Calculs éléments finis paramétrés à l'aide des dérivées d'ordre élevé, Applications à l'électromagnétisme

Thanh Nam Nguyen

Institut National Polytechnique de Grenoble
Laboratoire d'Électrotechnique de Grenoble

THÈSE

pour obtenir le grade de

DOCTEUR de l'INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE

Spécialité : **Génie Électrique**

présentée et soutenue publiquement

par

NGUYEN Thanh Nam

le **09 Septembre 1998**

Titre :

Calculs éléments finis paramétrés à l'aide des dérivées d'ordre élevé. Applications à l'électromagnétisme

Directeur de thèse:

Jean-Louis COULOMB

JURY

Messieurs	**Alain NICOLAS**	Président, Rapporteur
	Xavier BRUNOTTE	Examinateur
	Jean-Louis COULOMB	Examinateur
	Gérard MEUNIER	Examinateur
	Francis PIRIOU	Rapporteur

À mes parents,

qui m'ont donné le courage de suivre cette difficile route.

Vinh dự này tôi xin dành cho cha mẹ tôi,

Người đã cổ vũ tôi dấn thân trên con đường khó khăn này.

À Lan Anh et À petit Nam Khang

qui m'ont donné la force pour terminer ce travail de thèse.

Dành cho Lan Anh & Nam Khang yêu quý,

Người đã tiếp thêm nghị lực để tôi hoàn tất luận án này.

Remerciements

Je tiens tout d'abord à remercier **Jean-Louis COULOMB**, Professeur à l'Institut National Polytechnique de Grenoble, qui a assuré l'encadrement de mon travail. Mes trois années passées à ses côtés m'ont procuré de très bon moments; Jean-Louis, par sa gentillesse, sa simplicité et son authenticité a été une formidable source de motivation durant toute la thèse. Je ne peux que répéter: Merci beaucoup mon cher Jean-Louis, merci pour tout.

J'adresse mes sincères remerciements au jury:

Monsieur **Alain NICOLAS**, Professeur à l'École Centrale de Lyon, qui m'a fait l'honneur de présider le jury et qui a eu la tâche de juger cette thèse en tant que rapporteur.

Monsieur **Francis PIRIOU**, Professeur à l'Université des Sciences de Lille, qui m'a fait l'honneur d'être rapporteur de ce travail.

Monsieur **Gérard MEUNIER**, Directeur de Recherches du CNRS, qui a accepté de participer à mon jury. Je tiens aussi à remercier Gérard, tout particulièrement, pour l'atmosphère chaleureuse qu'il a su créer au sein de l'équipe modélisation du LEG.

Monsieur **Xavier BRUNOTTE**, Responsable de FLUX3D à CEDRAT-RECHERCHE, qui a accepté de participer à mon jury.

J'exprime mes remerciements:

À Pascal PETIN et Lucas SALUDJAN, mes coéquipiers, qui ont développé les autres aspects de la méthode des dérivées d'ordre élevé et de l'optimisation. À Louis DE MEDEIROS pour les outils de calculs de force et d'énergie dans FLUX3D.

À Anne Isabelle et Patrick GIRARD, les responsables de FLUX-PARAM à CEDRAT-RECHERCHE, qui m'ont donné la possibilité de tester FLUX-PARAM depuis sa toute

première version (pré-alpha...). Il faut dire que cela a été une expérience enrichissante dans ce travail de thèse.

À Stéphane COURTINE et Vincent LECONTE, pour le temps qu'ils ont bien voulu me consacrer en corrigeant mon français pendant la rédaction de ce mémoire. J'assure que cette expérience a été très vive et très amusante.

Aux responsables informatiques du réseau HP: Etiennette CALLEGHER, Patrick GUILLOT, Patrick EUSTACHE pour leur infinie patience.

À l'ensemble des chercheurs, ingénieurs, techniciens et personnels administratifs du laboratoire et particulièrement aux membres de l'équipe "Modélisation et Conception Assistée par Ordinateur en Électromagnétisme" pour l'ambiance chaleureuse qu'ils ont manifestée tout au long de ma présence au labo.

Enfin, je ne terminerai pas cet avant-propos sans remercier mes amis vietnamiens, avec qui j'ai eu le plaisir de partager de très bons moments pendant ces quatre dernières années à Grenoble et qui m'ont beaucoup encouragé dans les moments difficiles.

Grenoble, Septembre 1998.

NGUYEN Thanh Nam

TABLE DES MATIÈRES

TABLE DES MATIÈRES

NGUYEN Thanh Nam

INTRODUCTION GÉNÉRALE

INTRODUCTION GÉNÉRALE

Initiée dans le domaine de la mécanique des structures, la méthode des éléments finis (MEF) a, aujourd'hui, conquis les ingénieurs-chercheurs de nombreuses disciplines comme: le génie civil, l'électrotechnique, l'aéronautique, etc.. Étant un moyen puissant pour l'analyse de performance des structures, de plus en plus de logiciels de calcul EF deviennent des outils de conception, des plates-formes de travail indispensables pour les concepteurs-experts. Dans ces domaines d'applications, l'information de sensibilité(*) est très demandée, parfois on peut même dire que le gradient est devenu plus important que la solution elle-même.

Classiquement le gradient et même les dérivées secondes de la solution sont obtenus par plusieurs réévaluations (re-analyses) du problème. En utilisant la MEF, une telle procédure est alors assez limitée à cause de la perte de temps engendrée par plusieurs analyses complexes et coûteuses. L'idée d'obtenir dans une seule analyse toutes les informations nécessaires, est donc très séduisante, nous parlons d'une analyse globale de sensibilité qui va construire la solution EF sous la forme des fonctions dépendant des paramètres de conception.

Au LEG, notre travail fait suite aux travaux de KOUYOUMDJIAN [Kouyo85] puis GISTOSUSASTRO [Gitos88], SALDANHA [Salda92], WEEBER [Weebe92] et PETIN [Petin96] qui ont étudié les méthodes de sensibilité au sein de calculs EF et qui ont proposé diverses implantations pour les applications électrotechniques. Du côté mathématique, ce sujet s'inspire des travaux de MASMOUIDI, GUILLAUME et ROCHETTE [Guill94, GuiMa94].

* La sensibilité s'exprime comme la dérivée totale, le gradient, d'une fonction ou d'une grandeur par rapport aux paramètres

Quel environnement ?

0.1. Le calcul EF, un développement dans un environnement

a. *MEF et les grands codes de calcul existants.*

Pendant des années, les travaux des équipes de recherche ont donné naissance à plusieurs grands logiciels qui sont basés sur la méthode des éléments finis. Bien que les approches soient universelles, les implantations sont fréquemment très particulières. Par conséquent, une utilisation efficace du code nécessite beaucoup d'expériences et en demande encore plus pour la maîtrise en vue d'un développement global. En pratique, un chercheur, effectuant un développement à court terme, doit exercer un choix difficile entre deux approches:

◊ Valider l'algorithme dans un nouveau module de test indépendant. Cette approche donne au développeur une plus grande liberté de création, cependant il doit construire beaucoup d'outils complémentaires, surtout des interfaces complexes avec les outils existants. Dans ce cas, la poursuite de son travail et sa réutilisation sont généralement difficiles.

◊ Implanter l'algorithme comme une partie intégrée dans un logiciel existant. La réalisation doit respecter donc toutes les règles et les limitations internes, elle doit surtout être mise à jour régulièrement pour cohabiter avec les autres développements en cours.

En espérant une poursuite possible de notre travail, nous avons choisi la deuxième approche. Il est intéressant aussi de faire connaissance avec un grand logiciel de calcul, avec un environnement complet en apprenant les détails de son histoire, sa structure, son état et aussi les personnes qui l'ont créé.

b. *Notre environnement de développements.*

Au LEG, l'environnement commun est "FLUX" une famille de logiciels d'analyse par MEF dédiés aux applications électrotechniques (électromagnétisme et thermique). Ces logiciels sont développés en collaboration avec CEDRAT S.A. qui les commercialise. La famille se compose d'une plate-forme complète de traitements EF: le descripteur de la géométrie, le générateur de maillage, la description de problème, le processeur de calcul et le module d'exploitation des résultats. Notre mise en oeuvre informatique se situe principalement dans FLUX3D, la version 3D de cette famille.

Et le but ?

0.2. Le calcul paramétré

La méthode des éléments finis nous offre un moyen performant d'analyser le modèle donné, c'est-à-dire un dispositif entièrement défini. Aujourd'hui, la synthèse et l'optimisation utilisent de plus en plus les résultats précis de tels calculs EF. Traditionnellement, il faut changer plusieurs fois la description du problème à l'aide des paramètres et ensuite répéter le calcul afin de trouver la solution optimale.

La question se pose ainsi: Peut-on obtenir immédiatement (ou du moins très rapidement) une dérivée de solution ou une autre solution à chaque changement de paramètres ? La réponse est affirmative, dans une analyse de sensibilité d'ordre élevé, la solution EF peut être construite sous la forme de développement de Taylor dont l'évaluation est instantanée (ou du moins relativement rapide).

L'objectif de cette thèse consiste à développer la méthode de dérivées d'ordre élevé dans une réalisation pratique au sein de la méthode des éléments finis. Premièrement, nous éclaircirons les points caractéristiques de tels calculs en proposant une analyse aussi détaillée que possible. Ensuite, une implantation de cette méthode sera réalisée. Il est évident que l'investissement est très important, la mise en oeuvre doit être développée étape par étape. Dans ce travail, notre objectif prioritaire est un traitement global par rapport aux paramètres géométriques et non pas aux paramètres physiques dans la mesure où cet aspect a été déjà abordé par P.PETIN [Petin96].

Comment s'organise ce mémoire ?

0.3. Les chapitres

Le premier chapitre sera réservé à la description de la méthode des éléments finis, particulièrement en électrotechnique. Le deuxième chapitre présentera les bases théoriques de la méthode des dérivées d'ordre élevé. Nous y proposerons les concepts de base pour une réalisation pratique au sein de la méthode des éléments finis: les paramètres de sensibilité, les méthodes de dérivation, la convergence et la validité etc..

Le chapitre 3 sera consacré à la mise en oeuvre du dérivateur de maillages qui est une étape indispensable concernant les paramètres géométriques. Ce chapitre mettra en évidence une méthode de dérivation semi-analytique qui permet de construire rapidement le maillage EF "paramétré", c'est-à-dire de définir les polynômes décrivant la position des noeuds du maillage EF. Remarquons que le modèle de représentation géométrique est un élément essentiel dans cette opération. Bien que l'approche proposée soit générale, notre implantation sera concentrée principalement sur un modèle de représentation par des frontières.

Dans le quatrième chapitre, nous étudierons une organisation efficace (et portable) du solveur "paramétré". En analysant les diverses approches de dérivations, nous présenterons une mise en oeuvre "dynamique" qui se décompose en des procédures d'intégrations symboliques et de résolutions successives. En effet, un langage de commande permettra d'émuler cette phase de résolution paramétrée.

Enfin, nous présenterons dans le chapitre 5 des exemples de calculs pour valider la procédure. La méthode étant établie, nous proposerons alors quelques exploitations basées sur les résultats paramétrés de calculs.

NGUYEN Thanh Nam

LEG

Chapitre 1

LA MÉTHODE DES ÉLÉMENTS FINIS ÉTAT DE L'ART

Sommaire

Chapitre 1

LA MÉTHODE DES ÉLÉMENTS FINIS - ÉTAT DE L'ART

Le but de ce chapitre est de faire un état des lieux de la méthode des éléments finis et de présenter le cadre de notre recherche.

D'abord, nous donnerons quelques généralités sur la méthode des éléments finis (MEF) en tant qu'outil de calculs techniques. Puis, en s'appuyant sur une équation aux dérivées partielles particulière, nous rappellerons les principes mathématiques de la MEF et nous introduirons les notations utilisées dans les chapitres ultérieurs.

Ensuite, en insistant sur les points forts de la MEF, nous verrons pourquoi le calcul EF a trouvé sa place dans la conception et dans l'optimisation. Nous montrerons donc les nouveaux besoins créés et nous aurons alors posé les bases de notre étude.

Enfin, nous présenterons rapidement les particularités de la famille des logiciels FLUX, surtout celles de FLUX3D, la plate-forme d'accueil de nos développements informatiques.

1.1. La MEF, une méthode d'analyse.

☞ ***Un bref Résumé de la MEF:***

◊ Le besoin: Résolution d'un système d'équations aux dérivées partielles (EDP) dont la solution analytique en continu est impossible à définir.

◊ La Nature: Résolution numérique.

◊ Le Principe: Approximation discrète du problème continu.

☞ ***Des points caractéristiques:***

◊ Le milieu (dispositif étudié) est subdivisé en un nombre fini de sous domaines de formes géométriques simples, les éléments finis (EF). Sur chaque élément, les phénomènes physiques dépendent d'un nombre fini de variables, qui sont les variables d'état du problème choisi.

◊ La solution du problème global est définie par l'ensemble de toutes les variables d'état, en s'appuyant sur la topologie du problème, la topologie des EFs.

☞ ***Une découverte de physico-mathématiciens:***

Cette démarche existe depuis les années 50s avec comme point de départ un phénomène physique théorique continu (première application en mécanique des structures):

Équations différentielles d'un problème continu
—|| variables d'état ||—
→ formulation intégrale (fonctionnelle) —
→ domaine découpé en EF —
→ intégration sur chaque élément —
→ construction d'un système matriciel associé au problème —
→ itération/résolution numérique —

Fig.1.1. Première présentation de la MEF

☞ *L'Outil de calcul et la Plate-forme de travail*

Les développements des outils informatiques ont donné lieu à la naissance de logiciels de calculs EF conçus pour traiter de manière générique un grand nombre de problèmes techniques. Pour l'utilisateur, le traitement par la méthode EF se compose donc, dans l'ordre, de trois grands blocs (Fig.1.2).

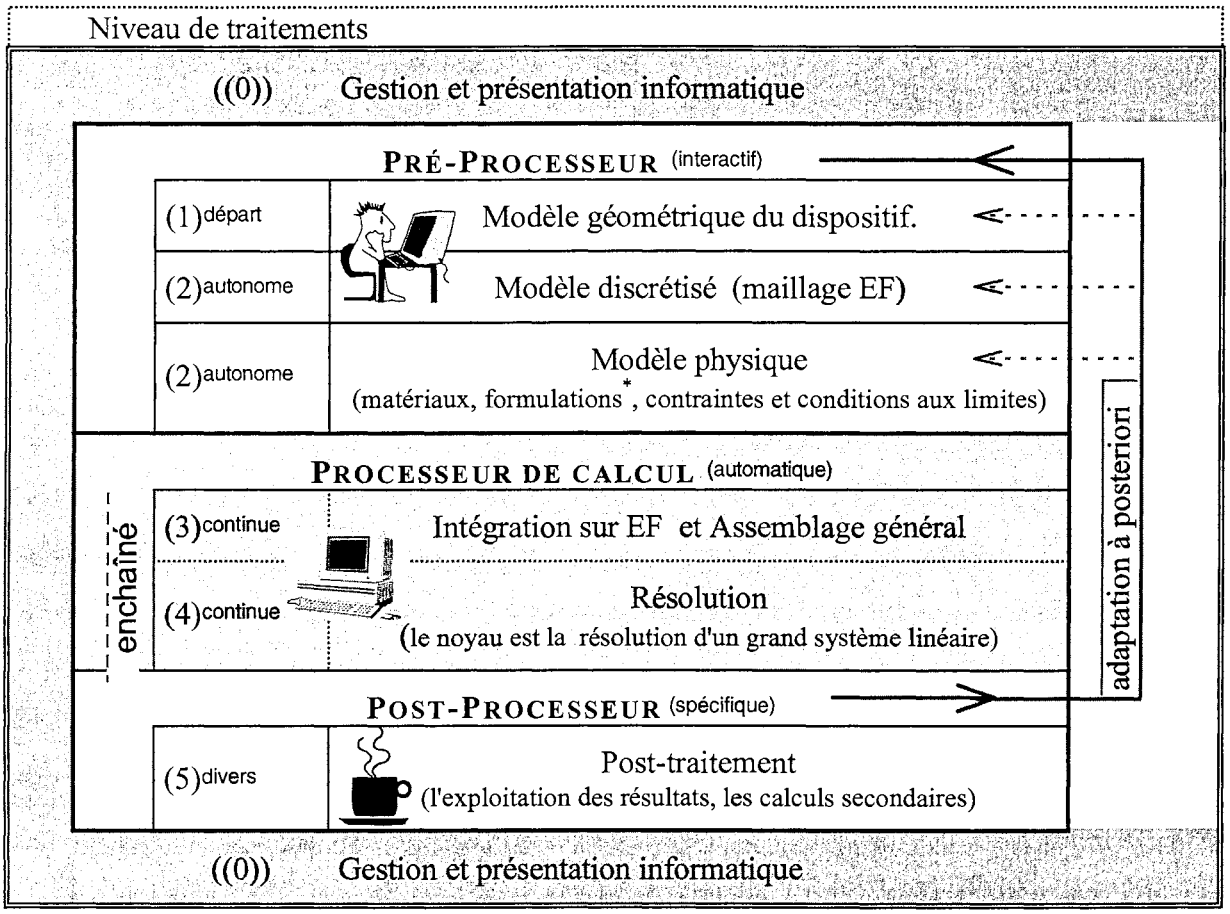


Fig.1.2. Organigramme typique d'une chaîne de calcul EF.

Remarque: Au fil des évolutions, les programmes EF se diversifient fortement, mais cette organisation est largement reconnue (en littérature). Elle devient une forme conventionnelle pour présenter les particularités de la MEF.

---

* formulation est un modèle simplifié d'EDP avec un choix définitif des variables d'état (degré de liberté d'un EF). En effet, les équations générales seraient trop complexes à résoudre dans leur intégralité. On effectue donc des hypothèses simplificatrices, suivant le dispositif et les phénomènes étudiés, pour se ramener à une EDP plus simple à traiter.

a. Le Pré-Processeur

Cette étape s'appelle aussi l'étape de ***pré-traitement***, c'est donc la première grande phase et la plus interactive dans le calcul EF. Dans cette étape, l'utilisateur est entièrement occupé à décrire son problème en communiquant avec les moyens informatiques. Une des particularités de cette phase réside en son organisation informatique. Ses trois modules sont très distincts et par conséquent ils sont souvent réalisés par des outils indépendants. De plus, les besoins en échange d'informations, sont très grands dans cette étape, ce qui exige plusieurs outils de communication et de conversion des données.

- ***Descripteur géométrique*** ([SabCo86 ,Terra85, Alber88]).

  La *description géométrique* est toujours la première étape d'un traitement EF mais elle n'est pas réalisable sans la connaissance du problème physique. Par exemple, pour un problème électromagnétique, on doit décrire les boîtes d'air et a priori encore une boîte englobante pour modéliser l'infini. Les descripteurs géométriques sont classés principalement par leur modèle de représentation géométrique. Le monde de la CAO utilise plusieurs représentations. Les plus utilisées sont: la Représentation par des Frontières B-Rep*(angl: B-Rep, Boundary Representation), la Représentation Surfacique (surface de Béziers, B-Splines, NURBS*(angl: Non Uniform Rational B-Splines)) et la Géométrie des Solides Constructive CSG*(angl: CSG, Constructive Solid Geometry).

  La comparaison des modèles est toujours très délicate, car les critères dépendent des applications envisagées. Il est évident qu'une géométrie très complexe ne peut être décrite que par un modeleur surfacique, et qu'une notion de sémantique métier est utilisé par des modeleurs CSG, où les opérations technologiques peuvent être simulées facilement. Ces deux modeleurs permettent une complexification importante de la géométrie, mais ils refusent en général les entités décrites par un autre modeleur. La représentation B-Rep est plus tolérante. Concernant les mailleurs, la B-Rep est pratiquement la seule représentation directement exploitable par un générateur (automatique) de maillage. En conclusion, on remarque que la B-Rep a un net avantage pour la représentation de base, en particulier pour les applications électrotechniques.

- ***Le Mailleur et le Modèle discrétisé*** ([SabCo86, Zgain96]).

  Le Mailleur est un module qui sert à découper le domaine étudié en EFs, c'est-à-dire à construire le maillage et le modèle discrétisé. Aujourd'hui, un algorithme de maillage automatique (générateur de maillage libre) est présent dans presque tous les mailleurs. En pratique, ce mailleur est assisté par l'utilisateur pour la définition de la densité moyenne de noeuds ou le taille de EFs. Le mailleur automatique est très rapide, pratique et largement utilisé, bien que les maillages générés ne soient pas toujours parfaits.

  À part les algorithmes de maillage automatique, il existe des mailleurs adaptatifs ou parfois manuels et un grand nombre d'outils pour améliorer la qualité du maillage. Les techniques de maillage et de contrôle de la qualité du maillage, très développées, restent assez mystérieux pour le simple utilisateur de la méthode des EFs.

- ***Le Descripteur du problème physique***

  C'est une étape proche du phénomène étudié, où il faut d'abord choisir le type de problème à résoudre et ensuite réaliser la description physique relative à ce problème. Il faut donner les propriétés des matériaux, imposer les contraintes, établir les conditions aux limites, etc.. Normalement c'est l'étape la plus courte mais elle demande beaucoup de connaissances et d'expérience aux utilisateurs, surtout dans les applications multi-domaines.

b. Le Processeur de calcul

C'est une partie entièrement automatisée et nécessitant le plus de puissante dans un calcul EF. En effet, les opérations sur les grands systèmes matriciels demandent beaucoup de mémoire et de la puissance de calcul arithmétique. Un problème assez simple est résolu rapidement, mais sur les applications industrielles réalistes, le calcul devient très long. On distingue deux modules dans cette étape:

- ***Intégration et Assemblage EF:*** s'appuyant sur la description physique et le maillage, l'intégration fonctionnelle est réalisée élément par élément, région par région, et est suivie par une procédure d'assemblage afin de construire un système matriciel associé au problème global. Ici, on utilise différentes méthodes d'intégration. La plus adaptée au calcul EF est la méthode d'intégration numérique de Gauss.

- ***Résolution du système matriciel:*** Le noyau de cette phase est une procédure de résolution d'un système linéaire creux (par exemple, une méthode itérative de type gradient conjugué ICCG [FLUX3D]). Un contrôle intervient à la fin de cette procédure pour signaler, soit la sortie, soit une nouvelle itération interne (non linéaire) ou une itération enchaînée avec un nouvel assemblage. La dimension très importante du problème EF et le bouclage de non-linéarité sont responsables des heures et parfois des jours de calculs.

  À la fin de cette étape, les valeurs des variables d'état sont connues aux noeuds du maillage.

c. Le Post-Processeur:

Ce module sert à exploiter au mieux les résultats fournis par le processeur central, c'est aussi une partie spécifique à la physique. En utilisant la variable d'état aux noeuds du maillage, connue à ce stade de l'analyse EF, le post-processeur fournit les moyens d'analyse complète du problème et de visualisation dans l'espace et le temps. On y trouve donc des outils de calcul de grandeurs dérivées et d'intégrales spécifiques qui peuvent intéresser l'utilisateur et des outils de visualisation de ces grandeurs selon plusieurs modes graphiques: courbes de niveaux, dégradés, tracés de courbes et de surfaces, etc.. En général, cette phase exige moins de ressource mémoire et de puissance arithmétique que l'étape précédente. Cependant, le calcul de certaines grandeurs intégrales peut être très coûteux, et la visualisation 3D est plus confortable avec de la mémoire et des moyens de calcul adaptés.

Enfin, il y a deux tendances pour la réalisation informatique. La première consiste en l'utilisation d'une seule plate-forme complète et performante spécifique au traitement EF. La deuxième, au contraire, revient à utiliser plusieurs modules spécialisés qui s'échangent les données EF.

En conclusion, la puissance de la méthode des EF est incontestablement établie. Cependant, elle reste très lourde à utiliser en tridimensionnel. Certes, les performances des ordinateurs augmentent sans cesse, mais les utilisateurs veulent toujours plus de précision et traitent des problèmes de plus en plus complexes avec plusieurs paramètres, les temps de calcul sont donc toujours importants.

1.2. La MEF en électrotechnique

Les premiers ouvrages et applications EF résolues en électromagnétisme sont apparus à la fin des années 1970, début des années 1980 [ChaSi80, SilFe83]. L'épanouissement des applications EF en électrotechnique a donné naissance à des outils-logiciels: FLUX2D, FLUX3D, PHI3D, TRIFOU, etc., qui sont dédiés notamment aux besoins spécifiques de ce domaine. Bien entendu, les électrotechniciens utilisent aussi les logiciels et les outils multidisciplinaires (mécanique, thermique, ...) dans leurs intégralités. Il y a plusieurs phénomènes à étudier en conception de dispositifs électrotechniques, mais nous nous intéressons ici à la distribution des champs électromagnétiques. Les équations générales décrivant la physique des problèmes sont connues sous le nom d'équations de Maxwell.

Modèle général électromagnétique de Maxwell

Équations de couplage électromagnétique

(1.1) $\vec{\nabla} \times \vec{\mathbf{E}} = -\partial\vec{\mathbf{B}}/\partial t$ équation Faraday

(1.2) $\vec{\nabla} \times \vec{\mathbf{H}} = \vec{\mathbf{J}} + \partial\vec{\mathbf{D}}/\partial t$ équation Ampère

Équations de conservation de Gauss

(1.3) $\vec{\nabla} \cdot \vec{\mathbf{D}} = \rho$

(1.4) $\vec{\nabla} \cdot \vec{\mathbf{B}} = 0$

Lois de comportement du milieu

(1.5) $\vec{\mathbf{D}} = \varepsilon . \vec{\mathbf{E}}$ Diélectrique $\varepsilon = \varepsilon_r \varepsilon_0$

(1.6) $\vec{\mathbf{B}} = \mu . \vec{\mathbf{H}} + \vec{\mathbf{B}}_r$ Matériau magnétique $\mu = \mu_r \mu_0$

(1.7) $\vec{\mathbf{J}} = \sigma . \vec{\mathbf{E}}$ Lois d'Ohm, conduction

Avec:	$\vec{\mathbf{E}}$	Champ électrique	[V/m]
	$\vec{\mathbf{D}}$	Induction électrique	[C/m^2]
	$\vec{\mathbf{H}}$	Champ magnétique	[A/m]
	$\vec{\mathbf{B}}$	Induction magnétique	[T]

$\vec{B}_r$	Induction rémanente des aimants	[T]
$\vec{J}$	Courant volumique ou densité de courant	[A/m^2]
ε	Permittivité électrique $\varepsilon=\varepsilon_o.\varepsilon_r$	[F/m]
ε_o	permittivité absolue du vide 8,85 10^{-12}	
ε_r	permittivité relative (diélectrique)	
μ	Perméabilité magnétique $\mu=\mu_o.\ \mu_r$	[H/m]
μ_o	perméabilité absolue du vide $4\pi.10^{-7}$	
μ_r	perméabilité relative	
ν	Réluctivité $\nu=1/\mu$	
ρ	Charge volumique ou densité de charge	[C/m^3]
σ	Conductivité	$[\Omega m]^{-1} \equiv [S/m]$

La résolution générale de ces équations dans toute leur intégralité est trop complexe. Selon les applications, il est possible de négliger certains phénomènes sans affecter la précision des résultats. Alors, on a recours à une formulation pratique, qui est un modèle simplifié d'EDP accompagné par un choix des variables d'état (des degrés de liberté d'un EF).

Une liste complète des formulations électromagnétiques et de leurs domaines d'applications est donnée dans [SabCo86, CouSa86]. Ici, pour démontrer le principe mathématique d'une analyse par MEF, nous détaillerons une seule formulation, la magnétodynamique vectorielle.

Formulation magnétodynamique vectorielle

Hypothèses simplificatrices dans un dispositif à induction classique [CouSa86]:

◊ Pas de courant de déplacement

◊ Pas de charge électrique (potentiel électrique $\mathcal{V}$ est considéré comme nul partout)

◊ Pas d'aimantation permanente

◊ Régime sinusoïdal

Imposant le vecteur potentiel magnétique: $\vec{\mathbf{A}} = \begin{vmatrix} A_x \\ A_y \\ A_z \end{vmatrix}$ pour que $\vec{\mathbf{B}} = \vec{\nabla} \times \vec{\mathbf{A}}$.

Il vient automatiquement $\vec{\nabla} \cdot \vec{\mathbf{B}} = \vec{\nabla} \cdot (\vec{\nabla} \times \vec{\mathbf{A}}) = 0$ et aussi

$$\vec{\nabla} \times (\vec{\mathbf{E}} + \partial\vec{\mathbf{A}}/\partial t) = 0 \quad \Rightarrow \quad \vec{\mathbf{E}} + \partial\vec{\mathbf{A}}/\partial t = (\nabla \mathcal{V}) = 0$$

Il reste à résoudre l'équation: $\vec{\nabla}\times\vec{\mathbf{H}}-\sigma\vec{\mathbf{E}}=\vec{\mathbf{J}}_{ex}$ (J_{ex} est le courant d'excitation connu)

Le modèle s'exprime alors en terme de potentiel vecteur:

$$\vec{\nabla}\times(\nu.\vec{\nabla}\times\vec{\mathbf{A}})+\sigma\frac{\partial\vec{\mathbf{A}}}{\partial t}=\vec{\mathbf{J}}_{ex}. \qquad (1.8)$$

Prenant en compte le régime sinusoïdal, (1.8) peut être écrit sous forme complexe:

$$\vec{\nabla}\times(\nu.\vec{\nabla}\times\vec{\mathbf{A}})+j\omega\sigma.\vec{\mathbf{A}}=\vec{\mathbf{J}}_{ex}. \qquad (1.9)$$

Dans le cas 2D linéaire les vecteurs $\vec{\mathbf{A}}$ et $\vec{\mathbf{J}}$ sont normaux au plan d'étude et on peut les traiter comme des grandeurs scalaires:

$$\vec{\mathbf{A}}=\begin{vmatrix}0\\0\\A\end{vmatrix},\ A=A_z=\text{real}\{A(x,y).e^{j\omega t}\}\ \text{ et }\ \mathbf{J}_s=\begin{vmatrix}0\\0\\J\end{vmatrix},\ J=J_z=\text{real}\{J_s(x,y).E^{j\omega t}\} \qquad (1.10)$$

Dans ce cas, équation (1.9) devient $\nabla(\nu.\nabla A)+j\omega\sigma A=J$, dont la résolution est effectuée en minimisant la fonction énergétique [ChaSi80, CouSa86]:

$$F(A)=\frac{1}{2}\iint_{\Omega}\left(\nu.(\nabla A)^2+j\omega\sigma A^2-2J.A\right)d\Omega \qquad (1.11)$$

La méthode des éléments finis permet donc d'interpoler les fonctions inconnues (vecteur $\vec{\mathbf{A}}$) par une approximation discrète, par exemple une interpolation nodale:

$$\vec{\mathbf{A}}_{disc}=\sum_{j=1}^{NN}\alpha_j.\vec{\mathbf{A}}_j\ \text{ ou }\ A_{disc}=\sum_{j=1}^{NN}\alpha_j.A_j \qquad (1.12)$$

où $\alpha_j=\alpha_j(x,y,z)$, sont les fonctions d'interpolation nodale, c'est-à-dire les fonctions de forme, qui sont définies sur chaque élément,

$\vec{\mathbf{A}}_j$ ou A_j, est la valeur au noeud numéro "j" dans l'ensemble de noeuds.

En utilisant cette approximation EF et la procédure de Galerkin, on obtient un système matriciel complexe qui est en général très creux:

$$[\mathbf{M}].\{\mathbf{A}\}=[\mathbf{P}+j\mathbf{Q}].\{\mathbf{A}\}=\{\mathbf{S}\} \qquad (1.13)$$

où **{A}** est un vecteur de NN valeurs nodales inconnues à déterminer

[M] = [P +jQ] est la matrice complexe des coefficients [NN×NN],

{S}, est le vecteur source.

Les matrices **P, Q** et le vecteur **S** sont construits à l'aide des intégrales élémentaires dans une procédure d'assemblage EF:

(1.14) $$[\mathbf{P}] = \mathbf{A}\Big\| \mathbf{P}_e = \mathbf{A}\Big\| \iint_{\Omega_e} \nu.[\nabla\alpha][\nabla\alpha]^T \, d\Omega$$

(1.15) $$[\mathbf{Q}] = \mathbf{A}\Big\| \mathbf{Q}_e = \mathbf{A}\Big\| \iint_{\Omega_e} \omega\sigma.\{\alpha\}\{\alpha\}^T \, d\Omega$$

(1.16) $$\{\mathbf{S}\} = \mathbf{A}\Big\| \mathbf{S}_e = \mathbf{A}\Big\| \iint_{\Omega_e} J.\{\alpha\} \, d\Omega$$

avec $\mathbf{A}\Big\|$: Opérateur de l'assemblage des matrices globales et du vecteur global, (s'appuyant sur la topologie du problème et du maillage).

$\{\alpha\}$: ensemble des fonctions d'interpolation sur les EFs

Par exemple, pour un triangle à trois noeuds {1N,2N,3N}on a les fonctions d'interpolation:

(1.17) $$\{\alpha\} = \begin{Bmatrix} \alpha_{1N} \\ \alpha_{2N} \\ \alpha_{3N} \end{Bmatrix} \text{ et } [\nabla\alpha] = \begin{Bmatrix} \{\nabla\alpha_{1N}\}^T \\ \{\nabla\alpha_{2N}\}^T \\ \{\nabla\alpha_{3N}\}^T \end{Bmatrix} = \begin{bmatrix} \frac{\partial\alpha_{1N}}{\partial x} & \frac{\partial\alpha_{1N}}{\partial y} \\ \frac{\partial\alpha_{2N}}{\partial x} & \frac{\partial\alpha_{2N}}{\partial y} \\ \frac{\partial\alpha_{3N}}{\partial x} & \frac{\partial\alpha_{3N}}{\partial y} \end{bmatrix}$$

Maintenant, regardons **le cas 3D**, en soulignant que le passage du 2D au 3D est toujours très compliqué, en particulier, il faut effectuer plusieurs traitements supplémentaires. Par exemple, si on utilise des éléments nodaux, la formulation (1.9) doit être modifiée afin d'assurer l'unicité de la solution. La méthode la plus usuelle consiste à introduire dans l'équation initiale un terme de pénalité $-\vec{\nabla}(\nu.\vec{\nabla}\cdot\vec{\mathbf{A}})$ [Coulo81]:

(1.18) $$\vec{\nabla}\times(\nu.\vec{\nabla}\times\vec{\mathbf{A}}) - \vec{\nabla}(\nu.\vec{\nabla}\cdot\vec{\mathbf{A}}) + j\omega\sigma.\vec{\mathbf{A}} = \vec{\mathbf{J}}$$

Cependant, l'utilisation d'éléments d'arête permet de dispenser de ce traitement [Bossa89, Webb93, Golov97]. En réalité, les éléments d'arête n'imposent que la continuité des composantes tangentielles du champ au passage, autrement dit, ce type d'éléments permet une discontinuité de composantes normales. Ces éléments sont les plus adaptés pour représenter le champ 3D vectoriel.

