Outils Formels pour la Modélisation en Mécanique

Yves A. Papegay

▶ **To cite this version:**

Yves A. Papegay. Outils Formels pour la Modélisation en Mécanique. Mécanique [physics.med-ph]. Université Nice Sophia Antipolis, 1992. Français. NNT : . tel-00784857

HAL Id: tel-00784857

https://theses.hal.science/tel-00784857

Submitted on 4 Feb 2013

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Outils Formels pour la Modélisation en Mécanique

Thèse de Doctorat

Y. Papegay

Institut National de Recherche en Informatique et en Automatique
Université de Nice Sophia-Antipolis
Projet Systèmes Algébriques Formels pour l'Industrie et la Recherche

2004 route des lucioles, B.P. 902 SOPHIA ANTIPOLIS Cedex
Tél : (33) 93 65 78 19 – Fax : (33) 93 65 78 58 – Email : papegay@sophia.inria.fr

13 novembre 1992

Table des matières

introduction **5**
0.1 l'étude des systèmes multicorps . . . 5
0.2 le calcul formel . . . 7
0.3 plan de l'étude . . . 7

1 préliminaires mécaniques **11**
1.1 de l'objet mécanique au mécanisme . . . 11
1.1.1 description du mouvement d'un objet . . . 12
1.1.2 notion de corps . . . 15
1.1.3 système de plusieurs corps rigides . . . 18
1.1.4 notion de liaison entre deux corps . . . 20
1.1.5 enfin les mécanismes . . . 22
1.2 modélisation des mécanismes en mouvement – étude cinématique . . . 27
1.2.1 variétés primitives de configurations . . . 27
1.2.2 variété d'un système de deux corps soumis à une liaison . . . 31
1.2.3 variété de configurations d'un mécanisme . . . 41
1.2.4 variétés d'étude d'un mécanisme . . . 44
1.3 modélisation du mouvement des mécanismes – étude dynamique . . . 46
1.3.1 grandeurs cinématiques, cinétiques et dynamiques . . . 47
1.3.2 les méthodes de la mécanique classique . . . 52
1.3.3 les méthodes de la mécanique analytique . . . 54
1.3.4 les méthodes "contemporaines" . . . 58

2 approche formelle : un exemple **61**
2.1 description du système étudié . . . 62
2.2 la loi fondamentale de la dynamique – Newton-Euler . . . 67
2.3 une autre méthode classique – équations de d'Alembert . . . 85
2.4 équations de Lagrange . . . 91
2.5 équations de Hamilton . . . 99
2.6 le formalisme de Kane . . . 104

3 outils pour la modélisation **113**
3.1 architecture d'un générateur formel des équations de la dynamique . . . 113
3.1.1 définition des besoins et des contraintes . . . 113
3.1.2 le point de vue informatique . . . 116

3.2 *LSD* : un Langage Symbolique de Description des mécanismes 122
3.2.1 un exemple de programme 123
3.2.2 la sémantique de *LSD* 125
3.2.3 la syntaxe de *LSD* 128
3.2.4 la compilation d'un programme *LSD* 130
3.3 un algorithme de génération des équations du mouvement 135
3.3.1 description du formalisme – établissement des équations 136
3.3.2 algorithme de génération des équations 144
3.4 manipulations d'expressions en *MAPLE* 150
3.4.1 les expressions matricielles 150
3.4.2 les expressions à base de quaternions 154
3.5 un manipulateur-simplificateur d'expressions matricielles 162
3.5.1 calcul équationnel, réécriture, et simplification modulo propriétés 163
3.5.2 le prototype *ULYSSE* 164
3.5.3 les expressions vectorielles et matricielles en *ULYSSE* 166

4 traitement des mécanismes bouclés 169
4.1 description des problèmes 170
4.1.1 degré de liberté local 170
4.1.2 détermination du degré de liberté 173
4.1.3 détermination d'un paramétrage 178
4.1.4 génération des équations 179
4.1.5 pour conclure 181
4.2 quelques exemples de mécanismes bouclés 182
4.3 quelques considérations mathématiques 192
4.3.1 première approche 193
4.3.2 deuxième approche 201
4.4 où l'on utilise des bases standards 201
4.4.1 bases standards d'idéaux de polynômes 202
4.4.2 bases standards de sous-modules 205
4.4.3 dimension d'une sous-variété de configurations 207
4.4.4 rang d'une matrice de contrainte 207
4.4.5 implémentations 209
4.5 expérimentations 211
4.5.1 première approche : algorithmes de calcul du rang 211
4.5.2 deuxième approche : utilisation de bases standards 215

5 conclusion 221

Annexes 223

A satellite : code de calcul *MAPLE* **225**
A.1 description du système (fichier `intro`) . . . 225
A.2 loi fondamentale de la dynamique . . . 228
A.3 principe de d'Alembert . . . 245
A.3.1 premières équations (fichier `dalembert1`) . . . 245
A.3.2 équations des corps isolés (fichier `dalembert2`) . . . 257
A.4 equations de Lagrange (fichier `lagrange`) . . . 260
A.5 equations de Hamilton . . . 297
A.5.1 mouvement relatif à $\mathcal{R}_1$ (fichier `hamilton1`) . . . 297
A.5.2 mouvement par rapport à R (fichier `hamilton2`) . . . 301
A.6 formalisme de Kane . . . 306
A.6.1 calculs préliminaires (fichier `kaneintro`) . . . 306
A.6.2 équations (fichier `kane`) . . . 316

B outils pour la modélisation **327**
B.1 *LSD* – code d'implémentation *Standard ML* . . . 327
B.1.1 la description lexicale de *LSD* (fichier `LSD.lex`) . . . 327
B.1.2 la description syntaxique de *LSD* (fichier `LSD.grm`) . . . 329
B.1.3 la syntaxe abstraite de *LSD* (fichier `absyn.sml`) . . . 332
B.1.4 le mécanisme de typage de *LSD* (fichier `typecheck.sml`) . . . 334
B.2 manipulation d'expressions matricielles – code *MAPLE* . . . 339
B.3 manipulation d'expressions matricielles – code *ULYSSE* . . . 352

C étude des mécanismes bouclés **357**
C.1 exemples de mécanismes bouclés . . . 357
C.1.1 maille d'antenne déployable à un niveau – fichier `maille1.gem` . . . 357
C.1.2 maille d'antenne déployable à deux niveaux – fichier `maille2.gem` . . . 367
C.2 calcul de rang : méthode probabiliste modulaire . . . 388
C.3 bases standards en *MAPLE* . . . 391
C.3.1 bases standards d'idéaux . . . 391
C.3.2 bases standards de sous-modules . . . 396
C.4 exemples de mécanisme bouclés – session *MACAULAY* . . . 403

introduction

La motivation qui a conduit à développer les outils présentés dans ce mémoire est d'améliorer, par l'apport de méthodes et de techniques de *calcul formel*, les performances et les capacités de calcul des logiciels qui modélisent et simulent le comportement dynamique de *systèmes de corps mécaniques reliés entre eux*[*].

La phase de modélisation de ce comportement est une étape essentielle du processus de conception de tout objet mécanique qui est amené par sa fonction à avoir un mouvement.

L'établissement des équations qui constituent un modèle dynamique demande généralement un travail long et complexe nécessitant d'importants calculs et qui doit être renouvelé à chaque modification apportée au système mécanique considéré.

On va s'intéresser ici à la possibilité d'écrire de manière automatique et littérale un ensemble d'équations différentielles régissant la dynamique des systèmes multicorps à partir de leur description physique.

0.1 l'étude des systèmes multicorps

On ressent, dans des secteurs technologiques extrèmement variés, le besoin de *calculer* précisément le comportement cinématique et dynamique de systèmes de corps polyarticulés.

En effet, ce comportement ne peut pas, dans de nombreux cas, être connu expérimentalement :

- soit parce que l'environnement n'est pas accessible, c'est le cas, par exemple dans le domaine spatial, lorsqu'il s'agit de déployer des panneaux solaires ou d'orienter des antennes sur un satellite,
- soit à cause d'un trop grand nombre d'expériences à réaliser, comme, par exemple, dans l'industrie automobile pendant la phase de conception d'un système de suspensions,
- soit par la nécessité pour un mécanisme de s'adapter à des circonstances changeantes ou difficilement prévisibles, comme cela peut être le cas en robotique,
- ou encore, parce qu'il est impossible d'effectuer des mesures de manière expérimentale, par exemple en biomécanique, pour déterminer quantitativement comment les muscles travaillent ou quels sont les efforts aux articulations.

L'évolution des technologies et des méthodes de fabrication pousse cependant à concevoir et à étudier des systèmes mécaniques de plus en plus complexes, au niveau de leurs structures, du nombre et de la diversité des corps qui les composent, et des liaisons qui interviennent.

Parallèlement, la nature des tâches imparties aux mécanismes nécessite une précision de plus en plus grande et exige des temps de calcul de plus en plus faible – l'objectif ultime est le temps réel – en ce qui concerne la simulation du mouvement.

[*] appelés *systèmes multicorps* ou encore *systèmes de corps polyarticulés*.

C'est ainsi que sont mises en évidence, de plus en plus fréquemment, les limites des logiciels de simulation basés sur des méthodes purement numériques, et la nécessité de faire appel à des techniques nouvelles et différentes.

un peu d'histoire

A la fin du dix-septième siècle et au dix-huitième, en posant les fondements de la mécanique classique, et en étudiant la dynamique du corps solide, Newton [Newton, 1687] puis d'Alembert [D'Alembert, 1743] ont ouvert la voie à l'étude du mouvement des objets physiques divers.

La mécanique analytique, depuis Lagrange [Lagrange, 1788] et avec Hamilton [Hamilton, 1835], permet théoriquement d'étudier la dynamique des systèmes multicorps quels qu'ils soient, mais conduit à des calculs qui se révèlent souvent impraticables même dans des cas relativement simples. Ainsi, peut-être par manque d'un réel intérêt technologique, mais surtout en l'absence d'outil permettant d'effectuer rapidement un gros volume de calculs, ce domaine n'a longtemps connu que très peu d'activité de recherche et les efforts des mécaniciens se sont focalisés sur l'étude théorique du comportement des milieux continus.

Un léger regain d'intérêt, aux alentours de 1900, a débouché sur quelques travaux importants – on peut citer Gibbs [Gibbs, 1879], Appell [Appell, 1900], Boltzmann [Boltzmann, 1902], Hamel [Hamel, 1904], ...– visant à rendre plus utilisable la théorie élaborée par Lagrange.

Toutefois, la dynamique des systèmes de corps polyarticulés n'a vraiment pris de l'essor qu'aux alentours de 1960, avec le développement d'ordinateurs possédant une grande puissance de traitement. A cette époque sont apparus des logiciels spécifiques à des systèmes mécaniques précis, pour simuler leur mouvement. Essentiellement écrits par des mécaniciens ou par des physiciens, il s'agissait d'énormes programmes codés en *FORTRAN*. Ces programmes calculaient l'évolution numérique d'un grand nombre de paramètres décrivant un système mécanique en résolvant, par des méthodes itératives, des équations différentielles établis manuellement au préalable.

Quelques années plus tard, il parut nécessaire, pour éviter des efforts de programmation redondants, de construire des logiciels plus généraux permettant d'étudier la dynamique de toute une classe de systèmes, à partir de la description physique de ces systèmes – *DRAM*, pour *Dynamic Response of Articulated Machinery*, développé par Chace et Korybalski [Chace et Korybalski, 1970], qui fut le premier de ces logiciels, se restreignait aux mouvements bidimensionnels, *ADAMS*, pour *Automatic Dynamic Analysis of Mechanical Systems* (voir [Rampalli, 1987]) est un produit commercial très performant actuellement utilisé par des milliers d'ingénieurs.

Parallèlement se sont développés de nombreuses études sur des problèmes connexes tels que l'optimisation de trajectoires en robotique, l'introduction de contrôle et/ou de commandes dans les mouvements, la modélisation de la flexibilité des corps, les singularités des mouvements, ...

Beaucoup plus récemment a été ressenti le besoin de connaître les équations gouvernant la dynamique des systèmes mécaniques sous une forme symbolique, essentiellement pour pouvoir manipuler ces équations avant d'effectuer de la simulation numérique afin de gagner en efficacité, et pour pouvoir étudier précisément le mouvement des systèmes au voisinage des configurations singulières. Sont alors apparus de nombreux logiciels ayant pour objet la génération sous forme littérale des équations de la dynamique de systèmes polyarticulés dont les deux principaux sont *AUTOLEV* développé par Kane et Levinson [Schaechter, Levinson et Kane, 1988] et *MESA VERDE* développé par Wittenburg (voir [Wittenburg, Woltz et Schmidt, 1990] et aussi [Wittenburg, 1977]).

Aucun de ces logiciels n'a malheureusement utilisé de véritables techniques de calcul formel.

0.2 le calcul formel

Le *calcul formel*, par opposition au calcul numérique, englobe toutes les manipulations d'objets mathématiques qui ne sont pas des nombres : calculs de primitives, de dérivées, factorisation d'expressions, développements, substitutions, utilisation de formules, de règles de simplification, résolution d'équation, manipulation de vecteurs, de matrices, d'opérateurs, ...

En tant que domaine scientifique, le calcul formel ou *calcul symbolique* dépend à la fois des mathématiques et de l'informatique et étudie tout ce qui permet de faire effectuer ces manipulations par un ordinateur ([Davenport, Siret et Tournier, 1988]).

Il existe actuellement des logiciels – ou systèmes – de calcul formel codés dans divers langages informatiques, qui sont capables d'effectuer toutes ces manipulations plus ou moins automatiquement en utilisant des algorithmes de calculs plus ou moins performants.

D'autres systèmes, plus récents, savent effectuer des calculs plus sophistiqués en autorisant l'utilisateur à définir les règles de calculs particulières au domaine des expressions qu'il utilise.

Pour la plupart, les logiciels de calcul formel intègrent un langage qui leur est propre et permet à l'utilisateur d'écrire de simples procédures comme de véritables programmes.

De plus, ces logiciels permettent aussi, bien entendu, d'effectuer des calculs numériques et acceptent le plus souvent de manipuler des entiers de taille arbitraire, des rationnels, et des approximations décimales de réels en précision arbitraire.

L'interêt, pour le scientifique en général et pour le mécanicien en particulier d'effectuer des calculs symboliquement en utilisant de tels logiciels est double :

- il s'agit d'une part de faire réaliser par une machine des calculs longs et fastidieux à faire à la main,
- et d'autre part, cela permet d'effectuer des calculs intrinsèquement justes par opposition aux calculs numériques, qui, dès lors qu'il ne font plus intervenir que des entiers ou des rationnels substituent aux nombres considérés des valeurs approchées.

Ce second avantage est particulièrement flagrant si coexistent dans un même calcul des valeurs dont les ordres de grandeurs sont très différents sans que l'on puisse pour autant négliger les "petites" quantités par rapport aux autres.

D'ailleurs ces avantages n'ont pas échappé aux scientifiques de toutes les origines puisque, si les mathématiciens ont consacré de nombreuses recherches au développement d'algorithmes de calculs, et si les informaticiens se sont penchés sur les problèmes spécifiques à la conception de logiciels de calcul formel, ce sont souvent des physiciens qui ont écrits, pour répondre à leurs besoins, de tels logiciels plus ou moins spécifiques – c'est notamment le cas de *MATHEMATICA* et aussi de *REDUCE* qui, avec *MACSYMA* et *MAPLE*, sont les quatre systèmes généraux les plus répandus (voir respectivement [Wolfram, 1988], [Hearn, 1985], [The Mathlab Group, 1983], [Char et al., 1988]).

0.3 plan de l'étude

Il a paru nécessaire, en l'absence d'une littérature qui puisse servir de référence sur ce point, de présenter soigneusement les objets mécaniques et mathématiques qui interviennent dans l'étude dynamique des systèmes de plusieurs corps reliés entre eux.

C'est l'objet du premier chapitre. Il permettra au lecteur non mécanicien d'acquérir les connaissances mécaniques essentielles pour la compréhension de ce mémoire.

> On s'attache ainsi, dans un premier temps, à définir ce qu'est un mécanisme, à l'aide d'entités mathématiques précises et adaptées. Pour cela, on part de la description

du mouvement d'un objet mécanique dans sa plus grande généralité, et on introduit successivement les notions de corps, de rigidité, d'ensemble de corps rigides, puis les notions de liaisons – holonomes ou non – entre les corps, et la notion d'architecture de systèmes de corps reliés entre eux.

Dans un deuxième temps, on introduit les bases nécessaires à l'étude cinématique des mécanismes, et notamment les variétés d'étude et les variétés de configurations. On montre au passage l'influence du caractère arborescent ou bouclé de la structure des mécanismes étudiés sur la complexité de cette étude et on dégage deux problèmes fondamentaux traités dans les chapitres suivants :

- le choix d'une variété d'étude adéquate,
- la nature de l'ensemble des configurations admissibles des mécanismes à structure bouclée.

On s'intéresse, dans un troisième temps, à l'étude dynamique des mécanismes. Ceci nécessite d'abord d'introduire soigneusement les grandeurs cinétiques et dynamiques, notamment les notions de masse, d'inertie et de force, et de faire le lien entre le vocabulaire développé jusqu'ici et le vocabulaire classique de la dynamique : vitesse et accélération en un point. On peut alors passer en revue les différentes méthodes d'établissement des équations de la dynamique, ainsi que les principes et les formalismes qui leur sont rattachés, ce qui fournira les bases essentielles pour le second chapitre.

Dans le deuxième chapitre, on étudie en détail sur un exemple simple mais non trivial les différentes méthodes existantes de génération des équations du mouvement de systèmes multicorps, dans l'optique de leur implémentation dans un logiciel de calcul symbolique.

Ce chapitre aidera le lecteur non familier à l'utilisation de logiciels de calcul formel à comprendre les possibilités et les contraintes qu'entraînent ces logiciels.

Tous les calculs effectués dans ce chapitre sont programmés et réalisés par le système de calcul formel *MAPLE*, ce qui permet d'analyser la plus ou moins bonne adéquation des différents formalismes à un traitement systématique, tant au niveau du travail nécessaire à leur emploi que du temps et du volume des calculs effectués et de la forme des équations obtenues.

Après une courte introduction au logiciel *MAPLE*, on commence par décrire avec soin le modèle de satellite qui servira d'exemple, en dégageant toutes les grandeurs indispensables à la mise en œuvre des différents formalismes.

On applique ensuite les méthodes de mises en équations issues de la mécanique classique.

Puis, on établit les équations analytiquement à partir des formalismes lagrangien et hamiltonien.

Et enfin, on met en œuvre un formalisme plus récent, provenant des travaux de Kane [Kane, 1978].

Ainsi ce second chapitre conduit naturellement à dégager une méthodologie pour construire un générateur automatique des équations de la dynamique de systèmes polyarticulés.

L'exposé de cette méthodologie constitue le troisième chapitre.

Plutôt que de concevoir et de présenter un nouveau logiciel, on a préféré mener une réflexion plus théorique et on expose dans ce chapitre des principes généraux qui aident à l'écriture de tels générateurs ainsi qu'un ensemble d'outils informatiques qui, en complétant les fonctionnalités des systèmes de calcul formel existants, permettent de mener à bien cette écriture.

On analyse, pour commencer, l'architecture informatique selon laquelle doit s'organiser le processus de génération des équations, ce qui permet de préciser, pour chaque tâche, les lacunes des systèmes de calcul formel classiques.

On présente ensuite successivement les outils qui ont été développés pour combler ces lacunes, à savoir :

- un langage de haut niveau pour la description des mécanismes,
- un algorithme de génération des équations, utilisant un formalisme spécifique, et particulièrement bien adapté à un traitement symbolique automatisé,
- diverses implémentations de bibliothèques de manipulation d'expressions vectorielles et matricielles – et d'expressions écrites à base de quaternions.

Le quatrième chapitre est entièrement consacré aux problèmes particuliers à la modélisation du mouvement des mécanismes dont l'architecture est bouclée, problèmes qui proviennent de la spécificité caractéristique de ces mécanismes en ce qui concerne essentiellement :

- la nature de l'ensemble des configurations admissibles,
- et la définition de la notion de degré de liberté.

Les problèmes mathématiques sous-jacents sont étudiés, des méthodes de résolutions sont proposées, leur implémentation et leur expérimentation sont présentées.

Tout d'abord, on analyse mathématiquement les deux caractéristiques spécifiques essentielles des mécanismes bouclés afin d'en déterminer une classe pour lesquels il est raisonnable de vouloir :

- calculer le degré de liberté,
- déterminer un paramétrage de la variété de configurations,
- et enfin générer les équations différentielles du mouvement.

ce qui conduit à définir une notion "locale" de degré de liberté et de variété de configurations.

On présente ensuite les deux types d'approches envisagées pour effectuer ces différentes tâches, qui sont :

i. une méthode de calcul du degré de liberté qui utilise des algorithmes probabilistes et modulaires de calcul de rang de matrices,
ii. une méthode d'étude de la variété de configurations faisant appel aux outils algébriques de la théorie des bases standard (ou de Gröbner).

Après avoir décrit quelques exemples de mécanismes à architecture bouclée - d'abord très simples puis graduellement plus complexes - et l'implémentation de ces deux méthodes dans des logiciels de calcul formel, on expérimente alors ces méthodes sur ces exemples en présentant les résultats qu'elles permettent d'obtenir, en analysant leurs performances et en dégageant aussi leurs limitations.

références bibliographiques

Tout au long de ce mémoire, il est fait référence à de nombreux textes présentant les travaux de leurs auteurs qu'ils s'agissent d'ouvrages généraux, de manuels, d'articles de revues spécialisés, d'actes de colloques, de thèses, de rapports ...

Ces textes sont répertoriés in fine selon un classement par nature puis par ordre alphabétique sur les noms des auteurs.

En ce qui concerne les ouvrages généraux, on trouvera pour chacun d'entre eux une courte analyse de leur contenu.

Chapitre Premier

préliminaires mécaniques

1.1 de l'objet mécanique au mécanisme

> On a tenu, dans cette section, à "décortiquer" sur le plan mathématique, des notions mécaniques qui sont pour la plupart très intuitives. Le lecteur averti pourra donc se dispenser d'une lecture qui risque de paraître inutilement complexe et fastidieuse.

L'objet le plus simple auquel est confronté le mécanicien est le *point matériel* qui, sans avoir de réelle existence physique, permet d'étudier de manière relativement simple certains phénomènes mécaniques.

Un objet mécanique, dans sa plus grande généralité, est une collection de points matériels, qui sont en quelque sorte ses *composants infinitésimaux*.

Si l'on veut considérer le mouvement d'un objet mécanique au cours d'un intervalle de temps, il faut :

- pouvoir représenter cet objet à chaque instant,
- connaître l'ensemble de ces représentations aux différents instants considérés,
- pouvoir "suivre" le mouvement de chacun des composants infinitésimaux de cet objet à l'intérieur de toutes ces représentations.

Ce dernier point est tout à fait indispensable en mécanique des milieux continus, lorsque l'on étudie le mouvement de fluides par exemple. Néanmoins il est aussi particulièrement important lorsque l'on étudie des corps rigides qui présentent des symétries au niveau de leurs formes. Par exemple, ce troisième point est nécessaire pour décider si un cylindre astreint à rester en contact avec un plan roule sur celui-ci sans glissement ou, au contraire, glisse sans roulement.

A un instant donné, un objet mécanique apparaît, à tout observateur, comme un ensemble de points dans l'espace*.

Chaque point représente la position à cet instant d'un composant infinitésimal de l'objet.

Ainsi, on dit que le point M^t ($\in \mathbb{E}$) est la *position* d'un composant infinitésimal de l'objet à l'instant t ou encore que (t, M^t) est un *état* de ce composant infinitésimal.

L'ensemble S^t ($\subset \mathbb{E}$) de tous les points M^t, et le couple (t, S^t) généralisent à l'objet les notions de position et d'état.

Pour un point matériel, et a fortiori pour un objet mécanique quelconque, les positions et les états diffèrent essentiellement pour plusieurs observateurs.

*on désigne par *espace* l'espace affine euclidien réel de dimension 3, noté $\mathbb{E}$.

La connaissance intrinsèque du mouvement demande de pouvoir représenter un objet mécanique indépendamment d'un observateur, ce qui nécessite de connaître les transformations qui font passer d'une observation à l'autre.

En mécanique relativiste, ces transformations dépendent explicitement du temps mais, dans le cadre de la mécanique classique dans lequel on se placera délibérément, les grandeurs spatiales et temporelles sont indépendantes.

Mathématiquement, on peut donc considérer que les objets mécaniques évoluent dans un espace fibré trivial sur $\mathbb{R}$ dont chaque fibre est l'espace affine $\mathbb{E}$, c'est à dire dans le produit cartésien $\mathbb{R} \times \mathbb{E}$.

La donnée d'une chronologie, et d'un repère de l'espace à chaque instant définit un *référentiel* de $\mathbb{R} \times \mathbb{E}$ et donne un sens à la notion d'observateur du mouvement.

La connaissance du mouvement d'un objet mécanique dans un référentiel suffit pour définir ce mouvement intrinsèquement dans la mesure où elle permet de le connaître dans n'importe quel autre référentiel.

1.1.1 description du mouvement d'un objet

Cette section s'inspire du premier chapitre du cours de mécanique [Germain, 1980].

On se donne un référentiel définissant tout au long du temps la position et l'orientation de l'observateur du mouvement.

Si I ($\subset \mathbb{R}$) est l'ensemble de tous les instants considérés lors du mouvement, on note

$$\mathcal{M} = \{(t, M^t), t \in I \text{ et } M^t \in S^t\}$$

l'ensemble de tous les états des composants infinitésimaux de l'objet au cours du mouvement.

Les propriétés physiques d'un objet en mouvement se traduisent sur $\mathcal{M}$ sous la forme de propriétés mathématiques que l'on va étudier ci-dessous.

Définition 1.1 *Une famille $(S^t)_{t\in I}$ de sous-ensembles de $\mathbb{E}$ dépendants du temps (I est une partie de $\mathbb{R}$) permet de décrire le mouvement d'un objet mécanique si et seulement si il existe une application Π, définie sur $I \times I$ ($\subset \mathbb{R}^2$) qui satisfait les conditions suivantes :*

i. pour tous réels t_1 et t_2 appartenants à I, $\Pi(t_1, t_2)$ est une bijection de S^{t_1} dans S^{t_2},

ii. pour tout réel t appartenant à I, $\Pi(t, t)$ est l'application identique de $\mathbb{E}$,

iii. pour tous réels t_1, t_2 et t_3 appartenants à I, $\Pi(t_1, t_3) = \Pi(t_2, t_3) \circ \Pi(t_1, t_2)$.

Comme le montre la figure (1.1), mécaniquement, c'est l'existence de l'application Π qui traduit le fait que l'on doit savoir comment évolue, au cours du mouvement, et à l'intérieur de l'objet, chacun de ses composants infinitésimaux. $\Pi(t_1, t_2)$ associe à tout point M_1 de S^{t_1} – position à l'instant t_1 d'un composant infinitésimal de l'objet mécanique considéré – le point M_2 de S^{t_2} – sa position à l'instant t_2.

Les conditions (*ii*) et (*iii*) de la définition (1.1) assurent la cohérence de la notion de composant infinitésimal d'un objet mécanique, tandis que la condition (*i*) traduit l'invariance de sa composition au cours du mouvement.

Mathématiquement, l'existence d'une application Π satisfaisant les conditions (*i*), (*ii*) et (*iii*) de la définition précédente permet de donner une signification précise aux notions d'objet mécanique et de composant infinitésimal en définissant les objets abstraits leur correspondant : le *système mécanique* et la *particule*.

En effet, on peut définir une relation d'équivalence ($\rightsquigarrow$) sur l'ensemble de tous les états des composants infinitésimaux de l'objet au cours du mouvement $\mathcal{M}$, en disant que (t_1, M_1) est en relation

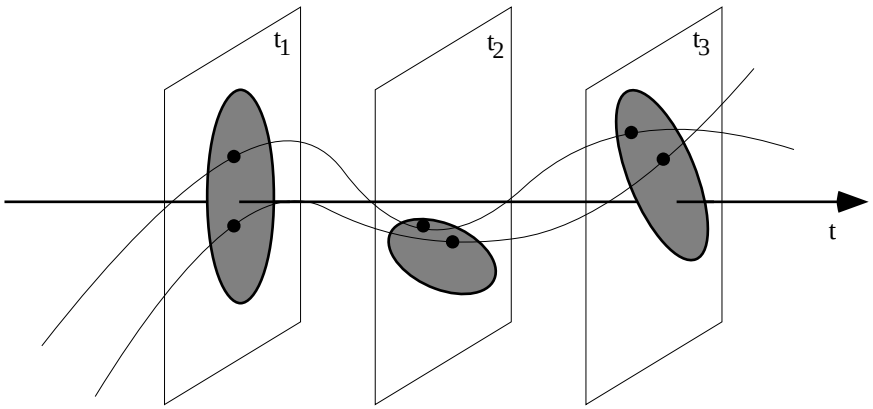


figure 1.1 : description du mouvement d'un objet mécanique

avec (t_2, M_2) si et seulement si M_2 est la position à l'instant t_2 du composant dont la position à l'instant t_1 est M_1, c'est à dire :

$$[(t_1, M_1) \rightsquigarrow (t_2, M_2)] \Longleftrightarrow \Pi(t_1, t_2)(M_1) = M_2$$

Définition 1.2 *On appelle particules d'un système mécanique les classes d'équivalence non vides de la relation* $(\rightsquigarrow)$ *définie ci-dessus ; un système mécanique* $\mathcal{S}$ *est l'ensemble de ses particules, c'est-à-dire l'espace quotient* $\mathcal{M}/_{(\rightsquigarrow)}$ *de l'ensemble de tous les états des composants infinitésimaux par cette relation.*

Dans la suite, on assimilera l'objet mécanique au système mécanique qu'il permet de définir, et ses composants infinitésimaux aux particules du système ; on parlera, notamment, par abus de langage, d'états et de positions d'une particule ou du système.

Comme on va s'intéresser seulement à l'étude de corps rigides ou flexibles, il n'est pas nécessaire de développer longuement le principe de description du mouvement qui est très intuitif.

On se bornera à présenter ici ce qui permet de définir rigoureusement les notions de corps rigides et flexibles.

La particularité d'un système mécanique est le fait qu'une particule possède, à tout instant, une position bien déterminée, précisément puisque Π satisfait la condition (i).

C'est pourquoi, on peut, pour tout instant t_0, exhiber une bijection ϕ^{t_0} entre le système $\mathcal{S}$ et l'ensemble S^{t_0} : à toute particule m, on fait correspondre sa position à l'instant t_0, le point M^{t_0}. Puisque m est une classe d'équivalence non vide, il existe un représentant de m : (t, M^t). Alors M^{t_0} est le point tel que : $M^{t_0} = \Pi(t, t_0)(M^t)$.

Cette application est bien définie puisque elle est indépendante du choix de l'état (t, M^t) de m d'après les conditions sur Π ; elle est bijective à cause de l'existence de l'application canonique de passage au quotient.

L'existence de ces bijections ϕ^t permet de donner une nouvelle caractérisation des modélisations d'objets mécaniques en mouvement qui ne fait plus intervenir l'application Π. C'est cette caractérisation que l'on utilisera essentiellement par la suite.

Définition 1.3 *Une famille* $(S^t)_{t\in I}$ *de sous-ensembles de* $\mathbb{E}$ *dépendants du temps (I est une partie de* $\mathbb{R}$*) permet de décrire le mouvement d'un objet mécanique si et seulement si il existe un ensemble* $\mathcal{S}$ *et une application surjective* ψ*, de* $\mathcal{M} = \{(t, M^t), t \in I \text{ et } M^t \in S^t\}$ *dans cet ensemble* $\mathcal{S}$*, d'une part, et d'autre part il existe une application* ϕ *de I dans* $\mathbb{E}^{\mathcal{S}}$*, telles que :*

i. pour tout réel t, $\phi(t)$ est une bijection de $\mathcal{S}$ dans S^t,

ii. pour tout réel t, et pour tout point M^t de S^t, $\phi(t)(\psi(t, M^t)) = M^t$,

iii. pour tout réel t, et pour tout élément m de $\mathcal{S}$, $\psi(t, \phi(t)(m)) = m$.

On montre facilement que cette définition est bien équivalente à la première définition : en effet, si Π est donnée, ψ est l'application qui, à tout état (t, M^t), associe sa classe d'équivalence et ϕ peut être définie par : pour tout réel t, $\phi(t) = \phi^t$; réciproquement, si ψ et ϕ sont données, Π peut être définie pour tous réels t_1 et t_2 par : $\Pi(t_1, t_2)$ est l'application de S^{t_1} dans S^{t_2} telle que $\Pi(t_1, t_2)(M^{t_1}) = \phi(t_2)(\psi(t_1, M^{t_1}))$ pour tout point M^{t_1} de S^{t_1}.
De plus, il est immédiat que l'ensemble $\mathcal{S} = \psi(\mathcal{M})$ introduit dans cette propriété est bien identique au système $\mathcal{S}$ tel qu'on la définit précédemment et que la notion de particule est conservée.
La figure (1.2) schématise la manière dont sont reliés ces objets mathématiques entre eux.

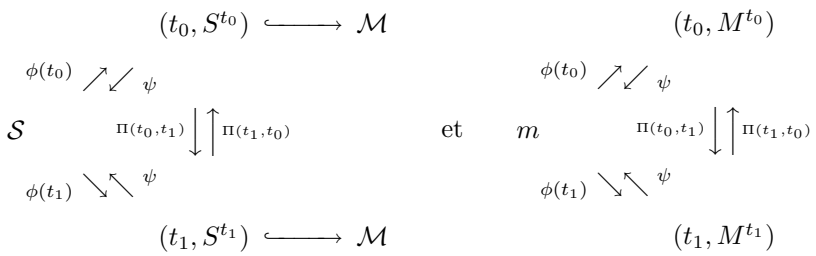


figure 1.2 : liens entre Π, ϕ et ψ

Remarque 1.4 L'intérêt de la définition (1.3) est que le mouvement d'un objet mécanique y apparaît comme une caractéristique de cet objet, alors que dans la définition (1.1) c'est l'objet qui apparaît comme une caractéristique d'un mouvement.

Définition 1.5 *Soit $\mathcal{S}$ un système mécanique, on appelle configuration de ce système à l'instant t l'application $\phi(t) = \phi^t$ de $\mathcal{S}$ dans $\mathbb{E}$ qui, à toute particule m de $\mathcal{S}$, associe M^t – sa position à l'instant t.*

On note $\mathcal{V}_{\mathit{eff}}$ l'ensemble de toutes les configurations du système[†] : $\mathcal{V}_{\mathit{eff}} = \{\phi^t, t \in I\}$.

Définition 1.6 *L'application ϕ de I dans $\mathcal{V}_{\mathit{eff}}$ qui, à tout instant t associe la configuration ϕ^t du système à cet instant est appelée mouvement du système.*

En conclusion, on dispose, pour modéliser un objet mécanique en mouvement, des différentes notions suivantes :

- la particule du système et le système $\mathcal{S}$, ensemble des particules,
- l'état du système à l'instant t : (t, S^t), l'état d'une particule du système à l'instant t : (t, M^t), avec $M^t \in S^t$ et l'ensemble de tous les états des particules du système au cours du mouvement : $\mathcal{M}$,
- la configuration du système à l'instant t : ϕ^t, l'ensemble de toutes les configurations du système : $\mathcal{V}_{\mathit{eff}} = \{\phi^t, t \in I\}$, et le mouvement du système : ϕ.

Remarque 1.7 Il est important de bien différencier les notions d'états, de positions et de configurations d'un système mécanique :

[†] l'indice *eff* est mis ici pour préciser qu'il s'agit des configurations effectivement atteintes par le système au cours du mouvement.

- à deux instants distincts t_1 et t_2 le système est dans des états différents (puisque $t_1 \neq t_2$) même s'il occupe des positions semblables ($S^{t_1} = S^{t_2}$) et même si il a une configuration identique aux deux instants ($\phi^{t_1} = \phi^{t_2}$),
- à deux instants distincts t_1 et t_2 le système peut occuper les mêmes positions ($S^{t_1} = S^{t_2}$) tout en se trouvant dans des configurations différentes ($\phi^{t_1} \neq \phi^{t_2}$).

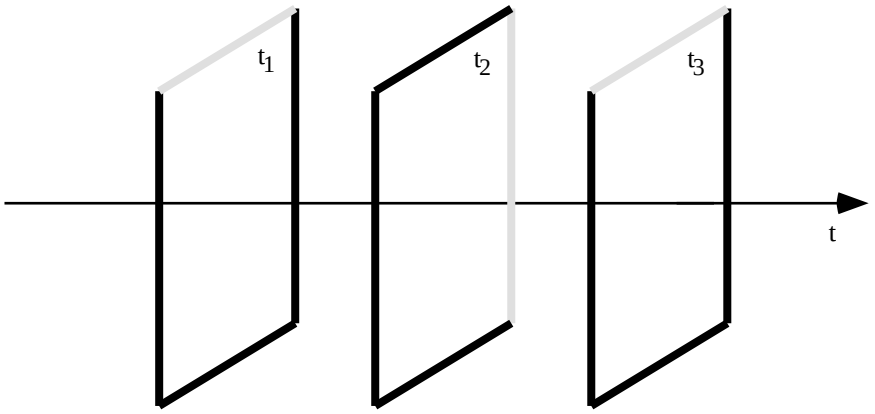


figure 1.3 : exemples de positions et de configurations

Dans la figure (1.3) représentant trois instants du mouvement d'un cadre métallique dans lequel on sait distinguer l'un des côtés, on a : $S^{t_1} = S^{t_2} = S^{t_3}$, $\phi^{t_1} = \phi^{t_3}$ et $\phi^{t_1} \neq \phi^{t_2}$ alors que : $(t_1, S^{t_1}) \neq (t_2, S^{t_2})$, $(t_1, S^{t_1}) \neq (t_3, S^{t_3})$, et $(t_2, S^{t_2}) \neq (t_3, S^{t_3})$.

Classiquement, la description du mouvement présentée ci-dessus est appelée description par trajectoires (ou Lagrangienne) ; en effet, l'ensemble $\{(t, \phi^t(m)), t \in I\}$ représente, d'un point de vue mécanique, la trajectoire de la particule m au cours du mouvement.

Il existe d'autres descriptions du mouvement, notamment la description par champs de vitesses, dite Eulerienne, que l'on n'abordera pas ici.

1.1.2 notion de corps

Par *corps*, on désigne intuitivement un objet mécanique tel que, pour connaître son mouvement, il suffise de connaître une de ses configurations et le mouvement d'un nombre fini de particules représentatives de cet objet – et non le mouvement de toutes ses particules .

On dira qu'un corps est rigide s'il suffit de connaître le mouvement de quatre particules bien choisies de ce corps – ou d'un repère attaché à ce corps.

On parlera de corps flexible sinon.

notion de rigidité

Définition 1.8 *On dit d'un système $\mathcal{S}$ qu'il est rigide si et seulement si l'application Π introduite dans la définition (1.1) est telle que : pour tous réels t et t' de I, l'application $\Pi(t, t')$ est une isométrie de S^t dans $S^{t'}$.*

Définition 1.9 *Une particule m est dite solidaire d'un système rigide $\mathcal{S}$ si et seulement si la réunion $\mathcal{S} \cup \{m\}$ forme encore un système rigide.*

On va voir ce que l'hypothèse de rigidité énoncée dans la définition (1.8) entraîne comme propriétés de l'application ϕ décrivant le mouvement.

Mais, auparavant, il est nécessaire de munir les systèmes mécaniques d'une structure topologique ; ceci est possible, notamment dans la mesure où l'hypothèse de rigidité autorise l'introduction de la notion de distance entre les particules.

On considère un système rigide $\mathcal{S}$; on peut définir sur $\mathcal{S}$ une distance δ de manière canonique à partir d'un état particulier du système (t_0, S^{t_0}) par :

$$\delta(m,n) = d(M^{t_0}, N^{t_0})$$

où $M^{t_0} = \phi^{t_0}(m)$, $N^{t_0} = \phi^{t_0}(n)$ et d est la distance usuelle de $\mathbb{E}$.

Le fait que les applications $\Pi(t,t')$ soient, pour tous réels t et t' des isométries, garantit que δ est indépendante du choix de t_0 et donc bien définie.

On peut alors munir $\mathcal{S}$ de la topologie induite par cette distance δ.

C'est ce que l'on supposera dans la suite ; ainsi, on peut affirmer :

Propriété 1.10 *Lorsque $\mathcal{S}$ est un système mécanique rigide, pour tout réel t, l'application ϕ^t est une isométrie de $\mathcal{S}$ dans $\mathbb{E}$.*

De manière analogue, on peut, par le choix d'une position particulière du système (t_0, S^{t_0}), transporter la structure affine canonique de $\mathbb{E}$ sur $\mathcal{S}$; pour cela, on commence par prolonger les isométries $\Pi(t,t')$ en des isométries de $\langle S^t \rangle$ dans $\langle S^{t'} \rangle$ – on note $\langle X \rangle$ le sous-espace affine engendré par X lorsque X est une partie d'un espace affine. Ceci permet de plonger $\mathcal{S}$ dans un ensemble $\langle \mathcal{S} \rangle$ défini comme image réciproque de $\langle S^{t_0} \rangle$ par le prolongement de ϕ^{t_0}, et sur lequel on transporte la structure d'espace affine de $\langle S^{t_0} \rangle$.

Définition 1.11 *Lorsqu'il existe un ensemble de 4 particules affinement indépendantes appartenant à (*resp. *solidaire d') un système rigide, on dit que cet ensemble forme un repère (*resp. *un repère solidaire) de ce système.*

Remarque 1.12 L'existence d'un repère solidaire du système rigide $\mathcal{S}$ a pour conséquence que $\langle S^{t_0} \rangle$ est égal à $\mathbb{E}$ pour tout t_0 et donc que $\langle \mathcal{S} \rangle$ est un espace affine de dimension 3 et d'espace vectoriel sous-jacent $\vec{\mathbb{E}} = \mathbb{R}^3$. Dans ce cas, on note cet espace $\mathcal{E}$. On peut alors, attacher au système rigide un référentiel.

Dans tout ce qui suit, on se placera dans les circonstances de la remarque précédente, ce qui, en d'autres termes signifie que l'on saura, pendant l'observation mécanique d'un système en mouvement, distinguer la position de chacun de ses composants et par exemple, décider de l'orientation angulaire d'une tige en rotation sur elle-même ou des deux faces d'un système plan.

Dans ce cas, on emploiera le terme de *corps rigide* au lieu de système rigide.

A ce stade de l'étude, on est amené à adopter une hypothèse supplémentaire : l'hypothèse de la continuité du mouvement.

Celle-ci nécessite tout d'abord de considérer que I est un intervalle de $\mathbb{R}$ puis se traduit par la continuité de l'application ϕ qui, à tout réel t fait correspondre l'isométrie ϕ^t donnant la position des particules du système à cet instant (cette continuité est relative à la topologie induite par $(\phi^{t_0})^{-1}$ – indépendamment de t_0 – sur l'ensemble des isométries de $\mathcal{E}$ dans $\mathbb{E}$ à partir de l'ensemble des isométries de $\mathbb{E}$).

Le mouvement étant désormais supposé continu, on peut alors transporter la structure d'espace affine *orienté* de $\mathbb{E}$ sur $\mathcal{E}$, toujours par une méthode analogue à celle utilisée précédemment ; en effet, à cause de la définition de ϕ, il est facile de voir que la continuité du mouvement entraîne la continuité de l'application Π par rapport à ses deux variables ; alors, puisque l'ensemble des

isométries de $\mathbb{E}$, $\mathrm{Is}(\mathbb{E})$, possède deux composantes connexes, et, puisque $\Pi(t,t) = \mathrm{Id}(\mathbb{E})$, on en déduit que toutes les applications $\Pi(t,t')$ sont des déplacements de $\mathbb{E}$.

Une orientation de $\mathcal{E}$ définie à partir d'une position particulière du système est donc bien indépendante du choix de cette position.

Dorénavant, on suppose $\mathcal{E}$ muni d'une orientation qui permet d'affiner la propriété (1.10) en écrivant que :

Propriété 1.13 *Lorsque $\mathcal{S}$ est un corps rigide, pour tout réel t, l'application ϕ^t est un déplacement de $\mathcal{S}$ dans $\mathbb{E}$.*

Ainsi, l'ensemble $\mathcal{V}_{\mathit{eff}} = \{\phi^t, t \in I\}$ de toutes les configurations d'un corps rigide est une partie de l'ensemble $\mathrm{Is}^+(\mathcal{E}, \mathbb{E})$ des déplacements de $\mathcal{E}$ dans $\mathbb{E}$.

Cet ensemble est bien sûr isomorphe, modulo le choix d'une position de référence E^{t_0} $(= \mathbb{E})$, à une partie de l'ensemble $\mathrm{Is}^+(\mathbb{E})$ des déplacements de $\mathbb{E}$:

> à ϕ^t – la configuration du système à l'instant t – on associe $\phi^{t_0,t}$ – le déplacement de $\mathbb{E}$ rendant le diagramme de la figure (1.4) commutatif, i.e. le déplacement $\phi^{t_0,t} = \phi^t(\phi^{t_0})^{-1}$.

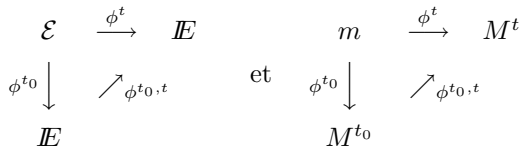


figure 1.4 : isomorphisme entre $\mathrm{Is}^+(\mathcal{E}, \mathbb{E})$ et $\mathrm{Is}^+(\mathbb{E})$

Concrètement, la donnée de la configuration ϕ^t d'un corps rigide à l'instant t correspond à la donnée de la position et de l'orientation, relativement à l'observateur du mouvement, d'un repère $\mathcal{R}$ solidaire de ce corps.

Le choix d'une position et d'une orientation de référence R^{t_0} permet, seul, de connaître cette configuration sous la forme d'un déplacement de l'espace, celui qui transforme R^{t_0} en R^t.

traitement de la flexibilité

Comme on l'a écrit plus haut, un corps flexible est un système mécanique tel que, pour connaître son mouvement, il suffise de connaître une de ses configurations et le mouvement d'un nombre fini n $(n > 4)$ de ses particules.

On ne s'intéressera pas ici à la modélisation de la dynamique des corps flexibles, ce thème pouvant, à lui seul, justifier un travail bien plus ambitieux.

On se contentera de présenter, pour information, les deux méthodes les plus utilisées dans ce domaine, basées sur la décomposition en corps multiples ou sur la notion de pseudo-rigidité.

décomposition en corps multiples En anticipant sur les paragraphes suivants qui présentent les systèmes de plusieurs corps rigides articulés, on remarque que l'on peut considérer un tel système comme un seul corps flexible, dans la mesure où la donnée d'une de ses configurations et du mouvement de 4 particules pour chaque corps suffira à connaître le mouvement du système.

Réciproquement, si la nature physique du corps flexible que l'on étudie le permet, on étudiera son mouvement en le décomposant en un certain nombre de corps rigides reliés, le cas échéant, par diverses articulations.

Par exemple une poutre pourra être considérée comme une succession de petites tiges reliées les unes aux autres par des articulations rotoïdes, ...

pseudo-rigidité L'étude physique de corps flexibles permet, dans certains cas, de mettre en évidence des modes vibratoires à fréquences déterminées dont seulement un nombre fini influe de manière non négligeable sur le mouvement du corps considéré.

Chaque mode est une application de $\mathcal{E} \times I\!R$ dans $I\!E$ décrivant un mouvement vibratoire particulier de chacune des particules du système.

Ces modes ayant été préalablement établis, la position du corps flexible à un instant donné, est entièrement déterminée, modulo le choix d'une position de référence, par la donnée d'un déplacement – en ce qui concerne le mouvement principal – et d'un point dans l'espace vectoriel fonctionnel engendré par les modes – en ce qui concerne le mouvement vibratoire du à la flexibilité.

On retrouve ainsi une notion étendue de *configuration*.

1.1.3 système de plusieurs corps rigides

On se donne un référentiel définissant, tout au long du temps, la position et l'orientation de l'observateur du mouvement.

Définition 1.14 *Un système de plusieurs corps rigides est un ensemble $\mathcal{S}$ de particules, tel qu'il existe une famille finie de parties $\mathcal{S}_1, \ldots, \mathcal{S}_n$ de $\mathcal{S}$ vérifiant :*

- $\mathcal{S} = \bigsqcup_{i=1}^{n} \mathcal{S}_i$ *(réunion disjointe des $\mathcal{S}_i$)*
- *pour tout entier i inférieur ou égal à n, $\mathcal{S}_i$ est un corps rigide*

Mécaniquement, le fait de définir un système de corps rigides comme une réunion disjointe de systèmes rigides autorise deux particules appartenant à deux corps distincts à occuper la même position à un instant donné.

Ainsi, tout ce qui provient de la notion de contact ou de chocs entre différents corps n'est pas pris en compte à ce stade de la modélisation.

Les contacts et les chocs interviendront lors de la définition de liaisons entre les corps ou lors de l'étude dynamique du mouvement.

On a vu, au paragraphe précédent, que pour chaque corps $\mathcal{S}_i$, il existe une application ϕ_i de I dans $I\!E^{\mathcal{S}_i}$ qui, à un instant donné, fait correspondre la configuration de ce corps.

Les $\mathcal{S}_i$ étant des corps rigides, on sait que, pour tout réel t, $\phi_i(t) = \phi_i^t$ est un déplacement de $\mathcal{S}_i$ dans $I\!E$, ou plutôt d'un exemplaire de $\mathcal{E}$ dans $I\!E$.

D'après le premier point de la définition (1.14), la connaissance de l'application ϕ permettant de décrire le mouvement de $\mathcal{S}$ est équivalente à la connaissance des n applications ϕ_i, en effet, $\prod_{i=1}^{n}(I\!E^{\mathcal{S}_i})$ et $(I\!E)^{\mathcal{S}}$ (i.e. $(I\!E)^{\sqcup_{i=1}^{n}\mathcal{S}_i}$) sont naturellement isomorphes.

Ceci conduit à poser :

Définition 1.15 *On appelle ensemble de toutes les configurations d'un système de plusieurs corps rigides, l'ensemble $\mathcal{V}_{eff} = \{\phi(t) = (\phi_1^t, \ldots, \phi_n^t), t \in I\}$.*

Ainsi, comme l'on sait associer, à chaque corps, un repère qui lui est solidaire, $\mathcal{V}_{eff}$ est une partie de l'espace produit $\mathrm{Is}^+(\mathcal{E}, I\!E)^n$.

De manière analogue à ce qui se passe lorsqu'on étudie le mouvement d'un seul corps, le choix d'un ensemble de positions de référence des corps du système $S^{t_0} = (S_1^{t_0}, \ldots, S_n^{t_0})$ – ou, ce qui est équivalent, des positions de repères qui leur sont solidaires $(R_1^{t_0}, \ldots, R_n^{t_0})$ – permet d'obtenir

une description effective de son mouvement en définissant un isomorphisme entre les espaces produits $\mathrm{Is}^+(\mathcal{E},\mathbb{E})^n$ et $\mathrm{Is}^+(\mathbb{E})^n$: à $(\phi_1^t,\ldots,\phi_n^t)$ la configuration du système à l'instant t, on associe $(\phi_1^{t_0,t},\ldots,\phi_n^{t_0,t}) = (\phi_1^t(\phi_1^{t_0})^{-1},\ldots,\phi_n^t(\phi_n^{t_0})^{-1})$.

Remarque 1.16 Le choix d'un ensemble de positions de référence $(R_1^{t_0},\ldots,R_n^{t_0})$ des repères solidaires des corps du système permet aussi de définir à cet instant, pour tout corps $\mathcal{S}_i$ et pour tout corps $\mathcal{S}_j$, un déplacement $\phi_{ij}^{t_0}$ de l'espace tel que $\phi_{ij}^{t_0}(R_i^{t_0}) = R_j^{t_0}$.

On déduit de $\phi_{ij}^{t_0}$ le déplacement ϕ_{ij}^t tel que $\phi_{ij}^t(R_i^t) = R_j^t$ par :

$$\phi_{ij}^t = \phi_j^{t_0,t}\phi_{ij}^{t_0}(\phi_i^{t_0,t})^{-1} = \phi_j^t(\phi_j^{t_0})^{-1}\phi_{ij}^{t_0}\phi_i^{t_0}(\phi_i^t)^{-1}$$

La relation ci-dessus est, bien sûr, vérifiée pour tout instant t_0 et pour tout instant t de l'intervalle de temps considéré.

Cela permet de définir un déplacement φ_{ij} de $\mathcal{S}_i$ ($\subset\mathcal{E}$) dans $\mathcal{S}_j$ ($\subset\mathcal{E}$) tel que $\varphi_{ij} = (\phi_j^{t_0})^{-1}\phi_{ij}^{t_0}\phi_i^{t_0}$, indépendamment du choix de t_0 – voir figure (1.5).
On appelle φ_{ij} *configuration structurelle relative* de $\mathcal{S}_j$ par rapport à $\mathcal{S}_i$.

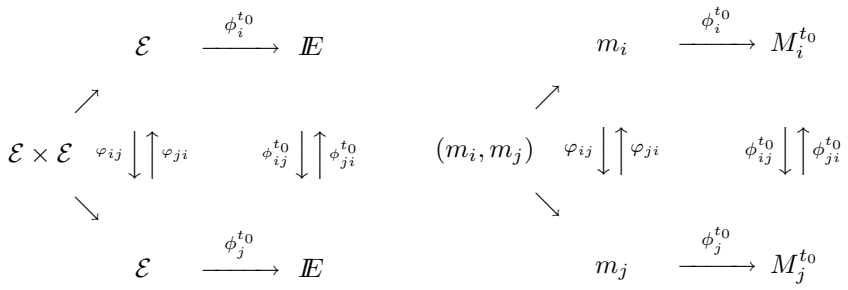


figure 1.5 : configurations et configurations relatives

Définition 1.17 *On appelle configuration relative du système mécanique $\mathcal{S}$ par rapport au corps $\mathcal{S}_i$ à l'instant t le $(n-1)$-uple :*

$$(\phi_{i1}^t,\ldots,\phi_{i(i-1)}^t,\phi_{i(i+1)}^t,\ldots,\phi_{in}^t)$$

L'application de I dans $Is^+(\mathbb{E})^{n-1}$ qui fait correspondre $(\phi_{i1}^t,\ldots,\phi_{i(i-1)}^t,\phi_{i(i+1)}^t,\ldots,\phi_{in}^t)$ à tout instant t est appelée mouvement relatif du système par rapport au corps $\mathcal{S}_i$.

De tout ce qui précède, on peut conclure que, si l'on se donne un système mécanique $\mathcal{S}$ de n corps rigides avec $n-1$ configurations structurelles fondamentales φ_{ij} de $\mathcal{E}$ dans $\mathcal{E}$ – ce qui revient à les connaître toutes puisque $\varphi_{jk} = \varphi_{ik}(\varphi_{ij})^{-1}$– on dispose de $n^{n-2}+1$ modélisations équivalentes du mouvement de $\mathcal{S}$:

i. par la donnée de l'application de I dans $\mathrm{Is}^+(\mathcal{E},\mathbb{E})^n$:

$$t \mapsto (\phi_1^t,\ldots,\phi_n^t)$$

ii. par la donnée d'une application de I dans $\mathrm{Is}^+(\mathcal{E},\mathbb{E})$, et de $n-1$ applications de I dans $\mathrm{Is}^+(\mathbb{E})^{n-1}$:

$$\begin{cases} t & \mapsto & \phi_k^t \\ t & \mapsto & (\phi_{\sigma(1)1}^t,\ldots,\phi_{\sigma(k-1)(k-1)}^t,\phi_{\sigma(k+1)(k+1)}^t,\ldots,\phi_{\sigma(n)n}^t) \end{cases}$$

où $k \in [\![1,n]\!]$, et σ est une application de $[\![1,n]\!]$ dans $[\![1,n]\!]$ que l'on peut définir à l'aide d'un arbre étiqueté par les n premiers entiers, dont la racine est étiquetée par k et tel que $\sigma(i)$ est l'étiquette du nœud adjacent à celui étiqueté par i appartenant au chemin qui le relie à la racine – voir figure (1.6).

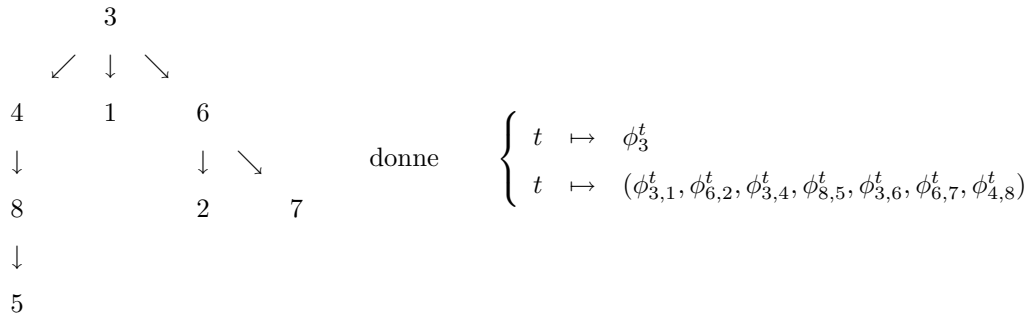


figure 1.6 : exemple de σ

Pratiquement, on utilisera plus fréquemment des modélisations de type (ii), qui sont mieux adaptées au traitement des liaisons entre les corps, comme on va le voir au paragraphe suivant.

De plus, ces modélisations sont particulièrement adaptées si l'on considère que l'observateur du mouvement est solidaire d'un des corps – i.e. si l'on considère que l'un des corps est fixe.

Dans ce cas, une configuration du système est en effet complètement caractérisée par la connaissance des $n-1$ déplacements de l'espace donnant les configurations relatives.

1.1.4 notion de liaison entre deux corps

Dans les systèmes mécaniques dont on va étudier le mouvement, les différents corps interagissent par le biais de contacts ou d'articulations, voire d'attirances ou de répulsions magnétiques.

Ces interactions dépendent exclusivement de la *structure mécanique* (ou physique) interne du système, c'est à dire de la forme et de la matière des corps, ainsi que de la nature des dispositifs mécaniques qui réalisent les articulations entre eux.

Cette structure contraint les mouvements relatifs des différents corps d'un système au niveau de leurs positions et de leurs orientations, voire de leurs vitesses, ou de leurs accélérations lorsque ces grandeurs sont définies.

D'une manière générale, il est intuitif de considérer que deux corps sont liés, à un instant donné, si il existe localement, au moins un mouvement du système qu'ils composent qui, bien que continu, n'est pas compatible avec la structure mécanique de ce système.

Il est clair que, comme cette structure, la nature d'une liaison d'un système peut évoluer dans le temps.

Néanmoins, il est raisonnable, lorsque l'on étudie des systèmes de corps rigides, de supposer que leur structure mécanique ne varie pas continuellement.

C'est pourquoi on peut considérer que cette structure reste constante pendant tout l'intervalle I pendant lequel on observe le mouvement, quitte à réduire cet intervalle, ce qui amènera à parler de liaisons indépendantes du temps, à condition que celles-ci ne fassent pas explicitement intervenir le temps – i.e. soient invariantes par changement de chronologie.

De plus, la notion de liaison doit être indépendante de l'observateur du mouvement, et notamment de sa position et de son propre mouvement.
Formellement, on dira qu'il existe une liaison entre deux corps $\mathcal{S}_i$ et $\mathcal{S}_j$ d'un système à l'instant t_0 lorsque :

> si l'on étudie, dans n'importe quel voisinage V_{t_0} de t_0, le mouvement du système restreint à $\mathcal{S}_i \sqcup \mathcal{S}_j$, l'ensemble $\Phi_{comp}(V_{t_0})$ des applications $\phi = (\phi_i, \phi_j)$ définissant un mouvement continu compatible avec la structure mécanique de ce système est strictement inclus dans l'ensemble des applications continues de ce voisinage dans l'ensemble des déplacements de $\mathcal{S}_i \sqcup \mathcal{S}_j$ dans $\mathbb{E}$, ce qui se note :
>
> $$\Phi_{comp}(V_{t_0}) \subsetneq C^0(V_{t_0}; \mathrm{Is}^+(\mathcal{E}, \mathbb{E})^2)$$

Remarque 1.18 Pour traduire l'indépendance vis à vis de l'observateur du mouvement, on assumera que $\Phi_{comp}(V_{t_0})$ possède la propriété suivante :

$$\forall \phi \in C(V_{t_0}; \mathrm{Is}^+(\mathbb{E})),\ \text{si } (\phi_i, \phi_j) \in \Phi_{comp}(V_{t_0}) \text{ alors } (\phi\phi_i, \phi\phi_j) \in \Phi_{comp}(V_{t_0})$$

Lorsque l'on utilise une modélisation du mouvement des systèmes de corps rigides faisant intervenir les mouvements relatifs des corps, on introduit, pour tout voisinage V_{t_0} d'un instant t_0, l'ensemble $\Phi^{ij}_{comp}(V_{t_0})$ des applications ϕ_{ij} définissant un mouvement relatif continu de $\mathcal{S}_j$ par rapport à $\mathcal{S}_i$ compatible avec la structure mécanique de $\mathcal{S}_i \sqcup \mathcal{S}_j$.
On dit alors que $\mathcal{S}_i$ et $\mathcal{S}_j$ sont reliés à l'instant t_0 si cet ensemble est strictement inclus dans l'ensemble des applications continues de V_{t_0} dans l'ensemble des déplacements de $\mathbb{E}$:

$$\Phi^{ij}_{comp}(V_{t_0}) \subsetneq C^0(V_{t_0}; \mathrm{Is}^+(\mathbb{E}))$$

Les deux moyens de définir des liaisons que l'on vient de voir sont totalement équivalents.

La première définition possède l'avantage d'être symétrique par rapport aux deux corps.

La deuxième, quant à elle, possède l'avantage d'exprimer intrinsèquement la propriété d'indépendance des liaisons vis à vis du mouvement de l'observateur, puisqu'aucune condition n'est imposée au mouvement propre de $\mathcal{S}_i$.

On va continuer, dans ce paragraphe, à étudier les liaisons dans le cadre de ces deux modélisations.

Dans le cas de liaisons indépendantes du temps, on pose $V_{t_0} = I$ – I est l'intervalle de temps pendant lequel on considère le mouvement et pendant lequel la structure mécanique du système reste constante – et on utilise les notations $\Phi_{comp}(I)$ et $\Phi^{ij}_{comp}(I)$.

Définition 1.19 *Il existe une liaison entre les corps $\mathcal{S}_i$ et $\mathcal{S}_j$ si et seulement si l'une des deux conditions équivalentes suivantes est réalisée :*

i.

$$\begin{cases} \Phi_{comp}(V_{t_0}) \subsetneq C^0(V_{t_0}; Is^+(\mathcal{E}, \mathbb{E})^2) \\ \forall \phi \in C(V_{t_0}; Is^+(\mathbb{E})),\ \textit{si } (\phi_i, \phi_j) \in \Phi_{comp}(V_{t_0}) \textit{ alors } (\phi\phi_i, \phi\phi_j) \in \Phi_{comp}(V_{t_0}) \end{cases}$$

ii.

$$\Phi^{ij}_{comp}(V_{t_0}) \subsetneq C^0(V_{t_0}; Is^+(\mathbb{E}))$$

L'ensemble $\Phi_{comp}(I)$ que l'on notera Φ_{comp} si il n'y a pas d'ambigüité, permet de définir l'ensemble $\mathcal{V}_{adm}(I)$ noté aussi $\mathcal{V}_{adm}$ des *configurations admissibles* pour la structure mécanique de $\mathcal{S}_i \sqcup \mathcal{S}_j$ par :

$$\mathcal{V}_{adm} = \{(\phi^t_i, \phi^t_j) \in \mathrm{Is}^+(\mathcal{E}\mathbb{E})^2 / \exists t \in I, \exists \phi \in \Phi_{comp}, \phi(t) = (\phi^t_i, \phi^t_j)\}$$

une configuration étant admissible pour la structure mécanique du système si elle peut être atteinte au cours d'un mouvement compatible.

On définit de manière analogue l'ensemble $\mathcal{V}_{adm}^{ij}$ des *configurations relatives admissibles* pour la structure mécanique de $\mathcal{S}_i \sqcup \mathcal{S}_j$ par :

$$\mathcal{V}_{adm}^{ij} = \{\phi_{ij}^t \in \mathrm{Is}^+(\mathbb{E})/\exists t \in I, \exists \phi \in \Phi_{comp}, \phi(t) = \phi_{ij}^t\}$$

une configuration relative étant admissible pour la structure mécanique du système si elle peut être atteinte au cours d'un mouvement relatif compatible.

Toujours à cause de l'indépendance de la notion de liaison par rapport à l'observateur du mouvement, l'ensemble des configurations admissibles possède la propriété suivante :

$$\forall \phi^0 \in \mathrm{Is}^+(\mathbb{E}),\ \text{si } (\phi_i^t, \phi_j^t) \in \mathcal{V}_{adm} \text{ alors } (\phi^0\phi_i^t, \phi^0\phi_j^t) \in \mathcal{V}_{adm}$$

Par contre, la définition de l'ensemble des configurations relatives admissibles intègre cette indépendance de façon intrinsèque.

Selon la nature des interactions entre les corps du système, deux cas peuvent être envisagés :

i. n'importe quelle application continue de I dans $\mathcal{V}_{adm}$ (*resp.* dans $\mathcal{V}_{adm}^{ij}$) définit un mouvement (*resp.* relatif) compatible avec la structure mécanique interne du système,

ii. il existe au moins une application continue de I dans $\mathcal{V}_{adm}$ (*resp.* dans $\mathcal{V}_{adm}^{ij}$) qui définit un mouvement (*resp.* relatif) qui n'est cependant pas compatible avec cette structure.

Définition 1.20 *On dit que deux corps d'un système sont reliés par une liaison holonome lorsque le système restreint à ces deux corps vérifie les conditions de l'alinea (i) ci-dessus, i.e. :*

$$\Phi_{comp} = C(I; \mathcal{V}_{adm}) \ \textit{(ou } \Phi_{comp}^{ij} = C(I; \mathcal{V}_{adm}^{ij})\textit{)}$$

On parle de liaison non-holonome sinon.

Mécaniquement, cette définition s'interprète ainsi :

- les liaisons holonomes introduisent, sur le mouvement des corps liés, des contraintes portant exclusivement sur les positions et les orientations relatives de ces corps,
- tandis que les liaisons non-holonomes introduisent aussi des contraintes sur les vitesses et/ou sur les accélérations relatives des corps.

Les figures (1.7) et (1.8) présentent deux exemples illustrant le caractère holonome ou non des liaisons.

1.1.5 enfin les mécanismes

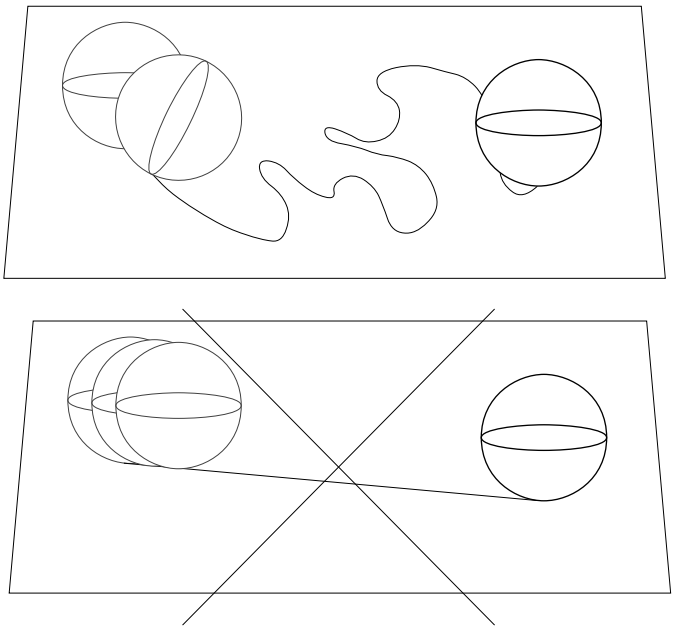
On considère un système mécanique $\mathcal{S}$ constitué de $n+1$ corps rigides $\mathcal{S}_0, \mathcal{S}_1, \ldots, \mathcal{S}_n$; on suppose que l'on sait, à chaque corps $\mathcal{S}_i$ pour $i = 0, 1, \ldots, n$ associer un repère $\mathcal{R}_i$ qui lui est solidaire.

On considère bien entendu qu'il existe des liaisons entre certains corps de ce système ; la structure mécanique de $\mathcal{S}$ est supposée constante pendant tout l'intervalle de temps I (i.e. les liaisons sont indépendantes du temps).

Enfin, on assume l'hypothèse que ces liaisons sont holonomes.

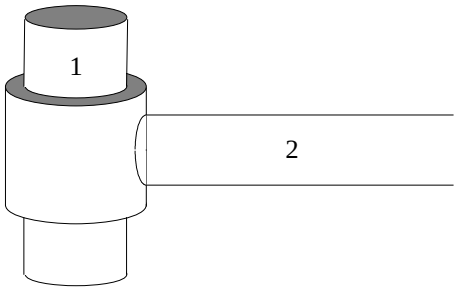
On appellera un tel système un *mécanisme*.

On a vu que, pour modéliser le mouvement de ce mécanisme, il suffit de connaître, à chaque instant de l'intervalle de temps considéré, les configurations dans lesquelles se trouvent les corps qui le composent.



Ici, le système est constitué d'une sphère et d'un plan, la sphère etant astreinte à rouler sur ce plan sans glissement. Etant donnée une orientation, on peut imaginer un mouvement qui permette à la sphère de se retrouver en contact avec n'importe quel point du plan dans cette orientation, alors que le mouvement qui consiste à la déplacer en lui conservant son orientation n'est pas compatible puisque c'est un glissement.

figure 1.7 : exemple de liaison non holonome



Ici, la tige 2 peut glisser le long de la tige 1 et tourner autour d'elle. N'importe quel mouvement défini à partir de ce glissement et de cette rotation est bien compatible.

figure 1.8 : exemple de liaison holonome

On se propose d'étudier ce mouvement relativement au référentiel $\mathcal{R}_0$ solidaire du corps $\mathcal{S}_0$.

Ainsi le corps $\mathcal{S}_0$ est considéré comme fixe et le système est composé de n corps rigides en mouvement.

D'après le paragraphe précédent, pour déterminer effectivement le mouvement du mécanisme, il s'agit de calculer, pour chaque instant,

i. soit les n déplacements du système dans l'espace qui donnent, par rapport au référentiel choisi, la position et l'orientation de chacun des corps, ou plutôt des repères associés à chacun des corps,

ii. soit les n déplacements de l'espace qui donnent, par rapport au référentiel choisi, la position et l'orientation relatives de chacun des corps par rapport au corps $\mathcal{S}_0$, ou plutôt des repères associés à chacun des corps par rapport à $\mathcal{R}_0$,

iii. soit n déplacements de l'espace qui donnent, par rapport au référentiel choisi, la position et l'orientation relatives de chaque corps par rapport à un autre corps convenablement choisi, ou plutôt de chaque repère associé à des corps par rapport à un repère associé à un autre corps convenablement choisi.

Concrètement, pour connaître les n déplacements de l'alinea (i), il est nécessaire de se rapporter à une configuration de référence du mécanisme, et il s'agit en fait de calculer les n déplacements de l'espace qui donnent la position et l'orientation de chacun des corps par rapport à sa position et à son orientation dans la configuration de référence.

Ceci revient exactement, si l'on choisit comme configuration de référence une configuration où les repères de tous les corps sont confondus avec $\mathcal{R}_0$, à calculer les n déplacements de l'alinea (ii).

C'est pourquoi, dans la suite, on utilisera essentiellement des descriptions du mouvement du type de l'alinea (ii) basé sur la notion de configuration relative.

On peut étendre aux mécanismes les notions décrites au paragraphe précédent :

- un mouvement relatif par rapport à $\mathcal{S}_0$ est *compatible* avec la structure mécanique interne du mécanisme, si, pour tout couple de corps $(\mathcal{S}_i, \mathcal{S}_j)$ reliés du mécanisme, le mouvement relatif induit par rapport au corps $\mathcal{S}_i$ du sous-système $\mathcal{S}_i \sqcup \mathcal{S}_j$ est compatible avec la structure mécanique de ce sous-système,
- on note Φ_{comp} l'ensemble des mouvements relatifs par rapport à $\mathcal{S}_0$ compatibles,
- on note $\mathcal{V}_{adm}$ l'ensemble des configurations relatives admissibles i.e. l'ensemble des configurations relatives du mécanisme qui peuvent être atteintes au cours de tout mouvement compatible.

Remarque 1.21 En toute généralité, on n'est pas en mesure de relier simplement l'ensemble $\mathcal{V}_{adm}$ des configurations relatives admissibles du mécanisme considéré globalement, aux ensembles $\mathcal{V}_{adm}^{ij}$ des configurations relatives admissibles pour chacun des couples de corps liés. On verra, au paragraphe suivant, une classe de mécanismes – les mécanismes à structure arborescente – pour lesquels il est facile de déduire $\mathcal{V}_{adm}$ à partir des ensembles $\mathcal{V}_{adm}^{ij}$.

Dans le cas où les corps du mécanisme considéré ne sont reliés que par des liaisons holonomes, on peut séparer l'approche cinématique du système de son approche dynamique et procéder en deux étapes pour étudier leur mouvement :

i. on détermine, dans un premier temps, l'ensemble $\mathcal{V}_{adm}$ de toutes les configurations relatives admissibles du mécanisme

ii. on détermine, pour chaque instant t, la configuration $\phi(t)$ $(\in \mathcal{V}_{adm})$ correspondante.

Remarque 1.22 Il y a lieu de bien distinguer $\mathcal{V}_{\mathit{eff}}$ et $\mathcal{V}_{adm}$ qui le contient, en effet :

- L'ensemble $\mathcal{V}_{adm}$, sous-ensemble de $\mathrm{Is}^+(\mathbb{E})^n$ ne dépend que de la structure mécanique interne du mécanisme c'est à dire du nombre, de la position et de la nature des corps et des liaisons entre les corps.
- L'ensemble $\mathcal{V}_{\mathit{eff}}$ est un sous-ensemble de $\mathcal{V}_{adm}$ qui contient toutes les configurations effectivement atteintes au cours du mouvement ; il dépend des propriétés dynamiques du système (masse, inertie, forme des corps, ...) et des actions externes qu'il subit. On l'obtient en résolvant les équations différentielles du mouvement écrites sur $\mathcal{V}_{adm}$.

Si le système comporte des liaisons non-holonomes, cette scission du travail en deux étapes n'est pas envisageable dans la mesure où :

- il faut introduire la notion de configuration compatible avec un certain mouvement puisque toutes les configurations admissibles ne sont pas forcément atteignables avec un mouvement compatible,
- le calcul de $\mathcal{V}_{comp}$, ensemble des configurations compatibles, nécessite la connaissance du mouvement.

On est alors amené à effectuer la mise en équation globalement ou à définir une notion de configuration étendue du système qui décrit non seulement les positions et les orientations de chaque corps mais aussi leurs vitesses et leurs accélérations ..., d'une manière générale, toutes les grandeurs dont dépendent les liaisons.

architecture d'un mécanisme

On note ℓ_{ij} la liaison reliant le corps $\mathcal{S}_i$ et le corps $\mathcal{S}_j$, le cas échéant et $\mathcal{L}$ l'ensemble des liaisons du mécanisme.

Pour étudier l'ensemble des configurations admissibles d'un mécanisme lorsqu'il est composé de plus de deux corps reliés entre eux, on est amené à représenter son architecture, c'est à dire la manière dont les corps sont reliés.

On utilise pour cela la notion mathématique de graphe.

A un système mécanique $\mathcal{S}$ composé de $n+1$ corps soumis à un ensemble de liaison $\mathcal{L}$, on associe le graphe simple $\mathfrak{G}(\mathcal{S}) = (\{0, 1, \ldots, n\}, \mathfrak{A}(\mathcal{S}))$ où $\mathfrak{A}(\mathcal{S})$ est un ensemble de paires d'éléments de $\{0, 1, \ldots, n\}$ tel que $\{i, j\} \in \mathfrak{A}(\mathcal{S})$ si et seulement si $\ell_{ij} \in \mathcal{L}$ ou $\ell_{ji} \in \mathcal{L}$.

Alors, les correspondances suivantes s'établissent entre le graphe du système mécanique et son architecture :

- à chaque corps, on associe un sommet du graphe,
- à chaque liaison, on associe une arête,
- si il y a une liaison entre deux corps alors les sommets correspondants sont voisins,
- tout sous-graphe de $\mathfrak{G}(\mathcal{S})$ représente un sous-système de $\mathcal{S}$ et en particulier, chaque composante connexe du graphe représente un sous-système mécanique dit *interconnecté*.

Remarque 1.23 Quitte à diviser le système mécanique en plusieurs sous-systèmes et à les étudier séparément, on peut considérer que le système est complètement interconnecté, c'est à dire que son graphe est connexe.

Remarque 1.24 Le fait que l'on représente un système mécanique par un graphe simple sous-entend les particularités suivantes sur les liaisons du système :

- un corps n'est jamais lié à lui-même,

- il n'y a pas de notion d'orientation sur les liaisons.

Définition 1.25 *On dit d'un sytème mécanique que son architecture est arborescente si le graphe qui lui est associé est un arbre. On parle d'architecture bouclée sinon.*

Par abus de langage, on parlera de mécanismes arborescents et de mécanismes bouclés.

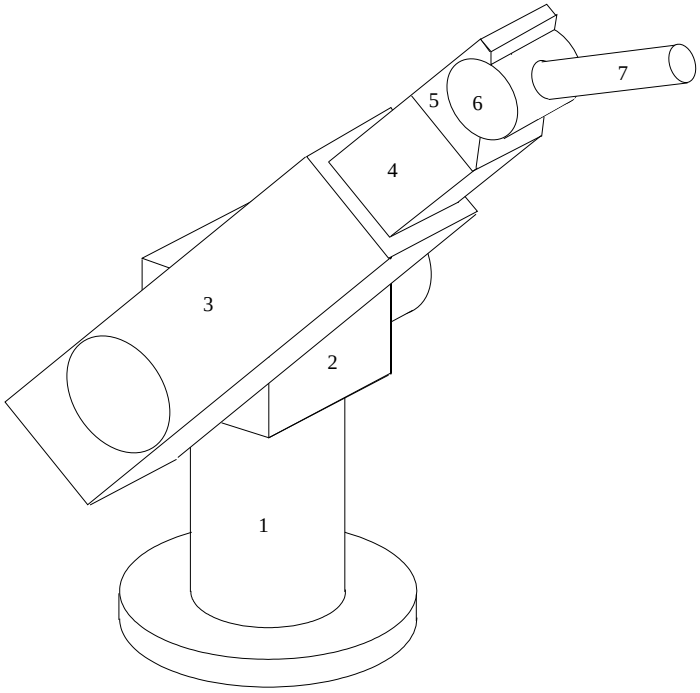


figure 1.9 : exemple de mécanisme arborescent

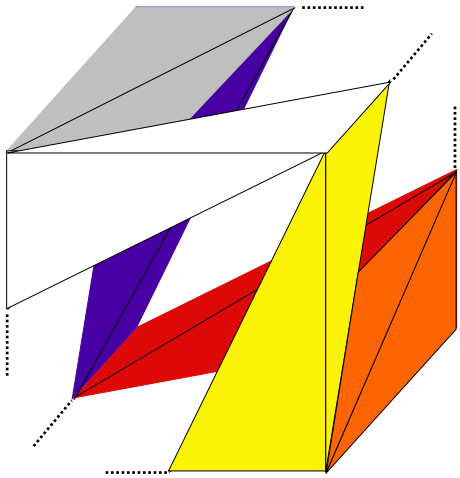


figure 1.10 : exemple de mécanisme bouclé

1.2 modélisation des mécanismes en mouvement – étude cinématique

On considère un mécanisme $\mathcal{S}$ composé de $n+1$ corps rigides $\mathcal{S}_0, \mathcal{S}_1, \ldots, \mathcal{S}_n$.
On se propose d'étudier le mouvement de ce mécanisme vu par un observateur solidaire du corps $\mathcal{S}_0$.
On utilisera, bien entendu, la notion de configuration relative par rapport à un corps, une configuration relative* du système $\mathcal{S}$ à un instant t étant, comme on l'a vu dans la définition (1.17), une application $\phi(t) = (\phi^t_{i1}, \ldots, \phi^t_{i(i-1)}, \phi^t_{i(i+1)}, \ldots, \phi^t_{in})$ appartenant à l'espace produit $\mathrm{Is}^+(\mathbb{E})^n$.

1.2.1 variétés primitives de configurations

Propriété 1.26 *$Is^+(\mathbb{E})$ est une variété différentielle† réelle de dimension 6.*

La propriété ci-dessus est un résultat classique de géométrie différentielle.
En fait, il est même connu, que l'ensemble des déplacements de l'espace affine euclidien réel de dimension trois, $\mathrm{Is}^+(\mathbb{E})$, est un groupe de Lie de dimension 6, i.e. une variété analytique réelle de dimension 6 munie d'une structure de groupe telle que les applications $[(x,y) \mapsto xy]$ et $[x \mapsto x^{-1}]$ sont analytiques.
Moyennant le choix d'une origine dans $\mathbb{E}$, $\mathrm{Is}^+(\mathbb{E})$ s'identifie au produit semi-direct $\mathbb{R}^3 \times \mathrm{SO}_3$.
En effet, tout déplacement ϕ se décompose en un produit $\tau \circ \rho$ d'une rotation autour de cette origine ρ par une translation τ, le produit de deux déplacements s'exprimant par :

$$(\tau_2, \rho_2) \circ (\tau_1, \rho_1) = (\rho_2(\tau_1) + \tau_2, \rho_2 \circ \rho_1)$$

et l'inverse du déplacement (τ, ρ) étant $(-\rho^{-1}(\tau), \rho^{-1})$.
Les cartes utilisées pour décrire $\mathrm{Is}^+(\mathbb{E})$ comportent donc trois coordonnées pour la translation et trois coordonnées pour la rotation.
En ce qui concerne la rotation, on utilise généralement :

- soit trois angles (angles d'Euler, angles de Bryant, ...),
- soit des quaternions unitaires.

rotation et quaternions unitaires

Le groupe des quaternions unitaires est la sphère S^3, ce groupe opère sur le corps des quaternions. Les restrictions des automorphismes intérieurs ainsi obtenus à $\mathbb{R}^3$ assimilé à l'ensemble des quaternions purs, sont des rotations.
On peut donc construire un homomorphisme surjectif entre le groupe des quaternions unitaires et le groupe des rotation SO_3.
Au quaternion unitaire $(\lambda_0, \lambda_1, \lambda_2, \lambda_3)$ (dans la base canonique $(1, i, j, k)$), on associe la rotation ρ d'axe dirigé selon $(\lambda_1, \lambda_2, \lambda_3)$ et d'angle $2\arccos(\lambda_0)$.
Si ρ est une rotation donnée par son angle α et un vecteur unitaire de son axe $\vec{v}$ alors un des quaternions qui la représentent est :

$$\lambda = \cos(\frac{\alpha}{2}) + \sin(\frac{\alpha}{2}).\vec{v}$$

l'autre étant son opposé.
La matrice $\bar{\rho}$ de la rotation ρ associé à $(\lambda_0, \lambda_1, \lambda_2, \lambda_3)$ s'écrit :

*lorsque aucune confusion n'est possible, on écrira simplement configuration.
†sauf mention du contraire, on écrira *différentielle* ou *différentiable* pour *indéfiniment différentiable*.

$$\begin{pmatrix} {\lambda_0}^2+{\lambda_1}^2-{\lambda_2}^2-{\lambda_3}^2 & 2(\lambda_1\lambda_2-\lambda_3\lambda_0) & 2(\lambda_0\lambda_2+\lambda_1\lambda_3) \\ 2(\lambda_0\lambda_3+\lambda_1\lambda_2) & {\lambda_0}^2+{\lambda_2}^2-{\lambda_1}^2-{\lambda_3}^2 & 2(\lambda_2\lambda_3-\lambda_0\lambda_1) \\ 2(\lambda_1\lambda_3-\lambda_0\lambda_2) & 2(\lambda_0\lambda_1+\lambda_2\lambda_3) & {\lambda_0}^2+{\lambda_3}^2-{\lambda_1}^2-{\lambda_2}^2 \end{pmatrix}$$

angles d'Euler

Les angles d'Euler (Ψ la *précession* autour d'Oz, Θ la *nutation* autour d'Ox et Φ la *rotation propre* autour d'Oz) fournissent une carte de SO_3 souvent employée mais qui ne permet pas d'atteindre de manière univoque les rotations d'axe Oz – voir figure (1.11).

La matrice de la rotation ainsi paramétrée par (Ψ,Θ,Φ) est :

$$\bar{\rho}=\begin{pmatrix} \cos(\Psi) & -\sin(\Psi) & 0 \\ \sin(\Psi) & \cos(\Psi) & 0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\Theta) & -\sin(\Theta) \\ 0 & \sin(\Theta) & \cos(\Theta) \end{pmatrix}\begin{pmatrix} \cos(\Phi) & -\sin(\Phi) & 0 \\ \sin(\Phi) & \cos(\Phi) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

c'est à dire :

$$\begin{pmatrix} \cos(\Phi)\cos(\Psi)-\sin(\Phi)\cos(\Theta)\sin(\Psi) & -\sin(\Phi)\cos(\Psi)-\cos(\Phi)\cos(\Theta)\sin(\Psi) & \sin(\Theta)\sin(\Psi) \\ \cos(\Phi)\sin(\Psi)+\sin(\Phi)\cos(\Theta)\cos(\Psi) & -\sin(\Phi)\sin(\Psi)+\cos(\Phi)\cos(\Theta)\cos(\Psi) & -\sin(\Theta)\cos(\Psi) \\ \sin(\Phi)\sin(\Theta) & \cos(\Phi)\sin(\Theta) & \cos(\Theta) \end{pmatrix}$$

angles de Bryant

Les angles de Bryant (${}^x\Phi$ autour d'Ox, ${}^y\Phi$ autour d'Oy, ${}^z\Phi$ autour d'Oz) fournissent aussi une carte de SO_3 souvent employée mais qui ne permet pas, cette fois, d'atteindre de manière univoque les rotations d'axe Ox – voir figure (1.12).

La matrice de la rotation paramétrée par $({}^x\Phi,{}^y\Phi,{}^z\Phi)$ est donnée par :

$$\bar{\rho}=\begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos({}^x\Phi) & -\sin({}^x\Phi) \\ 0 & \sin({}^x\Phi) & \cos({}^x\Phi) \end{pmatrix}\begin{pmatrix} \cos({}^y\Phi) & 0 & \sin({}^y\Phi) \\ 0 & 1 & 0 \\ -\sin({}^y\Phi) & 0 & \cos({}^y\Phi) \end{pmatrix}\begin{pmatrix} \cos({}^z\Phi) & -\sin({}^z\Phi) & 0 \\ \sin({}^z\Phi) & \cos({}^z\Phi) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

c'est à dire :

$$\begin{pmatrix} \cos({}^z\Phi)\cos({}^y\Phi) & -\sin({}^z\Phi)\cos({}^y\Phi) & \sin({}^y\Phi) \\ \sin({}^z\Phi)\cos({}^x\Phi)+\cos({}^z\Phi)\sin({}^y\Phi)\sin({}^x\Phi) & \cos({}^z\Phi)\cos({}^x\Phi)-\sin({}^z\Phi)\sin({}^y\Phi)\sin({}^x\Phi) & -\cos({}^y\Phi)\sin({}^x\Phi) \\ \sin({}^z\Phi)\sin({}^x\Phi)-\cos({}^z\Phi)\sin({}^y\Phi)\cos({}^x\Phi) & \cos({}^z\Phi)\sin({}^x\Phi)+\sin({}^z\Phi)\sin({}^y\Phi)\cos({}^x\Phi) & \cos({}^y\Phi)\cos({}^x\Phi) \end{pmatrix}$$

Le mécanisme ci-dessous réalise mécaniquement les angles d'Euler, Ψ autour de l'axe 1, Θ autour de l'axe 2 et Φ autour de l'axe 3.
On voit que, si il s'agit de représenter une rotation autour de l'axe Oz, les axes 1 et 3 doivent être confondus et seule la somme $\Psi + \Phi$ est fixée.

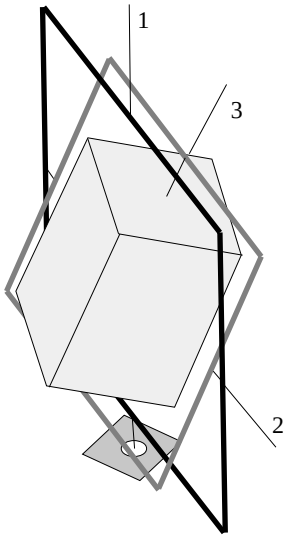


figure 1.11 : angles d'Euler

Le mécanisme ci-dessous réalise mécaniquement les angles de Bryant, ${}^{x}\Phi$ autour de l'axe x, ${}^{y}\Phi$ autour de l'axe y et ${}^{z}\Phi$ autour de l'axe z.
On voit que, si il s'agit de représenter une rotation autour de l'axe Ox, les axes x et z doivent être confondus et seule la somme ${}^{x}\Phi + {}^{z}\Phi$ est fixée.

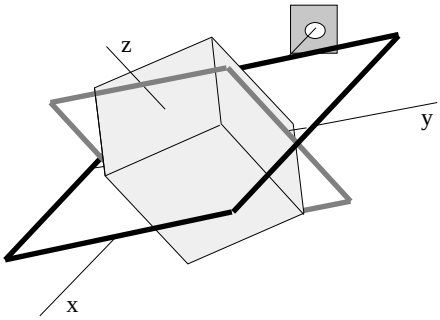


figure 1.12 : angles de Bryant

Quel que soit le jeu de paramètres utilisé pour représenter les rotations, on obtient alors les coordonnées $(x_{M'}, y_{M'}, z_{M'})$ d'un point M', image par ϕ du point M de coordonnées (x_M, y_M, z_M) en écrivant :

$$\begin{pmatrix} x_{M'} \\ y_{M'} \\ z_{M'} \end{pmatrix} = \bar{\rho} \begin{pmatrix} x_M \\ y_M \\ z_M \end{pmatrix} + \bar{\tau} \qquad \text{avec} \qquad \bar{\tau} = \begin{pmatrix} x_\tau \\ y_\tau \\ z_\tau \end{pmatrix}$$

Cette relation peut aussi s'écrire à l'aide de matrices 4×4 :

$$\begin{pmatrix} x_{M'} \\ y_{M'} \\ z_{M'} \\ 1 \end{pmatrix} = \left(\begin{array}{ccc|c} & \bar{\rho} & & \begin{matrix} x_\tau \\ y_\tau \\ z_\tau \end{matrix} \\ \hline 0 & 0 & 0 & 1 \end{array}\right) \begin{pmatrix} x_M \\ y_M \\ z_M \\ 1 \end{pmatrix}$$

En termes de telles matrices le déplacement $\phi = (\tau, \rho)$ et son inverse sont donc représentés respectivement par les matrices :

$$\left(\begin{array}{c|c} \bar{\rho} & \bar{\tau} \\ \hline \mathbf{0} & 1 \end{array}\right) \qquad \text{et} \qquad \left(\begin{array}{c|c} \bar{\rho}^{-1} & -\bar{\rho}^{-1}(\bar{\tau}) \\ \hline \mathbf{0} & 1 \end{array}\right)$$

Et le produit de deux déplacements s'écrit naturellement comme un produit de matrices :

$$\left(\begin{array}{c|c} \bar{\rho_2} & \bar{\tau_2} \\ \hline \mathbf{0} & 1 \end{array}\right) \left(\begin{array}{c|c} \bar{\rho_1} & \bar{\tau_1} \\ \hline \mathbf{0} & 1 \end{array}\right) = \left(\begin{array}{c|c} \bar{\rho_2}\bar{\rho_1} & \bar{\rho_2}\bar{\tau_1} + \bar{\tau_2} \\ \hline \mathbf{0} & 1 \end{array}\right)$$

Propriété 1.27 *$Is^+(\mathbb{E})^n$ est une variété différentielle réelle de dimension $6n$.*

Cette propriété découle immédiatement de la propriété précédente puisque $\mathrm{Is}^+(\mathbb{E})^n$ est le produit direct

$$\underbrace{\mathrm{Is}^+(\mathbb{E}) \times \ldots \times \mathrm{Is}^+(\mathbb{E})}_{n\text{fois}}$$

Définition 1.28 *On appelle $Is^+(\mathbb{E})^n$ variété primitive de configurations du système $\mathcal{S}$ relativement à $\mathcal{S}_0$[‡] que l'on note $\mathcal{V}'(\mathcal{S})$.*

En généralisant ce que l'on a vu plus haut, on dispose de quatre possibilités pour décrire, plus ou moins précisément, un élément ϕ de la variété primitive de configurations d'un mécanisme

- en termes de déplacement :
$$\phi = (\phi_1, \ldots, \phi_n)$$
- en termes de translations et de rotations :
$$\phi = ((\tau_1, \rho_1), \ldots, (\tau_n, \rho_n))$$

[‡]ou simplement variété de configurations de $\mathcal{S}$ lorsqu'il n'y a pas de risque de confusion.

- aux moyens de matrices 4×4

$$\left(\left(\begin{array}{c|c} \bar{\rho_1} & \bar{\tau_1} \\ \hline \mathbf{0} & 1 \end{array} \right), \ldots, \left(\begin{array}{c|c} \bar{\rho_n} & \bar{\tau_n} \\ \hline \mathbf{0} & 1 \end{array} \right) \right)$$

- en utilisant une *carte* $(q_1^1, q_1^2, \ldots, q_1^6, q_2^1, \ldots, q_n^6)$, par exemple :

$$(x_{\tau_1}, y_{\tau_1}, z_{\tau_1}, \Psi_1, \Theta_1, \Phi_1, \ldots, x_{\tau_n}, y_{\tau_n}, z_{\tau_n}, \Psi_n, \Theta_n, \Phi_n)$$

$$\text{ou} \qquad (x_{\tau_1}, y_{\tau_1}, z_{\tau_1}, {}^x\Phi_1, {}^y\Phi_1, {}^z\Phi_1, \ldots, x_{\tau_n}, y_{\tau_n}, z_{\tau_n}, {}^x\Phi_n, {}^y\Phi_n, {}^z\Phi_n)$$

Remarque 1.29 On gardera présent à l'esprit qu'une seule carte ne permet pas en général de décrire tout l'ensemble des déplacements. Il faudra donc s'assurer que tous les calculs effectués avec une telle description ne font intervenir que des déplacements qui font partie du domaine de validité de la carte choisi, ou alors effectuer un changement de carte le cas échéant.

1.2.2 variété d'un système de deux corps soumis à une liaison

Afin de ne pas compliquer inutilement les calculs que l'on sera amené à effectuer dans ce paragraphe, on suppose que l'on étudie le mouvement du système relativement au corps $\mathcal{S}_i$.

L'hypothèse d'holonomie, telle qu'elle a été présentée précédemment se traduit par le fait qu'une liaison ℓ_{ij} entre les corps $\mathcal{S}_i$ et $\mathcal{S}_j$ d'un système $\mathcal{S}$ est entièrement caractérisée par la donnée de l'ensemble des configurations relatives admissibles du système $\mathcal{S}_i \sqcup \mathcal{S}_j$ soumis à la liaison ℓ_{ij}, que l'on a noté $\mathcal{V}_{adm}^{ij}$.

Or on ne dispose d'aucune propriété sur $\mathcal{V}_{adm}^{ij}$.

Pour la suite de l'étude, et pour développer une théorie cinématique sur les mécanismes, il est nécessaire de restreindre la généralité des liaisons étudiées pour obtenir plus de renseignements sur cet ensemble.

C'est pourquoi, on rajoute à nos hypothèses une certaine condition de "régularité" des liaisons définie ci-dessous :

Définition 1.30 *Deux corps $\mathcal{S}_i$ et $\mathcal{S}_j$ d'un système $\mathcal{S}$ sont reliés par une liaison holonome régulière lorsque l'ensemble des configurations relatives à $\mathcal{S}_i$ admissibles du système restreint à ces deux corps $\mathcal{V}_{adm}^{ij}$ est une sous variété de $\mathcal{V}'(\mathcal{S}_i \sqcup \mathcal{S}_j)$, variété primitive de configurations de $\mathcal{S}_i \sqcup \mathcal{S}_j$ relativement à $\mathcal{S}_i$.*

Définition 1.31 *On appelle degré de contrainte d'une liaison holonome régulière ℓ_{ij} la codimension de $\mathcal{V}_{adm}^{ij}$, c'est à dire la différence entre la dimension de la variété primitive de configurations du système restreint aux deux corps $\mathcal{S}_i$ et $\mathcal{S}_j$ et la dimension de la variété de configurations de ce système.*

Ce degré est noté $\delta^c(\ell_{ij})$ ou $\delta(\ell_{ij})$ lorsqu'il n'y a pas de risque de confusion.

Remarque 1.32 Puisque la variété primitive d'un système de deux corps relativement à un de ces corps est de dimension 6, le degré de contrainte d'une liaison holonome régulière est toujours inférieur à 6, i.e. pour toute liaison ℓ_{ij}, $1 \leq \delta^c(\ell_{ij}) \leq 6$.

On peut exprimer la définition de la régularité des liaisons de deux autres façons, en effet, dire que $\mathcal{V}_{adm}^{ij}$ est une sous variété de $\mathcal{V}'(\mathcal{S}_i \sqcup \mathcal{S}_j)$ est équivalent à dire que :

i. toute configuration $[\phi^0_{ij}]$ de $\mathcal{V}^{ij}_{adm}$ possède un voisinage V dans $\mathcal{V}'(\mathcal{S}_i \sqcup \mathcal{S}_j)$ sur lequel on peut définir une submersion f vers $\mathbb{R}^{\delta(\ell_{ij})}$ telle que $\mathcal{V}^{ij}_{adm} \cap V = f^{-1}(0)$, ce qui est équivalent à définir sur V, $\delta(\ell_{ij})$ fonctions à valeurs réelles, dont les différentielles sont linéairement indépendantes, telles que

$$[\phi_{ij}] \in \mathcal{V}^{ij}_{adm} \Longleftrightarrow f_1([\phi_{ij}]) = f_2([\phi_{ij}]) = \ldots = f_{\delta(\ell_{ij})}([\phi_{ij}]) = 0$$

ii. autour de toute configuration $[\phi_{ij}]$ de $\mathcal{V}^{ij}_{adm}$ il existe un voisinage ouvert V de $[\phi_{ij}]$ dans $\mathcal{V}'(\mathcal{S}_i \sqcup \mathcal{S}_j)$ et un système de coordonnées locales $q_1, q_2, \ldots, q_6$ de $\mathcal{V}'(\mathcal{S}_i \sqcup \mathcal{S}_j)$ différentiable sur V, tel que $\mathcal{V}^{ij}_{adm} \cap V$ est défini par $q_1 = \ldots = q_{\delta(\ell_{ij})} = 0$

L'interprétation mécanique de l'existence d'une liaison holonome régulière entre deux corps est donc :

- d'une part, qu'il existe partout localement un certain nombre δ^c de relations $f([\phi_{ij}]) = 0$ que doivent satisfaire indépendamment les positions et les orientations relatives des corps liés, on appelle *contrainte* chacune de ces relations,
- d'autre part, que l'on peut toujours trouver, localement, un jeu de $6 - \delta^c$ paramètres pour décrire les configurations relatives de $\mathcal{S}_j$ par rapport à $\mathcal{S}_i$.

En fait, la plupart des liaisons que l'on rencontre dans les applications industrielles satisfont la condition de régularité définie au paragraphe précédent.

On va voir ci-dessous comment s'écrivent explicitement les contraintes sur des exemples de liaisons que l'on rencontre fréquemment à savoir :

- encastrement
- liaison rotoïde
- liaison prismatique
- liaison hélicoïdale

encastrement

Il s'agit d'une liaison possédant un degré de contrainte $\delta^c(enc) = 6$.

Les deux corps reliés sont totalement solidaires, voir figure (1.13), ce type de liaison n'est utilisé que lorsque les propriétés dynamiques des corps en question font qu'il n'est pas avantageux de ne considérer qu'un seul corps.

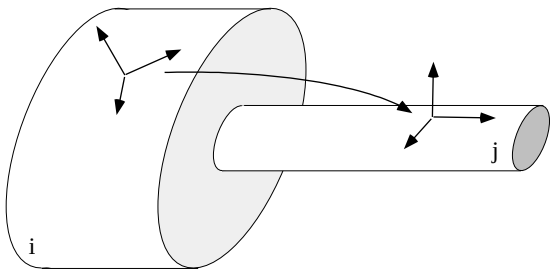


figure 1.13 : encastrement

On obtient les six contraintes en écrivant que la différence d'orientation entre les repères $\mathcal{R}_i$ et $\mathcal{R}_j$ est constante ainsi que la différence vectorielle entre leurs origines.

On peut obtenir des contraintes équivalentes en écrivant l'égalité entre deux repères l'un lié à $\mathcal{S}_i$, l'autre lié à $\mathcal{S}_j$.
Globalement, on obtient alors la relation suivante :

$$\exists d_{ij} \in \mathrm{Is}^+(I\!E), \forall t \in I, \phi_{ij}^t = d_{ij}$$

ou matriciellement :

$$\exists A_{ij} \in \mathrm{SO}_3, \exists b_{ij} \in I\!R^3, \forall t \in I, \left(\begin{array}{c|c} \bar{\rho}_{ij}^t & \bar{\tau}_{ij}^t \\ \hline \mathbf{0} & 1 \end{array}\right) = \left(\begin{array}{c|c} A_{ij} & b_{ij} \\ \hline \mathbf{0} & 1 \end{array}\right)$$

ou encore les six équations ci-dessous (en supposant que l'on se place dans le domaine de carte des angles d'Euler) :

$$\left\{\begin{array}{c} \exists x_{ij} \in I\!R, \\ \exists y_{ij} \in I\!R, \\ \exists y_{ij} \in I\!R, \\ \exists \Psi_{ij} \in [0, 2\pi[, \\ \exists \Theta_{ij} \in [0, 2\pi[, \\ \exists \Phi_{ij} \in [0, 2\pi[, \end{array}\right\} \forall t \in I, \left\{\begin{array}{rcl} x_{ij}^t - x_{ij} & = & 0 \\ y_{ij}^t - y_{ij} & = & 0 \\ z_{ij}^t - z_{ij} & = & 0 \\ \Psi_{ij}^t - \Psi_{ij} & = & 0 \\ \Theta_{ij}^t - \Theta_{ij} & = & 0 \\ \Phi_{ij}^t - \Phi_{ij} & = & 0 \end{array}\right.$$

La variété des configurations admissibles est un point, aucun paramètre n'est nécessaire pour définir la configuration relative de $\mathcal{S}_j$ par rapport à $\mathcal{S}_i$.

liaisons rotoïde, prismatique et hélicoïdale

Ces trois liaisons possèdent un degré de contrainte $\delta^c = 5$.
Les deux corps sont reliés par un axe de rotation, par un axe de translation, ou par un système vis-écrou – voir figures (1.14, 1.15, 1.16).

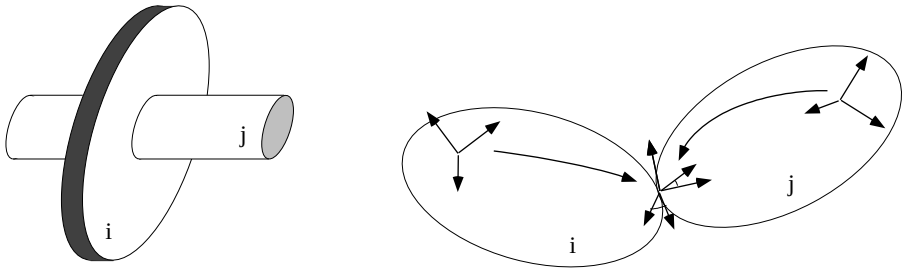


figure 1.14 : liaison rotoïde

Globalement, il s'agit d'écrire qu'il existe deux repères, l'un solidaire du corps $\mathcal{S}_i$, $\mathcal{R}_i^{ij}$, l'autre solidaire du corps $\mathcal{S}_j$, $\mathcal{R}_j^{ij}$ dont le mouvement relatif est soit une rotation d'axe donné, soit une translation le long d'un axe donné, soit un vissage d'axe et de pas donné.
Ceci revient à exprimer l'existence d'un sous-groupe à un paramètre de $\mathrm{Is}^+(I\!E)$ auquel appartient le déplacement transformant l'un de ces deux repères en l'autre.

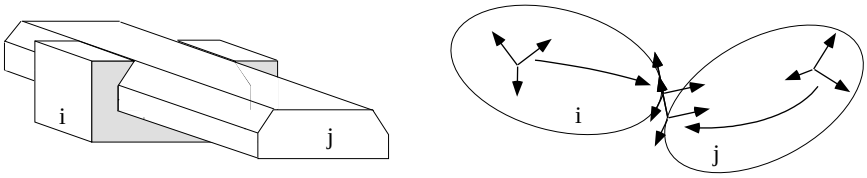


figure 1.15 : liaison prismatique

On obtient ainsi la relation suivante :

$$\left\{\begin{array}{c} \exists d_i \in \mathrm{Is}^+(I\!E), \\ \exists d_j \in \mathrm{Is}^+(I\!E), \end{array}\right\} \forall t \in I, (d_i)^{-1} \circ \phi_{ij}^t \circ d_j \in \Lambda_{\ell_{ij}}$$

où $\Lambda_{\ell_{ij}}$ est le groupe de déplacement à un paramètre, selon le cas, des rotations d'axe donné, des translations le long d'un axe donné, ou des vissages d'axe et de pas donné. d_i et d_j sont respectivement les déplacements qui déterminent les positions et les orientations des repères $\mathcal{R}_i^{ij}$ et $\mathcal{R}_j^{ij}$ par rapport aux repères de référence des corps $\mathcal{S}_i$ et $\mathcal{S}_j$.
Matriciellement, cette relation se traduit par :

- dans le cas de la liaison rotoïde,

$$\left\{\begin{array}{c} \exists A_i \in \mathrm{SO}_3, \exists b_i \in I\!R^3, \\ \exists A_j \in \mathrm{SO}_3, \exists b_j \in I\!R^3, \\ \exists A_{ij} \in \mathrm{SO}_3, \end{array}\right\} \qquad \forall t \in I, \qquad \exists \Theta_{ij}^t \in [0, 2\pi[,$$

$$\left(\begin{array}{c|c} \bar{\rho}_{ij}^t & \bar{\tau}_{ij}^t \\ \hline \mathbf{0} & 1 \end{array}\right) \left(\begin{array}{c|c} A_j & b_j \\ \hline \mathbf{0} & 1 \end{array}\right) \left(\begin{array}{c|c} A_{ij} & \mathbf{0} \\ \hline \mathbf{0} & 1 \end{array}\right) = \left(\begin{array}{c|c} A_i & b_i \\ \hline \mathbf{0} & 1 \end{array}\right) \left(\begin{array}{c|c} A_{ij} & \mathbf{0} \\ \hline \mathbf{0} & 1 \end{array}\right) \left(\begin{array}{c|c} \bar{\zeta}(\Theta_{ij}^t) & \mathbf{0} \\ \hline \mathbf{0} & 1 \end{array}\right)$$

$$\text{avec} \qquad \bar{\zeta}(\Theta_{ij}^t) = \begin{pmatrix} \cos(\Theta_{ij}^t) & -\sin(\Theta_{ij}^t) & 0 \\ \sin(\Theta_{ij}^t) & \cos(\Theta_{ij}^t) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$\text{et donc} \qquad \left(\begin{array}{c|c} A_{ij} & \mathbf{0} \\ \hline \mathbf{0} & 1 \end{array}\right) \left(\begin{array}{c|c} \bar{\zeta}(\Theta_{ij}^t) & \mathbf{0} \\ \hline \mathbf{0} & 1 \end{array}\right) \left(\begin{array}{c|c} A_{ij} & \mathbf{0} \\ \hline \mathbf{0} & 1 \end{array}\right)^{-1} \in \Lambda_{\ell_{ij}}$$

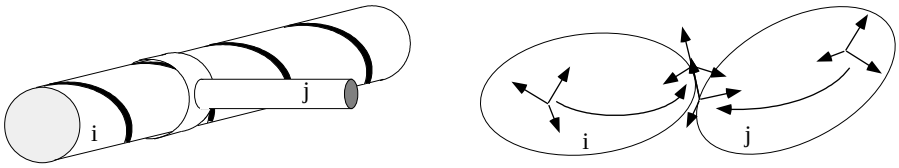


figure 1.16 : liaison hélicoïdale

- dans le cas d'une liaison prismatique,

$$\left\{\begin{array}{c} \exists A_i \in \mathrm{SO}_3, \exists b_i \in I\!R^3, \\ \exists A_j \in \mathrm{SO}_3, \exists b_j \in I\!R^3, \\ \exists b_{ij} \in I\!R^3, \end{array}\right\} \qquad \forall t \in I, \qquad \exists \Xi_{ij}^t \in I\!R,$$

$$\left(\begin{array}{c|c} \bar{\rho}_{ij}^t & \bar{\tau}_{ij}^t \\ \hline \mathbf{0} & 1 \end{array}\right)\left(\begin{array}{c|c} A_j & b_j \\ \hline \mathbf{0} & 1 \end{array}\right) = \left(\begin{array}{c|c} A_i & b_i \\ \hline \mathbf{0} & 1 \end{array}\right)\left(\begin{array}{c|c} \mathbf{Id} & \Xi_{ij}^t.b_{ij} \\ \hline \mathbf{0} & 1 \end{array}\right)$$

- dans le cas de la liaison hélicoïdale

$$\left\{\begin{array}{c} \exists A_i \in \mathrm{SO}_3, \exists b_i \in I\!R^3, \\ \exists A_j \in \mathrm{SO}_3, \exists b_j \in I\!R^3, \\ \exists A_{ij} \in \mathrm{SO}_3, \exists k \in I\!R, \end{array}\right\} \qquad \forall t \in I, \qquad \exists \Upsilon_{ij}^t \in I\!R,$$

$$\left(\begin{array}{c|c} \bar{\rho}_{ij}^t & \bar{\tau}_{ij}^t \\ \hline \mathbf{0} & 1 \end{array}\right)\left(\begin{array}{c|c} A_j & b_j \\ \hline \mathbf{0} & 1 \end{array}\right)\left(\begin{array}{c|c} A_{ij} & \mathbf{0} \\ \hline \mathbf{0} & 1 \end{array}\right) = \left(\begin{array}{c|c} A_i & b_i \\ \hline \mathbf{0} & 1 \end{array}\right)\left(\begin{array}{c|c} A_{ij} & \mathbf{0} \\ \hline \mathbf{0} & 1 \end{array}\right)\left(\begin{array}{c|c} \bar{\zeta}(\Upsilon_{ij}^t) & \vec{\zeta}_k(\Upsilon_{ij}^t) \\ \hline \mathbf{0} & 1 \end{array}\right)$$

$$\text{avec} \qquad \bar{\zeta}(\Upsilon_{ij}^t) = \begin{pmatrix} \cos(\Upsilon_{ij}^t) & -\sin(\Upsilon_{ij}^t) & 0 \\ \sin(\Upsilon_{ij}^t) & \cos(\Upsilon_{ij}^t) & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad \text{et} \qquad \vec{\zeta}_k(\Upsilon_{ij}^t) = \begin{pmatrix} 0 \\ 0 \\ \Upsilon_{ij}^t.k \end{pmatrix}$$

Remarque 1.33 Dans les expressions précédentes, A_i, b_i, A_j, b_j, A_{ij}, b_{ij} et k sont des constantes décrivant la nature de la liaison ℓ_{ij}, plus précisément, donnant l'axe de rotation, l'axe de translation, ou l'axe et le pas du vissage (c'est à dire les positions et les orientations de $\mathcal{R}_i^{ij}$ et de $\mathcal{R}_j^{ij}$ par rapport aux repères de références $\mathcal{R}_i$ et $\mathcal{R}_j$ des corps $\mathcal{S}_i$ et $\mathcal{S}_j$). Par contre, Θ_{ij}^t, Ξ_{ij}^t et Υ_{ij}^t sont les paramètres qui décrivent la configuration relative de $\mathcal{S}_j$ par rapport à $\mathcal{S}_i$ à l'instant t.

Pour obtenir les cinq contraintes scalaires, on exprime les conditions imposées par la liaison sur $\mathcal{R}_i^{ij}$ et $\mathcal{R}_j^{ij}$, ce qui donne :

- pour une liaison rotoïde, si l'on suppose que toutes les rotations considérées restent dans le domaine de carte des angles de Bryant,

$$\left\{\begin{array}{c} \exists x_i, \exists x_j \in I\!R, \\ \exists y_i, \exists y_j \in I\!R, \\ \exists z_i, \exists z_j \in I\!R, \\ \exists {}^x\Phi_i, \exists {}^x\Phi_j \in [0, 2\pi[, \\ \exists {}^y\Phi_i, \exists {}^y\Phi_j \in [0, 2\pi[, \\ \exists {}^z\Phi_i, \exists {}^z\Phi_j \in [0, 2\pi[, \end{array}\right\} \forall t \in I, \left\{\begin{array}{rcl} f_1(x_{ij}^t, {}^x\Phi_{ij}^t, {}^y\Phi_{ij}^t, {}^z\Phi_{ij}^t) & = & 0 \\ f_2(y_{ij}^t, {}^x\Phi_{ij}^t, {}^y\Phi_{ij}^t, {}^z\Phi_{ij}^t) & = & 0 \\ f_3(z_{ij}^t, {}^x\Phi_{ij}^t, {}^y\Phi_{ij}^t, {}^z\Phi_{ij}^t) & = & 0 \\ f_4({}^x\Phi_{ij}^t, {}^y\Phi_{ij}^t, {}^z\Phi_{ij}^t) & = & 0 \\ f_5({}^x\Phi_{ij}^t, {}^y\Phi_{ij}^t, {}^z\Phi_{ij}^t) & = & 0 \end{array}\right.$$

où f_1, f_2, f_3 qui traduisent l'égalité des origines des repères $\mathcal{R}_i^{ij}$ et $\mathcal{R}_j^{ij}$ sont obtenues en développant :

$$\left(\begin{array}{c|c} \bar{\rho}_{ij}^t & \bar{\tau}_{ij}^t \\ \hline \mathbf{0} & 1 \end{array}\right)\left(\begin{array}{c|c} A_j & b_j \\ \hline \mathbf{0} & 1 \end{array}\right)\left(\begin{array}{c} \mathbf{0} \\ \hline 1 \end{array}\right) = \left(\begin{array}{c|c} A_i & b_i \\ \hline \mathbf{0} & 1 \end{array}\right)\left(\begin{array}{c|c} \bar{\zeta}(\Theta_{ij}^t) & \mathbf{0} \\ \hline \mathbf{0} & 1 \end{array}\right)\left(\begin{array}{c} \mathbf{0} \\ \hline 1 \end{array}\right)$$

ont pour expressions :

$$\begin{aligned}
&f_1(x_{ij}^t, {}^x\Phi_{ij}^t, {}^y\Phi_{ij}^t, {}^z\Phi_{ij}^t) = \\
&\quad x_{ij}^t - x_i + \big(\cos({}^z\Phi_{ij}^t)\cos({}^y\Phi_{ij}^t)\big)x_j - \big(\sin({}^z\Phi_{ij}^t)\cos({}^y\Phi_{ij}^t)\big)y_j + \big(\sin({}^y\Phi_{ij}^t)\big)z_j \\
&f_2(y_{ij}^t, {}^x\Phi_{ij}^t, {}^y\Phi_{ij}^t, {}^z\Phi_{ij}^t) = \\
&\quad y_{ij}^t - y_i + \big(\sin({}^z\Phi_{ij}^t)\cos({}^x\Phi_{ij}^t) + \cos({}^z\Phi_{ij}^t)\sin({}^y\Phi_{ij}^t)\sin({}^x\Phi_{ij}^t)\big)x_j \\
&\quad + \big(\cos({}^z\Phi_{ij}^t)\cos({}^x\Phi_{ij}^t) - \sin({}^z\Phi_{ij}^t)\sin({}^y\Phi_{ij}^t)\sin({}^x\Phi_{ij}^t)\big)y_j - \big(\cos({}^y\Phi_{ij}^t)\sin({}^x\Phi_{ij}^t)\big)z_j \\
&f_3(z_{ij}^t, {}^x\Phi_{ij}^t, {}^y\Phi_{ij}^t, {}^z\Phi_{ij}^t) = \\
&\quad z_{ij}^t - z_i + \big(\sin({}^z\Phi_{ij}^t)\sin({}^x\Phi_{ij}^t) - \cos({}^z\Phi_{ij}^t)\sin({}^y\Phi_{ij}^t)\cos({}^x\Phi_{ij}^t)\big)x_j \\
&\quad + \big(\cos({}^z\Phi_{ij}^t)\sin({}^x\Phi_{ij}^t) + \sin({}^z\Phi_{ij}^t)\sin({}^y\Phi_{ij}^t)\cos({}^x\Phi_{ij}^t)\big)y_j + \big(\cos({}^y\Phi_{ij}^t)\cos({}^x\Phi_{ij}^t)\big)x_j
\end{aligned}$$

et où f_4 et f_5 qui traduisent que les repères $\mathcal{R}_i^{ij}$ et $\mathcal{R}_j^{ij}$ ont un axe commun – on choisit Oz pour ne pas compliquer inutilement les expressions calculées – sont obtenues en développant :

$${A_i}^{-1}\,\bar{\rho}_{ij}^t A_j \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

ont pour expressions :

$$\begin{aligned}
&f_4({}^x\Phi_{ij}^t, {}^y\Phi_{ij}^t, {}^z\Phi_{ij}^t) = \\
&\quad \cos({}^z\Phi_{ij}^t)\cos({}^y\Phi_{ij}^t)\sin({}^y\Phi_j) + \sin({}^z\Phi_{ij}^t)\cos({}^y\Phi_{ij}^t)\cos({}^y\Phi_j)\sin({}^x\Phi_j) + \sin({}^y\Phi_{ij}^t)\cos({}^y\Phi_j)\cos({}^x\Phi_j) - \sin({}^y\Phi_i) \\
&f_5({}^x\Phi_{ij}^t, {}^y\Phi_{ij}^t, {}^z\Phi_{ij}^t) = \\
&\quad -\sin({}^z\Phi_{ij}^t)\sin({}^y\Phi_{ij}^t)\cos({}^x\Phi_{ij}^t)\cos({}^y\Phi_j)\sin({}^x\Phi_j) - \cos({}^z\Phi_{ij}^t)\sin({}^y\Phi_{ij}^t)\cos({}^x\Phi_{ij}^t)\sin({}^y\Phi_j) \\
&\quad -\cos({}^z\Phi_{ij}^t)\sin({}^x\Phi_{ij}^t)\cos({}^y\Phi_j)\sin({}^x\Phi_j) + \sin({}^z\Phi_{ij}^t)\sin({}^x\Phi_{ij}^t)\sin({}^y\Phi_j) \\
&\quad +\cos({}^y\Phi_{ij}^t)\cos({}^x\Phi_{ij}^t)\cos({}^y\Phi_j)\cos({}^x\Phi_j) - \cos({}^y\Phi_i)\cos({}^x\Phi_i)
\end{aligned}$$

- pour une liaison prismatique, si l'on suppose que toutes les rotations considérées restent dans le domaine de carte des angles d'Euler,

$$\left\{\begin{array}{c} \exists x_i, \exists x_j, \exists x_{ij} \in I\!R, \\ \exists y_i, \exists y_j, \exists y_{ij} \in I\!R, \\ \exists z_i, \exists z_j, \exists y_{ij} \in I\!R, \\ \exists \Psi_{ij} \in [0, 2\pi[, \\ \exists \Theta_{ij} \in [0, 2\pi[, \\ \exists \Phi_{ij} \in [0, 2\pi[, \end{array}\right\} \forall t \in I, \left\{\begin{array}{rcl} f_1(x_{ij}^t, y_{ij}^t, \Psi_{ij}^t, \Theta_{ij}^t, \Phi_{ij}^t) & = & 0 \\ f_2(y_{ij}^t, z_{ij}^t, \Psi_{ij}^t, \Theta_{ij}^t, \Phi_{ij}^t) & = & 0 \\ \Psi_{ij}^t - \Psi_{ij} & = & 0 \\ \Theta_{ij}^t - \Theta_{ij} & = & 0 \\ \Phi_{ij}^t - \Phi_{ij} & = & 0 \end{array}\right.$$

où les trois dernières relations traduisent l'égalité des orientations des repères $\mathcal{R}_i^{ij}$ et $\mathcal{R}_j^{ij}$ et où f_1 et f_2 qui traduisent la colinéarité à un vecteur donné du vecteur reliant leurs origines sont obtenues en développant :

$$\overrightarrow{O_i^{ij}O_j^{ij}} \wedge b_{ij} = 0$$

avec
$$\left(\begin{array}{c|c} \overrightarrow{O_i^{ij}O_j^{ij}} & \\ \hline 1 & \end{array}\right) = \left(\begin{array}{c|c} \bar{\rho}_{ij}^t & \bar{\tau}_{ij}^t \\ \hline \mathbf{0} & 1 \end{array}\right) \left(\begin{array}{c|c} A_j & b_j \\ \hline \mathbf{0} & 1 \end{array}\right) \left(\begin{array}{c} \mathbf{0} \\ \hline 1 \end{array}\right) - \left(\begin{array}{c|c} A_i & b_i \\ \hline \mathbf{0} & 1 \end{array}\right) \left(\begin{array}{c} \mathbf{0} \\ \hline 1 \end{array}\right)$$

ont pour expressions :

$$\begin{aligned}
&f_1(x_{ij}^t, y_{ij}^t, \Psi_{ij}^t, \Theta_{ij}^t, \Phi_{ij}^t) = \\
&\quad x_{ij}\big[y_{ij}^t - y_i + \big(\sin(\Psi_{ij}^t)\cos(\Phi_{ij}^t) + \cos(\Psi_{ij}^t)\cos(\Theta_{ij}^t)\sin(\Phi_{ij}^t)\big)x_j \\
&\qquad + \big(\cos(\Psi_{ij}^t)\cos(\Theta_{ij}^t)\cos(\Phi_{ij}^t) - \sin(\Psi_{ij}^t)\sin(\Phi_{ij}^t)\big)y_j - \big(\cos(\Psi_{ij}^t)\sin(\Theta_{ij}^t)\big)z_j\big] \\
&\quad - y_{ij}\big[x_{ij}^t - x_i + \big(\cos(\Psi_{ij}^t)\cos(\Phi_{ij}^t) - \sin(\Psi_{ij}^t)\cos(\Theta_{ij}^t)\sin(\Phi_{ij}^t)\big)x_j \\
&\qquad - \big(\cos(\Psi_{ij}^t)\sin(\Phi_{ij}^t) + \sin(\Psi_{ij}^t)\cos(\Theta_{ij}^t)\cos(\Phi_{ij}^t)\big)y_j + \big(\sin(\Psi_{ij}^t)\sin(\Theta_{ij}^t)\big)z_j\big] \\
&f_2(y_{ij}^t, z_{ij}^t, \Psi_{ij}^t, \Theta_{ij}^t, \Phi_{ij}^t) = \\
&\quad y_{ij}\big[z_{ij}^t - z_i + \big(\sin(\Theta_{ij}^t)\sin(\Phi_{ij}^t)\big)x_j + \big(\sin(\Theta_{ij}^t)\cos(\Phi_{ij}^t)\big)y_j + \big(\cos(\Theta_{ij}^t)\big)z_j\big] \\
&\quad - z_{ij}\big[y_{ij}^t - y_i + \big(\sin(\Psi_{ij}^t)\cos(\Phi_{ij}^t) + \cos(\Psi_{ij}^t)\cos(\Theta_{ij}^t)\sin(\Phi_{ij}^t)\big)x_j \\
&\qquad - \big(\sin(\Psi_{ij}^t)\sin(\Phi_{ij}^t) - \cos(\Psi_{ij}^t)\cos(\Theta_{ij}^t)\cos(\Phi_{ij}^t)\big)y_j - \big(\cos(\Psi_{ij}^t)\sin(\Theta_{ij}^t)\big)z_j\big]
\end{aligned}$$

- pour une liaison hélicoïdale, si l'on suppose que toutes les rotations considérées restent dans le domaine de carte des angles de Bryant, et en choisissant Oz comme axe de la liaison :

$$\left\{\begin{array}{c} \exists x_i, \exists x_j \in I\!R, \\ \exists y_i, \exists y_j \in I\!R, \\ \exists z_i, \exists z_j \in I\!R, \\ \exists {}^x\Phi_i, \exists {}^x\Phi_j \in [0, 2\pi[, \\ \exists {}^y\Phi_i, \exists {}^y\Phi_j \in [0, 2\pi[, \\ \exists {}^z\Phi_i, \exists {}^z\Phi_j \in [0, 2\pi[, \end{array}\right\} \exists k \in I\!R, \forall t \in I, \left\{\begin{array}{lcl} f_1(x_{ij}^t, y_{ij}^t, {}^x\Phi_{ij}^t, {}^y\Phi_{ij}^t, {}^z\Phi_{ij}^t) & = & 0 \\ f_2(y_{ij}^t, z_{ij}^t, {}^x\Phi_{ij}^t, {}^y\Phi_{ij}^t, {}^z\Phi_{ij}^t) & = & 0 \\ f_3({}^x\Phi_{ij}^t, {}^y\Phi_{ij}^t, {}^z\Phi_{ij}^t) & = & 0 \\ f_4({}^x\Phi_{ij}^t, {}^y\Phi_{ij}^t, {}^z\Phi_{ij}^t) & = & 0 \\ f_5(x_{ij}^t, y_{ij}^t, z_{ij}^t, {}^x\Phi_{ij}^t, {}^y\Phi_{ij}^t, {}^z\Phi_{ij}^t) & = & 0 \end{array}\right.$$

où f_1 et f_2 qui traduisent l'appartenance à l'axe Oz – plus précisément l'axe $O_j^{ij}z_j^{ij}$ – du vecteur reliant les origines des repères $\mathcal{R}_i^{ij}$ et $\mathcal{R}_j^{ij}$ sont obtenues en développant :

$$\overrightarrow{O_i^{ij}O_j^{ij}} \wedge A_i \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = 0$$

avec
$$\left(\begin{array}{c|c} \overrightarrow{O_i^{ij}O_j^{ij}} & \\ \hline 1 & \end{array}\right) = \left(\begin{array}{c|c} \bar{\rho}_{ij}^t & \bar{\tau}_{ij}^t \\ \hline \mathbf{0} & 1 \end{array}\right) \left(\begin{array}{c|c} A_j & b_j \\ \hline \mathbf{0} & 1 \end{array}\right) \left(\begin{array}{c} \mathbf{0} \\ \hline 1 \end{array}\right) - \left(\begin{array}{c|c} A_i & b_i \\ \hline \mathbf{0} & 1 \end{array}\right) \left(\begin{array}{c} \mathbf{0} \\ \hline 1 \end{array}\right)$$

ont pour expressions :

$$
\begin{aligned}
&f_1(x_{ij}^t, y_{ij}^t, {}^x\Phi_{ij}^t, {}^y\Phi_{ij}^t, {}^z\Phi_{ij}^t) = \\
&\quad -[\cos({}^y\Phi_i)\sin({}^x\Phi_i)]\big[x_{ij}^t - x_i + \big(\cos({}^z\Phi_{ij}^t)\cos({}^y\Phi_{ij}^t)\big)x_j \\
&\qquad -\big(\sin({}^z\Phi_{ij}^t)\cos({}^y\Phi_{ij}^t)\big)y_j + \big(\sin({}^y\Phi_{ij}^t)\big)z_j\big] \\
&\quad -[\sin({}^y\Phi_i)]\big[y_{ij}^t - y_i + \big(\sin({}^z\Phi_{ij}^t)\cos({}^x\Phi_{ij}^t) + \cos({}^z\Phi_{ij}^t)\sin({}^y\Phi_{ij}^t)\sin({}^x\Phi_{ij}^t)\big)x_j \\
&\qquad +\big(\cos({}^z\Phi_{ij}^t)\cos({}^x\Phi_{ij}^t) - \sin({}^z\Phi_{ij}^t)\sin({}^y\Phi_{ij}^t)\sin({}^x\Phi_{ij}^t)\big)y_j - \big(\cos({}^y\Phi_{ij}^t)\sin({}^x\Phi_{ij}^t)\big)z_j\big] \\
&f_2(y_{ij}^t, z_{ij}^t, {}^x\Phi_{ij}^t, {}^y\Phi_{ij}^t, {}^z\Phi_{ij}^t) = \\
&\quad [\cos({}^y\Phi_i)\cos({}^x\Phi_i)]\big[y_{ij}^t - y_i + \big(\cos({}^z\Phi_{ij}^t)\sin({}^y\Phi_{ij}^t)\sin({}^x\Phi_{ij}^t) + \sin({}^z\Phi_{ij}^t)\cos({}^x\Phi_{ij}^t)\big)x_j \\
&\qquad +\big(\cos({}^z\Phi_{ij}^t)\cos({}^x\Phi_{ij}^t) - \sin({}^z\Phi_{ij}^t)\sin({}^y\Phi_{ij}^t)\sin({}^x\Phi_{ij}^t)\big)y_j - \big(\cos({}^y\Phi_{ij}^t)\sin({}^x\Phi_{ij}^t)\big)z_j\big] \\
&\quad +[\cos({}^y\Phi_i)\sin({}^x\Phi_i)]\big[z_{ij}^t - z_i + \big(\sin({}^z\Phi_{ij}^t)\sin({}^x\Phi_{ij}^t) - \cos({}^z\Phi_{ij}^t)\sin({}^y\Phi_{ij}^t)\cos({}^x\Phi_{ij}^t)\big)x_j \\
&\qquad +\big(\cos({}^z\Phi_{ij}^t)\sin({}^x\Phi_{ij}^t) + \sin({}^z\Phi_{ij}^t)\sin({}^y\Phi_{ij}^t)\cos({}^x\Phi_{ij}^t)\big)y_j + \big(\cos({}^y\Phi_{ij}^t)\cos({}^x\Phi_{ij}^t)\big)z_j\big]
\end{aligned}
$$

où f_3 et f_4 qui traduisent que les repères $\mathcal{R}_i^{ij}$ et $\mathcal{R}_j^{ij}$ ont un axe commun – en l'occurence Oz – sont obtenues comme f_4 et f_5 dans le cas des liaisons rotoïdes ont pour expressions :

$$
\begin{aligned}
&f_3({}^x\Phi_{ij}^t, {}^y\Phi_{ij}^t, {}^z\Phi_{ij}^t) = \\
&\quad \cos({}^z\Phi_{ij}^t)\cos({}^y\Phi_{ij}^t)\sin({}^y\Phi_j) + \sin({}^z\Phi_{ij}^t)\cos({}^y\Phi_{ij}^t)\cos({}^y\Phi_j)\sin({}^x\Phi_j) + \sin({}^y\Phi_{ij}^t)\cos({}^y\Phi_j)\cos({}^x\Phi_j) - \sin({}^y\Phi_i) \\
&f_4({}^x\Phi_{ij}^t, {}^y\Phi_{ij}^t, {}^z\Phi_{ij}^t) = \\
&\quad -\sin({}^z\Phi_{ij}^t)\sin({}^y\Phi_{ij}^t)\cos({}^x\Phi_{ij}^t)\cos({}^y\Phi_j)\sin({}^x\Phi_j) - \cos({}^z\Phi_{ij}^t)\sin({}^y\Phi_{ij}^t)\cos({}^x\Phi_{ij}^t)\sin({}^y\Phi_j) \\
&\quad -\cos({}^z\Phi_{ij}^t)\sin({}^x\Phi_{ij}^t)\cos({}^y\Phi_j)\sin({}^x\Phi_j) + \sin({}^z\Phi_{ij}^t)\sin({}^x\Phi_{ij}^t)\sin({}^y\Phi_j) \\
&\quad +\cos({}^y\Phi_{ij}^t)\cos({}^x\Phi_{ij}^t)\cos({}^y\Phi_j)\cos({}^x\Phi_j) - \cos({}^y\Phi_i)\cos({}^x\Phi_i)
\end{aligned}
$$

et où f_5 qui traduit que la translation le long de l'axe du vissage est proportionnelle à la rotation autour de cette axe et est obtenue en développant :

$$
\cos\Big(\frac{1}{k}\left[(\bar{\rho}_{ij}^t b_j + \bar{\tau}_{ij}^t - b_i)\cdot A_i \begin{pmatrix}0\\0\\1\end{pmatrix}\right]\Big) = \left[\bar{\rho}_{ij}^t A_j \begin{pmatrix}1\\0\\0\end{pmatrix}\cdot A_i \begin{pmatrix}1\\0\\0\end{pmatrix}\right]
$$

a pour expression :

$$
\begin{aligned}
&f_5(x_{ij}^t, y_{ij}^t, z_{ij}^t, {}^x\Phi_{ij}^t, {}^y\Phi_{ij}^t, {}^z\Phi_{ij}^t) = \\
&\cos\Big(\tfrac{1}{k}\big([\sin({}^y\Phi_i)]\big[x_{ij}^t - x_i + \big(\cos({}^z\Phi_{ij}^t)\cos({}^y\Phi_{ij}^t)\big)x_j - \big(\sin({}^z\Phi_{ij}^t)\cos({}^y\Phi_{ij}^t)\big)y_j + \big(\sin({}^y\Phi_{ij}^t)\big)z_j\big] \\
&\quad -[\cos({}^y\Phi_i)\sin({}^x\Phi_i)]\big[y_{ij}^t - y_i + \big(\sin({}^z\Phi_{ij}^t)\cos({}^x\Phi_{ij}^t) + \cos({}^z\Phi_{ij}^t)\sin({}^y\Phi_{ij}^t)\sin({}^x\Phi_{ij}^t)\big)x_j \\
&\quad\quad +\big(\cos({}^z\Phi_{ij}^t)\cos({}^x\Phi_{ij}^t) - \sin({}^z\Phi_{ij}^t)\sin({}^y\Phi_{ij}^t)\sin({}^x\Phi_{ij}^t)\big)y_j - \big(\cos({}^y\Phi_{ij}^t)\sin({}^x\Phi_{ij}^t)\big)z_j\big] \\
&\quad +[\cos({}^y\Phi_i)\cos({}^x\Phi_i)]\big[z_{ij}^t - z_i + \big(\sin({}^z\Phi_{ij}^t)\sin({}^x\Phi_{ij}^t) - \cos({}^z\Phi_{ij}^t)\sin({}^y\Phi_{ij}^t)\cos({}^x\Phi_{ij}^t)\big)x_j \\
&\quad\quad +\big(\cos({}^z\Phi_{ij}^t)\sin({}^x\Phi_{ij}^t) + \sin({}^z\Phi_{ij}^t)\sin({}^y\Phi_{ij}^t)\cos({}^x\Phi_{ij}^t)\big)y_j + \big(\cos({}^y\Phi_{ij}^t)\cos({}^x\Phi_{ij}^t)\big)z_j\big]\Big)\Big) \\
&-[\cos({}^z\Phi_i)\cos({}^y\Phi_i)]\big[\cos({}^z\Phi_{ij}^t)\sin({}^y\Phi_{ij}^t)\cos({}^z\Phi_j)\cos({}^y\Phi_j) - \sin({}^z\Phi_{ij}^t)\cos({}^y\Phi_{ij}^t)\cos({}^z\Phi_j)\sin({}^y\Phi_j)\sin({}^x\Phi_j) \\
&\quad -\sin({}^z\Phi_{ij}^t)\cos({}^y\Phi_{ij}^t)\sin({}^z\Phi_j)\cos({}^x\Phi_j) - \sin({}^y\Phi_{ij}^t)\cos({}^z\Phi_j)\sin({}^y\Phi_j)\cos({}^x\Phi_j) \\
&\quad +\sin({}^y\Phi_{ij}^t)\sin({}^z\Phi_j)\sin({}^x\Phi_j)\big] \\
&-[\cos({}^z\Phi_i)\sin({}^y\Phi_i)\sin({}^x\Phi_i) + \sin({}^z\Phi_i)\cos({}^x\Phi_i)]\big[\cos({}^z\Phi_{ij}^t)\sin({}^y\Phi_{ij}^t)\sin({}^x\Phi_{ij}^t)\cos({}^z\Phi_j)\cos({}^y\Phi_j) \\
&\quad +\sin({}^z\Phi_{ij}^t)\cos({}^x\Phi_{ij}^t)\cos({}^z\Phi_j)\cos({}^y\Phi_j) - \sin({}^z\Phi_{ij}^t)\sin({}^y\Phi_{ij}^t)\sin({}^x\Phi_{ij}^t)\cos({}^z\Phi_j)\sin({}^y\Phi_j)\sin({}^x\Phi_j) \\
&\quad -\sin({}^z\Phi_{ij}^t)\sin({}^y\Phi_{ij}^t)\sin({}^x\Phi_{ij}^t)\sin({}^z\Phi_j)\cos({}^x\Phi_j) + \cos({}^z\Phi_{ij}^t)\cos({}^x\Phi_{ij}^t)\cos({}^z\Phi_j)\sin({}^y\Phi_j)\sin({}^x\Phi_j) \\
&\quad +\cos({}^z\Phi_{ij}^t)\cos({}^x\Phi_{ij}^t)\sin({}^z\Phi_j)\cos({}^x\Phi_j) - \cos({}^y\Phi_{ij}^t)\sin({}^x\Phi_{ij}^t)\sin({}^z\Phi_j)\sin({}^x\Phi_j) \\
&\quad +\cos({}^y\Phi_{ij}^t)\sin({}^x\Phi_{ij}^t)\cos({}^z\Phi_j)\sin({}^y\Phi_j)\cos(\Phi_j)\big] \\
&-[\sin({}^z\Phi_i)\sin({}^x\Phi_i) - \cos({}^z\Phi_i)\sin({}^y\Phi_i)\cos({}^x\Phi_i)]\big[-\cos({}^z\Phi_{ij}^t)\sin({}^y\Phi_{ij}^t)\cos({}^x\Phi_{ij}^t)\cos({}^z\Phi_j)\cos({}^y\Phi_j) \\
&\quad +\sin({}^z\Phi_{ij}^t)\sin({}^x\Phi_{ij}^t)\cos({}^z\Phi_j)\cos({}^y\Phi_j) + \sin({}^z\Phi_{ij}^t)\sin({}^y\Phi_{ij}^t)\cos({}^x\Phi_{ij}^t)\cos({}^z\Phi_j)\sin({}^y\Phi_j)\sin({}^x\Phi_j) \\
&\quad +\sin({}^z\Phi_{ij}^t)\sin({}^y\Phi_{ij}^t)\cos({}^x\Phi_{ij}^t)\sin({}^z\Phi_j)\cos({}^x\Phi_j) + \cos({}^z\Phi_{ij}^t)\sin({}^x\Phi_{ij}^t)\cos({}^z\Phi_j)\sin({}^y\Phi_j)\sin({}^x\Phi_j) \\
&\quad +\cos({}^z\Phi_{ij}^t)\sin({}^x\Phi_{ij}^t)\sin({}^z\Phi_j)\cos({}^x\Phi_j) - \cos({}^y\Phi_{ij}^t)\cos({}^x\Phi_{ij}^t)\cos({}^z\Phi_j)\sin({}^y\Phi_j)\cos({}^x\Phi_j) \\
&\quad +\cos({}^y\Phi_{ij}^t)\cos({}^x\Phi_{ij}^t)\sin({}^z\Phi_j)\sin({}^x\Phi_j)\big]
\end{aligned}
$$

Remarque 1.34 Pour toutes ces liaisons, on a vu que l'on peut faire agir sur $\mathcal{V}_{adm}$ un sous-groupe de SO_3 qui est en fait un groupe de Lie à un paramètre, et que l'on peut écrire $\mathcal{V}_{adm}$ sous la forme :

$$\mathcal{V}_{adm} = d_i \Lambda_{\ell_{ij}} (d_j)^{-1}$$

Ce n'est bien sûr pas le cas de toutes les liaisons holonomes régulières. Considérons, par exemple, un mobile astreint à rester en contact avec un plan par le biais de deux attaches à un rail. Cette liaison satisfait bien la condition de régularité telle qu'elle est présentée dans la définition (1.30) dès que l'équation du rail dans le plan est suffisamment régulière mais sa variété de configurations n'est stable par l'action d'aucun sous-groupe du groupe des déplacements.

Toutefois la plupart des liaisons mécaniques que l'on rencontre dans les mécanismes industriels peuvent être caractérisées par un sous-groupe de SO_3, soit à un paramètre comme celles étudiées précédemment, soit à deux paramètres dans le cas du cardan (ou joint universel) et de la coulisse rotoïde, voire à trois paramètres pour la rotule ou pour le joint plan – voir figures (1.17, 1.18, 1.19, 1.20).

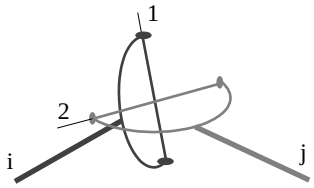


figure 1.17 : joint universel

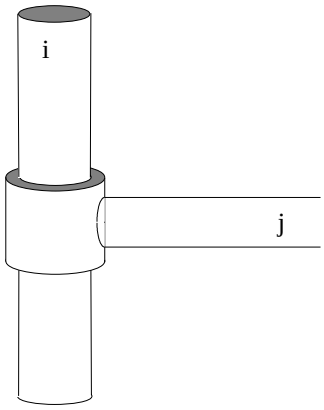


figure 1.18 : coulisse rotoïde

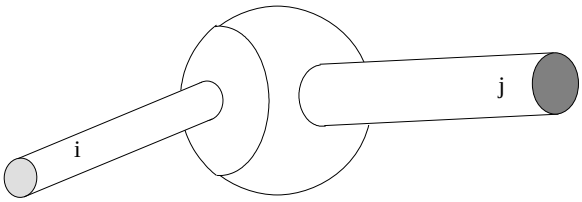


figure 1.19 : rotule

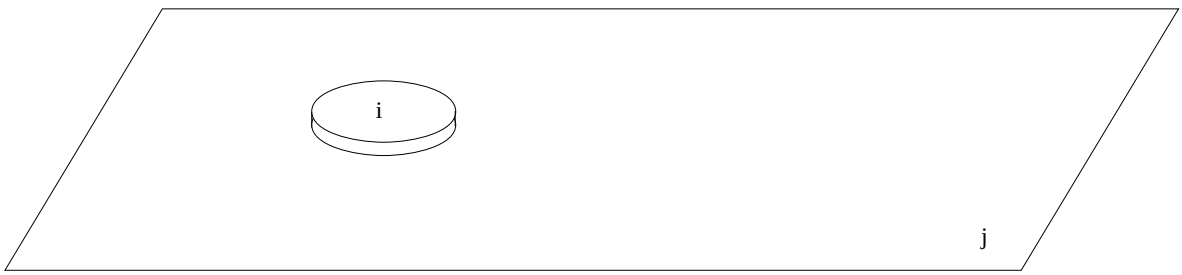


figure 1.20 : joint plan

1.2.3 variété de configurations d'un mécanisme

Propriété 1.35 *Lorsque l'ensemble des configurations relatives à $\mathcal{S}_i$ admissibles d'un système mécanique $\mathcal{S}$ composé de $n+1$ corps est une sous-variété de sa variété primitive de configurations relativement à $\mathcal{S}_i$ alors pour tout entier k plus petit que n, l'ensemble des configurations relatives à $\mathcal{S}_k$ admissibles de ce système est une sous-variété de sa variété primitive de configurations relativement à $\mathcal{S}_k$.*

Cette propriété provient de l'isomorphisme décrit ci-après entre l'ensemble des configurations relatives à $\mathcal{S}_i$ admissibles et l'ensemble des configurations relatives à $\mathcal{S}_k$ admissibles :

$$(\phi_{i1},\ldots,\phi_{i(i-1)},\phi_{i(i+1)},\ldots,\phi_{in})$$
$$\downarrow$$
$$(\phi_{i1}(\phi_{ik})^{-1},\ldots,\phi_{i(i-1)}(\phi_{ik})^{-1},(\phi_{ik})^{-1},\phi_{i(i+1)}(\phi_{ik})^{-1},\ldots,\phi_{i(k-1)}(\phi_{ik})^{-1},\phi_{i(k+1)}(\phi_{ik})^{-1},\ldots\phi_{in}(\phi_{ik})^{-1})$$

Remarque 1.36 Cet isomorphisme est en fait un difféomorphisme, à cause du caractère analytique de la multiplication et de l'inverse dans $\mathrm{Is}^+(\mathbb{E})$, donc, lorsque les ensembles de configurations relatives sont des sous-variétés des variétés primitives, elles sont toutes deux à deux difféomorphes.

Définition 1.37 *Lorsque le système vérifie l'hypothèse de la propriété précédente, on appelle degré de liberté de ce système la dimension de cette sous-variété.*

On note ce degré de liberté $\delta^l(\mathcal{S})$ ou $\delta(\mathcal{S})$ lorsque l'indice ^{l} n'est pas indispensable à la bonne compréhension.

Remarque 1.38 Le degré de contrainte d'une liaison ne dépend que de la nature de cette liaison, alors que le degré de liberté d'un système dépend, bien entendu, aussi du nombre de corps et de l'architecture du système.

Dans le cas d'un système composé de deux corps, dire que la liaison entre ces corps est régulière, c'est dire que le système satisfait aux hypothèses de la propriété (1.35), et la variété primitive de configurations du système étant de dimension 6, on doit avoir : $\delta^c(\ell_{ij}) + \delta^l(\mathcal{S}) = 6$.

Les deux propositions suivantes montre que lorsqu'un système composé de trois corps comporte deux liaisons régulières, les ensembles de configurations sont aussi des sous-variétés.

Proposition 1.39 *On considère le système $\mathcal{S}$ composé des corps $\mathcal{S}_i$, $\mathcal{S}_j$ et $\mathcal{S}_k$ comportant deux liaisons ℓ_{ij} et ℓ_{ik}. Si ces deux liaisons sont régulières alors l'ensemble des configurations relatives à $\mathcal{S}_i$ admissibles de $\mathcal{S}$ est une sous-variété de sa variété primitive de configurations relativement à $\mathcal{S}_0$.*

> Puisque la liaison ℓ_{ij} est régulière, $\mathcal{V}_{adm}^{ij}$ est une sous-variété de $\mathcal{V}'(\mathcal{S}_j)(=\mathbb{R}^3\times\mathrm{SO}_3)$, ainsi, tout point ϕ_{ij}^0 de $\mathcal{V}_{adm}^{ij}$ possède un voisinage V_{ij} dans $\mathcal{V}'(\mathcal{S}_j)$ sur lequel on peut définir une submersion f_{ij} vers $\mathbb{R}^{\delta(\ell_{ij})}$ telle que $\mathcal{V}_{adm}^{ij}\cap V = {f_{ij}}^{-1}(0)$.
> De la même manière, puisque la liaison ℓ_{ik} est régulière, $\mathcal{V}_{adm}^{ik}$ est une sous-variété de $\mathcal{V}'(\mathcal{S}_k)(=\mathbb{R}^3\times\mathrm{SO}_3)$, ainsi, tout point ϕ_{ik}^0 de $\mathcal{V}_{adm}^{ik}$ possède un voisinage V_{ik} dans $\mathcal{V}'(\mathcal{S}_k)$ sur lequel on peut définir une submersion f_{ik} vers $\mathbb{R}^{\delta(\ell_{ik})}$ telle que $\mathcal{V}_{adm}^{ik}\cap V = {f_{ik}}^{-1}(0)$.
> L'ensemble $\mathcal{V}_{adm}$ des configurations relatives à $\mathcal{S}_i$ admissibles de $\mathcal{S}$, sous-ensemble de sa variété primitive de configurations $\mathcal{V}'(\mathcal{S})$ $(=(\mathbb{R}^3\times\mathrm{SO}_3)\times(\mathbb{R}^3\times\mathrm{SO}_3))$ est donc tel que :

tout point $(\phi^0_{ij}, \phi_{ik}0)$ de cet ensemble possède un voisinage $V \times V'$ dans $\mathcal{V}'(\mathcal{S})$ sur lequel on peut définir une submersion f vers $I\!R^{\delta_{(\ell_{ij})}+\delta_{(\ell_{ik})}}$ telle que $\mathcal{V}_{adm} \cap (V \times V') = f^{-1}(0)$, on posera en effet

$$f(\phi_{ij}, \phi_{ik}) = \left\{ \begin{array}{c} f_{ij}(\phi_{ij}) \\ f_{ik}(\phi_{ik}) \end{array} \right\}$$

ce qui montre que $\mathcal{V}_{adm}$ est une sous-variété de $\mathcal{V}'(\mathcal{S})$ de dimension $12-(\delta_{(\ell_{ij})}+\delta_{(\ell_{ik})})$. ◇

Proposition 1.40 *On considère le système $\mathcal{S}$ composé des corps $\mathcal{S}_i$, $\mathcal{S}_j$ et $\mathcal{S}_k$ comportant deux liaisons ℓ_{ij} et ℓ_{jk}. Si ces deux liaisons sont régulières alors l'ensemble des configurations relatives à $\mathcal{S}_i$ admissibles de $\mathcal{S}$ est une sous-variété de sa variété primitive de configurations relativement à $\mathcal{S}_0$.*

On peut déduire cette proposition de la proposition précédente en s'appuyant sur la propriété (1.35) et sur la remarque (1.36) après avoir effectué une permutation sur les indices.

Une démonstration explicite ressemble à celle de la proposition précédente :

Puisque la liaison ℓ_{ij} est régulière, $\mathcal{V}^{ij}_{adm}$ est une sous-variété de $\mathcal{V}'(\mathcal{S}_j)(= I\!R^3 \times \mathrm{SO}_3)$, ainsi, tout point ϕ^0_{ij} de $\mathcal{V}^{ij}_{adm}$ possède un voisinage V_{ij} dans $\mathcal{V}'(\mathcal{S}_j)$ sur lequel on peut définir une submersion f_{ij} vers $I\!R^{\delta_{(\ell_{ij})}}$ telle que $\mathcal{V}^{ij}_{adm} \cap V = {f_{ij}}^{-1}(0)$.

De la même manière, puisque la liaison ℓ_{jk} est régulière, $\mathcal{V}^{jk}_{adm}$ est une sous-variété de $\mathcal{V}'(\mathcal{S}_k)(= I\!R^3 \times \mathrm{SO}_3)$, ainsi, tout point ϕ^0_{jk} de $\mathcal{V}^{jk}_{adm}$ possède un voisinage V_{jk} dans $\mathcal{V}'(\mathcal{S}_k)$ sur lequel on peut définir une submersion f_{jk} vers $I\!R^{\delta_{(\ell_{jk})}}$ telle que $\mathcal{V}^{jk}_{adm} \cap V = {f_{jk}}^{-1}(0)$.