Reprenons le cas ordinaire de l'équation (1.18). En utilisant la méthode des résidus pondérés, on obtient:

(1.19) $$\iiint_V w_i\left[\vec{\nabla}\times(\nu.\vec{\nabla}\times\vec{\mathbf{A}}) - \vec{\nabla}(\nu.\vec{\nabla}\cdot\vec{\mathbf{A}}) + j\omega\sigma\vec{\mathbf{A}}\right]dV = \iiint_V w_i.\vec{\mathbf{J}}\,dV$$

Où, $\mathbf{w}_i = w_i(x,y,z)$ sont des fonctions de pondération, qui doivent être choisies pour remplir toutes les exigences des conditions aux limites:

$\vec{\mathbf{H}} \times \vec{n} = 0$ Neuman sur Γ_n

$\vec{\mathbf{B}} \cdot \vec{n} = 0$ Diriclet sur Γ_0

Une intégrale par partie s'appuyant sur ces conditions aux limites, fait transformer (1.19) vers la forme:

$$\iiint_V (\vec{\nabla} w_i \times \nu . \vec{\nabla} \times \vec{\mathbf{A}} - \vec{\nabla} w_i \nu . \vec{\nabla} \cdot \vec{\mathbf{A}} + j\omega\sigma . w_i \vec{\mathbf{A}}) dV = \iiint_V w_i . \vec{\mathbf{J}} dV \quad \textbf{(1.20)}$$

En prenant l'interpolation (1.12) et en choisissant $\{\mathbf{w}\} \equiv \{\alpha\}$ (méthode de Galerkin), l'équation (1.20) prend donc la forme:

$$\sum_{j=1}^{NN} \vec{\mathbf{A}}_j \iiint_V (\vec{\nabla}\alpha_i \times \nu . \vec{\nabla}\alpha_j - \vec{\nabla}\alpha_i \nu . \vec{\nabla}\alpha_j + j\omega\sigma . \alpha_i . \alpha_j) dV = \iiint_V \alpha_i . \vec{\mathbf{J}} dV \quad \textbf{(1.21)}$$

Finalement, une procédure globale d'assemblage va construire un système matriciel ressemblant à (1.13), sauf que les formules d'assemblage (1.14), (1.15) et (1.16) deviennent beaucoup plus volumineuses et complexes.

Les développements actuels en calcul EF.

1.3. Deux axes de développements sur le calcul EF

Il faut noter que la majorité des programmes de calcul EF existent depuis des dizaines années et que leur implantation informatique doit subir au plus vite une transformation majeure pour arriver aux nouvelles normes de l'interactivité, pour s'adapter aux nouvelles techniques de programmations, etc.. Plusieurs milliers de lignes de codes attendent donc la modernisation, mais (un grand MAIS) ces travaux ne relèvent pas de la recherche en physique.

Aujourd'hui, on distingue deux différents types d'études (physiques) portant sur la MEF: les améliorations dans le calcul EF et les applications sur un calcul EF établi. En fait, ces besoins sont complémentaires dans plusieurs cas.

☞ Amélioration apportée au calcul EF

Dans cette catégorie, on trouve les développements consacrés à toutes les étapes du calcul EF (Fig.1.2). L'objectif est d'améliorer la performance et de diminuer le coût de calcul. Les principales directions de recherche sont:

⇒ L'augmentation de la précision par une amélioration du maillage EF: introduction de nouveaux éléments, de nouvelles fonctions d'approximation; étude de nouveaux algorithmes de maillage ou amélioration de la qualité du maillage.

⇒ Les améliorations numériques sur les formulations: amélioration du conditionnement des systèmes matriciels; accélération du calcul en utilisant de nouveaux algorithmes mathématiques (vectorisation, parallélisme), etc..

⇒ L'élaboration de formulations plus complexes pour décrire des problèmes plus réalistes; réalisation de couplages entre différentes formulations ou entre différents phénomènes.

⇒ Et enfin, les améliorations apportées aux calculs des grandeurs physiques utiles en post-traitement, par exemple, la force globale, l'énergie magnétique, etc..

☞ Applications sur le calcul EF bien établi

Bien qu'il y ait encore une large place pour la recherche sur la méthode des éléments finis, son enracinement est bien établi dans plusieurs domaines techniques. Les logiciels EF sont devenus des outils indispensables pour valider un modèle donné, c'est-à-dire un modèle entièrement défini.

Avec l'informatisation du travail des ingénieurs-physisiens, la conception assistée par ordinateur (CAO) remplace progressivement les formules empiriques. On s'attaque donc aux problèmes inverses, aux problèmes d'optimisation: Il faut trouver les "bonnes" valeurs et pas seulement vérifier que telles ou telles conviennent. Il y a eu de nombreux efforts pour coupler les logiciels EF avec diverses méthodes d'optimisations mathématiques. Une procédure générale consiste à prévoir un modèle paramétré, s'appuyant sur une analyse EF, et à définir ensuite les contraintes du problème, c'est-à-dire une fonction d'objectif par rapport aux paramètres. L'optimisation se fait par le bouclage automatique suivant:

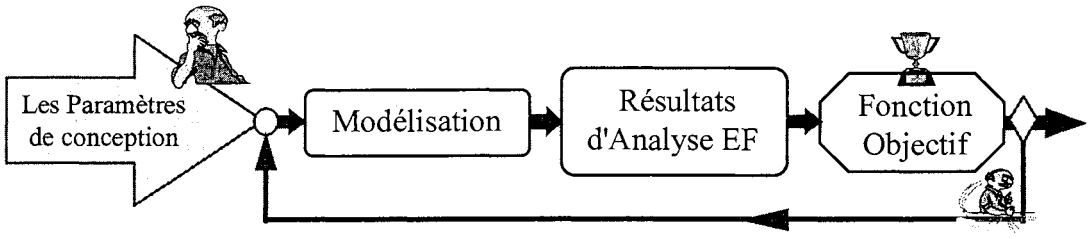


Fig.1.3. Cycle de travail en optimisation et en synthèse de dispositif.

Ce type de travail fait l'objet de beaucoup d'études, dans de nombreux domaines physiques. La première difficulté réside dans le choix de la fonction objectif, qui doit être continûment dérivable par rapport à chacun des paramètres de conception. L'autre difficulté plus fondamentale est le coût important d'analyses par la MEF, ceci représente l'essentiel du temps passé au cours de chaque itération. Cette lenteur limite aussi le nombre total d'itérations et par conséquent, elle limite le choix des algorithmes d'optimisation.

Le besoin est présent, il est important de porter les efforts pour le résoudre.

1.4. L'approche par l'Analyse de sensibilité

Dans le paragraphe précédant, nous avons remarqué les insuffisances de l'analyse EF classique par rapport aux problèmes inverses. En effet, apparaît un besoin d'exploiter plus en profondeur chaque itération EF. On parle alors d'étude de sensibilité [*]. Cette démarche repose sur l'idée qu'il vaut mieux donner à l'utilisateur le maximum d'informations pour qu'il puisse lui-même modifier les paramètres de son problème. Assisté, l'utilisateur gagne du temps, tout en gardant le contrôle de l'évolution. Il peut donc intégrer son savoir-faire. Du point de vue de la réalisation, le calcul de sensibilité correspond à un développement de la MEF, mais par sa nature, il est très proche de l'optimisation mathématique.

La populaire méthode du gradient

Dans la littérature, on constate que la majorité des efforts porte sur la vitesse de convergence de la boucle (Fig.1.3) en utilisant les dérivées premières, c'est-à-dire le gradient de la fonction objectif. Souvent, le calcul du gradient est réalisé à l'aide d'une procédure de différences finies. Dans ce cas, il faut ajouter une boucle interne de calculs EF dans la phase analyse.

Malgré le coût élevé des réévaluations EF, la méthode du gradient apporte une amélioration importante du point de vue de la convergence et de la précision, par conséquent, son utilisation est largement justifiée dans la pratique. Parfois, on peut dire que le gradient est plus important que la solution elle-même puisqu'il indique la direction des améliorations. Mais cette méthode a ses limites, notamment au niveau de la recherche de l'optimum global que l'on n'est pas assuré de trouver.

* Sensibilité s'exprime comme la dérivée totale d'une fonction ou d'une grandeur par rapport aux paramètres

Notre contribution

Notre travail prolonge l'analyse de sensibilité, va au-delà du calcul du gradient. Nous proposons de libérer la boucle d'optimisation de tout calcul EF. L'originalité de l'approche consiste donc à faire une importante analyse initiale dans l'espace des paramètres, puis par une valorisation directe (sans la procédure EF) de proposer des réponses à l'utilisateur. De cette manière, il pourra effectuer les diverses adaptations de paramètres rapidement et à son goût. Cela autorise par exemple l'utilisation d'algorithmes stochastiques pour localiser l'optimum global puis l'enchaînement d'algorithmes déterministes pour finaliser cette recherche.

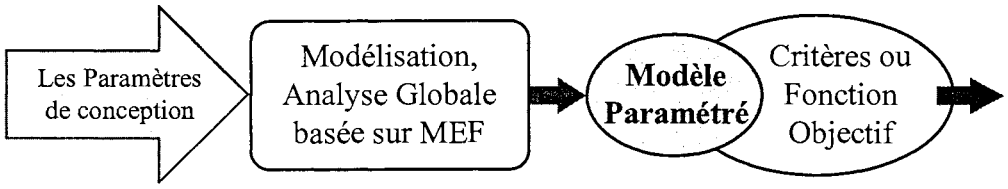


Fig.1.4. Optimisation par Modèle paramétré

Nous voulons donc effectuer une analyse de sensibilité d'ordre élevé par rapport à tous les paramètres de conception. Ceci afin de construire une approximation précise, un développement de Taylor d'ordre supérieur, de toutes les grandeurs EF (et notamment de la fonction objectif). En présence de ce développement de Taylor, la réponse est obtenue instantanément par une simple évaluation polynomiale (de degré **n**) en $p_0+\delta p$.

$$\mathbf{F}(p_0+\delta p)=\mathbf{F}(p_0)+\frac{\delta p}{1!}\frac{\partial \mathbf{F}}{\partial p}(p_0)+\frac{\delta p^2}{2!}\cdot\frac{\partial^2 \mathbf{F}}{\partial p^2}(p_0)+\cdots+\frac{\delta p^n}{n!}\cdot\frac{\partial^n \mathbf{F}}{\partial p^n}(p_0) \tag{1.22}$$

D'ores et déjà, on comprend que la réalisation soit très difficile et qu'elle suppose des interventions importantes dans la chaîne de calcul EF.

1.5. Présentation de la famille des logiciels FLUX

Bien que les idées et les approches théoriques de la MEF soient universelles, les implantations peuvent varier. Fréquemment, un développement s'appuie sur les outils existants (et disponibles) et par conséquent, il possède quelques particularités.

En ce qui concerne notre étude, la mise en oeuvre informatique a été réalisée dans l'environnement des logiciels FLUX, essentiellement de FLUX3D. Il convient donc de présenter cette plate-forme de travail.

a. Contexte et objectifs

Au début des années 80 une équipe de recherche du LEG a commencé la construction de l'un des premiers logiciels EF en électrotechnique. Ensuite, pendant près de 20 ans, les concepts EF ont été concrétisés dans un ensemble de logiciels ayant le nom commun FLUX et dont la commercialisation est gérée par CEDRAT S.A. (France) et MAGSOFT Co. (USA) à travers le monde. Les FLUX sont dédiés aux applications électrotechniques surtout aux analyses des problèmes électromagnétiques dans les appareillages électriques. Dans cette famille on distingue deux logiciels EF principaux FLUX2D et FLUX3D, FLUX3D étant la version orientée vers les problèmes 3D.

b. Particularités de FLUX3D

+ ***Base de données:*** FLUX3D est né avec l'apparition de **Bib-BD**, une bibliothèque d'opérations élémentaires. La base de données FLUX3D est très structurée grâce au concept d'entités, qui sont de type "classe d'objet" (bien qu'écrite exclusivement en FORTRAN). Toutes les entités sont donc distribuées sur un arbre de relations. Les outils de la bibliothèque permettent de modifier ou d'ajouter des entités (au niveau du modèle), ils assurent aussi le contrôle opérationnel de l'ensemble.

+ ***Descripteur géométrique:*** FLUX3D utilise le modèle de représentation par frontières (B-Rep). Ce modèle est enrichi par les concepts de repères, de transformations géométriques, de

construction automatique, etc.. Le paramétrage de la géométrie et la réévaluation automatique sont les points forts de ce module.

+ ***Mailleur:*** Le maillage libre de type Delaunay est guidé par la discrétisation des entités frontières. Les assistances possibles sont la densité des noeuds au voisinage des points, la taille des éléments distribués auprès des lignes. On dispose aussi de quelques outils pour vérifier et pour améliorer la qualité du maillage.

+ ***Problème physique:*** Dans FLUX3D, les formulations électromagnétiques et la formulation de la conduction thermique sont disponibles.

+ ***Résolution:*** L'assemblage est réalisé par l'intégration numérique de Gauss sur les EFs. Les résolutions matricielles linéaires sont effectuées par la méthode ICCG*(angl: Incomplete Cholesky Conjugate Gradient) et les itérations non-linéaires par la méthode de Newton-Raphson. Les résolutions partielles ou résolutions enchaînées sont également possibles.

+ ***Exploitation:*** C'est l'autre point fort de FLUX3D, ce module est spécialisé pour les traitements électromagnétiques, où l'on peut calculer toutes les grandeurs électriques globales ou locales du problème et les visualiser en plusieurs modes graphiques.

Divers:

◊ Deux modes d'exécution sont prévus: le mode normal graphique interactif et le mode "silence textuel", qui donne la possibilité d'exécution en "batch" (UNIX).

◊ FLUX3D possède un langage de commande, qui permet de lancer automatiquement une partie des opérations ou la totalité du calcul.

◊ A part ce langage, un nouveau concept de langage HE a été implanté dans la phase d'assemblage matriciel. Il accélère la procédure et donne à l'utilisateur la possibilité de créer/modifier facilement les formulations physiques [Coul95i].

c. Et la limite

Bien que FLUX3D soit une plate-forme complète et performante, son corps a été écrit en FORTRAN dans les années 80. Il possède donc les limites de son époque: sa structure paraît trop complexe, l'accès à la base de données est assez lourd, l'interface homme-machine trop sophistiquée pour un utilisateur moyen.

Bilan de ce chapitre introductif

Nous avons présenté succinctement la méthode des éléments finis, en distinguant les étapes d'utilisation et l'organisation typique des logiciels la mettant en oeuvre. Nous avons aussi montré quelques aspects du calcul EF en électrotechnique. L'organisation interne du calcul EF et les notations principales ont été expliquées sur une formulation particulière.

Nous avons abordé ensuite les axes de recherche concernant la méthode des éléments finis, nous avons donc dégagé les nouveaux besoins de développement et présenté les approches pour les résoudre. Enfin, dans l'intention d'implanter nos développements dans un code de calcul existant, nous avons présenté FLUX3D, un logiciel EF dédié aux problèmes électromagnétiques.

Nous avons défini l'objectif précis de notre travail: nous voulons réaliser une analyse de sensibilité d'ordre élevé au sein de calculs EF. Après une présentation générale, dans le chapitre suivant, de la méthode mise en oeuvre, nous détaillerons toutes les étapes de la réalisation dans les chapitres suivants et nous conclurons par quelques applications pratiques.

NGUYEN Thanh Nam

LEG

Chapitre 2

L'ANALYSE DE SENSIBILITÉ D'ORDRE ÉLEVÉ
UNE APPROCHE POLYNOMIALE DES SOLUTIONS EF

Sommaire

Chapitre 2

L'ANALYSE DE SENSIBILITÉ D'ORDRE ÉLEVÉ - UNE APPROCHE POLYNOMIALE DES SOLUTIONS EF

Nous avons signalé dans le premier chapitre que l'analyse par la MEF est devenue indispensable dans plusieurs domaines techniques. Nous avons remarqué aussi que le calcul EF classique ne suffit pas entièrement pour le traitement de problèmes inverses ou d'optimisation. Il y a eu, par le passé, de nombreux efforts pour adapter la MEF à ces calculs et aussi pour accélérer la boucle d'optimisation. Nous avons noté que l'analyse de sensibilité est un outil performant pour résoudre ce problème.

Nous proposons ici de prolonger l'analyse de sensibilité, pour séparer l'optimisation du calcul EF, afin de donner aux utilisateurs plus de souplesse et plus de liberté pour leurs applications. Plus exactement, nous voulons effectuer une analyse de sensibilité d'ordre élevé afin de construire le développement de Taylor pour toutes les grandeurs EF (champ, force, induction, ...).

Nous décrivons dans ce chapitre, les particularités d'une analyse de sensibilité d'ordre élevé, ainsi que les considérations générales pour sa mise en oeuvre. Nous faisons attention tout particulièrement au traitement des paramètres géométriques, dont les réalisations font l'objet des chapitres ultérieurs.

2.1. Considérations Générales

Nous présentons ici le contexte d'une utilisation efficace des calculs EF et aussi quelques considérations, qui justifient l'utilisation de notre méthode de dérivées d'ordre élevé.

2.1.1. Organisation des calculs EF

Nous avons vu que l'utilisation de calculs EF est coûteuse. Il faut cependant distinguer le temps mis par l'utilisateur pour décrire le problème et le temps pour résoudre les grands systèmes matriciels EF, c'est-à-dire le temps de calcul. En pratique, les améliorations sont consacrées à la diminution du temps de calcul. Naturellement, on pourrait disserter sur les machines plus puissantes, on pourrait utiliser les algorithmes vectoriels ou parallèles, etc.. Mais si cela n'est pas toujours possible, il faut penser à une organisation plus efficace des calculs. Traditionnellement, la résolution est traitée en "batch", c'est-à-dire que l'utilisateur la lance en tâche de fond ou la nuit, et qu'il récupère les résultats plus tard.

Le problème qui se pose avec une procédure d'optimisation, c'est que les calculs sont répétés plusieurs fois en intercalant l'exploitation de critères d'optimisation. On pourrait donc automatiser toute la procédure de calcul à l'aide d'une fonction objectif et de contraintes de pénalisation. Mais dans la pratique, cette approche est très contraignante: elle se base sur des hypothèses fortes sur la fonction objectif, elle utilise des contraintes de pénalisation qui sont généralement discutables, et le savoir-faire de l'utilisateur n'a pas vraiment sa place dans cette procédure.

2.1.2. Méthodes d'optimisation

Les méthodes d'optimisation sont très nombreuses. On peut toutefois les classer en deux grandes familles: les méthodes déterministes et les méthodes stochastiques. Les premières sont en général des méthodes efficaces, peu coûteuses mais qui nécessitent un point de départ "proche" de la solution optimale. Elles permettent l'optimisation locale. À l'opposé, les méthodes stochastiques explorent tout l'espace des solutions grâce, en partie, à des

mécanismes de transitions aléatoires. Leur intérêt réside dans leur capacité à trouver l'optimum global. Par contre, leur convergence est lente et demande habituellement un nombre très important d'évaluations de la fonction objectif.

Avec les méthodes EF classiques, les algorithmes déterministes sont préférables, parce que plus efficace. Mais, lorsque le problème possède plusieurs optima locaux, le choix du bon point de départ, permettant de localiser l'optimum global, devient très difficile. On a alors souvent recours à un algorithme stochastique pour la localisation de l'optimum global. Cependant, ce dernier couplage risque de perdre de son intérêt du fait de grand nombre d'évaluations nécessaire et du coût important des calculs EF qui en résulte.

2.1.3. Utilisation des dérivées d'ordre élevé

Un des points importants dans ce travail est l'utilisation des dérivées d'ordre supérieur. Nous nous sommes intéressés ici uniquement aux dérivées de la solution. Nous n'avons pas exploré la voie consistant à déterminer les dérivées à l'aide de nouvelles formulations (la dérivation des formulations, en quelque sorte). Les dérivées sont donc calculées à l'aide d'une procédure de dérivation numérique du système d'équations, c'est-à-dire qu'il faut introduire des nouveaux modules dérivateurs dans la procédure de calcul EF classique (Fig.2.1).

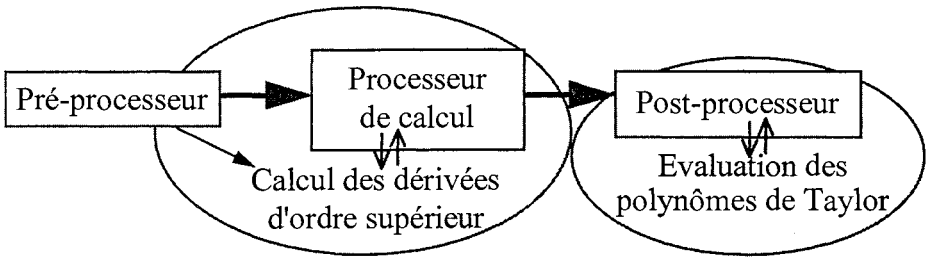


Fig.2.1. Première présentation d'une l'analyse de sensibilité d'ordre élevé

Dans l'analyse de sensibilité traditionnelle, la limitation des dérivées aux ordres 1 ou 2, s'explique par le coût supposé de la dérivation et aussi par la complexité d'obtention des dérivées supérieures. Les résultats théoriques de Ph. Guillaume [Guill94, GuiMa94] montrent que le calcul des dérivées par la dérivation de la matrice EF n'est pas trop coûteux et qu'il est aussi précis que le calcul de la fonction elle-même. Par conséquent, à partir d'une seule analyse,

l'approximation de la grandeur d'état (et de la fonction objectif) par son polynôme de Taylor est envisageable.

En utilisant les dérivées d'ordre élevé, il se pose cependant le problème du domaine de validité de la méthode, lié au rayon de convergence de la série de Taylor. Les premières implantations de cette méthode ont été effectuées par CADOE-CEDRAT [CADOE, CEDRAT] et par P.Pétin [Petin96]. Elles ont montré que les résultats numériques obtenus étaient très encourageants.

Notre approche consiste à faire une analyse de sensibilité d'ordre élevé afin de construire le polynôme d'approximation de la solution par rapport à tous les paramètres. Le calcul de plusieurs dérivées des variables nodales peut être assez long. Notons que le parallélisme est très efficace dans ce type de calcul, et qu'il peut être effectué, une fois pour toutes, en "batch" afin d'économiser le temps de l'utilisateur.

Dans tous les cas, l'évaluation polynomiale (très rapide) favorise l'utilisation des algorithmes stochastiques dans une optimisation globale. Nous espérons que la validité de chaque développement de Taylor obtenu sera assez étendue pour détecter et franchir les extrêma locaux. Si ce n'est pas le cas, nous espérons que le nombre des reconstructions nécessaires dans une phase d'optimisation, sera réduit.

2.2. Méthode des dérivées d'ordre élevé.

La méthode proposée est générale, mais pour alléger la présentation, nous nous contenterons d'expliquer son principe dans le cas d'un problème magnétodynamique vectoriel 2D linéaire (c'est-à-dire sans saturation), dont la formulation a été décrite dans le premier chapitre de ce mémoire. Reprenons le système matriciel associé au problème - l'équation (1.13):

(2.1) $$[\mathbf{M}]\cdot\{\mathbf{A}\} = [\mathbf{P} + j\mathbf{Q}]\cdot\{\mathbf{A}\} = \{\mathbf{S}\}$$

Soit **p** un paramètre du problème, on suppose que l'équation d'état (2.1) correspondant à la valeur $\mathbf{p}=\mathbf{p}_0$, a été résolu pour $\mathbf{A}(p_0)$.

$$[\mathbf{M}(p_0)]\cdot\{\mathbf{A}(p_0)\} = \{\mathbf{S}(p_0)\}$$

Habituellement, il faut construire et résoudre une nouvelle équation correspondant à une nouvelle valeur ($p_0+\delta p$) du paramètre

(2.2) $$[\mathbf{M}(p_0+\delta p)]\cdot\{\mathbf{A}(p_0+\delta p)\} = \{\mathbf{S}(p_0+\delta p)\}.$$

On se propose ici d'approcher la solution $\mathbf{A}(p_0+\delta p)$ du système (2.2) par son polynôme de Taylor d'ordre supérieur (limité à Np).

(2.3) $$\mathbf{A}(p_0+\delta p) \cong \mathbf{A}(p_0) + \frac{\delta p}{1!}\frac{\partial \mathbf{A}}{\partial p}(p_0) + \cdots + \frac{\delta p^{N_p}}{N_p!}\cdot\frac{\partial^{N_p}\mathbf{A}}{\partial p^{N_p}}(p_0)$$

$$\mathbf{A}(p_0+\delta p) \cong \sum_{i=0}^{N_p}\frac{\delta p^i}{i!}\cdot\frac{\partial^i \mathbf{A}}{\partial p^i}(p_0)$$

En effectuant une seule analyse centrée en p_0, on connaîtra la solution dans un voisinage du paramètre initial p_0. Nous renvoyons à [GuiMa94] et [Guill94] pour les aspects théoriques mathématiques et pour les premières estimations de l'erreur de l'approximation.

2.2.1. Calcul des dérivées d'ordre supérieur de la solution

Grâce au principe de stationnarité, nous pouvons dériver le système (2.1) par rapport au paramètre **p** et donc obtenir une nouvelle équation afin de calculer la première dérivée du vecteur d'état $\{\mathbf{A}\}$ au point $\mathbf{p}_0$.

(2.4) $$[\mathbf{M}(p_0)]\cdot\left\{\frac{\partial \mathbf{A}}{\partial p}(p_0)\right\}=\left\{\frac{\partial \mathbf{S}}{\partial p}(p_0)\right\}-\left[\frac{\partial \mathbf{M}}{\partial p}(p_0)\right]\cdot\{\mathbf{A}(p_0)\}$$

Remarquons que le système (2.4) comporte au premier membre, la même matrice [**M**] que celle de système (2.1). Il suffit donc de reconstruire le second membre, puis de relancer la résolution. D'ailleurs en utilisant le même principe, on peut calculer récursivement la dérivée d'ordre quelconque de la solution {**A**}.

(2.5) $$\mathbf{M}\cdot\frac{\partial^m \mathbf{A}}{\partial p^m}=\frac{\partial^m \mathbf{S}}{\partial p^m}-\sum_{i=0}^{m-1} C_m^i \frac{\partial^{m-i}\mathbf{M}}{\partial p^{m-i}}\cdot\frac{\partial^i \mathbf{A}}{\partial p^i} \quad \text{avec } C_m^i=\frac{m!}{i!(m-i)!} \quad \text{et } (0!=1)$$

Enfin, ce calcul est bien généralisable au problème à plusieurs paramètres. Par exemple, la dérivée croisée de deux paramètres **p** et **q** est calculée par l'équation récursive:

(2.6) $$\mathbf{M}\cdot\frac{\partial^{m+n}\mathbf{A}}{\partial p^m \partial q^n}=\frac{\partial^{m+n}\mathbf{S}}{\partial p^m \partial q^n}-\sum_{i=0}^{m}\sum_{j=0}^{(i\#m)\,n-1} C_m^i C_n^j \frac{\partial^{m-i+n-j}\mathbf{M}}{\partial p^{m-i}\partial q^{n-j}}\cdot\frac{\partial^{i+j}\mathbf{A}}{\partial p^i \partial q^j}.$$

Et le polynôme de Taylor s'écrit donc dans ce cas:

(2.7) $$\mathbf{A}(p_0+\delta p, q_0+\delta q)=\sum_{i=0}^{N_p}\sum_{j=0}^{N_q}\frac{\delta p^i \delta q^j}{i!\,j!}\cdot\frac{\partial^{i+j}\mathbf{A}}{\partial p^i \partial q^j}(p_0,q_0)$$

Évidement, pour des raisons d'encombrement et de coût des calculs, on ne peut pas (et on n'a pas de besoin de) calculer ni stocker toutes les dérivées de la solution {**A**}. En pratique, les ordres maximum de dérivation (N_p,N_q) sont choisis par l'utilisateur pour chaque paramètre. Ces ordres peuvent également être calculés à partir d'un critère de précision en s'appuyant sur les variations pré-définies des paramètres.

Remarques:

- La matrice [**M**] ayant été factorisée une fois pour toutes, les résolutions successives sont bien moins coûteuses en temps de calcul que la factorisation de la matrice elle-même. Cette méthode est particulièrement efficace pour les problèmes de grande taille, car le rapport *complexité de la factorisation / complexité de la résolution d'un système factorisé* augmente avec la dimension du problème.
- Dans le cas de plusieurs paramètres, la méthode proposée est parallélisable. En effet, pour un ordre de dérivation fixé, les calculs des différentes dérivées croisées par rapport aux

paramètres sont totalement indépendants. Le degré de parallélisme augmente rapidement avec le nombre de paramètres et l'ordre de dérivation.

2.2.2. Dérivation de la matrice et du second membre

Pour obtenir les dérivées de la solution, il faut résoudre plusieurs fois le même système matriciel avec différents seconds membres. Ces derniers sont construits récursivement à partir des équations (2.4), (2.5), (2.6), etc.. Pour définir ces seconds membres, il suffit donc de calculer les dérivées successives de [**M**] et de {**S**} au point initial $\mathbf{p}_0$. Cette dérivation de [**M**] et {**S**} est la tâche principale d'une analyse de sensibilité d'ordre élevé et la plus difficile. En effet, il y a de nombreux facteurs qui interviennent dans cette phase de calcul.

En premier lieu, étudions les facteurs dépendants des paramètres:

◊ Un paramètre peut intervenir dans le calcul de [**M**], dans celui de {**S**}, ou dans le calcul des deux, conjointement. L'influence d'un paramètre est fréquemment localisée, c'est-à-dire qu'il affecte seulement une partie de [**M**] et/ou de {**S**}.

◊ On peut classer les paramètres en deux grandes familles: les paramètres géométriques et les paramètres physiques. Le traitement d'un paramètre géométrique est plus complexe que celui d'un paramètre physique. Le paramètre physique n'influe que sur lui-même, tandis que le paramètre géométrique, en changeant de valeur, introduit une perturbation du maillage et une nonlinéarité en dérivation de l'équation matriciel (2.1) (voir paragraphe §2.3).

Ensuite, la dérivation concerne les matrices et les vecteurs élémentaires. En effet, la MEF se propose de construire la matrice [**M**] et le vecteur {**S**} à partir des matrices et des vecteurs élémentaires:

$$[\mathbf{M}] = \mathbf{A}\Big\|[\mathbf{M}_e] \quad \text{et} \quad \{\mathbf{S}\} = \mathbf{A}\Big\|\{\mathbf{S}_e\} \text{ avec } \mathbf{A}\Big\| \text{ est l'opérateur d'assemblage EF.}$$

L'associativité de la dérivation nous permet donc d'écrire:

(2.8) $$d\mathbf{M} = d(\mathbf{A}\Big\|\mathbf{M}_e) = \mathbf{A}\Big\| d\mathbf{M}_e$$

et $$d\mathbf{S} = d(\mathbf{A}\Big\|\mathbf{S}_e) = \mathbf{A}\Big\| d\mathbf{S}_e$$

Il existe deux principales méthodes pour dériver la matrice élémentaire et le vecteur élémentaire:

⇒ Méthode semi-analytique: Il faut calculer les formules analytiques des intégrales associées à chaque élément en utilisant effectivement l'algèbre symbolique. La procédure de dérivation est réalisée spécifiquement pour chaque type d'éléments finis et pour chaque formulation physique en s'appuyant sur une procédure d'intégration numérique. Une simulation indépendante est souhaitable pour automatiser toute la procédure de dérivation.

⇒ Méthode de différenciation automatique: En utilisant la technique de différenciation automatique (DA), le calcul de dérivées peut être réalisé à partir du code de calcul de la fonction (ou de l'intégrale). Il faut donc coupler le programme EF avec un différenciateur automatique (existant), ce travail ne comporte que des difficultés informatiques. Nous remarquons que c'est une approche performante, bien qu'elle exige une localisation précise des codes concernés.

Finalement, en ce qui concernent la mise en oeuvre pratique: il faut d'abord identifier toutes les régions et tous les EFs concernés; puis, calculer par la méthode choisie les dérivées sur les éléments identifiés; ensuite, réaliser tous les assemblages nécessaires; enfin, boucler la résolution pour obtenir la dérivée de la solution.

Les points caractéristiques et les exemples de la dérivation de matrices et de vecteurs élémentaires seront présentés dans le paragraphe suivant, après une présentation des types de paramètres utilisés dans un problème EF.

Pour résumer, nous proposons la structure complète d'une étude de sensibilité d'ordre élevé (Fig.2.2). Dans cette structure, nous remarquons trois blocs caractéristiques:

◊ *Au départ, il existe une chaîne de calculs EF.*

◊ *On ajoute* ***dans le calcul EF*** *des outils pour obtenir les dérivées d'ordre élevé de toutes les grandeurs désirées.*

◊ *Finalement, on propose à l'utilisateur la solution EF sous la forme de polynômes de Taylor, qui peut être considérée comme* ***un modèle équivalent paramétré.***

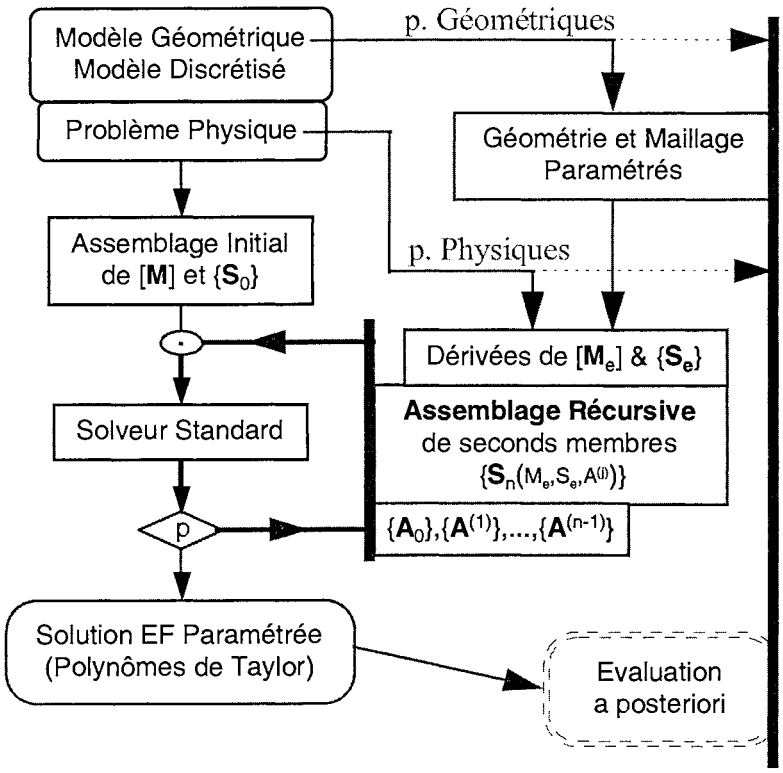


Fig.2.2. Organisation d'une étude de sensibilité d'ordre élevé.