L'ensemble $\mathcal{V}_{adm}$ des configurations relatives à $\mathcal{S}_i$ admissibles de $\mathcal{S}$, sous-ensemble de sa variété primitive de configurations $\mathcal{V}'(\mathcal{S})$ $(= (I\!R^3 \times \mathrm{SO}_3) \times (I\!R^3 \times \mathrm{SO}_3))$ est donc tel que :

tout point $(\phi^0_{ij}, \phi^0_{ik})$ de cet ensemble possède un voisinage $V \times V'$ dans $\mathcal{V}'(\mathcal{S})$ sur lequel on peut définir un morphisme f vers $I\!R^{\delta_{(\ell_{ij})}+\delta_{(\ell_{jk})}}$ telle que $\mathcal{V}_{adm} \cap (V \times V') = f^{-1}(0)$, on posera en effet

$$f(\phi_{ij}, \phi_{ik}) = \left\{ \begin{array}{c} f_{ij}(\phi_{ij}) \\ f_{jk}(\phi_{ik}(\phi_{ij})^{-1}) \end{array} \right\} \qquad \text{avec} \qquad f_{ik}(\phi_{ij}, \phi_{ik}) = f_{jk}(\phi_{ik}(\phi_{ij})^{-1})$$

Ce morphisme est bien une submersion dans la mesure où, en tout point $(\phi^0_{ij}, \phi^0_{ik})$, sa matrice jacobienne :

$$\left(\begin{array}{cc} \frac{df_{ij}}{d\phi_{ij}}\left(\phi^0_{ij}\right) & \mathbf{0} \\ \frac{\partial f_{ik}}{\partial \phi_{ij}}\left(\phi^0_{ij}, \phi^0_{ik}\right) & \frac{\partial f_{ik}}{\partial \phi_{ik}}\left(\phi^0_{ij}, \phi^0_{ik}\right) \end{array} \right)$$

est bien de rang maximum puisque si on note $\pi(\phi_1, \phi_2)$ le produit de deux déplacements,

$$\frac{\partial f_{ik}}{\partial \phi_{ik}}\left(\phi^0_{ij}, \phi^0_{ik}\right) = \frac{df_{jk}}{d\phi_{jk}}\left(\phi^0_{ik}(\phi^0_{ik})^{-1}\right).\frac{\partial \pi}{\partial \phi_1}\left(\phi^0_{ik}, (\phi^0_{ik})^{-1}\right)$$

Ainsi $\mathcal{V}_{adm}$ est bien une sous-variété de $\mathcal{V}'(\mathcal{S})$ de dimension $12-(\delta_{(\ell_{ij})}+\delta_{(\ell_{ik})})$. ◇

On va maintenant généraliser ce qui se passe pour les systèmes de deux et de trois corps aux systèmes de $n+1$ corps arborescents en établissant le théorème suivant :

Théorème 1.41 *On considère un système mécanique $\mathcal{S}$ composé de $n+1$ corps dont toutes les liaisons sont régulières. Si l'ensemble des liaisons $\mathcal{L}$ est tel que l'architecture de $\mathcal{S}$ soit arborescente, alors l'ensemble des configurations admissibles relatives de $\mathcal{S}$ est une sous-variété de sa variété primitive de configurations, et son degré de liberté vérifie :*

$$\delta^l(\mathcal{S}) = 6n - \sum_{\ell_{ij}\in\mathcal{L}} \delta^c(\ell_{ij})$$

On va démontrer ce théorème par récurrence sur le nombre de corps.
On a déjà vu plus haut, que pour un système à deux corps le théorème est vrai par la définition même de la régularité des liaisons, les propositions précédentes l'établisse pour $n = 3$.
Supposons donc que tout système mécanique d'au plus n corps vérifie l'énoncé du théorème et considérons un système mécanique $\mathcal{S}$ de $n+1$ corps dont l'architecture est arborescente.
Nécessairement, il existe parmi les corps qui composent le système $\mathcal{S}$, au moins un corps qui n'intervient que dans une liaison, notons le $\mathcal{S}_0$ et notons $\mathcal{S}_1$ le corps auquel il est lié.
Considérons le système $\mathcal{S}'$ composé des corps $\mathcal{S}_1, \ldots, \mathcal{S}_n$ et dont l'ensemble des liaisons est $\mathcal{L}' = \mathcal{L} \setminus \{\ell_{0,1}\}$.
L'architecture de ce système $\mathcal{S}'$ est arborescente et il vérifie donc l'hypothèse de récurrence. En conséquence l'ensemble de ses configurations relatives à $\mathcal{S}_1$ admissibles est une sous-variété de sa variété primitive de configurations $\mathcal{V}'(\mathcal{S}')$ $(= (I\!R^3 \times \mathrm{SO}_3)^{n-1})$.
Ainsi, tout point $(\phi^0_{1,2}, \ldots, \phi^0_{1,n})$ de $\mathcal{V}_{adm}(\mathcal{S}')$ possède un voisinage V' dans $\mathcal{V}'(\mathcal{S}')$ sur lequel on peut définir une submersion f' vers $I\!R^{6(n-1)-\delta^l(\mathcal{S}')}$ telle que $\mathcal{V}_{adm}(\mathcal{S}') \cap V = {f'}^{-1}(0)$.
D'autre part, puisque la liaison $\ell_{0,1}$ est régulière, $\mathcal{V}^{1,0}_{adm}$ est une sous-variété de $\mathcal{V}'(\mathcal{S}_0)(= I\!R^3 \times \mathrm{SO}_3)$, ainsi, tout point $\phi^0_{1,0}$ de $\mathcal{V}^{1,0}_{adm}$ possède un voisinage $V_{1,0}$ dans $\mathcal{V}'(\mathcal{S}_0)$ sur lequel on peut définir une submersion $f_{1,0}$ vers $I\!R^{\delta(\ell_{0,1})}$ telle que $\mathcal{V}^{1,0}_{adm} \cap V = {f_{1,0}}^{-1}(0)$.
L'ensemble $\mathcal{V}_{adm}$ des configurations relatives à $\mathcal{S}_1$ admissibles de $\mathcal{S}$, sous-ensemble de sa variété primitive de configurations $\mathcal{V}'(\mathcal{S})$ $(= (I\!R^3 \times \mathrm{SO}_3)^n)$ est donc tel que :
tout point $(\phi^0_{1,0}, \phi^0_{1,2}, \ldots, \phi^0_{1,n})$ de cet ensemble possède un voisinage $V \times V_{1,0}$ dans $\mathcal{V}'(\mathcal{S})$ sur lequel on peut définir une submersion f vers $I\!R^{\left(6(n-1)-\delta^l(\mathcal{S}')\right)+\delta_{(\ell_{1,0})}}$ telle que $\mathcal{V}_{adm} \cap (V' \times V_{1,0}) = f^{-1}(0)$, on posera en effet

$$f(\phi^0_{1,0}, \phi^0_{1,2}, \ldots, \phi^0_{1,n}) = \left\{ \begin{array}{c} f_{ij}(\phi^0_{1,2}, \ldots, \phi^0_{1,n}) \\ f_{1,0}(\phi_{1,0}) \end{array} \right\}$$

ce qui montre que $\mathcal{V}_{adm}$ est une sous-variété de $\mathcal{V}'(\mathcal{S})$ de dimension

$$6n - \big(6(n-1) - \delta^l(\mathcal{S}') + \delta_{(\ell_{1,0})}\big) = 6n - \left(\sum_{\ell_{ij}\in\mathcal{L}'} \delta^c(\ell_{ij}) + \delta_{(\ell_{1,0})}\right) = 6n - \sum_{\ell_{ij}\in\mathcal{L}} \delta^c(\ell_{ij})$$

◇

Remarque 1.42 L'hypothèse sur l'arborescence de la structure du mécanisme est essentielle dans la preuve du théorème précédent. Ceci explique pourquoi l'étude cinématique des mécanismes à structure bouclée – et, a fortiori leur étude dynamique – est beaucoup plus complexe. On verra, dans le troisième chapitre, différents moyens pour obtenir des renseignements sur l'ensemble des configurations admissibles relatives d'un système mécanique à structure bouclée.

1.2.4 variétés d'étude d'un mécanisme

Lorsque l'on a étudié les variétés de configurations de système de deux corps soumis à une liaison, on a vu que dans la plupart des cas on pouvait rattacher à la liaison un sous groupe à un ou plusieurs paramètres du groupe des déplacements.

Il semble alors naturel d'utiliser ce paramètre pour décrire le mouvement relatif d'un des corps lié, et lorsqu'on étudie un système comportant plus de deux corps d'écrire certaines contraintes à l'aide de ces paramètres.

Ceci amène à définir de manière générale les *variétés d'étude* des mécanismes :

Définition 1.43 *On appelle variété d'étude d'un mécanisme $\mathcal{S}$ toute sous-variété (au sens large) de sa variété primitive de configurations $\mathcal{V}'(\mathcal{S})$ qui contient l'ensemble de ses configurations (resp. relatives) admissibles $\mathcal{V}_{adm}(\mathcal{S})$.*

Remarque 1.44 Le degré de liberté d'un mécanisme est indépendant de la variété d'étude, mais il n'en est pas de même du degré de contrainte d'une liaison. En effet, une variété d'étude est souvent définie en tenant compte de certaines contraintes des liaisons du système.

On va voir sur un exemple simple l'intérêt de bien choisir les variétés d'études des mécanismes pour simplifier les calculs à effectuer.

On considère dans le plan[§] un système de 4 tiges rigides de longueurs l_0, l_1, l_2, l_3, articulées par des axes de rotations perpendiculaires au plan.

On se propose d'étudier le mouvement de ce système relativement à une des tiges notée $\mathcal{S}_0$, la variété primitive de ce système est l'ensemble $(\mathrm{Is}^+(P))^3$ produit cartésien de l'ensemble des déplacements du plan par lui-même, 3 fois. C'est une variété de dimension 9.

On représente classiquement un déplacement du plan comme le produit d'une rotation par une translation, $(\mathrm{Is}^+(P))^3$ étant assimilé au produit semi-direct $\mathbb{R}^2 \times \mathrm{SO}_2$, et totalement décrit par les deux cartes : (x, y, θ) avec $\theta \in]-\pi, \pi[$ et (x, y, θ) avec $\theta \in]0, 2\pi[$.

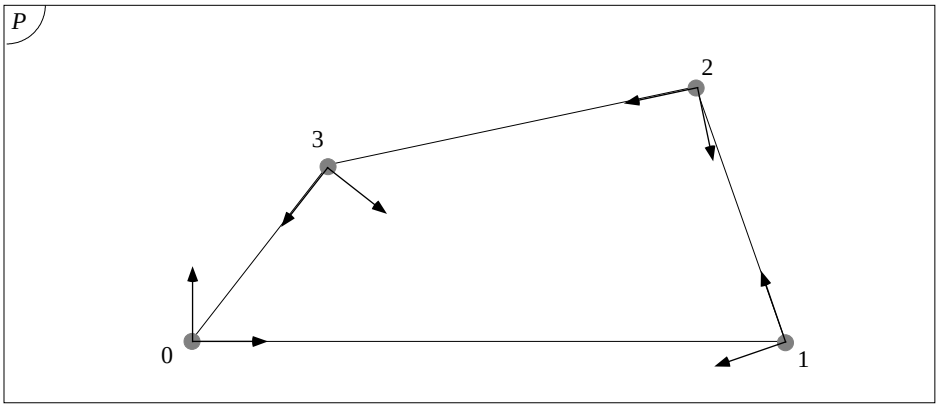


figure 1.21 : quatre tiges dans le plan

Ainsi, $\mathcal{V}'(\mathcal{S}) = \left(\mathbb{R}^2 \times \mathrm{SO}_2\right)^3$, et une configuration de $\mathcal{S}$ sera décrite par les neuf paramètres :

$$(x_1, y_1, \theta_1, x_2, y_2, \theta_2, x_3, y_3, \theta_3,)$$

[§]on désigne par *plan* l'espace affine euclidien réel de dimension 2, noté P.

et avec les choix des repères de références décrit figure (1.21), les contraintes s'écrivent (dans le domaine d'une des cartes) :

$$\left\{\begin{array}{rcl} x_1 & = & l_0 \\ y_1 & = & 0 \\ x_2 - x_1 & = & l_1\cos(\theta_1) \\ y_2 - y_1 & = & l_1\sin(\theta_1) \\ x_3 - x_2 & = & l_2\cos(\theta_2) \\ y_3 - y_2 & = & l_2\sin(\theta_2) \\ -x_3 & = & l_3\cos(\theta_3) \\ -y_3 & = & l_3\sin(\theta_3) \end{array}\right.$$

en traduisant simplement la coïncidence des extrémités des tiges.

Ce qui est intéressant, dans l'étude de ce système, est l'influence du caractère bouclé de son architecture ; si l'on ne tiens pas compte d'une (*resp.* de deux) des liaisons le système devient arborescent et il est clair que sa variété de configurations est alors difféomorphe à la sphère de dimension 3 (*resp.* au produit de $\mathbb{R}^2 \times \mathrm{SO}_2$ par la sphère de dimension 2).

En effet, dans le premier cas, c'est à dire en ne tenant pas compte de la liaison au point 0, les trois composantes en translation des configurations ϕ_1, ϕ_2 et ϕ_3 sont complètement déterminées par :

$$\begin{array}{rcl} \bar{\tau_1} & = & \begin{pmatrix} l_0 \\ 0 \end{pmatrix} \\ \bar{\tau_2} & = & \bar{\tau_1} + \bar{\rho_1}\begin{pmatrix} l_1 \\ 0 \end{pmatrix} \\ \bar{\tau_3} & = & \bar{\tau_2} + \bar{\rho_2}\begin{pmatrix} l_2 \\ 0 \end{pmatrix} \end{array}$$

on peut alors choisir comme variété d'étude la sous-variété de $\mathcal{V}'(\mathcal{S})$ des configurations satisfaisant ces conditions que l'on décrira par des cartes de la forme $(\Theta_1, \Theta_2, \Theta_3)$ avec $\Theta_1 = \theta_1$, $\Theta_2 = \theta_2 - \theta_1$ et $\Theta_3 = \theta_3 - \theta_2$ modulo les formules de changement de cartes.

La liaison dont on n'a pas tenu compte sera traduite par les contraintes :

$$\left\{\begin{array}{rcl} l_0 + l_1\cos(\Theta_1) + l_2\cos(\Theta_1+\Theta_2) + l_3\cos(\Theta_1+\Theta_2+\Theta_3) & = & 0 \\ l_1\sin(\Theta_1) + l_2\sin(\Theta_1+\Theta_2) + l_3\sin(\Theta_1+\Theta_2+\Theta_3) & = & 0 \end{array}\right.$$

Dans l'autre cas, c'est à dire en ne tenant compte ni de la liaison au point 2 ni de la liaison au point 3, les deux composantes en translation des configurations ϕ_1 et ϕ_3 sont

complètement déterminées par :

$$\begin{aligned} \bar{\tau_1} &= \begin{pmatrix} l_0 \\ 0 \end{pmatrix} \\ \bar{\tau_3} &= -\bar{\rho_3} \begin{pmatrix} l_3 \\ 0 \end{pmatrix} \end{aligned}$$

on peut alors choisir comme variété d'étude la sous-variété de $\mathcal{V}'(\mathcal{S})$ des configurations satisfaisant ces conditions que l'on décrira par des cartes de la forme $(\Theta_1, x_2, y_2, \theta_2, \Theta_3)$ avec $\Theta_1 = \theta_1$, $\Theta_3 = \theta_3 - \pi$ modulo les formules de changement de cartes.

Et les liaisons dont on n'a pas tenu compte seront traduites par les contraintes :

$$\begin{cases} x_2 &= l_0 + l_1 \cos(\Theta_1) \\ y_2 &= l_1 \sin(\Theta_1) \\ l_2 \cos(\Theta_2) &= l_3 \cos(\Theta_3) - l_1 \cos(\Theta_1) - l_0 \\ l_2 \sin(\Theta_2) &= l_3 \sin(\Theta_3) - l_1 \sin(\Theta_1) - l_0 \end{cases}$$

Il est clair que pour étudier la dynamique de ce système supposé arborescent – ne comportant que trois articulations – il est préférable de travailler dans une variété d'étude de dimension 3, donc de même dimension que le degré de liberté du système, plutôt que de travailler dans une variété d'étude de degré 5 en tenant compte de deux contraintes supplémentaires.

On reviendra sur cette exemple lorsque l'on s'intéressera à la modélisation des mécanismes à architectures bouclées pour voir ce qu'il en est dans ce cas.

1.3 modélisation du mouvement des mécanismes – étude dynamique

L'objet de l'étude dynamique du mouvement des mécanismes est l'obtention d'équations permettant de calculer la configuration d'un système à un instant donné en fonction d'une part de sa configuration – étendue aux paramètres cinématiques – initiale et d'autre part de ses paramètres physiques – les masses des corps, leurs inerties, leurs interactions connues dues par exemple à la présence de moteurs, de ressorts ou d'amortisseurs aux liaisons, les forces extérieures induites par exemple par un champ gravitationnel ou par un champ magnétique, ...

Dans le cas le plus simple du point matériel, le principe fondamental de la dynamique établit la proportionnalité entre l'accélération du mouvement de ce point et la somme des forces qui s'exercent sur lui par la traditionnelle formule vectorielle :

$$F = m.\gamma$$

On obtient ainsi trois équations différentielles du second ordre gouvernant la fonction vectorielle qui, à un instant donné, associe la position de ce point à cet instant.

La connaissance de la position du point à deux instants de référence permet alors de déterminer complètement cette fonction.

En ce qui concerne les mécanismes, il existe de nombreuses méthodes et de nombreux formalismes différents qui fournissent des ensembles d'équations différentielles gouvernant leurs mouvements. Ils ont été développés par ceux qui ont étudié, depuis quatre siècles, le comportement dynamique des systèmes mécaniques.

On va présenter ci-dessous les méthodes et les formalismes les plus courant existant dans le cadre de la mécanique classique – *Newton-Euler* et *d'Alembert* – et dans le cadre de la mécanique analytique – *Lagrange*, *Hamilton*, et plus récemment *Kane*.

Dans le chapitre suivant on illustrera ces méthodes en utilisant chacune d'entre elles pour mettre en équation le mouvement d'un système mécanique simple mais non trivial et on en tirera les enseignements concernant la génération automatique d'équations dans un logiciel de calcul symbolique.

Auparavant, il est nécessaire d'introduire quelques notions de base notamment à propos des grandeurs cinématiques, dynamiques et inertielles.

1.3.1 grandeurs cinématiques, cinétiques et dynamiques

On considère un mécanisme $\mathcal{S}$ composé de $n+1$ corps rigides $\mathcal{S}_0, \mathcal{S}_1, \ldots, \mathcal{S}_n$.

Quitte à être introduit arbitrairement pour cela, le corps $\mathcal{S}_0$ est fixe dans le repère de l'observateur du mouvement.

Ainsi, on utilisera systématiquement la notion de configuration relative au corps $\mathcal{S}_0$, et on se permettra, pour alléger les notations d'écrire ϕ_i au lieu de $\phi_{0,i}$.

On suppose que l'ensemble des configurations admissibles de $\mathcal{S}$ est une sous-variété de sa variété primitive de configurations $\mathcal{V}'$.

Un mouvement ϕ de ce mécanisme est, comme on l'a vu, une application d'un intervalle I de $\mathbb{R}$ dans $\mathcal{V}$ – la variété de configurations de $\mathcal{S}$ – qui, à un instant donné t de I, fait correspondre la configuration de $\mathcal{S}$ à cet instant :

$$\phi(t) = (\phi_1^t, \ldots, \phi_n^t)$$

On supposera que tous les mouvements considérés sont des applications deux fois continuement dérivables, ce qui, une fois de plus, exclut la possibilité de chocs entre les corps du système ou avec des objets extérieurs et suppose que la structure mécanique interne du système reste constante pendant tout l'intervalle de temps considéré – i.e. les liaisons sont indépendantes du temps.

trajectoires, vitesses et accélérations

Définition 1.45

- *On appelle* trajectoire *du système (resp. du i-ème corps du système) l'ensemble $\phi(I)$ (resp. $\phi_i(I)$) de ses configurations atteintes au cours du mouvement.*
- *On appelle* vitesse *du système (resp. du i-ème corps du système) à l'instant t, et on note $\dot{\phi}^t$ (resp. $\dot{\phi}_i^t$) le vecteur tangent à sa trajectoire, valeur de l'application linéaire tangente à ϕ (resp. à ϕ_i) en t :*

$$T_t\phi \;:\; \begin{array}{ccc} T_tI & \longrightarrow & T_{\phi^t}\mathcal{V} \\ 1 & \longmapsto & \dot{\phi}^t \end{array} \qquad \left(\text{resp.} \quad T_t\phi_i \;:\; \begin{array}{ccc} T_tI & \longrightarrow & T_{\phi_i^t}\mathcal{V}_i \\ 1 & \longmapsto & \dot{\phi}_i^t \end{array} \right)$$

l'application $\dot{\phi}$ (resp. $\dot{\phi}_i$) de I dans $T\mathcal{V}$ (resp. $T\mathcal{V}_i$) – le fibré tangent à $\mathcal{V}$ (resp. $\mathcal{V}_i$) – qui, à t fait correspondre $\dot{\phi}^t$ (resp. $\dot{\phi}_i^t$) est la vitesse associée au mouvement ϕ (resp. ϕ_i) du système (resp. du i-ème corps).

- *On appelle* accélération *du système (resp. du i-ème corps du système) à l'instant t, et on note $\ddot{\phi}^t$ (resp. $\ddot{\phi}_i^t$) le vecteur, valeur de l'application linéaire tangente à $\dot{\phi}$ (resp. à $\dot{\phi}_i$) en t :*

$$T_t\dot{\phi} \;:\; \begin{array}{ccc} T_tI & \longrightarrow & T_{\dot{\phi}^t}T\mathcal{V} \\ 1 & \longmapsto & \ddot{\phi}^t \end{array} \qquad \left(\text{resp.} \quad T_t\dot{\phi}_i \;:\; \begin{array}{ccc} T_tI & \longrightarrow & T_{\dot{\phi}_i^t}T\mathcal{V}_i \\ 1 & \longmapsto & \ddot{\phi}_i^t \end{array} \right)$$

La trajectoire, la vitesse et l'accélération – au sens habituel – de n'importe quel point matériel appartenant à un des corps du système peut se calculer à l'aide des grandeurs définies ci-dessus.

Considérons une particule m du système appartenant au corps $\mathcal{S}_i$ et $\overrightarrow{u_0}$ le vecteur $\overrightarrow{O_iM^0}$ exprimant la position de m à un instant de référence où le repère $\mathcal{R}_i$ coïncide avec le repère $\mathcal{R}_0$.
La position M^t de m à l'instant t est donnée par :

$$M^t = \phi_i^t(M^0) \qquad \text{i.e.} \qquad \overrightarrow{OM^t} = \overrightarrow{OO_i^t} + \overrightarrow{O_i^tM^t} = \tau_i^t + \rho_i^t(\overrightarrow{u_o})$$

Le vecteur vitesse $v(M^t)$ de m à l'instant t est donné par :

$$v(M^t) = v(O_i^t) + \omega_i^t \wedge \overrightarrow{O_i^tM^t} \tag{1.3.1.i}$$

où ω_i^t est le *vecteur vitesse de rotation instantanée* associé à la rotation ρ_i^t, porté par l'axe de la rotation et de longueur égale à la vitesse angulaire.
Le vecteur accélération $\gamma(M^t)$ de m à l'instant t est donné par :

$$\gamma(M^t) = \gamma(O_i^t) + \frac{d\omega_i^t}{dt} \wedge \overrightarrow{O_i^tM^t} + \omega_i^t \wedge \left(\omega_i^t \wedge \overrightarrow{O_i^tM^t}\right) \tag{1.3.1.ii}$$

La nature de la formule (1.3.1.i) qui donne le vecteur vitesse des particules d'un système exprime que le champ de ces vecteurs est un champ equiprojectif, ce qui conduit à poser la définition suivante :

Définition 1.46 *On appelle torseur distributeur des vitesses ou torseur cinématique d'un corps $\mathcal{S}_i$ pour un mouvement ϕ_i à l'instant t, l'application affine de $\mathbb{E}$ vers $\overrightarrow{\mathbb{E}}$ qui, à tout point M de l'espace, fait correspondre le vecteur vitesse $v(M)$ pour le mouvement ϕ_i de la particule $\psi(t, M)$ – dont la position à l'instant t serait M – de $\mathcal{E}$ considérée comme solidaire de $\mathcal{S}_i$.*

Propriété 1.47 *On peut identifier, par isomorphie, l'espace tangent en un point à la variété primitive de configurations d'un corps d'un système mécanique à l'espace vectoriel des champs équiprojectifs de l'espace. La vitesse ϕ_i^t de ce corps à l'instant t a pour image par cet isomorphisme le torseur distributeur des vitesses de ce corps.*

On a vu que la variété primitive de configurations $\mathcal{V}_i{}'$ d'un corps d'un système est le groupe des déplacements de l'espace $\mathbb{R}^3 \times \mathrm{SO}_3$.
L'espace tangent à $\mathbb{R}^3 \times \mathrm{SO}_3$ en l'identité, $T_{\mathrm{Id}}\mathcal{V}_i{}'$ est la somme directe $\mathbb{R}^3 \oplus H_3$ où H_3 est l'espace vectoriel des endomorphismes antisymétriques de $\mathbb{R}^3$ lui-même isomorphe à l'espace vectoriel des champs équiprojectifs de l'espace $\mathrm{Eq}(\mathbb{E})$ par l'application :

$$\begin{array}{ccc} \mathrm{Eq}(\mathbb{E}) & \longrightarrow & \mathbb{R}^3 \oplus H_3 \\ \left[\begin{array}{ccc} \mathbb{E} & \longrightarrow & \mathbb{E} \\ M & \longmapsto & X + A \wedge \overrightarrow{OM} \end{array}\right] & \longmapsto & (X, \tilde{A}) \end{array}$$

$$\text{avec} \qquad \tilde{A} = \begin{pmatrix} 0 & -c & b \\ c & 0 & -a \\ -b & a & 0 \end{pmatrix} \qquad \text{si} \qquad A : \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$

De plus, le produit des déplacements étant différentiable, en tout point ϕ de $\mathbb{R}^3 \times \mathrm{SO}_3$, l'espace tangent $T_\phi \mathcal{V}_i{}'$ est isomorphe à l'espace tangent en l'identité $T_{\mathrm{Id}} \mathcal{V}_i{}'$.
En effet, si, pour $\phi \in (\mathbb{R}^3 \times \mathrm{SO}_3)$ fixé, on note $\circ_\phi$ le produit à gauche d'un déplacement par ϕ :

$$\begin{array}{rccc} \circ_\phi & : & \mathbb{R}^3 \times \mathrm{SO}_3 & \longrightarrow & \mathbb{R}^3 \times \mathrm{SO}_3 \\ & & d & \longmapsto & \phi \circ d \end{array}$$

alors $[T_{\mathrm{Id}} \circ_\phi : T_{\mathrm{Id}} \mathcal{V}' \longrightarrow T_\phi \mathcal{V}']$ est bien un isomorphisme.
Concrètement à $(X, \tilde{A})$ de $T_{\mathrm{Id}} \mathcal{V}'$, $T_{\mathrm{Id}} \circ_\phi$ associe $(\rho(X), \bar{\rho}\tilde{A})$ dans $T_\phi \mathcal{V}'$. ◇

La propriété 1.47 permet effectivement de calculer les caractéristiques du torseur cinématique d'un corps.
Considérons un mouvement dont la trajectoire reste à l'intérieur du domaine de carte des angles d'Euler $\phi : t \longmapsto \phi^t{}_{(x(t),y(t),z(t),\Psi(t),\Theta(t),\Phi(t))}$ où :

$$\phi^t{}_{(x(t),y(t),z(t),\Psi(t),\Theta(t),\Phi(t))} = [M \longmapsto M^t]$$
$$\text{avec} \qquad \overrightarrow{OM^t} = \tau_{(x(t),y(t),z(t))} + \rho_{(\Psi(t),\Theta(t),\Phi(t))} \left(\overrightarrow{OM} \right)]$$

Dérivons, par rapport à t la matrice 4×4 de ce déplacement, on obtient une matrice assez encombrante mais qui peut s'écrire simplement* :

$$\left(\begin{array}{ccc|c} 0 & -\Psi' - \cos(\Theta)\Phi' & \sin(\Psi)\Theta' - \sin(\Theta)\cos(\Psi)\Phi' & x' \\ \Psi' + \cos(\Theta)\Phi' & 0 & -\cos(\Psi)\Theta' - \sin(\Theta)\sin(\Psi)\Phi' & y' \\ -\sin(\Psi)\Theta' + \sin(\Theta)\cos(\Psi)\Phi' & \sin(\Theta)\sin(\Psi)\Phi' + \cos(\Psi)\Theta' & 0 & z' \\ \hline \multicolumn{3}{c|}{\mathbf{0}} & 1 \end{array} \right) \left(\begin{array}{c|c} \bar{\rho}^t & \mathbf{0} \\ \hline \mathbf{0} & 1 \end{array} \right)$$

Ainsi, la vitesse du corps sera donnée par le couple de $\mathbb{R}^3 \oplus \mathrm{H}_3$ de composantes :

$$\left(\begin{pmatrix} x'(t) \\ y'(t) \\ z'(t) \end{pmatrix}, \begin{pmatrix} \cos(\Psi(t))\Theta'(t) + \widetilde{\sin(\Theta(t))}\sin(\Psi(t))\Phi'(t) \\ \sin(\Psi(t))\Theta'(t) - \sin(\Theta(t))\cos(\Psi(t))\Phi'(t) \\ \Psi'(t) + \cos(\Theta(t))\Phi'(t) \end{pmatrix} \right)$$

et le torseur cinématique pour ce corps sera donné par $M^t \longmapsto v(O^t) + \omega^t \wedge \overrightarrow{O^t M^t}$ soit analytiquement, si a, b et c sont les coordonnées de M^0 dans le repère $\mathcal{R}_0$:

$$\begin{pmatrix} x(t) \\ y(t) \\ z(t) \end{pmatrix} + \bar{\rho}^t \begin{pmatrix} a \\ b \\ c \end{pmatrix} \longmapsto \begin{pmatrix} x'(t) \\ y'(t) \\ z'(t) \end{pmatrix} + \begin{pmatrix} \cos(\Psi(t))\Theta'(t) + \sin(\Theta(t))\sin(\Psi(t))\Phi'(t) \\ \sin(\Psi(t))\Theta'(t) - \sin(\Theta(t))\cos(\Psi(t))\Phi'(t) \\ \Psi'(t) + \cos(\Theta(t))\Phi'(t) \end{pmatrix} \wedge \bar{\rho}^t \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$

Remarque 1.48 Dans la mesure où la variété primitive de configurations d'un système de plusieurs corps est le produit cartésien des variétés primitives de configurations de chacun de ces corps, on peut facilement étendre la propriété 1.47 en énonçant que, pour chaque corps du système, la composante de la vitesse du système correspondant à ce corps fournit le torseur cinématique de ce corps.

*Tous les paramètres dépendant du temps, on n'écrira pas explicitement (t) afin d'alléger les expressions ; et, pour toute fonction f dépendant du temps t, f' représentera sa dérivée par rapport à t

torseurs cinétique et dynamique, centre et opérateur d'inertie, énergie cinétique

Pour étudier la dynamique d'un système mécanique, on est amené à introduire la notion de *masse*.
La mécanique classique – non relativiste – postule l'existence d'une densité de masse $d\mu$ en toute particule des corps étudiés et l'invariance de cette masse au cours du temps.
La *masse du système* est obtenu à partir de cette densité par :

$$\mu(\mathcal{S}) = \int_{m\in\mathcal{S}} d\mu$$

Définition 1.49 *Le centre d'inertie d'un corps rigide $\mathcal{S}$ est défini comme la particule g de ce corps telle que :*

$$\int_{m\in\mathcal{S}} \overrightarrow{gm} d\mu = 0 \tag{1.3.1.iii}$$

Remarque 1.50 Si l'on considère un corps non rigide, ou si l'on considère un système composé de plusieurs corps, le centre d'inertie dépend de la configuration du système. Ce n'est donc plus une notion liée à une particule mais à une position et il varie donc au cours du temps. C'est le point G^t défini par :

$$\int_{M^t\in S^t} \overrightarrow{G^tM^t} d\mu = 0 \tag{1.3.1.iv}$$

où $d\mu$ représente cette fois une densité de masse induite sur les points à partir des particules dont ils sont les positions.

La formule1.3.1.iv entraîne, par dérivation immédiate :

$$\mu(\mathcal{S})v(G^t) = \int_{M^t\in S^t} v(M^t)d\mu \qquad \text{et} \qquad \mu(\mathcal{S})\gamma(G^t) = \int_{M^t\in S^t} \gamma(M^t)d\mu$$

et conduit à définir les torseurs *cinétique* et *dynamique*.

Définition 1.51 *On appelle torseur cinétique (resp. dynamique) d'un corps $\mathcal{S}$ à un instant t le torseur qui a pour moment en tout point M^t :*

$$\int_{P^t\in S^t} \overrightarrow{M^tP^t} \wedge v(P^t)d\mu \qquad \left(\text{resp.} \int_{P^t\in S^t} \overrightarrow{M^tP^t} \wedge \gamma(P^t)d\mu\right)$$

ce torseur a alors pour somme géométrique $\mu(\mathcal{S})v(G^t)$ (resp. $\mu(\mathcal{S})\gamma(G^t)$).

Ainsi, le torseur cinétique a pour éléments de réduction la quantité de mouvement du système et, en chaque point, son moment cinétique par rapport à ce point.
De même, le torseur dynamique a pour éléments de réduction la quantité d'accélération du système et, en chaque point, son moment dynamique par rapport à ce point.
On a les relations suivantes entre résultantes et moments des torseurs cinématiques et dynamiques :

$$\frac{d}{dt}\left(\mu(\mathcal{S})v(G^t)\right) = \mu(\mathcal{S})\gamma(G^t) \tag{1.3.1.v}$$

$$\int_{P^t\in S^t} \overrightarrow{M^tP^t} \wedge \gamma(P^t)d\mu = \frac{d}{dt}\left(\int_{P^t\in S^t} \overrightarrow{M^tP^t} \wedge v(P^t)d\mu\right) + \mu(\mathcal{S})v(M^t)\wedge v(G^t) \tag{1.3.1.vi}$$

Lors des calculs, on essaiera de se placer dans l'un des cas où la formule (1.3.1.vi) se simplifie, notamment :

- si $v(M^t) = 0$, ce qui arrive pour $M^t = O_i^t$ lorsque l'on écrit les torseurs cinétiques et dynamiques dans un repère lié au corps en mouvement ; alors (1.3.1.vi) s'écrit :

$$\int_{P^t \in S^t} \overrightarrow{O_i^t P^t} \wedge \gamma(P^t) d\mu = \frac{d}{dt} \left(\int_{P^t \in S^t} \overrightarrow{O_i^t P^t} \wedge v(P^t) d\mu \right) \tag{1.3.1.vii}$$

- pour $M^t = G^t$ – d'où $v(M^t) = v(G^t)$ – alors (1.3.1.vi) s'écrit :

$$\int_{P^t \in S^t} \overrightarrow{G^t P^t} \wedge \gamma(P^t) d\mu = \frac{d}{dt} \left(\int_{P^t \in S^t} \overrightarrow{G^t P^t} \wedge v(P^t) d\mu \right) \tag{1.3.1.viii}$$

Remarque 1.52 La définition 1.51 s'étend naturellement aux systèmes de plusieurs corps rigides et en particulier aux mécanismes. Il est néanmoins prudent, dans ce cas, de conserver à l'esprit que la notion de centre d'inertie n'est alors pas rattachée à une particule.

Définition 1.53 *On appelle énergie cinétique à l'instant t d'un système $\mathcal{S}$ de corps rigides en mouvement le scalaire $K(\mathcal{S}, t)$ défini par :*

$$K(\mathcal{S}, t) = \frac{1}{2} \left(\int_{M^t \in S^t} \left(v(M^t) \right)^2 d\mu \right)$$

Pour calculer explicitement le moment cinétique d'un corps par rapport à un point M^t, on est parfois amené à l'exprimer sous la forme :

$$\begin{aligned} \int_{P^t \in S^t} \overrightarrow{M^t P^t} \wedge v(P^t) d\mu &= \int_{P^t \in S^t} \overrightarrow{M^t P^t} \wedge \left(v(O_i^t) + \omega_i^t \wedge \overrightarrow{O_i^t P^t} \right) d\mu \\ &= \mu(\mathcal{S}) \overrightarrow{M^t G^t} \wedge v(O_i^t) + \overrightarrow{M^t O_i^t} \wedge \left(\mu(\mathcal{S}) \omega_i^t \wedge \overrightarrow{O_i^t G^t} \right) \\ &\quad + \int_{P^t \in S^t} \overrightarrow{O_i^t P^t} \wedge \left(\omega_i^t \wedge \overrightarrow{O_i^t P^t} \right) d\mu \end{aligned} \tag{1.3.1.ix}$$

C'est pourquoi, on définit en tout point de $\mathbb{E}$ un opérateur linéaire dépendant du corps considéré comme suit :

Définition 1.54 *On appelle opérateur d'inertie du corps $\mathcal{S}$ en un point M^t l'application linéaire $I(\mathcal{S}, M^t)$ qui à tout vecteur $\vec{u}$ fait correspondre $I(\mathcal{S}, M^t)\vec{u}$ défini par :*

$$I(\mathcal{S}, M^t)\vec{u} = \int_{P^t \in S^t} \overrightarrow{M^t P^t} \wedge \left(\vec{u} \wedge \overrightarrow{M^t P^t} \right) d\mu$$

Remarque 1.55 Si on exprime cet opérateur à l'aide de sa matrice dans une base liée au corps considéré, cette matrice ne dépend pas du temps, mais est caractéristique de la particule m de ce corps, on la note $\bar{I}(\mathcal{S}, m)$ et elle a pour expression (indépendamment de t) :

$$\begin{pmatrix} \int_{P^t \in S^t} \left((y_{P^t})^2 + (z_{P^t})^2 \right) d\mu & -\int_{P^t \in S^t} (x_{P^t})(y_{P^t}) d\mu & -\int_{P^t \in S^t} (x_{P^t})(z_{P^t}) d\mu \\ -\int_{P^t \in S^t} (x_{P^t})(y_{P^t}) d\mu & \int_{P^t \in S^t} \left((x_{P^t})^2 + (z_{P^t})^2 \right) d\mu & -\int_{P^t \in S^t} (y_{P^t})(z_{P^t}) d\mu \\ -\int_{P^t \in S^t} (x_{P^t})(z_{P^t}) d\mu & -\int_{P^t \in S^t} (y_{P^t})(z_{P^t}) d\mu & \int_{P^t \in S^t} \left((x_{P^t})^2 + (y_{P^t})^2 \right) d\mu \end{pmatrix}$$

Le moment cinétique au centre d'inertie du corps considéré s'exprime de façon particulièrement simple, à partir de la formule (1.3.1.ix), si l'origine du repère $\mathcal{R}_i$ est choisie au centre d'inertie :

$$\int_{P^t \in S^t} \overrightarrow{G^t P^t} \wedge v(P^t) d\mu = I(\mathcal{S}, G^t) \omega_i^t \tag{1.3.1.x}$$

torseurs des efforts extérieurs

Un des principes de la dynamique est que la somme des efforts exercés par un système mécanique sur un autre système mécanique est un torseur qui à tout point du système sur lequel s'exerce ces efforts associe un vecteur de l'espace représentant une force.

On modélise classiquement ces efforts au travers de lois physiques faisant intervenir des densités de forces – par exemple le champ gravitationnel – des densités de couples – par exemple un champ electro-magnétique – et des torseurs d'efforts concentrés – pour des actions ponctuelles.

On appelle *torseur des efforts extérieurs*[†] du système mécanique $\mathcal{S}$ le torseur représentant les efforts exercés sur ce système dont il est nécessaire de tenir compte dans le schéma d'étude physique choisi.

1.3.2 les méthodes de la mécanique classique

On va présenter ci-dessous les deux principales méthodes en mécanique classique, qui proviennent

- l'une de l'application de la loi fondamentale de la dynamique,
- l'autre de l'introduction en tout point du mécanisme de forces et de couples d'inertie.

La première méthode est historiquement attribuée à Newton [Newton, 1687] même si le théorème du moment cinétique, que l'on considèrera comme partie intégrante de la loi fondamentale, ne peut s'en déduire effectivement qu'en assumant l'hypothèse que les efforts de réactions entre deux corps ont même ligne d'action, et est considéré, dans le cas général comme un postulat d'Euler.

La deuxième méthode dérive des travaux de d'Alembert [D'Alembert, 1743] bien qu'elle ne fasse pas intervenir explicitement le principe des travaux virtuels que l'on désigne habituellement comme principe de d'Alembert.

la Loi fondamentale de la Dynamique – Newton-Euler

Il s'agit d'appliquer au système considéré la Loi fondamentale de la Dynamique qui exprime l'égalité des éléments de réduction du torseur dynamique et du torseur des forces extérieures à tout instant au cours du mouvement.

C'est à dire, en ce qui concerne la résultante, le théorème du centre d'inertie (ou seconde loi de Newton) :

> La somme des forces extérieures est égale au produit de sa masse totale par l'accélération de son centre d'inertie.

$$F_{ext} = \mu(\mathcal{S})\gamma(G^t) = \int_{M^t \in S^t} \gamma(M^t) d\mu \qquad (1.3.2.\text{i})$$

et, en ce qui concerne les moments, le théorème du moment dynamique :

$$C_{ext}(\mathcal{S}, M^t) = \int_{P^t \in S^t} \overrightarrow{M^t P^t} \wedge \gamma(P^t) d\mu$$

qui devient, si on l'écrit en un point fixe ou en G^t, le théorème du moment cinétique (ou postulat d'Euler) :

[†]On notera $F_{ext}(\mathcal{S})$ sa resultante et $C_{ext}(\mathcal{S}, M)$ son moment au point M, voire F et C_M si aucune ambiguïté n'est possible

Au centre d'inertie du système, la dérivée du moment cinétique est égale au moment des forces extérieures.

$$C_{ext}(\mathcal{S}, G^t) = \frac{d}{dt}\left(\int_{P^t \in S^t} \overrightarrow{G^t P^t} \wedge v(P^t) d\mu\right) \qquad (1.3.2.ii)$$

On obtient ainsi six équations différentielles du second ordre qui sont indépendantes et permettent donc de déterminer – au moins numériquement – six paramètres qui décrivent le système en fonction de t.

Or, dès que le système est constitué d'au moins deux corps, on a vu que six paramètres ne suffisent pas pour décrire les configurations des différents corps.

Pour obtenir des équations supplémentaires, on est contraint d'appliquer de nouveau la loi fondamentale à des sous-systèmes considérés comme libres.

Dans ce cas, les quantités inconnues qui interviennent dans les équations sont, non seulement les configurations des différents corps mais aussi les réactions entre certains de ces corps qui, bien qu'internes au système, sont extérieures au sous-système considéré.

Sur ces réactions, le principe de l'action et de la réaction (ou troisième loi de Newton) fournit les renseignements suivants :

Si on note F_{ij} l'action du corps $\mathcal{S}_j$ sur le corps $\mathcal{S}_i$, on sait seulement que $F_{ij} + F_{ji} = 0$.

On est donc amené à appliquer la loi fondamentale à des sous-systèmes de façon surabondante pour pouvoir éliminer, à l'intérieur de l'ensemble des équations, les variables correspondant aux réactions inconnues.

les équations de d'Alembert

Par antagonie avec l'approche de Newton, d'Alembert considère la notion de force comme une "causalité motrice" plutôt que comme une grandeur concrète. Il privilégie donc une représentation *fonctionnelle* des forces, ce qui aboutit à une forme *différentielle* au lieu d'intégrale des équations du mouvement.

Plus clairement, si Newton a établi que l'accélération du centre d'inertie d'un système est proportionnelle à la somme des forces extérieures qui s'appliquent sur ce système, d'Alembert démontre lui que l'accélération de chaque particule du système est proportionnelle aux forces, externes et internes, qui agissent sur cette particule.

Ainsi, d'Alembert établit, après avoir introduit en chaque particule m du système une force d'inertie $(-d\mu\, \gamma(M^t))$, que les sommes des forces et des moments des forces sont nulles en n'importe quelle particule du système :

$$\forall M^t \in S^t, \quad dF_{ext} - \gamma(M^t) d\mu = 0 \qquad (1.3.2.iii)$$

On ne va néanmois pas utiliser ce principe sous cette forme différentielle, mais en l'intégrant sur un des corps du mécanisme considéré et non sur tout le mécanisme.

Ainsi, sachant que l'on peut représenter l'ensemble des forces d'inertie qui s'appliquent à un corps rigide par un couple et une force appliquée au centre d'inertie de ce corps, on va être amené, une fois encore, à écrire et à dériver des expressions de vitesse comme précédemment, mais en un des centres d'inertie des corps du système que l'on essayera de choisir de manière à simplifier les calculs.

On écrira donc

- que la somme des forces – forces extérieures et force d'inertie – s'exerçant sur le système est nulle :

$$\sum_{\mathcal{S}} (F_{ext} + F_{inert}) = 0$$

- que la somme des moments des forces – forces extérieures et force d'inertie – en n'importe quel point M^t du système est nulle :

$$\sum_{\mathcal{S}} \big(C_{ext}(\mathcal{S}, M^t) + C_{inert}(\mathcal{S}, M^t)\big) = 0$$

Ce sont ces équations que l'on désignera par *principe de d'Alembert* contrairement à l'habitude de la littérature qui désigne sous ce nom le *principe des travaux virtuels* – dans le cas des mécanismes, ce dernier se réduit à ces équations (voir [Kane et Levinson, 1980]).

Par rapport à la méthode exposée plus haut, on n'est donc pas obligé de localiser le centre d'inertie global du mécanisme et la différence se compose essentiellement du choix supplémentaire à faire lors de l'établissement des équations – en quel corps appliquer le principe ? – qui conduit à une réduction de la complexité des calculs à exécuter si ce corp est bien choisi.

On obtient, bien entendu, de nouveau seulement six équations différentielles du mouvement, et on est donc là aussi amené à réappliquer cette méthode à des sous-systèmes pour obtenir des équations différentielles supplémentaires, et à le faire de façon surabondante puisque ces équations font elles aussi intervenir des efforts de réactions entre les corps qu'il faudra éliminer.

On pourra, bien entendu, choisir là encore en quels points des sous-systèmes on écrira que la somme des moments des forces s'annulle.

1.3.3 les méthodes de la mécanique analytique

Les travaux de Lagrange [Lagrange, 1788], à la fin du 18ème siècle, puis les travaux de Hamilton [Hamilton, 1835], au début du 19ème se sont appuyés sur le principe de d'Alembert et sur les recherches de Maupertuis pour jeter les bases de la mécanique analytique. Ils se sont développés autour de deux notions fondamentales :

- celle de *déplacement virtuel* infiniment petit associée au principe de moindre action – le mouvement réel d'un système apparaît comme un mouvement privilégié, celui qui dépense le moins d'énergie, dans l'ensemble de tous ses mouvements possibles –
- celle d'espace de configurations, qui ramène l'étude d'un système à l'étude d'un nombre restreint de paramètres indépendants et permet de séparer l'étude dynamique de l'étude cinématique du mouvement d'un système.

C'est depuis Lagrange que l'on s'accorde à considérer la mécanique comme une science théorique et abstraite, branche à part entière de l'analyse mathématique.

De nombreuses méthodes de mise en équation ont, depuis, étés développées, qui diffèrent entre elles essentiellement par le formalisme choisi – la nature de l'espace de configurations et la manière dont on le représente – et par les grandeurs à calculer pour obtenir les équations différentielles du mouvement – énergie cinétique, Hamiltonien, fonction de Gibbs, ...

On va en présenter quelques-unes ci-dessous.

les équations de Lagrange

On utilise précisément la notion de *configuration* d'un système introduite au paragraphe (§1.1) :

> Un point d'un espace de dimension $6n$ – pour un système de n corps – isomorphe, modulo le choix d'une configuration de référence, à $(\mathrm{Is}^+(I\!E))^n$ où $(\mathrm{Is}^+(I\!E))$ est l'ensemble des déplacements de l'espace affine de dimension 3.

En prenant en compte les liaisons holonomes du mécanisme étudié – que l'on suppose à structure arborescente – on définit comme une variété différentielle l'ensemble de ses configurations admissibles (cf. §1.2). On fait le choix d'un jeu de *coordonnées généralisées*, c'est à dire d'un certain nombre de paramètres indépendants q_i (autant que le *degré de liberté*) représentant une carte de la variété de configurations.

Les équations de la dynamique s'établissent en écrivant que, lors de n'importe quel mouvement virtuel compatible avec les liaisons du système, la puissance des quantités d'accélération est égale à la puissance des efforts connus s'exerçant sur le mécanisme – principe des puissances virtuelles.

Le théorème dit de Lagrange relie les quantités de mouvement et d'accélération à l'énergie cinétique K du système, les équations différentielles du mouvement s'écrivent ainsi :

$$\frac{d}{dt}\left(\frac{\partial K}{\partial \dot{q}_i}\right) - \frac{\partial K}{\partial q_i} = Q_i \tag{1.3.3.i}$$

où les Q_i sont les composantes généralisées des forces connues.

Il s'agit donc de calculer, à partir des paramètres q_i et $\dot{q}_i \left(= \frac{dq_i}{dt}\right)$, l'expression de l'énergie cinétique du système :

$$K = 1/2 \left[\int_S [\vec{v}(M)]^2 dm\right] \tag{1.3.3.ii}$$

L'avantage considérable de cette méthode est de ne pas nécessiter l'introduction puis l'élimination de paramètres représentant les réactions internes inconnues mais ceci n'est vrai que si l'on connait, a priori, le degré de liberté du mécanisme et un jeu de paramètres indépendants pour le décrire.

Par contre, l'établissement des équations est en général beaucoup plus coûteux que dans les méthodes précédentes et nécessite plusieurs dérivations.

les équations de Lagrange avec multiplicateurs

Comme on vient de le voir, l'établissement des équations de Lagrange nécessite le choix préalable d'un jeu de coordonnées généralisées. Ce choix n'est pas toujours faisable, notamment dans les deux cas suivants :

- si les liaisons intervenant dans les mécanismes étudiés ne sont pas holonomes (i.e. contraignent non seulement les positions mais aussi les vitesses), – voir (§1.1.4)
- si l'on ne connaît pas, a priori, le degré de liberté des mécanismes étudiés – notamment dans le cas de structures bouclées complexes.

Pour étudier ces cas, on est amené alors à utiliser des jeux de coordonnées généralisées contenant des paramètres surabondants qui satisfont des relations complémentaires de la forme :

$$f_j(q, \dot{q}) = \sum_i A_{ji}(q)\dot{q}_i = 0 \quad \text{pour } j = 1 \ldots p \tag{1.3.3.iii}$$

ce qui revient, d'une part, à ne tenir compte que de certaines liaisons pour définir la variété d'étude et introduire les q_i, d'autre part, à exprimer effectivement les liaisons, dont on n'a pas tenu compte, par les équations ci-dessus.

On doit ensuite introduire , en nombre égal, des inconnues complémentaires $\lambda_1, \lambda_2, \ldots, \lambda_p$ - appelées *multiplicateurs de Lagrange*. Les équations s'établissent alors comme au paragraphe précédent :

$$\frac{d}{dt}\left(\frac{\partial T}{\partial \dot{q}_i}\right) - \frac{\partial T}{\partial q_i} = Q_i + \sum_j \lambda_j A_{ji} \tag{1.3.3.iv}$$

le terme $\sum_j \lambda_j A_{ji}$ représentant la puissance des efforts (inconnus) des liaisons dont on n'a pas tenu compte dans la définition du paramétrage.

On obtient ainsi un système d'équations algébro-différentielles. L'intérêt de cette méthode, en dehors du fait qu'elle n'oblige pas à effectuer de choix préalable de coordonnées, est manifeste si l'on possède de bons outils d'intégration de tels système. Sinon, il faut éliminer les multiplicateurs, ce qui, formellement revient à expliciter les relations entre les colonnes de la matrice A, processus qui s'avère très couteûx dans le cas général.

Un autre intérêt de cette méthode est de fournir des renseignements sur les réactions internes au mécanisme lorsque l'on calcule effectivement les multiplicateurs.

les équations de Hamilton

La volonté de Hamilton de vouloir unifier les différentes sciences en dégageant en mécanique un *principe variationnel* comme il existait déjà en optique, l'a conduit à travailler non pas sur l'espace de configurations des mécanismes, mais sur leur espace de phases (configuration et vitesse). C'est pourquoi, dans le formalisme hamiltonien, on introduit comme paramètres supplémentaires les moments conjugués $p_i = \partial K / \partial \dot{q}_i$.

L'établissement des équations de Hamilton est facilement réalisable lorsque les efforts extérieurs s'appliquant au système dérivent d'un potentiel. Dans ce cas là, on définit le Lagrangien L du système comme la différence entre son énergie cinétique et le potentiel en question et les équations de Lagrange s'écrivent alors :

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = 0 \tag{1.3.3.v}$$

Le Hamiltonien du système H est alors défini par :

$$H = \sum_i p_i \dot{q}_i - L \tag{1.3.3.vi}$$

la fonction qui à L fait correspondre H est appelée transformation de Legendre.

Les équations du mouvement s'écrivent, à partir de (1.3.3.v) et en différenciant (1.3.3.vi) :

$$\frac{\partial H}{\partial p_i} = \frac{dq_i}{dt} \qquad \text{et} \qquad \frac{\partial H}{\partial q_i} = -\frac{dp_i}{dt} \tag{1.3.3.vii}$$

Pour former les équations du mouvement, on est ainsi amener à exprimer successivement :

- l'énergie cinétique du système en fonction des q_i
- les p_i en fonction des q_i et des $\dot{q}_i$
- les $\dot{q}_i$ en fonction des p_i et des q_i (ce qui revient à inverser la transformation de Legendre)

Si l'avantage de cette méthode est de fournir des équations différentielles du premier ordre, il est clair que son principal obstacle réside dans l'inversion formelle de la transformation de Legendre qui s'avère très coûteuse voire impossible.

les équations de Boltzmann-Hamel

On peut présenter le formalisme de Boltzmann-Hamel comme un raffinement (ou une extension) du formalisme Lagrangien. C'est d'ailleurs ainsi que ces auteurs ont présentés leurs travaux [Boltzmann, 1902] [Hamel, 1904] au début du siècle.

L'idée de Boltzmann et de Hamel consiste à travailler avec les coordonnées généralisées ainsi qu'avec des paramètres supplémentaires – parfois appelés *quasi-coordonnées* de Lagrange – tels que les mesures des vitesses ou des vitesses angulaires.

Ainsi, on introduit les paramètres supplémentaires u_j reliés aux q_i par :

$$u_j = \sum_i \alpha_{ij} \dot{q}_i \qquad \text{et} \qquad \dot{q}_i = \sum_j \beta_{ij} u_j \tag{1.3.3.viii}$$

Il s'agit alors d'écrire T en fonction des u_j et des q_i ce qui conduit à définir les variables intermédiaires ϵ_i et δ_i comme :

$$\epsilon_i = \frac{\partial T}{\partial u_i} \qquad \text{et} \qquad \delta_i = \sum_j \beta_{ji} \left(\frac{\partial T}{\partial q_j} \right) \tag{1.3.3.ix}$$

il ne reste plus qu'à former les équations, par substitution dans les équations de Lagrange, ce qui donne :

$$\dot{\epsilon}_i + \sum_j \sum_k \gamma_{ijk} u_k \epsilon_j - \delta_i = \sum_j \beta_{ji} Q_j \tag{1.3.3.x}$$

avec :

$$\gamma_{ijk} = \sum_r \sum_s \beta_{ri} \beta_{sk} \left(\frac{\partial \alpha_{rj}}{\partial q_s} - \frac{\partial \alpha_{sj}}{\partial q_r} \right)$$

Sauf dans quelques cas très particuliers, les équations obtenues ne sont pas de forme plus agréable que celles de Lagrange. Il est, en dehors de ces cas, tout à fait inintéressant d'utiliser cette méthode dans la mesure où elle nécessite, pour n coordonnées de départ, l'introduction et le calcul de $n^3 + 2n^2 + n$ variables supplémentaires.

les équations de Gibbs

Le formalisme de Gibbs [Gibbs, 1879] ressemble fortement au formalisme précédent puisqu'il utilise les mêmes paramètres et conduit à calculer les mêmes variables intermédiaires β_{ij}. Cependant l'établissement des équations est subordonné au calcul de la fonction (dite *de Gibbs*) faisant intervenir les accélérations inertielles des points de vitesse u_i :

$$G = \frac{1}{2} \sum_i m_i a_i^2 \tag{1.3.3.xi}$$

et il n'est donc plus nécessaire d'établir l'expression de l'énergie cinétique ce qui évite l'introduction des δ_i, ϵ_i et γ_{ijk}. Les équations s'écrivent :

$$\frac{\partial G}{\partial \dot{u}_i} = \sum_j \beta_{ji} Q_j \tag{1.3.3.xii}$$

Ce formalisme est, parmi les formalismes dérivés de l'approche Lagrangienne celui qui permet d'obtenir les équations du mouvement avec le moins d'effort et qui débouche sur la forme la plus simple de ces équations. Son principal inconvénient, qui est commun avec les précédents formalismes, à l'exception de celui de Lagrange avec multiplicateurs, est qu'il nécessite de connaître à l'avance le nombre de degrés de liberté du mécanisme étudié.

1.3.4 les méthodes “contemporaines”

Avec l'avènement des outils informatiques de calcul formel, des équipes de mécaniciens se sont récemment intéressées à la génération automatique des équations et ont utilisés des formalismes qui leurs sont propres à partir des formalismes existants.

On peut notamment citer les équipes des :

- Pr Wittenburg à l'Université de Karlsruhe qui ont développé un logiciel basé sur les méthodes de Newton utilisant aussi le principe de D'Alembert pour traiter les cas bouclés – voir [Wittenburg, 1977],
- Pr Kane à l'Université de Stanford qui ont introduit un formalisme original implanté dans divers logiciels – voir [Kane, 1978],
- Pr Chevallier à l'Ecole Nationale des Ponts et Chaussées qui, dans la continuité des travaux d'Arnold [Arnold, 1976], ont étudié l'utilisation de groupes de Lie pour la modélisation des mécanismes [Chevallier et Helmer, 1984] – voir aussi [Chevallier, 1984].

On va présenter ci-dessous les deux derniers formalismes.

formalisme de Kane

L'originalité du formalisme de Kane, qui s'applique à tout mécanisme que l'on peut caractériser par un jeu de coordonnées généralisées - et donc dont on *connaît* le nombre de degrés de liberté - tient dans l'introduction de nouvelles grandeurs appelées *vitesses partielles*.

Comme dans le formalisme de Boltzmann-Hamel, on utilise les u_j, mesures des vitesses et des vitesses angulaires. On se place dans le cas où l'expression des u_j en fonction des q_i et des $\dot{q}_i$ permet d'exprimer la vitesse angulaire de chaque corps et la vitesse de n'importe quel point du mécanisme étudié, comme une combinaison linéaire des u_j à coefficients vectoriels (3×1). Ces coefficients sont les vitesses partielles qui servent de base à l'établissement des équations. On note ainsi $v_i^{P_j}$ le vecteur coefficient de u_i dans l'expression de la vitesse de la particule P_j :

$$v^{P_j} = \sum_i u_i v_i^{P_j}$$

Les équations s'obtiennent alors en écrivant que la somme des forces généralisées et des forces d'inertie généralisées est nulle soit :

$$K_i + K_i^* = 0 \tag{1.3.4.i}$$

avec :

$$K_i = \sum_j (v_i^{P_j}.F_j) \qquad \text{et} \qquad K_i^* = \sum_j (v_i^{P_j}.(-m_j a_j)) \tag{1.3.4.ii}$$

Ce formalisme est très performant, aussi bien au niveau du travail nécessaire pour établir les équations qu'au niveau de la forme de ces équations. Mais ces performances dépendent beaucoup du choix préliminaire des u_j, et, une fois de plus, il est nécessaire de connaître à l'avance le nombre de degrés de liberté du système mécanique étudié. Les logiciels qui utilisent ce formalisme se sont donc restreint au traitement des cas de mécanismes à structure arborescente et dont les liaisons font parties d'un “catalogue de liaisons-type”.

théorie des groupes de Lie

Le formalisme développé à partir de cette théorie est radicalement différent des précédents, il est basé sur les trois principes suivants :

- utiliser la géométrie différentielle intrinsèque pour décrire les positions des solides indépendamment de tout choix de coordonnées,
- étudier, dans ce cadre, les propriétés géométriques des liaisons et la cinématique des mécanismes,
- et exprimer, toujours dans ce cadre, la cinétique et la théorie de l'inertie du solide rigide.

L'idée fondamentale est donc de décrire la position s d'un solide par un point d'un ensemble S homogène sur lequel agit le groupe des déplacements de l'espace et donc muni d'une structure de groupe de Lie. On caractérise alors la vitesse d'un solide par un élément $v(t)$ de l'algèbre de Lie du groupe des déplacements $\mathfrak{d}$ avec :

$$\dot{s}(t) = L^T_{A(t)}(v(t).r) \tag{1.3.4.iii}$$

où $L^T_{A(t)} : TS \to TS$ est l'application linéaire tangeante à $L_A : S \to S, s \mapsto A.s$ $(A \in \mathrm{Is}^+(I\!E))$.
L'ensemble des déplacements relatifs possibles d'un solide, par rapport à un autre solide auquel il est relié, est représenté (*sous certaines hypothèses*) comme un sous groupe de $\mathrm{Is}^+(I\!E)$.
Les concepts de masse, de tenseur d'inertie et de centre d'inertie laissent la place à un opérateur hexadimentionnel $H : \mathfrak{d} \to \mathfrak{d}$.
Les équations de la dynamique s'écrivent alors, pour un système constitué de n corps :

$$\dot{s}_k = L^T_{A_k}(v_k.r) \text{ pour } k = 1, \ldots, n \tag{1.3.4.iv}$$

$$H_k(\dot{v}_k) + [v_k, H_k(v_k)] = F_k(s, \dot{s}, t) + \sum_{i \neq k}(R_{ik}(s, \dot{s}) + N_{ik}(t)) \tag{1.3.4.v}$$

$$\text{relations d'orthogonalité diverses sur les } N_{ik}(t) \tag{1.3.4.vi}$$

les R_{ik} représentent les efforts internes connus (moteurs, raideurs, frottements,...) tandis que les N_k représentent les efforts de liaisons inconnus, les relations (1.3.4.vi) dépendent de la nature des liaisons.
Il ne reste plus qu'à choisir un jeu de coordonnées et à écrire les équations précédentes en fonction de ces coordonnées.
L'intérêt de ce formalisme est qu'il permet de réaliser l'essentiel des calculs avant de choisir le jeu de coordonnées - *mais ce choix est incontournable*. Dans l'optique d'un traitement automatique, ce formalisme est particulièrement agréable et performant à condition d'avoir implanté dans un système de calcul formel la manipulation d'objets caractéristiques des groupes de Lie et leurs propriétés, notamment le crochet de Lie et les opérations B et C de $\mathfrak{d} \times \mathfrak{d}$ dans $\mathfrak{d}$ définies par :

$$B(X, Y) = [X, H(Y)] \quad \text{et} \quad C(X, Y) = [X, H(Y)] + [Y, H(X)] + H([X, Y])$$

très utiles dans l'écriture des équations.

pour conclure

De ces préliminaires mécaniques, et pour la suite de cette étude, il faudra retenir essentiellement les points suivants :

i. Parmi les systèmes d'objets mécaniques dont on peut être amené à modéliser le comportement dynamique, on considérera – et on désignera par mécanismes – seulement ceux qui sont constitués par des corps rigides reliés entre eux par des articulations possédant des propriétés particulières – liaisons holonomes régulières.

ii. Le mouvement d'un système de corps rigides articulés, et donc plus particulièrement dún mécanisme, se modélise par la donnée, à chaque instant, de sa configuration, que l'on peut considérer comme constituée d'autant de déplacements de l'espace que le mécanisme contient de corps.

iii. L'étude du mouvement d'un mécanisme se décompose en deux étapes, à savoir :

- la détermination, à partir de sa nature physique – structure et type des articulations – de l'ensemble des configurations admissibles, et la paramétrisation de cet ensemble,
- l'écriture d'équations différentielles, à partir des caractéristiques dynamiques du mécanisme, gouvernant ces paramètres.

iv. La détermination de l'ensemble des configurations admissibles d'un mécanisme est aisée lorsque la structure de ce mécanisme est arborescente : dans ce cas, cet ensemble est une variété différentielle réelle dont on connait la dimension.

v. On dispose, pour l'écriture des équations différentielles du mouvement, de différentes méthodes associées à différents formalismes basés sur les théorèmes généraux de la mécanique classique ou de la mécanique analytique.

Chapitre 2

approche formelle : un exemple

On va maintenant appliquer certaines des méthodes de génération des équations de la dynamique vues au chapitre précédent à un exemple simple mais non trivial : la modélisation du mouvement d'un satellite transportant une antenne.

Plus précisément, pour ce mécanisme, on va établir les équations de Newton-Euler, de d'Alembert, de Lagrange – sans multiplicateurs –, de Hamilton et de Kane.

On pourra ainsi non seulement expliciter ces méthodes, mais surtout les étudier au niveau de la possibilité de les automatiser, de la quantité de travail que demande l'élaboration des équations et de la taille des expressions obtenues.

Tous les calculs de ce chapitre ont été réalisés avec le système de calcul formel *MAPLE* que l'on présente brièvement ci-dessous.

On trouvera le code et les résultats de calcul dans l'annexe (§A).

> *MAPLE* est un logiciel écrit en *C* à l'université de Waterloo qui permet de manipuler des objets symboliques.
>
> Pour l'utilisateur, il se présente comme un programme interactif, qui lit des commandes, les exécute et rend leurs résultats.
>
> Les commandes de *MAPLE* s'écrivent dans une syntaxe naturelle à partir :
>
> - d'un certain nombre d'opérateurs qui permettent de définir des expressions mathématiques – addition, multiplication, sommation, différentiation, intégration, ...
> - d'un certain nombre de déclarations qui permettent de représenter des objets mathématiques – vecteurs, matrices, applications, ensembles, ...
> - et d'un grand nombre de fonctions qui forment la bibliothèque du système et qui permettent de manipuler ces expressions et ces objets – développement, factorisation, simplification, substitution, calcul de dérivées, d'intégrales, résolution d'équations, ...– ces fonctions représentant toute la "connaissance" mathématique du système.
>
> L'utilisateur de *MAPLE* dispose, de plus, d'un langage de programmation spécifique ressemblant au langage *PASCAL* qui permet d'écrire des procédures, en utilisant diverses structures de données – tableaux, listes, ensembles, ...– et en faisant appel à des structures de contrôles classiques – instructions conditionnelles, boucles, ...
>
> Dans la mesure où la riche bibliothèque d'algorithmes mathématiques de *MAPLE* est elle-même écrite dans ce langage de programmation, il est facile pour l'utilisateur d'avoir accès à ces algorithmes et de les modifier si le besoin s'en fait sentir.
>
> Enfin *MAPLE* possède des interfaces particulièrement agréable, du point de vue :

- de la mise en forme des résultats à l'écran – à l'aide d'un *pretty-printer* paramétrable,
- du graphique, notamment pour le tracé de courbes et de surfaces,
- de la génération de codes d'autres langages
  - soit pour l'écriture de formules – génération de TEX et de LATEX,
  - soit pour l'écriture de code de calcul – génération de *FORTRAN* et de *C*.

On essaiera de dégager, dans les exemples qui suivent, quelques techniques de base de l'utilisation et de la programmation de *MAPLE*.

Dans un premier temps, il est nécessaire de décrire le système mécanique auquel l'on va s'intéresser.

2.1 description du système étudié

Le système $\mathcal{S}$ que l'on considère – voir figure (2.1) – est constitué de quatre corps $\mathcal{S}_1$, $\mathcal{S}_2$, $\mathcal{S}_3$ et $\mathcal{S}_4$ à savoir :

- $\mathcal{S}_1$ un satellite de forme cylindrique,
- $\mathcal{S}_2$ la partie primaire d'un bras articulé, en forme de tige, reliée au satellite par une articulation rotoïde, dont l'axe est contenu dans le plan d'une face du satellite,
- $\mathcal{S}_3$ la partie secondaire du même bras, qui coulisse sur la partie primaire en formant une articulation prismatique d'axe porté par le bras,
- $\mathcal{S}_4$ un réflecteur parabolique que l'on assimilera en fait à un disque relié à la partie secondaire du bras, en déport de son centre d'inertie, par une articulation rotoïde, d'axe confondu avec l'axe du bras.

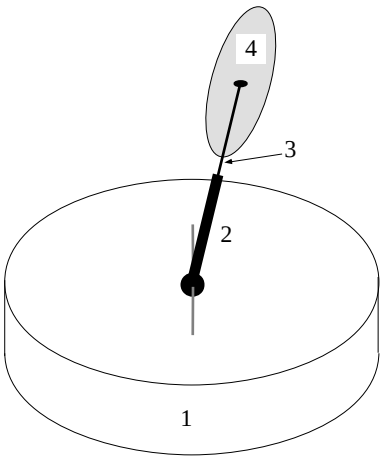


figure 2.1 : le satellite

Les deux parties du bras sont supposées de masses négligeables. En revanche, on note μ_1 et μ_4 les masses respectives du satellite et du réflecteur, g_1 et g_4 leurs centres d'inertie et $\mathcal{I}_1$ l'opérateur d'inertie de $\mathcal{S}_1$ – toute la masse du reflecteur sera supposée concentrée en son centre.

On se propose d'étudier le mouvement de ce système relativement à un repère Galiléen R d'origine O et de vecteurs unitaires i, j et k.

On associe au satellite un repère $\mathcal{R}_1$ d'origine o_1 $(= g_1)$ et dont les vecteurs unitaires i_1, j_1 et k_1 sont portés par ses axes principaux d'inertie – i.e. dans ce repère la matrice de l'opérateur d'inertie, I_1, est diagonale :

$$I_1 = \begin{pmatrix} I_{1x} & 0 & 0 \\ 0 & I_{1y} & 0 \\ 0 & 0 & I_{1z} \end{pmatrix}$$

La particule du satellite où le bras est articulé est notée m_{12}, et a pour coordonnées $(0, 0, z_{12})$ dans le repère $\mathcal{R}_1$. L'axe de cette articulation a pour direction la droite vectorielle engendrée par i_1.

On appelle m_{21} la particule du corps $\mathcal{S}_2$ au point d'articulation avec le corps $\mathcal{S}_1$. Le repère $\mathcal{R}_2$ associé à cette partie du bras a pour origine o_2 $(= m_{21})$ et est constitué des vecteurs unitaires i_2 $(= i_1)$, j_2 orienté selon l'axe du corps et k_2 tel que (i_2, j_2, k_2) forment un trièdre direct. A chaque instant t, l'angle $\widehat{k_1^t, k_2^t}$ est repéré par $\Theta_{12}(t)$.

$\mathcal{R}_3$, repère solidaire du corps $\mathcal{S}_3$ a les mêmes vecteurs unitaires que $\mathcal{R}_2$ et a son origine o_3 en m_{34}, point d'articulation entre la partie mobile du bras et le réflecteur. A tout instant t, le point M_{34}^t, position de la particule m_{34}, a pour coordonnées $(0, Y_{23}(t), 0)$ dans R_2^t.

La particule du réflecteur, point d'articulation avec le bras est m_{43}. Dans le repère $\mathcal{R}_4$, g_4, le centre d'inertie du réflecteur a pour coordonnées $(0, 0, z_{34})$.

Le point m_{43} est aussi l'origine o_4 du repère $\mathcal{R}_4$ associé au corps $\mathcal{S}_4$. Ce repère a pour vecteurs unitaires i_4, j_4, et k_4 tels que $j_4 = j_3$ et que i_4^t et k_4^t soient respectivement images de i_3^t et de k_3^t par la rotation autour de j_4^t d'angle $\Theta_{34}(t)$ à tout instant t.

Tous ces repères sont représentés sur la figure 2.2.

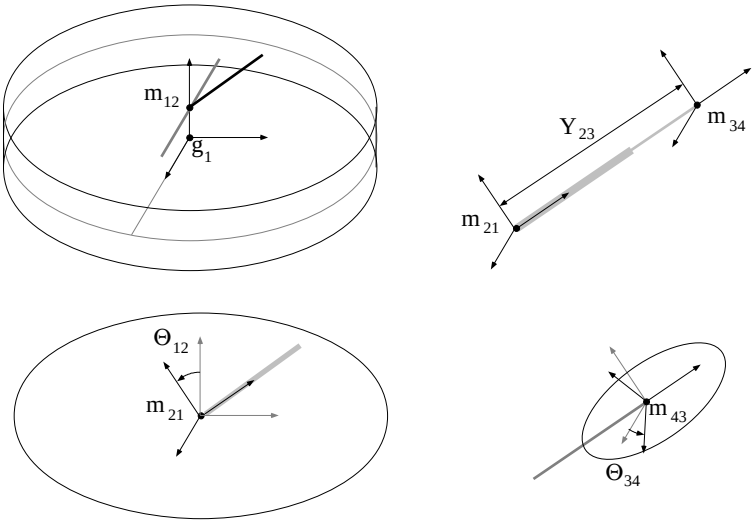


figure 2.2 : $\mathcal{R}_1$, $\mathcal{R}_2$, $\mathcal{R}_3$ et $\mathcal{R}_4$

La présence de ressorts et d'amortisseurs aux articulations entre $\mathcal{S}_1$ et $\mathcal{S}_2$, entre $\mathcal{S}_2$ et $\mathcal{S}_3$ et entre $\mathcal{S}_3$ et $\mathcal{S}_4$ est modélisée par des torseurs d'efforts s'exerçant sur $\mathcal{S}_2$, sur $\mathcal{S}_3$ et sur $\mathcal{S}_4$ dont les éléments de réduction sont connus sous la forme d'un couple de moment $C_{12}(\Theta_{12}, \dot{\Theta}_{12}).i_2$ s'appliquant sur $\mathcal{S}_2$, d'une force de direction j_3 et de module $F_{23}(Y_{23}, \dot{Y}_{23})$ s'appliquant en m_{34} et d'un couple de moment $C_{34}(\Theta_{34}, \dot{\Theta}_{34}).i_4$ s'appliquant sur $\mathcal{S}_4$.

Les efforts extérieurs s'exercant sur le système sont de deux types :

- les efforts s'exerçant sur $\mathcal{S}_1$ représentés par une force $F_{1x}.i_1 + F_{1y}.j_1 + F_{1z}.k_1$ s'appliquant en g_1 et un couple $C_{1x}.i_1 + C_{1y}.j_1 + C_{1z}.k_1$,
- les efforts s'exerçant sur $\mathcal{S}_4$ représentés par une force $F_{4x}.i_1 + F_{4y}.j_1 + F_{4z}.k_1$ s'appliquant en g_4.

Pour établir les équations du mouvement de $\mathcal{S}$ on est d'abord amené à exprimer différentes grandeurs cinétiques et donc à choisir un moyen de paramétrer les configurations de ce système.

Une rapide analyse mécanique montre que ce système possède un degré de liberté égal à neuf – le satellite est libre dans l'espace, et chacune des trois liaisons avec les trois autres corps possède un degré de contrainte égal à cinq – et conduit à choisir neuf paramètres q_1, q_2, q_3, q_4, q_5, q_6, q_7, q_8, q_9 pour le décrire.

On choisira, par exemple, ces paramètres tels que :

- q_1, q_2, q_3 soient les trois angles de Bryant décrivant l'orientation de R_1^t – image de $\mathcal{R}_1$ à l'instant t – par rapport à R,
- q_4, q_5, q_6 soient les trois coordonnées du point O_1^t – position de la particule o_1 à l'instant t – dans le repère R,
- q_7 soit l'angle Θ_{12} donnant l'orientation – on en déduira la position – du corps $\mathcal{S}_2$,
- q_8 soit la longueur Y_{23} donnant la position – on déduira de Θ_{12} l'orientation – du corps $\mathcal{S}_3$,
- q_9 soit l'angle Θ_{34} donnant l'orientation – on en déduira la position – du corps $\mathcal{S}_4$.

On peut alors, en fonction de ces paramètres, exprimer les positions et les orientations des repères liés à chacun des corps du système.

Ainsi, on note $\bar{\rho}_1^t$ (resp. $\bar{\rho}_{12}^t$, $\bar{\rho}_{23}^t$, $\bar{\rho}_{34}^t$) la matrice de la rotation associée au déplacement qui transforme R en R_1^t (resp. R_1^t en R_2^t, R_2^t en R_3^t, R_3^t en R_4^t) – qui est aussi la matrice de passage de la base associée à R_1^t dans la base associée à R (et respectivement) –, on a* :

$$\bar{\rho}_1^t = \begin{pmatrix} \cos(q_2)\cos(q_3) & -\cos(q_2)\sin(q_3) & \sin(q_2) \\ \sin(q_1)\sin(q_2)\cos(q_3)+\cos(q_1)\sin(q_3) & -\sin(q_1)\sin(q_2)\sin(q_3)+\cos(q_1)\cos(q_3) & -\sin(q_1)\cos(q_2) \\ -\cos(q_1)\sin(q_2)\cos(q_3)+\sin(q_1)\sin(q_3) & \cos(q_1)\sin(q_2)\sin(q_3)+\sin(q_1)\cos(q_3) & \cos(q_1)\cos(q_2) \end{pmatrix},$$

$$\bar{\rho}_{12}^t = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(q_7) & -\sin(q_7) \\ 0 & \sin(q_7) & \cos(q_7) \end{pmatrix}, \quad \bar{\rho}_{23}^t = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \text{ et } \bar{\rho}_{34}^t = \begin{pmatrix} \cos(q_9) & 0 & \sin(q_9) \\ 0 & 1 & 0 \\ -\sin(q_9) & 0 & \cos(q_9) \end{pmatrix}$$

Alors, d'après ce qui précède, les vecteurs $\overrightarrow{OO_1^t}$, $\overrightarrow{O_1^tO_2^t}$, $\overrightarrow{O_2^tO_3^t}$, $\overrightarrow{O_4^tG_4^t}$ ont pour coordonnées dans le repère R :

$$\overrightarrow{OO_1^t} = \begin{pmatrix} q_4 \\ q_5 \\ q_6 \end{pmatrix} \quad \overrightarrow{O_1^tO_2^t} = \bar{\rho}_1^t \begin{pmatrix} 0 \\ 0 \\ z_{12} \end{pmatrix} \quad \overrightarrow{O_2^tO_3^t} = \bar{\rho}_1^t\bar{\rho}_{12}^t \begin{pmatrix} 0 \\ q_8 \\ 0 \end{pmatrix} \quad \overrightarrow{O_4^tG_4^t} = \bar{\rho}_1^t\bar{\rho}_{12}^t\bar{\rho}_{34}^t \begin{pmatrix} 0 \\ 0 \\ z_{34} \end{pmatrix}$$

Pour simplifier l'écriture des grandeurs cinématiques, on sera parfois amené à utiliser des paramètres cinématiques, ainsi :

*l'indice ^{t} indique que ces matrices dépendent explicitement du temps et permet d'écrire seulement q_i à la place de $q_i(t)$ pour $i = 1, 2, \ldots, 9$

- la vitesse du corps $\mathcal{S}_1$ par rapport à R sera représentée par les coordonnées (v_x, v_y, v_z) du vecteur $v(G_1^t) = \left(\dfrac{d}{dt}\bigg|_R \overrightarrow{OG_1^t} \right)$ dans le repère R_1^t,
- la vitesse angulaire de $\mathcal{S}_1$ par rapport à R sera représentée par le vecteur vitesse instantanée de rotation ω_1 de coordonnées $(\omega_x, \omega_y, \omega_z)$ dans le repère R_1^t.

En tant que fonctions du temps t les paramètres dits "de position" q_1, q_2, q_3, q_4, q_5, q_6 et les paramètres dits "cinématiques" de $\mathcal{S}_1$ sont reliés par les systèmes d'équations différentielles suivant :

- pour ce qui est du vecteur vitesse de rotation instantanée,

$$\left\{ \begin{array}{rcl} \dot{q}_1 &=& \dfrac{\omega_x \cos(q_3) - \sin(q_3)\omega_y}{\cos(q_2)} \\ \dot{q}_2 &=& \sin(q_3)\omega_x + \omega_y \cos(q_3) \\ \dot{q}_3 &=& \omega_z - \dfrac{(\omega_x \cos(q_3) - \sin(q_3)\omega_y)\sin(q_2)}{\cos(q_2)} \end{array} \right. \tag{I.1}$$

- et, en ce qui concerne la vitesse linéaire,

$$\left\{ \begin{array}{rcl} \dot{q}_4 &=& \cos(q_2)\cos(q_3)v_x - \cos(q_2)\sin(q_3)v_y + \sin(q_2)v_z \\ \dot{q}_5 &=& (\sin(q_1)\sin(q_2)\cos(q_3) + \cos(q_1)\sin(q_3))\, v_x \\ && + (\cos(q_1)\cos(q_3) - \sin(q_1)\sin(q_2)\sin(q_3))\, v_y \\ && - \sin(q_1)\cos(q_2)v_z \\ \dot{q}_6 &=& (\sin(q_1)\sin(q_3) - \cos(q_1)\sin(q_2)\cos(q_3))\, v_x \\ && + (\cos(q_1)\sin(q_2)\sin(q_3) + \sin(q_1)\cos(q_3))\, v_y \\ && + \cos(q_1)\cos(q_2)v_z \end{array} \right. \tag{I.2}$$

Dans la mesure où le paramétrage des rotations de l'espace $\mathbb{E}$ par les angles de Bryant ne permet pas de représenter de manière unique les rotations autour de l'axe Ox et donc que l'on suppose – au risque de devoir utiliser localement un autre paramétrage – que ce cas n'est pas rencontré au cours du mouvement, on est assuré que $\cos(q_2)$ est non nul et que les relations (I.1) sont bien définies.

Les systèmes d'équations précédents ont été obtenus en "inversant" les équations qui, réciproquement, donnent les expressions de (v_x, v_y, v_z) et de $(\omega_x, \omega_y, \omega_z)$ en fonction des paramètres de position $(q_1, q_2, q_3, q_4, q_5, q_6)$ et de leurs dérivées $(\dot{q}_1, \dot{q}_2, \dot{q}_3, \dot{q}_4, \dot{q}_5, \dot{q}_6)$, c'est-à-dire respectivement :

$$\left\{ \begin{array}{rcl} \omega_x &=& \cos(q_2)\cos(q_3)\dot{q}_1 + \sin(q_3)\dot{q}_2 \\ \omega_y &=& -\cos(q_2)\sin(q_3)\dot{q}_1 + \cos(q_3)\dot{q}_2 \\ \omega_z &=& \sin(q_2)\dot{q}_1 + \dot{q}_3 \end{array} \right. \tag{I.3}$$

et

$$\left\{ \begin{array}{rcl} v_x &=& \cos(q_2)\cos(q_3)\dot{q}_4 + (\sin(q_1)\sin(q_2)\cos(q_3) + \cos(q_1)\sin(q_3))\, \dot{q}_5 \\ && - (\cos(q_1)\sin(q_2)\cos(q_3) - \sin(q_1)\sin(q_3))\, \dot{q}_6 \\ v_y &=& -\cos(q_2)\sin(q_3)\dot{q}_4 - (\sin(q_1)\sin(q_2)\sin(q_3) - \cos(q_1)\cos(q_3))\, \dot{q}_5 \\ && + (\cos(q_1)\sin(q_2)\sin(q_3) + \sin(q_1)\cos(q_3))\, \dot{q}_6 \\ v_z &=& \sin(q_2)\dot{q}_4 - \sin(q_1)\cos(q_2)\dot{q}_5 + \cos(q_1)\cos(q_2)\dot{q}_6 \end{array} \right. \tag{I.4}$$

Une fois établies les équations différentielles du mouvement du système mécanique faisant intervenir les paramètres cinématiques , on pourra :

- soit remplacer dans ces équations les paramètres cinématiques par leurs expressions en fonction des paramètres de position en utilisant les équations (I.3) et (I.4), on obtiendra alors un système de neuf équations différentielles du second ordre en les neuf paramètres de position,
- soit ajouter les six équations différentielles (I.1) et (I.2) aux neuf équations du mouvement et considérer ainsi le système de quinze équations différentielles en les paramètres de position et les paramètres cinématiques qui seront du premier ordre seulement en q_1, q_2, q_3, q_4, q_5, et q_6.

On pourrait très bien introduire de la même manière des paramètres cinématiques pour les autres corps du système, et les substituer ou les conserver après génération des équations du mouvement.

Il faut cependant garder à l'esprit que le gain en taille et en lisibilité des équations est ainsi acquis au détriment du nombre des variables et du nombre des expressions à calculer et n'est pas forcément intéressant surtout dans l'optique d'une simulation numérique effectuée à partir des équations du mouvement.

Le code *MAPLE* correspondant à cette description du mécanisme est le fichier `intro`.

L'examen de ce code amène quelques remarques qui se révèleront utiles.

Tout d'abord, en ce qui concerne les fonctionnalités logicielles utilisées, on notera le chargement initial de la bibliothèque d'algèbre linéaire pour avoir accès à toutes les fonctions de manipulation et de calcul sur les expressions matricielles et vectorielles. On utilise notamment, de cette bibliothèque :

- l'addition et la multiplication – non commutative – de matrices,
- l'addition de vecteurs,
- la multiplication d'un vecteur par une matrice,
- la multiplication d'une matrice par un scalaire

Il est important de remarquer aussi que l'on peut introduire des formules de deux manières différentes,

i. par une affectation,
ii. par une équation.

Dans le premier cas, il s'agit d'établir une correspondance irréversible entre un identificateur et une expression mathématique, ainsi la ligne de code

```
Rot1:=multiply(Rotq1,Rotq2,Rotq3)
```

affecte à `Rot1` la matrice résultat de l'évaluation du produit, soit composante par composante la matrice $\bar{\rho}^t$ telle qu'on l'a écrite plus haut.
Dans le deuxième cas, il s'agit de stocker une formule provenant d'un calcul qui permettra, le cas échéant, d'effectuer des substitutions d'expressions. Ainsi, en conservant le système d'équations (I.3) sous la forme

```
{omX(t) = cos(q2) cos(q3) qp1 + sin(q3) qp2,

    omY(t) = - cos(q2) sin(q3) qp1 + cos(q3) qp2,

       omZ(t) = sin(q2) qp1 + qp3}
```

on pourra, à la demande, substituer dans n'importe qu'elle expression `omZ(t)` par `sin(q2) * qp1 + qp3` mais rien n'est affecté à l'identificateur `omZ(t)` si ce n'est son propre nom.

On remarque encore les écritures de la forme `q1(t)` qui permettent sans affecter `q1` d'indiquer qu'il s'agit d'un paramètre dépendant de la variable `t`, et provoquent, notamment, la réaction suivante de l'opérateur de différentiation `diff` :

```
> diff(q1(t),t);
                                      d
                                     ---- q1(t)
                                      dt
> diff(q1,t);
                                        0
```

Enfin, on note l'utilisation de deux fonctions de manipulations d'expressions, `simplify` et `collect` qui réalisent respectivement la simplification et la factorisation partielle. Ainsi, l'expression

```
 qp3 = - (- sin(q2(t)) sin(q3(t)) omY(t) + sin(q2(t)) omX(t) cos(q3(t))

                                      2                                    2
      - omZ(t) cos(q2(t)) cos(q3(t))  - omZ(t) cos(q2(t)) sin(q3(t)) )

        /                           2             2
       /  (cos(q2(t)) (cos(q3(t))  + sin(q3(t)) ))
      /
```

obtenue au cours du calcul de `tmp16` devient, par un appel à `simplify` avec l'option `trig` :

```
 qp3 = - (- sin(q2(t)) sin(q3(t)) omY(t) + sin(q2(t)) omX(t) cos(q3(t))

           - omZ(t) cos(q2(t)))/cos(q2(t)),
```

et finalement après appel de la fonction `collect` avec le paramètre `sin(q2(t))` :

```
          (- sin(q3(t)) omY(t) + omX(t) cos(q3(t))) sin(q2(t))
 qp3 = - ---------------------------------------------------- + omZ(t)
                              cos(q2(t))
```

2.2 la loi fondamentale de la dynamique – Newton-Euler

Dans ce qui suit, lorsqu'on considère les vecteurs vitesse et accélération d'un point M, c'est par rapport au repère R et on les note respectivement $v(M)$ et $\gamma(M)$ sans indiquer explicitement R. On écrit pour commencer le théorème du centre d'inertie (1.3.2.i).

théorème du centre d'inertie

Pour tout point M de l'espace, et à tout instant t, on a :

$$\mu(\mathcal{S})\overrightarrow{MG^t} = \mu_1\overrightarrow{MG_1^t} + \mu_4\overrightarrow{MG_4^t} \tag{N.1}$$

En considérant un point M fixe dans le repère R, et en dérivant (N.1) par rapport au temps dans ce repère, on obtient l'expression de la quantité de mouvement totale du système, c'est à dire :

$$\mu(\mathcal{S})v(G^t) = \mu_1 v(G_1^t) + \mu_4 v(G_4^t) \tag{N.2}$$

Une nouvelle dérivation par rapport au temps, toujours dans le repère R, donne la quantité d'accélération totale du système :

$$\mu(\mathcal{S})\gamma(G^t) = \left(\frac{d}{dt}\bigg|_R \mu(\mathcal{S})v(G^t) \right) = \mu_1\gamma(G_1^t) + \mu_4\gamma(G_4^t) \tag{N.3}$$

Il vient le théorème du centre d'inertie :

$$\mu_1\gamma(G_1^t) + \mu_4\gamma(G_4^t) = F_1 + F_4 \tag{N.4}$$

Pour déduire de cette équation trois équations différentielles du mouvement, il faut maintenant exprimer les accélérations de G_1^t et de G_4^t en fonction des paramètres du mécanisme.
Pour établir ces expressions, on peut :

- soit utiliser traditionnellement les formules de composition de vitesses et d'accélérations et les formules de dérivation vectorielle,
- soit exprimer directement $\overrightarrow{OG_1^t}$ et $\overrightarrow{OG_4^t}$ dans le repère R et faire calculer automatiquement les dérivées secondes.

On va comparer ces deux possibilités.

> Le programme *MAPLE* correspondant à la première de ces possibilités se trouve dans le fichier `newton1alamain`.

Les formules de composition de vitesses et d'accélérations donnent :

$$\gamma(G_1^t) = \left(\frac{d}{dt}\bigg|_{R_1^t} v(G_1^t) \right) + \omega_1 \wedge v(G_1^t) \tag{N.5}$$

$$\begin{aligned} \gamma(G_4^t) \quad = \quad & \gamma(G_1^t) + \left(\frac{d}{dt}\bigg|_{R_1^t} \omega_1 \right) \wedge \overrightarrow{G_1^tG_4^t} + \omega_1 \wedge \left(\omega_1 \wedge \overrightarrow{G_1^tG_4^t} \right) \\ & + 2\omega_1 \wedge \left(\frac{d}{dt}\bigg|_{R_1^t} \overrightarrow{G_1^tG_4^t} \right) + \left(\frac{d^2}{(dt)^2}\bigg|_{R_1^t} \overrightarrow{G_1^tG_4^t} \right) \end{aligned} \tag{N.6}$$

ce qui permet d'obtenir, relativement simplement, les expressions voulues en fonction des paramètres cinématiques (v_x, v_y, v_z) et $(\omega_x, \omega_y, \omega_z)$.
Exprimées dans R_1^t, on a en effet :

$$v(G_1^t) = \begin{pmatrix} v_x \\ v_y \\ v_z \end{pmatrix},\ \omega_1 = \begin{pmatrix} \omega_x \\ \omega_y \\ \omega_z \end{pmatrix},\ \frac{d}{dt}\bigg|_{R_1^t} v(G_1^t) = \begin{pmatrix} \dot{v}_x \\ \dot{v}_y \\ \dot{v}_z \end{pmatrix} \text{ et } \frac{d}{dt}\bigg|_{R_1^t} \omega_1 = \begin{pmatrix} \dot{\omega}_x \\ \dot{\omega}_y \\ \dot{\omega}_z \end{pmatrix}$$

et donc directement :

$$\gamma(G_1^t) = \begin{pmatrix} \dot{v}_x + \omega_y v_z - \omega_z v_y \\ \dot{v}_y + \omega_z v_x - \omega_x v_z \\ \dot{v}_z + \omega_x v_y - \omega_y v_x \end{pmatrix}$$

Pour exprimer $\gamma(G_4^t)$ dans R_1^t, on se sert de :

$$\overrightarrow{G_1^tG_4^t} = \overrightarrow{G_1^tO_2^t} + \overrightarrow{O_2^tO_4^t} + \overrightarrow{O_4^tG_4^t} = \begin{pmatrix} 0 \\ 0 \\ z_{12} \end{pmatrix} + \bar{\rho}_{12}^t \begin{pmatrix} 0 \\ q_8 \\ 0 \end{pmatrix} + \bar{\rho}_{12}^t \bar{\rho}_{34}^t \begin{pmatrix} 0 \\ 0 \\ z_{34} \end{pmatrix}$$

Il est alors nécessaire d'utiliser de nouveau les formules classiques de dérivation vectorielle pour établir au préalable :

- l'expression de $\left.\dfrac{d}{dt}\right|_{R_1^t} \overrightarrow{G_1^tG_4^t}$

$$\begin{pmatrix} \dot{q}_7 \\ 0 \\ 0 \end{pmatrix} \wedge \bar{\rho}_{12}^t \begin{pmatrix} 0 \\ q_8 \\ 0 \end{pmatrix} + \bar{\rho}_{12}^t \begin{pmatrix} 0 \\ \dot{q}_8 \\ 0 \end{pmatrix} + \left[\begin{pmatrix} \dot{q}_7 \\ 0 \\ 0 \end{pmatrix} + \bar{\rho}_{12}^t \begin{pmatrix} 0 \\ \dot{q}_9 \\ 0 \end{pmatrix} \right] \wedge \bar{\rho}_{12}^t \bar{\rho}_{34}^t \begin{pmatrix} 0 \\ 0 \\ z_{34} \end{pmatrix}$$

- et l'expression de $\left.\dfrac{d^2}{(dt)^2}\right|_{R_1^t} \overrightarrow{G_1^tG_4^t}$

$$\begin{pmatrix} \ddot{q}_7 \\ 0 \\ 0 \end{pmatrix} \wedge \bar{\rho}_{12}^t \begin{pmatrix} 0 \\ q_8 \\ 0 \end{pmatrix} + \begin{pmatrix} \dot{q}_7 \\ 0 \\ 0 \end{pmatrix} \wedge \left[\begin{pmatrix} \dot{q}_7 \\ 0 \\ 0 \end{pmatrix} \wedge \bar{\rho}_{12}^t \begin{pmatrix} 0 \\ q_8 \\ 0 \end{pmatrix} \right]$$

$$+2 \begin{pmatrix} \dot{q}_7 \\ 0 \\ 0 \end{pmatrix} \wedge \bar{\rho}_{12}^t \begin{pmatrix} 0 \\ \dot{q}_8 \\ 0 \end{pmatrix} + \bar{\rho}_{12}^t \begin{pmatrix} 0 \\ \ddot{q}_8 \\ 0 \end{pmatrix}$$

$$+ \left[\begin{pmatrix} \ddot{q}_7 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} \dot{q}_7 \\ 0 \\ 0 \end{pmatrix} \wedge \bar{\rho}_{12}^t \begin{pmatrix} 0 \\ \dot{q}_9 \\ 0 \end{pmatrix} + \bar{\rho}_{12}^t \begin{pmatrix} 0 \\ \ddot{q}_9 \\ 0 \end{pmatrix} \right] \wedge \bar{\rho}_{12}^t \bar{\rho}_{34}^t \begin{pmatrix} 0 \\ 0 \\ z_{34} \end{pmatrix}$$

$$+ \left[\begin{pmatrix} \dot{q}_7 \\ 0 \\ 0 \end{pmatrix} + \bar{\rho}_{12}^t \begin{pmatrix} 0 \\ q_9 \\ 0 \end{pmatrix} \right] \wedge \left[\left[\begin{pmatrix} \dot{q}_7 \\ 0 \\ 0 \end{pmatrix} + \bar{\rho}_{12}^t \begin{pmatrix} 0 \\ \dot{q}_9 \\ 0 \end{pmatrix} \right] \wedge \bar{\rho}_{12}^t \bar{\rho}_{34}^t \begin{pmatrix} 0 \\ 0 \\ z_{34} \end{pmatrix} \right]$$

En ce qui concerne les termes représentant les coordonnées dans le repère R_1^t de la somme géométrique du torseur des efforts extérieurs s'exerçant sur $\mathcal{S}$, ils valent :

$$\begin{pmatrix} F_{1x} \\ F_{1y} \\ F_{1z} \end{pmatrix} + \begin{pmatrix} F_{4x} \\ F_{4y} \\ F_{4z} \end{pmatrix}$$

Le théorème du centre d'inertie fournit ainsi les trois premières équations du mouvement de $\mathcal{S}$:

$$
\begin{aligned}
&F_{1x}+F_{4x} \;=\\
&\quad \mu_1(\dot v_x+\omega_y v_z-\omega_z v_y)+\mu_4\\
&\qquad (\dot v_x+\omega_y v_z-\omega_z v_y+\dot\omega_y(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34})-\dot\omega_z(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})\\
&\qquad +\omega_y(\omega_x(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})-\omega_y\sin(q_9)z_{34})\\
&\qquad -\omega_z(\omega_z\sin(q_9)z_{34}-\omega_x(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34}))\\
&\qquad +2(\omega_y(\dot q_7\cos(q_7)q_8+\sin(q7)\dot q_8-(\dot q_7\sin(q_7)\cos(q_9)z_{34})-(\cos(q_7)\dot q_9\sin(q_9)z_{34}))\\
&\qquad\quad -\omega_z(-\dot q_7\sin(q_7)q_8+\cos(q_7)\dot q_8+\sin(q_7)\dot q_9\sin(q_9)z_{34}-\dot q_7\cos(q_7)\cos(q_9)z_{34}))\\
&\qquad +(-\dot q_7\sin(q_7)\dot q_9+\cos(q_7)\ddot q_9)\cos(q_7)\cos(q_9)z_{34}+(\dot q_7\cos(q_7)\dot q_9+\sin(q_7)\ddot q_9)\sin(q_7)\cos(q_9)z_{34}\\
&\qquad +\cos(q_7)\dot q_9(-(\dot q_7\sin(q_7)\cos(q_9)z_{34})-(\cos(q_7)\dot q_9\sin(q_9)z_{34}))\\
&\qquad -\sin(q_7)\dot q_9(\sin(q_7)\dot q_9\sin(q_9)z_{34}-\dot q_7\cos(q_7)\cos(q_9)z_{34}))
\end{aligned}
\tag{N.7.a}
$$

$$
\begin{aligned}
&F_{1y}+F_{4y} \;=\\
&\quad \mu_1(\dot v_y+\omega_z v_x-\omega_x v_z)+\mu_4\\
&\qquad (\dot v_y+\omega_z v_x-\omega_x v_z+\dot\omega_z\sin(q_9)z_{34}-\dot\omega_x(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34})\\
&\qquad +\omega_z(\omega_y(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34})-\omega_z(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34}))\\
&\qquad -\omega_x(\omega_x(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})-\omega_y\sin(q_9)z_{34})\\
&\qquad +2(\omega_z\dot q_9\cos(q_9)z_{34}-\omega_x(\dot q_7\cos(q_7)q_8+\sin(q_7)\dot q_8-(\dot q_7\sin(q_7)\cos(q_9)z_{34})-(\cos(q_7)\dot q_9\sin(q_9)z_{34})))\\
&\qquad -\dot q_7\sin(q_7)q_8-\dot q_7^2\cos(q_7)q_8-2\dot q_7\sin(q_7)\dot q_8+\cos(q_7)\dot q_8+(\dot q_7\cos(q_7)\dot q_9+\sin(q_7)\dot q_9)\sin(q_9)z_{34}\\
&\qquad -\dot q_7\cos(q_7)\cos(q_9)z_{34}+\sin(q_7)\dot q_9(\dot q_9\cos(q_9)z_{34})-\dot q_7(-(\dot q_7\sin(q_7)\cos(q_9)z_{34})-(\cos(q_7)\dot q_9\sin(q_9)z_{34})))
\end{aligned}
\tag{N.7.b}
$$

$$
\begin{aligned}
&F_{1z}+F_{4z} \;=\\
&\quad \mu_1(\dot v_z+\omega_x v_y-\omega_y v_x)+\mu_4\\
&\qquad (\dot v_z+\omega_x v_y-\omega_y v_x+\dot\omega_x(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})-\dot\omega_y\sin(q_9)z_{34}\\
&\qquad +\omega_x(\omega_z\sin(q_9)z_{34}-\omega_x(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34}))\\
&\qquad -\omega_y(\omega_y(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34})-\omega_z(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34}))\\
&\qquad +2(\omega_x(-\dot q_7\sin(q7)q_8+\cos(q_7)\dot q_8+\sin(q_7)\dot q_9\sin(q_9)z_{34}-(\dot q_7\cos(q_7)\cos(q_9)z_{34}))-\omega_y\dot q_9\cos(q_9)z_{34})\\
&\qquad +\dot q_7\cos(q_7)q_8-\dot q_7^2\sin(q_7)q_8+2\dot q_7\cos(q_7)\dot q_8+\sin(q_7)\dot q_8-\dot q_7\sin(q_7)\cos(q_9)z_{34}\\
&\qquad -(-\dot q_7\sin(q_7)\dot q_9+\cos(q_7)\dot q_9)\sin(q_9)z_{34}+\dot q_7(\sin(q_7)\dot q_9\sin(q_9)z_{34}-\dot q_7\cos(q_7)\cos(q_9)z_{34})\\
&\qquad -\cos(q7)\dot q_9(\dot q_9\cos(q_9)z_{34}))
\end{aligned}
\tag{N.7.c}
$$

Le programme *MAPLE* du fichier `newton1alamain` effectue précisément tous les calculs précédents.

Néanmoins, dans la mesure où les formules de dérivation vectorielle ne sont pas connues de *MAPLE*, et où il n'existe pas de structure de données permettant de les introduire facilement – on reviendra sur ce point – il a fallu développer "à la main" les formules

qui donnent les expressions de $\gamma(G_4^t)$, de $\left.\dfrac{d}{dt}\right|_{R_1^t} \overrightarrow{G_1^tG_4^t}$ et de $\left.\dfrac{d^2}{(dt)^2}\right|_{R_1^t} \overrightarrow{G_1^tG_4^t}$ avant de les programmer.

Ce travail peut devenir considérable – et donc source d'erreurs – si, au lieu de trois, comme c'était le cas, il fallait faire intervenir une dizaine de corps et de repères intermédiaires.

Avec un système de calcul formel et dans l'optique d'une génération automatique des équations, il semble préférable d'utiliser la deuxième possibilité à savoir d'exprimer directement $\mu(\mathcal{S})\overrightarrow{OG^t}$ dans le repère R en utilisant la formule (N.3) puis calculer $\mu(\mathcal{S})\gamma(G^t)$ en dérivant deux fois par rapport au temps dans ce repère, pour éviter d'avoir à faire appel au savoir du mécanicien à chaque étude d'un système mécanique différent.

Le programme correspondant à ces calculs se trouve dans le fichier `newton1`.

Pour cela, on commence par écrire les expressions de $\overrightarrow{OG_1^t}$ et de $\overrightarrow{OG_4^t}$ dans R :

$$\overrightarrow{OG_1^t} = \begin{pmatrix} q_4 \\ q_5 \\ q_6 \end{pmatrix} \qquad \text{et} \qquad \overrightarrow{G_1^tG_4^t} = \bar{\rho_1^t} \begin{pmatrix} \sin(q_9)z_{34} \\ \cos(q_7)q_8 - \sin(q_7)\cos(q_9)z_{34} \\ z_{12} + \sin(q_7)q_8 + \cos(q_7)\cos(q_9)z_{34} \end{pmatrix}$$

d'où il vient l'expression de $\mu(\mathcal{S})\overrightarrow{OG^t}$:

$$\begin{pmatrix}
\mu_1 q_4+\mu_4 \\
\quad (q_4+\cos(q_2)\cos(q_3)\sin(q9)z_{34}-\cos(q_2)\sin(q_3)\cos(q_7)q_8+\cos(q_2)\sin(q_3)\sin(q_7)\cos(q_9)z_{34} \\
\quad +\sin(q_2)z_{12}+\sin(q_2)\sin(q_7)q_8+\sin(q_2)\cos(q_7)\cos(q_9)z_{34}) \\
\\
\mu_1 q_5+\mu_4 \\
\quad (q_5+\sin(q_9)z_{34}\sin(q_1)\sin(q_2)\cos(q_3)+\sin(q_9)z_{34}\cos(q_1)\sin(q_3)-\sin(q_1)\sin(q_2)\sin(q_3)\cos(q_7)q_8 \\
\quad +\sin(q_1)\sin(q_2)\sin(q_3)\sin(q_7)\cos(q_9)z_{34}+\cos(q_1)\cos(q_3)\cos(q_7)q_8-\cos(q_1)\cos(q_3)\sin(q_7)\cos(q_9)z_{34} \\
\quad -\sin(q_1)\cos(q_2)z_{12}-\sin(q_1)\cos(q_2)\sin(q_7)q_8-\sin(q_1)\cos(q_2)\cos(q_7)\cos(q_9)z_{34}) \\
\\
\mu_1 q_6+\mu_4 \\
\quad (q_6-\sin(q_9)z_{34}\cos(q_1)\sin(q_2)\cos(q_3)+\sin(q_9)z_{34}\sin(q_1)\sin(q_3)+\cos(q_1)\sin(q_2)\sin(q_3)\cos(q_7)q_8 \\
\quad -\cos(q_1)\sin(q_2)\sin(q_3)\sin(q_7)\cos(q_9)z_{34}+\sin(q_1)\cos(q_3)\cos(q_7)q_8-\sin(q_1)\cos(q_3)\sin(q_7)\cos(q_9)z_{34} \\
\quad +\cos(q_1)\cos(q_2)z_{12}+\cos(q_1)\cos(q_2)\sin(q_7)q_8+\cos(q_1)\cos(q_2)\cos(q_7)\cos(q_9)z_{34})
\end{pmatrix}$$

On différentie alors deux fois par rapport au temps chacune des composantes du vecteur précédent et on multiplie par la transposée de $\bar{\rho}_1^t$ pour avoir l'expression de $\mu(\mathcal{S})\gamma(G^t)$ dans R_1^t.

On obtient ainsi de nouveau les trois premières équations différentielles du mouvement de $\mathcal{S}$:

$$
\begin{aligned}
F_{1x}+F_{4x} \;=\; & \\
& (\mu_1+\mu_4)v_z\omega_y-(\mu_1+\mu_4)v_y\omega_z+(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34})\mu_4\omega_z\omega_x \\
& +(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})\mu_4\omega_y\omega_x-\left(\omega_y^2+\omega_z^2\right)\mu_4\sin(q_9)z_{34} \\
& +(2\mu_4\omega_y\sin(q_7)-2\mu_4\omega_z\cos(q_7))\dot{q}_8 \\
& -(2\mu_4\omega_y\cos(q_7)\sin(q_9)z_{34}+2\mu_4\omega_z\sin(q_7)\sin(q_9)z_{34})\dot{q}_9 \\
& +((2\cos(q_7)q_8-2\sin(q_7)\cos(q_9)z_{34})\mu_4\omega_y+(2\cos(q_7)\cos(q_9)z_{34}+2\sin(q_7)q_8)\mu_4\omega_z)\dot{q}_7 \\
& -(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})\mu_4\dot{\omega}_z+(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34})\mu_4\dot{\omega}_y \\
& -\mu_4\sin(q_9)\dot{q}_9^2z_{34}+(\mu_1+\mu_4)\dot{v}_x+\mu_4\cos(q_9)\ddot{q}_9z_{34}
\end{aligned}
\tag{N.8.a}
$$

$$
\begin{aligned}
F_{1y}+F_{4y} \;=\; & \\
& (\mu_1+\mu_4)v_x\omega_z-(\mu_1+\mu_4)v_z\omega_x+(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34})\mu_4\omega_z\omega_y+\mu_4\omega_y\sin(q_9)z_{34}\omega_x \\
& -(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})\mu_4\left(\omega_x^2+\omega_z^2\right)-2\mu_4\omega_x\sin(q_7)\dot{q}_8 \\
& +(2\mu_4\omega_z\cos(q_9)z_{34}+2\mu_4\omega_x\cos(q_7)\sin(q_9)z_{34})\dot{q}_9+(2\sin(q_7)\cos(q_9)z_{34}-2\cos(q_7)q_8)\mu_4\omega_x\dot{q}_7 \\
& -(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34})\mu_4\dot{\omega}_x+\mu_4\dot{\omega}_z\sin(q_9)z_{34}+2\mu_4\cos(q_7)\dot{q}_7\sin(q_9)\dot{q}_9z_{34} \\
& -2\mu_4\sin(q_7)\dot{q}_7\dot{q}_8+\mu_4\sin(q_7)\cos(q_9)\dot{q}_9^2z_{34}+(\mu_1+\mu_4)\dot{v}_y \\
& -(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})\mu_4\dot{q}_7^2-(\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34})\mu_4\ddot{q}_7 \\
& +\mu_4\cos(q_7)\ddot{q}_8+\mu_4\sin(q_7)\sin(q_9)\ddot{q}_9z_{34}
\end{aligned}
\tag{N.8.b}
$$

$$
\begin{aligned}
F_{1z}+F_{4z} \;=\; & \\
& (\mu_1+\mu_4)v_y\omega_x-(\mu_1+\mu_4)v_x\omega_y+(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})\mu_4\omega_z\omega_y+\mu_4\omega_z\sin(q_9)z_{34}\omega_x \\
& -(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34})\mu_4\left(\omega_x^2+\omega_y^2\right)+2\mu_4\omega_x\cos(q_7)\dot{q}_8 \\
& -(2\mu_4\omega_y\cos(q_9)z_{34}-2\mu_4\omega_x\sin(q_7)\sin(q_9)z_{34})\dot{q}_9-(2\cos(q_7)\cos(q_9)z_{34}+2\sin(q_7)q_8)\mu_4\omega_x\dot{q}_7 \\
& +(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})\mu_4\dot{\omega}_x-\mu_4\dot{\omega}_y\sin(q_9)z_{34}+2\mu_4\sin(q_7)\dot{q}_7\sin(q_9)\dot{q}_9z_{34} \\
& +2\mu_4\cos(q_7)\dot{q}_7\dot{q}_8-\mu_4\cos(q_7)\cos(q_9)\dot{q}_9^2z_{34}+(\mu_1+\mu_4)\dot{v}_z \\
& -(\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34})\mu_4\dot{q}_7^2+(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})\mu_4\ddot{q}_7 \\
& +\mu_4\sin(q_7)\ddot{q}_8-\mu_4\cos(q_7)\sin(q_9)\ddot{q}_9z_{34}
\end{aligned}
\tag{N.8.c}
$$

Il est intéressant de constater que les équations trouvées ci-dessus sont plus compactes que celles obtenues par le calcul traditionnel.

Il ne s'agit pas seulement de factorisations différentes qui dans un cas donneraient une présentation mécaniquement significative et dans l'autre cas une présentation plus compacte. Certaines simplifications – notamment trigonométriques – ont, en effet, été réalisées dans les équations (N.8), ce qui diminue leur nombre de termes (par exemple le membre de droite de l'équation (N.8.a) est composé, une fois développé de 28 monômes pour 34 dans le membre de droite de l'équation équivalente (N.7.a)).

Malheureusement ces simplifications que l'on peut constater sont très coûteuses en temps de calcul. En effet, le temps CPU nécessaire pour engendrer les trois équations (N.7) – de l'ordre de 15

secondes sur une *DECstation 5000/200* – est environ quinze fois plus faible que le temps nécessaire pour engendrer les trois équations (N.8) - de l'ordre de 230 secondes.

Si l'on regarde de plus près les fichiers `newton1alamain` et `newton`, on s'aperçoit que cette différence de temps provient du nombre d'appel aux fonctions `simplify` et `clct` – qui est une version récursive de `collect` pour la factorisation partielle de sous-expressions.

Ces appels ont lieu :

- d'une part après la substitution dans l'expression de $\gamma(G^t)$ des dérivées des paramètres de position et d'orientation par leurs expressions en fonction des paramètres cinématiques,
- d'autre part après la multiplication de $\gamma(G^t)$ par la transposé de $\vec{\rho}_1^t$.

Ainsi, cette dépense de temps de calcul est directement liée à la volonté d'exprimer le principe fondamental de la dynamique dans R_1^t et en fonction des paramètres cinématiques. Ceci ne sert, a priori, qu'à conserver provisoirement – puisqu'au moment de l'intégration numérique, on substituera de nouveau les paramètres cinématiques par leurs expressions en fonction des dérivées des paramètres de position et d'orientation – une taille "humaine" aux équations différentielles obtenues.

Ici, ce mauvais choix a été fait délibérément pour insister sur le fait que lorsqu'on utilise des techniques de calcul formel, il est indispensable, avant de commencer à engendrer des équations, de bien définir l'usage que l'on veut avoir de ces équations et notamment de savoir :

i. si l'on veut obtenir des équations de tailles raisonnables, avec des paramètres mécaniquement significatifs,
ii. si par contre l'objectif est d'engendrer des équations puis un code – *FORTRAN* ou autre – pour la simulation numérique.

Dans le premier cas, il faut alors appliquer précisément les méthodes traditionnelles de mise en équation, c'est à dire procéder par substitutions successives de formules, en utilisant des variables intermédiaires ayant un sens mécanique, jusqu'à obtenir le niveau de détail voulu – considérant qu'au plus bas niveau de détail les équations s'écrivent : $\mu(\mathcal{S})\gamma(G^t) = F_{\text{ext}}$, qu'au niveau que l'on souhaite dans notre exemple, elle s'écrivent comme en (N.7), et qu'au plus haut niveau de détail il s'agit de ce que l'on obtiendrait en substituant dans (N.7) v_x, v_y, v_z, ω_x, ω_y et ω_z et leurs dérivées par leurs expressions en fonction des $\dot{q}_i$ et des $\ddot{q}_i$.

Dans le deuxième cas, il faut engendrer les équations le plus simplement possible, sans utiliser de variables intermédiaires, en n'effectuant que les simplifications indispensables et sans vouloir obtenir les résultats dans des formes inutilement factorisées.

Ainsi le code suivant calcule directement les équations différentielles en fonction des q_i, des $\dot{q}_i$ et des $\ddot{q}_i$ en un temps CPU comparable à celui du fichier `newton1alamain`.

```
read intro:
Og1:=vector([q4(t),q5(t),q6(t)]):
g1g4:=add(g1m12,add(multiply(Rot12,m21m34),multiply(Rot14,m43g4))):
Og4:=add(Og1,multiply(Rot1,g1g4)):
muOG:=add(scalarmul(Og1,mu1),scalarmul(Og4,mu4)):
tmp35:=map(diff,muOG,t):
tmp36:=map(diff,tmp35,t):
tmp38:=multiply(Rot1,add(f1,f4)):
eq1a:=subs(varslect,tmp36[1]=tmp38[1]):
```

```
eq2a:=subs(varslect,tmp36[2]=tmp38[2]):
eq3a:=subs(varslect,tmp36[3]=tmp38[3]):
```

Il serait agréable qu'un logiciel qui engendre les équations de la dynamique de mécanismes puisse donner le choix à l'utilisateur du type d'équations désirées – aisément interprétables ou bien conditionnées pour la simulation – à partir de la même description physique du système.

Pour cela, il faudrait donner au logiciel formel en question une base de connaissance mécanique, par exemple sous la forme d'une table de formules contenant pour le système mécanique étudié :

- les formule de changements de repères,
- les formules de composition des vitesses et des accélérations en fonction des différents repères.

Cette fonctionnalité n'existant pas de façon standard en *MAPLE*, il faut envisager de créer un outil, qui à partir des caractéristiques physiques d'un mécanisme, et à partir de quelques formules générales de changement de repères et de dérivation vectorielle produira les occurences de ces formules spécifiques à ce mécanisme.

théorème du moment cinétique

La deuxième partie de l'établissement des équations du mouvement de Newton-Euler consiste à écrire le théorème du moment cinétique.

Pour cela, on exprime tout d'abord le moment cinétique du système en G^t à l'aide du théorème de Koënig, ce qui donne :

$$\mathcal{I}_1\omega_1 + \mu_1\overrightarrow{G^tG_1^t} \wedge v(G_1^t) + \mathcal{I}_4\omega_4 + \mu_4\overrightarrow{G^tG_4^t} \wedge v(G_4^t)$$

ou, compte tenu de la définition du centre d'inertie G^t, et du fait que l'on suppose la masse de $\mathcal{S}_4$ concentrée en g_4 – i.e. que $\mathcal{I}_4\omega_4 = 0$:

$$\mathcal{I}_1\omega_1 + \mu\overrightarrow{G_1^tG_4^t} \wedge \left(-v(G_1^t) + v(G_4^t)\right) \qquad \text{en posant} \qquad \mu = \frac{\mu_1\mu_4}{\mu_1 + \mu_4}$$

Exprimées dans le repère R_1^t, on a :

$$\omega_1 = \begin{pmatrix} \omega_x \\ \omega_y \\ \omega_z \end{pmatrix} \qquad \text{et} \qquad \mathcal{I}_1\omega_1 = \begin{pmatrix} I_{1x}\omega_x \\ I_{1y}\omega_y \\ I_{1z}\omega_z \end{pmatrix}$$

Donc le moment cinétique de $\mathcal{S}$ en G^t s'ecrit, dans R_1^t :

$$\left(\begin{array}{l}
I_{1x}\omega_x+\mu(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34}) \\
\quad (\cos(q_7)\dot{q}_7q_8+\sin(q_7)\dot{q}_8-\sin(q_7)\dot{q}_7\cos(q_9)z_{34}-\cos(q_7)\sin(q_9)\dot{q}_9z_{34} \\
\quad\quad +\omega_x(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})-\omega_y\sin(q_9)z_{34}) \\
\quad -\mu(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34}) \\
\quad (-\sin(q_7)\dot{q}_7q_8+\cos(q_7)\dot{q}_8-\cos(q_7)\dot{q}_7\cos(q_9)z_{34}+\sin(q_7)\sin(q_9)\dot{q}_9z_{34}+\omega_z\sin(q_9)z_{34} \\
\quad\quad -\omega_x(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34})) \\
\\
I_{1y}\omega_y+\mu(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34}) \\
\quad (\cos(q_9)\dot{q}_9z_{34}+\omega_y(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34}) \\
\quad\quad -\omega_z(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})) \\
\quad -\mu\sin(q_9)z_{34} \\
\quad (\cos(q_7)\dot{q}_7q_8+\sin(q_7)\dot{q}_8-\sin(q_7)\dot{q}_7\cos(q_9)z_{34}-\cos(q_7)\sin(q_9)\dot{q}_9z_{34} \\
\quad\quad +\omega_x(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})-\omega_y\sin(q_9)z_{34}) \\
\\
I_{1z}\omega_z-\mu\sin(q_9)z_{34} \\
\quad (\sin(q_7)\dot{q}_7q_8-\cos(q_7)\dot{q}_8+\cos(q_7)\dot{q}_7\cos(q_9)z_{34}-\sin(q_7)\sin(q_9)\dot{q}_9z_{34}-\omega_z\sin(q_9)z_{34} \\
\quad\quad +\omega_x(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34})) \\
\quad -\mu(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34}) \\
\quad (\cos(q_9)\dot{q}_9z_{34}+\omega_y(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34}) \\
\quad\quad -\omega_z(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34}))
\end{array}\right)$$

Pour calculer le moment dynamique de $\mathcal{S}$ en G^t – i.e la dérivée du moment cinétique – il s'agit de différentier dans R l'expression précédente, donc de la multiplier au préalable par la matrice de passage de R_1^t dans R, de la différentier et de la remultiplier par la matrice de passage de R dans R_1^t.

Quant aux termes représentant le moment en G^t du torseur des efforts extérieurs, ils sont calculés par la formule suivante :

$$C_1+\overrightarrow{G^tG_1^t}\wedge F_1+\overrightarrow{G^tG_4^t}\wedge F_4$$

ou, pour ne pas devoir calculer explicitement la position de G^t,

$$C_1+\mu\overrightarrow{G_1^tG_4^t}\wedge\left(\frac{F_4}{\mu_4}-\frac{F_1}{\mu_1}\right)$$

ce qui s'exprime dans R_1^t par :

$$\left(\begin{array}{c}
C_{1x}+\left(\frac{F_{4z}}{\mu_4}-\frac{F_{1z}}{\mu_1}\right)\mu(q_8\cos(q_7)-\sin(q_7)\cos(q_9)z_{34})-\left(\frac{F_{4y}}{\mu_4}-\frac{F_{1y}}{\mu_1}\right)\mu(z_{12}+q_8\sin(q_7)+\cos(q_7)\cos(q_9)z_{34}) \\
C_{1y}+\left(\frac{F_{4x}}{\mu_4}-\frac{F_{1x}}{\mu_1}\right)\mu(z_{12}+q_8\sin(q_7)+\cos(q_7)\cos(q_9)z_{34})-\left(\frac{F_{4z}}{\mu_4}-\frac{F_{1z}}{\mu_1}\right)\mu\sin(q_9)z_{34} \\
C_{1z}+\left(\frac{F_{4y}}{\mu_4}-\frac{F_{1y}}{\mu_1}\right)\mu\sin(q_9)z_{34}-\left(\frac{F_{4x}}{\mu_4}-\frac{F_{1x}}{\mu_1}\right)\mu(q_8\cos(q_7)-\sin(q_7)\cos(q_9)z_{34})
\end{array}\right)$$

Par substitution dans la formule (1.3.2.ii), on obtient trois autres équations différentielles du mouvement, dont on ne reproduira ci-dessous que la première, étant données leurs tailles – on trouvera en annexe les deux autres équations :

$$
\begin{aligned}
&C_{1x}-\mu(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})\left(\tfrac{F_{1z}}{\mu_1}-\tfrac{F_{4z}}{\mu_4}\right)+\mu(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34})\left(\tfrac{F_{1y}}{\mu_1}-\tfrac{F_{4y}}{\mu_4}\right) = \\
&\quad \mu\omega_z^2\big(2\cos(q_7)^2\cos(q_9)z_{34}q_8-q_8\cos(q_9)z_{34}+z_{12}\cos(q_7)q_8-z_{12}\sin(q_7)\cos(q_9)z_{34} \\
&\qquad -\cos(q_7)\cos(q_9)^2{z_{34}}^2\sin(q_7)+\cos(q_7){q_8}^2\sin(q_7)\big) \\
&\quad +\mu\omega_y^2\big(z_{12}\sin(q_7)\cos(q_9)z_{34}-z_{12}\cos(q_7)q_8+q_8\cos(q_9)z_{34}+\cos(q_7)\cos(q_9)^2{z_{34}}^2\sin(q_7) \\
&\qquad -\cos(q_7){q_8}^2\sin(q_7)-2\cos(q_7)^2\cos(q_9)z_{34}q_8\big) \\
&\quad +\mu\omega_z\omega_x\big(\cos(q_7)q_8\sin(q_9)z_{34}-\sin(q_7)\cos(q_9){z_{34}}^2\sin(q_9)\big) \\
&\quad -\omega_z\omega_y\big(\mu\big({q_8}^2+2z_{12}\sin(q_7)q_8-\cos(q_9)^2{z_{34}}^2+2{z_{34}}^2\cos(q_9)^2\cos(q_7)^2-2\cos(q_7)^2{q_8}^2+{z_{12}}^2 \\
&\qquad +2z_{12}\cos(q_7)\cos(q_9)z_{34}+4\cos(q_7)q_8\sin(q_7)\cos(q_9)z_{34})+I_{1y}-I_{1z}\big) \\
&\quad -\mu\omega_y\omega_x\big(z_{12}\sin(q_9)z_{34}+\sin(q_7)q_8\sin(q_9)z_{34}+\cos(q_7)\cos(q_9){z_{34}}^2\sin(q_9)\big) \\
&\quad +\mu\omega_x\dot{q}_8(2z_{12}\sin(q_7)+2q_8)+\mu\omega_x\dot{q}_7(2z_{12}\cos(q_7)q_8-2z_{12}\sin(q_7)\cos(q_9)z_{34}) \\
&\quad -\mu\dot{q}_9\big(\omega_x\big(2z_{12}\cos(q_7)\sin(q_9)z_{34}+2\cos(q_9){z_{34}}^2\sin(q_9)\big)+ \\
&\qquad \omega_y\big(2\cos(q_7)q_8\cos(q_9)z_{34}-2\sin(q_7)\cos(q_9)^2{z_{34}}^2\big) \\
&\qquad +\omega_z\big(2\cos(q_7)\cos(q_9)^2{z_{34}}^2+2\sin(q_7)q_8\cos(q_9)z_{34}+2z_{12}\cos(q_9)z_{34}\big)\big) \\
&\quad +\dot{\omega}_x\big(I_{1x}+\mu\big(\cos(q_9)^2{z_{34}}^2+2z_{12}\cos(q_7)\cos(q_9)z_{34}+{z_{12}}^2+2z_{12}\sin(q_7)q_8+{q_8}^2\big)\big) \\
&\quad -\mu\dot{\omega}_z\big(z_{12}\sin(q_9)z_{34}+\sin(q_7)q_8\sin(q_9)z_{34}+\cos(q_7)\cos(q_9){z_{34}}^2\sin(q_9)\big) \\
&\quad +\mu\dot{\omega}_y\big(\sin(q_7)\cos(q_9)\sin(q_9){z_{34}}^2-\cos(q_7)q_8\sin(q_9)z_{34}\big) \\
&\quad -\mu\dot{q}_9^2(q_8\cos(q_9)z_{34}+z_{12}\sin(q_7)\cos(q_9)z_{34})-\mu\dot{q}_7^2(z_{12}\sin(q_7)\cos(q_9)z_{34}-z_{12}\cos(q_7)q_8) \\
&\quad -2\mu\dot{q}_9\dot{q}_7\big(z_{12}\cos(q_7)\sin(q_9)z_{34}+\cos(q_9)\sin(q_9){z_{34}}^2\big)+2\mu\dot{q}_8\dot{q}_7(z_{12}\sin(q_7)+q_8) \\
&\quad +\mu\ddot{q}_7\big(z_{12}\cos(q_7)\cos(q_9)z_{34}+\cos(q_9)^2{z_{34}}^2+{q_8}^2+z_{12}\sin(q_7)q_8\big)-\mu\ddot{q}_8(z_{12}\cos(q_7)+\cos(q_9)z_{34}) \\
&\quad -\mu\ddot{q}_9(q_8\sin(q_9)z_{34}+z_{12}\sin(q_7)\sin(q_9)z_{34})
\end{aligned}
\tag{N.9.a}
$$

Le fichier `newton2` comprend à la fois le code *MAPLE* pour le calcul de ces équations et pour le calcul des équations qui proviennent de l'application du théorème du centre d'inertie.

Il n'a pas été écrit dans un soucis d'économie de temps CPU dans la mesure où on a encore voulu exprimer le théorème du moment cinétique dans le repère R_1^t et en fonction des paramètres cinématiques, pour montrer que même en utilisant toute l'intuition mécanique pour limiter la taille des équations, on arrive, pour un mécanisme relativement simple, à des expressions trop grandes pour être lisibles.

Il faut ici faire intervenir la notion déjà évoquée de niveau de détail des équations : à un niveau de détail moindre, par exemple en conservant les expressions sous formes de produits et de sommes de matrices et de vecteurs, ces équations redeviendraient lisibles.

On met ainsi en évidence la nécessité de pouvoir manipuler – et simplifier, le cas échéant – des expressions matricielles et vectorielles – sans avoir affecté de composantes aux matrices et aux vecteurs considérés.

Cette possibilité n'existe pas, tout au moins sans un important travail de programmation préalable, dans les systèmes commerciaux actuels de calcul formel et en particulier en *MAPLE*.

En effet, en l'absence de véritable typage au niveau du langage, une expression de *MAPLE* n'est reconnue par le système comme étant une matrice ou un vecteur que

- dans le cas d'une expression complexe, si elle est composée d'opérateurs manipulant des matrices ou des vecteurs – `add`, `multiply`, `scalarmul`, ...– qui eux-mêmes ne peuvent s'appliquer qu'à des expressions reconnues comme matricielles ou vectorielles,
- dans le cas d'un seul identificateur, si il a été affecté au résultat d'une des fonctions `matrix` ou `vector` qui prennent en arguments les composantes scalaires.

On verra, au chapitre suivant, dans quelle mesure on peut, en *MAPLE*, définir des opérateurs comme la multiplication non-commutative, ou le produit vectoriel, permettant d'obtenir les fonctionnalités souhaitées.

D'autre part, il existe des logiciels dédiés au calcul formel – pour la plupart encore expérimentaux – qui ont étés conçus spécialement dans cet optique, et donnent la possibilité :

- de typer les expressions de manière sophistiquée,
- de définir des opérateurs pour certains type d'objets à partir de leurs propriétés algébriques,
- de définir et de faire opérer des règles de simplification spécifiques.

Parmi ces logiciels, le plus connu est *SCRATCHPAD I* – [Griesmer et Jenks, 1971] – et ses successeurs *SCRATCHPAD II* et *AXIOM* développés au centre de recherche T.J. Watson d'IBM dans l'équipe du Pr. R. Jenks. Il faut citer aussi *ULYSSE* en cours de développement à l'INRIA Sophia Antipolis au sein du projet SAFIR, que l'on aura l'occasion de présenter plus longuement et d'utiliser dans un prochain chapitre.

Les formules (N.8) et (N.9) fournissent ainsi six équations différentielles qui gouvernent les neufs paramètres $(v_x, v_y, v_z, \omega_x, \omega_y, \omega_z, q_7, q_8, q_9)$.

A l'aide des formules (I.1) et (I.2), on peut calculer $(q_1, q_2, q_3, q_4, q_5, q_6)$ dès que l'on connaît (v_x, v_y, v_z) et $(\omega_x, \omega_y, \omega_z)$.

Il est donc nécessaire, pour avoir le modèle dynamique complet, d'obtenir trois équations différentielles supplémentaires faisant intervenir soit q_1, q_2, q_3, q_4, q_5, q_6, q_7, q_8 et q_9, soit v_x, v_y, v_z, ω_x, ω_y, ω_z, q_7, q_8 et q_9.

equations complémentaires

Pour ce faire, on va isoler des corps ou des sous-systèmes du système $\mathcal{S}$ et appliquer le principe fondamental de la dynamique à chacun de ces sous-systèmes, considéré comme libre.

Il faudra bien entendu tenir compte des efforts de réaction aux liaisons, qui sont alors vus comme des efforts extérieurs aux sous-systèmes en question mais sont des grandeurs inconnues.

Ces efforts de liaisons inconnus vérifient toutefois les lois physiques de l'action-réaction, à savoir que l'effort d'un premier corps sur un second en une liaison est exactement l'opposé de l'effort du second corps sur le premier en cette liaison.

On écrira ainsi suffisamment d'équations pour pouvoir obtenir trois équations différentielles après élimination des termes d'effort inconnus.

Remarque 2.1 Il est toujours possible, quels que soient la nature et le degré de contrainte des liaisons d'un système mécanique et quelle que soit son architecture, d'écrire suffisamment

d'équations pour pouvoir obtenir autant d'équations différentielles en les paramètres du système que de paramètres.

> En effet, si on suppose que l'on doit étudier un système quelconque de n corps reliés par p liaisons ($n-1 \leq p \leq n(n-1)/2$), on a alors à déterminer au plus $6n-(n-1) = 5n+1$ paramètres.
>
> L'étude du système complet fournit 6 équations et l'étude de chaque corps considéré comme isolé fournit $6n$ équations en faisant intervenir au plus $6p$ composantes d'efforts inconnues.
>
> L'étude de tous les sous-systèmes de deux corps reliés fournit à son tour $6p$ équations ne faisant intervenir que les composantes d'efforts inconnues déjà rencontrées ci-dessus.
>
> On dispose alors de $6(n+p+1)$ (dès que $n > 2$) équations pour obtenir au plus $5n+1$ équations différentielles après élimination d'au plus $6p$ inconnues ce qui est théoriquement possible sous certaines conditions d'indépendance de ces équations.
>
> La provenance mécanique de ces équations permet d'assumer ces conditions d'indépendances, dans une certaine mesure équivalentes au fait que les efforts de liaisons inconnus ne "travaillent" pas lors du mouvement du système considéré.

premier sous-système On considère tout d'abord le corps $\mathcal{S}_1$ comme isolé du reste du système, il est soumis à trois types d'efforts extérieurs :

- la force $F_1 = F_{1x}i_1 + F_{1y}j_1 + F_{1z}k_1$ s'appliquant en g_1, et le couple $C_1 = C_{1x}i_1 + C_{1y}j_1 + C_{1z}k_1$,
- la force de réaction du corps $\mathcal{S}_2$ s'appliquant en m_{12} : $r_{21} = r_{21x}i_1 + r_{21y}j_1 + r_{21z}k_1$,
- le couple de réaction du corps $\mathcal{S}_2$: $c_{21} = c_{21x}i_1 + c_{21y}j_1 + c_{21z}k_1$, où la composante selon i_1 n'est autre, d'après le principe de l'action et de la réaction, que le couple développé par le mécanisme ressort-amortisseur de la liaison rotoïde – i.e $c_{21x} = -C_{12}$.

La loi fondamentale de la dynamique, appliquée à $\mathcal{S}_1$ ainsi isolé donne :

$$\left\{ \begin{array}{lcl} \mu_1 \gamma(G_1^t) & = & r_{21} + F_1 \\ \dfrac{d(\mathcal{I}_1\omega_1)}{dt} & = & \overrightarrow{G_1^t M_{21}^t} \wedge r_{21} + c_{21} + C_1 \end{array} \right.$$

Ce qui, si l'on se place dans le repère R_1^t, conduit aux six équations suivantes :

$$\left\{ \begin{array}{lcl} \dot{v}_x + \omega_y v_z - \omega_z v_y & = & r_{21x} + F_{1x} \\ \dot{v}_y + \omega_z v_x - \omega_x v_z & = & r_{21y} + F_{1y} \\ \dot{v}_z + \omega_x v_y - \omega_y v_x & = & r_{21z} + F_{1z} \end{array} \right. \tag{N.10}$$

$$\left\{ \begin{array}{lcl} I_{1x}\dot{\omega}_x + (I_{1z} - I_{1y})\,\omega_y\omega_z & = & -z_{12}r_{21y} - C_{12} + C_{1x} \\ I_{1y}\dot{\omega}_y + (I_{1x} - I_{1z})\,\omega_x\omega_z & = & z_{12}r_{21x} + c_{21y} + C_{1y} \\ I_{1z}\dot{\omega}_z + (I_{1y} - I_{1x})\,\omega_y\omega_x & = & c_{21z} + C_{1z} \end{array} \right. \tag{N.11}$$

étant donné que, exprimées dans le repère R_1^t, on a :

$$\gamma(G_1^t) = \left(\begin{array}{l} \dot{v}_x + \omega_y v_z - \omega_z v_y \\ \dot{v}_y + \omega_z v_x - \omega_x v_z \\ \dot{v}_z + \omega_x v_y - \omega_y v_x \end{array} \right) \quad \text{et} \quad \mathcal{I}_1\omega_1 = \left(\begin{array}{l} I_{1x}\dot{\omega}_x + (I_{1z} - I_{1y})\,\omega_y\omega_z \\ I_{1y}\dot{\omega}_y + (I_{1x} - I_{1z})\,\omega_x\omega_z \\ I_{1z}\dot{\omega}_z + (I_{1y} - I_{1x})\,\omega_y\omega_x \end{array} \right)$$

$$\text{ainsi que} \qquad r_{21} = \begin{pmatrix} r_{21x} \\ r_{21y} \\ r_{21z} \end{pmatrix}, \quad c_{21} = \begin{pmatrix} -C_{12} \\ c_{21y} \\ c_{21z} \end{pmatrix} \quad \text{et} \quad \overrightarrow{G_1^t M_{12}^t} = \begin{pmatrix} 0 \\ 0 \\ z_{12} \end{pmatrix}$$

Ces six équations introduisent cinq nouvelles inconnues, à savoir r_{21x}, r_{21y}, r_{21z}, c_{21y} et c_{21z}. On peut alors tirer de (N.10) la valeur de r_{21y} ce qui permet d'obtenir, par substitution dans la première équation de (N.11) une septième équation différentielle du mouvement :

$$I_{1x}\dot{\omega}_x + (I_{1z} - I_{1y})\,\omega_y\omega_z + z_{12}\,(\dot{v}_y + \omega_z v_x - \omega_x v_z - F_{1y}) = C_{1x} - C_{12} \tag{N.12}$$

Les calculs précédents, qui correspondent au début de fichier `newton3` de code *MAPLE*, font intervenir l'utilisateur, pour l'élimination des cinq nouvelles inconnues et l'obtention de l'équation différentielle.

Ces calculs peuvent être réalisés par *MAPLE* de manière autonome en faisant de l'élimination de Gauss sur la matrice définie à partir des systèmes d'équations (N.10) et (N.11).

En effet, si l'on écrit ces systèmes sous la forme :

$$(A)\begin{pmatrix} r_{21x} \\ r_{21y} \\ r_{21z} \\ c_{21y} \\ c_{21z} \\ 1 \end{pmatrix} = 0$$

ce qui se code facilement :

```
> eqs:=subs(c21X=-C12,[eq10a,eq10b,eq10c,eq11a,eq11b,eq11c]):
> vars:=[r21X,r21Y,r21Z,c21Y,c21Z]:
> A:=genmatrix(eqs,vars,flag);

A :=

[   -1    0    0   0   0             - vY omZ + vXp + vZ omY - f1X           ]

[   0     -1   0   0   0             - vZ omX + vYp + vX omZ - f1Y           ]

[   0     0   -1   0   0              vZp + vY omX - vX omY - f1Z            ]

[   0    z12   0   0   0  - i1Y omY omZ + i1X omXp + i1Z omZ omY - C21 - c1X ]

[ - z12   0    0  -1   0      i1Y omYp + i1X omX omZ - i1Z omZ omX - c1Y     ]

[   0     0    0   0  -1     - i1X omX omY + i1Y omY omX + i1Z omZp - c1Z    ]
```

et si l'on transforme cette matrice par la méthode de Gauss en une matrice triangulaire supérieure, alors le coefficient non nul de la dernière ligne fournit l'équation recherchée :

```
> gausselim(A);

                [-1, 0, 0, 0, 0, - vY omZ + vXp + vZ omY - f1X]

                [0, -1, 0, 0, 0, - vZ omX + vYp + vX omZ - f1Y]

                 [0, 0, -1, 0, 0, vZp + vY omX - vX omY - f1Z]

 [0, 0, 0, -1, 0, z12 f1X + z12 vY omZ - z12 vXp - z12 vZ omY - c1Y + i1Y omYp

      + i1X omX omZ - i1Z omZ omX]

        [0, 0, 0, 0, -1, - i1X omX omY + i1Y omY omX + i1Z omZp - c1Z]

[0, 0, 0, 0, 0, - i1Y omY omZ + i1X omXp + i1Z omZ omY - C21 - c1X - z12 vZ omX

     + z12 vYp + z12 vX omZ - z12 f1Y]
```

deuxième sous-système On considère ensuite le corps $\mathcal{S}_4$ comme isolé du reste du système, il est soumis aux mêmes types d'efforts extérieurs que le corps $\mathcal{S}_1$ à savoir :

- des efforts extérieurs à tout le système – i.e. la force $F_4 = F_{4x}i_1 + F_{4y}j_1 + F_{4z}k_1$ s'appliquant en g_4,
- une force de réaction du corps $\mathcal{S}_3$ s'appliquant en m_{43} : $r_{34} = r_{34x}i_4 + r_{34y}j_4 + r_{34z}k_4$ dont la composante selon j_4 est précisément, d'après le principe de l'action et de la réaction, la force exercée par le mécanisme ressort-amortisseur de la liaison prismatique entre $\mathcal{S}_2$ et $\mathcal{S}_3$ – ainsi $r_{34y} = F_{23}$,
- le couple de réaction du corps $\mathcal{S}_3$: $c_{34} = c_{34x}i_4 + c_{34y}j_4 + c_{34z}k_4$ dont, là encore, la composante selon j_4 correspond au couple développé par le mécanisme ressort-amortisseur i.e. $c_{34y} = C_{34}$.

Dans la mesure où toute la masse du corps $\mathcal{S}_4$ est supposée concentrée en g_4, l'inertie du corps $\mathcal{S}_4$ est nulle et donc le principe fondamental de la dynamique, appliqué à ce corps ainsi isolé donne :

$$\mu_4\gamma(G_4^t) = r_{34} + F_4 \qquad \text{et} \qquad 0 = \overrightarrow{G_4^t M_{43}^t} \wedge r_{34} + c_{34}$$

Les égalités précédentes se traduisent, dans le repère R_3^t, par les six équations suivantes :

$$
\left\{
\begin{array}{l}
\cos(q_9)r_{34x}+\sin(q_9)r_{34z}+F_{4x}= \\
\quad v_z\omega_y-v_y\omega_z-{\omega_y}^2\sin(q_9)z_{34}-{\omega_z}^2\sin(q_9)z_{34}+(\cos(q_7)\cos(q_9)z_{34}+\sin(q_7)q_8+z_{12})\omega_z\omega_x \\
\qquad +(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})\omega_y\omega_x \\
\qquad -((2\sin(q_7)\cos(q_9)z_{34}-2\cos(q_7)q_8)\omega_y-(2\sin(q_7)q_8+2\cos(q_7)\cos(q_9)z_{34})\omega_z)\dot{q}_7 \\
\qquad +(\cos(q_7)\cos(q_9)z_{34}+\sin(q_7)q_8+z_{12})\dot{\omega}_y+(\sin(q_7)\cos(q_9)z_{34}-\cos(q_7)q_8)\dot{\omega}_z \\
\qquad -(2\omega_z\cos(q_7)-2\omega_y\sin(q_7))\dot{q}_8 \\
\qquad -(2\omega_y\cos(q_7)\sin(q_9)z_{34}+2\omega_z\sin(q_7)\sin(q_9)z_{34})\dot{q}_9+\dot{v}_x-\sin(q_9)\dot{q}_9^2z_{34}+\cos(q_9)\ddot{q}_9z_{34} \\
\\
F_{23}+\cos(q_7)F_{4y}+\sin(q_7)F_{4z}= \\
\quad \sin(q_7)\omega_z\sin(q_9)z_{34}\omega_x+\cos(q_7)\omega_y\sin(q_9)z_{34}\omega_x \\
\qquad +\Big(2\cos(q_7)^2\cos(q_9)z_{34}+z_{12}\cos(q_7)+2\sin(q_7)q_8\cos(q_7)-\cos(q_9)z_{34}\Big)\omega_y\omega_z \\
\qquad -\sin(q_7)v_x\omega_y+\cos(q_7)v_x\omega_z+(\sin(q_7)v_y-\cos(q_7)v_z)\omega_x-(q_8+z_{12}\sin(q_7))\omega_x{}^2 \\
\qquad -\Big(\sin(q_7)\cos(q_7)\cos(q_9)z_{34}-q_8\cos(q_7)^2+z_{12}\sin(q_7)+q_8\Big)\omega_y^2 \\
\qquad +\Big(\sin(q_7)\cos(q_7)\cos(q_9)z_{34}-q_8\cos(q_7)^2\Big)\omega_z{}^2-2\omega_x\dot{q}_7q_8-(\cos(q_9)z_{34}+z_{12}\cos(q_7))\dot{\omega}_x \\
\qquad -\sin(q_7)\dot{\omega}_y\sin(q_9)z_{34}+\cos(q_7)\dot{\omega}_z\sin(q_9)z_{34} \\
\qquad -(2\sin(q_7)\cos(q_9)z_{34}\omega_y-2\sin(q_9)z_{34}\omega_x-2\cos(q_7)\omega_z\cos(q_9)z_{34})\dot{q}_9+\cos(q_7)\dot{v}_y \\
\qquad +\sin(q_7)\dot{v}_z+2\dot{q}_7\sin(q_9)\dot{q}_9z_{34}-\dot{q}_7^2q_8-\ddot{q}_7\cos(q_9)z_{34}+\ddot{q}_8 \\
\\
-\sin(q_9)r_{34x}+\cos(q_9)r_{34z}-\sin(q_7)F_{4y}+\cos(q_7)F_{4z}= \\
\quad \cos(q_7)\omega_z\sin(q_9)z_{34}\omega_x-\sin(q_7)\omega_y\sin(q_9)z_{34}\omega_x \\
\qquad -\Big(q_8+2\sin(q_7)\cos(q_7)\cos(q_9)z_{34}+z_{12}\sin(q_7)-2q_8\cos(q_7)^2\Big)\omega_y\omega_z-\cos(q_7)v_x\omega_y-\sin(q_7)v_x\omega_z \\
\qquad +(\sin(q_7)v_z+\cos(q_7)v_y)\omega_x-(\cos(q_9)z_{34}+z_{12}\cos(q_7))\omega_x{}^2 \\
\qquad -\Big(\cos(q_7)^2\cos(q_9)z_{34}+z_{12}\cos(q_7)+\sin(q_7)q_8\cos(q_7)\Big)\omega_y{}^2 \\
\qquad +\Big(\sin(q_7)q_8\cos(q_7)-\cos(q_9)z_{34}+\cos(q_7)^2\cos(q_9)z_{34}\Big)\omega_z{}^2-2\omega_x\dot{q}_7\cos(q_9)z_{34}+(q_8+z_{12}\sin(q_7))\dot{\omega}_x \\
\qquad -\cos(q_7)\dot{\omega}_y\sin(q_9)z_{34}-\sin(q_7)\dot{\omega}_z\sin(q_9)z_{34}+2\omega_x\dot{q}_8 \\
\qquad -(2\sin(q_7)\omega_z\cos(q_9)z_{34}+2\cos(q_7)\cos(q_9)z_{34}\omega_y)\dot{q}_9 \\
\qquad -\sin(q_7)\dot{v}_y+\cos(q_7)\dot{v}_z+2\dot{q}_7\dot{q}_8-\dot{q}_7^2\cos(q_9)z_{34}-\cos(q_9)\dot{q}_9^2z_{34}+\ddot{q}_7q_8-\sin(q_9)\ddot{q}_9z_{34}
\end{array}
\right. \tag{N.13}
$$

$$
\left\{
\begin{array}{l}
z_{34}F_{23}+c_{34x}=0 \\
-z_{34}r_{34x}+C_{34}=0 \\
c_{34z}=0
\end{array}
\right. \tag{N.14}
$$

La deuxième des équations (N.13) fournit immédiatement la huitième équation de la dynamique du système puisqu'elle ne fait pas intervenir de composante d'effort inconnue.

Bien entendu, *MAPLE* peut facilement tester si une des équations ne fait pas intervenir de réactions inconnues.

La fonction `genmatrix` que l'on a vu plus haut, sans l'option `flag` produira, dans ce cas, une ligne de zéros, et il existe aussi une fonction `coeff` qui rend le coefficient en une variable donnée d'un polynôme quelconque.

troisième sous-système On obtient la dernière équation de la dynamique de $\mathcal{S}$ en considérant le système $\mathcal{S}_2 \sqcup \mathcal{S}_3$ comme isolé.

Puisque la masse, et donc l'inertie de ce sous-système est nulle, le principe fondamental de la dynamique entraîne que le moment, par rapport à n'importe quel point de $\mathcal{S}_2 \sqcup \mathcal{S}_3$, du torseur des efforts extérieurs est nul ainsi que la résultante de ce torseur.

Si on note respectivement c_{12}, r_{43} et c_{43} le couple de réaction du corps $\mathcal{S}_1$ sur le corps $\mathcal{S}_2$ et la force et le couple de réaction du corps $\mathcal{S}_4$ sur le corps $\mathcal{S}_3$, on obtient :

$$\overrightarrow{m_{21}m_{34}} \wedge r_{43} + c_{12} + c_{43} = 0 \qquad \text{et} \qquad r_{43} + r_{12} = 0$$

c'est à dire, en projetant ces équations sur i_2^t, j_2^t et k_2^t :

$$\left\{\begin{array}{l} q_8\left(\sin(q_9)r_{34x} - \cos(q_9)r_{34z}\right) + C_{12} - \cos(q_9)c_{34x} - \sin(q_9)c_{34z} = 0 \\ -\cos(q_7)c_{21y} - \sin(q_7)c_{21z} - C_{34} = 0 \\ -q_8\left(-\cos(q_9)r_{34x} - \sin(q_9)r_{34z}\right) + \sin(q_7)c_{21y} \\ \qquad\qquad -\cos(q_7)c_{21z} + \sin(q_9)c_{34x} - \cos(q_9)c_{34z} = 0 \end{array}\right. \tag{N.15}$$

$$\left\{\begin{array}{l} -r_{21x} - \cos(q_9)r_{34x} - \sin(q_9)r_{34z} = 0 \\ -\cos(q_7)r_{21y} - \sin(q_7)r_{21z} - F_{23} = 0 \\ \sin(q_7)r_{21y} - \cos(q_7)r_{21z} + \sin(q_9)r_{34x} - \cos(q_9)r_{34z} = 0 \end{array}\right. \tag{N.16}$$

étant donné que dans R_2^t, on a, en tenant compte du principe de l'action et de la réaction :

$$\overrightarrow{m_{21}m_{34}} = \begin{pmatrix} 0 \\ q_8 \\ 0 \end{pmatrix}, \qquad r_{43} = \begin{pmatrix} -r_{34x}\cos(q_9) - r_{34z}\sin(q_9) \\ -F_{23} \\ r_{34x}\sin(q_9) - r_{34z}\cos(q_9) \end{pmatrix}$$

$$c_{43} = \begin{pmatrix} -c_{34x}\cos(q_9) - c_{34z}\sin(q_9) \\ -C_{34} \\ c_{34x}\sin(q_9) - c_{34z}\cos(q_9) \end{pmatrix},$$

$$r_{12} = \begin{pmatrix} -r_{21x} \\ -r_{21y}\cos(q_7) - r_{21z}\sin(q_7) \\ r_{21y}\sin(q_7) - r_{21z}\cos(q_7) \end{pmatrix} \quad \text{et} \quad c_{12} = \begin{pmatrix} C_{12} \\ -c_{21y}\cos(q_7) - c_{21z}\sin(q_7) \\ c_{21y}\sin(q_7) - c_{21z}\cos(q_7) \end{pmatrix}$$

Il reste à éliminer dans l'une des six équations (N.15) et (N.16) les composantes d'efforts inconnus, en utilisant d'autres équations les faisant intervenir, i.e. (N.10), (N.11), (N.13) ou (N.14).

Ainsi, en substituant par exemple dans l'équation (N.16.b) les expressions de r_{21y} et de r_{21z} déduites du système (N.10), on aboutit enfin à une neuvième équation de la dynamique de $\mathcal{S}$:

$$-\cos(q_7)\,(-v_z\omega_x + v_x\omega_z + \dot{v}_y - F_{1y}) - \sin(q_7)\,(\dot{v}_z - v_x\omega_y + v_y\omega_x - F_{1z}) - F_{23} = 0 \qquad \text{(N.17)}$$

Ces neuf équations différentielles permettent théoriquement de connaître le mouvement du satellite, même si elles nécessitent – peut-être – un certain conditionnement avant d'être transmises à un logiciel d'intégration numérique.