2.3. Paramètres dans un problème EF

Nous avons jusqu'à présent, décrit une méthode de façon générale, en parlant de paramètres. Il est temps de donner quelques précisions sur cette notion et aussi sur leur utilisation dans un problème EF.

Un paramètre sera donc une grandeur qui peut prendre ses valeurs dans un ensemble prédéfini, cette grandeur sert à la définition du problème EF.

2.3.1. Classement des paramètres

Les paramètres se regroupent, suivant leur influence dans la chaîne de calculs EF, en deux grandes familles: les paramètres géométriques et les paramètres physiques.

a. *Paramètres physiques*

C'est le type de paramètre fixé par le problème physique. On les définit formellement comme les paramètres qui n'influent pas sur le maillage.

Suivant le choix d'une formulation ils peuvent être: la fréquence ou la vitesse angulaire (f,ω), les propriétés physiques des matériaux (réluctivité ν, permittivité électrique ε, conductivité σ, etc.), les contraintes (flux, tension, courant ou densité de courant source, etc.), certaines conditions aux limites (type Diriclet par exemple). Dans une formulation, le nombre de paramètres physiques est généralement assez limité, leur définition est donc claire et unique.

Un caractère significatif est le mode de la présentation des paramètres physiques. On distingue la présentation externe, le paramètre "utilisateur" et la présentation interne qui est préférable pour le calcul EF. Par exemple, l'utilisateur peut s'intéresser à une perméabilité, alors qu'en interne, il est équivalent et peut-être plus facile de travailler avec son inverse, la réluctivité. Un autre exemple est le remplacement du courant total par la densité de courant dans les formulations magnétiques.

b. *Paramètres géométriques*

Naturellement, ce sont les paramètres qui définissent le modèle géométrique. Il y a deux types de paramètres géométriques de nature très différente.

- ***Les Paramètres topologiques*** sont les paramètres discrets, qui sont entiers par nature (exemple: nombre de pôles, nombre d'encoches, nombre de bobines ou nombre de canaux de refroidissement dans une machine tournante ou un transformateur). Leur définition est aussi claire et unique que celle des paramètres physiques.

  Ces paramètres, en changeant de valeur, introduisent un changement dans la topologie géométrique et une discontinuité dans le maillage (remaillage obligatoire).

- ***Les Paramètres de dimension ou Paramètres de mise en forme:*** Contrairement aux paramètres topologiques, ces paramètres font changer les dimensions ou/et la forme des régions mais ils ne font pas changer la topologie de la géométrie. C'est-à-dire que leur modification n'entraîne ni addition, ni suppression de régions géométriques, ni cassure des frontières.

  Ce sont les paramètres de dimension: longueur, largeur, hauteur, épaisseur, etc..

  Ce sont aussi les paramètres de mise en forme: position relative des entités, angle d'un arc, aire, volume, etc..

  Les paramètres de dimension peuvent prendre des valeurs dans un intervalle continu, on dit qu'ils sont de type "continu". On remarque donc une caractéristique de ces paramètres: bien que chaque paramètre porte une définition précise et unique, la définition de l'ensemble n'est pas toujours unique. Par exemple, on peut décrire le même segment de droite par la position des deux extrémités ou par un support, une extrémité et une longueur. Contrairement au cas des paramètres physiques, il est quasiment impossible de détecter automatiquement qu'une présentation est "préférable", ou "optimale" par rapport aux autres.

Il est évident que les paramètres topologiques discrets ne peuvent pas être traités par notre méthode. Dans la suite de ce travail, le terme "paramètre géométrique" signifiera exclusivement "paramètre de dimension".

Enfin, nous voudrions y ajouter une nuance: Il y a dans la classe des paramètres géométriques continus, des paramètres "discrets" par le choix normatif du concepteur. Par exemple, le

diamètre des fils de cuivre, l'épaisseur des tôles magnétiques sont choisis dans une gamme "discrète". On utilise des produits standards, beaucoup moins chers. Notons qu'un polynôme permet de traiter ces paramètres dans leur forme discrète.

2.3.2. Dérivation par rapport aux paramètres physiques

Nous reprenons les intégrales élémentaires du problème magnétodynamique 2D à partir des équations (1.14), (1.15), (1.16).

(2.9) $$[\mathbf{P}_e] = \iint_{\Omega_e} \nu.[\nabla\alpha][\nabla\alpha]^T \, d\Omega$$

(2.10) $$[\mathbf{Q}_e] = \iint_{\Omega_e} \omega\sigma.\{\alpha\}\{\alpha\}^T \, d\Omega$$

(2.11) $$\{\mathbf{S}_e\} = \iint_{\Omega_e} J_s.\{\alpha\} \, d\Omega$$

Dans cette formulation, il y a trois types de paramètres physiques ("internes"): réluctivité ν, densité de courant normal au plan étudié J_s, et conductivité complexe $\omega\sigma$. Ces paramètres sont indépendants et la dérivation ne présente plus de grande difficulté. Remarquons que les dérivées d'ordre supérieur par rapport à ces trois paramètres n'existent pas dans le cas linéaire.

Pour le paramètre "réluctivité ν_i" d'une région, les dérivées $\partial\mathbf{Q}_e$ et $\partial\mathbf{S}_e$ sont nulles, il suffit de calculer les termes $\partial\mathbf{P}_e$ et seulement pour les éléments de la région concernée:

(2.12) $$\frac{\partial}{\partial\nu_i}[\mathbf{P}_{ei}] = \iint_{\Omega_{ei}} [\nabla\alpha][\nabla\alpha]^T \, d\Omega$$

Pour la conductivité complexe $\omega\sigma_j$ (un bon exemple de paramètre "interne") il faut calculer seulement les $\partial\mathbf{Q}_e$ sur les éléments de la région de conduction:

(2.13) $$\frac{\partial}{\partial(\omega\sigma_j)}[\mathbf{Q}_{ej}] = \iint_{\Omega_{ej}} \{\alpha\}\{\alpha\}^T \, d\Omega$$

Pour la densité de courant J_{sk} qui est distribuée régulièrement dans une région, il faut calculer les $\partial\mathbf{S}_e$, toutes les dérivées correspondantes $\partial\mathbf{P}_e$ et $\partial\mathbf{Q}_e$ étant nulles.

(2.14) $$\frac{\partial}{\partial J_{sk}}\{\mathbf{S}_{ek}\} = \iint_{\Omega_{ek}} \{\alpha\} \, d\Omega$$

Comme les dérivées supérieures n'existent pas, il n'y a pas de dérivées croisées entre les paramètres physiques de cette formulation.

Enfin, il est clair que les formules (2.12), (2.13), (2.14) sont basées sur des hypothèses fortes de dérivation, en particulier la linéarisation des phénomènes physiques. En réalité, lorsque la solution dépend de la réluctivité dans un problème magnétique non-linéaire ou lorsque la densité de courant change par effet de peau ou par courant induit, etc., le calcul de dérivées devient très complexe. Il faut donc revenir à la base de la méthode, les équations (2.2), (2.3), (2.4) et intervenir fortement dans l'organisation des calculs. Notons qu'une première étude sur la non linéarité a été réalisée par P.Pétin [Petin96], une autre étude sur le développement de Padé est réalisée à CEDRAT/CADOE afin de proposer une nouvelle représentation de la solution non-linéaire [CEDRAT].

2.3.3. Dérivation par rapport aux paramètres géométriques

Le maillage en EFs est la base de la MEF et les paramètres géométriques en déformant les EFs influent sur les intégrales élémentaires.

Pour pouvoir effectuer la dérivation par rapport aux paramètres géométriques, il existe quelques conditions à respecter. De plus, certaines circonstances permettent d'en alléger effectivement le traitement.

◊ ***Exigence d'une topologie constante du maillage.***

Si on modifie un paramètre géométrique, il faut garder la topologie du maillage initial. La modification doit se propager seulement sur les noeuds et les éléments du problème. On parle alors de perturbation de maillage. La raison est évidente: il faut garder la structure de [M] et {S} afin de pouvoir les dériver.

◊ ***Réduction du nombre d'éléments à traiter***

En supposant que les caractéristiques physiques restent inchangées, on peut limiter la dérivation sur les éléments réellement déformés, car l'intégration est inchangée sur les éléments qui sont déplacés sans déformation.

◊ ***Influence sur toutes les intégrales élémentaires***

Le paramètre géométrique intervient dans toutes les intégrales de la formulation active.

Dans l'exemple présenté, il faut dériver conjointement les trois formules (2.9), (2.10) et (2.11) pour tous les éléments déformés.

Dans la suite de ce paragraphe, nous détaillerons quelques méthodes de dérivation par rapport aux paramètres géométriques.

a. *Dérivation directe*

Prenons le cas de la magnétodynamique 2D avec un maillage en triangles du premier ordre.

Les fonctions de forme $\alpha_i(x,y)$ dans un triangle $\Delta(N_1,N_2,N_3):N_i(x_i,y_i,z_i)$ s'expriment selon:

(2.15a) $$\alpha_1(x,y) = \frac{1}{2S_\Delta}\,[(x_2y_3-x_3y_2) + (y_2-y_3).x + (x_3-x_2).y]$$

(2.15b) $$\alpha_2(x,y) = \frac{1}{2S_\Delta}\,[(x_3y_1-x_1y_3) + (y_3-y_1).x + (x_1-x_3).y]$$

(2.15c) $$\alpha_3(x,y) = \frac{1}{2S_\Delta}\,[(x_1y_2-x_2y_1) + (y_1-y_2).x + (x_2-x_1).y]$$

où S_Δ est l'aire de triangle:

(2.16) $$2S_\Delta = (x_2y_3-x_3y_2) + (x_3y_1-x_1y_3) + (x_1y_2-x_2y_1)$$

$\mathbf{S_\Delta} = S1+S2+S3$

$\alpha_{1N} = S1/\mathbf{S_\Delta}$

$\alpha_{2N} = S2/\mathbf{S_\Delta}$

$\alpha_{3N} = S3/\mathbf{S_\Delta}$

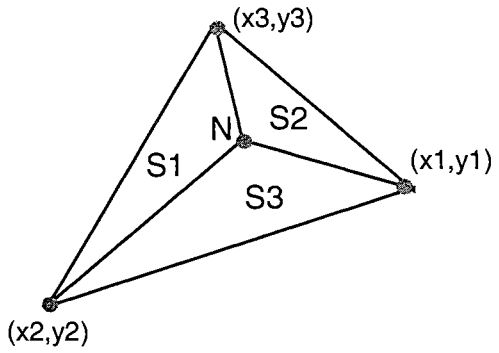


Fig.2.3. Le triangle et ses fonctions d'interpolation du premier ordre

Les intégrales (2.11) et (2.10) peuvent être calculées directement à l'aide des fonctions $\alpha_i(x,y)$:

(2.17) $$\{S_e\} = \iint_{\Omega_e} J_s.\{\alpha\}\ dxdy = J_s\,\frac{S_\Delta}{3}\begin{Bmatrix}1\\1\\1\end{Bmatrix}$$

(2.18) $$[\mathbf{Q}_e]=\iint_{\Omega_e}\omega\sigma.\{\alpha\}\{\alpha\}^T dxdy=\omega\sigma\frac{S_\Delta}{12}\begin{bmatrix}2&1&1\\1&2&1\\1&1&2\end{bmatrix}$$

Pour la matrice $[P_e]$, nous remarquons que:

(2.19) $$[\nabla\alpha]=\frac{1}{2S_\Delta}\left[\begin{array}{c|c}y_2-y_3 & x_3-x_2\\ y_3-y_1 & x_1-x_3\\ y_1-y_2 & x_2-x_1\end{array}\right]=\frac{1}{2S_\Delta}\left[\{\beta_x\}\middle|\{\beta_y\}\right]$$

En utilisant (2.19), l'intégrale (2.9) est calculée comme:

(2.20) $$[\mathbf{P}_e]=\iint_{\Omega_e}\nu.[\nabla\alpha][\nabla\alpha]^T dxdy=\iint_{\Omega_e}\frac{\nu}{4S_\Delta^2}\left[\{\beta_x\}\{\beta_x\}^T+\{\beta_y\}\{\beta_y\}^T\right]dxdy$$
$$=\frac{\nu}{4S_\Delta}\left[\{\beta_x\}\{\beta_x\}^T+\{\beta_y\}\{\beta_y\}^T\right]$$

Les équations (2.17), (2.18) et (2.20) nous permettent de passer de la dérivation d'intégrales élémentaires (2.9), (2.10), (2.11) à la dérivation de "S_Δ", de "1/ S_Δ" et de certain polynôme β.

(2.21) $$\frac{\partial^{(m_1+m_2+\ldots+m_n)}}{\partial p_1^{m_1}\partial p_2^{m_2}\ldots\partial p_n^{m_n}}\{\mathbf{S}_e\}=\frac{J_s}{3}\begin{Bmatrix}1\\1\\1\end{Bmatrix}\frac{\partial^{(m_1+m_2+\ldots+m_n)}S_\Delta}{\partial p_1^{m_1}\partial p_2^{m_2}\ldots\partial p_n^{m_n}}$$

(2.22) $$\frac{\partial^{(m_1+m_2+\ldots+m_n)}}{\partial p_1^{m_1}\partial p_2^{m_2}\ldots\partial p_n^{m_n}}[\mathbf{Q}_e]=\frac{\omega\sigma}{12}\begin{bmatrix}2&1&1\\1&2&1\\1&1&2\end{bmatrix}\frac{\partial^{(m_1+m_2+\ldots+m_n)}S_\Delta}{\partial p_1^{m_1}\partial p_2^{m_2}\ldots\partial p_n^{m_n}}$$

(2.23) $$\frac{\partial^{(m_1+m_2+\ldots+m_n)}}{\partial p_1^{m_1}\partial p_2^{m_2}\ldots\partial p_n^{m_n}}[\mathbf{P}_e]=\frac{\nu}{4}\frac{\partial^{(m_1+m_2+\ldots+m_n)}}{\partial p_1^{m_1}\partial p_2^{m_2}\ldots\partial p_n^{m_n}}\left(\frac{\{\beta_x\}\{\beta_x\}^T+\{\beta_y\}\{\beta_y\}^T}{S_\Delta}\right)$$

Cette méthode est très efficace pour les cas simples comme celui-ci, mais il faut réaliser le calcul spécifiquement pour chaque formulation et pour chaque type d'éléments finis (triangle, quadrilatère, ..., ordre 1, ordre 2), ce qui alourdit considérablement son usage

b. *Dérivation de la matrice de transformation géométrique.*

En pratique, toutes les intégrations qui découlent de la formulation intégrale du problème, sont ramenées sur des éléments de références. Naturellement, les éléments de références ne dépendent pas des paramètres, la dérivation est donc concentrée sur la fonction de transformation géométrique local-global des éléments.

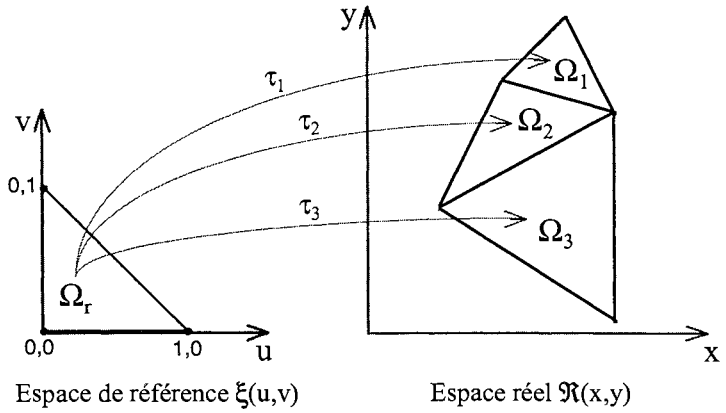


Fig.2.4. Le triangle de référence et la transformation géométrique local-global

On rappelle que le changement de variable $\xi(u,v) \rightarrow \Re(x,y)$ permet de passer de l'intégration d'une fonction f sur l'élément réel Ω_e à une intégrale (plus simple) sur l'élément de référence Ω_r. Par exemple, l'intégrale d'assemblage du vecteur $\{S\}$ peut être écrite:

(2.24) $$\{\mathbf{S}_e\} = \iint_{\Omega_e} J_s \cdot \{\alpha(x,y)\}\, dxdy = \iint_{\Omega_r} J_s \cdot \{\alpha(u,v)\} \cdot |G|\, dudv$$

où [G] est la matrice jacobienne de la transformation géométrique ci-dessus,

et |G| étant son déterminant.

Comme dans (2.24) |G| est le seul terme qui dépend du paramètre géométrique **p**, la première dérivée de $\{\mathbf{S}\}$ s'écrit:

(2.25) $$\frac{\partial}{\partial p}\{\mathbf{S}_e\} = \iint_{\Omega_r} J_s . \{\alpha(u,v)\} . \frac{\partial |G|}{\partial p}\, dudv$$

Ce calcul doit être réalisé exclusivement pour les éléments déformés par la modification de **p**. La formule généralisée pour calculer une dérivée d'ordre arbitraire supérieur par rapport aux paramètres géométriques $p_1, p_2, \ldots, p_n$ est donnée par:

(2.26) $$\frac{\partial^{(m_1+m_2+\ldots+m_n)}}{\partial p_1^{m_1} \partial p_2^{m_2} \ldots \partial p_n^{m_n}} \{\mathbf{S}_e\} = \iint_{\Omega_r} J_s . \{\alpha(u,v)\} . \frac{\partial^{(m_1+m_2+\ldots+m_n)} |G|}{\partial p_1^{m_1} \partial p_2^{m_2} \ldots \partial p_n^{m_n}}\, dudv$$

La dérivation de la matrice $[\mathbf{Q}_e]$ est similaire à celle du vecteur $\{\mathbf{S}_e\}$, nous étudions alors le cas plus complexe de $[\mathbf{P}_e]$. La difficulté se pose en présence de gradients $\{\nabla\alpha\}$ sous l'intégrale (2.9). Notons que la transformation est définie pour les dérivées premières d'une fonction de forme $\alpha_i(x,y)$ selon:

$$\nabla\alpha_i = \begin{Bmatrix} \frac{\partial\alpha_i}{\partial x} \\ \frac{\partial\alpha_i}{\partial y} \end{Bmatrix} = \begin{bmatrix} \frac{\partial u}{\partial x} & \frac{\partial v}{\partial x} \\ \frac{\partial u}{\partial y} & \frac{\partial v}{\partial y} \end{bmatrix} \cdot \begin{Bmatrix} \frac{\partial\alpha_i}{\partial u} \\ \frac{\partial\alpha_i}{\partial v} \end{Bmatrix} = [G]^{-1} \nabla_u \alpha_i \quad \textbf{(2.27)}$$

où $[G]^{-1}$ est l'inverse de la matrice [G] citée ci-dessus,

$\nabla_u \alpha_i$, est gradient local sur l'élément de référence

L'expression complète des gradients dans (2.9) peut être réécrite:

$$[\nabla\alpha]\cdot[\nabla\alpha]^T = [\nabla_u\alpha]\cdot[G]^{-1T}[G]^{-1}\cdot[\nabla_u\alpha]^T \quad \textbf{(2.28)}$$

En utilisant (2.28), l'intégrale (2.9) devient:

$$[\mathbf{P}_e] = \iint_{\Omega_r} \nu.[\nabla_u\alpha]\cdot|G|\cdot[G]^{-1T}[G]^{-1}\cdot[\nabla_u\alpha]^T \, dudv \quad \textbf{(2.29)}$$

Dans cette dernière expression le seul terme, qui dépend des paramètres (de l'élément réel), est $|G|\cdot[G]^{-1T}[G]^{-1}$ alors la dérivation de $[P_e]$ est calculée par:

$$\frac{\partial^{(m_1+m_2+\ldots+m_n)}[\mathbf{P}_e]}{\partial p_1^{m_1}\partial p_2^{m_2}\ldots\partial p_n^{m_n}} = \iint_{\Omega_r} \nu.[\nabla_u\alpha]\frac{\partial^{(m_1+m_2+\ldots+m_n)}\left[|G|\cdot[G]^{-1T}[G]^{-1}\right]}{\partial p_1^{m_1}\partial p_2^{m_2}\ldots\partial p_n^{m_n}}[\nabla_u\alpha]^T \, dudv \quad \textbf{(2.30)}$$

En principe, la dérivation du terme $|G|\cdot[G]^{-1T}[G]^{-1}$ peut être réalisée en s'appuyant sur les dérivées connues des noeuds d'élément [CoPSN96].

Les équations (2.26) et (2.30) sont générales pour toutes les formulations dans lesquelles l'expression des intégrales ne dépend que des fonctions de forme. Les cas particuliers seront étudiés dans la suite de ce paragraphe.

c. *Particularités des formulations.*

Dans cette partie, on va étudier un cas particulier pour montrer les adaptations nécessaires dans l'application de la méthode ci-dessus en fonction des formulations.

Prenons le cas similaire d'une formulation magnétodynamique 2D axisymétrique. En utilisant un système de coordonnées cylindriques (r,θ,z), nous supposons que les courants et les vecteurs potentiels magnétiques n'ont qu'une composante suivant θ:

$$\vec{\mathbf{A}} = \{0, A_\theta, 0\}^T \text{ et } \vec{\mathbf{J}}_s = \{0, J_\theta, 0\}^T \quad \textbf{(2.31)}$$

L'équation de champ (1.9): $\nabla(\nu.\nabla A) + j\omega\sigma A = J$ est écrite dans le plan de projection (r,z):

(2.32) $$\left(\frac{\partial}{\partial z}(\nu\frac{\partial A}{\partial z})+\frac{\partial}{\partial r}(\frac{\nu}{r}\frac{\partial(rA)}{\partial r})\right)+j\omega\sigma.A=J$$

En utilisant comme variable d'état $\mathbf{A^*} = r.\mathbf{A}$, l'équation (2.32) devient:

(2.33) $$\left(\frac{\partial}{\partial z}(\frac{\nu}{r}\frac{\partial A^*}{\partial z})+\frac{\partial}{\partial r}(\frac{\nu}{r}\frac{\partial A^*}{\partial r})\right)+j\omega\frac{\sigma}{r}.A^*=J$$

Ce qui correspond à un problème 2D au plan (r,z) avec une réluctivité ν/r et une conductivité σ/r. Le système matriciel associé est donc identique à (2.1), mais les intégrales pour calculer la matrice élémentaire, deviennent:

(2.34) $$\mathbf{P}_e = 2\pi\iint_{\Omega_e}\frac{\nu}{r}.[\nabla\alpha][\nabla\alpha]^T d\Omega$$

(2.35) $$\mathbf{Q}_e = 2\pi\iint_{\Omega_e}\omega\frac{\sigma}{r}.\{\alpha\}\{\alpha\}^T d\Omega$$

Dans ces intégrales, il est évident que:

◊ Le calcul analytique des intégrales est impossible dans une approche directe. Nous notons qu'il est possible cependant d'utiliser une procédure d'intégration numérique (exemple par des points de Gauss) pour résoudre cette difficulté.

◊ Compte tenu de la présence du rayon, il faut donc effectuer la dérivation sur le terme "1/r" en plus des dérivations de la matrice de transformation citées ci-dessus.

En conclusion, nous notons que la dérivation par rapport aux paramètres géométriques est très compliquée, elle dépend fortement des types d'éléments utilisés et des formulations à traiter.

2.4. Spécificité des paramètres géométriques

Ici, nous voulons détailler quelques points importants dans une analyse de sensibilité d'ordre élevé par rapport aux paramètres géométriques.

2.4.1. Particularités du modèle géométrique

Si les aspects fondamentaux relatifs à la gestion de la géométrie sont aujourd'hui bien maîtrisés, l'implantation d'un modeleur géométrique nécessite beaucoup de savoir-faire et de nombreuses années de travail. Lorsqu'on s'intéresse à la paramétrisation de la géométrie, de nombreux problèmes supplémentaires se posent, surtout en cas de multi-paramètrage.

a. Problèmes liés à la définition de la géométrie

Aujourd'hui, il existe de nombreux modèles de représentation géométrique, la paramétrisation dépend donc à un certain niveau du modèle utilisé. Par exemple, dans un modèle de B-Rep, les paramètres géométriques sont souvent liés à un objet de positionnement (coordonnées des points), ils ne décrivent pas alors une surface. Dans un modèle CSG, on ne connaît pas explicitement les frontières. Cette insuffisance rend difficile le contrôle de la variation de la géométrie (et ensuite du maillage) lors de la variation des paramètres.

La diversité des définitions est une autre particularité des paramètres géométriques. Un segment d'arc est facile à définir par trois points (origine P_1, intermédiaire P_m et final P_2), mais aussi par le centre O, le vecteur d'origine OP_1 (ou rayon) et l'angle d'ouverture. La deuxième définition est beaucoup plus explicite. Le choix de la définition dépend du modèle utilisé mais en général, c'est une tâche supplémentaire fastidieuse pour l'utilisateur.

b. Problèmes liés au multi-paramètrage

S'il existe plusieurs paramètres à traiter, il faut faire particulièrement attention à leur choix. Souvent, il y a différentes possibilités pour définir le paquet de paramètres indépendants. Le choix va influencer toute la procédure de paramétrisation. De nombreux critères sont en considération, cependant, le plus important est ***"l'indépendance des intervalles de variation"***.

Prenons le cas simple de la figure (Fig.2.5), où on a deux paquets équivalents X1 et X2 de quatre paramètres indépendants.

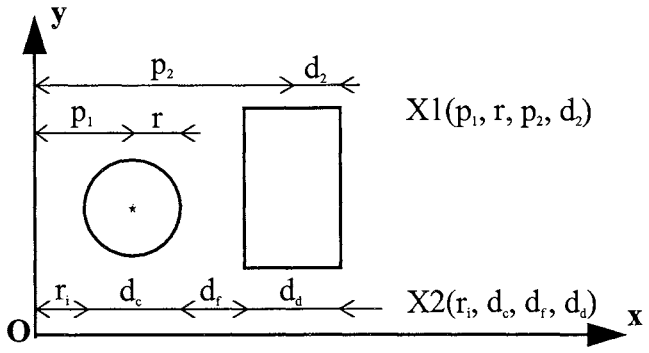


Fig.2.5. Exemple des paquets de paramètres géométriques indépendants.

Le premier paquet $X1(p_1,r,p_2,d_2)$ est le type de définition significative (deux positions et deux demi-largeurs), mais mauvaise pour la paramétrisation. La raison est évidente: les intervalles de variation des paramètres r et d_2 dépendent des valeurs des paramètres de positions p_1 et p_2. Le bon choix doit être $X2(r_i,d_c,d_f,d_d)$.

c. Dérivées croisées et dérivées d'ordre très élevé

Les dérivées sont croisées entre paramètres géométriques et paramètres physiques. Mais parfois, les paramètres géométriques influent sur les mêmes objets géométriques donc introduisent des dérivées géométriques croisées.

Dans certains cas, la géométrie est dépendante à des "ordres élevés" par rapport aux paramètres géométriques. Par exemple pour une rotation, le paramètre "angle de rotation" intervient sur les coordonnées par une expression trigonométrique, qui possède en principe toutes les dérivées. Même dans un cas plus courant, où la géométrie n'intervient qu'au premier ou au deuxième ordre, la dérivation des intégrales élémentaires va créer des ordres supérieurs. C'est le cas par exemple de l'équation (2.23) où la dérivation de "$1/S_\Delta$" est infinie lorsque S_Δ dépend des paramètres.

2.4.2. Perturbation de maillage

La perturbation de maillage est la principale différence entre le traitement des paramètres géométriques et le traitement des paramètres physiques. La perturbation est définie comme la variation du maillage lors d'une variation des paramètres, avec comme contrainte de maintenir la topologie du maillage initial.

En réalité, il faut réaliser un pré-calcul de dérivation sur le maillage afin de pouvoir ensuite dériver la matrice EF.

Dans notre optique, nous donnons la plus grande attention à deux aspects:

- ◊ Le choix du point initial de traitement. Pratiquement, le calcul initial est réalisé au point milieu de l'intervalle de variation.
- ◊ La qualité du maillage initial et la dégradation de la qualité des éléments consécutive aux modifications des paramètres géométriques.

En effet, la perturbation de maillage est une procédure complexe et très spécifique. Elle dépend de nombreux facteurs: le modèle géométrique, la nature de paramètres géométriques, le maillage initial, etc..

Quel ordre et quel intervalle de validité ?

2.4.3. Ordre des dérivées et rayon de convergence d'un paramètre

La base de notre méthode est l'utilisation des polynômes de Taylor. Naturellement il se pose le problème de la précision des calculs et de la validité des développements. Le rayon de convergence est l'intervalle de variation, dans lequel le polynôme s'approche suffisamment de la solution précise. Dans cet intervalle, plus on prend de termes dans le polynôme d'approximation, plus la solution est précise. Par contre, en dehors de cet intervalle, on ne peut pas améliorer la précision par l'augmentation de l'ordre d'approximation, bien au contraire.

Remarquons que pour les paramètres géométriques, il existe une facette supplémentaire de la validité: La validité du maillage perturbé.

Nous notons ici Π_i la table des positions des noeuds du maillage initial, un maillage perturbé a pour table $\Pi_i + \Gamma_p$ où $\Gamma_p(\delta p)$ est un déplacement imposé par le modèle suite à une variation δp des paramètres **p**. On s'aperçoit que cette fonction n'a de sens que sur un certain intervalle [p',p"]. En effet, pour de trop grandes valeurs $|\delta p|$ les mailles finissent en général par se croiser, l'aire (ou le volume) de certains éléments passe par zéro. Comme cette aire figure au dénominateur des matrices élémentaires, on a une division par zéro. L'intervalle [p',p"] est donc le rayon de convergence géométrique du modèle discrétisé.

En ce qui concernent le multi-paramètrage, l'expérience montre que la région de validité est toujours plus petite que le domaine combiné des simples rayons de convergence. Par exemple, la figure (Fig.2.6) présente la région de validité dans un modèle à deux paramètres géométriques **p** et **q**.

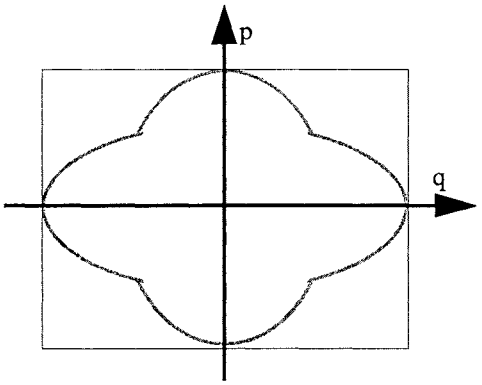


Fig.2.6. Exemple de la région de validité de deux paramètres géométriques

En conclusion, nous notons que l'étude de sensibilité d'ordre élevé par rapport aux paramètres géométriques est aussi complexe que le calcul EF lui-même. De nombreuses implantations sont à réaliser dans chaque module EF, il faut surtout faire attention aux croisements des paramètres et aussi à la validité du modèle.

Bilan de cette présentation

Nous avons dans ce chapitre présenté la méthode d'analyse de sensibilité d'ordre élevé dans le calcul EF, une méthode prometteuse mais initialement coûteuse. Nous avons abordé les problèmes "théoriques" de cette méthode et les variantes possibles de son implantation, nous avons vu les difficultés et quelques limitations dans les approches proposées.

Par rapport à l'objectif initial, nous avons donné ici quelques précisions sur le traitement des paramètres géométriques.

La mise en oeuvre du dérivateur de maillages et du processeur paramétré sera présentée aux chapitres 3 et 4. Naturellement, les critères du choix de réalisation sont:

◊ Diminuer le temps du calcul des dérivées,

◊ Agrandir le rayon de convergence (ou le domaine de validité) des paramètres.

Chapitre 3

MISE EN OEUVRE DU DÉRIVATEUR DE MAILLAGES

Sommaire

Chapitre 3

MISE EN OEUVRE DU DÉRIVATEUR DE MAILLAGES

Dans le chapitre précédent, nous avons décrit formellement le cadre de notre étude, qui est l'analyse de sensibilité à l'aide des dérivées d'ordre élevé, notre but étant de déterminer une solution Éléments Finis (EF) sous la forme d'un développement de Taylor par rapport aux paramètres de conception. Le paramétrage total d'un problème EF demande, en premier lieu, un pré-processeur EF paramétré, dans lequel les paramètres sont transportés du modèle physique ou/et du modèle géométrique, vers le modèle discrétisé.

Dans le pré-traitement, le paramétrage physique ne pose pas de grandes difficultés. En revanche, le paramétrage géométrique apparaît beaucoup plus complexe et demande une certaine initiative de la part du concepteur. En effet, la géométrie et le maillage sont très spécifiques d'une structure à l'autre et d'un logiciel à l'autre. Par conséquent, il est difficile de trouver une procédure universelle pour tous les cas. De plus, les paramètres géométriques peuvent avoir des natures très différentes. Leur choix lui même est une affaire très intuitive et donc délicate. Enfin, ce paramétrage concerne indirectement le maillage, qui est toujours un maillon difficile de la Méthode des Éléments Finis (MEF).

L'objectif de ce chapitre consiste donc à jeter les bases d'un module de pré-traitement performant, adapté aux besoins spécifiques de notre méthode d'analyse. Bien entendu,

les outils proposés pourront aussi être utilisés indépendamment de l'analyse de sensibilité d'ordre élevé.

On souligne qu'une telle mise en oeuvre a déjà été proposée par la société CADOE, et a été concrétisée par un produit industriel de traitement de maillage [ADOMESH]. Pour notre part, nous proposons une approche originale de dérivation du maillage, basée sur une dérivation analytique de la géométrie. Cette méthode est très rapide et très souple pour traiter les déformations complexes. Enfin, la validation informatique est réalisée en FORTRAN-77 et en ANSI-C, directement au sein du pré-processeur de FLUX3D, dont les sources sont en FORTRAN-77.

3.1. Paramètres géométriques et Modèle paramétré

Dans la suite de ce chapitre, les paramètres géométriques seront les seuls paramètres considérés. Pour alléger l'écriture, nous les appellerons simplement "paramètres".

3.1.1. Déclaration des paramètres

Le calcul d'un problème EF paramétré commence par la déclaration des paramètres. Leur définition peut découler directement du cahier des charges ou peut également être imposée par l'utilisateur-expert. Dans les grands codes de calcul EF, on dispose d'un mécanisme de déclaration des paramètres et aussi de certains moyens de les faire intervenir dans la construction de la géométrie. Bien souvent, par précaution et pour ne pas prendre le risque de devoir recommencer le paramétrage, on paramètre tout (et parfois n'importe quoi) a priori, ce qui conduit, en quelque sorte, à un sur-paramétrage. En définitive, une définition claire et significative des paramètres est toujours un point de départ souhaitable pour la suite des opérations, bien qu'elle soit difficile parfois.