D'après ce que l'on a vu précédemment, on pourrait croire qu'il est facile en *MAPLE* d'effectuer de manière complètement systématique l'élimination des composantes d'efforts inconnus, comme on l'a fait plus haut, en formant un gros système linéaire à partir des équations (N.10), (N.11), (N.13), (N.14), (N.15) et (N.16), en le mettant sous la forme du produit d'une matrice 18×13 – puisqu'il y a 18 équations et 12 inconnues – par le vecteur des inconnues augmenté d'une ligne de composante constante non nulle, puis en faisant de l'élimination de Gauss sur cette matrice et en récupérant le treizième coefficient, parmi les lignes dont les douzes premiers coefficients sont nuls.

Malheureusement, dans la plupart des cas et en particulier sur notre exemple, la matrice que l'on obtient a des coefficients qui dépendent de manière complexe des paramètres q_i du mécanisme et donc du temps.

Ces coefficients risquent donc de s'annuler pour toutes ou pour certaines des valeurs de t sans que l'on soit capable de le détecter – ce qui pourrait amener à effectuer des divisions par 0.

Pour éviter ces erreurs, *MAPLE* refuse de pratiquer la méthode de Gauss, dès que certains des coefficients de la matrice à trianguler – ceux par lesquels on est susceptible de diviser – ne sont pas reconnus comme appartenant à un corps de fractions rationnelles, comme en témoigne l'exemple suivant :

```
>  A := matrix(3,3,[x,1,0,0,0,1,1,y,1]);

                                  [ x  1  0 ]
                              A := [ 0  0  1 ]
                                  [ 1  y  1 ]

> gausselim(A);

                                [ x     1     0 ]
                                [    y x - 1    ]
                                [ 0  -------  1 ]
                                [       x       ]
                                [ 0     0     1 ]

>  A := matrix(3,3,[x(t),1,0,0,0,1,1,y,1]);

                                    [ x(t)  1  0 ]
                                A := [   0   0  1 ]
                                    [   1   y  1 ]

> gausselim(A);
Error, (in gausselim) matrix entries must be rational polynomials
```

On est en fait confronté ici à deux problèmes différents, inhérents à l'emploi de méthodes de calcul formel, qui sont fondamentaux :

i. Le fait de savoir si une expression qui dépend d'un certain nombre de paramètres est identiquement nulle ou non.
ii. Le fait de savoir si une expression qui dépend d'un certain nombre de paramètres peut s'annuler ou non pour certaines valeurs de ces paramètres.

Ces problèmes sont moins simples qu'ils ne le paraissent et font l'objet d'une importante activité de recherche.

Le problème (i) dépend du domaine mathématique auquel appartient l'expression en question :

- si cette expression est une fraction rationnelle, l'examen des coefficients – numériques – de cette fraction permet de décider immédiatement si elle est identiquement nulle ou non,
- si on considère, par exemple, un polynôme en les fonctions circulaires des multiples entiers d'un paramètre

$$P(\sin(q), \cos(q), \sin(2q), \cos(2q), \ldots, \sin(nq), \cos(nq), \ldots)$$

il faudra, pour tester s'il est identiquement nul, exprimer toutes les lignes trigonométriques en fonction de $\sin(q)$ et de $\cos(q)$ et tenir compte de l'égalité

$$\sin^2(q) + \cos^2(q) = 1$$

- tester si un polynôme de plusieurs variables est identiquement nul, modulo certaines relations polynômiales revient à tester l'appartenance de ce polynôme à un idéal et nécessite d'avoir recours à des algorithmes sophistiqués – bases de Gröbner – et en général coûteux
- il existe même des classes de fonctions – voir [Moses, 1971] – pour lesquelles la nullité d'une expression construite à partir de fonctions de ces classes est indécidable.

Le problème (ii) contient, quant à lui, à la fois le problème général de la résolution d'équations sous forme littérale sur le plan mathématique, et sur le plan informatique, le problème de pouvoir gérer une discussion en faisant des hypothèses successives, en étudiant leur compatibilité, en revenant sur des calculs antérieurs, ...

Dans l'exemple précédent, on aimerait, en effet avoir le résultat suivant :

```
>  A := matrix(3,3,[x,1,0,0,0,1,1,y,1]);
```

$$A := \begin{bmatrix} x & 1 & 0 \\ 0 & 0 & 1 \\ 1 & y & 1 \end{bmatrix}$$

```
> gausselim(A);

  si x <> 0
```

$$\begin{bmatrix} x & 1 & 0 \\ & y\,x - 1 & \\ 0 & \dfrac{}{x} & 1 \\ & x & \\ 0 & 0 & 1 \end{bmatrix}$$

```
sinon
                                  [ 1     y     1 ]
                                  [               ]
                                  [ 0     1     0 ]
                                  [               ]
                                  [ 0     0     1 ]
```

Pour conclure, à moins de connaître le domaine mathématique des expressions intervenant dans les calculs et de pouvoir facilement tester la nullité d'une expression dans ce domaine, on sera amené, pour automatiser le procédé d'élimination des efforts inconnus, à approfondir l'étude de la méthode de mise en équation du mouvement, et à écrire un algorithme décidant :

- d'une part du choix des sous-systèmes à considérer comme libres pour avoir des équations complémentaires,
- d'autre part, de l'élimination des composantes d'efforts inconnues.

Cet algorithme sera présenté au chapitre suivant.

2.3 une autre méthode classique – équations de d'Alembert

premières équations

On revient tout d'abord sur le fait qu'en ce qui concerne la résultante des torseurs dynamiques et des efforts extérieurs, il est complètement équivalent d'écrire le principe de d'Alembert et le principe fondamental de la dynamique.

Cela tient à la façon dont on exprime d'un côté les forces d'inertie et de l'autre côté l'accélération du centre d'inertie du système. En effet, au paragraphe précédent, on a vu que :

$$\mu(\mathcal{S})\gamma(G^t) = \sum_i \mu(\mathcal{S}_i)\gamma(G_i^t)$$

et, d'après les définitions de d'Alembert, pour chaque corps la résultante du torseur d'inertie s'écrit $\mu(\mathcal{S}_i)\gamma(G_i^t)$.

Ainsi les trois premières équations de la dynamique du système sont obtenues rigoureusement par les mêmes calculs qu'au paragraphe précédent et s'écrivent donc comme en (N.7.a), (N.7.b) et (N.7.c) :

$$\begin{aligned}
F_{1x}+F_{4x} \;=\; & \\
& (\mu_1+\mu_4)v_z\omega_y-(\mu_1+\mu_4)v_y\omega_z+(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34})\mu_4\omega_z\omega_x \\
& +(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})\mu_4\omega_y\omega_x-\left(\omega_y^2+\omega_z^2\right)\mu_4\sin(q_9)z_{34} \\
& +(2\mu_4\omega_y\sin(q_7)-2\mu_4\omega_z\cos(q_7))\dot{q}_8 \\
& -(2\mu_4\omega_y\cos(q_7)\sin(q_9)z_{34}+2\mu_4\omega_z\sin(q_7)\sin(q_9)z_{34})\dot{q}_9 \\
& +((2\cos(q_7)q_8-2\sin(q_7)\cos(q_9)z_{34})\mu_4\omega_y+(2\cos(q_7)\cos(q_9)z_{34}+2\sin(q_7)q_8)\mu_4\omega_z)\dot{q}_7 \\
& -(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})\mu_4\dot{\omega}_z+(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34})\mu_4\dot{\omega}_y \\
& -\mu_4\sin(q_9)\dot{q}_9^2z_{34}+(\mu_1+\mu_4)\dot{v}_x+\mu_4\cos(q_9)\ddot{q}_9z_{34}
\end{aligned} \tag{A.1.a}$$

$$
\begin{aligned}
F_{1y}+F_{4y} \;=\; & \\
& (\mu_1+\mu_4)v_x\omega_z-(\mu_1+\mu_4)v_z\omega_x+(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34})\mu_4\omega_z\omega_y+\mu_4\omega_y\sin(q_9)z_{34}\omega_x \\
& -(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})\mu_4\left(\omega_x^2+\omega_z^2\right)-2\mu_4\omega_x\sin(q_7)\dot{q}_8 \\
& +(2\mu_4\omega_z\cos(q_9)z_{34}+2\mu_4\omega_x\cos(q_7)\sin(q_9)z_{34})\dot{q}_9+(2\sin(q_7)\cos(q_9)z_{34}-2\cos(q_7)q_8)\mu_4\omega_x\dot{q}_7 \\
& -(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34})\mu_4\dot{\omega}_x+\mu_4\dot{\omega}_z\sin(q_9)z_{34}+2\mu_4\cos(q_7)\dot{q}_7\sin(q_9)\dot{q}_9z_{34} \\
& -2\mu_4\sin(q_7)\dot{q}_7\dot{q}_8+\mu_4\sin(q_7)\cos(q_9)\dot{q}_9^2z_{34}+(\mu_1+\mu_4)\dot{v}_y \\
& -(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})\mu_4\dot{q}_7^2-(\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34})\mu_4\ddot{q}_7 \\
& +\mu_4\cos(q_7)\ddot{q}_8+\mu_4\sin(q_7)\sin(q_9)\ddot{q}_9z_{34}
\end{aligned}
\tag{A.1.b}
$$

$$
\begin{aligned}
F_{1z}+F_{4z} \;=\; & \\
& (\mu_1+\mu_4)v_y\omega_x-(\mu_1+\mu_4)v_x\omega_y+(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})\mu_4\omega_z\omega_y+\mu_4\omega_z\sin(q_9)z_{34}\omega_x \\
& -(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34})\mu_4\left(\omega_x^2+\omega_y^2\right)+2\mu_4\omega_x\cos(q_7)\dot{q}_8 \\
& -(2\mu_4\omega_y\cos(q_9)z_{34}-2\mu_4\omega_x\sin(q_7)\sin(q_9)z_{34})\dot{q}_9-(2\cos(q_7)\cos(q_9)z_{34}+2\sin(q_7)q_8)\mu_4\omega_x\dot{q}_7 \\
& +(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})\mu_4\dot{\omega}_x-\mu_4\dot{\omega}_y\sin(q_9)z_{34}+2\mu_4\sin(q_7)\dot{q}_7\sin(q_9)\dot{q}_9z_{34} \\
& +2\mu_4\cos(q_7)\dot{q}_7\dot{q}_8-\mu_4\cos(q_7)\cos(q_9)\dot{q}_9^2z_{34}+(\mu_1+\mu_4)\dot{v}_z \\
& -(\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34})\mu_4\dot{q}_7^2+(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})\mu_4\ddot{q}_7 \\
& +\mu_4\sin(q_7)\ddot{q}_8-\mu_4\cos(q_7)\sin(q_9)\ddot{q}_9z_{34}
\end{aligned}
\tag{A.1.c}
$$

On obtient trois autres équations différentielles de la dynamique en écrivant que la somme des moments des forces (extérieures et d'inertie) en n'importe quel point du système considéré est nulle.

Si l'on choisit d'écrire ces moments au centre d'inertie G^t du système, on aboutit de nouveau aux équations (N.9) provenant de l'égalité des moments en G^t du torseur dynamique et du torseur des efforts extérieurs, et ceci en effectuant à peu près les mêmes calculs.

On a vu que l'intérêt du principe de d'Alembert est de permettre de choisir un autre point du système, où il sera plus facile de calculer la somme des moments des forces.

Ainsi, dans cet exemple, on choisira d'écrire que la somme des moments des forces extérieures et d'inertie est nulle au point G_1^t, position à l'instant t de la particule g_1, centre de masse de $\mathcal{S}_1$.

Les efforts extérieurs s'exerçant sur $\mathcal{S}$ sont, on le rappelle,

- un couple C_1 agissant sur $\mathcal{S}_1$ et une force F_1 de point d'application G_1^t,
- une force F_4 de point d'application G_4^t.

En G_1^t, le moment de C_1 est bien entendu C_1, le moment de F_1 est nul – puisque F_1 s'applique en G_1^t – et le moment de F_4 vaut :

$$\overrightarrow{G_1^tG_4^t}\wedge F_4$$

ce qui s'exprime dans R_1^t par :

$$
\begin{pmatrix}
\left(-\sin(q_7)\cos(q_9)z_{34}+\cos(q_7)q_8\right)F_{4z}-\left(\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34}+z_{12}\right)F_{4y} \\
\left(\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34}+z_{12}\right)F_{4x}-\sin(q_9)z_{34}F_{4z} \\
\sin(q_9)z_{34}F_{4y}-\left(-\sin(q_7)\cos(q_9)z_{34}+\cos(q_7)q_8\right)F_{4x}
\end{pmatrix}
$$

Quant aux forces d'inerties, on peut les décomposer en deux catégories :

- les forces s'exerçant sur des points de $\mathcal{S}_1$ que l'on peut représenter par :
  - un couple C_{i1} valant :
$$-\frac{d(\mathcal{I}_1\omega_1)}{dt} = -\left(\mathcal{I}_1\frac{d\omega_1}{dt} + \omega_1 \wedge \mathcal{I}_1\omega_1\right)$$
  - une force F_{i1} s'exerçant en G_1^t valant $-\mu_1\gamma(G_1^t)$,
- les forces s'exerçant sur des points de $\mathcal{S}_4$ que l'on représente, dans la mesure où toute la masse de $\mathcal{S}_4$ est concentrée en g_4, par une force F_{i4} s'appliquant en G_4^t et valant $-\mu_4\gamma(G_4^t)$.

En G_1^t, le moment de C_{i1} est bien entendu C_{i1}, le moment de F_{i1} est nul – puisque F_{i1} s'applique en G_1^t – et le moment de F_{i4} vaut :
$$\overrightarrow{G_1^tG_4^t} \wedge F_{i4}$$
Dans le repère R_1^t, C_{i1} s'exprime alors :
$$\begin{pmatrix} -I_{1x}\dot{\omega}_x - \omega_y\omega_z\,(I_{1z} - I_{1y}) \\ -I_{1y}\dot{\omega}_y - \omega_z\omega_x\,(I_{1x} - I_{1z}) \\ -I_{1z}\dot{\omega}_z - \omega_x\omega_y\,(I_{1y} - I_{1x}) \end{pmatrix}$$
Pour connaître l'expression de F_{i4}, on doit au préalable calculer $\gamma(G_4^t)$, par exemple en exprimant $v(G_4^t)$ dans R_1^t puis dans R, et en dérivant par rapport au temps.
On obtient l'expression ci-dessous, dans R_1^t, en remultipliant par la matrice de passage de R dans R_1^t, transposée de la matrice de la rotation $\bar{\rho}_1^t$.

$$\begin{pmatrix}
-v_y\omega_z+v_z\omega_y+(-\sin(q_7)\cos(q_9)z_{34}+\cos(q_7)q_8)\omega_y\omega_x+(\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34}+z_{12})\omega_z\omega_x \\
\quad -{\omega_z}^2\sin(q_9)z_{34}-{\omega_y}^2\sin(q_9)z_{34} \\
\quad +((2\cos(q_7)\cos(q_9)z_{34}+2\sin(q_7)q_8)\omega_z+(-2\sin(q_7)\cos(q_9)z_{34}+2\cos(q_7)q_8)\omega_y)\dot{q}_7 \\
\quad +(\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34}+z_{12})\dot{\omega}_y+(-\cos(q_7)q_8+\sin(q_7)\cos(q_9)z_{34})\dot{\omega}_z+(2\omega_y\sin(q_7)-2\omega_z\cos(q_7))\dot{q}_8 \\
\quad +(-2\omega_z\sin(q_7)\sin(q_9)z_{34}-2\omega_y\cos(q_7)\sin(q_9)z_{34})\dot{q}_9+\dot{v}_x-\sin(q_9)\dot{q}_9^2z_{34}+\cos(q_9)\ddot{q}_9z_{34} \\
\\
(-\cos(q_7)q_8+\sin(q_7)\cos(q_9)z_{34}){\omega_x}^2+v_x\omega_z-v_z\omega_x+(\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34}+z_{12})\omega_z\omega_y+\omega_y\sin(q_9)z_{34}\omega_x \\
\quad +(-\cos(q_7)q_8+\sin(q_7)\cos(q_9)z_{34}){\omega_z}^2+(2\sin(q_7)\cos(q_9)z_{34}-2\cos(q_7)q_8)\omega_x\dot{q}_7 \\
\quad +(-\cos(q_7)q_8+\sin(q_7)\cos(q_9)z_{34})\dot{q}_7^2-2\sin(q_7)\dot{q}_7\dot{q}_8+(-\cos(q_7)\cos(q_9)z_{34}-z_{12}-\sin(q_7)q_8)\dot{\omega}_x \\
\quad +\dot{\omega}_z\sin(q_9)z_{34}-2\omega_x\sin(q_7)\dot{q}_8+(2\omega_z\cos(q_9)z_{34}+2\omega_x\cos(q_7)\sin(q_9)z_{34})\dot{q}_9+\dot{v}_y+\sin(q_7)\cos(q_9)\dot{q}_9^2z_{34} \\
\quad +2\cos(q_7)\dot{q}_7\sin(q_9)\dot{q}_9z_{34}+(-\sin(q_7)q_8-\cos(q_7)\cos(q_9)z_{34})\ddot{q}_7+\cos(q_7)\ddot{q}_8+\sin(q_7)\sin(q_9)\ddot{q}_9z_{34} \\
\\
(-\cos(q_7)\cos(q_9)z_{34}-z_{12}-\sin(q_7)q_8)\omega_x^2-v_x\omega_y+v_y\omega_x+(-\sin(q_7)\cos(q_9)z_{34}+\cos(q_7)q_8)\omega_z\omega_y+\omega_z\sin(q_9)z_{34}\omega_x \\
\quad +(-\cos(q_7)\cos(q_9)z_{34}-z_{12}-\sin(q_7)q_8){\omega_y}^2+(-2\cos(q_7)\cos(q_9)z_{34}-2\sin(q_7)q_8)\omega_x\dot{q}_7 \\
\quad +(-\sin(q_7)q_8-\cos(q_7)\cos(q_9)z_{34})\dot{q}_7^2+2\cos(q_7)\dot{q}_7\dot{q}_8+(-\sin(q_7)\cos(q_9)z_{34}+\cos(q_7)q_8)\dot{\omega}_x-\dot{\omega}_y\sin(q_9)z_{34} \\
\quad +2\omega_x\cos(q_7)\dot{q}_8+(2\omega_x\sin(q_7)\sin(q_9)z_{34}-2\omega_y\cos(q_9)z_{34})\dot{q}_9+\dot{v}_z-\cos(q_7)\cos(q_9)\dot{q}_9^2z_{34}+2\sin(q_7)\dot{q}_7\sin(q_9)\dot{q}_9z_{34} \\
\quad +(-\sin(q_7)\cos(q_9)z_{34}+\cos(q_7)q_8)\ddot{q}_7+\sin(q_7)\ddot{q}_8-\cos(q_7)\sin(q_9)\ddot{q}_9z_{34}
\end{pmatrix}$$

On en déduit l'expression du moment de F_{i4} et la somme des moments en G_1^t des forces extérieures et d'inertie s'exerçant sur toutes les particules de $\mathcal{S}$ dans le repère R_1^t.

Cette expression fournit trois nouvelles équations différentielles du mouvement, par projection sur les trois axes du repère.

On reproduit, ci-dessous, la première de ces trois équations, les autres, étant donnée leurs tailles sont reproduites en annexes.

$$
\begin{aligned}
&C_{1x}+(q_8\sin(q_9)z_{34}+z_{12}\sin(q_7)\sin(q_9)z_{34})\mu_4\ddot{q}_9+q_8\cos(q_7)F_{4z}+(\cos(q_9)z_{34}+z_{12}\cos(q_7))\mu_4\ddot{q}_8\\
&\quad-(2q_8+2z_{12}\sin(q_7))\mu_4\omega_x\dot{q}_8-\big(\sin(q_9){z_{34}}^2\sin(q_7)\cos(q_9)-\sin(q_9)z_{34}\cos(q_7)q_8\big)\mu_4\dot{\omega}_y\\
&\quad+(z_{12}\sin(q_7)\cos(q_9)z_{34}+q_8\cos(q_9)z_{34})\mu_4\dot{q}_9^2+\big(2z_{12}\cos(q_7)\sin(q_9)z_{34}+2\sin(q_9){z_{34}}^2\cos(q_9)\big)\mu_4\dot{q}_9\dot{q}_7\\
&\quad-(-\sin(q_7)\cos(q_9)z_{34}+\cos(q_7)q_8)\mu_4\dot{v}_z+(z_{12}\sin(q_7)\cos(q_9)z_{34}-z_{12}\cos(q_7)q_8)\mu_4\dot{q}_7^2-q_8\sin(q_7)F_{4y}\\
&\quad-(2q_8+2z_{12}\sin(q_7))\mu_4\dot{q}_8\dot{q}_7+(\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34}+z_{12})\mu_4\dot{v}_y\\
&\quad+\dot{q}_9\Big(\big(2{z_{34}}^2\cos(q_7)\cos(q_9)^2+2\sin(q_7)q_8\cos(q_9)z_{34}+2z_{12}\cos(q_9)z_{34}\big)\mu_4\omega_z\\
&\qquad+\big(2\cos(q_7)q_8\cos(q_9)z_{34}-2{z_{34}}^2\sin(q_7)\cos(q_9)^2\big)\mu_4\omega_y\\
&\qquad+\big(2z_{12}\cos(q_7)\sin(q_9)z_{34}+2\sin(q_9){z_{34}}^2\cos(q_9)\big)\mu_4\omega_x\Big)\\
&\quad-\big(z_{12}\cos(q_7)\cos(q_9)z_{34}+{z_{34}}^2\cos(q_9)^2+z_{12}\sin(q_7)q_8+{q_8}^2\big)\mu_4\ddot{q}_7\\
&\quad+\big(\sin(q_9)z_{34}^2\cos(q_7)\cos(q_9)+\sin(q_7)q_8\sin(q_9)z_{34}+z_{12}\sin(q_9)z_{34}\big)\mu_4\dot{\omega}_z\\
&\quad-\Big(I_{1z}-I_{1y}+\big({z_{34}}^2\cos(q_9)^2-4\sin(q_7)q_8\cos(q_7)\cos(q_9)z_{34}-2z_{12}\cos(q_7)\cos(q_9)z_{34}-2z_{12}\sin(q_7)q_8-{z_{12}}^2\\
&\qquad+2{q_8}^2\cos(q_7)^2-2\cos(q_7)^2\cos(q_9)^2{z_{34}}^2-{q_8}^2\big)\mu_4\Big)\omega_y\omega_z \qquad \text{(A.2.a)}\\
&\quad+(2z_{12}\sin(q_7)\cos(q_9)z_{34}-2z_{12}\cos(q_7)q_8)\mu_4\omega_x\dot{q}_7\\
&\quad+\big(\sin(q_9){z_{34}}^2\cos(q_7)\cos(q_9)+\sin(q_7)q_8\sin(q_9)z_{34}+z_{12}\sin(q_9)z_{34}\big)\mu_4\omega_y\omega_x\\
&\quad+\big(\sin(q_9){z_{34}}^2\sin(q_7)\cos(q_9)-\sin(q_9)z_{34}\cos(q_7)q_8\big)\mu_4\omega_x\omega_z\\
&\quad+\big(q_8\cos(q_9)z_{34}+\cos(q_7)\cos(q_9)^2{z_{34}}^2\sin(q_7)-z_{12}\cos(q_7)q_8+z_{12}\sin(q_7)\cos(q_9)z_{34}\\
&\qquad-\sin(q_7){q_8}^2\cos(q_7)-2\cos(q_7)^2\cos(q_9)z_{34}q_8\big)\mu_4{\omega_z}^2\\
&\quad-\cos(q_7)\cos(q_9)z_{34}F_{4y}+(\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34}+z_{12})\mu_4v_x\omega_z\\
&\quad-\big(q_8\cos(q_9)z_{34}+\cos(q_7)\cos(q_9)^2{z_{34}}^2\sin(q_7)-z_{12}\cos(q_7)q_8+z_{12}\sin(q_7)\cos(q_9)z_{34}\\
&\qquad-\sin(q_7){q_8}^2\cos(q_7)-2\cos(q_7)^2\cos(q_9)z_{34}q_8\big)\mu_4{\omega_y}^2\\
&\quad-z_{12}F_{4y}-((-\sin(q_7)\cos(q_9)z_{34}+\cos(q_7)q_8)\mu_4v_y+(\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34}+z_{12})\mu_4v_z)\omega_x\\
&\quad-\big(I_{1x}+\big({z_{34}}^2\cos(q_9)^2+{q_8}^2+2z_{12}\cos(q_7)\cos(q_9)z_{34}+2z_{12}\sin(q_7)q_8+{z_{12}}^2\big)\mu_4\big)\dot{\omega}_x\\
&\quad-(-\cos(q_7)q_8+\sin(q_7)\cos(q_9)z_{34})\mu_4v_x\omega_y-z_{34}\sin(q_7)\cos(q_9)F_{4z}
\end{aligned}
$$

Ces équations diffèrent essentiellement des équations (N.9) par le fait qu'elle font intervenir le corps $\mathcal{S}_1$ par sa vitesse linéaire et non par la force extérieure qui s'y applique. Contrairement à ce qui était prévu, elles sont plus complexes que les équations (N.9) – 86 termes additifs pour l'expression précédente contre 78 pour l'équation correspondante établie avec la méthode de Newton.

Ceci montre la difficulté de faire un choix "simplificateur" du point où l'on annulera la somme des moments des forces.

En ce qui concerne l'établissement de ces équations à l'aide de *MAPLE* – le code correspondant se trouve dans le fichier `dalembert1` – on a utilisé les mêmes fonctionnalités du logiciel que lors de la mise en équation du satellite par la méthode de Newton-Euler.

Le gain de temps réalisé ici – 581 secondes de temps CPU contre 1351 précédemment – provient, cette fois encore, de l'utilisation de la fonction `simplify` et est du à la fois à la taille et à la nature des expressions sur lesquelles on applique cette fonction.

Ce gain dépend donc du système mécanique que l'on étudie, et on peut difficilement l'attribuer à la méthode elle-même.

équations des corps isolés

Comme lorsque l'on applique la loi fondamentale de la dynamique, il nous manque maintenant, pour avoir un modèle complet de la dynamique du système étudié, trois autres équations différentielles que l'on va obtenir en réappliquant le principe de d'Alembert à des corps ou à des sous-système de $\mathcal{S}$ considérés comme isolés du reste du système.

$\mathcal{S}_4$ isolé Considérons tout d'abord le corps $\mathcal{S}_4$ comme isolé. Etant données les suppositions que l'on a faites sur la distribution des masses de ce corps, les seuls points où il est intéressant de considérer la somme du moment des forces sont les points G_4^t, position du centre de masse g_4 et M_{43}^t, point d'articulation du corps $\mathcal{S}_4$ avec le corps $\mathcal{S}_3$.

Appliquer le principe de d'Alembert au point G_4^t est complètement équivalent à appliquer la loi fondamentale de la dynamique.

On obtiendrait donc les mêmes équations qu'au paragraphe précédent avec les mêmes calculs, c'est à dire, avec les notations déjà introduites :

$$\left\{\begin{array}{rcl} r_{34} + F_4 & = & \mu_4 \gamma(G_4^t) \\ c_{34} + \overrightarrow{G_4^t M_{43}^t} \wedge r_{34} & = & 0 \end{array}\right.$$

Tandis que si l'on applique le principe de d'Alembert au point M_{43}^t, la somme des moments des efforts extérieurs et d'inertie s'exprime en ce point par :

$$c_{34} + \overrightarrow{M_{43}^t G_4^t} F_4$$

$r_{34}+F_4=\mu_4\gamma(G_4^t)$ se traduit, comme précédemment dans le repère R_3^t par le système d'équations :

$$
\left\{
\begin{array}{l}
\cos(q_9)r_{34x}+\sin(q_9)r_{34z}+F_{4x}= \\
\quad v_z\omega_y-v_y\omega_z-{\omega_y}^2\sin(q_9)z_{34}-{\omega_z}^2\sin(q_9)z_{34}+(\cos(q_7)\cos(q_9)z_{34}+\sin(q_7)q_8+z_{12})\omega_z\omega_x \\
\qquad +(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})\omega_y\omega_x \\
\qquad -((2\sin(q_7)\cos(q_9)z_{34}-2\cos(q_7)q_8)\omega_y-(2\sin(q_7)q_8+2\cos(q_7)\cos(q_9)z_{34})\omega_z)\dot{q}_7 \\
\qquad +(\cos(q_7)\cos(q_9)z_{34}+\sin(q_7)q_8+z_{12})\dot{\omega}_y+(\sin(q_7)\cos(q_9)z_{34}-\cos(q_7)q_8)\dot{\omega}_z \\
\qquad -(2\omega_z\cos(q_7)-2\omega_y\sin(q_7))\dot{q}_8 \\
\qquad -(2\omega_y\cos(q_7)\sin(q_9)z_{34}+2\omega_z\sin(q_7)\sin(q_9)z_{34})\dot{q}_9+\dot{v}_x-\sin(q_9)\dot{q}_9^2z_{34}+\cos(q_9)\ddot{q}_9z_{34} \\
\\
F_{23}+\cos(q_7)F_{4y}+\sin(q_7)F_{4z}= \\
\quad \sin(q_7)\omega_z\sin(q_9)z_{34}\omega_x+\cos(q_7)\omega_y\sin(q_9)z_{34}\omega_x \\
\qquad +\Big(2\cos(q_7)^2\cos(q_9)z_{34}+z_{12}\cos(q_7)+2\sin(q_7)q_8\cos(q_7)-\cos(q_9)z_{34}\Big)\omega_y\omega_z \\
\qquad -\sin(q_7)v_x\omega_y+\cos(q_7)v_x\omega_z+(\sin(q_7)v_y-\cos(q_7)v_z)\omega_x-(q_8+z_{12}\sin(q_7))\omega_x{}^2 \\
\qquad -\Big(\sin(q_7)\cos(q_7)\cos(q_9)z_{34}-q_8\cos(q_7)^2+z_{12}\sin(q_7)+q_8\Big)\omega_y^2 \\
\qquad +\Big(\sin(q_7)\cos(q_7)\cos(q_9)z_{34}-q_8\cos(q_7)^2\Big){\omega_z}^2-2\omega_x\dot{q}_7q_8-(\cos(q_9)z_{34}+z_{12}\cos(q_7))\dot{\omega}_x \\
\qquad -\sin(q_7)\dot{\omega}_y\sin(q_9)z_{34}+\cos(q_7)\dot{\omega}_z\sin(q_9)z_{34} \\
\qquad -(2\sin(q_7)\cos(q_9)z_{34}\omega_y-2\sin(q_9)z_{34}\omega_x-2\cos(q_7)\omega_z\cos(q_9)z_{34})\dot{q}_9+\cos(q_7)\dot{v}_y \\
\qquad +\sin(q_7)\dot{v}_z+2\dot{q}_7\sin(q_9)\dot{q}_9z_{34}-\dot{q}_7^2q_8-\ddot{q}_7\cos(q_9)z_{34}+\ddot{q}_8 \\
\\
-\sin(q_9)r_{34x}+\cos(q_9)r_{34z}-\sin(q_7)F_{4y}+\cos(q_7)F_{4z}= \\
\quad \cos(q_7)\omega_z\sin(q_9)z_{34}\omega_x-\sin(q_7)\omega_y\sin(q_9)z_{34}\omega_x \\
\qquad -\Big(q_8+2\sin(q_7)\cos(q_7)\cos(q_9)z_{34}+z_{12}\sin(q_7)-2q_8\cos(q_7)^2\Big)\omega_y\omega_z-\cos(q_7)v_x\omega_y-\sin(q_7)v_x\omega_z \\
\qquad +(\sin(q_7)v_z+\cos(q_7)v_y)\omega_x-(\cos(q_9)z_{34}+z_{12}\cos(q_7))\omega_x{}^2 \\
\qquad -\Big(\cos(q_7)^2\cos(q_9)z_{34}+z_{12}\cos(q_7)+\sin(q_7)q_8\cos(q_7)\Big){\omega_y}^2 \\
\qquad +\Big(\sin(q_7)q_8\cos(q_7)-\cos(q_9)z_{34}+\cos(q_7)^2\cos(q_9)z_{34}\Big){\omega_z}^2-2\omega_x\dot{q}_7\cos(q_9)z_{34}+(q_8+z_{12}\sin(q_7))\dot{\omega}_x \\
\qquad -\cos(q_7)\dot{\omega}_y\sin(q_9)z_{34}-\sin(q_7)\dot{\omega}_z\sin(q_9)z_{34}+2\omega_x\dot{q}_8 \\
\qquad -(2\sin(q_7)\omega_z\cos(q_9)z_{34}+2\cos(q_7)\cos(q_9)z_{34}\omega_y)\dot{q}_9 \\
\qquad -\sin(q_7)\dot{v}_y+\cos(q_7)\dot{v}_z+2\dot{q}_7\dot{q}_8-\dot{q}_7^2\cos(q_9)z_{34}-\cos(q_9)\dot{q}_9^2z_{34}+\ddot{q}_7q_8-\sin(q_9)\ddot{q}_9z_{34}
\end{array}
\right. \tag{A.3}
$$

et, en ce qui concerne les moments, au lieu d'avoir :

$$
\left\{
\begin{array}{l}
z_{34}F_{23}+c_{34x}=0 \\
-z_{34}r_{34x}+C_{34}=0 \\
c_{34z}=0
\end{array}
\right. \tag{A.4}
$$

on obtient les équations (en projetant dans le repère R_4^t) :

$$\left\{\begin{array}{c} -z_{34}\left(\cos(q_7)F_{4y}+\sin(q_7)F_{4z}\right)+c_{34x}=0 \\ z_{34}(\cos(q_9)F_{4x}+\sin(q_7)\sin(q_9F_{4y}-\cos(q_7)\sin(q_9)F_{4z})+C_{34}=0 \\ c_{34z}=0 \end{array}\right. \tag{A.5}$$

La deuxième des équations (A.3) ne faisant pas intervenir de composantes d'effort inconnues, on peut la considérer comme la septième équation différentielle du mouvement du satellite.

Et on dispose de plus d'une huitième équation différentielle du mouvement puisque la deuxième équation de (A.5) ne fait pas non plus intervenir de composantes d'efforts inconnus.

Les quatre autres équations ne sont pas utilisables directement, mais pourront se révéler utiles, le cas échéant pour exprimer les composantes de r_{34} et de c_{34} en fonctions des paramètres du système.

Cette fois, l'interêt de pouvoir choisir le point où on annule la somme des moments des forces est beaucoup plus flagrant, puisque l'on obtient directement deux équations au lieu d'une en effectuant quasiment le même travail. En ce qui concerne l'étude des corps isolés, on peut même dégager, heuristiquement, un algorithme de choix de ce point en remarquant que la somme des moments de forces en un point ne comporte pas de contribution des forces dont la ligne d'action passe par ce point. Ainsi en choisissant un point de liaison avec un autre corps, on évite ipso facto les composantes inconnues de l'effort de réaction de cette liaison.

$\mathcal{S}_1$ isolé Pour tirer au maximum avantage des possibilités de choix du point où l'on écrit le principe de d'Alembert, on va chercher à obtenir la dernière équation différentielle du mouvement de $\mathcal{S}$ à partir du corps $\mathcal{S}_1$ considéré comme isolé en écrivant que la somme des moments des efforts extérieurs et d'inertie au point M_{12}^t est nulle, i.e. :

$$-\frac{d(\mathcal{I}_1\omega_1)}{dt}+\overrightarrow{M_{21}^tG_1^t}\wedge F_1+c_{21}+C_1=0$$

ce qui s'exprime, dans le repère R_1^t :

$$\left\{\begin{array}{l} -I_{1x}\dot{\omega}_x-\omega_yI_{1z}\omega_z+\omega_zI_{1y}\omega_y+z_{12}F_{1y}-C_{12}+C_{1x}=0 \\ -I_{1y}\dot{\omega}_y-\omega_zI_{1x}\omega_x+\omega_xI_{1z}\omega_z-z_{12}F_{1x}+c_{21y}+C_{1y}=0 \\ -I_{1z}\dot{\omega}_z-\omega_xI_{1y}\omega_y+\omega_yI_{1x}\omega_x+c_{21z}+C_{1z}=0 \end{array}\right. \tag{A.6}$$

La première de ces trois équations ne faisant intervenir aucune composante d'effort inconnue, elle donne directement la dernière équation différentielle du mouvement du satellite.

> On trouvera le code de calcul *MAPLE* relatif à l'obtention de ces équations dans le fichier `dalembert2`

2.4 équations de Lagrange

A cause de la structure mécanique du système $\mathcal{S}$, on sait que l'ensemble de ses configurations admissibles est une sous-variété de dimension 9 de $\left(\mathrm{Is}^+(\mathcal{E})\right)^6$ – et donc, l'ensemble de ses configurations relatives par rapport à une configuration de référence est une sous-variété de dimension 9 de $\left(\mathrm{Is}^+(I\!E)\right)^6$.

En effet, les liaisons entre $\mathcal{S}_1$ et $\mathcal{S}_2$, entre $\mathcal{S}_2$ et $\mathcal{S}_3$ et entre $\mathcal{S}_3$ et $\mathcal{S}_4$ ont toutes un degré de contrainte égal à 5 puisqu'elles n'autorisent que le mouvenment autour ou le long d'un axe. En vertu du théorème (1.41) du paragraphe (§1.2.3), la structure de $\mathcal{S}$ étant arborescente, l'ensemble des configurations est bien une sous-variété de dimension $(6 \times 6) - (3 \times 5) = 9$.

Il est intuitif que les 9 paramètres q_1, q_2, q_3, q_4, q_5, q_6, q_7, q_8, et q_9 que l'on a choisis précédemment forment une carte de cette sous-variété, tout au moins sur le produit cartésien du domaine où $(q_1, q_2, q_3, q_4, q_5, q_6)$ forme une carte de la variété de configurations de $\mathcal{S}_1$ avec les variétés de configurations relatives de $\mathcal{S}_2$, $\mathcal{S}_3$ et de $\mathcal{S}_4$ par rapport respectivement à $\mathcal{S}_1$, à $\mathcal{S}_3$ et à $\mathcal{S}_4$.

Pratiquement, étant donné que les angles de Bryant ne forment pas une carte de toutes les rotations de l'espace mais seulement des rotations dont l'axe n'est pas parallèle à l'axe Ox du repère considéré, le domaine de validité des équations que l'on va développer sera ainsi restreint.

Une fois cette restriction assumée, il n'est par contre pas nécessaire d'ecrire des équations de Lagrange avec multiplicateurs puisque puisqu'on dispose du bon nombre de paramètres.

Pour écrire les neuf équations différentielles du mouvement (1.3.3.i), il est donc nécessaire de calculer en fonction de ces paramètres,

- d'une part l'énergie cinétique totale du système $K(\mathcal{S})$,
- d'autre part les neuf composantes généralisées des efforts connus Q_i.

Les neufs équations s'obtiendront alors en répétant pour chacun des paramètres q_i $(i = 1, 2, \ldots, 9)$ le même procédé :

i. différentier l'énergie cinétique par rapport à $\dot{q}_i$, la dérivée par rapport au temps du paramètre pour obtenir
$$\frac{\partial K}{\partial \dot{q}_i}$$

ii. dériver par rapport au temps l'expression obtenue à l'étape précédente pour avoir
$$\frac{d}{dt}\left(\frac{\partial K}{\partial \dot{q}_i}\right)$$

iii. différentier l'énergie cinétique par rapport au paramètre, ce qui donne
$$\frac{\partial K}{\partial q_i}$$

iv. former l'équation :
$$\frac{d}{dt}\left(\frac{\partial K}{\partial \dot{q}_i}\right) - \frac{\partial K}{\partial q_i} = Q_i$$

On calcule l'énergie cinétique totale du système $\mathcal{S}$ à l'aide du théorème de König en développant la formule suivante :
$$\frac{1}{2}\left(\mu_1 (v(G_1^t))^2 + \omega_1.(\mathcal{I}\omega_1) + \mu_4 (v(G_4^t))^2\right)$$

Ce qui donne comme expression dans R_1^t – et donc dans tout autre repère orthonormé direct :

$$\begin{aligned}
K(\mathcal{S}) \quad = \quad & \tfrac{1}{2}\mu_1\left({v_x}^2+{v_y}^2+{v_z}^2\right)+\tfrac{1}{2}\left(I_{1x}{\omega_x}^2+I_{1y}{\omega_y}^2+I_{1z}{\omega_z}^2\right) \\
& +\tfrac{1}{2}\mu_4((v_x+\cos(q_9)\dot{q}_9 z_{34}+\omega_y(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34}) \\
& \qquad -\omega_z(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34}))^2 \\
& \qquad +(v_y-\sin(q_7)\dot{q}_7 q_8+\cos(q_7)\dot{q}_8-\cos(q_7)\dot{q}_7\cos(q_9)z_{34}+\sin(q_7)\sin(q_9)\dot{q}_9 z_{34} \\
& \qquad +\omega_z\sin(q_9)z_{34}-\omega_x(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34}))^2 \\
& \qquad +(v_z+\cos(q_7)\dot{q}_7 q_8+\sin(q_7)\dot{q}_8-\sin(q_7)\dot{q}_7\cos(q_9)z_{34}-\cos(q_7)\sin(q_9)\dot{q}_9 z_{34} \\
& \qquad +\omega_x(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})-\omega_y\sin(q_9)z_{34})^2\big)
\end{aligned} \tag{L.1}$$

Pour calculer les termes de gauche des six premières équations, il est nécessaire d'avoir une expression de l'énergie cinétique faisant intervenir explicitement q_1, q_2, q_3, q_4, q_5, et q_6.
Pour cela, on substitue dans la formule précédente v_x, v_y et v_z et ω_x, ω_y et ω_z par leurs expressions en fonction de q_1, q_2, q_3, q_4, q_5, et de q_6, provenant des systèmes (I.3) et (I.4) – cf. (§2.1).
Il en découle, pour $K(\mathcal{S})$:

$$\begin{aligned}
& \tfrac{1}{2}\mu_1(((-\cos(q_1)\sin(q_2)\cos(q_3)+\sin(q_1)\sin(q_3))\dot{q}_6+(\sin(q_1)\sin(q_2)\cos(q_3)+\cos(q_1)\sin(q_3))\dot{q}_5 \\
& \qquad +\cos(q_2)\dot{q}_4\cos(q_3))^2 \\
& \quad +((\cos(q_1)\sin(q_2)\sin(q_3)+\sin(q_1)\cos(q_3))\dot{q}_6+(-\sin(q_1)\sin(q_2)\sin(q_3)+\cos(q_1)\cos(q_3))\dot{q}_5 \\
& \qquad -\sin(q_3)\cos(q_2)\dot{q}_4)^2 \\
& \quad +(-\cos(q_2)\dot{q}_5\sin(q_1)+\cos(q_2)\cos(q_1)\dot{q}_6+\dot{q}_4\sin(q_2))^2\big) \\
& +\tfrac{1}{2}I_{1x}(\cos(q_2)\cos(q_3)\dot{q}_1+\sin(q_3)\dot{q}_2)^2+\tfrac{1}{2}I_{1y}(-\cos(q_2)\sin(q_3)\dot{q}_1+\cos(q_3)\dot{q}_2)^2+\tfrac{1}{2}I_{1z}(\sin(q_2)\dot{q}_1+\dot{q}_3)^2 \\
& +\tfrac{1}{2}\mu_4(((-\cos(q_1)\sin(q_2)\cos(q_3)+\sin(q_1)\sin(q_3))\dot{q}_6+(\sin(q_1)\sin(q_2)\cos(q_3)+\cos(q_1)\sin(q_3))\dot{q}_5 \\
& \qquad +\cos(q_2)\dot{q}_4\cos(q_3)+\cos(q_9)\dot{q}_9 z_{34} \\
& \qquad +(-\cos(q_2)\sin(q_3)\dot{q}_1+\cos(q_3)\dot{q}_2)(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34}) \\
& \qquad -(\sin(q_2)\dot{q}_1+\dot{q}_3)(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34}))^2 \\
& \quad +((\cos(q_1)\sin(q_2)\sin(q_3)+\sin(q_1)\cos(q_3))\dot{q}_6+(-\sin(q_1)\sin(q_2)\sin(q_3)+\cos(q_1)\cos(q_3))\dot{q}_5 \\
& \qquad -\sin(q_3)\cos(q_2)\dot{q}_4-\sin(q_7)\dot{q}_7 q_8+\cos(q_7)\dot{q}_8-\cos(q_7)\dot{q}_7\cos(q_9)z_{34}+\sin(q_7)\sin(q_9)\dot{q}_9 z_{34} \\
& \qquad +(\sin(q_2)\dot{q}_1+\dot{q}_3)\sin(q_9)z_{34}-(\cos(q_2)\cos(q_3)\dot{q}_1+\sin(q_3)\dot{q}_2)(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34}))^2 \\
& \quad +(-\cos(q_2)\dot{q}_5\sin(q_1)+\cos(q_2)\cos(q_1)\dot{q}_6+\dot{q}_4\sin(q_2)+\cos(q_7)\dot{q}_7 q_8+\sin(q_7)\dot{q}_8-\sin(q_7)\dot{q}_7\cos(q_9)z_{34} \\
& \qquad -\cos(q_7)\sin(q_9)\dot{q}_9 z_{34}+(\cos(q_2)\cos(q_3)\dot{q}_1+\sin(q_3)\dot{q}_2)(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34}) \\
& \qquad -(-\cos(q_2)\sin(q_3)\dot{q}_1+\cos(q_3)\dot{q}_2)\sin(q_9)z_{34})^2\big)
\end{aligned} \tag{L.2}$$

Il faut ensuite calculer les neuf composantes généralisées des forces.
Par définition, la composante généralisée des forces relative au paramètre q_i est le coefficient de $\dot{q}_i$ dans l'expression de la puissance de toutes les forces connues développée au cours de n'importe quel mouvement compatible avec la structure mécanique du système $\mathcal{S}$.
Cette puissance est donc la somme globale :

- des produits scalaires des forces connues qui s'appliquent en des points du système par la vitesse de ces points dans le mouvement considéré,

- des produits scalaires des couples connus exercés sur les corps du système par la vitesse instantanée de rotation de ces corps.

Dans notre exemple, en prenant le mouvement compatible le plus général – en n'imposant aucune contrainte sur les $\dot{q}_i$ – cette puissance s'écrit, avec les notations introduites au paragraphe précédent :

$$\begin{aligned} &F_1.v(G_1^t) + C_1.\omega_1 - C_{12}\overrightarrow{i_1}.\omega_1 - F_{23}\overrightarrow{j_2}.v(M_{21}^t) + C_{12}\overrightarrow{i_2}.\omega_2 \\ &+F_{23}\overrightarrow{j_3}.v(M_{34}^t) - C_{34}\overrightarrow{j_3}.\omega_3 + F_4.v(G_4^t) + C_{34}\overrightarrow{j_4}.\omega_4 \end{aligned}$$

où ω_2, ω_3 et ω_4 sont respectivement, à chaque instant t les vecteurs de rotation instantanée des repères attachés aux corps $\mathcal{S}_2$, $\mathcal{S}_3$ et $\mathcal{S}_4$ par rapport au repère R.

Ainsi elle a comme expression dans R_1^t, ou dans tout autre repère orthonormé direct :

$$\begin{aligned} &F_{1x}v_x+F_{1y}v_y+F_{1z}v_z+(C_{1x}-C_{12})\omega_x+C_{1y}\omega_y+C_{1z}\omega_z-\cos(q_7)F_{23}(v_y-\omega_x z_{12})-\sin(q_7)F_{23}v_z+C_{12}(\omega_x+\dot{q}_7) \\ &\quad+\cos(q_7)F_{23}(v_y-\sin(q_7)\dot{q}_7q_8+\cos(q_7)\dot{q}_8-\omega_x(z_{12}+\sin(q_7)q_8)) \\ &\quad+\sin(q_7)F_{23}(v_z+\cos(q_7)\dot{q}_7q_8+\sin(q_7)\dot{q}_8+\omega_x\cos(q_7)q_8)+C_{34}\dot{q}_9 \\ &\quad+F_{4x}(v_x+\cos(q_9)\dot{q}_9z_{34}+\omega_y(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34})-\omega_z(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})) \\ &\quad+F_{4y}(v_y-\sin(q_7)\dot{q}_7q_8+\cos(q_7)\dot{q}_8-\cos(q_7)\dot{q}_7\cos(q_9)z_{34}+\sin(q_7)\sin(q_9)\dot{q}_9z_{34}+\omega_z\sin(q_9)z_{34} \\ &\qquad\quad-\omega_x(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34})) \\ &\quad+F_{4z}(v_z+\cos(q_7)\dot{q}_7q_8+\sin(q_7)\dot{q}_8-\sin(q_7)\dot{q}_7\cos(q_9)z_{34}-\cos(q_7)\sin(q_9)\dot{q}_9z_{34} \\ &\qquad\quad+\omega_x(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})-\omega_y\sin(q_9)z_{34}) \end{aligned} \tag{L.3}$$

ou, en faisant intervenir q_1, q_2, q_3, q_4, q_5, et q_6 à la place de v_x, v_y, v_z et de ω_x, ω_y et ω_z,

$$
\begin{aligned}
&F_{1x}((-\cos(q_1)\sin(q_2)\cos(q_3)+\sin(q_1)\sin(q_3))\dot{q}_6+(\sin(q_1)\sin(q_2)\cos(q_3)+\cos(q_1)\sin(q_3))\dot{q}_5+\cos(q_2)\dot{q}_4\cos(q_3))\\
&\quad+F_{1y}((\cos(q_1)\sin(q_2)\sin(q_3)+\sin(q_1)\cos(q_3))\dot{q}_6+(-\sin(q_1)\sin(q_2)\sin(q_3)+\cos(q_1)\cos(q_3))\dot{q}_5\\
&\qquad-\sin(q_3)\cos(q_2)\dot{q}_4)\\
&\quad+F_{1z}(-\cos(q_2)\dot{q}_5\sin(q_1)+\cos(q_2)\cos(q_1)\dot{q}_6+\dot{q}_4\sin(q_2))+(C_{1x}-C_{12})(\cos(q_2)\cos(q_3)\dot{q}_1+\sin(q_3)\dot{q}_2)\\
&\quad+C_{1y}(-\cos(q_2)\sin(q_3)\dot{q}_1+\cos(q_3)\dot{q}_2)+C_{1z}(\sin(q_2)\dot{q}_1+\dot{q}_3)+C_{34}\dot{q}_9\\
&\quad-\cos(q_7)F_{23}\\
&\qquad((\cos(q_1)\sin(q_2)\sin(q_3)+\sin(q_1)\cos(q_3))\dot{q}_6+(-\sin(q_1)\sin(q_2)\sin(q_3)+\cos(q_1)\cos(q_3))\dot{q}_5\\
&\qquad\quad-\sin(q_3)\cos(q_2)\dot{q}_4-(\cos(q_2)\cos(q_3)\dot{q}_1+\sin(q_3)\dot{q}_2)z_{12})\\
&\quad-\sin(q_7)F_{23}\\
&\qquad(-\cos(q_2)\dot{q}_5\sin(q_1)+\cos(q_2)\cos(q_1)\dot{q}_6+\dot{q}_4\sin(q_2))+C_{12}(\cos(q_2)\cos(q_3)\dot{q}_1+\sin(q_3)\dot{q}_2+\dot{q}_7)\\
&\quad+\cos(q_7)F_{23}\\
&\qquad((\cos(q_1)\sin(q_2)\sin(q_3)+\sin(q_1)\cos(q_3))\dot{q}_6+(-\sin(q_1)\sin(q_2)\sin(q_3)+\cos(q_1)\cos(q_3))\dot{q}_5\\
&\qquad\quad-\sin(q_3)\cos(q_2)\dot{q}_4-\sin(q_7)\dot{q}_7q_8+\cos(q_7)\dot{q}_8-(\cos(q_2)\cos(q_3)\dot{q}_1+\sin(q_3)\dot{q}_2)(z_{12}+\sin(q_7)q_8))\\
&\quad+\sin(q_7)F_{23}\\
&\qquad(-\cos(q_2)\dot{q}_5\sin(q_1)+\cos(q_2)\cos(q_1)\dot{q}_6+\dot{q}_4\sin(q_2)+\cos(q_7)\dot{q}_7q_8+\sin(q_7)\dot{q}_8\\
&\qquad\quad+(\cos(q_2)\cos(q_3)\dot{q}_1+\sin(q_3)\dot{q}_2)\cos(q_7)q_8)\\
&\quad+F_{4x}\\
&\qquad((-\cos(q_1)\sin(q_2)\cos(q_3)+\sin(q_1)\sin(q_3))\dot{q}_6+(\sin(q_1)\sin(q_2)\cos(q_3)+\cos(q_1)\sin(q_3))\dot{q}_5\\
&\qquad\quad+\cos(q_2)\dot{q}_4\cos(q_3)+\cos(q_9)\dot{q}_9z_{34}+(-\cos(q_2)\sin(q_3)\dot{q}_1+\cos(q_3)\dot{q}_2)(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34})\\
&\qquad\quad-(\sin(q_2)\dot{q}_1+\dot{q}_3)(cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34}))\\
&\quad+F_{4y}\\
&\qquad((\cos(q_1)\sin(q_2)\sin(q_3)+\sin(q_1)\cos(q_3))\dot{q}_6+(-\sin(q_1)\sin(q_2)\sin(q_3)+\cos(q_1)\cos(q_3))\dot{q}_5\\
&\qquad\quad-\sin(q_3)\cos(q_2)\dot{q}_4-\sin(q_7)\dot{q}_7q_8+\cos(q_7)\dot{q}_8-\cos(q_7)\dot{q}_7\cos(q_9)z_{34}+\sin(q_7)\sin(q_9)\dot{q}_9z_{34}\\
&\qquad\quad+(\sin(q_2)\dot{q}_1+\dot{q}_3)\sin(q_9)z_{34}-(\cos(q_2)\cos(q_3)\dot{q}_1+\sin(q_3)\dot{q}_2)(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34}))\\
&\quad+F_{4z}\\
&\qquad(-\cos(q_2)\dot{q}_5\sin(q_1)+\cos(q_2)\cos(q_1)\dot{q}_6+\dot{q}_4\sin(q_2)+\cos(q_7)\dot{q}_7q_8+\sin(q_7)\dot{q}_8-\sin(q_7)\dot{q}_7\cos(q_9)z_{34}\\
&\qquad\quad-\cos(q_7)\sin(q_9)\dot{q}_9z_{34}+(\cos(q_2)\cos(q_3)\dot{q}_1+\sin(q_3)\dot{q}_2)(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})\\
&\qquad\quad-(-\cos(q_2)\sin(q_3)\dot{q}_1+\cos(q_3)\dot{q}_2)\sin(q_9)z_{34})
\end{aligned}
\tag{L.4}
$$

On est alors amené à choisir pour chaque calcul de composante, le mouvement compatible avec la structure mécanique du système qui rend l'expression de la puissance des forces la plus simple possible – simplement pour diminuer le temps de calcul puisqu'il est clair que les Q_i sont indépendants du mouvement compatible choisi pour les calculer.

Commençons par calculer Q_1. On va choisir comme mouvement compatible, celui qui est défini par :

$$\dot{q}_2 = \dot{q}_3 = \dot{q}_4 = \dot{q}_5 = \dot{q}_6 = \dot{\Theta}_{12} = \dot{Y}_{23} = \dot{\Theta}_{34} = 0$$

Par substitution dans (L.4), on obtient :

$$\begin{aligned} Q_1 \quad = \quad & C_{1x}\cos(q_2)\cos(q_3) - C_{1y}\cos(q_2)\sin(q_3) + C_{1z}\sin(q_2) \\ & + F_{4x}(-\cos(q_2)\sin(q_3)(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34}) - \sin(q_2)(\cos(q_7)q_8 - \sin(q_7)\cos(q_9)z_{34})) \\ & + F_{4y}(\sin(q_2)\sin(q_9)z_{34} - \cos(q_2)\cos(q_3)(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34})) \\ & + F_{4z}(\cos(q_2)\cos(q_3)(\cos(q_7)q_8 - \sin(q_7)\cos(q_9)z_{34}) + \cos(q_2)\sin(q_3)\sin(q_9)z_{34}) \end{aligned}$$

De la même manière, en choisissant des mouvements compatibles adéquats, il vient aisément :

- avec $\dot{q}_1 = \dot{q}_3 = \dot{q}_4 = \dot{q}_5 = \dot{q}_6 = \dot{\Theta}_{12} = \dot{Y}_{23} = \dot{\Theta}_{34} = 0$

$$\begin{aligned} Q_2 \quad = \quad & C_{1x}\sin(q_3) + C_{1y}\cos(q_3) + F_{4x}\cos(q_3)(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34}) \\ & - F_{4y}\sin(q_3)(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34}) \\ & + F_{4z}(\sin(q_3)(\cos(q_7)q_8 - \sin(q_7)\cos(q_9)z_{34}) - \cos(q_3)\sin(q_9)z_{34}) \end{aligned}$$

- avec $\dot{q}_1 = \dot{q}_2 = \dot{q}_4 = \dot{q}_5 = \dot{q}_6 = \dot{\Theta}_{12} = \dot{Y}_{23} = \dot{\Theta}_{34} = 0$

$$Q_3 \quad = \quad C_{1z} - F_{4x}(\cos(q_7)q_8 - \sin(q_7)\cos(q_9)z_{34}) + F_{4y}\sin(q_9)z_{34}$$

- avec $\dot{q}_1 = \dot{q}_2 = \dot{q}_3 = \dot{q}_5 = \dot{q}_6 = \dot{\Theta}_{12} = \dot{Y}_{23} = \dot{\Theta}_{34} = 0$

$$\begin{aligned} Q_4 \quad = \quad & F_{1x}\cos(q_2)\cos(q_3) - F_{1y}\sin(q_3)\cos(q_2) + F_{1z}\sin(q_2) \\ & + F_{4x}\cos(q_2)\cos(q_3) - F_{4y}\sin(q_3)\cos(q_2) + F_{4z}\sin(q_2) \end{aligned}$$

- avec $\dot{q}_1 = \dot{q}_2 = \dot{q}_3 = \dot{q}_4 = \dot{q}_6 = \dot{\Theta}_{12} = \dot{Y}_{23} = \dot{\Theta}_{34} = 0$

$$\begin{aligned} Q_5 \quad = \quad & F_{1x}(\sin(q_1)\sin(q_2)\cos(q_3) + \cos(q_1)\sin(q_3)) + F_{1y}(-\sin(q_1)\sin(q_2)\sin(q_3) + \cos(q_1)\cos(q_3)) \\ & - F_{1z}\cos(q_2)\sin(q_1) + F_{4x}(\sin(q_1)\sin(q_2)\cos(q_3) + \cos(q_1)\sin(q_3)) \\ & + F_{4y}(-\sin(q_1)\sin(q_2)\sin(q_3) + \cos(q_1)\cos(q_3)) - F_{4z}\cos(q_2)\sin(q_1) \end{aligned}$$

- avec $\dot{q}_1 = \dot{q}_2 = \dot{q}_3 = \dot{q}_4 = \dot{q}_5 = \dot{\Theta}_{12} = \dot{Y}_{23} = \dot{\Theta}_{34} = 0$

$$\begin{aligned} Q_6 \quad = \quad & F_{1x}(-\cos(q_1)\sin(q_2)\cos(q_3) + \sin(q_1)\sin(q_3)) + F_{1y}(\cos(q_1)\sin(q_2)\sin(q_3) + \sin(q_1)\cos(q_3)) \\ & + F_{1z}\cos(q_2)\cos(q_1) + F_{4x}(-\cos(q_1)\sin(q_2)\cos(q_3) + \sin(q_1)\sin(q_3)) \\ & + F_{4y}(\cos(q_1)\sin(q_2)\sin(q_3) + \sin(q_1)\cos(q_3)) + F_{4z}\cos(q_2)\cos(q_1) \end{aligned}$$

Et les composantes généralisées des forces correspondant aux paramètres Θ_{12}, Y_{23} et Θ_{34} s'obtiennent aussi de la même manière, à partir de l'expression plus simple (L.3), en utilisant des mouvements compatibles dans lesquels, respectivement, seules $\dot{\Theta}_{12}$, $\dot{Y}_{23}$ et $\dot{\Theta}_{34}$ sont non nulles. Ainsi :

$$Q_7 = C_{12} + F_{4y}\left(-\sin(q_7)q_8 - \cos(q_7)\cos(q_9)z_{34}\right) + F_{4z}\left(\cos(q_7)q_8 - \sin(q_7)\cos(q_9)z_{34}\right)$$

$$Q_8 = F_{23} + F_{4y}\cos(q_7) + F_{4z}\sin(q_7)$$

$$Q_9 = C_{34} + F_{4x}\cos(q_9)z_{34} + F_{4y}\sin(q_7)\sin(q_9)z_{34} - F_{4z}\cos(q_7)\sin(q_9)z_{34}$$

Il reste maintenant,pour avoir les équations différentielles du mouvement à calculer les neufs expressions, pour $i = 1, \ldots, 9$

$$\frac{d}{dt}\left(\frac{\partial K}{\partial \dot{q}_i}\right) - \frac{\partial K}{\partial q_i}$$

en répétant, pour chaque équation, les quatre étapes du procédé décrit plus haut.

Effectuons ces calculs pour $i = 8$ – c'est pour $i = 8$ que l'on obtient les équations les moins volumineuses, pour certaines valeurs de i, on obtient des expressions qui dépasseraient, même après simplification, le volume d'une page.

La première étape consiste à établir l'expression de la dérivée partielle de l'énergie cinétique par rapport à $\dot{q}_8$, soit :

$$\frac{\partial K}{\partial \dot{q}_8} = \mu_4(\sin(q_7)v_z + \cos(q_7)v_y + \cos(q_7)\omega_z\sin(q_9)z_{34} - \sin(q_7)\sin(q_9)z_{34}\omega_y - (\cos(q_9)z_{34} + z_{12}\cos(q_7))\omega_x - \dot{q}_7\cos(q_9)z_{34} + \dot{q}_8)$$

Ensuite, on dérive cette expression par rapport au temps.

Puis, on calcule la dérivée partielle de l'énergie cinétique par rapport à q_8 soit :

$$\begin{aligned}
\frac{\partial K}{\partial q_8} \;=\; & \mu_4(-v_x\omega_z\cos(q_7) + v_x\omega_y\sin(q_7) + (v_z\cos(q_7) - v_y\sin(q_7))\omega_x - \cos(q_7)\sin(q_9)z_{34}\omega_y\omega_x \\
& - \big(2\cos(q_9)z_{34}\cos(q_7)^2 - \cos(q_9)z_{34} + z_{12}\cos(q_7) + 2\sin(q_7)q_8\cos(q_7)\big)\omega_y\omega_z - \sin(q_7)\omega_z\sin(q_9)z_{34}\omega_x \\
& + \big(z_{12}\sin(q_7) + q_8 - \cos(q_7)^2 q_8 + \sin(q_7)\cos(q_9)z_{34}\cos(q_7)\big){\omega_y}^2 \\
& - \big(\sin(q_7)\cos(q_9)z_{34}\cos(q_7) - \cos(q_7)^2 q_8\big){\omega_z}^2 + (z_{12}\sin(q_7) + q_8){\omega_x}^2 \\
& + (v_z\cos(q_7) - v_y\sin(q_7) - \omega_z\sin(q_9)z_{34}\sin(q_7) - \sin(q_9)z_{34}\omega_y\cos(q_7) + (z_{12}\sin(q_7) + 2q_8)\omega_x)\dot{q}_7 \\
& - (\omega_x\sin(q_9)z_{34} - \sin(q_7)\cos(q_9)z_{34}\omega_y + \cos(q_7)\omega_z\cos(q_9)z_{34})\dot{q}_9 - \dot{q}_7\sin(q_9)\dot{q}_9 z_{34} + \dot{q}_7^2 q_8\big)
\end{aligned}$$

Et enfin, on écrit l'équation du mouvement :

$$\begin{aligned}
& F_{23} + F_{4y}\cos(q_7) + F_{4z}\sin(q_7) = \\
& \mu_4\big(\big(\sin(q_7)\cos(q_9)z_{34}\cos(q_7) - \cos(q_7)^2 q_8\big){\omega_z}^2 + v_x\omega_z\cos(q_7) - v_x\omega_y\sin(q_7) - (v_z\cos(q_7) - v_y\sin(q_7))\omega_x \\
& \quad - \big(z_{12}\sin(q_7) + q_8 - \cos(q_7)^2 q_8 + \sin(q_7)\cos(q_9)z_{34}\cos(q_7)\big){\omega_y}^2 - (z_{12}\sin(q_7) + q_8){\omega_x}^2 \\
& \quad + \sin(q_7)\omega_z\sin(q_9)z_{34}\omega_x + \cos(q_7)\sin(q_9)z_{34}\omega_y\omega_x \\
& \quad + \big(2\cos(q_9)z_{34}\cos(q_7)^2 - \cos(q_9)z_{34} + z_{12}\cos(q_7) + 2\sin(q_7)q_8\cos(q_7)\big)\omega_y\omega_z + 2\dot{q}_7\sin(q_9)\dot{q}_9 z_{34} - 2\dot{q}_7\omega_x q_8 \\
& \quad - (\cos(q_9)z_{34} + z_{12}\cos(q_7))\dot{\omega}_x - \sin(q_7)\sin(q_9)z_{34}\dot{\omega}_y + \cos(q_7)\dot{\omega}_z\sin(q_9)z_{34} \\
& \quad + (2\omega_x\sin(q_9)z_{34} - 2\sin(q_7)\cos(q_9)z_{34}\omega_y + 2\cos(q_7)\omega_z\cos(q_9)z_{34})\dot{q}_9 \\
& \quad + \cos(q_7)\dot{v}_y + \sin(q_7)\dot{v}_z - \dot{q}_7^2 q_8 - \ddot{q}_7\cos(q_9)z_{34} + \ddot{q}_8\big)
\end{aligned} \tag{L.5}$$

L'examen du code de calcul *MAPLE* correspondant à l'écriture des équations de Lagrange – qui se trouve dans le fichier `lagrange` – amène plusieurs remarques.

En plus des fonctionnalités de manipulation matricielle et vectorielle dont on a déjà parlé et qui servent ici surtout au calcul de l'énergie cinétique et de la puissance généralisée des efforts connus, on note l'utilisation massive de la fonction de dérivation formelle `diff` conjointement avec la fonction de substitution d'expression `subs`.

On a vu déjà que pour indiquer à *MAPLE* qu'une expression `q` non affectée dépend d'une variable `t`, il suffit de l'écrire `q(t)`.

Par contre, syntaxiquement, *MAPLE* n'accepte pas de différentier une expression par rapport à une variable qui elle-même dépend d'une autre variable :

```
> diff(eq2,q1(t))
Error, wrong number (or type) of parameters in function diff;
```

Ainsi, auparavant, on avait défini `varslect` – qui permet de réaliser les substitutions de la forme `q(t)` $\longrightarrow$ `q` et `diff(q(t),t)` $\longrightarrow$ `qp` – simplement pour obtenir à l'ecran des expressions plus lisibles.

Ici, `varslect` est indispensable aux calculs. On a été amené de plus à définir `varstq` – qui réalise les substitutions de la forme `q` $\longrightarrow$ `q(t)` – et `varstqp` – qui réalise les substitutions de la forme `qp` $\longrightarrow$ `diff(q(t),t)` – et à faire des substitutions successives dans les deux sens pour pouvoir utiliser `diff`.

Mais, dans la mesure où, en *MAPLE* les substitutions sont "syntaxiques", c'est à dire ne sont réalisées que si l'objet que l'on veut modifier se trouve explicitement – au niveau de la représentation interne – dans l'expression considérée, ces substitutions ne sont pas coûteuses en temps de calcul.

Par contre, l'utilisation de ce mécanisme de substitution syntaxique nécessite une bonne connaissance de la représentation interne des objets *MAPLE* comme le montre l'exemple suivant :

```
> exp:=a+b+c;
                                  exp := a + b + c
> subs(a+b=d,exp);
                                     a + b + c
> exp:=(a+b)*c;
                                  exp := (a + b) c
> subs(a+b=d,exp);
                                        d c
> exp:=(a+b)+c;
                                  exp := a + b + c
> subs(a+b=d,exp);
                                     a + b + c
```

les résultats surprenants proviennent de ce que même introduite par `(a+b)+c` l'expression `a + b + c` est représentée comme $+(a,b,c)$ et non comme $+(+(a,b),c)$ tandis que `(a + b) c` est représentée comme $*(+(a,b),c)$ et que la substitution se traduit par $+(a,b) \longrightarrow d$.

Une deuxième remarque sur le fichier `lagrange` concerne le temps de calcul nécessaire à son éxécution, beaucoup plus court – de l'ordre de 315 secondes CPU – que les temps nécessaires pour établir les équations avec les méthodes précédentes.

Ce gain est, cette fois, bien dû à la méthode employée dans la mesure où celle-ci utilise massivement des dérivations plutôt que de nombreux produits de matrices qui induisent des produits de grosses expressions au niveau des coefficients.

En revanche la forme des expressions obtenues et leurs tailles interdisent toute tentative d'interprétation mécanique des équations obtenues et ceci est aussi intrinsèque à la méthode.

Enfin, concernant l'écriture du code en lui-même, ou tout au moins de la partie de code qui calcule les Q_i et les $\frac{d}{dt}\left(\frac{\partial K}{\partial \dot{q}_i}\right) - \frac{\partial K}{\partial q_i}$, on voit qu'il consiste en une répétition des mêmes instructions pour chacun des paramètres q_i.

Il est donc parfaitement adapté à une écriture à l'aide d'instructions controlées par une boucle. Le langage de programmation de *MAPLE* possède bien entendu les structures de contrôle nécessaires ainsi qu'un mécanisme de création de nom de variables intéressant. Ainsi cette partie du code du fichier `lagrange` peut se réécrire par :

```
for i from 1 to 9 do
  Q.i:=collect(
         coeff(
           collect(
             subs({seq(qp.j=0,j=1..i-1),seq(qp.j=0,j=i+1..9)},tmp62),
             qp.i),
           qp.i),
         [f4X,f4Y,f4Z])
od;

for i from 1 to 9 do
  diff(eq2,qp.i):
  subs(varstq,"):
  subs(tmp17 union convert(tmp21,set),"):
  simplify(subs(varstqp,"),trig):
  dK2dqp.i.dt:=subs(varslect,diff(",t)):
  diff(eq2,q.i):
  subs(varstq,"):
  subs(tmp17 union convert(tmp21,set),"):
  simplify(tmp108,trig):
  K.i:=rlms(
         clct(
           simplify(dK2dqp.i.dt-dKdq.i,trig),
            [qpp7,qpp8,qpp9],
            [omXp,omYp,omZp,vXp,vYp,vZp,
             qp1,qp2,qp3,qp4,qp5,qp6,qp7,qp8,qp9],
            [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4],factor));
od:
```

Ainsi, la méthode de Lagrange, décrite par un algorithme indépendant de la structure du système mécanique et de la nature des corps qui le composent permettrait facilement de programmer un générateur automatique du code de mise en équation.

2.5 équations de Hamilton

Pour établir les dix-huit équations différentielles du premier ordre du mouvement du satellite – on rapelle que les équations de Hamilton sont toutes des équations du premier ordre sur l'espace des phases hamiltonien du système, i.e. le fibré cotangent à sa variété de configurations – on est

amené à effectuer les deux étapes suivantes :

i. calculer le lagrangien du système,

ii. puis calculer le hamiltonien du système.

Comme on va le voir, aucun de ces deux calculs ne peut facilement être réalisé de manière automatique.

On ne peut déterminer le lagrangien d'un système mécanique que si les efforts connus qui s'exercent sur ce système dérivent :

- soit d'un potentiel, c'est-à-dire qu'il existe une fonction $V(q)$ à valeurs réelles telle que, pour $i = 1, 2, \ldots, n$,
$$Q_i = -\frac{\partial V(q)}{\partial q_i}$$
où les Q_i sont les composantes généralisées des forces connues que l'on a introduites au paragraphe précédent,
- soit d'un potentiel lagrangien, c'est-à-dire qu'il existe une fonction $V(q, \dot{q})$ à valeurs réelles telle que, pour $i = 1, 2, \ldots, n$,
$$Q_i = \frac{d}{dt}\left(\frac{\partial V(q,\dot{q})}{\dot{q}}_i\right) - \frac{\partial V(q)}{\partial q_i}$$

Dans ces deux cas, on voit facilement que si l'on pose

$$L(q,\dot{q}) = K(q,\dot{q}) - V(q) \qquad (\text{resp.} \quad L(q,\dot{q}) = K(q,\dot{q}) - V(q,\dot{q}))$$

les équations différentielles du mouvement de Lagrange,

$$\frac{d}{dt}\left(\frac{\partial K}{\partial \dot{q}_i}\right) - \frac{\partial K}{\partial q_i} = Q_i$$

s'écrivent :

$$\frac{d}{dt}\left(\frac{\partial K}{\partial \dot{q}_i}\right) - \frac{\partial K}{\partial q_i} = -\frac{\partial V(q)}{\partial q_i} \qquad (\text{resp.} \quad \frac{d}{dt}\left(\frac{\partial K}{\partial \dot{q}_i}\right) - \frac{\partial K}{\partial q_i} = \frac{d}{dt}\left(\frac{\partial V(q,\dot{q})}{q_i}\right) - \frac{\partial V(q)}{\partial q_i})$$

$$\text{soit} \qquad \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = 0 \tag{H.1}$$

On dit alors que L est le lagrangien du système considéré.

Si l'on sait que les efforts s'exerçant sur le système étudié dérivent effectivement d'un potentiel – ou d'un potentiel lagrangien – et que l'on connaît ce potentiel, l'écriture du lagrangien ne pose aucun problème.

Dans le cas contraire, qui est bien sûr celui dans lequel rentre une approche automatique de la génération des équations de la dynamique, il faut "récupérer" le potentiel voulu à partir des composantes généralisées des efforts donnés.

Toutefois, ce potentiel étant additif – relativement aux forces –, on peut essayer de le déterminer terme à terme en examinant les différentes forces successivement.

Ainsi, sur l'exemple du satellite, on obtiendrait assez facilement, si l'on étudiait son mouvement dans le repère $\mathcal{R}_1$, le potentiel dont dérivent les efforts connus, tant extérieurs que ceux exercés par les mécanismes ressorts-amortisseurs – à condition de se donner explicitement $C_{12}(q,\dot{q})$, $F_{23}(q,\dot{q})$ et $C_{34}(q,\dot{q})$, et que ces efforts dérivent effectivement d'un potentiel !

En supposant, pour simplifier que C_{12}, F_{23} et C_{34} sont constantes, on trouverait alors $V(q)$ en utilisant l'algorithme suivant :

- $V(q) \longleftarrow \int_0^{q_7} Q_7 dq_7$

- pour $i = 8, 9$
  - $V(q) \longleftarrow V(q) + \displaystyle\int_0^{q_i} \left(Q_i - \frac{\partial V}{\partial q_i} \right) dq_i$

En effet, si l'on étudie le mouvement relativement au repère $\mathcal{R}_1$, des calculs similaires à ceux du paragraphe précédent – écriture de l'expression de la puissance généralisée des efforts connus – donnent :

- $Q_7 = C_{12} + F_{4y} \left(-\sin(q_7) q_8 - \cos(q_7)\cos(q_9) z_{34} \right) + F_{4z} \left(\cos(q_7) q_8 - \sin(q_7)\cos(q_9) z_{34} \right)$
- $Q_8 = F_{23} + F_{4y}\cos(q_7) + F_{4z}\sin(q_7)$
- $Q_9 = C_{34} + F_{4x}\cos(q_9) z_{34} + F_{4y}\sin(q_7)\sin(q_9) z_{34} - F_{4z}\cos(q_7)\sin(q_9) z_{34}$

et l'algorithme fournit alors le potentiel $V(q)$

$$\begin{aligned} V(q) \quad = \quad & \sin(q_9) F_{4x} z_{34} + \left(\cos(q_7) q_8 - \sin(q_7)\cos(q_9) z_{34} \right) F_{4y} \\ & + \left(\sin(q_7) q_8 + \cos(q_7)\cos(q_9) z_{34} \right) F_{4z} + C_{12} q_7 + F_{23} q_8 + C_{34} q_9 \end{aligned} \tag{H.2}$$

> *MAPLE* possède une bibliothèque puissante d'intégration formelle qui permet de calculer, lorsqu'elles existent les expressions des intégrales définies et des primitives. C'est la fonction `int` de cette bibliothèque que l'on utilise pour réaliser les calculs précédents.

Par contre, si l'on étudie le mouvement du satellite relativement à R, comme on l'a fait jusqu'à présent, il est immédiat de voir que les Q_i ne dérivent pas d'un simple potentiel puisqu'ils ne satisfont pas les trente-six conditions nécessaires :

$$\frac{\partial Q_i}{\partial q_j} = \frac{\partial Q_j}{\partial q_i}$$

et le problème qui consiste à trouver un algorithme pour obtenir, lorsqu'il existe, un potentiel lagrangien dont les Q_i dérivent est beaucoup plus difficile.

Même en supposant que l'on sache effectuer cette première étape, – le calcul du lagrangien – pour une certaine classe de systèmes mécaniques, il reste à calculer, pour établir les équations de la dynamique, le hamiltonien du système :

$$H(p, q) = \sum_i p_i \dot{q}_i - L(q, \dot{q})$$

Cette expression fait intervenir les *vitesses généralisées* p_i pour $i = 1, \ldots, 9$, qui forment une carte particulière de l'espace cotangent à la variété de configurations du système mécanique et sont définies comme :

$$\text{pour} \quad i = 1, \ldots, 9 \qquad p_i = \frac{\partial K(q, \dot{q})}{\partial \dot{q}_i} \tag{H.3}$$

Les équations différentielles du mouvement viennent facilement à partir des équations de Lagrange sous la forme (H.1) et de la différentielle de $H(p, q)$:

$$dH(p, q) = \sum_i \dot{q}_i dp_i - \sum_i \frac{\partial L(q, \dot{q})}{\partial q_i} dq_i$$

et s'écrivent :

$$\dot{q}_i = \frac{\partial H(p, q)}{\partial p_i} \quad \text{et} \quad \dot{p}_i = -\frac{\partial H(p, q)}{\partial q_i} \quad \left(\text{resp.} \quad \dot{p}_i = -\frac{\partial H(p, q)}{\partial q_i} + \frac{d}{dt} \left(\frac{\partial V(q, \dot{q})}{\partial \dot{q}_i} \right) \right)$$

selon que les efforts connus dépendent d'un simple potentiel ou d'un potentiel lagrangien.

mouvement relatif à $\mathcal{R}_1$ Si l'on étudie le mouvement du satellite relativement au repère $\mathcal{R}_1$, on arrive après un volume raisonnable de calcul à établir les équations du mouvement.
En effet, dans ce cas, les p_i sont définis par :

$$\begin{pmatrix} p_7 \\ p_8 \\ p_9 \end{pmatrix} = \mu_4 \begin{pmatrix} {q_8}^2 + {z_{34}}^2 \cos(q_9)^2 & -\cos(q_9) z_{34} & -\sin(q_9) z_{34} q_8 \\ -\cos(q_9) z_{34} & 1 & 0 \\ -\sin(q_9) z_{34} q_8 & 0 & {z_{34}}^2 \end{pmatrix} \begin{pmatrix} \dot{q}_7 \\ \dot{q}_8 \\ \dot{q}_9 \end{pmatrix}$$

ce système linéaire ayant pour déterminant :

$$\mu_4{}^3 {q_8}^2 \cos(q_9)^2 {z_{34}}^2$$

on obtient, à condition que ni q_8 ni $\cos(q_9)$ ne soit nul, les expressions de $\dot{q}_7$, de $\dot{q}_8$, et de $\dot{q}_9$ en fonction de (p_7, p_8, p_9) :

$$\dot{q}_7 = \frac{\cos(q_9) {z_{34}}^2 p_8 + \sin(q_9) q_8 p_9 + p_7 z_{34}}{\cos(q_9)^2 z_{34} \mu_4 {q_8}^2}$$

$$\dot{q}_8 = \frac{\cos(q_9) {z_{34}}^2 p_8 + \sin(q_9) q_8 p_9 + p_7 z_{34} + p_8 \cos(q_9) {q_8}^2}{\cos(q_9) {q_8}^2 \mu_4}$$

$$\dot{q}_9 = \frac{\cos(q_9) {z_{34}}^2 p_8 \sin(q_9) + q_8 p_9 + p_7 \sin(q_9) z_{34}}{\mu_4 q_8 {z_{34}}^2 \cos(q_9)^2}$$

On peut alors écrire successivement les expressions du lagrangien et du hamiltonien en fonction de (q_7, q_8, q_9) et de (p_7, p_8, p_9) :

$$\begin{aligned} H(p,q) = \frac{1}{2\cos(q_9)^2 \mu_4 {q_8}^2 {z_{34}}^2} \big(& {p_7}^2 {z_{34}}^2 + 2 C_{34} q_9 \mu_4 {q_8}^2 {z_{34}}^2 \cos(q_9)^2 + 2 F_{4y} \mu_4 {q_8}^3 {z_{34}}^2 \cos(q_7) \cos(q_9)^2 \\ & + 2 C_{12} q_7 \mu_4 {q_8}^2 {z_{34}}^2 \cos(q_9)^2 + {p_8}^2 {z_{34}}^2 {q_8}^2 \cos(q_9)^2 + 2 \sin(q_9) p_7 z_{34} p_9 q_8 \\ & + 2 \sin(q_9) \cos(q_9) {z_{34}}^2 p_8 p_9 q_8 + 2 F_{4z} \mu_4 {q_8}^2 {z_{34}}^3 \cos(q_7) \cos(q_9)^3 + {p_9}^2 {q_8}^2 \\ & + {p_8}^2 {z_{34}}^4 \cos(q_9)^2 + 2 F_{4z} \mu_4 {q_8}^3 {z_{34}}^2 \sin(q_7) \cos(q_9)^2 + 2 F_{23} {q_8}^3 \mu_4 {z_{34}}^2 \cos(q_9)^2 \\ & - 2 F_{4y} \mu_4 {q_8}^2 {z_{34}}^3 \sin(q_7) \cos(q_9)^3 + 2 p_7 {z_{34}}^3 \cos(q_9) p_8 \\ & + 2 F_{4x} {z_{34}}^3 \mu_4 {q_8}^2 \sin(q_9) \cos(q_9)^2 \big) \end{aligned}$$

et finalement établir les six équations différentielles du premier ordre du mouvement.
Les trois premières de ces équations proviennent de :

$$\dot{p}_i = \frac{\partial H(p,q)}{\partial q_i}$$

et s'écrivent :

$$\begin{cases} \dot{p}_7 = \dfrac{p_7 z_{34} + \cos(q_9) {z_{34}}^2 p_8 + \sin(q_9) q_8 p_9}{\mu_4 {q_8}^2 z_{34} \cos(q_9)^2} \\ \dot{p}_8 = \dfrac{p_7 z_{34} + \cos(q_9) {z_{34}}^2 p_8 + \sin(q_9) q_8 p_9 + p_8 {q_8}^2 \cos(q_9)}{\mu_4 {q_8}^2 \cos(q_9)} \\ \dot{p}_9 = \dfrac{\sin(q_9) p_7 z_{34} + \sin(q_9) \cos(q_9) {z_{34}}^2 p_8 + p_9 q_8}{\mu_4 q_8 {z_{34}}^2 \cos(q_9)^2} \end{cases} \tag{H.4}$$

Les trois autres équations qui proviennent de :

$$\dot{q}_i = \frac{\partial H(p,q)}{\partial p_i}$$

s'écrivent :

$$\left\{\begin{array}{rcl}
\dot{q}_7 & = & F_{4y}\sin(q_7)q_8 - C_{12} + F_{4z}\sin(q_7)\cos(q_9)z_{34} - F_{4z}\cos(q_7)q_8 + F_{4y}\cos(q_7)\cos(q_9)z_{34} \\
\dot{q}_8 & = & \frac{1}{\cos(q_9)^2\mu_4 {q_8}^3 z_{34}} \quad \big(F_{4y}\mu_4 {q_8}^3 z_{34}\cos(q_7)\cos(q_9)^2 + 2p_7 {z_{34}}^2\cos(q_9)p_8 + \sin(q_9)p_7 p_9 q_8 \\
 & & \quad + \sin(q_9)\cos(q_9)z_{34}p_8 p_9 q_8 + {p_8}^2 {z_{34}}^3\cos(q_9)^2 \\
 & & \quad - F_{4z}\mu_4 {q_8}^3 z_{34}\sin(q_7)\cos(q_9)^2 + {p_7}^2 z_{34} - F_{23}{q_8}^3\mu_4 z_{34}\cos(q_9)^2\big) \\
\dot{q}_9 & = & \frac{1}{\cos(q_9)^3\mu_4 {q_8}^2 {z_{34}}^2} \quad \big(\cos(q_9)^3 {z_{34}}^2 C_{34}\mu_4 {q_8}^2 + \cos(q_9)^2 p_7 z_{34}p_9 q_8 - \sin(q_9){p_7}^2 {z_{34}}^2 \\
 & & \quad - \sin(q_9){p_9}^2 {q_8}^2 - 2p_7 z_{34}p_9 q_8 - \cos(q_9){z_{34}}^2 p_8 p_9 q_8 \\
 & & \quad + \sin(q_9)F_{4z}\mu_4 {q_8}^2 {z_{34}}^3\cos(q_7)\cos(q_9)^3 \\
 & & \quad - \sin(q_9)F_{4y}\mu_4 {q_8}^2 {z_{34}}^3\sin(q_7)\cos(q_9)^3 - \sin(q_9)p_7 {z_{34}}^3\cos(q_9)p_8 \\
 & & \quad - \cos(q_9)^4 {z_{34}}^3 F_{4x}\mu_4 {q_8}^2\big)
\end{array}\right. \tag{H.5}$$

Ce qui est nouveau, par rapport aux fonctionnalités déjà vues, dans l'utilisation de *MAPLE* pour obtenir ces équations – le code se trouve dans le fichier `hamilton1` – est la résolution du système linéaire pour obtenir les $\dot{q}_i$ en fonction des p_i.

Les restrictions que l'on a faite dans le paragraphe (§2.2) concernant l'élimination de Gauss, à savoir la possibilité de savoir formellement si une expression est identiquement nulle ou non, s'appliquent aussi pour la résolution de systèmes linéaires puisque l'on est amené à diviser par des coefficients des équations du système linéaire considéré.

Toutefois, *MAPLE* est moins prudent dans la fonction `linsolve` – qui résout les systèmes linéaires – que dans la fonction `gausselim` et accepte de traiter des systèmes avec des équations dont les coefficients sont quelconques.

C'est alors l'utilisateur qui doit prendre les précautions nécessaires pour que le résultat soit correct. C'est pourquoi, avant d'appeler `linsolve` on a eu recours à la fonction `det` qui calcule le déterminant de la matrice du système linéaire en développant tous les mineurs.

mouvement relatif à R

Le code *MAPLE* correspondant aux calculs décrit ci-dessous se trouve dans le fichier `hamilton2`.

Si on étudie le mouvement du satellite relativement à R, il faut, en premier lieu, établir les 9 équations (H.3), ce qui se fait à partir des calculs déjà effectués au paragraphe précédent.

On obtient, comme précédemment, un système linéaire – en annexe §A à cause de sa taille importante – sous la forme $p = M.\dot{q}$ qu'il faut résoudre en $\dot{q}$.

Malheureusement, si on arrive à obtenir – laborieusement : plus de neuf mille secondes de calcul – le déterminant de ce système :

$$I_{1y}\cos(q_2)^2 I_{1x}I_{1z}{\mu_1}^2{\mu_4}^2{q_8}^2\cos(q_9)^2{z_{34}}^2$$

on ne dispose pas d'une puissance de calcul suffisante pour réussir à avoir les expressions des $\dot{q}_i$.

En fait, il faut relativiser cet “echec”. D'une part, ce n'est pas un problème de temps de calcul, mais de volume de calcul qui contraint *MAPLE* à s'arrêter avant d'obtenir le résultat voulu. Pendant la résolution du système, *MAPLE* stocke en effet toutes les expressions intermédiaires qu'il est amené à calculer. Lorsque l'ensemble de ces expressions atteint un volume proche du volume maximal que l'ordinateur accepte d'allouer à *MAPLE*, l'essentiel du temps CPU est alors employé par le programme pour gérer cette espace en écrivant de façon dispersée dans la mémoire pour optimiser son occupation et en recherchant systèmatiquement – très souvent – les expressions qu'il peut “oublier” pour effacer des secteurs de mémoire. Ainsi, on aurait pu, pour ce calcul particulier faire en sorte que l'ordinateur alloue à *MAPLE* un espace de travail plus grand et donc suffisant pour mener à bien sa tâche.

D'autre part, on aurait pu faire effectuer étape par étape l'algorithme d'élimination de Gauss, pour minimiser le volume des expressions intermédiaires à conserver.

Ce que l'on a voulu mettre ici en évidence est plutôt le handicap que représente l'éxécution d'un calcul aussi volumineux, incontournable si l'on emploie ce formalisme, dans l'optique d'une génération automatique des équations de la dynamique pour des mécanismes quelconques.

2.6 le formalisme de Kane

La méthode développée par Kane pour engendrer les équations de la dynamique est fondée sur le principe de d'Alembert puisqu'il s'agit d'annuler une somme de termes de forces extérieures et de forces d'inertie.

Sans anticiper sur la conclusion de ce chapitre, il est clair qu'un inconvénient de la méthode attribuée à Lagrange réside dans le calcul puis les dérivations successives de l'énergie cinétique et de la puissance généralisée des efforts connus qui fournissent des équations particulièrement volumineuses et difficilement interprétables mécaniquement.

Les méthodes issues des travaux de Newton, et de d'Alembert ont, quant à elles, pour inconvénient principal, dans l'optique d'une génération automatique des équations, de nécessiter de faire des choix, tant à propos des points où il est opportun d'appliquer le principe de d'Alembert, qu'à propos des sous-systèmes qu'il faut considérer comme isolés pour établir toutes les équations du mouvement.

La méthode prônée par Kane que l'on va appliquer ci-dessous essaye d'éviter ces inconvénients en se basant sur l'utilisation de :

- variables décrivant l'espace tangent à la variété de configurations du système mécanique, introduites initialement, et appelées *vitesses généralisées*,
- paramètres vectoriels, appelés respectivement *vitesses angulaires partielles* et *vitesses partielles*, qui sont les coefficients dans les expressions de la vitesse angulaire de n'importe quel corps du système et de la vitesse linéaire de n'importe quel point du système mécanique, sous forme de combinaisons linéaires des vitesses généralisées.

  Ce formalisme n'utilise donc plus du tout les paramètres “cinématiques” que l'on a introduits au paragraphe (§2.1).

  Ainsi, le code *MAPLE* correspondant à la mise en équation du satellite dans ce formalisme est divisé en deux fichiers :

  - le premier `kaneintro` définissant les paramètres utilisés et réalisant les calculs préliminaires à la mise en équation,
  - le second `kane` calculant les équations du mouvement.

calculs préliminaires Il est nécessaire de se donner, au préalable, neuf paramètres u_1, u_2, u_3, u_4, u_5, u_6, u_7, u_8, et u_9 provenant d'un changement de carte affine inversible à partir des paramètres $\dot{q}_i$.

Il faut ici constater que l'introduction de ces paramètres u_i ne comporte pas les mêmes désagréments que l'introduction des p_i dans le formalisme de Hamilton, à savoir l'inversion du changement de carte, puisque ces u_i sont choisis par l'utilisateur.

Il faut de plus faire, à propos des u_i les remarques suivantes :

- d'une part, on peut toujours poser pour tout i, $u_i = \dot{q}_i$ si l'on ne sait pas dégager une stratégie systématique de choix de ces paramètres.

- d'autre part, le choix des u_i peut n'être effectué qu'à la fin de l'élaboration des équations de la dynamique, et au vu de leurs expressions.

Ici, on va poser :

- $u_1 = \omega_x = \cos(q_2)\cos(q_3)\dot{q}_1 + \sin(q_3)\dot{q}_2$

- $u_2 = \omega_y = -\cos(q_2)\sin(q_3)\dot{q}_1 + \cos(q_3)\dot{q}_2$

- $u_3 = \omega_z = \sin(q_2)\dot{q}_1 + \dot{q}_3$

- $u_4 = v_x = \cos(q_2)\cos(q_3)\dot{q}_4 + (\sin(q_1)\sin(q_2)\cos(q_3) + \cos(q_1)\sin(q_3))\dot{q}_5 - (\cos(q_1)\sin(q_2)\cos(q_3) - \sin(q_1)\sin(q_3))\dot{q}_6$

- $u_5 = v_y = -\cos(q_2)\sin(q_3)\dot{q}_4 - (\sin(q_1)\sin(q_2)\sin(q_3) - \cos(q_1)\cos(q_3))\dot{q}_5 + (\cos(q_1)\sin(q_2)\sin(q_3) + \sin(q_1)\cos(q_3))\dot{q}_6$

- $u_6 = v_z = \sin(q_2)\dot{q}_4 - \sin(q_1)\cos(q_2)\dot{q}_5 + \cos(q_1)\cos(q_2)\dot{q}_6$

- $u_7 = q_8\dot{q}_7 - \sin(q_9)z_{34}\dot{q}_9$

- $u_8 = -\cos(q_9)z_{34}\dot{q}_7 + \dot{q}_8$

- $u_9 = z_{34}\dot{q}_9$

Ceci, à condition toutefois que le déterminant de ce système linéaire, i.e. $\cos(q_2)q_8z_{34}$ ne soit pas nul – ce qui est le cas dans le domaine de validité des angles de Bryant en assumant de plus que

$q_8 \neq 0$ – conduit immédiatement aux équations réciproques suivantes :

$$\left\{\begin{array}{rcl}
\dot{q}_1 & = & \dfrac{\cos(q_3)u_1 - \sin(q_3)u_2}{\cos(q_2)} \\
\dot{q}_2 & = & \sin(q_3)u_1 + \cos(q_3)u_2 \\
\dot{q}_3 & = & \dfrac{\sin(q_2)}{\cos(q_2)}\left(\cos(q_3)u_1 + \sin(q_3)u_2\right) + u_3 \\
\dot{q}_4 & = & \cos(q_2)\cos(q_3)u_4 - \cos(q_2)\sin(q_3)u_5 + \sin(q_2)u_6 \\
\dot{q}_5 & = & \left(\cos(q_1)\sin(q_3) + \sin(q_1)\sin(q_2)\cos(q_3)\right)u_4 \\
 & & \quad + \left(\cos(q_1)\cos(q_3) - \sin(q_1)\sin(q2)\sin(q_3)\right)u_5 - \sin(q_1)\cos(q_2)u_6 \\
\dot{q}_6 & = & \left(\sin(q_1)\sin(q_3) - \cos(q_1)\sin(q_2)\cos(q_3)\right)u_4 \\
 & & \quad + \left(\sin(q_1)\cos(q_3) + \cos(q_1)\sin(q_2)\sin(q_3)\right)u_5 + \cos(q_1)\cos(q_2)u_6 \\
\dot{q}_7 & = & \dfrac{u_7 + \sin(q_9)u_9}{q_8} \\
\dot{q}_8 & = & \dfrac{\cos(q_9)z_{34}u_7 + \cos(q_9)\sin(q_9)z_{34}u_9}{q_8} + u_8 \\
\dot{q}_9 & = & \dfrac{u_9}{z_{34}}
\end{array}\right. \qquad \text{(K.1)}$$

Maintenant que l'on a défini les vitesses généralisées, il s'agit d'exprimer, en fonctions de ces paramètres, les vitesses angulaires de chacun des corps et les vitesses linéaires de chaque point du système où s'applique une force.

On va exceptionnellement, tout au long de ce chapitre, noter

- $\omega^{(i)}$ le vecteur vitesse instantanée de rotation du corps $\mathcal{S}_i$ par rapport à R,
- et v^M la vitessse linéaire de n'importe quel point M du système par rapport à ce repère.

On obtient alors, en ce qui concerne les vitesses de rotation :

- $\omega^{(1)} = u_1\overrightarrow{i_1} + u_2\overrightarrow{j_1} + u_3\overrightarrow{k_1}$
- $\omega^{(2)} = \left(u_1 + \dfrac{u_7 + \sin(q_9)u_9}{q_8}\right)\overrightarrow{i_1} + u_2\overrightarrow{j_1} + u_3\overrightarrow{k_1}$
- $\omega^{(3)} = \omega^{(2)}$
- $\omega^{(4)} = \left(u_1 + \dfrac{u_7 + \sin(q_9)u_9}{q_8}\right)\overrightarrow{i_1} + \left(u_2 + \dfrac{\cos(q_7)u_9}{z_{34}}\right)\overrightarrow{j_1} + \left(u_3 + \dfrac{\sin(q_7)u_9}{z_{34}}\right)\overrightarrow{k_1}$

Quant aux vitesses linéaires elles s'expriment par :

$$v^{(G_1^t)} = u_4\overrightarrow{i_1} + u_5\overrightarrow{j_1} + u_6\overrightarrow{k_1} \quad \text{et} \quad v^{(M_{12}^t)} = (z_{12}u_2 + u_4)\overrightarrow{i_1} + (-z_{12}u_1 + u_5)\overrightarrow{j_1} + u_6\overrightarrow{k_1},$$

$$v^{(M_{21}^t)} = v^{(M_{12}^t)},$$

$$\begin{aligned} v^{(M_{34}^t)} &= ((\sin(q_7) + q_8 z_{12})\, u_2 - \cos(q_7)u_3 + u_4)\overrightarrow{i_1} \\ &\quad + \left(-(\sin(q_7)q_8 + z_{12})\, u_1 + u_5 + \left(\frac{\cos(q_7)\cos(q_9)z_{34}}{q_8} - \sin(q_7)\right)u_7\right. \\ &\quad \left. + \cos(q_7)u_8 + \left(\frac{\cos(q_7)\cos(q_9)\sin(q_9)z_{34}}{q_8} - \sin(q_7)\sin(q_9)\right)u_9\right)\overrightarrow{j_1}\ , \\ &\quad + \left(\cos(q_7)q_8 u_1 + u_6 + \left(\frac{\sin(q_7)\cos(q_9)z_{34}}{q_8} + \cos(q_7)\right)u_7\right. \\ &\quad \left. + \sin(q_7)u_8 + \left(\frac{\sin(q_7)\cos(q_9)\sin(q_9)z_{34}}{q_8} + \cos(q_7)\sin(q_9)\right)u_9\right)\overrightarrow{k_1} \end{aligned}$$

$$v^{(M_{43}^t)} = v^{(M_{34}^t)} \quad \text{et}$$

$$\begin{aligned} v^{(G_4^t)} &= ((z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34})u_2-(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})u_3+u_4+\cos(q_9)u_9)\overrightarrow{i_1} \\ &\quad +(-(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34})u_1+\sin(q_9)z_{34}u_3+u_5-\sin(q_7)u_7+\cos(q_7)u_8)\overrightarrow{j_1} \\ &\quad +((\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})u_1-\sin(q_9)z_{34}u_2+u_6+\cos(q_7)u_7+\sin(q_7)u_8)\overrightarrow{k_1} \end{aligned}$$

On peut maintenant établir les expressions des vitesses partielles et des vitesses angulaires partielles qui sont, on le rappelle, les coefficients vectoriels des u_i dans les expressions précédentes.
On obtient ainsi les vitesses angulaires partielles :

- $\omega_1^{(1)} = \overrightarrow{i_1}$, $\omega_2^{(1)} = \overrightarrow{j_1}$, $\omega_3^{(1)} = \overrightarrow{k_1}$ et $\omega_4^{(1)} = \omega_5^{(1)} = \omega_6^{(1)} = \omega_7^{(1)} = \omega_8^{(1)} = \omega_9^{(1)} = \overrightarrow{0}$,
- $\omega_1^{(2)} = \overrightarrow{i_1}$, $\omega_2^{(2)} = \overrightarrow{j_1}$, $\omega_3^{(2)} = \overrightarrow{k_1}$, $\omega_4^{(2)} = \omega_5^{(2)} = \omega_6^{(2)} = \overrightarrow{0}$, $\omega_7^{(2)} = \dfrac{1}{q_8}\overrightarrow{i_1}$, $\omega_8^{(2)} = \overrightarrow{0}$ et $\omega_9^{(2)} = \dfrac{\sin(q_9)}{q_8}\overrightarrow{i_1}$,
- $\omega_1^{(4)} = \overrightarrow{i_1}$, $\omega_2^{(4)} = \overrightarrow{j_1}$, $\omega_3^{(4)} = \overrightarrow{k_1}$, $\omega_4^{(4)} = \omega_5^{(4)} = \omega_6^{(4)} = \overrightarrow{0}$, $\omega_7^{(4)} = \dfrac{1}{q_8}\overrightarrow{i_1}$, $\omega_8^{(4)} = \overrightarrow{0}$ et $\omega_9^{(4)} = \dfrac{\sin(q_9)}{q_8}\overrightarrow{i_1} + \dfrac{\cos(q_7)}{z_{34}}\overrightarrow{j_1} + \dfrac{\sin(q_7)}{z_{34}}\overrightarrow{k_1}$

et les vitesses partielles :

- $v_1^{G_1^t} = v_2^{G_1^t} = v_3^{G_1^t} = \overrightarrow{0}$, $v_4^{G_1^t} = \overrightarrow{i_1}$, $v_5^{G_1^t} = \overrightarrow{j_1}$, $v_6^{G_1^t} = \overrightarrow{k_1}$ et $v_7^{G_1^t} = v_8^{G_1^t} = v_9^{G_1^t} = \overrightarrow{0}$,
- $v_1^{M_{12}^t} = -z12\overrightarrow{j_1}$, $v_2^{M_{12}^t} = z_{12}\overrightarrow{i_1}$, $v_3^{M_{12}^t} = \overrightarrow{0}$, $v_4^{M_{12}^t} = \overrightarrow{i_1}$, $v_5^{M_{12}^t} = \overrightarrow{j_1}$, $v_6^{M_{12}^t} = \overrightarrow{k_1}$ et $v_7^{M_{12}^t} = v_8^{M_{12}^t} = v_9^{M_{12}^t} = \overrightarrow{0}$,
- $v_1^{M_{34}^t} = (-z12 - \sin(q_7)q_8)\overrightarrow{j_1} + \cos(q_7)q_8\overrightarrow{k_1}$, $v_2^{M_{34}^t} = (z_{12} + \sin(q_7)q_8)\overrightarrow{i_1}$, $v_3^{M_{34}^t} = -\cos(q_7)q_8\overrightarrow{i_1}$, $v_4^{M_{34}^t} = \overrightarrow{i_1}$, $v_5^{M_{34}^t} = \overrightarrow{j_1}$, $v_6^{M_{34}^t} = \overrightarrow{k_1}$,

$$v_7^{M_{34}^t} = -\left(\sin(q_7) + \frac{\cos(q_7)\cos(q_9)z_{34}}{q_8}\right)\overrightarrow{j_1} + \left(\cos(q_7) + \frac{\sin(q_7)\cos(q_9)z_{34}}{q_8}\right)\overrightarrow{k_1},$$

$v_8^{M_{34}^t} = \cos(q_7)\overrightarrow{j_1} + \sin(q_7)\overrightarrow{k_1}$ et

$$\begin{aligned} v_9^{M_{34}^t} &= -\sin(q_9)\left(\sin(q_7) + \frac{\cos(q_7)\cos(q_9)z_{34}}{q_8}\right)\overrightarrow{j_1} \\ &\quad + \sin(q_9)\left(\cos(q_7) + \frac{\sin(q_7)\cos(q_9)z_{34}}{q_8}\right)\overrightarrow{k_1}\ , \end{aligned}$$

- $v_1^{G_4^t} = (-z12 - \sin(q_7)q_8 - \cos(q_7)\cos(q_9)z_{34})\,\overrightarrow{j_1} + (\cos(q_7)q_8 - \sin(q_7)\cos(q_9)z_{34})\,\overrightarrow{k_1}$,
  $v_2^{G_4^t} = (z_{12} + \sin(q_7)q_8 + \cos(q_7)\cos(q_9)z_{34})\,\overrightarrow{i_1} - \sin(q_9)z_{34}\overrightarrow{k_1}$,
  $v_3^{G_4^t} = (\sin(q_7)\cos(q_9)z_{34} - \cos(q_7)q_8)\,\overrightarrow{i_1} + \sin(q_9)z_{34}\overrightarrow{j_1}$,
  $v_4^{G_4^t} = \overrightarrow{i_1}$, $v_5^{G_4^t} = \overrightarrow{j_1}$, $v_6^{G_4^t} = \overrightarrow{k_1}$, $v_7^{G_4^t} = -\sin(q_7)\overrightarrow{j_1} + \cos(q_7)\overrightarrow{k_1}$, $v_8^{G_4^t} = \cos(q_7)\overrightarrow{j_1} + \sin(q_7)\overrightarrow{k_1}$
  et $v_9^{G_4^t} = \cos(q_9)\,\overrightarrow{i_1}$

La méthode de mise en équation de Kane repose sur l'utilisation exclusive de ces vitesses et vitesses angulaires partielles comme paramètres pour l'écriture des lois de la dynamique.

De manière tout à fait systématique, on peut en effet exprimer le principe de d'Alembert – la somme des forces extérieures et d'inerties est nulle ainsi que la somme des moments de ces forces en tout point – sous la forme synthétique suivante :

$$\text{pour } i = 1, \ldots, p \qquad \mathcal{K}_i + \mathcal{K}_i^* = 0 \tag{K.2}$$

où p est la dimension de la variété de configurations du système, et où $\mathcal{K}_i$ et $\mathcal{K}_i^*$, que l'on appelle respectivement les forces généralisées connues et les forces généralisées d'inertie sont définies par :

- $\mathcal{K}_i = \sum_{j=1}^{\nu} \left(v_i^{M_j^t}.F_j \right) + \sum_{j=1}^{n} \left(\omega_i^{(j)}.C_j \right)$ en notant :
  - ν : le nombre de particules du systèmes où des forces connues s'exercent,
  - M_j^t pour $j = 1, \ldots, \nu$: les positions de ces particules,
  - F_j pour $j = 1, \ldots, \nu$: les forces qui s'exercent en ces points,
  - n : le nombre de corps du système,
  - et C_j pour $j = 1, \ldots, n$: les couples connus qui s'exercent sur ces corps.
- $\mathcal{K}_i^* = \sum_{j=1}^{n} \left(v_i^{G_j^t}.\left(-\mu(G_j^t)\gamma(G_j^t)\right) \right) + \sum_{j=1}^{n} \left(\omega_i^{(j)}.\left(-\mathcal{I}_j \frac{d\omega^{(j)}}{dt} - \omega^{(j)} \wedge \mathcal{I}_j\omega^{(j)} \right) \right)$
  avec les mêmes notations que précédemment, et les notations habituelles.

En particularisant à l'exemple du satellite, les forces généralisées connues se développent à partir de :

$$\begin{aligned} \mathcal{K}_i \quad = \quad & v_i^{G_1^t}.F_1 + \omega_i^{(1)}.\left(C_1 - C_{12}\,\overrightarrow{i_1}\right) + v_i^{M_{21}^t}.\left(-F_{23}\,\overrightarrow{j_2}\right) + \omega_i^{(2)}.C_{12}\,\overrightarrow{i_2} \\ & + v_i^{M_{34}^t}.F_{23}\,\overrightarrow{j_3} + \omega_i^{(3)}.\left(-C_{34}\,\overrightarrow{j_3}\right) + v_i^{G_4^t}.F_4 + \omega_i^{(4)}.C_{34}\,\overrightarrow{j_4} \end{aligned} \tag{K.3}$$

pour $i = 1, \ldots, 9$ et s'écrivent donc :

- $\mathcal{K}_1 = C_{1x} - (z_{12} + \sin(q_7)q_8 + \cos(q_7)\cos(q_9)z_{34})\,F_{4y} + (\cos(q_7)q_8 - \sin(q_7)\cos(q_9)z_{34})\,F_{4z}$
- $\mathcal{K}_2 = C_{1y} + (z_{12} + \sin(q_7)q_8 + \cos(q_7)\cos(q_9)z_{34})\,F_{4x} - \sin(q_9)z_{34}F_{4z}$
- $\mathcal{K}_3 = C_{1z} + (\sin(q_7)\cos(q_9)z_{34} - \cos(q_7)q_8)\,F_{4x} + \sin(q_9)z_{34}F_{4y}$
- $\mathcal{K}_4 = F_{1x} + F_{4x}$
- $\mathcal{K}_5 = F_{1y} + F_{4y}$
- $\mathcal{K}_6 = F_{1z} + F_{4z}$
- $\mathcal{K}_7 = \dfrac{C_{12} + F_{23}\cos(q_9)z_{34} - \sin(q_7)q_8F_{4y} + \cos(q_7)q_8F_{4z}}{q_8}$
- $\mathcal{K}_8 = F_{23} + \cos(q_7)F_{4y} + \sin(q_7)F_{4z}$

- $\mathcal{K}_9 = \dfrac{\sin(q_9)\,(C_{12} + \cos(q_9)F_{23}z_{34})}{q_8} + \dfrac{\cos(q_9)F_{4x}z_{34} + C34}{z_{34}}$

Quant aux forces d'inertie généralisée, elles proviennent du développement de l'expression suivante :

$$\mathcal{K}_i^* = v_i^{G_1^t}.\left(-\mu_1\gamma(G_1^t)\right) + \omega_i^{(1)}.\left(-I_1\frac{d\omega^{(1)}}{dt} - \omega^{(1)} \wedge I_1\omega^{(1)}\right) + v_i^{G_4^t}.\left(-\mu_4\gamma(G_4^t)\right) \tag{K.4}$$

ce qui nous amène à calculer tout d'abord les expressions de l'accélération des points G_1^t et G_4^t et de l'accélération angulaire $\frac{d\omega^{(1)}}{dt}$ du corps $\mathcal{S}_1$.

En utilisant l'une des deux méthodes déjà utilisées lors de l'étude du formalisme de Newton pour calculer ces mêmes grandeurs, on obtient, en projetant dans R_1^t :

$$\gamma(G_1^t) = \begin{pmatrix} \dot{u}_4 + u_6u_2 - u_5u_3 \\ \dot{u}_5 + u_4u_3 - u_6u_1 \\ \dot{u}_6 + u_5u_1 - u_4u_2 \end{pmatrix}, \qquad \frac{d\omega^{(1)}}{dt} = \begin{pmatrix} \dot{u}_1 \\ \dot{u}_2 \\ \dot{u}_3 \end{pmatrix}$$

et pour $\gamma(G_4^t)$:

$$\begin{pmatrix}
(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34})\dot{u}_2+(\sin(q_7)\cos(q_9)z_{34}-\cos(q_7)q_8)\dot{u}_3+\dot{u}_4+\cos(q_9)\dot{u}_9+\sin(q_7)q_8u_1u_3 \\
-\sin(q_7)\cos(q_9)z_{34}u_1u_2-2\cos(q_7)u_3u_8-u_5u_3-\sin(q_9)z_{34}{u_3}^2+2\cos(q_7)u_2u_7+\cos(q_7)q_8u_1u_2+u_6u_2 \\
+z_{12}u_1u_3+\cos(q_7)\cos(q_9)z_{34}u_1u_3+2\sin(q_7)u_2u_8-\sin(q_9)z_{34}{u_2}^2+2\sin(q_7)u_7u_3 \\
-\frac{\sin(q_9){u_9}^2}{z_{34}} \\
-(q_8\sin(q_7)+\cos(q_7)\cos(q_9)z_{34}+z_{12})\dot{u}_1+\sin(q_9)z_{34}\dot{u}_3+\dot{u}_5-\sin(q_7)\dot{u}_7+\cos(q_7)\dot{u}_8+\sin(q_7)\cos(q_9)z_{34}{u_1}^2 \\
+u_4u_3-\cos(q_7)q_8{u_1}^2-\cos(q_7)q_8{u_3}^2+\sin(q_9)z_{34}u_2u_1+\sin(q_7)q_8u_2u_3+\sin(q_7)\cos(q_9)z_{34}{u_3}^2 \\
+\cos(q_7)\cos(q_9)z_{34}u_2u_3-2\cos(q_7)u_1u_7-2\sin(q_7)u_8u_1-u_6u_1+2\cos(q_9)u_9u_3+z_{12}u_2u_3 \\
-\frac{\cos(q_7){u_7}^2+\cos(q_7)u_7\sin(q_9)u_9+\sin(q_7)u_8u_7+\sin(q_7)u_8\sin(q_9)u_9}{q_8} \\
(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})\dot{u}_1-\sin(q_9)z_{34}\dot{u}_2+\dot{u}_6+\cos(q_7)\dot{u}_7+\sin(q_7)\dot{u}_8-u_4u_2-z_{12}{u_1}^2-\sin(q_7)q_8{u_2}^2 \\
-z_{12}{u_2}^2+\cos(q_7)q_8u_3u_2-\cos(q_7)\cos(q_9)z_{34}{u_1}^2+u_5u_1-\sin(q_7)q_8{u_1}^2-2\sin(q_7)u_1u_7-2\cos(q_9)u_9u_2 \\
+\sin(q_9)z_{34}u_3u_1-\sin(q_7)\cos(q_9)z_{34}u_3u_2+2\cos(q_7)u_1u_8-\cos(q_7)\cos(q_9)z_{34}{u_2}^2 \\
+\frac{-\sin(q_7)u_7\sin(q_9)u_9-\sin(q_7){u_7}^2+\cos(q_7)u_8\sin(q_9)u_9+\cos(q_7)u_8u_7}{q_8}
\end{pmatrix}$$

Ce qui donne, par substitution dans (K.4) les 9 composantes généralisées des efforts d'inertie :

$$
\begin{aligned}
\mathcal{K}_1^* \;=\;& -I_{1x}\dot{u}_1+u_3u_2I_{1y}-u_2u_3I_{1z}\\
&+\mu_4(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34})\\
&\quad(-(q_8\sin(q_7)+\cos(q_7)\cos(q_9)z_{34}+z_{12})\dot{u}_1+\sin(q_9)z_{34}\dot{u}_3+\dot{u}_5-\sin(q_7)\dot{u}_7+\cos(q_7)\dot{u}_8\\
&\qquad+u_4u_3-\cos(q_7)q_8{u_1}^2+\sin(q_7)q_8u_2u_3-\cos(q_7)q_8{u_3}^2-u_6u_1+\sin(q_9)z_{34}u_2u_1\\
&\qquad+\cos(q_7)\cos(q_9)z_{34}u_2u_3+\sin(q_7)\cos(q_9)z_{34}{u_3}^2\\
&\qquad-2\cos(q_7)u_7u_1+\sin(q_7)\cos(q_9)z_{34}{u_1}^2+z_{12}u_2u_3-2\sin(q_7)u_1u_8+2\cos(q_9)u_9u_3\\
&\qquad+\frac{-\cos(q_7)u_7\sin(q_9)u_9-\sin(q_7)u_8u_7-\sin(q_7)u_8\sin(q_9)u_9-\cos(q_7){u_7}^2}{q_8}\Big)\\
&-\mu_4(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})\\
&\quad((\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})\dot{u}_1-\sin(q_9)z_{34}\dot{u}_2+\dot{u}_6+\cos(q_7)\dot{u}_7+\sin(q_7)\dot{u}_8\\
&\qquad+\sin(q_9)z_{34}u_3u_1-\sin(q_7)\cos(q_9)z_{34}u_3u_2+\cos(q_7)q_8u_3u_2+2\cos(q_7)u_1u_8\\
&\qquad-\cos(q_7)\cos(q_9)z_{34}{u_2}^2-\sin(q_7)q_8{u_2}^2-z_{12}{u_2}^2-\cos(q_7)\cos(q_9)z_{34}{u_1}^2\\
&\qquad-2\cos(q_9)u_9u_2-\sin(q_7)q_8{u_1}^2+u_5u_1-z_{12}{u_1}^2-2\sin(q_7)u_1u_7-u_4u_2\\
&\qquad+\frac{\cos(q_7)u_8\sin(q_9)u_9+\cos(q_7)u_8u_7-\sin(q_7)u_7\sin(q_9)u_9-\sin(q_7){u_7}^2}{q_8}\Big)
\end{aligned}
$$

$$
\begin{aligned}
\mathcal{K}_2^* \;=\;& -I_{1y}\dot{u}_2+u_1u_3I_{1z}-u_3u_1I_{1x}\\
&-\mu_4(z_{12}+\sin(q_7)q_8+\cos(q_7)\cos(q_9)z_{34})\\
&\quad((\sin(q_7)q_8+z_{12}+\cos(q_7)\cos(q_9)z_{34})\dot{u}_2-(\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})\dot{u}_3+\dot{u}_4+\cos(q_9)\dot{u}_9\\
&\qquad+\sin(q_7)q_8u_1u_3+z_{12}u_1u_3+\cos(q_7)\cos(q_9)z_{34}u_1u_3+2\sin(q_7)u_2u_8-\sin(q_9)z_{34}{u_3}^2\\
&\qquad-\sin(q_7)\cos(q_9)z_{34}u_1u_2+2\sin(q_7)u_7u_3-\sin(q_9)z_{34}{u_2}^2-2\cos(q_7)u_8u_3-u_5u_3\\
&\qquad+\cos(q_7)q_8u_1u_2+u_6u_2+2\cos(q_7)u_7u_2\\
&\qquad-\frac{\sin(q_9){u_9}^2}{z_{34}}\Big)\\
&+\sin(q_9)z_{34}\mu_4\\
&\quad((\cos(q_7)q_8-\sin(q_7)\cos(q_9)z_{34})\dot{u}_1-\sin(q_9)z_{34}\dot{u}_2+\dot{u}_6+\cos(q_7)\dot{u}_7+\sin(q_7)\dot{u}_8+\sin(q_9)z_{34}u_3u_1\\
&\qquad-\sin(q_7)\cos(q_9)z_{34}u_3u_2+\cos(q_7)q_8u_3u_2+2\cos(q_7)u_1u_8-\cos(q_7)\cos(q_9)z_{34}{u_2}^2\\
&\qquad-\sin(q_7)q_8{u_2}^2-z_{12}{u_2}^2-\cos(q_7)\cos(q_9)z_{34}{u_1}^2-2\cos(q_9)u_9u_2-\sin(q_7)q_8{u_1}^2\\
&\qquad+u_5u_1-z_{12}{u_1}^2-2\sin(q_7)u_1u_7-u_4u_2\\
&\qquad+\frac{\cos(q_7)u_8\sin(q_9)u_9+\cos(q_7)u_8u_7-\sin(q_7)u_7\sin(q_9)u_9-\sin(q_7){u_7}^2}{q_8}\Big)
\end{aligned}
$$

et sept autres expressions de tailles analogues.

Il reste alors à former les équations d'après (K.2), et si on le désire substituer dans ces équations les expressions des u_i et des $\dot{u}_i$ en fonction des q_i, des $\dot{q}_i$ et des $\ddot{q}_i$ – mais il semble plus facile de résoudre le système différentiel à dix-huit variables constitué des neufs équations du mouvement et des équations (K.1) .

De tous les formalismes que l'on a employé dans ce chapitre, le formalisme de Kane est celui qui a conduit aux équations les plus concises en demandant le temps de calcul le plus faible – moins de 150 secondes CPU.

Ces performances proviennent

- d'une part de l'utilisation des grandeurs dites partielles qui factorisent naturellement les expressions des forces généralisées connues et des forces généralisées d'inertie et permettent de réduire les appels à la fonction `simplify` par leur nature vectorielle qui évite d'avoir à effectuer des changements de repères
- d'autre part du bon choix qui a été fait ici des vitesses généralisées.

Il est bien sûr, en l'absence d'algorithme, difficile d'envisager qu'un système de calcul formel puisse faire ce choix, sans intervention de l'utilisateur.

Par contre le découplage entre la phase de mise en équation en elle-même, qui peut se faire en conservant les grandeurs partielles sous forme littérale, et la phase de choix des u_i, qui peut se faire a posteriori permet de guider l'utilisateur dans ce choix, dès lors qu'on possède un système de calcul formel ayant de bonnes fonctionnalités de manipulation d'expressions vectorielles sous forme littérale.

en guise de conclusion

A propos des formalismes La nature particulière de certains mécanismes fait que, selon le cas, tel ou tel formalisme paraîtra beaucoup plus performant qu'un autre, ou engendrera des équations beaucoup plus significatives mécaniquement, ou au contraire sera pratiquement impossible à mettre en œuvre.

En choisissant un mécanisme relativement "neutre" on n'a pas réussi à mettre en évidence de réelles impossibilités d'utiliser un des formalismes étudiés – sauf en ce qui concerne le formalisme de Hamilton si le potentiel dont dérivent les efforts n'est pas connu.

L'inconvénient des méthodes que l'on a qualifiées de "classiques" vient de ce qu'elles ne se servent pas de l'ensemble des configurations des mécanismes étudiés.

En effet, toutes les grandeurs que l'on est amené à exprimer avec ces méthodes sont des vecteurs de l'espace et les équations sont obtenues en projetant ces grandeurs dans des repères – plus ou moins – convenablement choisis.

De plus cette absence de référence à la variété de configurations, débouche sur l'introduction nécessaire d'efforts de liaisons qu'il faut ensuite éliminer.

Un programme de génération des équations du mouvement sous forme littérale qui utiliserait ces méthodes devra pallier, en utilisant un savoir-faire algébrique à cet inconvénient – en éliminant algébriquement les variables inutiles, ce qui sera généralement possible mais jamais habile.

L'inconvénient par contre des méthodes purement analytiques – on a étudié seulement la méthode de Lagrange dans ce chapitre, mais la génération des équations de Boltzmann-Hamel ou de Gibbs conduirait aux mêmes remarques – est de trop s'éloigner des notions dynamiques de bases, forces et quantités d'accélération, et d'établir les équations du mouvement en faisant du calcul purement différentiel sur la variété de configurations des mécanismes étudiés.

Les formalismes plus récents ont le plus souvent étés développés dans le but d'echapper à ces inconvénients en adoptant des positions moins extrèmes.

Ils sont, pour la plupart, dérivés des principes de Newton-Euler ou de d'Alembert et proviennent de l'application des théorèmes généraux de la mécanique classique. Cependant, en exprimant les grandeurs dynamiques fondamentales directement à partir de paramétrages de la variété de configurations des mécanismes étudiés, ces formalismes évitent d'introduire des efforts inconnus ou tout au moins systématisent leur élimination.

C'est le cas du formalisme de Kane, mais aussi de tous les formalismes qui ont été récemment implémentés dans un logiciel et plus particulièrement dans un logiciel de génération formelle des équations.

En tenant compte à la fois de la nature spatiale des notions de force et d'accélération et de la nature de variété de l'ensemble des configurations des mécanismes, ces formalismes réalisent le bon compromis entre l'étendue de leur champs d'application, le volume des calculs à effectuer et le caractère significatif des équations obtenues.

En ce qui concerne les outils formels Un logiciel de calcul formel classique, comme *MAPLE*, a permis de mener à bien la génération des équations, quelle que soit la méthode employée.

Les principales fonctionnalités utilisées concernent la manipulation d'expressions, notamment la substitution et la simplification et, dans une moindre mesure, la différentiation et les opérations algébriques courantes.

Seules les fonctionnalités de manipulation d'expressions vectorielles, matricielles ou matricielles par bloc – qui sont absentes de *MAPLE* mais existent dans d'autres systèmes – devront donc être écrites.

Chapitre 3

outils pour la modélisation

Dans ce chapitre on va s'intéresser à la modélisation dynamique, par des logiciels de calcul formel, des mécanismes en général. Les problèmes de modélisation qui sont spécifiques aux mécanismes à structure bouclée ne seront étudiés qu'au chapitre suivant.

Le but, ici, n'est pas de présenter un nouveau logiciel de génération des équations de la dynamique des mécanismes, mais plutôt des principes généraux qui aident à l'écriture de tels logiciels ainsi qu'un ensemble d'outils informatiques qui, en complétant les fonctionnalités des systèmes de calcul formel classiques, permettent de mener à bien cette écriture.

3.1 architecture d'un générateur formel des équations de la dynamique

On propose dans cette section une architecture informatique adaptée aux générateurs formels des équations de la dynamique, qui permet, de plus, de séparer ces logiciels en différents modules indépendants et interconnectés.

On pourra alors traiter séparément les problèmes de conception attachés à chacun de ces modules, voire par la suite utiliser les différents systèmes de calcul formel les mieux adaptés pour programmer chacun d'entre eux.

Il est nécessaire, au préalable, de bien analyser quelles sont les tâches que doivent réaliser les générateurs formels d'équations, quelles sont, parmi les données, celles que l'on doit leur fournir à chaque utilisation, et quelles sont celles qui doivent être résidentes dans ces logiciels.

On de propose donc d'essayer de définir les besoins des utilisateurs dans ce domaine et, si nécessaire, les contraintes qu'il faut leur imposer.

On rappelle que l'utilisation de tels logiciels se situe à un stade avancé de la phase de conception assistée par ordinateur, puisqu'il s'agit d'étudier des systèmes mécaniques déjà bien définis, même si certaines de leurs caractéristiques restent paramétrées par des valeurs numériques ou formelles.

Ainsi, il n'y a pas, entre l'utilisateur et le logiciel, une interaction telle, que la nature physique du mécanisme étudié puisse être modifiée en cours d'éxécution.

3.1.1 définition des besoins et des contraintes

Le processus de génération des équations de la dynamique de mécanismes se décompose naturellement en trois tâches :

i. la description physique du mécanisme,

ii. la génération des équations en elle-même,

iii. l'exploitation de ces équations par l'utilisateur, c'est à dire :

- soit leur manipulation – simplification, linéarisation, ...– en vue d'une interprétation "humaine" interactive basée sur leur signification mécanique,
- soit la génération d'un code – paramétrable par des valeurs numériques – dédié à une simulation numérique postérieure.

Ce processus peut se schématiser par le diagramme de la figure (3.1).

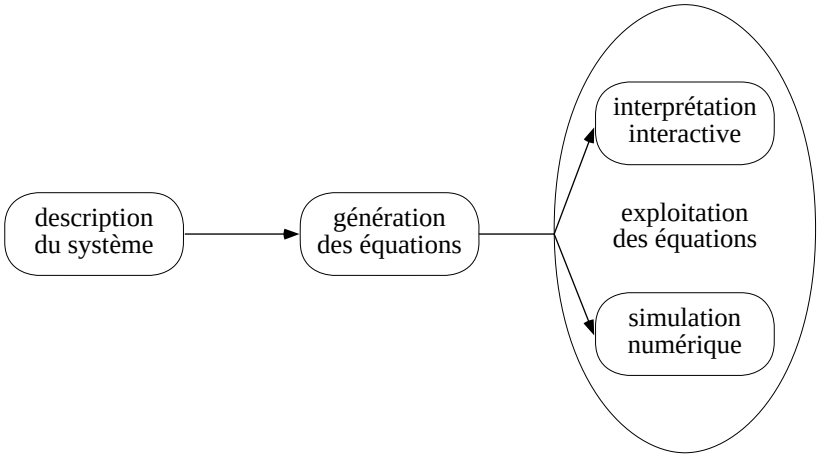


figure 3.1 : architecture du processus de génération des équations

La description physique du système mécanique doit donner au logiciel toutes les informations nécessaires – propres au mécanisme étudié – pour calculer les expressions des équations de la dynamique.

Parmi ces informations, il est indispensable d'avoir :

- le nombre de corps du système mécanique,
- une représentation de son architecture – le nombre de liaisons, et les corps reliés par chacune des liaisons,
- la nature des liaisons entre les corps – rotoïde, prismatique, universelle, ...
- pour chaque corps et pour chaque liaison, la position des particules de liaison,
- les caractéristiques cinétiques de chacun des corps, i.e la masse et l'inertie, ainsi que la position du centre d'inertie,
- les efforts exercés sur le systèmes, leurs natures et leurs points d'action,
- le cas échéant, les efforts de liaisons connus – dus par exemple à la présence de ressorts, d'amortisseurs ou de motorisations aux articulations.

Certains mécanismes étudiés peuvent être considérés comme plans et/ou leur mouvement peut être limité à être planaire. Toute la théorie développée au premier chapitre se simplifie alors de manière non négligeable, puisque, dans ce cas, une configuration d'un corps est non plus un déplacement de l'espace mais un déplacement du plan, et que l'ensemble $\mathrm{Is}^+(\mathbb{P})$ est un groupe de dimension 3, en tant qu'espace vectoriel – au lieu de 6. Il est donc bon de pouvoir informer le logiciel de cette particularité.

La génération des équations dépend, comme on l'a vu au chapitre précédent, du formalisme choisi pour l'étude de la dynamique du mécanisme considéré. On peut imaginer que l'utilisateur ne possède pas le choix parmi différents formalismes mais dispose seulement d'un formalisme "moderne", exprimant les équations de Newton-Euler ou le principe de d'Alembert dans la variété de configurations du mécanisme étudié.

Par contre, l'utilisateur doit pouvoir choisir le *niveau de détail* des équations obtenues, et disposer d'un ensemble de formules qui lui permettent de changer de niveau de détail. On va préciser cette notion déjà évoquée au paragraphe (§2.2).

Pour les équations de la dynamique, on peut définir essentiellement trois niveaux de détail :

- Au niveau le plus bas, les équations de la dynamique sont écrites exclusivement en fonction des grandeurs cinétiques et cinématiques des corps, ce sont des instanciations des théorèmes généraux pour le mécanisme étudié. Par exemple, si l'on utilise un formalisme de Newton-Euler, pour mettre en équation le mouvement d'un système de deux corps, on écrira, à ce niveau – avec des notations évidentes :
$$F_1 + F_2 = \mu_1 \gamma(G_1^t) + \mu_2 \gamma(G_2^t)$$
- Au niveau intermédiaire, on aura des équations sous forme d'expressions matricielles ou vectorielles, voire matricielles par bloc – selon la méthode de mise en équation utilisée – ne faisant pas intervenir explicitement les paramètres choisis pour décrire le mécanisme et donc relativement compactes. Les équations de ce niveau sont obtenues à partir des équations du niveau précédent en les développant le plus possible par les formules mécaniques spécifiques au mécanisme étudié. L'exemple correspondant est – toujours avec des notations évidentes :
$$F_1 + F_2 = \mu_1 \left(\gamma(O_1^t) + \frac{d\omega_1^t}{dt} \wedge \overrightarrow{O_1^t G_1^t} + \omega_1^t \wedge \left(\omega_1^t \wedge \overrightarrow{O_1^t G_1^t} \right) \right) + \dots$$
- Le plus haut niveau de détail correspond aux équations différentielles scalaires en fonctions des paramètres choisis pour décrire le système considéré.

Pour passer du plus bas niveau de détail au niveau intermédiaire, l'utilisateur devra choisir la nature de la modélisation du système mécanique étudié – modélisation du mouvement absolu, par rapport à un repère donné, non nécessairement fixe, ou modélisation des mouvements relatifs des corps les uns par rapport aux autres.

Et c'est au moment du passage entre le niveau de détail intermédiaire et le niveau le plus haut que l'utilisateur pourra décider de la manière dont il veut représenter les rotations de l'espace – à l'aide de quaternions, d'angles d'Euler, d'angles de Bryant, ...

L'exploitation des équations demande de pouvoir accéder à la fois aux fonctionnalités générales de manipulation d'expression dans le cas d'une interprétation "humaine", et aux fonctionnalités de génération de code dans le cas d'une simulation numérique.

Au cours de la génération des équations de la dynamique, le logiciel aura, vraisemblablement, eu besoin de calculer – au moins en partie – le modèle statique et les modèles cinématiques directs et inverses du système étudié, c'est à dire certaines formules concernant la géométrie ou la cinématique de ce système.

Ces renseignements devront rester à la portée de l'utilisateur pendant la phase d'exploitation des résultats, phase pendant laquelle l'accès à toutes les fonctionnalités du système de calcul formel sous-jacent au logiciel doit rester possible, pour permettre notamment d'effectuer des linéarisations des systèmes d'équations différentielles obtenus, ou d'étudier formellement, lorsque c'est faisable, les singularités de ces systèmes.

3.1.2 le point de vue informatique

On appelera *descripteur* la partie du générateur formel des équations de la dynamique dédiée à la description, par l'utilisateur, des caractéristiques physiques du mécanisme étudié.

Le *moteur* regroupe tout ce qui permet de réaliser la génération des équations. Sans trop rentrer dans les détails – on le fera par la suite – il est clair qu'il se compose d'au moins deux parties qui interagissent entre elles :

- le *calculateur* en lui-même,
- une *bibliothèque* de connaissances mécaniques portant à la fois
  - sur le *formalisme* et la méthode de mise en équation,
  - sur la nature des *liaisons*,
  - sur les différents *paramétrages*,
  - sur des *formules* mécaniques générales – changement de repères, formule de composition de vitesses et d'accélérations, théorème de Huygens, ...

Ainsi, en fonction de la méthode de mise en équation, de la nature des liaisons et des paramétrages choisis, le calculateur fait appel à la bibliothèque pour déterminer quelles sont les expressions qu'il doit calculer, et puis fait de nouveau appel à la bibliothèque pour connaître les formules générales qui lui permettent de calculer ces expressions. Il crée enfin une table de *formules spécifiques* au système considéré.

Il fournit, comme *résultats*, les *équations* en elle-mêmes ainsi que cette table de formules spécifiques, qui est en fait une instanciation pour le mécanisme considéré de certaines formules générales de mécanique.

Ensuite, selon le cas, c'est le *manipulateur–simplificateur* ou le *générateur de code* qui intervient pour travailler sur ces résultats, l'un n'excluant pas l'autre dans la mesure où l'on doit pouvoir faire de la simulation numérique après avoir manipulé les équations du mouvement.

Le manipulateur–simplificateur doit, de plus, avoir accès à la "connaissance" mécanique générale du logiciel.

On peut maintenant raffiner un peu le diagramme de la figure (3.1), en présentant dans la figure (3.2) les diverses composantes d'un générateur formel d'équations et leurs interactions mutuelles.

le descripteur

Le descripteur est, par essence, l'interface d'entrée du logiciel.

Les objets qu'il doit permettre de définir sont des corps, des liaisons, des forces, ... des objets mécaniques assez éloignés des objets mathématiques que les systèmes de calcul formel manipulent habituellement.

Dans les logiciels de modélisation dynamique existants, cette interface peut prendre trois formes différentes :

- Le plus couramment, ces logiciels ne sont pas interactifs et la description du mécanisme consiste en un fichier de données, c'est à dire une succession de champs que l'utilisateur doit instancier. Un tel fichier pour le logiciel *ROBOTRAN* (développé à l'université de Louvain, voir [Maes, Samin et Willems, 1990]) est présenté ci-dessous, il décrit un robot représenté figure (3.3) formé de trois corps reliés à une base fixe – les deux premières articulations étant des articulations cylindriques, la troisième une articulation rotoïde – sur lequel n'agit que la gravité.

```
Spatial Motion of Robot
5                          |       0;      c;      0;
0 1 2 3 4                  |       0;      0;      0;
```

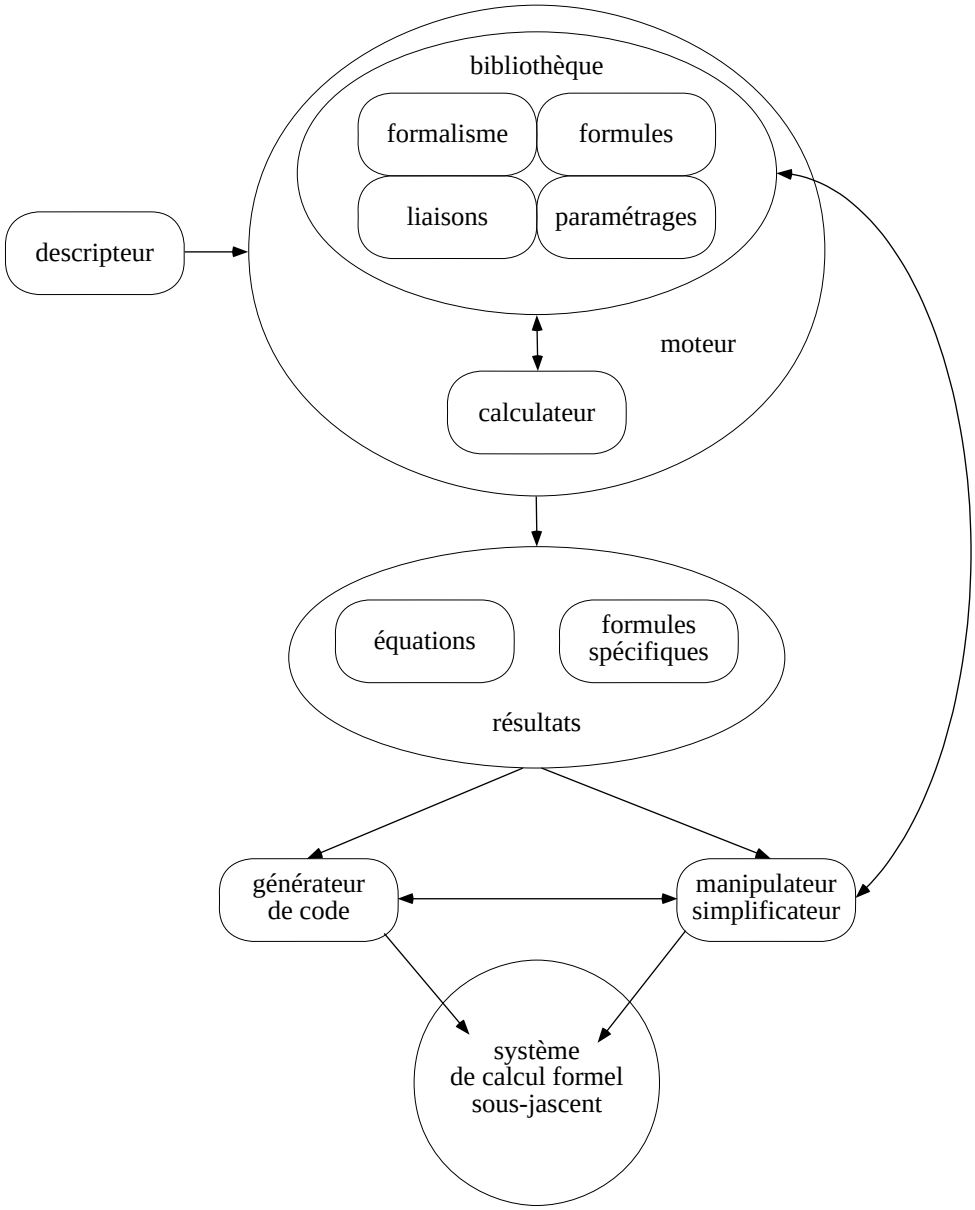


figure 3.2 : architecture d'un générateur formel des équations

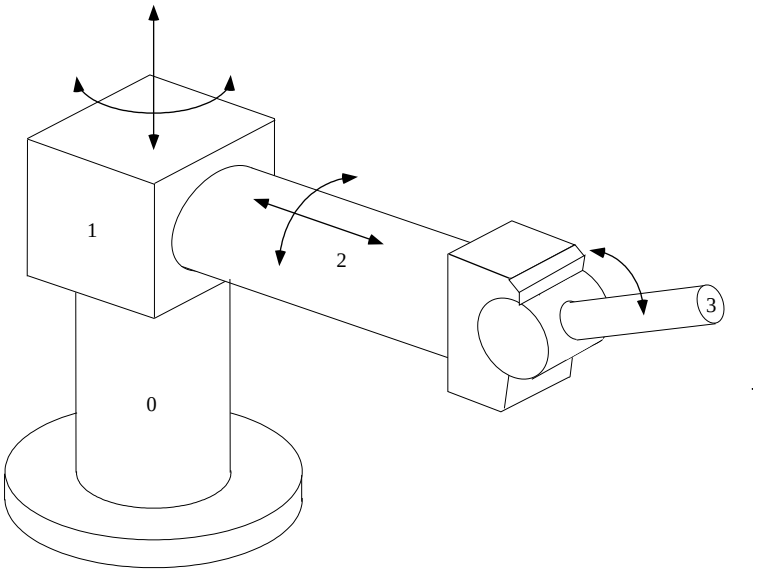


figure 3.3 : robot

```
0; m1; 0; m2; m3;            |       0;      0;      0;
0;      0;      0;           |       0;      0;      0;
        0;      0;           |       0;      0;      0;
                0;           |       0;      l;      0;
I111;   0;      0;           |       R3      q1;     qp1;    qpp1;
        I122;   0;           |       T3      q2;     qp2;    qpp2;
                I133;        |       R2      q3;     qp3;    qpp3;
0;      0;      0;           |       T2      q4;     qp4;    qpp4;
        0;      0;           |       R1      q5;     qp5;    qpp5;
                0;           |       0;      0;      0;
I211;   0;      0;           |       0;      0;      0;
        I222;   0;           |       0;      0;      0;
                I233;        |       0;      0;      0;
I311;   0;      0;           |       0;      0;      0;
        I322;   0;           |       0;      0;      0;
                I333;        |       0;      0;      0;
0;      0;      z1;          |       0;      0;      0;
0;      0;      0;           |       0;      0;      0;
0;      0;      0;           |       0;      0;      0;
0;      0;      0;           |       0;      0;      -g;
```

- Dans les logiciels interactifs, un préprocesseur de données, plus ou moins convivial, assiste l'utilisateur pour l'introduction de caractéristiques physiques du mécanisme en lui posant les questions adéquates. Ci-dessous sont représentées les deux phases – initialisation et assignation – d'une session d'utilisation du préprocesseur du logiciel *JAMES* – voir [Garnier et Rideau, 1989b] – développé à l'aerospatiale–Cannes, qui décrit le même robot* :

```
                Bienvenue sur JAMES V2.0 (Copyright 1991)
                        par C.Garnier & P.Rideau
                      Version pour MAPLE V du 03/01/92

        Ce logiciel est propriete de AEROSPATIALE Societe Nationale Industrielle
          et ne peut etre reproduit ou communique sans son autorisation ecrite

      > donnees;
```

*Les commandes et les expressions entrées par l'utilisateur sont, à chaque ligne, situées à droite du signe `>`.

```
COM-> init;
                              INITIALISATION DES DONNEES

nombre de corps > 3;

numero du 1-ieme corps > 10;
nombre de modes du corps 10 > 0;
numero du 2-ieme corps > 20;
nombre de modes du corps 20 > 0;
numero du 3-ieme corps > 30;
nombre de modes du corps 30 > 0;

nombre de points > 6;

numero du 1-ieme point > 1;
numero du corps a qui appartient le point 1 > 0;
numero du 2-ieme point > 101;
numero du corps a qui appartient le point 101 > 10;
numero du 3-ieme point > 102;
numero du corps a qui appartient le point 102 > 10;
numero du 4-ieme point > 202;
numero du corps a qui appartient le point 202 > 20;
numero du 5-ieme point > 203;
numero du corps a qui appartient le point 203 > 20;
numero du 6-ieme point > 303;
numero du corps a qui appartient le point 303 > 30;

nombre de liaisons > 3;

numero de la 1-ieme liaison > 1;
type de la liaison numero 1 > cyl;
premier point de la liaison 1 > 1;
deuxieme point de la liaison 1 > 101;
Voulez-vous associer des forces a cette liaison > n;

numero de la 2-ieme liaison > 2;
type de la liaison numero 2 > cyl;
premier point de la liaison 2 > 102;
deuxieme point de la liaison 2 > 202;
Voulez-vous associer des forces a cette liaison > n;

numero de la 3-ieme liaison > 3;
type de la liaison numero 3 > rj;
premier point de la liaison 3 > 203;
deuxieme point de la liaison 3 > 303;
Voulez-vous associer des forces a cette liaison > n;

nombre de forces > 0;

                              GRAVITE

La gravite agit-elle > oui;

entree des donnees terminee


COM-> assgn;
                              ASSIGNATIONS DES VALEURS
ASSIGN-> assgn;

Numero du corps > 0;
Voulez-vous assigner :
  1- gravite
  2- distance point de reference point 1
  3- matrice de passage locale du point 1
Numero de la commande > 1;
  Nouvelle valeur de GRAV > mat;
  Type de la matrice > sparse;

  Position de l'element a initialiser > [3,1];
  GRAV(3,1) > -g;
```

```
  Position de l'element a initialiser > fin;
                                      [  0  ]
                                      [     ]
                                      [  0  ]
                                      [     ]
                                      [ - g ]
Numero de la commande > 2;
  Nouvelle valeur de r01 > 0;
Numero de la commande > 3;
  Nouvelle valeur de PL1 > id;

Numero du corps > 10;
Voulez-vous assigner :
  1- masse
  2- inertie
  3- distance point de reference CdG
  4- distance point de reference point 101
  5- matrice de passage locale du point 101
  6- distance point de reference point 102
  7- matrice de passage locale du point 102
Numero de la commande > 2;
  Nouvelle valeur de In010 > mat;
  In010(1,1) > I111;
  In010(1,2) > 0;
  In010(1,3) > 0;
  In010(2,2) > I122;
  In010(2,3) > 0;
  In010(3,3) > I133;
                                [ I111    0     0  ]
                                [                  ]
                                [   0   I122    0  ]
                                [                  ]
                                [   0     0   I133 ]
Numero de la commande >
...
```

- les logiciels les plus sophistiqués permettent de décrire le système mécanique étudié dans un langage approprié, c'est le cas du logiciel *MESAVERDE* développé à l'université de Karlsruhe – voir [Wittenburg, Woltz et Schmidt, 1990] – dont on présente ci-dessous la description de données pour le même robot,

```
body 0 alias
       motion zero
body 1 alias tower
       force 1 (z) frame inertial
body 2 alias arm
       force 1 (z) frame inertial
body 3 alias hand
       force 1 (z) frame inertial
joint 1 type transzrotz
        from inertial to tower
joint 2 type transyroty
        from tower to arm
joint 3 type rotx
        from arm to hand
include joint.lib
```

  la concision de cette description est en partie dûe au fait que, dans ce robot, les repères de référence et les liaisons correspondent précisément aux repères et aux liaisons génériques prédéfinies dans la bibliothèque `joint.lib`.

L'existence d'un langage de description des objets mécaniques est nécessaire si l'on veut pouvoir décrire ces objets avec une syntaxe simple et/ou avec des outils graphiques – les primitives graphiques sont traduites en "macros" du langage – et pour utiliser facilement des bibliothèques d'objets réutilisables.

Dans la section (§3.2), on présente une maquette d'un langage évolué de description des mécanismes dont une originalité est d'être indépendant de tout système de calcul formel, et de tout logiciel

de génération symbolique des équations déjà existant, mais d'être conçu pour pouvoir s'interfacer avec n'importe lequel de ces logiciels.

le moteur – bibliothèque et calculateur

Les bibliothèques représentent la "connaissance" mécanique du système.

Cette connaissance est constituée d'ensembles d'équations qui servent à effectuer des substitutions.

On peut aussi considérer ces équations comme des règles de réécriture : au cours des calculs, chaque occurence du terme de gauche d'une formule devra être remplacée par le membre de droite correspondant.

La plupart des systèmes de calcul formel actuellement disponibles peuvent manipuler simplement des équations et appliquer des règles de réécriture. Néanmoins, la gestion et l'utilisation de telles bibliothèques pose des problèmes particuliers au niveau :

- de la compatibilité des équations,
- du typage des objets qu'elles font intervenir,
- de leurs stratégies d'utilisation.

On exposera, dans la section (§3.5) des reflexions théoriques sur ces problèmes.

Dans la section (§3.3), on présentera un formalisme original, et un algorithme permettant d'engendrer les équations du mouvement avec ce formalisme.

le manipulateur-simplificateur

Le plus souvent, les logiciels de génération des équations de la dynamique sous forme symbolique ont été écrit dans un langage classique – *FORTRAN* ou *C* – et non à partir d'un système de calcul formel.

C'est pourquoi leur principale lacune se situe au niveau de la manipulation et de la simplification des équations obtenues.

Comme on l'a vu au chapitre précédent, *MAPLE* possède des fonctionnalités de manipulation et de simplifications d'expressions intéressantes. De manière générale, un logiciel de calcul formel permet d'effectuer sur une expression faisant intervenir des nombres ou des fonctions numériques :

- des factorisations, entières ou partielles, et des développements,
- l'application de règles définies par l'utilisateur,

et simplifie automatiquement les expressions en reconnaissant – le plus souvent – 0, 1, l'opposé et l'inverse d'une expression.

En ce qui concerne la manipulation d'expressions vectorielles et matricielles, il va être nécessaire de définir les opérateurs utilisés et les règles de simplification qui peuvent s'appliquer.

On présentera, dans la section (§3.4), les procédures définissant, en *MAPLE*, ces opérateurs et ces règles ainsi qu'un ensemble de procédures permettant de manipuler les quaternions représentant des rotations.

A titre de comparaison, on présentera, dans la section (§3.5) un manipulateur-simplificateur spécialisé pour le traitement des matrices et des vecteurs développé au-dessus d'*ULYSSE* – un logiciel de calcul formel expérimental développé à l'INRIA Sophia-Antipolis, au sein du projet *SAFIR* – précisément pour permettre d'utiliser des règles de réécriture de termes et de définir des opérateurs en leur attachant des axiomes.

le générateur de code

La simulation numérique est le plus souvent l'étape ultime de l'étude du mouvement d'un mécanisme.

C'est pourquoi, un logiciel de génération des équations de la dynamique doit pouvoir fournir, in fine, un simulateur numérique du mouvement,

- soit en produisant un fichier de données ayant le format d'entrée de certaines bibliothèques de routines numériques – par exemple NAG, voir [NAG Inc.]
- soit en créant, dans un langage approprié – *FORTRAN*, *ADA*, *C*, ... – un code complet d'intégration numérique des équations différentielles du mouvement.

Dans le deuxième cas, le générateur de code doit alors être capable de :

- transcrire dans le langage souhaité les expressions algébriques,
- créer les routines de calcul – inversion de matrices, résolution d'équations – et les routines annexes notamment d'entrée et de sortie,
- gérer les déclarations de variables, de constantes, les passages de paramètres, ...

Ces fonctionnalités existent en *MAPLE*, en ce qui concerne le langage *FORTRAN*. Elles ont été développée, à l'INRIA Rocquencourt, sous la forme de la bibliothèque de procédures *MACROFORT* – voir [Gomez, 1990].

Des travaux ont été réalisés pour améliorer cette bibliothèque au niveau de la transcription des expressions algébriques en optimisant les calculs matriciels et en évitant les calculs redondants – voir [Capolsini, 1990] – et une bibliothèque similaire concernant le langage *C* a été développée à l'Université de Nice – Sophia Antipolis – voir [Capolsini, 1992].

3.2 *LSD* : un Langage Symbolique de Description des mécanismes

Le langage de description des mécanismes qui est présenté ci-dessous a été conçu et implémenté en tenant compte des recherches les plus récentes sur les problèmes logiciels, et notamment de langages, en calcul formel – voir [Dalmas, 1991].

Décrire un système mécanique, c'est d'abord déclarer que ce système est constitué d'un certain nombre de corps, puis comment ces corps sont reliés entre eux, et quels sont les efforts connus aux articulations, et, enfin, quels sont les efforts extérieurs qui s'appliquent sur ce système.

C'est pourquoi *LSD* est un langage déclaratif, c'est à dire qu'un programme de ce langage est une suite de déclarations qui introduisent et définissent la nature physique du mécanisme qui va être étudié.

Lorsque l'on conçoit des mécanismes, on est amené à souvent utiliser des corps qui possèdent des caractéristiques voisines, à réutiliser certains types de liaisons et certains types d'efforts.

C'est pourquoi un programme de *LSD* peut aussi servir à créer une bibliothèque dans laquelle seront conservées les définitions d'objets mécaniques divers – corps, forces ou liaisons – amenés à intervenir souvent dans les mécanismes considérés.