Les paramètres se présentent habituellement comme des entités spécifiques qui participent à la définition des entités géométriques. La déclaration de l'entité "paramètre géométrique" dans le formalisme propre à FLUX3D est la suivante :

```
type entité    : PARAM          Paramètre géométrique
       NOM     : C80[1:1]       Nom de référence (une chaîne de caractères)
       EXP     : SCAL_I[1:1]    Expression algébrique (une sous-structure)
       VAL     : R08[1:1]       Valeur courante du paramètre (un réel double)
```

Le noyau de cette déclaration est une expression algébrique (le champ EXP dans la déclaration) fonction de constantes et/ou d'autres paramètres. Cette expression supporte aussi toutes les fonctions mathématiques usuelles admises par le FORTRAN (sin, cos, abs,...). Signalons qu'à tout moment, chaque paramètre FLUX3D porte une valeur numérique courante (le champ VAL), et toutes les opérations de construction et d'évaluation géométrique font référence à cette valeur, sans tenir compte de la dépendance paramétrique explicite du paramètre.

La définition en cascade des paramètres (un paramètre dépendant de paramètres définis préalablement) augmente beaucoup leur fonctionnalité et leur puissance. Elle donne une forme plus simple et plus claire aux définitions géométriques. Cependant, parmi l'ensemble des paramètres déclarés (et utilisés) dans la géométrie, il faut identifier clairement ceux qui pilotent réellement l'étude paramétrique. Dans ce but, on peut envisager soit une modification des attributs des paramètres standards, soit une surcharge des paramètres retenus par une nouvelle entité opérationnelle. La deuxième solution nous paraît plus générale, car elle répond mieux au besoin de compatibilité du modèle et aux critères de paramétrage présentées dans le paragraphe (§2.4). En définitive, pour simuler réellement le calcul paramétré dans FLUX3D, nous avons utilisé une variable de pilotage PARAM_DRV.

```
type entité    : PARAM_DRV      Paramètre à dériver
       PARAM   : PARAM[1:1]     Param. mère (référence vers param. standard)
       ETAT    : I04[1:1]       Identification de marquage (un entier)
       VAL     : R08[2:2]       Valeurs courant et initiale(deux doubles)
       IVAL    : R08[2:2]       Intervalle de validité (provisoire).
```

La déclaration des paramètres étant présentée, nous pouvons passer à la construction d'une géométrie paramétrée.

3.1.2. Un modèle géométrique réaliste

Dans le premier chapitre, les modèles géométriques ont été présentés dans toute leur diversité. L'accent a été surtout mis sur leurs possibilités théoriques de description des dispositifs réels. Dans ce paragraphe, nous allons étudier la façon pratique, et donc réaliste, de paramétrer un modèle géométrique. Il est clair que le paramétrage du maillage, et au-delà, de toute la chaîne de calcul, dépend fortement de la manière dont la géométrie elle même est paramétrée.

Si le descripteur géométrique permet la création de paramètres, il doit disposer également des outils de validation des dépendances. Néanmoins, ces outils peuvent ne pas répondre entièrement aux problèmes spécifiques à la mise en oeuvre de notre méthode. Pour cette raison, une étude détaillée de la représentation géométrique est indispensable pour comprendre les développements effectués, surtout pour le paramétrage du maillage. Notre étude s'appuie donc sur la représentation géométrie, utilisée dans FLUX3D [FLUX3D]. Nous ferons remarquer tout d'abord, que le module géométrique date d'une quinzaine d'années, qu'il est volumineux et d'une grande complexité. Les modifications ne sont donc pas faciles à apporter.

Le descripteur géométrique de FLUX3D est de type frontalier, ce qui signifie: qu'un ***volume*** est décrit par les faces qui le bordent, qu'une ***face*** est décrite par les lignes qui la bordent et qu'une ***ligne*** est décrite par une relation vers des ***points***. La géométrie est construite donc de manière ascendante (Fig.3.1).

Les points sont bien naturellement les entités de base d'une telle géométrie, ils sont créés en donnant leurs coordonnées dans un système défini - le ***repère*** ou à l'aide des points existants et d'une transformation géométrique. Le repère et les transformations géométriques sont des outils spéciaux, à l'aide desquels le modèle B-Rep (représentation de frontières) de FLUX3D s'approche des performances des modèles CSG (modeleur de solides).

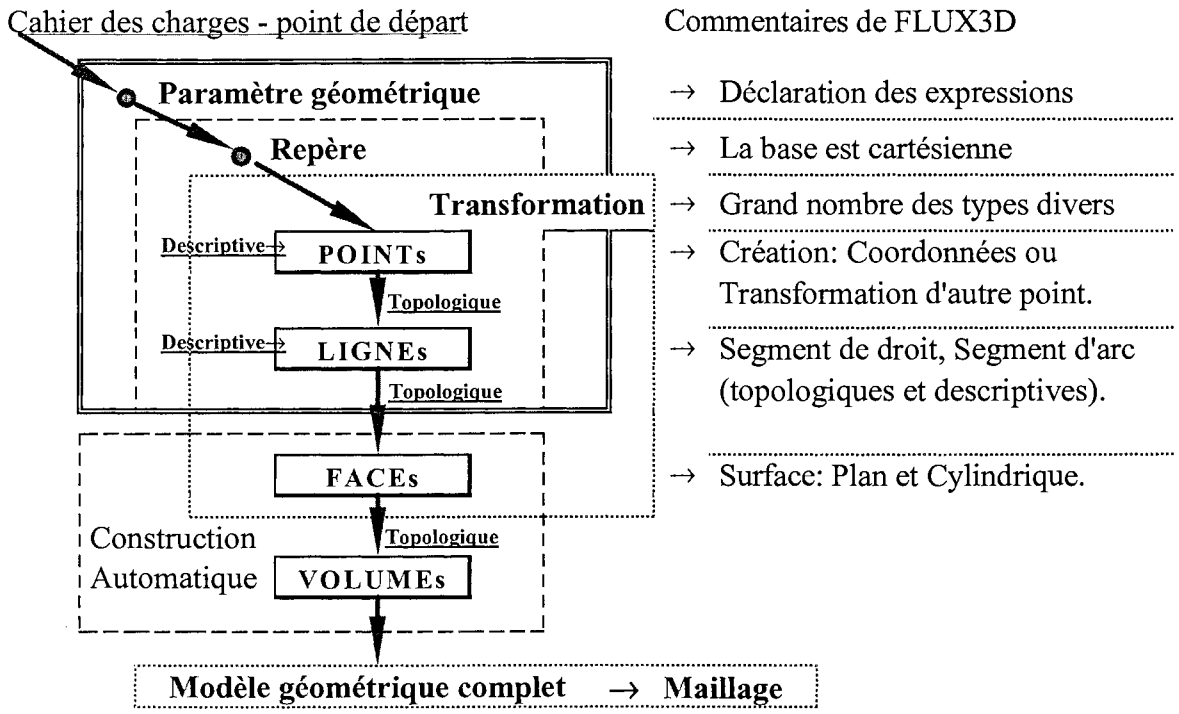


Fig.3.1. Arbre de description géométrique FLUX3D, caractéristiques des éléments et des principaux outils (avec leurs régions d'influence directe) [FLUX3D].

Le paramétrage de la géométrie est l'un des points forts du descripteur géométrique de FLUX3D. Grâce au concept de l'expression algébrique, les paramètres peuvent participer directement à la définition:

◊ d'un point, aux expressions de ses coordonnées,

◊ d'un arc (entité "ligne"), par les coordonnées du centre, le rayon ou l'angle d'arc,

◊ d'un repère, par la position absolue de l'origine et par les angles de rotation relative,

◊ d'une transformation, par le rapport, l'angle de rotation, etc..

La paramétrisation totale de la géométrie se réalise donc par l'intermédiaire de ces définitions primitives. On remarque trois points caractéristiques importants:

◊ Grâce aux outils de la base de données (géométriques), l'autre point fort de FLUX3D, toutes les entités dépendant directement ou indirectement d'un paramètre ou entité primitive modifié, seront automatiquement réévaluées.

◊ Du fait de la déclaration en cascade et combinée des paramètres, une entité géométrique, même la plus simple comme une coordonnée de point, peut dépendre de plusieurs paramètres.

◊ L'utilisation des repères et des transformations paramétrées apporte beaucoup de souplesse sur la construction et modification de la géométrie (on illustrera cette puissance dans les applications spéciales qui se trouvent dans les derniers paragraphes de ce chapitre).

3.1.3. Un modèle plus direct

Les définitions en cascade et les définitions par entités secondaires sont des points forts qui facilitent la description paramétrée de la géométrie (B-Rep). Cependant, elles introduisent des difficultés de contrôle et complexifient la mise en oeuvre de la dérivation géométrique. Par ailleurs, elles posent des problèmes de portabilité du modèle.

La définition directe par l'utilisateur d'une représentation B-Rep avec des points et des lignes a des limites quant à la complexité des objets représentables. Les modeleurs solides actuels CSG permettent de traiter directement tous les volumes avec plus de souplesse et de facilité. En ce qui concerne le maillage, les approches sont équivalentes puisque dans les deux cas les noeuds sont créés numériquement.

Enfin, il faut bien souligner que dans tous les cas, les déclarations et utilisations des paramètres exigent une bonne maîtrise de la part des utilisateurs.

3.2. Méthodes de dérivation du maillage

Une fois que la géométrie a été paramétrée, il faut définir la manière dont le maillage varie lors d'une modification des paramètres. Plus précisément, il faut réaliser un pré-calcul de dérivation sur le maillage afin de pouvoir ensuite dériver la matrice EF (voir §2.2). En raison de sa nature, le maillage EF est défini par la position des noeuds et la topologie des éléments. En supposant que la topologie du maillage reste inchangée, tant que la topologie de la géométrie garde sa configuration, alors, ce sont les déplacements des noeuds, et eux seuls, qui assurent la perturbation du maillage. La dérivation du maillage consiste donc à définir la dépendance explicite des coordonnées des noeuds par rapport aux paramètres. En effet, notre méthode demande une approximation sous la forme de développement de Taylor, il faut donc calculer toutes les dérivées de chaque noeud par rapport aux paramètres géométriques.

La figure 3.2 donne un classement simple des noeuds s'appuyant sur les entités d'un modèle géométrique. Ce classement va servir à justifier l'utilisation des différents noeuds-types dans la procédure de perturbation du maillage et dans ce qui suit.

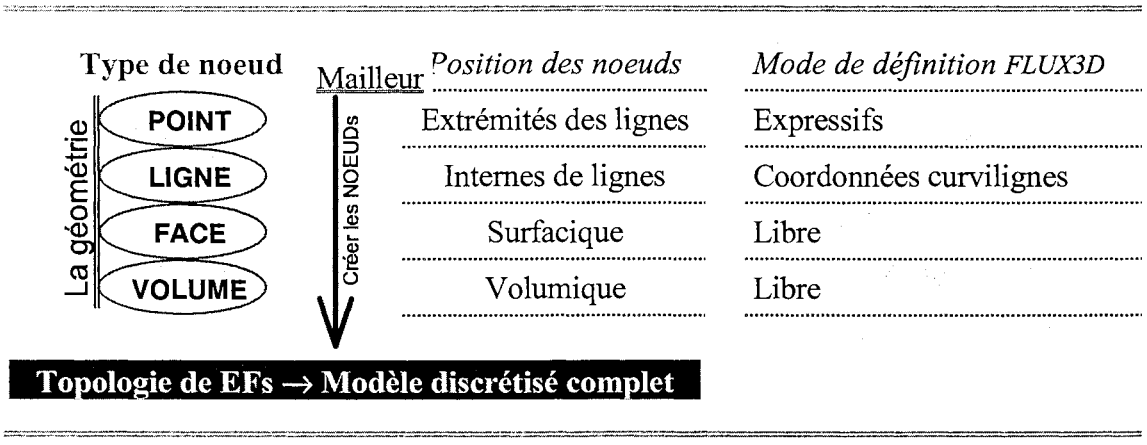


Fig.3.2. Classement topologique des noeuds au niveau du mailleur

3.2.1. Méthodes de calcul des dérivées d'un noeud

Les dérivées d'un noeud peuvent se calculer par une des manières suivantes:

◊ Calcul analytique, c'est-à-dire direct, à condition que le noeud soit défini explicitement par des expressions analytiques dépendant des paramètres.

◊ Par différence finie $\Delta\zeta/\Delta p$, avec $\Delta\zeta$ un déplacement (ou déplacement virtuel) de noeud lors d'une petite variation Δp du paramètre.

◊ Par approximation fonctionnelle, on cherche donc une fonction analytique qui décrit au mieux les positions des noeuds correspondant aux valeurs choisies des paramètres. Ayant cette fonction, un dérivateur analytique nous donnera les dérivées nécessaires.

a. *Méthode analytique*

C'est une méthode précise et relativement simple à implanter, mais elle n'est applicable qu'à des formes très simples, car elle demande un paramétrage direct de toutes les coordonnées des noeuds. C'est le type de définition naturel pour un noeud lié à un point. Quand un noeud est sur une ligne, la dépendance peut se définir à partir d'une coordonnée curviligne en s'appuyant sur la description analytique de la ligne. Avec une représentation surfacique par carreaux, on pourrait étendre cette dépendance par l'intermédiaire de coordonnées surfaciques. De même avec une représentation volumique par blocs (parallélépipédiques, prismatiques, ...), on pourrait utiliser un système de coordonnées volumiques.

Malheureusement, les objets de l'électromagnétisme et tout particulièrement l'air englobant la matière solide ont des formes particulièrement complexes qu'il est souvent impossible de découper simplement en blocs élémentaires.

b. *Calcul par différences finies*

C'est une méthode de dérivation numérique. Au premier ordre, elle est précise dans le cas de variations linéaires ou pour de petites perturbations quasi-linéaires. Dans la méthode des éléments finis, le gradient est souvent utilisé, il y a donc de nombreux outils de dérivation automatique (disponibles sur INTERNET), qui utilisent principalement la dérivation du premier

ordre. C'est un des avantages de la mise en oeuvre de ce calcul. Notons que des dérivations d'ordres plus élevés sont parfois nécessaires pour augmenter la validité de l'approximation.

Deux difficultés majeures sont à résoudre: la manière dont on fait bouger les noeuds et le traitement des dérivées d'ordre supérieur.

c. *Calcul à l'aide d'une approximation*

Cette approche est une généralisation de l'approximation linéaire de l'algorithme précédent. En premier lieu, il faut réévaluer plusieurs fois le modèle géométrique et le modèle discrétisé pour différentes valeurs des paramètres. Pour chaque noeud, on obtient un nuage de positions. Ensuite, en s'appuyant sur ce nuage de positions, on construit une fonction d'approximation pour la position de noeud. En générale, pour résoudre ce problème, différents types d'approximation sont possibles.

À ce jour, la synthèse et l'optimisation n'étudient que des variations (de la géométrie) de complexité assez basse, l'utilisation d'un polynôme de faible degré est donc généralement suffisante. Finalement, les dérivées du maillage seront calculées à partir de la fonction obtenue.

Dans la mise en oeuvre de cette méthode, une bonne stratégie d'évaluation, c'est-à-dire un bon échantillonnage des paramètres, peut apporter des améliorations essentielles aux résultats. En pratique, on utilise fréquemment un traitement de type des "plans d'expérience".

3.2.2. Méthodes de Perturbation de Maillage

Sauf dans les cas très rares d'un traitement analytique global, les dérivateurs de maillage existants nécessitent toujours un moyen de faire varier (de perturber) le maillage en fonction de l'évolution des paramètres. La méthode la plus directe consiste, pour de nouvelles valeurs des paramètres géométriques, à reprendre toutes les étapes de construction, et donc à remailler le problème. Avec les mailleurs automatiques présents dans presque tous les pré-processeurs EF modernes, les maillages libres générés par de tels mailleurs ne remplissent pas l'exigence d'une topologie constante lorsque la géométrie varie [Weeb92, Moren93]. Les changements de topologie introduisent des bruits numériques et des discontinuités artificielles sur toutes les grandeurs globales ou locales résultant des résolutions éléments finis successives.

Le remaillage est donc exclu. Il faut d'abord évaluer la géométrie et puis chercher un algorithme pour propager la déformation obtenue sur la totalité du modèle discrétisé. Quelques solutions sont proposées aujourd'hui: le bougé de noeuds, la position optimale, la transformation structurale.

a. *Bougé de noeuds*

Le bougé de noeuds est la technique générale d'amélioration non-topologique de la qualité du maillage. La perturbation est obtenue en définissant localement le déplacement optimal de chaque noeud.

Le plus simple à réaliser est une procédure itérative de perturbation par barycentrage aléatoire. Chaque noeud est déplacé au barycentre de ses proches voisins. L'utilisation du barycentrage est efficace en 2D [Alber88], mais pour un maillage 3D la performance de cette méthode est bien limitée. Un autre point faible du barycentrage et qu'il génère des éléments plats.

Pour le maillage libre tétraédrique 3D, une approche plus générale consiste à déplacer un noeud P pas à pas (via un coefficient α) vers un point optimal P_{opt} calculé à partir des tétraèdres idéaux, en s'appuyant sur les facettes extérieures de la boule de tétraèdres connectés au noeud [HenGe93]. Schématiquement on peut écrire:

$$P = P + \bar{d} \quad \text{avec } \bar{d} = \alpha . \overline{PP}_{opt} \text{ le déplacement}$$

et $$P_{opt} = \sum_{j=1}^{n} \beta_j P_{idj} \quad \text{avec } \beta_j = 1/n \text{ par exemple.}$$

où n, le nombre de points connectés à P

P_{idj}, la position idéale de P pour la facette triangulaire de numéro j

β_j, le poids associé au point j ($\Sigma\beta_j = 1$).

Le couplage entre un déplacement grossier et l'une des techniques mentionnées ci-dessus est nécessaire pour accélérer la convergence du bougé de noeuds vers une perturbation optimale du maillage. Ce pré-traitement est surtout important en cas de déformation assez forte [Ng-TN95].

b. *Position Optimale*

Le bougé de noeuds est une optimisation locale. Il existe aussi des critères définissant la qualité globale du maillage. En s'appuyant sur ces critères, un algorithme d'optimisation performant, de type stochastique, par exemple algorithme génétique ou recuit simulé, peut produire les positions optimales des noeuds [Salud97]. Une grande faiblesse de ces techniques réside en leur coût de calcul trop élevé, car la convergence demande généralement un nombre très important d'itérations.

c. *Transformation Structurale*

Une approche plus physique consiste à utiliser les équations d'élasticité (c'est l'idée initiale du barycentrage). La perturbation du maillage est obtenue en prolongeant les déformations géométriques à l'intérieur du maillage par résolution d'un problème d'élasticité avec déplacement imposé sur le bord. L'analyse de performance de cette méthode, accompagnée d'une réalisation 2D, est décrite par Weeber dans [Weebe94]. Une autre réalisation plus complète se trouve dans ADOMESH [CADOE], un produit-outil de perturbation de maillage. Le système d'équations à résoudre s'écrit:

$$[K]\cdot\{u\} = \left[\begin{array}{c|c} K_{uu} & K_{us} \\ \hline K_{su} & K_{ss} \end{array}\right]\cdot\left\{\begin{array}{c} u_u \\ \hline u_s \end{array}\right\} = 0$$

où [K], la matrice de rigidité (matrice de transformation), assemblage EF réalisé sur les régions déformées de la géométrie.

{u}, les déplacements nodaux, dans lesquels u_s sont les déplacements imposés aux bords et u_u, les composantes internes inconnues.

En absence de force interne, et en supposant que le milieu est identique partout, cette formulation devient vraiment une transformation topologique, c'est-à-dire une transformation structurale, dont l'implantation ne pose pas de grandes difficultés. Enfin, on note que cette méthode peut générer un maillage peu satisfaisant avec des éléments aplatis (point commun avec le barycentrage).

La figure 3.3. montre un schéma usuel de dérivation du maillage, où les phases de déformation de la géométrie et de perturbation du maillage doivent être réévaluées plusieurs fois suivant la stratégie choisie pour la variation des paramètres.

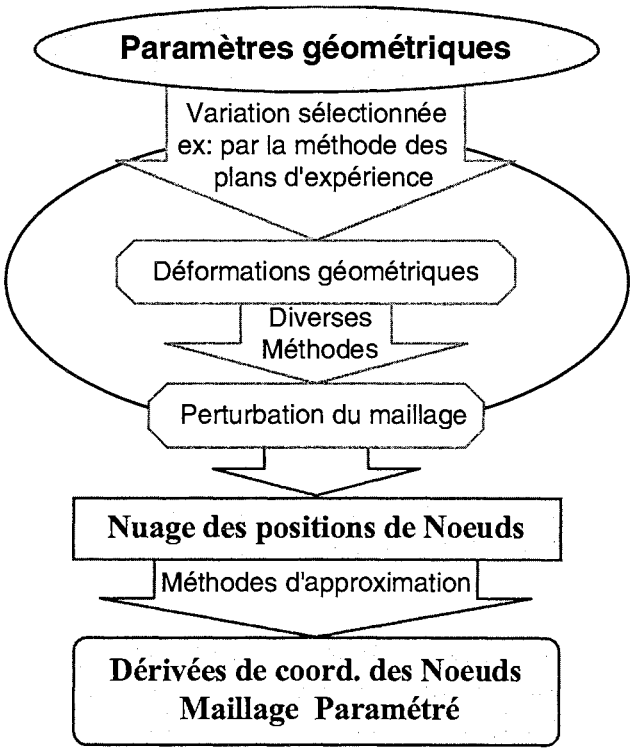


Fig.3.3. Dérivation du maillage EF

3.3. Un dérivateur semi-analytique

Nous présentons ici le travail réalisé dans le module de pré-traitement de FLUX3D. Il s'agit de la première mise en oeuvre de la méthode de dérivées d'ordre élevé concernant le traitement de problèmes électromagnétiques. Mais nous mettrons en évidence qu'elle n'est pas spécifique aux applications de ce domaine.

3.3.1. Le choix de la méthode

Dans les méthodes citées ci-dessus, les dérivées de noeuds se calculent par l'intermédiaire du nuage de leurs positions ou de leurs déplacements relatifs. Ici, au contraire, notre approche consiste à calculer directement les polynômes d'approximation, c'est-à-dire les dérivées, à partir des expressions géométriques des frontières et de la topologie du maillage.

Un tel calcul est envisageable, d'un part, grâce à la dérivation analytique des expressions géométriques des frontières, et d'autre part, grâce à la propagation des dérivées obtenues sur les frontières par une des méthodes ci-dessus.

Ici, le premier point fort réside dans l'utilisation directe et maximale des informations stockées dans le modèle géométrique paramétré, ce qui donnera un coût de calcul assez bas. L'autre point fort est la possibilité de gérer parfaitement l'ordre des dérivées lors de la dérivation du maillage, car les dérivées de la géométrie sont connues précisément. Enfin, l'approximation polynomiale de la géométrie conduit à une synchronisation parfaite au niveau du post-traitement, alors l'évaluation du résultat peut être réalisée totalement indépendamment du calcul initial, c'est-à-dire dans un autre programme EF ou même en dehors des calculs EF.

La figure 3.4 présente le schéma de dérivation du maillage par cette nouvelle méthode. Dans la suite de ce paragraphe, nous détaillerons étape par étape l'implantation de cette procédure au sein de FLUX3D. Nous donnerons aussi, à la fin, un bilan comparatif, qui présente toutes les particularités de cette implantation.

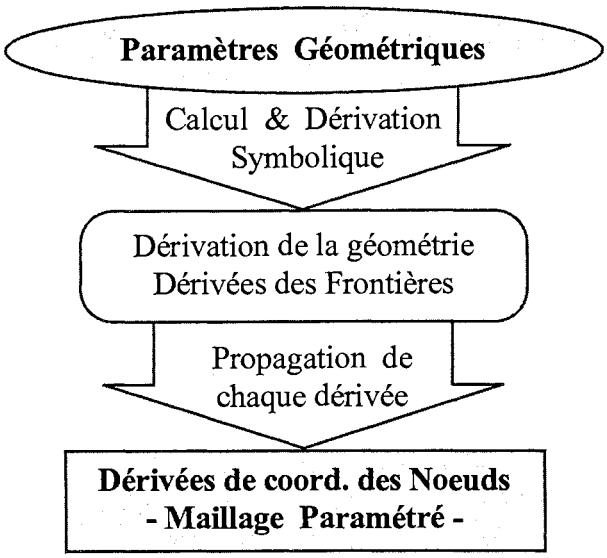


Fig.3.4. Dérivation semi-analytique du maillage EF

3.3.2. Dérivation symbolique de la géométrie

C'est la première étape de la dérivation et la partie analytique du traitement. Au paragraphe (§3.1), nous avons constaté qu'il est préférable que les entités géométriques de frontière (points, lignes) possèdent une définition directe, c'est-à-dire une définition par expressions algébriques. Alors, il suffit de dériver analytiquement ces expressions. Cependant, le besoin d'utiliser les définitions indirectes est grand dans la majorité des cas. Pour pouvoir dériver les noeuds frontaliers, il est donc nécessaire de définir pour chaque entité, une dépendance explicite.

Dans ce but, nous avons construit dans FLUX3D un module de calcul symbolique élémentaire. Premièrement, les expressions ordinaires doivent être transformées sous la forme d'expressions "polonaises-inverses". Ensuite, un calcul symbolique sur ces expressions primitives nous donne les définitions nécessaires des entités désirées. Enfin, le dérivateur symbolique produit les dérivées nécessaires (Fig.3.5).

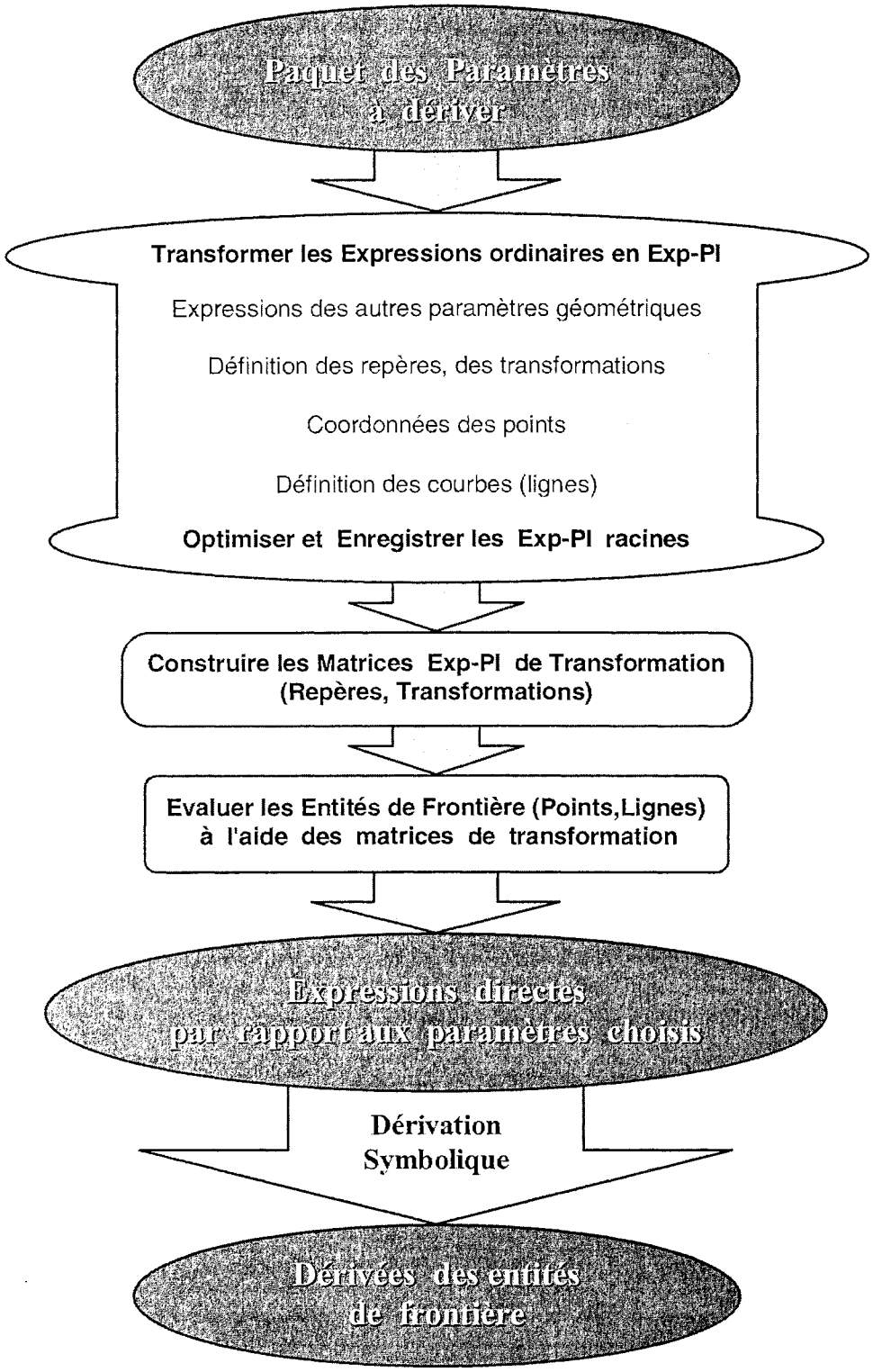


Fig.3.5. Calcul symbolique par Exp-PI dans le module de pré-traitement FLUX3D. Dérivation analytique des entités géométriques frontalières.

a. *L'expression "polonaise-inverse"*

Avant de traiter le calcul symbolique dans la dérivation du maillage, nous rappelons le concept d'expression "polonaise-inverse". L'Expression Polonaise-Inverse (abréviation Exp-PI) est une autre présentation de l'expression algébrique ordinaire. Elle se forme en supprimant tous les composants topologiques (les parenthèses et les virgules) et en mettant les opérateurs et les opérandes dans l'ordre d'exécution, chaque opérateur immédiatement après ses opérandes. Par exemple les expressions ordinaires:

(3.1) $5+10\times\sin(\varpi\times t+\pi/4)$ et

(3.2) $7\times\sqrt{p}-5$

deviennent sous la forme polonaise-inverse (les virgules servent seulement à la présentation):

(3.3) $\{5\ ,10\ ,\varpi\ ,t\ ,\times\ ,\pi\ ,4\ ,/\ ,+,\ \sin\ ,\times\ ,+\}$ et

(3.4) $\{7\ ,p\ ,\sqrt{\ }\ ,\times,\ 5\ ,-\}$

On peut améliorer la lisibilité de l'expression (3.3), en groupant les opérandes:

$$(5\ (10\ (((\varpi\ ,t)\times\ (\pi\ ,4)\ /)+\ )\sin)\times\ )+$$

Les opérations arithmétiques sont réalisées d'une manière très naturelle sur les Exp-PI. Une addition, par exemple, de (3.1) et (3.2) sous forme polonaise-inverse (3.3),(3.4) donne:

$$\{5\ ,10\ ,\varpi\ ,t\ ,\times\ ,\pi\ ,4\ ,/\ ,+,\ \sin\ ,\times\ ,+\}\{7\ ,p\ ,\sqrt{\ }\ ,\times,\ 5\ ,-\}+$$

Après une simple optimisation, elle devient:

$$\{10\ ,\varpi\ ,t\ ,\times\ ,\pi\ ,4\ ,/\ ,+,\ \sin\ ,\times\ ,7\ ,p\ ,\sqrt{\ }\ ,\times,+\}$$

qui est $\{10\times\sin(\varpi\times t+\pi/4)+7\times\sqrt{p}\ \}$ sous la forme ordinaire

Dans le cas où p est le seul paramètre actif, une simplification calculée sur les constantes fait réduire cette Exp-PI à: $(7.071\ ,7\ ,p\ ,\sqrt{\ }\ ,\times,+)$ (prise à $\varpi t=0$).

La structure de Exp-PI déclarée en FORTRAN (formalisme FLUX3D) est:

```
Structure      : EXP_PI  Expression de polonaise inverse
        INTEGER NDIM            : Dimension
        INTEGER ITYP(NDIM)      : Type data, opérateur ou fonction
        REAL*8  RDAT(NDIM)      : Valeurs ou numéro de référence
```

Tous les opérateurs arithmétiques, les fonctions usuelles, le type de paramètre et le type de constante sont identifiés par un code unique. Tous les codes d'une Exp-PI sont stockés dans le tableau index ITYP. La série des valeurs "double précision" RDAT sert à stocker les valeurs des constantes, elle garde aussi en cas de besoin, le numéro de référence du type codé, par exemple un numéro du paramètre, un code complémentaire d'une fonction, etc..

Dans le module de pré-traitement, les Exp-PI des entités géométriques sont gérées par un type de relation, dont la déclaration dans le formalisme de FLUX3D est la suivante:

```
Structure      : GEOM_EXP       Relation des entités géom. vers des Exp-PI
     CAS*      GEOM_EN          : Une entité géométrique
     EXP_PI    EXP_DEF(1:MAX)   : Liste compacte des Exp-PI en relation
     INTEGER   ID_EXP           : Identification des Exp-PI dans la définition
```

* CAS est un type de relation multi-modale dans la base de données de FLUX3D, qui permet de brancher la relation directe de l'entité courante vers d'autres objets, mais un seul à la fois.

Dans cette déclaration MAX est le nombre maximal de Exp-PI reliées à une entité géométrique. L'entier ID-Exp sert à identifier les positions des Exp-PI dans la définition de l'entité géométrique, on utilise ici un codage binaire.

Prenons un point dans le système de coordonnées 3D, qui utilise au maximum 3 Exp-PI pour présenter ses coordonnées P:{Ex-x,Ex-y,Ex-z}. Si toutes les trois coordonnées dépendent des paramètres, ID-Exp est donc égal à: $2^0+2^1+2^2=7$, si le composant "y" ne dépend pas de paramètres, ID-Exp devient $(2^0+2^2=5)$ (avec 2 Exp-PI).

Prenons le cas plus complexe, d'une matrice de transformation 3D qui comporte au maximum 12 Exp-PI (Fig.3.6a). Pour un repère qui est défini par l'origine $\{0,0,z_0\}$ et la rotation $\{0,0,\alpha\}$, la matrice de transformation local-global contient 5 Exp-PI (Fig.3.6b).

Ex1	Ex4	Ex7	Ex10
Ex2	Ex5	Ex8	Ex11
Ex3	Ex6	Ex9	Ex12
0	0	0	1

a.) Structure complète de la matrice Exp-PI
ID-Exp:07777 (Octane)

α,cos	α,sin,-	0	0
α,sin	α,cos	0	0
0	0	1	z_0
0	0	0	1

b.) Exemple
ID-Exp:03304 (Octane)

Fig.3.6. Matrice de transformation sous la forme des Exp-PI

Dans la base de données, la structure ***Geom-Exp*** est créée exclusivement pour les entités possédant les Exp-PI, c'est-à-dire les entités dépendant des paramètres.

b. *Le calcul symbolique sur les Exp-PI*

La description ci-dessus donne une idée du calculateur symbolique basé sur les Exp-PI. En utilisant ces Exp-PI et la structure Geom-Exp, nous avons construit un dérivateur de la géométrie s'appuyant sur une représentation de frontière (B-Rep) (Fig.3.5).

Les outils de calcul symbolique se regroupent, suivant leurs fonctionnalités, en quatre grands modules:

◊ Un traducteur, qui traduit l'expression ordinaire en Exp-PI;

◊ Un analyseur-calculateur, qui réalise les opérations arithmétiques sur les Exp-PI, y compris les opérations matricielles, vectorielles et le calcul de la valeur d'une Exp-PI;

◊ Un compacteur, qui optimise et compacte les Exp-PI après chaque composition;

◊ Un dérivateur, pour produire toutes les dérivées désirées de l'Exp-PI en valeurs numériques.

En particulier, le traducteur transforme les expressions algébriques des entités primitives, sous la forme Exp-PI. Les expressions des coordonnées d'un point se traduisent directement. Par contre, une matrice de transformation géométrique est construite en plusieurs étapes et avec prudence, comme nous allons le voir plus tard. À la fin de cette traduction, on possède des expressions transformées grossièrement en Exp-PI.

Une optimisation globale suit immédiatement afin de minimiser la complexité des Exp-PI. D'abord elle élimine les paramètres complémentaires en les remplaçant par leur valeur numérique ou par leur Exp-PI explicite. Ensuite elle regroupe au maximum les membres constants, calculant les opérations et fonctions explicites sur eux.

L'étape suivante consiste à évaluer les entités dépendantes secondaires. Les Exp-PI d'un point transformé sont créées en appliquant une transformation constante sur les Exp-PI d'un autre point, ou en appliquant une matrice d'Exp-PI sur un point fixe, etc.. Le traitement continue en examinant toutes les entités de niveau supérieur (ligne, face, volume) à la dépendance spécifique, c'est-à-dire la dépendance qui facilite le calcul de dérivées. Par exemple, on peut trouver:

◊ une ligne, une face ou un volume ne dépendant pas de certains paramètres,

◊ un volume en déplacement global, ou au moins sans déformation,

◊ une ligne, une surface constantes locale sur un repère, etc..

Enfin, le dérivateur symbolique calcule les dérivées d'une Exp-PI, afin d'obtenir les dérivées de noeuds-points et noeuds-lignes. Les premières dérivées (des entités primitives) sont stockées dans la base de données. On va les utiliser d'abord pour calculer analytiquement les dérivées des autres noeuds frontaliers et ensuite en propagation numérique à l'intérieur des surfaces et des volumes.

c. *La dérivation des noeuds frontaliers*

Par noeuds frontaliers, on entend tous les noeuds confondus avec les points (les noeuds-points) et les noeuds reliés aux lignes (Fig.3.1). La dérivation des noeuds frontaliers comporte beaucoup de spécificité. Nous allons donc décrire, en détail, la procédure de dérivation de chaque type de noeud.

Pour un Noeud-point, il suffit d'introduire simplement ses Exp-PI dans le dérivateur.

Pour un Noeud situé sur un segment de droite, on peut calculer ses dérivées facilement à l'aide d'une coordonnée curviligne et des dérivées des points extrémités:

(3.5) $dN = (1- k).\ dE_1 + k.\ dE_2$

où dE_1, dE_2 sont les dérivées des extrémités

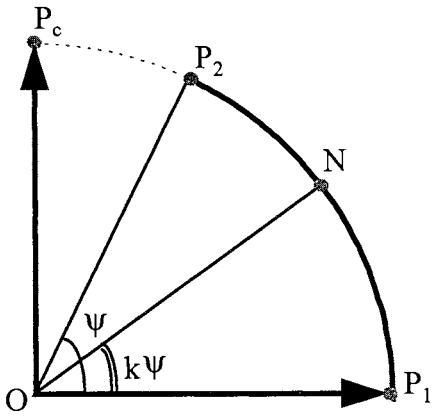
k est la coordonnée curviligne du noeud

dN sont les dérivées cherchées du noeud interne d'un segment de droit.

Pour un Noeud situé sur un segment d'arc de cercle ou de courbe, il faut prendre une des trois méthodes suivantes, qui s'appuie sur la définition géométrique:

⇒ Si une ligne est constante localement dans un repère ou si elle est la transformée d'une ligne constante, alors quand c'est possible, on utilise une procédure analytique. D'abord, on calcule (ou on récupère) les dérivées analytiques de la matrice de transformation et/ou de la ligne primitive. Ensuite, on définit pour chaque noeud, des coefficients de relation qui dépendent généralement des coordonnées globales et/ou de la coordonnée curviligne du noeud. Par exemple, les coordonnées locales du repère sont calculées à partir des coordonnées globales. Enfin, en utilisant les coefficients et s'appuyant sur les dérivées déjà calculées, on peut calculer toutes les dérivées du noeud.

⇒ Une ligne ayant une définition analytique standard, est transformée en une forme normalisée, pour laquelle une dérivation analytique est directement applicable. Par exemple pour un segment d'arc, nous avons utilisé le modèle présenté à la figure (Fig.3.7).



P_1,P_2 —Extrêmités
O —Centre
P_c —Point de définition
ψ —Angle de l'arc
N —Un noeud
k —Coord. curviligne
avec
eX EPI de l'entité X
dX Dérivées de X

$$\vec{ON} = \vec{OP}_1.\cos(k\psi) + \vec{OP}_c.\sin(k\psi)$$

(3.6) $$dN = dO + d[(eP_1\text{-}eO).\cos(k.e\psi) + (eP_c\text{-}eO).\sin(k.e\psi)]$$

Fig.3.7. Segment d'arc normalisé et sa dérivation analytique

Dans ce modèle là, les paramètres peuvent intervenir dans les expressions (Exp-PI) du centre O, du point géométrique P_1, du point P_c ou même dans l'expression de l'angle ψ. Évidement, la possibilité d'une telle dérivation est limitée aux lignes qui supportent les formes normales (celle ci-dessus ou une autre).

⇒ Si les deux méthodes citées ci-dessus sont inapplicables, un simple dérivateur numérique se met automatiquement en service. Cet outil utilise une approximation sur le nuage de noeuds positions, dont le principe est décrit dans le paragraphe précédent (§3.2). Dans notre étude, le dérivateur numérique est réservé aux lignes, il ne doit pas être confondu avec la procédure de perturbation du maillage local. Cette perturbation se réalise ici en interne, analytiquement à l'aide des Exp-PI définissant la ligne, et à l'aide d'une coordonnée curviligne du noeud. C'est donc un calcul rapide, sûr et sans bruit numérique. La description détaillée de ce module de dérivation se trouve dans le paragraphe suivant.

Enfin, nous notons que les formules (3.5) et (3.6) sont effectivement calculées à partir des dérivées primitives, c'est-à-dire que pour chaque noeud il faut calculer et stocker seulement des coefficients de relations.

d. *La dérivation numérique des lignes*

Un dérivateur numérique est mise en oeuvre dans FLUX3D, son utilisation est réservée à la dérivation des noeuds-lignes.

Pour la fonction d'approximation, on utilise un développement de Taylor d'ordre deux complet (dérivées simples et dérivées croisées d'ordre deux). La perturbation, c'est à dire, le nuage des positions est construit en s'appuyant sur un plan de variation, qui combine trois valeurs de paramètres: la valeur originelle, le bord inférieur et le bord supérieur.

On calcule les coefficients, c'est-à-dire les dérivées; par une méthode de type différence finie, dont nous allons tout d'abord rappeler le principe dans le cas d'un seul paramètre.

La dérivée du premier ordre est calculée par la relation:

$$\partial N/\partial p = dN = 0.5\left(\frac{P'-P^\circ}{p'-p^\circ}+\frac{P''-P^\circ}{p''-p^\circ}\right)=0.5\left(\frac{P'}{p'-p^\circ}+\left(\frac{P^\circ}{p^\circ-p'}+\frac{P^\circ}{p^\circ-p''}\right)+\frac{P''}{p''-p^\circ}\right). \tag{3.7}$$

avec p',p°,p", valeurs min., initiale et max. du paramètre
P',P°,P", trois positions correspondantes de noeud
N, le noeud étudié (ses coordonnées)

Supposons que la fonction de position P prenne la forme d'une série de Taylor d'ordre deux:

$$P = P^\circ + dN(p-p^\circ) + 0.5\, d^2N(p-p^\circ)^2 \tag{3.8}$$

$$\Rightarrow \quad 2\frac{P-P^\circ}{p-p^\circ} - 2dN = d^2N(p-p^\circ).$$

En remplaçant dN par (3.7), le développement de (3.8) aux points p' et p" nous donne

$$\frac{P'-P^\circ}{(p'-p^\circ)} - \frac{P''-P^\circ}{(p''-p^\circ)} = d^2N(p'-p^\circ) \tag{3.9}$$

$$\frac{P''-P^\circ}{(p''-p^\circ)} - \frac{P'-P^\circ}{(p'-p^\circ)} = d^2N(p''-p^\circ). \tag{3.10}$$

Enfin, à partir de (3.9) et (3.10) on calcule une correction du deuxième ordre d^2N

$$2\left(\frac{P''-P^\circ}{p''-p^\circ} - \frac{P'-P^\circ}{p'-p^\circ}\right) = d^2N(p''-p')$$

(3.11) $$\partial^2 N/\partial p^2 = d^2 N = \frac{2}{p''-p'}\left(\frac{P''-P^\circ}{p''-p^\circ} - \frac{P'-P^\circ}{p'-p^\circ}\right)$$
$$= \frac{P'}{(p'-p'')(p'-p^\circ)} + \frac{P^\circ}{(p^\circ-p')(p^\circ-p'')} + \frac{P''}{(p''-p')(p''-p^\circ)}$$

Notons que les équations (3.7) et (3.11) peuvent se généraliser facilement au cas multi-paramètres. On doit combiner donc à chaque position de noeud les coefficients de tous les paramètres concernés. Prenons l'exemple du cas de deux paramètres p et q, la dérivée du premier ordre par rapport au paramètre p va être calculée par le moyen de trois dérivées de (3.7):

(3.12) $$\partial N/\partial p = 1/3\left(\partial N/\partial p(q') + \partial N/\partial p(q^\circ) + \partial N/\partial p(q'')\right).$$
$$= \frac{1}{6}\left(\frac{P'^{\sim}+P'+P'^{*}}{p'-p^\circ} + (P^{\circ\sim}+P^\circ+P^{\circ *})\left(\frac{1}{p^\circ-p'} + \frac{1}{p^\circ-p''}\right) + \frac{P''^{\sim}+P''+P''^{*}}{p''-p^\circ}\right)$$

La dérivée croisante premier ordre de p et q s'écrit:

(3.13) $$\frac{\partial^2 N}{\partial p \partial q} = \frac{\partial}{\partial q}\frac{\partial N}{\partial p} = 0.5\left[\frac{1}{q'-q^\circ}\left(\frac{\partial N}{\partial p}(q') - \frac{\partial N}{\partial p}(q^\circ)\right) - \frac{1}{q''-q^\circ}\left(\frac{\partial N}{\partial p}(q'') - \frac{\partial N}{\partial p}(q^\circ)\right)\right]$$
$$= 0.25\left\{\frac{1}{q'-q^\circ}\left[\left(\frac{P'^{\sim}-P^{\circ\sim}}{p'-p^\circ} + \frac{P''^{\sim}-P^{\circ\sim}}{p''-p^\circ}\right) - \left(\frac{P'-P^\circ}{p'-p^\circ} + \frac{P''-P^\circ}{p''-p^\circ}\right)\right] - \right.$$
$$\left.\frac{1}{q''-q^\circ}\left[\left(\frac{P'^{*}-P^{\circ *}}{p'-p^\circ} + \frac{P''^{*}-P^{\circ *}}{p''-p^\circ}\right) - \left(\frac{P'-P^\circ}{p'-p^\circ} + \frac{P''-P^\circ}{p''-p^\circ}\right)\right]\right\}$$

Avec les positions P du noeud:

p \ q	q'	q°	q''
p'	P'~	P'	P'*
p°	P°~	P°	P°*
p''	P''~	P''	P''*

Suivant le même principe, on arrive à calculer toutes les dérivées et dérivées croisées des paramètres. Ensuite, il faut contrôler le résultat obtenu pour éliminer toutes les dérivée nulles, surtout les dérivées croisées inexistantes.

Évidement, cette méthode est simplificatrice. Son utilisation vient en complément (ou en secours) des méthodes précédentes. Lorsque la complexité des paramètres géométriques est assez faible (en fait dans la majorité des problèmes), les deux premières dérivées sont suffisantes. Cependant, la porte est encore ouverte pour la recherche d'approximations locales plus performantes.

3.3.3. La propagation des dérivées

Une fois que les dérivées des noeuds-points et des noeuds-lignes sont calculées, il faut les propager à l'intérieur des faces et des volumes. Lorsqu'un calcul direct à l'aide de la matrice de transformation est possible, c'est la méthode la plus souple et la plus précise. Malheureusement, le calcul direct ne permet pas de traiter toutes les surfaces et tous les volumes d'un modèle B-Rep. C'est pourquoi, ce paragraphe étudiera une méthode numérique permettant de propager de manière générale les dérivées frontalières.

Les méthodes de perturbation de maillages (propagation de déplacements) présentées au paragraphe (§3.2.2), sont utilisables pour propager les dérivées. Dans FLUX3D, pour définir les dérivées de noeuds libres, nous avons utilisé une approximation barycentrique, c'est-à-dire un calcul par moyenne sur les premiers voisins. Ce choix s'explique essentiellement par la simplicité de cette méthode. En mettant chaque noeud au barycentre d'un polygone (2D)/d'un polyèdre (3D), dont les bords sont les arêtes/facettes extérieures de la boule d'éléments connectés à ce noeud, le vecteur de position peut s'écrire (sur le polygone de la Fig.3.8):

$$\mathbf{(3.14)} \qquad \vec{oN} = \frac{\vec{oN_1} + \vec{oN_2} + \vec{oN_3} + \vec{oN_4} + \vec{oN_5} + \vec{oN_6}}{6} = \frac{1}{6}\sum_{i=1}^{6} \vec{oN_i}$$

et les dérivées:

$$\mathbf{(3.15)} \qquad dN = \frac{dN_1 + dN_2 + dN_3 + dN_4 + dN_5 + dN_6}{6} = \frac{1}{6}\sum_{i=1}^{6} dN_i$$

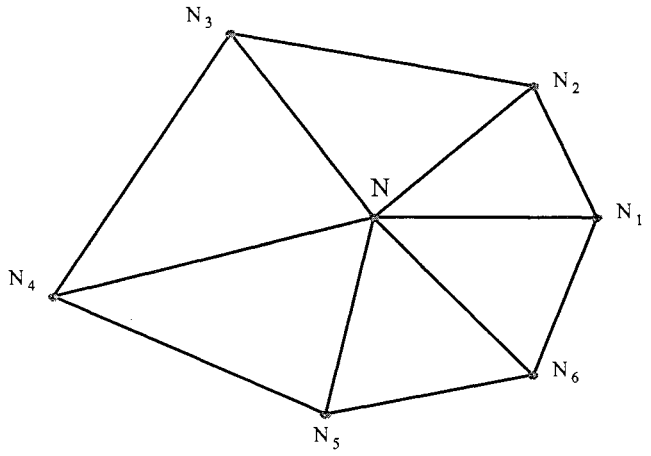


Fig.3.8. Exemple d'une boule de triangles connectés au noeud N

Dans un maillage triangulaire régulier généré par l'algorithme de Delaunay en 2D, cette approche donne un résultat satisfaisant. Si le maillage est fortement déséquilibré, on peut utiliser des coefficients (poids) de type distance pour améliorer l'équation (3.14) et éventuellement l'équation (3.15):

(3.16) $$o\vec{N}\sum_{i=1}^{n}\frac{1}{\overline{NN_i}}=\frac{1}{n}\sum_{i=1}^{n}\frac{o\vec{N}_i}{\overline{NN_i}}$$

(3.17) $$dN\sum_{i=1}^{n}\frac{1}{\overline{NN_i}}=\frac{1}{n}\sum_{i=1}^{n}\frac{dN_i}{\overline{NN_i}}$$

En appliquant (3.15) ou (3.17) à tous les noeuds internes des surfaces et volumes déformés, on obtient un système matriciel linéaire. Comme le système est défini positif, il peut être résolu facilement par une méthode directe, par exemple la méthode de Cholesky. Pour chaque ordre de dérivation, une résolution donne les dérivées. Par évaluations successives, on arrive donc à calculer toutes les dérivées et les dérivées croisées par rapports aux paramètres désirés.

Nous venons de présenter tous les modules d'un dérivateur semi-analytique de maillages. Au niveau de l'organisation des données, l'information primitive du calcul symbolique, les Exp-PI, sont stockées en base de données (FLUX3D). Par contre, pour des raisons d'encombrement mémoire, toutes les dérivées calculées sont enregistrées sur fichier externe. Le lecteur intéressé peut consulter le paragraphe consacré à ce sujet au chapitre suivant (§4.4) pour connaître la structure de stockage des dérivées d'un noeud, l'enchaînement pour les créer (suivant les formules présentées ci-dessus) et aussi les arithmétiques élémentaires sur cette structure.

Avant de présenter les premiers résultats de validation, nous proposons un bilan comparatif des méthodes de dérivations de maillages (Table 3.1).

Table 3.1. Caractéristiques de la procédure de dérivation du maillage

	Dérivation semi-analytique	**Dérivation numérique existante**
Caractéristiques Procédurales		
La géométrie	Paramétrée	- idem
Les paramètres	Paramètres de conception généralisés Stratégie de dérivations successives	- idem Stratégie de variation (ex: par plan d'expérience)
Première phase: Évaluation de la géométrie Résultat	Calcul symbolique des dérivées de l'entité de frontière (point, lignes), (ou numérique si impossible) Dérivées de noeuds frontaliers.	Réévaluer les param. et définir les déplacements des frontières géométriques Déplacements des frontières.
Étape intermédiaire Résultat	- Aucune	Perturbation du maillage, s'appuyant sur les déplacements aux frontières Nuage des positions de noeuds
Phase productive	Propagation des dérivées (ex.: au moyen de voisinage)	Approximation de la dérivée pour obtenir la dérivation numérique
Autres Caractéristiques		
Multi-paramètres	Pas de problème pour la propagation de chaque dérivée	Problème de dérivations croisées dans l'approximation par nuages des positions
Intervalle de variation des paramètres	Pas de besoin explicite.	Doit être prédéfini ou défini par itération a priori.
Rayon de convergence	Dépend fortement de la qualité des éléments du maillage (point faible)	Dépend moins de la qualité des éléments
Implantation informatique	Procédure intégrée dans le logiciel - Outils de dérivation symbolique et dérivation numérique des entités	Procédure semi-autonome - Dérivation procédurale, utilisation de la dérivation automatique.
Exploitation a posteriori de la variation	Variation synchrone de toute la géométrie et de tout le maillage	Difficulté de localisation des variations et propagations inverses sur les entités géométriques
Temps de calcul	Très rapide	Assez long, à cause de l'interfacage de logiciels, peut être amélioré par une implantation directe

3.4. Tests effectués sur quelques exemples

Dans cette partie, nous allons valider nos algorithmes et les outils correspondants, sur divers maillages, de manière à évaluer leurs performances intrinsèques. Les maillages 2D, dont la visualisation est plus aisée, sont choisis pour illustrer les principaux types de perturbation. Concernant le maillage 3D, un exemple complet, accompagné par son histogramme, apparaîtra en fin de paragraphe. Un autre exemple très spécial sera présenté au paragraphe suivant.

a. *La déformation équilibrée*

C'est la déformation la plus simple à traiter. On la rencontre assez souvent en pratique en phase de conception. Un simple assemblage de surfaces ou de volumes convexes, un déplacement de toute une région, conduisent à une déformation (ou déplacement) quasi-proportionnelle de tous les éléments. Lors d'une déformation équilibrée, on peut donc créer des variations très importantes sans craindre d'incohérence du maillage (retournement d'élément), en revanche la dégradation de la qualité des éléments peut être importante (Fig.3.9.).

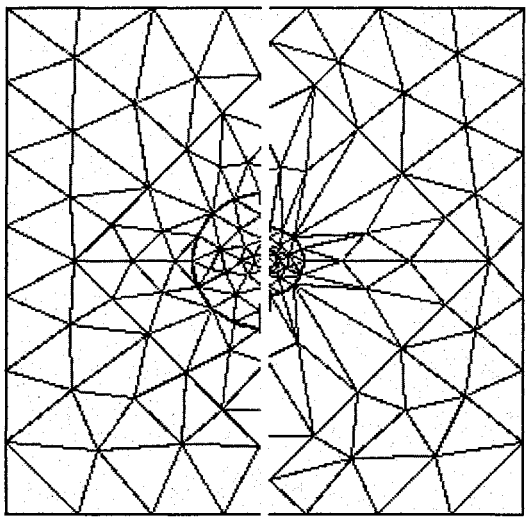


Fig.3.9. Une perturbation par la déformation équilibrée des régions

b. *La déformation locale forte*

Ceci est l'autre type de déformation très courante en optimisation de forme. Lorqu'on fait bouger des points pour des régions convexes, une déformation importante se crée dans la région voisine. Généralement, les effets de déformation sont plus graves dans une région de forme complexe (par exemple une région d'air dans un problème électromagnétique). La dégradation du maillage se produit donc essentiellement dans les parties concaves entourant les pointes déplacées. Nous avons validé notre algorithme sur la géométrie du TEAM Worshop Problème 22 [TEAM22] avec ses six paramètres géométriques. Les résultats d'évaluation du maillage paramétré sont satisfaisants (Fig.3.10)

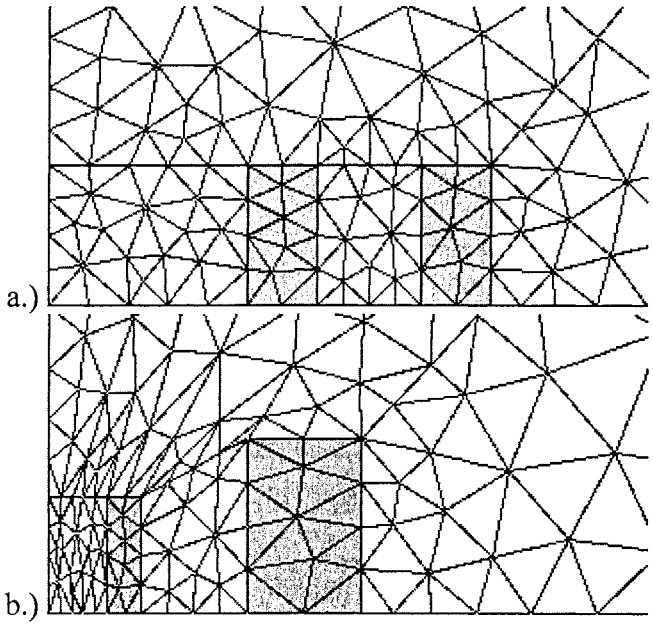


Fig.3.10. La perturbation forte de plusieurs paramètres

	Maillage	
Nombre d'éléments	**a) initial**	**b) perturbé**
Bonne qualité	98.1 %	85.4 %
Moyenne qualité	1.1 %	9.7 %
Médiocre qualité	0.0 %	4.0 %
Mauvaise qualité	0.0 %	0.9 %

c. *La rotation*

Une particularité de l'électrotechnique est l'étude des machines tournantes. Ici, si on a pris la précaution d'utiliser un repère local propre au rotor, la rotation est facilement gérée en

continue avec toute la cohérence souhaitable. En d'autres termes, la déformation se produit exclusivement dans la région entrefer, le maillage du rotor tourne donc sans se déformer.

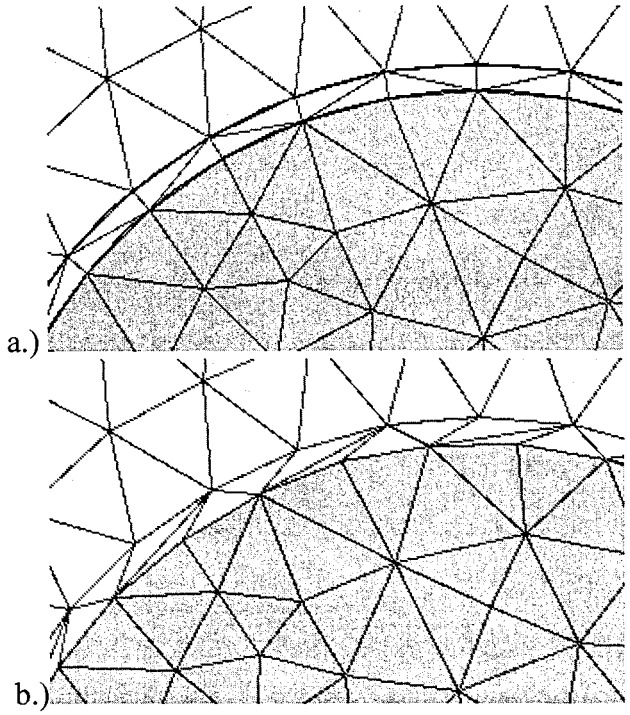


Fig.3.11. La perturbation par rotation 2D
a) Maillage initial b) Maillage perturbé

Bien sûr, cette rotation est limitée à un petit débattement angulaire qui est insuffisant pour étudier le comportement global d'une machine tournante. Son utilisation peut, cependant, apporter des résultats continus sur un intervalle de variation autour de la position angulaire étudiée. Pour une rotation plus importante, il faudra enchaîner plusieurs positions angulaires.

d. *L'exemple 3D*

Enfin, voici un exemple 3D, un contacteur électrique avec une partie mobile pivotante. On utilise un repère pour décrire la rotation et le déplacement de la partie mobile. Le dimensionnement du circuit magnétique est décrit (en fait, sur-décrit) par plusieurs paramètres. Un maillage tétraédrique de densité variable a été généré par un mailleur libre de Delaunay, pour une valeur centrale des paramètres. Dans l'exemple (Fig.3.12), le contacteur est soumis à une déformation non uniforme assez forte des 3 paramètres. L'histogramme montre, cependant, une dégradation acceptable de la qualité des EF.

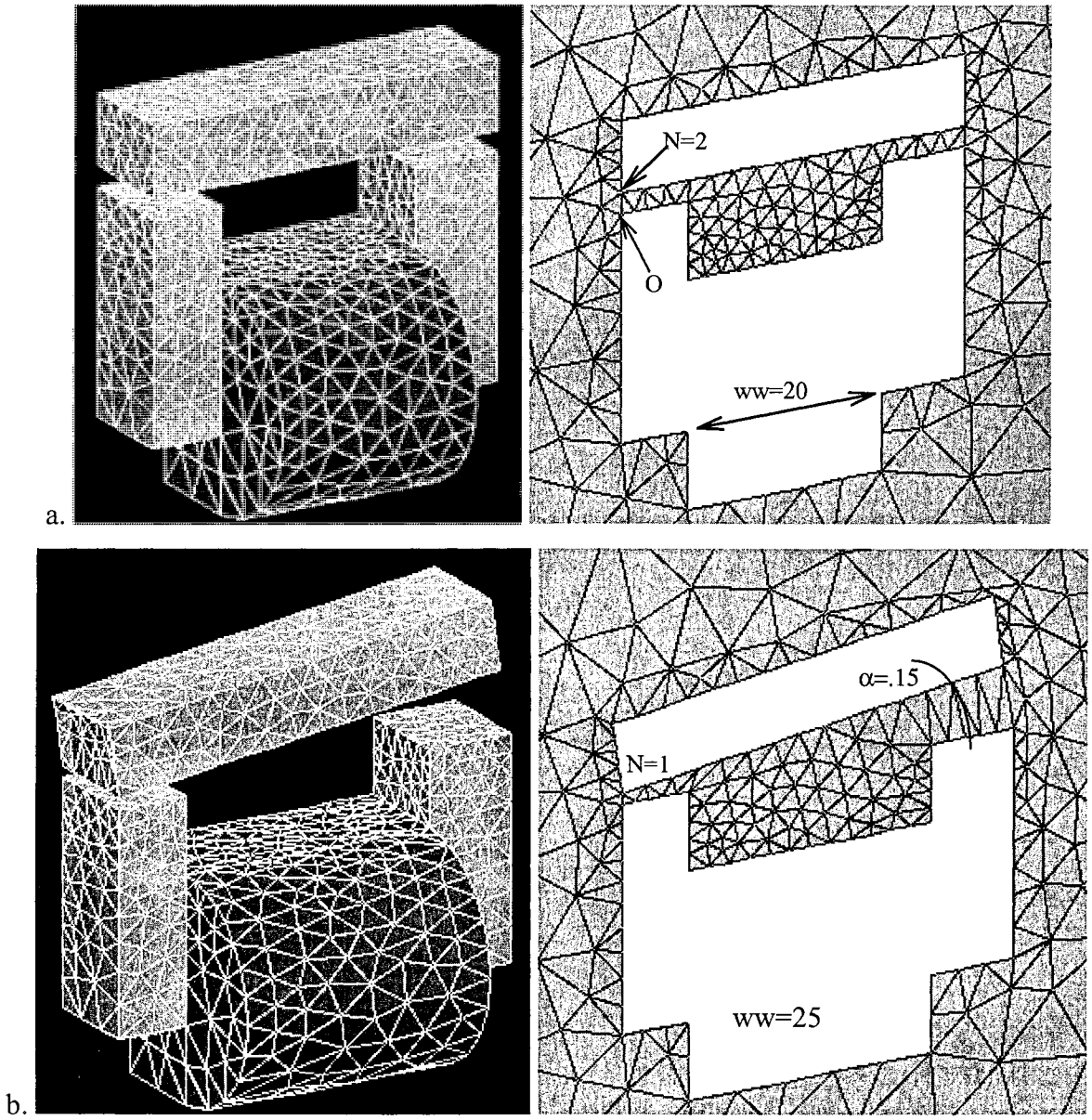


Fig.3.12. Exemple de perturbation du maillage 3D, un contacteur avec 3 paramètres

Nombre d'éléments	Maillage	
	a) initial	b) perturbé
Bonne qualité	46.1 %	43.1 %
Moyenne qualité	34.4 %	35.4 %
Médiocre qualité	14.5 %	16.0 %
Mauvaise qualité	5.0 %	5.4 %

3. 5. Les améliorations du maillage paramétré

Dans ce paragraphe, nous présentons quelques critères et techniques permettant d'améliorer la qualité d'un maillage paramétré. Nous avons donc intérêt à choisir le maillage initial, la topologie géométrique et les techniques de dérivation de telle sorte que le rayon de convergence soit le plus grand possible. Notons que la définition du rayon de convergence géométrique a été donnée au paragraphe (§2.4).

a. *Concernant le Maillage Initial*

Du point de vue de la qualité du maillage paramétré et du rayon de convergence du modèle, le maillage initial a une importance décisive (Fig.3.13).

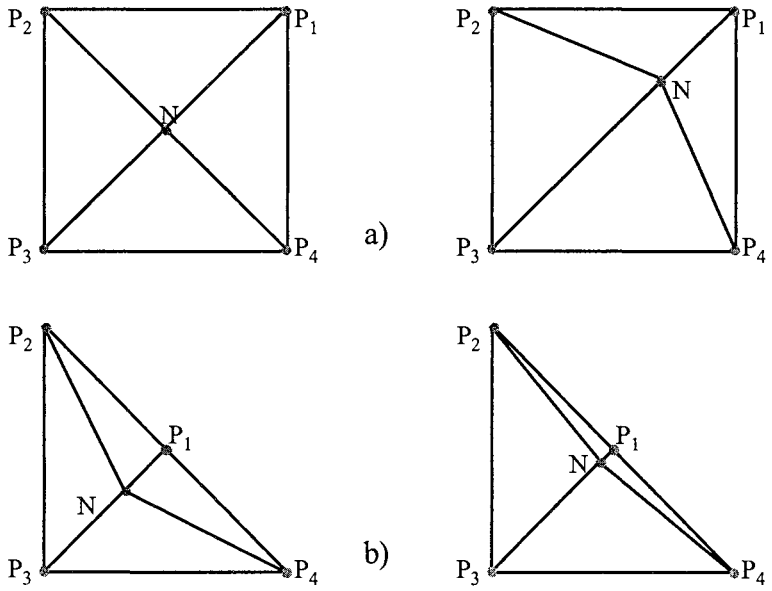


Fig.3.13. Dépendance de la Qualité du maillage perturbé par rapport à celle du Maillage Initial. a) Maillages initiaux, b) Maillages perturbés.

Meilleur est le maillage initial, meilleur est la qualité du maillage paramétré. Mais, à part la qualité globale, il y a quelques points intéressants:

◊ Le choix du point initial de développement. Usuellement, on choisit un développement de Taylor au centre du domaine d'intérêt. Mais ce choix n'est pas toujours le meilleur parce que la déformation géométrique est souvent non symétrique.

◊ Les éléments du maillage. En plus de la qualité, le type et la dimension des éléments influent sur la validité du modèle. En particulier, les éléments rectangulaires sont meilleurs que les éléments simplexes (triangles, tétraèdres). Les grands éléments supportent mieux la déformation que les petits.

b. *Une amélioration de la forme géométrique des régions*

Il est clair que la dégradation du maillage se produit en premier lieu dans les régions de forme complexe, surtout dans leurs parties concaves. On a donc intérêt à imposer certaines contraintes dans les zones sensibles du maillage afin d'éviter une détérioration prématurée. En utilisant des "poids" adaptés dans la formule barycentrique ou en ajoutant des tenseurs dans une transformation structurale, on arrive généralement à consolider les EF dans les zones sensibles. De plus, il existe une technique simple et efficace qui consiste à découper la région complexe (concave) en sous régions de forme simple (convexe), qui réagissent mieux aux déformations. Par exemple, dans un contacteur électrique, la variation du paramètre "entrefer" devient beaucoup plus douce si on ajoute quelques lignes artificielles dans la région boîte d'air (Fig.3.14). Les résultats de cet exemple peuvent être encore améliorés en utilisant des éléments rectilignes et en attachant toute la partie mobile à un repère. Cette technique est bien efficace pour améliorer la qualité du maillage initial. Elle est malheureusement moins évidente à mettre en oeuvre dans un modèle 3D.