On peut aussi, en *LSD*, définir des classes d'objets mécaniques dépendant d'un ou plusieurs paramètres et récupérer, par la suite, les objets voulus en instanciant ces paramètres. Cette caractéristique accroît considérablement l'intérêt de l'utilisation de bibliothèques.

Un programme, une fois écrit, est transmis à un compilateur, qui successivement,

- vérifie qu'il est correct en l'analysant lexicalement et syntaxiquement,

- vérifie sa cohérence en examinant le typage des différents objets introduits.

Le mécanisme de typage de *LSD* est à la fois simple et performant : il ne pénalise pas le temps de compilation mais permet la "surcharge" – `0` peut représenter, selon le contexte, le nombre 0, le vecteur nul, le déplacement nul, ...

Après la phase de vérification, et si le programme en question est la description d'un système mécanique, le compilateur fournit un code compréhensible comprenant toutes les expressions spécifiques au système mécanique et nécessaires à l'écriture des équations qui gouvernent son mouvement.

Selon l'option choisie, ce code pourra alors être traduit puis transmis à un logiciel de calcul formel ou à un logiciel de génération formelle des équations de la dynamique.

Avant de décrire précisément la syntaxe et la sémantique du langage, on va présenter ses caractéristiques principales à partir d'un exemple concret de programme *LSD*.

3.2.1 un exemple de programme

Décrire un système mécanique en *LSD* ne demande pas, de la part de l'utilisateur, d'effectuer une analyse mécanique préalable de ce système dans la mesure où l'utilisateur n'a pas :

i. à nommer explicitement les différents objets manipulés, ni les grandeurs qui interviennent dans l'étude dynamique du système, mais seulement – et pour des raisons évidentes – les objets rencontrés à plusieurs endroits dans la description du système,

ii. ni à déclarer les grandeurs qui ont des valeurs formelles génériques mais doit simplement décrire les grandeurs qui possèdent des propriétés particulières.

Le code ci-dessous décrit en *LSD* le satellite supportant un reflecteur articulé sur un bras télescopique que l'on a déjà rencontré au chapitre (§2) :

```
(* les corps *)
let satellite = body iframe = 0 end ;
let bras_primaire = body mass = 0, inertia = 0 end ;
let bras_secondaire = body mass = 0, inertia = 0 end ;
let reflecteur = body iframe = [0,0,?s] ++ 1, inertia = 0 end ;

(* les liaisons *)
join S1.frame and earth
  with link(x,y,z,ps,th,ph)
         dof1 = [x,y,z],
         dof2 = ps || [0,0,1],
         dof3 = th || [1,0,0],
         dof4 = ph || [0,0,1]
       end ;
join [0,0,?s] ++ 1 @ satellite and bras_primaire.frame
  with link(x) dof = x || [1,0,0], torque = [?fs(x,x'),0,0] end ;
join bras_primaire.frame and bras_secondaire.frame
  with link(x) dof = [0,x,0], force = [0,?fs(x,x'),0] end ;
join bras_secondaire.frame and reflecteur.frame
  with link(x) dof = x || [0,1,0], torque = [0,?fs(x,x'),0] end ;

(* les forces *)
apply force body = satellite end on satellite.iframe ;
apply force body = satellite end on reflecteur.iframe ;
```

Le point (ii) est illustré par les deux premières déclarations du programme,

```
let satellite = body iframe = 0 end ;
let bras_primaire = body mass = 0, inertia = 0 end ;
```

qui traduisent que physiquement le satellite est un corps dont le repère de référence coïncide avec le repère d'inertie et que la partie primaire du bras est de masse et d'inertie nulle.

Le point (i) se traduit dans ce code, par le fait que les seuls objets nommés explicitement sont les corps puisqu'on y fait référence aussi dans la définition des liaisons.

C'est la compilation du programme qui se charge de numéroter les corps, et de créer tous les identificateurs nécessaires pour désigner les repères, les déplacements, les vecteurs, ... et qui fournit une table de notation précisant quel identificateur représente quel objet.

> Etant donné le nombre de grandeurs qui interviennent dans l'établissement des équations du mouvement d'un système mécanique – à chaque corps, on associe au moins un repère de référence, un repère inertiel et un repère à chaque point d'articulation ; à chaque repère, on associe une origine, sa position, son orientation, sa vitesse linéaire, sa vitesse instantanée de rotation, ...– cette fonctionnalité de *LSD* le rend particulièrement agréable à utiliser puisqu'elle évite d'avoir à gérer un grand nombre d'identificateurs.

Il existe en *LSD* un mécanisme de gestion de variables muettes, capable de distinguer les paramètres mécaniques du système des paramètres dépendant du temps sur lesquels porteront les équations de la dynamique. L'interêt de ce mécanisme est de permettre de traduire n'importe quelle propriété physique, sans imposer, là non plus, de contrainte de nommage.

> Ainsi, dans l'expression ci-dessous :
>
> ```
> join [0,0,?s] ++ 1 @ satellite and bras_primaire.frame
>   with link(x) dof = x || [1,0,0], torque = [?fs(x,x'),0,0] end ;
> ```
>
> le fait que :
>
> - le référentiel du point d'articulation du satellite avec le bras est l'image du référentiel de référence du satellite par une translation selon l'axe Oz de ce dernier se traduit simplement par `[0,0,?s] ++ 1 @ satellite`
> - la liaison fait intervenir un paramètre dépendant du temps est contenu dans `link(x) dof = x || [1,0,0],`
>
> et il n'est pas nécessaire de désigner explicitement la valeur de la translation – que l'on avait désignée au chapitre (§2) par $(0, 0, z_{12})$ – ou le paramêtre utilisé – $q_7(t)$.

Cette fonctionnalité de *LSD* est importante en ce qui concerne :

- d'une part sa réusabilité,

  > Si l'on veut, par exemple, considérer, dans une étude ultérieure, le même satellite relié cette fois à une navette par un verin, il ne sera pas nécessaire de redéfinir la liaison entre le satellite et le bras alors que vraisemblablement on préférera désigner le satellite par $\mathcal{S}_2$, le bras par $\mathcal{S}_3$ et donc la translation en question par $(0, 0, z_{23})$ au lieu de $(0, 0, z_{12})$.

- d'autre part sa concision, et l'utilisation de bibliothèque.

  > On peut en effet remplacer les deux déclarations concernant les liaisons :
  >
  > ```
  > join [0,0,?s] ++ 1 @ satellite and bras_primaire.frame
  >   with link(x) dof = x || [1,0,0], torque = [?fs(x,x'),0,0] end ;
  > join bras_secondaire.frame and reflecteur.frame
  >   with link(x) dof = x || [0,1,0], torque [0,?fs(x,x'),0] end ;
  > ```

par la déclaration suivante qui définit d'une manière générale les liaisons rotoïdes avec rappel et amortissement – que l'on conservera dans une bibliothèque :

```
let rotoide(axe) = link(x) dof = x || axe,
                           torque = ?fs(x,x')*axe
end ;
```

et les deux déclarations de liaisons – qui feront partie du programe :

```
join [0,0,?s] ++ 1 @ satellite and bras_primaire.frame
  with rotoide([1,0,0]) ;
join bras_secondaire.frame and reflecteur.frame
  with rotoide([0,1,0]) ;
```

3.2.2 la sémantique de *LSD*

LSD permet de décrire et manipule des objets de deux natures différentes :

- les objets mécaniques – corps, liaisons, forces et référentiels,
- les objets mathématiques – scalaires, vecteurs, matrices, déplacements, translations et rotations.

Les objets mécaniques sont des enregistrements de plusieurs champs décrivant leurs caractéristiques.

Ils doivent être explicitement créés par l'utilisateur au moyen d'une déclaration, tandis que les objets mathématiques sont automatiquement créés par le langage, et notamment à l'aide de son mécanisme de typage.

Lorsque certains champs sont absents de la déclaration d'un objet mécanique, cela signifie simplement que la caractéristique correspondante de l'objet ne possède pas de propriété particulière, mais ces champs son créés par le langage et sont dès lors accessibles.

Ainsi la déclaration `let satellite = body iframe = 0 end ;` est comprise par le langage de la manière suivante :

> `satellite` est un corps et, à ce titre, possède une masse, une inertie, ... qui sont quelconques, mais ses repères de référence et d'inertie sont confondus.

Comme on l'a déjà dit, *LSD* reconnaît quatre types d'objets mécaniques :

- les corps,
- les liaisons,
- les forces (ou couples),
- et les référentiels.

Un corps est nécessairement rigide – voir paragraphe (§1.1.2) –. L'objet du langage <*corps*> possède quatre champs :

- <*corps*>`.frame` désigne un référentiel de référence qui lui est solidaire – toutes les propriétés de nature géométrique sont définies à partir de ce repère de référence ;
- <*corps*>`.mass` désigne la masse du corps ;
- <*corps*>`.iframe` désigne le référentiel principal d'inertie du corps – i.e. le référentiel ayant pour origine le centre d'inertie du corps et de base telle que la matrice de l'opérateur d'inertie du corps par rapport à cette base soit diagonale ;

- <*corps*>`.inertia` désigne l'opérateur central d'inertie du corps, i.e. l'opérateur d'inertie par rapport à son centre d'inertie, sous la forme du vecteur des valeurs propres de sa matrice dans la base associée au référentiel principal d'inertie.

On décrit un corps à l'aide du morceau de programme suivant :

`body mass =` <expression>`, iframe =` <expression> `, inertia =` <expression> `end`

L'expression attendue dans l'égalité concernant `iframe` est celle du déplacement qui transforme le repère de référence du corps en son référentiel principal d'inertie. Si ce déplacement est donné explicitement, il doit l'être par rapport au repère de référence du corps.

Les liaisons que l'on peut décrire en *LSD* sont les liaisons holonomes régulieres telles que l'ensemble des déplacements relatifs admissibles des deux corps liés soit un sous-groupe du groupe des déplacements – voir paragraphe (§1.2.2) et notamment la remarque (1.34). Une liaison existe de manière intrinsèque et peut être utilisée à plusieurs endroits dans un même système. Les renseignements concernant les corps reliés par une liaison ne sont donc pas considérés comme des caractéristiques d'une liaison et font l'objet d'une déclaration séparée :

`join` <*référentiel*> `and` <*référentiel*> `with` <*liaison*>

Ainsi, l'objet mécanique <*liaison*> possède trois types de champs :

- les champs du premier type, <*liaison*>`.dof`i où i est un entier optionnel compris entre 1 et 6 décrivent précisément la nature du sous-groupe des déplacements relatifs admissibles et ont la forme d'une liste de déplacements dépendant de paramètres ;
- les champs des deux autres types , <*liaison*>`.force`i et <*liaison*>`.torque`i désignent les effort de liaisons, respectivement forces et couples, lorsque ce sont des efforts connus.

La description de liaisons a une forme similaire à celle des corps :

`link dof =` <expression>`, force =` <expression> `, torque =` <expression> `end`

ou

```
link dof1 = <expression>, ... dof6 = <expression>,
     force1 = <expression>, ... force6 = <expression>,
     torque1 = <expression>, ... torque6 = <expression>
end
```

On utilise la première forme si les mouvements relatifs admissibles pour la liaison font parties d'un groupe de translation, ou d'un groupe à un seul paramètre. On doit alors avoir, en regard de `dof`, la translation en question ou un déplacement du groupe, exprimé relativement au premier référentiel de la liaison. Par exemple, une liaison prismatique le long de l'axe de vecteur directeur $\vec{\imath}+\vec{\jmath}$ du premier référentiel de la liaison se déclarera par :

```
link(x) dof = [x,x,0] end
```

et une liaison parallèle dans le plan $(O,\vec{\imath},\vec{\jmath})$ de ce même référentiel se déclarera par :

```
link(x,y) dof = [x,y,0] end
```

Si, par contre, l'ensemble des mouvements relatifs admissibles est un groupe à plusieurs paramètres qui ne contient pas que des translations, on utilisera une liste de champs `dof`i contenant chacun un déplacement, le premier déplacement étant exprimé dans le premier référentiel de la liaison, le déplacement suivant étant exprimé dans le référentiel obtenu en appliquant le premier déplacement et ainsi de suite.

Ainsi, la déclaration :

```
link(a,b) dof1 = [0,a,0], dof2 = b || [1,0,0] end
```

désigne une liaison que l'on peut décomposer en une translation selon l'axe Oy suivi d'une rotation d'angle b autour de l'axe Ox, tandis que la déclaration :

```
link(a,b) dof1 = b || [1,0,0], dof2 = [0,a,0] end
```

désigne une liaison que l'on peut décomposer en une rotation selon l'axe Ox suivie d'une translation selon l'axe Oy_1 image par la rotation de l'axe Oy.

De manière analogue, dans l'exemple de programme vu plus haut, la liaison entre le référentiel d'inertie du satellite et le repère fixe dénommé `earth` que l'on a déclaré par :

```
link(x,y,z,ps,th,ph)
        dof1 = [x,y,z],
        dof2 = ps || [0,0,1],
        dof3 = th || [1,0,0],
        dof4 = ph || [0,0,1]]
end ;
```

possède un degré de liberté égal à 6, et est paramétrée par la translation entre les origines et, en ce qui concerne la rotation, par les angles d'Euler.

Les expressions à l'intérieur des égalités relatives aux champs `force`, `torque`, `force`*i* et `torque`*i* sont des vecteurs exprimés, le cas échéant, dans le même repère que le déplacement auquel ils sont associés.

Les forces et les couples qui ne sont pas des efforts de liaisons sont considérés par le langage comme des objets mécaniques à part entière. Comme pour les liaisons, on ne considère pas que le point d'application d'une force est une caractéristique de cette force, ni que le corps sur lequel s'applique un couple est une caractéristique de ce couple. On utilise pour définir le point d'application d'une force ou le corps sur lequel s'applique un couple, une déclaration séparée :

`apply` <*force*> `on` <*référentiel*> ou
`apply` <*torque*> `on` <*corps*>

Cela permet de pouvoir utiliser la même force – par exemple la gravité – en plusieurs points du système.

Les objets <*force*> (resp. <*torque*>) possèdent donc deux champs :

- le premier <*force*>`.value` (resp. <*torque*>`.value`) désigne le vecteur force (resp. le moment du couple) ;
- le second <*force*>`.body` (resp. <*torque*>`.body`) désigne le corps par rapport auquel ce vecteur (resp. ce moment) est exprimé.

On utilise, pour déclarer les forces et les couples, une construction similaire à celle déjà vue pour les corps et les liaisons :

`force value =` <expression>`, body =` <expression> `end`
ou
`torque value =` <expression>`, body =` <expression> `end`

Les expressions des champs <*force*>`.value` et <*torque*>`.value` sont des vecteurs tandis que les expressions des champs <*force*>`.body` et <*torque*>`.body` sont des corps déjà déclarés.

Un référentiel peut-être vu comme l'image d'un autre référentiel par un déplacement. En *LSD*, il existe un référentiel privilégié que l'on désigne par `earth` qui est le référentiel galiléen que l'on considérera fixe pendant l'étude du mouvement du système mécanique considéré. Les autres référentiels sont définis à partir du référentiel de référence d'un corps – ou de `earth` – et d'un déplacement. Ainsi, l'objet mécanique <référentiel> possède deux champs :

- <*référentiel*>`.body` désigne le corps duquel le référentiel est solidaire ;

- <*référentiel*>`.displ` désigne le déplacement qui transforme le repère de référence du corps <*référentiel*>`.body` en le référentiel voulu.

Contrairement aux précédents objets mécaniques, un référentiel doit se déclarer à l'aide d'un constructeur qui ne fait pas explicitement référence aux champs, sous la forme :

<*déplacement*> `@` <*corps*>

Les objets mathématiques peuvent être définis à l'aide de certains opérateurs et de certains constructeurs, mais cela n'est pas indispensable, un identificateur ou toute autre expression syntaxiquement correcte et compatible au niveau du typage pouvant en effet représenter n'importe quel objet mathématique.

Les constructeurs définis en *LSD* sont les suivants :

- `[`$< s_1 >$`,`$< s_2 >$`,`$< s_3 >$ `]` qui crée un vecteur à partir de ses composantes scalaires,
- <*angle*>`||`<*axe*> qui crée une rotation à partir de son angle et de son axe,
- et <*translation*>`++`<*rotation*> qui crée un déplacement à partir d'une translation et d'une rotation.

3.2.3 la syntaxe de *LSD*

Les objets manipulés par ce langage sont répartis dans deux classes syntaxiques, à savoir :

- les *déclarations*,
- et les *expressions*.

Un programme du langage est une suite de déclarations. Une déclaration a pour effet d'introduire de nouveaux objets, elle peut être construite de deux manières différentes :

- récursivement, comme une suite de déclarations,
- ou en liant un *identificateur* à une expression,

On peut regrouper les déclarations en bibliothèques, ce qui permet, entre autre, de disposer de différents domaines de nommage.

Les identificateurs acceptés sont des chaînes de caractères commençant par une lettre ou par un underscore (`_`) et contenant autant de lettres et de chiffres que souhaité. La différence entre les lettres minuscules et majuscules est reconnue.

On peut aussi utiliser des *identificateurs paramétrés* qui sont construits à partir d'un identificateur et paramétrés par d'autres identificateurs.

Les expressions admises par le langages peuvent être :

- des identificateurs ou des identificateurs paramétrés.
- des identificateurs qualifiés, construits automatiquement à partir d'identificateurs et d'opérateurs de sélection, pour récupérer les déclarations d'une bibliothèque ou les champs d'un objet,
- créées récursivement à partir d'autres expressions par :
  - des constructeurs d'objets mécaniques,
  - des opérateurs mathématiques,
  - un mécanisme qui permet de leur associer localement des déclarations.

- données sous forme de notation fonctionnelle faisant intervenir un identificateur et d'autres expressions.

La syntaxe du langage est résumée ci-dessous :

- <programme> `::=` <liste de déclarations>
- <liste de déclarations> `::=` <>
  `::=` <déclaration> `;` <liste de déclarations>
- <déclaration> `::= let` <liste d'équations>
  `::= let` <identificateur simple> `(` <liste d'identificateurs typés> `)`
  `=` <expression>
  `::= join` <expression> `and` <expression> `with` <expression>
  `::= package` <identificateur simple> `=` <liste de déclarations> `end`
  `::= apply` <expression> `on` <expression>
- <liste d'équations> `::=` <>
  `::=` <équation>
  `::=` <équation> `and` <liste d'équations>
- <identificateur simple> $\in$ *Lang* $\left([\_, A..Z, a..z] + [A..Z, a..z, 0..9]^*\right)$
- <liste d'identificateurs typés> `::=` <>
  `::=` <identificateur simple> `:` <identificateur simple>
  `::=` <identificateur simple> `:` <identificateur simple> `,`
  <liste d'identificateurs typés>
- <expression> `::=` <identificateur>
  `::=` <identificateur d'inconnue>
  `::=` <construction>
  `::=` <expression mathématique>
  `::= [` <expression> `,` <expression> `,` <expression> `]`
  `::= local` <liste d'équations> `in` <expression> `end`
  `::=` <identificateur> `()`
  `::=` <identificateur> `(` <liste d'expressions> `)`
  `::=` <identificateur d'inconnue> `(` <liste d'expressions> `)`
  `::=` <identificateur> `:` <identificateur> `(` <séquence d'équations–1> `)`
  `::=` <identificateur> `(` <séquence d'équations–1> `)`
  `::= (` <expression> `)`
- <équation> `::=` <identificateur simple> `=` <expression>
  `::= force =` <expression>
  `::= torque =` <expression>
- <identificateur> `::=` <identificateur simple>
  `::=` <identificateur qualifié>
  `::=` <identificateur paramétré>
- <identificateur d'inconnue> $\in$ *Lang* $\left([?] + [\_, A..Z, a..z]^+\right)$
- <construction> `::= body` <séquence d'équations–2> `end`
  `::= link` <séquence d'équations–2> `end`
  `::= force` <séquence d'équations–2> `end`
  `::= torque` <séquence d'équations–2> `end`

- <expression mathématique> ::= <expression> + <expression>
  ::= <expression> * <expression>
  ::= <expression> / <expression>
  ::= <expression> - <expression>
  ::= <expression> ^ <expression>
  ::= <expression> @ <expression>
  ::= <expression> || <expression>
  ::= <expression> ++ <expression>
  ::= <nombre entier>
- <liste d'expressions> ::= <expression>
  ::= <expression> , <liste d'expressions>
- <séquence d'équations–1> ::= <équation>
  ::= <équation> , <séquence d'équations–1>
- <identificateur qualifié> ::= <identificateur simple> . <identificateur simple>
- <identificateur paramétré> ::= <identificateur simple> : <identificateur simple>
  ::= <identificateur simple> :
  <identificateur simple> . <identificateur simple>
- <séquence d'équations–2> ::= <>
  ::= <équation>
  ::= <équation> , <séquence d'équations–1>

3.2.4 la compilation d'un programme *LSD*

Pour implémenter un compilateur de *LSD*, on a utilisé le langage *Standard ML*, qui est un langage fonctionnel polymorphe particulièrement bien adapté à la création de prototypes d'autres langages – voir [Milner, Tofte et Harper, 1990] ou [Paulson, 1991].

Ainsi le compilateur a été construit à partir de différents fichiers qui sont fournis en annexe (§B.1) :

- `LSD.lex` qui décrit l'alphabet du langage, les caractères et les mots réservés, sert à créer un analyseur lexical,
- `LSD.grm` qui décrit la grammaire du langage et sert à créer son analyseur syntaxique,
- `absyn.sml` et `typecheck.sml` sont des fichiers spécifiques qui décrivent la syntaxe abstraite utilisée par l'analyseur du compilateur et le mécanisme de typage.

On n'abordera pas ici les détails techniques concernant la compilation des programmes *LSD*, on va se limiter à décrire le mécanisme de nommage inclus dans le compilateur, et, sur un exemple, les expressions fournies, in fine, par celui-ci.

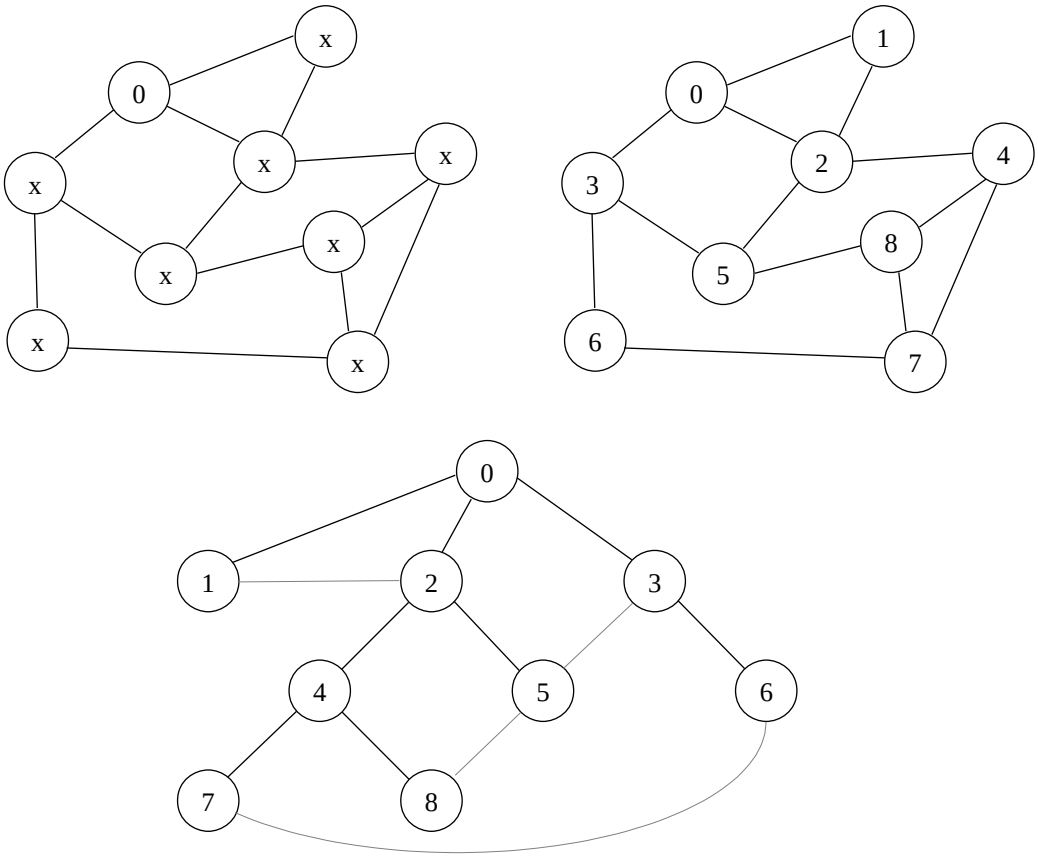
le mécanisme de nommage

En ce qui concerne le nommage, la première tâche effectuée à la compilation consiste à récupérer, à partir des déclarations de liaisons entre les corps, l'architecture du mécanisme décrit par le programme et à numéroter les corps en fonction de cette architecture.

Ainsi, on crée le graphe associé au système mécanique – voir la fin du paragraphe (§1.1).

Ensuite, on parcourt ce graphe en utilisant un algorithme classiquement appelé "Breadth-first search" produisant un arbre de recouvrement du graphe de profondeur minimal – voir pour plus de détails [Cormen, Leiserson et Rivest, 1989]. Cet algorithme numérote les sommets au fur et à mesure et localise les boucles – on présente son fonctionnement sur un exemple dans la figure (3.4).

L'arbre de recouvrement correspond à un système mécanique arborescent sous-jacent au mécanisme de départ. Les liaisons représentés par les arêtes de cet arbre sont dites *principales*, les autres liaisons sont dites *complémentaires*. Cette distinction sera utile lorsque l'on s'intéressera à l'étude du mouvement de mécanismes bouclés – voir (§4).



Le schéma en haut à gauche montre l'architecture avant la numérotation, seul le sommet correspondant au repère considéré comme fixe – `earth` – a été distingué comme la racine de l'arbre de recouvrement, c'est le point d'entrée de l'algorithme, et il est noté 0.

En haut à droite est représentée la même architecture après numérotation.

Sur le schéma du bas, on voit l'arbre de recouvrement, les arêtes en pointillés correpondent aux liaisons complémentaires qui rendent l'architecture bouclée.

figure 3.4 : exemple de parcours de graphe en "Breadth-first search"

Les corps étant numérotés de 1 à n , on peut numéroter les liaisons :

Il existe une liaison principale et une seule qui relie le corps i à un corps j si $i < j$, à cause de l'algorithme de parcours de graphe utilisé. On numérote donc les liaisons principales de 1 à n comme les corps, une liaison ayant pour numéro le plus grand des numéros attribués aux deux corps qu'elle relie. Les liaisons complémentaires sont numérotées à partir de $n + 1$, selon un ordre arbitraire.

Les référentiels qui sont introduits dans les liaisons sont repérés en premier lieu par le numéro du corps duquel ils sont solidaires et aussi par le numéro du second corps intervenant dans la

liaison. Si, dans la description du système mécanique, interviennent d'autres référentiels, ils sont numérotés toujours par le corps duquel ils sont solidaires et de plus par un entier supérieur à n dans un ordre arbitraire, $n+1$ étant réservé pour le référentiel d'inertie du corps.

Quant aux couples (resp. forces), ils (resp. elles) sont numéroté(e)s par le corps sur lequel ils (resp. le référentiel où elles) s'appliquent.

Il reste alors à introduire les notations suivantes :

- n le nombre de corps,

- pour le corps i,

	$\mathcal{S}_i$		le corps en lui-même
	$\mathcal{R}_i$		son repère de référence
	$\mathcal{R}_{i(n+1)}$		son référentiel d'inertie
	μ_i	désignent	sa masse
	$\mathcal{I}_i$		son opérateur d'inertie
	$\mathcal{P}_i$		six paramètres de position et d'orientation
	$\mathcal{Q}_i$		six paramètres cinématiques

- pour la liaison principale j, reliant le corps i au corps j avec $i < j$,

$\mathcal{L}_j$		la liaison en elle-même
$\mathcal{R}_{ij}$		le référentiel à l'articulation pour le premier corps
$\mathcal{R}_{ji}$	désignent	le référentiel à l'articulation pour le deuxième corps
$\mathcal{X}_j$		les paramètres articulaires
$\mathcal{V}_j$		les paramètres cinématiques articulaires

- pour la liaison complémentaire k, reliant le corps i au corps j avec $i < j$,

$\mathcal{L}_k$		la liaison en elle-même
$\mathcal{R}_{ik}$		le référentiel à l'articulation pour le premier corps
$\mathcal{R}_{jk}$	désignent	le référentiel à l'articulation pour le deuxième corps
$\mathcal{X}_k$		les paramètres articulaires
$\mathcal{V}_k$		les paramètres cinématiques articulaires

	$\mathcal{R}_{i(n+2)}, \mathcal{R}_{i(n+3)}, \ldots$		d'autres référentiels solidaires du corps $\mathcal{S}_i$
	$\mathcal{FE}_i$ ou $\mathcal{FE}_{ik}$		une force extérieure s'exerçant sur le référentiel $\mathcal{R}_i$ ou $\mathcal{R}_{ik}$ (ici k est quelconque)
et, le cas échéant,	$\mathcal{CE}_i$	désignent	un couple extérieur s'exerçant sur le corps $\mathcal{S}_i$
	$\mathcal{FL}_{ij}$		une force de liaison s'appliquant sur l'origine du référentiel $\mathcal{R}_{ij}$ à l'articulation
	$\mathcal{CL}_{ij}$		un couple de liaison s'appliquant sur le corps i concernant la liaison entre le corps $\mathcal{S}_i$ et le corps $\mathcal{S}_j$

Les seuls objets qui seront manipulés par le générateur des équations de la dynamique sont des objets mathématiques : scalaires, vecteurs, matrices, voire translations, rotations et déplacements. C'est pourquoi, pour chacun des objets qui ont été nommés ci-dessus, *LSD* va produire les variables correspondantes, s'il en existe. Ainsi, il sera défini :

- pour chaque repère de référence $\mathcal{R}_i$ d'un corps, le déplacement qui transforme $\mathcal{R}_0$ (le repère fixe associé à `earth`) en $\mathcal{R}_i$: `D_0_`i.
- pour chaque référentiel $\mathcal{R}_{ik}$ (ici k est quelconque), deux déplacements
  - le déplacement qui transforme $\mathcal{R}_0$ en $\mathcal{R}_{ik}$: `D_0_`ik,
  - le déplacement qui transforme le repère de référence $\mathcal{R}_i$ en $\mathcal{R}_{ik}$: `D_`i`_`ik,
- pour chaque opérateur d'inertie $\mathcal{I}_i$,
  - sa matrice dans la base du référentiel d'inertie $\mathcal{R}_{i(n+1)}$: `MIg_`i,
  - et sa matrice dans la base du repère fixe : `M0Ig_`i,
  - et les matrices dans les mêmes bases de l'opérateur d'inertie par rapport à l'origine O_i du repère de référence du corps, à savoir respectivement `MIo_`i et `M0Io_`i,
- pour les paramètres de position et d'orientation $\mathcal{P}_i$, les scalaires correspondant : `p`i`[1](t)`, ..., `p`i`[6](t)`,
- pour les paramètres cinématiques $\mathcal{Q}_i$, les scalaires correspondant : `q`i`[1](t)`, ..., `q`i`[6](t)`,
- pour les paramètres articulaires $\mathcal{X}_j$, les scalaires correspondant : `x`j`[1](t)`, ... (il y en a autant que le degré de liberté de la liaison)
- pour les paramètres cinématiques articulaires $\mathcal{V}_j$, les scalaires correspondant : `v`j`[1](t)`, ...(il y en a autant que le degré de liberté de la liaison)
- pour chaque force extérieure $\mathcal{FE}_i$ (resp. $\mathcal{FE}_{ik}$),
  - son vecteur : `FE_`i (resp. `FE_`ik),
  - l'expression de ce vecteur dans la base du référentiel attaché à son point d'application : `VFE_`i (resp. `VFE_`ik),
  - l'expression de ce vecteur dans la base du référentiel fixe : `V0FE_`i (resp. `V0FE_`ik),
- pour chaque couple extérieur $\mathcal{CE}_i$
  - son vecteur : `CE_`i,
  - l'expression de ce vecteur dans la base du référentiel attaché à son point d'application : `VCE_`i,
  - l'expression de ce vecteur dans la base du référentiel fixe : `V0CE_`i,

- pour chaque force de liaison $\mathcal{FL}_{ij}$,
  - son vecteur : FL_ij,
  - l'expression de ce vecteur dans la base du référentiel attaché à son point d'application, i.e. le référentiel $\mathcal{R}_{ij}$ de la liaison : VFL_ij,
  - l'expression de ce vecteur dans la base du référentiel fixe : V0FL_ij,
- pour chaque couple de liaison $\mathcal{CL}_{ij}$,
  - son vecteur : CL_ij,
  - l'expression de ce vecteur dans la base du référentiel du corps $\mathcal{S}_i$ attaché au point d'articulation, i.e. le référentiel $\mathcal{R}_{ij}$: VCL_ij,
  - l'expression de ce vecteur dans la base du référentiel fixe : V0CL_ij.

De plus, à chaque fois que *LSD* aura défini une variable D_α – α pouvant prendre n'importe laquelle des formes que l'on a rencontré plus haut – il définira les variables correspondantes en translation et en rotation :

- le vecteur de la translation : T_α,
- ses expressions dans les différents repères $\mathcal{R}_\beta$ – β étant soit un entier, soit un couple d'entier – concernés : VβT_α,
- la rotation : R_α,
- sa matrice dans les bases associées aux différents repères $\mathcal{R}_\beta$ concernés : MβT_α.

Une fois définies toutes ces variables, *LSD* écrit toutes les équations qui les relient, soit par leur nature même, soit à cause des propriétés qui ont été déclarées dans le programme de description du mécanisme considéré.

On reproduit ci-dessous, à titre d'exemple, le fichier fourni par la compilation du programme de description du satellite telle qu'on l'a montré dans la section (§3.2.1).

```
(* definition de la structure *)
> N := 4 ;
> BODIES := S_1 = 'satellite',
>           S_2 = 'bras_primaire',
>           S_3 = 'bras_secondaire',
>           S_4 = 'reflecteur';

(* definition des variables *)
...

(* affectations declarees *)
> V1T_1_15 := 0 ;
> M1T_1_15 := Id ;
> mu_2 := 0 ;
> mu_3 := 0 ;
> MIg_2 := 0 ;
> MIg_3 := 0 ;
> V4T_4_45 := [0,0,v4T_4_45_z] ;
> MIg_4 := 0 ;

(* affectations deduites *)
> V0T_1_15 = 0 ;
> M0Io_2 := 0 ;
```

```
> M0Io_3 := 0 ;
```

3.3 un algorithme de génération des équations du mouvement

On va décrire dans cette section un formalisme original, particulièrement bien adapté à la génération automatique des équations du mouvement de mécanismes.

Ce formalisme a été développé dans l'optique d'étudier le mouvement de mécanismes dont on ne connaît pas, a priori, le degré de liberté.

En ce qui concerne les mécanismes à structure arborescente, on a montré – théorème (1.41) du paragraphe (§1.2.3) – que leur degré de liberté est égal à six fois le nombre de corps mobiles diminué de la somme des degrés de contrainte des liaisons. L'intérêt de ce formalisme intervient donc surtout lors de l'étude du mouvement de mécanismes à structure bouclée – mais aussi lorsqu'un mécanisme à structure arborescente possède un grand nombre de corps, pour éviter d'avoir à calculer au préalable son degré de liberté.

Un autre intérêt de ce formalisme est de faire intervenir, comme on va le voir, essentiellement des grandeurs mécaniquement significatives.

Les équations du mouvement sont obtenues en suivant une approche mixte (Newton-Euler et Lagrange) de la modélisation dynamique dans la mesure où :

- les équations du mouvement d'un système mécanique quelconque sont obtenues par dérivation de la quantité de mouvement du système et de son moment cinétique, conformément à la méthode de Newton,
- les efforts inconnus des réactions aux liaisons sont exprimés à partir de multiplicateur de Lagrange, dans la variété de configurations,
- les équations du mouvement d'un système mécanique à structure bouclée font aussi intervenir des multiplicateurs de Lagrange.

Toutefois, la partie du travail d'élaboration des équations qui fait intervenir des dérivations a été exécutée une fois pour toute et n'apparaît plus lors de l'utilisation du formalisme.

Cela est rendu possible en scindant le procédé de modélisation dynamique d'un mécanisme en deux étapes :

i. les différents corps du système sont considérés comme indépendants, on écrit :
   - les équations différentielles du mouvement gouvernant un paramétrage maximal de la variété primitive de configurations,
   - les contraintes algébriques induites sur ces paramètres par les liaisons que l'on n'a pas considérées,

ii. dans le cas arborescent, on calcule explicitement la variété de configurations du mécanisme soumis aux liaisons, on crée un paramétrage minimal de cette variété, et on réécrit les équations précédentes dans ce nouveau paramétrage.

Lors de la première étape, les spécificités du système mécanique considéré n'interviennent pas directement dans l'élaboration des équations différentielles, c'est pourquoi l'on a pu effectuer les dérivations de la quantité de mouvement et du moment cinétique une fois pour toute en considérant un système mécanique générique.

De plus, dans le cas arborescent, l'élimination des termes concernant les efforts inconnus de réactions aux articulations est réalisée automatiquement par le calcul explicite de la variété de configurations, et par la création automatique d'un paramétrage minimal du système.

Si l'on considère un mécanisme à architecture bouclée, il n'est pas possible, tout au moins actuellement et dans le cas général, d'effectuer systématiquement la deuxième étape dans un temps et avec un volume de calcul raisonnables – ceci sera étudié plus en détail dans le prochain chapitre. On se ramène donc au mécanisme à architecture arborescente sous-jascent que l'on traitera comme n'importe quel mécanisme à architecture arborescente. Les liaisons complémentaires fourniront alors des équations algébriques supplémentaires sur les paramètres du mécanisme.

Ce formalisme a été implémenté dans le cadre du logiciel *GEMMES* développé pour l'"aerospatiale" – voir [Garnier et Rideau, 1989a] – et consacré à la génération des équation de la mécanique et à la manipulation d'expressions symboliques.

3.3.1 description du formalisme – établissement des équations

> Les noms choisis pour les paramètres généralisés et les vitesses généralisées peuvent apparaître inversés par rapport à l'usage de certains traités de mécanique, notamment en mécanique hamiltonnienne. Ils ont été néanmoins choisis ainsi pour rester cohérent avec le logiciel *GEMMES*.

On considère ici un mécanisme – au sens de la section (§1.1.5) – constitué de n corps et dont les laisons sont holonomes régulières. On étudie son mouvement par rapport à un référentiel inertiel R_0 d'origine O.

L'idée qui a prévalu à la conception de ce formalisme est d'obtenir, in fine, des équations différentielles qui puissent se traduire sous forme matricielle par l'expression – dans le cas d'un mécanisme à structure arborescente :

$$M(p).\dot{q} = S(p,q) + F(p,q) \tag{3.3.1.i}$$

expression dans laquelle :

- p est un vecteur de taille δ^l – on rapelle que δ^l désigne le nombre de degré de liberté du mécanisme étudié – dont les composantes sont les paramètres généralisés du mécanisme et forment une carte de sa variété de configurations.
- q (resp. $\dot{q}$) est un vecteur de taille δ^l dont les composantes sont les (resp. les dérivées par rapport au temps des) paramètres cinématiques généralisés du mécanisme et forment une carte de l'espace tangent à sa variété de configurations. Les $\dot{q}$ représentent une grandeur homogène à une accélération dite "accélération généralisée".
- $M(p)$ est une matrice $(\delta^l \times \delta^l)$ dépendant de p dite "de masse généralisée" car elle regroupe des termes de masse et d'inertie de chacun des n corps.
- $S(p,q)$ est un vecteur de taille δ^l dépendant de p et de q dit "de second membre" qui regroupe tous les termes indépendant des efforts exercés sur le mécanisme, qui ne sont pas facteurs de $\dot{q}$. Il inclut notamment ce que l'on appelle classiquement les forces centrifuges et de Coriolis, dues aux accélérations normales et d'entraînement.
- $F(p,q)$ est un vecteur de taille δ^l dépendant de p et de q dit "des forces généralisées" qui réunit les termes dépendant des efforts exercés sur le mécanisme.

Si l'on considère un mécanisme à structure bouclée, on cherchera à obtenir une expression similaire :

$$M(p).\dot{q} = S(p,q) + F(p,q) + {}^T\left(\Delta_{\mathcal{V}}(p)\right).\Lambda \tag{3.3.1.ii}$$

ainsi que les équations algébriques dites "de contrainte" – voir (§1.3.3) :

$$\Delta_{\mathcal{V}}(p).q = 0$$

Dans ces expressions, si on désigne par $\underline{\mathcal{S}}$ le système mécanique arborescent sous-jascent au mécanisme $\mathcal{S}$, et par $\underline{\delta^l}$ son degré de liberté, $M(p)$ est une matrice $(\underline{\delta^l} \times \underline{\delta^l})$, p, q, $\dot{q}$, $S(p,q)$, $F(p,q)$ sont des vecteurs de taille $\underline{\delta^l}$, et

- $\Delta_{\mathcal{V}}(p)$ est une matrice $(\sum \delta^c \times \underline{\delta^l})$ – où $\sum \delta^c$ désigne la somme des degrés de contrainte des liaisons complémentaires – c'est la matrice jacobienne des équations de liaisons.
- Λ est un vecteur de taille $\sum \delta^c$ dont les composantes sont des variables dépendantes du temps, les multiplicateurs de Lagrange.

équations des corps libres

Dans un premier temps, on regarde les corps comme libres dans l'espace. A chaque corps $\mathcal{S}_i$, on attache un référentiel $\mathcal{R}_i$ dont la position à l'instant t est le repère* R_i ayant pour origine le point O_i.

Ainsi, pour chaque corps $\mathcal{S}_i$ on repère sa configuration à l'instant t par six paramètres de position et d'orientation, qui sont les composantes du vecteur $\mathfrak{p}_i$, les trois premières étant les coordonnées du vecteur $\overrightarrow{OO_i}$ dans le repère R_0, les trois autres définissant l'orientation de $\mathcal{S}_i$ – il s'agit, par exemple, des trois angles d'Euler ou de Bryant.

On utilise aussi six paramètres cinématiques, les composantes du vecteur $\mathfrak{q}_i$.

- Les trois premiers paramètres cinématiques sont les coordonnées de la vitesse en translation du point O_i par rapport au repère R_0 – que l'on notera v_i exprimées dans ce repère. On a donc :

$$\begin{pmatrix} \mathfrak{q}_{i1} \\ \mathfrak{q}_{i2} \\ \mathfrak{q}_{i3} \end{pmatrix} = \begin{pmatrix} \dot{\mathfrak{p}}_{i1} \\ \dot{\mathfrak{p}}_{i2} \\ \dot{\mathfrak{p}}_{i3} \end{pmatrix} \tag{3.3.1.iii}$$

- Les trois derniers sont les coordonnées du vecteur vitesse instatanée de rotation du corps $\mathcal{S}_i$ – que l'on notera ω_i – exprimées par rapport au repère R_0. Ce ne sont donc pas exactement les dérivées des paramètres de position correspondant, mais ils leur sont reliés par une égalité de la forme :

$$\begin{pmatrix} \mathfrak{q}_{i4} \\ \mathfrak{q}_{i5} \\ \mathfrak{q}_{i6} \end{pmatrix} = \mathfrak{P}_i(\mathfrak{p}_i) \cdot \begin{pmatrix} \dot{\mathfrak{p}}_{i4} \\ \dot{\mathfrak{p}}_{i5} \\ \dot{\mathfrak{p}}_{i6} \end{pmatrix}$$

La matrice $\mathfrak{P}_i(\mathfrak{p}_i)$ qui dépend de la représentation des rotations choisie, est calculée d'après la propriété suivante – si on note $\bar{\rho}_i$ la matrice de la rotation qui transforme R_0 en R_i :

$$\text{pour tout vecteur } X, \qquad \left.\frac{d}{dt}\right|_R (\bar{\rho}_i.X) = \omega_i \wedge \bar{\rho}_i.X$$

Si l'on décide d'utiliser les angles de Bryant – voir (§1.2.1) – et que l'on pose :

$$\begin{cases} \mathfrak{p}_{i4} = {}^x\Phi_i \\ \mathfrak{p}_{i5} = {}^y\Phi_i \\ \mathfrak{p}_{i6} = {}^z\Phi_i \end{cases}$$

*Aucune confusion n'étant possible, on ommetra l'indice t accompagnant habituellement les grandeurs dépendant du temps.

$$\text{alors :} \qquad \mathfrak{P}_i(\mathfrak{p}_i) = \begin{pmatrix} 1 & 0 & \sin(\mathfrak{p}_{i5}) \\ 0 & \cos(\mathfrak{p}_{i4}) & -\sin(\mathfrak{p}_{i4})\cos(\mathfrak{p}_{i5}) \\ 0 & \sin(\mathfrak{p}_{i4}) & \cos(\mathfrak{p}_{i4})\cos(\mathfrak{p}_{i5}) \end{pmatrix}$$

On va établir, pour chaque corps $\mathcal{S}_i$, les équations de Newton-Euler du mouvement qui gouvernent $\mathfrak{p}_i$ et $\mathfrak{q}_i$, à savoir :

- la loi fondamentale de la dynamique :

$$\mu_i \gamma(G_i) = F_{\text{ext}} \tag{3.3.1.iv}$$

- le théorème du moment cinétique :

$$\mathcal{I}_i(G_i)\dot{\omega}_i + \omega_i \wedge \mathcal{I}_i(G_i)\omega_i = C_{\text{ext}}(G_i) \tag{3.3.1.v}$$

On rappelle que dans ces équations, on désigne par :

- μ_i la masse du corps $\mathcal{S}_i$, G_i la position de son centre d'inertie, $\mathcal{I}_i$ son inertie par rapport au point G_i,
- F_{ext}, la résultante du torseur des efforts extérieurs,
- $C_{\text{ext}}(G_i)$, le moment, au point G_i du torseur des efforts extérieurs.

Si on réécrit les équations (3.3.1.iv) et (3.3.1.v) en faisant apparaître l'origine O_i du repère de référence attaché au corps, on obtient[†] :

$$\left\{ \begin{array}{l} \mu \left(\gamma(O_i) + \dot{\omega}_i \wedge \overrightarrow{O_iG_i} + \omega_i \wedge \left(\omega_i \wedge \overrightarrow{O_iG_i} \right) \right) = F_{\text{ext}} \\ \mathcal{I}_i(O_i)\dot{\omega}_i + \omega_i \wedge \mathcal{I}_i(O_i)\omega_i + \mu\ \widetilde{O_iG_i}\widetilde{O_iG_i}\ \dot{\omega}_i + \omega_i \wedge \left(\mu\ \widetilde{O_iG_i}\widetilde{O_iG_i}\ \omega_i \right) = C_{\text{ext}}(G_i) \end{array} \right. \tag{3.3.1.vi}$$

puisque :

$$O_i = G_i + \overrightarrow{G_iO_i} \quad \text{donc} \quad v(G_i) = v(O_i) + \omega_i \wedge \overrightarrow{O_iG_i} \quad \text{et} \quad \gamma(G_i) = \gamma(O_i) + \dot{\omega}_i \wedge \overrightarrow{O_iG_i} + \omega_i \wedge \left(\omega_i \wedge \overrightarrow{O_iG_i} \right)$$

et que :

$$\mathcal{I}_i(G_i) = \mathcal{I}_i(O_i) + \mu\ \widetilde{O_iG_i}\widetilde{O_iG_i}$$

Si de plus, on utilise le fait que les efforts sont représentés par un torseur, c'est à dire que l'on a :

$$C_{\text{ext}}(O_i) = C_{\text{ext}}(G_i) + \overrightarrow{O_iG_i} \wedge F_{\text{ext}}$$

alors on peut combiner les équations (3.3.1.vi) pour obtenir :

$$\left\{ \begin{array}{l} \mu \left(\gamma(O_i) + \dot{\omega}_i \wedge \overrightarrow{O_iG_i} + \omega_i \wedge \left(\omega_i \wedge \overrightarrow{O_iG_i} \right) \right) = F_{\text{ext}} \\ \mu\ \overrightarrow{O_iG_i}\gamma(O_i) + \mathcal{I}_i(O_i)\dot{\omega}_i + \omega_i \wedge \mathcal{I}_i(O_i)\omega_i = C_{\text{ext}}(O_i) \end{array} \right. \tag{3.3.1.vii}$$

en utilisant la propriété remarquable du produit vectoriel :

$$\text{pour tous vecteurs } X \text{ et } Y, \qquad X \wedge (Y \wedge (Y \wedge X)) = -Y \wedge (X \wedge (X \wedge Y))$$

[†]Dans tout ce qui suit, on note $\tilde{X}$ (resp. $\widetilde{MN}$) l'opérateur linéaire associé au vecteur X (resp. $\overrightarrow{MN}$) tel que $\tilde{X}Y = X \wedge Y$ (resp. $\widetilde{MN}\ Y = \overrightarrow{MN} \wedge Y$) pour tout vecteur Y.

Les équations (3.3.1.vii) peuvent s'écrire – en exprimant les différents vecteurs et les différents opérateurs par rapport au repère R_i – matriciellement par blocs :

$$\left(\begin{array}{ccc|c} \mu & 0 & 0 & \\ 0 & \mu & 0 & -\mu\ \widetilde{O_iG_i} \\ 0 & 0 & \mu & \\ \hline \multicolumn{3}{c|}{\mu\ \widetilde{O_iG_i}} & \mathcal{I}_i(O_i) \end{array}\right) \left(\begin{array}{c} \dot{\mathfrak{q}}_{i1} \\ \dot{\mathfrak{q}}_{i2} \\ \dot{\mathfrak{q}}_{i3} \\ \hline \dot{\mathfrak{q}}_{i4} \\ \dot{\mathfrak{q}}_{i5} \\ \dot{\mathfrak{q}}_{i6} \end{array}\right) = \left(\begin{array}{c} -\tilde{\omega}_i\ \tilde{\omega}_i\ \overrightarrow{O_iG_i} \\ \hline -\tilde{\omega}_i\ \mathcal{I}_i(O_i)\omega_i \end{array}\right) + \left(\begin{array}{c} F_{\text{ext}} \\ \hline C_{\text{ext}}(O_i) \end{array}\right) \qquad \text{(3.3.1.viii)}$$

soit, avec des notations évidentes :

$$\mathfrak{M}_i(\mathfrak{p}_i)\dot{\mathfrak{q}}_i = \mathfrak{S}_i(\mathfrak{p}_i, \mathfrak{q}_i) + \mathfrak{F}_i(\mathfrak{p}_i, \mathfrak{q}_i)$$

Si l'on regroupe maintenant tous les corps considérés comme libres en un seul système mécanique, on obtient les équations par construction :

$$\left(\begin{array}{cccc} \mathfrak{M}_1(\mathfrak{p}_1) & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathfrak{M}_2(\mathfrak{p}_2) & \ddots & \vdots \\ \vdots & \ddots & \ddots & \mathbf{0} \\ \mathbf{0} & \dots & \mathbf{0} & \mathfrak{M}_n(\mathfrak{p}_n) \end{array}\right) \left(\begin{array}{c} \dot{\mathfrak{q}}_1 \\ \dot{\mathfrak{q}}_2 \\ \vdots \\ \dot{\mathfrak{q}}_n \end{array}\right) = \left(\begin{array}{c} \mathfrak{S}_1(\mathfrak{p}_1, \mathfrak{q}_1) \\ \mathfrak{S}_2(\mathfrak{p}_2, \mathfrak{q}_2) \\ \vdots \\ \mathfrak{S}_n(\mathfrak{p}_n, \mathfrak{q}_n) \end{array}\right) + \left(\begin{array}{c} \mathfrak{F}_1(\mathfrak{p}_1, \mathfrak{q}_1)) \\ \mathfrak{F}_2(\mathfrak{p}_2, \mathfrak{q}_2) \\ \vdots \\ \mathfrak{F}_n(\mathfrak{p}_n, \mathfrak{q}_n) \end{array}\right)$$

ou, pour abbrévier :

$$\mathfrak{M}(\mathfrak{p})\dot{\mathfrak{q}} = \mathfrak{S}(\mathfrak{p}, \mathfrak{q}) + \mathfrak{F}(\mathfrak{p}, \mathfrak{q}) \qquad \text{(3.3.1.ix)}$$

équations avec multiplicateurs

On va maintenant considérer les liaisons existantes entre les corps du système et on va être amené à introduire des "multiplicateurs de Lagrange" pour représenter les composantes inconnues des efforts de liaisons.

On suppose qu'il existe une liaison ℓ_{ij} de degré de contrainte $\delta^c(\ell_{ij})$ entre les corps $\mathcal{S}_i$ et $\mathcal{S}_j$ du système.

On a vu – au paragraphe (§1.2.2) – que l'on peut associer, au moins localement, à cette liaison, une submersion $f_{(ij)}$ de la variété primitive de configurations $\mathcal{V}'(\mathcal{S}_i \sqcup \mathcal{S}_j)$ du système restreint au deux corps liés vers $\mathbb{R}^{\delta^c(\ell_{ij})}$.

Cela revient à traduire la liaison par $\delta^c(\ell_{ij})$ équations contraignant les paramètres du système. Ces équations sont de la forme $f_{(ij)k}(\mathfrak{p}_i, \mathfrak{p}_j)$ avec $k = 1, \dots, \delta^c(\ell_{ij})$. Elles permettent d'obtenir, par différentiation, $\delta^c(\ell_{ij})$ équations indépendantes contraignant linéairement les $\dot{\mathfrak{p}}_i$ et les $\dot{\mathfrak{p}}_j$:

$$\left\{\begin{array}{l} \displaystyle\sum_{k=1}^{6} \frac{\partial f_{(ij)1}}{\partial \mathfrak{p}_{ik}}\dot{\mathfrak{p}}_{ik} + \sum_{k=1}^{6} \frac{\partial f_{(ij)1}}{\partial \mathfrak{p}_{jk}}\dot{\mathfrak{p}}_{jk} = 0 \\ \qquad \dots \\ \displaystyle\sum_{k=1}^{6} \frac{\partial f_{(ij)\delta^c}}{\partial \mathfrak{p}_{ik}}\dot{\mathfrak{p}}_{ik} + \sum_{k=1}^{6} \frac{\partial f_{(ij)\delta^c}}{\partial \mathfrak{p}_{jk}}\dot{\mathfrak{p}}_{jk} = 0 \end{array}\right. \qquad \text{(3.3.1.x)}$$

ce que l'on écrira matriciellement :

$$\left(\begin{array}{cc} \mathfrak{L}^i_{(ij)}(\mathfrak{p}_i,\mathfrak{p}_j) & \mathfrak{L}^j_{(ij)}(\mathfrak{p}_i,\mathfrak{p}_j) \end{array} \right) \left(\begin{array}{c} \dot{\mathfrak{p}}_i \\ \dot{\mathfrak{p}}_j \end{array} \right) = 0$$

ou en faisant intervenir les $\mathfrak{q}_i$ et les $\mathfrak{q}_j$ dans la mesure où la matrice $\mathfrak{P}_i(\mathfrak{p}_i)$ est inversible – ce qui est le cas presque partout :

$$\left(\begin{array}{cc} \mathfrak{D}^i_{(ij)}(\mathfrak{p}_i,\mathfrak{p}_j) & \mathfrak{D}^j_{(ij)}(\mathfrak{p}_i,\mathfrak{p}_j) \end{array} \right) \left(\begin{array}{c} \mathfrak{q}_i \\ \mathfrak{q}_j \end{array} \right) = 0$$

avec
$$\left\{ \begin{array}{l} \mathfrak{D}^i_{(ij)}(\mathfrak{p}_i,\mathfrak{p}_j) = \mathfrak{L}^i_{(ij)}(\mathfrak{p}_i,\mathfrak{p}_j) \left(\begin{array}{ccc|c} 1 & 0 & 0 & \\ 0 & 1 & 0 & \mathbf{0} \\ 0 & 0 & 1 & \\ \hline & \mathbf{0} & & (\mathfrak{P}_i(\mathfrak{p}_i))^{-1} \end{array} \right) \\ \\ \mathfrak{D}^j_{(ij)}(\mathfrak{p}_i,\mathfrak{p}_j) = \mathfrak{L}^j_{(ij)}(\mathfrak{p}_i,\mathfrak{p}_j) \left(\begin{array}{ccc|c} 1 & 0 & 0 & \\ 0 & 1 & 0 & \mathbf{0} \\ 0 & 0 & 1 & \\ \hline & \mathbf{0} & & (\mathfrak{P}_j(\mathfrak{p}_j))^{-1} \end{array} \right) \end{array} \right.$$

La théorie Lagrangienne établit que la puissance développée par les composantes inconnues des efforts de liaison est nulle dans tout mouvement, réel ou virtuel, compatible avec cette liaison.

On démontre alors facilement que la puissance développée par ces composantes au cours d'un mouvement quelconque décrit par $(\dot{\mathfrak{p}}_i, \dot{\mathfrak{p}}_j)$ s'exprime sous la forme :

$$\left(\begin{array}{cccccc} \dot{\mathfrak{p}}_{i1} & \dots & \dot{\mathfrak{p}}_{i6} & \dot{\mathfrak{p}}_{j1} & \dots & \dot{\mathfrak{p}}_{j6} \end{array} \right) \left(\begin{array}{c} \sum_{k=1}^{\delta^c} \Lambda_{(ij)k} \dfrac{\partial f_{(ij)k}}{\partial \mathfrak{p}_{i1}} \\ \vdots \\ \sum_{k=1}^{\delta^c} \Lambda_{(ij)k} \dfrac{\partial f_{(ij)k}}{\partial \mathfrak{p}_{i6}} \\ \\ \sum_{k=1}^{\delta^c} \Lambda_{(ij)k} \dfrac{\partial f_{(ij)k}}{\partial \mathfrak{p}_{j1}} \\ \vdots \\ \sum_{k=1}^{\delta^c} \Lambda_{(ij)k} \dfrac{\partial f_{(ij)k}}{\partial \mathfrak{p}_{j6}} \end{array} \right) \qquad (3.3.1.xi)$$

dans laquelle les $\Lambda_{(ij)k}$ sont des inconnues appelées "multiplicateurs de Lagrange".

En effet, on peut regarder la puissance développée par les composantes inconnues des efforts de liaison comme une forme linéaire $\Pi_{(ij)}$ définie sur l'espace tangent à la variété primitive de configurations du système $\mathcal{S}_i \sqcup \mathcal{S}_j$ qui, à un mouvement $(\dot{\mathfrak{p}}_i, \dot{\mathfrak{p}}_j)$ associe la puissance développée au cours de ce mouvement $< \Pi_{(ij)} \mid (\dot{\mathfrak{p}}_i, \dot{\mathfrak{p}}_j) >$.

Dire que cette puissance est nulle si le mouvement considéré est compatible avec la liaison, c'est dire que si les $\delta^c(\ell_{ij})$ contraintes (3.3.1.x) – qui sont aussi des formes linéaires définies sur l'espace tangent à la variété primitive de configurations du système – sont nulles alors $\Pi_{(ij)}$ est nulle.

Et comme la dimension de l'espace tangent est finie, cela entraîne, par dualité, que $\Pi_{(ij)}$ appartient, en tant que forme linéaire, au sous-espace vectoriel dual engendré par les formes linéaires de contrainte.

Les multiplicateurs de Lagrange $\Lambda_{(ij)k}$ ne sont autres que les coefficients de la combinaison linéaire exprimant $\Pi_{(ij)}$ en fonction des formes linéaires de contrainte. ◇

On peut exprimer la puissance (3.3.1.xi) sous forme matricielle :

$$\left(\begin{array}{cc} {}^T\dot{\mathfrak{p}}_i & {}^T\dot{\mathfrak{p}}_j \end{array} \right) \left(\begin{array}{c} {}^T\Big(\mathfrak{L}^i_{(ij)}(\mathfrak{p}_i, \mathfrak{p}_j)\Big) \\ \\ {}^T\Big(\mathfrak{L}^j_{(ij)}(\mathfrak{p}_i, \mathfrak{p}_j)\Big) \end{array} \right) \left(\begin{array}{c} \Lambda_{(ij)} \end{array} \right)$$

ou, en faisant intervenir les paramètres cinématiques :

$$\left(\begin{array}{cc} {}^T\dot{\mathfrak{q}}_i & {}^T\dot{\mathfrak{q}}_j \end{array} \right) \left(\begin{array}{c} {}^T\Big(\mathfrak{D}^i_{(ij)}(\mathfrak{p}_i, \mathfrak{p}_j)\Big) \\ \\ {}^T\Big(\mathfrak{D}^j_{(ij)}(\mathfrak{p}_i, \mathfrak{p}_j)\Big) \end{array} \right) \left(\begin{array}{c} \Lambda_{(ij)} \end{array} \right) \qquad \text{(3.3.1.xii)}$$

Or, mécaniquement, la puissance des composantes inconnues des efforts de la liaison ℓ_{ij} développée au cours d'un mouvement quelconque du système $\mathcal{S}_i \sqcup \mathcal{S}_j$, est définie par :

$$v(G_i) \;.\; F_{\ell_{ij}i} + \omega_i \;.\; C_{\ell_{ij}i} + v(G_j) \;.\; F_{\ell_{ij}j} + \omega_j \;.\; C_{\ell_{ij}j} \qquad \text{(3.3.1.xiii)}$$

Donc, si l'on ne regarde que le corps $\mathcal{S}_i$, les efforts extérieurs correspondant aux composantes inconnues des efforts de la liaison avec le corps $\mathcal{S}_j$ s'expriment, par identification entre les expressions (3.3.1.xii) et (3.3.1.xiii), par le vecteur :

$${}^T\Big(\mathfrak{D}^i_{(ij)}(\mathfrak{p}_i, \mathfrak{p}_j)\Big) \;\; \big(\Lambda_{(ij)}\big)$$

et les équations du mouvement de ce corps soumis à la liaison ℓ_{ij} s'écrivent :

$$\mathfrak{M}_i(\mathfrak{p}_i)\dot{\mathfrak{q}}_i = \mathfrak{S}_i(\mathfrak{p}_i, \mathfrak{q}_i) + \mathfrak{F}_i(\mathfrak{p}_i, \mathfrak{q}_i) + {}^T\Big(\mathfrak{D}^i_{(ij)}(\mathfrak{p}_i, \mathfrak{p}_j)\Big) \;\; \big(\Lambda_{(ij)}\big)$$

Si l'on considère maintenant toutes les liaisons présentes dans l'architecture du système, on peut obtenir, par construction, les équations de son mouvement sous la forme :

$$\mathfrak{M}(\mathfrak{p})\dot{\mathfrak{q}} = \mathfrak{S}(\mathfrak{p}, \mathfrak{q}) + \mathfrak{F}(\mathfrak{p}, \mathfrak{q}) + {}^T(\mathfrak{D}(\mathfrak{p})) \;\; (\Lambda) \qquad \text{(3.3.1.xiv)}$$

où $\mathfrak{M}$, $\mathfrak{S}$ et $\mathfrak{F}$ sont définis comme précédemment – dans le cas où les corps étaient considérés comme libres, et où :

- $\mathit{\Lambda}$ est le vecteur colonne des multiplicateurs de Lagrange :

$$\mathit{\Lambda} = \begin{pmatrix} \mathit{\Lambda}_{\ell_1 1} \\ \vdots \\ \mathit{\Lambda}_{\ell_1 \delta^c(\ell_1)} \\ \\ \vdots \\ \\ \mathit{\Lambda}_{\ell_\nu 1} \\ \vdots \\ \mathit{\Lambda}_{\ell_\nu \delta^c(\ell_\nu)} \end{pmatrix}$$

  en renumérotant les liaisons de 1 à ν,

- ${}^T(\mathfrak{D}(\mathfrak{p}))$ est construite par blocs à partir des ${}^T\Big(\mathfrak{D}_{(ij)}^{i \text{ ou } j}(\mathfrak{p})\Big)$:

  elle possède autant de colonnes de blocs qu'il y a de liaisons dans le système, et autant de lignes de blocs qu'il y a de corps dans le système, le bloc correpondant à la i-ème ligne et à la k-ième colonne étant :
  - ${}^T\Big(\mathfrak{D}_{(ij)}^{i}(\mathfrak{p})\Big)$ si la k-ième liaison relie le corps $\mathcal{S}_i$ au corps $\mathcal{S}_j$,
  - nul si cette liaison ne fait pas intervenir le corps $\mathcal{S}_i$.

A ces équations différentielles gouvernant les paramètres $\mathfrak{p}$ et $\mathfrak{q}$, il faut, bien entendu ajouter les les équations algébriques dite "de contrainte" que ces paramètres doivent satisfaire à tout instant t lors du mouvement réel et qui s'écrivent :

$$\mathfrak{D}(\mathfrak{p}) \; . \; \begin{pmatrix} \\ \mathfrak{q} \\ \\ \end{pmatrix} = 0$$

équations sans multiplicateurs

Pour avoir les équations du mouvement d'un mécanisme sous l'une des deux formes voulues – (3.3.1.i) dans le cas d'un mécanisme à structure arborescente, (3.3.1.ii) dans le cas d'un mécanisme à structure bouclé – on est maintenant amené à éliminer, au moins partiellement les multiplicateurs de Lagrange, et changer de paramétrage.

On considère un paramétrage minimal de la variété de configurations du mécanisme (ou du système mécanique arborescent sous-jascent, si la structure du mécanisme étudié est bouclée).

Soient donc p le vecteur donc les composantes sont les nouveaux paramètres de position et q le vecteur dont les composantes sont les nouveaux paramètres cinématiques.

On admet que l'on peut exprimer les anciens paramètres $\mathfrak{p}$ et $\mathfrak{q}$ en fonction des nouveaux p et q par les relations suivantes :

- pour $i = 1, \ldots, n, \quad k = 1, \ldots, 6, \qquad \mathfrak{p}_{ik} = \psi_{ik}(p)$
- $$\begin{pmatrix} \mathfrak{q}_1 \\ \vdots \\ \mathfrak{q}_n \end{pmatrix} = \Psi(p) \begin{pmatrix} q_1 \\ \vdots \\ q_n \end{pmatrix}$$

d'où l'on déduit :

$$\begin{pmatrix} q_1 \\ \vdots \\ q_n \end{pmatrix} = P(p) \begin{pmatrix} \dot{p}_1 \\ \vdots \\ \dot{p}_n \end{pmatrix}$$

à l'aide du diagramme commutatif suivant[‡] :

$$\begin{array}{ccccc} \mathfrak{p} & \xrightarrow{\frac{d}{dt}} & \dot{\mathfrak{p}} & \xrightarrow{.\mathfrak{P}(\mathfrak{p})} & \mathfrak{q} \\ {\scriptstyle\psi}\uparrow & & {\scriptstyle\frac{\partial\psi}{\partial p}}\uparrow & & \uparrow{\scriptstyle .\Psi(p)} \\ p & \xrightarrow[\frac{d}{dt}]{} & \dot{p} & \xrightarrow[.P(p)]{} & q \end{array}$$

On verra plus loin un algorithme qui permet d'obtenir les nouveaux paramètres et les relations à partir de la structure du mécanisme étudié et de la nature des liaisons rencontrées.

Dans le cas arborescent, puisque un paramétrage minimal de la variété de configurations, par définition, prend en compte toutes les liaisons, les équations de contrainte que l'on peut réécrire :

$$\mathfrak{D}(\mathfrak{p})\Psi(p) \ . \ q = 0$$

sont vérifiées quelles que soient les valeurs des composantes de q et donc :

$$\mathfrak{D}(\mathfrak{p})\Psi(p) = 0$$

On peut donc simplifier sensiblement les équations du mouvement (3.3.1.xiv) en utilisant ψ et $\Psi(p)$. On obtient successivement :

$$^T(\Psi(p))\,\mathfrak{M}(\mathfrak{p})\dot{\mathfrak{q}} = {}^T(\Psi(p))\,\mathfrak{S}(\mathfrak{p},\mathfrak{q}) + {}^T(\Psi(p))\,\mathfrak{F}(\mathfrak{p},\mathfrak{q})$$

$$\text{puis} \qquad {}^T(\Psi(p))\,\mathfrak{M}(\psi(p))\left(\Psi(p)\dot{q} + \dot{\Psi}(p,\dot{p})\right) = {}^T(\Psi(p))\,\mathfrak{S}(\psi(p),\Psi(p)q) + {}^T(\Psi(p))\,\mathfrak{F}(\psi(p),\Psi(p)q)$$

ce qui, avec les notations adéquates, s'écrit – dans la mesure où $P(p)$ est inversible :

$$M(p)\ \dot{q} = S(p,q) + F(p,q) \tag{3.3.1.xv}$$

Dans le cas bouclé, on ne sait pas toujours – comme on le verra au chapitre suivant – trouver un paramétrage minimal de tout le système. On détermine alors un système mécanique arborescent sous-jacent au mécanisme étudié et on distingue parmi les liaisons, celles qui font partie du système sous-jacent des liaisons dites complémentaires.

On peut alors scinder la matrice de contrainte et le vecteur des multiplicateurs de Lagrange en deux blocs, l'un concernant exclusivement les liaisons complémentaires. Par un jeu de notation, et en prenant soin de bien dimensionner les nouvelles matrices de contrainte quitte à ajouter des blocs nuls, on écrira les équations du mouvement du mécanisme et ses équations de contrainte :

$$\begin{cases} \mathfrak{M}(\mathfrak{p})\dot{\mathfrak{q}} = \mathfrak{S}(\mathfrak{p},\mathfrak{q}) + \mathfrak{F}(\mathfrak{p},\mathfrak{q}) + {}^T(\mathfrak{D}_{\text{arb}}(\mathfrak{p}))\ (\Lambda_{\text{arb}}) + {}^T(\mathfrak{D}_{\text{cmp}}(\mathfrak{p}))\ (\Lambda_{\text{cmp}}) \\ \begin{pmatrix} \mathfrak{D}_{\text{arb}}(\mathfrak{p}) \\ \mathfrak{D}_{\text{cmp}}(\mathfrak{p}) \end{pmatrix} \begin{pmatrix} \mathfrak{q} \end{pmatrix} = 0 \end{cases} \tag{3.3.1.xvi}$$

[‡]Dans ce diagramme, on a mis $\mathfrak{P}(\mathfrak{p})$ pour désigner en fait $\begin{pmatrix} \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathfrak{P}(\mathfrak{p}) \end{pmatrix}$.

les indices $_{\text{arb}}$ et $_{\text{cmp}}$ se rapportant respectivement aux liaisons du système mécanique arborescent sous-jacent et aux liaisons complémentaires

La connaissance d'un paramétrage minimal (p, q) de la variété de configurations du système mécanique arborescent sous-jacent nous fournit alors, comme précédemment, des relations ψ_{arb} (resp. une matrice Ψ_{arb}) reliant les anciens paramètres de position (resp. cinématiques) aux nouveaux.

Et la matrice Ψ_{arb} possède bien entendu la propriété :

$$\mathfrak{D}_{\text{arb}}(\mathfrak{p})\Psi_{\text{arb}}(p) = 0$$

En utilisant ψ_{arb} et Ψ_{arb}, on peut alors réécrire les équations (3.3.1.xvi) successivement en :

$$\begin{cases} {}^T(\Psi_{\text{arb}}(p))\,\mathfrak{M}(\mathfrak{p})\dot{\mathfrak{q}} = {}^T(\Psi_{\text{arb}}(p))\,\mathfrak{S}(\mathfrak{p},\mathfrak{q}) + {}^T(\Psi_{\text{arb}}(p))\,\mathfrak{F}(\mathfrak{p},\mathfrak{q}) + {}^T(\Psi_{\text{arb}}(p))\,{}^T(\mathfrak{D}_{\text{cmp}}(\mathfrak{p}))\ \ (\Lambda_{\text{cmp}}) \\ \mathfrak{D}_{\text{cmp}}(\mathfrak{p})\ .\ \begin{pmatrix} \mathfrak{q} \end{pmatrix} = 0 \end{cases}$$

puis en :

$$\begin{cases} {}^T(\Psi_{\text{arb}}(p))\,\mathfrak{M}(\psi_{\text{arb}}(p))\left(\Psi_{\text{arb}}(p)\dot{q} + \dot{\Psi}_{\text{arb}}(p,\dot{p})\right) = \\ \qquad {}^T(\Psi_{\text{arb}}(p))\,\mathfrak{S}(\psi_{\text{arb}}(p), \Psi_{\text{arb}}(p)q) + {}^T(\Psi_{\text{arb}}(p))\,\mathfrak{F}(\psi_{\text{arb}}(p), \Psi_{\text{arb}}(p)q) \\ \qquad\quad + {}^T(\Psi_{\text{arb}}(p))\,{}^T(\mathfrak{D}_{\text{cmp}}(\psi_{\text{arb}}(p)))\ \ (\Lambda_{\text{cmp}}) \\ \\ \mathfrak{D}_{\text{cmp}}(\psi_{\text{arb}}(p))\ .\ \Psi_{\text{arb}}(p)\ .\ \begin{pmatrix} q \end{pmatrix} = 0 \end{cases}$$

ce qui nous permet, avec les bonnes notations, d'obtenir les équations du mouvement sous la forme souhaitée :

$$\begin{cases} M(p).\dot{q} = S(p,q) + F(p,q) + {}^T\left(\Delta_{\mathcal{V}}(p)\right).\Lambda \\ \Delta_{\mathcal{V}}(p)\ .\ q = 0 \end{cases} \tag{3.3.1.xvii}$$

3.3.2 algorithme de génération des équations

Il reste à présenter l'algorithme qui permet d'engendrer automatiquement, pour un mécanisme quelconque, les équations de la dynamique qui modélisent son mouvement en utilisant le formalisme décrit ci-dessus.

On considère que la description du mécanisme étudié a fourni les renseignements nécessaires sur son architecture, ses liaisons, les caractéristiques cinétiques de chacun des corps et les forces qui s'appliquent en différents points de ce mécanisme.

Schématiquement, on peut alors écrire cet algorithme comme suit :

i. si la structure du mécanisme est bouclée,
   - déterminer les liaisons complémentaires,
   - introduire Λ_{cmp} et établir l'expression de $\mathfrak{D}_{\text{cmp}}(\mathfrak{p})$,
   - remplacer le mécanisme par le système mécanique arborescent sous-jacent,

ii. établir l'expression de la matrice de masse $\mathfrak{M}(\mathfrak{p})$ en fonction des paramètres initiaux de position,

iii. établir l'expression du vecteur de second membre $\mathfrak{S}(\mathfrak{p},\mathfrak{q})$, en fonction des paramètres initiaux de position et cinématiques,

iv. établir l'expression du vecteur des forces généralisées en fonction des paramètres initiaux de position et cinématiques,

v. établir l'expression de la matrice de contrainte $\mathfrak{D}(\mathfrak{p})$ en fonction des paramètres initiaux de position, et introduire les multiplicateurs de Lagrange Λ,

vi. éliminer les multiplicateurs de Lagrange, et pour cela :

- déterminer un paramétrage minimal de la variété de configurations du système,
- exprimer les relations existant entre le nouveau et l'ancien paramétrage tant en ce qui concerne les paramètres de position qu'en ce qui concerne les paramètres cinématiques, i.e. les relations ψ et la matrice Ψ,
- multiplier terme à terme les équations :

$$\mathfrak{M}(\mathfrak{p})\dot{\mathfrak{q}} = \mathfrak{S}(\mathfrak{p},\mathfrak{q}) + \mathfrak{F}(\mathfrak{p},\mathfrak{q}) + {}^{T}(\mathfrak{D}(\mathfrak{p}))\ \ (\Lambda)$$

par la transposée de Ψ,

vii. exprimer les équations en fonction du paramétrage minimal,

viii. si le mécanisme initial était bouclé,

- ajouter aux équations le terme

$${}^{T}(\Psi(p))\,{}^{T}(\mathfrak{D}_{\mathrm{cmp}}(\psi(p)))\ \ (\Lambda_{\mathrm{cmp}})$$

- et établir les équations de contrainte en calculant la matrice $\Delta_{\mathcal{V}}$

les étapes simples

La réalisation des étapes (ii) à (iv), (vii) et (viii) ne présente pas de difficultés majeures à condition toutefois :

- de posséder les variables adéquates décrivant les différents corps du système – par exemple celles qui sont fournies en sortie du compilateur *LSD*, voir (§3.2.4) – et, le cas échéant, les formules qui les relient (formules de changement de repère, formules de dérivation vectorielle, formules de composition des vitesses ...),
- d'utiliser un logiciel de calcul formel permettant de manipuler des expressions matricielles, d'effectuer des substitutions à l'intérieur des expressions et possédant un langage de programmation simple.

On va ici se contenter de présenter le calcul de la matrice $\mathfrak{M}(\mathfrak{p})$ sur un exemple. Pour cela, on considère à nouveau le satellite qui a servi d'exemple dans le chapitre (§2), et on utilise les variables définies dans la section (§3.2.4) – in fine – ainsi que les variables `MMassMax` et `MMassMax_`i qui designent respectivement $\mathfrak{M}(\mathfrak{p})$ et $\mathfrak{M}_i(\mathfrak{p})$.

Le calcul est alors réalisé par la création successive d'équations comme le montre le code ci-dessous[§] :

```
> MMassMax = BlockDiagonal(seq(MMassMax_.i, i = 1..N) ;

                  [ MMassMax_1       0           0           0     ]
                  [                                                ]
                  [      0      MMassMax_2       0           0     ]
        MMassMax = [                                               ]
                  [      0           0      MMassMax_3       0     ]
                  [                                                ]
```

[§] `&s`, `&*` et `&~` sont des opérateurs qui implémentent respectivement la multiplication par un scalaire, la multiplication et l'opérateur *tilde*

```
                    [      0           0           0      MMassMax_4 ]

> seq(MMassMax_.i = BlockMatrix([[&s(mu_.i,Id),&s(-mu_.i,&~(VOT_.i._.i.5))],
                                 [&s(mu,&~(VOT_.i._.i.5))),MOIo_.i],
      i = 1..N) ;

                                 [  mu_1 &s Id     0     ]
                    MMassMax_1 = [                         ],
                                 [       0       MOIo_1  ]

                                          [  0  0  ]
                            MMassMax_2 = [        ],
                                          [  0  0  ]

                                          [  0  0  ]
                            MMassMax_3 = [        ],
                                          [  0  0  ]

                    [        mu_4 &s Id       - mu_4 &s &~( VOT_4_45 )  ]
      MMassMax_4 = [                                                    ]
                    [  mu_4 &s &~( VOT_4_45 )           MOIo_4          ]
```

MMassMax_2 et MMassMax_3 sont directement évaluées comme des matrices nulles dans la mesure où la description du système a déjà produit les affectations :

```
> VOT_1_15 = 0 ;
> mu_2 := 0 ;
> mu_3 := 0 ;
> MOIo_2 := 0 ;
> MOIo_3 := 0 ;
```

De plus la compilation du programme *LSD* de déscription du mécanisme a aussi engendré :

- au niveau matriciel, des équations exprimant les caractéristiques cinétiques des corps, et des formules de changement de repères :

  ```
  > MOIo_1 = MOIg_1 ;
  > VOT_4_45 = MOR_0_4 &* V4T_4_45 ;
  > MOIo_4 = &s(-mu_4,&~(VOT_4_45) &* &~(VOT_4_45)) ;
  > MOIg_1 = MOR_0_1 &* MIg_1 &* &t(MOR_0_1) ;
  ```

- et au niveau scalaire des affectations transcrivant les propriétés des corps et définissant les vecteurs et les matrices soit en fonction des paramètres du système soit en fonction des caractéristiques physiques des corps :

  ```
  > MOR_0_4 ;

    [cos(p4[5](t)) cos(p4[6](t)), - cos(p4[5](t)) sin(p4[6](t)), sin(p4[5](t))]

    [sin(p4[4](t)) sin(p4[5](t)) cos(p4[6](t)) + cos(p4[4](t)) sin(p4[6](t)),

       - sin(p4[4](t)) sin(p4[5](t)) sin(p4[6](t)) + cos(p4[4](t)) cos(p4[6](t)),

                   - sin(p4[4](t)) cos(p4[5](t))]

    [- cos(p4[4](t)) sin(p4[5](t)) cos(p4[6](t)) + sin(p4[4](t)) sin(p4[6](t)),

       cos(p4[4](t)) sin(p4[5](t)) sin(p4[6](t)) + sin(p4[4](t)) cos(p4[6](t)),

                    cos(p4[4](t)) cos(p4[5](t))]

  > V4T_4_45 ;
                                   [      0     ]
                                   [            ]
                                   [      0     ]
                                   [            ]
                                   [ V4T_4_45_z ]
  ```

```
> MOR_0_1 ;

 [cos(p1[5](t)) cos(p1[6](t)), - cos(p1[5](t)) sin(p1[6](t)), sin(p1[5](t))]

 [sin(p1[4](t)) sin(p1[5](t)) cos(p1[6](t)) + cos(p1[4](t)) sin(p1[6](t)),

    - sin(p1[4](t)) sin(p1[5](t)) sin(p1[6](t)) + cos(p1[4](t)) cos(p1[6](t)),

                - sin(p1[4](t)) cos(p1[5](t))]

  [- cos(p1[4](t)) sin(p1[5](t)) cos(p1[6](t)) + sin(p1[4](t)) sin(p1[6](t)),

     cos(p1[4](t)) sin(p1[5](t)) sin(p1[6](t)) + sin(p1[4](t)) cos(p1[6](t)),

                  cos(p1[4](t)) cos(p1[5](t))]


> MIg_1 = matrix([[M1Ig_1_xx,0,0],[0,M1Ig_1_yy,0],[0,0,M1Ig_1_zz]]) ;

                            [ M1Ig_1_xx      0          0     ]
                            [                               ]
                  M1Ig_1 = [     0      M1Ig_1_yy      0     ]
                            [                               ]
                            [     0          0      M1Ig_1_zz ]
```

On peut ainsi avoir, si on le désire, l'expression de tous les composants scalaires de la matrice $\mathfrak{M}(\mathfrak{p})$.

les étapes (i) et (v)

On a vu comment le parcours du graphe de la structure du mécanisme en "Breath First Search" permet de décomposer l'ensemble des liaisons en liaisons principales et complémentaires et permet de déterminer l'arbre de recouvrement qui définit la structure du système mécanique arborescent sous-jacent.

Pour achever ces deux étapes, il s'agit donc de pouvoir exprimer $\mathfrak{D}(\mathfrak{p})$, et le cas échéant $\mathfrak{D}_{\text{cmp}}(\mathfrak{p})$, à partir de la description de la nature des liaisons.

> On considère une liaison entre les corps $\mathcal{S}_i$ et $\mathcal{S}_j$ faisant intervenir les repères $\mathcal{R}_{ij}$ et $\mathcal{R}_{ji}$.
>
> Les renseignements fournis par la description de la nature de cette liaison consistent en la donnée sous forme paramétrique de l'ensemble des déplacements relatifs admissibles transformant, à l'instant t, la position du premier repère en la position du second.
>
> Si $d_{ij} = (\tau_{ij}, \rho_{ij})$ est un tel déplacement, on peut décrire la position de l'origine du repère de référence du corps $\mathcal{S}_j$ de deux manières différentes, ce qui donne :
>
> $$\overrightarrow{OO_j} = \overrightarrow{OO_i} + \overrightarrow{O_iO_{ij}} + \tau_{ij} + \overrightarrow{O_{ji}O_j}$$
>
> dans cette équation $\overrightarrow{OO_j}$ et $\overrightarrow{O_{ji}O_j}$ ne dépendent que des paramètres de position $\mathfrak{p}_j$, $\overrightarrow{OO_i}$ et $\overrightarrow{O_iO_{ij}}$ des paramètres de position $\mathfrak{p}_i$ et τ_{ij} des paramètres articulaires p_j.
>
> De même, si on note Ω_{ij} le vecteur rotation instantanée associé à d_{ij}, on a :
>
> $$\omega_j = \Omega_{ij} + \omega_j$$
>
> En différentiant ces six équations, on obtient une expresssion de la forme :
>
> $$D^j_{ij}\mathfrak{q}_j + D^i_{ij}\mathfrak{q}_i + D_{ij}q_j = 0 \qquad (3.3.2.\text{i})$$

d'où il est théoriquement possible d'éliminer les paramètres articulaires cinématiques q_j pour obtenir la matrice de contrainte – on trouvera explicitement les équations (3.3.2.i) dans [Papegay et al., 1991].

Cet élimination peut être coûteuse en temps de calcul. C'est pourquoi, en fait, on préferera dans une implémentation, utiliser une bibliothèque de liaisons prédéfinies paramétrables – par exemple par la direction du (ou des) axe(s) de rotation et/ou de translation – et posséder, au niveau du générateur des équations de la dynamique une bibliothèque correspondante – d'expressions précalculées – fournissant automatiquement les équations de contrainte en fonction du type de liaison utilisée et des paramètres la définissant.

l'étape (vi)

Pour chaque liaison ℓ_{ij}, on suppose ici que l'on sait, soit par le calcul direct, soit en faisant appel à une bibliothèque d'expressions précalculées :

- déterminer un paramétrage minimal (p_j, q_j) de la variété de configurations relative du sous-système $\mathcal{S}_i \sqcup \mathcal{S}_j$ soumis exclusivement à ℓ_{ij},
- exprimer les relations existant entre le nouveau et l'ancien paramétrage tant en ce qui concerne les paramètres de position qu'en ce qui concerne les paramètres cinématiques, c'est à dire avec des notations habituelles, écrire – voir (§1.2.2)) :

$$\mathfrak{p}_j = \psi_{ij}(\mathfrak{p}_i, p_j) \text{ et } \mathfrak{q}_j = \Psi^i_{ij}(p_j)\mathfrak{q}_i + \Psi^j_{ij}(p_j)q_j \quad \Big(\text{si } i = 0,\ \mathfrak{p}_j = \psi_{0j}(p_j) \text{ et } \mathfrak{q}_j = \Psi^j_{0j}(p_j)q_j\Big)$$

On pose alors :

$$\Psi_j(p_j) = \begin{pmatrix}
\mathbf{1} & \mathbf{0} & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \mathbf{0} \\
\mathbf{0} & \ddots & \ddots & & & & & & & & & \vdots \\
\vdots & \ddots & \ddots & \ddots & & & & & & & & \vdots \\
\vdots & & \ddots & \ddots & \ddots & & & & & & & \vdots \\
\vdots & & & \ddots & \ddots & \ddots & & & & & & \vdots \\
\vdots & & & & \ddots & \ddots & \ddots & & & & & \vdots \\
\mathbf{0} & \dots & \dots & \mathbf{0} & \dots & \mathbf{0} & \mathbf{1} & \mathbf{0} & \dots & \dots & \dots & \mathbf{0} \\
\mathbf{0} & \dots & \mathbf{0} & \Psi^i_{ij}(p_j) & \mathbf{0} & \dots & \mathbf{0} & \Psi^j_{ij}(p_j) & \mathbf{0} & \dots & \dots & \mathbf{0} \\
\mathbf{0} & \dots & \dots & \mathbf{0} & \dots & \dots & \dots & \mathbf{0} & \mathbf{1} & \mathbf{0} & \dots & \mathbf{0} \\
\vdots & & & & & & & & \ddots & \ddots & \ddots & \vdots \\
\vdots & & & & & & & & & \ddots & \ddots & \mathbf{0} \\
\mathbf{0} & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \mathbf{0} & \mathbf{1}
\end{pmatrix}$$

$$\left(\text{si } i=0, \qquad \Psi_j(p_j) = \begin{pmatrix} \mathbf{1} & \mathbf{0} & \dots & \dots & \dots & \dots & \mathbf{0} \\ \mathbf{0} & \ddots & \ddots & & & & \vdots \\ \vdots & \ddots & \mathbf{1} & \ddots & & & \vdots \\ \mathbf{0} & \dots & \mathbf{0} & \Psi^j_{0j}(p_j) & \mathbf{0} & \dots & 0 \\ \vdots & & & \ddots & \mathbf{1} & \ddots & \vdots \\ \vdots & & & & \ddots & \ddots & 0 \\ \mathbf{0} & \dots & \dots & \dots & \dots & \mathbf{0} & \mathbf{1} \end{pmatrix}\right)$$

de sorte que :

$$\begin{pmatrix} \mathfrak{q}_1 \\ \vdots \\ \mathfrak{q}_{j-1} \\ \mathfrak{q}_j \\ q_{j+1} \\ \vdots \\ q_n \end{pmatrix} = \Psi_j(p_j) \begin{pmatrix} \mathfrak{q}_1 \\ \vdots \\ \mathfrak{q}_{j-1} \\ q_j \\ q_{j+1} \\ \vdots \\ q_n \end{pmatrix}$$

alors :

$$\begin{pmatrix} \mathfrak{q}_1 \\ \mathfrak{q}_2 \\ \vdots \\ \mathfrak{q}_{n-1}\mathfrak{q}_n \end{pmatrix} = \Psi_n(p_n)\ \Psi_{n-1}(p_{n-1})\ \dots\ \Psi_2(p_2)\ \Psi_1(p_1) \begin{pmatrix} q_1 \\ q_2 \\ \vdots \\ q_{n-1} \\ q_n \end{pmatrix}$$

d'où : $\Psi(p) = \Psi_n(p_n)\ \Psi_{n-1}(p_{n-1})\ \dots\ \Psi_2(p_2)\ \Psi_1(p_1)$

Remarque 3.1 Il est important de remarquer que c'est la numérotation des liaisons à l'aide du parcours du graphe de la structure en "Breath First Search" qui permet d'obtenir cette expression de $\Psi(p)$ dans la mesure où l'on n'exprime la position d'un corps en fonction des coordonnées articulaires d'une liaison que si les positions de tous les autres corps qui lui sont reliés sont déjà exprimées en fonction des coordonnées articulaires des liaisons correspondantes.

De manière analogue, on obtient les expressions des anciens paramètres de position en fonction des nouveaux en écrivant :

$$\mathfrak{p}_j = \psi_{i_1 j}(\psi_{i_2 i_1}(\dots\ (\psi_{i_k i_{k-1}}(\psi_{0 i_k}(p_{i_k}), p_{i_{k-1}}), \dots),\ p_{i_1}), p_j) = \psi_j(p)$$

où la suite d'indices $i_1, i_2, \dots, i_k$ est obtenue en parcourant, de la feuille vers la racine, le chemin de l'arbre qui relie la racine au nœud correspondant au corps $\mathcal{S}_j$.
On est ainsi en mesure, grâce à l'algorithme décrit ci-dessus, d'obtenir automatiquement les équations du mouvement de mécanismes à structure arborescente, sous une forme particulièrement agréable, à condition de pouvoir manipuler des expressions matricielles et vectorielles.

3.4 manipulations d'expressions en *MAPLE*

La manipulation des équations décrivant la dynamique des mécanismes et, plus généralement, de toutes les expressions algébriques que l'on peut rencontrer au cours de la génération de ces équations est un intérêt fondamental de l'utilisation du calcul formel pour cette tâche.

Cet intérêt se manifeste à double titre :

- en ce qui concerne la simplification, d'une part, parce que les équations qui vont être transmises à un simulateur numériques sont destinées à être évaluées un grand nombre de fois – à chaque pas d'intégration – et donc qu'un gain de taille, même faible, sur les expressions au niveau formel entraînera un gain de temps de simulation non négligeable,
- d'autre part, en ce qui concerne la manipulation proprement dite, dans la mesure où l'interprétation mécanique des équations n'est réalisable que si certaines grandeurs y apparaissent explicitement, ce qui peut nécessiter des factorisations, des regroupements de termes, des changements de variables, ...

On présente dans cette section deux bibliothèques de procédures *MAPLE* qui implémentent dans ce système de calcul formel :

- l'une, la manipulation d'expressions matricielles,
- l'autre, la manipulation de quaternions et d'expressions construites à partir de quaternions.

Ces deux bibliothèques sont essentiellement différentes.

La première a été conçue pour être intégrée à un logiciel de génération formelle des équations de la dynamique des mécanismes utilisant le formalisme de Kane – développé dans le cadre d'une étude supportée par le Centre National d'Etudes Spatiales de Toulouse, voir [Capolsini, 1991]. Elle exploite au mieux les fonctionnalités de *MAPLE* concernant l'introduction, la définition et l'utilisation de nouveaux opérateurs.

La deuxième intervient à un niveau de détail des expressions plus élevé puisqu'elle permet de représenter les rotations de l'espace par des quaternions. Elle est complètement autonome et n'utilise de *MAPLE* que le langage de programmation et quelques fonctionnalités de base. Cette bibliothèque – voir [Capolsini, Dalmas et Papegay, 1991] et [Capolsini, Dalmas et Papegay, 1992] – a été développée à la demande du CNES précisément pour résoudre des problèmes de représentation des rotations.

3.4.1 les expressions matricielles

MAPLE dispose, de manière standard d'une large bibliothèque de procédures de manipulation des expressions matricielles mais celles-ci ne sont utilisables que sur des matrices complètement instanciées.

On a vu cependant – au chapitre (§2) – l'importance de pouvoir considérer des termes qui représentent des matrices sans avoir accès à leurs composantes scalaires.

Il existe également en *MAPLE* un évaluateur matriciel spécialisé, la fonction `evalm` qui, appelée sur une expression l'évalue en considérant qu'elle est constituée de matrices. Mais son emploi n'est pas satisfaisant car, d'une part, elle ne reconnaît pas les opérateurs de produit scalaire et de produit vectoriel et, d'autre part, les performances de simplification ne sont pas suffisantes, comme on peut le constater sur l'exemple suivant :

```
> evalm(A &* B - A &* (B - A));
                            (A &* B) - (A &* (B - A))
> evalm(A &* B &* (2*B) - A &* B);
                                      2
```

```
                              2 (A &* B ) - (A &* B)
> evalm(A &* B &* (2*B)-B &* A^0);
Error, (in evalm/amperstar) &* is reserved for matrix multiplication
```

Par contre il existe en *MAPLE* une fonction `define` qui permet de définir de nouveaux opérateurs en leur associant des propriétés algébriques et/ou des règles de simplification décrivant leurs comportements.

On a utilisé cette fonction – ou tout au moins on s'est inspiré de son code – pour définir les opérateurs suivants qui peuvent être utilisé indifféremment en notation préfixe ou infixe, exceptés la transposition et l'opérateur *tilde*[*] qui sont unaires.

La multiplication par un scalaire est notée &s, est binaire, reconnaît les scalaires 0 et 1 et regroupe les multiplications scalaires imbriquées.

```
> &s(0, A);
                                        0
> &s(1, A);
                                        A
> &s(2, &s(3, A));
                                    6 &s A
```

L'addition est notée &+ et est n-aire. Elle est commutative, reconnaît l'élément neutre 0 – notée comme le scalaire 0 –, regroupe les additions imbriquées et extrait les multiplications scalaires.

```
>  A &+ 0;
                                        A
> &+(A, &+(B ,C));
                                  &+(A, B, C)

> &s(a, A) &+ &s(b, A);
                                  (a + b) &s A
```

L'opérateur *tilde* transforme un vecteur – une matrice 3×1 en une matrice 3×3. Il est noté &~, est unaire, reconnaît le vecteur nul et extrait les multiplications scalaires.

```
>  &~(0);
                                        0
> &~(&s(a, A));
                                  a &s (&~ A)
```

La transposition est notée &t, est unaire, reconnaît la matrice nulle, la matice identité – notée Id. Elle est involutive et tient compte du caractère antisymétrique de l'image d'un vecteur par l'opérateur *tilde*. Elle extrait également les multiplications scalaires.

```
> &t(0);
                                        0
> &t(Id);
                                        Id
> &t(&t A);
```

[*] L'opérateur *tilde* a déjà été introduit dans la section précédente : à un vecteur de l'espace il associe une matrice antisymétrique telle que le produit d'un vecteur par cette matrice réalise le produit vectoriel des deux vecteurs considérés.

```
                                        A
> &t(&~ A);
                                   (-1) &s (&~ A)
> &t(&s(a, A));
                                    a &s (&t A)
```

La puissance est notée &^ et est évidemment binaire. Elle reconnaît les exposants 0 et 1 et extrait egalement les multiplications scalaires.

```
> A &^ 1;
                                        A
> A &^ 0;
                                        Id
> 0 &^ x;
                                        0
> &s(x, A) &^ y;
                                    y
                                   x  &s (A &^ y)
```

La multiplication est notée &* et est n-aire comme l'addition mais non-commutative. Elle accepte comme argument aussi bien des termes représentant des matrices 3×3 que des termes représentant des vecteurs, mais ne vérifie pas le bon dimensionnement des matrices manipulées. Par défaut, elle :

- regroupe les multiplications imbriquées,
- reconnaît la matrice identité et la matrice nulle,
- regroupe éventuellement un produit sous forme de puissance,
- extrait les multiplications par des scalaires,
- reconnaît que la multiplication d'un vecteur par son image par l'opérateur *tilde* est nulle.

```
> &*(A, &*(B,C));
                                   &*(A, B, C)
> &*(Id,A);
                                        A
> &*(A,Id);
                                        A
> &*(A, 0);
                                        0
> &*(0, A);
                                        0
> &*(A, A, A);
                                     A &^ 3
> A &* (&s(x, B));
                                   x &s (A &* B)
> (&~ A) &* A;
                                        0
```

L'inverse est un opérateur unaire noté &i défini à partir de l'opérateur de puissance. Il est involutif et extrait la multiplication par un scalaire.

```
> &i(A);
                                  A &^ (-1)
> &i(&i (A));
                                      A
> &i (&s (a, A));
                                1/a &s (A &^ (-1))
```

Le produit scalaire est noté `&.` et est binaire et symétrique. Il est défini comme étant la multiplication de l'un des vecteurs par la transposée de l'autre et hérite ses propriétés de bilinéarité des propriétés de la multiplication.

```
 > &.(B, A);
                                  (&t A) &* B
> &.(A,B);
                                  (&t A) &* B
> &.(&s(2, A), B);
                                2 &s ((&t A) &* B)
```

Le produit vectoriel est défini à partir de l'opérateur *tilde* et hérite de ses propriétés. Il est noté &x, est binaire et antisymétrique.

```
> &x(A, B);
                                  (&~ A) &* B
> &x(B, A);
                              (-1) &s ((&~ A) &* B)
> &x(A, 0);
                                      0
> &x(0,A);
                                      0
> &x(A, A);
                                      0
> &x(&s(2, A), B);
                                2 &s ((&~ A) &* B)
```

Avec les opérateurs définis ci-dessus, on est à même d'écrire les expressions matricielles les plus diverses et notamment celles rencontrées dans la génération des équations de la dynamique des mécanismes.

Pour manipuler ces expressions, on dispose de deux fonctions spécifiques `expand` et `evalmat`.

Le but premier d'`expand` est de distribuer les produits par rapport aux sommes, mais cette fonction utilise aussi les propriétés suivantes :

- la transposée d'un produit est le produit des transposées pris dans l'ordre inverse,

$$^T(AB) = {}^T B\, {}^T A$$

- la transposée d'une somme est la somme des transposées,

$$^T(A+B) = {}^T A + {}^T B$$

- l'inverse d'un produit est le produit des inverses dans l'ordre opposé,

$$(AB)^{-1} = B^{-1} A^{-1}$$

- la puissance d'un produit est le produit autant de fois que nécessaire,

$$(AB)^3 = ABABAB$$

- l'image par l'opérateur *tilde* d'une somme est la somme des images par cet opérateur,

$$\widetilde{(AB)} = \tilde{A} + \tilde{B}$$

- enfin, une règle sur le double produit vectoriel qui se traduit sur l'opérateur *tilde* :

$$\widetilde{(\tilde{A}B)} = \tilde{A}\tilde{B} - \tilde{B}\tilde{A}$$

comme on le voit ci-dessous :

```
> expand(A &* (B &+ C));
                                (A &* B) &+ (A &* C)
> expand(&t(A &* B));
                                  (&t B) &* (&t A)
> expand(&t(A &+ B));
                                  (&t A) &+ (&t B)
> expand(&^(A &* B, -1));
                            (B &^ (-1)) &* (A &^ (-1))
> expand(&^(A &* B, 3));
                               &*(A, B, A, B, A, B)
> expand(&~(A &+ B &+ C));
                               &+(&~ A, &~ B, &~ C)
> expand(&~ ((&~ A) &* B));
               ((-1) &s ((&~ B) &* (&~ A))) &+ ((&~ A) &* (&~ B))
```

Quant à `evalmat`, elle évalue les expressions matricielles en réappliquant récursivement les règles de simplifications utilisées par défaut dans la définition des opérateurs et en reconnaissant les opposés dans les sommes et les inverses dans les produits. Ainsi, on arrive à simplifier l'expression donnée plus haut en exemple du fonctionnement imparfait de la fonction `evalm` de *MAPLE* :

```
> (A &* B) &+ ((-1 &s A) &* (B &+ (-1 &s A)));
                  (A &* B) &+ (&s(-1,  A) &* (B &+ &s(-1,  A)))
> expand(");
                         &+(A &* B, &s(-1,  A &* B), A &^ 2)
> evalmat(");
                                          2
                                         A
```

Par contre, on n'est pas en mesure de tenir compte des propriétés de certaines matrices – matrices antisymétriques, matrices de rotation – et de certains vecteurs – orthogonalités – manipulés. Pour cela, il faudrait utiliser un système de calcul formel possédant un mécanisme de typage des expressions plus performant que celui de *MAPLE* – voir (§3.5.

On trouvera en annexe (§B.2) le code *MAPLE* qui implémente toutes les procédures présentées dans cette section.

3.4.2 les expressions à base de quaternions

"`QUATERMAN`", pour MANipulateur de QUATERnions est une bibliothèque de procédures *MAPLE* qui permettent de travailler dans l'algèbre des quaternions. Elle est composée d'un ensemble de

procédures implémentant les opérations de base du corps des quaternions, des opérations sur les vecteurs de l'espace à trois dimensions (qui interviennent naturellement dans les quaternions), des manipulations symboliques classiques sur ces objets (utilisation de la distributivité des produits, simplifications usuelles), les opérations qui permettent d'utiliser les quaternions pour faire de la géométrie euclidienne dans l'espace tridimensionnel (représentation des rotations comme des quaternions de norme 1) ainsi qu'un peu de calcul différentiel (quaternion dépendant d'un paramètre réel).

La principale caractéristique de `QUATERMAN` est la possibilité de manipuler des quaternions sous trois formes différentes :

- purement symbolique – un identificateur,
- purement numérique – les quatres composantes sont connues,
- mixte : on connait la composante scalaire et la composante vectorielle (sous la forme d'une expression formelle).

Manipuler cette dernière forme est indispensable pour les applications des quaternions à la géométrie. Cela a donc nécessité l'écriture des opérations de base du calcul vectoriel (somme, produit scalaire et vectoriel) qui puissent manipuler des vecteurs et des expressions vectorielles sous forme symbolique (ce qui n'existait pas dans *MAPLE*, la bibliothèque `linalg` n'étant capable de manipuler que des vecteurs dont les composantes sont explicitement connues i.e. des listes de trois expressions ou des tableaux).

Dans cette section, on va d'abord rappeler quelques propriétés mathématiques des quaternions – le lecteur intéressé pourra utilement se référer à [Borgne, 1987] – ensuite, on verra la façon d'exprimer les quaternions et les vecteurs dans `QUATERMAN` ainsi que les opérations de base de la bibliothèque, enfin on rappelera brièvement le lien entre les quaternions unitaires et les rotations de l'espace.

l'algèbre des quaternions

L'ensemble des quaternions est simultanément :

- un espace vectoriel réel lorsqu'on le munit de l'addition de $\mathbb{R}^4$ ($\boxplus$) et de la multiplication externe par un réel ($\boxdot$),
- un corps non commutatif si on le munit de $\boxplus$ et d'une multiplication interne ($\boxtimes$) telle que les vecteurs de la base canonique 1, i, j et k vérifient :

$$\begin{cases} i \boxtimes i = j \boxtimes j = k \boxtimes k = -1 \\ j \boxtimes k = -k \boxtimes j = i \\ k \boxtimes i = -i \boxtimes k = j \\ i \boxtimes j = -j \boxtimes i = k \end{cases}$$

  et bien sûr $1 \boxtimes 1 = 1$, $1 \boxtimes i = i$, $1 \boxtimes j = j$ et $1 \boxtimes k = k$.
- et par conséquent une algèbre non commutative de dimension quatre sur le corps des réels (munit de $\boxplus$, de $\boxtimes$ et de $\boxdot$).

Le produit de deux quaternions s'écrit, si q_1 et q_2 ont respectivement comme coordonnées dans la base canonique (s_1, x_1, y_1, z_1) et (s_2, x_2, y_2, z_2) :

$$q_1 \boxtimes q_2 : \begin{pmatrix} s_1 s_2 - x_1 x_2 - y_1 y_2 - z_1 z_2 \\ s_1 x_2 + s_2 x_1 + y_1 z_2 - y_2 z_1 \\ s_1 y_2 + s_2 y_1 + z_1 x_2 - z_2 x_1 \\ s_1 z_2 + s_2 z_1 + x_1 y_2 - x_2 z_1 \end{pmatrix}$$

Puisque l'ensemble des quaternions est un corps, il existe une opération qui, à tout quaternion q non nul associe son inverse que nous noterons $q^{[-1]}$.

Le corps des quaternions contient un sous-corps commutatif $\mathbb{R} \boxdot 1$ isomorphe à $\mathbb{R}$ et un sous-espace vectoriel sur ce corps $\mathbb{R} \boxdot i + \mathbb{R} \boxdot j + \mathbb{R} \boxdot k$ isomorphe à $\mathbb{R}^3$.

Ainsi on est amené à décomposer tout quaternion en la somme de sa partie scalaire et de sa partie vectorielle. L'opération involutive de conjugaison transforme un quaternion q en le quaternion $\overline{q}$ de même partie scalaire et de partie vectorielle opposée.

Le produit de deux quaternions peut alors s'écrire naturellement à partir des opérations courantes de $\mathbb{R}$ $(+,\times)$ et de $\mathbb{R}^3$ $(\oplus, \odot, \wedge, <.,.>)$, si q_1 et q_2 se décomposent respectivement en (s_1, v_1) et (s_2, v_2) :

$$q_1 \boxtimes q_2 : \begin{pmatrix} s_1 s_2 - < v_1, v_2 > \\ (s_1 \odot v_2) \oplus (s_2 \odot v_1) \oplus (v_1 \wedge v_2) \end{pmatrix}$$

De plus, l'algèbre des quaternions est une algèbre normée, la norme d'un quaternion étant définie à partir du produit scalaire de $\mathbb{R}^4$, $\ll \cdot, \cdot \gg$ par :

$$\|q\| = \sqrt{\ll q, q \gg} = \sqrt{\frac{1}{2}\left((\overline{q} \boxtimes q) \boxplus (q \boxtimes \overline{q})\right)}$$

le carré de cette norme intervient dans le calcul de l'inverse d'un quaternion car :

$$q^{[-1]} = \frac{1}{\|q\|^2} \boxdot \overline{q}$$

représentation des quaternions

Lorsque l'on calcule dans l'algèbre des quaternions, on est amené à utiliser des quaternions sous trois formes différentes qui correspondent à des niveaux de détail supplémentaires. En effet, comme on l'a vu plus haut, on peut désigner un quaternion par :

i. un symbole,

ii. un couple, dont la première composante est la partie scalaire, et dont la deuxième est la partie vectorielle du quaternion, cette dernière partie pouvant être elle-même :
   - un symbole,
   - le triplet de ses coordonnées scalaires dans une base prédéfinie,

iii. le 4-uplet de ses coordonnées dans la base canonique (voir le §3.4.2)

Si l'on veut simplifier des expressions faisant intervenir des quaternions – ou des vecteurs – que l'on ne connaît que par un symbole, les seules règles que l'on peut utiliser sont dues aux propriétés des opérations (commutativité, associativité, distributivité, ...).

Par contre, si l'on connaît, même formellement, la composante scalaire et la composante vectorielle des quaternions rencontrés dans une expression, il est possible d'utiliser, pour simplifier cette expression, des règles faisant intervenir des propriétés de ces composantes (parties scalaires opposées, parties vectorielles colinéaires ou orthogonales, ...).

Enfin, il est clair que la connaissance des quatre coordonnées des quaternions considérés permet d'effectuer complètement les calculs et d'atteindre – au détriment en général de la taille des expressions obtenues – un degré de simplification maximal.

Dans `QUATERMAN`, on accepte donc que n'importe quel symbole représente un quaternion, l'utilisation d'un symbole dans une expression écrite à base d'opérateurs dédiés aux quaternions étant suffisante pour que ce symbole soit considéré comme représentant un quaternion. Il en va bien entendu de même pour les vecteurs.

Par contre, pour manipuler des quaternions sous une des deux autres formes, il sera nécessaire d'indiquer à *MAPLE* que les couples ou les 4-uplets en question représentent bien des quaternions, au moyen d'une fonction de création `qmake`. De plus, déclarer qu'un couple représente un quaternion entraînera implicitement que la deuxième composante soit considérée par la bibliothèque comme un vecteur.

Des types (au sens de la fonction *MAPLE* `type`) sont associés aux objets ainsi déclarés et aux expressions construites à partir d'opérateurs dédiés pour permettre d'utiliser simultanément, en limitant les risques d'erreurs, à la fois `QUATERMAN` et les objets traditionnellement manipulés par d'autres librairies de *MAPLE*.

Le fait de pouvoir manipuler des quaternions sous des formes différentes oblige à prendre en compte différents niveaux de calcul et à pouvoir à tout moment passer d'un niveau à un autre lorsque c'est mathématiquement correct et désirable.

Aux trois formes sous lesquelles peuvent apparaître les quaternions, correspondent trois niveaux d'évaluation des expressions :

- Lorsque l'on écrit des expressions avec des opérateurs dédiés aux quaternions ne faisant intervenir que des quaternions sous forme de symboles *MAPLE*, aucune évaluation n'est effectuée à part l'évaluation standard de *MAPLE*. Néanmoins, des règles de simplification rattachées aux opérations sont appliquées.
- Le deuxième niveau d'évaluation correspond à la manipulation des quaternions sous forme de parties scalaires et de parties vectorielles. On calcule à ce niveau en le demandant explicitement (par la fonction `qeval`). A ce niveau interviennent des opérations sur les vecteurs formels et les règles de simplification qui leur sont rattachées.
- Le troisième niveau correspond à la manipulation de quaternions dont on connait les 4 composantes. Les calculs sont alors fait totalement par défaut. On peut évidemment accéder à ce niveau d'évaluation à partir du niveau précédent et du mécanisme d'évaluation de *MAPLE* en affectant aux vecteurs représentant des parties vectorielles de quaternions leurs coordonnées dans la base canonique.

opérations sur les quaternions

Les six opérations présentées au paragraphe 3.4.2 sont représentées par les six procédures :

- `qadd` : l'addition n-aire,
- `qsmult` : la multiplication par un scalaire,
- `qmult` : la multiplication interne n-aire,
- `qinv` : l'inversion,

- `qconj` : la conjugaison,
- `qnorm` : le carré de la norme.

Conformément à la philosophie *MAPLE*, les opérations qui font intervenir des quaternions dont on connaît numériquement les composantes sont effectuées par ces procédures.

L'évaluation d'une expression contenant ces opérations de bases entraîne l'application – récursive – des règles de simplification suivantes :

$$\begin{cases} q \boxplus 0_{\mathcal{Q}} \longrightarrow q \\ 0_{\mathcal{Q}} \boxplus q \longrightarrow q \end{cases} \qquad (\mathcal{Q}.\text{i})$$

$$\begin{cases} q_1 \boxplus (q_2 \boxplus q_3) \longrightarrow q_1 \boxplus q_2 \boxplus q_3 \\ (q_1 \boxplus q_2) \boxplus q_3 \longrightarrow q_1 \boxplus q_2 \boxplus q_3 \end{cases} \qquad (\mathcal{Q}.\text{ii})$$

$$\left.\begin{array}{c} q_1 \boxplus q_2 \\ q_2 \prec q_1 \end{array}\right\} \longrightarrow q_2 \boxplus q_1 \qquad (\mathcal{Q}.\text{iii})$$

$$\begin{cases} (a \boxdot q) \boxplus (b \boxdot q) \longrightarrow (a+b) \boxdot q \\ (a \boxdot q) \boxplus q \longrightarrow (a+1) \boxdot q \\ q \boxplus (a \boxdot q) \longrightarrow (1+a) \boxdot q \end{cases} \qquad (\mathcal{Q}.\text{iv})$$

$$(0 \boxdot q) \longrightarrow 0_{\mathcal{Q}} \qquad (\mathcal{Q}.\text{v})$$

$$(1 \boxdot q) \longrightarrow q \qquad (\mathcal{Q}.\text{vi})$$

$$(k \boxdot 0_{\mathcal{Q}}) \longrightarrow 0_{\mathcal{Q}} \qquad (\mathcal{Q}.\text{vii})$$

$$a \boxdot (b \boxdot q) \longrightarrow (a * b) \boxdot q \qquad (\mathcal{Q}.\text{viii})$$

$$\begin{array}{c} 1_{\mathcal{Q}} \boxtimes q \longrightarrow q \\ q \boxtimes 1_{\mathcal{Q}} \longrightarrow q \end{array} \qquad (\mathcal{Q}.\text{ix})$$

$$\begin{array}{c} 0_{\mathcal{Q}} \boxtimes q \longrightarrow 0_{\mathcal{Q}} \\ q \boxtimes 0_{\mathcal{Q}} \longrightarrow 0_{\mathcal{Q}} \end{array} \qquad (\mathcal{Q}.\text{x})$$

$$\begin{cases} q_1 \boxtimes (q_2 \boxtimes q_3) \longrightarrow q_1 \boxtimes q_2 \boxtimes q_3 \\ (q_1 \boxtimes q_2) \boxtimes q_3 \longrightarrow q_1 \boxtimes q_2 \boxtimes q_3 \end{cases} \qquad (\mathcal{Q}.\text{xi})$$

$$(a \boxdot q_1) \boxtimes (b \boxdot q_2) \longrightarrow (a * b) \boxdot (q_1 \boxtimes q_2) \qquad (\mathcal{Q}.\text{xii})$$

$$\left(q^{[-1]}\right)^{[-1]} \longrightarrow q \tag{$\mathcal{Q}$.xiii}$$

$$\begin{cases} q^{[-1]} \boxtimes q \longrightarrow 1_{\mathcal{Q}} \\ q \boxtimes q^{[-1]} \longrightarrow 1_{\mathcal{Q}} \end{cases} \tag{$\mathcal{Q}$.xiv}$$

$$(a \boxdot q)^{[-1]} \longrightarrow \frac{1}{a} \boxdot q^{[-1]} \tag{$\mathcal{Q}$.xv}$$

$$\overline{\overline{q}} \longrightarrow q \tag{$\mathcal{Q}$.xvi}$$

$$\overline{a \boxdot q} \longrightarrow a \boxdot \overline{q} \tag{$\mathcal{Q}$.xvii}$$

$$\left\|q^{[-1]}\right\| \longrightarrow \|q\| \tag{$\mathcal{Q}$.xviii}$$

$$\|a \boxdot q\| \longrightarrow a^2 \|q\| \tag{$\mathcal{Q}$.xix}$$

Comme on peut le voir, il s'agit de la combinaisons des règles classiques de calcul dans un corps et une algèbre involutive (pour la conjugaison). On reconnaît (et on simplifie) les éléments neutres de la somme et du produit, les sommes et les produits deviennent n-aires (application de la loi d'associativité), le zéro est absorbant, l'inversion et la conjugaison sont involutives, les opposés et les inverses sont reconnus dans les sommes et les produits (et plus généralement les scalaires qui multiplient un même quaternion sont factorisés)... De plus, les termes d'une somme ou d'un produit sont ordonnés suivant l'ordre classique de *MAPLE* (fonction `sort` avec le paramètre `address`, notée $\prec$ ci-dessus).

Ces règles ont été choisies pour correspondre aux règles de simplification appliquées par défaut en *MAPLE* sur les expressions à valeurs dans le corps des réels ou des complexes.

On peut effectuer, en le demandant explicitement, d'autres transformations algébriques élémentaires sur ces expressions, qui font intervenir les propriétés de distributivité. L'appel des procédures correspondantes (en fait, la fonction `expand` de *MAPLE* correctement étendue) permet d'appliquer les règles suivantes :

$$a \boxdot (q_1 \boxplus q_2) \longrightarrow (a \boxdot q_1) \boxplus (a \boxdot q_2) \tag{$\mathcal{D}$.i}$$

$$(q_1 \boxplus q_2) \boxtimes q_3 \longrightarrow q_1 \boxtimes q_3 \boxplus q_2 \boxtimes q_3 \tag{$\mathcal{D}$.ii}$$

$$(q_1 \boxtimes q_2)^{[-1]} \longrightarrow {q_2}^{[-1]} \boxtimes {q_1}^{[-1]} \tag{$\mathcal{D}$.iii}$$

opérations sur les vecteurs

L'évaluation au deuxième niveau des expressions écrites à partir des procédures décrites au paragraphe précédent nécessite que soient disponibles certaines opérations sur les vecteurs de dimension 3. Ceci est réalisé par les cinq procédures :

- `vadd` : l'addition n-aire,

- `vsmult` : la multiplication par un scalaire,
- `vprod` : le produit vectoriel,
- `vscal` : le produit scalaire,
- `vnorm` : la norme d'un vecteur.

Ces procédures effectuent les calculs lorsque les coordonnées des vecteurs sont connues même seulement de manière formelle.

Voici les règles utilisées récursivement par ces procédures pour la simplification des expressions vectorielles.

$$\begin{cases} v \oplus 0_{\mathcal{V}} \longrightarrow v \\ 0_{\mathcal{V}} \oplus v \longrightarrow v \end{cases} \tag{$\mathcal{V}$.i}$$

$$\begin{cases} v_1 \oplus (v_2 \oplus v_3) \longrightarrow v_1 \oplus v_2 \oplus v_3 \\ (v_1 \oplus v_2) \oplus v_3 \longrightarrow v_1 \oplus v_2 \oplus v_3 \end{cases} \tag{$\mathcal{V}$.ii}$$

$$\left.\begin{array}{l} v_1 \oplus v_2 \\ v_2 \prec v_1 \end{array}\right\} \longrightarrow v_2 \oplus v_1 \tag{$\mathcal{V}$.iii}$$

$$\begin{cases} (a \odot v) \oplus (b \odot v) \longrightarrow (a+b) \odot v \\ (a \odot v) \oplus v \longrightarrow (a+1) \odot v \\ v \oplus (a \odot v) \longrightarrow (a+1) \odot v \end{cases} \tag{$\mathcal{V}$.iv}$$

$$(0 \odot v) \longrightarrow 0_{\mathcal{V}} \tag{$\mathcal{V}$.v}$$

$$(1 \odot v) \longrightarrow v \tag{$\mathcal{V}$.vi}$$

$$(k \odot 0_{\mathcal{V}}) \longrightarrow 0_{\mathcal{V}} \tag{$\mathcal{V}$.vii}$$

$$a \odot (b \odot v) \longrightarrow (a * b) \odot v \tag{$\mathcal{V}$.viii}$$

$$\begin{cases} v \wedge 0_{\mathcal{V}} \longrightarrow 0_{\mathcal{V}} \\ 0_{\mathcal{V}} \wedge v \longrightarrow 0_{\mathcal{V}} \end{cases} \tag{$\mathcal{V}$.ix}$$

$$v \wedge v \longrightarrow 0_{\mathcal{V}} \tag{$\mathcal{V}$.x}$$

$$\left.\begin{array}{l} v_1 \wedge v_2 \\ v_2 \prec v_1 \end{array}\right\} \longrightarrow (-1) \odot (v_2 \wedge v_1) \tag{$\mathcal{V}$.xi}$$

$$\begin{cases} (a \odot v_1) \wedge v_2 \longrightarrow a \odot (v_1 \wedge v_2) \\ v_1 \wedge (a \odot v_2) \longrightarrow a \odot (v_1 \wedge v_2) \end{cases} \quad (\mathcal{V}.\text{xii})$$

$$\begin{cases} < v, 0_\nu > \longrightarrow 0 \\ < 0_\nu, v > \longrightarrow 0 \end{cases} \quad (\mathcal{V}.\text{xiii})$$

$$\left.\begin{array}{c} < v_1, v_2 > \\ v_2 \prec v_1 \end{array}\right\} \longrightarrow < v_2, v_1 > \quad (\mathcal{V}.\text{xiv})$$

$$\begin{cases} < (a \odot v_1), v_2 > \longrightarrow a* < v_1, v_2 > \\ < v_1, (a \odot v_2) > \longrightarrow a* < v_1, v_2 > \end{cases} \quad (\mathcal{V}.\text{xv})$$

$$\begin{cases} < (v_1 \wedge v_2), v_1 > \longrightarrow 0_\nu \\ < (v_1 \wedge v_2), v_2 > \longrightarrow 0_\nu \\ < v_1, (v_1 \wedge v_2) > \longrightarrow 0_\nu \\ < v_2, (v_1 \wedge v_2) > \longrightarrow 0_\nu \end{cases} \quad (\mathcal{V}.\text{xvi})$$

Les règles suivantes correspondent à de la distributivité et ne sont pas appliquées par défaut. Elles sont disponibles grâce à la fonction `expand`.

$$a \odot (v_1 \oplus v_2) \longrightarrow (a \odot v_1) \oplus (a \odot v_2) \quad (\mathcal{D}.\text{iv})$$

$$(v_1 \oplus v_2) \wedge v_3 \longrightarrow v_1 \wedge v_3 \oplus v_2 \wedge v_3 \quad (\mathcal{D}.\text{v})$$

$$< (v_1 \oplus v_2), v_3 > \longrightarrow < v_1, v_3 > + < v_2, v_3 > \quad (\mathcal{D}.\text{vi})$$

Il est aussi possible de déclarer que certains vecteurs ont des propriétés particulières, et notamment de déclarer que deux vecteurs sont colinéaires, ou sont orthogonaux, ou encore que trois vecteurs forment une base orthonormée. Ceci peut se faire en utilisant les possibilités classiques de *MAPLE*, par assignation du résultat d'une expression fonctionnelle. Par exemple `vprod(v1, v2) = 0` exprime que `v1` et `v2` sont colinéaires (on aurait pu aussi l'exprimer par `v1 := vsmult(k, v2)`), `vscal(v1, v2) = 0` exprime que `v1` et `v2` sont orthogonaux. L'assignation d'une norme entraîne la prise en compte du produit scalaire correspondant i.e. après `vnorm(v2) := 1` le produit scalaire de `v2` par lui même vaudra 1. On peut aussi utiliser la fonction `qtrans` qui permet de manipuler un ensemble de telles propriétés d'une manière plus souple (on peut revenir en arrière plus facilement, tester les effets de plusieurs ensembles d'hypothèses...).

quaternions et rotations

L'ensemble des quaternions unitaires (de norme 1) qui est isomorphe à la sphère de dimension 3 a une structure de groupe multiplicatif (pour $\boxtimes$).
En tant que tel, il opère sur le corps des quaternions par :

$$(\lambda, q) \longmapsto \lambda \boxtimes q \boxtimes \lambda^{[-1]}$$

A chaque quaternion unitaire, on peut donc associer un automorphisme intérieur du corps des quaternions.
On démontre que cet automorphisme laisse $\mathbb{R}^3$ (en tant qu'ensemble des quaternions purs) invariant et que sa restriction sur cet ensemble est une rotation de $\mathbb{R}^3$.
La réciproque est vraie, à toute rotation r de $\mathbb{R}^3$, on peut associer un quaternion unitaire λ tel que pour tout vecteur v, on ait :

$$r(v) = \lambda \boxtimes (0, v) \boxtimes \lambda^{[-1]}$$

Notons toutefois que λ et $-1 \boxdot \lambda$ sont associés à la même rotation i.e que $SO(3)$ est isomorphe au quotient du groupe des quaternions unitaires par le groupe $\{-1, 1\}$.
Si l'on connaît une rotation par un vecteur unitaire u de son axe et la mesure θ de son angle, un des quaternions unitaires associés à cette rotation est :

$$\lambda : \begin{pmatrix} \cos(\theta/2) \\ \sin(\theta/2) \odot u \end{pmatrix}$$

Si l'on connaît une rotation par un des quaternions unitaires $\lambda : (\lambda_0, \lambda_1, \lambda_2, \lambda_3)$ qui lui sont associés, sa matrice dans la base (i, j, k) s'écrit :

$$\begin{pmatrix} {\lambda_0}^2 + {\lambda_1}^2 - {\lambda_2}^2 - {\lambda_3}^2 & 2(\lambda_0\lambda_3 + \lambda_1\lambda_2) & 2(\lambda_1\lambda_3 - \lambda_0\lambda_2) \\ 2(\lambda_1\lambda_2 - \lambda_3\lambda_0) & {\lambda_0}^2 - {\lambda_1}^2 + {\lambda_2}^2 - {\lambda_3}^2 & 2(\lambda_0\lambda_1 + \lambda_2\lambda_3) \\ 2(\lambda_0\lambda_2 + \lambda_1\lambda_3) & 2(\lambda_2\lambda_3 + \lambda_1\lambda_3) & {\lambda_0}^2 - {\lambda_1}^2 - {\lambda_2}^2 + {\lambda_3}^2 \end{pmatrix}$$

Réciproquement, si l'on connaît une rotation par sa matrice M, on sait que l'on peut obtenir un vecteur unitaire u de son axe et une mesure θ de son angle par les formules suivantes :

$$\begin{cases} \mathit{trace}(M) = 1 + 2\cos(\theta) \\ \dfrac{1}{2}(M - M^t) = \sin(\theta)\tilde{u} \end{cases}$$

(où l'on note $\sim$ l'opérateur qui à un vecteur u associe la matrice $\tilde{u}$ telle que pour tout vecteur v, on ait $\tilde{u}.v = u \wedge v$), ce qui donne un des quaternions unitaires $\lambda : (s, v)$ représentant la rotation :

$$\begin{cases} s = \dfrac{1}{2}\sqrt{1 + \mathit{trace}(M)} \\ \tilde{v} = \dfrac{1}{2\sqrt{1 + \mathit{trace}(M)}}(M - M^t) \end{cases}$$

On trouve dans `QUATERMAN` un ensemble de procédures qui permettent ainsi de faire l'interface entre les rotations, les matrices et les quaternions : trouver un quaternion représentant la rotation d'angle et d'axe donnés (`rot2quat`, passage inverse (`quat2rot`), passage d'une matrice de rotation au quaternion correspondant (`mat2quat`)...

3.5 un manipulateur-simplificateur d'expressions matricielles

On a vu, dans la section précédente, comment il est possible d'utiliser un système de calcul formel classique pour manipuler des matrices et des vecteurs de manière relativement satisfaisante.

Néanmoins, si on a pu prendre en compte, dans la simplification des expressions en question, la plupart des propriétés standards des opérations (commutativité, associativité, élément neutre, symétrique, ...), cela a nécessité de gros efforts de programmation.

De même, l'adjonction de nouvelles propriétés aux opérateurs ou l'introduction de nouvelles règles de calcul nécessiterait, non seulement l'écriture de nouvelles procédures, mais aussi de reconsidérer et de modifier la plupart des procédures existantes.

De plus, les stratégies de simplification choisies ont été déterminées une fois pour toute et vont parfois à l'encontre des résultats souhaités :

> Ainsi, bien que la multiplication de matrice soit distributive par rapport à l'addition, $A*A+A*B+B*A+B*B$ n'est pas une expression plus simple que $(A+B)(A+B)$, alors que la forme la plus simple de $(A+B)(A+B)-A*B-A*A$, est bien $B*A+B*B$.

Enfin, on n'a pas pu, faute d'un mécanisme de typage des variables performant, attacher des propriétés spécifiques à certaines expressions, par exemple écrire que $A^{-1} = {}^{T}A$ si A est une matrice de rotation.

Ces inconvénients des systèmes de calcul formel classique ont motivé des réflexions théoriques sur la simplification en calcul formel – voir [Faure, 1992] – qui ont débouché, dans le projet *SAFIR*, à l'*INRIA*, sur l'écriture d'un prototype de système de calcul formel, *ULYSSE* dédié à la manipulation d'expressions et à la simplification.

On présente ci-dessous, après avoir introduit la philosophie de ce logiciel, une implémentation des manipulations matricielles utiles à la génération automatique des équations de la dynamique des mécanismes.

3.5.1 calcul équationnel, réécriture, et simplification modulo propriétés

Manipuler des expressions, en calcul formel, c'est effectuer du *calcul équationnel*, c'est à dire calculer dans l'ensemble des expressions modulo des équations.

> Ainsi, calculer l'expression $0 + sin(x)$ en $sin(x)$ c'est calculer $0 + sin(x)$ modulo l'équation $0 + X = X$.

Dans les systèmes de calcul formel, les hypothèses qui sont faites sur les opérateurs de base peuvent être exprimées sous forme d'équations, par exemple $X + 0 = X$, $0 + X = X$, $X + Y = Y + X$, $X + -X = 0$, $-X + X = 0$ ...

On a l'habitude, en calcul équationnel, d'utiliser le vocabulaire suivant :

> Une expression est appelée *terme* et est vue comme un arbre dont les noeuds sont étiquetés par des symboles d'opérateurs et les feuilles par des symboles de constantes ou des symboles de variables. Le symbole qui étiquette la racine de l'arbre s'appelle *symbole de tête* du terme (ou *opérateur de tête*). A un sous-arbre correspond alors un *sous-terme*.

réécriture

Pour arriver à calculer un terme modulo des équations, il faut choisir, à chaque étape, une bonne équation, et l'appliquer dans le bon sens et au bon sous-terme. Le calcul équationnel ne peut donc être automatisé facilement, c'est pourquoi on introduit la *réécriture*, calcul équationnel dans lequel les équations sont orientées et qui peut être automatisé.

Définition 3.2 *On appelle règle de réécriture une équation orientée – ainsi l'équation* $0+X = X$ *donne la règle* $0+X \longrightarrow X$ *– Un ensemble de règles de réécriture est appelé système de réécriture.*

Réécrire un terme t consiste à choisir un sous terme t' de t et une règle de réécriture $G \longrightarrow D$, à vérifier que t' est une instance de G (on dit aussi que G filtre t') puis à remplacer t' dans t par l'instance correspondante de D. Instancier G ou D consiste à remplacer chaque variable par un terme (sa valeur).

Dans ce contexte, calculer un terme t consiste à le réécrire jusqu'à l'obtention d'un terme *irréductible* – qui ne peut être réécrit par aucune règle du système – appelé *forme normale* de t. La réécriture est un processus non déterministe : un terme pourra (en général) avoir plusieurs formes normales selon la succession de réécritures effectuées. Si le résultat de la réécriture d'un terme est indépendant du choix de la règle appliquée, le système est dit *confluent*.

De plus, il est possible que certains calculs ne terminent pas. Si tous les calculs sont finis, le système a la propriété de *terminaison*. Pour orienter une équation on utilise un ordre sur les termes. Une équation $t_1 = t_2$ est orientée en $t_1 \longrightarrow t_2$ si t_1 est "plus grand" que t_2. Intuitivement, si une réécriture augmente la complexité des termes donc est mal orientée, les calculs peuvent ne pas terminer. Lorsqu'un système termine et est confluent, on a une et une seule forme normale pour chaque terme dite *forme canonique*, et le système est alors dit *convergent*.

Malheureusement, toutes les équations ne sont pas orientables. Parmi les équations usuelles, la commutativité, par exemple, $X+Y = Y+X$ est non orientable, et si un opérateur est commutatif alors l'équation signifiant son associativité est elle aussi non orientable. Pour prendre en compte ces équations, il faut faire de la *réécriture équationnelle*, c'est à dire réécrire non plus des termes, mais des *termes généralisés*, i.e. des classes de termes modulo ces équations.

Les équations usuelles modulo lesquelles se fait la réécriture correspondent à l'associativité et à la commutativité, le cas échéant, et des algorithmes de filtrage modulo associativité/commutativité ont été définis.

simplification modulo propriétés

Parmi les règles de réécriture courantes, il est aisé de voir émerger des modèles. Par exemple, les règles $+(X,0) \longrightarrow X$ et $*(X,1) \longrightarrow X$ peuvent être considérées comme des instances du modèle de règles $f(X, \text{neutre}) \longrightarrow X$. A chacun de ces modèles est associé un nom appelé *propriété* et des paramètres – au modèle précédent correspondra la propriété *possède-neutre-à-gauche*. Ainsi, il sera équivalent de déclarer *possède-neutre-à-gauche*$(+,0)$ ou $+(X,0) \longrightarrow X$.

Pour simplifier un terme modulo un ensemble de propriétés il faut

- associer à chaque propriété un algorithme de simplification en observant la règle équivalente à la propriété,
- définir un ordre sur ces propriétés pour utiliser successivement les algorithmes de simplification correspondant.

3.5.2 le prototype *ULYSSE*

ULYSSE est un prototype de système de calcul formel écrit en *LISP* qui permet de définir des objets mathématiques typés et de manipuler ces objets modulo propriétés.

On peut ainsi définir des opérateurs en leur associant une ou plusieurs des propriétés élémentaires suivantes :

i. associativité,

$$\text{associative}(f) \ : \ f(X, f(Y,Z)) = f(f(X,Y), Z)$$

ii. g-symétrie, i.e commutativité ou anticommutativité

$$\text{commutative}(f) \ : \ f(X,Y) = f(Y,X)$$

$$\text{anticommutative}(f) \ : \ f(X,Y) = -f(Y,X)$$

iii. opérateur externe,

$$\text{external–operator}(f,g) \ : \ f(X,f(Y,Z)) \longrightarrow f(g(X,Y),Z)$$

iv. n-homogénéité,

$$n\text{–homogeneous} \ : \ f(X_1,\ldots,X_{i-1},g(Y,X_i),X_{i+1},\ldots,X_n) \longrightarrow h(Y,f(X_1,\ldots,X_n))$$

v. $n \times m$-morphisme,

$$\begin{array}{l} n \times m\text{–morphism} \ : \\ \quad f(X_1,\ldots,X_{i-1},g(X_{i1},\ldots,X_{im}),X_{i+1},\ldots,X_n) \longrightarrow \\ \qquad h(f(X_1,\ldots,X_{i-1},X_{i1},X_{i+1},\ldots,X_n),\ldots f(X_1,\ldots,X_{i-1},X_{im},X_{i+1},\ldots,X_n)) \end{array}$$

vi. projecteur,

$$f(f(X)) \longrightarrow f(X)$$

vii. involution,

$$f(f(X)) \longrightarrow X$$

viii. nilpotence,

$$f(f(\ldots f(X)\ldots)) \longrightarrow X$$

ix. idempotence,

$$f(X,X) \longrightarrow X$$

x. élément neutre (à gauche ou à droite),

$$f(\text{neutre},X) \longrightarrow X \quad \text{ou} \quad f(X,\text{neutre}) \longrightarrow X$$

xi. élément symétrique (à gauche ou à droite),

$$f(h(X),X) \longrightarrow \text{neutre} \quad \text{ou} \quad f(X,h(X)) \longrightarrow \text{neutre}$$

xii. élément absorbant (à gauche ou à droite).

$$f(\text{abs},X) \longrightarrow \text{abs} \quad \text{ou} \quad f(X,\text{abs}) \longrightarrow \text{abs}$$

Ces propriétés élémentaires sont les plus couramment utilisées en algèbre, et correspondent aux axiomes équationnels habituellement utilisés en réécriture et en réécriture équationnelle. Elles permettent de définir toutes les propriétés algébriques utilisées dans les systèmes de calcul formel classiques.

Ces propriétés ont été réparties en classes de manière à ce que toutes les propriétés appartenant à la même classe aient des algorithmes de simplification semblables.

- Les propriétés de classe 1 sont l'associativité et la g-symétrie. Elles sont utilisées sous forme d'équation et servent de contexte aux autres propriétés.

- Les propriétés de classe 2 – (iii), (iv) et (v) – sont celles dont l'utilisation implique la recherche d'un sous-terme de forme donnée et transforme le sous-terme direct en un terme non encore simplifié,
- Les propriétés de classe 3 – (vi), (vii) et (viii) – sont celles dont l'utilisation implique la recherche d'un sous-terme de forme donnée et transforme le sous-terme en un terme simplifié,
- Les propriétés de classe 4 – (ix) à (xv) – sont celles dont l'utilisation implique la recherche de plusieurs sous-termes de formes données et transforme le sous-terme en un terme déjà simplifié.

Lorsqu'on souhaite associer plusieurs de ces propriétés à un même opérateur, on peut être confronté à des conflits ou à des incompatibilités.

La compatibilité des propriétés implémentées en *ULYSSE* a été étudiée et des listes de propriétés valides maximales ont été définies.

Pour chacune de ces listes, un ordre a été défini, pour déterminer l'emploi successif des algorithmes de simplification correspondant.

Ainsi, on peut, dans *ULYSSE*, définir n'importe quelles règles de manipulation sur des expressions, à partir des propriétés élémentaires, et il sera engendré automatiquement le simplificateur correspondant utilisant un algorithme de simplification spécifique.

3.5.3 les expressions vectorielles et matricielles en *ULYSSE*

On présente ici les fonctionnalités correspondant à la manipulation d'expressions vectorielles et matricielles dans *ULYSSE*. Le code, en lui-même se trouve en anexe (§B.3)

les opérateurs

On retrouve les opérateurs classiques agissant selon le cas sur les matrices, les vecteurs, et les scalaires à savoir :

- l'addition, `+`, n-aire, associative, commutative, possédant un neutre `_0`, un symétrique défini par l'opérateur `-`,
- l'opérateur `-`, unaire et involutif,
- le produit sur les scalaires `*`, n-aire, associatif, commutatif, possédant un neutre `1`, et un élément absorbant `0`, distributif par rapport à `+` et à `-`, homogène par rapport à la multiplication `*e` d'une matrice par un scalaire,
- le produit `.` n-aire, associatif, possédant un élément neutre à gauche et à droite `Id`, un élément absorbant à gauche et à droite noté encore `_0`, un symétrique à droite et à gauche défini par l'opérateur `inv`, homogène par rapport à `*e`, et distributif par rapport à `+` et à `-`,
- l'inverse de `.` noté `inv`, unaire et involutif,
- la transposition `trans`, unaire, involutive, homogène par rapport à `*e`, linéaire par rapport à `+` et `-`,
- l'opérateur *tilde* `tild`, unaire, homogène par rapport à `*e` et linéaire par rapport à `+` et `-`,

les simplifications élémentaires

Les simplifications élémentaires qui n'ont pas été prises en compte par les propriétés des opérateurs ci-dessus, et notamment celles qui définissent l'opérateur `*e` de multiplication d'une matrice par un scalaire sont données sous la forme d'un système de règles :

```
$ proprietes de *e
#x *e (- #y) -> (- #x) *e #y ,
(#x *e (#y *e #z)) -> (#x * #y) *e #z ,
- (#x *e #y) -> (- #x) *e #y ,
#x *e #y + #z *e #y -> (#x + #z) *e #y ,

$ quelques regles complementaires
inv(#x . #y) -> inv(#y) . inv(#x) ,
trans(#x . #y) -> trans(#y) . trans(#x) ,
tild(#a) . #b + tild(#b) . #a -> _0 ,
trans(#x . #y) -> trans(#y) . trans(#x) ,
trans(tild(#v)) -> - tild(#v) ,
tild(#v) . #v -> _0 ,

$ des regles sur les vecteurs de base
tild(X0) . Z0 -> -Y0 ,
tild(X0) . Y0 -> Z0 ,
tild(Y0) . Z0 -> X0 ,
tild(Y0) . X0 -> -Z0 ,
tild(Z0) . X0 -> Y0 ,
tild(Z0) . Y0 -> -X0 ,

$ des proprietes de _0
tild(_0) -> _0 ,
#x *e _0 -> _0 ,
trans(#x) . tild(#x) -> _0 ,

$ le double produit vectoriel
tild(#a) . tild(#b) . tild(#b) . #a
  + tild(#b) . tild(#a) . tild(#a) . #b -> _0 ,

#x *e (tild(#a) . tild(#b) . tild(#b) . #a)
  + #x *e (tild(#b) . tild(#a) . tild(#a) . #b) -> _0
```

le typage des variables

Pour typer les variables à l'intérieur des règles, on utilise des règles d'inférence de type :

```
tild(#x):#x1 . #y:(antisym & #y1) . tild(#x):#x2
 -> (tild(#x):#x1 . #y:(antisym & #y1) . tild(#x):#x2):(matrix & antisym),

tild(#x) -> tild(#x:vector):(matrix & antisym)
```

On a alors des règles de simplification portant sur des objets typés comme possédant des propriétés particulières :

```
(trans(#a:vector) . #b:vector -> trans(#b:vector) . #a:vector) when (#a > #b),

(tild(#a:vector):#x . #b:vector -> - (tild(#b:vector):#x . #a:vector))
  when (#a > #b),

tild(#a):#p1 . tild(#b):#p2 . (#c:vector) ->
```

```
  (trans(#a) . #c) *e #b + -(trans(#a) . #b) *e #c,

#x . tild(#a) . #b:vector + #x . tild(#b) . #a:vector -> _0,

#y *e (#x . tild(#a) . #b:vector) + #y *e (#x . tild(#b) . #a:vector) -> _0,

#y *e (tild(#a) . #b:vector) + #y *e (tild(#b) . #a:vector) -> _0,

trans(#x:(sym & #y)) -> #x:(sym & #y),
trans(#x:(antisym & #x1)) -> -(#x:(antisym & #x1)),
inv(#x:(ortho & #y)) -> trans(#x:(ortho & #y)),
trans(#x:vector) . #y:(antisym & #z) . #x:vector -> _0,
trans(#x:(antisym & #y)) -> - #x:(antisym & #y)
```

un exemple

Au cours d'une session de mise en équation d'un mécanisme – en utilisant le logiciel *GEMMES* – on a rencontré l'expression suivante :

$$-\left(\dot{\Theta}_1^2\left({}^{T}R_1.\widetilde{R_1}.I_{10}.R_1\right)+\left(-m_{10}\dot{\Theta}_1^2\left({}^{T}R_1\widetilde{r_{10}}\widetilde{R_1}\widetilde{R_1}y_{10}\right)\right)\right.$$
$$\left.+\left(-\dot{\Theta}_1\left(m_{10}{}^{T}R_1\widetilde{r_{10}}+\left(-m_{10}{}^{T}R_1\widetilde{y_{10}}\right)\right)\left(\dot{\Theta}_1\widetilde{R_1}\widetilde{r_{10}}+\left(-\dot{\Theta}_1\widetilde{r_{10}}\widetilde{R_1}\right)\right)R_1\right)\right)$$

où R_1, r_{10} et y_{10} sont des vecteurs et qui n'a pas été simplifiée alors qu'elle est nulle :

```
> R1 -> R1:vector :
> r10 -> r10:vector :
> y10 -> y10:vector :
> exp ;

 - (smul(THp1*THp1 ,
        trans(R1) . tild(R1) . In10 . R1)

    + -smul(m10* THp1*Thp1 ,
    trans(R1). tild(r10). tild(R1).tild(R1). y10)

    + -smul(THp1,

    (smul(m10, trans(R1) . tild(r10))

             + - smul(m10, trans(R1) . tild(y10)))

    . (smul(THp1, tild(R1). tild(r10))

       + - smul(THp1, tild(r10) . tild(R1)))

    . R1))

> rewrite(calcprop(‘exp),‘rsglobal) ;
                                          _0
```

Chapitre 4

traitement des mécanismes bouclés

On a vu, dès le premier chapitre de cette étude, combien intervient la nature bouclée ou arborescente de l'architecture d'un mécanisme dans son comportement dynamique, et surtout dans l'approche mécanique et mathématique que l'on peut avoir de la modélisation de ce comportement.

Dans le cas d'un mécanisme à structure arborescente, on peut scinder le procédé de modélisation dynamique en deux étapes :

i. la détermination de la variété de configurations admissibles,

ii. la détermination effective du mouvement en tant qu'application de l'intervalle de temps considéré vers cette variété.

Mais dans le cas d'un mécanisme à structure bouclée, cette scission n'est pas, a priori, directement envisageable, puisque les hypothèses du théorème (1.41) qui assure l'existence de la variété des configurations admissibles ne sont pas satisfaites.

Ainsi, se trouve posé avant même le problème de la détermination du degré de liberté d'un mécanisme et d'un paramétrage adéquat, le problème de l'existence – i.e. de la définition – du degré de liberté d'un mécanisme à structure bouclé et d'un tel paramétrage.

Essentiellement, dans l'optique que l'on s'est fixée, les problèmes à résoudre sont de trois ordres et se formulent en ces termes :

i. Considérant un mécanisme dont on connaît simplement la nature des liaisons, mais dont on sait qu'il possède, au moins localement, un degré de liberté – voire nul – comment déterminer ce degré de liberté ?

ii. Considérant un mécanisme dont on connaît la nature des liaisons, et dont on connaît, au moins localement, le degré de liberté, comment trouver un paramétrage, même partiel de sa variété de configurations ?

iii. Considérant un mécanisme dont on connaît la nature des liaisons, dont on connaît, au moins localement, le degré de liberté, et dont on sait paramétrer, au moins en partie, la variété de configurations, comment engendrer systématiquement les équations de la dynamique de ce mécanisme ?

Dans un premier temps, on va approfondir les concepts mathématiques sous-jacents aux notions de "degré de liberté" et de "paramétrage" pour déterminer quelles sont les hypothèses et les restrictions que l'on devra assumer pour mener à bien la modélisation dynamique des mécanismes considérés.

On verra ensuite différentes approches mises en œuvre pour aborder, de manière systématique, dans le cadre de ces hypothèses et de ces restrictions :

- la détermination du degré de liberté d'un mécanisme,
- la détermination d'un paramétrage de sa variété de configurations,

- et, les deux points précédents étant supposés résolus, la génération des équations différentielles qui gouvernent son mouvement.

4.1 description des problèmes

On considère un mécanisme $\mathcal{S}$ constitué de n corps $\mathcal{S}_1$, ..., $\mathcal{S}_n$ dont les liaisons sont holonomes régulières et dont la structure est bouclée.

On note $\mathcal{L}$ l'ensemble des liaisons de ce mécanisme et on partitionne $\mathcal{L}$ en deux ensembles :

- $\mathcal{L}_{\text{arb}}$ l'ensemble des liaisons d'un mécanisme à structure arborescente sous-jacent $\mathcal{S}_{\text{arb}}$ qui est constitué des mêmes corps que $\mathcal{S}$,
- et $\mathcal{L}_{\text{cmp}} = \{\ell_1, \ldots, \ell_\nu\}$ l'ensemble des liaisons complémentaires.

4.1.1 degré de liberté local

On introduit avec des notations habituelles :

- $\mathcal{V}'$, la variété de configurations primitive du mécanisme $\mathcal{S}$ de dimension $6n$,
- $\mathcal{V}_{\text{arb}}$, la variété des configurations admissibles du système arborescent sous-jacent $\mathcal{S}_{\text{arb}}$, sous-variété de $\mathcal{V}'$ – on note m sa dimension, i.e. le degré de liberté $\delta^l(\mathcal{S}_{\text{arb}})$,
- $\mathcal{V}$, l'*ensemble* – on ne peut pas, a priori, parler de variété – des configurations admissibles de $\mathcal{S}$.
- et, pour chaque liaison ℓ_i $(i = 1 \ldots \nu)$, contraignant les mouvements relatifs des corps $\mathcal{S}_{i_1}$ et $\mathcal{S}_{i_2}$,

  $\mathcal{V}_{\ell_i}$, la variété des configurations admissibles du système mécanique $\mathcal{S}_{\ell_i}$ constitué des mêmes corps que $\mathcal{S}$ et soumis à la seule liaison ℓ_i.

D'après le théorème (1.41), et les hypothèses de régularité faites sur les liaisons, on sait que :

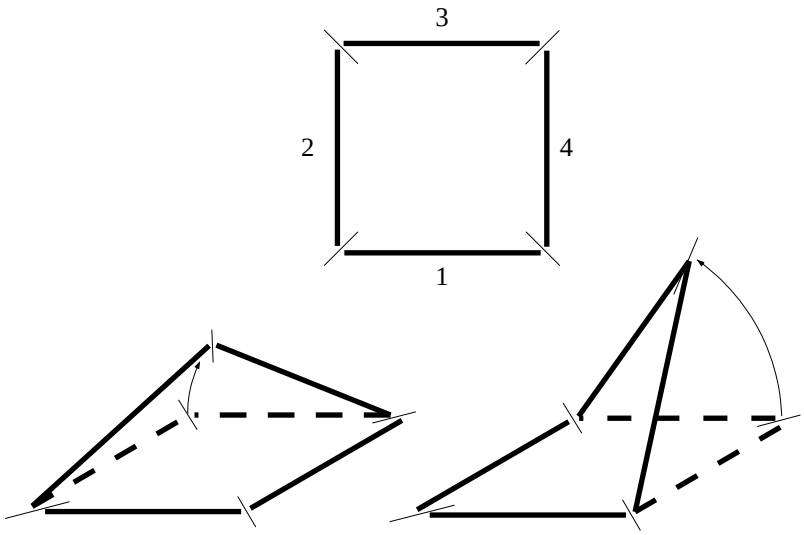
- $\mathcal{V}_{\text{arb}}$ est une sous-variété de $\mathcal{V}'$,
- $\mathcal{V}_{\ell_i}$ est une sous-variété de $\mathcal{V}'$ pour tout i tel que $1 \leq i \leq \nu$,
- et que :

$$\mathcal{V} = \mathcal{V}_{\text{arb}} \cap \left(\bigcap_{i=1}^{\nu} \mathcal{V}_{\ell_i} \right)$$

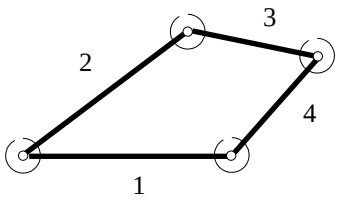
Il n'y a donc aucune raison pour que, en l'absence d'hypothèses supplémentaires, $\mathcal{V}$ soit une sous-variété de $\mathcal{V}'$, comme on va le voir sur les deux exemples suivants.

On considère deux mécanismes composés de quatre tiges reliées entre elles par quatre axes de rotation.