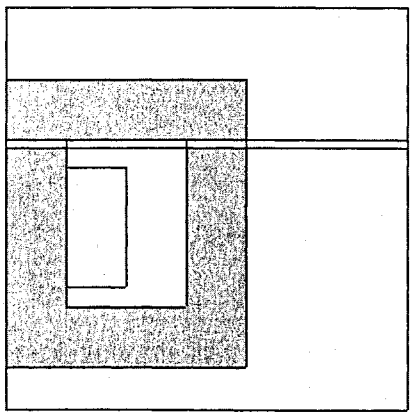
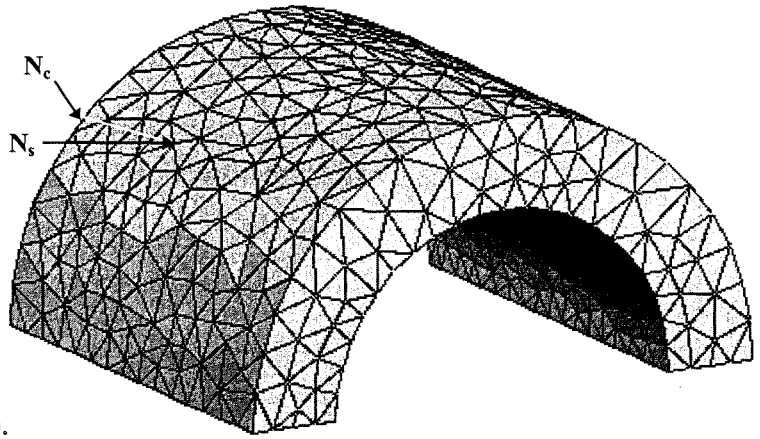


Fig.3.14. Limites des régions perturbées dans un contacteur électrique (2D)

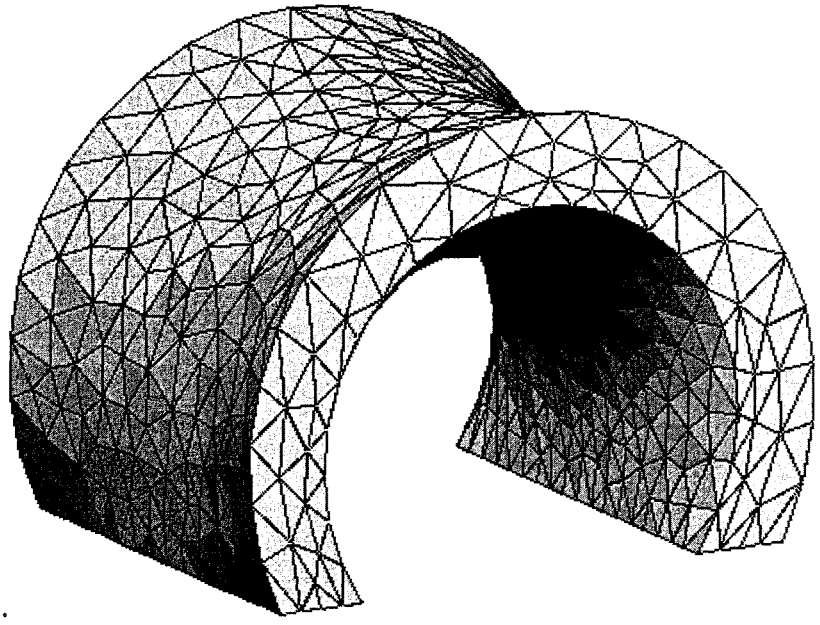
c. *Et une amélioration afin de surmonter les effets 3D surfaciques*

On constate que le passage du 2D au 3D en MEF n'est pas qu'une simple montée en dimension. En effet, il amène des problèmes supplémentaires.

Pour un maillage paramétré 3D, la difficulté majeure consiste à conserver la forme des surfaces non planes lors de la mise en oeuvre des perturbations. Les noeuds peuvent s'éloigner de la surface qui les porte lorsqu'on impose un fort changement de courbure. C'est un point faible commun aux méthodes basées sur les maillages élastiques (Fig.3.15).



a).



b).

Fig.3.15. L'effet de la déformation élastique.
a). Maillage initial b). Un fort changement de courbure

En fait, cette imprécision est liée à la nature même de la représentation géométrique. On peut donc l'éviter si la perturbation s'appuie sur une modification analytique des noeuds surfaciques, c'est-à-dire, si les noeuds sont définis sur un carreau surfacique. Dans notre démarche, où la géométrie est à base de B-Rep, les dérivées surfaciques sont obtenues par une propagation barycentrique de dérivées frontalières (noeuds-lignes), par conséquent, l'effet surfacique pose des problèmes assez graves. S'appuyant sur le calcul symbolique (§3.3), nous avons construit deux types d'amélioration "manuelle":

◊ *Libération d'une composante des dérivées.*

En l'absence d'une représentation surfacique précise, on propose une amélioration a posteriori à l'aide des expressions explicites données par l'utilisateur. En effet, on refait calculer une dérivée (dx, dy ou dz) à partir des autres dérivées et des coordonnées du noeud. Par exemple, si la surface est définie par ($x^2+y^2+z^2=R^2$) la correction pour dx est:

$$dx= (dR - ydy - zdz)/x$$

C'est un développement qui est orienté vers le modèle surfacique, mais qui est dans l'état actuel très limitée car elle nécessite des interventions de l'utilisateur.

◊ *Amélioration à l'aide du concept de la transformation géométrique.*

C'est aussi une méthode assistée, dont nous allons expliquer le principe sur un exemple. Reprenons les surfaces cylindriques ci-dessus (Fig.3.15a). En l'absence de déplacement axial, on constate que les dérivées du noeud surfacique Ns{x,y,z} sont identiques à celles d'un noeud-ligne Nc dont la coordonnée curviligne k est à calculer (Fig.3.15a). Le dérivateur symbolique peut identifier automatiquement le cas si la surface est construite par l'extrusion d'une ligne. Dans le cas contraire, cette tâche doit être remplie par une expression explicite. Dans le cas simple de cet exemple, la coordonnée curviligne a été calculée par: $k = arctg(y/x) / \pi$.

En définitive, on a réalisé un bon maillage paramétré, dont l'évaluation dans les conditions équivalentes à celles de la (Fig.3.15b), est donnée dans la figure (Fig.3.16).

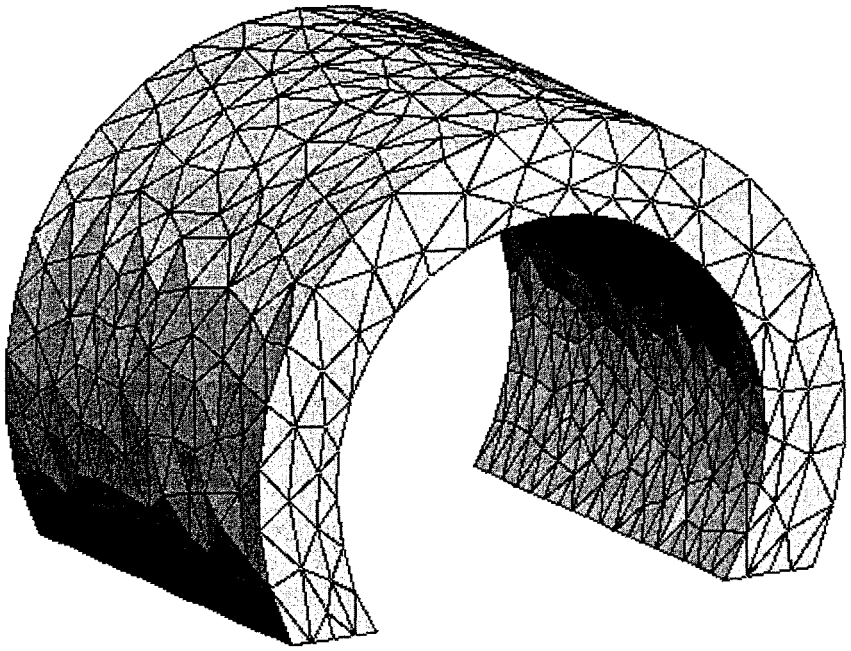


Fig.3.16. Mise en surface à l'aide du calcul symbolique.

On conclut cette partie par un petit bilan des performances des méthodes. Les études sur un seul paramètre montre que la transformation structurale supporte mieux les déformations convexes assez fortes, par contre, elle accepte mal la rotation ou le changement de courbure qui sont le point fort de notre méthode semi-analytique. En ce qui concernant les traitements multi-paramètres, nous n'avons pas pour l'instant de conclusions générales.

Conclusion et perpectives de développement

Après avoir fait le bilan des étapes de construction d'un maillage paramétré, nous avons proposé une méthode originale de dérivation du maillage, qui est basée sur une dérivation analytique des frontières géométriques. Étape par étape, nous avons alors détaillé toutes les particularités de l'implantation de cette méthode dans un grand code de calcul. Ensuite, pour valider cette méthode, nous avons donné quelques exemples de maillages paramétrés, en détaillant la réalisation et l'optimisation des points clés. Les expériences montrent que cette nouvelle approche est très rapide et qu'elle donne généralement des résultats satisfaisants. Enfin, quelques outils interactifs ont été réalisés pour améliorer la qualité du maillage 3D paramétré suivant les besoins de l'utilisateur.

Les développements envisageables sont les suivants:

◊ Amélioration de l'algorithme de propagation des dérivées frontales à l'intérieur des surfaces et des volumes.

◊ Utilisation d'éléments spéciaux à topologie constante (par exemple macro-élément) ou d'approximation nodale sans éléments dans les déplacements spécifiques à l'électrotechnique (rotation du rotor, translation du noyau,...).

Le calcul paramétré est relativement nouveau dans la pratique de la MEF, ce chapitre présentait la construction du maillage paramétré. Notre préoccupation suivante va être la réalisation d'un solveur paramétré.

Chapitre 4

MISE EN OEUVRE DU SOLVEUR PARAMÉTRÉ

Sommaire

Chapitre 4

MISE EN OEUVRE DU SOLVEUR PARAMÉTRÉ

Nous avons présenté notre approche de résolution EF paramétrée dans le deuxième chapitre et dans le chapitre précédant nous avons mis en oeuvre le dérivateur du maillage. L'étape suivante consiste à organiser la résolution paramétrée, c'est-à-dire une boucle de résolution EF accompagnée par une procédure d'assemblage paramétré. En entrée de cette étape, on dispose d'un modèle discrétisé dont la position des noeuds se présente sous forme de développements de Taylor par rapport aux paramètres géométriques. À la fin, nous devons obtenir les polynômes de Taylor pour toutes les variables d'état.

Signalons qu'une mise en oeuvre a été effectuée par les sociétés CADOE et CEDRAT, pour obtenir FLUX-PARAM, un premier produit 2D. De son côté, Pascal PÉTIN a testé, dans l'environnement SIC, une implantation par rapport aux paramètres physiques seuls. En ce qui nous concerne, nous avons repris ce travail pour implanter la méthode dans FLUX3D en axant nos efforts sur les paramètres géométriques. Nous avons cette fois travaillé sur la nouvelle organisation des calculs, en s'appuyant sur un langage interprété. Avant de parler de la mise en oeuvre, nous présenterons les modules du solveur EF et les méthodes de calcul des dérivées. Nous préciserons ensuite notre choix et nous détaillerons la procédure de mise en oeuvre de cette approche. Nous porterons

une attention toute particulière sur l'implantation informatique du module qui a une structure ouverte basée sur un langage interprété.

4.1. La résolution EF classique

Dans le premier chapitre de ce mémoire nous avons abordé le fonctionnement du module de résolution EF (le processeur) dans une chaîne de calculs EF. Nous présentons maintenant les détails de sa réalisation, et quelques généralités, qui seront, par la suite, à la base de l'implantation du solveur paramétré.

La tâche de ce module est de construire et résoudre un système matriciel: $[\mathbf{M}].\{\mathbf{A}\} = \{\mathbf{S}\}$

La figure (Fig.4.1) présente les trois blocs d'opérations de la résolution EF.

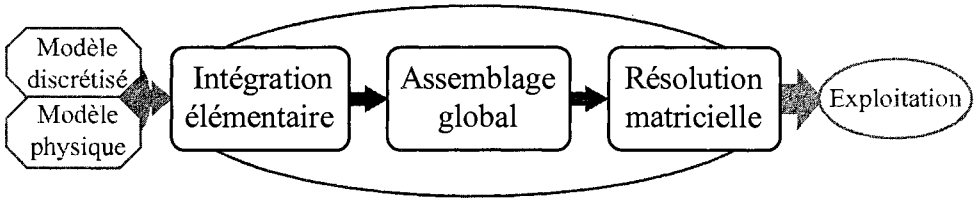


Fig.4.1. Un solveur EF classique

a. L'intégration élémentaire

Cette opération se réalise indépendamment sur chaque élément fini pour obtenir la matrice et le vecteur élémentaires. En pratique, les intégrales d'assemblage (ex: les équations (2.9), (2.10) et (2.11)) sont calculables par différentes méthodes d'intégration. La plus adaptée au calcul EF est la méthode d'intégration numérique par des points de Gauss. Le principe de cette méthode consiste à remplacer le calcul de l'intégrale d'une fonction par le somme des valeurs pondérées de cette fonction en certains points prédéfinis. Par exemple, l'intégrale exacte d'un polynôme à l'ordre 2n+1 est calculé par:

(4.1) $$\int_{-1}^{1} f(u)\,du = \sum_{i=1}^{n} \kappa_i\, f(u_i)$$

où u_i sont les points (abscisses) de Gauss et κ_i sont les coefficients (poids)

$f(u_i)$ est la valeur de f au point u_i

En pratique, l'intégration EF se réalise en s'appuyant sur l'élément de référence. L'élément géométrique "réel" Ω^e se ramène à un élément de référence Ω^r dans un système de coordonnées $\mathbf{u}(u,v,w)$, dit "local", par une transformation géométrique τ. On utilise souvent une transformation identique pour trois coordonnées, qui est basée sur les mêmes fonctions que celles de l'interpolation EF.

$$\tau: \mathbf{u} \rightarrow \mathbf{X}(u) = \langle \alpha(u) \rangle \{X_n\}$$

où $\{X_n\}$ sont les valeurs nodales d'une grandeur quelconque (ou les coordonnées des noeuds)

$\langle \alpha(u) \rangle$ est l'ensemble des fonctions de forme (fonctions de base)

Les éléments de référence doivent être définis par le concepteur, ils sont les EF de forme normalisée (carrée, cube, triangle rectangle, etc.). Toutes les informations nécessaires sont stockées dans la bibliothèque d'éléments de référence:

◊ le nombre des points de Gauss (la précision de calculs) de l'élément,

◊ les coordonnées locales $\mathbf{u}_i\{u_i,v_i,w_i\}$ et les coefficients κ_i des points de Gauss,

◊ les fonctions de forme $\langle \alpha(\mathbf{u}) \rangle$ (en local),

◊ les gradients des fonctions de forme $\langle \nabla_u \alpha \rangle$ (aux points de Gauss).

Généralement, l'intégration est faite élément par élément. Cette procédure nécessite trois opérations:

☞ Premièrement, il faut récupérer les "données" de l'élément Ω^e:

La définition géométrique: le type de l'élément, la liste des noeuds et leurs coordonnées. Le modèle physique associé: les formulations actives, les propriétés physiques des matériaux, les contraintes (sources et condition aux limites)

☞ Ensuite, en s'appuyant sur le type d'élément (et la précision désirée) on définit les caractéristiques de l'élément de référence Ω^r et récupère le modèle d'intégration.

☞ Enfin, on calcule les intégrales, en utilisant ce modèle de référence et les "données" de l'élément réel Ω^e.

Remarquons qu'une organisation très efficace de l'assemblage EF est proposée dans [Coul95i]. Le principe consiste donc à réaliser le calcul par paquet de plusieurs éléments de même type.

b. La procédure globale d'assemblage

L'opération d'assemblage est la procédure qui consiste à remplir le système matriciel global (matrice globale et vecteur global) avec les matrices et les vecteurs élémentaires calculés. Ici, le principe est assez simple, ce sont des additions de vecteurs et de matrices. Mais l'organisation pratique de cette opération est beaucoup plus complexe à cause principalement de la dimension importante du système. Il faut donc utiliser un mode de stockage adapté, par exemple le stockage "bande", "ligne de ciel", "Morse" ou même sur la mémoire auxiliaire (sur "disques"). La numérotation des variables est un traitement supplémentaire, qui permet d'utiliser efficacement le stockage et qui améliore la performance de l'algorithme de résolution choisi.

En général, l'assemblage contient une affectation complète des conditions aux limites et quelques traitements spéciaux en fonction des besoins de la formulation (libération des composantes, préparation pour l'enchaînement, etc.)

c. La résolution du système matriciel

Les méthodes de résolution du système matriciel sont nombreuses et bien maîtrisées. On doit choisir l'une ou l'autre selon la nature du problème (linéaire ou non-linéaire, statique ou dynamique, simple ou enchaîné). Pour le problème non-linéaire, la plus utilisée est la méthode de Newton-Raphson. La procédure de résolution d'un système linéaire utilise fréquemment la méthode ICCG*(angl: Incomplete Cholesky Conjugate Gradient), qui est bien adapté aux systèmes creux, par contre, dans une procédure de la résolution enchaînée (comme dans notre approche) il vaut mieux utiliser une méthode de factorisation complète.

4.2. Les méthodes de dérivation

Dans le chapitre 2, nous avons décrit de façon générale, en nous appuyant sur quelques exemples simples, les approches de dérivation du système d'équations EF. Dans le chapitre 3 nous avons présenté des méthodes de dérivation du maillage. Nous reprenons ici ces méthodes générales pour la mise en oeuvre d'un module-dérivateur de la solution EF. Nous nous préoccuperons principalement de deux approches: la dérivation directe grâce à la différenciation symbolique et l'utilisation des techniques de différenciation automatique.

4.2.1. Les différences finies (les Différences divisées)

L'avantage de la méthode des différences finies est sa simplicité: on a juste besoin du module standard de calcul en tant que "boite noire". Pourtant sa précision est difficile à contrôler et elle n'est applicable en pratique que pour les dérivées du premier et du second ordre. En général, on peut utiliser cette méthode pour la dérivation du maillage mais elle est insuffisante pour la dérivation de la solution, où il faut calculer avec précision des dérivées d'ordre élevé et plusieurs dérivées croisées par rapport aux paramètres.

4.2.2. La dérivation directe

La dérivation analytique est toujours souhaitable, c'est la solution la plus précise et la plus facile à interpréter par l'utilisateur. Elle permet de "toucher" à la physique du problème et consiste donc à dériver les formulations. Dans le chapitre 2, nous avons présenté quelques interprétations simples de cette approche. En général, la solution totalement analytique est inaccessible. On a recours alors au calcul symbolique pour définir les formules de base, puis à l'intégration numérique de ces formules pour la phase de calcul.

La Différenciation Symbolique (DS)

La différenciation symbolique est la procédure assistée par l'ordinateur de différenciation analytique. La représentation symbolique des fonctions est transformée par la logique de la

machine en formule permettant de calculer les dérivées d'ordre quelconque par rapport aux paramètres (les arguments des fonctions). La différenciation symbolique elle-même est une partie de l'algèbre symbolique, dont les versions existent dans Maple, Masyma, Matlab.... C'est un outil puissant de manipulation des expressions algébriques mais l'interprétation peut être très coûteuse à cause de la génération inefficace de plusieurs exemplaires inutiles de même sous-expressions.

Dans une analyse de sensibilité d'ordre élevé, nous remarquons que si on utilise exclusivement les développements de Taylor comme type de (représentation de) fonctions, on peut éviter ce dernier point faible.

La dérivation du solveur EF

Il est clair que la différenciation symbolique (et la dérivation analytique) ne peut pas prendre en compte automatiquement les constructions comme les branchements, les boucles ou sous-programmes cachés dans le code de calcul. Autrement dit elle est inapplicable pour dériver globalement toute la procédure de résolution EF. Le calcul doit donc être organisé au niveau de l'intégration élémentaire (§4.1.1 et §2.3). Dans notre optique, il y a deux possibilités:

- Dans la première, les intégrales sont calculées d'abord analytiquement, la dérivation se réalise ensuite sur les résultats de l'intégration (ex: les équations (2.21), (2.22), (2.23)).
- Dans la seconde, les expressions sous les intégrales sont dérivées pour obtenir les formules "dérivées" (ex: les équations (2.26) et (2.30)). Ensuite, on peut calculer les dérivées désirées en appliquant les intégrales sur les formules obtenues. Par exemple, en utilisant la méthode de Gauss, on a alors besoin de calculer les valeurs des formules dérivées aux points de Gauss.

Dans les deux cas, les dérivées des variables d'état sont obtenues par des résolutions successives de systèmes matriciels (voir paragraphe §2.2).

4.2.3. La Différenciation Automatique (DA)

C'est une méthode purement informatique qui permet d'évaluer assez rapidement la dérivée d'ordre quelconque d'une fonction. En principe, c'est une méthode très similaire à la différenciation symbolique citée ci-dessus, sauf que la dérivée est calculée sans la génération explicite de la formule analytique.

a. Le principe de la méthode

La différenciation automatique est une technique qui permet, à partir du programme source, de construire la partie concernant le calcul automatique des dérivées. Cette technique est basée principalement sur la composition enchaînée de la dérivée. On sait que chaque fonction est codée dans l'ordinateur par une chaîne (qui peut être très longue) d'opérations élémentaires comme l'addition, la multiplication, etc. et de fonctions élémentaires comme "sin", "cos", etc.. En appliquant récursivement la "règle de dérivation en chaîne":

$$\frac{\partial}{\partial t} f(g(t))\Big|_{t=t_0} = \left(\frac{\partial}{\partial s} f(s)\Big|_{s=g(t_0)}\right)\left(\frac{\partial}{\partial t} g(t)\Big|_{t=t_0}\right) \tag{4.2}$$

pour chaque composition des fonctions élémentaires, la dérivée peut être calculée jusqu'à la précision de la machine.

Différentes techniques sont développées dans la différenciation automatique. Certaines permettent même de dériver "les exceptions" telles que les fonctions "max","min","mod",... et les structures de branchements. Nous renvoyons à [ADIFOR] et [ADOL-C] pour les aspects théoriques mathématiques et aussi pour les aspects techniques informatiques de la différenciation automatique. Dans le paragraphe (§4.4.5), on proposera un exemple d'utilisation de la "composition en chaîne" dans le traitement des polynômes de Taylor.

b. Les différenciateurs automatiques

En général, la réalisation informatique de la technique de différenciation automatique est très complexe, l'implantation des outils de base est considérable, de plus, cette implantation dépend encore de la plate-forme informatique (DOS, MAC, SUN, HP, etc.). Par bonheur, de nombreux produits de démonstration se trouvent dans le domaine public (sur Internet par exemple). Les

modules de différenciation automatique (les différenciateurs automatiques) proposent des bibliothèques d'outils de base qui assurent la génération "automatique" des codes de dérivées et qui permettent de calculer ensuite les dérivées désirées à partir de codes enregistrés. En pratique, l'utilisateur (le développeur du programme de calculs) doit réaliser un couplage fort entre le logiciel (le code de calcul) et les outils de différenciation automatique.

Les différenciateurs automatiques peuvent être classés en deux groupes: les "précompilateurs" (de codes) et les "surchargeurs" (de types). Entre eux, la différence est grande du point de vue de l'utilisateur.

❶ ***Précompilation***: Le code existant est traduit en un code "dérivée", qui doit ensuite remplacer totalement le code original. Les outils du différenciateur automatique assurent donc les analyses et les contrôles nécessaires pour la traduction. Notons que les idées de DA ont été initiées dans cette direction par Bert Speelpenning (1980), ce travail ensuite s'est concrétisé dans le précompilateur JAKE (JAKEF). Les autres précompilateurs plus récents sont GRESS/ADGEN, PADRE2 et surtout ADIFOR [ADIFOR].

❷ ***Surcharge****(angl:Overloading): Cette technique est plus moderne, elle permet de remplacer (surcharger) toutes les fonctions et les opérateurs reliées à un "type de données" par les nouvelles procédures. En utilisant la "surcharge", il n'y a pas besoin de génération de codes intermédiaires. Théoriquement, il suffit d'effectuer des modifications mineures, principalement dans la déclaration de type des variables et des fonctions. Pourtant la surcharge est moins tolérante concernante les programmes, elle exige généralement une structure moderne du programme original (programmation en langage C, C++, etc.).

Le représentant le plus connu (dans le domaine public) de cette famille est ADOL-C (et ADOL-F) [ADOL-C]. Signalons que le produit industriel ADOC de CADOE S.A. [ADOC] utilise lui-aussi le principe de "surcharge".

Il faut dire que la plupart des différenciateurs automatiques se limitent aux dérivées premières; ils sont surtout optimisés pour le calcul du gradient de fonctions. Les outils gérant les dérivées supérieures sont apparus pourtant dans quelques produits (ADOLC, ADIFOR, ADOC).

En utilisation, on distingue deux modes de déroulement du différenciateur automatique qui sont basés sur deux méthodes différentes de la génération de dérivées.

- Propagation en avant*(angl: forward propagation): Le calcul des gradients des fonctions se réalise en même temps que le calcul des fonctions. Le coût de différenciation est donc proportionnel au nombre de paramètres (entrées).
- Reverse propagation*(angl: reverse propagation): Il faut enregistrer toute la trace d'exécution pour une analyse a posteriori qui va produire la trace pour calculer les dérivées. En général, le coût de reverse propagation ne dépend ni du nombre de paramètres (les entrées), ni du nombre de variables d'état à calculer (les sorties), mais l'enregistrement est peut-être très long et la lenteur du post-traitement est aussi essentielle.

Notons que l'utilisation du différenciateur automatique est encore limitée dans le calcul EF. Les raisons sont leur absence de quelques plates-formes informatiques et l'insuffisance de leur performance. Mais le plus souvent, la raison principale se trouve dans les codes EF, qui sont trop complexe, et peut-être trop anciens.

Enfin, nous pouvons imaginer deux manières d'utiliser la technique de différenciation automatique dans notre approche de dérivation de la résolution EF:

⇒ L'utilisation dans une procédure globale: Il faut dériver toute la procédure de résolution EF. Le différenciateur est activé une fois en entrée et il doit générer toutes les dérivées des variables d'état par rapport aux paramètres de conception, en s'appuyant sur les dépendances décrites aux noeuds du maillage. Ici, l'implantation et l'adaptation sont très importantes, la complexité augmente donc rapidement avec la taille du programme.

⇒ Une application locale: Le traitement est similaire de celui de l'approche semi-analytique ci-dessus. Le différenciateur automatique est utilisé pour définir les dérivées de la matrice et du vecteur élémentaire. Les dérivées des variables d'état sont obtenues ensuite par des résolutions successives du système matriciel. Le calcul est probablement un peu plus lent, par contre, les résultats sont directement exploitables (sans différenciation automatique). Ceci peut être un bon compromis entre le coût d'implantation du programme et le temps de dérivation.

4.3. Les particularités de la réalisation

Dans cette partie, nous voulons réaliser une analyse comparative des implantations possibles d'un dérivateur de la résolution EF. En proposant quelques suggestions d'améliorations, nous définirons alors notre choix de mise en oeuvre du solveur paramétré dans FLUX3D.

4.3.1. Les problèmes de l'interface

La dérivation du solveur EF est une étape dans le calcul par la méthode de dérivées d'ordre élevé. Les critères de compatibilité et d'échange d'informations sont importants.

En entrée, le traitement des paramètres géométriques exige un couplage fort avec le dérivateur de maillage. Il faut donc assurer la transparence des données en réalisant les interfaces nécessaires. À la sortie du solveur, les résultats seront transmis au module d'exploitation. Une représentation compatible des données est très souhaitable, ce qui permet de réutiliser au maximum les outils existants dans l'exploitation. Notons que ces outils sont généralement spécifiques et assez complexes à reconstruire. En pratique, la compatibilité totale n'est pas toujours facile, il faut y penser depuis le début de la réalisation.

Enfin, les résultats (et les données intermédiaires) sont conservés en mémoire secondaire (sur disques). Cette solution est absolument nécessaire pour traiter le volume très important de données "supplémentaires". Cela implique donc un traitement supplémentaire à optimiser.

4.3.2. Les problèmes liés aux paramètres de dérivation

Dans le chapitre 2, nous avons parlé des paramètres de conception, de leurs spécificités. Nous avons alors remarqué plusieurs points permettant d'améliorer le calcul de dérivées. En ce qui concernent la dérivation du solveur, cela se résume par:

◊ Les paramètres "utilisateur" peuvent parfois être remplacés par les variables "internes" équivalentes, lorsque c'est préférable pour le calcul de dérivées. Cette technique est surtout efficace pour le traitement des paramètres physiques (§2.3).

◊ Dans le cas des paramètres géométriques, il existe des associations de variables qui sont équivalentes à des variables. Par exemple, dans le dispositif (Fig.4.2), la région "3" dépend, en réalité, d'une seule "super-variable" $ll=l_1+dd+l_2$.

◊ Enfin, le paramètre géométrique est à ignorer s'il ne déforme pas la région géométrique. Par exemple, la région "2" (Fig.4.2) est changée lorsque les paramètres "l_1" ou "dd" sont modifiés, mais le paramètre "dd" est le seul qui déforme la région et donc le seul à traiter (§2.3.3).

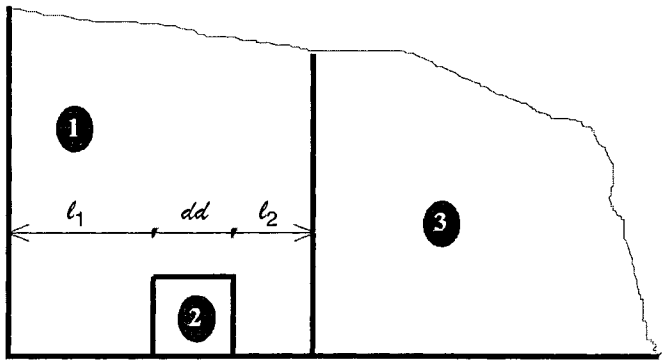


Fig.4.2. Exemple de l'influence des paramètres géométriques

Précisons que la détection puis le traitement des relations spéciales ci-dessus permettent de réduire efficacement le volume des calculs et à un certain niveau d'augmenter la précision. Cependant c'est un travail assez complexe.

4.3.3. La différenciation automatique et La différenciation symbolique

L'intérêt de la différenciation automatique (DA) réside dans sa capacité à traiter globalement (et de manière systématique) toute la procédure de la résolution EF. Quant à la dérivation directe, par différenciation symbolique (DS), elle est un traitement spécifique de l'algorithme de calculs.

Lorsqu'on crée un tout nouveau code bien adapté au calcul de dérivées, la technique de DA a donc un avantage très net. Par contre, si l'on part d'une plate-forme de calculs complexe déjà existante, le développement d'un module de calculs symboliques indépendant paraît beaucoup plus abordable. D'ailleurs, une implantation globale de DA est souvent trop compliquée. Elle

demande, en particulier, un accès à toutes les sources du programme, ce qui est bien souvent impossible. On peut toutefois avoir recours à une application locale de DA, c'est-à-dire une implantation pour traiter les fonctions "élémentaires". Mais dans ce cas, la DA ne présente plus d'avantages par rapport à la DS. En effet, la difficulté principale réside dans le contrôle et dans la localisation des codes, qui sont parfois beaucoup plus complexes que la création de nouveaux codes de dérivées.

En ce qui concerne l'implantation de base, les modules de DA sont généralement prêts à utiliser, néanmoins, il faut créer plusieurs outils pour assurer l'adaptation des données et pour communiquer avec les parties existantes. Dans une approche de DS, pratiquement tous les outils de base sont à créer. Notons cependant que le traitement par développement de Taylor est relativement simple. En effet, les opérations se réalisent exclusivement sur des polynômes (voir §4.4).

4.3.4. Les problèmes divers

En ce qui concerne la mise en oeuvre informatique, il existe de nombreux objectifs contradictoires:

◊ Dans la réalisation de base du programme, il faut choisir entre des orientations favorisant l'algorithme de calculs ou la gestion des "données"; la vitesse d'exécution ou l'évolutivité des codes. Du côté des langages de programmation, un langage simple comme le "vieux" FORTRAN s'adapte bien aux algorithmes complexes, il reste encore une référence pour le calcul lourd (le calcul EF en particulier). De l'autre côté, les langages "modernes", surtout les "orientés d'objet", donnent une grande souplesse pour manipuler de données, ce qui apporte une autre point de vue dans le développement qui devient plus évolutif. Notons que la "surcharge" (de DA) exige un éditeur de liens "moderne" qui supporte cette méthode spéciale.

◊ Une autre contradiction se présente entre la réutilisation (l'adaptation) et la reconstruction des outils. En pratique, de nombreuses versions similaires d'une même "opération" peuvent coexister dans un énorme code.

Les explications des expériences acquises sont rares ou ont été perdues. En définitive, la reconstruction est parfois beaucoup plus simple que la réutilisation de vieux codes

(chercher, récupérer et s'adapter).

Remarquons que les normes de programmation et d'échange des objets facilitent la réutilisation. La notion de langage de commande spécialisé est aussi une approche permettant de mieux comprendre et utiliser les ressources existantes.

◊ La dernière remarque repose sur le contexte de développement. On distingue le développement à "long-terme", dont l'objectif est de construire un produit industriel fiable, et le développement à "court terme" qui correspond à un objectif d'expérimentation.

Du point de vue du calcul paramétré, la programmation objet est préférable. En effet, plusieurs types de "données" doivent être gérées globalement. En particulier, il y a de nombreuses opérations complexes dans le traitement symbolique ou dans la DA. Leur efficacité ne peut être assurée qu'en utilisant plusieurs relations directes (par exemple les pointeurs dans les langages C ou C++).

4.3.5. Le choix d'une procédure assistée

En ce qui concerne la dérivation du solveur EF, notre choix a été dicté par les circonstances suivantes:

◊ Le respect des nos objectifs initiaux: Il s'agissait de créer une chaîne (indépendante) de calcul paramétré dans la chaîne initiale de calcul EF en assurant la cohérence de fonctionnement avec les parties existantes.

◊ La complexité et la diversité des approches et leurs particularités d'implantation.

◊ La limitation du temps imparti: La durée des développements nécessaires devait être compatible avec la durée d'une thèse.

Une implantation globale basée sur la technique de DA est envisageable. Mais nous avons dû abandonner cette idée à cause de l'exigence d'une réalisation informatique très lourde (de tous niveaux) et la limitation à la durée d'une thèse. Nous avons donc choisi un développement par étapes, basé sur la dérivation des intégrales élémentaires. Nous avons fait tout particulièrement attention à l'organisation efficace du module favorisant la poursuite des développements.

Le noyau de cette implantation est un interpréteur pour simuler la dérivation élémentaire (et assemblage global du vecteur second membre). En s'appuyant sur les commandes élémentaires, enregistrées séparément, l'interpréteur simule toute la procédure. Il y a donc deux parties (deux modules) à réaliser:

⇒ l'ensemble des outils qui vont assurer toutes les opérations élémentaires. Dans la dérivation semi-analytique, ce sont essentiellement les outils de calculs sur les polynômes.

⇒ les enregistrements des commandes et les assistants reliés.

Les divers outils d'interprétation sont réalisés principalement en langage C qui donne plus de souplesse dans la programmation. L'interface et la connexion sont, par contre, écrites en FORTRAN qui assure la compatibilité nécessaire avec le corps du logiciel.

4.4. Le traitement des polynômes de Taylor

Les polynômes de Taylor ont été utilisés dans le maillage paramétré (chapitre 3) pour représenter la position des noeuds. Comme les polynômes des noeuds sont généralement compacts et leur traitement dans la dérivation du maillage est assez simple, cette structure n'a pas été explicitée jusqu'ici.

Par contre, le polynôme est l'entité fondamentale dans la phase de résolution paramétrée. Dans notre implantation, le polynôme de Taylor est aussi le seul type de représentation des dépendances par rapport aux paramètres: en entrée ce sont les polynômes des noeuds et en sortie ce sont les polynômes des variables d'état. Dans ce paragraphe, nous présentons tout d'abord le principe et la structure pratique permettant un stockage opérationnel des polynômes de Taylor. Nous décrivons ensuite quelques opérations élémentaires sur les polynômes, ceux qui vont servir à réaliser les calculs dans une partie ultérieure de ce chapitre. L'objectif est de présenter l'efficacité et la souplesse des réalisations informatiques nécessaires dans ce type de traitement.

Naturellement, une telle présentation sera un peu technique, mais nous pensons qu'elle a sa place ici. Premièrement, nous considérons que les aspects informatiques sont devenus importants dans tous les travaux de recherche. L'implantation informatique du dérivateur de solution EF qui est assez complexe, mérite donc quelques développements. Deuxièmement, les analyses de solutions proposées sont elles-mêmes une partie de l'expérience acquise. Certes, ces solutions peuvent être éloignées des préoccupations du programmeur professionnel, elles n'en méritent pas moins l'attention des "développeurs par nécessité", qui seraient un jour, amenés à poursuivre ce travail.

4.4.1. L'ordre de priorité des paramètres

Le besoin d'une priorité s'explique principalement par deux raisons:

◊ Pour éviter les répétitions de dérivées dans un polynôme et pour assurer l'uniformité de la représentation de tous les polynômes dans les parties symboliques du traitement.

◊ Pour synchroniser les données, ce qui permet de réaliser rapidement les fusions possibles (addition, suppression, multiplication...) sur les polynômes de Taylor.

L'ordre unique de priorité doit être défini donc globalement pour la liste complète de tous les paramètres. En pratique, le choix le plus efficace consiste à utiliser une priorité qui est conforme au numéro d'identification des paramètres (les PA-DRV, voir §3.1.1). Par exemple, dans notre réalisation la création (et stockage) de dérivées est effectuée en sens inverse des numéros de PA-DRV reliés. En effet, la montée dans l'ordre de dérivation est effectuée d'abord pour le paramètre de numéro le plus grand (si cela est encore possible, c'est-à-dire si la dérivée existe et si l'ordre de dérivation est moins que l'ordre maximal prédéfini).

Par exemple la dérivée croisée des trois paramètres (p_1,p_2,p_3)

(4.3) $$\frac{\partial^{m1+m2+m3} X}{\partial p_1^{m1} \partial p_2^{m2} \partial p_3^{m3}}$$

doit être calculée exclusivement à partir de $\frac{\partial^{m1+m2+m3-1} X}{\partial p_1^{m1} \partial p_2^{m2} \partial p_3^{m3-1}}$ (dériver par rapport au dernier paramètre "p_3"). On n'utilise pas la génération directe à partir de

(4.4) $\frac{\partial^{m1+m2+m3-1} X}{\partial p_1^{m1} \partial p_2^{m2-1} \partial p_3^{m3}}$ (paramètre "p_2") ou de $\frac{\partial^{m1+m2+m3-1} X}{\partial p_1^{m1-1} \partial p_2^{m2} \partial p_3^{m3}}$ (paramètre "p_1").

Autrement dit, les dérivées suivantes des dérivées (4.4) sont:

(4.5) $\frac{\partial^{m1+m2+m3-1} X}{\partial p_1^{m1} \partial p_2^{m2-1} \partial p_3^{m3+1}}$ et $\frac{\partial^{m1+m2+m3-1} X}{\partial p_1^{m1-1} \partial p_2^{m2} \partial p_3^{m3+1}}$ (prendre tout d'abord le "p_3")

si elles existent, c'est-à-dire si $\left(\frac{\partial}{\partial p_3^{m3+1}}\right)$ existe. Sinon les dérivées suivantes de (4.4) sont:

(4.6) $\frac{\partial^{m1+m2} X}{\partial p_1^{m1} \partial p_2^{m2}}$ et $\frac{\partial^{m1+m2} X}{\partial p_1^{m1-1} \partial p_2^{m2+1}}$ (ou $\frac{\partial^{m1} X}{\partial p_1^{m1}}$ si $\left(\frac{\partial}{\partial p_2^{m2+1}}\right)$ n'existe pas)

Cet ordonnancement est très efficace pour une génération récursive des dérivées. Une amélioration consiste à calculer (et vérifier) en premier lieu les dérivées premières (de la fonction), ceci permet d'identifier les paramètres existants dans un polynôme.

Pour une compréhension plus globale de la procédure, un exemple complet de la séquence de dérivation (la même que pour le stockage), sera présenté dans la (table 4.1). Notons que toutes les dérivées premières sont effectivement stockées avant les dérivées supérieures (et croisées).

4.4.2. La structure de stockage des polynômes

La structure de stockage a été choisie pour s'adapter aux traitements en C et en FORTRAN-77. Elle correspond à l'arbre de génération des dérivées.

```
Structure      : DRVV_PP        Dérivées par rapport aux paramètres
     INTEGER   NBDRV            : Nombre de dérivées
     INTEGER   IORG(NBDRV)      : Indice de dérivée mère (branche mère)
     INTEGER   IPAR(NBDRV)      : Numéros des paramètres
     DOUBLE    DRVV(NBDRV)      : Les valeurs de dérivées
```

La (table 4.1) présente la structure complète d'un DRVV-PP à trois paramètres (u,v,w) ayant les ordres de dérivation correspondants (1,3,2). Dans cette table, "u,v,w" représentent effectivement les numéros des paramètres concernés. Notons que le champ DRVV est utilisé pour stocker les valeurs de dérivées correspondantes.
Le nombre total de dérivées de cette DRVV-PP est Nb=(1+1)x(3+1)x(2+1)-1=23

Table **4.1**: Exemple de la structure d'un DRVV-PP

No	1	2	3	4	5	6	7	8	9	10	11	12	13
Drv	du	dv	dw	dw^2	dvdw	$dvdw^2$	dv^2	dv^2dw	dv^2dw^2	dv^3	dv^3dw	dv^3dw^2	dudw
IORG	0*	0	0	3	2	5	2	7	8	7	10	11	1
IPAR	u	v	w	w	w	w	v	w	w	v	w	w	w

14	15	16	17	18	19	20	21	22	23
$dudw^2$	dudv	dudvdw	$dudvdw^2$	$dudv^2$	$dudv^2dw$	$dudv^2dw^2$	$dudv^3$	$dudv^3dw$	$dudv^3dw^2$
13	1	15	16	15	18	19	18	21	22
w	v	w	w	v	w	w	v	w	w

*IORG=0 signifie que la dérivée est première (calculé directement à partir de l'expression originale)

La liste de PA-DRV est stockée au début du tableau IPAR (où IORG=0), c'est une exception nécessaire dans la génération des indices, celle ci facilite l'identification a posteriori des paramètres concernés. Dans certaines applications on utilise également un tableau supplémentaire des entiers ORD-DRV(Nombre-Paramètres) pour stocker tous les ordres de dérivation des paramètres.

Dans un DRVV-PP donné, la recherche d'une dérivée quelconque, qui est définie par la liste des paramètres PA-DRV(Nbr-Par) et la liste des ordres ORD-DRV(Nbr-Par), est réalisée par l'algorithme

présenté Fig.4.3 (formalisme compact du langage C). Le résultat est la position "Npos" de la dérivée, où "Npos==0" indique l'absence de dérivée correspondante. Remarquons que:

◊ la liste PA-DRV(Nbr-Par) peut être différente de la liste PA-DRV du polynôme.

◊ les ordres (maximaux) du polynôme sont récupérés facilement par une procédure similaire en traçant les tableaux IORG et IPAR en entier.

```
// NbrPar : Nombre de paramètres
// NbrDrv : Nombre de dérivées du DRVV-PP
Début
Imère = Npos = 0
Np = ip = 1       // Indices
Tantque ( IORG(Np)==0 et Np<=NbrDrv )
   Np = Np+1
   Si IPAR(Np)==IPAR(Np) Alors
      Si ( Imère==0 et ORD_DRV(ip)>0 ) Imère=Np
      ip = ip +1
   Finsi
FinTantque
Si (ip < NbrPar) Terminer;  // Au moins un paramètre non-présenté
Pour ip = 1,NbrPar
   Nordre = ORD_DRV(ip)
   Si ( IORG(Imère)==0 ) Nordre = Nordre -1 // 1er Paramètre
   Tantque Nordre > 0
      Tantque (IPAR(Np)>PA_DRV(ip) et Np<=NbrDrv) Np = Np +1
      Si ( Np>NbrDrv ou IPAR(Np)<PA_DRV(ip) )Terminer;
      Si (IORG(Np)==Imère et IPAR(Np)==PA_DRV(ip) ) Alors
         Nordre = Nordre -1
         Imère = k
      Finsi
   FinTantque
FinPour
Npos = Np
Fin
```

Fig.4.3. Chercher une dérivée quelconque dans un DRVV-PP

Pour une dérivée du DRVV-PP (donnée par sa position Npos dans les tableaux), les ordres sont calculés efficacement par une simple boucle (Fig.4.4). Notons que cet algorithme retourne toujours un tableau des ordres correspondant à la liste de paramètres du polynôme, et il est possible d'avoir des zéros.

```
// Npos    : La position de la dérivée dans DRVV-PP
// Nbpa    : Nombre de paramètres du polynôme
// NORD(Nbpa) : les ordres de la dérivée
// IPAR(Nbpa) sont effectivement la liste des PA-DRV
Début
Np = Npos
Nbpa = 1      // Indices
Tantque ( IORG(Nbpa)==0 et IPAR(Nbpa)<IPAR(Np) )
   Nbpa = Nbpa +1
   NORD(Nbpa) = 0
FinTantque
IORD(Nbpa) = 1              // Paramètre courant
ipa = Nbpa
Tantque (IORG(Np)>0)
   Np = IORG(Np)
   Tantque (IPAR(ipa)<IPAR(Np)) ipa = ipa -1
   NORD(ipa) = NORD(ipa)+1
FinTantque
Fin
```

Fig.4.4. Décoder une dérivée de DRVV-PP

En pratique, la synchronisation avec un tableau (de référence) des paramètres $\{\text{PA-DRV}\}^{ref}$ est fréquemment nécessaire, il faut donc modifier les algorithmes ci-dessus en améliorant la boucle pour identifier le paramètre courant (qui permet d'ignorer les paramètres "non-actifs").

4.4.3. L'arithmétique élémentaire des polynômes de Taylor

En utilisant la structure proposée du polynôme de Taylor, on peut réaliser facilement toutes les opérations arithmétiques usuelles. Prenons l'exemple d'une addition multi-modale:

$$\text{DRVV\_PP} = \sum_{j=1}^{n} \text{Cff}_j \times \text{DRVV\_PP}_j$$

où Cff_j sont les coefficients de composition

Cette opération se compose apparemment d'une simple boucle (Fig.4.5). En réalité, l'addition se réalise en trois opérations:

❶ ***Pré-traitement***: Il faut créer la liste commune des PA-DRV et la liste des ordres du polynôme résultant (le maximal de tous les ordres de DRVV-PP_j).

❷ ***Initialisation***: Tous les compteurs $(\mathbf{k},k_1,k_2,..,k_n)$ et les indicateurs de dérivées (DRV) (voir §4.4.4) sont mis au point de départ (zéros).

❸ ***Processeur***: En générant une à une les dérivées de DRVV-PP "somme" (**k**=**k**+1), une vérification simple sur les indicateurs (DRV_j) permet de récupérer la dérivée DRVV-PP_j($\mathbf{k}_j$) qui est identique à la dérivée DRVV-PP(**k**)(du polynôme "somme"), et de réaliser l'addition (composition par le coefficient Cff_j). Le compteur correspondant est avancé (k_j=k_j+1) et l'indicateur (DRV_j) le suit automatiquement.

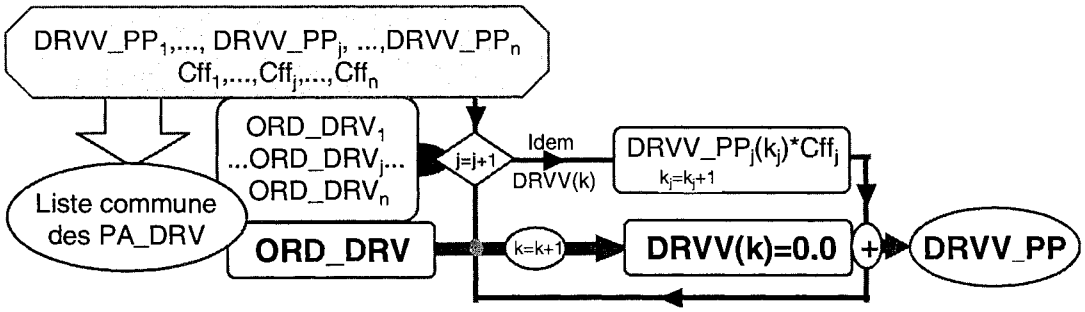


Fig.4.5. Une addition multi-modale des polynômes

Cette addition ressemble à une intégration de Gauss. Ici, la notion d'ordre de priorité de la génération de dérivées est très avantageuse. En effet:

◊ Il n'y a pas besoin de boucler sur tous les coefficients du polynôme pour chercher la dérivée désirée. Il faut comparer avec une seule dérivée, dite courante. Un compteur (un entier) est donc suffisant pour indiquer la position courante de chaque DRVV-PP

◊ En plus, l'indicateur de dérivées (voir §4.4.4) est aussi un entier, qui ne présente aucune difficulté pour les comparaisons.

La multiplication, quant à elle, se compose dans une boucle "inverse". Chaque dérivée calculée (composée) trouve sa place dans un DRVV-PP "modèle", qui a été défini auparavant à partir de la liste des PA-DRVV et des ordres composés.

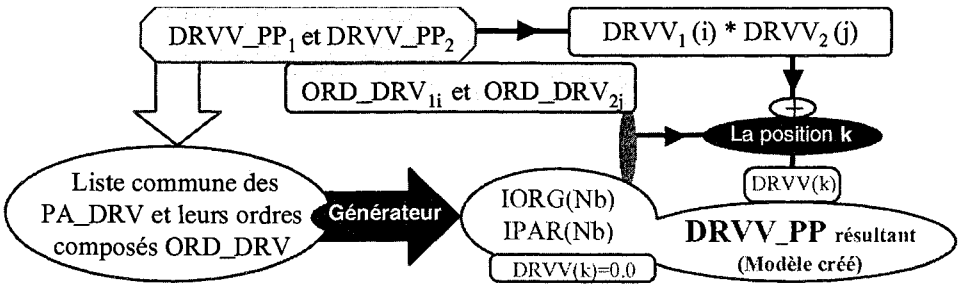


Fig.4.6. Multiplication de deux polynômes

4.4.4. Le générateur de séquence

Dans la partie ci-dessus, nous avons vu que le polynôme (DRVV-PP) est créé effectivement à partir d'une liste des PA-DRV (§3.1.1) et d'une liste des ordres ORD-DRV (correspondants). En pratique, un générateur (unique dans le programme) sert à créer successivement les indices IORG, IPAR et aussi un indicateur spécial DRV. La nature de IORG et de IPAR a été présentée au paragraphe précédent (§4.4.2). L'indicateur DRV est donc un entier binaire qui enregistre tous les PA-DRV, pour lesquels la "dérivée courante" est formé:

$$\mathrm{DRVV-PP(p1,p2,p3,p4,p5)}:\begin{cases}\dfrac{\partial}{\partial p_1^{m1}\partial p_3^{m3}} \mapsto \mathrm{DRV} = 0\mathrm{b}00101 \\ \dfrac{\partial}{\partial p_2^{m2}\partial p_3^{m3}\partial p_5^{m5}} \mapsto \mathrm{DRV} = 0\mathrm{b}10110\end{cases}$$

La fonctionnalité de cet indicateur est assurée par la priorité stricte des paramètres (voir paragraphe §4.4.1).

Les interventions sur le générateur (les changements de la séquence) peuvent être réalisées soit directement dans le tableau des ordres de référence (ORD-DRV), soit sur un tableau (de travail) des ordres de la dérivée courante. Les besoins d'interventions apparaissent dans certaines circonstances, comme:

◊ Lorsqu'une branche n'existe plus (une dérivée s'annule): Il y a le besoin d'un retour brutal,

◊ Pour limiter l'ordre de dérivation d'un paramètre par un critère particulier (calculé en même temps que la création des dérivées)

◊ Pour supprimer une branche ou la totalité des branches de quelques paramètres dans un traitement a posteriori.

4.4.5. L'inverse d'un polynôme de Taylor

Dans les chapitres antérieurs nous avons évoqué l'apparition de dérivées d'ordre très élevé par rapport aux paramètres géométriques (§2.4). Nous avons remarqué que le responsable principal est l'inversion de certaine fonction dépendant de paramètres (terme "$1/S_\Delta$" dans l'équation (2.23)). Dans cette partie, nous allons décrire l'outil qui sert à l'inversion d'un polynôme de Taylor. Certains problèmes se posent avant le démarrage de cette opération:

◊ Le calcul paramétré ne peut pas traiter toutes les dérivées jusqu'à l'infini. La limitation doit être imposée a priori.

◊ Même avec les limitations imposées, le nombre de coefficients (dérivées) dans un développement de Taylor peut encore être très important. Par exemple, avec six paramètres possédant chacun des dérivées jusqu'à l'ordre six, cela nous donne $(6+1)^6-1=117.648$ dérivées. Il faut donc essayer par tous les moyens de réduire effectivement ce nombre de dérivées.

Comme l'inversion va produire a priori des dérivées d'ordre très élevé, l'efficacité et la précision sont deux critères décisifs dans la construction de l'outil. La dérivation symbolique, s'appuyant sur la présentation complète de série Taylor, nous paraît inefficace, car la présentation est très lourde, les coefficients et les sous-séries doivent être répétés plusieurs fois dans la formule de dérivées. Nous avons choisi une approche voisine du principe de différenciation automatique. Nous calculons directement les nouveaux coefficients (les dérivées) à partir des coefficients (dérivées) du polynôme original. Un dérivateur assure donc la génération successive des traces (les chaînes) de dérivation. La figure (Fig.4.7) présente la boucle récursive pour inverser complètement un polynôme, toutes les dérivées sont calculées dans l'ordre généré par le générateur de séquence mentionné ci-dessus.

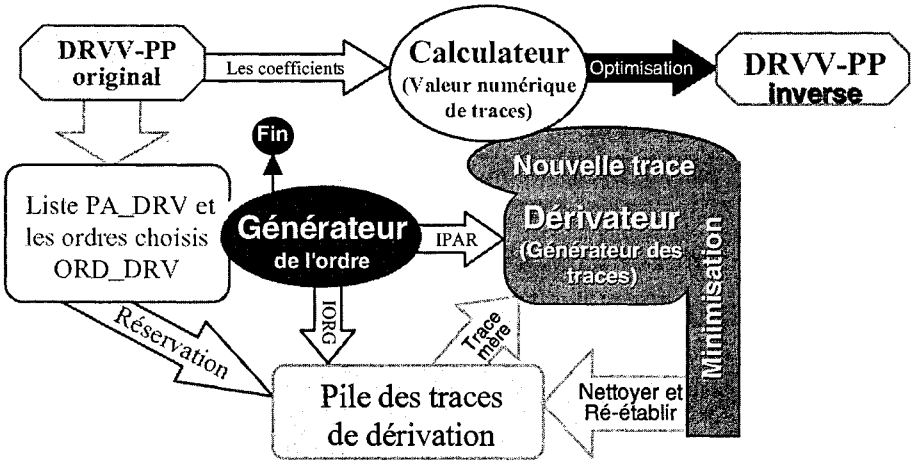


Fig.4.7. Inversion d'un polynôme de Taylor

Prenons un exemple simple d'un polynôme d'ordre 3 où PP est la valeur initiale du polynôme et 1P, 2P, 3P sont ses trois dérivées (les valeurs numériques). En limitant l'ordre du polynôme

"inverse" à cinq, la procédure de l'inversion se réalise dans une boucle récursive de 5 traces comme indiqué ci-dessous:

Table **4.2**: Exemple des traces d'un DRVV-PP "inverse"

inv	Le polynôme original d'ordre 3: DRVV-PP(PP, $\underline{1P}$, $\underline{2P}$, $\underline{3P}$)
1D	$-(\underline{1P})/PP^2$
2D	$2.(\underline{1P})^2/PP^3 - (\underline{2P})/PP^2$
3D	$-6.(\underline{1P})^3/PP^4 + 6.(\underline{1P}.\underline{2P})/PP^3 - (\underline{3P})/PP^2$
4D	$24.(\underline{1P})^4/PP^5 - 36.(\underline{1P}^2.\underline{2P})/PP^4 + 6.(\underline{2P})^2/PP^3 + 8.(\underline{1P}.\underline{3P})/PP^3$ (Composant $\underline{4P}$ n'existe pas)
5D	$-120.(\underline{1P})^5/PP^6 + 240.(\underline{1P}^3.\underline{2P})/PP^5 - 90.(\underline{1P}.\underline{2P}^2)/PP^4 - 60.(\underline{1P}^2.\underline{3P})/PP^4 + 20.(\underline{2P}.\underline{3P})/PP^3$ (Pas de dérivées $\underline{4P}$ et $\underline{5P}$)

Dans le cas d'un seul paramètre, on n'a pas vraiment besoin de stockage dans la pile. Par contre, la pile devient absolument nécessaire pour créer efficacement toutes les dérivées de "DRVV-PP inverse" s'il y a plusieurs paramètres. Notons que le nombre maximal des "traces" stockées dans la pile, est égal au nombre de paramètres.

En ce qui concerne la limitation des ordres, il est possible de la définir globalement pour tous les paramètres (PA-DRV), elle est ensuite affectée à toutes les opérations (ce type de limitation est très pratique pour l'assemblage). Une autre possibilité consiste à la définir localement à partir des ordres de polynômes origines. Par exemple, nous avons choisi d'ajouter un nombre constant sur les ordres du polynôme à inverser ($ORD\_DRV_{inverse} = ORD\_DRV_{origine} + 5$).

Enfin, une optimisation sur le DRVV-PP résultant est absolument nécessaire afin de réduire effectivement sa dimension. Cette minimisation se réalise pratiquement en s'appuyant sur un critère des erreurs (par exemple, l'erreur relative $<10^{-4}$), un contrôle dès la création est donc très souhaitable.

4.5. Mise en oeuvre d'un interpréteur de la procédure

Nous avons choisi une organisation ouverte pour la résolution "paramétrée", basée sur une procédure d'interprétation d'opérations élémentaires. Notons qu'il existe déjà dans FLUX3D un langage d'interprétation de l'assemblage EF, langage HE (Hyper-matrice Élémentaire) [Coul95i]. Ce langage possède un ensemble très puissant de commandes généralisées.

Dans notre cas, le calcul de dérivées a des besoins très spécifiques (différents du calcul EF classique):

◊ les données dans ce problème sont peu similaires aux données EF "normaux" et elles exigent plusieurs traitements spéciaux;

◊ les calculs de dérivées (sur les éléments) sont très particuliers et complexes;

◊ l'assemblage du vecteur second membre est récursif (et très répétitif) et ne touche pas à la matrice globale.

L'efficacité de la procédure demande une organisation toute particulière. Une étude de faisabilité nous a conduit à créer un nouveau module d'interprétation, dans lequel l'ensemble de commandes est choisi à un niveau beaucoup plus élémentaire (surtout au niveau de l'intégration) que celui du langage HE. Notons que, l'interpréteur et la majorité de ses "opérations primitives" sont écrits effectivement en C (ANSI-C). La gestion de mémoire dynamique et les formats d'écriture sur les fichiers sont donc gérés en interne au module (indépendamment des outils FORTRAN du corps de FLUX3D).

4.5.1. L'organisation générale du module

Nous parlons d'un problème linéaire, avec les hypothèses suivantes:

- Le système d'équations $[\mathbf{M}].\{\mathbf{A}\}=\{\mathbf{S}\}$ est déjà établi et résolu pour une solution initiale (la matrice $[\mathbf{M}]$est déjà factorisée).
- Le problème de dérivation est décrit par un paquet de paramètres, dont le nombre n'est pas limité, si ce n'est par les ressources informatiques (temps de calcul, espace mémoire).

- Les influences des paramètres sur toutes les entités géométriques sont connues. Elles portent sur les éléments finis à calculer (pratiquement ceux des faces et des volumes des régions actives).
- Les polynômes de Taylor des positions de tous les noeuds sont connus.

Notre objectif consiste à établir le polynôme de Taylor de la solution {**A**} (des variables d'état). Comme nous l'avons vu au chapitre 2, ce travail se décompose en un assemblage récursif du vecteur second membre d'après les dérivées de la matrice **M** et du vecteur **S**, puis en une résolution successive de système d'équations avec la matrice "initiale" et le second membre recalculé. Dans notre réalisation, la résolution reste une opération "standard". L'implantation est consacrée à la partie "assemblage", qui est constituée de deux grandes opérations:

❶ La détermination des dérivées de **M** et de **S**

❷ L'assemblage récursif.

En pratique, il est préférable de calculer en une seule fois toutes les dérivées des intégrales élémentaires et ensuite, de réaliser la boucle d'assemblage séparément (Fig.4.8). Cela permet d'augmenter l'efficacité du calcul des dérivées (de **M** et de **S**) et aussi d'alléger beaucoup la procédure récursive d'assemblage.

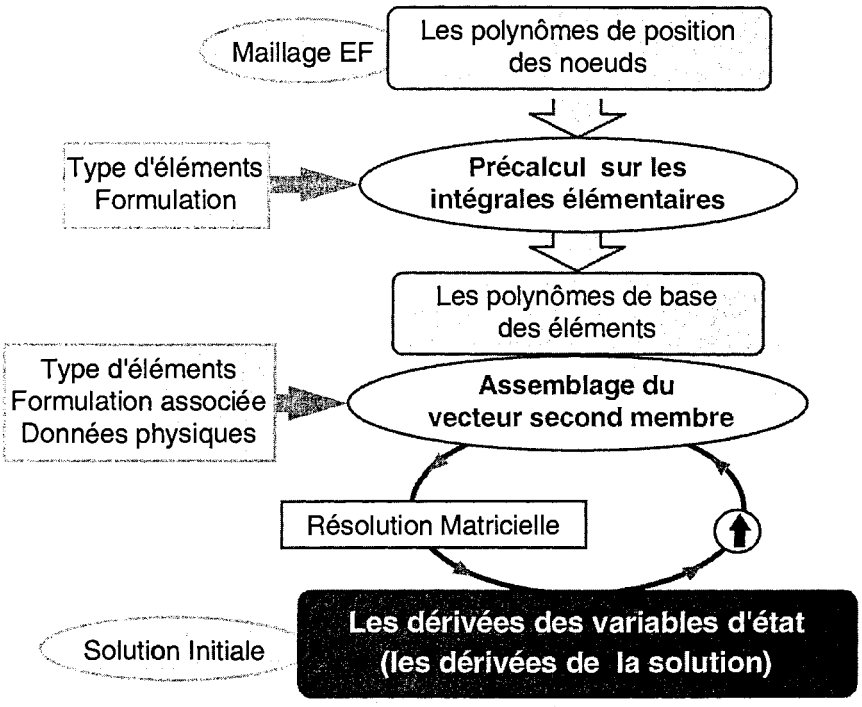


Fig.4.8. L'organisation de la dérivation du solveur EF

Dans cette procédure, le besoin en stockage secondaire est évident, car le volume de "données" est très important. Pratiquement, les polynômes des noeuds, puis les polynômes des éléments et les dérivées des variables d'état, sont tous enregistrés sur disques. Le mode de stockage binaire est très utile, en raison:

◊ de la rapidité des opérations binaires;

◊ de l'existence de plusieurs boucles dans l'assemblage récursif intercalées avec la résolution et qui enchaînent lecture et écriture sur fichier.

Deux groupes d'opérations (commandes) élémentaires sont créées pour interpréter deux phases de traitement ("Précalcul" et "Assemblage") présentées dans l'organigramme de la (Fig.4.8). Ces programmes "modèles de dérivation" et "modèles d'assemblage" dépendent de la formulation et sont construits à partir des commandes disponibles dans notre langage.

Les modèles sont gérés en dehors du code compilé. Tous les modèles sont rassemblés dans une "bibliothèque de modèles" consultable par l'interpréteur intégré dans le logiciel éléments finis. En général, de tels programmes sont écrits sous forme de lignes de commandes (de syntaxe conventionnelle) et stockés sous forme de fichier "texte" (sur disques). L'interprétation est réalisée ensuite, soit directement par une analyse textuelle, soit par l'utilisation des codes pré-compilés. Compte tenu des particularités de notre simulation, nous avons organisé notre bibliothèque autour de modèles "binaires". Elle est ainsi plus compacte et aussi plus efficace. Un module assistant est disponible pour faciliter la création des nouveaux modèles et pour modifier les modèles existants (voir §4.5.4).

Enfin, nous remarquons que l'utilisation du stockage secondaire des résultats (séparée de la base de données principale) crée des problèmes de synchronisation des données. En effet, il faut enregistrer aussi toutes les informations qui assurent la cohérence des données, par exemple les informations concernant le type et la position des variables dans un enregistrement des dérivées de la solution EF.

4.5.2. La mise en oeuvre du précalcul sur les EFs

Il est évident que cette boucle de calculs doit être réalisée exclusivement sur les éléments finis qui sont "actifs" (pour le problème) et qui sont déformés par les changements des paramètres PA-DRV (voir §2.5). Le module se décompose donc en un traitement préliminaire de polynômes

des noeuds (d'un élément) et le lancement de l'algorithme de dérivation qui est dans ce cas, une interprétation de modèles enregistrés (Fig.4.9).

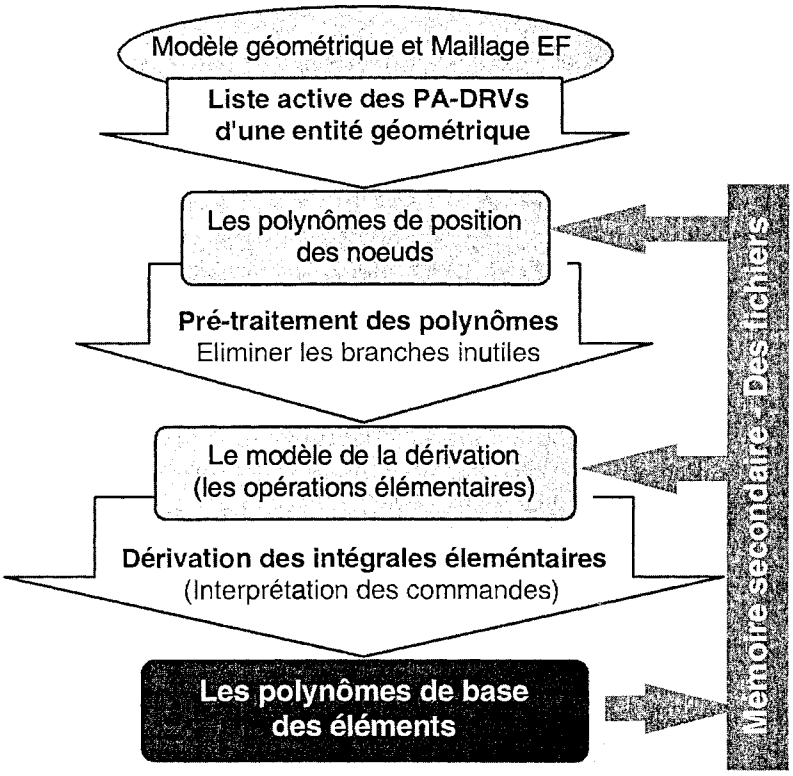


Fig.4.9. La dérivation des intégrales élémentaires

Naturellement, il vaut mieux organiser la dérivation en groupant les éléments d'après l'entité géométrique qui les porte. En effet, la dérivation semi-analytique du maillage (chapitre 3) nous permet d'identifier clairement le comportement d'une entité géométrique par rapport aux PA-DRV. Ici l'information précieuse est donc la liste des PA-DRV réellement "à dériver". Les polynômes des noeuds, quant à eux, contiennent tous les PA-DRV concernés, ce qui est nécessaire pour réaliser une perturbation de maillage par exemple et qui est indispensable pour les noeuds frontaliers. En présence des PA-DRV, dit "non actifs", il faut optimiser le calcul en éliminant toutes les branches inutiles des polynômes.

Nous avons parlé antérieurement de la possibilité d'utilisation du calcul symbolique ou de la technique de DA dans la dérivation des intégrales élémentaires. Notre travail utilise principalement une procédure symbolique. On peut imaginer cependant une coexistence intéressante entre la DS et la DA: Dans ce cadre, l'interpréteur (par le biais des outils

symboliques) assure l'interface (en entrée et sortie) et permet d'activer les codes DA (généré par un précompilateur) pour calculer les dérivées élémentaires.

Retournons à la réalisation interne d'un langage de commande (l'interpréteur de la procédure), il y a trois parties à implanter (et organiser) lors de sa conception:

◊ La partie automatique: C'est une partie fixe de l'algorithme (constantes, opérations communes), mais qui est cachée de l'utilisateur.

◊ Les données publiques: Ce sont les emplacements en mémoire réservée (éventuellement initialisée). Chacun possède un "nom" et des propriétés destinés à l'interpréteur de commandes. En général, on distingue les données non-modifiables et les données modifiables.

◊ Les opérations, c'est-à-dire les commandes et leur syntaxe: C'est la partie active de l'algorithme. L'ensemble de ces commandes détermine la puissance et la souplesse de notre interpréteur.