- dans le premier cas, la "salière", les tiges sont de même longueur et les axes sont orientés comme décrit sur les dessins ci-dessous :
  en haut est représenté le mécanisme en configuration plane, vu de dessus, et en bas deux configurations possibles,

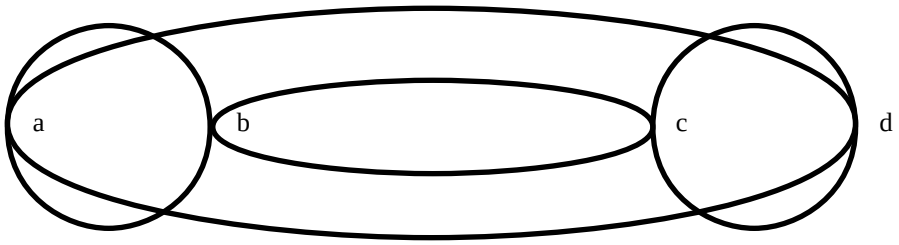


- dans le deuxième exemple, les tiges sont de longueurs égales deux à deux, les quatre axes sont parallèles et les tiges se meuvent dans le plan perpendiculaire à la direction des axes :

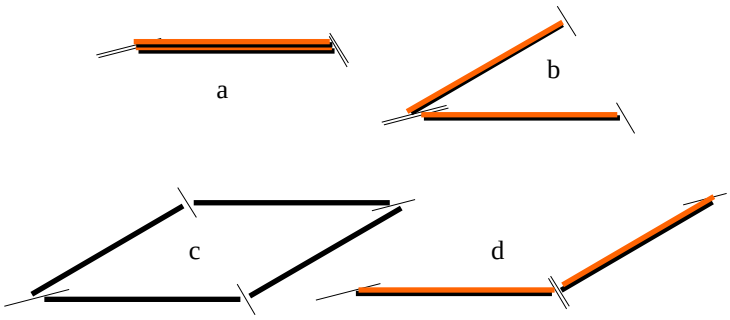


Si $\mathcal{S}_{\rm arb}$ est le système mécanique obtenu en ne considérant pas la liaison $\mathcal{S}_{\ell_1}$ entre le corps $\mathcal{S}_1$ et le corps $\mathcal{S}_4$ et si, de plus, on suppose le corps $\mathcal{S}_1$ comme fixe alors $\mathcal{V}_{\rm arb}$ peut être vue comme le produit de trois cercles, $\mathcal{V}_1$ comme le produit de $(I\!R^3 \times {\rm SO}_3)^2$ par un cercle et $\mathcal{V}$ peut être représenté,

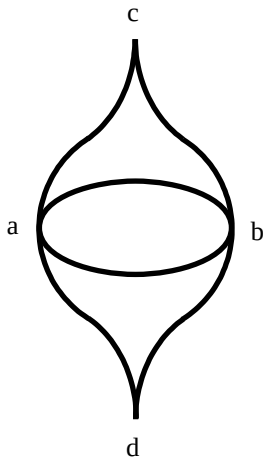
- dans le premier cas par le dessin ci-dessous,



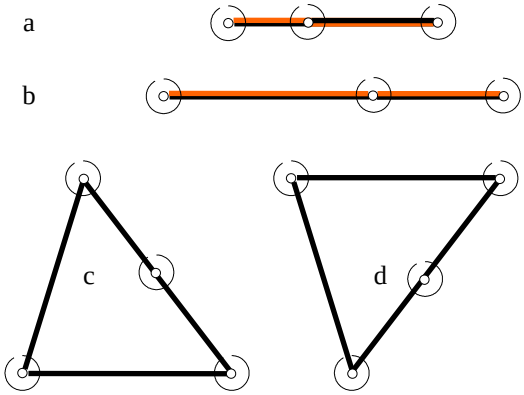
les points a, b, c, et d correspondant aux configurations :



- dans le deuxième cas par :



les points a, b, c, et d correspondant aux configurations :



Ainsi, il existe des *lieux singuliers* où $\mathcal{V}$ n'est pas une sous-variété de $\mathcal{V}'$. En ces lieux, évoquer le degré de liberté du mécanisme n'a pas de sens. On se limitera donc à étudier les mouvements des mécanismes qui ne "passent" pas par de telles configurations singulières.

Pour la suite de cette étude, on va ainsi écarter délibérément les mécanismes tels que l'ensemble des lieux singuliers est "trop grand", et on ne s'intéressera qu'aux mécanismes satisfaisant la définition suivante ;

Définition 4.1 *Soit $\mathcal{S}$ un mécanisme, $\mathcal{V}'$ sa variété primitive de configurations, $\mathcal{L}$ l'ensemble de ses liaisons, $\mathcal{V}$ l'ensemble de ses configurations admissibles, et, pour toute liaison $\ell_i \in \mathcal{L}$, $\mathcal{V}_{\ell_i}$*

l'ensemble des configurations admissibles du système $\mathcal{S}_{\ell_i}$ constitué des mêmes corps que $\mathcal{S}$ et soumis exclusivement à la liaison ℓ_i, alors, on dit de $\mathcal{S}$ que c'est un mécanisme régulier si et seulement si, il existe une configuration ϕ^0 de $\mathcal{V}$ et un voisinage ouvert V de ϕ^0 dans $\mathcal{V}'$ tel que,

> *pour tout sous-ensemble $\mathcal{M}$ de $\mathcal{L}$, $V \cap (\bigcap_{\ell_i \in \mathcal{M}} \mathcal{V}_{\ell_i})$ est non vide et est une sous-variété de $\mathcal{V}'$.*

En particulier un mécanisme régulier satisfait l'hypothèse :

> Il existe un voisinage V, dans $\mathcal{V}'$, d'une configuration de $\mathcal{V}$ tel que $V \cap \mathcal{V}$ soit une sous-variété de $\mathcal{V}'$.

ce qui permet localement de définir son degré de liberté.

Dans les chapitres précédents, on a toujours considéré une seule carte pour représenter les différentes variétés et sous-variétés rencontrées, en étant conscient de se restreindre à l'étude de mouvements dont les images, en tant qu'application de t dans $\mathcal{V}'$, restaient dans le domaine de validité de la carte en question – voir remarque (1.29).

Dorénavant, on se restreindra, de plus, à l'étude de mouvements dont les images restent dans un voisinage V de la définition (4.1) sur lequel la variété des configurations admissibles du mécanisme est une sous-variété.

De manière abusive, on conservera les notations habituelles, et en particulier $\mathcal{V}'$, $\mathcal{V}_{\text{arb}}$ et $\mathcal{V}$ désigneront en fait respectivement $\mathcal{V}' \cap V$, $\mathcal{V}_{\text{arb}} \cap V$ et $\mathcal{V} \cap V$.

4.1.2 détermination du degré de liberté

On utilise pour paramétrer $\mathcal{V}'$ une carte $(\mathfrak{p}_{1,1}, \ldots, \mathfrak{p}_{1,6}, \mathfrak{p}_{2,1}, \ldots, \mathfrak{p}_{n,1}, \ldots, \mathfrak{p}_{n,6})$ dans $\mathbb{R}^{6n}$. On suppose, quitte à réduire le voisinage V, que le domaine de validité de cette carte contient V. On utilise de plus un jeu de paramètres cinématiques $(\mathfrak{q}_{1,1}, \ldots, \mathfrak{q}_{1,6}, \mathfrak{q}_{2,1}, \ldots, \mathfrak{q}_{n,1}, \ldots, \mathfrak{q}_{n,6})$ pour décrire, en tout point de V les espaces tangents à $\mathcal{V}'$.

Similairement, on utilise pour paramétrer $\mathcal{V}_{\text{arb}}$ une carte $(p_{1,1}, \ldots, p_{1,j_1}, \ldots, p_{2,1}, \ldots, p_{n,j_n})$ dans $\mathbb{R}^m$ (avec $m = \sum_{i=1}^{n} j_i$). On suppose encore, quitte à réduire de nouveau le voisinage V, que le domaine de validité de cette carte contient V.

Et on utilise aussi un jeu de paramètres cinématiques $(q_{1,1}, \ldots, q_{1,j_1}, \ldots, q_{2,1}, \ldots, q_{n,j_n})$ pour décrire, en tout point de V les espaces tangents à $\mathcal{V}_{\text{arb}}$.

On notera parfois, pour abréger, $\mathfrak{p}_i$ à la place de $(\mathfrak{p}_{i,1}, \ldots, \mathfrak{p}_{i,6})$, $\mathfrak{q}_i$ à la place de $(\mathfrak{q}_{i,1}, \ldots, \mathfrak{q}_{i,6})$, ou p_i à la place de $(p_{i,1}, \ldots, p_{i,j_i})$, ou encore q_i à la place de $(q_{i,1}, \ldots, q_{i,j_i})$.

A chaque *boucle* dans la structure du mécanisme correspond une liaison complémentaire ℓ_i, qui est holonome régulière et donc possède un degré de contrainte δ_i^c.

Cette liaison s'exprime, localement, dans la variété primitive de configurations du mécanisme par l'existence de δ_i^c fonctions $f_{i1}, \ldots, f_{i\delta_i^c}$ de $\mathcal{V}'$ dans $\mathbb{R}$ telles que pour toute configuration admissible ϕ :

i. $f_{i1}(\phi) = \ldots = f_{i\delta_i^c}(\phi) = 0$,

ii. les différentielles de ces fonctions, en tant que formes linéaires sur l'espace tangent à $\mathcal{V}'$ en ϕ sont indépendantes.

Ceci peut se traduire par l'existence de δ_i^c fonctions $\overline{f}_{i1}, \ldots, \overline{f}_{i\delta_i^c}$ de $\mathbb{R}^{6n}$ dans $\mathbb{R}$ telles que, pour toute configuration admissible ϕ paramétrée par $\mathfrak{p} = (\mathfrak{p}_1, \ldots, \mathfrak{p}_n)$:

i. $\overline{f}_{i1}(\mathfrak{p}_1, \ldots, \mathfrak{p}_n) = \ldots = \overline{f}_{i\delta_i^c}(\mathfrak{p}_1, \ldots, \mathfrak{p}_n) = 0$,

ii. la matrice jacobienne, $\left(\dfrac{\partial \overline{f}_i}{\partial \mathfrak{p}}(\mathfrak{p})\right)$ des $\overline{f}_i$ en $\mathfrak{p}$:

$$\begin{pmatrix} \dfrac{\partial \overline{f}_{i1}}{\partial \mathfrak{p}_{1,1}}(\mathfrak{p}) & \dots & \dfrac{\partial \overline{f}_{i1}}{\partial \mathfrak{p}_{1,6}}(\mathfrak{p}) & \dfrac{\partial \overline{f}_{i1}}{\partial \mathfrak{p}_{2,1}}(\mathfrak{p}) & \dots & \dfrac{\partial \overline{f}_{i1}}{\partial \mathfrak{p}_{n,6}}(\mathfrak{p}) \\ \vdots & & \vdots & \vdots & & \vdots \\ \dfrac{\partial \overline{f}_{i\delta_i^c}}{\partial \mathfrak{p}_{1,1}}(\mathfrak{p}) & \dots & \dfrac{\partial \overline{f}_{i\delta_i^c}}{\partial \mathfrak{p}_{1,6}}(\mathfrak{p}) & \dfrac{\partial \overline{f}_{i\delta_i^c}}{\partial \mathfrak{p}_{2,1}}(\mathfrak{p}) & \dots & \dfrac{\partial \overline{f}_{i\delta_i^c}}{\partial \mathfrak{p}_{n,6}}(\mathfrak{p}) \end{pmatrix}$$

est de rang maximum (i.e. δ_i^c).

Et, si l'on veut faire intervenir les paramètres cinématiques, dans la mesure où la matrice $\mathfrak{P}(\mathfrak{p})$ – telle que $\mathfrak{q} = \mathfrak{P}(\mathfrak{p})\dot{\mathfrak{p}}$ – est inversible pour toute configuration admissible ϕ paramétrée par $\mathfrak{p}$, la liaison peut se traduire par :

i. $\overline{f}_{i1}(\mathfrak{p}_1, \dots, \mathfrak{p}_n) = \dots = \overline{f}_{i\delta_i^c}(\mathfrak{p}_1, \dots, \mathfrak{p}_n) = 0$,

ii. la matrice $\mathfrak{D}(\mathfrak{p})$ définie par :

$$\mathfrak{D}_i(\mathfrak{p}) = \left(\frac{\partial \overline{f}_i}{\partial \mathfrak{p}}(\mathfrak{p})\right) \mathfrak{P}(\mathfrak{p})^{-1}$$

est de rang maximum (i.e. δ_i^c).

D'autre part, puisque $\mathcal{V}_{\text{arb}}$ est une sous-variété de $\mathcal{V}'$, et que $(p_{1,1}, \dots, p_{1,j_1}, p_{2,1}, \dots, p_{n,j_n})$ est une carte de $\mathcal{V}_{\text{arb}}$, il existe une carte de $\mathcal{V}'$ de la forme :

$$(p_{1,1}, \dots, p_{1,j_1}, \dots, p_{2,1}, \dots, p_{n,j_n}, \varpi_1, \dots, \varpi_{6n-m})$$

telle que, pour toute configuration ϕ de $\mathcal{V}'$ – dans le domaine de la carte :

$$\text{si} \quad \phi \in \mathcal{V}_{\text{arb}} \ , \quad \text{alors} \quad \varpi_1 = \dots = \varpi_{6n-m} = 0$$

Soit $\mathit{\Pi}$ le difféomorphisme de changement de carte :

$$\mathit{\Pi} \ : \ (p_{1,1}, \dots, p_{1,j_1}, \dots, p_{2,1}, \dots, p_{n,j_n}, \varpi_1, \dots, \varpi_{6n-m}) \longmapsto (\mathfrak{p}_{1,1}, \dots, \mathfrak{p}_{1,6}, \mathfrak{p}_{2,1}, \dots, \mathfrak{p}_{n,1}, \dots, \mathfrak{p}_{n,1})$$

on peut alors exprimer la liaison ℓ_i dans cette nouvelle carte en écrivant que, pour toute configuration admissible ϕ paramétrée par* $(p_1, \dots, p_n, \varpi)$:

i. en posant, pour tout j compris entre 1 et δ_i^c, $g_{ij} = \overline{f}_{ij} \circ \mathit{\Pi}$,

$$g_{i1}(p_1, \dots, p_n, \varpi) = \dots = g_{i\delta_i^c}(p_1, \dots, p_n, \varpi) = 0$$

ii. la matrice $D_i(p_1, \dots, p_n, \varpi)$, définie par :

$$\begin{pmatrix} \frac{\partial(\overline{f}_{i1} \circ \mathit{\Pi})}{\partial p_{1,1}}(p, \varpi) & \dots & \frac{\partial(\overline{f}_{i1} \circ \mathit{\Pi})}{\partial p_{n,j_n}}(p, \varpi) & \frac{\partial(\overline{f}_{i1} \circ \mathit{\Pi})}{\partial \varpi_1}(p, \varpi) & \dots & \frac{\partial(\overline{f}_{i1} \circ \mathit{\Pi})}{\partial \varpi_{6n-m}}(p, \varpi) \\ \vdots & & \vdots & \vdots & & \vdots \\ \frac{\partial(\overline{f}_{i\delta_i^c} \circ \mathit{\Pi})}{\partial p_{1,1}}(p, \varpi) & \dots & \frac{\partial(\overline{f}_{i\delta_i^c} \circ \mathit{\Pi})}{\partial p_{n,j_n}}(p, \varpi) & \frac{\partial(\overline{f}_{i\delta_i^c} \circ \mathit{\Pi})}{\partial \varpi_1}(p, \varpi) & \dots & \frac{\partial(\overline{f}_{i\delta_i^c} \circ \mathit{\Pi})}{\partial \varpi_{6n-m}}(p, \varpi) \end{pmatrix}$$

est de rang maximum (i.e. δ_i^c).

Puisque, par définition de (p, ϖ), on a $\varpi_1 = \dots = \varpi_{6n-m} = 0$ sur $\mathcal{V}_{\text{arb}}$, on peut naturellement caractériser $\mathcal{V}_{\text{arb}} \cap \mathcal{V}_{\ell_i}$ en tant que sous-variété de $\mathcal{V}_{\text{arb}}$ par la donnée :

- de la carte $(p_{1,1}, \dots, p_{1,j_1}, \dots, p_{2,1}, \dots, p_{n,j_n})$,

*On note, pour abréger, (p, ϖ) à la place de $(p_{1,1}, \dots, p_{1,j_1}, p_{2,1}, \dots, p_{n,j_n}, \varpi_1, \dots, \varpi_{6n-m})$ et ϖ à la place de $(\varpi_1, \dots, \varpi_{6n-m})$.

- et des δ_i^c conditions :

$$\left\{\begin{array}{c} g_{i1}(p_1,\ldots,p_n)=0 \\ \vdots \\ g_{i\delta_i^c}(p_1,\ldots,p_n)=0 \end{array}\right.$$

Ainsi, la dimension de $\mathcal{V}\cap\mathcal{V}_{\ell_i}$ est obtenue comme la différence entre la dimension de $\mathcal{V}_{\rm arb}$ et le rang de la matrice :

$$\left(\begin{array}{ccc} \dfrac{\partial g_{i1}}{\partial p_{1,1}}(p) & \ldots & \dfrac{\partial g_{i1}}{\partial p_{n,j_n}}(p) \\ \vdots & & \vdots \\ \dfrac{\partial g_{i\delta_i^c}}{\partial p_{1,1}}(p) & \ldots & \dfrac{\partial g_{i\delta_i^c}}{\partial p_{n,j_n}}(p) \end{array}\right)$$

ou, si l'on préfère faire intervenir les paramètres cinématiques, et dans la mesure où la matrice $P(p)$ – telle que $q=P(p)\dot{p}$ – est inversible, la dimension de $\mathcal{V}\cap\mathcal{V}_{\ell_i}$ est obtenue comme la différence entre la dimension de $\mathcal{V}_{\rm arb}$ et le rang de la matrice $\Delta_{(\mathcal{V}\cap\mathcal{V}_{\ell_i})}(p)$ définie par :

$$\Delta_{(\mathcal{V}\cap\mathcal{V}_{\ell_i})}(p) \quad = \quad \left(\begin{array}{ccc} \dfrac{\partial g_{i1}}{\partial p_{1,1}}(p) & \ldots & \dfrac{\partial g_{i1}}{\partial p_{n,j_n}}(p) \\ \vdots & & \vdots \\ \dfrac{\partial g_{i\delta_i^c}}{\partial p_{1,1}}(p) & \ldots & \dfrac{\partial g_{i\delta_i^c}}{\partial p_{n,j_n}}(p) \end{array}\right) P(p)^{-1}$$

Si l'on considère maintenant une seconde boucle, c'est à dire une seconde liaison complémentaire ℓ_k, on peut définir, comme on vient de le faire pour la liaison ℓ_i :

- δ_k^c fonctions $f_{k1},\ldots,f_{k\delta_k^c}$ de $\mathcal{V}'$ dans $\mathbb{R}$ telles que pour toute configuration ϕ admissible – pour la liaison ℓ_k :
  i. $f_{k1}(\phi)=\ldots=f_{k\delta_k^c}(\phi)=0$,
  ii. les différentielles de ces fonctions, en tant que formes linéaires sur l'espace tangent à $\mathcal{V}'$ en ϕ sont indépendantes.
- puis δ_k^c fonctions $\overline{f}_{k1},\ldots,\overline{f}_{k\delta_k^c}$ de $\mathbb{R}^{6n}$ dans $\mathbb{R}$ telles que, si la configuration ϕ est paramétrée par $\mathfrak{p}=(\mathfrak{p}_1,\ldots,\mathfrak{p}_n)$:
  i. $\overline{f}_{k1}(\mathfrak{p}_1,\ldots,\mathfrak{p}_n)=\ldots=\overline{f}_{k\delta_k^c}(\mathfrak{p}_1,\ldots,\mathfrak{p}_n)=0$,
  ii. la matrice jacobienne, $\left(\dfrac{\partial\overline{f}_k}{\partial\mathfrak{p}}(\mathfrak{p})\right)$ des $\overline{f}_k$ en $\mathfrak{p}$:

$$\left(\begin{array}{cccccc} \dfrac{\partial\overline{f}_{k1}}{\partial\mathfrak{p}_{1,1}}(\mathfrak{p}) & \ldots & \dfrac{\partial\overline{f}_{k1}}{\partial\mathfrak{p}_{1,6}}(\mathfrak{p}) & \dfrac{\partial\overline{f}_{k1}}{\partial\mathfrak{p}_{2,1}}(\mathfrak{p}) & \ldots & \dfrac{\partial\overline{f}_{k1}}{\partial\mathfrak{p}_{n,6}}(\mathfrak{p}) \\ \vdots & & \vdots & \vdots & & \vdots \\ \dfrac{\partial\overline{f}_{k\delta_k^c}}{\partial\mathfrak{p}_{1,1}}(\mathfrak{p}) & \ldots & \dfrac{\partial\overline{f}_{k\delta_k^c}}{\partial\mathfrak{p}_{1,6}}(\mathfrak{p}) & \dfrac{\partial\overline{f}_{k\delta_k^c}}{\partial\mathfrak{p}_{2,1}}(\mathfrak{p}) & \ldots & \dfrac{\partial\overline{f}_{k\delta_k^c}}{\partial\mathfrak{p}_{n,6}}(\mathfrak{p}) \end{array}\right)$$

  est de rang maximum (i.e. δ_k^c).

- et enfin les δ_k^c fonctions, $g_{k1}, \ldots, g_{k\delta_k^c}$ définies par :
  pour tout j compris entre 1 et δ_k^c,
  $$g_{kj} = \overline{f}_{kj} \circ \Pi$$
  et telles que :
  i. $g_{k1}(p_1, \ldots, p_n, \varpi) = \ldots = g_{k\delta_k^c}(p_1, \ldots, p_n, \varpi) = 0$
  ii. la matrice $D_k(p_1, \ldots, p_n, \varpi)$, définie par :
  $$\begin{pmatrix} \frac{\partial g_{k1}}{\partial p_{1,1}}(p,\varpi) & \ldots & \frac{\partial g_{k1}}{\partial p_{n,j_n}}(p,\varpi) & \frac{\partial g_{k1}}{\partial \varpi_1}(p,\varpi) & \ldots & \frac{\partial g_{k1}}{\partial \varpi_{6n-m}}(p,\varpi) \\ \vdots & & \vdots & \vdots & & \vdots \\ \frac{\partial g_{k\delta_k^c}}{\partial p_{1,1}}(p,\varpi) & \ldots & \frac{\partial g_{k\delta_k^c}}{\partial p_{n,j_n}}(p,\varpi) & \frac{\partial g_{k\delta_k^c}}{\partial \varpi_1}(p,\varpi) & \ldots & \frac{\partial g_{k\delta_k^c}}{\partial \varpi_{6n-m}}(p,\varpi) \end{pmatrix}$$
  est de rang maximum (i.e. δ_k^c).

Et on peut, comme précédemment, caractériser naturellement $\mathcal{V}_{\text{arb}} \cap \mathcal{V}_{\ell_k}$ par la donnée :

- de la carte $(p_{1,1}, \ldots, p_{1,j_1}, \ldots, p_{2,1}, \ldots, p_{n,j_n})$,
- et des δ_k^c conditions :
$$\begin{cases} g_{k1}(p_1, \ldots, p_n) = 0 \\ \vdots \\ g_{k\delta_k^c}(p_1, \ldots, p_n) = 0 \end{cases}$$

Alors la variété $\mathcal{V}_{\text{arb}} \cap \mathcal{V}_{\ell_i} \cap \mathcal{V}_{\ell_k}$ sera caractérisée par la donnée :

- de la carte $(p_{1,1}, \ldots, p_{1,j_1}, \ldots, p_{2,1}, \ldots, p_{n,j_n})$,
- et des $\delta_i^c + \delta_k^c$ conditions :
$$\begin{cases} g_{i1}(p_1, \ldots, p_n) = 0 \\ \vdots \\ g_{i\delta_i^c}(p_1, \ldots, p_n) = 0 \\ g_{k1}(p_1, \ldots, p_n) = 0 \\ \vdots \\ g_{k\delta_k^c}(p_1, \ldots, p_n) = 0 \end{cases}$$

Et la codimension de $\mathcal{V}_{\text{arb}} \cap \mathcal{V}_{\ell_i} \cap \mathcal{V}_{\ell_k}$, en tant que sous-variété de $\mathcal{V}_{\text{arb}}$, sera donnée par le rang de la matrice :
$$\begin{pmatrix} \frac{\partial g_{i1}}{\partial p_{1,1}}(p) & \ldots & \frac{\partial g_{i1}}{\partial p_{n,j_n}}(p) \\ \vdots & & \vdots \\ \frac{\partial g_{i\delta_i^c}}{\partial p_{1,1}}(p) & \ldots & \frac{\partial g_{i\delta_i^c}}{\partial p_{n,j_n}}(p) \\ \frac{\partial g_{k1}}{\partial p_{1,1}}(p) & \ldots & \frac{\partial g_{k1}}{\partial p_{n,j_n}}(p) \\ \vdots & & \vdots \\ \frac{\partial g_{k\delta_k^c}}{\partial p_{1,1}}(p) & \ldots & \frac{\partial g_{k\delta_k^c}}{\partial p_{n,j_n}}(p) \end{pmatrix}$$

ou, ce qui est équivalent, par le rang de la matrice $\Delta_{(\mathcal{V}\cap\mathcal{V}_{\ell_i}\cap\mathcal{V}_{\ell_k})}(p)$ définie par :

$$\Delta_{(\mathcal{V}\cap\mathcal{V}_{\ell_i}\cap\mathcal{V}_{\ell_k})}(p) \quad = \quad \begin{pmatrix} \dfrac{\partial g_{i1}}{\partial p_{1,1}}(p) & \dots & \dfrac{\partial g_{i1}}{\partial p_{n,j_n}}(p) \\ \vdots & & \vdots \\ \dfrac{\partial g_{i\delta_i^c}}{\partial p_{1,1}}(p) & \dots & \dfrac{\partial g_{i\delta_i^c}}{\partial p_{n,j_n}}(p) \\ \dfrac{\partial g_{k1}}{\partial p_{1,1}}(p) & \dots & \dfrac{\partial g_{k1}}{\partial p_{n,j_n}}(p) \\ \vdots & & \vdots \\ \dfrac{\partial g_{k\delta_k^c}}{\partial p_{1,1}}(p) & \dots & \dfrac{\partial g_{k\delta_k^c}}{\partial p_{n,j_n}}(p) \end{pmatrix} P(p)^{-1}$$

En itérant le procédé décrit ci-dessus pour toutes les liaisons complémentaires du mécanisme $\mathcal{S}$ étudié, on obtient la caractérisation de $\mathcal{V}$ en tant que sous-variété de $\mathcal{V}_{\rm arb}$ dans le domaine de la carte $(p_{1,1},\dots,p_{1,j_1},\dots,p_{2,1},\dots,p_{n,j_n})$:

$$\forall\phi\in\mathcal{V}_{\rm arb}, \qquad \phi\in\mathcal{V} \quad\Longleftrightarrow\quad \left\{ \begin{array}{l} g_{1,1}(p_1,\dots,p_n)=0 \\ \vdots \\ g_{1\delta_1^c}(p_1,\dots,p_n)=0 \\ g_{2,1}(p_1,\dots,p_n)=0 \\ \vdots \\ \vdots \\ g_{\nu\delta_\nu^c}(p_1,\dots,p_n)=0 \end{array} \right.$$

Et la codimension de $\mathcal{V}$ en tant que sous-variété de $\mathcal{V}_{\rm arb}$, est donnée par le rang de la matrice :

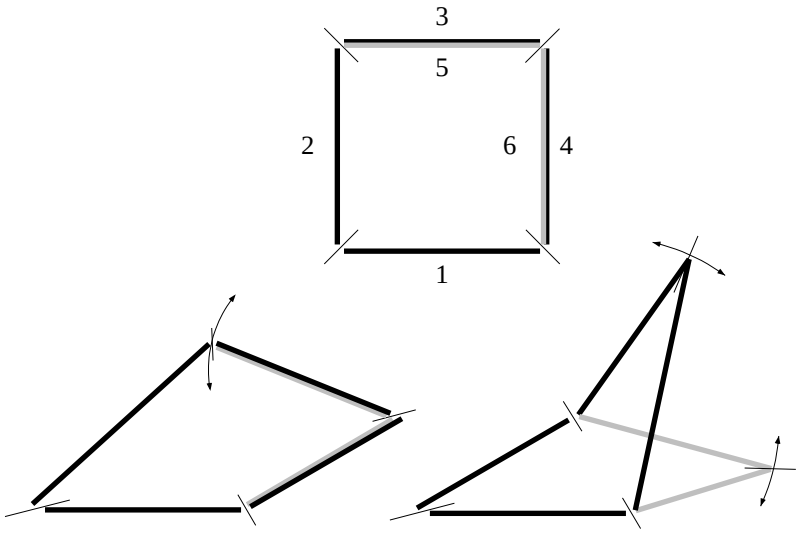
$$\begin{pmatrix} \dfrac{\partial g_{1,1}}{\partial p_{1,1}}(p) & \dots & \dfrac{\partial g_{1,1}}{\partial p_{n,j_n}}(p) \\ \vdots & & \vdots \\ \dfrac{\partial g_{1\delta_1^c}}{\partial p_{1,1}}(p) & \dots & \dfrac{\partial g_{1\delta_1^c}}{\partial p_{n,j_n}}(p) \\ \dfrac{\partial g_{2,1}}{\partial p_{1,1}}(p) & \dots & \dfrac{\partial g_{2,1}}{\partial p_{n,j_n}}(p) \\ \vdots & & \vdots \\ \vdots & & \vdots \\ \dfrac{\partial g_{\nu\delta_\nu^c}}{\partial p_{1,1}}(p) & \dots & \dfrac{\partial g_{\nu\delta_\nu^c}}{\partial p_{n,j_n}}(p) \end{pmatrix}$$

ou, ce qui est équivalent, par le rang de la matrice $\Delta_{\mathcal{V}}(p)$ dite *matrice de contrainte* du mécanisme et définie comme le produit de la matrice précédente par $P(p)^{-1}$ – cette matrice est précisément celle que l'on a déjà notée $\Delta_{\mathcal{V}}(p)$ dans la section (§3.3.1) et notamment dans l'équation (3.3.1.xvii) – voir aussi (§4.1.4).

Le degré de liberté du mécanisme étudié est, bien entendu, cette codimension, c'est à dire la différence entre la dimension de la variété des configurations admissibles du système arborescent sous-jacent et le rang de la matrice de contrainte. Soit alors $\mathrm{rg}(\Delta_{\mathcal{V}})$ ce rang, on a :

$$\delta^l(\mathcal{S}) = \delta^l(\mathcal{S}_{\mathrm{arb}}) - \mathrm{rg}(\Delta_{\mathcal{V}}) = m - \mathrm{rg}(\Delta_{\mathcal{V}})$$

Remarque 4.2 On rappelle que le degré de liberté du mécanisme que l'on vient de calculer est "local", puisque ce calcul n'est valide qu'au voisinage V d'une configuration donnée. Pour certains mécanismes, le choix de configurations différentes pour la définition de V peut amener à trouver des valeurs différentes pour le degré de liberté – voir figure (4.1).



Ce mécanisme, constitué de trois "demi-salières" est représenté en configuration plane et vu de dessus sur le schéma du haut. Les dessins du bas représentent respectivement des configurations au voisinage desquelles son degré de liberté est 1 ou 2.

figure 4.1 : mécanisme avec différents degrés de liberté

4.1.3 détermination d'un paramétrage

On suppose, dans ce paragraphe, que l'on connaît un paramétrage $(p_1, \ldots, p_m)$ de $\mathcal{V}_{\mathrm{arb}}$ et les ν $(= \sum_{i=1}^{n} \delta_i^c)$ fonctions $g_1, \ldots, g_\nu$ telles que :

$$g_1(p_1, \ldots, p_m) = \ldots = g_\nu(p_1, \ldots, p_m) = 0$$

qui caractérisent la sous-variété $\mathcal{V}$ des configurations admissibles du mécanisme étudié ainsi que la matrice de contrainte $\Delta_{\mathcal{V}}(p)$.
On suppose que l'on connaît le rang de la matrice de contrainte – que l'on note r – et le degré de liberté du mécanisme est donc $\delta^l(\mathcal{S}) = m - r$.

Par définition de la notion de sous-variété, il existe, au moins localement, une carte de $\mathcal{V}_{\text{arb}}$:

$$(\mathcal{P}_1, \ldots, \mathcal{P}_{m-r}, \mathcal{P}_{m-r+1}, \ldots, \mathcal{P}_m)$$

telle que pour toute configuration ϕ de $\mathcal{V}_{\text{arb}}$ dans le domaine de cette carte,

$$x \in \mathcal{V} \text{ si et seulement si } \mathcal{P}_{m-r+1} = \ldots = \mathcal{P}_m = 0$$

Si on note ψ le difféomorphisme de changement de carte :

$$\psi \ : \quad (\mathcal{P}_1, \ldots, \mathcal{P}_m) \ \longmapsto \ (p_1, \ldots, p_m)$$

alors, pour n'importe quelles valeurs des paramètres $\mathcal{P}_1, \ldots, \mathcal{P}_{m-r}$ on a :

$$\left\{ \begin{array}{l} g_1(\psi(\mathcal{P}_1, \ldots, \mathcal{P}_{m-r}, 0, \ldots, 0) \ = \ 0 \\ \vdots \\ g_\nu(\psi(\mathcal{P}_1, \ldots, \mathcal{P}_{m-r}, 0, \ldots, 0) \ = \ 0 \end{array} \right.$$

Ainsi, $\mathcal{V}$ est entièrement caractérisée par la donnée de la carte $(\mathcal{P}_1, \ldots, \mathcal{P}_{m-r})$.
De manière pratique, déterminer un paramétrage de $\mathcal{V}$, c'est donc exhiber un jeu de paramètres $(\mathcal{P}_1, \ldots, \mathcal{P}_{m-r})$ ainsi que le difféomorphisme ψ de changement de carte ou tout au moins sa restriction à $\mathcal{V}$:

$$\psi_{\mathcal{V}} \ : \quad (\mathcal{P}_1, \ldots, \mathcal{P}_{m-r}) \ \longmapsto \ (p_1, \ldots, p_m) = \psi(\mathcal{P}_1, \ldots, \mathcal{P}_{m-r}, 0, \ldots, 0)$$

4.1.4 génération des équations

On a vu dans la section (§3.3.1) que, en utilisant un formalisme approprié, on pouvait obtenir les équations du mouvement d'un système mécanique à structure bouclée sous la forme :

$$\left\{ \begin{array}{l} M(p).\dot{q} = S(p,q) + F(p,q) + {}^T\left(\Delta_{\mathcal{V}}(p)\right).\Lambda \\ \Delta_{\mathcal{V}}(p) \ . \ q = 0 \end{array} \right.$$

où p et q sont respectivements des jeux de paramètres de position et cinématiques de la variété $\mathcal{V}_{\text{arb}}$ des configurations admissibles du système mécanique arborescent sous-jacent.

Remarque 4.3 Parmi ces équations, celles dites de contrainte,

$$\Delta_{\mathcal{V}}(p) \ . \ q = 0$$

traduisent simplement le fait qu'un mouvement du mécanisme considéré est bien une application (d'un intervalle de temps) vers sa variété $\mathcal{V}$ des configurations admissibles puisqu'elles sont obtenues par différenciation à partir des fonctions f_i caractérisant $\mathcal{V}$.

Si l'on suppose que l'on a su déterminer une carte $(\mathcal{P}_1, \ldots, \mathcal{P}_{m-r})$ de $\mathcal{V}$ ainsi que le difféomorphisme $\psi_{\mathcal{V}}$ de changement de carte correspondant, on a, par définition :

$$(p_1, \ldots, p_m) = \psi_{\mathcal{V}}(\mathcal{P}_1, \ldots, \mathcal{P}_{m-r}) \qquad \text{et} \qquad \begin{pmatrix} \dot{p}_1 \\ \vdots \\ \dot{p}_m \end{pmatrix} = \left(\frac{\partial \psi_{\mathcal{V}}}{\partial \mathcal{P}}(\mathcal{P}) \right) \begin{pmatrix} \dot{\mathcal{P}}_1 \\ \vdots \\ \dot{\mathcal{P}}_{m-r} \end{pmatrix}$$

en notant $\left(\frac{\partial \psi_{\mathcal{V}}}{\partial \mathcal{P}}(\mathcal{P})\right)$ la matrice jacobienne du changement de carte :

$$\begin{pmatrix} \frac{\partial p_1}{\partial \mathcal{P}_1}(\mathcal{P}_1,\ldots,\mathcal{P}_{m-r}) & \cdots & \frac{\partial p_1}{\partial \mathcal{P}_{m-r}}(\mathcal{P}_1,\ldots,\mathcal{P}_{m-r}) \\ \vdots & & \vdots \\ \frac{\partial p_m}{\partial \mathcal{P}_1}(\mathcal{P}_1,\ldots,\mathcal{P}_{m-r}) & \cdots & \frac{\partial p_m}{\partial \mathcal{P}_{m-r}}(\mathcal{P}_1,\ldots,\mathcal{P}_{m-r}) \end{pmatrix}$$

On a aussi :

$$\begin{pmatrix} q_1 \\ \vdots \\ q_m \end{pmatrix} = P(p) \left(\frac{\partial \psi_{\mathcal{V}}}{\partial \mathcal{P}}(\mathcal{P})\right) \begin{pmatrix} \dot{\mathcal{P}}_1 \\ \vdots \\ \dot{\mathcal{P}}_{m-r} \end{pmatrix}$$

$$\text{ou} \quad \begin{pmatrix} q_1 \\ \vdots \\ q_m \end{pmatrix} = \Psi(\mathcal{P}) \begin{pmatrix} \dot{\mathcal{P}}_1 \\ \vdots \\ \dot{\mathcal{P}}_{m-r} \end{pmatrix} \quad \text{avec} \quad \Psi(\mathcal{P}) = P(\psi_{\mathcal{V}}(\mathcal{P})) \left(\frac{\partial \psi_{\mathcal{V}}}{\partial \mathcal{P}}(\mathcal{P})\right)$$

puis :

$$\begin{pmatrix} \dot{q}_1 \\ \vdots \\ \dot{q}_m \end{pmatrix} = \Psi(\mathcal{P}) \begin{pmatrix} \ddot{\mathcal{P}}_1 \\ \vdots \\ \ddot{\mathcal{P}}_{m-r} \end{pmatrix} + \dot{\Psi}(\mathcal{P}, \dot{\mathcal{P}}) \begin{pmatrix} \dot{\mathcal{P}}_1 \\ \vdots \\ \dot{\mathcal{P}}_{m-r} \end{pmatrix}$$

De plus, sur le domaine de définition de la carte :

$$\begin{cases} (g_1 \circ \psi_{\mathcal{V}})(\mathcal{P}_1,\ldots,\mathcal{P}_{m-r}) = 0 \\ \vdots \\ (g_\nu \circ \psi_{\mathcal{V}})(\mathcal{P}_1,\ldots,\mathcal{P}_{m-r}) = 0 \end{cases}$$

et consécutivement, par dérivation par rapport au temps :

$$\begin{pmatrix} \frac{\partial (g_1 \circ \psi_{\mathcal{V}})}{\partial \mathcal{P}_1}(\mathcal{P}_1,\ldots,\mathcal{P}_{m-r}) & \cdots & \frac{\partial (g_1 \circ \psi_{\mathcal{V}})}{\partial \mathcal{P}_{m-r}}(\mathcal{P}_1,\ldots,\mathcal{P}_{m-r}) \\ \vdots & & \vdots \\ \frac{\partial (g_\nu \circ \psi_{\mathcal{V}})}{\partial \mathcal{P}_1}(\mathcal{P}_1,\ldots,\mathcal{P}_{m-r}) & \cdots & \frac{\partial (g_\nu \circ \psi_{\mathcal{V}})}{\partial \mathcal{P}_{m-r}}(\mathcal{P}_1,\ldots,\mathcal{P}_{m-r}) \end{pmatrix} \begin{pmatrix} \dot{\mathcal{P}}_1 \\ \vdots \\ \dot{\mathcal{P}}_{m-r} \end{pmatrix} = 0$$

c'est à dire :

$$\left(\frac{\partial g}{\partial p}(\psi_{\mathcal{V}}(\mathcal{P}))\right) \left(\frac{\partial \psi_{\mathcal{V}}}{\partial \mathcal{P}}(\mathcal{P})\right) \begin{pmatrix} \dot{\mathcal{P}}_1 \\ \vdots \\ \dot{\mathcal{P}}_{m-r} \end{pmatrix} = 0$$

ce qui peut s'écrire :

$$\left(\frac{\partial g}{\partial p}(\psi_{\mathcal{V}}(\mathcal{P}))\right)\ [P(\psi_{\mathcal{V}}(\mathcal{P}))]^{-1}\ P(\psi_{\mathcal{V}}(\mathcal{P}))\ \left(\frac{\partial \psi_{\mathcal{V}}}{\partial \mathcal{P}}(\mathcal{P})\right)\ \begin{pmatrix} \dot{\mathcal{P}}_1 \\ \vdots \\ \dot{\mathcal{P}}_{m-r} \end{pmatrix} = 0$$

$$\text{i.e.}\quad \Delta_{\mathcal{V}}(\psi_{\mathcal{V}}(\mathcal{P})).\Psi(\mathcal{P})\ \begin{pmatrix} \dot{\mathcal{P}}_1 \\ \vdots \\ \dot{\mathcal{P}}_{m-r} \end{pmatrix} = 0$$

et comme cette égalité doit être vraie indépendamment des valeurs prises par $(\dot{\mathcal{P}}_1, \ldots, \dot{\mathcal{P}}_{m-r})$, on a :

$$\Delta_{\mathcal{V}}(\psi_{\mathcal{V}}(\mathcal{P})).\Psi(\mathcal{P}) = 0 \qquad \text{et aussi} \qquad {}^T(\Psi(\mathcal{P}))\,.^T(\Delta_{\mathcal{V}}(\psi_{\mathcal{V}}(\mathcal{P}))) = 0$$

Alors on peut, par substitution dans les équations de départ, exprimer les équations du mouvement du mécanisme considéré par :

$$\begin{aligned} {}^T(\Psi(\mathcal{P}))M(\psi_{\mathcal{V}}(\mathcal{P})).\left(\Psi(\mathcal{P})\ddot{\mathcal{P}} + \dot{\Psi}(\mathcal{P},\dot{\mathcal{P}})\right) = \\ {}^T(\psi_{\mathcal{V}}(\mathcal{P}))S(\psi_{\mathcal{V}}(\mathcal{P}),\Psi(\mathcal{P})\dot{P}) + {}^T(\psi_{\mathcal{V}}(\mathcal{P}))F(\psi_{\mathcal{V}}(\mathcal{P}),\Psi(\mathcal{P})\dot{P}) \end{aligned}$$

ce qui peut, avec les bonnes notations, se mettre sous la forme :

$$\mathbf{M}(\mathcal{P}).\ddot{\mathcal{P}} = \mathbf{S}(\mathcal{P},\dot{\mathcal{P}}) + \mathbf{F}(\mathcal{P},\dot{\mathcal{P}})$$

4.1.5 pour conclure

On peut revenir un peu sur le problème de la détermination d'un paramétrage de la variété des configurations admissibles.

> En effet, on vient de voir que la connaissance d'une carte $(\mathcal{P}_1, \ldots, \mathcal{P}_{m-r})$ de $\mathcal{V}$, du difféomorphisme de changement de carte $\psi_{\mathcal{V}}$ et de sa matrice jacobienne $\left(\frac{\partial \psi_{\mathcal{V}}}{\partial \mathcal{P}}\right)$ permet d'obtenir une matrice Ψ de taille $(m \times (m - \mathrm{rg}(\Delta_{\mathcal{V}})))$ et de rang $m - \mathrm{rg}(\Delta_{\mathcal{V}})$ telle que $\Delta_{\mathcal{V}}.\Psi = 0$.
>
> Réciproquement, si l'on suppose que l'on a calculé, par des méthodes purement mathématiques, une matrice $\Psi(p)$ de taille $(m \times (m - \mathrm{rg}(\Delta_{\mathcal{V}})))$ de rang $m - \mathrm{rg}(\Delta_{\mathcal{V}})$ et telle que $\Delta_{\mathcal{V}}.\Psi = 0$, alors on peut toujours poser :
>
> $$\begin{pmatrix} q_1 \\ \vdots \\ q_m \end{pmatrix} = \Psi(p) \begin{pmatrix} \dot{\mathcal{P}}_1 \\ \vdots \\ \dot{\mathcal{P}}_{m-r} \end{pmatrix}$$
>
> et espérer récupérer :
>
> - les paramètres $\dot{\mathcal{P}}_i$ en résolvant formellement ce système lorsque c'est possible,

- les paramètres $\mathcal{P}_i$ en intégrant formellement, le cas échéant, les expressions obtenues pour définir les $\dot{\mathcal{P}}_i$.

De toute cette section, on retiendra donc essentiellement les points suivants :

i. La détermination du degré de liberté d'un mécanisme se ramène au calcul du rang d'une matrice dont les coefficients sont des expressions algébriques en les fonctions élémentaires des paramètres du système mécanique à structure arborescente sous-jacent au mécanisme étudié.

ii. Si, en plus du rang de cette matrice $\Delta_{\mathcal{V}}(p)$, on a calculé une matrice $\Psi(p)$ – dont les coefficients sont aussi des expressions algébriques en les fonctions élémentaires des paramètres de ce système – de taille $(m \times (m - \mathrm{rg}(\Delta_{\mathcal{V}})))$ et de rang $m - \mathrm{rg}(\Delta_{\mathcal{V}})$, on pourra peut-être alors déterminer automatiquement un paramétrage de $\mathcal{V}$.

iii. Enfin, si l'on a pu calculer la dimension de la sous-variété des configurations admissibles du mécanisme étudié, et si l'on connaît un paramétrage de cette sous-variété, l'établissement des équations, de manière systématique, par un logiciel spécialisé ne pose pas de problème particulier, à condition de pouvoir effectuer des manipulations formelles sur des expressions et notamment des dérivations.

4.2 quelques exemples de mécanismes bouclés

On va présenter, dans les paragraphes suivants, des méthodes et des outils qui ont été successivement développés pour déterminer le degré de liberté d'un mécanisme à structure bouclée, et, si possible, un paramétrage de sa variété de configurations admissibles.

Mais auparavant, on décrit les exemples sur lesquels ces méthodes ont été expérimentées.

exemple 1 : le mécanisme le plus simple

Ce premier exemple consiste en un mécanisme planaire – et à mouvement dans le plan – constitué de trois tiges $\mathcal{S}_1$, $\mathcal{S}_2$ et $\mathcal{S}_3$ de longueurs inégales (respectivement λ_1, λ_2 et λ_3) reliées entre elles par des articulations rotoïdes. Il est représenté dans la figure (4.2).

figure 4.2 : le mécanisme le plus simple (exemple 1)

Aucune de ces trois tiges n'est supposé fixe par rapport au repère dans lequel on étudie le mouvement.

Si on note respectivement M_{12}^t, M_{13}^t, M_{21}^t, M_{23}^t, M_{32}^t et M_{31}^t les positions des extrémités des trois tiges à tout instant t, on paramètre la variété primitive de configurations $\mathcal{V}'$ de ce mécanisme par

$(\mathfrak{p}_1, \mathfrak{p}_2, \mathfrak{p}_3)$ avec :

$$\mathfrak{p}_1 = \begin{pmatrix} x_{M^t_{13}} \\ y_{M^t_{13}} \\ \alpha_1 \end{pmatrix}, \qquad \mathfrak{p}_2 = \begin{pmatrix} x_{M^t_{21}} \\ y_{M^t_{21}} \\ \alpha_2 \end{pmatrix} \quad \text{et} \quad \mathfrak{p}_3 = \begin{pmatrix} x_{M^t_{32}} \\ y_{M^t_{32}} \\ \alpha_3 \end{pmatrix}$$

où α_1, α_2 et α_3 sont respectivement les mesures, comprises entre 0 et 2π, des angles de demi-droites $((Ox), (M^t_{13}, M^t_{12}))$, $((Ox), (M^t_{21}, M^t_{23}))$ et $((Ox), (M^t_{31}, M^t_{32}))$.

Remarque 4.4 Le domaine de validité de cette carte exclut les positions du système où l'une des tiges forme un angle de mesure nulle avec l'axe Ox.

Les liaisons s'expriment dans ce paramétrage :

$$\left\{ \begin{array}{l} x_{M^t_{13}} + \lambda_1 \cos(\alpha_1) - x_{M^t_{21}} = 0 \\ y_{M^t_{13}} + \lambda_1 \sin(\alpha_1) - y_{M^t_{21}} = 0 \\ x_{M^t_{21}} + \lambda_2 \cos(\alpha_2) - x_{M^t_{32}} = 0 \\ y_{M^t_{21}} + \lambda_2 \sin(\alpha_2) - y_{M^t_{32}} = 0 \\ x_{M^t_{32}} + \lambda_3 \cos(\alpha_3) - x_{M^t_{13}} = 0 \\ y_{M^t_{32}} + \lambda_3 \sin(\alpha_3) - y_{M^t_{13}} = 0 \end{array} \right. \qquad (ex1.\text{i})$$

et la matrice de contrainte $\Delta_{\mathcal{V}}(\mathfrak{p})$ s'écrit :

$$\left(\begin{array}{ccc|ccc|ccc} -1 & 0 & \lambda_1 \sin(\alpha_1) & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & -\lambda_1 \cos(\alpha_1) & 0 & 1 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & -1 & 0 & \lambda_2 \sin(\alpha_2) & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & -\lambda_2 \cos(\alpha_2) & 0 & 1 & 0 \\ \hline 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & \lambda_3 \sin(\alpha_3) \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & -\lambda_3 \cos(\alpha_3) \end{array} \right) \qquad (ex1.\text{ii})$$

Le système mécanique arborescent sous-jacent est obtenu en ne considérant pas la liaison rotoïde entre les tiges $\mathcal{S}_2$ et $\mathcal{S}_3$. Il est paramétré naturellement par (p_1, p_2, p_3) avec :

$$p_1 = \mathfrak{p}_1, \qquad p_2 = \alpha_2, \quad \text{et} \quad p_3 = \alpha_3$$

Dans ce paramétrage, la liaison ℓ_3 entre $\mathcal{S}_2$ et $\mathcal{S}_3$ s'exprime par :

$$\left\{ \begin{array}{l} \lambda_1 \cos(\alpha_1) + \lambda_2 \cos(\alpha_2) + \lambda_3 \cos(\alpha_3) = 0 \\ \lambda_1 \sin(\alpha_1) + \lambda_2 \sin(\alpha_2) + \lambda_3 \sin(\alpha_3) = 0 \end{array} \right. \qquad (ex1.\text{iii})$$

ce qui donne pour la matrice de contrainte :

$$\Delta_{\mathcal{V}}(p) = \left(\begin{array}{ccc|c|c} 0 & 0 & \lambda_1 \sin(\alpha_1) & \lambda_2 \sin(\alpha_2) & \lambda_3 \sin(\alpha_3) \\ 0 & 0 & -\lambda_1 \cos(\alpha_1) & \lambda_2 \cos(\alpha_2) & -\lambda_3 \cos(\alpha_3) \end{array} \right) \qquad (ex1.\text{iv})$$

Il est trivial que ces trois tiges forment un triangle indéformable qui peut se déplacer dans le plan – donc le mécanisme possède un degré de liberté égal à 3 – et qu'une bonne façon de paramétrer ce mécanisme pour écrire les équations de la dynamique serait de ne considérer que la position et l'orientation d'une des trois tiges. On retrouvera ce résultat par le calcul.

exemple 2 : une bielle

Dans cet exemple, on considère encore un mécanisme planaire. Il s'agit d'une bielle – au sens classique – composée d'un bâti supposé fixe $\mathcal{S}_0$ et de trois tiges $\mathcal{S}_1$, $\mathcal{S}_2$ et $\mathcal{S}_3$ de longueurs respectives λ_1, λ_2 et λ_3. Ces corps sont reliés entre eux par trois articulations rotoïdes (entre $\mathcal{S}_0$ et $\mathcal{S}_1$, entre $\mathcal{S}_1$ et $\mathcal{S}_2$ et entre $\mathcal{S}_2$ et $\mathcal{S}_3$) et par une articulation prismatique (entre $\mathcal{S}_3$ et $\mathcal{S}_0$). Il est représenté dans la figure (4.3).

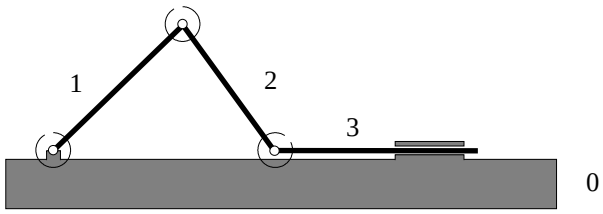


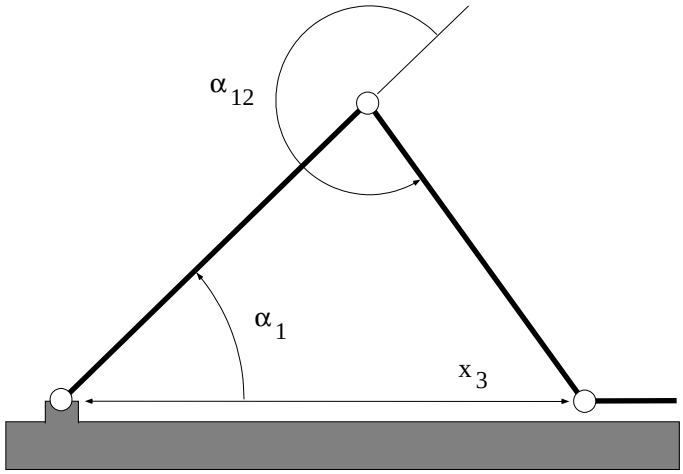
figure 4.3 : la bielle (exemple 2)

On désigne par M^t_{10}, M^t_{12}, M^t_{21}, M^t_{23}, M^t_{31} et M^t_{32} les positions des extrémités des trois tiges, et par M^t_{01} et M^t_{03}, les positions des particules du bâti, aux articulations avec les tiges $\mathcal{S}_1$ et $\mathcal{S}_3$.

Le système mécanique arborescent sous-jacent est obtenu en ne tenant pas compte de la liaison ℓ_3 entre le corps $\mathcal{S}_2$ et le corps $\mathcal{S}_3$. Il est paramétré par $p = (\alpha_1, \alpha_{12}, x_3)$ où :

- α_1 et α_{12} sont les mesures, entre 0 et 2π, des angles de demi-droites $((M^t_{01}M^t_{03})(M^t_{10}M^t_{12}))$ et $((M^t_{10}M^t_{12})(M^t_{21}M^t_{23}))$,
- et x_3 la mesure orientée $\overline{M^t_{01}M^t_{32}}$.

comme indiqué sur le dessin ci-dessous.



On traduit alors l'existence de la liaison ℓ_3 par les équations :

$$\left\{\begin{array}{c} \lambda_1\cos(\alpha_1)+\lambda_2\cos(\alpha_1+\alpha_{12})-x_3=0 \\ \lambda_1\sin(\alpha_1)+\lambda_2\sin(\alpha_1+\alpha_{12})=0 \end{array}\right. \qquad (ex2.\text{i})$$

et par la matrice de contrainte $\Delta_{\mathcal{V}}(p)$:

$$\left(\begin{array}{ccc} -\lambda_1\sin(\alpha_1)-\lambda_2(\sin(\alpha_1)\cos(\alpha_{12})+\cos(\alpha_1)\sin(\alpha_{12})) & -\lambda_2(\sin(\alpha_1)\cos(\alpha_{12})+\cos(\alpha_1)\sin(\alpha_{12})) & -1 \\ \lambda_1\cos(\alpha_1)+\lambda_2(\cos(\alpha_1)\cos(\alpha_{12})-\sin(\alpha_1)\sin(\alpha_{12})) & \lambda_2(\cos(\alpha_1)\cos(\alpha_{12})-\sin(\alpha_1)\sin(\alpha_{12})) & 0 \end{array}\right) \qquad (ex2.\text{ii})$$

Ce mécanisme ayant été conçu pour tranformer un mouvement de rotation en un mouvement de translation alternatif, on sait qu'il possède un degré de liberté égal à 1, et qu'il peut, en dehors de configurations singulières être paramétré par n'importe lequel des trois paramètres α_1, α_{12} ou x_3. On verra comment on peut retrouver ces résultats de manière calculatoire.

exemple 3 : la salière

Il s'agit du mécanisme que l'on a déjà rencontré dans la section (§4.1) constitué de 4 tiges de même longueur reliées entre elles par des articulations rotoïdes.

On définit le système mécanique arborescent sous-jacent en "coupant" la liaison ℓ_3 entre les corps $\mathcal{S}_2$ et $\mathcal{S}_3$ – voir figure (4.4).

La tige $\mathcal{S}_0$ est supposé fixe par rapport au repère dans lequel on étudie le mouvement, elle reste donc dans le plan Oxy. Pour paramétrer le système mécanique arborescent sous-jacent, on utilise $p=(\alpha_1,\alpha_2,\alpha_3)$, c'est à dire respectivement, les mesures, entre $-\pi$ et π, des angles des rotations autour des axes des liaisons de telle sorte que ces angles soient nuls lorsque la salière est en configuration plane déployée et que les angles correspondant à α_2 et à α_3 aient la même orientation.

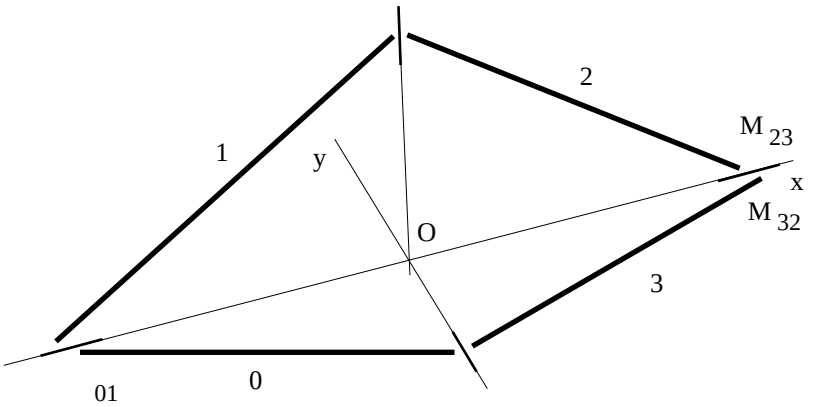
Si, de plus, pour éviter de faire apparaître une constante dans les équations, on suppose que les tiges sont toutes de longueur $\sqrt{2}$. Alors les contraintes dues à la liaison ℓ_3 s'écrivent normalement à partir de l'égalité de $M_{23}^t=M_{32}^t$, et du parallèlisme des deux axes liés à chacun de ces point. Mais puisque ces axes sont respectivement colinéaires à $\overrightarrow{OM_{23}^t}$ et à $\overrightarrow{OM_{32}^t}$, écrire leur parallélisme serait totalement redondant avec l'égalité des points. On a donc :

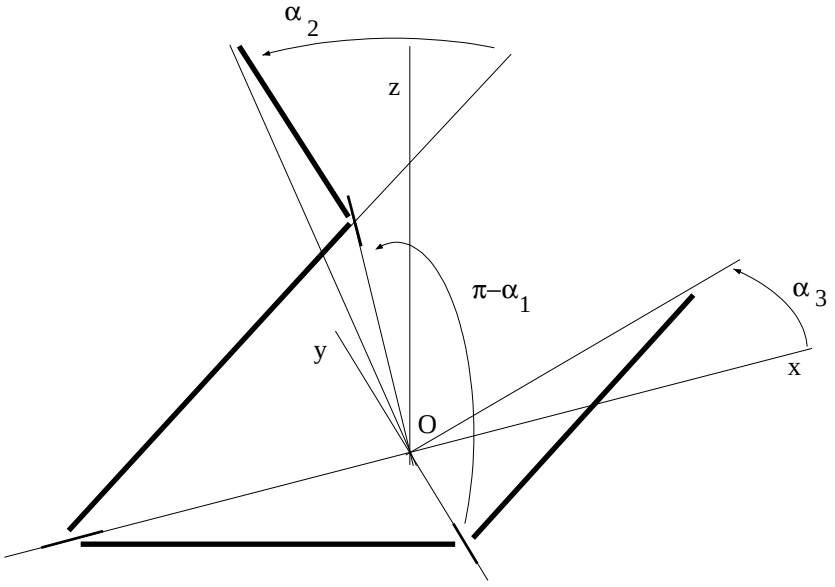
$$\left\{\begin{array}{c} \cos(\alpha_2)-\cos(\alpha_3)=0 \\ \sin(\alpha_1)\sin(\alpha_2)=0 \\ \cos(\alpha_1)\sin(\alpha_2)-\sin(\alpha_3)=0 \end{array}\right. \qquad (ex3.\text{i})$$

puis la matrice de contrainte :

$$\left(\begin{array}{ccc} 0 & -\sin(\alpha_2) & \sin(\alpha_3) \\ \cos(\alpha_1)\sin(\alpha_2) & \sin(\alpha_1)\cos(\alpha_2) & 0 \\ -\sin(\alpha_1)\sin(\alpha_2) & \cos(\alpha_1)\cos(\alpha_2) & -\cos(\alpha_3) \end{array}\right) \qquad (ex3.\text{ii})$$

Sur cet exemple, on verra comment le calcul du rang de sa matrice de contrainte peut permettre d'exhiber les lieux singuliers de la variété de configurations du mécanisme étudié.





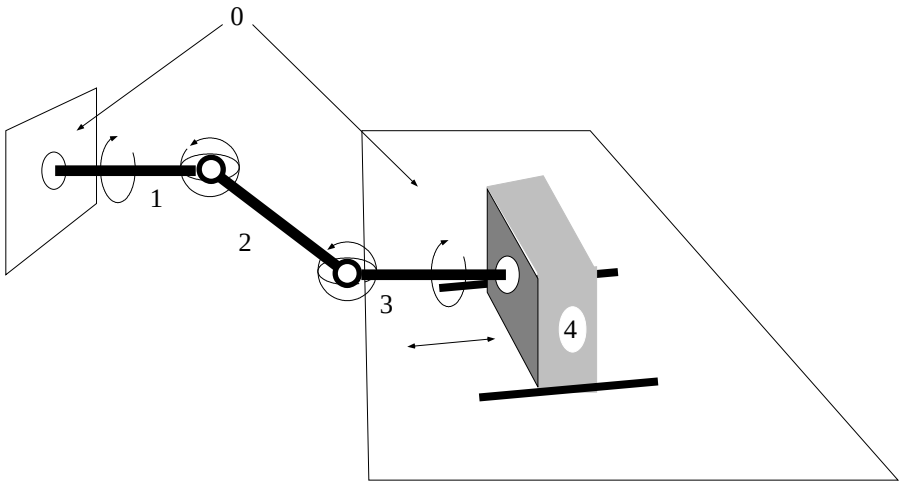
Le dessin du bas représente une configuration du système mécanique arborescent sous-jacent.

figure 4.4 : la salière (exemple 3)

exemple 4 : le joint homocinétique

Le joint homocinétique est un mécanisme spatial dont les applications industrielles sont nombreuses puisqu'il schématise un des procédés de transmission d'un mouvement de rotation entre deux points disposés de manière quelconque – il peut s'agir par exemple d'une transmission boîte de vitesse - essieu. Il est composé d'un bâti $\mathcal{S}_0$ et de quatre corps en mouvement, $\mathcal{S}_1$, $\mathcal{S}_2$, $\mathcal{S}_3$ et $\mathcal{S}_4$.

Les liaisons entre ces différents corps et avec le bâti sont deux liaisons rotoïdes, deux liaisons *universelles** – voir figure (1.17) –, et une liaison prismatique – voir figure (4.5).



Il est important de remarquer que les deux parties du bâti sont solidaires et donc toutes les deux fixes.

figure 4.5 : le joint homocinétique (exemple 4)

Le système mécanique arborescent sous-jacent est obtenu en ne ne tenant pas compte de la liaison universelle ℓ_3 entre les corps $\mathcal{S}_2$ et $\mathcal{S}_3$.

Pour mettre en équation ce système mécanique, on va utiliser les paramètres suivants :

- α_1 une mesure de l'angle de rotation du corps $\mathcal{S}_1$ sur lui-même,
- β_2 et γ_2 des mesures des angles de rotation autour, respectivement, des deux axes du cardan,
- α_3 une mesure de l'angle de rotation du corps $\mathcal{S}_3$ sur lui-même,
- et z_4 une mesure de la translation du corps $\mathcal{S}_4$ par rapport au bâti, correspondant à la coordonnée selon Oz d'une particule du corps $\mathcal{S}_4$ – le repère de référence a été choisi pour que l'axe de la liaison prismatique soit parallèle à Oz.

De plus, on notera λ_2 la longueur de la tige $\mathcal{S}_2$ et (x_4, y_4) les deux autres coordonnées de la particule du corps $\mathcal{S}_4$, ces deux coordonnées étant indépendantes du temps.

Alors la liaison ℓ_3 est traduite par l'égalité de $M_{23}^t = M_{32}^t$, et en écrivant que les deux axes du cardan de la liaison ℓ_3 sont toujours orthogonaux.

*Une liaison *universelle* possède un degré de liberté égal à 2, correspondant à deux rotations possibles. Elle est généralement réalisée par un cardan.

On obtient les équations suivantes :

$$\left\{\begin{array}{c}(\cos(\alpha_1)\sin(\beta_2)\cos(\gamma_2)-\sin(\alpha_1)\sin(\gamma_2))\,l-x=0\\(\sin(\alpha_1)\sin(\beta_2)\cos(\gamma_2)+\cos(\alpha_1)\sin(\gamma_2))\,l-y=0\\\cos(\beta_2)\cos(\gamma_2)l-z=0\\\cos(\alpha_4)\sin(\alpha_1)\cos(\beta_2)-\sin(\alpha_4)\cos(\alpha_1)\cos(\beta_2)\end{array}\right. \qquad (ex4.\text{i})$$

puis la matrice de contrainte, qui s'écrit :

$$\left(\begin{array}{c|c}\text{Bloc1} & \text{Bloc2}\end{array}\right) \qquad \text{avec :} \qquad (ex4.\text{ii})$$

$$\text{Bloc1} = \left(\begin{array}{cc}(-\sin(\alpha_1)\sin(\beta_2)\cos(\gamma_2)-\cos(\alpha_1)\sin(\gamma_2))l & \cos(\alpha_1)\cos(\beta_2)\cos(\gamma_2)l\\(\cos(\alpha_1)\sin(\beta_2)\cos(\gamma_2)-\sin(\alpha_1)\sin(\gamma_2))l & \sin(\alpha_1)\cos(\beta_2)\cos(\gamma_2)l\\0 & -\sin(\beta_2)\cos(\gamma_2)l\\(\cos(\alpha_1)\cos(\alpha_4)+\sin(\alpha_1)\sin(\alpha_4))\cos(\beta_2) & (-\sin(\alpha_1)\cos(\alpha_4)+\cos(\alpha_1)\sin(\alpha_4))\sin(\beta_2)\end{array}\right|$$

$$\text{Bloc2} = \left|\begin{array}{ccc}(-\cos(\alpha_1)\sin(\beta_2)\sin(\gamma_2)-\sin(\alpha_1)\cos(\gamma_2))l & 0 & 0\\(-\sin(\alpha_1)\sin(\beta_2)\sin(\gamma_2)+\cos(\alpha_1)\cos(\gamma_2))l & 0 & 0\\-\cos(\beta_2)\sin(\gamma_2)l & 0 & -1\\0 & (-\cos(\alpha_1)\cos(\alpha_4)-\sin(\alpha_1)\sin(\alpha_4))\cos(\beta_2) & 0\end{array}\right)$$

On retrouvera par le calcul, le fait que la liaison prismatique n'ajoute rien ici au degré de liberté du système. En effet, pour avoir, en plus de la liberté de rotation des tiges sur elles-mêmes, la possibilité de bouger le corps $\mathcal{S}_4$, il aurait fallu avoir une liaison planaire à la place de la liaison prismatique – ce qui est le cas dans les joints homocinétiques réellement employés.

exemples 5 et 6 : une maille d'antenne déployable

Ces deux exemples sont des modèles simplifiés et partiels d'une antenne déployable déstinée à être emportée dans l'espace par un satellite. Cette antenne est constituée d'un grand nombre de mailles identiques composées chacune de plusieurs tiges articulées entre elles par des axes de rotations.

Parmi les tiges d'une maille, trois d'entre elles sont des tiges principales et sont des éléments actifs de l'antenne. Elles sont reliés deux à deux par :

- un bras articulé constitué de deux tiges secondaires, dans l'exemple 5 – voir figures (4.6),
- deux bras articulés constitués chacun de deux tiges secondaires, dans l'exemple 6 – voir figures (4.7).

Toutes les tiges principales sont de même longueur ainsi que les tiges secondaires. Les axes de rotations sont placés de sorte que la maille soit toujours, par dessus, vue comme un triangle équilatéral dont les sommets sont les tiges principales :

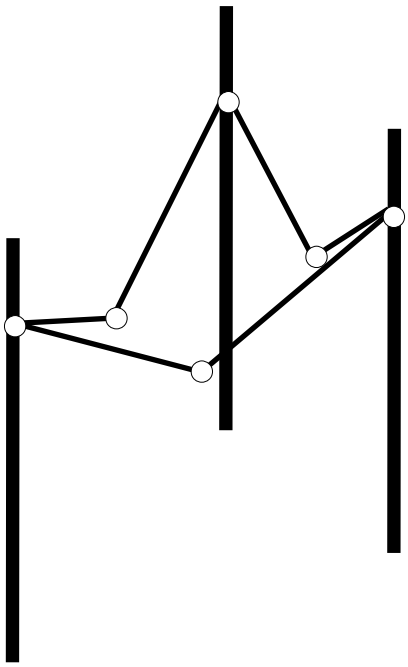


figure 4.6 : maille d'antenne déployable à un niveau (exemple 5)

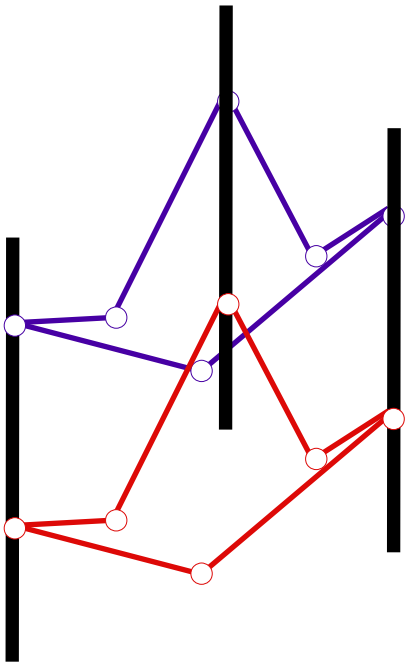
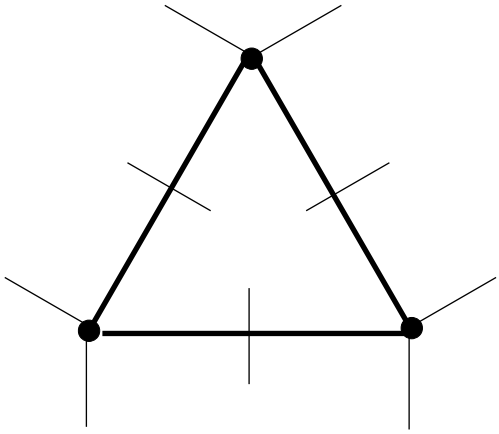


figure 4.7 : maille d'antenne déployable à deux niveaux (exemple 6)



Pour obtenir les équations de contrainte, et les matrices de contrainte des systèmes mécaniques arborescent sous-jacents à ces deux modèles de maille d'antenne, on a utilisé le logiciel *GEMMES* déjà présenté – les sessions correspondantes sont reproduites en annexe (§C.1) ainsi que les expressions obtenues.

maille à un niveau

Concernant la maille de l'exemple 5, le système mécanique initial (bouclé) possède neuf liaisons et une seule boucle. Une de ces liaisons est donc coupée par *GEMMES* pour avoir le système arborescent sous-jacent. On considère qu'une des tiges principales est fixe dans le repère par rapport auquel on étudie le mouvement.

Les configurations du système mécanique arborescent sous-jacent sont ainsi décrites par huit paramètres, soit huit mesures d'angles correspondant chacune à un des axes de rotation. Pour simplifier les expressions obtenues la longueur des tiges est supposée égale à 1.

La matrice de contrainte traduit le fait que deux points du systèmes ont leur vitesse relative de translation nulle (soit trois égalités) et leur vitesse relative de rotation parallèle à un axe donné (soit trois autres égalités). Il en résulte donc six égalités portant sur huit paramètres.

Cette matrice posséde donc six lignes et huit colonnes et fait intervenir huit variables.

Pour avoir une idée de la forme et de la complexité des coefficients de la matrice de contrainte,

voici celui de la cinquième ligne et de la cinquième colonne :

$$\frac{1}{8}\sqrt{3}(2\sin(\theta_{76})\cos(\theta_{80})\cos(\theta_{87})-2\sin(\theta_{76})\sin(\theta_{80})\sin(\theta_{87})+2\cos(\theta_{76})\cos(\theta_{80})\sin(\theta_{87})$$
$$+2\cos(\theta_{76})\sin(\theta_{80})\cos(\theta_{87})-3\sin(\theta_{32})\cos(\theta_{10})\cos(\theta_{21})\sin(\theta_{76})\sin(\theta_{80})\cos(\theta_{87})$$
$$-3\sin(\theta_{32})\cos(\theta_{10})\cos(\theta_{21})\sin(\theta_{76})\cos(\theta_{80})\sin(\theta_{87})-3\sin(\theta_{32})\cos(\theta_{10})\cos(\theta_{21})\cos(\theta_{76})\sin(\theta_{80})\sin(\theta_{87})$$
$$+3\sin(\theta_{32})\cos(\theta_{10})\cos(\theta_{21})\cos(\theta_{76})\cos(\theta_{80})\cos(\theta_{87})-\sin(\theta_{32})\cos(\theta_{10})\cos(\theta_{21})$$
$$+3\sin(\theta_{32})\sin(\theta_{10})\sin(\theta_{21})\sin(\theta_{76})\sin(\theta_{80})\cos(\theta_{87})+3\sin(\theta_{32})\sin(\theta_{10})\sin(\theta_{21})\sin(\theta_{76})\cos(\theta_{80})\sin(\theta_{87})$$
$$+3\sin(\theta_{32})\sin(\theta_{10})\sin(\theta_{21})\cos(\theta_{76})\sin(\theta_{80})\sin(\theta_{87})-3\sin(\theta_{32})\sin(\theta_{10})\sin(\theta_{21})\cos(\theta_{76})\cos(\theta_{80})\cos(\theta_{87})$$
$$+\sin(\theta_{32})\sin(\theta_{10})\sin(\theta_{21})-3\cos(\theta_{32})\cos(\theta_{10})\sin(\theta_{21})\sin(\theta_{76})\sin(\theta_{80})\cos(\theta_{87})$$
$$-3\cos(\theta_{32})\cos(\theta_{10})\sin(\theta_{21})\sin(\theta_{76})\cos(\theta_{80})\sin(\theta_{87})-3\cos(\theta_{32})\cos(\theta_{10})\sin(\theta_{21})\cos(\theta_{76})\sin(\theta_{80})\sin(\theta_{87})$$
$$+3\cos(\theta_{32})\cos(\theta_{10})\sin(\theta_{21})\cos(\theta_{76})\cos(\theta_{80})\cos(\theta_{87})-\cos(\theta_{32})\cos(\theta_{10})\sin(\theta_{21})$$
$$-3\cos(\theta_{32})\sin(\theta_{10})\cos(\theta_{21})\sin(\theta_{76})\sin(\theta_{80})\cos(\theta_{87})-3\cos(\theta_{32})\sin(\theta_{10})\cos(\theta_{21})\sin(\theta_{76})\cos(\theta_{80})\sin(\theta_{87})$$
$$-3\cos(\theta_{32})\sin(\theta_{10})\cos(\theta_{21})\cos(\theta_{76})\sin(\theta_{80})\sin(\theta_{87})+3\cos(\theta_{32})\sin(\theta_{10})\cos(\theta_{21})\cos(\theta_{76})\cos(\theta_{80})\cos(\theta_{87})$$
$$-\cos(\theta_{32})\sin(\theta_{10})\cos(\theta_{21}))$$

Une étude mécanique “à la main” de ce mécanisme, validée par des expérimentations a montré que son degré de liberté est égal à trois. En effet,

- chaque tige principale – mise à part celle supposée fixe – peut se déplacer verticalement,
- la maille peut se déployer – la longueur des côtés du triangle équilatéral varie.

maille à deux niveaux

Comme on l'a vu, la maille de l'antenne déployable à deux niveaux est construite à partir de l'antenne précédente en rajoutant dans chaque maille un nouveau jeu de trois bras articulés reliant les tiges principales deux à deux pour améliorer la rigidité de l'ensemble.

La matrice de contrainte, telle qu'elle est fournie par le logiciel *GEMMES* est une matrice possédant vingt-quatre lignes et quatorze colonnes et faisant intervenir quatorze variables.

En effet, le système mécanique possède initialement 18 liaisons. Quatres de ces liaisons sont coupés par *GEMMES* pour obtenir le système arborescent sous-jacent dont les configurations sont paramétrées par les quatorze mesures angulaires correspondant chacune à un des axes de rotation.

La matrice de contrainte traduit le même phénomène que dans l'exemple précédent mais pour quatre boucles différentes, ce qui donne donc vingt-quatre égalités portant sur quatorze paramètres.

Pour avoir une idée de la forme et de la compléxité des coefficients de la matrice de contrainte, il suffit de savoir que leur forme est sensiblement égale à celle des coefficients de l'exemple précédent dans la mesure où chaque branche de l'arborescence du système est de la même longueur que dans ce cas. On retrouve d'ailleurs la matrice précédente comme sous-matrice de celle-ci.

De plus sept lignes de cette matrice sont nulles car certaines égalités sont trivialement vérifiées en raison de la redondance des contraintes.

En effet, le deuxième niveau de barres articulées que l'on a rajouté n'intervient qu'en ce qui concerne la rigidité du mécanisme, et le degré de liberté de celui-ci reste égale à trois.

4.3 quelques considérations mathématiques

On a montré, dans la section (§4.1) que le calcul du degré de liberté d'un mécanisme à structure bouclé se ramène au problème purement mathématique suivant :

> Etant donné un jeu de variables $p = (p_1, \ldots, p_m)$ reliées par ν équations :
>
> $$g_1(p) = \ldots = g_\nu(p) = 0$$
>
> et étant donnée une matrice $\Delta_\mathcal{V}$ dont les coefficients dépendent explicitement de ces variables, quel est le rang de cette matrice ?

Pour un ensemble de valeurs données des variables p, les coefficients de $\Delta_\mathcal{V}$ sont des nombres réels et le rang de $\Delta_\mathcal{V}$ est défini comme le nombre de ses lignes (ou de ses colonnes) qui sont indépendantes – en tant que vecteurs de $\mathbb{R}^m$ (ou de $\mathbb{R}^\nu$) – ou, de manière équivalente comme l'ordre du plus grand mineur extrait non nul.

Cette définition s'étend facilement au voisinage d'un point donné non singulier dans la mesure où tous les mineurs, en tant que fonctions numériques des variables p sont continus et que si l'image d'un point par une fonction continue est non nulle, il existe un voisinage de ce point sur lequelle la fonction ne s'annule pas.

C'est pourquoi une approche numérique de la détermination du rang ne présente pas de difficultés particulières.

Toutefois, dans l'optique d'une génération automatique des équations de la dynamique sous forme symbolique, on est contraint de conserver les coefficients de la matrice $\Delta_\mathcal{V}$ sous forme symbolique pour effectuer le calcul du rang – même si l'on assume par ailleurs des restrictions quant au domaine de validité de ce calcul pour la détermination du degré de liberté du mécanisme étudié.

Il est alors important de bien préciser le domaine auquel appartiennent ces coefficients qui sont des expressions formelles pour pouvoir définir, sans ambiguïté, la notion de *nullité* d'une expression, puis d'*indépendance* de deux familles d'expressions et enfin de rang.

> En effet, si l'on ne fait aucune hypothèse sur les expressions qui forment les coefficients de $\Delta_\mathcal{V}$, on ne dispose d'aucun renseignement sur l'ensemble des valeurs des paramètres p qui peuvent les annuler. Or pour calculer le rang, on est amené, soit à effectuer des divisions – dans les méthodes basées sur le pivot de Gauss – soit à discuter de la nullité d'expressions – dans les méthodes basées sur le calcul des mineurs. Et le rang – que l'on qualifiera de *générique* – que l'on obtient en se plaçant délibérément en dehors de l'ensemble des zéros des expressions qui apparaissent dans les calculs, ne correspond pas nécessairement au rang que l'on obtiendrait en évaluant, au préalable, la matrice de contrainte puisqu'il est fort possible que l'ensemble des valeurs de p telles que $g_1(p) = \ldots = g_\nu(p) = 0$ soit inclus dans cet ensemble de zéros.

Pour développer des méthodes de calcul de rang, on va, dans le cadre de deux approches différentes, assumer l'une ou l'autre des hypothèses suivantes :

i. on suppose que les coefficients de $\Delta_\mathcal{V}$ sont des fractions rationnelles sur le corps des rationnels $\mathbb{Q}$ en des indéterminées :

$$(X_1, \ldots, X_l, T_{l+1}, \ldots, T_m)$$

ces indéterminées étant concrètement – quitte à renuméroter les paramètres p_i :

- pour les l premiers paramètres p_i, les paramètres eux-mêmes :

$$X_i = p_i$$

- pour les autres, les demi-tangentes de ces paramètres :

$$T_{l+i} = \tan(\frac{p_{l+i}}{2})$$

ii. on suppose que les coefficients de $\Delta_{\mathcal{V}}$ sont des polynômes sur le corps des rationnels $\mathbb{Q}$ en des indéterminées :

$$(X_1, \ldots, X_l, Y_{l+1}, \ldots, Y_m, Z_{l+1}, \ldots, Z_m)$$

ces indéterminées étant concrètement – quitte à renuméroter les paramètres p_i :

- pour les l premiers paramètres p_i, les paramètres eux-mêmes :

$$X_i = p_i$$

- pour les autres, les lignes trigonométriques de ces paramètres :

$$\begin{cases} Y_{l+i} = \sin(p_{l+i}) \\ Z_{l+i} = \cos(p_{l+i}) \end{cases}$$

Remarque 4.5 L'hypothèse (i) réduit le domaine de validité du paramétrage p puisque $\tan(\frac{p_i}{2})$ n'est définie que sur un intervalle de longueur π où p_i est différent d'un multiple impair de π.

Ces hypothèses, qui sont les plus faibles pour pouvoir effectuer un calcul de rang en conservant la matrice $\Delta_{\mathcal{V}}$ sous forme symbolique sont notamment vérifiées si les liaisons qui interviennent dans l'architecture du mécanisme étudié sont telles que leurs degrés de libertés en rotation et en translation sont indépendants (liaisons rotoïdes, prismatiques, planaires, parallèles, universelles, rotules, ...). Elles ne permettent malheureusement pas de considérer des mécanismes dans lesquels certains corps sont soumis à des liaisons hélicoïdales (de type vis-ecrou).

4.3.1 première approche

Dans cette première approche, on se place dans les conditions de la première hypothèse. On considère donc $\Delta_{\mathcal{V}}$ comme une matrice à coefficients dans le corps

$$\mathbb{Q}(X_1, \ldots, X_l, T_{l+1}, \ldots, T_m)$$

Comme c'est le cas pour une matrice à coefficients réels, on définit le rang d'une matrice à coefficients dans un corps comme l'ordre du plus grand déterminant non nul extrait de cette matrice.

La méthode la plus naturelle pour calculer le rang consiste alors à calculer tous les mineurs par ordre *décroissant* jusqu'à ce que l'on trouve un mineur non nul. En fait, comme pour calculer un mineur d'un certain ordre, il est nécessaire de connaître les mineurs d'ordre immédiatement inférieur, on préfère donc calculer tous les mineurs par ordre *croissant* jusqu'à ce que tous les mineurs d'un même ordre soient nuls.

Mais ces deux méthodes requierent un nombre d'opérations qui croit plus qu'exponentiellement avec la taille de la matrice et ne sont donc presque jamais utilisées.

l'algorithme classique

La méthode classiquement mise en œuvre pour calculer le rang d'une matrice à coefficients dans un corps est la méthode dite du *pivot de Gauss* – ou d'*élimination de Gauss* – qui consiste à

triangulariser la matrice en faisant des opérations sur ses lignes et à compter le nombre de lignes non nulles de la matrice ainsi triangularisée.

Cette méthode requiert un nombre de multiplications qui croît avec le cube de la taille de la matrice. Dans le cas où les coefficients sont des fractions rationnelles, il faut, après chaque multiplication, simplifier le produit pour limiter la taille des expressions. C'est pourquoi les calculs deviennent rapidement impraticables car ils nécessitent le traitement d'un grand nombre d'expressions de grandes tailles, comme le montre l'exemple simple ci-dessous :

$$\begin{pmatrix} \frac{x+y}{xy} & \frac{2x+y}{1+y^2} & \frac{y}{x-y} \\ \frac{x-y}{x^2} & \frac{2x}{x^2-y} & \frac{x}{x-y^2} \\ \frac{x^2-y^2}{xy} & \frac{x^2+1}{x+y} & \frac{y}{x} \end{pmatrix}$$

se transforme en la matrice triangulaire :

$$\begin{pmatrix} \frac{x-y}{x^2} & \frac{2x}{x^2-y} & \frac{x}{x-y^2} \\ 0 & \frac{yx^4-3y^2x^2+x^2y-y^2-2x^4-4x^3y}{(x+y)y(x^2-y)} & \frac{-y^4-xy^2+x^4+x^3y}{xy(x-y^2)} \\ 0 & 0 & \cdots \end{pmatrix}$$

où "$\cdots$" représente :

$$\frac{yx^4+x^3y^2-y^3x^2+y^8+3x^5y^3-3x^3y^4+2x^3y^3-3x^5y^2-2x^4y^3+2x^5y^5-7x^3y^6+5x^3y^5-xy^6-3x^5y^4-4x^4y^5}{x(1+y^2)(x-y)(yx^4-3y^2x^2+x^2y-y^2-2x^4-4x^3y)(x-y^2)}$$
$$-\frac{y^6x^2-5y^5x^2+14y^4x^4-3y^7x^2+4y^6x^4+3y^7x^3-3x^7y+x^5y-x^6y-x^7y^3-x^7y^2-4x^6y^3-5x^6y^2}{x(1+y^2)(x-y)(yx^4-3y^2x^2+x^2y-y^2-2x^4-4x^3y)(x-y^2)}+\cdots$$

Ainsi la fonction `gausselim` qui implémente cet algorithme dans la librairie d'algèbre linéaire de MAPLE ne permet-elle pas – pour des raisons de temps et surtout de volume de calcul – de calculer le rang de grosses matrices et notamment pas de la matrice de contrainte (6×8) qui provient de l'etude mécanique de la maille d'antenne à un niveau.

De plus, pour tenir compte du fait que les variables p sont reliées par les équations

$$g_1(p) = \ldots = g_\nu(p) = 0$$

il faut considérer que cet algorithme ne fournit qu'une majoration du rang.

Si l'on veut avoir exactement le rang, il faut tout d'abord exprimer les équations

$$g_1(p) = \ldots = g_\nu(p) = 0$$

dans le corps $\mathcal{Q}(X_1, \ldots, X_l, T_{l+1}, \ldots, T_m)$, ce qui définit ν fractions rationnelles $G_1, \ldots, G_\nu$ telles que :

$$G_1(p) = \ldots = G_\nu(p) = 0$$

puis vérifier, à chaque étape de l'algorithme, que l'on n'effectue pas de division par une expression nulle, i.e par une expression dont le dénominateur peut s'écrire comme une combinaison linéaire à coefficients polynomiaux des dénominateurs des fractions rationnelles G_i.

Toutefois, étant données les faibles capacités de traitement de l'algorithme d'élimination de Gauss, en raison de sa complexité et de la taille des matrices que l'on est amené à considérer, on n'a pas cru bon d'implémenter ces vérifications, ce qui nécessiterait de faire appel à la théorie des bases standards d'idéaux de polynômes – voir (§4.4.1) – et grèverait encore lourdement la complexité de calcul.

Du reste, sur les exemples que l'on a pu traiter avec `gausselim`, il a été facile d'interpréter "cas par cas" les résultats obtenus – voir la section (§4.5) où sont réalisées les expérimentations.

une méthode probabiliste

Une méthode probabiliste pour effectuer un calcul consiste à effectuer un calcul plus simple qui en dérive et tel que la probabilité que les résultats soient les mêmes soit très voisine de 1.

Pour calculer le rang d'une matrice, on va choisir arbitrairement des valeurs entières pour chacune des variables présentes dans les coefficients de la matrice.

Ceci revient à associer à une matrice notée M, dont les coefficients appartiennent à

$$\mathbb{Q}(X_1, \ldots, X_l, T_{l+1}, \ldots, T_m)$$

une matrice à coefficients rationnels, que l'on notera $M_{\mathbb{Q}}$.

Pour être sûr que la fonction d'évaluation rationnelle $f_{\mathbb{Q}}$ définie par :

$$\begin{array}{rcccl} f_{\mathbb{Q}} & : & \mathbb{Q}(X_1, \ldots, X_l, T_{l+1}, \ldots, T_m) & \rightarrow & \mathbb{Q} \\ & & X_i \text{ (pour } i = 1, \ldots, l \text{)} & \mapsto & x_i \text{ (pour } i = 1, \ldots, l \text{)} \end{array}$$

associe bien à M une matrice $M_{\mathbb{Q}}$, il suffit de vérifier que le m-uplet $(x_1, \ldots, x_m)$ n'est pas racine d'un des dénominateurs des coefficients de M.

Il est clair que si un mineur de $M_{\mathbb{Q}}$ n'est pas nul, le mineur correspondant de M ne l'est pas non plus, aussi le rang de M est certainement supérieur ou égal au rang de $M_{\mathbb{Q}}$.

Réciproquement, si un mineur de $M_{\mathbb{Q}}$ est nul, le mineur correspondant de M l'est aussi, sauf si le m-uplet $(x_1, \ldots, x_m)$ est racine d'un des polynômes intervenant pendant le calcul de ce mineur.

Si l'on nomme Ω l'ensemble de ces polynômes, la probabilité ϱ de rencontrer ce cas défavorable est donnée par le rapport entre le nombre de m-uplets $(x_1, \ldots, x_m)$ que l'on peut choisir et le nombre de m-uplets $(x_1, \ldots, x_m)$ qui sont racines de polynômes de Ω.

Si l'on utilise un processus de choix aléatoire des x_i dans un intervalle de valeurs entières comprises entre 1 et une valeur maximale s, on démontre aisément que ϱ tend vers 0 lorsque s tend vers l'infini.

En fait, même pour des valeurs de s assez petites, la probabilité ϱ est minime, et les résultats sont assez probants, comme on va le voir en reprenant l'exemple étudié plus haut.

Il y a 18 polynômes qui interviennent dans le calcul des mineurs :

- 8 à l'ordre 1 :

$$x + y,\ 2x + y,\ x,\ y,\ 2x,\ x - y,\ x^2 - y^2,\ x^2 + 1$$

- 9 à l'ordre 2 :

$$2yx^2 + 2y^2x - 2y^3x - 2y^4 - x^5 + yx^3 - x^3 + xy,$$
$$x\left(2x^4 - 2yx^2 - yx^3 + y^2x - x^2y^2 + 3y^3 - 2xy - 2y^3x + 2y^5\right),$$
$$x^2y^2 - y^4x - y^3x + y^5 - x^5 + x^3y^2,$$
$$x^5y + x^3y^2 + yx^3 - y^2x - x^4y^2 + 3y^3x^2 - x^2y^2 + y^3 - 2x^5 - 2yx^4,$$
$$-y\left(-x^3 - yx^2 + 3y^2x + y^3 + x^3y^2 + x\right),$$
$$x - x^2 - xy + y,$$
$$x^2 + x^2y^2 + 1 + y^2 - 2x^3 - yx^2 + 2y^2x + y^3,$$
$$x^3 + yx^2 - y^2x + y^4,$$
$$2x^3 + 3x^3y^2 + 2yx^2 + 3y^3x^2 - 2yx^4 + 2x^2y^2 - y^3x - y^4$$

- et 1 à l'ordre 3 :

$$\begin{aligned}x^3y^2+y^8-y^3x^2+y^5x-3x^5y^2+2y^3x^3-5y^5x^2-y^6x+x^5y-5x^6y^2+3x^5y^3\\+yx^4-2y^3x^4-3y^4x^3+5x^4y^2+3y^7x+5y^5x^3-y^6x^2-x^6y-x^7y^2-4x^6y^3\\-3x^5y^4+14y^4x^4-x^6+2x^9-4x^4y^5-7x^3y^6-3x^7y-x^7y^3+3x^3y^7+2x^5y^5\\+4x^4y^6+x^6y^4-3y^7x^2-x^8y^2+3x^8y-x^5y^6-x^8\end{aligned}$$

Si on prend comme valeur maximale $s = 9$ pour x et pour y, il y a quatre-vingt-un couples possibles, dont treize annulent des dénominateurs, soit en fait soixante-huit couples possibles, sur ces soixante huit couples, seuls huit annulent certains polynômes de Ω et aucun n'annule le mineur d'ordre 3 et donc ne change la valeur trouvée pour le rang.

Si on prend comme valeur maximale $s = 99$ pour x et pour y, il y a neuf-mille-huit-cent-un couples possibles, dont cent-quinze annulent des dénominateurs, soit en fait neuf-mille-six-cent-quatre-vingt-six couples possibles, sur ces neuf-mille-six-cent-quatre-vingt-six couples, seuls quatre-vingt-dix-huit annulent certains polynômes de Ω et aucun n'annule le mineur d'ordre 3 et donc ne change la valeur trouvée pour le rang.

En fait, dans cet exemple le nombre de cas défavorables est de l'ordre de $\frac{1}{s^2}$ par ce que l'un des polynômes peut s'écrire $(x-1)(x-y)$ ce qu'il est raisonnable de considérer comme de la malchance. Et néanmoins, la méthode donne le rang exact!

Bien sûr, le gain de temps apporté par la méthode probabiliste est énorme, cependant, les calculs des mineurs font intervenir des entiers très grands, ce qui peut devenir génant lorsque le degré des polynômes à évaluer est élevé – dans notre exemple, le déterminant d'ordre 3 est un entier supérieur à 10^{20} pour certaines valeurs de (x, y).

C'est pourquoi l'on va compléter la méthode probabiliste par une méthode modulaire.

Remarque 4.6 Il faut cependant remarquer que cette méthode, et a fortiori les méthodes modulaire et probabiliste-modulaire que l'on va voir ci-dessous, fourniront le rang d'une matrice de contrainte $\Delta_{\mathcal{V}}(p)$ sans tenir compte des équations $g_1(p) = \ldots = g_\nu(p) = 0$.

> On pourrait, bien sûr, contraindre les nombres des m-uplets $(x_1, \ldots, x_m)$ à satisfaire les équations $f_{\mathbb{Q}}(G_1) = \ldots = f_{\mathbb{Q}}(G_\nu) = 0$ – où, pour $i = 1, \ldots, \nu$, G_i est la fraction rationnelle associée à g_i – tout au moins en donnant à certains x_i des valeurs non pas entières mais rationnelles. Mais, dans ce cas, il serait très difficile d'avoir des arguments probabilistes satisfaisants concernant l'égalité des rangs de $\Delta_{\mathcal{V}}$ et de $(\Delta_{\mathcal{V}})_{\mathbb{Q}}$.

On utilisera donc seulement ces méthodes pour obtenir une majoration du rang d'une matrice de contrainte, ce qui s'avèrera utile si des arguments mécaniques permettent, par ailleurs, d'en obtenir une minoration.

une méthode modulaire

Une méthode modulaire pour effectuer un calcul sur des nombres entiers consiste à effectuer ce même calcul modulo un ou plusieurs nombres premiers de telle sorte que l'on puisse déduire le résultat sur les nombres entiers du résultat des calculs modulaires.

Pour évaluer un polynôme à coefficients entiers $P(X_1, \ldots, X_l, T_{l+1}, \ldots, T_m)$ en un point entier $(x_1, \ldots, x_m)$, on transforme le polynôme et le point en $\breve{P}(X_1, \ldots, X_l, T_{l+1}, \ldots, T_m)$ et $(\breve{x}_1, \ldots, \breve{x}_l)$

par l'application de passage au quotient :

$$\begin{array}{rccl} f_{\mathcal{Z}/a\mathcal{Z}} : & \mathcal{Z} & \rightarrow & \mathcal{Z}/a\mathcal{Z} \\ & x & \mapsto & f_{\mathcal{Z}/a\mathcal{Z}}(x) = \breve{x} \text{ tel que } x \equiv \breve{x} \pmod{a} \end{array}$$

où $\mathcal{Z}$ désigne l'ensemble des entiers relatifs et a est un nombre premier.

On effectue ensuite le calcul dans $\mathcal{Z}/a\mathcal{Z}$ et on récupère le résultat dans $\mathcal{Z}$ en utilisant le fait que $\mathcal{Z}/a\mathcal{Z}$ est isomorphe à $\{x \in \mathcal{Z},\ x < a\}$. Ainsi, pour être sûr de récupérer le résultat exact, il faut avoir choisi a plus grand que tous les nombres qui interviennent au cours du calcul.

L'inconvénient de choisir une grande valeur de a se contourne aisément grâce au théorème du reste chinois qui assure que $\mathcal{Z}/a_1\mathcal{Z} \times \mathcal{Z}/a_2\mathcal{Z}$ est isomorphe à $\mathcal{Z}/a_1a_2\mathcal{Z}$ dès que a_1 et a_2 sont premiers entre eux. Il faudra donc effectuer le calcul modulo différents nombres premiers en s'assurant que le produit de ces nombres est plus grand que tous les nombres qui interviennent au cours du calcul.

Pour calculer le rang d'une matrice à coefficients rationnels M de taille n, il faut d'abord la convertir en une matrice à coefficients entiers $M_{\mathcal{Z}}$ en multipliant tous ses coefficients par le plus petit commun multiple de leurs dénominateurs.

On peut alors borner l'ensemble des nombres qui apparaissent au cours du calcul des mineurs de $M_{\mathcal{Z}}$, par définition même des déterminants, par :

$$n!\ (max_{i,j}\ \{M_{\mathcal{Z}}\,[i,j]\})^n$$

Il s'agit donc de choisir des nombres premiers $a_1, \ldots, a_r$ en quantité suffisante pour que $\prod_{i=1}^{r} a_i$ soit supérieur à cette borne.

On associe ensuite à M, r matrices $\breve{M}_1, \ldots, \breve{M}_r$ à coefficients dans les $\mathcal{Z}/a_i\mathcal{Z}$ telles que $\breve{M}_i = g_{\mathcal{Z}/a_i\mathcal{Z}}(M)$, on calcule les divers mineurs de ces matrices, et on obtient les divers mineurs de $M_{\mathcal{Z}}$ et par conséquent le rang de M en utilisant la propriété suivante, qui découle du théorème du reste chinois :

$$\text{si } \left\{ \begin{array}{ll} z \equiv z_1 & (\text{mod } a_i) \\ z \equiv z_2 & (\text{mod } a_j) \end{array} \right. \text{ et si } a_iu + a_jv = 1 \text{ alors } z \equiv a_iuz_1 + a_jvz_2 \pmod{a_ia_j}$$

On va appliquer cette méthode à l'exemple déjà rencontré plus haut :

- on remplace x et y par deux valeurs entières aléatoires comprises entre 0 et 100, en l'occurence 71 et 98, dans la matrice :

$$\begin{pmatrix} \frac{x+y}{xy} & \frac{2x+y}{1+y^2} & \frac{y}{x-y} \\ \frac{x-y}{x^2} & \frac{2x}{x^2-y} & \frac{x}{x-y^2} \\ \frac{x^2-y^2}{xy} & \frac{x^2+1}{x+y} & \frac{y}{x} \end{pmatrix}$$

  on obtient la matrice à coefficients rationnels :

$$\begin{pmatrix} \frac{169}{6958} & \frac{48}{1921} & \frac{-98}{27} \\ \frac{-27}{5041} & \frac{142}{4943} & \frac{-71}{9533} \\ \frac{-4563}{6958} & \frac{5042}{169} & \frac{98}{71} \end{pmatrix}$$

  qui se transforme en la matrice à coefficients entiers :

$$\begin{pmatrix} 4956134576065702263 & 5098643920827501408 & -740633107443783765484 \\ -1092918750585036102 & 5861901873066152004 & -1519740950307667542 \\ -133815633553773961101 & 6087751929039491554788 & 281649209872988192508 \end{pmatrix}$$

- le plus grand coefficient de la matrice précédente est :

$$6087751929039491554788$$

la borne des coefficients est donc :

$$135369894768128611068185704593946765900719638866321952001860539123 2$$

- il s'agit donc de considérer les 40 premiers nombres premiers, leur produit étant supérieur à cette borne, et de calculer les différents mineurs modulo ces 40 nombres. On aurait aussi pu considérer les 30 premiers nombres premiers supérieurs à 100, ou les 17 derniers nombres premiers inférieurs à 10000,...
  - modulo 2 la matrice associée est :

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}$$

    elle est de rang 1,
  - modulo 11 la matrice associée est :

$$\begin{pmatrix} 3 & 1 & 2 \\ 9 & 3 & 7 \\ 7 & 10 & 9 \end{pmatrix}$$

    et elle est de rang 3
  - modulo 97 la matrice associée est :

$$\begin{pmatrix} 6 & 31 & 39 \\ 29 & 85 & 53 \\ 32 & 5 & 89 \end{pmatrix}$$

    et elle est aussi de rang 3
  - etc ...
- les valeurs obtenues pour le déterminant d'ordre 3 sont respectivement : 0, 0, 1, 0, 10, 0, 0, 7, 19, 1, 21, 0, 8, 5, 27, 47, 41, 12, 58, 0, 7, 58, 47, 65, 39, 85, 77, 0, 108, 0, 109, 21, 122, 48, 52, 124, 153, 18, 51, 101, ce qui permet de reconstruire le déterminant de la matrice à coefficients entiers :

$$44034208811268166199503865680868074012305693496417857843887 76$$

En fait, cet exemple montre que le gain au niveau de la taille des nombres à traiter n'est pas évident en utilisant cette méthode, même si l'essentiel des calculs se pratiquant dans les corps $\mathcal{Z}/p_i\mathcal{Z}$ sont beaucoup plus rapides.

Pour obtenir de meilleurs performances de calcul on va envisager une méthode, tenant compte des deux précédentes et intégrant les avantages du calcul probabiliste et du calcul modulaire.

une méthode probabiliste modulaire

Pour calculer le rang d'une matrice dont les coefficients sont des fractions rationnelles, on va choisir un nombre premier a et des valeurs arbitraires de $\mathcal{Z}/a\mathcal{Z}$ pour chacune des variables présentes dans les coefficients de la matrice.

Ceci revient à associer à la matrice M, dont les coefficients appartiennent à

$$\mathcal{Q}(X_1, \ldots, X_l, T_{l+1}, \ldots, T_m)$$

une matrice à coefficients dans $\mathcal{Z}/a\mathcal{Z}$, que l'on notera $M_{\mathcal{Z}/a\mathcal{Z}}$.

Pour être sûr que la fonction d'évaluation modulaire $f_{\mathcal{Z}/a\mathcal{Z}}$ définie par :

$$\begin{array}{rcccl} f_{\mathcal{Z}/a\mathcal{Z}} & : & \mathcal{Q}(X_1, \ldots, X_l, T_{l+1}, \ldots, T_m) & \rightarrow & \mathcal{Z}/a\mathcal{Z} \\ & & X_i \text{ pour } i = 1, \ldots, m & \mapsto & \breve{x}_i \text{ pour } i = 1, \ldots, m \end{array}$$

associe bien à M une matrice $M_{\mathcal{Z}/a\mathcal{Z}}$, il suffit de vérifier que le m-uplet $(\breve{x}_1, \ldots, \breve{x}_m)$ n'est pas racine, modulo a, d'un des dénominateurs des coefficients de M – et, s'il n'existe pas de tels m-uplets, considérer un autre nombre premier a.

Comme dans la méthode probabiliste vue précédemment, il est clair que si un mineur de $M_{\mathcal{Z}/a\mathcal{Z}}$ n'est pas nul, le mineur correspondant de M ne l'est pas non plus, aussi le rang de M est certainement supérieur ou égal au rang de $M_{\mathcal{Z}/a\mathcal{Z}}$.

Réciproquement, si un mineur de $M_{\mathcal{Z}/a\mathcal{Z}}$ est nul, le mineur correspondant de M l'est aussi, sauf si le m-uplet $(\breve{x}_1, \ldots, \breve{x}_m)$ est racine, modulo a, d'un des polynômes intervenant pendant le calcul de ce mineur. Ainsi, la probabilité ϱ de rencontrer ce cas défavorable est donnée par le rapport entre le nombre de m-uplets $(\breve{x}_1, \ldots, \breve{x}_m)$ que l'on peut choisir et le nombre de m-uplets $(\breve{x}_1, \ldots, \breve{x}_m)$ qui sont racines, modulo a, de polynômes de Ω – l'ensemble de tous les polynômes intervenant pendant le calcul du déterminant.

Si l'on utilise un processus de choix aléatoire des $\breve{x}_i$, toutes les valeurs que peut prendre un polynôme, modulo a, en ce point ne peuvent être considérées comme équiprobables et au nombre de a que si a est grand par rapport au degré maximal de ce polynôme.

On peut de plus borner le degré du polynôme de plus haut degré qui intervient dans le calcul du déterminant d'une matrice M de taille n par n fois le degré du polynôme de plus haut degré parmi ses coefficients.

On considèrera, dans ce qui suit que a est grand par rapport à cette borne.

Dans ce cas, la probabilité qu'un m-uplet soit racine d'un polynôme est donc voisine de $\frac{1}{a}$ et, si n est la taille de M, la probabilité ϱ que le rang de $M_{\mathcal{Z}/a\mathcal{Z}}$ soit le même que celui de M est majorée par :

$$\frac{2^{2n}}{a}$$

puisque le nombre de polynômes à calculer est majoré par 2^{2n}.

On peut faire tendre cette probabilité vers 0 en prenant des nombres premiers de plus en plus grands, ou en utilisant la même méthode avec plusieurs nombres premiers différents – et suffisament grands. En effet si le rang calculé à partir de $M_{\mathcal{Z}/a_1\mathcal{Z}}$ et de $M_{\mathcal{Z}/a_2\mathcal{Z}}$ est le même, la probabilité ϱ devient

$$\frac{2^{4n}}{a_1 a_2}$$

Appliquée toujours au même exemple, cette méthode consiste à :

- choisir un nombre premier a, on prend ici 997,

- remplacer x et y par deux valeurs entières aléatoires comprises entre 0 et 997, en l'occurence 440 et 43, dans la matrice :

$$\begin{pmatrix} \frac{x+y}{xy} & \frac{2x+y}{1+y^2} & \frac{y}{x-y} \\ \frac{x-y}{x^2} & \frac{2x}{x^2-y} & \frac{x}{x-y^2} \\ \frac{x^2-y^2}{xy} & \frac{x^2+1}{x+y} & \frac{y}{x} \end{pmatrix}$$

  ce qui donne la matrice à coefficients dans $\mathcal{Z}/997\mathcal{Z}$:

$$\begin{pmatrix} 976 & 319 & 530 \\ 172 & 408 & 299 \\ 636 & 149 & 93 \end{pmatrix}$$

- calculer les mineurs dans $\mathcal{Z}/997\mathcal{Z}$, on obtient 180 pour le déterminant d'ordre 3.

Sur cet exemple, la matrice est trop petite pour que l'on puisse avoir un gain de temps et de place vraiment appréciable par rapport aux précédentes méthodes. Par contre, on note déjà un gain de l'ordre d'un facteur 10 en vitesse par rapport au calcul "classique" du rang à l'aide de la méthode de Gauss.

implémentation

L'algorithme implémentant la méthode probabiliste modulaire décrite ci-dessus prend comme entrée la matrice M à coefficients polynômiaux en les variables $(X_1, \ldots, X_l, T_{l+1}, \ldots, T_m)$ et fournit en sortie son rang. Il est le suivant :

i. choisir un nombre premier a - on le prendra le plus grand possible en dessous de 10000 car les entiers de MAPLE sont codés en base 10000,

ii. **faire**

- générer aléatoirement m entiers modulo a, $(x_1, \ldots, x_m)$,
- pour $i = 1, \ldots, m$ substituer X_i (resp. T_{l+i}) par x_i (resp. x_{l+i}) dans M,
- calculer modulo a l'ensemble des dénominateurs des coefficients de M,

jusqu'à ce que 0 n'appartienne pas à cet ensemble,

iii. calculer M modulo a,

iv. calculer le rang de M modulo a après l'avoir triangularisée à l'aide de la méthode de Gauss.

Les étapes (2) à (5) peuvent être éxécutées plusieurs fois pour des valeurs de a différentes pour diminuer la probabilité que le rang obtenu ne soit pas le rang exact de la matrice de contrainte.

Cet algorithme, en MAPLE version V, peut se coder au moyen des deux procédures ci-dessous,

- `modgauss` calculant le rang d'une matrice à coefficents modulaires, après l'avoir triangularisée,
- `rangpm` éxécutant l'algorithme, pour différentes valeurs de a.

On trouvera les codes correspondants à ces procédures, ainsi que le code de la procédure `trigfree` – le filtre transformant la matrice de contrainte dépendante des p_i en une matrice à coefficients dans $\mathcal{Q}(X_1, \ldots, X_l, T_{l+1}, \ldots, T_m)$, en annexe (§C.2).

4.3.2 deuxième approche

Ici, on assume l'hypothèse (ii), à savoir que les coefficients de $\Delta_{\mathcal{V}}$ appartiennent à l'anneau quotient :

$$\mathcal{Q}[X_1, \ldots, X_l, Y_{l+1}, \ldots, Y_m, Z_{l+1}, \ldots, Z_m]/(Y_{l+1}^2 + Z_{l+1}^2 - 1, \ldots, Y_m^2 + Z_m^2 - 1)$$

En effet, puisque Y_{l+i} et Z_{l+i} représentent respectivement le sinus et le cosinus d'une même variable p_{l+i}, on a bien évidemment : $Y_{l+i}^2 + Z_{l+i}^2 = 1$. Il est donc nécessaire, si l'on regarde les coefficients d'une matrice de contrainte comme des polynômes en $(X_1, \ldots, X_l, Y_{l+1}, \ldots, Y_m, Z_{l+1}, \ldots, Z_m)$ de les quotienter par ces relations. ◇

Cet anneau quotient ne possède pas de diviseur de zéro, on peut donc définir la notion de rang d'une matrice, en disant que c'est la dimension de l'image de l'application linéaire qu'elle représente – en tant qu'espace vectoriel sur le corps des fractions de cette anneau.

De plus, cette hypothèse entraîne aussi que les équations $g_1(p) = \ldots = g_\nu(p) = 0$ qui définissent la sous-variété $\mathcal{V}$ de $\mathcal{V}_{\text{arb}}$ peuvent s'écrire sous forme polynômiale :

$$\begin{cases} G_1(X_1, \ldots, X_l, Y_{l+1}, \ldots, Y_m, Z_{l+1}, \ldots, Z_m) = 0 \\ \vdots \\ G_\nu(X_1, \ldots, X_l, Y_{l+1}, \ldots, Y_m, Z_{l+1}, \ldots, Z_m) = 0 \end{cases}$$

Et dans ce cas, on peut se ramener au problème de la détermination de la dimension *réelle* d'une variété algébrique.

Que l'on veuille ainsi calculer directement la dimension de $\mathcal{V}$ ou calculer le rang de la matrice $\Delta_{\mathcal{V}}$ dans ce cadre, il est necessaire de faire appel à des outils algébriques puissants et notamment à la théorie des bases standards (ou base de Gröbner).

On décrira les méthodes envisagées et expérimentées après avoir présenté ces outils.

4.4 où l'on utilise des bases standards

On considère un corps K et n indéterminées $X_1, X_2, \ldots, X_n$. On note $K[X_1, X_2, \ldots, X_n]$ – ou de manière abrégée $K[X]$ l'anneau des polynômes en les indéterminées $X_1, X_2, \ldots, X_n$ à coefficients dans K

On rappelle les deux définitions fondamentales suivantes :

Définition 4.7 *Un idéal I de $K[X]$ est un sous-ensemble de $K[X]$ qui vérifie les deux propriétés suivantes :*

i. I est un sous-groupe du groupe additif $(K[X], +)$,

ii. pour tout polynôme de $K[X]$, son produit par tout polynôme de I est un polynôme de I :

$$\forall P \in K[X], \quad \forall Q \in I, \quad PQ \in I$$

Définition 4.8 *Soit A une partie de $K[X]$, il existe un plus petit – au sens de l'inclusion ensembliste – idéal de $K[X]$ qui contient A ; c'est l'intersection de tous les idéaux de $K[X]$ qui contiennent A ; on l'appelle idéal engendré par A et on dit des polynômes de A qu'ils forment une famille génératrice de cet idéal.*

Tout idéal I possède une infinité de familles génératrices, puisqu'il suffit d'adjoindre à une famille génératrice une combinaison à coefficients polynômiaux des polynômes de cette famille pour obtenir une nouvelle famille génératrice du même idéal.

4.4.1 bases standards d'idéaux de polynômes

La notion de *base standard* (ou *base de Gröbner*) est née de la volonté de pouvoir associer – algorithmiquement – à un idéal donné par une de ses familles de générateurs $g_1, \ldots, g_r$, une autre famille de générateurs $f_1, \ldots, f_s$ qui satisfasse la propriété suivante :

Tout polynôme P de $K[X]$ peut s'écrire :

$$\sum_{i=1}^{s} h_i f_i + h \qquad (h_i \in K[X])$$

par divisions successives et,

- si $P \in I$ alors $h = 0$,
- sinon h ne dépend pas de l'ordre dans lequel sont effectuées les divisions.

Ainsi la connaissance d'une base standard d'un idéal fournit, entre autre, un critère simple d'appartenance à un idéal, et un moyen de classifier les polynômes.

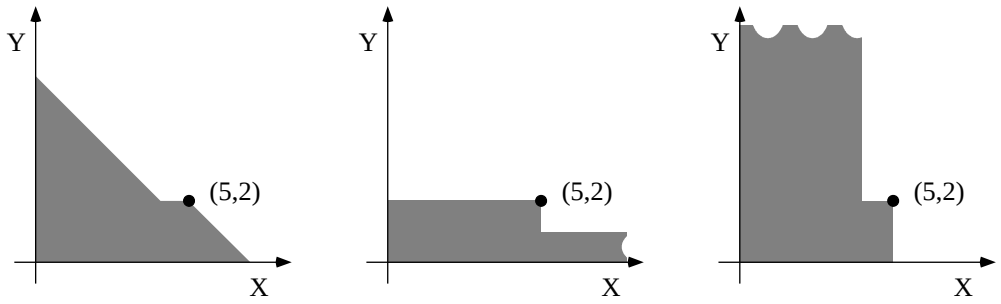
Avant de présenter un algorithme de calcul de bases standards, il est nécessaire de préciser la notion de division euclidienne des polynômes à n variables qui généralise la division euclidienne bien connue des polynômes à une seule variable.

ordres et division euclidienne

Pour cela, il faut, au préalable définir un ordre total sur $\mathbb{N}^n$ pour pouvoir ordonner les monômes selon leurs degrés et considérer le terme de plus haut degré d'un polynôme.

En effet, à tout monôme ${X_1}^{p_1} \ldots {X_n}^{p_n}$ on associe naturellement le n-uplet $(p_1, \ldots, p_n)$ de $\mathbb{N}^n$ pour définir son degré.

Les ordres totaux les plus couramment employés sont l'ordre diagonal, l'ordre lexicographique inverse et l'ordre lexicographique – voir figure (4.8).



Pour chaque ordre, sont teintées les régions des monômes inférieurs au monôme X^5Y^2. De gauche à droite sont représentés les ordres diagonal, lexicographique inverse et lexicographique.

figure 4.8 : exemples d'ordres totaux dans $\mathbb{N}^2$

On suppose qu'un tel ordre a été fixé une fois pour toute. On peut ainsi définir le terme de plus haut degré d'un polynôme P, $\mathrm{lt}(P)$ et son exposant $\mathrm{le}(P)$.

Par exemple, si $P = XY^4Z + X^3Y^2 + Z^5$, on a en fonction de l'ordre fixé :

- diagonal : $\mathrm{lt}(P) = XY^4Z$ et $\mathrm{le}(P) = (1, 4, 1)$

- lexicographique : $\mathrm{lt}(P) = X^3Y^2$ et $\mathrm{le}(P) = (3,2,0)$
- lexicographique inverse : $\mathrm{lt}(P) = Z^5$ et $\mathrm{le}(P) = (0,0,5)$

Si on considère une famille de polynômes $B_1, \ldots, B_r$, on peut partitionner $\mathbb{N}^n$ sous la forme* :

$$\mathbb{N}^n = \bigsqcup_{i=1}^{r} \Delta_i \quad \text{avec} \quad \begin{cases} \Delta_1 = \{\mathrm{le}(B_1)\} \dot{+} \mathbb{N}^n \\ \Delta_i = (\{\mathrm{le}(B_1)\} \dot{+} \mathbb{N}^n) \setminus \left(\bigcup_{j=1}^{i-1} \Delta_j\right) \quad \text{si } 1 < i \leq r \\ \Delta_0 = \mathbb{N}^n \setminus \left(\bigcup_{j=1}^{r} \Delta_j\right) \end{cases}$$

Le théorème suivant permet de généraliser la division euclidienne aux polynômes à n indéterminées.

Théorème 4.9 *Soit $B_1, \ldots, B_r$ une famille de polynômes de $K[X]$ et $\Delta_0, \ldots, \Delta_r$ la partition de $\mathbb{N}^n$ qui lui est associée, soit P un polynôme de $K[X]$, il existe de manière unique une famille de polynôme $Q_1, \ldots, Q_r$ et un polynôme R de $K[X]$ tels que :*

i. $P = \left(\sum_{i=1}^{r} B_i Q_i\right) + R$,

ii. pour $i = 1 \ldots r$,
les monômes de Q_i ont tous leur exposant dans $\Delta_i \dot{-} \{\mathrm{le}(B_i)\}$ et $\mathrm{le}(B_i Q_i) \leq \mathrm{le}(P)$,

iii. les monômes de R ont tous leur exposant dans Δ_0 et $\mathrm{le}(R) \leq \mathrm{le}(P)$.

Remarque 4.10 Pour un ordre fixé sur $\mathbb{N}^n$, la partition $\Delta_0, \ldots, \Delta_r$ et par là même la division euclidienne dépendent de la manière dont est ordonnée la famille $B_1, \ldots, B_r$.

En effet, si dans $\mathbb{R}[X,Y]$ on effectue la division de $P = X^6Y^3 + X^5Y - X^2Y^2$ – en considérant l'ordre lexicographique – par (B_1, B_2) et par (B_2, B_1) avec $B_1 = X^5$ et $B_2 = X^3Y - Y^2$, on obtient respectivement :

- $P = (XY^3 + Y)B_1 + R$ avec $R = -X^2Y^2$,
- et $P = (X^3Y^2 + X^2 + Y^3)B_2 + R$ avec $R = Y^5$.

et les partitions correspondantes – voir figure (4.9)

base standard et algorithme de calcul

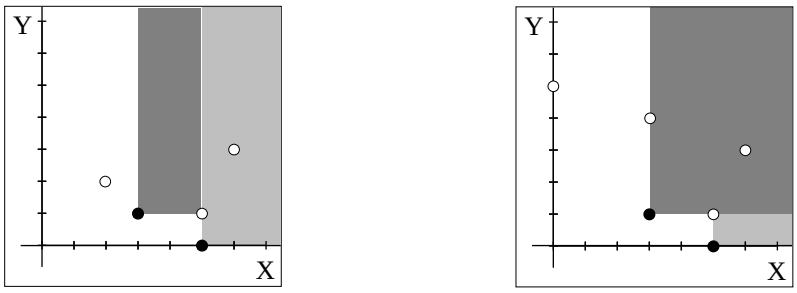
On peut maintenant donner la définition d'une base standard d'un idéal :

Définition 4.11 *On appelle base standard (ou base de Gröbner) d'un idéal I de $K[X]$ tout ensemble de polynômes de $K[X]$, $\{f_1, \ldots, f_s\}$ tel que :*

$$\{\mathrm{le}(f),\ f \in I\} = \bigcup_{i=1}^{s} (\{\mathrm{le}(f_i)\} \dot{+} \mathbb{N}^n)$$

Remarque 4.12 Dans la mesure où le reste de la division d'un polynôme par une base standard ne dépend pas de l'ordre dans lequel sont effectuées les divisions successives et comme une base standard sert essentiellement à tester l'appartenance d'un polynôme à un idéal, c'est à dire à déterminer si ce reste est nul ou non, il n'est pas nécessaire d'ordonner les polynômes d'une base standard. C'est pourquoi on a défini une base standard comme un ensemble – et non une liste – de polynômes.

*Si E et F sont deux sous-ensembles de $\mathbb{N}$, on note $E \dot{+} F$ (resp. $E \dot{-} F$ l'ensemble des sommes (resp. différences) d'un élément de E et d'un élément de F : $E \dot{+} F = \{x + y \ / \ x \in E, y \in F\}$ (resp. $E \dot{-} F = \{x - y \ / \ x \in E, y \in F\}$ et on note $E \setminus F$ la différence ensembliste de E et de F.



A gauche est représentée la partition associée à la division par (B_1, B_2), à droite celle associée à la division par (B_2, B_1). Les parties sombres (resp. claires) matérialisent l'ensemble Δ associé à B_1 (resp. à B_2). Les point noirs positionnent les monômes de tête de B_1 et de B_2. Les points blancs positionnent les monômes qui interviennent dans la division.

figure 4.9 : exemples de partitions associées à la division euclidienne

On considère un idéal I de $K[X_1, X_2, \ldots, X_n]$ et $g_1, \ldots, g_r$ une famille de polynômes génératrice de I.
Pour deux polynômes quelconques, P et Q de $K[X]$, on définit l'opération binaire $\curlywedge$ par :

$$P \curlywedge Q = bX^{\gamma-\aleph}P - aX^{\gamma-\beth}$$

dès que $\mathrm{lt}(P) = aX^{\aleph}$ et $\mathrm{lt}(Q) = bX^{\beth}$ et en posant $\gamma = \max(\aleph, \beth)$ i.e. $\gamma_i = \max(\aleph_i, \beth_i)$, pour $i = 1 \ldots n$.
De plus, on note $P \circledast f$ le reste de la division euclidienne d'un polynôme P par une famille de polynômes $f = (f_1, f_2, \ldots, f_{s(f)})$.
Alors on obtient une base standard de l'idéal I en itérant le principe suivant :

- si, pout tout couple d'indice (i, j) compris entre 1 et r avec $i < j$, on a :

  $$(g_i \curlywedge g_j) \circledast (g_1, \ldots, g_r) = 0$$

  alors $(g_1, \ldots, g_r)$ est une base standard,
- sinon, il existe au moins un couple (i_0, j_0) tel que cette quantité soit non nulle, alors on pose :

  $$g_{r+1} = (g_{i_0} \curlywedge g_{j_0}) \circledast (g_1, \ldots, g_r)$$

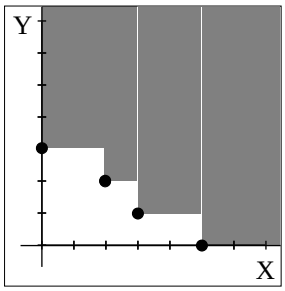
  et on s'intéresse à la famille $(g_1, \ldots, g_{r+1})$.

  Par exemple, on veut calculer une base standard – pour l'ordre lexicographique – de $I \subset \mathbb{R}[X, Y]$ engendré par $g_1 = X^5$ et $g_2 = X^3Y - Y^2$. On obtient successivement :

  i. $(g_1 \curlywedge g_2) \circledast (g_1, g_2) = X^2Y^2$ donc on pose $g_3 = X^2Y^2$,
  ii. $(g_1 \curlywedge g_3) \circledast (g_1, g_2, g_3) = 0$ et $(g_2 \curlywedge g_3) \circledast (g_1, g_2, g_3) = -Y^3$ donc on pose $g_4 = -Y^3$,
  iii. $(g_1 \curlywedge g_4) \circledast (g_1, g_2, g_3, g_4) = 0$, $(g_2 \curlywedge g_4) \circledast (g_1, g_2, g_3, g_4) = 0$ et $(g_3 \curlywedge g_4) \circledast (g_1, g_2, g_3, g_4) = 0$ donc donc (g_1, g_2, g_3, g_4) est une base standard – voir figure (4.10) la partition correspondante.

Des considérations algébriques dont la plus simple est la suivante :

$$\text{si} \quad (g_i \curlywedge g_j) \circledast (g_1, \ldots, g_{r+k}) = 0 \quad \text{alors} \quad (g_i \curlywedge g_j) \circledast (g_1, \ldots, g_{r+k+1}) = 0$$



En blanc est représenté Δ_0, en sombre $\bigcup_{i=1}^{4} \Delta_i$.

figure 4.10 : exemples de partition associée à une base standard

permettent de se dispenser de calculer, à chaque étape, toutes les quantités $(g_i \curlywedge g_j) \divideontimes (g_1, \ldots, g_{r+k})$.

A chaque méthode de choix des couples (g_i, g_j) à considérer – par analogie avec la théorie de la réécriture on parlera de *méthode d'élimination de paires critiques* – on associera donc un algorithme de calcul de bases standards. Et, bien entendu, c'est de la qualité de cette méthode que dépendront les performances de ces algorithmes.

Toutefois, la complexité de calcul, en temps et en espace, de tous ces algorithmes est au moins exponentielle en la taille des entrées – à savoir le nombre de monômes et leurs degrés – ce qui limitera grandement leurs possibilités d'utilisation.

4.4.2 bases standards de sous-modules

Les notions que l'on vient de voir dans la section précédente peuvent se généraliser aux sous-modules de $(K[X])^p$.

On rapelle qu'un *module* est à un anneau ce qu'un espace vectoriel est à un corps, c'est à dire qu'un module est un ensemble E muni d'une loi additive $\boxplus$ qui en fait un groupe abélien et muni d'une loi externe ., dont le domaine d'opérateurs est un anneau $(A, +, \times)$, qui possède les propriétés classiques :

i. $\forall \alpha \in A, \quad \forall (x, y) \in E^2, \quad \alpha.(x \boxplus y) = \alpha.x \boxplus \alpha.y,$

ii. $\forall (\alpha, \beta) \in A^2, \quad \forall x \in E, \quad (\alpha + \beta).x = \alpha.x \boxplus \beta.x,$

iii. $\forall (\alpha, \beta) \in A^2, \quad \forall x \in E, \quad \alpha.(\beta.x) = (\alpha \times \beta).x,$

iv. $\forall x \in E, 1_A.x = x$

De plus, on définit la *dimension* d'un module comme sa dimension en tant qu'espace vectoriel sur le corps des fractions de son anneau de base.

Ainsi, $(K[X])^p$ est un module construit sur l'anneau $K[X]$ dont les éléments sont des p-uplets de polynômes et qui est muni des deux lois + et . définies naturellement par :

- si $\hat{P} = (P_1, \ldots, P_p)$ et $\hat{Q} = (Q_1, \ldots, Q_p)$, alors $\hat{P} + \hat{Q} = (P_1 + Q_1, \ldots, P_p + Q_p)$,
- si, de plus R est un polynôme de $K[X]$, alors $R.\hat{P} = (RP_1, \ldots, RP_p)$.

On peut définir canoniquement, en introduisant p indéterminées supplémentaires, un isomorphisme – de $K[X]$-modules – entre $(K[X])^p$ et $K[X][Y_1 \ldots Y_p]$ en faisant correspondre au p-uplet $(P_1, \ldots, P_p)$ le polynôme $P_1Y_1 + \ldots + P_pY_p$.

Alors, si on considère un sous-module de $(K[X])^p$ engendré par une famille $\hat{P}_1, \ldots, \hat{P}_r$ de vecteurs[†] – un sous-module engendré par une famille de vecteurs est, comme on le pense, le plus petit sous-module, au sens de l'inclusion des ensembles, qui contient cette famille – on peut définir la notion de base standard de ce sous-module en utilisant l'isomorphisme canonique comme le montre le diagramme suivant :

$$\begin{array}{ccc} (K[X])^p & \Longleftrightarrow & K[X][Y_1 \ldots Y_p] \\ \hat{P}_1, \ldots, \hat{P}_r & \longrightarrow & \check{P}_1, \ldots, \check{P}_r \\ & & \text{base} \downarrow \text{standard} \\ \hat{Q}_1, \ldots, \hat{Q}_s & \longleftarrow & \check{Q}_1, \ldots, \check{Q}_s \end{array}$$

Pour calculer une base standard d'un sous module, on peut ainsi utiliser l'algorithme vu précédemment après avoir transformé les vecteurs de $(K[X])^p$ en polynômes de $K[X][Y_1 \ldots Y_p]$.

syzygies et pseudo-noyaux d'une matrice

En conservant la trace des calculs faits lorsqu'on veut déterminer une base standard d'un idéal donné par une famille de générateurs $(g_1, \ldots, g_r)$, on possède des *relations* de la forme :

$$\sum_{i=1}^{s} a_i g_i = 0 \quad \text{avec} \quad a_i \in K[X]$$

et dans la mesure où les polynômes $g_{r+1}, \ldots, g_s$ sont obtenus comme des combinaisons linéaires à coefficients polynômiaux des $g_1, \ldots, g_r$, on peut facilement, par substitution, obtenir des relations de la forme :

$$\sum_{i=1}^{r} b_i g_i = 0 \quad \text{avec} \quad b_i \in K[X]$$

De manière analogue, on peut obtenir, en calculant une base standard d'un sous-module engendré par une famille de vecteurs $\hat{P}_1, \ldots, \hat{P}_r$, un ensemble de relations de la forme :

$$\sum_{i=1}^{r} Q_i \hat{P}_i = 0 \quad \text{avec} \quad Q_i \in K[X]$$

Ces relations sont appelées *premières syzygies* du sous-module.

A travers le calcul des premières syzygies d'un sous-module, la mise en œuvre d'un algorithme de calcul de bases standards permet donc de calculer un *pseudo-noyau* d'une matrice M à coefficients dans un anneau de polynômes, c'est à dire une matrice N à coefficients dans le même anneau et telle que $M.N = 0$.

> En effet, si M est une matrice $l \times r$ dont les coefficients sont des polynômes de $K[X]$, et si on considère le sous-module engendré par les colonnes $M_1, \ldots, M_r$ de M (avec $M_i = (M_{1i}, M_{2i}, \ldots, M_{li})$ pour $1 \leq i \leq r$), les premières syzygies de ce sous-module consistent en m relations qui s'écrivent :
>
> $$\sum_{i=1}^{r} N_{im} M_i = 0$$

[†]Par abus de langage, on parlera de *vecteurs* pour désigner les éléments d'un module tout en gardant présent à l'esprit la différence avec un espace vectoriel.

On peut alors les regrouper sous la forme matricielle :

$$\left(\begin{array}{c} M \end{array} \right) \left(\begin{array}{c} N \end{array} \right) = 0$$

où N est la matrice $r \times m$ dont le coefficient de la i-ème ligne et de la j-ème colonne n'est autre que N_{ij}.

4.4.3 dimension d'une sous-variété de configurations

Si l'on considère une variété algébrique donnée par un idéal de polynômes, et si ces polynômes sont homogènes, le calcul d'une base standard de cet idéal pour l'ordre diagonal permet de calculer facilement la dimension sur $\mathbb{C}$ – l'ensemble des nombres complexes – de cette variété.

En effet, la forme et la taille de l'ensemble Δ_0 de la partition de $\mathbb{N}^n$ associée à cette base standard – on dit de cet ensemble que c'est l'*escalier* de la base standard – peuvent se représenter au moyen d'un polynôme – appelé *polynôme de Hilbert* de l'idéal – dont le degré est la dimension de la variété.

La connaissance de la base standard détermine l'escalier qui permet de calculer le polynôme de Hilbert et d'avoir, in fine, la dimension de la variété.

Remarque 4.13 Toutefois la dimension obtenue est une dimension sur $\mathbb{C}$ alors que la dimension intéressante pour le problème mécanique que l'on regarde est la dimension sur $\mathbb{R}$.

Les méthodes pour déterminer la dimension réelle d'une variété algébrique dont on connaît la dimension complexe sont encore l'objet d'une importante activité de recherche dans la communauté mathématique internationale.

On se contentera donc de considérer cette dimension complexe comme une majoration de la dimension réelle.

4.4.4 rang d'une matrice de contrainte

Lorsque l'on considère une matrice M $(\nu \times m)$ comme une représentation d'une application linéaire d'un espace vectoriel E^m dans un espace vectoriel E^ν, on peut dire que son rang, en tant que nombre de ses colonnes qui sont indépendantes est la dimension du sous-espace vectoriel $\mathrm{Im}M$ de E^ν, l'image de cette application.

On peut alors écrire la suite exacte[‡] :

$$0 \hookrightarrow \mathrm{Ker}M \xrightarrow{N} E^m \xrightarrow{M} \mathrm{Im}M (\subset E^\nu) \longrightarrow 0$$

où N est une matrice dont les colonnes forment une base du noyau de M.

Une des propriétés d'une suite exacte étant que la somme alternée des dimensions des espaces qui y interviennent est nulle, c'est à partir de cette suite exacte que l'on retrouve la formule bien connue :

$$\dim(\mathrm{Ker}M) + \dim(\mathrm{Im}M) = m$$

Si maintenant on considère M $(\nu \times m)$ comme une représentation d'une application linéaire d'un module E^m dans un module E^ν – ce qui est le cas dès que les coefficients de M ne font plus partie

[‡]Soient $E_0, \ldots, E_n$ des espaces vectoriels, on dit que le diagramme : $E_0 \xrightarrow{f_1} E_1 \xrightarrow{f_2} \ldots \xrightarrow{f_{n-1}} E_{n-1} \xrightarrow{f_n} E_n$ est une suite exacte si et seulement si : $\mathrm{Im}f_i = \mathrm{Ker}f_{i+1}, \quad (1 \leq i \leq n-1)$.

d'un corps mais d'un anneau, ici E représente $K[X]$ – alors on peut écrire la suite exacte suivante , que l'on appelle *résolution* du sous-module image de M :

$$0 \hookrightarrow E^{\mu_d} \xrightarrow{\Psi_d} E^{\mu_{d-1}} \xrightarrow{\Psi_{d-1}} \dots \xrightarrow{\Psi_2} E^{\mu_1} \xrightarrow{\Psi_1} E^m \xrightarrow{M} \mathrm{Im}M(\subset E^\nu) \longrightarrow 0$$

où :

- Ψ_1 est un pseudo-noyau de M construit à partir de ses premières syzygies,
- et pour tout i compris entre 1 et $d-1$,
  - Ψ_{i+1} est un annulateur de Ψ_i construit à partir de ses premières syzygies,
  - μ_{i+1} désigne le nombre de premières syzygies de Ψ_i i.e. le nombre de colonnes de Ψ_{i+1} ou le nombre de générateurs de E^{μ_i}.

En écrivant que la somme alternée des nombres de générateurs des modules qui interviennent dans la résolution précédente est nulle, on obtient :

$$m - \dim(\mathrm{Im}M) = m - \mathrm{rg}(M) = \sum_{i=1}^{d} \left((-1)^{i+1}\mu_i\right)$$

et cette notion de rang correspond bien à ce que l'on obtiendrait en évaluant au préalable la matrice de contrainte en un point non-singulier i.e. en dehors de l'ensemble des racines des polynômes présents dans les coefficients de M.

Toutefois, si l'on veut utiliser ce résultat pour calculer le rang d'une matrice de contrainte $\Delta_{\mathcal{V}}$, il ne faut pas oublier que le rang cherché est celui que l'on obtiendrait en évaluant la matrice $\Delta_{\mathcal{V}}$ sur les points de $\mathcal{V}$.

Il faut donc tenir compte des équations de contrainte :

$$g_1(p_1, \dots, p_m) = \dots = g_\nu(p_1, \dots, p_m) = 0$$

En assumant l'hypothèse que les fonctions g_i peuvent s'écrire comme des polynômes en

$$(X_1, \dots, X_l, Y_{l+1}, \dots, Y_m, Z_{l+1}, \dots, Z_m)$$

$$\text{par :} \qquad g_i(p_1, \dots, p_m) = G_i(X_1, \dots, X_l, Y_{l+1}, \dots, Y_m, Z_{l+1}, \dots, Z_m) \quad \text{pour} \quad i = 1, \dots, \nu$$

il suffirait alors de considérer que les coefficients de la matrice $\Delta_{\mathcal{V}}$ appartiennent non pas à :

$$\mathcal{Q}[X_1, \dots, X_l, Y_{l+1}, \dots, Y_m, Z_{l+1}, \dots, Z_m]$$

mais à cet anneau quotienté par l'idéal engendré par les G_i, et par les relations dues à la trigonométrie, c'est à dire :

$$\mathcal{Q}[X_1, \dots, X_l, Y_{l+1}, \dots, Y_m, Z_{l+1}, \dots, Z_m]/(Y_{l+1}^2 + Z_{l+1}^2 - 1, \dots, Y_m^2 + Z_m^2 - 1, G_1, \dots, G_\nu)$$

Remarque 4.14 Toute la théorie exposée dans le paragraphe (§4.4.1) (resp. dans le paragraphe (§4.4.2)) concernant les polynômes (resp. les vecteurs de polynômes) d'un anneau $K[X]$ (resp. d'un module $(k[X])^p$) se généralise immédiatement à des polynômes (resp. vecteurs de polynômes) d'un anneau quotient $K[X]/(Q_1, Q_2, \dots, Q_l)$ (resp. d'un module quotient $(K[X]/(Q_1, Q_2, \dots, Q_l))^p$.

Remarque 4.15 Malheureusement, le fait que la résolution d'un sous-module puisse s'écrire en un nombre fini d'étapes est lié à une propriété de *régularité* de l'anneau de base qui n'est en général pas vérifiée par les anneaux-quotients.

On ne pourra donc pas toujours tenir compte, en utilisant cette méthode, des équations définies par les fonctions g_i ou des relations imposées par la trigonométrie.

Dans le cas général, on n'obtiendra donc qu'une majoration du rang que l'on cherche comme on le verra dans l'expérimentation basée sur cette méthode – voir (§4.5).

On emploiera alors l'algorithme suivant :

i. calculer une base standard du sous-module engendré par les vecteurs colonnes de $\Delta_{\mathcal{V}}$ et récuperer les premières syzygies sous la forme d'une matrice Ψ_1

ii. faire, à partir de $i = 1$ et tant que Ψ_i est non nulle :
   - calculer une base standard du sous-module engendré par les vecteurs colonnes de Ψ_i et récuperer les premières syzygies sous la forme d'une matrice Ψ_{i+1}

iii. calculer la majoration du rang à partir des nombres de générateurs μ_i des modules de syzygies successifs en utilisant la formule :

$$m - \mathrm{rg}(\Delta_{\mathcal{V}}) = \sum_{i=1}^{d} \left((-1)^{i+1} \mu_i \right)$$

Remarque 4.16 Un effet de bord de cet algorithme est de nous fournir un moyen d'obtenir facilement une matrice Ψ de taille $(m \times (m - \mathrm{rg}(\Delta_{\mathcal{V}})))$ et de rang $m - \mathrm{rg}(\Delta_{\mathcal{V}})$.

> En effet, si l'on arrive à déterminer le rang de $\Delta_{\mathcal{V}}$, il suffira de considérer une sous-matrice de la matrice des premières syzygies, Ψ_1, de la bonne taille et dont les colonnes sont polynômialement indépendantes – i.e. sur lesquelles il n'existe pas de relation – ce qui se vérifiera rapidement à l'aide de la matrice de syzygies Ψ_2.

4.4.5 implémentations

De manière habituelle, sont implémentés dans la plupart des logiciels de calcul formel classiques – *MAPLE*, *MACSYMA*, *REDUCE* – des algorithmes de calcul de bases standards d'idéaux de polynômes proches de celui que l'on a présenté plus haut.

Il existe de plus un logiciel de calcul formel *MACAULAY* spécialisé dans les manipulations de géométrie algébrique et notamment les calculs de bases standards, de syzygies et de résolutions de modules.

C'est à la fois le seul logiciel qui calcule des bases standards de sous-modules et celui qui possède l'implémentation la plus efficace des algorithmes, mais il ne travaille qu'avec des polynômes, des idéaux et des sous-modules *homogènes*.

Pour expérimenter les méthodes de calcul de la dimension et du rang présentées aux paragraphes précédents, on s'est donc servi de *MACAULAY*. Les résultats obtenus seront exposés et commentés au paragraphe suivant.

Mais, on s'est aussi intéressé à implémenter intégralement cette méthode en *MAPLE*, dans l'optique d'inclure la fonctionnalité de calcul du rang – ou tout au moins d'une majoration du rang – de la matrice de contrainte de mécanismes à structure bouclée dans le logiciel *GEMMES* développé en *MAPLE*.

La bibliothèque standard `grobner` de *MAPLE* ne donnant pas entière satisfaction sur le plan des performances obtenues et ne permettant pas de traiter directement les sous-modules, on s'est attaché, tout d'abord, à implémenter dans ce système de nouveaux algorithmes de calcul de bases standards.

Vu la complexité inhérente à ces algorithmes, on a apporté un grand soin à leur programmation en suivant deux directions :

- l'amélioration théorique des algorithmes,
- les variantes d'implémentations et de représentation des données – en l'occurence des polynômes.

On a décidé, dans un premier temps, d'explorer ces deux voies dans le cadre restreint du calcul de bases standards dans les anneaux, ceci afin d'avoir une programmation et surtout une expérimentation plus facile et plus rapide – puisque le cas des sous-modules n'est qu'une "vectorisation" du cas des idéaux, toute amélioration se transpose immédiatement.

En ce qui concerne la comparaison entre les programmes résidents en *MAPLE* et ceux que l'on a écrit, on a utilisé une approche statistique et systématique portant sur un grand nombre de tests en faisant varier des paramètres tels que le degré total, le nombre de variables ou le nombre de termes des polynômes. Pour cela, on a développé un certain nombre d'outils à des fins d'analyse de performances : génération automatique de jeux de tests pseudo-aléatoires, comparaison et présentation des résultats.

En ce qui concerne l'implémentation des algorithmes, on a pris comme base de départ la procédure `gbasis` de la version 4.0 de *MAPLE* – cette procédure ayant été écrite par des gens connaissant bien le système de calcul formel utilisé, le style de programmation devait par conséquent être quasi-optimal.

Après une étude du code de `gbasis`, il est apparu que celui-ci était perfectible sur au moins deux points :

- la procédure de division `redall`,
- la méthode d'élimination des paires critiques – procédure `addpairs`.

Une nouvelle procédure de division a donc remplacé l'originale, le gain étant sur cette opération de 20% à 50% en temps.

Ensuite, on a programmé l'algorithme d'élimination des paires critiques dit "des β_i^j" – voir [Galligo, 1983] – basé sur une étude fine de la partition de $\mathbb{N}^n$ associée aux exposants de plus haut degré. Pour ce faire, on a introduit des changements majeurs dans le code qui s'est trouvé réduit – d'un tiers – et simplifié.

L'amélioration a été indiscutable puisque dans presque tous les cas testés la nouvelle procédure a été deux fois plus rapide que la procédure originale de *MAPLE*.

Ce doublement de la vitesse d'exécution, allié à un code simplifié et clarifié, bien qu'étant en soi un résultat intéressant ne suffit pourtant pas pour s'attaquer à une classe de problèmes sensiblement plus vaste que celle qui était à la portée de la procédure `gbasis`.

L'utilisation d'un outil de type *profiler* que l'on a développé spécifiquement, a permis de déterminer que la consommation de temps excessive se produisaient lors des opérations élémentaires sur les polynômes (addition, multiplication, determination du monôme de plus haut degré, ...) et était donc intrinsèque à leur représentation.

Toutefois les essais de programmation utilisant des représentations différentes des monômes, à savoir :

- des couples ordonnés d'entiers – coefficient, liste des exposants,
- des expressions mixtes, le degré d'un monôme étant représenté par un entier à l'aide d'une fonction de "hash-code" compatible avec la multiplication,

n'ont pas donné de résultats intéressants dans le contexte de programmation de *MAPLE*.

Les enseignement que l'on a pu tirer de cette expérimentation ont ouvert de nouvelles perspectives que l'on présentera dans la conclusion de cette étude.

Le code *MAPLE* correspondant à la procédure `basis` que l'on a développé ainsi qu'aux procédures subalternes appelées par `basis` se trouve en annexe (§C.3).

4.5 expérimentations

4.5.1 première approche : algorithmes de calcul du rang

Dans cette section on présente les résultats obtenus avec les algorithmes de calcul du rang classique et probabiliste modulaire en traitant les exemples décrits au paragraphe (§4.2).

algorithme classique

Chronologiquement, les premiers tests effectués pour calculer le rang des matrices de contrainte des mécanismes pris en exemples, l'ont été avec l'algorithme classique du pivot de Gauss implémenté en *MAPLE* par la procédure `gausselim`.

Pour l'exemple le plus simple, le résultat est satisfaisant, comme le montre la session transcrite ci-dessous :

```
> M1:=matrix([[0,0,l1*(1-t1^2)/(1+t1^2),
              l2*(1-t2^2)/(1+t2^2),l3*(1-t3^2)/(1+t3^2)],
             [0,0,-l1*2*t1/(1+t1^2),-l2*2*t2/(1+t2^2),-l3*2*t3/(1+t3^2)]]);

                [                2             2             2  ]
                [       l1 (1 - t1 )  l2 (1 - t2 )  l3 (1 - t3 ) ]
                [ 0  0  ------------  ------------  ------------ ]
                [                2             2             2   ]
                [          1 + t1        1 + t2        1 + t3    ]
          M1 := [                                                ]
                [             l1 t1         l2 t2         l3 t3  ]
                [ 0  0   - 2 -------   - 2 -------   - 2 ------- ]
                [                 2             2             2 ]
                [            1 + t1        1 + t2        1 + t3  ]

> map(factor,gausselim(M1,'r'));

 [           l1 t1                l2 t2                     l3 t3         ]
 [ 0  0  - 2 -------          - 2 -------                - 2 -------        ]
 [              2                    2                          2         ]
 [           1 + t1               1 + t2                     1 + t3        ]
 [                                                                         ]
 [                    l2 (t2 t1 + 1) (- t2 + t1)  l3 (t3 t1 + 1) (- t3 + t1) ]
 [ 0  0       0       -------------------------  ------------------------- ]
 [                                  2                          2           ]
 [                         t1 (1 + t2 )               t1 (1 + t3 )        ]

> r;

                                    2

> eq11:=M1[2,3]+M1[2,4]+M1[2,5];

                           l1 t1       l2 t2       l3 t3
                eq1 := - 2 ------- - 2 ------- - 2 -------
                                  2           2           2
                             1 + t1      1 + t2      1 + t3

> eq12:=M1[1,3]+M1[1,4]+M1[1,5];

                              2             2             2
                         l1 (1 - t1 )  l2 (1 - t2 )  l3 (1 - t3 )
             eq2 := ------------ + ------------ + ------------
                              2             2             2
                        1 + t1        1 + t2        1 + t3
```

On constate assez facilement que les équations de contrainte `eq11` et `eq12` ne permettent pas d'annuler un des coefficients de la matrice triangularisée.

Pour l'exemple de la bielle, on trouve encore le rang attendu par l'intuition mécanique, mais il n'est pas aisé de vérifier que les coefficients de la matrice triangularisée ne sont pas des combinaisons polynômiales des fractions rationnelles exprimant les équations de contrainte :

```
> eq21:= expand(l1*cos(a1)+l2*cos(a1+a12)-x3,trig);
> eq22:= expand(l1*sin(a1)+l2*sin(a1+a12),trig);
> vector([eq21,eq22]):
> M2:=jacobian(",[a1,a12,x3]);
> lvar:=[sin(a1)=2*t1/(1+t1^2),sin(a12)=2*t2/(1+t2^2),
         cos(a1)=(1-t1^2)/(1+t1^2),cos(a12)=(1-t2^2)/(1+t2^2)];
> M2:=subs(lvar,op(M2));
> gausselim(M2,'r');

                          2                   2                   2
          - l1 t1 - l1 t1 t2  - l2 t1 + l2 t1 t2  - l2 t2 + l2 t2 t1
       [2 ------------------------------------------------------------,
                                    2         2
                             (1 + t1 ) (1 + t2 )

                          2             2
            l2 (- t1 + t1 t2  - t2 + t2 t1 )
          2 --------------------------------, -1]
                        2         2
                 (1 + t1 ) (1 + t2 )

                                          2
                          l2 l1 t2 (1 + t1 )
   [0, ------------------------------------------------------------, 1/2 (l1
                        2                   2                   2
       - l1 t1 - l1 t1 t2  - l2 t1 + l2 t1 t2  - l2 t2 + l2 t2 t1

              2        2        2   2             2        2        2   2
        + l1 t2  - l1 t1  - l1 t1  t2  + l2 - l2 t2  - l2 t1  + l2 t1  t2

        - 4 l2 t1 t2)

          /                   2                   2                   2
         /  (- l1 t1 - l1 t1 t2  - l2 t1 + l2 t1 t2  - l2 t2 + l2 t2 t1 )]
        /

> r;

                                    2

> eq21:=subs(lvar,eq21);

                    2             2         2
          l1 (1 - t1 )   l2 (1 - t1 ) (1 - t2 )           l2 t1 t2
   eq21 := ------------ + ---------------------- - 4 ------------------- - x3
                  2             2         2             2         2
             1 + t1        (1 + t1 ) (1 + t2 )      (1 + t1 ) (1 + t2 )

> eq22:=subs(lvar,eq22);

                                          2                   2
                l1 t1        l2 t1 (1 - t2 )         l2 (1 - t1 ) t2
      eq22 := 2 ------- + 2 ------------------- + 2 -------------------
                     2            2         2             2         2
               1 + t1      (1 + t1 ) (1 + t2 )     (1 + t1 ) (1 + t2 )
```

On pourra cependant le vérifier assez rapidement en testant au moyen d'un calcul de base standard l'appartenance des numérateurs des fractions rationnelles intervenant dans les coefficients à l'idéal engendré par les numérateurs des fractions rationnelles exprimant les équations de contrainte – après avoir réduit toutes ces fractions rationnelles au même dénominateur.

En ce qui concerne l'exemple de la salière, le calcul direct ne fournit pas le rang correct :

```
> M3;
```

```
              [                                     t2             t3   ]
              [          0                   - 2 -------       2 ------- ]
              [                                       2              2 ]
              [                                  1 + t2        1 + t3  ]
              [                                                         ]
              [             2                          2                ]
              [        (1 - t1 ) t2          t1 (1 - t2 )               ]
              [  2 -------------------   2 -------------------      0   ]
       M3 := [         2         2              2         2             ]
              [    (1 + t1 ) (1 + t2 )     (1 + t1 ) (1 + t2 )          ]
              [                                                         ]
              [                                2         2            2 ]
              [            t1 t2          (1 - t1 ) (1 - t2 )     1 - t3 ]
              [ - 4 -------------------   -------------------   - ------- ]
              [            2         2          2         2           2 ]
              [     (1 + t1 ) (1 + t2 )   (1 + t1 ) (1 + t2 )     1 + t3 ]

> eqs;

eqs :=

        2          2                                  2
  1 - t2    1 - t3            t1 t2             (1 - t1 ) t2           t3
[ ------- - -------, 4 -------------------, 2 ------------------- - 2 ------- ]
        2          2            2         2            2         2           2
  1 + t2    1 + t3     (1 + t1 ) (1 + t2 )    (1 + t1 ) (1 + t2 )     1 + t3


> M3t:=gausselim(M3,'r');

M3t :=
                                             2           2            2
                   t1 t2           (- 1 + t1 ) (- 1 + t2 )  - 1 + t3
         [- 4 -------------------, -----------------------, ---------]
                       2         2              2         2          2
               (1 + t1 ) (1 + t2 )    (1 + t1 ) (1 + t2 )     1 + t3

                                          t2         t3
                             [0, - 2 -------, 2 -------]
                                              2          2
                                        1 + t2     1 + t3

[0, 0,

                         2        2        2   2             2        2        2   2
            t2 - t2 t3  - t2 t1  + t2 t1  t3  - t3 + t3 t2  - t3 t1  + t3 t1  t2
    - 1/2 ----------------------------------------------------------------------
                                                        2
                                          t1 t2 (1 + t3 )

    ]

> r;

                                          3
```

En effet, on sait que ce mécanisme peut bouger et donc le rang de la matrice de contrainte ne peut être supérieur à 2.

On remarque alors que le coefficient `M3t[1,1]` de la première ligne et de la première colonne de la matrice triangularisée est multiple d'une des équations de contrainte.

En annulant ce coefficient et en recalculant la matrice, on obtient cette fois le rang correct.

```
> M3t[1,1]:=0:
> gausselim(M3t,'r');

                                          t2         t3
                             [0, - 2 -------, 2 -------]
                                              2          2
```

```
                                  1 + t2     1 + t3

[0, 0,

                  2        2        2   2             2        2        2   2
         t2 - t2 t3  - t2 t1  + t2 t1  t3  - t3 + t3 t2  - t3 t1  + t3 t1  t2
    - 1/2 ------------------------------------------------------------------
                                                  2
                                     t1 t2 (1 + t3 )

    ]

                                  [0, 0, 0]
> r;

                                       2
```

En triangularisant la matrice de contrainte du joint homocinétique, on obtient, là encore, le rang correct par rapport à la nature du mécanisme.

```
M4 :=

      /                        2                        2       \
      |           t1 t2 (1 - t3 )                (1 - t1 ) t3   |
     [|- 4 ----------------------------- - 2 -------------------| l,
      |           2         2         2             2         2 |
      \    (1 + t1 ) (1 + t2 ) (1 + t3 )     (1 + t1 ) (1 + t3 )/

                2         2         2
         (1 - t1 ) (1 - t2 ) (1 - t3 ) l
         -------------------------------,
                 2         2         2
          (1 + t1 ) (1 + t2 ) (1 + t3 )

         /                 2                                2    \
         |           (1 - t1 ) t2 t3                t1 (1 - t3 )   |
         |- 4 ----------------------------- - 2 -------------------| l, 0, 0]
         |           2         2         2             2         2 |
         \    (1 + t1 ) (1 + t2 ) (1 + t3 )     (1 + t1 ) (1 + t3 )/

       /             2            2                            \
       |      (1 - t1 ) t2 (1 - t3 )               t1 t3       |
      [|2 ----------------------------- - 4 -------------------| l,
       |         2         2         2             2         2 |
       \  (1 + t1 ) (1 + t2 ) (1 + t3 )     (1 + t1 ) (1 + t3 )/

                          2         2
              l t1 (1 - t2 ) (1 - t3 )
          2 -----------------------------,
                  2         2         2
            (1 + t1 ) (1 + t2 ) (1 + t3 )

            /                                          2         2 \
            |               t1 t2 t3             (1 - t1 ) (1 - t3 )|
          l |- 8 ----------------------------- + -------------------|, 0, 0]
            |           2         2         2             2         2 |
            \    (1 + t1 ) (1 + t2 ) (1 + t3 )   (1 + t1 ) (1 + t3 )/

                              2                     2
                     t2 (1 - t3 ) l           (1 - t2 ) t3 l

> gausselim(M4,`r`):
> r;

                                       4
```

La taille de la matrice triangularisée est trop importante pour qu'on puisse la reproduire ici.

L'utilisation les procédures de la bibliothèque `grobner` pour verifier que ses coefficients ne sont pas annulés par les équations de contrainte, nécessite plus de 30 minutes CPU sur une DECSTATION

5000/200.

Quant aux calculs des rangs des matrices de contrainte des exemples de mailles d'antenne, il a été nécessaire de les interrompre, le programme *MAPLE* occupant plus de 40 Mbytes d'espace mémoire – au dessus d'un certain seuil d'espace mémoire, le calcul ne progresse presque plus, le programme occupant plus de 90 % de son temps à la réorganisation de l'espace mémoire.

algorithme probabiliste modulaire

L'expérimentation des procédures que l'on a développées pour implémenter l'algorithme probabiliste modulaire de calcul du rang a donné, en ce qui concerne les quatre premiers exemples – l'exemple le plus simple, la bielle, la salière, le joint homocinétique – exactement les mêmes résultats que ceux obtenus avec l'algorithme classique.

Sur ces exemples, les temps de calculs avec les deux méthodes étaient trop faibles pour que l'on puisse tirer des conclusions.

Par contre l'étude des mailles d'antenne déployables à un et à deux niveaux a permis d'apprécier la rapidité de l'algorithme probabiliste modulaire.

Dans le premier cas – maille à un niveau – la matrice de contrainte posséde six lignes et huit colonnes et fait intervenir neuf variable.

La procédure `rangpm` donne la majoration de son rang en 100 secondes CPU sur un sun4.

En fait la valeur obtenu, 5, correspond au rang recherché puisque ce mécanisme possède un degré de liberté égal à 3.

Mais, il est difficile d'exhiber des arguments qui puissent prévoir cette coïncidence.

Dans le deuxième cas – maille à deux niveaux – la matrice de contrainte posséde 24 lignes et 14 colonnes et fait intervenir 14 variables.

Cette fois, la procédure `rangpm` donne le rang de cette matrice en 850 secondes CPU sur la même machine.

Intuitivement, cette matrice est de rang 11, puisque ce mécanisme peut se déplacer exactement comme le mécanisme précédent.

Ainsi, la procédure `rangpm` qui rend comme résultat 13 fournit cette fois une majoration stricte du rang.

4.5.2 deuxième approche : utilisation de bases standards

Dans cette section, on va présenter et commenter les résultats obtenus en utilisant *MACAULAY* pour traiter les exemples décrits au paragraphe (§4.2).

Les sessions complètes correspondantes d'utilisation du logiciel sont fournies en annexe (§C.4).

exemple1 : le mécanisme le plus simple

Dans cet exemple, on pose :

$$\begin{cases} X_1 = p_{1,1} = x_{M_{13}^t} \\ X_2 = p_{1,2} = y_{M_{13}^t} \\ Y_3 = \sin(\alpha_1) \\ Z_3 = \cos(\alpha_1) \\ Y_4 = \sin(\alpha_2) \\ Z_4 = \cos(\alpha_2) \\ Y_5 = \sin(\alpha_3) \\ Z_5 = \cos(\alpha_3) \end{cases}$$

La matrice de contrainte (*ex1*.iv) s'écrit alors :

$$\begin{pmatrix} 0 & 0 & \lambda_1 Y_3 & \lambda_2 Y_4 & \lambda_3 Y_5 \\ 0 & 0 & -\lambda_1 Z_3 & \lambda_2 Z_4 & -\lambda_3 Z_5 \end{pmatrix}$$

Dans un premier temps, on calcule une base standard de l'idéal engendré par les polynômes traduisant les équations de contrainte.

MACAULAY calcule sa codimension, c'est à dire la différence entre le nombre d'indéterminées et la dimension de la variété algébrique définie par ces polynômes.

On obtient 2, ce qui donne comme dimension 3 dans la mesure où *MACAULAY* considère aussi λ_1, λ_2 et λ_3 comme des indéterminées et utilise une indéterminée de plus pour l'homogénéisation.

Ici, il est clair que le degré de liberté du mécanisme est au moins égal à 3, ce qui permet de valider le résultat obtenu.

On fournit ensuite la matrice de contrainte à *MACAULAY* en lui précisant que ses coefficients appartiennent à l'anneau quotient :

$$\mathbb{Q}[X_1, X_2, Y_3, Y_4, Y_5, Z_3, Z_4, Z_5]/(\lambda_1 Y_3 + \lambda_2 Y_4 + \lambda_3 Y_5, \lambda_1 Z_3 + \lambda_2 Z_4 + \lambda_3 Z_5)$$

qui est un anneau régulier.

La résolution du sous-module engendré par les colonnes de la matrice de contrainte s'arrête à la matrice des premières syzygies et donne :

$$0 \hookrightarrow E^3 \xrightarrow{\Psi} E^m \xrightarrow{M} \mathrm{Im}M(\subset E^\nu) \longrightarrow 0$$

avec :

$$\Psi = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix}$$

On retrouve donc par le calcul que le rang de la matrice de contrainte est $5 - 3 = 2$ et donc que le degré de liberté du système est 3.

De plus, on peut obtenir un paramétrage (P_1, P_2, P_3) adapté de la variété des configurations admissibles en écrivant $\dot{p} = \Psi \dot{P}$ soit :

$$\left\{\begin{array}{rcl} \dot{P}_1 &=& \dot{x}_{M^t_{13}} \\ \dot{P}_2 &=& \dot{y}_{M^t_{13}} \\ \dot{P}_3 &=& \dot{\alpha}_1 \\ \dot{P}_3 &=& \dot{\alpha}_2 \\ \dot{P}_3 &=& \dot{\alpha}_3 \end{array}\right.$$

d'où l'on tire :

$$\left\{\begin{array}{l} \left\{\begin{array}{rcl} \dot{P}_1 &=& \dot{x}_1 \\ \dot{P}_2 &=& \dot{y}_2 \\ \dot{P}_3 &=& \dot{\alpha}_1 \end{array}\right. \\ \\ \left\{\begin{array}{rcl} \dot{\alpha}_2 &=& \dot{\alpha}_1 \\ \dot{\alpha}_3 &=& \dot{\alpha}_1 \end{array}\right. \end{array}\right. \quad \text{puis} \quad \left\{\begin{array}{l} \left\{\begin{array}{rcl} P_1 &=& x_{M^t_{13}} \\ P_2 &=& y_{M^t_{13}} \\ P_3 &=& \alpha_1 \end{array}\right. \\ \\ \left\{\begin{array}{rcl} \alpha_2 &=& \alpha_1 + \mathrm{c}^{\mathrm{st}} \\ \alpha_3 &=& \alpha_1 + \mathrm{c}^{\mathrm{st}} \end{array}\right. \end{array}\right.$$

On pourra déterminer les constantes en utilisant les équations de contrainte.

exemple2 : la bielle

En ce qui concerne la bielle, on fait calculer à *MACAULAY* la résolution du sous-module engendré par les colonnes de la matrice de contrainte (*ex2*.ii) en considérant ses coefficients comme appartenant :

i. à l'anneau :
$$\mathbb{Q}[X_1, Y_2, Y_3, Z_2, Z_3]$$

ii. à l'anneau quotient :
$$\mathbb{Q}[X_1, Y_2, Y_3, Z_2, Z_3]/(Y_2^2 + Z_2^2 - 1)$$

en posant $X_1 = x_3$, $Y_2 = \sin(\alpha_1)$, $Z_2 = \cos(\alpha_1)$, $Y_3 = \sin(\alpha_{12})$ et $z_3 = \cos(\alpha_{12})$.
Ces deux anneaux sont réguliers.
Dans les deux cas, la résolution du sous-module s'arrête après le calcul des premières syzygies et on obtient respectivement pour l'annulateur :

$$\Psi_{\mathrm{i}} = \begin{pmatrix} -\sin(\alpha_1)\sin(\alpha_{12})\lambda_2 + \cos(\alpha_1)\cos(\alpha_{12})\lambda_2 \\ \sin(\alpha_1)\sin(\alpha_{12})\lambda_2 - \cos(\alpha_1)\cos(\alpha_{12})\lambda_2 - \cos(\alpha_1)\lambda_1 \\ \sin(\alpha_1)^2\lambda_1\sin(\alpha_{12})\lambda_2 + \cos(\alpha_1)^2\lambda_1\sin(\alpha_{12})\lambda_2 \end{pmatrix}$$

$$\text{et} \qquad \Psi_{\mathrm{ii}} = \begin{pmatrix} -\sin(\alpha_1)\sin(\alpha_{12})\lambda_2 + \cos(\alpha_1)\cos(\alpha_{12})\lambda_2 \\ \sin(\alpha_1)\sin(\alpha_{12})\lambda_2 - \cos(\alpha_1)\cos(\alpha_{12})\lambda_2 - \cos(\alpha_1)\lambda_1 \\ \lambda_1\sin(\alpha_{12})\lambda_2 \end{pmatrix}$$

Remarque 4.17 Dans cet exemple, la relation trigonométrique $Y_2^2 + Z_2^2 = 1$ n'influence pas la nature de la résolution du sous-module et il est plus intéressant – au niveau du temps et du volume de calcul – d'effectuer le calcul sans en tenir compte puis de considérer la matrice Ψ obtenue modulo cette relation que d'en tenir compte dès le début.

Remarque 4.18 De la même manière, on n'a pas tenu compte ici des équations de contrainte dans la mesure où, sur cet exemple, il est équivalent de calculer le rang de $\Delta_{\mathcal{V}}$ en tout point non singulier de $\mathcal{V}_{\text{arb}}$ ou en tout point non singulier de $\mathcal{V}$.

Dans cet exemple, il n'est pas possible de "récupérer" simplement un paramétrage de $\mathcal{V}$ à partir de la matrice Ψ.

Le calcul de la dimension de $\mathcal{V}$ effectué directement à partir de la base standard des polynômes représentant les équations de liaison donne, ici, un résultat supérieur au résultat correct puisque l'on obtient 3 comme codimension et donc 2 comme dimension.

exemple3 : la salière

Ici le calcul de la dimension de $\mathcal{V}$ donne comme résultat : 1.

Dans la mesure, où l'on sait qu'il existe des mouvements possibles pour ce mécanisme, on peut donc conclure.

Il est par contre difficile, en toute rigueur, de calculer le rang de la matrice de contrainte puisque l'anneau quotient :

$$\mathbb{Q}[Y_1, Z_1, Y_2, Y_3, Z_2, Z_3]/(Z_2 - Z_3, Y_1Y_2, Z_1Y_2 - Y_3)$$

n'est pas régulier – il n'est en particulier pas intègre.

On peut néanmoins calculer la matrice des premières syzygies. *MACAULAY* donne la matrice suivante :

$$\begin{pmatrix} 0 & \sin(\alpha_1) & \sin(\alpha_2) - \cos\alpha_1)\sin(\alpha_3) & 0 & 0 \\ \sin(\alpha_2) & 0 & 0 & 0 & \sin(\alpha_3)\cos\alpha_3) \\ \sin(\alpha_3) & 0 & 0 & \sin(\alpha_2) - \cos\alpha_1)\sin(\alpha_3) & \sin(\alpha_2)\cos\alpha_3) \end{pmatrix}$$

Cette matrice peut s'interpréter en fonction des équations de contrainte :

De $\sin(\alpha_1)\sin(\alpha_2) = 0$, on déduit $\sin(\alpha_1) = 0$ ou $\sin(\alpha_2) = 0$,

- si $\sin(\alpha_1) = 0$ alors $\cos(\alpha_1) = 1$ et $\alpha_2 = \alpha_3$, seules la première et la dernière colonne de la matrice sont non nulles et elles sont égales,
- sinon $\sin(\alpha_2) = 0$, et donc $\sin(\alpha_3) = 0$, et seule la deuxième colonne de la matrice est non nulle.

Dans les deux cas, on trouve bien alors que la matrice de contrainte est de rang 2 puisqu'il n'existe qu'une seule relation non triviale entre ses colonnes.

exemple4 : le joint homocinétique

Le calcul direct de la dimension de $\mathcal{V}$ donne, ici encore, un résultat correspondant à l'intuition mécanique c'est à dire 3.

On retrouve ce résultat en calculant le rang de la matrice de contrainte sans tenir compte ni des équations de contrainte, ni des relations trigonométriques.

Dans ce cas, *MACAULAY* fournit une matrice qui, modulo $Y_2^2 + Z_2^2 = 1$ et la dernière équation de contrainte vaut – si $\cos(\beta_2) \neq 0$:

$$\begin{pmatrix} \cos(\gamma_2) \\ \cos(\beta_2)\sin(\gamma_2) \\ -\sin(\beta_2)\cos(\gamma_2) \\ \cos(\gamma_2) \\ 0 \end{pmatrix}$$

Il n'est pas facile, à partir de cette matrice, de "récupérer" un paramétrage de $\mathcal{V}$. Néanmoins, on peut y lire que le paramètre z est constant, ce qui n'était, mécaniquement pas complètement évident.

exemple5 : les mailles d'antennes

Ces deux exemples n'ont pas pu être traités avec *MACAULAY*, la taille des matrices de contrainte entraînant des temps et des volumes de calculs trop importants.

Chapitre 5

conclusion

Au terme de ce mémoire, et pour mieux tirer les conclusions des travaux qui y sont présentés, il est bon de revenir brièvement sur les motivations qui ont conduit à réaliser cette étude ainsi que sur son déroulement chronologique.

C'est à la demande d'une équipe d'ingénieurs de la division des Systèmes Stratégiques et Spatiaux de l'*AEROSPATIALE – Cannes* qu'ont débuté ces recherches.

Cette équipe s'occupait de la réalisation en *MAPLE* d'un prototype de générateur formel des équations de la mécanique de systèmes de corps polyarticulés.

Ce prototype, utilisant un formalisme proche du formalisme de Lagrange, engendraient les équations différentielles du mouvement ainsi que les équations algébriques de contrainte dues à la présence des liaisons dans les systèmes mécaniques étudiées.

A première vue, l'élimination des multiplicateurs de Lagrange dans les équations du mouvement de systèmes mécaniques à architecture bouclée semblait se ramener simplement à la détermination algébrique du noyau d'une matrice.

L'étude commandée alors avait essentiellement pour thème de résoudre le problème suivant :

> *Etant donnée une matrice M à coefficients dans un anneau de polynôme, calculer une matrice N de la "bonne" dimension telle que $M.N = 0$.*

ce problème devant être abordé à l'aide des outils algébriques provenant de la théorie des bases de Gröbner.

Il était ainsi principalement prévu de réaliser :

- une implémentation en *MAPLE* d'un algorithme efficace de calcul de bases standards,
- l'interfaçage entre cette implémentation et le prototype en développement,
- l'expérimentation de cet algorithme sur des exemples simples provenant de l'étude de systèmes mécaniques concrets.

Comme on a pu le voir au chapitre §4, les résultats obtenus par l'utilisation "brutale" de méthodes purement algébriques ont un intérêt plutôt théorique que pratique dans la mesure où l'expérimentation de ces méthodes sur des exemples de systèmes mécaniques relativement simples débouche déjà sur des temps et des volumes de calculs très importants.

Il est ainsi très vite apparu la nécessité de prendre en compte la spécificité "mécanique" des objets mathématiques auquels on s'interessait.

C'est ce qui a modifié l'orientation des recherches entreprises et a débouché – in fine – sur le travail qui fait l'objet de ce mémoire.

Le premier chapitre n'avait pas d'autre ambition que de synthétiser les connaissances mécaniques nécessaires à l'étude du comportement dynamique des systèmes de corps mécaniques rigides reliés entre eux. Il aura atteint son objectif et trouvé sa raison d'être s'il a :

- d'une part, permis d'établir clairement une correspondance précise entre les notions et le vocabulaire de la modélisation du mouvement des objets mécaniques et les concepts et le vocabulaire mathématiques qui s'y rattachent,
- d'autre part, bien mis en évidence la différence intrinsèque entre les systèmes à architecture arborescente et les systèmes à architecture bouclée, et l'influence de cette différence sur les méthodes de mise en équation de leurs mouvements.

Son influence sur la suite de l'étude est loin d'être négligeable, puisque l'acquisition de connaissances qu'a demandé sa rédaction a conduit à remettre en question le formalisme de mise en équation préconisé par l'équipe de l'*AEROSPATIALE* et à effectuer une étude comparative de différents formalismes existants, dans l'optique de leur utilisation pour la génération des équations du mouvement automatisée à l'aide d'un logiciel de calcul formel.

Le deuxième chapitre, dans lequel est réalisée cette étude comparative, a tout d'abord mis en valeur l'interêt de l'emploi de logiciels de calcul formel pour la modélisation en fournissant des indications concrètes sur la taille des calculs nécessaires, et a aussi fait apparaître les différents problèmes qui résultent de cet emploi. Il a notamment mis en évidence :

- les inconvénients des différents formalismes étudiés, qu'ils proviennent de la mécanique "classique" ou "analytique", et la nécessité de développer un formalisme spécifique à l'emploi de logiciels de calcul formel,
- les lacunes des systèmes de calcul formel disponibles, surtout en ce qui concerne la manipulation et la simplification des expressions et notamment des expressions vectorielles et matricielles,
- la nécessité de pouvoir contrôler le processus de génération des équations au cours de leur développement pour obtenir :
  - soit des expressions bien conditionnées pour effectuer des simulations numériques efficaces,
  - soit des expressions conservant un sens "mécanique" significatif pour être interpréter par l'utilisateur.

Dans le troisième chapitre, on s'est attaché à proposer des éléments de solution aux problèmes dégagés précédemment.

Le nombre de générateurs formels des équations de la dynamique actuellement en développement dans différents centres de recherche universitaires ou industriels a conduit à ne pas développer un prototype de plus mais à se focaliser sur des problèmes plus théoriques tels que :

- l'architecture informatique de tels logiciels et leur nécessaire modularité,
- la création d'un langage typé de description des mécanismes qui, en prenant en compte toutes les caractéristiques physiques des systèmes mécaniques à modéliser puisse servir d'interface d'entrée de données à la plupart de ces logiciels,
- l'implémentation dans divers systèmes de calcul formel de bibliothèques de manipulation d'expression matricielles et vectorielles.

Les travaux présentés dans ce chapitre devraient constituer les premiers éléments d'une "boîte à outil logicielle" formelle pour la modélisation en mécanique qui permettrait de répondre aux besoins de l'ingénieur intéressé par la conception et la modélisation de mécanismes.

Enfin le dernier chapitre regroupe les diverses tentatives effectuées pour mener à bien le traitement des mécanismes dont l'architecture est bouclée.

C'est dans ce domaine qu'il y a le plus de perspectives théoriques, puisque ce chapitre ramène le processus de modélisation dynamique à l'étude de la dimension réelle d'une variété algébrique, étude qui est actuellement l'objet d'une importante et fertiel activité de recherche dans la communauté mathématique internationale.

Annexe A

satellite : code de calcul *MAPLE*

A.1 description du système (fichier `intro`)

```
    |\^/|      MAPLE V
._|\|   |/|_.  Copyright (c) 1981-1990 by the University of Waterloo.
 \  MAPLE  /   All rights reserved.  MAPLE is a registered trademark of
 <____ ____>   Waterloo Maple Software.
      |        Type ? for help.

> with(linalg):
Warning: new definition for   norm
Warning: new definition for   trace
#####
> i1:=matrix([[i1X,0,0],[0,i1Y,0],[0,0,i1Z]]):
#####
> g1m12:=vector([0,0,z12]):
#####
> m21m34:=vector([0,q8(t),0]):
#####
> m43g4:=vector([0,0,z34]):
#####
> c1:=vector([c1X,c1Y,c1Z]):
> f1:=vector([f1X,f1Y,f1Z]):
#####
> f4:=vector([f4X,f4Y,f4Z]):
#####
> Rotq1:=matrix([[1,0,0],[0,cos(q1(t)),-sin(q1(t))],[0,sin(q1(t)),cos(q1(t))]]):
> Rotq2:=matrix([[cos(q2(t)),0,sin(q2(t))],[0,1,0],[-sin(q2(t)),0,cos(q2(t))]]):
> Rotq3:=matrix([[cos(q3(t)),-sin(q3(t)),0],[sin(q3(t)),cos(q3(t)),0],[0,0,1]]):
#####
> Rot1:=multiply(Rotq1,Rotq2,Rotq3):
#####
> Rot12:=subs(q1(t)=q7(t),op(Rotq1)):
#####
> Rot34:=subs(q2(t)=q9(t),op(Rotq2)):
#####
> v1:=vector([vX(t),vY(t),vZ(t)]):
> om1:=vector([omX(t),omY(t),omZ(t)]):
#####
> Rot14:=multiply(Rot12,Rot34):
> Rot1p:=map(diff,Rot1,t):
> Rot14p:=map(diff,Rot14,t):
#####

> tmp10:=multiply(Rot1p,transpose(Rot1)):
> tmp11:=map(simplify,tmp10,trig):
> tmp12:=vector([tmp11[3,2],tmp11[1,3],tmp11[2,1]]):
> tmp13:=multiply(transpose(Rot1),tmp12):
> tmp14:=map(simplify,tmp13,trig):

> tmp15:=
>   subs({diff(q1(t),t)=qp1,diff(q2(t),t)=qp2,diff(q3(t),t)=qp3},op(tmp14)):
```

```
> tmp16:=
>   solve({tmp15[1]=om1[1],tmp15[2]=om1[2],tmp15[3]=om1[3]},
>         {qp1,qp2,qp3}):
> tmp17:=map(simplify,tmp16,trig):
> tmp18:=
>   subs({q1(t)=q1,q2(t)=q2,q3(t)=q3,omX(t)=omX,omY(t)=omY,omZ(t)=omZ},tmp17):
#####
> eq1:=map(collect,tmp18,sin(q2));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                   (- sin(q3) omY + omX cos(q3)) sin(q2)
    eq1 := {qp3 = - ------------------------------------- + omZ,
                                  cos(q2)

                                               - sin(q3) omY + omX cos(q3)
        qp2 = sin(q3) omX + omY cos(q3), qp1 = ---------------------------}
                                                         cos(q2)

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp20:=multiply(Rot1,v1):
> tmp21:=
>   map(collect,[qp4=tmp20[1],qp5=tmp20[2],qp6=tmp20[3]],
>       {vX(t),vY(t),vZ(t)}):
#####
> eq2:=
>   convert(subs({q1(t)=q1,q2(t)=q2,q3(t)=q3,vX(t)=vX,vY(t)=vY,vZ(t)=vZ},
>                tmp21),set);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

eq2 := {qp4 = cos(q2) cos(q3) vX - cos(q2) sin(q3) vY + sin(q2) vZ,

 qp5 = (sin(q1) sin(q2) cos(q3) + cos(q1) sin(q3)) vX

      + (- sin(q1) sin(q2) sin(q3) + cos(q1) cos(q3)) vY - sin(q1) cos(q2) vZ,

 qp6 = (- cos(q1) sin(q2) cos(q3) + sin(q1) sin(q3)) vX

      + (cos(q1) sin(q2) sin(q3) + sin(q1) cos(q3)) vY + cos(q1) cos(q2) vZ}

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp25:={om1[1]=tmp15[1],om1[2]=tmp15[2],om1[3]=tmp15[3]}:
#####
> eq3:=subs({q1(t)=q1,q2(t)=q2,q3(t)=q3,vX(t)=vX,vY(t)=vY,vZ(t)=vZ},tmp25);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

               eq3 := {omZ(t) = sin(q2) qp1 + qp3,

                   omX(t) = cos(q2) cos(q3) qp1 + sin(q3) qp2,

                   omY(t) = - cos(q2) sin(q3) qp1 + cos(q3) qp2}

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp30:=solve({qp4=tmp20[1],qp5=tmp20[2],qp6=tmp20[3]},{vX(t),vY(t),vZ(t)}):
> tmp31:=map(simplify,tmp30,trig):
> tmp32:=map(collect,tmp31,{qp4,qp5,qp6},distributed):
#####

> eq4:=subs({q1(t)=q1,q2(t)=q2,q3(t)=q3,vX(t)=vX,vY(t)=vY,vZ(t)=vZ},tmp32);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

eq4 := {

   vY = (- sin(q1) sin(q2) sin(q3) + cos(q1) cos(q3)) qp5 - sin(q3) cos(q2) qp4
```

```
        + (cos(q1) sin(q2) sin(q3) + sin(q1) cos(q3)) qp6,
   vZ = - cos(q2) sin(q1) qp5 + cos(q2) qp6 cos(q1) + qp4 sin(q2),
   vX = (sin(q1) sin(q2) cos(q3) + cos(q1) sin(q3)) qp5 + cos(q3) cos(q2) qp4
        + (- cos(q1) sin(q2) cos(q3) + sin(q1) sin(q3)) qp6                 }
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
bytes used=3024412, alloc=1244956, time=13.383
```

A.2 loi fondamentale de la dynamique

théorème du centre d'inertie – calculs traditionnels (fichier `newton1alamain`)

```
> read intro:
#####
> om2:=vector([diff(q7(t),t),0,0]):
#####
> om4:=vector([0,diff(q9(t),t),0]):
#####
> tmp35:=map(diff,v1,t):
> gamma1:=add(tmp35,crossprod(om1,v1)):
####
> g1g4:=add(g1m12,add(multiply(Rot12,m21m34),multiply(Rot12,Rot34,m43g4))):
####
> tmp36:=map(diff,om1,t):
> tmp37:=
>   add(gamma1,add(crossprod(tmp36,g1g4),crossprod(om1,crossprod(om1,g1g4)))):
> tmp38:=
>   map(simplify,
>       add(crossprod(om2,multiply(Rot12,m21m34)),
>       add(multiply(Rot12,map(diff,m21m34,t)),
>           crossprod(add(om2,multiply(Rot12,om4)),
>                     multiply(Rot12,Rot34,m43g4)))),
>       trig):
> tmp39:=
>   add(crossprod(map(diff,om2,t),multiply(Rot12,m21m34)),
>   add(crossprod(om2,crossprod(om2,multiply(Rot12,m21m34))),
>   add(scalarmul(crossprod(om2,multiply(Rot12,map(diff,m21m34,t))),2),
>   add(multiply(Rot12,map(diff,map(diff,m21m34,t),t)),
>   add(crossprod(add(map(diff,om2,t),add(crossprod(om2,multiply(Rot12,om4)),
>                                         multiply(Rot12,map(diff,om4,t)))),
>                 multiply(Rot12,Rot34,m43g4)),
>       crossprod(add(om2,multiply(Rot12,om4)),
>                 crossprod(add(om2,multiply(Rot12,om4)),
>                           multiply(Rot12,Rot34,m43g4)))))))):
> gamma4:=add(tmp37,add(scalarmul(crossprod(om1,tmp38),2),tmp39)):#
#####

> Qacc:=
>   subs(varslect,add(scalarmul(gamma1,mu1),scalarmul(gamma4,mu4)));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

Qacc := [

mu1 (vXp + omY vZ - omZ vY) + mu4 (vXp + omY vZ - omZ vY + omYp %4 - omZp %3

     + omY (omX %3 - omY sin(q9) z34) - omZ (omZ sin(q9) z34 - omX %4)

     + 2 omY (qp7 cos(q7) q8 + sin(q7) qp8 - %6 - %5)

     - 2 omZ (- qp7 sin(q7) q8 + cos(q7) qp8 + %2 - %1)

     + (- qp7 sin(q7) qp9 + cos(q7) qpp9) cos(q7) cos(q9) z34

     + (qp7 cos(q7) qp9 + sin(q7) qpp9) sin(q7) cos(q9) z34

     + cos(q7) qp9 (- %6 - %5) - sin(q7) qp9 (%2 - %1)),

mu1 (vYp + omZ vX - omX vZ) + mu4 (vYp + omZ vX - omX vZ + omZp sin(q9) z34

     - omXp %4 + omZ (omY %4 - omZ %3) - omX (omX %3 - omY sin(q9) z34)

     + 2 omZ qp9 cos(q9) z34 - 2 omX (qp7 cos(q7) q8 + sin(q7) qp8 - %6 - %5)

                              2
     - qpp7 sin(q7) q8 - qp7  cos(q7) q8 - 2 qp7 sin(q7) qp8 + cos(q7) qpp8
```

```
     + (qp7 cos(q7) qp9 + sin(q7) qpp9) sin(q9) z34 - qpp7 cos(q7) cos(q9) z34

                            2                              2
     + sin(q7) qp9 (cos(q7)  qp9 cos(q9) z34 + sin(q7)  qp9 cos(q9) z34)

     - qp7 (- %6 - %5)),

mu1 (vZp + omX vY - omY vX) + mu4 (vZp + omX vY - omY vX + omXp %3

     - omYp sin(q9) z34 + omX (omZ sin(q9) z34 - omX %4)

     - omY (omY %4 - omZ %3) + 2 omX (- qp7 sin(q7) q8 + cos(q7) qp8 + %2 - %1)

                                                    2
     - 2 omY qp9 cos(q9) z34 + qpp7 cos(q7) q8 - qp7  sin(q7) q8

     + 2 qp7 cos(q7) qp8 + sin(q7) qpp8 - qpp7 sin(q7) cos(q9) z34

     - (- qp7 sin(q7) qp9 + cos(q7) qpp9) sin(q9) z34 + qp7 (%2 - %1)

                            2                              2
     - cos(q7) qp9 (cos(q7)  qp9 cos(q9) z34 + sin(q7)  qp9 cos(q9) z34))

 ]

%1 :=                       qp7 cos(q7) cos(q9) z34

%2 :=                       sin(q7) qp9 sin(q9) z34

%3 :=                  cos(q7) q8 - sin(q7) cos(q9) z34

%4 :=              z12 + sin(q7) q8 + cos(q7) cos(q9) z34

%5 :=                       cos(q7) qp9 sin(q9) z34

%6 :=                       qp7 sin(q7) cos(q9) z34

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

bytes used=3505512, alloc=1244956, time=15.850
```

théorème du centre d'inertie – calculs directs (fichier `newton1`)

```
> read intro:
#####
> Og1:=vector([q4(t),q5(t),q6(t)]):
####
> g1g4:=add(g1m12,add(multiply(Rot12,m21m34),multiply(Rot14,m43g4)));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

    g1g4 := [ sin(q9(t)) z34, cos(q7(t)) q8(t) - sin(q7(t)) cos(q9(t)) z34,

        z12 + sin(q7(t)) q8(t) + cos(q7(t)) cos(q9(t)) z34 ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> Og4:=add(Og1,multiply(Rot1,g1g4)):
####
> muOG:=add(scalarmul(Og1,mu1),scalarmul(Og4,mu4));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

muOG := [

   mu1 q4(t) + mu4 (q4(t) + cos(q2(t)) cos(q3(t)) sin(q9(t)) z34

        - cos(q2(t)) sin(q3(t)) cos(q7(t)) q8(t)

        + cos(q2(t)) sin(q3(t)) sin(q7(t)) cos(q9(t)) z34 + sin(q2(t)) z12

        + sin(q2(t)) sin(q7(t)) q8(t) + sin(q2(t)) cos(q7(t)) cos(q9(t)) z34),

   mu1 q5(t) + mu4 (q5(t) + sin(q9(t)) z34 sin(q1(t)) sin(q2(t)) cos(q3(t))

        + sin(q9(t)) z34 cos(q1(t)) sin(q3(t))

        - sin(q1(t)) sin(q2(t)) sin(q3(t)) cos(q7(t)) q8(t)

        + sin(q1(t)) sin(q2(t)) sin(q3(t)) sin(q7(t)) cos(q9(t)) z34

        + cos(q1(t)) cos(q3(t)) cos(q7(t)) q8(t)

        - cos(q1(t)) cos(q3(t)) sin(q7(t)) cos(q9(t)) z34

        - sin(q1(t)) cos(q2(t)) z12 - sin(q1(t)) cos(q2(t)) sin(q7(t)) q8(t)

        - sin(q1(t)) cos(q2(t)) cos(q7(t)) cos(q9(t)) z34),

   mu1 q6(t) + mu4 (q6(t) - sin(q9(t)) z34 cos(q1(t)) sin(q2(t)) cos(q3(t))

        + sin(q9(t)) z34 sin(q1(t)) sin(q3(t))

        + cos(q1(t)) sin(q2(t)) sin(q3(t)) cos(q7(t)) q8(t)

        - cos(q1(t)) sin(q2(t)) sin(q3(t)) sin(q7(t)) cos(q9(t)) z34

        + sin(q1(t)) cos(q3(t)) cos(q7(t)) q8(t)

        - sin(q1(t)) cos(q3(t)) sin(q7(t)) cos(q9(t)) z34

        + cos(q1(t)) cos(q2(t)) z12 + cos(q1(t)) cos(q2(t)) sin(q7(t)) q8(t)

        + cos(q1(t)) cos(q2(t)) cos(q7(t)) cos(q9(t)) z34)                  ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp35:=map(diff,muOG,t):
> tmp36:=
>    subs({diff(q1(t),t)=qp1,diff(q2(t),t)=qp2,diff(q3(t),t)=qp3,
>          diff(q4(t),t)=qp4,diff(q5(t),t)=qp5,diff(q6(t),t)=qp6},op(tmp35)):
```

```
> tmp37:=map(simplify,subs(tmp17 union convert(tmp21,set),op(tmp36)),trig):
> tmp38:=map(diff,tmp37,t):
> tmp39:=
>   subs({diff(q1(t),t)=qp1,diff(q2(t),t)=qp2,diff(q3(t),t)=qp3,
>         diff(q4(t),t)=qp4,diff(q5(t),t)=qp5,diff(q6(t),t)=qp6},op(tmp38)):
> tmp40:=map(simplify,subs(tmp17 union convert(tmp21,set),op(tmp39)),trig):
> tmp41:=map(simplify,multiply(transpose(Rot1),tmp40)):
> tmp42:=subs(varslect,op(tmp41)):
> tmp43:=add(f1,f4):
#####
> eq8a:=
>   tmp43[1]=rlms(clct(tmp42[1],[qpp7,qpp8,qpp9],
                               [omXp,omYp,omZp,vXp,vYp,vZp,qp7,qp8,qp9],
>                              [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4]));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

   eq8a := f1X + f4X = - (mu1 + mu4) vY omZ + (mu1 + mu4) vZ omY

        + (cos(q7) q8 - sin(q7) cos(q9) z34) mu4 omY omX

        + (z12 + cos(q7) cos(q9) z34 + sin(q7) q8) mu4 omZ omX

                                 2                      2
        - mu4 sin(q9) z34 omY  - mu4 sin(q9) z34 omZ  + (

       (2 sin(q7) q8 + 2 cos(q7) cos(q9) z34) mu4 omZ

        + (2 cos(q7) q8 - 2 sin(q7) cos(q9) z34) mu4 omY) qp7

        + (z12 + cos(q7) cos(q9) z34 + sin(q7) q8) mu4 omYp

        + (sin(q7) cos(q9) z34 - cos(q7) q8) mu4 omZp

        + (2 mu4 sin(q7) omY - 2 mu4 cos(q7) omZ) qp8

        - (2 mu4 sin(q7) sin(q9) z34 omZ + 2 mu4 cos(q7) sin(q9) z34 omY) qp9

                                           2
        + (mu1 + mu4) vXp - mu4 sin(q9) qp9  z34 + mu4 cos(q9) qpp9 z34

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> eq8b:=
>   tmp43[2]=rlms(clct(tmp42[2],[qpp7,qpp8,qpp9],
>                              [omXp,omYp,omZp,vXp,vYp,vZp,qp7,qp8,qp9],
>                              [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4]));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

eq8b := f1Y + f4Y = (mu1 + mu4) vX omZ - (mu1 + mu4) vZ omX

     + (z12 + cos(q7) cos(q9) z34 + sin(q7) q8) mu4 omZ omY

                                 2
     + (%1 - cos(q7) q8) mu4 omX  + mu4 sin(q9) z34 omY omX

                                 2
     + (%1 - cos(q7) q8) mu4 omZ  - (2 cos(q7) q8 - 2 %1) mu4 omX qp7

     - (z12 + cos(q7) cos(q9) z34 + sin(q7) q8) mu4 omXp + mu4 sin(q9) z34 omZp

     - 2 mu4 sin(q7) qp8 omX

     + (2 mu4 cos(q9) z34 omZ + 2 mu4 cos(q7) sin(q9) z34 omX) qp9

     + (mu1 + mu4) vYp - 2 mu4 sin(q7) qp7 qp8

                                                                          2
     + 2 mu4 cos(q7) qp7 sin(q9) qp9 z34 + (%1 - cos(q7) q8) mu4 qp7
```

```
                                  2
     + mu4 sin(q7) cos(q9) qp9  z34

     - (cos(q7) cos(q9) z34 + sin(q7) q8) mu4 qpp7 + mu4 cos(q7) qpp8

     + mu4 sin(q7) sin(q9) qpp9 z34

%1 :=                         sin(q7) cos(q9) z34

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> eq8c:=
>   tmp43[3]=rlms(clct(tmp42[3],[qpp7,qpp8,qpp9],
>                               [omXp,omYp,omZp,vXp,vYp,vZp,qp7,qp8,qp9],
>                               [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4]));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

  eq8c := f1Z + f4Z = - (mu1 + mu4) vX omY + (mu1 + mu4) vY omX

       + (cos(q7) q8 - sin(q7) cos(q9) z34) mu4 omZ omY

                                        2
       - (z12 + %1 + sin(q7) q8) mu4 omX  + mu4 sin(q9) z34 omZ omX

                                       2
       - (z12 + %1 + sin(q7) q8) mu4 omY  - (2 sin(q7) q8 + 2 %1) mu4 omX qp7

       + (cos(q7) q8 - sin(q7) cos(q9) z34) mu4 omXp - mu4 sin(q9) z34 omYp

       + 2 mu4 cos(q7) qp8 omX

       + (2 mu4 sin(q7) sin(q9) z34 omX - 2 mu4 cos(q9) z34 omY) qp9

       + (mu1 + mu4) vZp + 2 mu4 cos(q7) qp7 qp8

                                                                    2
       + 2 mu4 sin(q7) qp7 sin(q9) qp9 z34 - (%1 + sin(q7) q8) mu4 qp7

                                   2
       - mu4 cos(q7) cos(q9) qp9  z34

       + (cos(q7) q8 - sin(q7) cos(q9) z34) mu4 qpp7 + mu4 sin(q7) qpp8

       - mu4 cos(q7) sin(q9) qpp9 z34

%1 :=                         cos(q7) cos(q9) z34

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

bytes used=39681096, alloc=3603820, time=230.250
```

théorème du moment cinétique (fichier newton2)

```
> read intro:
#####

> g1g4:=add(g1m12,add(multiply(Rot12,m21m34),multiply(Rot14,m43g4))):
#####
> tmp40:=map(diff,g1g4,t):
> tmp41:=crossprod(om1,g1g4):
> v4:=add(v1,add(tmp40,tmp41)):
#####
> MV:=add(scalarmul(v1,mu1),scalarmul(v4,mu4)):
#####
> tmp45:=multiply(Rot1,MV):
> tmp46:=map(simplify,tmp45,trig):
> tmp47:=map(diff,tmp46,t):
> tmp48:=map(simplify,tmp47,trig):
> tmp49:=
>   subs({diff(q1(t),t)=qp1,diff(q2(t),t)=qp2,diff(q3(t),t)=qp3},op(tmp48)):
> tmp50:=subs(tmp17,op(tmp49)):
> tmp51:=multiply(transpose(Rot1),tmp50):
> tmp52:=map(simplify,tmp51,trig):
> tmp53:=
>   subs(varslect,op(tmp52)):
> tmp54:=add(f1,f4):
#####

> eq8a:=
>   tmp54[1]=rlms(clct(tmp53[1],[qpp7,qpp8,qpp9],
>                              [omXp,omYp,omZp,vXp,vYp,vZp,qp7,qp8,qp9],
>                              [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4])):
#####
> eq8b:=
>   tmp54[2]=rlms(clct(tmp53[2],[qpp7,qpp8,qpp9],
>                              [omXp,omYp,omZp,vXp,vYp,vZp,qp7,qp8,qp9],
>                              [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4])):
#####
> eq8c:=
>   tmp54[3]=rlms(clct(tmp53[3],[qpp7,qpp8,qpp9],
>                              [omXp,omYp,omZp,vXp,vYp,vZp,qp7,qp8,qp9],
>                              [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4])):
#####

> tmp55:=scalarmul(g1g4,mu):
> tmp56:=add(scalarmul(v1,-1),v4):
> tmp57:=crossprod(tmp55,tmp56):
> tmp58:=add(multiply(i1,om1),tmp57):
#####
> Momc:=subs(varslect,op(tmp58));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

 Momc := [

     i1X omX + mu %1 (cos(q7) qp7 q8 + sin(q7) qp8 - sin(q7) qp7 cos(q9) z34

          - cos(q7) sin(q9) qp9 z34 + omX %1 - omY sin(q9) z34) - mu %2 (

         - sin(q7) qp7 q8 + cos(q7) qp8 - cos(q7) qp7 cos(q9) z34

          + sin(q7) sin(q9) qp9 z34 + omZ sin(q9) z34 - omX %2),

     i1Y omY + mu %2 (cos(q9) qp9 z34 + omY %2 - omZ %1) - mu sin(q9) z34 (

         cos(q7) qp7 q8 + sin(q7) qp8 - sin(q7) qp7 cos(q9) z34

          - cos(q7) sin(q9) qp9 z34 + omX %1 - omY sin(q9) z34),

     i1Z omZ + mu sin(q9) z34 (- sin(q7) qp7 q8 + cos(q7) qp8
```

```
          - cos(q7) qp7 cos(q9) z34 + sin(q7) sin(q9) qp9 z34 + omZ sin(q9) z34

          - omX %2) - mu %1 (cos(q9) qp9 z34 + omY %2 - omZ %1)

      ]

%1 :=                 cos(q7) q8 - sin(q7) cos(q9) z34

%2 :=              z12 + sin(q7) q8 + cos(q7) cos(q9) z34
```

```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

```
> tmp59:=multiply(Rot1,tmp58):
> tmp60:=map(simplify,tmp59,trig):
> tmp61:=map(diff,tmp60,t):
> tmp62:=map(simplify,tmp61,trig):
> tmp63:=
>   subs({diff(q1(t),t)=qp1,diff(q2(t),t)=qp2,diff(q3(t),t)=qp3},op(tmp62)):
> tmp64:=subs(tmp17,op(tmp63)):
> tmp65:=multiply(transpose(Rot1),tmp64):
> tmp66:=map(simplify,tmp65,trig):
> tmp67:=add(scalarmul(f1,-1/mu1),scalarmul(f4,1/mu4)):
> tmp68:=crossprod(g1g4,tmp67):
> tmp69:=add(c1,tmp68):
#####

> Fext:=subs(varslect,op(tmp69));
```

```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

```
                                                        /  f1Z   f4Z\
   Fext := [ c1X + (cos(q7) q8 - sin(q7) cos(q9) z34) |- --- + ---|
                                                        \  mu1   mu4/

                                                            /  f1Y   f4Y\
                 - (z12 + sin(q7) q8 + cos(q7) cos(q9) z34) |- --- + ---|,
                                                            \  mu1   mu4/

                                                        /  f1X   f4X\
       c1Y + (z12 + sin(q7) q8 + cos(q7) cos(q9) z34) |- --- + ---|
                                                        \  mu1   mu4/

                           /  f1Z   f4Z\
            - sin(q9) z34 |- --- + ---|,
                           \  mu1   mu4/

                          /  f1Y   f4Y\
       c1Z + sin(q9) z34 |- --- + ---|
                          \  mu1   mu4/

                                                   /  f1X   f4X\
            - (cos(q7) q8 - sin(q7) cos(q9) z34) |- --- + ---| ]
                                                   \  mu1   mu4/
```

```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

```
> tmp70:=
>   subs(varslect,op(tmp69)):
> tmp71:=
>   subs(varslect,op(tmp66)):
#####

> eq9a:=
>   rlms(clct(tmp70[1]=tmp71[1],[qpp7,qpp8,qpp9],
>                               [omXp,omYp,omZp,vXp,vYp,vZp,qp7,qp8,qp9],
>                               [omX,omY,omZ],[vX,vY,vZ],[mu]));
```

```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

```
                                                    /f1Z   f4Z\
eq9a := c1X - (cos(q7) q8 - sin(q7) cos(q9) z34) |--- - ---|
                                                    \mu1   mu4/

                                                /f1Y   f4Y\                      2
 + (z12 + sin(q7) q8 + cos(q7) cos(q9) z34) |--- - ---| = - (i1Y - i1Z + (z12
                                                \mu1   mu4/

          2    2     2
 - cos(q9)  z34  + q8  + 4 cos(q7) q8 sin(q7) cos(q9) z34

                                        2    2        2           2   2
 + 2 z12 cos(q7) cos(q9) z34 + 2 cos(q9)  z34  cos(q7)  - 2 cos(q7)  q8

 + 2 z12 sin(q7) q8) mu) omZ omY

                       2
 - (sin(q7) cos(q9) z34  sin(q9) - cos(q7) q8 sin(q9) z34) mu omZ omX -

                   2
(cos(q7) cos(q9) z34  sin(q9) + z12 sin(q9) z34 + sin(q7) q8 sin(q9) z34) mu

                            2    2
omY omX - (- cos(q7) cos(q9)  z34  sin(q7) - q8 cos(q9) z34 - %1

           2                                2                                 2
 + 2 cos(q7)  cos(q9) z34 q8 + sin(q7) q8  cos(q7) + z12 cos(q7) q8) mu omY  -

                                                          2
(- z12 cos(q7) q8 + %1 + q8 cos(q9) z34 - sin(q7) q8  cos(q7)

           2                                    2    2                2
 - 2 cos(q7)  cos(q9) z34 q8 + cos(q7) cos(q9)  z34  sin(q7)) mu omZ

 + (2 z12 cos(q7) q8 - 2 %1) mu omX qp7 + (i1X +

        2    2                                                    2      2
(cos(q9)  z34  + 2 z12 cos(q7) cos(q9) z34 + 2 z12 sin(q7) q8 + q8  + z12 ) mu)

                             2
 omXp + (sin(q7) cos(q9) z34  sin(q9) - cos(q7) q8 sin(q9) z34) mu omYp -

                   2
(cos(q7) cos(q9) z34  sin(q9) + z12 sin(q9) z34 + sin(q7) q8 sin(q9) z34) mu

omZp + (2 z12 sin(q7) + 2 q8) mu omX qp8 - (

                                      2    2
(2 z12 cos(q9) z34 + 2 cos(q7) cos(q9)  z34  + 2 sin(q7) q8 cos(q9) z34) mu omZ

                                                2    2
 + (2 cos(q7) q8 cos(q9) z34 - 2 sin(q7) cos(q9)  z34 ) mu omY

               2
 + (2 cos(q9) z34  sin(q9) + 2 z12 cos(q7) sin(q9) z34) mu omX) qp9

                                                                    2
 + (2 z12 sin(q7) + 2 q8) mu qp8 qp7 - (%1 + q8 cos(q9) z34) mu qp9

                                 2
 - (%1 - z12 cos(q7) q8) mu qp7

               2
 - (2 cos(q9) z34  sin(q9) + 2 z12 cos(q7) sin(q9) z34) mu qp9 qp7

                                                      2    2     2
 + (z12 sin(q7) q8 + z12 cos(q7) cos(q9) z34 + cos(q9)  z34  + q8 ) mu qpp7

 - (z12 cos(q7) + cos(q9) z34) mu qpp8
```

```
 - (q8 sin(q9) z34 + z12 sin(q7) sin(q9) z34) mu qpp9

%1 :=                     z12 sin(q7) cos(q9) z34

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> eq9b:=
>   rlms(clct(tmp70[2]=tmp71[2],[qpp7,qpp8,qpp9],
>                              [omXp,omYp,omZp,vXp,vYp,vZp,qp7,qp8,qp9],
>                              [omX,omY,omZ],[vX,vY,vZ],[mu]));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                                                      /f1X   f4X\
eq9b := c1Y - (z12 + sin(q7) q8 + cos(q7) cos(q9) z34) |--- - ---|
                                                      \mu1   mu4/

                 /f1Z   f4Z\
   + sin(q9) z34 |--- - ---| =
                 \mu1   mu4/

                      2
  (sin(q7) cos(q9) z34  sin(q9) - cos(q7) q8 sin(q9) z34) mu omZ omY

                                    2                        2    2
   + (%1 + z12 sin(q9) z34 + %2) mu omX  + (i1X - i1Z + (cos(q9)  z34

          2   2                        2      2
   - cos(q7)  q8  + 2 z12 sin(q7) q8 + q8  + z12

                                                                      2
   + 2 cos(q7) q8 sin(q7) cos(q9) z34 + 2 z12 cos(q7) cos(q9) z34 - z34

          2    2        2
   + cos(q9)  z34  cos(q7) ) mu) omZ omX - (- z12 cos(q7) q8

                                                          2
   + z12 sin(q7) cos(q9) z34 + q8 cos(q9) z34 - sin(q7) q8  cos(q7)

            2                                   2    2
   - 2 cos(q7)  cos(q9) z34 q8 + cos(q7) cos(q9)  z34  sin(q7)) mu omY omX

                                    2              2   2
   - (%1 + z12 sin(q9) z34 + %2) mu omZ  - ((2 cos(q7)  q8

                                                       2
   - 2 z12 cos(q7) cos(q9) z34 - 2 z12 sin(q7) q8 - 2 q8

                                                 2    2        2
   - 4 cos(q7) q8 sin(q7) cos(q9) z34 - 2 cos(q9)  z34  cos(q7) ) mu omZ - (

          2                            2
  4 cos(q7)  cos(q9) z34 q8 + 2 sin(q7) q8  cos(q7) + 2 z12 cos(q7) q8

                          2    2
   - 2 cos(q7) cos(q9)  z34  sin(q7) - 2 q8 cos(q9) z34

   - 2 z12 sin(q7) cos(q9) z34) mu omY - (2 %1 + 2 %2) mu omX) qp7

                          2
   + (sin(q7) cos(q9) z34  sin(q9) - cos(q7) q8 sin(q9) z34) mu omXp + (i1Y + (

                                          2
  2 cos(q7) q8 sin(q7) cos(q9) z34 + q8  + 2 z12 cos(q7) cos(q9) z34

                                2    2        2          2    2      2
   + 2 z12 sin(q7) q8 + cos(q9)  z34  cos(q7)  - cos(q9)  z34  + z34

          2   2      2                               2    2
   - cos(q7)  q8  + z12 ) mu) omYp - (- cos(q7) cos(q9)  z34  sin(q7)
```

```
                                                                    2
   - q8 cos(q9) z34 - z12 sin(q7) cos(q9) z34 + 2 cos(q7)  cos(q9) z34 q8

             2
   + sin(q7) q8  cos(q7) + z12 cos(q7) q8) mu omZp - (

                                                        2
  (2 z12 cos(q7) + 2 sin(q7) q8 cos(q7) + 2 cos(q7)  cos(q9) z34) mu omZ -

                                                                        2
  (2 z12 sin(q7) + 2 sin(q7) cos(q7) cos(q9) z34 + 2 q8 - 2 q8 cos(q7)  ) mu omY

   + 2 mu sin(q9) z34 omX cos(q7)) qp8 - ((2 q8 sin(q9) z34

                                        2
   + 2 z12 sin(q7) sin(q9) z34 - 2 cos(q7)  sin(q9) z34 q8

                          2
   + 2 cos(q7) cos(q9) z34  sin(q7) sin(q9)) mu omZ + (

                                      2           2
  2 z12 cos(q7) sin(q9) z34 + 2 cos(q7)  cos(q9) z34  sin(q9)

                                                    2
   + 2 sin(q7) q8 cos(q7) sin(q9) z34 - 2 cos(q9) z34  sin(q9)) mu omY

                       2    2        2
   - (2 sin(q7) cos(q9)  z34  - 2 z34  sin(q7)) mu omX) qp9

                                                                    2
   - 2 mu sin(q9) z34 cos(q7) qp7 qp8 - (%2 + z12 sin(q9) z34) mu qp9

                     2                       2    2        2
   + (%2 + %1) mu qp7  + (2 sin(q7) cos(q9)  z34  - 2 z34  sin(q7)) mu qp9 qp7

                         2
   + (sin(q7) cos(q9) z34  sin(q9) - cos(q7) q8 sin(q9) z34) mu qpp7

   - mu sin(q9) z34 sin(q7) qpp8

         2
   + (z34  cos(q7) + sin(q7) q8 cos(q9) z34 + z12 cos(q9) z34) mu qpp9

                                       2
%1 :=                   cos(q7) cos(q9) z34  sin(q9)

%2 :=                      sin(q7) q8 sin(q9) z34

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> eq9c:=
>   rlms(clct(tmp70[2]=tmp71[2],[qpp7,qpp8,qpp9],
>                               [omXp,omYp,omZp,vXp,vYp,vZp,qp7,qp8,qp9],
>                               [omX,omY,omZ],[vX,vY,vZ],[mu]));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>


                                                          /f1X   f4X\
eq9c := c1Y - (z12 + sin(q7) q8 + cos(q7) cos(q9) z34) |--- - ---|
                                                          \mu1   mu4/

                /f1Z   f4Z\
   + sin(q9) z34 |--- - ---| =
                \mu1   mu4/

                      2
  (sin(q7) cos(q9) z34  sin(q9) - cos(q7) q8 sin(q9) z34) mu omZ omY
```

```
                                      2                          2    2
 + (%1 + z12 sin(q9) z34 + %2) mu omX  + (i1X - i1Z + (cos(q9)  z34

          2   2                         2      2
 - cos(q7)  q8  + 2 z12 sin(q7) q8 + q8  + z12

                                                                      2
 + 2 cos(q7) q8 sin(q7) cos(q9) z34 + 2 z12 cos(q7) cos(q9) z34 - z34

          2    2        2
 + cos(q9)  z34  cos(q7) ) mu) omZ omX - (- z12 cos(q7) q8

                                                    2
 + z12 sin(q7) cos(q9) z34 + q8 cos(q9) z34 - sin(q7) q8  cos(q7)

           2                                  2    2
 - 2 cos(q7)  cos(q9) z34 q8 + cos(q7) cos(q9)  z34  sin(q7)) mu omY omX

                                      2              2   2
 - (%1 + z12 sin(q9) z34 + %2) mu omZ  - ((2 cos(q7)  q8

                                                      2
 - 2 z12 cos(q7) cos(q9) z34 - 2 z12 sin(q7) q8 - 2 q8

                                             2    2        2
 - 4 cos(q7) q8 sin(q7) cos(q9) z34 - 2 cos(q9)  z34  cos(q7) ) mu omZ - (

         2                                 2
4 cos(q7)  cos(q9) z34 q8 + 2 sin(q7) q8  cos(q7) + 2 z12 cos(q7) q8

                    2    2
 - 2 cos(q7) cos(q9)  z34  sin(q7) - 2 q8 cos(q9) z34

 - 2 z12 sin(q7) cos(q9) z34) mu omY - (2 %1 + 2 %2) mu omX) qp7

                       2
 + (sin(q7) cos(q9) z34  sin(q9) - cos(q7) q8 sin(q9) z34) mu omXp + (i1Y + (

                                      2
2 cos(q7) q8 sin(q7) cos(q9) z34 + q8  + 2 z12 cos(q7) cos(q9) z34

                              2    2        2          2    2      2
 + 2 z12 sin(q7) q8 + cos(q9)  z34  cos(q7)  - cos(q9)  z34  + z34

          2   2      2                                  2    2
 - cos(q7)  q8  + z12 ) mu) omYp - (- cos(q7) cos(q9)  z34  sin(q7)

                                                       2
 - q8 cos(q9) z34 - z12 sin(q7) cos(q9) z34 + 2 cos(q7)  cos(q9) z34 q8

             2
 + sin(q7) q8  cos(q7) + z12 cos(q7) q8) mu omZp - (

                                                  2
(2 z12 cos(q7) + 2 sin(q7) q8 cos(q7) + 2 cos(q7)  cos(q9) z34) mu omZ -

                                                                    2
(2 z12 sin(q7) + 2 sin(q7) cos(q7) cos(q9) z34 + 2 q8 - 2 q8 cos(q7) ) mu omY

 + 2 mu sin(q9) z34 omX cos(q7)) qp8 - ((2 q8 sin(q9) z34

                                         2
 + 2 z12 sin(q7) sin(q9) z34 - 2 cos(q7)  sin(q9) z34 q8

                        2
 + 2 cos(q7) cos(q9) z34  sin(q7) sin(q9)) mu omZ + (

                                      2           2
2 z12 cos(q7) sin(q9) z34 + 2 cos(q7)  cos(q9) z34  sin(q9)
```

```
                                                                  2
   + 2 sin(q7) q8 cos(q7) sin(q9) z34 - 2 cos(q9) z34  sin(q9)) mu omY

                        2    2        2
   - (2 sin(q7) cos(q9)  z34  - 2 z34  sin(q7)) mu omX) qp9

                                                                      2
   - 2 mu sin(q9) z34 cos(q7) qp7 qp8 - (%2 + z12 sin(q9) z34) mu qp9

                     2                      2    2        2
   + (%2 + %1) mu qp7  + (2 sin(q7) cos(q9)  z34  - 2 z34  sin(q7)) mu qp9 qp7

                          2
   + (sin(q7) cos(q9) z34  sin(q9) - cos(q7) q8 sin(q9) z34) mu qpp7

   - mu sin(q9) z34 sin(q7) qpp8

         2
   + (z34  cos(q7) + sin(q7) q8 cos(q9) z34 + z12 cos(q9) z34) mu qpp9

                                          2
%1 :=                   cos(q7) cos(q9) z34  sin(q9)

%2 :=                      sin(q7) q8 sin(q9) z34

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

bytes used=181809808, alloc=10418316, time=1351.866
```

équations complémentaires (fichier newton3)

```
> read newton2:
#####
>
> r21:=vector([r21X,r21Y,r21Z]):
> c21:=vector([c21X,c21Y,c21Z]):
> r34:=vector([r34X,r34Y,r34Z]):
> c34:=vector([c34X,c34Y,c34Z]):
#####
>
> tmp75:=add(r21,f1):
> tmp76:=map(diff,multiply(Rot1,v1),t):
> tmp77:=
>   subs({diff(q1(t),t)=qp1,diff(q2(t),t)=qp2,diff(q3(t),t)=qp3},op(tmp76)):
> tmp78:=subs(tmp17,op(tmp77)):
> tmp79:=map(simplify,multiply(transpose(Rot1),tmp78),trig):
#####
> eq10a:=subs(varslect,tmp79[1]=tmp75[1]);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                 eq10a := vZ omY - vY omZ + vXp = r21X + f1X

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> eq10b:=subs(varslect,tmp79[2]=tmp75[2]);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                 eq10b := vX omZ - vZ omX + vYp = r21Y + f1Y

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> eq10c:=subs(varslect,tmp79[3]=tmp75[3]);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                 eq10c := vZp - vX omY + vY omX = r21Z + f1Z

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp80:=add(crossprod(g1m12,r21),add(c21,c1)):
> tmp81:=map(diff,multiply(Rot1,i1,om1),t):
> tmp82:=
>   subs({diff(q1(t),t)=qp1,diff(q2(t),t)=qp2,diff(q3(t),t)=qp3},op(tmp81)):
> tmp83:=subs(tmp17,op(tmp82)):
> tmp84:=map(simplify,multiply(transpose(Rot1),tmp83),trig):
#####
> eq11a:=subs(varslect,tmp84[1]=tmp80[1]);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

    eq11a := i1X omXp - i1Y omY omZ + i1Z omZ omY = - z12 r21Y + c21X + c1X

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> eq11b:=subs(varslect,tmp84[2]=tmp80[2]);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

    eq11b := - i1Z omZ omX + i1Y omYp + i1X omX omZ = z12 r21X + c21Y + c1Y

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> eq11c:=subs(varslect,tmp84[3]=tmp80[3]);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

          eq11c := - i1X omX omY + i1Z omZp + i1Y omY omX = c21Z + c1Z
```

```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> eq12:=subs(r21Y=solve(eq10b,r21Y),eq11a);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

              eq12 := i1X omXp - i1Y omY omZ + i1Z omZ omY =

                  - z12 (vX omZ - vZ omX + vYp - f1Y) + c21X + c1X

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp85:=add(multiply(Rot34,r34),multiply(transpose(Rot12),f4)):
> tmp86:=map(diff,multiply(Rot1,v4),t):
> tmp87:=
>   subs({diff(q1(t),t)=qp1,diff(q2(t),t)=qp2,diff(q3(t),t)=qp3},op(tmp86)):
> tmp88:=subs(tmp17,op(tmp87)):
> tmp89:=map(simplify,multiply(transpose(Rot12),transpose(Rot1),tmp88),trig):
#####
> eq13a:=
>   tmp85[1]=rlms(clct(subs(varslect,tmp89[1]),
>                      [qpp7,qpp8,qpp9],
>                      [omXp,omYp,omZp,vXp,vYp,vZp,qp7,qp8,qp9],
>                      [omX,omY,omZ],[vX,vY,vZ]));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

eq13a := cos(q9(t)) r34X + sin(q9(t)) r34Z + f4X = vZ omY - vY omZ

              2                   2
         - omZ  sin(q9) z34 - omY  sin(q9) z34

         + (cos(q7) cos(q9) z34 + z12 + sin(q7) q8) omZ omX

         - (- cos(q7) q8 + sin(q7) cos(q9) z34) omY omX - (

        (2 sin(q7) cos(q9) z34 - 2 cos(q7) q8) omY

         - (2 sin(q7) q8 + 2 cos(q7) cos(q9) z34) omZ) qp7

         + (cos(q7) cos(q9) z34 + z12 + sin(q7) q8) omYp

         - (- sin(q7) cos(q9) z34 + cos(q7) q8) omZp

         - (2 omZ cos(q7) - 2 omY sin(q7)) qp8

         - (2 omZ sin(q7) sin(q9) z34 + 2 omY cos(q7) sin(q9) z34) qp9 + vXp

                          2
         - sin(q9) qp9  z34 + cos(q9) qpp9 z34

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> eq13b:=
>   tmp85[2]=rlms(clct(subs(varslect,tmp89[2]),
>                      [qpp7,qpp8,qpp9],
>                      [omXp,omYp,omZp,vXp,vYp,vZp,qp7,qp8,qp9],
>                      [omX,omY,omZ],[vX,vY,vZ]));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>


eq13b := r34Y + cos(q7(t)) f4Y + sin(q7(t)) f4Z = - sin(q7) vX omY

 + cos(q7) vX omZ + (sin(q7) vY - cos(q7) vZ) omX

                                                  2     2
 + (sin(q7) cos(q7) cos(q9) z34 - q8 cos(q7) ) omZ
```

```
                                                              2     2
 - (q8 + sin(q7) cos(q7) cos(q9) z34 + z12 sin(q7) - q8 cos(q7) ) omY

 + sin(q7) omZ sin(q9) z34 omX + cos(q7) omY sin(q9) z34 omX

                           2
 - (z12 sin(q7) + q8) omX   -

                                            2
(cos(q9) z34 - z12 cos(q7) - 2 cos(q7)  cos(q9) z34 - 2 sin(q7) q8 cos(q7)) omY

 omZ - 2 omX qp7 q8 - (z12 cos(q7) + cos(q9) z34) omXp

 - sin(q7) omYp sin(q9) z34 + cos(q7) omZp sin(q9) z34 +

(2 omZ cos(q7) cos(q9) z34 + 2 sin(q9) z34 omX - 2 omY sin(q7) cos(q9) z34) qp9

                                       2
 + cos(q7) vYp + sin(q7) vZp - qp7  q8 + 2 qp7 sin(q9) qp9 z34

 - qpp7 cos(q9) z34 + qpp8
```

>>>

```
> eq13c:=
>    tmp85[3]=rlms(clct(subs(varslect,tmp89[3]),
>                       [qpp7,qpp8,qpp9],
>                       [omXp,omYp,omZp,vXp,vYp,vZp,qp7,qp8,qp9],
>                       [omX,omY,omZ],[vX,vY,vZ]));
```

>>>

```
eq13c := - sin(q9(t)) r34X + cos(q9(t)) r34Z - sin(q7(t)) f4Y + cos(q7(t)) f4Z

   = - cos(q7) vX omY - sin(q7) vX omZ + (sin(q7) vZ + cos(q7) vY) omX

                                                  2                  2
   + (sin(q7) q8 cos(q7) - cos(q9) z34 + cos(q7)  cos(q9) z34) omZ

              2                                                     2
   - (cos(q7)  cos(q9) z34 + sin(q7) q8 cos(q7) + z12 cos(q7)) omY

   + cos(q7) omZ sin(q9) z34 omX - sin(q7) omY sin(q9) z34 omX

                                     2
   - (z12 cos(q7) + cos(q9) z34) omX

                                                              2
   - (z12 sin(q7) + 2 sin(q7) cos(q7) cos(q9) z34 - 2 q8 cos(q7)  + q8) omY omZ

   - 2 omX qp7 cos(q9) z34 + (z12 sin(q7) + q8) omXp - cos(q7) omYp sin(q9) z34

   - sin(q7) omZp sin(q9) z34 + 2 omX qp8

   - (2 cos(q7) omY cos(q9) z34 + 2 sin(q7) omZ cos(q9) z34) qp9 - sin(q7) vYp

                                          2          2
   + cos(q7) vZp + 2 qp7 qp8 - cos(q9) qp9  z34 - qp7  cos(q9) z34 + qpp7 q8

   - sin(q9) qpp9 z34
```

>>>

```
> tmp90:=add(crossprod(scalarmul(m43g4,-1),r34),c34):
#####
> eq14a:=subs(varslect,tmp90[1]=0);
```

>>>

```
                          eq14a := z34 r34Y + c34X = 0
```

```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
> eq14b:=subs(varslect,tmp90[2]=0);
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
                         eq14b := - z34 r34X + c34Y = 0
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
> eq14c:=subs(varslect,tmp90[3]=0);
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
                               eq14c := c34Z = 0
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
> r43:=multiply(Rot34,scalarmul(r34,-1));
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
r43 := [
 - cos(q9(t)) r34X - sin(q9(t)) r34Z, - r34Y, sin(q9(t)) r34X - cos(q9(t)) r34Z
  ]
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
> c43:=multiply(Rot34,scalarmul(c34,-1));
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
c43 := [
 - cos(q9(t)) c34X - sin(q9(t)) c34Z, - c34Y, sin(q9(t)) c34X - cos(q9(t)) c34Z
  ]
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
> c12:=multiply(transpose(Rot12),scalarmul(c21,-1));
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
c12 := [
 - c21X, - cos(q7(t)) c21Y - sin(q7(t)) c21Z, sin(q7(t)) c21Y - cos(q7(t)) c21Z
  ]
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
> r12:=multiply(transpose(Rot12),scalarmul(r21,-1));
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
r12 := [
 - r21X, - cos(q7(t)) r21Y - sin(q7(t)) r21Z, sin(q7(t)) r21Y - cos(q7(t)) r21Z
  ]
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
> tmp95:=add(crossprod(m21m34,r43),add(c12,c43)):
#####
> eq15:=subs(varslect,op(tmp95));
```

```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

  eq15 := [

      q8 (sin(q9) r34X - cos(q9) r34Z) - c21X - cos(q9) c34X - sin(q9) c34Z,

      - cos(q7) c21Y - sin(q7) c21Z - c34Y,

      - q8 (- cos(q9) r34X - sin(q9) r34Z) + sin(q7) c21Y - cos(q7) c21Z

           + sin(q9) c34X - cos(q9) c34Z                                ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> eq16:=subs(varslect,add(r12,r43));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

        eq16 := [ - r21X - cos(q9) r34X - sin(q9) r34Z,

            - cos(q7) r21Y - sin(q7) r21Z - r34Y,

            sin(q7) r21Y - cos(q7) r21Z + sin(q9) r34X - cos(q9) r34Z ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp100:=r21Y=solve(eq10b,r21Y):
> tmp101:=r21Z=solve(eq10c,r21Z):
#####
> eq17:=subs({tmp100,tmp101},eq16[2])=0;

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

            eq17 := - cos(q7) (vX omZ - vZ omX + vYp - f1Y)

                 - sin(q7) (vZp - vX omY + vY omX - f1Z) - r34Y = 0

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

bytes used=239950444, alloc=10680412, time=1734.750
```

A.3 principe de d'Alembert

A.3.1 premières équations (fichier dalembert1)

```
> read intro:
#####
>
> g1g4:=add(g1m12,add(multiply(Rot12,m21m34),multiply(Rot14,m43g4))):
#####
> tmp40:=map(diff,g1g4,t):
> tmp41:=crossprod(om1,g1g4):
> v4:=add(v1,add(tmp40,tmp41)):
#####
> MV:=add(scalarmul(v1,mu1),scalarmul(v4,mu4)):
#####
> tmp45:=multiply(Rot1,MV):
> tmp46:=map(simplify,tmp45,trig):
> tmp47:=map(diff,tmp46,t):
> tmp48:=map(simplify,tmp47,trig):
> tmp49:=
>   subs({diff(q1(t),t)=qp1,diff(q2(t),t)=qp2,diff(q3(t),t)=qp3},op(tmp48)):
> tmp50:=subs(tmp17,op(tmp49)):
> tmp51:=multiply(transpose(Rot1),tmp50):
> tmp52:=map(simplify,tmp51,trig):
> tmp53:=
>   subs(varslect,op(tmp52)):
> tmp54:=add(f1,f4):
#####

> eq1a:=
>   tmp54[1]=rlms(clct(tmp53[1],[qpp7,qpp8,qpp9],
>                              [omXp,omYp,omZp,vXp,vYp,vZp,qp7,qp8,qp9],
>                              [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4]));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

   eq1a := f1X + f4X = - (mu4 + mu1) vY omZ + (mu4 + mu1) vZ omY

        - (- cos(q7) q8 + sin(q7) cos(q9) z34) mu4 omY omX

        + (sin(q7) q8 + cos(q7) cos(q9) z34 + z12) mu4 omZ omX

                   2                     2
        - mu4 omZ  sin(q9) z34 - mu4 omY  sin(q9) z34 + (

       (2 cos(q7) cos(q9) z34 + 2 sin(q7) q8) mu4 omZ

        + (2 cos(q7) q8 - 2 sin(q7) cos(q9) z34) mu4 omY) qp7

        + (sin(q7) q8 + cos(q7) cos(q9) z34 + z12) mu4 omYp

        - (- sin(q7) cos(q9) z34 + cos(q7) q8) mu4 omZp

        + (2 mu4 omY sin(q7) - 2 mu4 omZ cos(q7)) qp8

        - (2 mu4 omZ sin(q7) sin(q9) z34 + 2 mu4 omY cos(q7) sin(q9) z34) qp9

                                              2
        + (mu4 + mu1) vXp - mu4 sin(q9) qp9  z34 + mu4 cos(q9) qpp9 z34

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> eq1b:=
>   tmp54[2]=rlms(clct(tmp53[2],[qpp7,qpp8,qpp9],
>                              [omXp,omYp,omZp,vXp,vYp,vZp,qp7,qp8,qp9],
>                              [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4]));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

```
eq1b := f1Y + f4Y = (mu4 + mu1) vX omZ - (mu4 + mu1) vZ omX

     + (sin(q7) q8 + cos(q7) cos(q9) z34 + z12) mu4 omZ omY

                                    2
     - (- %1 + cos(q7) q8) mu4 omX  + mu4 omY sin(q9) z34 omX

                                    2
     - (- %1 + cos(q7) q8) mu4 omZ  + (2 %1 - 2 cos(q7) q8) mu4 omX qp7

     - (sin(q7) q8 + cos(q7) cos(q9) z34 + z12) mu4 omXp + mu4 omZp sin(q9) z34

     - 2 mu4 omX sin(q7) qp8

     + (2 mu4 omX cos(q7) sin(q9) z34 + 2 mu4 omZ cos(q9) z34) qp9

     + (mu4 + mu1) vYp + 2 mu4 cos(q7) qp7 sin(q9) qp9 z34

                                                              2
     - 2 mu4 sin(q7) qp7 qp8 - (- %1 + cos(q7) q8) mu4 qp7

                               2
     + mu4 sin(q7) cos(q9) qp9  z34

     - (cos(q7) cos(q9) z34 + sin(q7) q8) mu4 qpp7 + mu4 cos(q7) qpp8

     + mu4 sin(q7) sin(q9) qpp9 z34

%1 :=                        sin(q7) cos(q9) z34

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> eq1c:=
>   tmp54[3]=rlms(clct(tmp53[3],[qpp7,qpp8,qpp9],
>                              [omXp,omYp,omZp,vXp,vYp,vZp,qp7,qp8,qp9],
>                              [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4]));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

  eq1c := f1Z + f4Z = - (mu4 + mu1) vX omY + (mu4 + mu1) vY omX

       - (- cos(q7) q8 + sin(q7) cos(q9) z34) mu4 omZ omY

                                      2
       - (sin(q7) q8 + %1 + z12) mu4 omX  + mu4 omZ sin(q9) z34 omX

                                      2
       - (sin(q7) q8 + %1 + z12) mu4 omY  - (2 %1 + 2 sin(q7) q8) mu4 omX qp7

       - (- cos(q7) q8 + sin(q7) cos(q9) z34) mu4 omXp - mu4 omYp sin(q9) z34

       + 2 mu4 omX cos(q7) qp8

       + (2 mu4 omX sin(q7) sin(q9) z34 - 2 mu4 omY cos(q9) z34) qp9

       + (mu4 + mu1) vZp + 2 mu4 sin(q7) qp7 sin(q9) qp9 z34

                                                              2
       + 2 mu4 cos(q7) qp7 qp8 - (%1 + sin(q7) q8) mu4 qp7

                                  2
       - mu4 cos(q7) cos(q9) qp9  z34

       - (- cos(q7) q8 + sin(q7) cos(q9) z34) mu4 qpp7 + mu4 sin(q7) qpp8

       - mu4 cos(q7) sin(q9) qpp9 z34

%1 :=                        cos(q7) cos(q9) z34

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

```
> tmp55:=crossprod(g1g4,multiply(Rot14,f4)):
#####
> Momf4:=subs(varslect,op(tmp55));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

      Momf4 := [ (- sin(q7) cos(q9) z34 + cos(q7) q8) f4Z

                      - (sin(q7) q8 + cos(q7) cos(q9) z34 + z12) f4Y,

          (sin(q7) q8 + cos(q7) cos(q9) z34 + z12) f4X - sin(q9) z34 f4Z,

          sin(q9) z34 f4Y - (- sin(q7) cos(q9) z34 + cos(q7) q8) f4X ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp60:=
>   scalarmul(add(multiply(i1,map(diff,om1,t)),crossprod(om1,multiply(i1,om1))),-1):
#####
> ci1:=subs(varslect,op(tmp60));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

               ci1 := [ - i1X omXp - omY i1Z omZ + omZ i1Y omY,

                   - i1Y omYp - omZ i1X omX + omX i1Z omZ,

                   - i1Z omZp - omX i1Y omY + omY i1X omX ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp65:=multiply(Rot1,v4):
> tmp66:=map(diff,tmp65,t):
> tmp67:=map(simplify,tmp66,trig):
> tmp68:=
>   subs({diff(q1(t),t)=qp1,diff(q2(t),t)=qp2,diff(q3(t),t)=qp3},op(tmp67)):
> tmp69:=subs(tmp17,op(tmp68)):
> tmp70:=multiply(transpose(Rot1),tmp69):
> tmp71:=map(simplify,tmp70,trig):
> tmp72:=subs(varslect,op(tmp71)):
#####

> gamma4:=
>   map(clct,op(tmp72),[qpp7,qpp8,qpp9],
>                      [omXp,omYp,omZp,vXp,vYp,vZp,qp7,qp8,qp9],
>                      [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4]);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

gamma4 := [

                                                                2
    - vY omZ + vZ omY + (sin(q7) q8 + %2 + z12) omX omZ - omY  sin(q9) z34

                                            2
         + (- %1 + cos(q7) q8) omX omY - omZ  sin(q9) z34

         + ((2 %2 + 2 sin(q7) q8) omZ + (2 cos(q7) q8 - 2 %1) omY) qp7

         + (sin(q7) q8 + %2 + z12) omYp + (- cos(q7) q8 + %1) omZp

         + (- 2 omZ cos(q7) + 2 omY sin(q7)) qp8

         + (- 2 omZ sin(q7) sin(q9) z34 - 2 omY cos(q7) sin(q9) z34) qp9 + vXp

                          2
         - sin(q9) qp9  z34 + cos(q9) qpp9 z34,

    vX omZ - vZ omX + omY sin(q9) z34 omX + (sin(q7) q8 + %2 + z12) omZ omY
```

```
                                  2                              2
        + (- cos(q7) q8 + %1) omZ  + (- cos(q7) q8 + %1) omX

        + (2 %1 - 2 cos(q7) q8) omX qp7 + (- %2 - sin(q7) q8 - z12) omXp

        + omZp sin(q9) z34 - 2 omX sin(q7) qp8

        + (2 omZ cos(q9) z34 + 2 omX cos(q7) sin(q9) z34) qp9 + vYp

        + 2 cos(q7) qp7 sin(q9) qp9 z34 - 2 sin(q7) qp7 qp8

                               2                              2
        + sin(q7) cos(q9) qp9  z34 + (- cos(q7) q8 + %1) qp7

        + (- sin(q7) q8 - %2) qpp7 + cos(q7) qpp8 + sin(q7) sin(q9) qpp9 z34,

                                                                        2
    - vX omY + vY omX + omZ sin(q9) z34 omX + (- %2 - sin(q7) q8 - z12) omY

                                                                     2
        + (- %1 + cos(q7) q8) omZ omY + (- %2 - sin(q7) q8 - z12) omX

        + (- 2 sin(q7) q8 - 2 %2) omX qp7 + (- %1 + cos(q7) q8) omXp

        - omYp sin(q9) z34 + 2 cos(q7) qp8 omX

        + (- 2 cos(q9) z34 omY + 2 sin(q7) sin(q9) z34 omX) qp9 + vZp

        + 2 sin(q7) qp7 sin(q9) qp9 z34 + 2 cos(q7) qp7 qp8

                               2                              2
        - cos(q7) cos(q9) qp9  z34 + (- sin(q7) q8 - %2) qp7

        + (- %1 + cos(q7) q8) qpp7 + sin(q7) qpp8 - cos(q7) sin(q9) qpp9 z34 ]

%1 :=                       sin(q7) cos(q9) z34

%2 :=                       cos(q7) cos(q9) z34

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp75:=scalarmul(tmp71,-mu4):
> tmp76:=crossprod(g1g4,tmp75):
> tmp77:=map(simplify,op(tmp76),trig):
> tmp78:=subs(varslect,op(tmp77)):
#####
> Momfi4:=
>   map(clct,op(tmp78),[qpp7,qpp8,qpp9],
>                      [omXp,omYp,omZp,vXp,vYp,vZp,qp7,qp8,qp9],
>                      [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4]);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

Momfi4 := [

(sin(q7) q8 + %15 + z12) mu4 vX omZ + (- %5 + cos(q7) q8) mu4 vX omY

    + ((- cos(q7) q8 + %5) mu4 vY + (- %15 - sin(q7) q8 - z12) mu4 vZ) omX

    + %13 mu4 omY omX +

      2             2                              2         2
   (q8  + 4 %7 + z12  + 2 %6 - %14 + 2 %16 - 2 q8  cos(q7)  + 2 z12 sin(q7) q8)

    mu4 omY omZ + (%3 - %4) mu4 omX omZ

                                                                      2
    + (- %8 - z12 cos(q7) q8 + %10 - 2 %11 + q8 cos(q9) z34 + %9) mu4 omZ
```

```
                  2
    + %12 mu4 omY  + (2 %9 - 2 z12 cos(q7) q8) mu4 omX qp7

          2                                  2
    + (- q8  - 2 z12 sin(q7) q8 - %14 - z12  - 2 %16) mu4 omXp

    + (%4 - %3) mu4 omYp + %13 mu4 omZp + (- 2 z12 sin(q7) - 2 q8) mu4 omX qp8

            2                2
    + ((2 z34  cos(q7) cos(q9)  + 2 sin(q7) q8 cos(q9) z34 + 2 z12 cos(q9) z34)

                     2                2
    mu4 omZ + (- 2 z34  sin(q7) cos(q9)  + 2 cos(q7) q8 cos(q9) z34) mu4 omY

                   2
    + (2 sin(q9) z34  cos(q9) + 2 z12 cos(q7) sin(q9) z34) mu4 omX) qp9

    + (sin(q7) q8 + %15 + z12) mu4 vYp + (- cos(q7) q8 + %5) mu4 vZp

    + (- 2 z12 sin(q7) - 2 q8) mu4 qp8 qp7

                   2
    + (2 sin(q9) z34  cos(q9) + 2 z12 cos(q7) sin(q9) z34) mu4 qp9 qp7

                                    2                                  2
    + (%9 + q8 cos(q9) z34) mu4 qp9  + (%9 - z12 cos(q7) q8) mu4 qp7

                                          2
    + (- z12 sin(q7) q8 - %16 - %14 - q8 ) mu4 qpp7

    + (z12 cos(q7) + cos(q9) z34) mu4 qpp8

    + (z12 sin(q7) sin(q9) z34 + q8 sin(q9) z34) mu4 qpp9,

(sin(q7) q8 + %15 + z12) mu4 vY omZ

 + ((- %15 - sin(q7) q8 - z12) mu4 vZ - sin(q9) z34 mu4 vX) omY

 + sin(q9) z34 mu4 vY omX

 + (- %8 - z12 cos(q7) q8 + %10 - 2 %11 + q8 cos(q9) z34 + %9) mu4 omY omX

 + (%4 - %3) mu4 omY omZ + (

                          2                                2         2      2      2
- %14 - 2 %7 - 2 %16 - q8  - 2 z12 sin(q7) q8 - %6 + q8  cos(q7)  + z34  - z12

                         2                                        2
) mu4 omX omZ + %13 mu4 omZ  + (- z12 sin(q9) z34 - %2 - %1) mu4 omX  + (

       2                                    2        2
(- 2 q8  - 2 z12 sin(q7) q8 - 4 %7 + 2 q8  cos(q7)  - 2 %16 - 2 %6) mu4 omZ

 + (2 q8 cos(q9) z34 + 2 %10 + 2 %9 - 2 %8 - 2 z12 cos(q7) q8 - 4 %11) mu4 omY

 + (- 2 %2 - 2 %1) mu4 omX) qp7 + (%4 - %3) mu4 omXp + (

                                        2     2        2      2      2
- %6 - 2 %7 - 2 z12 sin(q7) q8 + %14 - q8  + q8  cos(q7)  - z12  - z34  - 2 %16

) mu4 omYp + %12 mu4 omZp + (

                                                 2
(2 z12 cos(q7) + 2 sin(q7) q8 cos(q7) + 2 cos(q7)  cos(q9) z34) mu4 omZ +

                                                                      2
(- 2 cos(q7) cos(q9) z34 sin(q7) - 2 z12 sin(q7) - 2 q8 + 2 q8 cos(q7) ) mu4

omY + 2 mu4 omX cos(q7) sin(q9) z34) qp8 + ((2 q8 sin(q9) z34
```

```
                          2                        2
 - 2 q8 sin(q9) z34 cos(q7)  + 2 cos(q7) cos(q9) z34  sin(q7) sin(q9)

                                                        2
 + 2 z12 sin(q7) sin(q9) z34) mu4 omZ + (- 2 sin(q9) z34  cos(q9)

           2            2
 + 2 cos(q7)  cos(q9) z34  sin(q9) + 2 sin(q7) q8 cos(q7) sin(q9) z34

 + 2 z12 cos(q7) sin(q9) z34) mu4 omY

          2               2        2
 + (- 2 z34  sin(q7) cos(q9)  + 2 z34  sin(q7)) mu4 omX) qp9

 + (- %15 - sin(q7) q8 - z12) mu4 vXp + sin(q9) z34 mu4 vZp

 + 2 sin(q9) z34 mu4 cos(q7) qp7 qp8

          2               2        2
 + (- 2 z34  sin(q7) cos(q9)  + 2 z34  sin(q7)) mu4 qp9 qp7

                                  2                       2
 + (%2 + z12 sin(q9) z34) mu4 qp9  + (- %2 - %1) mu4 qp7  + (%4 - %3) mu4 qpp7

 + sin(q9) z34 mu4 sin(q7) qpp8

                          2
 + (- z12 cos(q9) z34 - z34  cos(q7) - sin(q7) q8 cos(q9) z34) mu4 qpp9,

((- cos(q7) q8 + %5) mu4 vY - sin(q9) z34 mu4 vX) omZ

 + (- %5 + cos(q7) q8) mu4 vZ omY + sin(q9) z34 mu4 vZ omX

                     2             2        2
 + (- 2 %7 - %6 - z34  + 2 %14 + q8  cos(q7) ) mu4 omY omX

 + (- z12 sin(q9) z34 - %2 - %1) mu4 omY omZ + %12 mu4 omX omZ

                    2                    2
 + (%4 - %3) mu4 omX  + (%3 - %4) mu4 omY  + (

(- 2 q8 cos(q9) z34 + 4 %11 + 2 %8 - 2 %10) mu4 omZ

                         2        2
 + (- 4 %7 - 2 %6 + 2 q8  cos(q7)  + 2 %14) mu4 omY + (2 %4 - 2 %3) mu4 omX)

qp7 + %13 mu4 omXp + %12 mu4 omYp

         2     2        2
 + (- z34  - q8  cos(q7)  + 2 %7 + %6) mu4 omZp + (

              2
(- 2 q8 cos(q7)  + 2 cos(q7) cos(q9) z34 sin(q7)) mu4 omZ

            2
 + (2 cos(q7)  cos(q9) z34 + 2 sin(q7) q8 cos(q7) - 2 cos(q9) z34) mu4 omY

 + 2 mu4 omX sin(q7) sin(q9) z34) qp8 + (

                                                  2            2
(- 2 sin(q7) q8 cos(q7) sin(q9) z34 - 2 cos(q7)  cos(q9) z34  sin(q9)) mu4 omZ

                          2                                      2
 + (2 cos(q7) cos(q9) z34  sin(q7) sin(q9) - 2 q8 sin(q9) z34 cos(q7) ) mu4 omY

         2               2        2
 + (2 z34  cos(q7) cos(q9)  - 2 z34  cos(q7)) mu4 omX) qp9

 + (- %5 + cos(q7) q8) mu4 vXp - sin(q9) z34 mu4 vYp
```

```
 + 2 sin(q9) z34 mu4 sin(q7) qp7 qp8

        2                2        2
 + (2 z34  cos(q7) cos(q9)  - 2 z34  cos(q7)) mu4 qp9 qp7

                              2                          2
 - mu4 cos(q7) q8 sin(q9) qp9  z34 + (%4 - %3) mu4 qp7  + (%2 + %1) mu4 qpp7

 - sin(q9) z34 mu4 cos(q7) qpp8

        2
 + (- z34  sin(q7) + cos(q7) q8 cos(q9) z34) mu4 qpp9

 ]

                                  2
%1 :=                   sin(q9) z34  cos(q7) cos(q9)

%2 :=                      sin(q9) z34 sin(q7) q8

                                  2
%3 :=                   sin(q9) z34  sin(q7) cos(q9)

%4 :=                      sin(q9) z34 cos(q7) q8

%5 :=                        sin(q7) cos(q9) z34

                                 2        2    2
%6 :=                      cos(q7)  cos(q9)  z34

%7 :=                  sin(q7) q8 cos(q7) cos(q9) z34

                                       2
%8 :=                        sin(q7) q8  cos(q7)

%9 :=                      z12 sin(q7) cos(q9) z34

                                       2    2
%10 :=                  cos(q7) cos(q9)  z34  sin(q7)

                                  2
%11 :=                     cos(q7)  cos(q9) z34 q8

%12 :=     2 %11 - %10 - %9 + z12 cos(q7) q8 + %8 - q8 cos(q9) z34

%13 :=                     %2 + %1 + z12 sin(q9) z34

                                  2        2
%14 :=                         z34  cos(q9)

%15 :=                       cos(q7) cos(q9) z34

%16 :=                     z12 cos(q7) cos(q9) z34

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp80:=add(c1,add(ci1,add(Momf4,Momfi4))):
> tmp81:=map(expand,op(tmp80)):
> tmp82:=map(simplify,op(tmp81),trig):
#####
> eq2a:=
>   rlms(clct(tmp82[1],[qpp7,qpp8,qpp9],
>                      [omXp,omYp,omZp,vXp,vYp,vZp,qp7,qp8,qp9],
>                      [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4]));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

eq2a := c1X + f4Z cos(q7) q8 - f4Y sin(q7) q8 - f4Y z12

 - f4Z sin(q7) cos(q9) z34 - f4Y cos(q7) cos(q9) z34
```

```
 + (z12 sin(q7) sin(q9) z34 + q8 sin(q9) z34) mu4 qpp9

                                                                      2
 + (z12 cos(q7) + cos(q9) z34) mu4 qpp8 + (%1 + q8 cos(q9) z34) mu4 qp9

                 2
 + (2 sin(q9) z34  cos(q9) + 2 z12 cos(q7) sin(q9) z34) mu4 qp9 qp7

 + (sin(q7) q8 + cos(q7) cos(q9) z34 + z12) mu4 vYp +

           2
(sin(q9) z34  cos(q7) cos(q9) + sin(q9) z34 sin(q7) q8 + z12 sin(q9) z34) mu4

                                                  2
omZp + (sin(q9) z34 cos(q7) q8 - sin(q9) z34  sin(q7) cos(q9)) mu4 omYp

                                                          2    2
 + (2 %1 - 2 z12 cos(q7) q8) mu4 omX qp7 + (cos(q7) cos(q9)  z34  sin(q7)

                                                    2
 + q8 cos(q9) z34 - z12 cos(q7) q8 - 2 cos(q7)  cos(q9) z34 q8

            2                         2
 - sin(q7) q8  cos(q7) + %1) mu4 omZ  +

           2
(sin(q9) z34  cos(q7) cos(q9) + sin(q9) z34 sin(q7) q8 + z12 sin(q9) z34) mu4

omY omX + (sin(q7) q8 + cos(q7) cos(q9) z34 + z12) mu4 vX omZ + (i1Y - i1Z - (

    2        2                                    2      2        2
2 q8  cos(q7)  - 2 z12 cos(q7) cos(q9) z34 - z12  + z34  cos(q9)

 - 4 sin(q7) q8 cos(q7) cos(q9) z34 - 2 z12 sin(q7) q8

           2        2    2     2
 - 2 cos(q7)  cos(q9)  z34  - q8 ) mu4) omZ omY

 - (- cos(q7) q8 + sin(q7) cos(q9) z34) mu4 vX omY - (i1X +

                       2        2      2     2
(2 z12 sin(q7) q8 + z34  cos(q9)  + z12  + q8  + 2 z12 cos(q7) cos(q9) z34) mu4

) omXp - ((- sin(q7) cos(q9) z34 + cos(q7) q8) mu4 vY

 + (sin(q7) q8 + cos(q7) cos(q9) z34 + z12) mu4 vZ) omX

                                      2
 - (sin(q9) z34 cos(q7) q8 - sin(q9) z34  sin(q7) cos(q9)) mu4 omX omZ - (

               2    2
cos(q7) cos(q9)  z34  sin(q7) + q8 cos(q9) z34 - z12 cos(q7) q8

           2                                2                       2
 - 2 cos(q7)  cos(q9) z34 q8 - sin(q7) q8  cos(q7) + %1) mu4 omY

 - (2 z12 sin(q7) + 2 q8) mu4 omX qp8 - (2 z12 sin(q7) + 2 q8) mu4 qp8 qp7 + (

                                    2                2
(2 sin(q7) q8 cos(q9) z34 + 2 z34  cos(q7) cos(q9)  + 2 z12 cos(q9) z34) mu4

           2                2
omZ - (2 z34  sin(q7) cos(q9)  - 2 cos(q7) q8 cos(q9) z34) mu4 omY

                 2
 + (2 sin(q9) z34  cos(q9) + 2 z12 cos(q7) sin(q9) z34) mu4 omX) qp9

 - (- sin(q7) cos(q9) z34 + cos(q7) q8) mu4 vZp
```

```
                        2        2     2
 - (z12 sin(q7) q8 + z34  cos(q9)  + q8  + z12 cos(q7) cos(q9) z34) mu4 qpp7

                                  2
 - (z12 cos(q7) q8 - %1) mu4 qp7

%1 :=                      z12 sin(q7) cos(q9) z34

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> eq2b:=
>   rlms(clct(tmp82[2],[qpp7,qpp8,qpp9],
>                      [omXp,omYp,omZp,vXp,vYp,vZp,qp7,qp8,qp9],
>                      [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4]));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                                                 2          2        2    2
eq2b := f4X sin(q7) q8 + f4X z12 - (i1Y + (z12  + cos(q7)  cos(q9)  z34

                                                                 2     2        2
  + 2 z12 sin(q7) q8 + 2 sin(q7) q8 cos(q7) cos(q9) z34 + q8  - q8  cos(q7)

                                      2      2        2
  + 2 z12 cos(q7) cos(q9) z34 + z34  - z34  cos(q9) ) mu4) omYp - (i1X - i1Z +

     2        2
 (z34  cos(q9)  + 2 sin(q7) q8 cos(q7) cos(q9) z34 + 2 z12 cos(q7) cos(q9) z34

           2        2    2      2                          2        2      2
  + cos(q7)  cos(q9)  z34  + z12  + 2 z12 sin(q7) q8 - q8  cos(q7)  - z34

      2
  + q8 ) mu4) omX omZ + f4X cos(q7) cos(q9) z34 - sin(q9) z34 f4Z + c1Y

  - (sin(q7) q8 + cos(q7) cos(q9) z34 + z12) mu4 vXp - ((2 z12 sin(q7) q8

        2        2       2
  - 2 q8  cos(q7)  + 2 q8  + 2 z12 cos(q7) cos(q9) z34

                                                  2        2    2
  + 4 sin(q7) q8 cos(q7) cos(q9) z34 + 2 cos(q7)  cos(q9)  z34 ) mu4 omZ - (

                                        2
 2 z12 sin(q7) cos(q9) z34 - 4 cos(q7)  cos(q9) z34 q8 + 2 q8 cos(q9) z34

                                      2    2                          2
  - 2 z12 cos(q7) q8 + 2 cos(q7) cos(q9)  z34  sin(q7) - 2 sin(q7) q8  cos(q7))

  mu4 omY + (2 %2 + 2 %1) mu4 omX) qp7 + sin(q9) z34 mu4 vZp

  + sin(q9) z34 mu4 vY omX + 2 sin(q9) z34 mu4 cos(q7) qp7 qp8

                                                    2
  + sin(q9) z34 mu4 sin(q7) qpp8 - (%2 + %1) mu4 qp7  + (

                                                   2
 (2 z12 cos(q7) + 2 sin(q7) q8 cos(q7) + 2 cos(q7)  cos(q9) z34) mu4 omZ -

                                                        2
 (2 q8 + 2 cos(q7) cos(q9) z34 sin(q7) - 2 q8 cos(q7)  + 2 z12 sin(q7)) mu4 omY

  + 2 mu4 omX cos(q7) sin(q9) z34) qp8

       2
  - (z34  cos(q7) + sin(q7) q8 cos(q9) z34 + z12 cos(q9) z34) mu4 qpp9 - (

                2    2
 cos(q7) cos(q9)  z34  sin(q7) + q8 cos(q9) z34 - z12 cos(q7) q8
```

```
                 2                             2
  - 2 cos(q7)  cos(q9) z34 q8 - sin(q7) q8  cos(q7) + z12 sin(q7) cos(q9) z34)

 mu4 omZp

  - ((sin(q7) q8 + cos(q7) cos(q9) z34 + z12) mu4 vZ + sin(q9) z34 mu4 vX) omY

                                                2
  + (sin(q9) z34 cos(q7) q8 - sin(q9) z34  sin(q7) cos(q9)) mu4 qpp7

                                         2
  + (z12 sin(q9) z34 + %1) mu4 qp9

              2                2                2
  + (2 z34  sin(q7) - 2 z34  sin(q7) cos(q9) ) mu4 qp9 qp7 + ((

                          2                                          2
 2 cos(q7) cos(q9) z34  sin(q7) sin(q9) - 2 q8 sin(q9) z34 cos(q7)

  + 2 q8 sin(q9) z34 + 2 z12 sin(q7) sin(q9) z34) mu4 omZ + (

             2           2
 2 cos(q7)  cos(q9) z34  sin(q9) + 2 z12 cos(q7) sin(q9) z34

                                                         2
  + 2 sin(q7) q8 cos(q7) sin(q9) z34 - 2 sin(q9) z34  cos(q9)) mu4 omY

              2                2                2
  + (2 z34  sin(q7) - 2 z34  sin(q7) cos(q9) ) mu4 omX) qp9

                                                2
  + (sin(q9) z34 cos(q7) q8 - sin(q9) z34  sin(q7) cos(q9)) mu4 omXp

                                              2                  2    2
  + (%2 + %1 + z12 sin(q9) z34) mu4 omZ  + (cos(q7) cos(q9)  z34  sin(q7)

                                                      2
  + q8 cos(q9) z34 - z12 cos(q7) q8 - 2 cos(q7)  cos(q9) z34 q8

                  2
  - sin(q7) q8  cos(q7) + z12 sin(q7) cos(q9) z34) mu4 omY omX

                                                2
  + (sin(q9) z34 cos(q7) q8 - sin(q9) z34  sin(q7) cos(q9)) mu4 omY omZ

                                              2
  - (%2 + %1 + z12 sin(q9) z34) mu4 omX

  + (sin(q7) q8 + cos(q7) cos(q9) z34 + z12) mu4 vY omZ

%1 :=                     sin(q9) z34 sin(q7) q8

                                     2
%2 :=                  sin(q9) z34  cos(q7) cos(q9)
```

```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> eq2c:=
>   rlms(clct(tmp82[3],[qpp7,qpp8,qpp9],
>                     [omXp,omYp,omZp,vXp,vYp,vZp,qp7,qp8,qp9],
>                     [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4]));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

```
                                                                 2
eq2c := sin(q9) z34 f4Y + c1Z - mu4 cos(q7) q8 sin(q9) qp9  z34

 - sin(q9) z34 mu4 vYp + sin(q9) z34 mu4 vZ omX

 + 2 sin(q9) z34 mu4 sin(q7) qp7 qp8 - sin(q9) z34 mu4 cos(q7) qpp8
```

```
 - f4X cos(q7) q8 + f4X sin(q7) cos(q9) z34

                                  2                                    2
 + (cos(q7) q8 cos(q9) z34 - z34  sin(q7)) mu4 qpp9 + (%1 - %2) mu4 qp7

              2
 + (sin(q9) z34  cos(q7) cos(q9) + sin(q9) z34 sin(q7) q8) mu4 qpp7 + (

                                                 2
(2 cos(q7) cos(q9) z34 sin(q7) - 2 q8 cos(q7) ) mu4 omZ

             2
 + (2 cos(q7)  cos(q9) z34 + 2 sin(q7) q8 cos(q7) - 2 cos(q9) z34) mu4 omY

 + 2 mu4 omX sin(q7) sin(q9) z34) qp8 +

           2
(sin(q9) z34  cos(q7) cos(q9) + sin(q9) z34 sin(q7) q8 + z12 sin(q9) z34) mu4

                         2                   2    2
omXp + (%1 - %2) mu4 omX  - (cos(q7) cos(q9)  z34  sin(q7) + q8 cos(q9) z34

                                  2                                   2
 - z12 cos(q7) q8 - 2 cos(q7)  cos(q9) z34 q8 - sin(q7) q8  cos(q7)

                                                                    2
 + z12 sin(q7) cos(q9) z34) mu4 omYp - (%1 - %2) mu4 omY  - (

               2    2
cos(q7) cos(q9)  z34  sin(q7) + q8 cos(q9) z34 - z12 cos(q7) q8

            2                                2
 - 2 cos(q7)  cos(q9) z34 q8 - sin(q7) q8  cos(q7) + z12 sin(q7) cos(q9) z34)

mu4 omX omZ - (

          2            2
(2 cos(q7)  cos(q9) z34  sin(q9) + 2 sin(q7) q8 cos(q7) sin(q9) z34) mu4 omZ

                               2                                 2
 + (2 q8 sin(q9) z34 cos(q7)  - 2 cos(q7) cos(q9) z34  sin(q7) sin(q9)) mu4 omY

         2                2                2
 + (2 z34  cos(q7) - 2 z34  cos(q7) cos(q9) ) mu4 omX) qp9

 - (- cos(q7) q8 + sin(q7) cos(q9) z34) mu4 vZ omY

         2                2                2
 - (2 z34  cos(q7) - 2 z34  cos(q7) cos(q9) ) mu4 qp9 qp7 - (i1Y - i1X - (

     2        2      2          2        2    2     2        2
2 z34  cos(q9)  - z34  - cos(q7)  cos(q9)  z34  + q8  cos(q7)

 - 2 sin(q7) q8 cos(q7) cos(q9) z34) mu4) omX omY

                                                                     2
 - (- cos(q7) q8 + sin(q7) cos(q9) z34) mu4 vXp + ((4 cos(q7)  cos(q9) z34 q8

               2                                 2    2
 + 2 sin(q7) q8  cos(q7) - 2 cos(q7) cos(q9)  z34  sin(q7) - 2 q8 cos(q9) z34)

                2        2            2        2    2
mu4 omZ + (2 z34  cos(q9)  - 2 cos(q7)  cos(q9)  z34

                                           2        2
 - 4 sin(q7) q8 cos(q7) cos(q9) z34 + 2 q8  cos(q7) ) mu4 omY

 - (2 %2 - 2 %1) mu4 omX) qp7
```

```
 - ((- sin(q7) cos(q9) z34 + cos(q7) q8) mu4 vY + sin(q9) z34 mu4 vX) omZ - (

i1Z - (

                                        2     2        2          2        2    2
2 sin(q7) q8 cos(q7) cos(q9) z34 - z34  - q8  cos(q7)  + cos(q7)  cos(q9)  z34

) mu4) omZp -

           2
(sin(q9) z34  cos(q7) cos(q9) + sin(q9) z34 sin(q7) q8 + z12 sin(q9) z34) mu4

omY omZ

%1 :=                       sin(q9) z34 cos(q7) q8

                                    2
%2 :=                    sin(q9) z34  sin(q7) cos(q9)

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

bytes used=93069060, alloc=4652204, time=581.766
```

A.3.2 équations des corps isolés (fichier dalembert2)

```
> read dalembert1:
#####
>
> c21:=vector([c21X,c21Y,c21Z]):
> r34:=vector([r34X,r34Y,r34Z]):
> c34:=vector([c34X,c34Y,c34Z]):
#####
>
> tmp85:=add(multiply(Rot34,r34),multiply(transpose(Rot12),f4)):
> tmp86:=map(diff,multiply(Rot1,v4),t):

> tmp87:=
>   subs({diff(q1(t),t)=qp1,diff(q2(t),t)=qp2,diff(q3(t),t)=qp3},op(tmp86)):
> tmp88:=subs(tmp17,op(tmp87)):
> tmp89:=map(simplify,multiply(transpose(Rot12),transpose(Rot1),tmp88),trig):
#####
> eq3a:=
>   tmp85[1]=rlms(clct(subs(varslect,tmp89[1]),
>                      [qpp7,qpp8,qpp9],
>                      [omXp,omYp,omZp,vXp,vYp,vZp,qp7,qp8,qp9],
>                      [omX,omY,omZ],[vX,vY,vZ]));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

   eq3a := cos(q9(t)) r34X + sin(q9(t)) r34Z + f4X = vZ omY - vY omZ

        - (- cos(q7) q8 + sin(q7) cos(q9) z34) omX omY

                                                                  2
        + (cos(q7) cos(q9) z34 + sin(q7) q8 + z12) omX omZ - omZ  sin(q9) z34

             2
        - omY  sin(q9) z34 + ((2 cos(q7) q8 - 2 sin(q7) cos(q9) z34) omY

        + (2 cos(q7) cos(q9) z34 + 2 sin(q7) q8) omZ) qp7

        + (2 omY sin(q7) - 2 omZ cos(q7)) qp8

        - (2 omY cos(q7) sin(q9) z34 + 2 omZ sin(q7) sin(q9) z34) qp9

        + (cos(q7) cos(q9) z34 + sin(q7) q8 + z12) omYp

                                                                         2
        - (- sin(q7) cos(q9) z34 + cos(q7) q8) omZp + vXp - sin(q9) qp9  z34

        + cos(q9) qpp9 z34

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> eq3b:=
>   tmp85[2]=rlms(clct(subs(varslect,tmp89[2]),
>                      [qpp7,qpp8,qpp9],
>                      [omXp,omYp,omZp,vXp,vYp,vZp,qp7,qp8,qp9],
>                      [omX,omY,omZ],[vX,vY,vZ]));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

eq3b := r34Y + cos(q7(t)) f4Y + sin(q7(t)) f4Z = - sin(q7) vX omY

 + cos(q7) vX omZ + (sin(q7) vY - cos(q7) vZ) omX -

                                         2
(cos(q9) z34 - z12 cos(q7) - 2 cos(q7)  cos(q9) z34 - 2 sin(q7) q8 cos(q7)) omY

 omZ + sin(q7) omZ sin(q9) z34 omX + cos(q7) omY sin(q9) z34 omX

                                                2     2
 + (cos(q7) cos(q9) z34 sin(q7) - q8 cos(q7) ) omZ
```

```
                              2                                   2
 - (z12 sin(q7) - q8 cos(q7)  + cos(q7) cos(q9) z34 sin(q7) + q8) omY

                        2
 - (z12 sin(q7) + q8) omX  - 2 qp7 q8 omX +

(2 sin(q9) z34 omX - 2 omY sin(q7) cos(q9) z34 + 2 omZ cos(q7) cos(q9) z34) qp9

 - (cos(q9) z34 + z12 cos(q7)) omXp - sin(q7) omYp sin(q9) z34

                                                                 2
 + cos(q7) omZp sin(q9) z34 + cos(q7) vYp + sin(q7) vZp - qp7  q8

 + 2 qp7 sin(q9) qp9 z34 - qpp7 cos(q9) z34 + qpp8

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> eq3c:=
>   tmp85[3]=rlms(clct(subs(varslect,tmp89[3]),
>                      [qpp7,qpp8,qpp9],
>                      [omXp,omYp,omZp,vXp,vYp,vZp,qp7,qp8,qp9],
>                      [omX,omY,omZ],[vX,vY,vZ]));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

eq3c := - sin(q9(t)) r34X + cos(q9(t)) r34Z - sin(q7(t)) f4Y + cos(q7(t)) f4Z

   = - cos(q7) vX omY - sin(q7) vX omZ + (cos(q7) vY + sin(q7) vZ) omX

                                                                     2
   - (2 cos(q7) cos(q9) z34 sin(q7) + q8 + z12 sin(q7) - 2 q8 cos(q7)  ) omY omZ

   + cos(q7) omZ sin(q9) z34 omX - sin(q7) omY sin(q9) z34 omX

              2                                                    2
   + (cos(q7)  cos(q9) z34 + sin(q7) q8 cos(q7) - cos(q9) z34) omZ

                                                     2                2
   - (sin(q7) q8 cos(q7) + z12 cos(q7) + cos(q7)  cos(q9) z34) omY

                                         2
   - (cos(q9) z34 + z12 cos(q7)) omX  - 2 omX qp7 cos(q9) z34 + 2 omX qp8

   - (2 sin(q7) omZ cos(q9) z34 + 2 cos(q7) cos(q9) z34 omY) qp9

   + (z12 sin(q7) + q8) omXp - cos(q7) omYp sin(q9) z34

                                                                        2
   - sin(q7) omZp sin(q9) z34 - sin(q7) vYp + cos(q7) vZp - cos(q9) qp9  z34

                      2
   + 2 qp7 qp8 - qp7  cos(q9) z34 + qpp7 q8 - sin(q9) qpp9 z34

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp90:=add(crossprod(scalarmul(m43g4,-1),r34),c34):
#####
> eq4a:=subs(varslect,tmp90[1]=0);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                          eq4a := z34 r34Y + c34X = 0

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> eq4b:=subs(varslect,tmp90[2]=0);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

```
                         eq4b := - z34 r34X + c34Y = 0

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> eq4c:=subs(varslect,tmp90[3]=0);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                                eq4c := c34Z = 0

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp95:=add(crossprod(m43g4,multiply(transpose(Rot14),f4)),c34):
#####
> eq5a:=subs(varslect,tmp95[1]=0);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

              eq5a := - z34 (cos(q7) f4Y + sin(q7) f4Z) + c34X = 0

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> eq5b:=subs(varslect,tmp95[2]=0);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

  eq5b :=

      z34 (cos(q9) f4X + sin(q7) sin(q9) f4Y - cos(q7) sin(q9) f4Z) + c34Y = 0

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> eq5c:=subs(varslect,tmp95[3]=0);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                                eq5c := c34Z = 0

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp100:=add(crossprod(scalarmul(g1m12,-1),f1),add(c21,c1)):
> tmp101:=map(diff,multiply(Rot1,i1,om1),t):
> tmp102:=
>   subs({diff(q1(t),t)=qp1,diff(q2(t),t)=qp2,diff(q3(t),t)=qp3},op(tmp101)):
> tmp103:=subs(tmp17,op(tmp102)):
> tmp104:=map(simplify,multiply(transpose(Rot1),tmp103),trig):
> tmp105:=add(scalarmul(tmp104,-1),tmp100):
#####

> eq6a:=subs(varslect,tmp105[1]=0);
> eq6b:=subs(varslect,tmp105[2]=0);
> eq6c:=subs(varslect,tmp105[3]=0);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

   eq6a := - i1X omXp - omY i1Z omZ + omZ i1Y omY + z12 f1Y + c21X + c1X = 0

   eq6b := - i1Y omYp - omZ i1X omX + omX i1Z omZ - z12 f1X + c21Y + c1Y = 0

        eq6c := - i1Z omZp - omX i1Y omY + omY i1X omX + c21Z + c1Z = 0

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

bytes used=139509612, alloc=6683448, time=938.133
```

A.4 equations de Lagrange (fichier `lagrange`)

```
> read intro:
#####
>
> c12:=vector([C12,0,0]):
> f:=vector([0,F23,0]):
> c34:=vector([0,C34,0]):
#####
>
> g1m34:=add(g1m12,multiply(Rot12,m21m34)):
#####
> tmp35:=map(diff,g1m34,t):
> tmp36:=crossprod(om1,g1m34):
> v34:=add(v1,add(tmp35,tmp36)):
#####
> g1g4:=add(g1m34,multiply(Rot14,m43g4)):
bytes used=3000344, alloc=1244956, time=13.383
#####
> tmp37:=map(diff,g1g4,t):
> tmp38:=crossprod(om1,g1g4):
> v4:=add(v1,add(tmp37,tmp38)):
#####
> tmp39:=crossprod(om1,g1m12):
> v12:=add(v1,tmp39):
#####
> om2:=add(om1,vector([diff(q7(t),t),0,0])):
#####
> om4:=add(om2,multiply(Rot12,vector([0,diff(q9(t),t),0]))):
#####
>
> tmp45:=
>   (dotprod(v1,v1)*mu1+dotprod(multiply(i1,om1),om1)+dotprod(v4,v4)*mu4)/2:
#####
> eq1:=subs(varslect,tmp45);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

               2     2     2                   2              2              2
 eq1 := 1/2 (vX  + vY  + vZ ) mu1 + 1/2 i1X omX  + 1/2 i1Y omY  + 1/2 i1Z omZ

      + 1/2 ((vX + cos(q9) qp9 z34

      + omY (z12 + sin(q7) q8 + cos(q7) cos(q9) z34)

      - omZ (cos(q7) q8 - sin(q7) cos(q9) z34))^2 + (vY - sin(q7) qp7 q8

      + cos(q7) qp8 - cos(q7) qp7 cos(q9) z34 + sin(q7) sin(q9) qp9 z34

      + omZ sin(q9) z34 - omX (z12 + sin(q7) q8 + cos(q7) cos(q9) z34))^2 + (vZ

      + cos(q7) qp7 q8 + sin(q7) qp8 - sin(q7) qp7 cos(q9) z34

      - cos(q7) sin(q9) qp9 z34 + omX (cos(q7) q8 - sin(q7) cos(q9) z34)

      - omY sin(q9) z34)^2) mu4

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp50:=subs(tmp25 union tmp32,tmp45):
> tmp51:=
>   subs({qp1=diff(q1(t),t),qp2=diff(q2(t),t),qp3=diff(q3(t),t),
>         qp4=diff(q4(t),t),qp5=diff(q5(t),t),qp6=diff(q6(t),t)},tmp50):
#####

> eq2:=subs(varslect,tmp51);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

eq2 := 1/2 (((- cos(q1) sin(q2) cos(q3) + sin(q1) sin(q3)) qp6
```

```
 + (sin(q1) sin(q2) cos(q3) + cos(q1) sin(q3)) qp5 + cos(q2) qp4 cos(q3))^2 + (
(cos(q1) sin(q2) sin(q3) + sin(q1) cos(q3)) qp6
 + (- sin(q1) sin(q2) sin(q3) + cos(q1) cos(q3)) qp5 - sin(q3) cos(q2) qp4)^2
                                                                 2
 + (- cos(q2) qp5 sin(q1) + cos(q2) cos(q1) qp6 + qp4 sin(q2)) ) mu1
                                                  2
 + 1/2 i1X (cos(q2) cos(q3) qp1 + sin(q3) qp2)
                                                    2
 + 1/2 i1Y (- cos(q2) sin(q3) qp1 + cos(q3) qp2)
                                  2
 + 1/2 i1Z (sin(q2) qp1 + qp3)  + 1/2 ((
(- cos(q1) sin(q2) cos(q3) + sin(q1) sin(q3)) qp6
 + (sin(q1) sin(q2) cos(q3) + cos(q1) sin(q3)) qp5 + cos(q2) qp4 cos(q3)
 + cos(q9) qp9 z34 +
(- cos(q2) sin(q3) qp1 + cos(q3) qp2) (z12 + sin(q7) q8 + cos(q7) cos(q9) z34)
 - (sin(q2) qp1 + qp3) (cos(q7) q8 - sin(q7) cos(q9) z34))^2 + (
(cos(q1) sin(q2) sin(q3) + sin(q1) cos(q3)) qp6
 + (- sin(q1) sin(q2) sin(q3) + cos(q1) cos(q3)) qp5 - sin(q3) cos(q2) qp4
 - sin(q7) qp7 q8 + cos(q7) qp8 - cos(q7) qp7 cos(q9) z34
 + sin(q7) sin(q9) qp9 z34 + (sin(q2) qp1 + qp3) sin(q9) z34
 - (cos(q2) cos(q3) qp1 + sin(q3) qp2) (z12 + sin(q7) q8 + cos(q7) cos(q9) z34)
)^2 + (- cos(q2) qp5 sin(q1) + cos(q2) cos(q1) qp6 + qp4 sin(q2)
 + cos(q7) qp7 q8 + sin(q7) qp8 - sin(q7) qp7 cos(q9) z34
 - cos(q7) sin(q9) qp9 z34
 + (cos(q2) cos(q3) qp1 + sin(q3) qp2) (cos(q7) q8 - sin(q7) cos(q9) z34)
 - (- cos(q2) sin(q3) qp1 + cos(q3) qp2) sin(q9) z34)^2) mu4
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
> tmp55:=dotprod(f1,v1)+dotprod(add(c1,scalarmul(c12,-1)),om1):
> tmp56:=-dotprod(multiply(Rot12,f),v12)+dotprod(c12,om2):
> tmp57:=dotprod(multiply(Rot12,f),v34)-dotprod(multiply(Rot12,c34),om2):
> tmp58:=dotprod(multiply(Rot12,c34),om4)+dotprod(f4,v4):
> tmp59:=tmp55+tmp56+tmp57+tmp58:
#####

> eq3:=subs(varslect,tmp59);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

eq3 := f1X vX + f1Y vY + f1Z vZ + (c1X - C12) omX + c1Y omY + c1Z omZ
     - cos(q7) F23 (vY - omX z12) - sin(q7) F23 vZ + C12 (omX + qp7)
     + cos(q7) F23 (vY - sin(q7) qp7 q8 + cos(q7) qp8 - omX (z12 + sin(q7) q8))
     + sin(q7) F23 (vZ + cos(q7) qp7 q8 + sin(q7) qp8 + omX cos(q7) q8)
```

```
     - cos(q7) C34 omY - sin(q7) C34 omZ + cos(q7) C34 (omY + cos(q7) qp9)

     + sin(q7) C34 (omZ + sin(q7) qp9) + f4X (vX + cos(q9) qp9 z34

     + omY (z12 + sin(q7) q8 + cos(q7) cos(q9) z34)

     - omZ (cos(q7) q8 - sin(q7) cos(q9) z34)) + f4Y (vY - sin(q7) qp7 q8

     + cos(q7) qp8 - cos(q7) qp7 cos(q9) z34 + sin(q7) sin(q9) qp9 z34

     + omZ sin(q9) z34 - omX (z12 + sin(q7) q8 + cos(q7) cos(q9) z34)) + f4Z (

    vZ + cos(q7) qp7 q8 + sin(q7) qp8 - sin(q7) qp7 cos(q9) z34

     - cos(q7) sin(q9) qp9 z34 + omX (cos(q7) q8 - sin(q7) cos(q9) z34)

     - omY sin(q9) z34)

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp60:=subs(tmp25 union tmp32,tmp59):
> tmp61:=
>   subs({qp1=diff(q1(t),t),qp2=diff(q2(t),t),qp3=diff(q3(t),t),
>         qp4=diff(q4(t),t),qp5=diff(q5(t),t),qp6=diff(q6(t),t)},tmp60):
> tmp62:=subs(varslect,simplify(tmp61,trig)):
#####
> eq4:=subs(varslect,tmp61);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

eq4 := f1X ((- cos(q1) sin(q2) cos(q3) + sin(q1) sin(q3)) qp6

     + (sin(q1) sin(q2) cos(q3) + cos(q1) sin(q3)) qp5 + cos(q2) qp4 cos(q3))

     + f1Y (%7 + %6 - %5) + f1Z (- %4 + %3 + qp4 sin(q2))

     + (c1X - C12) (%2 + sin(q3) qp2) + c1Y (- %1 + cos(q3) qp2)

     + c1Z (sin(q2) qp1 + qp3)

     - cos(q7) F23 (%7 + %6 - %5 - (%2 + sin(q3) qp2) z12)

     - sin(q7) F23 (- %4 + %3 + qp4 sin(q2)) + C12 (%2 + sin(q3) qp2 + qp7) +

    cos(q7) F23 (%7 + %6 - %5 - sin(q7) qp7 q8 + cos(q7) qp8

     - (%2 + sin(q3) qp2) (z12 + sin(q7) q8)) + sin(q7) F23 (- %4 + %3

     + qp4 sin(q2) + cos(q7) qp7 q8 + sin(q7) qp8

     + (%2 + sin(q3) qp2) cos(q7) q8) - cos(q7) C34 (- %1 + cos(q3) qp2)

     - sin(q7) C34 (sin(q2) qp1 + qp3)

     + cos(q7) C34 (- %1 + cos(q3) qp2 + cos(q7) qp9)

     + sin(q7) C34 (sin(q2) qp1 + qp3 + sin(q7) qp9) + f4X (

    (- cos(q1) sin(q2) cos(q3) + sin(q1) sin(q3)) qp6

     + (sin(q1) sin(q2) cos(q3) + cos(q1) sin(q3)) qp5 + cos(q2) qp4 cos(q3)

     + cos(q9) qp9 z34

     + (- %1 + cos(q3) qp2) (z12 + sin(q7) q8 + cos(q7) cos(q9) z34)

     - (sin(q2) qp1 + qp3) (cos(q7) q8 - sin(q7) cos(q9) z34)) + f4Y (%7 + %6

     - %5 - sin(q7) qp7 q8 + cos(q7) qp8 - cos(q7) qp7 cos(q9) z34
```

```
     + sin(q7) sin(q9) qp9 z34 + (sin(q2) qp1 + qp3) sin(q9) z34

     - (%2 + sin(q3) qp2) (z12 + sin(q7) q8 + cos(q7) cos(q9) z34)) + f4Z (- %4

     + %3 + qp4 sin(q2) + cos(q7) qp7 q8 + sin(q7) qp8

     - sin(q7) qp7 cos(q9) z34 - cos(q7) sin(q9) qp9 z34

     + (%2 + sin(q3) qp2) (cos(q7) q8 - sin(q7) cos(q9) z34)

     - (- %1 + cos(q3) qp2) sin(q9) z34)

%1 :=                        cos(q2) sin(q3) qp1

%2 :=                        cos(q2) cos(q3) qp1

%3 :=                        cos(q2) cos(q1) qp6

%4 :=                        cos(q2) qp5 sin(q1)

%5 :=                        sin(q3) cos(q2) qp4

%6 :=         (- sin(q1) sin(q2) sin(q3) + cos(q1) cos(q3)) qp5

%7 :=          (cos(q1) sin(q2) sin(q3) + sin(q1) cos(q3)) qp6

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp65:=subs({qp2=0,qp3=0,qp4=0,qp5=0,qp6=0,qp7=0,qp8=0,qp9=0},tmp62):
#####
> Q1:=collect(coeff(collect(tmp65,qp1),qp1),[f4X,f4Y,f4Z]);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

Q1 := (- sin(q2) cos(q7) q8 - cos(q2) sin(q3) sin(q7) q8

     + sin(q2) sin(q7) cos(q9) z34 - cos(q2) sin(q3) cos(q7) cos(q9) z34

     - cos(q2) sin(q3) z12) f4X + (sin(q9) z34 sin(q2) - z12 cos(q2) cos(q3)

     - cos(q2) cos(q3) sin(q7) q8 - cos(q2) cos(q3) cos(q7) cos(q9) z34) f4Y +

    (- cos(q2) cos(q3) sin(q7) cos(q9) z34 + sin(q9) z34 cos(q2) sin(q3)

     + cos(q7) q8 cos(q2) cos(q3)) f4Z + c1X cos(q2) cos(q3)

     - c1Y cos(q2) sin(q3) + c1Z sin(q2)

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp66:=subs({qp1=0,qp3=0,qp4=0,qp5=0,qp6=0,qp7=0,qp8=0,qp9=0},tmp62):
#####

> Q2:=collect(coeff(collect(tmp66,qp2),qp2),[f4X,f4Y,f4Z]);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

Q2 := (cos(q3) z12 + cos(q3) sin(q7) q8 + cos(q3) cos(q7) cos(q9) z34) f4X

 + (- sin(q3) sin(q7) q8 - sin(q3) cos(q7) cos(q9) z34 - z12 sin(q3)) f4Y

 + (cos(q7) q8 sin(q3) - sin(q3) sin(q7) cos(q9) z34 - sin(q9) z34 cos(q3)) f4Z

 + c1Y cos(q3) + c1X sin(q3)

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp67:=subs({qp1=0,qp2=0,qp4=0,qp5=0,qp6=0,qp7=0,qp8=0,qp9=0},tmp62):
#####
> Q3:=collect(coeff(collect(tmp67,qp3),qp3),[f4X,f4Y,f4Z]);
```

```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

      Q3 := (sin(q7) cos(q9) z34 - cos(q7) q8) f4X + c1Z + f4Y sin(q9) z34

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp68:=
>   subs({qp1=0,qp2=0,qp3=0,qp5=0,qp6=0,qp7=0,qp8=0,qp9=0},tmp62):
#####
> Q4:=collect(coeff(collect(tmp68,qp4),qp4),[f4X,f4Y,f4Z]);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

        Q4 := - f4Y sin(q3) cos(q2) + f4Z sin(q2) + f4X cos(q2) cos(q3)

             + f1X cos(q2) cos(q3) - f1Y sin(q3) cos(q2) + f1Z sin(q2)

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp69:=subs({qp1=0,qp2=0,qp3=0,qp4=0,qp6=0,qp7=0,qp8=0,qp9=0},tmp62):
#####
> Q5:=collect(coeff(collect(tmp69,qp5),qp5),[f4X,f4Y,f4Z]);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

 Q5 := (sin(q1) sin(q2) cos(q3) + cos(q1) sin(q3)) f4X

      + (- sin(q1) sin(q2) sin(q3) + cos(q1) cos(q3)) f4Y - f4Z cos(q2) sin(q1)

      - f1Z cos(q2) sin(q1) + f1X cos(q1) sin(q3) - f1Y sin(q1) sin(q2) sin(q3)

      + f1Y cos(q1) cos(q3) + f1X sin(q1) sin(q2) cos(q3)

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp70:=subs({qp1=0,qp2=0,qp3=0,qp4=0,qp5=0,qp7=0,qp8=0,qp9=0},tmp62):
#####
> Q6:=collect(coeff(collect(tmp70,qp6),qp6),[f4X,f4Y,f4Z]);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

 Q6 := (- cos(q1) sin(q2) cos(q3) + sin(q1) sin(q3)) f4X

      + (cos(q1) sin(q2) sin(q3) + sin(q1) cos(q3)) f4Y + f4Z cos(q2) cos(q1)

      - f1X cos(q1) sin(q2) cos(q3) + f1X sin(q1) sin(q3) + f1Z cos(q2) cos(q1)

      + f1Y sin(q1) cos(q3) + f1Y cos(q1) sin(q2) sin(q3)

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp71:=subs({qp1=0,qp2=0,qp3=0,qp4=0,qp5=0,qp6=0,qp8=0,qp9=0},tmp62):
#####
> Q7:=collect(coeff(collect(tmp71,qp7),qp7),[f4X,f4Y,f4Z]);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

              Q7 := (- sin(q7) q8 - cos(q7) cos(q9) z34) f4Y

                   + (cos(q7) q8 - sin(q7) cos(q9) z34) f4Z + C12

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp72:=subs({qp1=0,qp2=0,qp3=0,qp4=0,qp5=0,qp6=0,qp7=0,qp9=0},tmp62):
#####
> Q8:=collect(coeff(collect(tmp72,qp8),qp8),[f4X,f4Y,f4Z]);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

```
                    Q8 := F23 + f4Y cos(q7) + f4Z sin(q7)

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp73:=subs({qp1=0,qp2=0,qp3=0,qp4=0,qp5=0,qp6=0,qp7=0,qp8=0},tmp62):
#####
> Q9:=collect(coeff(collect(tmp73,qp9),qp9),[f4X,f4Y,f4Z]);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

Q9 :=

    - f4Z cos(q7) sin(q9) z34 + f4Y sin(q7) sin(q9) z34 + C34 + f4X cos(q9) z34

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> varstqpp:=[qpp1=diff(diff(q1(t),t),t),qpp2=diff(diff(q2(t),t),t),
>            qpp3=diff(diff(q3(t),t),t),qpp4=diff(diff(q4(t),t),t),
>            qpp5=diff(diff(q5(t),t),t),qpp6=diff(diff(q6(t),t),t),
>            qpp7=diff(diff(q7(t),t),t),qpp8=diff(diff(q8(t),t),t),
>            qpp9=diff(diff(q9(t),t),t)]:
> varstqp:=[qp1=diff(q1(t),t),qp2=diff(q2(t),t),qp3=diff(q3(t),t),
>           qp4=diff(q4(t),t),qp5=diff(q5(t),t),qp6=diff(q6(t),t),
>           qp7=diff(q7(t),t),qp8=diff(q8(t),t),qp9=diff(q9(t),t)]:
> varstq:=[q1=q1(t),q2=q2(t),q3=q3(t),q4=q4(t),q5=q5(t),q6=q6(t),
>          q7=q7(t),q8=q8(t),q9=q9(t)]:
> varstomvpp:=[omXpp=diff(diff(omX(t),t),t),omYpp=diff(diff(omY(t),t),t),
>              omZpp=diff(diff(omZ(t),t),t),
>              vXpp=diff(diff(vX(t),t),t),vYpp=diff(diff(vY(t),t),t),
>              vZpp=diff(diff(vZ(t),t),t)]:
> varstomvp:=[omXp=diff(omX(t),t),omYp=diff(omY(t),t),omZp=diff(omZ(t),t),
>            vXp=diff(vX(t),t),vYp=diff(vY(t),t),vZp=diff(vZ(t),t)]:
> varstomv:=[omX=omX(t),omY=omY(t),omZ=omZ(t),vX=vX(t),vY=vY(t),vZ=vZ(t)]:
#####
>
> tmp90:=diff(eq2,qp1):
> tmp91:=subs(varstq,tmp90):
> tmp92:=subs(tmp17 union convert(tmp21,set),tmp91):
> tmp93:=simplify(subs(varstqp,tmp92),trig):
> tmp94:=subs(varslect,tmp93):
#####
> dKdqp1:=
>   rlms(clct(tmp94,
>             [qpp7,qpp8,qpp9],
>             [omXp,omYp,omZp,vXp,vYp,vZp,qp1,qp2,qp3,qp4,qp5,qp6,qp7,qp8,qp9],
>             [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4])):
#####
> dK2dqp1dt:=subs(varslect,diff(tmp93,t)):
#####
> tmp96:=diff(eq2,q1):
> tmp97:=subs(varstq,tmp96):
> tmp98:=subs(tmp17 union convert(tmp21,set),tmp97):
> tmp99:=simplify(tmp98,trig):
#####
> dKdq1:=
>   rlms(clct(subs(varslect,tmp99),
>             [qpp7,qpp8,qpp9],
>             [omXp,omYp,omZp,vXp,vYp,vZp,qp1,qp2,qp3,qp4,qp5,qp6,qp7,qp8,qp9],
>             [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4])):
#####
> K1:=
>   rlms(clct(simplify(dK2dqp1dt-dKdq1,trig),
>             [qpp7,qpp8,qpp9],
>             [omXp,omYp,omZp,vXp,vYp,vZp,qp1,qp2,qp3,qp4,qp5,qp6,qp7,qp8,qp9],
>             [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4],factor));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

K1 := (i1X cos(q2) cos(q3) + (2 sin(q7) q8 cos(q2) cos(q3) z12
```

```
 + 2 cos(q7) cos(q9) z34 cos(q2) cos(q3) z12

 - sin(q7) q8 sin(q9) z34 sin(q2)

                                2                          2
 - cos(q2) sin(q7) cos(q9) z34  sin(q9) sin(q3) + z12  cos(q2) cos(q3)

                                 2
 - z12 sin(q9) z34 sin(q2) + q8  cos(q2) cos(q3)

 + cos(q2) cos(q7) q8 sin(q9) z34 sin(q3)

                      2                                      2    2
 - cos(q7) cos(q9) z34  sin(q9) sin(q2) + cos(q2) cos(q9)  z34  cos(q3))

                                                                    2
mu4) omXp + (i1Z sin(q2) + (cos(q7) q8 sin(q3) cos(q2) z12 + z34  sin(q2)

            2
 + 2 cos(q7)  q8 sin(q3) cos(q2) cos(q9) z34

 - 2 sin(q7) cos(q9) z34 sin(q2) cos(q7) q8

          2    2                2
 - cos(q9)  z34  sin(q2) cos(q7)  - sin(q7) cos(q9) z34 sin(q3) cos(q2) z12

 - cos(q9) z34 sin(q3) cos(q2) q8 - sin(q9) z34 cos(q2) cos(q3) sin(q7) q8

          2   2                     2
 + cos(q7)  q8  sin(q2) + cos(q7) q8  sin(q3) cos(q2) sin(q7)

              2
 - sin(q9) z34  cos(q2) cos(q3) cos(q7) cos(q9)

                 2    2
 - sin(q7) cos(q9)  z34  sin(q3) cos(q2) cos(q7)

 - sin(q9) z34 cos(q2) cos(q3) z12) mu4) omZp + ((cos(q2) sin(q3) z12

 + cos(q2) sin(q3) cos(q7) cos(q9) z34 + cos(q2) sin(q3) sin(q7) q8) mu4 vY

 + (sin(q9) z34 cos(q3) cos(q2) - cos(q7) q8 sin(q3) cos(q2)

 + sin(q7) cos(q9) z34 sin(q3) cos(q2)) mu4 vZ - (

cos(q2) cos(q3) sin(q7) q8 + cos(q2) cos(q3) cos(q7) cos(q9) z34

 + z12 cos(q2) cos(q3)) mu4 vX - (cos(q9) z34 cos(q2) cos(q3) q8

                                2
 - cos(q2) cos(q7) cos(q9) z34  sin(q9) sin(q3)

            2
 - 2 cos(q7)  q8 cos(q3) cos(q2) cos(q9) z34

             2
 - cos(q7) q8  cos(q3) cos(q2) sin(q7) - cos(q7) q8 cos(q2) cos(q3) z12

                      2    2
 - %2 + sin(q7) cos(q9)  z34  cos(q3) cos(q2) cos(q7) + %1

 - sin(q9) z34 sin(q3) cos(q2) z12) mu4 omZ - (i1Y cos(q3) cos(q2) + (

                                                  2
2 cos(q7) cos(q9) z34 cos(q2) cos(q3) z12 + q8  cos(q2) cos(q3)

 + 2 sin(q7) q8 cos(q2) cos(q7) cos(q9) z34 cos(q3)

     2                        2
 - q8  cos(q2) cos(q3) cos(q7)  - cos(q2) cos(q7) q8 sin(q9) z34 sin(q3)
```

```
                2    2                     2
 - cos(q2) cos(q9)  z34  cos(q3) + cos(q2) z34  cos(q3)

         2        2    2                      2
 + cos(q7)  cos(q9)  z34  cos(q2) cos(q3) + z12  cos(q2) cos(q3)

                                 2
 + cos(q2) sin(q7) cos(q9) z34  sin(q9) sin(q3)

 + 2 sin(q7) q8 cos(q2) cos(q3) z12) mu4) omY - (i1X cos(q2) sin(q3)

             2
 + (2 %8 + z12  cos(q2) sin(q3) + %3 - %6 + %5 + %7 + 2 %4) mu4) omX) qp3 -

 (sin(q2) sin(q7) sin(q9) z34 sin(q3) - cos(q7) sin(q9) z34 cos(q2)

 - sin(q2) cos(q3) cos(q9) z34 - cos(q7) sin(q2) cos(q3) z12) mu4 qp8 qp2

 - (%5 + %3 + %8 + %7 + %4 - %6) mu4 qp3 qp7 - (

cos(q7) cos(q9) z34 sin(q2) cos(q3) z12 + sin(q7) q8 sin(q9) z34 cos(q2)

                       2
 + cos(q7) cos(q9) z34  sin(q9) cos(q2)

                                 2
 - sin(q2) sin(q7) cos(q9) z34  sin(q9) sin(q3)

 + sin(q2) cos(q7) q8 sin(q9) z34 sin(q3) + sin(q7) q8 sin(q2) cos(q3) z12

                2    2             2
 + sin(q2) cos(q9)  z34  cos(q3) + q8  sin(q2) cos(q3)) mu4 qp2 qp7 - (

2 sin(q2) sin(q7) sin(q9) z34 - 2 sin(q7) cos(q2) cos(q3) z12

 - 2 cos(q2) cos(q7) sin(q9) z34 sin(q3) - 2 q8 cos(q2) cos(q3)) mu4 qp8

           2                         2
qp7 + ((2 q8  cos(q2) sin(q3) cos(q7)  + 2 q8 sin(q2) cos(q9) z34

                   2    2
 + 2 cos(q7) cos(q9)  z34  sin(q2) sin(q7) + %5 - %3

 - 4 sin(q7) q8 cos(q2) cos(q7) cos(q9) z34 sin(q3) + %7

              2                                    2
 - 2 sin(q7) q8  sin(q2) cos(q7) - %4 - 4 cos(q7)  cos(q9) z34 sin(q2) q8

           2        2    2
 - 2 cos(q7)  cos(q9)  z34  cos(q2) sin(q3) - %8 - %6) mu4 omZ + (

z12 sin(q2) cos(q7) cos(q9) z34 + 2 cos(q9) z34 sin(q3) cos(q2) q8

           2    2                2
 + 2 cos(q9)  z34  sin(q2) cos(q7)

                                                      2    2
 + 4 sin(q7) cos(q9) z34 sin(q2) cos(q7) q8 - cos(q9)  z34  sin(q2)

 - 2 cos(q7) q8 sin(q3) cos(q2) z12

             2                                          2   2
 + sin(q9) z34  cos(q2) cos(q3) cos(q7) cos(q9) - 2 cos(q7)  q8  sin(q2)

 + z12 sin(q2) sin(q7) q8 + sin(q9) z34 cos(q2) cos(q3) sin(q7) q8

              2                                2
 - 2 cos(q7) q8  sin(q3) cos(q2) sin(q7) + q8  sin(q2)
```

```
          2
 - 4 cos(q7)  q8 sin(q3) cos(q2) cos(q9) z34

 + 2 sin(q7) cos(q9) z34 sin(q3) cos(q2) z12

                    2    2
 + 2 sin(q7) cos(q9)  z34  sin(q3) cos(q2) cos(q7)) mu4 omY - (%2

 + cos(q7) q8 sin(q9) z34 sin(q2)

                               2
 + cos(q2) cos(q7) cos(q9) z34  sin(q9) sin(q3)

 - 2 cos(q7) q8 cos(q2) cos(q3) z12 + 2 %1

                      2
 - sin(q7) cos(q9) z34  sin(q9) sin(q2)) mu4 omX) qp7 - (

i1Y sin(q3) cos(q2) - (q8 sin(q2) cos(q9) z34

          2        2    2
 - cos(q7)  cos(q9)  z34  cos(q2) sin(q3) - 2 %8

            2                                 2
 - sin(q7) q8  sin(q2) cos(q7) - 2 cos(q7)  cos(q9) z34 sin(q2) q8

    2                         2
 + q8  cos(q2) sin(q3) cos(q7)  + %7 - z12 sin(q2) cos(q7) q8 - %6

              2                           2
 - cos(q2) z34  sin(q3) + %5 - 2 %4 - z12  cos(q2) sin(q3) - %3

 + z12 sin(q2) sin(q7) cos(q9) z34

 - 2 sin(q7) q8 cos(q2) cos(q7) cos(q9) z34 sin(q3)

                  2    2
 + cos(q7) cos(q9)  z34  sin(q2) sin(q7)) mu4) omYp - (

sin(q7) sin(q9) z34 cos(q2) cos(q3) z12

                                                                  2
 + cos(q9) z34 sin(q3) cos(q2) sin(q7) q8 + cos(q2) cos(q7) z34  sin(q3)

 + sin(q9) z34 cos(q2) cos(q3) q8 + cos(q9) z34 sin(q3) cos(q2) z12

                                                  2
 + cos(q9) z34 sin(q2) cos(q7) q8 - sin(q7) z34  sin(q2)) mu4 qpp9 + ((

                                                       2
sin(q9) z34 cos(q2) sin(q3) q8 + cos(q2) cos(q7) z34  cos(q3)

          2
 - 2 cos(q7)  q8 sin(q3) cos(q2) sin(q9) z34

          2    2
 - 2 cos(q9)  z34  cos(q2) cos(q3) cos(q7)

 + sin(q7) sin(q9) z34 cos(q2) sin(q3) z12

                       2
 + 2 sin(q7) cos(q9) z34  sin(q3) cos(q2) cos(q7) sin(q9)

 - cos(q9) z34 cos(q3) cos(q2) sin(q7) q8

              2                2
 + 2 cos(q9) z34  sin(q2) cos(q7)  sin(q9)

 + 2 sin(q7) sin(q9) z34 sin(q2) cos(q7) q8
```

```
 - cos(q9) z34 cos(q3) cos(q2) z12) mu4 omZ + (

         2            2
2 cos(q7)  cos(q9) z34  cos(q2) sin(q3) sin(q9)

 + 2 z12 cos(q2) cos(q7) sin(q9) z34 sin(q3)

                     2    2
 + 2 cos(q2) cos(q9)  z34  cos(q3) sin(q7)

 - z12 sin(q2) sin(q7) sin(q9) z34 - q8 sin(q2) sin(q9) z34

                        2
 - 2 cos(q7) cos(q9) z34  sin(q2) sin(q7) sin(q9)

 + 2 sin(q7) q8 cos(q2) cos(q7) sin(q9) z34 sin(q3)

           2                                        2
 + 2 cos(q7)  sin(q9) z34 sin(q2) q8 - cos(q2) z34  cos(q3) sin(q7)

                2
 - 2 cos(q2) z34  cos(q9) sin(q3) sin(q9)

 - cos(q2) cos(q9) z34 cos(q3) cos(q7) q8) mu4 omY + (

                   2
cos(q2) sin(q7) z34  sin(q3) - z12 cos(q9) z34 sin(q2)

                     2    2
 - 2 cos(q7) cos(q9)  z34  sin(q2) - sin(q7) q8 cos(q9) z34 sin(q2)

 - 2 cos(q7) sin(q9) z34 cos(q2) cos(q3) z12

 + cos(q2) cos(q7) q8 cos(q9) z34 sin(q3)

                        2
 - 2 cos(q2) cos(q9) z34  cos(q3) sin(q9)

                               2    2                        2
 - 2 cos(q2) sin(q7) cos(q9)  z34  sin(q3) + cos(q7) z34  sin(q2)) mu4 omX)

                              2
 qp9 - (2 cos(q2) cos(q9) z34  cos(q3) sin(q9)

                         2                               2    2
 - 2 cos(q2) sin(q7) z34  sin(q3) + 2 cos(q7) cos(q9)  z34  sin(q2)

 + 2 cos(q7) sin(q9) z34 cos(q2) cos(q3) z12

                               2    2                         2
 + 2 cos(q2) sin(q7) cos(q9)  z34  sin(q3) - 2 cos(q7) z34  sin(q2)) mu4

qp9 qp7 + (cos(q2) sin(q7) sin(q9) z34 sin(q3)

 - cos(q7) cos(q2) cos(q3) z12 - cos(q2) cos(q3) cos(q9) z34

                                                          2    2
 + cos(q7) sin(q9) z34 sin(q2)) mu4 qpp8 + (cos(q2) cos(q9)  z34  cos(q3)

 + cos(q7) cos(q9) z34 cos(q2) cos(q3) z12 + sin(q7) q8 cos(q2) cos(q3) z12

                                  2
 - cos(q2) sin(q7) cos(q9) z34  sin(q9) sin(q3)

                                                       2
 - sin(q7) q8 sin(q9) z34 sin(q2) - cos(q7) cos(q9) z34  sin(q9) sin(q2)

                                                 2
 + cos(q2) cos(q7) q8 sin(q9) z34 sin(q3) + q8  cos(q2) cos(q3)) mu4 qpp7
```

```
 + (sin(q9) z34 sin(q3) cos(q2) z12 + %2 + cos(q7) q8 sin(q9) z34 sin(q2)
                                                          2
 - cos(q9) z34 cos(q2) cos(q3) q8 - %1) mu4 qp9  + (
sin(q9) z34 cos(q2) sin(q3) q8 - cos(q9) z34 cos(q3) cos(q2) sin(q7) q8
                       2
 - cos(q2) cos(q7) z34  cos(q3) + sin(q7) sin(q9) z34 cos(q2) sin(q3) z12
 - cos(q9) z34 cos(q3) cos(q2) z12) mu4 qp3 qp9 + (
                                                   2
cos(q9) z34 sin(q3) sin(q2) z12 + sin(q7) z34  cos(q2)
                                                        2
 - cos(q9) z34 cos(q2) cos(q7) q8 + sin(q2) cos(q7) z34  sin(q3)
 + sin(q7) sin(q9) z34 sin(q2) cos(q3) z12 + sin(q9) z34 sin(q2) cos(q3) q8
 + cos(q9) z34 sin(q3) sin(q2) sin(q7) q8) mu4 qp2 qp9 + (
(cos(q7) q8 cos(q2) cos(q3) - cos(q2) cos(q3) sin(q7) cos(q9) z34) mu4 vY
 + (cos(q2) cos(q3) sin(q7) q8 + cos(q2) cos(q3) cos(q7) cos(q9) z34
 + z12 cos(q2) cos(q3)) mu4 vZ + (cos(q7) q8 sin(q3) cos(q2)
 - sin(q7) cos(q9) z34 sin(q3) cos(q2) - z12 sin(q2) - sin(q7) q8 sin(q2)
 - cos(q7) cos(q9) z34 sin(q2)) mu4 vX) omX - ((
                                                  2
cos(q3) cos(q2) sin(q7) sin(q9) z34 - 2 cos(q7)  q8 sin(q2)
 - 2 cos(q7) q8 sin(q3) cos(q2) sin(q7) + sin(q3) cos(q2) cos(q9) z34
           2
 - 2 cos(q7)  sin(q3) cos(q2) cos(q9) z34
 + 2 sin(q7) cos(q9) z34 sin(q2) cos(q7) - cos(q7) cos(q2) sin(q3) z12) mu4
                                      2
 omZ - (2 q8 cos(q2) sin(q3) cos(q7)
 - 2 sin(q7) cos(q2) cos(q7) cos(q9) z34 sin(q3)
                                             2
 - 2 sin(q7) cos(q2) z12 sin(q3) - 2 cos(q7)  cos(q9) z34 sin(q2)
 - z12 sin(q2) cos(q7) - 2 q8 cos(q2) sin(q3)
 - 2 sin(q7) q8 sin(q2) cos(q7) - cos(q3) cos(q2) cos(q7) sin(q9) z34
 + cos(q9) z34 sin(q2)) mu4 omY - (2 sin(q7) cos(q2) cos(q3) z12
 + 2 q8 cos(q2) cos(q3) - sin(q2) sin(q7) sin(q9) z34
 + cos(q2) cos(q7) sin(q9) z34 sin(q3)) mu4 omX) qp8 + (
sin(q3) cos(q2) cos(q9) z34 + cos(q3) cos(q2) sin(q7) sin(q9) z34
 + cos(q7) cos(q2) sin(q3) z12) mu4 qp3 qp8 + (sin(q9) z34 sin(q2)
 - z12 cos(q2) cos(q3) - cos(q2) cos(q3) sin(q7) q8
 - cos(q2) cos(q3) cos(q7) cos(q9) z34) mu4 vYp - (%2 + %1
 - cos(q7) q8 cos(q2) cos(q3) z12 + cos(q7) q8 sin(q9) z34 sin(q2)
```

```
                                2
 + cos(q2) cos(q7) cos(q9) z34  sin(q9) sin(q3)

                      2                          2
 - sin(q7) cos(q9) z34  sin(q9) sin(q2)) mu4 qp7  + ((sin(q9) z34 cos(q2)

 + cos(q3) sin(q2) z12 + cos(q3) sin(q2) cos(q7) cos(q9) z34

 + cos(q3) sin(q2) sin(q7) q8) mu4 vY - (cos(q3) sin(q2) cos(q7) q8

 + sin(q3) sin(q2) sin(q9) z34 - cos(q3) sin(q2) sin(q7) cos(q9) z34) mu4

vZ + (sin(q2) sin(q7) q8 sin(q3) + sin(q2) cos(q7) cos(q9) z34 sin(q3)

 - cos(q2) cos(q7) q8 + sin(q2) z12 sin(q3) + cos(q2) sin(q7) cos(q9) z34)

mu4 vX + (i1Z cos(q2) + (sin(q9) z34 sin(q2) cos(q3) sin(q7) q8

          2   2                            2    2
 + cos(q7)  q8  cos(q2) + sin(q7) cos(q9)  z34  sin(q3) sin(q2) cos(q7)

 + cos(q9) z34 sin(q3) sin(q2) q8 + sin(q7) cos(q9) z34 sin(q3) sin(q2) z12

      2                  2    2                2
 + z34  cos(q2) - cos(q9)  z34  cos(q2) cos(q7)

            2
 - cos(q7) q8  sin(q3) sin(q2) sin(q7) - cos(q7) q8 sin(q3) sin(q2) z12

 + sin(q9) z34 sin(q2) cos(q3) z12

             2
 + sin(q9) z34  sin(q2) cos(q3) cos(q7) cos(q9)

           2
 - 2 cos(q7)  q8 sin(q3) sin(q2) cos(q9) z34

 - 2 sin(q7) cos(q9) z34 cos(q2) cos(q7) q8) mu4) omZ + (

                                      2    2
i1Y sin(q3) sin(q2) + (cos(q7) cos(q9)  z34  cos(q2) sin(q7)

 + 2 sin(q7) q8 sin(q2) z12 sin(q3)

 + sin(q2) sin(q9) z34 cos(q3) cos(q7) q8 + q8 cos(q2) cos(q9) z34

 + 2 z12 sin(q2) cos(q7) cos(q9) z34 sin(q3)

                                                 2
 + z12 cos(q2) sin(q7) cos(q9) z34 + sin(q2) z34  sin(q3)

          2        2    2
 + cos(q7)  cos(q9)  z34  sin(q2) sin(q3)

 + 2 sin(q7) q8 sin(q2) cos(q7) cos(q9) z34 sin(q3)

            2                      2
 - sin(q7) q8  cos(q2) cos(q7) + q8  sin(q2) sin(q3)

                     2
 - sin(q2) sin(q9) z34  cos(q3) sin(q7) cos(q9)

     2                        2
 - q8  sin(q2) sin(q3) cos(q7)  - z12 cos(q2) cos(q7) q8

           2                             2
 - 2 cos(q7)  cos(q9) z34 cos(q2) q8 + z12  sin(q2) sin(q3)

             2        2
 - sin(q2) z34  cos(q9)  sin(q3)) mu4) omY - (i1X sin(q2) cos(q3) + (
```

```
                      2
    cos(q7) cos(q9) z34  sin(q9) cos(q2)

                                  2                      2
     - sin(q2) sin(q7) cos(q9) z34  sin(q9) sin(q3) + q8  sin(q2) cos(q3)

          2
     + z12  sin(q2) cos(q3) + 2 cos(q7) cos(q9) z34 sin(q2) cos(q3) z12

     + 2 sin(q7) q8 sin(q2) cos(q3) z12 + z12 sin(q9) z34 cos(q2)

                     2    2
     + sin(q2) cos(q9)  z34  cos(q3) + sin(q2) cos(q7) q8 sin(q9) z34 sin(q3)

     + sin(q7) q8 sin(q9) z34 cos(q2)) mu4) omX) qp2 - (sin(q2) cos(q7) q8

     + cos(q2) sin(q3) sin(q7) q8 - sin(q2) sin(q7) cos(q9) z34

     + cos(q2) sin(q3) cos(q7) cos(q9) z34 + cos(q2) sin(q3) z12) mu4 vXp - ((

    cos(q7) cos(q9) z34 sin(q2) + z12 sin(q2) + sin(q9) z34 cos(q3) cos(q2)

     + sin(q7) q8 sin(q2)) mu4 vY + (cos(q2) sin(q3) z12

     + cos(q2) sin(q3) cos(q7) cos(q9) z34 + cos(q2) sin(q3) sin(q7) q8) mu4 vZ

     + mu4 cos(q2) sin(q9) z34 vX sin(q3)) omY - (

    (sin(q2) sin(q7) cos(q9) z34 - sin(q2) cos(q7) q8) mu4 vY - (

    cos(q7) q8 sin(q3) cos(q2) - sin(q7) cos(q9) z34 sin(q3) cos(q2)

     - sin(q9) z34 cos(q3) cos(q2)) mu4 vZ - mu4 sin(q9) z34 sin(q2) vX) omZ -

    (cos(q2) cos(q3) sin(q7) cos(q9) z34 - sin(q9) z34 cos(q2) sin(q3)

     - cos(q7) q8 cos(q2) cos(q3)) mu4 vZp

%1 :=              sin(q7) cos(q9) z34 cos(q2) cos(q3) z12

%2 :=              cos(q2) sin(q7) q8 sin(q9) z34 sin(q3)

                           2
%3 :=                     q8  cos(q2) sin(q3)

%4 :=              z12 cos(q2) cos(q7) cos(q9) z34 sin(q3)

                                2        2
%5 :=                  cos(q2) z34  cos(q9)  sin(q3)

%6 :=              cos(q2) sin(q9) z34 cos(q3) cos(q7) q8

                                2
%7 :=           cos(q2) sin(q9) z34  cos(q3) sin(q7) cos(q9)

%8 :=                sin(q7) q8 cos(q2) z12 sin(q3)

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp100:=diff(eq2,qp2):
> tmp101:=subs(varstq,tmp100):
> tmp102:=subs(tmp17 union convert(tmp21,set),tmp101):
> tmp103:=simplify(subs(varstqp,tmp102),trig):
> tmp104:=subs(varslect,tmp103):
#####
> dKdqp2:=
>   rlms(clct(tmp104,
>             [qpp7,qpp8,qpp9],
>             [omXp,omYp,omZp,vXp,vYp,vZp,qp1,qp2,qp3,qp4,qp5,qp6,qp7,qp8,qp9],
```

```
>             [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4])):
#####
> dK2dqp2dt:=subs(varslect,diff(tmp103,t)):
#####
> tmp106:=diff(eq2,q2):
> tmp107:=subs(varstq,tmp106):
> tmp108:=subs(tmp17 union convert(tmp21,set),tmp107):
> tmp109:=simplify(tmp108,trig):
#####
> dKdq2:=
>   rlms(clct(subs(varslect,tmp109),
>             [qpp7,qpp8,qpp9],
>             [omXp,omYp,omZp,vXp,vYp,vZp,qp1,qp2,qp3,qp4,qp5,qp6,qp7,qp8,qp9],
>             [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4])):
#####
> K2:=
>   rlms(clct(simplify(dK2dqp2dt-dKdq2,trig),
>             [qpp7,qpp8,qpp9],
>             [omXp,omYp,omZp,vXp,vYp,vZp,qp1,qp2,qp3,qp4,qp5,qp6,qp7,qp8,qp9],
>             [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4],factor));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                    2    2
K2 := - ((cos(q9)  z34  sin(q2) - 2 z12 sin(q2) cos(q7) cos(q9) z34

              2    2                2            2
   - cos(q9)  z34  sin(q2) cos(q7)  + 2 cos(q7)  q8 sin(q3) cos(q2) cos(q9) z34

                                                     2
   - cos(q9) z34 sin(q3) cos(q2) q8 + cos(q7) q8  sin(q3) cos(q2) sin(q7)

   - 2 sin(q7) cos(q9) z34 sin(q2) cos(q7) q8 + %9

                      2    2
   - sin(q7) cos(q9)  z34  sin(q3) cos(q2) cos(q7) - %10

              2         2    2              2
   + cos(q7)  cos(q9)  z34  sin(q2) cos(q3)

                2
   + 2 cos(q3)  sin(q7) q8 sin(q2) cos(q7) cos(q9) z34

              2    2               2      2         2            2
   - cos(q3)  q8  sin(q2) cos(q7)  + z34  cos(q3)  sin(q2) - z34  sin(q2)

              2    2                 2    2               2
   + cos(q7)  q8  sin(q2) - cos(q9)  z34  sin(q2) cos(q3)

                                  2
   + sin(q7) cos(q9) z34  sin(q2) sin(q3) sin(q9) cos(q3)

   - cos(q7) q8 sin(q2) sin(q3) sin(q9) z34 cos(q3)

                                               2
   + 2 cos(q7) cos(q9) z34 sin(q2) cos(q3)  z12

                                  2          2               2
   + 2 sin(q7) q8 sin(q2) cos(q3)  z12 + z12  sin(q2) cos(q3)

         2              2      2
   + q8  sin(q2) cos(q3)  - z12  sin(q2) - 2 z12 sin(q2) sin(q7) q8

                                                                   2
        2                       i1Y sin(q2) - i1Y sin(q2) cos(q3)       2
   - q8  sin(q2)) mu4/cos(q2) - ---------------------------------) omY  + ((
                                              cos(q2)

                                                                              2
  sin(q9) z34 sin(q2) cos(q3) sin(q3) z12 + cos(q7) q8 sin(q2) z12 cos(q3)
```

```
         2       2    2
 - cos(q7)  cos(q9)  z34  cos(q2) sin(q3)

            2                          2            2
 + cos(q7) q8  sin(q2) sin(q7) cos(q3)  - sin(q7) q8  sin(q2) cos(q7)

    2                         2
 + q8  cos(q2) sin(q3) cos(q7)  - z12 sin(q2) cos(q7) q8

                                        2
 - sin(q7) cos(q9) z34 sin(q2) z12 cos(q3)  + q8 sin(q2) cos(q9) z34

                 2    2                        2
 - sin(q7) cos(q9)  z34  sin(q2) cos(q7) cos(q3)

          2                                        2    2
 - 2 cos(q7)  cos(q9) z34 sin(q2) q8 + cos(q7) cos(q9)  z34  sin(q2) sin(q7)

                                  2
 - cos(q9) z34 sin(q2) q8 cos(q3)

                                                                  2
 + sin(q9) z34 sin(q2) cos(q3) sin(q3) sin(q7) q8 + cos(q2) z34  sin(q3)

 - 2 sin(q7) q8 cos(q2) cos(q7) cos(q9) z34 sin(q3)

             2
 + sin(q9) z34  sin(q2) cos(q3) sin(q3) cos(q7) cos(q9)

 + z12 sin(q2) sin(q7) cos(q9) z34

          2                                  2
 + 2 cos(q7)  q8 sin(q2) cos(q9) z34 cos(q3) ) mu4/cos(q2) + i1Z sin(q3)) omY

 omZ + ((2 %8 + %3 + 2 sin(q7) q8 cos(q2) cos(q7) cos(q9) z34 cos(q3)

    2                         2                        2
 - q8  cos(q2) cos(q3) cos(q7)  - %7 - %6 + cos(q2) z34  cos(q3)

         2       2    2                       2
 + cos(q7)  cos(q9)  z34  cos(q2) cos(q3) + z12  cos(q2) cos(q3) + %5 + 2 %4)

 mu4/cos(q2) + i1Y cos(q3)) omYp - ((

                  2
sin(q7) cos(q9) z34  sin(q2) sin(q3) sin(q9) cos(q3)

         2    2               2
 - cos(q9)  z34  sin(q2) cos(q3)  + 2 %10

 - cos(q7) q8 sin(q2) sin(q3) sin(q9) z34 cos(q3) - 2 %9 - 2 %1

                                     2                              2
 - cos(q7) cos(q9) z34 sin(q2) cos(q3)  z12 - sin(q7) q8 sin(q2) cos(q3)  z12

    2               2
 - q8  sin(q2) cos(q3)  - 2 %2) mu4 omX/cos(q2) + (

                  2                       2
sin(q7) cos(q9) z34  sin(q2) sin(q9) cos(q3)

          2
 - 4 cos(q7)  q8 cos(q3) cos(q2) cos(q9) z34

 + 2 sin(q7) cos(q9) z34 cos(q2) cos(q3) z12

         2    2
 + cos(q9)  z34  sin(q2) cos(q3) sin(q3)
```

```
 + sin(q7) q8 sin(q2) cos(q3) sin(q3) z12 - 2 cos(q7) q8 cos(q2) cos(q3) z12

              2
 - 2 cos(q7) q8  cos(q3) cos(q2) sin(q7) + cos(q7) q8 sin(q9) z34 sin(q2)

 + cos(q7) cos(q9) z34 sin(q2) cos(q3) sin(q3) z12

                     2
 - sin(q7) cos(q9) z34  sin(q9) sin(q2) + 2 cos(q9) z34 cos(q2) cos(q3) q8

                   2    2
 + 2 sin(q7) cos(q9)  z34  cos(q3) cos(q2) cos(q7)

     2                                                                  2
 + q8  sin(q2) cos(q3) sin(q3) - cos(q7) q8 sin(q2) sin(q9) z34 cos(q3) ) mu4

 omY/cos(q2) - (%3 + 4 sin(q7) q8 cos(q2) cos(q7) cos(q9) z34 cos(q3) + %4

       2                          2            2        2    2
 - 2 q8  cos(q2) cos(q3) cos(q7)  + 2 cos(q7)  cos(q9)  z34  cos(q2) cos(q3)

 + %8 - %6 + %5 - %7) mu4 omZ/cos(q2)) qp7 + ((

                                      2                     2
2 sin(q7) q8 cos(q2) z12 sin(q3) + z12  cos(q2) sin(q3) + q8  cos(q2) sin(q3)

                                                        2        2
 - cos(q2) sin(q9) z34 cos(q3) cos(q7) q8 + cos(q2) z34  cos(q9)  sin(q3)

                     2
 + cos(q2) sin(q9) z34  cos(q3) sin(q7) cos(q9)

 + 2 z12 cos(q2) cos(q7) cos(q9) z34 sin(q3)) mu4/cos(q2) + i1X sin(q3)) omXp

                     2
 + (cos(q2) cos(q7) z34  cos(q3) - sin(q9) z34 cos(q2) sin(q3) q8

 + cos(q9) z34 cos(q3) cos(q2) sin(q7) q8

 - sin(q7) sin(q9) z34 cos(q2) sin(q3) z12 + cos(q9) z34 cos(q3) cos(q2) z12)

 mu4 qpp9/cos(q2) + (cos(q9) z34 cos(q2) cos(q3) q8

                            2
 - cos(q2) cos(q7) cos(q9) z34  sin(q9) sin(q3)

           2
 - 2 cos(q7)  q8 cos(q3) cos(q2) cos(q9) z34

            2
 - cos(q7) q8  cos(q3) cos(q2) sin(q7) - cos(q7) q8 cos(q2) cos(q3) z12

 - cos(q2) sin(q7) q8 sin(q9) z34 sin(q3)

                 2    2
 + sin(q7) cos(q9)  z34  cos(q3) cos(q2) cos(q7)

 + sin(q7) cos(q9) z34 cos(q2) cos(q3) z12 - sin(q9) z34 sin(q3) cos(q2) z12)

 mu4 omZp/cos(q2) + (cos(q7) q8 sin(q3) cos(q2)

 - sin(q7) cos(q9) z34 sin(q3) cos(q2) - sin(q9) z34 cos(q3) cos(q2)) mu4 vZp

/cos(q2) - (cos(q2) sin(q3) z12 + cos(q2) sin(q3) cos(q7) cos(q9) z34

 + cos(q2) sin(q3) sin(q7) q8) mu4 vYp/cos(q2) + ((

sin(q7) cos(q9) z34 sin(q3) cos(q2) + z12 sin(q2)

 - cos(q7) q8 sin(q3) cos(q2) + sin(q9) z34 cos(q3) cos(q2)
```

```
                 2
 - cos(q3)  sin(q2) cos(q7) cos(q9) z34 + cos(q7) cos(q9) z34 sin(q2)

                                      2                      2
 + sin(q7) q8 sin(q2) - cos(q3)  sin(q2) z12 - cos(q3)  sin(q2) sin(q7) q8)

mu4 vX/cos(q2) + (sin(q2) cos(q3) cos(q7) cos(q9) z34 sin(q3)

 + sin(q2) cos(q3) z12 sin(q3) + sin(q2) cos(q3) sin(q7) q8 sin(q3)) mu4 vY/

cos(q2) + (cos(q2) cos(q3) cos(q7) cos(q9) z34 - sin(q9) z34 sin(q2)

 + z12 cos(q2) cos(q3) + cos(q3) sin(q3) sin(q7) cos(q9) z34 sin(q2)

 + cos(q2) cos(q3) sin(q7) q8 - cos(q3) sin(q3) cos(q7) q8 sin(q2)

                                     2
 + sin(q9) z34 sin(q2) cos(q3)  ) mu4 vZ/cos(q2)) omY - (

             2                                     2
2 cos(q2) z34  cos(q9) sin(q3) sin(q9) + 2 cos(q2) z34  cos(q3) sin(q7)

 + 2 z12 cos(q2) cos(q7) sin(q9) z34 sin(q3)

                         2    2
 - 2 cos(q2) cos(q9)  z34  cos(q3) sin(q7)) mu4 qp9 qp7/cos(q2) + ((

2 q8 cos(q2) sin(q3) + sin(q7) sin(q2) sin(q3) sin(q9) z34 cos(q3)

                               2
 - cos(q7) sin(q2) cos(q3)  z12 - 2 cos(q3) cos(q2) cos(q7) sin(q9) z34

                     2
 - sin(q2) cos(q3)  cos(q9) z34 + 2 sin(q7) cos(q2) z12 sin(q3)) mu4 omX/

                                                    2
cos(q2) + (sin(q7) sin(q2) sin(q9) z34 cos(q3)

 + 2 sin(q7) cos(q2) cos(q3) z12

 + 2 cos(q3) sin(q7) cos(q7) cos(q9) z34 cos(q2)

                                                                  2
 + sin(q2) cos(q3) sin(q3) cos(q9) z34 - 2 cos(q3) q8 cos(q7)  cos(q2)

 - sin(q2) sin(q7) sin(q9) z34 + cos(q7) sin(q2) cos(q3) sin(q3) z12

 + 2 q8 cos(q2) cos(q3)) mu4 omY/cos(q2) - (

cos(q2) sin(q7) sin(q9) z34 sin(q3) + 2 cos(q3) cos(q7) q8 sin(q7) cos(q2)

                         2
 + 2 cos(q3) cos(q7)  cos(q9) z34 cos(q2) + cos(q7) cos(q2) cos(q3) z12

 - cos(q2) cos(q3) cos(q9) z34) mu4 omZ/cos(q2)) qp8 -

(%2 + sin(q9) z34 cos(q2) cos(q3) z12 + %10 + cos(q9) z34 sin(q3) cos(q2) q8)

         2
 mu4 qp9 /cos(q2) - (sin(q3) cos(q2) cos(q9) z34

 + cos(q3) cos(q2) sin(q7) sin(q9) z34 + cos(q7) cos(q2) sin(q3) z12) mu4

                                    2
qpp8/cos(q2) - ((2 cos(q2) z34  cos(q3) sin(q7)

 + cos(q9) z34 sin(q3) sin(q2) cos(q3) sin(q7) q8

 + 2 z12 cos(q2) cos(q7) sin(q9) z34 sin(q3)
```

```
 + cos(q9) z34 sin(q3) sin(q2) cos(q3) z12

                                      2
 + sin(q7) sin(q9) z34 sin(q2) cos(q3)  z12

               2                                                      2
 + 2 cos(q2) z34  cos(q9) sin(q3) sin(q9) + sin(q9) z34 sin(q2) cos(q3)  q8

             2
 + cos(q7) z34  sin(q2) sin(q3) cos(q3)

                    2    2
 - 2 cos(q2) cos(q9)  z34  cos(q3) sin(q7)) mu4 omX/cos(q2) - (

z12 cos(q9) z34 sin(q2) - 2 cos(q3) sin(q7) q8 cos(q7) sin(q9) z34 cos(q2)

                                  2              2                2
 - cos(q9) z34 sin(q2) z12 cos(q3)  - cos(q7) z34  sin(q2) cos(q3)

 + sin(q9) z34 sin(q2) cos(q3) sin(q3) q8

 - 2 cos(q2) cos(q7) q8 cos(q9) z34 sin(q3)

                                                          2
 - 2 cos(q7) sin(q9) z34 cos(q2) cos(q3) z12 + cos(q7) z34  sin(q2)

                           2    2
 + 2 cos(q2) sin(q7) cos(q9)  z34  sin(q3) + sin(q7) q8 cos(q9) z34 sin(q2)

 + sin(q7) sin(q9) z34 sin(q2) cos(q3) sin(q3) z12

                                       2
 - cos(q9) z34 sin(q2) sin(q7) q8 cos(q3)

                         2
 + 2 cos(q2) cos(q9) z34  cos(q3) sin(q9)

                    2            2
 - 2 cos(q3) cos(q7)  cos(q9) z34  sin(q9) cos(q2)) mu4 omY/cos(q2) + (

                                                              2
sin(q7) sin(q9) z34 cos(q2) cos(q3) z12 - cos(q2) cos(q7) z34  sin(q3)

 + cos(q9) z34 sin(q3) cos(q2) z12

                    2
 - 2 cos(q3) cos(q7)  q8 sin(q9) z34 cos(q2)

 + cos(q9) z34 sin(q3) cos(q2) sin(q7) q8

                    2    2
 + 2 sin(q3) cos(q9)  z34  cos(q7) cos(q2)

                                  2
 + 2 cos(q3) sin(q7) cos(q9) z34  cos(q7) sin(q9) cos(q2)

 + sin(q9) z34 cos(q2) cos(q3) q8) mu4 omZ/cos(q2)) qp9 - (

cos(q2) cos(q3) cos(q9) z34 - cos(q2) sin(q7) sin(q9) z34 sin(q3)

 + cos(q7) cos(q2) cos(q3) z12) mu4 qp3 qp8/cos(q2) + (

/           2                                                          \
|(2 %8 + z12  cos(q2) cos(q3) + %6 + 2 %4 + %7 - %5 + %3) mu4           |
|------------------------------------------------------------ + i1X cos(q3)|
\                           cos(q2)                                      /

omX - ((2 sin(q7) q8 cos(q2) z12 sin(q3)
```

```
                                                                  2
 + 2 z12 cos(q2) cos(q7) cos(q9) z34 sin(q3) + z12  cos(q2) sin(q3)

 + 2 sin(q7) q8 cos(q2) cos(q7) cos(q9) z34 sin(q3)

              2        2
 - cos(q2) z34  cos(q9)  sin(q3) + cos(q2) sin(q9) z34 cos(q3) cos(q7) q8

              2                                 2
 + cos(q2) z34  sin(q3) - cos(q2) sin(q9) z34  cos(q3) sin(q7) cos(q9)

    2                         2          2        2    2
 - q8  cos(q2) sin(q3) cos(q7)  + cos(q7)  cos(q9)  z34  cos(q2) sin(q3)

    2
 + q8  cos(q2) sin(q3)) mu4/cos(q2) + i1Y sin(q3)) omY - (%1

 + sin(q9) z34 cos(q2) cos(q3) z12 + %2 + %10

           2
 - 2 cos(q7)  q8 sin(q3) cos(q2) cos(q9) z34 + cos(q9) z34 sin(q3) cos(q2) q8

                2    2
 + sin(q7) cos(q9)  z34  sin(q3) cos(q2) cos(q7) - %9

            2
 - cos(q7) q8  sin(q3) cos(q2) sin(q7)) mu4 omZ/cos(q2) - (

cos(q2) sin(q3) z12 + cos(q2) sin(q3) cos(q7) cos(q9) z34

 + cos(q2) sin(q3) sin(q7) q8) mu4 vX/cos(q2) - (cos(q2) cos(q3) sin(q7) q8

 + cos(q2) cos(q3) cos(q7) cos(q9) z34 + z12 cos(q2) cos(q3)) mu4 vY/cos(q2)

 - (cos(q2) cos(q3) sin(q7) cos(q9) z34 - sin(q9) z34 cos(q2) sin(q3)

 - cos(q7) q8 cos(q2) cos(q3)) mu4 vZ/cos(q2)) qp3 - ((

sin(q2) cos(q3) cos(q7) cos(q9) z34 sin(q3) + sin(q2) cos(q3) z12 sin(q3)

 + sin(q2) cos(q3) sin(q7) q8 sin(q3)) mu4 vX/cos(q2) - (

cos(q7) q8 sin(q3) cos(q2) - sin(q9) z34 cos(q3) cos(q2)

         2                                2
 - cos(q3)  sin(q2) sin(q7) q8 - cos(q3)  sin(q2) z12

                                                    2
 - sin(q7) cos(q9) z34 sin(q3) cos(q2) - cos(q3)  sin(q2) cos(q7) cos(q9) z34

) mu4 vY/cos(q2) - (sin(q3) sin(q9) z34 cos(q3) sin(q2)

                                         2
 + cos(q2) sin(q3) sin(q7) q8 - cos(q3)  sin(q7) cos(q9) z34 sin(q2)

                                                   2
 + cos(q2) sin(q3) cos(q7) cos(q9) z34 + cos(q3)  cos(q7) q8 sin(q2)

 + cos(q2) sin(q3) z12) mu4 vZ/cos(q2)) omX + (2 q8 cos(q2) sin(q3)

 - 2 cos(q3) cos(q2) cos(q7) sin(q9) z34 + 2 sin(q7) cos(q2) z12 sin(q3)) mu4

                                                   2
                   (%2 + %1 - %10 + %9) mu4 qp7
 qp8 qp7/cos(q2) + ---------------------------- - (
                              cos(q2)

cos(q9) z34 sin(q3) cos(q2) z12 + cos(q9) z34 sin(q3) cos(q2) sin(q7) q8
```

```
                      2
 + cos(q2) cos(q7) z34  sin(q3) + sin(q7) sin(q9) z34 cos(q2) cos(q3) z12

 + sin(q9) z34 cos(q2) cos(q3) q8) mu4 qp9 qp3/cos(q2)

   (%8 + %7 + %6 - %5 + %4 + %3) mu4 qp3 qp7
 + ---------------------------------------- - ((
                    cos(q2)

           2                 2
 sin(q9) z34  sin(q2) cos(q3)  cos(q7) cos(q9)

 + cos(q9) z34 sin(q3) sin(q2) cos(q3) q8

 - 2 sin(q7) q8 cos(q2) cos(q7) cos(q9) z34 cos(q3)

             2
 - cos(q7) q8  sin(q3) sin(q2) cos(q3) sin(q7)

                                2                        2
 + sin(q9) z34 sin(q2) cos(q3)  sin(q7) q8 + cos(q2) z34  cos(q3)

           2
 - 2 cos(q7)  q8 sin(q3) sin(q2) cos(q3) cos(q9) z34

                                2              2        2    2
 + sin(q9) z34 sin(q2) cos(q3)  z12 - cos(q7)  cos(q9)  z34  cos(q2) cos(q3)

 - cos(q7) q8 sin(q3) sin(q2) cos(q3) z12

 + sin(q7) cos(q9) z34 sin(q3) sin(q2) cos(q3) z12

     2                         2
 + q8  cos(q2) cos(q3) cos(q7)

                  2    2
 + sin(q7) cos(q9)  z34  sin(q3) sin(q2) cos(q3) cos(q7)) mu4/cos(q2)

                                                  2
 + i1Z cos(q3)) omX omZ + ((sin(q7) cos(q9) z34  sin(q9) sin(q2)

       2
 - 2 q8  sin(q2) cos(q3) sin(q3) - sin(q7) cos(q9) z34 cos(q2) cos(q3) z12

          2        2    2
 - cos(q7)  cos(q9)  z34  sin(q2) cos(q3) sin(q3)

           2
 + 2 cos(q7)  q8 cos(q3) cos(q2) cos(q9) z34 + cos(q7) q8 cos(q2) cos(q3) z12

                                       2                                 2
 - sin(q9) z34 sin(q3) cos(q2) z12 + q8  sin(q2) cos(q3) sin(q3) cos(q7)

        2
 - 2 z12  sin(q2) cos(q3) sin(q3)

 - 2 sin(q7) q8 sin(q2) cos(q3) cos(q7) cos(q9) z34 sin(q3)

 - cos(q7) q8 sin(q9) z34 sin(q2)

                  2    2
 - sin(q7) cos(q9)  z34  cos(q3) cos(q2) cos(q7)

             2                                2
 + cos(q7) q8  cos(q3) cos(q2) sin(q7) - z34  sin(q2) cos(q3) sin(q3)

                                2
 - cos(q2) cos(q7) cos(q9) z34  sin(q9) sin(q3)

 - 4 sin(q7) q8 sin(q2) cos(q3) sin(q3) z12
```

```
  - 4 cos(q7) cos(q9) z34 sin(q2) cos(q3) sin(q3) z12

  - cos(q9) z34 cos(q2) cos(q3) q8 - cos(q2) sin(q7) q8 sin(q9) z34 sin(q3))

                i1X sin(q2) cos(q3) sin(q3) + i1Y sin(q2) cos(q3) sin(q3)
 mu4/cos(q2) - ---------------------------------------------------------) omX
                                        cos(q2)

                 2                 2
 omY + ((%2 + q8  sin(q2) cos(q3)

  + cos(q7) q8 sin(q2) sin(q3) sin(q9) z34 cos(q3)

                                 2
  + 2 sin(q7) q8 sin(q2) cos(q3)  z12

                                          2
  + 2 cos(q7) cos(q9) z34 sin(q2) cos(q3)  z12

           2    2                 2
  + cos(q9)  z34  sin(q2) cos(q3)  + %1 + sin(q9) z34 cos(q2) cos(q3) z12

       2                2
  + z12  sin(q2) cos(q3)

                        2
  - sin(q7) cos(q9) z34  sin(q2) sin(q3) sin(q9) cos(q3)) mu4/cos(q2)

                        2
    i1X sin(q2) cos(q3)      2
  + --------------------) omX  + (cos(q2) cos(q3) sin(q7) cos(q9) z34
           cos(q2)

  - sin(q9) z34 cos(q2) sin(q3) - cos(q7) q8 cos(q2) cos(q3)) mu4 vZ omZ/

                        2        2             2
 cos(q2) + (cos(q2) z34  cos(q9)  sin(q3) + q8  cos(q2) sin(q3)

  + sin(q7) q8 cos(q2) z12 sin(q3)

                        2
  + cos(q2) sin(q9) z34  cos(q3) sin(q7) cos(q9)

  + z12 cos(q2) cos(q7) cos(q9) z34 sin(q3)

  - cos(q2) sin(q9) z34 cos(q3) cos(q7) q8) mu4 qpp7/cos(q2) + (

 cos(q2) cos(q3) sin(q7) q8 + cos(q2) cos(q3) cos(q7) cos(q9) z34

  + z12 cos(q2) cos(q3)) mu4 vXp/cos(q2)

                          2
%1 :=          sin(q9) z34  cos(q2) cos(q3) cos(q7) cos(q9)

%2 :=             sin(q9) z34 cos(q2) cos(q3) sin(q7) q8

                              2
%3 :=                        q8  cos(q2) cos(q3)

%4 :=                 sin(q7) q8 cos(q2) cos(q3) z12

                                          2
%5 :=          cos(q2) sin(q7) cos(q9) z34  sin(q9) sin(q3)

                                      2    2
%6 :=                  cos(q2) cos(q9)  z34  cos(q3)

%7 :=             cos(q2) cos(q7) q8 sin(q9) z34 sin(q3)
```

```
%8 :=             cos(q7) cos(q9) z34 cos(q2) cos(q3) z12

%9 :=                 cos(q7) q8 sin(q3) cos(q2) z12

%10 :=            sin(q7) cos(q9) z34 sin(q3) cos(q2) z12

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp110:=diff(eq2,qp3):
> tmp111:=subs(varstq,tmp110):
> tmp112:=subs(tmp17 union convert(tmp21,set),tmp111):
> tmp113:=simplify(subs(varstqp,tmp112),trig):
> tmp114:=subs(varslect,tmp113):
#####
> dKdqp3:=
>   rlms(clct(tmp114,
>             [qpp7,qpp8,qpp9],
>             [omXp,omYp,omZp,vXp,vYp,vZp,qp1,qp2,qp3,qp4,qp5,qp6,qp7,qp8,qp9],
>             [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4])):
#####

> dK2dqp3dt:=subs(varslect,diff(tmp113,t)):
#####
> tmp116:=diff(eq2,q3):
> tmp117:=subs(varstq,tmp116):
> tmp118:=subs(tmp17 union convert(tmp21,set),tmp117):
> tmp119:=simplify(tmp118,trig):
#####
> dKdq3:=
>   rlms(clct(subs(varslect,tmp119),
>             [qpp7,qpp8,qpp9],
>             [omXp,omYp,omZp,vXp,vYp,vZp,qp1,qp2,qp3,qp4,qp5,qp6,qp7,qp8,qp9],
>             [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4])):
#####
> K3:=
>   rlms(clct(simplify(dK2dqp3dt-dKdq3,trig),
>             [qpp7,qpp8,qpp9],
>             [omXp,omYp,omZp,vXp,vYp,vZp,qp1,qp2,qp3,qp4,qp5,qp6,qp7,qp8,qp9],
>             [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4],factor));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

K3 := ((cos(q7) q8 - sin(q7) cos(q9) z34) mu4 vY + mu4 sin(q9) z34 vX) omZ

 - (cos(q7) q8 - sin(q7) cos(q9) z34) mu4 vZ omY - mu4 vZ sin(q9) z34 omX +

            2
(sin(q9) z34  cos(q7) cos(q9) + sin(q9) z34 z12 + sin(q9) z34 sin(q7) q8) mu4

                              2        2      2          2    2        2
omY omZ + (i1Y - i1X - (2 z34  cos(q9)  - z34  - cos(q9)  z34  cos(q7)

                                              2   2
 - 2 sin(q7) cos(q9) z34 cos(q7) q8 + cos(q7)  q8 ) mu4) omX omY + (

                 2    2                  2
cos(q7) cos(q9)  z34  sin(q7) - sin(q7) q8  cos(q7) - z12 cos(q7) q8

                                          2
 + z12 sin(q7) cos(q9) z34 - 2 cos(q7)  cos(q9) z34 q8 + q8 cos(q9) z34) mu4

                          2                        2
omX omZ + (%2 - %1) mu4 omX  - (%2 - %1) mu4 omY  + ((2 q8 cos(q9) z34

                   2    2                      2
 + 2 cos(q7) cos(q9)  z34  sin(q7) - 2 sin(q7) q8  cos(q7)

           2                                   2    2         2
 - 4 cos(q7)  cos(q9) z34 q8) mu4 omZ + (2 cos(q9)  z34  cos(q7)
```

```
                                          2        2           2   2
 + 4 sin(q7) cos(q9) z34 cos(q7) q8 - 2 z34  cos(q9)  - 2 cos(q7)  q8 ) mu4 omY

 - (2 %1 - 2 %2) mu4 omX) qp7 -

           2
(sin(q9) z34  cos(q7) cos(q9) + sin(q9) z34 z12 + sin(q9) z34 sin(q7) q8) mu4

                       2    2                     2
omXp + (cos(q7) cos(q9)  z34  sin(q7) - sin(q7) q8  cos(q7) - z12 cos(q7) q8

                                    2
 + z12 sin(q7) cos(q9) z34 - 2 cos(q7)  cos(q9) z34 q8 + q8 cos(q9) z34) mu4

omYp + (i1Z + (

   2          2   2                                          2    2        2
z34  + cos(q7)  q8  - 2 sin(q7) cos(q9) z34 cos(q7) q8 - cos(q9)  z34  cos(q7)

                                                          2
) mu4) omZp - ((2 sin(q7) cos(q9) z34 cos(q7) - 2 cos(q7)  q8) mu4 omZ

                             2
 - (2 cos(q9) z34 - 2 cos(q7)  cos(q9) z34 - 2 sin(q7) q8 cos(q7)) mu4 omY

 + 2 mu4 sin(q9) z34 omX sin(q7)) qp8 + (

                                                  2        2
(2 sin(q7) sin(q9) z34 cos(q7) q8 + 2 cos(q9) z34  cos(q7)  sin(q9)) mu4 omZ

                         2                              2
 - (2 cos(q7) cos(q9) z34  sin(q7) sin(q9) - 2 cos(q7)  sin(q9) z34 q8) mu4 omY

             2    2                          2
 - (2 cos(q9)  z34  cos(q7) - 2 cos(q7) z34 ) mu4 omX) qp9

 - (cos(q7) q8 - sin(q7) cos(q9) z34) mu4 vXp + mu4 sin(q9) z34 vYp

                   2                                2
 + mu4 sin(q9) qp9  z34 cos(q7) q8 + (%2 - %1) mu4 qp7

             2    2                          2
 - (2 cos(q9)  z34  cos(q7) - 2 cos(q7) z34 ) mu4 qp7 qp9

 - 2 mu4 sin(q9) z34 sin(q7) qp7 qp8

                                         2
 - (sin(q9) z34 sin(q7) q8 + sin(q9) z34  cos(q7) cos(q9)) mu4 qpp7

 + mu4 sin(q9) z34 cos(q7) qpp8

       2
 + (z34  sin(q7) - cos(q9) z34 cos(q7) q8) mu4 qpp9

%1 :=                       sin(q9) z34 cos(q7) q8

                                          2
%2 :=                    sin(q7) cos(q9) z34  sin(q9)

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp120:=diff(eq2,qp4):
> tmp121:=subs(varstq,tmp120):
> tmp122:=subs(tmp17 union convert(tmp21,set),tmp121):
> tmp123:=simplify(subs(varstqp,tmp122),trig):
> tmp124:=subs(varslect,tmp123):
#####
> dKdqp4:=
>   rlms(clct(tmp124,
>             [qpp7,qpp8,qpp9],
```

```
>             [omXp,omYp,omZp,vXp,vYp,vZp,qp1,qp2,qp3,qp4,qp5,qp6,qp7,qp8,qp9],
>             [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4])):
#####
> dK2dqp4dt:=subs(varslect,diff(tmp123,t)):
#####
> tmp126:=diff(eq2,q4):
> tmp127:=subs(varstq,tmp126):
> tmp128:=subs(tmp17 union convert(tmp21,set),tmp127):
> tmp129:=simplify(tmp128,trig):
#####
> dKdq4:=
>   rlms(clct(subs(varslect,tmp129),
>             [qpp7,qpp8,qpp9],
>             [omXp,omYp,omZp,vXp,vYp,vZp,qp1,qp2,qp3,qp4,qp5,qp6,qp7,qp8,qp9],
>             [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4])):
#####

> K4:=
>   rlms(clct(simplify(dK2dqp4dt-dKdq4,trig),
>             [qpp7,qpp8,qpp9],
>             [omXp,omYp,omZp,vXp,vYp,vZp,qp1,qp2,qp3,qp4,qp5,qp6,qp7,qp8,qp9],
>             [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4],factor));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

K4 := ((%3 + %2) mu4 omZ

 - (cos(q3) cos(q2) sin(q7) cos(q9) z34 - cos(q3) cos(q2) cos(q7) q8) mu4 omY

 + (cos(q7) q8 sin(q3) cos(q2) - cos(q7) cos(q9) z34 sin(q2)

 - sin(q7) q8 sin(q2) - %1) mu4 omX) qp7 + (cos(q2) z12 sin(q3)

 - sin(q2) sin(q7) cos(q9) z34 + cos(q2) sin(q7) q8 sin(q3)

 + cos(q2) cos(q7) cos(q9) z34 sin(q3) + sin(q2) cos(q7) q8) mu4 omXp

 + (%2 + %3 + cos(q3) cos(q2) z12 - sin(q2) sin(q9) z34) mu4 omYp - (

cos(q3) cos(q2) cos(q7) q8 - cos(q3) cos(q2) sin(q7) cos(q9) z34

 + cos(q2) sin(q3) sin(q9) z34) mu4 omZp - (mu4 cos(q2) cos(q3) omZ cos(q7)

 - mu4 cos(q2) cos(q3) omY sin(q7)

 - (cos(q2) sin(q7) sin(q3) + sin(q2) cos(q7)) mu4 omX) qp8 - (

(cos(q3) cos(q2) sin(q7) sin(q9) z34 + sin(q3) cos(q2) cos(q9) z34) mu4 omZ

 + (cos(q3) cos(q2) cos(q7) sin(q9) z34 + cos(q9) z34 sin(q2)) mu4 omY

 - (sin(q2) sin(q7) sin(q9) z34 - cos(q2) cos(q7) sin(q9) z34 sin(q3)) mu4 omX)

 qp9 + ((mu1 sin(q2) sin(q3) + mu4 sin(q2) sin(q3)) vY

 + (mu4 cos(q2) + mu1 cos(q2)) vZ

 - (mu4 sin(q2) cos(q3) + mu1 sin(q2) cos(q3)) vX - (

cos(q3) sin(q2) sin(q7) cos(q9) z34 - cos(q3) sin(q2) cos(q7) q8

 - sin(q3) sin(q2) sin(q9) z34) mu4 omZ - (sin(q9) z34 cos(q2)

 + cos(q3) sin(q2) cos(q7) cos(q9) z34 + cos(q3) sin(q2) z12

 + cos(q3) sin(q2) sin(q7) q8) mu4 omY - (cos(q2) sin(q7) cos(q9) z34

 + sin(q2) z12 sin(q3) - cos(q2) cos(q7) q8 + sin(q2) sin(q7) q8 sin(q3)

 + sin(q2) cos(q7) cos(q9) z34 sin(q3)) mu4 omX) qp2 - (
```

```
(mu1 cos(q2) cos(q3) + mu4 cos(q2) cos(q3)) vY
 + (mu1 cos(q2) sin(q3) + mu4 cos(q2) sin(q3)) vX
 - (cos(q7) q8 sin(q3) cos(q2) - %1 - sin(q9) z34 cos(q3) cos(q2)) mu4 omZ + (
cos(q2) z12 sin(q3) + cos(q2) sin(q7) q8 sin(q3)
 + cos(q2) cos(q7) cos(q9) z34 sin(q3)) mu4 omY
 - (cos(q3) cos(q2) z12 + %2 + %3) mu4 omX) qp3
 + (mu1 cos(q2) cos(q3) + mu4 cos(q2) cos(q3)) vXp
 - (mu1 cos(q2) sin(q3) + mu4 cos(q2) sin(q3)) vYp
 + (mu1 sin(q2) + mu4 sin(q2)) vZp + (cos(q7) q8 sin(q3) cos(q2)
                                                                  2
 - cos(q7) cos(q9) z34 sin(q2) - sin(q7) q8 sin(q2) - %1) mu4 qp7
 + (sin(q3) sin(q2) cos(q7) + cos(q2) sin(q7)) mu4 qp8 qp2
 + (%3 + %2) mu4 qp7 qp3 -
(cos(q3) cos(q2) sin(q7) sin(q9) z34 + sin(q3) cos(q2) cos(q9) z34) mu4 qp9 qp3
 - (cos(q7) cos(q2) sin(q9) z34 + sin(q2) cos(q3) cos(q9) z34
 - sin(q2) sin(q7) sin(q3) sin(q9) z34) mu4 qp9 qp2
 - mu4 cos(q3) qp3 cos(q2) cos(q7) qp8
 + (2 cos(q2) sin(q7) sin(q3) + 2 sin(q2) cos(q7)) mu4 qp8 qp7 - (
cos(q2) sin(q7) cos(q9) z34 - cos(q2) cos(q7) q8
 + sin(q2) cos(q7) cos(q9) z34 sin(q3) + sin(q2) sin(q7) q8 sin(q3)) mu4 qp7
qp2 + (2 sin(q2) sin(q7) sin(q9) z34 - 2 cos(q2) cos(q7) sin(q9) z34 sin(q3))
mu4 qp9 qp7
                                                                       2
 - (sin(q9) z34 cos(q3) cos(q2) + cos(q7) cos(q9) z34 sin(q2) + %1) mu4 qp9  +
(cos(q2) sin(q7) q8 sin(q3) - sin(q2) sin(q7) cos(q9) z34 + sin(q2) cos(q7) q8
 + cos(q2) cos(q7) cos(q9) z34 sin(q3)) mu4 qpp7
 + (sin(q2) sin(q7) - sin(q3) cos(q2) cos(q7)) mu4 qpp8 + (
cos(q2) cos(q3) cos(q9) z34 - cos(q2) sin(q7) sin(q3) sin(q9) z34
 - cos(q7) sin(q2) sin(q9) z34) mu4 qpp9
%1 :=                sin(q7) cos(q9) z34 sin(q3) cos(q2)
%2 :=                cos(q3) cos(q2) cos(q7) cos(q9) z34
%3 :=                    cos(q3) cos(q2) sin(q7) q8
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
> tmp130:=diff(eq2,qp5):
> tmp131:=subs(varstq,tmp130):
> tmp132:=subs(tmp17 union convert(tmp21,set),tmp131):
> tmp133:=simplify(subs(varstqp,tmp132),trig):
> tmp134:=subs(varslect,tmp133):
```

```
#####
> dKdqp5:=
>   rlms(clct(tmp134,
>             [qpp7,qpp8,qpp9],
>             [omXp,omYp,omZp,vXp,vYp,vZp,qp1,qp2,qp3,qp4,qp5,qp6,qp7,qp8,qp9],
>             [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4])):
#####
> dK2dqp5dt:=subs(varslect,diff(tmp133,t)):
#####
> tmp136:=diff(eq2,q5):
> tmp137:=subs(varstq,tmp136):
> tmp138:=subs(tmp17 union convert(tmp21,set),tmp137):
> tmp139:=simplify(tmp138,trig):
#####

> dKdq5:=
>   rlms(clct(subs(varslect,tmp139),
>             [qpp7,qpp8,qpp9],
>             [omXp,omYp,omZp,vXp,vYp,vZp,qp1,qp2,qp3,qp4,qp5,qp6,qp7,qp8,qp9],
>             [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4])):
#####
> K5:=
>   rlms(clct(simplify(dK2dqp5dt-dKdq5,trig),
>             [qpp7,qpp8,qpp9],
>             [omXp,omYp,omZp,vXp,vYp,vZp,qp1,qp2,qp3,qp4,qp5,qp6,qp7,qp8,qp9],
>             [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4],factor));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

K5 := (%7 + cos(q2) sin(q1) sin(q7) q8 + cos(q2) sin(q1) cos(q7) cos(q9) z34

 - %6 + cos(q7) q8 sin(q1) sin(q2) sin(q3) - cos(q7) q8 cos(q1) cos(q3)) mu4

   2
qp7  - ((%1 mu4 + %1 mu1) vY + (%8 mu4 + %8 mu1) vX - (

cos(q7) q8 sin(q1) sin(q2) sin(q3) - sin(q9) z34 cos(q1) sin(q3) - %6

 - cos(q7) q8 cos(q1) cos(q3) - sin(q9) z34 sin(q1) sin(q2) cos(q3) + %7) mu4

omZ + (cos(q7) cos(q9) z34 sin(q1) sin(q2) sin(q3) - z12 cos(q1) cos(q3)

 - sin(q7) q8 cos(q1) cos(q3) + z12 sin(q1) sin(q2) sin(q3)

 + sin(q7) q8 sin(q1) sin(q2) sin(q3) - cos(q7) cos(q9) z34 cos(q1) cos(q3))

mu4 omY -

(%3 + z12 cos(q1) sin(q3) + %4 + %5 + %2 + z12 sin(q1) sin(q2) cos(q3)) mu4 omX

) qp3 - (mu4 cos(q2) sin(q1) + mu1 cos(q2) sin(q1)) vZp - (%8 mu4 + %8 mu1) vYp

 - ((cos(q9) z34 sin(q1) sin(q2) sin(q3) - cos(q9) z34 cos(q1) cos(q3)

 + sin(q7) sin(q9) z34 sin(q1) sin(q2) cos(q3)

 + sin(q7) sin(q9) z34 cos(q1) sin(q3)) mu4 omZ + (

cos(q7) sin(q9) z34 cos(q1) sin(q3) - cos(q2) sin(q1) cos(q9) z34

 + cos(q7) sin(q9) z34 sin(q1) sin(q2) cos(q3)) mu4 omY - (

cos(q7) sin(q9) z34 cos(q1) cos(q3) - cos(q2) sin(q1) sin(q7) sin(q9) z34

 - cos(q7) sin(q9) z34 sin(q1) sin(q2) sin(q3)) mu4 omX) qp9 - (

(cos(q7) cos(q1) sin(q3) + cos(q7) sin(q1) sin(q2) cos(q3)) mu4 omZ

 - (sin(q7) sin(q1) sin(q2) cos(q3) + sin(q7) cos(q1) sin(q3)) mu4 omY + (
```

```
sin(q7) cos(q1) cos(q3) + cos(q2) sin(q1) cos(q7)
 - sin(q7) sin(q1) sin(q2) sin(q3)) mu4 omX) qp8 - (
sin(q9) z34 sin(q1) sin(q2) cos(q3) - cos(q2) sin(q1) cos(q7) cos(q9) z34
                                                                  2
 + sin(q9) z34 cos(q1) sin(q3) - %7 + %6) mu4 qp9  - (cos(q7) sin(q1) cos(q3)
 + cos(q7) cos(q1) sin(q2) sin(q3) + cos(q2) cos(q1) sin(q7)) mu4 qp1 qp8
 - (cos(q7) sin(q1) cos(q2) sin(q3) - sin(q2) sin(q1) sin(q7)) mu4 qp2 qp8 + (
(%3 + %5 + %4 + %2) mu4 omZ + (cos(q7) q8 cos(q1) sin(q3)
 + cos(q7) q8 sin(q1) sin(q2) cos(q3) - sin(q7) cos(q9) z34 cos(q1) sin(q3)
 - sin(q7) cos(q9) z34 sin(q1) sin(q2) cos(q3)) mu4 omY + (%7
 + cos(q2) sin(q1) sin(q7) q8 + cos(q2) sin(q1) cos(q7) cos(q9) z34 - %6
 + cos(q7) q8 sin(q1) sin(q2) sin(q3) - cos(q7) q8 cos(q1) cos(q3)) mu4 omX)
qp7 - (cos(q7) cos(q1) sin(q3) + cos(q7) sin(q1) sin(q2) cos(q3)) mu4 qp3 qp8
 + (cos(q9) z34 sin(q1) sin(q2) cos(q3) + cos(q9) z34 cos(q1) sin(q3)
 - sin(q7) sin(q9) z34 sin(q1) sin(q2) sin(q3)
 + sin(q7) sin(q9) z34 cos(q1) cos(q3) + cos(q2) sin(q1) cos(q7) sin(q9) z34)
mu4 qpp9 - (cos(q9) z34 sin(q1) sin(q2) sin(q3) - cos(q9) z34 cos(q1) cos(q3)
 + sin(q7) sin(q9) z34 sin(q1) sin(q2) cos(q3)
 + sin(q7) sin(q9) z34 cos(q1) sin(q3)) mu4 qp3 qp9 - (
2 cos(q7) sin(q9) z34 sin(q1) sin(q2) sin(q3)
 - 2 cos(q7) sin(q9) z34 cos(q1) cos(q3)
 + 2 cos(q2) sin(q1) sin(q7) sin(q9) z34) mu4 qp9 qp7 + (
cos(q7) cos(q9) z34 sin(q1) sin(q2) sin(q3)
 - cos(q7) cos(q9) z34 cos(q1) cos(q3) + sin(q7) q8 sin(q1) sin(q2) sin(q3)
 - cos(q2) sin(q1) cos(q7) q8 + cos(q2) sin(q1) sin(q7) cos(q9) z34
 - sin(q7) q8 cos(q1) cos(q3)) mu4 qpp7 + (%3 + %5 + %4 + %2) mu4 qp3 qp7 - (
cos(q2) cos(q1) cos(q7) q8 - cos(q7) cos(q9) z34 sin(q1) cos(q3)
 - cos(q2) cos(q1) sin(q7) cos(q9) z34
 - cos(q7) cos(q9) z34 cos(q1) sin(q2) sin(q3)
 - sin(q7) q8 cos(q1) sin(q2) sin(q3) - sin(q7) q8 sin(q1) cos(q3)) mu4 qp1 qp7
 - (sin(q2) sin(q1) sin(q7) cos(q9) z34
 - cos(q7) cos(q9) z34 sin(q1) cos(q2) sin(q3)
 - sin(q7) q8 sin(q1) cos(q2) sin(q3) - sin(q2) sin(q1) cos(q7) q8) mu4 qp2 qp7
 + (cos(q2) cos(q1) cos(q7) sin(q9) z34
 - sin(q7) sin(q9) z34 cos(q1) sin(q2) sin(q3)
 - sin(q7) sin(q9) z34 sin(q1) cos(q3) - cos(q9) z34 sin(q1) sin(q3)
```

```
 + cos(q9) z34 cos(q1) sin(q2) cos(q3)) mu4 qp1 qp9 - (
(mu1 sin(q1) cos(q2) sin(q3) + mu4 sin(q1) cos(q2) sin(q3)) vY
 - (mu4 sin(q2) sin(q1) + mu1 sin(q2) sin(q1)) vZ
 - (mu4 cos(q3) sin(q1) cos(q2) + mu1 cos(q3) sin(q1) cos(q2)) vX + (
cos(q7) q8 sin(q1) cos(q2) cos(q3)
 - sin(q7) cos(q9) z34 sin(q1) cos(q2) cos(q3)
 + sin(q9) z34 sin(q1) cos(q2) sin(q3)) mu4 omZ - (z12 sin(q1) cos(q2) cos(q3)
 + cos(q7) cos(q9) z34 sin(q1) cos(q2) cos(q3) - sin(q2) sin(q1) sin(q9) z34
 + sin(q7) q8 sin(q1) cos(q2) cos(q3)) mu4 omY - (
sin(q7) q8 sin(q1) cos(q2) sin(q3) - sin(q2) sin(q1) sin(q7) cos(q9) z34
 + cos(q7) cos(q9) z34 sin(q1) cos(q2) sin(q3) + sin(q2) sin(q1) cos(q7) q8
 + z12 sin(q1) cos(q2) sin(q3)) mu4 omX) qp2 + (
cos(q9) z34 sin(q1) cos(q2) cos(q3) - sin(q2) sin(q1) cos(q7) sin(q9) z34
 - sin(q7) sin(q9) z34 sin(q1) cos(q2) sin(q3)) mu4 qp2 qp9 - (
cos(q7) sin(q1) sin(q2) sin(q3) - cos(q7) cos(q1) cos(q3)
 + cos(q2) sin(q1) sin(q7)) mu4 qpp8 + (2 sin(q7) sin(q1) sin(q2) sin(q3)
 - 2 cos(q2) sin(q1) cos(q7) - 2 sin(q7) cos(q1) cos(q3)) mu4 qp8 qp7 + (
sin(q7) cos(q9) z34 sin(q1) sin(q2) cos(q3)
 - sin(q9) z34 sin(q1) sin(q2) sin(q3) - cos(q7) q8 cos(q1) sin(q3)
 + sin(q9) z34 cos(q1) cos(q3) + sin(q7) cos(q9) z34 cos(q1) sin(q3)
 - cos(q7) q8 sin(q1) sin(q2) cos(q3)) mu4 omZp + (
cos(q7) cos(q9) z34 sin(q1) sin(q2) sin(q3)
 + sin(q7) q8 sin(q1) sin(q2) sin(q3) - cos(q2) sin(q1) cos(q7) q8
 + z12 sin(q1) sin(q2) sin(q3) - z12 cos(q1) cos(q3)
 - sin(q7) q8 cos(q1) cos(q3) - cos(q7) cos(q9) z34 cos(q1) cos(q3)
 + cos(q2) sin(q1) sin(q7) cos(q9) z34) mu4 omXp + (cos(q2) sin(q1) sin(q9) z34
 + %5 + z12 sin(q1) sin(q2) cos(q3) + %4 + %3 + z12 cos(q1) sin(q3) + %2) mu4
omYp - (((cos(q1) sin(q2) sin(q3) + sin(q1) cos(q3)) mu4
 + (cos(q1) sin(q2) sin(q3) + sin(q1) cos(q3)) mu1) vY
 + (mu1 cos(q2) cos(q1) + mu4 cos(q2) cos(q1)) vZ - (
(cos(q1) sin(q2) cos(q3) - sin(q1) sin(q3)) mu4
 + (cos(q1) sin(q2) cos(q3) - sin(q1) sin(q3)) mu1) vX + (
sin(q9) z34 sin(q1) cos(q3) + cos(q7) q8 cos(q1) sin(q2) cos(q3)
 - sin(q7) cos(q9) z34 cos(q1) sin(q2) cos(q3)
 + sin(q9) z34 cos(q1) sin(q2) sin(q3) - cos(q7) q8 sin(q1) sin(q3)
```

```
 + sin(q7) cos(q9) z34 sin(q1) sin(q3)) mu4 omZ - (cos(q2) cos(q1) sin(q9) z34

 - z12 sin(q1) sin(q3) + z12 cos(q1) sin(q2) cos(q3)

 - sin(q7) q8 sin(q1) sin(q3) - cos(q7) cos(q9) z34 sin(q1) sin(q3)

 + sin(q7) q8 cos(q1) sin(q2) cos(q3)

 + cos(q7) cos(q9) z34 cos(q1) sin(q2) cos(q3)) mu4 omY - (

cos(q7) cos(q9) z34 cos(q1) sin(q2) sin(q3)

 + cos(q2) cos(q1) sin(q7) cos(q9) z34 + z12 cos(q1) sin(q2) sin(q3)

 + cos(q7) cos(q9) z34 sin(q1) cos(q3) - cos(q2) cos(q1) cos(q7) q8

 + sin(q7) q8 cos(q1) sin(q2) sin(q3) + z12 sin(q1) cos(q3)

 + sin(q7) q8 sin(q1) cos(q3)) mu4 omX) qp1 + (%1 mu4 + %1 mu1) vXp

%1 :=             sin(q1) sin(q2) cos(q3) + cos(q1) sin(q3)

%2 :=                cos(q7) cos(q9) z34 cos(q1) sin(q3)

%3 :=                sin(q7) q8 sin(q1) sin(q2) cos(q3)

%4 :=                    sin(q7) q8 cos(q1) sin(q3)

%5 :=            cos(q7) cos(q9) z34 sin(q1) sin(q2) cos(q3)

%6 :=            sin(q7) cos(q9) z34 sin(q1) sin(q2) sin(q3)

%7 :=                sin(q7) cos(q9) z34 cos(q1) cos(q3)

%8 :=             sin(q1) sin(q2) sin(q3) - cos(q1) cos(q3)

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp140:=diff(eq2,qp6):
> tmp141:=subs(varstq,tmp140):
> tmp142:=subs(tmp17 union convert(tmp21,set),tmp141):
> tmp143:=simplify(subs(varstqp,tmp142),trig):
> tmp144:=subs(varslect,tmp143):
#####
> dKdqp6:=
>   rlms(clct(tmp144,
>             [qpp7,qpp8,qpp9],
>             [omXp,omYp,omZp,vXp,vYp,vZp,qp1,qp2,qp3,qp4,qp5,qp6,qp7,qp8,qp9],
>             [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4])):
#####
> dK2dqp6dt:=subs(varslect,diff(tmp143,t)):
#####
> tmp146:=diff(eq2,q6):
> tmp147:=subs(varstq,tmp146):
> tmp148:=subs(tmp17 union convert(tmp21,set),tmp147):
> tmp149:=simplify(tmp148,trig):
#####
> dKdq6:=
>   rlms(clct(subs(varslect,tmp149),
>             [qpp7,qpp8,qpp9],
>             [omXp,omYp,omZp,vXp,vYp,vZp,qp1,qp2,qp3,qp4,qp5,qp6,qp7,qp8,qp9],
>             [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4])):
#####
> K6:=
>   rlms(clct(simplify(dK2dqp6dt-dKdq6,trig),
>             [qpp7,qpp8,qpp9],
>             [omXp,omYp,omZp,vXp,vYp,vZp,qp1,qp2,qp3,qp4,qp5,qp6,qp7,qp8,qp9],
>             [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4],factor));
```

```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

K6 := - (cos(q7) q8 sin(q1) cos(q3) - %4 - %3 + cos(q2) cos(q1) sin(q7) q8

 + cos(q2) cos(q1) cos(q7) cos(q9) z34 + cos(q7) q8 cos(q1) sin(q2) sin(q3))

       2
mu4 qp7  - (cos(q2) cos(q1) cos(q7) cos(q9) z34 + sin(q9) z34 sin(q1) sin(q3)

                                                         2
 - sin(q9) z34 cos(q1) sin(q2) cos(q3) - %3 - %4) mu4 qp9  + (

(%1 mu4 + %1 mu1) vY + (%2 mu4 + %2 mu1) vX + (%3 - sin(q9) z34 sin(q1) sin(q3)

 - cos(q7) q8 cos(q1) sin(q2) sin(q3) + sin(q9) z34 cos(q1) sin(q2) cos(q3)

 + %4 - cos(q7) q8 sin(q1) cos(q3)) mu4 omZ + (

cos(q7) cos(q9) z34 cos(q1) sin(q2) sin(q3)

 + cos(q7) cos(q9) z34 sin(q1) cos(q3) + sin(q7) q8 sin(q1) cos(q3)

 + z12 sin(q1) cos(q3) + sin(q7) q8 cos(q1) sin(q2) sin(q3)

 + z12 cos(q1) sin(q2) sin(q3)) mu4 omY +

(%5 - z12 cos(q1) sin(q2) cos(q3) - %7 - %8 + z12 sin(q1) sin(q3) + %6) mu4 omX

) qp3 - ((cos(q7) sin(q1) sin(q3) - cos(q7) cos(q1) sin(q2) cos(q3)) mu4 omZ

 + (sin(q7) cos(q1) sin(q2) cos(q3) - sin(q7) sin(q1) sin(q3)) mu4 omY - (

cos(q2) cos(q1) cos(q7) - sin(q7) cos(q1) sin(q2) sin(q3)

 - sin(q7) sin(q1) cos(q3)) mu4 omX) qp8 - (cos(q9) z34 cos(q1) cos(q2) cos(q3)

 - sin(q7) sin(q9) z34 cos(q1) cos(q2) sin(q3)

 - sin(q2) cos(q1) cos(q7) sin(q9) z34) mu4 qp2 qp9 - (

2 sin(q7) sin(q1) cos(q3) + 2 sin(q7) cos(q1) sin(q2) sin(q3)

 - 2 cos(q2) cos(q1) cos(q7)) mu4 qp8 qp7 - (%8 + %7 - %6 - %5) mu4 qp3 qp7

 - (cos(q7) sin(q1) sin(q3) - cos(q7) cos(q1) sin(q2) cos(q3)) mu4 qp3 qp8 - (

cos(q7) cos(q9) z34 cos(q1) sin(q2) sin(q3)

 + cos(q2) cos(q1) sin(q7) cos(q9) z34 + z12 cos(q1) sin(q2) sin(q3)

 + cos(q7) cos(q9) z34 sin(q1) cos(q3) - cos(q2) cos(q1) cos(q7) q8

 + sin(q7) q8 cos(q1) sin(q2) sin(q3) + z12 sin(q1) cos(q3)

 + sin(q7) q8 sin(q1) cos(q3)) mu4 omXp - ((

(sin(q1) sin(q2) sin(q3) - cos(q1) cos(q3)) mu4

 + (sin(q1) sin(q2) sin(q3) - cos(q1) cos(q3)) mu1) vY

 + (mu4 cos(q2) sin(q1) + mu1 cos(q2) sin(q1)) vZ - (

(sin(q1) sin(q2) cos(q3) + cos(q1) sin(q3)) mu4

 + (sin(q1) sin(q2) cos(q3) + cos(q1) sin(q3)) mu1) vX - (

sin(q7) cos(q9) z34 sin(q1) sin(q2) cos(q3)

 - sin(q9) z34 sin(q1) sin(q2) sin(q3) - cos(q7) q8 cos(q1) sin(q3)
```

```
 + sin(q9) z34 cos(q1) cos(q3) + sin(q7) cos(q9) z34 cos(q1) sin(q3)
 - cos(q7) q8 sin(q1) sin(q2) cos(q3)) mu4 omZ - (cos(q2) sin(q1) sin(q9) z34
 + cos(q7) cos(q9) z34 sin(q1) sin(q2) cos(q3) + z12 sin(q1) sin(q2) cos(q3)
 + sin(q7) q8 cos(q1) sin(q3) + sin(q7) q8 sin(q1) sin(q2) cos(q3)
 + z12 cos(q1) sin(q3) + cos(q7) cos(q9) z34 cos(q1) sin(q3)) mu4 omY - (
cos(q7) cos(q9) z34 sin(q1) sin(q2) sin(q3)
 + sin(q7) q8 sin(q1) sin(q2) sin(q3) - cos(q2) sin(q1) cos(q7) q8
 + z12 sin(q1) sin(q2) sin(q3) - z12 cos(q1) cos(q3)
 - sin(q7) q8 cos(q1) cos(q3) - cos(q7) cos(q9) z34 cos(q1) cos(q3)
 + cos(q2) sin(q1) sin(q7) cos(q9) z34) mu4 omX) qp1 + ((
cos(q9) z34 cos(q1) sin(q2) sin(q3) - sin(q7) sin(q9) z34 sin(q1) sin(q3)
 + cos(q9) z34 sin(q1) cos(q3) + sin(q7) sin(q9) z34 cos(q1) sin(q2) cos(q3))
mu4 omZ + (cos(q7) sin(q9) z34 cos(q1) sin(q2) cos(q3)
 - cos(q7) sin(q9) z34 sin(q1) sin(q3) - cos(q2) cos(q1) cos(q9) z34) mu4 omY
 + (cos(q7) sin(q9) z34 cos(q1) sin(q2) sin(q3)
 + cos(q2) cos(q1) sin(q7) sin(q9) z34 + cos(q7) sin(q9) z34 sin(q1) cos(q3))
mu4 omX) qp9 + (sin(q9) z34 sin(q1) cos(q3)
 + cos(q7) q8 cos(q1) sin(q2) cos(q3)
 - sin(q7) cos(q9) z34 cos(q1) sin(q2) cos(q3)
 + sin(q9) z34 cos(q1) sin(q2) sin(q3) - cos(q7) q8 sin(q1) sin(q3)
 + sin(q7) cos(q9) z34 sin(q1) sin(q3)) mu4 omZp + (z12 sin(q1) sin(q3) + %5
 - cos(q2) cos(q1) sin(q9) z34 - %8 - %7 - z12 cos(q1) sin(q2) cos(q3) + %6)
mu4 omYp + ((mu4 cos(q1) cos(q2) sin(q3) + mu1 cos(q1) cos(q2) sin(q3)) vY
 - (mu4 sin(q2) cos(q1) + mu1 sin(q2) cos(q1)) vZ
 - (mu1 cos(q3) cos(q1) cos(q2) + mu4 cos(q3) cos(q1) cos(q2)) vX + (
cos(q7) q8 cos(q1) cos(q2) cos(q3)
 - sin(q7) cos(q9) z34 cos(q1) cos(q2) cos(q3)
 + sin(q9) z34 cos(q1) cos(q2) sin(q3)) mu4 omZ - (
sin(q7) q8 cos(q1) cos(q2) cos(q3)
 + cos(q7) cos(q9) z34 cos(q1) cos(q2) cos(q3) + z12 cos(q1) cos(q2) cos(q3)
 - sin(q2) cos(q1) sin(q9) z34) mu4 omY - (z12 cos(q1) cos(q2) sin(q3)
 + sin(q2) cos(q1) cos(q7) q8 + sin(q7) q8 cos(q1) cos(q2) sin(q3)
 - sin(q2) cos(q1) sin(q7) cos(q9) z34
 + cos(q7) cos(q9) z34 cos(q1) cos(q2) sin(q3)) mu4 omX) qp2 + (
cos(q7) cos(q9) z34 sin(q1) sin(q2) sin(q3)
```

```
 - cos(q7) cos(q9) z34 cos(q1) cos(q3) + sin(q7) q8 sin(q1) sin(q2) sin(q3)
 - cos(q2) sin(q1) cos(q7) q8 + cos(q2) sin(q1) sin(q7) cos(q9) z34
 - sin(q7) q8 cos(q1) cos(q3)) mu4 qp1 qp7 + (
sin(q2) cos(q1) sin(q7) cos(q9) z34 - sin(q7) q8 cos(q1) cos(q2) sin(q3)
 - cos(q7) cos(q9) z34 cos(q1) cos(q2) sin(q3) - sin(q2) cos(q1) cos(q7) q8)
mu4 qp2 qp7 + (2 cos(q7) sin(q9) z34 cos(q1) sin(q2) sin(q3)
 + 2 cos(q2) cos(q1) sin(q7) sin(q9) z34
 + 2 cos(q7) sin(q9) z34 sin(q1) cos(q3)) mu4 qp9 qp7 + (
cos(q9) z34 sin(q1) sin(q2) cos(q3) + cos(q9) z34 cos(q1) sin(q3)
 - sin(q7) sin(q9) z34 sin(q1) sin(q2) sin(q3)
 + sin(q7) sin(q9) z34 cos(q1) cos(q3) + cos(q2) sin(q1) cos(q7) sin(q9) z34)
mu4 qp1 qp9 - ((%8 + %7 - %6 - %5) mu4 omZ + (
cos(q7) q8 cos(q1) sin(q2) cos(q3) + sin(q7) cos(q9) z34 sin(q1) sin(q3)
 - cos(q7) q8 sin(q1) sin(q3) - sin(q7) cos(q9) z34 cos(q1) sin(q2) cos(q3))
mu4 omY + (cos(q7) q8 sin(q1) cos(q3) - %4 - %3 + cos(q2) cos(q1) sin(q7) q8
 + cos(q2) cos(q1) cos(q7) cos(q9) z34 + cos(q7) q8 cos(q1) sin(q2) sin(q3))
mu4 omX) qp7 + (cos(q9) z34 cos(q1) sin(q2) sin(q3)
 - sin(q7) sin(q9) z34 sin(q1) sin(q3) + cos(q9) z34 sin(q1) cos(q3)
 + sin(q7) sin(q9) z34 cos(q1) sin(q2) cos(q3)) mu4 qp3 qp9
 - (sin(q2) cos(q1) sin(q7) - cos(q7) cos(q1) cos(q2) sin(q3)) mu4 qp8 qp2 + (
cos(q7) cos(q1) cos(q3) - cos(q2) sin(q1) sin(q7)
 - cos(q7) sin(q1) sin(q2) sin(q3)) mu4 qp8 qp1 + (%2 mu4 + %2 mu1) vYp
 + (mu1 cos(q2) cos(q1) + mu4 cos(q2) cos(q1)) vZp - (%1 mu4 + %1 mu1) vXp + (
cos(q9) z34 sin(q1) sin(q3) - cos(q9) z34 cos(q1) sin(q2) cos(q3)
 - cos(q2) cos(q1) cos(q7) sin(q9) z34
 + sin(q7) sin(q9) z34 cos(q1) sin(q2) sin(q3)
 + sin(q7) sin(q9) z34 sin(q1) cos(q3)) mu4 qpp9 + (cos(q7) sin(q1) cos(q3)
 + cos(q7) cos(q1) sin(q2) sin(q3) + cos(q2) cos(q1) sin(q7)) mu4 qpp8 + (
cos(q2) cos(q1) cos(q7) q8 - cos(q7) cos(q9) z34 sin(q1) cos(q3)
 - cos(q2) cos(q1) sin(q7) cos(q9) z34
 - cos(q7) cos(q9) z34 cos(q1) sin(q2) sin(q3)
 - sin(q7) q8 cos(q1) sin(q2) sin(q3) - sin(q7) q8 sin(q1) cos(q3)) mu4 qpp7
%1 :=             cos(q1) sin(q2) cos(q3) - sin(q1) sin(q3)
%2 :=             cos(q1) sin(q2) sin(q3) + sin(q1) cos(q3)
%3 :=            sin(q7) cos(q9) z34 cos(q1) sin(q2) sin(q3)
```

```
%4 :=                sin(q7) cos(q9) z34 sin(q1) cos(q3)

%5 :=                    sin(q7) q8 sin(q1) sin(q3)

%6 :=                cos(q7) cos(q9) z34 sin(q1) sin(q3)

%7 :=            cos(q7) cos(q9) z34 cos(q1) sin(q2) cos(q3)

%8 :=                sin(q7) q8 cos(q1) sin(q2) cos(q3)

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp150:=diff(eq2,qp7):
> tmp151:=subs(varstq,tmp150):
> tmp152:=subs(tmp17 union convert(tmp21,set),tmp151):
> tmp153:=simplify(subs(varstqp,tmp152),trig):
> tmp154:=subs(varslect,tmp153):
#####
> dKdqp7:=
>   rlms(clct(tmp154,
>              [qpp7,qpp8,qpp9],
>              [omXp,omYp,omZp,vXp,vYp,vZp,qp1,qp2,qp3,qp4,qp5,qp6,qp7,qp8,qp9],
>              [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4])):
#####
> dK2dqp7dt:=subs(varslect,diff(tmp153,t)):
#####
> tmp156:=diff(eq2,q7):
> tmp157:=subs(varstq,tmp156):
> tmp158:=subs(tmp17 union convert(tmp21,set),tmp157):
> tmp159:=simplify(tmp158,trig):
bytes used=50913812, alloc=4521156, time=261.550
bytes used=51914936, alloc=4521156, time=266.700
#####
> dKdq7:=
>   rlms(clct(subs(varslect,tmp159),
>              [qpp7,qpp8,qpp9],
>              [omXp,omYp,omZp,vXp,vYp,vZp,qp1,qp2,qp3,qp4,qp5,qp6,qp7,qp8,qp9],
>              [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4])):
#####
> K7:=
>   rlms(clct(simplify(dK2dqp7dt-dKdq7,trig),
>              [qpp7,qpp8,qpp9],
>              [omXp,omYp,omZp,vXp,vYp,vZp,qp1,qp2,qp3,qp4,qp5,qp6,qp7,qp8,qp9],
>              [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4],factor));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

K7 := - (sin(q7) q8 + cos(q7) cos(q9) z34) mu4 vX omZ

    - (cos(q7) q8 - sin(q7) cos(q9) z34) mu4 vX omY + (

   (cos(q7) q8 - sin(q7) cos(q9) z34) mu4 vY

    + (sin(q7) q8 + cos(q7) cos(q9) z34) mu4 vZ) omX

                                          2
    - (sin(q9) z34 sin(q7) q8 + sin(q9) z34  cos(q7) cos(q9)) mu4 omX omY

                                                2
    + (sin(q9) z34 cos(q7) q8 - sin(q7) cos(q9) z34  sin(q9)) mu4 omX omZ - (

                                             2   2      2        2     2
   4 sin(q7) cos(q9) z34 cos(q7) q8 - 2 cos(q7)  q8  - z34  cos(q9)  + q8

              2    2        2
    + 2 cos(q9)  z34  cos(q7)  + z12 cos(q7) cos(q9) z34 + z12 sin(q7) q8) mu4

                             2    2                     2
   omZ omY + (cos(q7) cos(q9)  z34  sin(q7) - sin(q7) q8  cos(q7)
```

```
                                                        2
    - z12 cos(q7) q8 + z12 sin(q7) cos(q9) z34 - 2 cos(q7)  cos(q9) z34 q8

                                  2             2
    + q8 cos(q9) z34) mu4 omY  + (2 cos(q7)  cos(q9) z34 q8

                         2    2                  2
    - cos(q7) cos(q9)  z34  sin(q7) + sin(q7) q8  cos(q7) - q8 cos(q9) z34) mu4

       2                                                     2
    omZ  + (z12 sin(q7) cos(q9) z34 - z12 cos(q7) q8) mu4 omX

         2      2         2
    + (q8  + z34  cos(q9)  + z12 sin(q7) q8 + z12 cos(q7) cos(q9) z34) mu4 omXp

                             2
    + (sin(q7) cos(q9) z34  sin(q9) - sin(q9) z34 cos(q7) q8) mu4 omYp

                                              2
    - (sin(q9) z34 sin(q7) q8 + sin(q9) z34  cos(q7) cos(q9)) mu4 omZp

    + 2 mu4 omX q8 qp8 - (

                                          2    2
   (2 cos(q9) z34 sin(q7) q8 + 2 cos(q9)  z34  cos(q7)) mu4 omZ

                2    2
    - (2 cos(q9)  z34  sin(q7) - 2 cos(q9) z34 cos(q7) q8) mu4 omY

                              2
    + 2 mu4 omX cos(q9) z34  sin(q9)) qp9

    - (sin(q7) q8 + cos(q7) cos(q9) z34) mu4 vYp

    + (cos(q7) q8 - sin(q7) cos(q9) z34) mu4 vZp

                              2
    - 2 mu4 qp7 cos(q9) z34  sin(q9) qp9 + 2 mu4 qp7 q8 qp8

                         2             2      2         2
    - mu4 cos(q9) qp9  z34 q8 + (q8  + z34  cos(q9) ) mu4 qpp7

    - mu4 qpp8 cos(q9) z34 - mu4 sin(q9) qpp9 z34 q8
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp160:=diff(eq2,qp8):
> tmp161:=subs(varstq,tmp160):
> tmp162:=subs(tmp17 union convert(tmp21,set),tmp161):
> tmp163:=simplify(subs(varstqp,tmp162),trig):
> tmp164:=subs(varslect,tmp163):
#####
> dKdqp8:=
>   rlms(clct(tmp164,
>             [qpp7,qpp8,qpp9],
>             [omXp,omYp,omZp,vXp,vYp,vZp,qp1,qp2,qp3,qp4,qp5,qp6,qp7,qp8,qp9],
>             [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4]));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

    dKdqp8 := mu4 cos(q7) vY + mu4 sin(q7) vZ + mu4 cos(q7) omZ sin(q9) z34

         - mu4 sin(q7) sin(q9) z34 omY - (z12 cos(q7) + cos(q9) z34) mu4 omX

         - mu4 qp7 cos(q9) z34 + mu4 qp8

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

```
> dK2dqp8dt:=subs(varslect,diff(tmp163,t));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

    dK2dqp8dt := mu4 sin(q7) qp7 omX z12 - mu4 cos(q7) omXp z12 + mu4 qpp8

         - mu4 sin(q7) qp7 vY + mu4 cos(q7) vYp - mu4 omXp cos(q9) z34

         + mu4 omX sin(q9) qp9 z34 + mu4 cos(q7) qp7 vZ + mu4 sin(q7) vZp

         - mu4 qpp7 cos(q9) z34 + mu4 qp7 sin(q9) qp9 z34

         - mu4 cos(q7) qp7 sin(q9) z34 omY - mu4 sin(q7) cos(q9) qp9 z34 omY

         - mu4 sin(q7) sin(q9) z34 omYp - mu4 sin(q7) qp7 omZ sin(q9) z34

         + mu4 cos(q7) omZp sin(q9) z34 + mu4 cos(q7) omZ cos(q9) qp9 z34

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp166:=diff(eq2,q8):
> tmp167:=subs(varstq,tmp166):
> tmp168:=subs(tmp17 union convert(tmp21,set),tmp167):
> tmp169:=simplify(tmp168,trig):
#####
> dKdq8:=
>   rlms(clct(subs(varslect,tmp169),
>             [qpp7,qpp8,qpp9],
>             [omXp,omYp,omZp,vXp,vYp,vZp,qp1,qp2,qp3,qp4,qp5,qp6,qp7,qp8,qp9],
>             [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4]));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

dKdq8 := - mu4 vX omZ cos(q7) + mu4 vX omY sin(q7)

 + (mu4 vZ cos(q7) - mu4 vY sin(q7)) omX - mu4 sin(q7) omZ sin(q9) z34 omX

 - mu4 cos(q7) sin(q9) z34 omY omX -

                                                                2
(2 sin(q7) q8 cos(q7) + z12 cos(q7) - cos(q9) z34 + 2 cos(q7)  cos(q9) z34) mu4

                                        2
 omZ omY + (q8 + z12 sin(q7)) mu4 omX

           2                                                               2
 - (cos(q7)  q8 - q8 - sin(q7) cos(q9) z34 cos(q7) - z12 sin(q7)) mu4 omY

           2                                         2
 + (cos(q7)  q8 - sin(q7) cos(q9) z34 cos(q7)) mu4 omZ  + (mu4 vZ cos(q7)

 - mu4 vY sin(q7) - mu4 omZ sin(q9) z34 sin(q7) - mu4 sin(q9) z34 omY cos(q7)

 + (2 q8 + z12 sin(q7)) mu4 omX) qp7 - (

mu4 cos(q7) omZ cos(q9) z34 + mu4 omX sin(q9) z34 - mu4 sin(q7) cos(q9) z34 omY

               2
) qp9 + mu4 qp7  q8 - mu4 qp7 sin(q9) qp9 z34

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> K8:=
>   rlms(clct(simplify(dK2dqp8dt-dKdq8,trig),
>             [qpp7,qpp8,qpp9],
>             [omXp,omYp,omZp,vXp,vYp,vZp,qp1,qp2,qp3,qp4,qp5,qp6,qp7,qp8,qp9],
>             [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4],factor));
```

```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

K8 := mu4 vX omZ cos(q7) - mu4 vX omY sin(q7)

     + (mu4 vY sin(q7) - mu4 vZ cos(q7)) omX + mu4 sin(q7) omZ sin(q9) z34 omX

     + mu4 cos(q7) sin(q9) z34 omY omX +

                                                              2
    (2 sin(q7) q8 cos(q7) + z12 cos(q7) - cos(q9) z34 + 2 cos(q7)  cos(q9) z34)

                                                2
     mu4 omZ omY - (q8 + z12 sin(q7)) mu4 omX

                 2                                                              2
     + (cos(q7)  q8 - q8 - sin(q7) cos(q9) z34 cos(q7) - z12 sin(q7)) mu4 omY

                                                2            2
     + (sin(q7) cos(q9) z34 cos(q7) - cos(q7)  q8) mu4 omZ  - 2 mu4 omX q8 qp7

     - (z12 cos(q7) + cos(q9) z34) mu4 omXp - mu4 sin(q7) sin(q9) z34 omYp

     + mu4 cos(q7) omZp sin(q9) z34 - (2 mu4 sin(q7) cos(q9) z34 omY

     - 2 mu4 cos(q7) omZ cos(q9) z34 - 2 mu4 omX sin(q9) z34) qp9

                                                   2
     + mu4 cos(q7) vYp + mu4 sin(q7) vZp - mu4 qp7  q8

     + 2 mu4 qp7 sin(q9) qp9 z34 - mu4 qpp7 cos(q9) z34 + mu4 qpp8

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

```
> tmp170:=diff(eq2,qp9):
> tmp171:=subs(varstq,tmp170):
> tmp172:=subs(tmp17 union convert(tmp21,set),tmp171):
> tmp173:=simplify(subs(varstqp,tmp172),trig):
> tmp174:=subs(varslect,tmp173):
#####
> dKdqp9:=
>   rlms(clct(tmp174,
>             [qpp7,qpp8,qpp9],
>             [omXp,omYp,omZp,vXp,vYp,vZp,qp1,qp2,qp3,qp4,qp5,qp6,qp7,qp8,qp9],
>             [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4])):
#####
> dK2dqp9dt:=subs(varslect,diff(tmp173,t)):
#####
> tmp176:=diff(eq2,q9):
> tmp177:=subs(varstq,tmp176):
> tmp178:=subs(tmp17 union convert(tmp21,set),tmp177):
> tmp179:=simplify(tmp178,trig):
#####
> dKdq9:=
>   rlms(clct(subs(varslect,tmp179),
>             [qpp7,qpp8,qpp9],
>             [omXp,omYp,omZp,vXp,vYp,vZp,qp1,qp2,qp3,qp4,qp5,qp6,qp7,qp8,qp9],
>             [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4])):
#####
> K9:=
>   rlms(clct(simplify(dK2dqp9dt-dKdq9,trig),
>             [qpp7,qpp8,qpp9],
>             [omXp,omYp,omZp,vXp,vYp,vZp,qp1,qp2,qp3,qp4,qp5,qp6,qp7,qp8,qp9],
>             [omX,omY,omZ],[vX,vY,vZ],[mu1,mu4],factor));
```

```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

K9 := (mu4 vX sin(q7) sin(q9) z34 - mu4 cos(q9) z34 vY) omZ

   + (z34 mu4 sin(q9) vX cos(q7) + z34 mu4 vZ cos(q9)) omY
```

```
   - (mu4 vZ sin(q7) sin(q9) z34 + mu4 vY cos(q7) sin(q9) z34) omX + (

          2    2                                                  2
  2 cos(q9)  z34  cos(q7) + cos(q9) z34 sin(q7) q8 - cos(q7) z34

                                              2
   + cos(q9) z34 z12) mu4 omZ omX - (2 cos(q7)  sin(q9) z34 q8

                                                 2
   - z12 sin(q7) sin(q9) z34 - 2 cos(q7) cos(q9) z34  sin(q7) sin(q9)

   - q8 sin(q9) z34) mu4 omY omZ -

           2    2              2
  (2 cos(q9)  z34  sin(q7) - z34  sin(q7) - cos(q9) z34 cos(q7) q8) mu4 omX omY

                2        2                                                 2
   - (cos(q9) z34  cos(q7)  sin(q9) + sin(q7) sin(q9) z34 cos(q7) q8) mu4 omZ

                                          2
   + (sin(q7) sin(q9) z34 cos(q7) q8 - z34  cos(q9) sin(q9)

              2        2                                              2
   + cos(q9) z34  cos(q7)  sin(q9) + z34 sin(q9) z12 cos(q7)) mu4 omY

         2                                                  2
   + (z34  cos(q9) sin(q9) + z34 sin(q9) z12 cos(q7)) mu4 omX  + (

                                       2    2
  (2 cos(q9) z34 sin(q7) q8 + 2 cos(q9)  z34  cos(q7)) mu4 omZ

                                       2    2
   + (2 cos(q9) z34 cos(q7) q8 - 2 cos(q9)  z34  sin(q7)) mu4 omY

                          2
   + 2 mu4 omX cos(q9) z34  sin(q9)) qp7

   - (q8 sin(q9) z34 + z12 sin(q7) sin(q9) z34) mu4 omXp

                                          2
   + (cos(q9) z34 sin(q7) q8 + cos(q7) z34  + cos(q9) z34 z12) mu4 omYp

         2
   + (z34  sin(q7) - cos(q9) z34 cos(q7) q8) mu4 omZp + (

  2 mu4 sin(q7) cos(q9) z34 omY - 2 mu4 cos(q7) omZ cos(q9) z34

   - 2 mu4 omX sin(q9) z34) qp8 + z34 mu4 cos(q9) vXp

   + z34 mu4 sin(q7) sin(q9) vYp - z34 mu4 cos(q7) sin(q9) vZp

        2        2
   + z34  mu4 qp7  cos(q9) sin(q9) - 2 z34 mu4 sin(q9) qp7 qp8

                                    2
   - z34 mu4 sin(q9) qpp7 q8 + z34  mu4 qpp9

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

bytes used=61512072, alloc=4521156, time=315.666
```

A.5 equations de Hamilton

A.5.1 mouvement relatif à $\mathcal{R}_1$ (fichier `hamilton1`)

```
> read intro:
#####
>
> c12:=vector([C12,0,0]):
> f:=vector([0,F23,0]):
> c34:=vector([0,C34,0]):
#####
>
> g1m34:=add(g1m12,multiply(Rot12,m21m34)):
#####
> v34:=map(diff,g1m34,t):
#####
> g1g4:=add(add(g1m12,multiply(Rot12,m21m34)),multiply(Rot14,m43g4)):
#####
> v4:=map(diff,g1g4,t):
#####
> om2:=vector([diff(q7(t),t),0,0]):
#####
> om4:=add(om2,multiply(Rot12,vector([0,diff(q9(t),t),0]))):
#####
>
> tmp35:=subs(varslect,(dotprod(v4,v4)*mu4)/2):
#####
> tmp36:=dotprod(c12,om2):
> tmp37:=dotprod(multiply(Rot12,f),v34)-dotprod(multiply(Rot12,c34),om2):
> tmp38:=dotprod(multiply(Rot12,c34),om4)+dotprod(f4,v4):
> tmp39:=subs(varslect,tmp36+tmp37+tmp38):
#####
>
> tmp40:=simplify(subs({qp8=0,qp9=0},tmp39),trig):
#####
> Q7:=collect(coeff(collect(tmp40,qp7),qp7),[f4X,f4Y,f4Z]);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

              Q7 := (- sin(q7) q8 - cos(q7) cos(q9) z34) f4Y

                   + (cos(q7) q8 - sin(q7) cos(q9) z34) f4Z + C12

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp41:=simplify(subs({qp7=0,qp9=0},tmp39),trig):
#####
> Q8:=collect(coeff(collect(tmp41,qp8),qp8),[f4X,f4Y,f4Z]);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                     Q8 := F23 + f4Y cos(q7) + f4Z sin(q7)

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp42:=simplify(subs({qp7=0,qp8=0},tmp39),trig):
#####
> Q9:=collect(coeff(collect(tmp42,qp9),qp9),[f4X,f4Y,f4Z]);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

Q9 := C34 + f4X cos(q9) z34 + f4Y sin(q7) sin(q9) z34 - f4Z cos(q7) sin(q9) z34

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp45:=int(Q7,q7):
> tmp46:=tmp45+int(Q8-diff(tmp45,q8),q8):

#####
```

```
> V:=tmp46+int(Q9-diff(tmp46,q9),q9);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

    V := (cos(q7) q8 - sin(q7) cos(q9) z34) f4Y

         + (sin(q7) q8 + cos(q7) cos(q9) z34) f4Z + C12 q7 + F23 q8 + C34 q9

         + sin(q9) f4X z34

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp50:=diff(tmp35,qp7):
> tmp51:=diff(tmp35,qp8):
> tmp52:=diff(tmp35,qp9):
#####
> A:=map(simplify,genmatrix([tmp50=0,tmp51=0,tmp52=0],[qp7,qp8,qp9]),trig);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

      [       2      2            2                                         ]
      [ mu4 q8  + z34  mu4 cos(q9)   - cos(q9) z34 mu4  - sin(q9) z34 q8 mu4 ]
      [                                                                     ]
 A := [      - cos(q9) z34 mu4              mu4                  0          ]
      [                                                                     ]
      [                                                          2          ]
      [     - sin(q9) z34 q8 mu4             0                z34  mu4      ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> p:=matrix([[p7],[p8],[p9]]):
> tmp55:=det(A):
#####
> DET:=simplify(tmp55,trig);
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                                 3    2   2        2
                        DET := mu4  z34  q8  cos(q9)

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp60:=linsolve(A,p):
#####
> qp = map(simplify,tmp60,trig);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

           [                                   2                     ]
           [          sin(q9) q8 p9 + cos(q9) z34  p8 + p7 z34        ]
           [          ----------------------------------------        ]
           [                        2          2                     ]
           [                  mu4 q8  z34 cos(q9)                    ]
           [                                                         ]
           [                          2                    2         ]
           [ sin(q9) q8 p9 + cos(q9) z34  p8 + p7 z34 + p8 q8  cos(q9) ]
      qp = [ ------------------------------------------------------- ]
           [                                  2                      ]
           [                    cos(q9) mu4 q8                       ]
           [                                                         ]
           [                        2                                ]
           [      sin(q9) cos(q9) z34  p8 + sin(q9) p7 z34 + p9 q8    ]
           [      ------------------------------------------------    ]
           [                            2          2                 ]
           [                  mu4 q8 z34  cos(q9)                    ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> L:=tmp35-V:
#####
```

```
> tmp65:=p7*qp7+p8*qp8+p9*qp9-L:
> tmp66:=subs([qp7=tmp60[1,1],qp8=tmp60[2,1],qp9=tmp60[3,1]],tmp65):
#####
> H:=simplify(tmp66,trig);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                             3    2                  2
  H := 1/2 (2 f4Y mu4 q8  z34  cos(q7) cos(q9)

                  3       2                2     2    4        2
       + 2 f4X z34  mu4 q8  sin(q9) cos(q9)  + p8  z34  cos(q9)

                                 2                                      2   2
       + 2 sin(q9) cos(q9) z34  p8 p9 q8 + 2 sin(q9) p7 z34 p9 q8 + p9  q8

            2    2           3                 2    2   2        2
       + p7  z34  + 2 p7 z34  cos(q9) p8 + p8  z34  q8  cos(q9)

                             2    2        2           3       2        2
       + 2 C34 q9 mu4 q8  z34  cos(q9)  + 2 F23 q8  mu4 z34  cos(q9)

                             2    2        2            3    2                 2
       + 2 C12 q7 mu4 q8  z34  cos(q9)  + 2 f4Z mu4 q8  z34  sin(q7) cos(q9)

                          2    3                3
       + 2 f4Z mu4 q8  z34  cos(q7) cos(q9)

                          2    3                3    /         2       2    2
       - 2 f4Y mu4 q8  z34  sin(q7) cos(q9) )  /  (cos(q9)  mu4 q8  z34 )
                                              /

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> eq4a:=simplify(diff(H,p7));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                                                       2
                     sin(q9) q8 p9 + cos(q9) z34  p8 + p7 z34
            eq4a := ----------------------------------------
                                            2          2
                                 z34 mu4 q8  cos(q9)

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> eq4b:=simplify(diff(H,p8));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                        2                                      2
            cos(q9) z34  p8 + sin(q9) q8 p9 + p7 z34 + p8 q8  cos(q9)
   eq4b := ---------------------------------------------------------
                                                 2
                                  cos(q9) mu4 q8

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> eq4c:=simplify(diff(H,p9));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                                    2
               sin(q9) cos(q9) z34  p8 + sin(q9) p7 z34 + p9 q8
       eq4c := ------------------------------------------------
                                           2          2
                                q8 mu4 z34  cos(q9)

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

```
> eq5a:=simplify(-diff(H,q7));
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
    eq5a := f4Y q8 sin(q7) - C12 - f4Z q8 cos(q7) + f4Z z34 sin(q7) cos(q9)

         + f4Y z34 cos(q7) cos(q9)
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
> eq5b:=simplify(-diff(H,q8));
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                         3                    2           2
 eq5b := - (f4Y mu4 q8  z34 cos(q7) cos(q9)  - 2 p7 z34  cos(q9) p8

          2    3         2
      - p8  z34  cos(q9)  - sin(q9) cos(q9) z34 p8 p9 q8 - sin(q9) p7 p9 q8

          2              3                 2             3                    2
      - p7  z34 + F23 q8  mu4 z34 cos(q9)  + f4Z mu4 q8  z34 sin(q7) cos(q9)  )

        /             2       3
       /  (z34 cos(q9)  mu4 q8 )
      /
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
> eq5c:=simplify(-diff(H,q9));
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                          2   2                2    2           2
  eq5c := - (sin(q9) p9  q8  + sin(q9) p7  z34  - cos(q9)  p7 z34 p9 q8

                     2                                            3
       + cos(q9) z34  p8 p9 q8 + 2 p7 z34 p9 q8 + sin(q9) p7 z34  cos(q9) p8

                            2    3                3
       - sin(q9) f4Z mu4 q8  z34  cos(q7) cos(q9)

                            2    3                3      3        4           2
       + sin(q9) f4Y mu4 q8  z34  sin(q7) cos(q9)  + z34  cos(q9)  f4X mu4 q8

           2        3           2    /         3       2    2
       + z34  cos(q9)  C34 mu4 q8 )  /  (cos(q9)  mu4 q8  z34 )
                                    /
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
bytes used=5173092, alloc=1376004, time=22.966
```

A.5.2 mouvement par rapport à R (fichier hamilton2)

```
> read intro:
#####
> g1g4:=add(add(g1m12,multiply(Rot12,m21m34)),multiply(Rot14,m43g4)):
#####
> tmp35:=map(diff,g1g4,t):
> tmp36:=crossprod(om1,g1g4):
> v4:=add(v1,add(tmp35,tmp36)):
#####

> tmp40:=
>   (dotprod(v1,v1)*mu1+dotprod(multiply(i1,om1),om1)+dotprod(v4,v4)*mu4)/2:
> tmp41:=subs(tmp25 union tmp32,tmp40):
> tmp42:=subs(varslect,tmp41):
> tmp43:=diff(tmp42,qp1):
> tmp44:=diff(tmp42,qp2):
> tmp45:=diff(tmp42,qp3):
> tmp46:=diff(tmp42,qp4):
> tmp47:=diff(tmp42,qp5):
> tmp48:=diff(tmp42,qp6):
> tmp49:=diff(tmp42,qp7):
> tmp50:=diff(tmp42,qp8):
> tmp51:=diff(tmp42,qp9):
> tmp52:=
>   genmatrix([tmp43=0,tmp44=0,tmp45=0,tmp46=0,tmp47=0,tmp48=0,tmp49=0,tmp50=0,
>              tmp51=0],[qp1,qp2,qp3,qp4,qp5,qp6,qp7,qp8,qp9]);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

tmp52 :=

               2         2               2         2               2
[i1X cos(q2)  cos(q3)  + i1Y cos(q2)  sin(q3)  + i1Z sin(q2)

               2       2       2
    + 1/2 (2 %8  + 2 %6  + 2 %3 ) mu4, i1X sin(q3) cos(q2) cos(q3)

    - i1Y cos(q3) cos(q2) sin(q3)

    + 1/2 (2 cos(q3) %5 %8 - 2 sin(q3) %5 %6 + 2 %9 %3) mu4,

   i1Z sin(q2) + 1/2 (2 (- cos(q7) q8 + %2) %8 + 2 sin(q9) z34 %6) mu4,

   1/2 (2 cos(q2) cos(q3) %8 - 2 cos(q2) sin(q3) %6 + 2 sin(q2) %3) mu4,

   1/2 (2 %11 %8 + 2 %10 %6 - 2 cos(q2) sin(q1) %3) mu4,

   1/2 (2 %13 %8 + 2 %12 %6 + 2 cos(q2) cos(q1) %3) mu4,

   1/2 (2 (- sin(q7) q8 - %4) %6 + 2 (cos(q7) q8 - %2) %3) mu4,

   1/2 (2 cos(q7) %6 + 2 sin(q7) %3) mu4, 1/2

   (2 cos(q9) z34 %8 + 2 sin(q7) sin(q9) z34 %6 - 2 cos(q7) sin(q9) z34 %3) mu4

   ]

[i1X sin(q3) cos(q2) cos(q3) - i1Y cos(q3) cos(q2) sin(q3)

 + 1/2 (2 cos(q3) %5 %8 - 2 sin(q3) %5 %6 + 2 %9 %3) mu4,

            2              2                   2   2              2   2       2
i1X sin(q3)  + i1Y cos(q3)  + 1/2 (2 cos(q3)  %5  + 2 sin(q3)  %5  + 2 %9 ) mu4

, 1/2 (2 (- cos(q7) q8 + %2) cos(q3) %5 - 2 sin(q9) z34 sin(q3) %5) mu4,

                         2                           2
1/2 (2 cos(q2) cos(q3)  %5 + 2 cos(q2) sin(q3)  %5 + 2 sin(q2) %9) mu4,
```

```
1/2 (2 %11 cos(q3) %5 - 2 %10 sin(q3) %5 - 2 cos(q2) sin(q1) %9) mu4,

1/2 (2 %13 cos(q3) %5 - 2 %12 sin(q3) %5 + 2 cos(q2) cos(q1) %9) mu4,

1/2 (- 2 (- sin(q7) q8 - %4) sin(q3) %5 + 2 (cos(q7) q8 - %2) %9) mu4,

1/2 (- 2 cos(q7) sin(q3) %5 + 2 sin(q7) %9) mu4, 1/2 (2 cos(q9) z34 cos(q3) %5

 - 2 sin(q7) sin(q9) z34 sin(q3) %5 - 2 cos(q7) sin(q9) z34 %9) mu4]

  [i1Z sin(q2) + 1/2 (2 (- cos(q7) q8 + %2) %8 + 2 sin(q9) z34 %6) mu4,

      1/2 (2 (- cos(q7) q8 + %2) cos(q3) %5 - 2 sin(q9) z34 sin(q3) %5) mu4,

                                        2             2    2
      i1Z + 1/2 (2 (- cos(q7) q8 + %2)  + 2 sin(q9)  z34 ) mu4,

      1/2 (2 cos(q2) cos(q3) (- cos(q7) q8 + %2) - 2 %1) mu4,

      1/2 (2 %11 (- cos(q7) q8 + %2) + 2 %10 sin(q9) z34) mu4,

      1/2 (2 %13 (- cos(q7) q8 + %2) + 2 %12 sin(q9) z34) mu4,

      (- sin(q7) q8 - %4) sin(q9) z34 mu4, cos(q7) sin(q9) z34 mu4,

                                                                2    2
      1/2 (2 cos(q9) z34 (- cos(q7) q8 + %2) + 2 sin(q7) sin(q9)  z34 ) mu4]

 [1/2 (2 cos(q2) cos(q3) %8 - 2 cos(q2) sin(q3) %6 + 2 sin(q2) %3) mu4,

                            2                             2
     1/2 (2 cos(q2) cos(q3)  %5 + 2 cos(q2) sin(q3)  %5 + 2 sin(q2) %9) mu4,

     1/2 (2 cos(q2) cos(q3) (- cos(q7) q8 + %2) - 2 %1) mu4,

                  2        2            2        2            2
     1/2 (2 cos(q2)  cos(q3)  + 2 cos(q2)  sin(q3)  + 2 sin(q2) ) mu1

                     2        2            2        2            2
      + 1/2 (2 cos(q2)  cos(q3)  + 2 cos(q2)  sin(q3)  + 2 sin(q2) ) mu4,

     1/2 %14 mu1 + 1/2 %14 mu4, 1/2 %15 mu1 + 1/2 %15 mu4,

     1/2 (- 2 (- sin(q7) q8 - %4) cos(q2) sin(q3) + 2 %7) mu4,

     1/2 (- 2 cos(q7) cos(q2) sin(q3) + 2 sin(q7) sin(q2)) mu4, 1/2 (

     2 cos(q9) z34 cos(q2) cos(q3) - 2 sin(q7) sin(q9) z34 cos(q2) sin(q3)

      - 2 cos(q7) sin(q9) z34 sin(q2)) mu4]

[1/2 (2 %11 %8 + 2 %10 %6 - 2 cos(q2) sin(q1) %3) mu4,

    1/2 (2 %11 cos(q3) %5 - 2 %10 sin(q3) %5 - 2 cos(q2) sin(q1) %9) mu4,

    1/2 (2 %11 (- cos(q7) q8 + %2) + 2 %10 sin(q9) z34) mu4,

                                          2        2            2        2
    1/2 %14 mu1 + 1/2 %14 mu4, 1/2 (2 %11  + 2 %10  + 2 cos(q2)  sin(q1) ) mu1

                 2        2            2        2
     + 1/2 (2 %11  + 2 %10  + 2 cos(q2)  sin(q1) ) mu4,

    1/2 %16 mu1 + 1/2 %16 mu4,

    1/2 (2 (- sin(q7) q8 - %4) %10 - 2 (cos(q7) q8 - %2) cos(q2) sin(q1)) mu4,

    1/2 (2 cos(q7) %10 - 2 sin(q7) cos(q2) sin(q1)) mu4, 1/2 (2 cos(q9) z34 %11

     + 2 sin(q7) sin(q9) z34 %10 + 2 cos(q7) sin(q9) z34 cos(q2) sin(q1)) mu4]
```

```
[1/2 (2 %13 %8 + 2 %12 %6 + 2 cos(q2) cos(q1) %3) mu4,
    1/2 (2 %13 cos(q3) %5 - 2 %12 sin(q3) %5 + 2 cos(q2) cos(q1) %9) mu4,
    1/2 (2 %13 (- cos(q7) q8 + %2) + 2 %12 sin(q9) z34) mu4,
    1/2 %15 mu1 + 1/2 %15 mu4, 1/2 %16 mu1 + 1/2 %16 mu4,
              2        2            2        2
    1/2 (2 %13  + 2 %12  + 2 cos(q2)  cos(q1) ) mu1
                 2        2            2        2
     + 1/2 (2 %13  + 2 %12  + 2 cos(q2)  cos(q1) ) mu4,
    1/2 (2 (- sin(q7) q8 - %4) %12 + 2 (cos(q7) q8 - %2) cos(q2) cos(q1)) mu4,
    1/2 (2 cos(q7) %12 + 2 sin(q7) cos(q2) cos(q1)) mu4, 1/2 (2 cos(q9) z34 %13
     + 2 sin(q7) sin(q9) z34 %12 - 2 cos(q7) sin(q9) z34 cos(q2) cos(q1)) mu4]
[1/2 (2 (- sin(q7) q8 - %4) %6 + 2 (cos(q7) q8 - %2) %3) mu4,
    1/2 (- 2 (- sin(q7) q8 - %4) sin(q3) %5 + 2 (cos(q7) q8 - %2) %9) mu4,
    (- sin(q7) q8 - %4) sin(q9) z34 mu4,
    1/2 (- 2 (- sin(q7) q8 - %4) cos(q2) sin(q3) + 2 %7) mu4,
    1/2 (2 (- sin(q7) q8 - %4) %10 - 2 (cos(q7) q8 - %2) cos(q2) sin(q1)) mu4,
    1/2 (2 (- sin(q7) q8 - %4) %12 + 2 (cos(q7) q8 - %2) cos(q2) cos(q1)) mu4,
                              2                        2
    1/2 (2 (- sin(q7) q8 - %4)  + 2 (cos(q7) q8 - %2) ) mu4,
    1/2 (2 cos(q7) (- sin(q7) q8 - %4) + 2 sin(q7) (cos(q7) q8 - %2)) mu4, 1/2
    (2 sin(q7) sin(q9) z34 (- sin(q7) q8 - %4)
     - 2 cos(q7) sin(q9) z34 (cos(q7) q8 - %2)) mu4]
[1/2 (2 cos(q7) %6 + 2 sin(q7) %3) mu4,
    1/2 (- 2 cos(q7) sin(q3) %5 + 2 sin(q7) %9) mu4, cos(q7) sin(q9) z34 mu4,
    1/2 (- 2 cos(q7) cos(q2) sin(q3) + 2 sin(q7) sin(q2)) mu4,
    1/2 (2 cos(q7) %10 - 2 sin(q7) cos(q2) sin(q1)) mu4,
    1/2 (2 cos(q7) %12 + 2 sin(q7) cos(q2) cos(q1)) mu4,
    1/2 (2 cos(q7) (- sin(q7) q8 - %4) + 2 sin(q7) (cos(q7) q8 - %2)) mu4,
              2            2
    1/2 (2 cos(q7)  + 2 sin(q7) ) mu4, 0]
[1/2
(2 cos(q9) z34 %8 + 2 sin(q7) sin(q9) z34 %6 - 2 cos(q7) sin(q9) z34 %3) mu4,
1/2 (2 cos(q9) z34 cos(q3) %5 - 2 sin(q7) sin(q9) z34 sin(q3) %5
 - 2 cos(q7) sin(q9) z34 %9) mu4,
                                                          2    2
1/2 (2 cos(q9) z34 (- cos(q7) q8 + %2) + 2 sin(q7) sin(q9)  z34 ) mu4, 1/2 (
2 cos(q9) z34 cos(q2) cos(q3) - 2 sin(q7) sin(q9) z34 cos(q2) sin(q3)
```

```
 - 2 cos(q7) sin(q9) z34 sin(q2)) mu4, 1/2 (2 cos(q9) z34 %11

 + 2 sin(q7) sin(q9) z34 %10 + 2 cos(q7) sin(q9) z34 cos(q2) sin(q1)) mu4, 1/2

(2 cos(q9) z34 %13 + 2 sin(q7) sin(q9) z34 %12

 - 2 cos(q7) sin(q9) z34 cos(q2) cos(q1)) mu4, 1/2 (

2 sin(q7) sin(q9) z34 (- sin(q7) q8 - %4)

 - 2 cos(q7) sin(q9) z34 (cos(q7) q8 - %2)) mu4, 0,

              2    2            2        2    2            2        2    2
1/2 (2 cos(q9)  z34  + 2 sin(q7)  sin(q9)  z34  + 2 cos(q7)  sin(q9)  z34 ) mu4

]


%1 :=                    cos(q2) sin(q3) sin(q9) z34

%2 :=                        sin(q7) cos(q9) z34

%3 :=              cos(q2) cos(q3) (cos(q7) q8 - %2) + %1

%4 :=                        cos(q7) cos(q9) z34

%5 :=                       z12 + sin(q7) q8 + %4

%6 :=             sin(q2) sin(q9) z34 - cos(q2) cos(q3) %5

%7 :=                      sin(q2) (cos(q7) q8 - %2)

%8 :=                      - cos(q2) sin(q3) %5 - %7

%9 :=          sin(q3) (cos(q7) q8 - %2) - cos(q3) sin(q9) z34

%10 :=           - sin(q1) sin(q2) sin(q3) + cos(q1) cos(q3)

%11 :=            sin(q1) sin(q2) cos(q3) + cos(q1) sin(q3)

%12 :=            cos(q1) sin(q2) sin(q3) + sin(q1) cos(q3)

%13 :=           - cos(q1) sin(q2) cos(q3) + sin(q1) sin(q3)

%14 :=

    2 %11 cos(q2) cos(q3) - 2 %10 cos(q2) sin(q3) - 2 cos(q2) sin(q1) sin(q2)

%15 :=

    2 %13 cos(q2) cos(q3) - 2 %12 cos(q2) sin(q3) + 2 cos(q2) cos(q1) sin(q2)

                                                 2
%16 :=       2 %13 %11 + 2 %12 %10 - 2 cos(q2)  cos(q1) sin(q1)

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp53:=matrix([[p1],[p2],[p3],[p4],[p5],[p6],[p7],[p8],[p9]]):
#####
> tmp54:=det(tmp52):
#####
> DET:=simplify(tmp54,trig);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                           2            2    2   2        2    2
          DET := i1Y cos(q2)  i1x i1z mu1  mu4  q8  cos(q9)  z34

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

```
bytes used=797730384, alloc=18412244, time=9278.100

#####
RES:=linsolve(tmp52,tmp53);
#####

bytes used=982889932, alloc=52222628, time=12308.733
System error, object too large
```

A.6 formalisme de Kane

A.6.1 calculs préliminaires (fichier kaneintro)

```
> with(linalg):
#####

> varslect:=[diff(diff(q1(t),t),t)=qpp1,diff(diff(q2(t),t),t)=qpp2,
>            diff(diff(q3(t),t),t)=qpp3,diff(diff(q4(t),t),t)=qpp4,
>            diff(diff(q5(t),t),t)=qpp5,diff(diff(q6(t),t),t)=qpp6,
>            diff(diff(q7(t),t),t)=qpp7,diff(diff(q8(t),t),t)=qpp8,
>            diff(diff(q9(t),t),t)=qpp9,
>            diff(q1(t),t)=qp1,diff(q2(t),t)=qp2,diff(q3(t),t)=qp3,
>            diff(q4(t),t)=qp4,diff(q5(t),t)=qp5,diff(q6(t),t)=qp6,
>            diff(q7(t),t)=qp7,diff(q8(t),t)=qp8,diff(q9(t),t)=qp9,
>            q1(t)=q1,q2(t)=q2,q3(t)=q3,q4(t)=q4,q5(t)=q5,q6(t)=q6,
>            q7(t)=q7,q8(t)=q8,q9(t)=q9,
>            diff(diff(omX(t),t),t)=omXpp,diff(diff(omY(t),t),t)=omYpp,
>            diff(diff(omZ(t),t),t)=omZpp,
>            diff(diff(vX(t),t),t)=vXpp,diff(diff(vY(t),t),t)=vYpp,
>            diff(diff(vZ(t),t),t)=vZpp,
>            diff(omX(t),t)=omXp,diff(omY(t),t)=omYp,diff(omZ(t),t)=omZp,
>            diff(vX(t),t)=vXp,diff(vY(t),t)=vYp,diff(vZ(t),t)=vZp,
>            omX(t)=omX,omY(t)=omY,omZ(t)=omZ,vX(t)=vX,vY(t)=vY,vZ(t)=vZ]:
#####

> U1:=cos(q2)*cos(q3)*qp1+sin(q3)*qp2;

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                    U1 := cos(q2) cos(q3) qp1 + sin(q3) qp2

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> U2:=-cos(q2)*sin(q3)*qp1+cos(q3)*qp2;

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                   U2 := - cos(q2) sin(q3) qp1 + cos(q3) qp2

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> U3:=sin(q2)*qp1+qp3;

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                           U3 := sin(q2) qp1 + qp3

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> U4:=
>   cos(q2)*cos(q3)*qp4+(sin(q1)*sin(q2)*cos(q3)+cos(q1)*sin(q3))*qp5
>   -(cos(q1)*sin(q2)*cos(q3)-sin(q1)*sin(q3))*qp6;

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

  U4 := cos(q2) cos(q3) qp4 + (sin(q1) sin(q2) cos(q3) + cos(q1) sin(q3)) qp5

       - (cos(q1) sin(q2) cos(q3) - sin(q1) sin(q3)) qp6

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> U5:=
>   -cos(q2)*sin(q3)*qp4-(sin(q1)*sin(q2)*sin(q3)-cos(q1)*cos(q3))*qp5
>   +(cos(q1)*sin(q2)*sin(q3)+sin(q1)*cos(q3))*qp6;
```

```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

 U5 := - cos(q2) sin(q3) qp4 - (sin(q1) sin(q2) sin(q3) - cos(q1) cos(q3)) qp5

      + (cos(q1) sin(q2) sin(q3) + sin(q1) cos(q3)) qp6

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> U6:=sin(q2)*qp4-sin(q1)*cos(q2)*qp5+cos(q1)*cos(q2)*qp6;

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

         U6 := sin(q2) qp4 - sin(q1) cos(q2) qp5 + cos(q1) cos(q2) qp6

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> U7:=q8*qp7-sin(q9)*z34*qp9;

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                         U7 := q8 qp7 - sin(q9) z34 qp9

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> U8:=-cos(q9)*z34*qp7+qp8;

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                         U8 := - cos(q9) z34 qp7 + qp8

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> U9:=z34*qp9;

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                                 U9 := z34 qp9

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp5:=
>   genmatrix([U1,U2,U3,U4,U5,U6,U7,U8,U9],
>             [qp1,qp2,qp3,qp4,qp5,qp6,qp7,qp8,qp9]):
> tmp6:=vector([u1,u2,u3,u4,u5,u6,u7,u8,u9]):
> tmp7:=linsolve(tmp5,tmp6):
#####
> DET:=simplify(det(tmp5),trig);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                             DET := cos(q2) q8 z34

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> eq1:=map(simplify,tmp7,trig);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

             - cos(q3) u1 + u2 sin(q3)
  eq1 := [ - --------------------------, cos(q3) u2 + u1 sin(q3),
                      cos(q2)

      - sin(q2) cos(q3) u1 + sin(q2) u2 sin(q3) + u3 cos(q2)
      ------------------------------------------------------,
                              cos(q2)

      - cos(q2) sin(q3) u5 + cos(q2) u4 cos(q3) + u6 sin(q2),

      cos(q1) cos(q3) u5 - sin(q1) cos(q2) u6 - sin(q1) sin(q2) sin(q3) u5
```

```
          + u4 cos(q1) sin(q3) + u4 sin(q2) sin(q1) cos(q3),

      cos(q2) u6 cos(q1) + sin(q3) u5 sin(q2) cos(q1) + u4 sin(q1) sin(q3)

          - u4 sin(q2) cos(q3) cos(q1) + sin(q1) cos(q3) u5,

      sin(q9) u9 + u7  cos(q9) z34 sin(q9) u9 + cos(q9) z34 u7 + u8 q8   u9
      ---------------, ---------------------------------------------, --- ]
            q8                              q8                          z34

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> Rotq1:=matrix([[1,0,0],[0,cos(q1(t)),-sin(q1(t))],[0,sin(q1(t)),cos(q1(t))]]):
>
> Rotq2:=matrix([[cos(q2(t)),0,sin(q2(t))],[0,1,0],[-sin(q2(t)),0,cos(q2(t))]]):
>
> Rotq3:=matrix([[cos(q3(t)),-sin(q3(t)),0],[sin(q3(t)),cos(q3(t)),0],[0,0,1]]):
>
> Rot1:=multiply(Rotq1,Rotq2,Rotq3):
> Rot12:=subs(q1(t)=q7(t),op(Rotq1)):
> Rot34:=subs(q2(t)=q9(t),op(Rotq2)):
> Rot1p:=map(diff,Rot1,t):
#####
> tmp10:=multiply(Rot1p,transpose(Rot1)):
> tmp11:=map(simplify,tmp10,trig):
> tmp12:=vector([tmp11[3,2],tmp11[1,3],tmp11[2,1]]):
> tmp13:=multiply(transpose(Rot1),tmp12):
> tmp14:=map(simplify,tmp13,trig):
> tmp15:=
>   subs({diff(q1(t),t)=qp1,diff(q2(t),t)=qp2,diff(q3(t),t)=qp3},op(tmp14)):
> tmp16:=subs({qp1=eq1[1],qp2=eq1[2],qp3=eq1[3]},op(tmp15)):
> tmp17:=subs(varslect,op(tmp16)):
#####
> om1:=map(simplify,tmp17,trig);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                             om1 := [ u1, u2, u3 ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp20:=add(om1,vector([qp7,0,0])):
#####
> om2:=subs(qp7=eq1[7],op(tmp20));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                                  sin(q9) u9 + u7
                    om2 := [ u1 + ---------------, u2, u3 ]
                                        q8

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp25:=add(om2,multiply(Rot12,vector([0,qp9,0]))):
> tmp26:=subs(varslect,op(tmp25)):
#####
> om4:=subs(qp9=eq1[9],op(tmp26));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                    sin(q9) u9 + u7       cos(q7) u9       sin(q7) u9
       om4 := [ u1 + ---------------, u2 + ----------, u3 + ---------- ]
                           q8                  z34              z34

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> Og1:=vector([q4(t),q5(t),q6(t)]):
> tmp30:=map(diff,Og1,t):
> tmp31:=multiply(transpose(Rot1),tmp30):
```

```
> tmp32:=subs(varslect,op(tmp31)):
> tmp33:=subs({qp4=eq1[4],qp5=eq1[5],qp6=eq1[6]},op(tmp32)):
#####
> v1:=map(simplify,tmp33,trig);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                              v1 := [ u4, u5, u6 ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> g1m12:=vector([0,0,z12]):
#####
> v12:=add(v1,crossprod(om1,g1m12));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                    v12 := [ u4 + u2 z12, u5 - u1 z12, u6 ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> m21m34:=vector([0,q8(t),0]):
#####
> g1m34:=add(g1m12,multiply(Rot12,m21m34)):
#####
> tmp35:=map(diff,g1m34,t):
> tmp36:=crossprod(om1,g1m34):
> tmp37:=add(v1,add(tmp35,tmp36)):
> tmp38:=subs(varslect,op(tmp37)):
> tmp39:=subs({qp7=eq1[7],qp8=eq1[8]},op(tmp38)):
#####
> v34:=map(simplify,tmp39,trig);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

v34 := [ u4 + u2 z12 + u2 sin(q7) q8 - u3 cos(q7) q8,

- (- u5 q8 + sin(q7) q8 sin(q9) u9 + sin(q7) q8 u7

     - cos(q7) cos(q9) z34 sin(q9) u9 - cos(q7) cos(q9) z34 u7 - cos(q7) u8 q8

                        2
     + u1 q8 z12 + u1 q8  sin(q7))/q8,

(u6 q8 + cos(q7) q8 sin(q9) u9 + cos(q7) q8 u7 + sin(q7) cos(q9) z34 sin(q9) u9

                                                            2
     + sin(q7) cos(q9) z34 u7 + sin(q7) u8 q8 + u1 cos(q7) q8 )/q8

 ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> Rot14:=multiply(Rot12,Rot34):
#####

> m43g4:=vector([0,0,z34]):
#####
> g1g4:=add(g1m34,multiply(Rot14,m43g4)):
#####
> tmp40:=map(diff,g1g4,t):
> tmp41:=crossprod(om1,g1g4):
> tmp42:=add(v1,add(tmp40,tmp41)):
> tmp43:=subs(varslect,op(tmp42)):
> tmp44:=subs({qp7=eq1[7],qp8=eq1[8],qp9=eq1[9]},op(tmp43)):
#####
> v4:=map(simplify,tmp44,trig);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

```
  v4 := [ u4 + cos(q9) u9 + u2 z12 + u2 sin(q7) q8 + u2 cos(q7) cos(q9) z34

              - u3 cos(q7) q8 + u3 sin(q7) cos(q9) z34,

      u5 - sin(q7) u7 + cos(q7) u8 + u3 sin(q9) z34 - u1 z12 - u1 q8 sin(q7)

           - u1 cos(q7) cos(q9) z34,

      u6 + cos(q7) u7 + sin(q7) u8 + u1 cos(q7) q8 - u1 sin(q7) cos(q9) z34

           - u2 sin(q9) z34                                                  ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp45:=map(collect,om1,convert(tmp6,list)):
#####
> om1_1:=map(coeff,tmp45,u1);
> om1_2:=map(coeff,tmp45,u2);
> om1_3:=map(coeff,tmp45,u3);
> om1_4:=map(coeff,tmp45,u4);
> om1_5:=map(coeff,tmp45,u5);
> om1_6:=map(coeff,tmp45,u6);
> om1_7:=map(coeff,tmp45,u7);
> om1_8:=map(coeff,tmp45,u8);
> om1_9:=map(coeff,tmp45,u9);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                              om1_1 := [ 1, 0, 0 ]

                              om1_2 := [ 0, 1, 0 ]

                              om1_3 := [ 0, 0, 1 ]

                              om1_4 := [ 0, 0, 0 ]

                              om1_5 := [ 0, 0, 0 ]

                              om1_6 := [ 0, 0, 0 ]

                              om1_7 := [ 0, 0, 0 ]

                              om1_8 := [ 0, 0, 0 ]

                              om1_9 := [ 0, 0, 0 ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp46:=map(collect,om2,convert(tmp6,list)):
#####




> om2_1:=map(coeff,tmp46,u1);
> om2_2:=map(coeff,tmp46,u2);
> om2_3:=map(coeff,tmp46,u3);
> om2_4:=map(coeff,tmp46,u4);
> om2_5:=map(coeff,tmp46,u5);
> om2_6:=map(coeff,tmp46,u6);
> om2_7:=map(coeff,tmp46,u7);
> om2_8:=map(coeff,tmp46,u8);
> om2_9:=map(coeff,tmp46,u9);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                              om2_1 := [ 1, 0, 0 ]

                              om2_2 := [ 0, 1, 0 ]
```

```
                    om2_3 := [ 0, 0, 1 ]

                    om2_4 := [ 0, 0, 0 ]

                    om2_5 := [ 0, 0, 0 ]

                    om2_6 := [ 0, 0, 0 ]

                                 1
                  om2_7 := [ ----, 0, 0 ]
                                q8

                    om2_8 := [ 0, 0, 0 ]

                               sin(q9)
                 om2_9 := [ -------, 0, 0 ]
                                q8

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp47:=map(collect,om4,convert(tmp6,list)):
#####

> om4_1:=map(coeff,tmp47,u1);
> om4_2:=map(coeff,tmp47,u2);
> om4_3:=map(coeff,tmp47,u3);
> om4_4:=map(coeff,tmp47,u4);
> om4_5:=map(coeff,tmp47,u5);
> om4_6:=map(coeff,tmp47,u6);
> om4_7:=map(coeff,tmp47,u7);
> om4_8:=map(coeff,tmp47,u8);
> om4_9:=map(coeff,tmp47,u9);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                    om4_1 := [ 1, 0, 0 ]

                    om4_2 := [ 0, 1, 0 ]

                    om4_3 := [ 0, 0, 1 ]

                    om4_4 := [ 0, 0, 0 ]

                    om4_5 := [ 0, 0, 0 ]

                    om4_6 := [ 0, 0, 0 ]

                                 1
                  om4_7 := [ ----, 0, 0 ]
                                q8

                    om4_8 := [ 0, 0, 0 ]

                         sin(q9)  cos(q7)  sin(q7)
           om4_9 := [ -------, -------, ------- ]
                           q8       z34      z34

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp50:=map(collect,v1,convert(tmp6,list)):
#####
> v1_1:=map(coeff,tmp50,u1);
> v1_2:=map(coeff,tmp50,u2);
> v1_3:=map(coeff,tmp50,u3);
> v1_4:=map(coeff,tmp50,u4);
> v1_5:=map(coeff,tmp50,u5);
> v1_6:=map(coeff,tmp50,u6);
> v1_7:=map(coeff,tmp50,u7);
> v1_8:=map(coeff,tmp50,u8);
```

```
> v1_9:=map(coeff,tmp50,u9);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                              v1_1 := [ 0, 0, 0 ]

                              v1_2 := [ 0, 0, 0 ]

                              v1_3 := [ 0, 0, 0 ]

                              v1_4 := [ 1, 0, 0 ]

                              v1_5 := [ 0, 1, 0 ]

                              v1_6 := [ 0, 0, 1 ]

                              v1_7 := [ 0, 0, 0 ]

                              v1_8 := [ 0, 0, 0 ]

                              v1_9 := [ 0, 0, 0 ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp51:=map(collect,v12,convert(tmp6,list)):
#####
> v12_1:=map(coeff,tmp51,u1);
> v12_2:=map(coeff,tmp51,u2);
> v12_3:=map(coeff,tmp51,u3);
> v12_4:=map(coeff,tmp51,u4);
> v12_5:=map(coeff,tmp51,u5);
> v12_6:=map(coeff,tmp51,u6);
> v12_7:=map(coeff,tmp51,u7);
> v12_8:=map(coeff,tmp51,u8);
> v12_9:=map(coeff,tmp51,u9);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                            v12_1 := [ 0, - z12, 0 ]

                             v12_2 := [ z12, 0, 0 ]

                              v12_3 := [ 0, 0, 0 ]

                              v12_4 := [ 1, 0, 0 ]

                              v12_5 := [ 0, 1, 0 ]

                              v12_6 := [ 0, 0, 1 ]

                              v12_7 := [ 0, 0, 0 ]

                              v12_8 := [ 0, 0, 0 ]

                              v12_9 := [ 0, 0, 0 ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp52:=map(collect,v34,convert(tmp6,list)):
#####
> v34_1:=map(simplify,map(coeff,tmp52,u1));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                v34_1 := [ 0, - z12 - sin(q7) q8, cos(q7) q8 ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> v34_2:=map(coeff,tmp52,u2);
```

```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                      v34_2 := [ z12 + sin(q7) q8, 0, 0 ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> v34_3:=map(coeff,tmp52,u3);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                        v34_3 := [ - cos(q7) q8, 0, 0 ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> v34_4:=map(coeff,tmp52,u4);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                              v34_4 := [ 1, 0, 0 ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>



> v34_5:=map(coeff,tmp52,u5);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                              v34_5 := [ 0, 1, 0 ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> v34_6:=map(coeff,tmp52,u6);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                              v34_6 := [ 0, 0, 1 ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> v34_7:=map(coeff,tmp52,u7);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

v34_7 :=

           sin(q7) q8 - cos(q7) cos(q9) z34  cos(q7) q8 + sin(q7) cos(q9) z34
    [ 0, - --------------------------------, -------------------------------- ]
                          q8                                 q8

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> v34_8:=map(coeff,tmp52,u8);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                        v34_8 := [ 0, cos(q7), sin(q7) ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> v34_9:=map(coeff,tmp52,u9);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                       sin(q7) q8 sin(q9) - cos(q7) cos(q9) z34 sin(q9)
       v34_9 := [ 0, - ------------------------------------------------,
                                            q8
```

```
          cos(q7) q8 sin(q9) + sin(q7) cos(q9) z34 sin(q9)
          ------------------------------------------------ ]
                                 q8

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp53:=map(collect,v4,convert(tmp6,list)):
#####
> v4_1:=map(coeff,tmp53,u1);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

v4_1 := [

  0, - z12 - sin(q7) q8 - cos(q7) cos(q9) z34, cos(q7) q8 - sin(q7) cos(q9) z34

   ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> v4_2:=map(coeff,tmp53,u2);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

      v4_2 := [ z12 + sin(q7) q8 + cos(q7) cos(q9) z34, 0, - sin(q9) z34 ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> v4_3:=map(coeff,tmp53,u3);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

          v4_3 := [ sin(q7) cos(q9) z34 - cos(q7) q8, sin(q9) z34, 0 ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> v4_4:=map(coeff,tmp53,u4);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                              v4_4 := [ 1, 0, 0 ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> v4_5:=map(coeff,tmp53,u5);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                              v4_5 := [ 0, 1, 0 ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> v4_6:=map(coeff,tmp53,u6);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                              v4_6 := [ 0, 0, 1 ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> v4_7:=map(coeff,tmp53,u7);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                       v4_7 := [ 0, - sin(q7), cos(q7) ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> v4_8:=map(coeff,tmp53,u8);
```

```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                        v4_8 := [ 0, cos(q7), sin(q7) ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> v4_9:=map(coeff,tmp53,u9);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                           v4_9 := [ cos(q9), 0, 0 ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

bytes used=5009156, alloc=1310480, time=19.766
```

A.6.2 équations (fichier kane)

```
> read kaneintro:
#####

> c1:=vector([c1X,c1Y,c1Z]):
> f1:=vector([f1X,f1Y,f1Z]):
#####
> c12:=vector([C12,0,0]):
#####
> f:=vector([0,F23,0]):
#####
> c34:=vector([0,C34,0]):
#####
> f4:=vector([f4X,f4Y,f4Z]):
#####

> tmp55:=add(c1,scalarmul(c12,-1)):
> tmp56:=scalarmul(multiply(Rot12,f),-1):
> tmp57:=scalarmul(tmp56,-1):
> tmp58:=scalarmul(multiply(Rot12,c34),-1):
> tmp59:=scalarmul(tmp58,-1):
#####
> K_1:=
>   simplify(subs(varslect,dotprod(v1_1,f1)+dotprod(om1_1,tmp55)
>                          +dotprod(v12_1,tmp56)+dotprod(om2_1,c12)
>                          +dotprod(v34_1,tmp57)+dotprod(om2_1,tmp58)
>                          +dotprod(v4_1,f4)+dotprod(om4_1,tmp59)));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

        K_1 := c1X - f4Y z12 - f4Y sin(q7) q8 - f4Y cos(q7) cos(q9) z34

             + f4Z cos(q7) q8 - f4Z sin(q7) cos(q9) z34

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> K_2:=
>   dotprod(v1_2,f1)+dotprod(om1_2,tmp55)+dotprod(v12_2,tmp56)
>   +dotprod(om2_2,c12)+dotprod(v34_2,tmp57)+dotprod(om2_2,tmp58)
>   +dotprod(v4_2,f4)+dotprod(om4_2,tmp59);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

  K_2 := c1Y + (z12 + sin(q7) q8 + cos(q7) cos(q9) z34) f4X - sin(q9) z34 f4Z

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> K_3:=
>   dotprod(v1_3,f1)+dotprod(om1_3,tmp55)+dotprod(v12_3,tmp56)
>   +dotprod(om2_3,c12)+dotprod(v34_3,tmp57)+dotprod(om2_3,tmp58)
>   +dotprod(v4_3,f4)+dotprod(om4_3,tmp59);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

     K_3 := c1Z + (sin(q7) cos(q9) z34 - cos(q7) q8) f4X + sin(q9) z34 f4Y

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> K_4:=
>   dotprod(v1_4,f1)+dotprod(om1_4,tmp55)+dotprod(v12_4,tmp56)
>   +dotprod(om2_4,c12)+dotprod(v34_4,tmp57)+dotprod(om2_4,tmp58)
>   +dotprod(v4_4,f4)+dotprod(om4_4,tmp59);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                                K_4 := f1X + f4X

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

```
> K_5:=
>   dotprod(v1_5,f1)+dotprod(om1_5,tmp55)+dotprod(v12_5,tmp56)
>   +dotprod(om2_5,c12)+dotprod(v34_5,tmp57)+dotprod(om2_5,tmp58)
>   +dotprod(v4_5,f4)+dotprod(om4_5,tmp59);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                               K_5 := f1Y + f4Y

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> K_6:=
>   dotprod(v1_6,f1)+dotprod(om1_6,tmp55)+dotprod(v12_6,tmp56)
>   +dotprod(om2_6,c12)+dotprod(v34_6,tmp57)+dotprod(om2_6,tmp58)
>   +dotprod(v4_6,f4)+dotprod(om4_6,tmp59);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                               K_6 := f1Z + f4Z

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> K_7:=
>   simplify(subs(varslect,dotprod(v1_7,f1)+dotprod(om1_7,tmp55)
>                          +dotprod(v12_7,tmp56)+dotprod(om2_7,c12)
>                          +dotprod(v34_7,tmp57)+dotprod(om2_7,tmp58)
>                          +dotprod(v4_7,f4)+dotprod(om4_7,tmp59)));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                - C12 - F23 cos(q9) z34 + f4Y sin(q7) q8 - f4Z cos(q7) q8
       K_7 := - ---------------------------------------------------------
                                            q8

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> K_8:=
>   simplify(subs(varslect,dotprod(v1_8,f1)+dotprod(om1_8,tmp55)
>                          +dotprod(v12_8,tmp56)+dotprod(om2_8,c12)
>                          +dotprod(v34_8,tmp57)+dotprod(om2_8,tmp58)
>                          +dotprod(v4_8,f4)+dotprod(om4_8,tmp59)));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                    K_8 := F23 + cos(q7) f4Y + sin(q7) f4Z

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> K_9:=
>   simplify(subs(varslect,dotprod(v1_9,f1)+dotprod(om1_9,tmp55)
>                          +dotprod(v12_9,tmp56)+dotprod(om2_9,c12)
>                          +dotprod(v34_9,tmp57)+dotprod(om2_9,tmp58)
>                          +dotprod(v4_9,f4)+dotprod(om4_9,tmp59)));

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                                         2
       sin(q9) C12 z34 + sin(q9) F23 z34  cos(q9) + cos(q9) f4X q8 z34 + C34 q8
K_9 := ------------------------------------------------------------------------
                                       q8 z34

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> varstu:=[u1=u1(t),u2=u2(t),u3=u3(t),u4=u4(t),u5=u5(t),u6=u6(t),
>          u7=u7(t),u8=u8(t),u9=u9(t)]:
> varstq:=[q1=q1(t),q2=q2(t),q3=q3(t),q4=q4(t),q5=q5(t),q6=q6(t),
>          q7=q7(t),q8=q8(t),q9=q9(t)]:
```

```
> varslectu:=[diff(u1(t),t)=up1,diff(u2(t),t)=up2,diff(u3(t),t)=up3,
>             diff(u4(t),t)=up4,diff(u5(t),t)=up5,diff(u6(t),t)=up6,
>             diff(u7(t),t)=up7,diff(u8(t),t)=up8,diff(u9(t),t)=up9,
>             u1(t)=u1,u2(t)=u2,u3(t)=u3,u4(t)=u4,u5(t)=u5,u6(t)=u6,
>             u7(t)=u7,u8(t)=u8,u9(t)=u9]:
#####
> tmp60:=
>   subs(diff(q4(t),t)=eq1[4],diff(q5(t),t)=eq1[5],diff(q6(t),t)=eq1[6],
>        op(tmp30)):
> tmp61:=subs(varstq,subs(varstu,op(tmp60))):
> tmp62:=map(diff,tmp61,t):
> tmp63:=multiply(transpose(Rot1),tmp62):
> tmp64:=subs(varslect,subs(varslectu,op(tmp63))):
> tmp65:=subs(qp1=eq1[1],qp2=eq1[2],qp3=eq1[3],op(tmp64)):
#####
> gamma1:=map(simplify,tmp65,trig);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

  gamma1 := [ up4 + u6 u2 - u5 u3, up5 + u4 u3 - u6 u1, up6 + u5 u1 - u4 u2 ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> Og4:=add(Og1,multiply(Rot1,g1g4)):
> tmp70:=map(diff,Og4,t):
> tmp71:=subs(varslect,op(tmp70)):
> tmp72:=
>   subs({qp1=eq1[1],qp2=eq1[2],qp3=eq1[3],qp4=eq1[4],qp5=eq1[5],
>         qp6=eq1[6],qp7=eq1[7],qp8=eq1[8],qp9=eq1[9]},op(tmp71)):
> tmp73:=subs(varstu,subs(varstq,op(tmp72))):
> tmp74:=map(simplify,tmp73,trig):
> tmp75:=map(diff,tmp74,t):
> tmp76:=multiply(transpose(Rot1),tmp75):
> tmp77:=subs(varslect,subs(varslectu,op(tmp76))):
> tmp78:=
>   subs({qp1=eq1[1],qp2=eq1[2],qp3=eq1[3],qp4=eq1[4],qp5=eq1[5],
>         qp6=eq1[6],qp7=eq1[7],qp8=eq1[8],qp9=eq1[9]},op(tmp77)):
> tmp79:=map(simplify,tmp78,trig):
#####
> gamma4:=map(collect,tmp79,[up1,up2,up3,up4,up5,up6,up7,up8,up9]);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

gamma4 := [

                      2
(cos(q7) cos(q9) z34  + sin(q7) q8 z34 + z12 z34) up2
-----------------------------------------------------
                        z34

                         2
       (sin(q7) cos(q9) z34  - cos(q7) q8 z34) up3
     + ------------------------------------------- + up4 + cos(q9) up9 + (
                          z34

                         2                    2   2              2   2
    - sin(q7) cos(q9) z34  u1 u2 - sin(q9) z34  u2  - sin(q9) z34  u3

                                                 2
     - 2 cos(q7) u8 z34 u3 + cos(q7) cos(q9) z34  u1 u3 + 2 sin(q7) u2 z34 u8

     + sin(q7) q8 u1 z34 u3 - u5 z34 u3 + 2 sin(q7) u7 z34 u3

                                             2
     + cos(q7) q8 u1 z34 u2 - sin(q9) u9  + z12 u1 z34 u3 + 2 cos(q7) u7 z34 u2

     + u6 z34 u2)/z34,
```

```
     2
(- q8  sin(q7) - cos(q7) cos(q9) z34 q8 - q8 z12) up1
----------------------------------------------------- + sin(q9) z34 up3 + up5
                         q8

     - sin(q7) up7 + cos(q7) up8 + (- 2 sin(q7) u1 u8 q8 + sin(q9) z34 u2 q8 u1

                             2                                      2
     + sin(q7) cos(q9) z34 u1  q8 - 2 cos(q7) u1 q8 u7 - cos(q7) u7

                                 2                   2   2
     - sin(q7) u8 u7 + sin(q7) q8  u2 u3 - cos(q7) q8  u3

                             2
     + sin(q7) cos(q9) z34 u3  q8 + cos(q7) cos(q9) z34 u2 q8 u3

                                         2   2
     - sin(q7) u8 sin(q9) u9 - cos(q7) q8  u1  + u4 q8 u3 + 2 cos(q9) u9 u3 q8

     + z12 u2 q8 u3 - cos(q7) u7 sin(q9) u9 - u6 q8 u1)/q8,

           2
(cos(q7) q8  - sin(q7) cos(q9) z34 q8) up1
------------------------------------------ - sin(q9) z34 up2 + up6
                    q8

     + cos(q7) up7 + sin(q7) up8 + (sin(q9) z34 u3 q8 u1

                             2            2                 2   2
     - cos(q7) cos(q9) z34 u2  q8 - z12 u2  q8 - sin(q7) q8  u1

                             2
     - cos(q7) cos(q9) z34 u1  q8 + u5 q8 u1 - sin(q7) cos(q9) z34 u3 q8 u2

                 2             2   2
     - sin(q7) u7  - sin(q7) q8  u2  - sin(q7) u7 sin(q9) u9 - u4 q8 u2

                                  2
     + 2 cos(q7) u1 u8 q8 - z12 u1  q8 + cos(q7) u8 sin(q9) u9 + cos(q7) u8 u7

                                                           2
     - 2 sin(q7) u1 q8 u7 - 2 cos(q9) u9 u2 q8 + cos(q7) q8  u3 u2)/q8

 ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> tmp80:=
>   subs(diff(q1(t),t)=eq1[1],diff(q2(t),t)=eq1[2],diff(q3(t),t)=eq1[3],
>       op(tmp12)):
> tmp81:=subs(varstq,subs(varstu,subs(varslect,op(tmp80)))):
> tmp82:=map(diff,tmp81,t):
> tmp83:=multiply(transpose(Rot1),tmp82):
> tmp84:=subs(varslect,subs(varslectu,op(tmp83))):
> tmp85:=subs(qp1=eq1[1],qp2=eq1[2],qp3=eq1[3],op(tmp84)):
#####
> alpha1:=map(simplify,tmp85,trig);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                          alpha1 := [ up1, up2, up3 ]

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> i1:=matrix([[i1X,0,0],[0,i1Y,0],[0,0,i1Z]]):
#####
> tmp90:=scalarmul(gamma1,-mu1):
> tmp91:=scalarmul(add(multiply(i1,alpha1),crossprod(om1,multiply(i1,om1))),-1):
> tmp92:=scalarmul(gamma4,-mu4):
#####
```

```
> Ki_1:=
>   dotprod(v1_1,tmp90)+dotprod(om1_1,tmp91)+dotprod(v4_1,tmp92);
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

 Ki_1 := - i1X up1 - u2 i1Z u3 + u3 i1Y u2 -

     (- z12 - sin(q7) q8 - cos(q7) cos(q9) z34) mu4 (

           2
     (- q8  sin(q7) - cos(q7) cos(q9) z34 q8 - q8 z12) up1
     ----------------------------------------------------- + sin(q9) z34 up3
                               q8

      + up5 - sin(q7) up7 + cos(q7) up8 + (- 2 sin(q7) u1 u8 q8

                                                       2
      + sin(q9) z34 u2 q8 u1 + sin(q7) cos(q9) z34 u1  q8 - 2 cos(q7) u1 q8 u7

                   2                              2                    2   2
      - cos(q7) u7  - sin(q7) u8 u7 + sin(q7) q8  u2 u3 - cos(q7) q8  u3

                                  2
      + sin(q7) cos(q9) z34 u3  q8 + cos(q7) cos(q9) z34 u2 q8 u3

                                              2   2
      - sin(q7) u8 sin(q9) u9 - cos(q7) q8  u1  + u4 q8 u3 + 2 cos(q9) u9 u3 q8

      + z12 u2 q8 u3 - cos(q7) u7 sin(q9) u9 - u6 q8 u1)/q8) -

     (cos(q7) q8 - sin(q7) cos(q9) z34) mu4 (

                  2
     (cos(q7) q8  - sin(q7) cos(q9) z34 q8) up1
     ------------------------------------------ - sin(q9) z34 up2 + up6
                         q8

      + cos(q7) up7 + sin(q7) up8 + (sin(q9) z34 u3 q8 u1

                                  2             2                  2   2
      - cos(q7) cos(q9) z34 u2  q8 - z12 u2  q8 - sin(q7) q8  u1

                                  2
      - cos(q7) cos(q9) z34 u1  q8 + u5 q8 u1 - sin(q7) cos(q9) z34 u3 q8 u2

                   2              2   2
      - sin(q7) u7  - sin(q7) q8  u2  - sin(q7) u7 sin(q9) u9 - u4 q8 u2

                                     2
      + 2 cos(q7) u1 u8 q8 - z12 u1  q8 + cos(q7) u8 sin(q9) u9 + cos(q7) u8 u7

                                                                 2
      - 2 sin(q7) u1 q8 u7 - 2 cos(q9) u9 u2 q8 + cos(q7) q8  u3 u2)/q8)
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> Ki_2:=
>   dotprod(v1_2,tmp90)+dotprod(om1_2,tmp91)+dotprod(v4_2,tmp92);
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

Ki_2 := - i1Y up2 - u3 i1X u1 + u1 i1Z u3 -

    (z12 + sin(q7) q8 + cos(q7) cos(q9) z34) mu4 (

                               2
    (cos(q7) cos(q9) z34  + sin(q7) q8 z34 + z12 z34) up2
    -----------------------------------------------------
                              z34
```

```
                          2
      (sin(q7) cos(q9) z34  - cos(q7) q8 z34) up3
    + ------------------------------------------ + up4 + cos(q9) up9 + (
                         z34

                        2                     2   2              2   2
   - sin(q7) cos(q9) z34  u1 u2 - sin(q9) z34  u2  - sin(q9) z34  u3

                                                 2
    - 2 cos(q7) u8 z34 u3 + cos(q7) cos(q9) z34  u1 u3 + 2 sin(q7) u2 z34 u8

    + sin(q7) q8 u1 z34 u3 - u5 z34 u3 + 2 sin(q7) u7 z34 u3

                                       2
    + cos(q7) q8 u1 z34 u2 - sin(q9) u9  + z12 u1 z34 u3 + 2 cos(q7) u7 z34 u2

    + u6 z34 u2)/z34) + sin(q9) z34 mu4 (

              2
   (cos(q7) q8  - sin(q7) cos(q9) z34 q8) up1
   ----------------------------------------- - sin(q9) z34 up2 + up6
                       q8

    + cos(q7) up7 + sin(q7) up8 + (sin(q9) z34 u3 q8 u1

                             2            2                2   2
    - cos(q7) cos(q9) z34 u2  q8 - z12 u2  q8 - sin(q7) q8  u1

                            2
    - cos(q7) cos(q9) z34 u1  q8 + u5 q8 u1 - sin(q7) cos(q9) z34 u3 q8 u2

                2             2   2
    - sin(q7) u7  - sin(q7) q8  u2  - sin(q7) u7 sin(q9) u9 - u4 q8 u2

                                  2
    + 2 cos(q7) u1 u8 q8 - z12 u1  q8 + cos(q7) u8 sin(q9) u9 + cos(q7) u8 u7

                                                          2
    - 2 sin(q7) u1 q8 u7 - 2 cos(q9) u9 u2 q8 + cos(q7) q8  u3 u2)/q8)

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> Ki_3:=
>   dotprod(v1_3,tmp90)+dotprod(om1_3,tmp91)+dotprod(v4_3,tmp92);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

Ki_3 := - i1Z up3 - u1 i1Y u2 + u2 i1X u1 - (sin(q7) cos(q9) z34 - cos(q7) q8)

                            2
         (cos(q7) cos(q9) z34  + sin(q7) q8 z34 + z12 z34) up2
    mu4 (-------------------------------------------------
                                z34

                          2
      (sin(q7) cos(q9) z34  - cos(q7) q8 z34) up3
    + ------------------------------------------ + up4 + cos(q9) up9 + (
                         z34

                        2                     2   2              2   2
   - sin(q7) cos(q9) z34  u1 u2 - sin(q9) z34  u2  - sin(q9) z34  u3

                                                 2
    - 2 cos(q7) u8 z34 u3 + cos(q7) cos(q9) z34  u1 u3 + 2 sin(q7) u2 z34 u8

    + sin(q7) q8 u1 z34 u3 - u5 z34 u3 + 2 sin(q7) u7 z34 u3

                                       2
    + cos(q7) q8 u1 z34 u2 - sin(q9) u9  + z12 u1 z34 u3 + 2 cos(q7) u7 z34 u2
```

```
     + u6 z34 u2)/z34) - sin(q9) z34 mu4 (

         2
    (- q8  sin(q7) - cos(q7) cos(q9) z34 q8 - q8 z12) up1
    ---------------------------------------------------- + sin(q9) z34 up3
                            q8

     + up5 - sin(q7) up7 + cos(q7) up8 + (- 2 sin(q7) u1 u8 q8

                                                      2
     + sin(q9) z34 u2 q8 u1 + sin(q7) cos(q9) z34 u1  q8 - 2 cos(q7) u1 q8 u7

                2                                 2                    2   2
     - cos(q7) u7  - sin(q7) u8 u7 + sin(q7) q8  u2 u3 - cos(q7) q8  u3

                             2
     + sin(q7) cos(q9) z34 u3  q8 + cos(q7) cos(q9) z34 u2 q8 u3

                                         2   2
     - sin(q7) u8 sin(q9) u9 - cos(q7) q8  u1  + u4 q8 u3 + 2 cos(q9) u9 u3 q8

     + z12 u2 q8 u3 - cos(q7) u7 sin(q9) u9 - u6 q8 u1)/q8)

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> Ki_4:=
>   dotprod(v1_4,tmp90)+dotprod(om1_4,tmp91)+dotprod(v4_4,tmp92);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

Ki_4 := - mu1 (up4 + u6 u2 - u5 u3) - mu4 (

                         2
    (cos(q7) cos(q9) z34  + sin(q7) q8 z34 + z12 z34) up2
    -----------------------------------------------------
                            z34

                          2
       (sin(q7) cos(q9) z34  - cos(q7) q8 z34) up3
     + ---------------------------------------- + up4 + cos(q9) up9 + (
                         z34

                        2                     2   2              2   2
    - sin(q7) cos(q9) z34  u1 u2 - sin(q9) z34  u2  - sin(q9) z34  u3

                                                 2
     - 2 cos(q7) u8 z34 u3 + cos(q7) cos(q9) z34  u1 u3 + 2 sin(q7) u2 z34 u8

     + sin(q7) q8 u1 z34 u3 - u5 z34 u3 + 2 sin(q7) u7 z34 u3

                                        2
     + cos(q7) q8 u1 z34 u2 - sin(q9) u9  + z12 u1 z34 u3 + 2 cos(q7) u7 z34 u2

     + u6 z34 u2)/z34)

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> Ki_5:=
>   dotprod(v1_5,tmp90)+dotprod(om1_5,tmp91)+dotprod(v4_5,tmp92);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

 Ki_5 := - mu1 (up5 + u4 u3 - u6 u1) - mu4 (

         2
     (- q8  sin(q7) - cos(q7) cos(q9) z34 q8 - q8 z12) up1
     ---------------------------------------------------- + sin(q9) z34 up3
                             q8
```

```
      + up5 - sin(q7) up7 + cos(q7) up8 + (- 2 sin(q7) u1 u8 q8

                                                      2
      + sin(q9) z34 u2 q8 u1 + sin(q7) cos(q9) z34 u1  q8 - 2 cos(q7) u1 q8 u7

                   2                             2                  2   2
      - cos(q7) u7  - sin(q7) u8 u7 + sin(q7) q8  u2 u3 - cos(q7) q8  u3

                             2
      + sin(q7) cos(q9) z34 u3  q8 + cos(q7) cos(q9) z34 u2 q8 u3

                                         2   2
      - sin(q7) u8 sin(q9) u9 - cos(q7) q8  u1  + u4 q8 u3 + 2 cos(q9) u9 u3 q8

      + z12 u2 q8 u3 - cos(q7) u7 sin(q9) u9 - u6 q8 u1)/q8)
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
> Ki_6:=
>   dotprod(v1_6,tmp90)+dotprod(om1_6,tmp91)+dotprod(v4_6,tmp92);
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
 Ki_6 := - mu1 (up6 + u5 u1 - u4 u2) - mu4 (

                 2
     (cos(q7) q8  - sin(q7) cos(q9) z34 q8) up1
     ----------------------------------------- - sin(q9) z34 up2 + up6
                         q8

      + cos(q7) up7 + sin(q7) up8 + (sin(q9) z34 u3 q8 u1

                             2             2                2   2
      - cos(q7) cos(q9) z34 u2  q8 - z12 u2  q8 - sin(q7) q8  u1

                             2
      - cos(q7) cos(q9) z34 u1  q8 + u5 q8 u1 - sin(q7) cos(q9) z34 u3 q8 u2

                   2             2   2
      - sin(q7) u7  - sin(q7) q8  u2  - sin(q7) u7 sin(q9) u9 - u4 q8 u2

                                  2
      + 2 cos(q7) u1 u8 q8 - z12 u1  q8 + cos(q7) u8 sin(q9) u9 + cos(q7) u8 u7

                                                          2
      - 2 sin(q7) u1 q8 u7 - 2 cos(q9) u9 u2 q8 + cos(q7) q8  u3 u2)/q8)
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
> Ki_7:=
>   dotprod(v1_7,tmp90)+dotprod(om1_7,tmp91)+dotprod(v4_7,tmp92);
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
                           2
                       (- q8  sin(q7) - cos(q7) cos(q9) z34 q8 - q8 z12) up1
 Ki_7 := sin(q7) mu4 (-----------------------------------------------------
                                               q8

      + sin(q9) z34 up3 + up5 - sin(q7) up7 + cos(q7) up8 + (

                                                                       2
     - 2 sin(q7) u1 u8 q8 + sin(q9) z34 u2 q8 u1 + sin(q7) cos(q9) z34 u1  q8

                                        2                         2
      - 2 cos(q7) u1 q8 u7 - cos(q7) u7  - sin(q7) u8 u7 + sin(q7) q8  u2 u3

                   2   2                         2
      - cos(q7) q8  u3  + sin(q7) cos(q9) z34 u3  q8
```

```
                                                                           2   2
      + cos(q7) cos(q9) z34 u2 q8 u3 - sin(q7) u8 sin(q9) u9 - cos(q7) q8  u1

      + u4 q8 u3 + 2 cos(q9) u9 u3 q8 + z12 u2 q8 u3 - cos(q7) u7 sin(q9) u9

                                                 2
                                     (cos(q7) q8  - sin(q7) cos(q9) z34 q8) up1
      - u6 q8 u1)/q8) - cos(q7) mu4 (------------------------------------------
                                                        q8

      - sin(q9) z34 up2 + up6 + cos(q7) up7 + sin(q7) up8 + (

                                                   2             2
     sin(q9) z34 u3 q8 u1 - cos(q7) cos(q9) z34 u2  q8 - z12 u2  q8

                   2   2                          2
      - sin(q7) q8  u1  - cos(q7) cos(q9) z34 u1  q8 + u5 q8 u1

                                                    2             2   2
      - sin(q7) cos(q9) z34 u3 q8 u2 - sin(q7) u7  - sin(q7) q8  u2

                                                                     2
      - sin(q7) u7 sin(q9) u9 - u4 q8 u2 + 2 cos(q7) u1 u8 q8 - z12 u1  q8

      + cos(q7) u8 sin(q9) u9 + cos(q7) u8 u7 - 2 sin(q7) u1 q8 u7

                                          2
      - 2 cos(q9) u9 u2 q8 + cos(q7) q8  u3 u2)/q8)
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> Ki_8:=
>   dotprod(v1_8,tmp90)+dotprod(om1_8,tmp91)+dotprod(v4_8,tmp92);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                              2
                       (- q8  sin(q7) - cos(q7) cos(q9) z34 q8 - q8 z12) up1
 Ki_8 := - cos(q7) mu4 (-----------------------------------------------------
                                                q8

      + sin(q9) z34 up3 + up5 - sin(q7) up7 + cos(q7) up8 + (

                                                                          2
     - 2 sin(q7) u1 u8 q8 + sin(q9) z34 u2 q8 u1 + sin(q7) cos(q9) z34 u1  q8

                                          2                            2
      - 2 cos(q7) u1 q8 u7 - cos(q7) u7  - sin(q7) u8 u7 + sin(q7) q8  u2 u3

                   2   2                          2
      - cos(q7) q8  u3  + sin(q7) cos(q9) z34 u3  q8

                                                                           2   2
      + cos(q7) cos(q9) z34 u2 q8 u3 - sin(q7) u8 sin(q9) u9 - cos(q7) q8  u1

      + u4 q8 u3 + 2 cos(q9) u9 u3 q8 + z12 u2 q8 u3 - cos(q7) u7 sin(q9) u9

                                                 2
                                     (cos(q7) q8  - sin(q7) cos(q9) z34 q8) up1
      - u6 q8 u1)/q8) - sin(q7) mu4 (------------------------------------------
                                                        q8

      - sin(q9) z34 up2 + up6 + cos(q7) up7 + sin(q7) up8 + (

                                                   2             2
     sin(q9) z34 u3 q8 u1 - cos(q7) cos(q9) z34 u2  q8 - z12 u2  q8

                   2   2                          2
      - sin(q7) q8  u1  - cos(q7) cos(q9) z34 u1  q8 + u5 q8 u1
```

```
                                                  2            2   2
      - sin(q7) cos(q9) z34 u3 q8 u2 - sin(q7) u7  - sin(q7) q8  u2

                                                                      2
      - sin(q7) u7 sin(q9) u9 - u4 q8 u2 + 2 cos(q7) u1 u8 q8 - z12 u1  q8

      + cos(q7) u8 sin(q9) u9 + cos(q7) u8 u7 - 2 sin(q7) u1 q8 u7

                                       2
      - 2 cos(q9) u9 u2 q8 + cos(q7) q8  u3 u2)/q8)

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

> Ki_9:=
>   dotprod(v1_9,tmp90)+dotprod(om1_9,tmp91)+dotprod(v4_9,tmp92);

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

                                          2
                        (cos(q7) cos(q9) z34  + sin(q7) q8 z34 + z12 z34) up2
Ki_9 := - cos(q9) mu4 (-----------------------------------------------------
                                             z34

                           2
       (sin(q7) cos(q9) z34  - cos(q7) q8 z34) up3
     + ------------------------------------------ + up4 + cos(q9) up9 + (
                          z34

                         2                     2   2              2   2
    - sin(q7) cos(q9) z34  u1 u2 - sin(q9) z34  u2  - sin(q9) z34  u3

                                              2
     - 2 cos(q7) u8 z34 u3 + cos(q7) cos(q9) z34  u1 u3 + 2 sin(q7) u2 z34 u8

     + sin(q7) q8 u1 z34 u3 - u5 z34 u3 + 2 sin(q7) u7 z34 u3

                                         2
     + cos(q7) q8 u1 z34 u2 - sin(q9) u9  + z12 u1 z34 u3 + 2 cos(q7) u7 z34 u2

     + u6 z34 u2)/z34)

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
bytes used=26593920, alloc=3210676, time=144.600
```

Annexe B

outils pour la modélisation

B.1 *LSD* – code d'implémentation *Standard ML*

B.1.1 la description lexicale de *LSD* (fichier `LSD.lex`)

```
structure Tokens = Tokens

type pos = int
type svalue = Tokens.svalue
type ('a,'b) token = ('a,'b) Tokens.token
type lexresult= (svalue,pos) token

val pos = ref 0
val eof = fn () => Tokens.EOF(!pos,!pos)
val error = fn (e,l : int,_) =>
              (output(std_out,"line " ^ (makestring l) ^
                              ": " ^ e ^ "\n"))
%%
%header (functor LSDLexFun(structure Tokens: LSD_TOKENS));
%s C;
alpha=[_A-Za-z];
digit=[0-9];
id={alpha}({digit}|{alpha})*;
optsign=("+"|"-")?;
integer={digit}+;
frac="."{digit}+;
exp=(e|E){optsign}{digit}+;
float={integer}({frac}{exp}?|{frac}?{exp});
ws = [\ \t];

%%
<INITIAL>\n       => (pos := (!pos) + 1; lex());
<INITIAL>{ws}+    => (lex());
<INITIAL>"and"    => (Tokens.AND(!pos,!pos));
<INITIAL>"body"   => (Tokens.BODY(!pos,!pos));
<INITIAL>"^"      => (Tokens.CARAT(!pos,!pos));
<INITIAL>":"      => (Tokens.COLON(!pos,!pos));
<INITIAL>","      => (Tokens.COMMA(!pos,!pos));
```

```
<INITIAL>"["      => (Tokens.LBRACK(!pos,!pos));
<INITIAL>"]''     => (Tokens.RBRACK(!pos,!pos));
<INITIAL>"("      => (Tokens.LPAREN(!pos,!pos));
<INITIAL>")"      => (Tokens.RPAREN(!pos,!pos));
<INITIAL>"."      => (Tokens.DOT(!pos,!pos));
<INITIAL>"="      => (Tokens.EQUAL(!pos,!pos));
<INITIAL>"apply"  => (Tokens.APPLY(!pos,!pos));
<INITIAL>"end"    => (Tokens.END(!pos,!pos));
<INITIAL>"force"  => (Tokens.FORCE(!pos,!pos));
<INITIAL>"in"     => (Tokens.IN(!pos,!pos));
<INITIAL>"join"   => (Tokens.JOIN(!pos,!pos));
<INITIAL>"let"    => (Tokens.LET(!pos,!pos));
<INITIAL>"link"   => (Tokens.LINK(!pos,!pos));
<INITIAL>"local"  => (Tokens.LOCAL(!pos,!pos));
<INITIAL>"on"     => (Tokens.ON(!pos,!pos));
<INITIAL>"package"=> (Tokens.PACKAGE(!pos,!pos));
<INITIAL>"torque" => (Tokens.TORQUE(!pos,!pos));
<INITIAL>"with"   => (Tokens.WITH(!pos,!pos));
<INITIAL>"+"      => (Tokens.PLUS(!pos,!pos));
<INITIAL>"++"     => (Tokens.PLUSPLUS(!pos,!pos));
<INITIAL>"*"      => (Tokens.TIMES(!pos,!pos));
<INITIAL>"-"      => (Tokens.SUB(!pos,!pos));
<INITIAL>"/"      => (Tokens.DIV(!pos,!pos));
<INITIAL>"@"      => (Tokens.AT(!pos,!pos));
<INITIAL>"||"     => (Tokens.PARA(!pos,!pos));
<INITIAL>";"      => (Tokens.SEMI(!pos,!pos));
<INITIAL>{optsign}{integer} => (Tokens.INT(yytext, !pos, !pos));
<INITIAL>{optsign}{float}   => (Tokens.FLOAT(yytext, !pos, !pos));
<INITIAL>{id}     => (Tokens.ID(yytext,!pos,!pos));
<INITIAL>"?"[_A-Za-z]+      =>
           (Tokens.UNKID(substring(yytext, 1, String.length yytext -1),
                         !pos,!pos));
<INITIAL>"(*"   => (YYBEGIN C; lex());
<C>"(*"         => (lex());
<C>"*)"         => (YYBEGIN INITIAL; lex());
<C>\n+          => (pos := (!pos) + (String.length yytext); lex());
<C>.            => (lex());
```

B.1.2 la description syntaxique de *LSD* (fichier `LSD.grm`)

```
open Utils
val mkVar = Absyn.mkVar
val eqn2bindings = map (fn (x, e) => (mkVar x, e))
%%

%verbose

%term
  EOF | AND | APPLY | AT | BODY | CARAT | COLON | COMMA
  | DIV | DOT | END | EQUAL | FORCE | IN
  | JOIN | LBRACK | LET | LINK | LOCAL | LPAREN | ON | PACKAGE | PARA
  | PLUS | PLUSPLUS
  | RBRACK
  | RPAREN | SEMI | SUB  | TIMES | TORQUE | WITH
  | FLOAT of string
  | ID of string
  | INT of string
  | UNKID of string

%nonterm
  program of Absyn.decl list | decl_ss of Absyn.decl list |
  decl of Absyn.decl | eqn of string * Absyn.expr |
  id of Absyn.expr | tid_cs of (Absyn.var * string) list |
  expr of Absyn.expr |
  pid of Absyn.expr | qid of Absyn.expr |
  constr of Absyn.expr |
  mathexpr of Absyn.expr |
  eqn_s of (string * Absyn.expr) list |
  eqn_cs of (string * Absyn.expr) list |
  eqn_cp of (string * Absyn.expr) list |
  expr_cs of Absyn.expr list |
  expr_cp of Absyn.expr list

%pos int
%start program
%eop EOF
%noshift EOF

%name LSD

%keyword AND BODY END IN LET LINK LOCAL PACKAGE APPLY

%nonassoc PARA
%nonassoc AT
%nonassoc PLUSPLUS

%left PLUS SUB
%left TIMES DIV
```

```
%left CARAT

%prefer SEMI

%%

program : decl_ss                 (decl_ss)

decl_ss :                         ([])
        | decl SEMI decl_ss       (decl :: decl_ss)

decl : LET eqn_s                  (Absyn.LET (eqn2bindings eqn_s))
     | LET ID LPAREN tid_cs RPAREN EQUAL expr
       (Absyn.PARAM (mkVar ID, tid_cs, expr))
     | JOIN expr AND expr WITH expr
       (Absyn.JOIN({body1 = expr1, body2 = expr2, link = expr2}))
     | PACKAGE ID EQUAL decl_ss END
       (Absyn.PACKAGE(mkVar ID, decl_ss))
     | APPLY expr ON expr
       (Absyn.APPLY({body = expr2, force = expr1}))

eqn : ID EQUAL expr               ((ID, expr))
    | FORCE EQUAL expr            (("force", expr))
    | TORQUE EQUAL expr           (("torque", expr))

id : ID                           (Absyn.VAR (mkVar ID))
   | qid                          (qid)
   | pid                          (pid)

tid_cs :                          ([])
       | ID COLON ID              ([(mkVar ID1, ID2)])
       | ID COLON ID COMMA tid_cs ((mkVar ID1, ID2) :: tid_cs)

expr : id                         (id)
     | UNKID                      (Absyn.UVAR UNKID)
     | constr                     (constr)
     | mathexpr                   (mathexpr)
     | LBRACK expr COMMA expr COMMA expr RBRACK
       (Absyn.VECTOR (expr1, expr2, expr3))
     | LOCAL eqn_s IN expr END    (Absyn.LOCAL(eqn2bindings eqn_s, expr))
     | ID LPAREN RPAREN           (Absyn.CALL(mkVar ID, []))
     | ID LPAREN expr_cp RPAREN   (Absyn.CALL((mkVar ID), expr_cp))
     | UNKID LPAREN expr_cp RPAREN (Absyn.UCALL(UNKID, expr_cp))
     | ID COLON ID LPAREN eqn_cp RPAREN
       (Absyn.INST(Absyn.PFID(mkVar ID1, mkVar ID2), sortField eqn_cp))
     | ID LPAREN eqn_cp RPAREN
       (Absyn.INST(Absyn.FID (mkVar ID), sortField eqn_cp))
     | LPAREN expr RPAREN         (expr)

qid : ID DOT ID                (Absyn.SELECT(mkVar ID1, ID2))
```

```
pid : ID COLON ID              (Absyn.FROM (mkVar ID1, Absyn.PID (mkVar ID2)))
    | ID COLON ID DOT ID
      (Absyn.FROM (mkVar ID1, Absyn.SPID (mkVar ID2, ID3)))

constr : BODY eqn_cs END       (Absyn.BODY (sortField eqn_cs))
       | LINK eqn_cs END       (Absyn.LINK (sortField eqn_cs))
       | FORCE eqn_cs END      (Absyn.FORCE (sortField eqn_cs))
       | TORQUE eqn_cs END     (Absyn.TORQUE (sortField eqn_cs))

mathexpr : expr PLUS expr      (Absyn.BINOP(Absyn.PLUS, expr1, expr2))
         | expr TIMES expr     (Absyn.BINOP(Absyn.MULT, expr1, expr2))
         | expr DIV expr       (Absyn.BINOP(Absyn.DIV, expr1, expr2))
         | expr SUB expr       (Absyn.BINOP(Absyn.SUB, expr1, expr2))
         | expr CARAT expr     (Absyn.BINOP(Absyn.POWER, expr1, expr2))
         | expr AT expr
           (Absyn.FRAME {body = expr2,displacement = expr1})
         | expr PARA expr      (Absyn.ROTATION {angle = expr1, axis = expr2})
         | expr PLUSPLUS expr
           (Absyn.DISPLACEMENT {rotation = expr2, translation = expr2})
         | INT                 (Absyn.INT INT)
         | FLOAT               (Absyn.FLOAT FLOAT)

eqn_s :                        ([])
     | eqn                     ([eqn])
     | eqn AND eqn_s           (eqn :: eqn_s)

(* suite d'e'galite's (x = expr) se'pare'es par des "," *)
eqn_cs :                       ([])
       | eqn                   ([eqn])
       | eqn COMMA eqn_cs      (eqn :: eqn_cs)

eqn_cp : eqn                   ([eqn])
       | eqn COMMA eqn_cp      (eqn :: eqn_cp)

(* suite d'expressions se'pare'es par des "," *)
expr_cs :                      ([])
        | expr                 ([expr])
        | expr COMMA expr_cs   (expr :: expr_cs)

expr_cp : expr                 ([expr])
        | expr COMMA expr_cp   (expr :: expr_cp)
```

B.1.3 la syntaxe abstraite de *LSD* (fichier `absyn.sml`)

```
(* La syntaxe abstraire de LSD *)

structure Absyn =
  struct

    (* les variables ont un nom et un "identifiant" entier *)
    datatype var = V of string * int ref

    (* les de'clarations *)
    datatype decl =
      LET of (var * expr) list
    | PARAM of var * (var * string) list * expr
    | PACKAGE of var * decl list
    | JOIN of {body1 : expr, body2 : expr, link : expr}
    | APPLY of {force : expr, body : expr}

    (* les expressions *)
    and expr =
      VAR of var
    | UVAR of string
    | INT of string
    | FLOAT of string
    | SELECT of var * string
    (* re'fe'rence a' ce qui se trouve dans un "package" *)
    | FROM of var * pcomp
    | VECTOR of expr * expr * expr
    | CALL of var * expr list
    | UCALL of string * expr list
    | INST of fid * (string * expr) list
    | BINOP of binop * expr * expr
    | BODY of (string * expr) list
    | LINK of (string * expr) list
    | FORCE of (string * expr) list
    | TORQUE of (string * expr) list
    | ROTATION of {angle : expr, axis : expr}
    | DISPLACEMENT of {translation : expr, rotation : expr}
    | FRAME of {body : expr, displacement : expr}
    | LOCAL of (var * expr ) list * expr

    and binop =
      PLUS | SUB | MULT | DIV | POWER

    (* ce qui peut e^tre dans un package P:X ou P:X.mass *)
    and pcomp =
      PID of var
    | SPID of var * string
    (* ce qui peut e^tre en position de nom de fonction *)
    and fid =
```

```
    FID of var
  | PFID of var * var

  fun mkVar s = V(s, ref (Clock.clock()))

  fun getName (V(name, _)) = name

  datatype ty =
    INTty
  | FLOATty
  | SCALARty
  | BODYty
  | FRAMEty
  | LINKty
  | FORCEty
  | TORQUEty
  | DISPLty
  | ROTty
  | VECTORty
  | MATRIXty
  | PARAMty of (string * ty) * ty
  | PACKAGEty of (string * ty) list
  | UNKNOWNty

  type varInfo = {ty : ty}
  exception Unknown

  (* la table dans laquelle on range les infos sur les variables *)
  val VarTable : varInfo Intmap.intmap = Intmap.new (1024,  Unknown)

  fun getType (V(name, ref id)) =
    #ty (Intmap.map VarTable id)
    handle Unknown => (print ("unbound variable "^name);
                       raise Unknown)

  fun setType (V(_, ref id), ty) =
    (Intmap.add VarTable (id, {ty = ty}); ty)

end
```

B.1.4 le mécanisme de typage de *LSD* (fichier `typecheck.sml`)

```
structure Components =
  struct

    val body = Utils.sortField [("mass", Absyn.SCALARty)]

    val force = Utils.sortField []

    val link = Utils.sortField []

    val torque = Utils.sortField []

  end



structure TypeChecker =
  struct

    open Absyn

    exception unbound

    fun binder [] k = raise unbound
      | binder ((V(k, ref i))::t) k' = if k = k' then i
                                       else binder t k'
    fun bind e v = v::e
    fun binded (v,e) =
      let val _ = binder e (getName v) in true end
    handle unbound => false
    fun compose e1 e2 = e1 @ e2
    fun merge (lenv) = fold (op @) lenv []

    fun bindvar env (v as V(name, id)) =
      if binded (v, env) then id := binder env name else ()

    fun bindDecl env decl =
      case decl of
        LET l =>
         (app (fn (_, e) => bindExpr env e) l;
           rev (map (fn (v, _) => v) l))
      | PARAM (f, l, e) =>
         (bindExpr ((map (fn (x, t) => x) l) @ env) e;
           [f])
      | PACKAGE (v, dl) =>
         (bindDecls [] dl;  [v])
      | JOIN {body1, body2, link} =>
           (bindExpr env body1;
            bindExpr env body2;
```

```
        bindExpr env link;
        [])
  | APPLY {body, force} =>
      (bindExpr env body; bindExpr env force; [])

and bindExpr env expr =
  case expr of
     VAR v => bindvar env v
  | UVAR v => ()
  | INT i => ()
  | FLOAT f => ()
  | SELECT (v, c) => bindvar env v
  | FROM(p, PID v) => (bindvar env p; bindvar env v)
  | FROM(p, SPID(v, c)) => (bindvar env p; bindvar env v)
  | VECTOR (e1, e2, e3) => app (bindExpr env) [e1, e2, e3]
  | CALL (v, l) => (bindvar env v; app (bindExpr env) l)
  | INST (FID v, l) => (bindvar env v; app (fn (_, e) => bindExpr env e) l)
  | INST (PFID(p, v), l) =>
      (bindvar env v; app (fn (_, e) => bindExpr env e) l)
  | BINOP(_, e1, e2) => (bindExpr env e1; bindExpr env e2)
  | BODY l => (app (fn (_, e) => bindExpr env e) l)
  | LINK l => (app (fn (_, e) => bindExpr env e) l)
  | FORCE l => (app (fn (_, e) => bindExpr env e) l)
  | TORQUE l => (app (fn (_, e) => bindExpr env e) l)
  | ROTATION {angle, axis} => (bindExpr env angle; bindExpr env axis)
  | DISPLACEMENT {translation, rotation} =>
      (bindExpr env translation; bindExpr env rotation)
  | FRAME {body, displacement} =>
      (bindExpr env body; bindExpr env displacement)
  | LOCAL (dl, e) =>
      let val nenv = (app (fn (_, e) => bindExpr env e) dl;
                      rev (map (fn (v, _) => v) dl))
      in bindExpr (compose nenv env) e end

and bindDecls env l =
  let fun iter env [] = []
        | iter env (h::t) =
          let val nenv = bindDecl env h
          in
            compose nenv (iter (compose nenv env) t)
          end
  in
    rev (iter env l)
  end

(* la gestion simpliste des erreurs *)
exception TCexn of string
fun TCError mesg = raise (TCexn mesg)

fun isScalar INTy = true
```

```
  | isScalar FLOATty = true
  | isScalar _ = false

(* la fonction qui donne le type des variables "inconnues" *)
fun TCUvar s =
  case s of
  "s" => SCALARty
  | "scalar" => SCALARty
  | _ => TCError "unknown unknown"

fun TypeCheck decl =
  case decl of
    LET l =>
      map (fn (v, e) => (v, setType (v, TC e))) l
  | PACKAGE (v, dl) =>
      let val tdl = fold (op @) (map TypeCheck dl) [] in
        [(v,
         setType(v, PACKAGEty (map (fn (v, t) => (getName v, t)) tdl)))]
      end
  | PARAM (f, l, e) => TCError "TypeCheck/NYI PARAM"
  | JOIN {body1, body2, link} =>
      (case (TC body1, TC body2, TC link) of
          (BODYty, BODYty, LINKty) => []
       | _ => TCError "invalid JOIN")
  | APPLY {force, body} =>
      (case (TC force, TC body) of
         (FORCEty, BODYty) => []
       | _ => TCError "invalid FORCE")

and TC expr =
  case expr of
    VAR v => getType v
  | UVAR s => TCUvar s
  | INT i => SCALARty
  | FLOAT f => SCALARty
  | SELECT (v, c) =>
      let fun validComp (c, spec) =
        Utils.getField (spec, c)
        handle Utils.NotAField _ => TCError "not a field"
      in
        case getType v of
          BODYty => validComp (c, Components.body)
        | LINKty => validComp (c, Components.link)
        | FORCEty => validComp (c, Components.force)
        | TORQUEty => validComp (c, Components.torque)
        | _ => TCError "non valid selection"
      end
  | FROM(p, PID v) =>
      let val t = getType p in
        case t of
```

```
          PACKAGEty l => UNKNOWNty
        | _ => TCError ((getName p)^" is not a package")
      end
  | FROM(p, SPID(v, c)) => UNKNOWNty
  | VECTOR (e1, e2, e3) =>
      let val (t1, t2, t3) = (TC e1, TC e2, TC e3) in
        UNKNOWNty
      end
  | CALL (v, l) => UNKNOWNty
  | INST (FID v, l) => UNKNOWNty
  | INST (PFID(p, v), l) => UNKNOWNty
  | BINOP(bop, e1, e2) =>
      (case (TC e1, TC e2) of
         _ => UNKNOWNty)
  | BODY l => (TCSpec (l, Components.body); BODYty)
  | LINK l => (TCSpec (l, Components.link); LINKty)
  | FORCE l => (TCSpec (l, Components.force); FORCEty)
  | TORQUE l => (TCSpec (l, Components.torque); TORQUEty)

  | ROTATION {angle, axis} =>
      (case (TC angle, TC axis) of
         (SCALARty, VECTORty) => ROTty
       | _ => TCError "not a valid rotation")
  | DISPLACEMENT {translation, rotation} =>
      (case (TC translation, TC rotation) of
         (VECTORty, ROTATIONty) => DISPLty
       | _ => TCError "not a valid displacement")

  | FRAME {body, displacement} =>
      (case (TC body, TC displacement) of
         (BODYty, DISPLty) => FRAMEty
       | _ => TCError "not a valid frame")

  | LOCAL (dl, e) =>
      let
        val t = UNKNOWNty
      in
        UNKNOWNty
      end
  | _ => TCError "strange expression"

(* suppose que l et spec sont trie'es *)
and TCSpec (l, spec) =
  case (l, spec) of
    ([], _) => true
  | ((c, e)::r, (c', t')::r') =>
      if c = c' then
        let val t = TC e in
          if t = t' then
            TCSpec(r,r')
```

```
              else
                TCError "wrong component type"
            end
          else
            TCSpec(l, r')
      | _ => TCError "wrong component"

  end




(*
        LET l =>
      | PARAM (f, l, e) =>
      | PACKAGE (v, dl) =>
      | JOIN {body1, body2, link} =>

        VAR v =>
      | INT i =>
      | FLOAT f =>
      | SELECT (v, c) =>
      | FROM(p, PID v) =>
      | FROM(p, SPID(v, c)) =>
      | VECTOR (e1, e2, e3) =>
      | CALL (v, l) =>
      | INST (FID v, l) =>
      | INST (PFID(p, v), l) =>
      | BINOP(bop, e1, e2) =>
      | BODY l =>
      | LINK l =>
      | ROTATION {angle, axis} =>
      | DISPLACEMENT {translation, rotation} =>
      | FRAME {body, displacement} =>
      | LOCAL (dl, e) =>
*)
```

B.2 manipulation d'expressions matricielles – code *MAPLE*

```
#################################################################
#
#                    CAPOLSINI Patrick
#               INRIA - Universite' de Nice
#                         1991
#
#  * De'finition d'ope'rateurs matriciels
#  * Les re'gles de simplifications associe'es
#  * Les re'gles d'expansion associe'es
#  * L'e'valuation effective des expressions (apres remplissage des
#    matrices)
#  * Les utilitaires de calcul de taille et de proprie'te'
#
#
#################################################################

#         LES OPERATEURS MATRICIELS

# Remise a' zero des tables de remember
# ENTREE : NULL
# SORTIE : NULL
# EFFET DE BORD : remise a zero des tables de remember
init_mat_op := proc()

# les ope'rateurs
    forget(`&*`);
    forget(`&+`);
    forget(`&^`);
    forget(`&t`);
    forget(`&~`);
    forget(`&s`);
    forget(`&i`);
    forget(`&.`);
    forget(`&x`);
# les expansions
    forget(expand);
    forget(`expand/&*`);
    forget(`expand/&+`);
    forget(`expand/&s`);
    forget(`expand/&t`);
    forget(`expand/&~`);
    forget(`expand/&^`);
    forget(`expand/frame`);
# les utilitaires matriciels
    forget(get_dim);
    NULL;
end:

# Remise a' zero des tables de remember
# ENTREE : NULL
# SORTIE : NULL
# EFFET DE BORD : remise a zero des tables de remember
init_evalmat := proc()

    forget(`evalmat/&~`);
    forget(`evalmat/&s`);
    forget(`evalmat/&*`);
    forget(`evalmat/&+`);
    forget(`evalmat/&^`);
    forget(`evalmat/&t`);
    forget(`evalmat/auto_assign`);
    NULL;
end:

# ENTREE : les arguments de l'operateur
# SORTIE : l'expression matricielle
# la multiplication matricielle
# &*(A, &*(B,C)) = &*(A,B,C)
```

```
# &*(A,Id) = &*(Id,A) = Id
# &*(A, A &^ 2) = A &^ 3
# &*(A, 0) = &*(0, A) = 0
# A &* (&s(x, B)) = &s(x, (A &* B))
# A &* (&~ A) = (&~ A) &* A =0
`&*`:=proc()
local _AA, i, j, s, aux;
options remember;

    _AA := [args];
    _AA := map(proc(x) if x <> Id then x fi end,_AA);
    _AA := map(proc(x,pn)
                  if type(x,function) and (op(0,x) = pn) then op(x) else x fi
              end                                                              ,
        _AA,procname);
# Pour regrouper les puissances
    for i from 2 while i <= nops(_AA) do
        aux:='bidon';
        if type(_AA[i],`&^`) and
           (op(1,_AA[i]) =_AA[i-1]) then           #A &* (A^2)
           aux:=`&^`(_AA[i-1],op(2,_AA[i])+1);
        elif type(_AA[i],`&^`) and type(_AA[i-1],`&^`) and
           (op(1,_AA[i])=op(1,_AA[i-1])) then      #(A^2) &* (A^3)
           aux:=`&^`(op(1,_AA[i]),op(2,_AA[i])+op(2,_AA[i-1]));
        elif type(_AA[i-1],`&^`) and
             (op(1,_AA[i-1])=_AA[i]) then          #(A^2) &* A
           aux:=`&^`(_AA[i],op(2,_AA[i-1])+1);
        elif _AA[i]=_AA[i-1] then                   #A &* A
           aux:=`&^`(_AA[i],2);
        elif type(_AA[i], `&~`) and                 #A &* (&~ A)
             (op(1, _AA[i]) = _AA[i-1]) then
           RETURN(0);
        elif type(_AA[i-1], `&~`) and               #(&~ A) &* A
             (op(1, _AA[i-1]) = _AA[i]) then
           RETURN(0);
        fi;
        if aux<>'bidon' then
            _AA := subsop(i-1 = NULL,i = aux,_AA);
            _AA := subs(Id=NULL,_AA);
            i:=i-1;
        fi;
    od;
    if member(0,_AA) then RETURN(0) fi;
    if _AA = [] then RETURN(Id) fi;
    if nops(_AA) = 1 then RETURN(op(_AA)) fi;
# Pour extraire les multiplications scalaires
    if member(true,map(type, _AA,`&s`)) and nops(_AA) <> 1 then
        s:=1;
        for i to nops(_AA) do
           if type(_AA[i], `&s`) then
             s := s * op(1, _AA[i]);
             _AA := subsop(i = op(2, _AA[i]), _AA);
           fi;
        od;
        RETURN(`&s`(s, `&*`(op(_AA))));
    fi;
    'procname'(op(_AA))
end:

# ENTREE : les arguments de l'operateur
# SORTIE : l'expression matricielle
# l'addition matricielle
# A &+ 0 = 0 &+ A = 0
# &s(a, A) &+ &s(b, A) = &s(a+b, A)
`&+` := proc()
local _AA, i, t, s, m, mats, result;
options remember;

    _AA := [args];
    _AA := map(proc(x) if x <> 0 then x fi end,_AA);
```

```
    _AA := map(proc(x,pn)
                  if type(x,function) and (op(0,x) = pn) then op(x) else x fi
              end                                                              ,
        _AA,procname);
    _AA := sort(_AA, `address`);
    if _AA = [] then RETURN(0) fi;
# Rajouts personnels
    t := table(); mats := NULL; result := NULL;
    for i in _AA do
       if type(i, `&s`) then
         s := op(1, i);
         m := op(2, i);
         if assigned(t[m]) then
           t[m] := t[m] + s;
         else
           t[m] := s; mats := mats, m;
         fi;
       else
         if assigned(t[i]) then
           t[i] := t[i] + 1;
         else
           t[i] := 1; mats := mats, i;
         fi;
       fi;
    od;
    mats := sort([mats], `address`);
    for i to nops(mats) do
       if t[mats[i]] = 1 then
         result := result, mats[i];
       elif t[mats[i]] <> 0 then
         result := result, `&s`(t[mats[i]], mats[i]);
       fi;
    od;
    if nops([result]) = 1 then
      result
    elif nops([result]) = 0 then
      0
    else
      'procname'(result)
    fi;
end:

# ENTREE : les arguments de l'operateur
# SORTIE : l'expression matricielle
# la puissance matricielle
# A &^ 1 = A
# A &^ 0 = Id
# 0 &^ x = 0
# &s(x, A) &^ y = &s(x^y, (A &^ y))
`&^`:=proc()
local _AA;
options remember;

    _AA := [args];
    _AA := map(proc(x) if x <> 1 then x fi end,_AA);
    if nops(_AA) = 1 then RETURN(_AA[1]) fi;
    if _AA = [] then RETURN(1) fi;
# Rajouts personnels
    if _AA[2]=0 then RETURN(Id) fi;
    if _AA[1]=0 then RETURN(0) fi;
    if type(_AA[1],`&s`) then
      RETURN(`&s`(op(1, _AA[1])^_AA[2],
                  `&^`(op(2, _AA[1]), _AA[2])));
    fi;
    'procname'(op(_AA))
end:

# ENTREE : l'argument de l'operateur
# SORTIE : l'expression matricielle
# la transposee matricielle
```

```
# &t(0) = 0
# &t(Id) = Id
# &t(&t A) = A
# &t(&~ A) = - &~A
# &t(&s(ab, A)) = &s(ab, &t(A))
‘&t‘:=proc()
local _AA, aux;
options remember;

    _AA := [args];
    if nops(_AA) <> 1 then ERROR(‘not used as unary‘) fi;
# Rajouts personnels
    if _AA[] = 0 then
      RETURN(0);
    elif _AA[] = Id then
      RETURN(Id);
    elif type(_AA[], ‘&t‘) then
      RETURN(op(_AA[]));
    elif type(_AA[], ‘&~‘) then
      RETURN(‘&s‘(-1, _AA[]));
    elif type(_AA[], ‘&s‘) then
      RETURN(‘&s‘(op(1, _AA[]), ‘&t‘(op(2, _AA[]))))
    fi;
    ’procname’(op(_AA));
end:

# ENTREE : l’argument de l’operateur
# SORTIE : l’expression matricielle
# le tild matriciel
# &~(0) = 0
# &~(&s(ab, A)) = &s(ab, (&~ A))
‘&~‘:=proc()
local _BB, aux;
options remember;

    _BB := [args];
    if nops(_BB) <> 1 then ERROR(‘not used as unary‘) fi;
# Rajouts personnels
    if _BB[]=0 then
      RETURN(0);
    elif type(_BB[],‘&s‘) then
      RETURN(‘&s‘(op(1, _BB[]), ‘&~‘(op(2, _BB[]))));
     fi;
    ’procname’(op(_BB));
end:

# ENTREE : les arguments de l’operateur
# SORTIE : l’expression matricielle
# la multiplication par un sclaire
# &s(0, A) = 0
# &s(1, A) = A
# &s(2, &s(3, A)) = &s(6, A)
‘&s‘ := proc()
local _BB;
options remember;

    _BB := [args];
    if nops(_BB) <> 2 then ERROR(‘not used as binary‘) fi;
# Rajouts personnels
    if _BB[1] = 0 or _BB[2] = 0 then
      RETURN(0);
    elif _BB[1] = 1 then
      RETURN(_BB[2]);
    elif type(_BB[2], ‘&s‘) then
      RETURN(‘&s‘(_BB[1] * op(1, _BB[2]), op(2, _BB[2])))
    fi;
    ’procname’(op(_BB));
end:

# Operateurs definis a partir des autres
```

```
# ENTREE : l'argument de l'operateur
# SORTIE : l'expression matricielle
# l'inverse matriciel
# definition par &i(A) = A &^ -1
# &i(&i (A)) = A
`&i`:=proc(x)
options remember;

    if nargs <> 1 then ERROR(`invalid arguments`) fi;
    if type(x, `&^`) and (op(2, x) = -1) then
      RETURN(op(1, x));
    fi;
    if op(1, x) = Id then RETURN(Id) fi;
    `&^`(x, -1);
end:

# ENTREE : les arguments de l'operateur
# SORTIE : l'expression matricielle
# le produit scalaire
# definition par &.(A, B) = &t(A) &* B
# &.(&s(2, A), B) = &s(2, (&t(A) &* B))
# &.(B, A) = &t(A) &* B
# mise sous forme canonique (cf ci-dessous)
`&.`:=proc()
local _AA, aux, i;
options remember;

    _AA := [args];
    if nops(_AA) <> 2 then
      ERROR(`bad number of arguments`)
    elif member(0, _AA) then
      RETURN(0);
    fi;
# Rajouts personnels
    aux:=1;
    for i in _AA do
       if type(i,`&s`) then
         aux := aux * op(1, i);
         _AA := subs(i = op(2, i), _AA);
       fi;
    od;
    _AA := sort(_AA, `address`);
    `&s`(aux, `&*`(`&t`(_AA[1]), _AA[2]));
end:

# ENTREE : les arguments de l'operateur
# SORTIE : l'expression matricielle
# le produit vectoriel
# definition par &x(A, B)=&*(&~ A , B)
# &x(A, A) = 0
# &x(A, 0) = 0
# &x(&s(2, A), B) = &s(2, (&~(A) &* B))
# mise sous forme canonique (cf ci-dessous)
`&x`:=proc()
local _BB, _BB2, aux, i;
options remember;

    _BB:=[args];
    if nops(_BB)<>2 then
      ERROR(`bad number of arguments`)
    elif member(0,_BB) or _BB[1] = _BB[2] then
      RETURN(0);
    fi;
# Rajouts personnels
    aux := 1;
    for i in _BB do
       if type(i,`&s`) then
         aux := aux * op(1, i);
         _BB := subs(i = op(2, i),_BB);
```

```
      fi;
    od;
     _BB2 := sort(_BB, `address`);
     if _BB2 <> _BB then aux := - aux fi;
     `&s`(aux, `&*`(`&~`(_BB2[1]), _BB2[2]));
end:

#          LES TYPES DES OPERATEURS
# ENTREE : une expression matricielle
# SORTIE : true ou false
`type/&*`:=proc(x)
    type(x,function) and evalb(op(0,x)=`&*`);
end:

`type/&+`:=proc(x)
    type(x,function) and evalb(op(0,x)=`&+`);
end:

`type/&^`:=proc(x)
    type(x,function) and evalb(op(0,x)=`&^`);
end:

`type/&t`:=proc(x)
    type(x,function) and evalb(op(0,x)=`&t`);
end:

`type/&~`:=proc(x)
    type(x,function) and evalb(op(0,x)=`&~`);
end:

`type/&s`:=proc(x)
    type(x,function) and evalb(op(0,x)=`&s`);
end:

`type/&.`:=proc(x)
    type(x,function) and evalb(op(0,x)=`&.`);
end:

`type/&x`:=proc(x)
    type(x,function) and evalb(op(0,x)=`&x`);
end:

#          LES REGLES D'EXPANSION

# Attention expand/toto est appelle surles arguments de toto et non sur
# l'expression elle meme

# ENTREE : les arguments de l'operateur
# SORTIE : l'expression matricielle expansee
# le expand/&* a 2 arguments
`expand_mult2` := proc(e1, e2)
    if type(e2, `&+`) then
      map((x,c) -> expand(`&*`(c, x)), e2, e1); ";
    elif type(e1, `&+`) then
      map((x,c) -> expand(`&*`(x, c)), e1, e2); ";
    else
      `&*`(e1, e2);
    fi;
end:

# ENTREE : les arguments de l'operateur
# SORTIE : l'expression matricielle expansee
# A &* (B &+ C &+ D) = A&*B &+ A&*C &+ A&*D
`expand/&*`:=proc()
local l_args, i, aux;
options remember;

    l_args := map(expand,[args]);
    if nops(l_args) = 2 then
      `expand_mult2`(l_args[1], l_args[2]);
```

```
    else
      `expand/&*`(`expand_mult2`(l_args[1], l_args[2]),
                  l_args[3..nops(l_args)]);
    fi;
end:

# ENTREE : les arguments de l'operateur
# SORTIE : l'expression matricielle expansee
# expansion du &+
`expand/&+` := proc()
local l_args;
options remember;

    l_args := map(expand, [args]);
    `&+`(l_args[]);
end:

# ENTREE : les arguments de l'operateur
# SORTIE : l'expression matricielle expansee
# expansion du &s
`expand/&s` := proc()
local l_args;
options remember;

    l_args := map(expand, [args]);
    `&s`(l_args[]);
end:

# ENTREE : les arguments de l'operateur
# SORTIE : l'expression matricielle expansee
# &t(A &* B) = &t(B) &* &t(A)
# &t(A &+ B) = &t(A) &+ &t(B)
`expand/&t`:=proc()
local l, i, aux, x;
options remember;

    x := expand(args);
    aux := [];
    if type(x, `&+`) then
      expand(map(`expand/&t`, x)); ";
    elif type(x, `&*`) then
      l := [op(x)];
      for i from nops(l) by -1 to 1 do
         aux:=[aux[], expand(`&t`(l[i]))];
      od;
      expand(`&*`(aux[]));
    else
      `&t`(x);
    fi;
end:

# ENTREE : les arguments de l'operateur
# SORTIE : l'expression matricielle expansee
# &~(A &+ B &+ C) = &~(A) &+ &~(B) &+ &~(C)
# &~(&~(A) &* B) = &~(A) &* &~(B) &+ &s(-1, &~(B) &* &~(A))
`expand/&~`:=proc()
local op1x, op2x, x;
options remember;

    x := expand(args);
    if type(x, `&+`) then
      expand(map(proc(x) expand(`&~`(x)) end, x)); ";
    elif type(x, `&*`) and type(op(1, x), `&~`) then
      op1x := op(1, x);
      subsop(1 = NULL, x); op2x := ";
      expand(expand(`&*`(op1x, expand(`&~`(op2x)))) &+
                &s(-1, expand(`&*`(expand(`&~`(op2x)), op1x)))
            ); ";
    else
    `&~`(x);
```

```
    fi;
end:

# ENTREE : les arguments de l'operateur
# SORTIE : l'expression matricielle expansee
# &^(A &* B, -1) = (B &^ (-1)) &* (A &^ (-1))
# &^(A &* B, x) = &*(A,B, ... ,A,B) x fois bien sur
`expand/&^`:=proc()
local i, l_args;
options remember;

    l_args:=map(expand,[args]);
    if type(l_args[1],`&*`) then
      if l_args[2]=-1 then
        map(proc(x) `&^`(x,-1) end,
            [op(nops(l_args[1])+1-i,l_args[1]) $ (i = 1 .. nops(l_args[1]))]);
        RETURN(`&*`(op(")));
      elif type(l_args[2],numeric) then
        RETURN(`&*`(op(map(op,[[l_args[1]] $ l_args[2]]))));
       fi;
    else
     `&^`(op(l_args));
    fi;
end:

#        EVALUATION EFFECTIVE D'UNE EXPRESSION MATRICIELLE

# ENTREE : un vecteur
# SORTIE : une matrice representant le tild du vecteur
# Construction du tild d'un vecteur
# Internal use only
`evalmat/&^` := proc(v)
local res;
options remember;

    if not(type(v, vector)) then
      ERROR(`Bad type of argument`);
    elif op(2, op(2, eval(v))) <> 3 then
      ERROR(`Bad vector size`);
    else
      res := array(1..3,1..3,antisymmetric);
      res[2,1] := v[3];
      res[3,1] := - v[2];
      res[3,2] := v[1];
    fi;
    op(res);
end:

# Detection de la multiplication de deux scalaires
# Multiplication par un scalaire (linalg[scalarmul] mais avec les arguments
# inverses)
# ENTREE : les arguments d'un `&s`
# SORTIE : la matrice ou le vecteur resultat
# Internal use only
`evalmat/&s` := proc()
options remember;

    if get_dim(args[2]) = [1, 1] then
      args[1] * args[2];
    else
      linalg[scalarmul](args[2], args[1]);
    fi;
end:

# Detection de l'addition de deux scalaires
# Addition de matrices (linalg[add] mais avec plusieurs arguments)
# ENTREE : les arguments d'un `&+`
# SORTIE : la matrice ou le vecteur resultat
`evalmat/&+` := proc()
options remember;
```

```
    if nargs = 1 then
      args[1];
    elif nargs = 2 then
      if get_dim(args[1]) = [1, 1] and get_dim(args[2]) = [1, 1] then
        args[1] + args[2];
      else
        linalg[add](args[1], args[2]);
      fi;
    else
      `evalmat/&+`(args[1], `evalmat/&+`(args[2..nargs]));
    fi;
end:

# Detection de la mise a la puissance d'un scalaire scalaire
# Mise a la puisance d'une matrice
# ENTREE : les arguments d'un `&^`
# SORTIE : la matrice ou le vecteur resultat
# Internal use only
`evalmat/&^` := proc()
options remember;

    if args[2] = -1 then
      linalg[inverse](args[1]);
    elif get_dim(args[1]) = [1, 1] then
      args[1] ^ args[2];
    else
      linalg[multiply](args[1]$args[2]);
    fi;
end:

# Detection de la multiplication de deux scalaires
#              la multiplication par 0
# Multiplication des matrices
# ENTREE : les arguments d'un `&*`
# SORTIE : la matrice ou le vecteur resultat
# Internal use only
`evalmat/&*` := proc()
local dim;
options remember;

    if map(get_dim, [args]) = [[1, 1]$nargs] then
      convert([args], `*`);
    elif member(0, [args]) then
      dim := get_dim(`&*`(args));
      if dim = [1, 1] then
        0;
      else
        array(1..dim[1], 1..dim[2],sparse);
      fi;
    else
      linalg[multiply](args);
    fi;
end:

# Detection du bug de linalg[transpose] pour les antisymmetric
# Transposition des matrices
# ENTREE : l'argument d'un `&t`
# SORTIE : la matrice ou le vecteur resultat
# Internal use only
`evalmat/&t` := proc()
local res, dim, i, j;
options remember;

    dim := get_dim(args[1]);
    if get_prop(args[1]) = antisymmetric then
      res := array(1..dim[1], 1..dim[2], antisymmetric);
      for i to dim[1] do
         for j from (i+1) to dim[2] do
            res[i, j] := - args[1][i, j];
```

```
        od;
      od;
      RETURN(op(res));
    else
      linalg[transpose](args[1]);
    fi;
end:

# ENTREE : une expression matricielle, <true>
#          Si appellee avec un second argument true (c'est un appel
#          recursif), il n'y a alors pas de calcul des valeurs non
#          assignees
#          Si appellee avec un second argument false il y a mise a
#          zero des tables de remember
# SORTIE : la matrice ou le vecteur resultat
# Toutes les expressions doivent etre des vecteurs ou des matrices
# Ceci n'est qu'une ebauche de ce que devra etre la fonction plus tard

evalmat := proc()
local mats, exp, dim;
options remember;

    exp := args[1];
    if nargs = 1 then
      mats := map(proc(x) if get_dim(x) <> [1, 1] and
                             not(type(x, array)) and x <> 0 then
                           x fi; end,
              indets(exp, name));
      if mats <> {} then
        print(`WARNING automatical assignation for :`);
        print(op(mats));
      fi;
    fi;
    if nargs = 2 and not(args[2]) then      # appel avec false
      `init_evalmat`();                     # modif le 7/04
    fi;
    if type(exp, name) then
      if exp = `Id` then
        array(1..3, 1..3, identity);
      elif type(exp, array) then               # modif le 12/02
        map(evalmat, exp);
      elif get_dim(exp) = [1, 1] then
        op(exp);
      else
         `evalmat/auto_assign`(exp, get_dim(exp), get_prop(exp));
      fi;
    elif type(exp, array) then
#op(exp);
      map(evalmat, exp, true);
#    elif exp = 0 and get_dim(exp) <> [1, 1] then
#      dim := get_dim(exp);
#      array(1..dim[1], 1..dim[2], sparse);

    elif type(exp, `&s`) then
      `evalmat/&s`(op(map(evalmat, exp, true)));
    elif type(exp, `&+`) then
      `evalmat/&+`(op(map(evalmat, exp,true)));
    elif type(exp, `&^`) then
      `evalmat/&^`(op(map(evalmat, exp, true)));
    elif type(exp, `&*`) then
      `evalmat/&*`(op(map(evalmat, exp, true)));
    elif type(exp, `&t`) then
      `evalmat/&t`(evalmat(op(exp), true));
    elif type(exp, `&~`) then
      `evalmat/&~`(evalmat(op(exp), true));
    elif type(exp, function) and op(0, exp) = frame then
      evalmat(_frame[op(exp)])
#      evalmat(_frame[op(exp)], true)
    elif type(exp, {numeric, name, function}) then  # modif le 12/02
      exp;
```

```
    elif type(exp, {‘*‘, ‘**‘, ‘+‘}) then           # modif le 12/02
      map(evalmat, exp);
    else
      ERROR(‘Matricial evaluation not defined‘);
    fi;
end:

# ENTREE : un nom de matrice ou de vecteur, sa taille, sa propriete
# SORTIE : le vacteur ou la matrice apres "remplissage"
# "Remplissage" d'une matrice  a partir de sa taille et de son nom
‘evalmat/auto_assign‘ := proc(nom, dim, prop)
local res, i, j;
options remember;

    if dim = [1, 1] then
      RETURN(nom);
    elif dim = [3, 1] then
      res := array(1..3);
      for i to 3 do
         res[i] := nom.‘[‘.i.‘]‘;
      od;
    elif prop = identity then
      res := array(1..dim[1], 1..dim[2], prop);
    elif prop = antisymmetric then
      res := array(1..dim[1], 1..dim[2], prop);
      for i to dim[1] do
         for j from (i+1) to dim[2] do
            res[i, j] := nom.‘[‘.i.‘,‘.j.‘]‘;
         od;
      od;
    else
      if prop = symmetric then
        res := array(1..dim[1], 1..dim[2], prop);
      else
        res := array(1..dim[1], 1..dim[2]);
      fi;
      for i to dim[1] do
         for j to dim[2] do
            res[i, j] := nom.‘[‘.i.‘,‘.j.‘]‘;
         od;
      od;
    fi;
op(res);
end:

#               UTILITAIRES MATRICIELS
#       Les versions table globale et mat(_nom, dim, prop) st acceptees

# ENTREE : une expression matricielle, <true>
#          si second argument est true, il y a verification de
#          coherence des tailles
# SORTIE : la dimension recherchee
get_dim := proc(exp, bool)
local aux;
options remember;

    if nargs = 2 and bool and not(verif_dim(exp)) then
      ERROR(‘Bad dimension in expression‘);
    fi;
    if type(exp, array) then
      aux := [op(2, eval(exp))];
      if nops(aux) = 2 then
        [op(2, aux[1]), op(2, aux[2])];
      else
        [op(2, aux[1]), 1];
      fi;
    elif assigned(_mat) and
         _mat[exp] <> 0 then     # si dimension dans tables globale
      _mat[exp][1];
    elif type(exp, function) and # si objet du genre mat(_nom, dim,...)
```

```
        op(0, exp) = mat then
      op(2, exp);
    elif type(exp, `&~`) then
      [3, 3];
    elif type(exp, `&t`) then
      get_dim(op(exp));
      ["[2], "[1]];
    elif type(exp, `&^`) then
      get_dim(op(1, exp));
    elif type(exp, `&*`) then
      [get_dim(op(1, exp))[1],get_dim(op(nops(exp), exp))[2]];
    elif type(exp, `&s`) then
      get_dim(op(2, exp));
    elif type(exp, `&+`) then
      get_dim(op(1, exp));
    elif type(exp, `function`) and op(0, exp) = `frame` then
      get_dim(op(1, exp));
    else              # un scalaire
      [1,1];
    fi;
end:

# ENTREE : un nom de variable, une taille
# SORTIE : NULL
# EFFET DE BORD : affectation du resultat de "get_dim"
# De'clare la dimension d'une variable
put_dim := proc(exp, ltaille)

    if _mat[exp] <> 0 then
      _mat[exp] := op(subsop(1=ltaille, [_mat[exp]]));
    else
      get_dim(exp) := ltaille;
    fi;
    NULL;
end:

# Verification de la coherence des dimensions d'une expression
# ENTREE : une expression matricielle
# SORTIE : true ou false
verif_dim := proc(exp)
local l_dim;

    if type(exp, `&~`) then
      RETURN(evalb(get_dim(op(exp)) = [3, 1]));
    elif type(exp, `&^`) then
      get_dim(op(1, exp));
      RETURN(evalb("[1] = "[2]));
    elif type(exp, `&*`) then
      l_dim := map(get_dim, [op(exp)]);
      for i to nops(l_dim)-1 do
         if l_dim[i][2] <> l_dim[i+1][1] then
           RETURN(false);
         fi;
      od;
    elif type(exp,`&s`) then
      RETURN(type(op(1, exp), algebraic));
    elif type(exp,`&+`) then  # tous de meme dimension
      l_dim := map(get_dim,[op(exp)]);
      RETURN(type(l_dim, [l_dim[1]$(nops(l_dim))]));
    elif type(exp, `function`) and op(0, exp) = `frame` then
      RETURN(get_dim(op(1, exp), true))
    fi;
    true;
end:

# ENTREE : un nom de matrice ou de vecteur
# SORTIE : la propriete attachee a la matrice (si il y a lieu)
#          a savoir : vector, matrix, symmetric, antisymmetric, rotation,
#          translation
get_prop := proc(exp)
```

```
    if type(exp, array) then
      RETURN(linalg[indexfunc](exp));
    elif assigned(_mat) and
         _mat[exp] <> 0 then     # si propriete dans tables globale
      RETURN(_mat[exp][2]);
    elif type(exp, function) and # si objet du genre mat(_nom, dim,...)
         op(0, exp) = mat then
      RETURN(op(3, exp));
    fi;
    NULL;
end:

# ENTREE : un nom de variable, une propriete
# SORTIE : NULL
# EFFET DE BORD : affectation du resultat de "get_prop"
# De'clare la proprie'te' d'une variable
put_prop := proc(exp, prop)

    if _mat[exp] <> 0 then
      _mat[exp] := op(subsop(2=prop, [_mat[exp]]));
    else
      get_prop(exp) := prop;
    fi;
    NULL;
end:
```

B.3 manipulation d'expressions matricielles – code *ULYSSE*

```
$d=====================================================================
$$ \titresource{Gemmes en \ulysse\ }
$$
$=====================================================================

new_operator ([name = "|+|":Lisp,
                      properties =
                  "((is-a)(is-c)(has-n . _0)(has-s . -))":Lisp]);

new_operator ([name = "|-|":Lisp,
           properties = "((is-inv))":Lisp]);

new_operator ([name = "|*|":Lisp,
           properties = "((is-a)(is-c)(has-n . 1)
                          (has-abs . _0)(is-morph . ((+ +) (- -)))
                          (is-hom . ((*e *e))))":Lisp]);

new_operator ([name = "|.|":Lisp,
           properties = "((is-a) (has-ng . Id)(has-nd . Id) (has-absg . _0)
                          (has-absd . _0)(has-sg . inv) (has-sd . inv)
                          (is-hom . ((*e *e)))
                          (is-morph . ((- -)(+ +))))":Lisp]);

new_operator ([name = "|inv|":Lisp,
           properties = "((is-inv))":Lisp]);

new_operator ([name = "trans":Lisp,
           properties = "((is-inv)(is-hom . ((*e *e)))
                          (is-morph . ((+ +)(- -))))":Lisp]);

new_operator ([name = "tild":Lisp,
           properties = "((is-hom . ((*e *e)))
                          (is-morph . ((+ +)(- -))))":Lisp]);

new_operator ([name = "|&|":Lisp,
           properties = "((is-a)(is-c)(is-id))":Lisp]);

rules();

$d=====================================================================
$$ Traduction gemmes -> \ulysse\ .
$=====================================================================
rs0:= [smul(#x,#y) -> #x *e #y]:RS;

$d=====================================================================
$$ Simplifications des termes sans proprietes.
$=====================================================================
rs1:=
[#x *e (-#y) -> (-#x) *e #y,
(#x *e (#y *e #z)) -> (#x * #y) *e #z,
- (#x *e #y) -> (-#x) *e #y,
#x *e #y + #z *e #y -> (#x + #z) *e #y,
inv(#x . #y) -> inv(#y) . inv(#x),
trans(#x . #y) -> trans(#y) . trans(#x),
tild(#a) . #b + tild(#b) . #a -> _0,
trans(#x . #y) -> trans(#y) . trans(#x),
trans(tild(#v)) -> -tild(#v),
tild(#v) . #v -> _0,
tild(X0) . Z0 -> -Y0,
tild(X0) . Y0 -> Z0,
tild(Y0) . Z0 -> X0,
tild(Y0) . X0 -> -Z0,
tild(Z0) . X0 -> Y0,
tild(Z0) . Y0 -> -X0,
tild(_0) -> _0,
#x *e _0 -> _0,
rot(#v,#a) . #v -> #v,
trans(rot(#v,#a)) . rot(#v,#a) -> Id,
```

```
rot(#v,#a) . trans(rot(#v,#a)) -> Id,
rot(Z0,#a) . X0 -> (sin(#a) *e Y0 + cos(#a) *e X0),
rot(Z0,#a) . Y0 -> (cos(#a) *e Y0 + -sin(#a) *e X0),
tild(rot(#v,#a) . #x) ->
         rot(#v,#a) . tild(#x) . trans(rot(#v,#a)),

trans(#x) . tild(#x) -> _0

/*
tild(#a) . tild(#b) . tild(#b) . #a
  + tild(#b) . tild(#a) . tild(#a) . #b -> _0,

#x *e (tild(#a) . tild(#b) . tild(#b) . #a)
  + #x *e (tild(#b) . tild(#a) . tild(#a) . #b) -> _0,
*/
]:RS;

$d=========================================================================
$$ Met des proprietes aux termes.
$=========================================================================
rs3:= [a -> a:vector,
       b -> b:(matrix & sym),
       c -> c:vector,
       d -> d:(matrix & antisym),
       RJ1 -> RJ1:vector,
       r10 -> r10:vector,
       y10 -> y10:vector,
       tild(#x):#x1 . #y:(antisym & #y1) . tild(#x):#x2
         -> (tild(#x):#x1 . #y:(antisym & #y1) . tild(#x):#x2)
              :(matrix & antisym),
       tild(#x) -> tild(#x:vector):(matrix & antisym)
       ]:RS;

$d=========================================================================
$$ Simplification de termes avec proprie'te's.
$=========================================================================
rs3b:= [
       (trans(#a:vector) . #b:vector ->
           trans(#b:vector) . #a:vector)
             when (#a > #b),

       (tild(#a:vector):#x . #b:vector ->
          - (tild(#b:vector):#x . #a:vector))
             when (#a > #b),

       tild(#a):#p1 . tild(#b):#p2 . (#c:vector) ->
          (trans(#a) . #c) *e #b + -(trans(#a) . #b) *e #c,

       #x . tild(#a) . #b:vector + #x . tild(#b) . #a:vector -> _0,

       #y *e (#x . tild(#a) . #b:vector)
         + #y *e (#x . tild(#b) . #a:vector) -> _0,

       #y *e (tild(#a) . #b:vector)
         + #y *e (tild(#b) . #a:vector) -> _0,

       trans(#x:(sym & #y)) -> #x:(sym & #y),
       trans(#x:(antisym & #x1)) -> -(#x:(antisym & #x1)),
       inv(#x:(ortho & #y)) -> trans(#x:(ortho & #y)),
       trans(#x:vector) . #y:(antisym & #z) . #x:vector -> _0,
       trans(#x:(antisym & #y)) -> - #x:(antisym & #y)
       ]:RS;

$d=========================================================================
$$ Normalisation des proprietes.
$=========================================================================
rs6:= [(#x:#y):#y -> #x:#y,
       (#x:#y):#z -> #x:(#y & #z)
       ]:RS;
```

```
$d=========================================================================
$$ Simplification et normalisation des proprietes.
$=========================================================================
rs5:= [calc(#b) -> rewrite(replace(calc(#b),‘rs3),‘rs6)]:RS;

$d=========================================================================
$$ Vire les propri\’et\’es.
$=========================================================================
rs2:= [#xxx:#y  -> #xxx]:RS;

$d=========================================================================
$$ Expand.
$=========================================================================
rsexpand:= [#x . (#y + #z) -> #x . #y + #x . #z,
            (#y + #z) . #x  -> #y . #x + #z . #x,
            #x *e (#y + #z) -> #x *e #y + #x *e #z,
            (#y + #z) *e #x  -> #y *e #x + #z *e #x,
            tild(tild(#x).#y)
               -> tild(#x).tild(#y) + - tild(#y).tild(#x)
            ]:RS;

$d=========================================================================
$$ Met les proprietes, simplifie, les vire.
$=========================================================================
rs71:= [calc(#xx) ->  rewrite(rewrite(replace_n(calc(#xx),
                                                ‘rs5),
                                          ‘rs3b),
                                    ‘rs2)]:RS;

$ test:= replace(calc(trans(a) . c + - trans(c) . a), ‘rs71);

$d=========================================================================
$$ Met les proprietes, simplifie, les vire, et simplifie sans les prop.
$=========================================================================
rs7:= [calc(#xx) ->  rewrite(replace(calc(#xx), ‘rs71),‘rs1)]:RS;

rsglobal:=[calcprop(#a) -> replace_n(calc(rewrite(#a,‘rs0)),‘rs7)]:RS;


rsglobal1:=[calcprop(#a) -> replace(calc(rewrite(#a,‘rs0)),‘rs7)]:RS;

$d=========================================================================
$$ rewrite(calcprop(‘toto),‘rsglobal) = 0~?~?~?
$=========================================================================
toto :=

 - (smul(THp1*THp1 ,
        trans(RJ1) . tild(RJ1) . In10 . RJ1)

    + -smul(m10* THp1*Thp1 ,
            trans(RJ1). tild(r10). tild(RJ1).tild(RJ1). y10)

    + -smul(THp1,

            (smul(m10, trans(RJ1) . tild(r10))

             + - smul(m10, trans(RJ1) . tild(y10)))

            . (smul(THp1, tild(RJ1). tild(r10))

               + - smul(THp1, tild(r10) . tild(RJ1)))

            . RJ1));

/*
e1:= trans(RJ1).tild(r10).RJ1;

e2 := tild(a) . tild(c) . tild(c) . a
      + tild(c) . tild(a) . tild(a) . c ;
```

```
e3 := tild(a:vector) . tild(c:vector) . tild(c:vector) . a:vector
      + tild(c:vector) . tild(a:vector) . tild(a:vector) . c:vector ;

e4:= tild(c:vector) . tild(c:vector) . a:vector;
e5:= tild(c) . tild(c) . a;

rewrite(‘e4,[tild(#a:vector) . tild(#b:vector) . (#c:vector) ->
          (trans(#a) . #c) *e #b + -(trans(#a) . #b) *e #c]:RS);

rewrite(‘e5,[tild(#a) . tild(#b) . (#c) ->
          (trans(#a) . #c) *e #b + -(trans(#a) . #b) *e #c]:RS);

rules(simplify(#x) -> rewrite(calcprop(#x),‘rsglobal),
      calc(#x) -> #x);

simplify(tild(a) . a);

simplify(rot(v,a) . trans(rot(v,a)));
*/
```

Annexe C

étude des mécanismes bouclés

C.1 exemples de mécanismes bouclés

C.1.1 maille d'antenne déployable à un niveau – fichier `maille1.gem`

```
> donnees;
COM-> lect;

                    IMPRESSION DES DONNEES DISPONIBLES

 Les valeurs des donnees initialisees sont :

   nombre de corps :   8

   numeros des corps et nombre de modes:

      corps :   1   nombre de modes :   0
      corps :   2   nombre de modes :   0
      corps :   3   nombre de modes :   0
      corps :   4   nombre de modes :   0
      corps :   5   nombre de modes :   0
      corps :   6   nombre de modes :   0
      corps :   7   nombre de modes :   0
      corps :   8   nombre de modes :   0

   nombre de points :   18

   numeros des points et proprietaires:

      point :   100   proprietaire :   0
      point :   101   proprietaire :   1
      point :   211   proprietaire :   1
      point :   212   proprietaire :   2
      point :   322   proprietaire :   2
      point :   323   proprietaire :   3
      point :   433   proprietaire :   3
      point :   434   proprietaire :   4
      point :   544   proprietaire :   4
      point :   545   proprietaire :   5
      point :   655   proprietaire :   5
      point :   656   proprietaire :   6
      point :   766   proprietaire :   6
      point :   767   proprietaire :   7
      point :   877   proprietaire :   7
      point :   878   proprietaire :   8
      point :   808   proprietaire :   8
      point :   800   proprietaire :   0

   nombre de liaisons :   9

      liaison   10   de type   rj
      entre le point   101   et le point   100
```

```
      liaison   21   de type   rj
      entre le point   211   et le point   212

      liaison   32   de type   rj
      entre le point   322   et le point   323

      liaison   43   de type   rj
      entre le point   433   et le point   434

      liaison   54   de type   rj
      entre le point   544   et le point   545

      liaison   65   de type   rj
      entre le point   655   et le point   656

      liaison   76   de type   rj
      entre le point   766   et le point   767

      liaison   87   de type   rj
      entre le point   877   et le point   878

      liaison   80   de type   rj
      entre le point   800   et le point   808

   nombre de forces :   0

 Donnees assignees pour le corps 1 :

   - masse ( 0 )
   - inertie ( 0(3, 3) )
   - distance point de reference CdG ( 0(3, 1) )
   - distance point de reference point 101 ( 0(3, 1) )
   - matrice de passage locale du point 101 ( id_(3) )
   - distance point de reference point 211 ( r0211 )
   - matrice de passage locale du point 211 ( id_(3) )

...
> op(r0211);

                                     [ 0 ]
                                     [   ]
                                     [ L ]
                                     [   ]
                                     [ 0 ]
...

 Donnees assignees pour le corps 2 :

   - masse ( 0 )
   - inertie ( 0(3, 3) )
   - distance point de reference CdG ( 0(3, 1) )
   - distance point de reference point 212 ( 0(3, 1) )
   - matrice de passage locale du point 212 ( id_(3) )
   - distance point de reference point 322 ( r0211 )
   - matrice de passage locale du point 322 ( id_(3) )

 Donnees assignees pour le corps 3 :

   - masse ( 0 )
   - inertie ( 0(3, 3) )
   - distance point de reference CdG ( 0(3, 1) )
   - distance point de reference point 323 ( 0(3, 1) )
   - matrice de passage locale du point 323 ( id_(3) )
   - distance point de reference point 433 ( 0(3, 1) )
   - matrice de passage locale du point 433 ( PL433 )
```

```
...
> op(PL433);

                          [ 1      0         0    ]
                          [                       ]
                          [ 0     -1/2    1/2 sq3 ]
                          [                       ]
                          [ 0  - 1/2 sq3    -1/2  ]
...

 Donnees assignees pour le corps 4 :

   - masse ( 0 )
   - inertie ( 0(3, 3) )
   - distance point de reference CdG ( 0(3, 1) )
   - distance point de reference point 434 ( 0(3, 1) )
   - matrice de passage locale du point 434 ( id_(3) )
   - distance point de reference point 544 ( r0544 )
   - matrice de passage locale du point 544 ( id_(3) )

...
> op(r0544);

                                    [ 0 ]
                                    [   ]
                                    [ L ]
                                    [   ]
                                    [ 0 ]
...

 Donnees assignees pour le corps 5 :

   - masse ( 0 )
   - inertie ( 0(3, 3) )
   - distance point de reference CdG ( 0(3, 1) )
   - distance point de reference point 545 ( 0(3, 1) )
   - matrice de passage locale du point 545 ( id_(3) )
   - distance point de reference point 655 ( r0544 )
   - matrice de passage locale du point 655 ( id_(3) )

 Donnees assignees pour le corps 6 :

   - masse ( 0 )
   - inertie ( 0(3, 3) )
   - distance point de reference CdG ( 0(3, 1) )
   - distance point de reference point 656 ( 0(3, 1) )
   - matrice de passage locale du point 656 ( id_(3) )
   - distance point de reference point 766 ( 0(3, 1) )
   - matrice de passage locale du point 766 ( PL433 )

 Donnees assignees pour le corps 7 :

   - masse ( 0 )
   - inertie ( 0(3, 3) )
   - distance point de reference CdG ( 0(3, 1) )
   - distance point de reference point 767 ( 0(3, 1) )
   - matrice de passage locale du point 767 ( id_(3) )
   - distance point de reference point 877 ( r0877 )
   - matrice de passage locale du point 877 ( id_(3) )

...
> op(r0877);

                                    [ 0 ]
                                    [   ]
                                    [ L ]
                                    [   ]
                                    [ 0 ]
...
```

```
 Donnees assignees pour le corps 8 :

   - masse ( 0 )
   - inertie ( 0(3, 3) )
   - distance point de reference CdG ( 0(3, 1) )
   - distance point de reference point 878 ( 0(3, 1) )
   - matrice de passage locale du point 878 ( id_(3) )
   - distance point de reference point 808 ( r0877 )
   - matrice de passage locale du point 808 ( id_(3) )

 Donnees assignees pour le corps 0 :

   - gravite ( 0(3, 1) )
   - distance point de reference point 100 ( 0(3, 1) )
   - matrice de passage locale du point 100 ( id_(3) )
   - distance point de reference point 800 ( 0(3, 1) )
   - matrice de passage locale du point 800 ( PL800 )
...
> op(PL800);

                          [ 1     0         0     ]
                          [                       ]
                          [ 0    -1/2   - 1/2 sq3 ]
                          [                       ]
                          [ 0  1/2 sq3     -1/2   ]
...

> evalmat(matcontb2);

[- (- ((sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) cos(TH32)

  + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)) sin(TH43) - 1/2 (

 - (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)

  + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) cos(TH43)) L

  - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) L - cos(TH10) L, - (- (

 (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) cos(TH32)

  + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)) sin(TH43) - 1/2 (

 - (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)

  + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) cos(TH43)) L

  - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) L, - (- (

 (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) cos(TH32)

  + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)) sin(TH43) - 1/2 (

 - (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)

  + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) cos(TH43)) L, - 1/4

    2
 sq3  cos(TH43) L (- (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)

  + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) + 1/2 (- (

 (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) cos(TH32)

  + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)) sin(TH43) - 1/2 (

 - (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
```

```
 + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) cos(TH43)) L,

(- ( (1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) cos(TH76)
 + (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) sin(TH76))
sin(TH65) + (
1/2 (1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) sin(TH76)
 - 1/2 (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) cos(TH76)
 - 1/4 sq3) cos(TH65)) L (
1/2 (- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) sin(TH76) sq3
 - 1/2 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) cos(TH76) sq3
 + 1/4 sq3) - (- (
(- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) cos(TH76)
 + (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) sin(TH76)) sin(TH65) +
(1/2 (- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) sin(TH76)
 - 1/2 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) cos(TH76)
          2
 - 1/4 sq3 ) cos(TH65)) L (
1/2 (1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) sin(TH76) sq3
 - 1/2
(- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) cos(TH76) sq3
 + 1/4), - 1/2 (- (
(1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) cos(TH76)
 + (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) sin(TH76))
sin(TH65) + (
1/2 (1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) sin(TH76)
 - 1/2 (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) cos(TH76)
 - 1/4 sq3) cos(TH65)) L sq3 + 1/2 (- (
(- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) cos(TH76)
 + (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) sin(TH76)) sin(TH65) +
(1/2 (- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) sin(TH76)
 - 1/2 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) cos(TH76)
          2
 - 1/4 sq3 ) cos(TH65)) L, 1/2 (
- (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) L - (- (
(1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) cos(TH76)
 + (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) sin(TH76))
```

```
sin(TH65) + (
1/2 (1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) sin(TH76)

 - 1/2 (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) cos(TH76)
 - 1/4 sq3) cos(TH65)) L) sq3
 + 1/2 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) L + 1/2 (- (
(- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) cos(TH76)
 + (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) sin(TH76)) sin(TH65) +
(1/2 (- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) sin(TH76)
 - 1/2 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) cos(TH76)
          2
 - 1/4 sq3 ) cos(TH65)) L, 1/2 (- 1/2 sq3 cos(TH80) L
 - (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) L - (- (
(1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) cos(TH76)
 + (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) sin(TH76))
sin(TH65) + (
1/2 (1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) sin(TH76)
 - 1/2 (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) cos(TH76)
 - 1/4 sq3) cos(TH65)) L) sq3 - 1/4 cos(TH80) L
 + 1/2 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) L + 1/2 (- (
(- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) cos(TH76)
 + (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) sin(TH76)) sin(TH65) +
(1/2 (- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) sin(TH76)
 - 1/2 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) cos(TH76)
          2
 - 1/4 sq3 ) cos(TH65)) L]
[(- ((cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)
  + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) sin(TH32)) sin(TH43) - 1/2 (
 - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
  + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) cos(TH43)) L
  + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) L - sin(TH10) L, (- (
 (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)
  + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) sin(TH32)) sin(TH43) - 1/2 (
 - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
  + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) cos(TH43)) L
  + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) L, (- (
```

```
(cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)
 + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) sin(TH32)) sin(TH43) - 1/2 (
- (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)

 + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) cos(TH43)) L, 1/4
   2
sq3  cos(TH43) L (- (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
 + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) - 1/2 (- (
(cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)
 + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) sin(TH32)) sin(TH43) - 1/2 (
- (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
 + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) cos(TH43)) L, - (-
 ((1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) cos(TH76)
 + (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) sin(TH76))
sin(TH65) + (
1/2 (1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) sin(TH76)
 - 1/2 (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) cos(TH76)
 - 1/4 sq3) cos(TH65)) L (
1/2 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76) sq3
 - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76) sq3) + (- (
(cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
 + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) sin(TH65) + (
1/2 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76)
 - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76)) cos(TH65)) L (
1/2 (1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) sin(TH76) sq3
 - 1/2
(- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) cos(TH76) sq3
 + 1/4), - 1/2 (- ((cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
 + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) sin(TH65) + (
1/2 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76)
 - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76)) cos(TH65)) L,
- 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) L - 1/2 (- (
(cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
 + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) sin(TH65) + (
1/2 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76)
```

```
  - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76)) cos(TH65)) L,
 1/2 sin(TH80) L - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) L - 1/2 (-
  ((cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
  + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) sin(TH65) + (
 1/2 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76)

  - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76)) cos(TH65)) L]
[0, 0, 0, 1/2 (- ((sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) cos(TH32)
     + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)) sin(TH43) - 1/2 (
    - (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
     + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) cos(TH43)) L (
    - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
     + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3 - 1/2 (- (
    (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)
     + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) sin(TH32)) sin(TH43) - 1/2
     (- (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
     + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) cos(TH43)) L (
    - (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
     + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) sq3, (- (
    (- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) cos(TH76)
     + (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) sin(TH76)) sin(TH65)
     + (1/2 (- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) sin(TH76)
     - 1/2 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) cos(TH76)
              2
     - 1/4 sq3 ) cos(TH65)) L (
    1/2 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76) sq3
     - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76) sq3) - (- (
    (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
     + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) sin(TH65) + (
    1/2 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76)
     - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76)) cos(TH65))
    L (1/2 (- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) sin(TH76) sq3
     - 1/2 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) cos(TH76) sq3
     + 1/4 sq3), 1/2 (- ((cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
     + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) sin(TH65) + (
    1/2 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76)
```

```
     - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76)) cos(TH65))
    L sq3, 1/2 ((- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) L + (- (
    (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
     + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) sin(TH65) + (
    1/2 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76)

     - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76)) cos(TH65))
    L) sq3, 1/2 (- sin(TH80) L
     + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) L + (- (
    (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
     + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) sin(TH65) + (
    1/2 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76)
     - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76)) cos(TH65))
    L) sq3]
[1/2 (- (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
   + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) sq3, 1/2 (
  - (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
   + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) sq3, 1/2 (
  - (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
   + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) sq3, 0,
  - 1/4 (- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) sin(TH76) sq3
   + 1/4 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) cos(TH76) sq3
   - 1/8 sq3 - 1/2 (- (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
   + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) sq3 (
  1/2 (1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) sin(TH76) sq3
   - 1/2
  (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) cos(TH76) sq3
   + 1/4), 1/4 sq3 + 1/4 (
  - (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
   + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) sq3, 1/4 sq3 + 1/4
  (- (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
   + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) sq3, 1/4 sq3 + 1/4
  (- (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
   + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) sq3]
[- 1/2 (- (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
```

```
   + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3, - 1/2 (
  - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
   + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3, - 1/2 (
  - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
   + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3, 0,
  1/4 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76) sq3


   - 1/4 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76) sq3 + 1/2 (
  - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
   + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3 (
  1/2 (1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) sin(TH76) sq3
   - 1/2
  (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) cos(TH76) sq3
   + 1/4), - 1/4 (- (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
   + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3, - 1/4 (
  - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
   + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3, - 1/4 (
  - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
   + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3]
[0, 0, 0, 0, 1/2 (- (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
     + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) sq3 (
    1/2 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76) sq3
     - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76) sq3) - 1/2 (
    - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
     + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3 (
    1/2 (- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) sin(TH76) sq3
     - 1/2 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) cos(TH76) sq3
     + 1/4 sq3), 1/4 (- (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
                                                                    2
     + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3 , 1/4 (
    - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
                                                                    2
     + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3 , 1/4 (
    - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
                                                                    2
     + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3 ]
```

C.1.2 maille d'antenne déployable à deux niveaux – fichier `maille2.gem`

```
> donnees;
COM-> lect;

                      IMPRESSION DES DONNEES DISPONIBLES

 Les valeurs des donnees initialisees sont :

   nombre de corps :   14

   numeros des corps et nombre de modes:

      corps :   1   nombre de modes :   0
      corps :   2   nombre de modes :   0
      corps :   3   nombre de modes :   0
      corps :   4   nombre de modes :   0
      corps :   5   nombre de modes :   0
      corps :   6   nombre de modes :   0
      corps :   7   nombre de modes :   0
      corps :   8   nombre de modes :   0
      corps :   9   nombre de modes :   0
      corps :   10   nombre de modes :   0
      corps :   11   nombre de modes :   0
      corps :   12   nombre de modes :   0
      corps :   13   nombre de modes :   0
      corps :   14   nombre de modes :   0

   nombre de points :   36

   numeros des points et proprietaires:

      point :   100   proprietaire :   0
      point :   101   proprietaire :   1
      point :   211   proprietaire :   1
      point :   212   proprietaire :   2
      point :   322   proprietaire :   2
      point :   323   proprietaire :   3
      point :   433   proprietaire :   3
      point :   434   proprietaire :   4
      point :   544   proprietaire :   4
      point :   545   proprietaire :   5
      point :   655   proprietaire :   5
      point :   656   proprietaire :   6
      point :   766   proprietaire :   6
      point :   767   proprietaire :   7
      point :   877   proprietaire :   7
      point :   878   proprietaire :   8
      point :   808   proprietaire :   8
      point :   800   proprietaire :   0
      point :   900   proprietaire :   0
      point :   909   proprietaire :   9
      point :   1099   proprietaire :   9
      point :   10910   proprietaire :   10
      point :   121111   proprietaire :   11
      point :   121112   proprietaire :   12
      point :   12612   proprietaire :   12
      point :   13613   proprietaire :   13
      point :   141313   proprietaire :   13
      point :   141314   proprietaire :   14
      point :   1400   proprietaire :   0
      point :   14014   proprietaire :   14
      point :   10310   proprietaire :   10
      point :   1033   proprietaire :   3
      point :   1133   proprietaire :   3
      point :   11311   proprietaire :   11
      point :   1266   proprietaire :   6
      point :   1366   proprietaire :   6

   nombre de liaisons :   18
```

```
    liaison   10   de type   rj
    entre le point   101   et le point   100

    liaison   21   de type   rj
    entre le point   211   et le point   212

    liaison   32   de type   rj
    entre le point   322   et le point   323

    liaison   43   de type   rj
    entre le point   433   et le point   434

    liaison   54   de type   rj
    entre le point   544   et le point   545

    liaison   65   de type   rj
    entre le point   655   et le point   656

    liaison   76   de type   rj
    entre le point   766   et le point   767

    liaison   87   de type   rj
    entre le point   877   et le point   878

    liaison   80   de type   rj
    entre le point   800   et le point   808

    liaison   90   de type   rj
    entre le point   900   et le point   909

    liaison   109   de type   rj
    entre le point   1099   et le point   10910

    liaison   103   de type   rj
    entre le point   1033   et le point   10310

    liaison   113   de type   rj
    entre le point   1133   et le point   11311

    liaison   1211   de type   rj
    entre le point   121111   et le point   121112

    liaison   126   de type   rj
    entre le point   12612   et le point   1266

    liaison   136   de type   rj
    entre le point   13613   et le point   1366

    liaison   1413   de type   rj
    entre le point   141313   et le point   141314

    liaison   140   de type   rj
    entre le point   1400   et le point   14014

  nombre de forces :   0

Donnees assignees pour le corps 1 :

  - masse ( 0 )
  - inertie ( 0(3, 3) )
  - distance point de reference CdG ( 0(3, 1) )
  - distance point de reference point 101 ( 0(3, 1) )
  - matrice de passage locale du point 101 ( id_(3) )
  - distance point de reference point 211 ( r0211 )
  - matrice de passage locale du point 211 ( id_(3) )

Donnees assignees pour le corps 2 :
```

```
   - masse ( 0 )
   - inertie ( 0(3, 3) )
   - distance point de reference CdG ( 0(3, 1) )
   - distance point de reference point 212 ( 0(3, 1) )
   - matrice de passage locale du point 212 ( id_(3) )
   - distance point de reference point 322 ( r0211 )
   - matrice de passage locale du point 322 ( id_(3) )

 Donnees assignees pour le corps 3 :

   - masse ( 0 )
   - inertie ( 0(3, 3) )
   - distance point de reference CdG ( 0(3, 1) )
   - distance point de reference point 323 ( 0(3, 1) )
   - matrice de passage locale du point 323 ( id_(3) )
   - distance point de reference point 433 ( 0(3, 1) )
   - matrice de passage locale du point 433 ( PL433 )
   - distance point de reference point 1033 ( r0900 )
   - matrice de passage locale du point 1033 ( id_(3) )
   - distance point de reference point 1133 ( r0900 )
   - matrice de passage locale du point 1133 ( PL433 )
...
> op(r0900);

                                    [ - R ]
                                    [     ]
                                    [  0  ]
                                    [     ]
                                    [  0  ]
...

 Donnees assignees pour le corps 4 :

   - masse ( 0 )
   - inertie ( 0(3, 3) )
   - distance point de reference CdG ( 0(3, 1) )
   - distance point de reference point 434 ( 0(3, 1) )
   - matrice de passage locale du point 434 ( id_(3) )
   - distance point de reference point 544 ( r0211 )
   - matrice de passage locale du point 544 ( id_(3) )

 Donnees assignees pour le corps 5 :

   - masse ( 0 )
   - inertie ( 0(3, 3) )
   - distance point de reference CdG ( 0(3, 1) )
   - distance point de reference point 545 ( 0(3, 1) )
   - matrice de passage locale du point 545 ( id_(3) )
   - distance point de reference point 655 ( r0211 )
   - matrice de passage locale du point 655 ( id_(3) )

 Donnees assignees pour le corps 6 :

   - masse ( 0 )
   - inertie ( 0(3, 3) )
   - distance point de reference CdG ( 0(3, 1) )
   - distance point de reference point 656 ( 0(3, 1) )
   - matrice de passage locale du point 656 ( id_(3) )
   - distance point de reference point 766 ( 0(3, 1) )
   - matrice de passage locale du point 766 ( PL433 )
   - distance point de reference point 1266 ( r0900 )
   - matrice de passage locale du point 1266 ( id_(3) )
   - distance point de reference point 1366 ( r0900 )
   - matrice de passage locale du point 1366 ( PL433 )

 Donnees assignees pour le corps 7 :

   - masse ( 0 )
   - inertie ( 0(3, 3) )
```

```
  - distance point de reference CdG ( 0(3, 1) )
  - distance point de reference point 767 ( 0(3, 1) )
  - matrice de passage locale du point 767 ( id_(3) )
  - distance point de reference point 877 ( r0211 )
  - matrice de passage locale du point 877 ( id_(3) )

Donnees assignees pour le corps 8 :

  - masse ( 0 )
  - inertie ( 0(3, 3) )
  - distance point de reference CdG ( 0(3, 1) )
  - distance point de reference point 878 ( 0(3, 1) )
  - matrice de passage locale du point 878 ( id_(3) )
  - distance point de reference point 808 ( r0211 )
  - matrice de passage locale du point 808 ( id_(3) )

Donnees assignees pour le corps 9 :

  - masse ( 0 )
  - inertie ( 0(3, 3) )
  - distance point de reference CdG ( 0(3, 1) )
  - distance point de reference point 909 ( 0(3, 1) )
  - matrice de passage locale du point 909 ( id_(3) )
  - distance point de reference point 1099 ( r0211 )
  - matrice de passage locale du point 1099 ( id_(3) )

Donnees assignees pour le corps 10 :

  - masse ( 0 )
  - inertie ( 0(3, 3) )
  - distance point de reference CdG ( 0(3, 1) )
  - distance point de reference point 10910 ( 0(3, 1) )
  - matrice de passage locale du point 10910 ( id_(3) )
  - distance point de reference point 10310 ( r0211 )
  - matrice de passage locale du point 10310 ( id_(3) )

Donnees assignees pour le corps 11 :

  - masse ( 0 )
  - inertie ( 0(3, 3) )
  - distance point de reference CdG ( 0(3, 1) )
  - distance point de reference point 121111 ( r0211 )
  - matrice de passage locale du point 121111 ( id_(3) )
  - distance point de reference point 11311 ( 0(3, 1) )
  - matrice de passage locale du point 11311 ( id_(3) )

Donnees assignees pour le corps 12 :

  - masse ( 0 )
  - inertie ( 0(3, 3) )
  - distance point de reference CdG ( 0(3, 1) )
  - distance point de reference point 121112 ( 0(3, 1) )
  - matrice de passage locale du point 121112 ( id_(3) )
  - distance point de reference point 12612 ( r0211 )
  - matrice de passage locale du point 12612 ( id_(3) )

Donnees assignees pour le corps 13 :

  - masse ( 0 )
  - inertie ( 0(3, 3) )
  - distance point de reference CdG ( 0(3, 1) )
  - distance point de reference point 13613 ( 0(3, 1) )
  - matrice de passage locale du point 13613 ( id_(3) )
  - distance point de reference point 141313 ( r0211 )
  - matrice de passage locale du point 141313 ( id_(3) )

Donnees assignees pour le corps 14 :

  - masse ( 0 )
  - inertie ( 0(3, 3) )
```

```
   - distance point de reference CdG ( 0(3, 1) )
   - distance point de reference point 141314 ( 0(3, 1) )
   - matrice de passage locale du point 141314 ( id_(3) )
   - distance point de reference point 14014 ( r0211 )
   - matrice de passage locale du point 14014 ( id_(3) )

 Donnees assignees pour le corps 0 :

   - gravite ( 0(3, 1) )
   - distance point de reference point 100 ( 0(3, 1) )
   - matrice de passage locale du point 100 ( id_(3) )
   - distance point de reference point 800 ( 0(3, 1) )
   - matrice de passage locale du point 800 ( PL800 )
   - distance point de reference point 900 ( r0900 )
   - matrice de passage locale du point 900 ( id_(3) )
   - distance point de reference point 1400 ( r0900 )
   - matrice de passage locale du point 1400 ( PL800 )

...

> evalmat(matcontb2);

[- (- ((sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) cos(TH32)

  + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)) sin(TH43) - 1/2 (

 - (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)

  + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) cos(TH43)) L

  - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) L - cos(TH10) L, - (- (

 (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) cos(TH32)

  + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)) sin(TH43) - 1/2 (

 - (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)

  + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) cos(TH43)) L

  - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) L, - (- (

 (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) cos(TH32)

  + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)) sin(TH43) - 1/2 (

 - (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)

  + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) cos(TH43)) L, - 1/4

    2
 sq3  cos(TH43) L (- (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)

  + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) + 1/2 (- (

 (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) cos(TH32)

  + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)) sin(TH43) - 1/2 (

 - (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)

  + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) cos(TH43)) L, (- (

 (1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) cos(TH76)

  + (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) sin(TH76))

 sin(TH65) + (

 1/2 (1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) sin(TH76)
```

```
 - 1/2 (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) cos(TH76)


 - 1/4 sq3) cos(TH65)) L (
1/2 (- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) sin(TH76) sq3
 - 1/2 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) cos(TH76) sq3
 + 1/4 sq3) - (- (
(- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) cos(TH76)
 + (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) sin(TH76)) sin(TH65) +
(1/2 (- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) sin(TH76)
 - 1/2 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) cos(TH76)
          2
 - 1/4 sq3 ) cos(TH65)) L (
1/2 (1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) sin(TH76) sq3
 - 1/2
(- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) cos(TH76) sq3
 + 1/4), - 1/2 (- (
(1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) cos(TH76)
 + (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) sin(TH76))
sin(TH65) + (
1/2 (1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) sin(TH76)
 - 1/2 (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) cos(TH76)
 - 1/4 sq3) cos(TH65)) L sq3 + 1/2 (- (
(- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) cos(TH76)
 + (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) sin(TH76)) sin(TH65) +
(1/2 (- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) sin(TH76)
 - 1/2 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) cos(TH76)
          2
 - 1/4 sq3 ) cos(TH65)) L, 1/2 (
- (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) L - (- (
(1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) cos(TH76)
 + (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) sin(TH76))
sin(TH65) + (
1/2 (1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) sin(TH76)
 - 1/2 (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) cos(TH76)
 - 1/4 sq3) cos(TH65)) L) sq3
 + 1/2 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) L + 1/2 (- (
```

```
 (- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) cos(TH76)
  + (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) sin(TH76)) sin(TH65) +

 (1/2 (- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) sin(TH76)
  - 1/2 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) cos(TH76)
           2
  - 1/4 sq3 ) cos(TH65)) L, 1/2 (- 1/2 sq3 cos(TH80) L
  - (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) L - (- (
 (1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) cos(TH76)
  + (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) sin(TH76))
 sin(TH65) + (
 1/2 (1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) sin(TH76)
  - 1/2 (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) cos(TH76)
  - 1/4 sq3) cos(TH65)) L) sq3 - 1/4 cos(TH80) L
  + 1/2 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) L + 1/2 (- (
 (- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) cos(TH76)
  + (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) sin(TH76)) sin(TH65) +
 (1/2 (- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) sin(TH76)
  - 1/2 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) cos(TH76)
           2
  - 1/4 sq3 ) cos(TH65)) L, 0, 0, 0, 0, 0, 0]
[(- ((cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)
   + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) sin(TH32)) sin(TH43) - 1/2 (
 - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
   + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) cos(TH43)) L
   + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) L - sin(TH10) L, (- (
 (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)
   + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) sin(TH32)) sin(TH43) - 1/2 (
 - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
   + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) cos(TH43)) L
   + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) L, (- (
 (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)
   + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) sin(TH32)) sin(TH43) - 1/2 (
 - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
   + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) cos(TH43)) L, 1/4
```

```
   2
sq3  cos(TH43) L (- (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
 + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) - 1/2 (- (
(cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)
 + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) sin(TH32)) sin(TH43) - 1/2 (
- (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
 + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) cos(TH43)) L, - (-
 ((1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) cos(TH76)
 + (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) sin(TH76))
sin(TH65) + (
1/2 (1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) sin(TH76)
 - 1/2 (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) cos(TH76)
 - 1/4 sq3) cos(TH65)) L (
1/2 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76) sq3
 - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76) sq3) + (- (
(cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
 + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) sin(TH65) + (
1/2 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76)
 - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76)) cos(TH65)) L (
1/2 (1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) sin(TH76) sq3
 - 1/2
(- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) cos(TH76) sq3
 + 1/4), - 1/2 (- ((cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
 + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) sin(TH65) + (
1/2 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76)
 - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76)) cos(TH65)) L,
- 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) L - 1/2 (- (
(cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
 + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) sin(TH65) + (
1/2 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76)
 - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76)) cos(TH65)) L,
1/2 sin(TH80) L - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) L - 1/2 (-
 ((cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
 + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) sin(TH65) + (
1/2 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76)
 - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76)) cos(TH65)) L,
```

```
 0, 0, 0, 0, 0, 0]
[0, 0, 0, 1/2 (- ((sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) cos(TH32)
     + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)) sin(TH43) - 1/2 (
    - (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
     + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) cos(TH43)) L (
    - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
     + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3 - 1/2 (- (
    (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)
     + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) sin(TH32)) sin(TH43) - 1/2
     (- (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
     + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) cos(TH43)) L (
    - (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
     + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) sq3, (- (
    (- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) cos(TH76)
     + (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) sin(TH76)) sin(TH65)
     + (1/2 (- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) sin(TH76)
     - 1/2 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) cos(TH76)
              2
     - 1/4 sq3 ) cos(TH65)) L (
    1/2 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76) sq3
     - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76) sq3) - (- (
    (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
     + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) sin(TH65) + (
    1/2 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76)
     - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76)) cos(TH65))
    L (1/2 (- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) sin(TH76) sq3
     - 1/2 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) cos(TH76) sq3
     + 1/4 sq3), 1/2 (- ((cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
     + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) sin(TH65) + (
    1/2 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76)
     - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76)) cos(TH65))
    L sq3, 1/2 ((- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) L + (- (
    (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
     + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) sin(TH65) + (
    1/2 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76)
```

```
     - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76)) cos(TH65))
    L) sq3, 1/2 (- sin(TH80) L
     + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) L + (- (
    (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
     + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) sin(TH65) + (
    1/2 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76)
     - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76)) cos(TH65))
    L) sq3, 0, 0, 0, 0, 0, 0]
[1/2 (- (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
   + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) sq3, 1/2 (
  - (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
   + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) sq3, 1/2 (
  - (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
   + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) sq3, 0,
  - 1/4 (- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) sin(TH76) sq3
   + 1/4 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) cos(TH76) sq3
   - 1/8 sq3 - 1/2 (- (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
   + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) sq3 (
  1/2 (1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) sin(TH76) sq3
   - 1/2
  (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) cos(TH76) sq3
   + 1/4), 1/4 sq3 + 1/4 (
  - (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
   + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) sq3, 1/4 sq3 + 1/4
  (- (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
   + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) sq3, 1/4 sq3 + 1/4
  (- (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
   + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) sq3, 0, 0, 0, 0, 0
  , 0]
[- 1/2 (- (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
   + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3, - 1/2 (
  - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
   + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3, - 1/2 (
  - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
   + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3, 0,
```

```
  1/4 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76) sq3
   - 1/4 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76) sq3 + 1/2 (
  - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
   + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3 (
  1/2 (1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) sin(TH76) sq3
   - 1/2
  (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) cos(TH76) sq3
   + 1/4), - 1/4 (- (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
   + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3, - 1/4 (
  - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
   + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3, - 1/4 (
  - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
   + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3, 0, 0, 0, 0,
  0, 0]
[0, 0, 0, 0, 1/2 (- (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
     + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) sq3 (
    1/2 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76) sq3
     - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76) sq3) - 1/2 (
    - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
     + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3 (
    1/2 (- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) sin(TH76) sq3
     - 1/2 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) cos(TH76) sq3
     + 1/4 sq3), 1/4 (- (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
                                                                    2
     + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3 , 1/4 (
    - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
                                                                    2
     + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3 , 1/4 (
    - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
                                                                    2
     + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3 , 0, 0, 0,
    0, 0, 0]
[- (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) L - cos(TH10) L + (
    (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) cos(TH32)
     + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)) R,
    - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) L + (
    (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) cos(TH32)
```

```
     + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)) R, (
    (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) cos(TH32)
     + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)) R, 0, 0, 0, 0, 0
    , (cos(TH90) cos(TH109) - sin(TH90) sin(TH109)) L + cos(TH90) L,
    (cos(TH90) cos(TH109) - sin(TH90) sin(TH109)) L, 0, 0, 0, 0]
[(- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) L - sin(TH10) L - (
    (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)
     + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) sin(TH32)) R,
    (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) L - (
    (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)
     + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) sin(TH32)) R, - (
    (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)
     + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) sin(TH32)) R, 0, 0, 0, 0,
    0, - (- cos(TH90) sin(TH109) - sin(TH90) cos(TH109)) L + sin(TH90) L,
    - (- cos(TH90) sin(TH109) - sin(TH90) cos(TH109)) L, 0, 0, 0, 0]
                [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
                [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
                [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
                [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[- (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) L - cos(TH10) L + (
 (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) cos(TH32)
  + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)) R - (- (
 (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) cos(TH32)
  + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)) sin(TH113) - 1/2 (
 - (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
  + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) cos(TH113)) L,
 - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) L + (
 (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) cos(TH32)
  + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)) R - (- (
 (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) cos(TH32)
  + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)) sin(TH113) - 1/2 (
 - (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
  + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) cos(TH113)) L, (
 (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) cos(TH32)
  + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)) R - (- (
```

```
(sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) cos(TH32)
 + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)) sin(TH113) - 1/2 (
- (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
 + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) cos(TH113)) L, 0, 0
, - 1/2 ((
(1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) cos(TH76)
 + (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) sin(TH76)) R
 + (- ((1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) cos(TH76)
 + (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) sin(TH76))
sin(TH126) + (
1/2 (1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) sin(TH76)
 - 1/2 (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) cos(TH76)
 - 1/4 sq3) cos(TH126)) L) sq3 + 1/2 (
(- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) cos(TH76)
 + (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) sin(TH76)) R + 1/2 (- (
(- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) cos(TH76)
 + (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) sin(TH76)) sin(TH126)
 + (1/2 (- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) sin(TH76)
 - 1/2 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) cos(TH76)
          2
 - 1/4 sq3 ) cos(TH126)) L, - 1/2 ((
(1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) cos(TH76)
 + (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) sin(TH76)) R
 + (- ((1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) cos(TH76)
 + (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) sin(TH76))
sin(TH126) + (
1/2 (1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) sin(TH76)
 - 1/2 (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) cos(TH76)
 - 1/4 sq3) cos(TH126)) L
 + (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) L) sq3 + 1/2
((- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) cos(TH76)
 + (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) sin(TH76)) R + 1/2 (- (
(- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) cos(TH76)
 + (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) sin(TH76)) sin(TH126)
 + (1/2 (- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) sin(TH76)
```

```
 - 1/2 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) cos(TH76)
          2
 - 1/4 sq3 ) cos(TH126)) L
 + 1/2 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) L, 1/2 (
- 1/2 sq3 cos(TH80) L
 - (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) L - (- (
(1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) cos(TH76)
 + (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) sin(TH76))
sin(TH126) + (
1/2 (1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) sin(TH76)
 - 1/2 (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) cos(TH76)
 - 1/4 sq3) cos(TH126)) L - (
(1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) cos(TH76)
 + (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) sin(TH76)) R)
 sq3 - 1/4 cos(TH80) L
 + 1/2 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) L + 1/2 (- (
(- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) cos(TH76)
 + (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) sin(TH76)) sin(TH126)
 + (1/2 (- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) sin(TH76)
 - 1/2 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) cos(TH76)
          2
 - 1/4 sq3 ) cos(TH126)) L + 1/2 (
(- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) cos(TH76)
 + (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) sin(TH76)) R, 0, 0,
         2
- 1/4 sq3  cos(TH113) L (
- (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
 + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) + 1/2 (- (
(sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) cos(TH32)
 + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)) sin(TH113) - 1/2 (
- (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
 + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) cos(TH113)) L, (- (
(1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) cos(TH76)
 + (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) sin(TH76))
sin(TH126) + (
1/2 (1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) sin(TH76)
 - 1/2 (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) cos(TH76)
```

```
 - 1/4 sq3) cos(TH126)) L (
1/2 (- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) sin(TH76) sq3
 - 1/2 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) cos(TH76) sq3
 + 1/4 sq3) - (- (
(- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) cos(TH76)
 + (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) sin(TH76)) sin(TH126)
 + (1/2 (- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) sin(TH76)
 - 1/2 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) cos(TH76)
          2
 - 1/4 sq3 ) cos(TH126)) L (
1/2 (1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) sin(TH76) sq3
 - 1/2
(- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) cos(TH76) sq3
 + 1/4), 0, 0]
[(- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) L - sin(TH10) L - (
(cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)
 + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) sin(TH32)) R + (- (
(cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)
 + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) sin(TH32)) sin(TH113) - 1/2 (
- (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
 + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) cos(TH113)) L,
(- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) L - (
(cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)
 + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) sin(TH32)) R + (- (
(cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)
 + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) sin(TH32)) sin(TH113) - 1/2 (
- (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
 + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) cos(TH113)) L, - (
(cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)
 + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) sin(TH32)) R + (- (
(cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)
 + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) sin(TH32)) sin(TH113) - 1/2 (
- (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
 + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) cos(TH113)) L, 0,
0, - 1/2 ((cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
```

```
 + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) R - 1/2 (- (
(cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
 + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) sin(TH126) + (
1/2 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76)
 - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76)) cos(TH126)) L
, - 1/2 ((cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
 + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) R - 1/2 (- (
(cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
 + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) sin(TH126) + (
1/2 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76)
 - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76)) cos(TH126)) L
 - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) L, 1/2 sin(TH80) L
 - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) L - 1/2 (- (
(cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
 + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) sin(TH126) + (
1/2 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76)
 - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76)) cos(TH126)) L
 - 1/2 ((cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
                                                                          2
 + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) R, 0, 0, 1/4 sq3
cos(TH113) L (- (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
 + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) - 1/2 (- (
(cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)
 + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) sin(TH32)) sin(TH113) - 1/2 (
- (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
 + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) cos(TH113)) L, - (
- ((1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) cos(TH76)
 + (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) sin(TH76))
sin(TH126) + (
1/2 (1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) sin(TH76)
 - 1/2 (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) cos(TH76)
 - 1/4 sq3) cos(TH126)) L (
1/2 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76) sq3
 - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76) sq3) + (- (
(cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
 + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) sin(TH126) + (
```

```
 1/2 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76)
  - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76)) cos(TH126)) L
 (1/2 (1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) sin(TH76) sq3
  - 1/2
 (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) cos(TH76) sq3
  + 1/4), 0, 0]
[0, 0, 0, 0, 0, - 1/2 (- ((cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
     + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) R - (- (
    (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
     + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) sin(TH126) + (
    1/2 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76)
     - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76)) cos(TH126))
     L) sq3, - 1/2 (- ((cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
     + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) R - (- (
    (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
     + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) sin(TH126) + (
    1/2 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76)
     - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76)) cos(TH126))
     L - (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) L) sq3, 1/2 (
    - sin(TH80) L + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) L + (- (
    (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
     + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) sin(TH126) + (
    1/2 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76)
     - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76)) cos(TH126))
     L + ((cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
     + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) R) sq3, 0, 0,
    1/2 (- ((sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) cos(TH32)
     + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)) sin(TH113) - 1/2
    (- (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
     + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) cos(TH113)) L (
    - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
     + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3 - 1/2 (- (
    (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)
     + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) sin(TH32)) sin(TH113)
     - 1/2 (- (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
```

```
     + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) cos(TH113)) L (
    - (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
     + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) sq3, (- (
    (- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) cos(TH76)
     + (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) sin(TH76))
    sin(TH126) + (
    1/2 (- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) sin(TH76)
     - 1/2 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) cos(TH76)
              2
     - 1/4 sq3 ) cos(TH126)) L (
    1/2 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76) sq3
     - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76) sq3) - (- (
    (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
     + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) sin(TH126) + (
    1/2 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76)
     - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76)) cos(TH126))
     L (1/2 (- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) sin(TH76) sq3
     - 1/2 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) cos(TH76) sq3
     + 1/4 sq3), 0, 0]
[1/2 (- (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
   + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) sq3, 1/2 (
  - (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
   + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) sq3, 1/2 (
  - (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
   + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) sq3, 0, 0, 1/4 sq3
   + 1/4 (- (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
   + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) sq3, 1/4 sq3 + 1/4
  (- (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
   + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) sq3, 1/4 sq3 + 1/4
  (- (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
   + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) sq3, 0, 0, 0,
  - 1/4 (- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) sin(TH76) sq3
   + 1/4 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) cos(TH76) sq3
   - 1/8 sq3 - 1/2 (- (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
   + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) sq3 (
```

```
  1/2 (1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) sin(TH76) sq3
   - 1/2
  (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) cos(TH76) sq3
   + 1/4), 0, 0]
[- 1/2 (- (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
   + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3, - 1/2 (
  - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
   + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3, - 1/2 (
  - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
   + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3, 0, 0, - 1/4
  (- (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
   + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3, - 1/4 (
  - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
   + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3, - 1/4 (
  - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
   + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3, 0, 0, 0,
  1/4 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76) sq3
   - 1/4 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76) sq3 + 1/2 (
  - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
   + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3 (
  1/2 (1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) sin(TH76) sq3
   - 1/2
  (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) cos(TH76) sq3
   + 1/4), 0, 0]
[0, 0, 0, 0, 0, 1/4 (- (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
                                                                    2
     + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3 , 1/4 (
    - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
                                                                    2
     + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3 , 1/4 (
    - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
                                                                    2
     + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3 , 0, 0, 0,
    1/2 (- (sin(TH10) cos(TH21) + cos(TH10) sin(TH21)) sin(TH32)
     + (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) cos(TH32)) sq3 (
    1/2 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) sin(TH76) sq3
     - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) cos(TH76) sq3) - 1/2 (
```

```
    - (cos(TH10) cos(TH21) - sin(TH10) sin(TH21)) sin(TH32)
     + (- cos(TH10) sin(TH21) - sin(TH10) cos(TH21)) cos(TH32)) sq3 (
    1/2 (- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) sin(TH76) sq3
     - 1/2 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) cos(TH76) sq3
     + 1/4 sq3), 0, 0]
[0, 0, 0, 0, 0, - 1/2 (
  (1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) cos(TH76)
   + (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) sin(TH76)) R
   sq3 + 1/2 ((- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) cos(TH76)
   + (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) sin(TH76)) R, - 1/2 (
  (1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) cos(TH76)
   + (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) sin(TH76)) R
   sq3 + 1/2 ((- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) cos(TH76)
   + (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) sin(TH76)) R
   - 1/2 (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) L sq3
   + 1/2 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) L, - 1/2 (
  (1/2 sq3 sin(TH80) cos(TH87) + 1/2 sq3 cos(TH80) sin(TH87)) cos(TH76)
   + (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) sin(TH76)) R
   sq3 + 1/2 ((- 1/2 sin(TH80) cos(TH87) - 1/2 cos(TH80) sin(TH87)) cos(TH76)
   + (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) sin(TH76)) R + 1/2 (
  - 1/2 sq3 cos(TH80) L
   - (- 1/2 sq3 sin(TH80) sin(TH87) + 1/2 sq3 cos(TH80) cos(TH87)) L) sq3
   - 1/4 cos(TH80) L
   + 1/2 (1/2 sin(TH80) sin(TH87) - 1/2 cos(TH80) cos(TH87)) L, 0, 0, 0, 0,
  1/2 (- 1/2 sq3 sin(TH140) sin(TH1413) + 1/2 sq3 cos(TH140) cos(TH1413)) L sq3
   - 1/2 (1/2 sin(TH140) sin(TH1413) - 1/2 cos(TH140) cos(TH1413)) L, - 1/2 (
  - 1/2 sq3 cos(TH140) L
   - (- 1/2 sq3 sin(TH140) sin(TH1413) + 1/2 sq3 cos(TH140) cos(TH1413)) L) sq3
   + 1/4 cos(TH140) L
   - 1/2 (1/2 sin(TH140) sin(TH1413) - 1/2 cos(TH140) cos(TH1413)) L]
[0, 0, 0, 0, 0, - 1/2 ((cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
  + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) R, - 1/2 (
 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
  + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) R
  - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) L, - 1/2 (
```

```
 (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
  + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) R
  + 1/2 sin(TH80) L - 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) L, 0, 0
 , 0, 0, 1/2 (- cos(TH140) sin(TH1413) - sin(TH140) cos(TH1413)) L,
 - 1/2 sin(TH140) L + 1/2 (- cos(TH140) sin(TH1413) - sin(TH140) cos(TH1413)) L
 ]
[0, 0, 0, 0, 0, 1/2 ((cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
    + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) R sq3, 1/2 (
   (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
    + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) R sq3
    + 1/2 (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) L sq3, 1/2 (
   (cos(TH80) cos(TH87) - sin(TH80) sin(TH87)) cos(TH76)
    + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) sin(TH76)) R sq3
    + 1/2 (- sin(TH80) L + (- cos(TH80) sin(TH87) - sin(TH80) cos(TH87)) L) sq3
   , 0, 0, 0, 0,
   - 1/2 (- cos(TH140) sin(TH1413) - sin(TH140) cos(TH1413)) L sq3, - 1/2
   (- sin(TH140) L + (- cos(TH140) sin(TH1413) - sin(TH140) cos(TH1413)) L) sq3
   ]
                 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
                 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
                 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

C.2 calcul de rang : méthode probabiliste modulaire

filtre de changement de variables – fichier `trigfree`

```
trigfree:=proc(mat)
###################
local i,j,k,v;                          # indices

for i to rowdim(mat) do
   for j to coldim(mat) do
      v := [op(indets(mat[i,j],trig))]; # determine les lignes
                                        # trigonometriques presentes
                                        # dans la matrices
      for k to nops(v) do
         if op(0,v[k])=sin then mat[i,j]:=
           subs(v[k]=2*op(1,v[k])/(1+op(1,v[k])^2),mat[i,j])
         fi;                            # remplace les sinus
         if op(0,v[k])=cos then mat[i,j]:=
           subs(v[k]=(1-op(1,v[k])^2)/(1+op(1,v[k])^2),mat[i,j])
         fi;                            # remplace les cosinus
         if op(0,v[k])=tan then mat[i,j]:=
           subs(v[k]=2*op(1,v[k])/(1-op(1,v[k])^2),mat[i,j])
         fi                             # remplace les tangentes
      od
   od
od

end;
```

procedure principale – fichier `rangpm`

```
modgauss:=proc(M,p)
###################
local r,         # nombre de lignes
      c,         # nombre de colonnes
      v,         # variable intermediaire
      i,j,k,s,t; # indices
r:=rowdim(M);
c:=coldim(M);
if r<c then M:=transpose(M); v:=r; r:=c; c:=v fi;
                 # fait en sorte que la matrice a traiter ait plus de
                 # lignes que de colonnes
for i to c do    # cherche les pivots et transforme les lignes
  if not({seq(M[s,i],s=i..r)}={0})
  then
    if M[i,i]<>0
    then
      for j from i+1 to r do
        for k from i+1 to c do
          M[j,k]:=M[j,k]-M[j,i]*M[i,k]/M[i,i] mod p
        od;
        M[j,i]:=0;
      od;
    else
      for j from i+1 to r do
        if M[j,i]<>0
        then
          for t from j+1 to r do
            for k from i+1 to c do
              M[t,k]:=M[t,k]-M[t,i]*M[j,k]/M[j,i] mod p
            od;
            M[t,i]:=0
          od;
          for k from i to c do
            v:=M[i,k];
            M[i,k]:=M[j,k];
            M[j,k]:=v
          od
        fi
      od
    fi
  fi
od;
rk:=0;           # calcule le nombre de lignes non nulles
for i from 1 to r do
  if not({seq(M[i,s],s=1..c)}={0}) then rk:=rk+1 fi
od; rk
end;
```

élimination de Gauss – fichier `modgauss`

```
rangpm:=proc(M,n)                # n : nombres de calcul a effectuer
#################
local X,                         # liste des variables de M
      p,                         # le nombre premier
      ok,                        # marqueur
      r,                         # procedure de choix aleatoire
      x,                         # liste de valeurs modulaires des variables
      ss,                        # sequence de substitution
      m,                         # matrice de travail
      rk,                        # matrice des rangs obtenus
      i,j,k;                     # indices

if not type(M,matrix)
then ERROR(‘wrong number or type of arguments‘);
fi;
if not (indets(convert(M,set),trig)={0})
then trigfree(M)
fi;
X:=[op(indets(convert(M,set)))];  # etape (1)
p:=nextprime(10000);
rk:=matrix(1,n);
for k to n do
  p:=prevprime(p);               # etape (2)
  ok:=false;                     # etape (3)
  while not(ok) do
    r:=rand(0..p);
    x:=[seq(r(),i=1..nops(X))];
    denom(convert(M,set));
    ss:=seq(X[i]=x[i],i=1..nops(X));
    subs(",[op("")]);
    ok:=not(member(0,"))
  od;
  m:=matrix(rowdim(M),coldim(M)); # etape (4)
  for i to rowdim(M) do
    for j to coldim(M) do
      m[i,j]:=subs(ss,M[i,j]) mod p
    od;
  od;
  rk[1,k]:=modgauss(m,p)         # etape (5)
od;
op(rk)
end;
```

C.3 bases standards en *MAPLE*

C.3.1 bases standards d'idéaux

procédure principale – fichier `basis`

```
 #####     ##     ####      #     ####
 #    #   #  #   #          #    #
 #####   #    #   ####      #     ####
 #    #  ######       #     #         #
 #    #  #    #  #    #     #    #    #
 #####   #    #   ####      #     ####
```

```
# main procedure : compute standard basis algorithm
# F : a list of polynomials
# X : the list of variables

basis := proc(F,X)
    local nvar,HT,HC,G,B,r,i,j ;

# initializing parameters :
# nvar : number of variables
# G : increasing family of generators
# HT : list of head terms
# HC : list of head coefficients
# see file "head" for procedures "headterm" and "headcoef"

    nvar := nops(X) ;
    G := [ 'expand(F[i])' $i=1..nops(F) ] ;
    HT := [ 'headterm(G[i],X)' $i=1..nops(G) ] ;
    HC := [ 'headcoef(G[i],X)' $i=1..nops(G) ] ;

# algorithm using BETAij's method
# see files "betalist" and "rest"

    for i from 2 while i<=nops(G) do
        B := betalist(HT,i,nvar) ;
        for j from 1 to nops(B) do
            r := rest(Xexp(B[j],X)*G[i],G,HT,HC,X) ;
            if r<>0 then
                       G := [ op(G),r ] ;
                HT := [ op(HT),headterm(r,X) ] ;
                HC := [ op(HC),headcoef(r,X) ] ;
                fi
        od
    od ;
    G
end ;
```

procédure de division – fichier `rest`

```
 #####   ######   ####    #####
 #    #  #       #          #
 #    #  #####    ####      #
 #####   #            #     #
 #   #   #       #    #     #
 #    #  ######   ####      #

# REST
# divide a polynomial by a family of polynomials

rest := proc(f,G,HT,HC,X)
    local rp,r,dum,flag,pt,pc,j,q,m1,m2 ;
    if f=0 then RETURN (0) fi ;
    rp := f ;
    r := 0 ;
    for dum while rp<>0 do
        flag := true ;
        pt := headterm(rp,X) ;
        pc := headcoef(rp,X) ;
        for j to nops(G) while flag do
            if divide(Xexp(pt,X),Xexp(HT[j],X),'q') then
                flag := false ;
                igcd(HC[j],pc) ;
                divide(HC[j],",'m1') ;
                divide(pc,"",'m2') ;
                (m1*rp) ;
                rp := expand ((-m2)*q*G[j]) ;
                rp := expand(rp+"")
            fi
        od ;
        if flag then
            rp := expand(rp-Xexp(pt,X)*pc) ;
            r := expand(r+pc*Xexp(pt,X))
        fi
    od ;
    r
end ;

# X-EXP
# convert the vector representation of a monomial in an expression

Xexp := proc(m,X)
     local e,i ;
     option remember ;
     e := 1 ;
     for i from 1 to nops(X) do e := e*X[i]^m[i] od ;
     e
end ;
```

procédure d'élimination des paires critiques – fichier `betalist`

```
 #####   ######   #####    ##    #          #     ####    #####
 #    #  #          #     #  #   #          #    #          #
 #####   #####      #    #    #  #          #     ####      #
 #    #  #          #    ######  #          #         #     #
 #    #  #          #    #    #  #          #    #    #     #
 #####   ######     #    #    #  ######     #     ####      #


# compute a minimal list of beta[i,j] for a given i
# see files "calculbeta" and "compar"

betalist := proc(M,i,nvar)
    local B,B1,Bij,flag,j,k ;
    B := [calculbeta(M,i,1,nvar)] ;
    for j from 2 to i-1 do
        B1 := NULL ;
        Bij := calculbeta(M,i,j,nvar) ;
        flag := true ;
        for k from 1 to nops(B) do
            if compar(Bij,B[k],nvar)=false then B1 := B1,B[k]
            elif compar(Bij,B[k],nvar)=B[k] then flag:= false ; break
            fi
        od ;
        if flag then B := [B1,Bij] fi
    od ;
    B
end ;
```

procédure de calculs sur les monômes – fichiers `head` et `compar`

```
 #    #  ######    ##    #####
 #    #  #        #  #   #    #
 ######  #####   #    #  #    #
 #    #  #       ######  #    #
 #    #  #       #    #  #    #
 #    #  ######  #    #  #####
```

```
# HEADTERM
# give head term of a polynomial (see below)

headterm := proc(p,X)
    op(1,head(p,X))
end ;

# HEADCOEF
# give head coefficient of a polynomial (see below)

headcoef := proc(p,X)
    op(2,head(p,X))
end ;

# HEAD
# compute head of a polynomial

head := proc(p,X)
    local q,h,td,t ;
    option remember ;

# separe terms which have highest total degree

    q := expand(p) ;
    h := 0 ;
    td := degree(q,{op(X)}) ;
    if type(q,`+`) then
        for i to nops(q) do
            op(i,q) ;
            if degree(",{op(X)})=td then h := h+" fi
        od
    else  h := q
    fi ;

# separe in last expression term of highest degree using lexicographic
# order on variables

    t := [] ;
    for i from 1 to nops(X) do
        t := [op(t),degree(h,X[i])] ;
```

```
        if t[i]<>0 then h := lcoeff(h,X[i]) fi
    od ;
    [t,h]
end ;
```

```
  ####    ####   #    #  #####     ##    #####
 #    #  #    #  ##  ##  #    #   #  #   #    #
 #       #    #  # ## #  #    #  #    #  #    #
 #       #    #  #    #  #####   ######  #####
 #    #  #    #  #    #  #       #    #  #   #
  ####    ####   #    #  #       #    #  #    #
```

```
# compare two monomials in order to know which one divide the other
# return false in the other cases

compar := proc(m1,m2,nvar)
    local c1,c2,c,rep ;
    c1 := 0 ;
    c2 := 0 ;
    for i from 1 to nvar do
        if m1[i]<m2[i] then c1 := c1+1
        elif m1[i]>m2[i] then c2 := c2+1
        fi
    od ;
    c := c1*c2 ;
    if c<>0 then rep := false elif c1>c2 then rep := m1 else rep := m2 fi ;
    rep
end ;
```

C.3.2 bases standards de sous-modules

procédure principale – fichier `basis`

```
 #####     ##     ####      #     ####
 #    #   #  #   #          #    #
 #####   #    #   ####      #     ####
 #    #  ######       #     #         #
 #    #  #    #  #    #     #    #    #
 #####   #    #   ####      #     ####
```

```
# Procedure principale :  calcule la base standard et les relations entre les
# vecteurs de la liste F.
# retourne une liste [Base, Rel] ou Base est la liste des vecteurs de la base
# standard est Rel celle des relations entre les vecteurs initiaux.
# F : listes des vecteurs (definis comme "array" Maple, tous de la meme
# dimension, dont les composantes sont des polynomes en les variables de
# la liste X. F est "un peu modifie" i.e. ses composantes sont expansees.
# X : la liste des variables.

basis := proc(F,X)
  local t0, nvar, nbvec, nbveci, G, HT, nbrel, rel, i, B, nbeta, j, r, q, k,
        coord, tmp;

  t0 := time();         # le temps CPU consommme
  nvar := nops(X) ;         # le nombre de variables
  nbveci := nops(F);        # le nombre de vecteurs au debut
  nbvec := nbveci;        # le nombre de vecteurs (va croitre)

  rel := table();         # Les relations
  nbrel := 0;               # le nombre de relations
  coord := table();         # La table des "coordonnees" des vecteurs rajoutes
                        # en fonction des nveci premiers.

  # On expanse les composantes et on recupere les monomes de tete
  for i to nbvec do
    G[i] := Expand(F[i]);        # modifie les F[i]
    HT[i] := Head(G[i], X);
  od;

  # l'algorithme utilisant les "Beta i j" :

  for i from 2 while i<=nbvec do

    B := BetaList(HT,i,nvar,'nbeta');         # Betaij associes au i-eme vecteur

    for j from 1 to nbeta do
      xbeta := BetaToM(B[j], X);
      r := Divide(Xexp(xbeta, G[i]), G, HT, nbvec, X, 'q');
```

```
      if IsNull(r) then         # On a trouve une nouvelle relation
              nbrel := nbrel + 1;
         UpdateRel(nbrel, coord, nbvec, nbveci, rel, q, xbeta, i);
      else                          # On a trouve un nouveau vecteur ou on a fini
        if IsConstant(r) then
           print(`total time : `,time()-t0);
           RETURN([1]) # a changer peut-etre ?
        else
          nbvec := nbvec + 1;
          G[nbvec] := copy(r);
          HT[nbvec] := Head(r,X);
          # On prend les "coordonnees" du nouveau vecteur ...
          UpdateCoord(coord, nbvec, nbveci, q, xbeta, i);
        fi
      fi
    od;

  od;

# On pourrait obtenir ici une base standard minimale ... mais il faudrait
# remettre les relations dans le bon ordre. Plus tard peut-etre ...

  print(`total time : `,time()-t0);

  RETURN([ConvertToList(G, nbvec), ConvertToList(rel, nbrel)]);

end;

# Pour multiplier un vecteur de polynomes par un monome
# Ne modifie pas son argument.
Xexp := proc(xbeta, V)
   local i, C;
   C := copy(V);
   for i from 1 to vectdim(V) do
       C[i] := expand(xbeta*C[i]);
   od;
   RETURN(C);
end;
```

procédure de division – fichier `rest`

```
 #####   ######   ####    #####
 #    #  #       #          #
 #    #  #####    ####      #
 #####   #            #     #
 #   #   #       #    #     #
 #    #  ######   ####      #
```

```
# La procedure de division (rend le reste)
# On divise f (vecteur de polynome) par la liste G, HT est la liste des
# monomes de tete de G, X la liste des variables, nvec le nombre de vecteurs
# dans G et q est le quotient qu'on retourne
# Attention, f est modifie !

Divide := proc(f, G, HT, nvec, X, q)

  local r, flag, rm, i, j, quo, pos, mo, fv;

  r := NullVect(vectdim(f));         # le reste
  for i to nvec do
      q[i] := 0;
  od;

  while not IsNull(f) do
    flag := true; rm := Head(f, X);
    for j to nvec while flag do
        if CanDivide(HT[j], rm, 'quo') then
           flag := false;
           StripQuotient(f, mtopoly(quo, X), G[j], q, j);
        fi
    od;
    if flag then
       StripRest(f, mtopoly(rm, X), mpos(rm), r);
    fi
  od;

  RETURN(r)

end;

# Deux fonctions "auxilliaires" :

# On enleve le quotient et on met a jour les qi
StripQuotient := proc(f, quo, V, q, i0)
   local i;
   for i from 1 to vectdim(f) do
       f[i] := f[i] - expand(quo*V[i]);
   od;
```

```
   q[i0] := q[i0] + quo;
end;

# On enleve le terme de tete et on le place dans le reste
StripRest := proc(f, ht, pht, r)
   f[pht] := f[pht] - ht;
   r[pht] := r[pht] + ht;
end;
```

procédure d'élimination des paires critiques – fichier `betalist`

```
 #####   ######   #####    ##    #          #     ####    #####
 #    #  #          #     #  #   #          #    #          #
 #####   #####      #    #    #  #          #     ####      #
 #    #  #          #    ######  #          #         #     #
 #    #  #          #    #    #  #          #    #    #     #
 #####   ######     #    #    #  ######     #     ####      #
```

```
# BetaList : calcul de la table des "betas" associee au i-ieme vecteur
# de monomes de tete de la liste M.
# nvar est le nombre de variables, et nbeta est au retour le nombre de
# "betas".

BetaList := proc(M, i, nvar, nbeta)

   local j, posi, nb, nbetan, Bij, j, k, B, B1, flag;

   posi := mpos(M[i]);
   nb := 0;

   for j from 1 to i-1 do
       if mpos(M[j]) = posi then
          Bij := CalculBeta(M, i, j, nvar);
          nbetan := 1;
          flag := true;
          for k from 1 to nb do
              if compar(Bij, B[k], nvar) = 0 then
                 B1[nbetan] := B[k];
                 nbetan := nbetan + 1;
              elif compar(Bij, B[k], nvar) = 2 then
                 flag := false;
                 break;
              fi;
          od;
          if flag then
             B := B1; B[nbetan] := Bij; nb := nbetan
            fi;
       fi;
   od;
   nbeta := nb;
   RETURN(op(B));

end;



# CALCULBETA
# Pour deux monomes M[i1] et M[i2], calcule le "candidat betaij" :
```

```
# beta[i,j]=exp(ppcm(M[i1],M[i2])/M[i1])

CalculBeta := proc(M,i1,i2,nvar)
  local m1,m,i;

  m1 := NULL;
  m := NULL;
  for i to nvar do
    if mpow(M[i1],i) > mpow(M[i2],i) then
       m1 := m1,mpow(M[i1],i);
    else
       m1 := m1,mpow(M[i2],i);
    fi
  od;
  for i to nvar do m := m,m1[i]-mpow(M[i1],i) od;
  RETURN([m])
end;


# compar :
# retourne 1 si m1 divise m2 | 2 si m2 divise m1 | 0 sinon.

compar := proc(m1,m2,nvar)
  local c1,c2,i,c,rep;
  option remember;
  c1 := 0;
  c2 := 0;
  for i from 1 to nvar do
    if m1[i]<m2[i] then c1 := c1+1
    elif  m1[i]>m2[i] then c2 := c2+1  fi
  od;
  c := c1*c2;
  if c<>0 then rep := 0 elif c1>c2 then rep := 1 else rep := 2 fi;
  rep
end;

# BetaToM : transforme une liste de puissance en monome
BetaToM := proc(p, vars)
   local tmp;
   tmp := 1;
   for i from 1 to nops(vars) do
       tmp := tmp * vars[i]^p[i]
   od;
   RETURN(tmp);
end;
```

procédure de calcul du monôme de tête – fichier `head`

```
 #    #  ######    ##    #####
 #    #  #        #  #   #    #
 ######  #####   #    #  #    #
 #    #  #       ######  #    #
 #    #  #       #    #  #    #
 #    #  ######  #    #  #####

Head := proc(v,X)
  local i, d;
  d := vectdim(v);
  for i to d - 1 do
     if v[i] <> 0 then
        RETURN([i, HeadPol(v[i], X)]);
     fi
  od;
  [d, HeadPol(v[d], X)]
end;

# Calcule le terme de tete d'un polynome.
# 4*x^2*y^3 -> 4, [2,3]

HeadPol := proc(p,X)
  local q,h,td,i,t;
  option remember;

# separe les termes de plus haut degre total.

  q := expand(p) ; h := 0 ; td := degree(q,{op(X)});
  if type(q,`+`) then
    for i to nops(q) do
      op(i,q);
      if degree(",{op(X)})=td then h := h+" fi
    od
  else h := q
  fi;

# On utilise l'ordre lexicographique sur les variables

  t := [];
  for i from 1 to nops(X) do
    t := [op(t),degree(h,X[i])];
    if t[i]<>0 then h := lcoeff(h,X[i]) fi
  od;
  h, t
end;
```

C.4 exemples de mécanisme bouclés – session *MACAULAY*

exemple1 : le mécanisme le plus simple – fichier `ex1.mac`

```
                        Macaulay
       A computer algebra system for algebraic geometry

    This program, Macaulay, may be freely copied for others.  We
request that you write us to join a mailing list of Macaulay users,
so that we can keep you informed of updates to Macaulay.

    Dave Bayer                  Mike Stillman
    Department of Mathematics   Department of Mathematics
    Barnard College             Cornell University
    New York, NY 10027          Ithaca, NY 14853
    (212)854-2643, 864-4235     (607)255-7240, 277-1835
    dab@math.columbia.edu       mike@mssun7.msi.cornell.edu

Macaulay version 3.0, created 8/14/89

% ring q
! characteristic (if not 31991)       ?
! number of variables                 ? 9
!   9 variables, please                ? l[1]s[1]c[1]l[2]s[2]c[2]l[3]s[3]c[3]
! variable weights (if not all 1)     ?
! monomial order (if not rev. lex.)   ?
;   largest degree of a monomial        : 43

% ideal i
! number of generators ? 2
! (1,1) ? l[1]c[1]+l[2]c[2]+l[3]c[3]
! (1,2) ? -l[1]s[1]-l[2]s[2]-l[3]s[3]

% std i i_std
; 23.4.
; computation complete after degree 4

% codim is
; codimension : 2

% qring i_std Q

% setring Q

% pring
; current ring is Q
; characteristic           : 31991
; number of variables      : 9
; variables                : l[1]s[1]c[1]l[2]s[2]c[2]l[3]s[3]c[3]
; weights                  : 1 1 1 1 1 1 1 1 1
; monomial order           : 9 c
; top degree of a monomial : 43
; quotient ring by ideal:
; l[1]s[1]+l[2]s[2]+l[3]s[3]
; l[1]c[1]+l[2]c[2]+l[3]c[3]
; c[1]l[2]s[2]-s[1]l[2]c[2]+c[1]l[3]s[3]-s[1]l[3]c[3]

% mat m
! number of rows    ? 2
! number of columns ? 5
! (1,1) ? 0
! (2,1) ? 0
! (1,2) ? 0
! (2,2) ? 0
! (1,3) ? l[1]s[1]
! (2,3) ? -l[1]c[1]
! (1,4) ? l[2]s[2]
! (2,4) ? -l[2]c[2]
! (1,5) ? l[3]s[3]
```

```
! (2,5) ? -l[3]c[3]

% res m m_res
; -1..0..1..2..3..4..5..
; computation complete after degree 5

% pres m_res
;
; ----------------------------------

; 0 0 -l[2]s[2]-l[3]s[3] l[2]s[2]  l[3]s[3]
; 0 0 l[2]c[2]+l[3]c[3]  -l[2]c[2] -l[3]c[3]
;
; ----------------------------------

; 1 0 0
; 0 1 0
; 0 0 1
; 0 0 1
; 0 0 1
;
; ----------------------------------
```

exemple 2 : la bielle – fichiers `ex2_std.mac` et `ex2_trig.mac`

sans relation trigonométrique – fichier `ex2_std.mac`

```
% ring q
! characteristic (if not 31991)       ?
! number of variables                 ? 8
!   8 variables, please                ? txs[1]c[1]l[1]s[2]c[2]l[2]
! variable weights (if not all 1)     ?
! monomial order (if not rev. lex.)   ?
;   largest degree of a monomial        : 55

% ideal i
! number of generators ? 2
! (1,1) ? l[1]c[1]+l[2]c[1]c[2]-l[2]s[1]s[2]
! (1,2) ? l[1]s[1]+l[2]s[1]c[2]+l[2]c[1]s[2]

% ideal j
! number of generators ? 4
! (1,1) ? l[1]c[1]+l[2]c[1]c[2]-l[2]s[1]s[2]
! (1,2) ? l[1]s[1]+l[2]s[1]c[2]+l[2]c[1]s[2]
! (1,3) ? c[1]2+s[1]2-1
! (1,4) ? c[2]2+s[2]2-1

% homog j t jh

% std jh js
; 23.4.5.6.7.8.9.10.
; computation complete after degree 10

% codim js
; codimension : 3
% mat m
! number of rows    ? 2
! number of columns ? 3
! (1,1) ? -l[1]s[1]-l[2]s[1]c[2]-l[2]c[1]s[2]
! (2,1) ? l[1]c[1]+l[2]c[1]c[2]-l[2]s[1]s[2]
! (1,2) ? -l[2]s[1]c[2]-l[2]c[1]s[2]
! (2,2) ? l[2]c[1]c[2]-l[2]s[1]s[2]
! (1,3) ? -1
! (2,3) ? 0

% homog m t mh

% type mh
; -ts[1]l[1]-c[1]s[2]l[2]-s[1]c[2]l[2] -c[1]s[2]l[2]-s[1]c[2]l[2] -1
; tc[1]l[1]-s[1]s[2]l[2]+c[1]c[2]l[2]  -s[1]s[2]l[2]+c[1]c[2]l[2] 0

% res mh mh_res
; -1.0.1.2.3.4.5..
; computation complete after degree 5

% numinfo mh_res
; name    #rows #cols #standard  degrees
; 1       2     3     3          3 3 0
; 2       3     1     1          6

% pres mh_res.2
;
; ---------------------------------

; -s[1]s[2]l[2]+c[1]c[2]l[2]
; -tc[1]l[1]+s[1]s[2]l[2]-c[1]c[2]l[2]
; ts[1]2l[1]s[2]l[2]+tc[1]2l[1]s[2]l[2]
;
; ---------------------------------
```

avec relation trigonométrique – fichier `ex2_trig.mac`

```
% ring q
! characteristic (if not 31991)       ?
! number of variables                 ? 8
!   8 variables, please                ? tx
!   6 more variables, please           ? s[1]c[1]l[1]
!   3 more variables, please           ? s[2]c[2]l[2]
! variable weights (if not all 1)     ?
! monomial order (if not rev. lex.)   ?
;   largest degree of a monomial        : 55

% ideal i
! number of generators ? 1
! (1,1) ? s[1]2+c[1]2-t2

% std i is
; 23.
; computation complete after degree 3

% qring is qi

% setring qi

% pring qi
; ring qi
; characteristic           : 31991
; number of variables      : 8
; variables                : xs[1]c[1]l[1]s[2]c[2]l[2]t
; weights                  : 1 1 1 1 1 1 1 1
; monomial order           : 8 c
; top degree of a monomial : 55
; quotient ring by ideal:
; s[1]2+c[1]2-t2

% mat m
! number of rows    ? 2
! number of columns ? 3
! (1,1) ? -l[1]s[1]-l[2]s[1]c[2]-l[2]c[1]s[2]
! (2,1) ?  l[1]c[1]+l[2]c[1]c[2]-l[2]s[1]s[2]
! (1,2) ? -l[2]s[1]c[2]-l[2]c[1]s[2]
! (2,2) ? l[2]c[1]c[2]-l[2]s[1]s[2]
! (1,3) ? -1
! (2,3) ? 0

% homog m t mh

% res mh mh_res
; -1.0.1.2.3.4.5..6..
; computation complete after degree 6

% pres mh_res.2
;
; ---------------------------------

; -s[1]s[2]l[2]+c[1]c[2]l[2]
; s[1]s[2]l[2]-c[1]c[2]l[2]-c[1]l[1]t
; l[1]s[2]l[2]t3
;
; ---------------------------------
```

exemple3 : la salière – fichier `ex3.mac`

```
% ring q
! characteristic (if not 31991)       ?
! number of variables                 ? 7
!   7 variables, please                ? ts[1]c[1]s[2]c[2]s[3]c[3]
! variable weights (if not all 1)     ?
! monomial order (if not rev. lex.)   ?
;   largest degree of a monomial        : 76

% ideal i
! number of generators ? 3
! (1,1) ? c[2]-c[3]
! (1,2) ? s[1]s[2]
! (1,3) ? c[1]s[2]-s[3]t

% std i is
; 12.3.4.
; computation complete after degree 4

% codim
; codim <standard basis> [integer result]

% codim is
; codimension : 3

% ideal j js
; ideal <resulting matrix>

% ideal j
! number of generators ? 6
! (1,1) ?  c[2]-c[3]
! (1,2) ? s[1]s[2]
! (1,3) ? c[1]s[2]-s[3]t
! (1,4) ? c[1]2+s[1]2-t2
! (1,5) ? c[2]2+s[2]2-t2
! (1,6) ? c[3]2+s[3]2-t2

% std j js
; 12.3.4.5.6.
; computation complete after degree 6

% codim js
; codimension : 5

% mat m
! number of rows    ? 3
! number of columns ? 3
! (1,1) ? 0
! (2,1) ? c[1]s[2]
! (3,1) ? -s[1]s[2]
! (1,2) ? -s[2]t
! (2,2) ? s[1]c[2]
! (3,2) ? c[1]c[2]
! (1,3) ? s[3]t
! (2,3) ? 0
! (3,3) ? -c[3]t
% res mh m_res
; 1.2.
; computation complete after degree 2

% numinfo m_res
; name    #rows #cols #standard  degrees
; 1       3     3     4          2:3

% qring is qq
[126k]

% pring
; current ring is qq
```

```
; characteristic           : 31991
; number of variables      : 7
; variables                : ts[1]c[1]s[2]c[2]s[3]c[3]
; weights                  : 1 1 1 1 1 1 1
; monomial order           : 7 c
; top degree of a monomial : 76
; quotient ring by ideal:
; ts[1]s[3]
; s[1]s[2]
; c[1]s[2]-ts[3]
; c[2]-c[3]

% mat n
! number of rows    ? 3
! number of columns ? 3
! (1,1) ? 0
! (2,1) ? c[1]s[2]
! (3,1) ?  -s[1]s[2]
! (1,2) ?  -s[2]t
! (2,2) ? s[1]c[2]
! (3,2) ?  c[1]c[2]
! (1,3) ?  s[3]t
! (2,3) ? 0
! (3,3) ?  -c[3]t

% homog n t nh

% type nh
; 0     -ts[2]   ts[3]
; ts[3] s[1]c[3] 0
; 0     c[1]c[3] -tc[3]

% type mh
; 0         -ts[2]   ts[3]
; c[1]s[2]  s[1]c[2] 0
; -s[1]s[2] c[1]c[2] -tc[3]

% res nh n_res
; 1.2.......3.......4.......5.......
; computation complete after degree 5
% numinfo n_res
; name    #rows #cols #standard  degrees
; 1       3     3     7          2:3
; 2       3     2     2          3 4
; 3       2     3     4          4 5 5
; 4       3     4     5          5 6:3
; 5       4     5     7          6 7:4
; 6       5     6     8          7 8:5
; 7       6     7     10         8 9:6

% type n_res.2
; s[1] 0
; 0    0
; 0    s[1]s[3]

% setring q

% qring js qq

% mat p
! number of rows    ? 3
! number of columns ? 3
! (1,1) ? 0
! (2,1) ? c[1]s[2]
! (3,1) ?  -s[1]s[2]
! (1,2) ?  -s[2]t
! (2,2) ?  s[1]c[2]
! (3,2) ?  c[1]c[2]
! (1,3) ?  s[3]t
! (2,3) ?  0
```

```
! (3,3) ?  -c[3]t

% homog p t ph

% type ph
; 0     -ts[2]   ts[3]
; ts[3] s[1]c[3] 0
; 0     c[1]c[3] -tc[3]

% res ph p_res
; 1.2......[189k].3....[252k].[315k].[378k][441k][504k]^C

% 1% type p_res.2
; 0    s[1] ts[2]-c[1]s[3] 0              0
; s[2] 0    0              0              s[3]c[3]
; s[3] 0    0              ts[2]-c[1]s[3] s[2]c[3]
```

exemple4 : le joint homocinétique – fichier `ex4.mac`

```
% ring q
! characteristic (if not 31991)       ?
! number of variables                  ? 10
!  10 variables, please                ? tc[1]s[1]c[2]s[2]c[3]s[3]c[4]s[4]z
! variable weights (if not all 1)     ?
! monomial order (if not rev. lex.)   ?
;   largest degree of a monomial        : 36

% ideal i
! number of generators ? 2
! (1,1) ? c[2]2+s[2]2-t2
! (1,2) ? c[1]s[4]-s[1]c[4]

% std i is
; 23.4.
; computation complete after degree 4

% ideal j
! number of generators ? 8
! (1,1) ? c[1]s[2]c[3]-s[1]s[3]
! (1,2) ? s[1]s[2]c[3]-c[1]s[3]
! (1,3) ? c[2]c[3]-z
! (1,4) ? c[4]s[1]c[2]-s[1]c[4]c[2]
! (1,5) ? c[1]2+s[1]2-1
! (1,6) ? c[2]2+s[2]2-1
! (1,7) ? c[3]2+s[3]2-1
! (1,8) ? c[4]2+s[4]2-1
% homog j t jh

% std jh js
; 23.4.5.6.7.8.9.
; computation complete after degree 9

% codim js
; codimension : 6

% mat m
! number of rows    ? 4
! number of columns ? 5
! (1,1) ? -s[1]s[2]c[3]-c[1]s[3]
! (2,1) ? c[1]s[2]c[3]-s[1]s[3]
! (3,1) ? 0
! (4,1) ? c[1]c[4]c[2]+s[1]s[4]c[2]
! (1,2) ? c[1]c[2]c[3]
! (2,2) ? s[1]c[2]c[3]
! (3,2) ? -s[2]c[3]
! (4,2) ? -s[1]c[4]s[2]+c[1]s[4]s[2]
! (1,3) ? -c[1]s[2]s[3]-s[1]c[3]
! (2,3) ? -s[1]s[2]s[3]+c[1]c[3]
! (3,3) ? -c[2]s[3]
! (4,3) ? 0
! (1,4) ? 0
! (2,4) ? 0
! (3,4) ? 0
! (4,4) ? -c[1]c[4]c[2]-s[1]s[4]c[2]
! (1,5) ? 0
! (2,5) ? 0
! (3,5) ? -1
! (4,5) ? 0

% homog m t mh
[126k]
% res m m_res
; matrix m isn't homogeneous; either homogenize it
; using 'homog', or change degrees using 'setdegs'

% res mh m_res
; -1.0.1.2.3.4.5.6.7.8..
```

```
; computation complete after degree 8

% numinfo m_res
; name    #rows #cols #standard  degrees
; 1       4     5     8          3:4 0
; 2       5     1     1          9

% type m_res.2
; -tc[1]c[2]2c[3]c[4]-ts[1]c[2]2c[3]s[4]

; -t2c[1]c[2]s[3]c[4]+c[1]c[2]s[2]2s[3]c[4]-t2s[1]c[2]s[3]s[4] \
;     +s[1]c[2]s[2]2s[3]s[4]

; c[1]c[2]2s[2]c[3]c[4]+s[1]c[2]2s[2]c[3]s[4]

; -tc[1]c[2]2c[3]c[4]+t2s[1]s[2]s[3]c[4]-s[1]s[2]3s[3]c[4]-ts[1]c[2]2c[3]s[4] \
;     -t2c[1]s[2]s[3]s[4]+c[1]s[2]3s[3]s[4]

; t3c[1]c[2]s[2]c[3]s[3]c[4]-tc[1]c[2]3s[2]c[3]s[3]c[4] \
;     -tc[1]c[2]s[2]3c[3]s[3]c[4]+t3s[1]c[2]s[2]c[3]s[3]s[4] \
;     -ts[1]c[2]3s[2]c[3]s[3]s[4]-ts[1]c[2]s[2]3c[3]s[3]s[4]

% qring is qq

% mat n
! number of rows    ? 4
! number of columns ? 5
! (1,1) ? -s[1]s[2]c[3]-c[1]s[3]
! (2,1) ? c[1]s[2]c[3]-s[1]s[3]
! (3,1) ?  0
! (4,1) ?  c[1]c[4]c[2]+s[1]s[4]c[2]
! (1,2) ? c[1]c[2]c[3]
! (2,2) ?  s[1]c[2]c[3]
! (3,2) ?  -s[2]c[3]
! (4,2) ?  -s[1]c[4]s[2]+c[1]s[4]s[2]
! (1,3) ?  -c[1]s[2]s[3]-s[1]c[3]
! (2,3) ?  -s[1]s[2]s[3]+c[1]c[3]
! (3,3) ? -c[2]s[3]
! (4,3) ? 0
! (1,4) ? 0
! (2,4) ? 0
! (3,4) ? 0
! (4,4) ? -c[1]c[4]c[2]-s[1]s[4]c[2]
! (1,5) ? 0! (1,5) ? ^R
! (1,5) ? 0
! (2,5) ? 0
! (3,5) ? -1
! (4,5) ? 0

% homog n t nh

% res n n_res
; matrix n isn't homogeneous; either homogenize it
; using 'homog', or change degrees using 'setdegs'

% res nh n_res
; -1.0.1.2.3.4..5..6..
; computation complete after degree 6

% numinfo n_res
; name    #rows #cols #standard  degrees
; 1       4     5     8          3:4 0
; 2       5     1     2          5

% type n_res.2
; tc[3]
; c[2]s[3]
; -s[2]c[3]
; tc[3]
; 0
```

Bibliographie

articles

[Appell, 1900]
P. Appell. Sur une forme générale des équations de la dynamique. *Journal für die Reine und Angewandte Mathematique*, 121:310–319, 1900.

[Boltzmann, 1902]
L. Boltzmann. Uber die form der lagrange'schen gleichungen für nichtholonome, generalisierte koordinaten. *Sitzungsberichte der Mathematisch-Naturwissenchaftlichen Klasse der Kaiserlichen Akademie der Wissenchaften*, 61:1603–1614, 1902.

[Chevallier et Helmer, 1984]
D. Chevallier and J.M. Helmer. Formation des équations de la dynamique. examen des diverses méthodes. *Annales des Ponts et Chaussées*, 1er trim. 1984.

[Gibbs, 1879]
J.W. Gibbs. On the fundamental formulae of dynamics. *American Journal of Mathematics*, 11:49–64, 1879.

[Hamel, 1904]
G. Hamel. Die lagrange-eulerschen gleichungen der mechanik. *Zeitschrift für Mathematik und Physik*, 50:1–57, 1904.

[Kane et Levinson, 1980]
T.R. Kane and D.A. Levinson. Formulation of equations of motion for complex spacecraft. *Journal of Guidance and Control*, 3(80-4014), march-april 1980.

[Moses, 1971]
J. Moses. Algebraic simplification : A guide for the perplexed. *Communications of the ACM*, 14(8), august 1971.

livres

[Arnold, 1976]
V. Arnold. *Méthodes mathématiques de la Mécanique classique*. Mir, Moscou, 1976.
Une interprétation mathématique de très haut niveau des objets de la mécanique classique, et notamment l'étude de la dynamique du corps rigide en terme de groupes de Lie.

[Cormen, Leiserson et Rivest, 1989]
T.H. Cormen, C.E. Leiserson, and R.L. Rivest. *Introduction to Algorithms*. The MIT Press, Cambridge, Massachussetts, 1989.
Un recueil rigoureux et complet sur les algorithmes classiques de l'informatique moderne.

[D'Alembert, 1743]
J.L. D'Alembert. *Traité de Dynamique*. Chez David l'aîné, Paris, 1743.
D'un grand intérêt historique et épistémologique.

[Davenport, Siret et Tournier, 1988]
J.H. Davenport, Y. Siret, and E. Tournier. *Computer Algebra - Systems and Algorithms for Algebraic Computations*. Academic Press, 1988.
Un des premiers ouvrage sur le calcul formel, un panorama complet et détaillé de l'existant

[Germain, 1980]
P. Germain. *Cours de mécanique*, volume 1. Ecole Polytechnique, Paris, 1980.
Le support d'un cours plus axé sur la mécanique des milieux continus que sur la mécanique rationnelle.

[Hamilton, 1835]
W.R. Hamilton. *Second Essay on a General Method in Dynamics*. Philosophical Transactions of the Royal Society of London, 1835.
D'un grand intérêt historique et épistémologique.

[Kane, 1978]
T.R. Kane. *Dynamics*. Stanford University, California, 3rd edition, 1978.
Un support de cours présentant le formalisme développé par son auteur.

[Lagrange, 1788]
J.L. Lagrange. *Méchanique Analytique*. Chez la veuve Desaint, Paris, 1788.
D'un grand intérêt historique et épistémologique.

[Milner, Tofte et Harper, 1990]
R. Milner, M. Tofte, and R. Harper. *The Definition of standard ML*. The MIT press, Cambridge, Massachussetts, 1990.
Presque un manuel de référence : la définition formelle du langage ML.

[Newton, 1687]
I. Newton. *Philosophiae Naturalis Principia Mathematica*. Jussu Societatis Regiae ac Typis Josephi Streater, London, 1687.
D'un grand intérêt historique et épistémologique.

[Paulson, 1991]
L.C. Paulson. *ML for the working programmer*. Cambridge University Press, Cambridge, UK, 1991.
Un excellent ouvrage pour apprendre à programmer en ML.

[Wittenburg, 1977]
J. Wittenburg. *Dynamics of Systems of Rigid Bodies*. B.G. Teubner, Stuttgart, 1977.
La description complète d'une approche des problèmes de la dynamique des systèmes polyarticulés – de la définition des notions de base à la génération des équations.

[Wolfram, 1988]
S. Wolfram. *Mathematica - A System for doing Mathematics by Computer*. Addison-Wesley Publishing Company, 1988.
Le livre de référence de l'utilisateur de Mathematica.

ouvrages collectifs

[Maes, Samin et Willems, 1990]
P. Maes, J.C. Samin, and P.Y. Willems. Autodyn & robotran – computer programmes. In W. Schielen, editor, *Multibody Systems Handbook*. Springer-Verlag, 1990.

[Wittenburg, Woltz et Schmidt, 1990]
J. Wittenburg, U. Woltz, and A. Schmidt. Mesaverde – a general purpose program package for dynamics simulations of multibody systems. In W. Schielen, editor, *Multibody Systems Handbook*. Springer-Verlag, 1990.

actes de conférences

[Chace et Korybalski, 1970]
M.A. Chace and M.E. Korybalski. Computer graphics in the schematic representation of nonlinear, constrained, multifreedom mechanical systems. In *Computer Graphics Conference*. Brunel University, 1970.

[Chevallier, 1984]
D.P. Chevallier. Groupes de lie et mécanique des structures de corps rigides. In McGraw-Hill, editor, *Mathematische Modellierung*, Kassel, RFA, Octobre 1984.

[Griesmer et Jenks, 1971]
J. Griesmer and R. Jenks. Scratchpad/1 – an interactive facility for symbolic mathematics. In *Second Symposium on Symbolic and Algebraic Manipulation*, Kassel, RFA, 1971. ACM.

[Rampalli, 1987]
R. Rampalli. Adams – a sparse matrix approach to solving multibody dynamics problems. In *Proceedings of the SDIO/NASA Workshop on Multibody Simulation*, Pasadena, California, Septembre 1987. Jet Propulsion Laboratory.

manuels d'utilisation

[Char et al., 1988]
B.W. Char, K.O. Geddes, G.H. Gonnet, M.B. Monagan, and S.M. Watt. *MAPLE Reference Manual*. WATCOM Publications Limited, Waterloo, Ontario, 5th edition, 1988.

[Garnier et Rideau, 1989b]
C. Garnier and P. Rideau. *Gemmes 1.0 - Manuel d'utilisation*. aerospatiale, Cannes, France, 1989.

[Hearn, 1985]
A.C. Hearn. *Reduce User's Manual*. Rand publication, April 1985.

[NAG Inc.]
NAG Inc. *The NAG Fortran Library Manual*. NAG Inc.

[Schaechter, Levinson et Kane, 1988]
D.B. Schaechter, D.A. Levinson, and T.R. Kane. *AUTOLEV User's Manual*. Online Dynamics Inc., Sunnyvale, California, 1988.

[The Mathlab Group, 1983]
The Mathlab Group. *Macsyma Reference Manual, version 10*. MIT, 1983.

thèses

[Capolsini, 1990]
P. Capolsini. Optimisation du code numérique engendré par le logiciel gemmes. Master's thesis, Université de Nice – Sophia Antipolis, septembre 1990.

[Dalmas, 1991]
S. Dalmas. *Un langage fonctionnel polymorphe. Application aux problèmes logiciels du calcul formel*. PhD thesis, Université de Nice Sophia-Antipolis, France, Avril 1991.

[Faure, 1992]
C. Faure. *Quelques aspects de la Simplification en Calcul Formel*. PhD thesis, Université de Nice Sophia-Antipolis, France, Avril 1992.

rapports techniques

[Borgne, 1987]
M. Le Borgne. Quaternions et contrôle sur l'espace des rotations. Rapport de Recherche 751, INRIA, novembre 1987.

[Capolsini, Dalmas et Papegay, 1991]
P. Capolsini, S. Dalmas, and Y. Papegay. Quaterman : une bibliothèque *maple* de calculs et de manipulations sur l'algèbre des quaternions. Rapport final de contrat, Centre National d'Etudes Spatial – Toulouse, 1991.

[Capolsini, Dalmas et Papegay, 1992]
P. Capolsini, S. Dalmas, and Y. Papegay. Calcul formel sur les quaternions. Rapport technique, INRIA / ESSI (UNSA), 1992.

[Capolsini, 1991]
P. Capolsini. Un logiciel formel de simulation des mécanismes. Rapport final de contrat, Centre National d'Etudes Spatial – Toulouse, 1991.

[Capolsini, 1992]
P. Capolsini. macroC : C code generation using maple. Rapport de recherche, INRIA, 1992.

[Galligo, 1983]
A. Galligo. Algorithme de constructions de bases standards. P.u.p.e, Universite' de Nice, mai 1983.

[Garnier et Rideau, 1989a]
C. Garnier and P. Rideau. Gemmes : a new tool for dynamic modelling. Rapport technique, aerospatiale, 1989.

[Gomez, 1990]
C. Gomez. Macrofort : a fortran code generator in maple. Rapport technique 119, INRIA, mai 1990.

[Papegay et al., 1991]
Y. Papegay, P. Capolsini, L. Pottier, and P. Rideau. Outils d'aide à la modélisation formelle en mécanique et en automatique. Rapport final de contrat MRT CA/TSV 173, aerospatiale / INRIA, Decembre 1991.