Une liste complète de toutes les commandes avec leurs syntaxes principales est présentée dans l'Annexe **A**. Un exemple simple est visible plus loin à la figure Fig.4.12 dans le paragraphe §4.6.1. Ici, nous décrirons uniquement les concepts principaux essentiels à la création d'un "modèle de dérivation".

Dans la dérivation, dite élémentaire, les données principales sont naturellement les polynômes de position des noeuds et la structure de stockage de tous les polynômes créés au long du calcul. Les polynômes des noeuds se présentent sous la forme {xN,yN[,zN]}, où N est le numéro du noeud (avec la numérotation interne propre à chaque élément). Les polynômes opérationnels sont identifiés par leur numéro (de position) dans un stockage de pile. À la fin du traitement, les polynômes résultants doivent être accessibles et identifiables, afin que l'utilisateur puisse les exploiter ensuite dans le module d'assemblage. Notons que le remplissage mémoire (de tous les tableaux) est "compact" sans trou, ce qui est totalement compatible pour une lecture/écriture massive sur disques.

Les opérations de lecture et d'écriture de données ci-dessus sont automatiques.

En ce qui concerne la structure d'un "modèle de dérivation", la première ligne (significative) doit être une déclaration par commande «PBL». Dans cette déclaration, il faut donner la formulation associée, le type d'élément relié et aussi quelques options de contrôle (par

exemple: nombre de polynômes résultants). Les autres commandes de type "contrôle" sont «INI», «IF», «GSS» (pour Gauss procédure) et «FIN» (ou «ASM» pour lancer l'assemblage).

Les traitements de "données" constituent un deuxième groupe de commandes. Naturellement, les commandes principales sont les opérations "polynomiales" décrites dans (§4.4): «SUM», «MLT» (Multiplication) et «INV». Il y a aussi les commandes reliées aux constantes «CFF» (assignation des coefficients), «CST» (multiplicateur constant). Notons que les opérations arithmétiques créent toujours un nouveau polynôme, qui s'écrit au point "courant" de la pile. En ce qui concerne la pile (mémoire dynamique), la gestion s'organise par un pointeur "début" et un pointeur d'écriture (le "courant"). Le positionnement (des polynômes) est donné en absolu (par rapport au "début") ou relatif à partir du pointeur "courant" (par un nombre négatif). De plus, une commande «MEM» sert à compacter la pile en supprimant un certain nombre d'éléments (des polynômes).

Remarquons que le calcul complet de la matrice et du vecteur élémentaire peut produire un stockage inefficace de données, qui sont parfois les grands polynômes identiques (ou de composants identiques). Par exemple, le déterminant $|\mathbf{G}|$ (eq.2.25) ou l'aire "S" (eq.2.22) sont présents dans tous les membres du vecteur élémentaires et le "1/S" (eq.2.23) est entré dans tous les membres de la matrice élémentaire. Pour ces cas, un stockage des "pièces de base" est donc très efficace (et économique), éventuellement un traitement supplémentaire (qui doit être assez simple) est demandé ensuite dans l'assemblage.

4.5.3. La mise en oeuvre de l'assemblage du second membre

Cette phase doit assurer l'assemblage récursif des vecteurs seconds membres afin de calculer toutes les dérivées (désirées) de la solution. Reprenons par exemple, le système d'équation (2.6) qui est écrit pour calculer une dérivée quelconque par rapport à deux paramètres **p** et **q**:

$$\mathbf{M}\cdot\frac{\partial^{m+n}\mathbf{A}}{\partial p^m \partial q^n}=\frac{\partial^{m+n}\mathbf{S}}{\partial p^m \partial q^n}-\sum_{i=0}^{m}\sum_{j=0}^{(i\#m)\,n-1} C_m^i C_n^j \frac{\partial^{m-i+n-j}\mathbf{M}}{\partial p^{m-i}\partial q^{n-j}}\cdot\frac{\partial^{i+j}\mathbf{A}}{\partial p^i \partial q^j}. \tag{4.7}$$

Le vecteur second membre est calculé récursivement en s'appuyant sur les dérivées de la matrice et du vecteur second membre initial. Ces dernières sont calculées à partir des intégrales élémentaires dans le "précalcul" décrit précédemment. Une structure générale du module est présentée sur la figure (Fig.4.10).

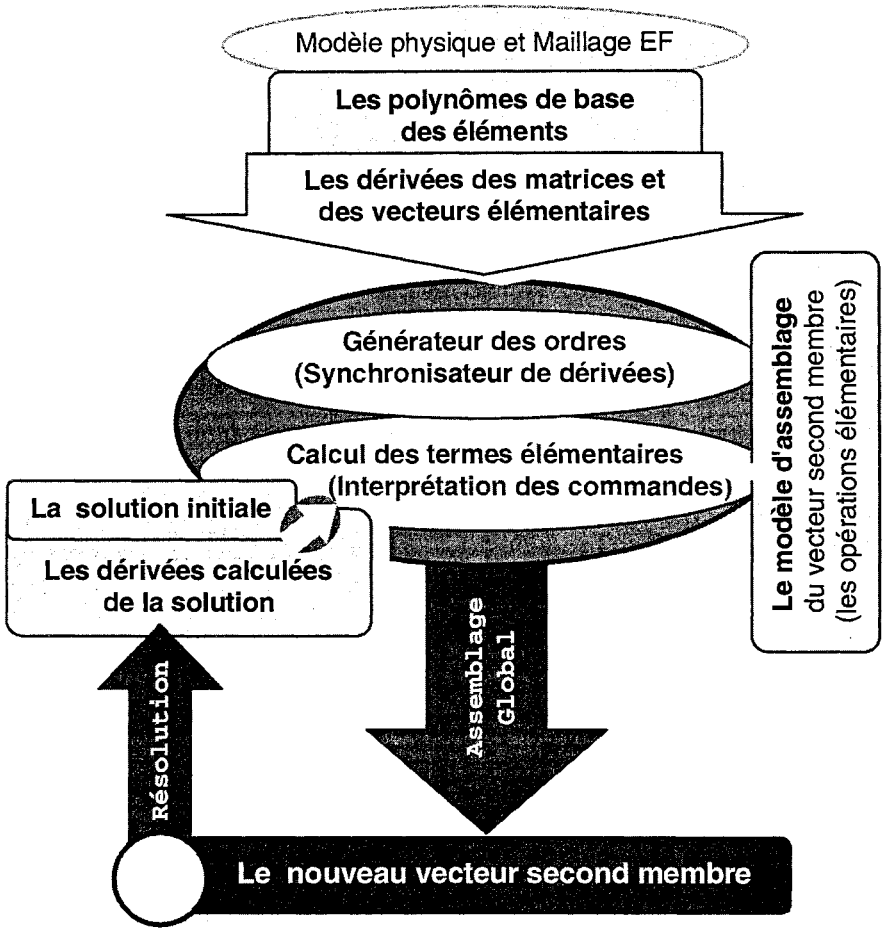


Fig.4.10. L'assemblage récursif du vecteur second membre

```
NbrE   = Nombre total des éléments concernés (déformés)
Pol-E  = Les polynômes de Taylor élémentaires
NbrD   = Nombre total des dérivées calculées de la solution
NbrR   = Nombre des dérivées concernées (prédécesseurs de la "courante")
```

```
Pour n:=1,NbrE
   Récupérer Pol-E
   Pour k:=1,NbrD
      Si prédécesseurs alors
         Positionner dans Pol-E
         Calcul d'assemblage
      Finsi
   FinPour
FinPour

Lecture des fichiers:
1 fois de NbrE Pol-E
NbrE fois de NbrD solutions
```

```
Pour k:=1,NbrD
   Si Non prédécesseurs Répéter
   Pour n:=1,NbrE
      Récupérer Pol-E
      Positionner dans Pol-E
      Calcul d'assemblage
   FinPour
FinPour


Lecture des fichiers:
NbrR fois de NbrE Pol-E
1 fois de NbrD solutions
```

Fig.4.11. Les deux algorithmes d'assemblage récursif du vecteur second membre

En pratique, il y a deux réalisations possibles pour cette phase d'assemblage: le bouclage sur les éléments (déformés) et le bouclage sur les dérivées successives (4.7). Les algorithmes correspondants sont présentés sur la figure (Fig.4.11).

A première vue, ces deux algorithmes sont comparables. Celui avec boucle sur les éléments paraît même plus naturel. Cependant, une étude plus détaillée montre quelques avantages pour le second:

◊ Le calcul est plus stable en matière de données internes (les coefficients, les tableaux des PA-DRV, etc.) car la dérivée de référence reste inchangée dans toutes les boucles internes.

◊ Dans le cas de plusieurs paramètres, le nombre de prédécesseurs de la dérivée "courante" **NbrR** devient au fur et à mesure de création beaucoup plus petit que le nombre total **NbrD**.

◊ Le fichier de "Pol-E" est en lecture (seulement), par contre, le fichier des solutions (de ses dérivées) est en "mise à jour", par conséquent son traitement doit être plus soigné.

Dans cette phase, une grande différence par rapport à la phase précédente réside dans le type des données, dites principales. Ici, nous revenons au calcul "numérique" et non plus au calcul "polynomial". En effet, on traite à la fois une dérivée de la solution et les dérivées correspondantes des intégrales élémentaires. Les polynômes (des éléments) sont immédiatement filtrés pour devenir simplement un tableau (de type réel) des dérivées (zéro si la dérivée correspondante n'existe pas dans un polynôme). Notons que cette opération de pré-traitement doit compléter aussi tous les traitements réservés des polynômes (voir les remarques dans le paragraphe précédent).

En principe, à chaque itération le vecteur de la solution (sa dérivée) $\{\mathbf{V}\}$ est restauré à partir du vecteur global. Il faut ensuite initialiser le vecteur second membre $\{\mathbf{R}\}$ et calculer la dérivée de la matrice élémentaire $[\mathbf{M}]$. L'assemblage élémentaire se réalise enfin par:

$$\{\mathbf{R}\} = \{\mathbf{R}\} + [\mathbf{M}].\{\mathbf{V}\}$$

En ce qui concernent le "modèle d'assemblage", une déclaration par commande «PBL» est nécessaire pour identifier la formulation associée (et le type d'élément concerné). Le seul branchement «IF» est là pour isoler la partie d'assemblage de la dérivée du vecteur second membre qui ne doit boucler qu'une fois.

L'arithmétique vectorielle et matricielle élémentaire se décompose principalement suivant les opérations «ADD» et «MLT». La préparation consiste essentiellement en des opérations d'assignation:

◊ les «CFF» pour créer les coefficients

◊ les «SET» pour remplir par une constante ou assigner deux entités de dimension équivalente.

◊ les «DRV» assurent les affectations à partir du tableau de dérivées de base (calculées précédemment)

◊ les «CFS» pour affecter les coefficients à un tableau (vecteur ou matrice)

La procédure assemblage-résolution nous donne à chaque fois une dérivée de la solution. Dans notre réalisation, les dérivées sont créées en ordre généré par le générateur décrit au paragraphe (§4.4.4). Les dérivées calculées sont enregistrées alors dans un fichier (binaire) et accompagnées par l'indicateur de dérivée (DRV) (voir §4.4.4). Remarquons que cet ordre de dérivation respecte parfaitement l'exigence de l'assemblage récursif, c'est-à-dire que toutes les dérivées-prédécesseurs sont calculées avant le calcul de la dérivée souhaitée.

En ce qui concernent la DA, son utilisation est envisageable dans le contexte suivant:

◊ Il faudrait d'abord localiser et précompiler les codes reliés à une formulation et/ou un type d'élément

◊ Ensuite, une "commande" ferait activer les codes précompilés au moment propice pour créer toutes les dérivées désirées (des intégrales). L'interpréteur aurait la charge de fournir toutes les données nécessaires et de gérer les résultats.

◊ L'assemblage serait réalisé (normalement) par une boucle des "commandes".

4.5.4. Les assistants de la création des "modèles"

Avant de donner un exemple complet de "modèle" (programme) de dérivation et d'assemblage, nous allons présenter les outils qui servent à assister l'utilisateur dans la création de ces "modèles". Ils sont au nombre de deux: le compilateur (codeur) et le décodeur.

⇒ le **compilateur** sert à transformer le modèle "textuel" (écrit en format "texte") en un modèle binaire et à enregistrer ce dernier dans la bibliothèque.

⇒ le **décodeur** fait l'inverse, il décompose les modèles binaires en lignes de commandes (avec les commentaires minimaux).

Naturellement, la création primitive des modèles (et leur modification) est réalisée sous forme textuelle. Un éditeur de texte quelconque suffit pour effectuer cette tâche. D'un autre côté, les modèles binaires sont plus compacts, plus convenables pour la machine, de plus, leur interprétation interne est beaucoup plus simple et rapide. L'utilisation d'un compilateur nous permet de combiner les avantages de ces deux types de modèles. Dans le cas d'un interpréteur, nous remarquons encore quelques particularités avantageuses:

◊ Le compilateur peut donner un rapport plus précis des fautes de syntaxe ce que n'aura plus à faire l'interpréteur en cours d'exécution.

◊ Lorsqu'une vérification de syntaxe est déjà effectuée, on peut simplifier le contrôle à l'exécution (contrôler au point critique seulement).

◊ Le compilateur vérifie et assure la cohérence des interfaces de modules (entrée et sortie). Également, il peut donner les informations précieuses de dimensions, détecter les commandes spéciales pour préparer les procédures spécifiques (par exemple l'intégration de Gauss ou activer un outil de type de DA).

◊ À la suite de compilations, les codes binaires permettent de gérer efficacement tous les modèles. Ils sont installés dans une bibliothèque centralisée.

Notons que le décodeur sert principalement à récupérer (pour modifier) les modèles binaires existants en forme "texte", mais en pratique c'est aussi un outil très efficace pour vérifier la compilation d'un nouveau modèle.

4.6. La validation des modèles

Dans cette partie nous voulons présenter la validation des modèles. Nous étudierons aussi les particularités de calculs et quelques problèmes concernants l'exploitabilité des résultats.

4.6.1. Un exemple simple en 2D

Nous avons décrit dans le chapitre 2 la dérivation des intégrales élémentaires du problème magnétodynamique pour un maillage triangulaire (voir §2.3.3). C'est le "modèle" le plus simple. Il est totalement analytique. Il nous servira à montrer le principe d'interprétation et la structure typique d'un fichier de commandes.

```
// Triangle formulation (MS2VEC,MD2VEC)
// **********************************
PBL TRIG ALL 8
//  ============================================================
//  Input  : {x1,y1},{x2,y2},{x3,y3}
//  Output : S2, S2inv/2, [gradt]*[grad] 8 DRVV-PO
//            S2, Deux fois d'Aire du triangle S2inv=(1/S2)
//          [gradt]*[grad], Matrice symétrique, Stockage triangle
//                          M11,M12,M13,M22,M23,M33
//  ============================================================
INI 2 3                    //
CFF s2 1 -1                // Déclarer pour SUM : a-b

SUM y2 y3                  // y2-y3 -> Output 1
SUM y3 y1                  // y3-y1 -> Output 2
SUM y1 y2                  // y1-y2 -> Output 3
SUM x3 x2                  // x3-x2 -> Output 4
SUM x1 x3                  // x1-x3 -> Output 5
SUM x2 x1                  // x2-x1 -> Output 6

// Remettre au tableau "Input" et re-définir
// la dimension des vecteurs
MEM 0 6          // {y2-y3,y3-y1,y1-y2,x3-x2,x1-x3,x2-x1}
INI 3 2 u        // {Dx,Dy} # {u1,v1,w1},{u2,v2,w2}

// S2 = (y3-y1)(x2-x1) - (y1-y2)(x1-x3) l'aire du triangle
MLT v1 w2        // (y3-y1)(x2-x1) -> Output 1
MLT w1 v2        // (y1-y2)(x1-x3) -> Output 2
SUM -2 -1        // Out(1)-Out(2)  -> Output 3
MEM -3 -1        // Out(3) -> Output 1 (reposition à 1)
INV  1           // Inv. of Out(1) -> Output 2 (S2inv)
```

```
CFF  3 1 2        // Coeff(3) = 1/2
CST  2 3          // Out(2)*coeff(3) (=>S2inv2)

CFF s2 1 1        // pour SUM : a+b
//[M]=[Dx.Dxt + Dy.Dyt]*S2inv stockage compact diagonal
MLT u1 u1         // Dx1*Dx1 -> Output 3
MLT u2 u2         // Dy1*Dy1 -> Output 4
SUM -2 -1         // Dx1*Dx1+Dy1*Dy1 -> Output 5
MLT -1  1         // (D.Dt)*S2inv2 -> Output 6 (élément M11)
MEM -4 -1         // Out(6) -> Output 3 (reposition à 3)
MLT u1 v1         // Dx1*Dx2 -> Output
MLT u2 v2         // Dy1*Dy2 -> Output
SUM -2 -1         // SUM -> Output
MLT -1  1         // (D.Dt)*S2inv2 -> Output (élément M12)
MEM -4 -1         // Out(-1) -> Out(-4) (position 4)

!!... Similaires pour les position de 5 à 8 !!

//================================================================
// Fin de pre-calcul, passer à l'assemblage
// Entrée: + Le tableau DRVV[8] calculé
//         + Le vecteur: valeurs de la solution (dérivées) V[3]
//         + double: réluctivité mr
//         + double: densité du courant scr
//         + double: conductivité complexe osgm
// Les trois derniers sont constants(inchangés dans une région)
//================================================================
ASM 3 8 2         // Output M[3x3],R[3] Input DRVV[8] (problème 2D)
PBL MS2VEC        // Formulation MS2VEC

IF@== 0           // Calcul du vecteur source (une dérivée)
DRV R3 1 1 1      // S2 (voir équation 2.17)
CFF ccf 1 6       // 1/6 -> coeff.
MLT ccf scr       // J/6  -> coeff
MLT R3  ccf       // J/6 * S2 -> R (Initialiser le vecteur R)
IF#====
// La partie réelle, voir l'équation (2.20)
DRV Mr 3 4 5 |=   // Affecter matrice M[3x3] à partir de DRVV[8]
       4 6 7 |=   // lignes continues
       5 7 8 |#
MLT M3 mrl        // NUconst * M[3x3] -> M
MLT V3 M3         // M.V -> V
MLT V3 tcff       // Coefficient de combinaison
ADD R3 V3         // t.V + R -> R

//============================================================
PBL MD2VEC 2      // Problème Dynamique [2:=Complexe]
IF@== 0
DRV R3 1 1 1      // Idem de statique
SET Rj 0          // Zéro pour R[3] Imaginaire
SET ucf scr       // scr nonmodifiable (scr est le nombre complexe)
CFF ccf 1 6       // coefficient réel
MLT ucf ccf       //
MLT R3 scr        // R*J/6
IF#====
```

```
DRV Mr 3 4 5 |=            // Matrice M[3x3] Réelle partie [P]
       4 6 7 |=            // lignes continues
       5 7 8 |#
MLT Mr mr                  // NUconst * M[3x3] -> M
DRV Mj 1 1 1 1 1 1 1 1 1   // Matrice M[3x3] Partie Imaginaire [Q]
CFF ccf 1 12
MLT ccf osgm               // (1/12)* ωσ
MLT Mj ccf                 // (ωσ/12) * Mj[3x3] -> Mj
CFS Mj 2 2 2               // Diagonale Multiplication
MLT V3 M3                  // M.V -> V
MLT V3 tcff
ADD R3 V3                  // C.V + R -> R

FIN
```

Fig.4.12. Exemple du fichier de commandes

4.6.2. Réflexion sur les "modèles" plus complexes

Dans le paragraphe précédent, nous avons montré la structure d'un modèle de simulation. Le code obtenu est très "élémentaire" et assez simple car c'est un (rare) modèle direct analytique. Dans le cas général, l'interprétation exclusivement par les opérations de base conduit à un code très long. A la place, il vaut mieux créer (et utiliser) les sous-modèles d'utilisateurs et quelques commandes, dites généralisées.

Par exemple, l'intégrale du terme "1/x" ("1/r" dans une formulation axisymétrique) est calculée par une boucle standardisée de Gauss (en utilisant la méthode d'intégration de Gauss):

```
#GS 7 3       // Initialiser le modèle de Gauss (une fois en entrée)
              // 7 point et 3 fonctions de forme (un triangle 1er ordre)
GSS  all      // Boucler tous les points de Gauss en utilisant les
              // fonctions de forme comme les coefficients
              // dans l'opération SUM (3 membres)
   SUM x1 x2 x3   // (α^1i x1+α^2i x2+α^3i x3) : X^i
   INV -1         // (1/X^i)
   MEM -2 -1      // Supprimer X^i
GS#           // Fermer le premier boucle Gauss actif
GSS  poids    // Utiliser les poids pour les coefficients SUM(7 membres)
   SUM -1 -2 -3 -4 _-5 -6 -7
   MEM -8 -1  // Supprimer les (1/X^i)
GS#
```

Fig.4.13. L'exemple d'une boucle de Gauss

Remarquons que dans cet exemple, le positionnement relatif (par un numéro négatif) permet de transporter le code librement dans la procédure (une portabilité toute relative).

Étudions maintenant une procédure générale de calculs des intégrales élémentaires, l'intégration de Gauss se décompose pratiquement en trois étapes:

❶ La construction des fonctions de forme et de leur gradient aux points de Gauss

❷ La définition de la valeur des expressions sous l'intégrale aux points de Gauss

❸ La composition des intégrales en calculant les sommes de la forme (eq.4.1).

En utilisant les éléments de référence, le calcul classique interprète l'étape ❶ comme une simple récupération du modèle existant, l'étape ❸ est elle aussi une opération standard, la difficulté principale est donc dans le calcul des expressions de l'étape ❷. Dans une procédure de dérivation, la solution la plus naturelle consiste à dériver directement tous les termes de ces trois étapes, c'est-à-dire répéter tous les calculs "classiques" avec les données et les outils de différenciation (DA ou DS). En pratique, il vaut mieux optimiser le calcul pour chaque formulation et chaque type d'élément car la procédure de dérivation est basée sur les opérations beaucoup plus coûteuses que celles du calcul ordinaire. Cette optimisation est quasiment obligatoire dans le cas des calculs symboliques. En pratique, elle est nécessaire également pour une utilisation, dite locale, de la technique de DA.

Dans notre première implantation (très primitive), la création et la vérification d'un modèle sont encore assez compliquées. En effet, c'est une procédure difficile et très intuitive (une vraie programmation), qui demande tout d'abord une analyse détaillée de l'algorithme (des codes). La réalisation, ensuite, consiste à découper la procédure (déjà optimisée) en des commandes et peut-être à créer les sous-programmes utilisateur nécessaires (par exemple les codes de DA).

4.6.3. La dérivation par rapport aux paramètres physiques

La limitation du temps disponible ne nous a pas permis d'implanter toutes les opérations nécessaires pour assurer le calcul des dérivées par rapport aux paramètres physiques. Dans l'interprétation des commandes (ci-dessus), les données physiques sont récupérées, pour l'instant, comme des constantes. L'identification de ces données est réalisée à l'aide des variables "par défaut": la propriété du matériaux "mtr" et le terme de source "src". L'initialisation est effectuée donc par les outils internes (automatiquement et fixe).

Avec l'organisation actuelle de l'interpréteur, la possibilité d'ajouter le traitement des paramètres physiques existe. Par exemple, pour un problème linéaire l'implantation consisterait à réaliser les modules suivants:

◊ La déclaration des paramètres physiques: Tout d'abord il faudrait créer une liste de paramètres admissibles pour chaque formulation, et ensuite une structure pour identifier les paramètres actifs (similaire à PA-DRV).

◊ L'intégration paramétrée: L'intégration (et assemblage récursif) peut être organisé par l'interpréteur ci-dessus en ajoutant les identifications (les formats des variables internes) de tous les paramètres physiques et aussi en créant les nouveaux outils pour les traiter. S'il y a des dérivées croisées entre les paramètres physiques et les paramètres géométriques, il serait possible alors d'ajouter les nouveaux paramètres et les dérivées reliées dans les polynômes de base (§4.5.2).

◊ L'assemblage et résolution successive: il faudrait ajouter une boucle, en plus de celles existantes, pour assurer le calcul des dérivées par rapport aux paramètres physiques, et pour le calcul des dérivées croisées. Dans le concept du "langage", il faudrait renforcer les commandes de contrôle et ajouter les boucles internes.

Notons que dans le calcul symbolique, une représentation composée (de deux polynômes en combinaison ou en "padé") est une solution intéressante, qui permet d'améliorer le calcul et de réduire le volume de stockage des données (§4.5.2). Les outils existants permettent d'utiliser directement ce type de représentation, la création de nouvelle arithmétique est également possible. Enfin, le "padé" est une autre solution efficace pour le problème non-linéaire [CEDRAT].

Conclusion et perspectives de développement

Dans ce chapitre nous avons présenté les différentes techniques numériques pour obtenir les dérivées de la solution EF et aussi les organisations possibles du solveur paramétré. En étudiant les aspects pratiques des approches, nous avons choisi d'implanter un interpréteur pour simuler la phase de dérivation des intégrales élémentaires et la phase d'assemblage récursif du vecteur second membre. Nous avons réalisé les outils et les structures primitives d'un langage d'interprétation. Ce langage est orienté vers une nouvelle structure du programme EF, dans laquelle l'ordre des opérations et les branchements sont enregistrés en dehors des codes compilés.

En s'appuyant sur ce langage, nous avons créé quelques modèles de résolutions paramétrées. Dans le cadre d'un travail de thèse, nous n'avons pas pu compléter tous les aspects pratiques de l'approche proposée. Il faut dire que l'insuffisance du nombre de modèles créés et surtout l'absence des outils traitants les paramètres physiques, limite beaucoup l'utilisation de nos résultats. Dans le chapitre suivant nous présenterons quelques exemples validés de calculs paramétrés en étudiant les applications possibles.

Chapitre 5

L'EXPLOITATION DES RÉSULTATS PARAMÉTRÉS

Sommaire

Chapitre 5

L'EXPLOITATION DES RÉSULTATS PARAMÉTRÉS

Dans les chapitres précédants nous avons développé les principaux modules d'une analyse de sensibilité d'ordre élevé: le dérivateur de maillage et le solveur paramétré. Le premier nous a donné les polynômes de Taylor représentant la position des noeuds et le second les polynômes pour évaluer toutes les variables d'état dépendant des paramètres géométriques.

Fort d'une méthode établie, ce chapitre est consacré à la validation des calculs paramétrés et à l'exploitation des résultats d'analyses. Nous étudierons, tout d'abord, quelques problèmes généraux concernant la performance et la validité du calcul, nous y soulignerons les points délicats dans une extraction de l'information, dite "paramétrée". Ensuite nous présenterons quelques exemples de calcul en faisant le point sur la réalisation pratique. À la fin, nous proposerons une manière complète d'exploiter les résultats de calculs EF paramétrés en s'appuyant sur une procédure de construction du modèle analytique équivalent d'un dispositif électromécanique.

5.1. Les problèmes généraux concernant l'exploitation des résultats

La validité des résultats

En s'appuyant sur les polynômes de Taylor la validité et la précision des développements sont les questions essentielles qui se posent durant l'utilisation de notre méthode. Un critère explicite pour déterminer a priori le rayon de convergence est généralement introuvable [Petin96, Guill94], l'expérience montre qu'il dépend des matrices EF et à ce titre du maillage. En pratique, le rayon de convergence doit être estimé a posteriori, c'est-à-dire une fois les polynômes calculés. Même si l'erreur d'approximation peut être déterminée assez précisément dans ce cas, il n'est pas évident de trouver un bon critère de validité sur toutes les grandeurs désirées. Il faut donc compter sur l'expérience de l'utilisateur. Souvent, il est nécessaire d'effectuer quelques calculs aux limites du domaine de variations pour étalonner la validité et la précision.

Les problèmes concernant l'extraction d'information paramétrée

Nous avons réalisé les outils permettant de calculer toutes les dérivées de la solution et éventuellement de construire les polynômes de Taylor représentant les variables d'état. Il est évident qu'une utilisation directe de ces polynômes est idéale, c'est aussi la manière la plus simple et la plus rapide d'exploiter les résultats. Cependant, la grandeur résolue n'est pas toujours celle qui intéresse l'utilisateur, par conséquent il faut souvent avoir recours à une des approches suivantes:

❶ Choisir (construire) une nouvelle formulation qui sera résolue pour les variables désirées,

❷ Définir (par un moyen quelconque) les nouveaux polynômes de cette grandeur à partir des polynômes de variables d'état,

❸ Calculer numériquement cette grandeur aux points définis, c'est-à-dire correspondant à certaines valeurs de paramètres.

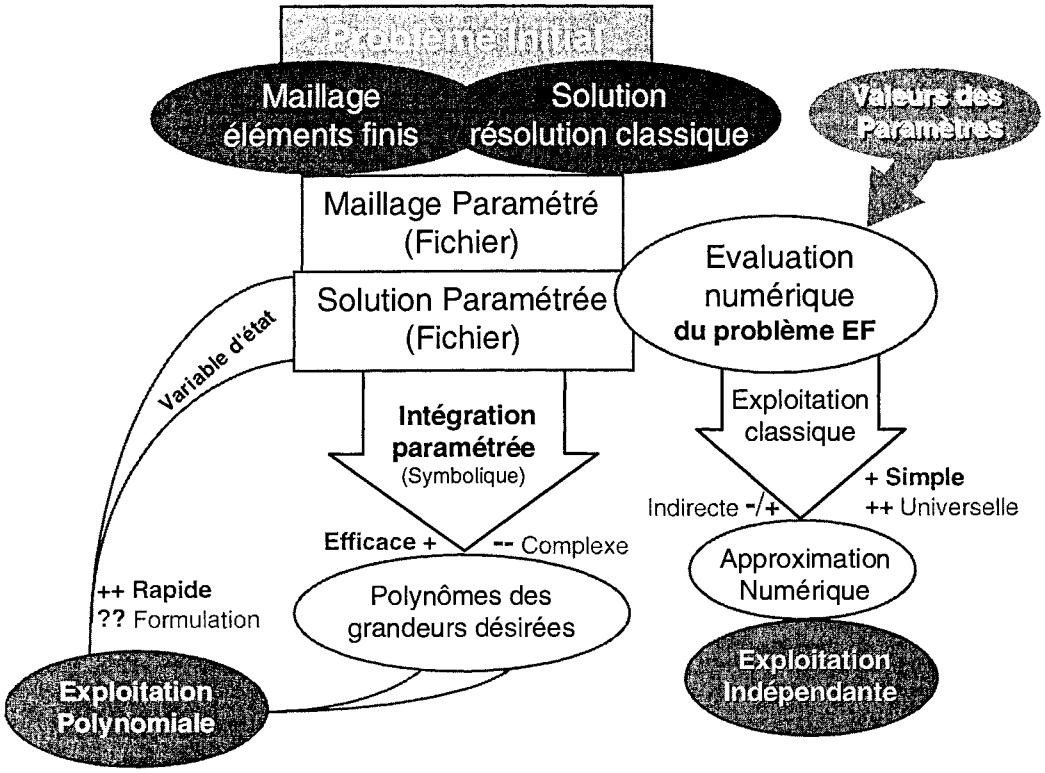


Fig. 5.1. Les trois approches de l'exploitation des résultats paramétrés

La première approche est toujours efficace (et réutilisable facilement) mais son implantation est très coûteuse surtout en calcul paramétré. D'ailleurs, cette solution est partielle et n'est pas applicable, par exemple, pour calculer les grandeurs globales: l'énergie, la force, etc..

Pour la deuxième approche, l'extraction s'organise effectivement d'une manière très similaire à celle de la phase d'intégration élémentaire (voir le chapitre précédent). Regardons, par exemple, un problème linéaire 2D, dans lequel l'énergie magnétique est calculée par l'équation:

$$\mathbf{W} = \iint_{\Omega} \frac{\nu}{2} \mathbf{B}^2 \, d\Omega = \iint_{\Omega} \frac{\nu}{2} (\mathbf{B}_x^2 + \mathbf{B}_y^2) \cdot d\Omega \tag{5.1}$$

En utilisant le domaine de référence Ω_r, nous pouvons dériver cette équation afin de calculer les dérivées énergétiques comme suit:

$$\frac{\partial \mathbf{W}}{\partial p} = \frac{\nu}{2} \iint_{\Omega} (2\mathbf{B}_x \frac{\partial \mathbf{B}_x}{\partial p} + 2\mathbf{B}_y \frac{\partial \mathbf{B}_y}{\partial p}) \cdot d\Omega + \frac{\nu}{2} \iint_{\Omega r} \mathbf{B}^2 \frac{\partial |G|}{\partial p} \cdot d\Omega_r \tag{5.2}$$

où |G| est le déterminant de la matrice jacobienne de la transformation géométrique.

Pour les paramètres physiques, la formule généralisée s'écrit:

$$\frac{\partial^n \mathbf{W}}{\partial p^n} = \frac{\nu}{2} \iint_{\Omega} \sum_{k=0}^{n} C_n^k \left(\frac{\partial^k \mathbf{B}_x}{\partial p^k} \frac{\partial^{n-k} \mathbf{B}_x}{\partial p^{n-k}} + \frac{\partial^k \mathbf{B}_y}{\partial p^k} \frac{\partial^{n-k} \mathbf{B}_y}{\partial p^{n-k}} \right) \cdot d\Omega \tag{5.3}$$

Si le problème est résolu pour le vecteur magnétique $\vec{\mathbf{A}} = \{0,0,A\}^T$,

nous avons $B_x = \frac{\partial A}{\partial y}$, $B_y = -\frac{\partial A}{\partial x}$.

En utilisant les outils de calculs symboliques, en particulier l'intégration de Gauss (chapitre 4), les équations (5.2) et (5.3) peuvent être calculée en s'appuyant sur les fonctions de base des éléments et les polynômes de $\{\mathbf{A}\}$. Il est évident que l'extraction du polynôme sera beaucoup plus compliquée que la déduction classique. La difficulté réside encore dans certains aspects secondaires de calculs. Prenons comme exemple très simple, l'interpolation nodale de la variable d'état "**A**" au point fixe **P**(x,y,z) qui s'écrit:

$$\mathbf{A}(x,y,z) = \sum_{i=1}^{NN} \alpha_i(x,y,z) \cdot \mathbf{A}_i \tag{5.4}$$

Classiquement et en cas de paramétrage physique, il faut simplement composer le somme (5.4) dans un élément "connu" en utilisant les fonctions de forme α_i qui sont aussi constantes. Dans le cas de paramétrage géométrique, le calcul est beaucoup plus complexe. Premièrement les fonctions α_i dépendent de paramètres; deuxièmement, il n'est pas sûr que ce point P reste dans le même élément initial.

À cause de toutes les difficultés mentionnées ci-dessus, la troisième approche, l'approche numérique nous paraît la plus simple et la plus universelle. Dans ce cas là, une grandeur est calculée à partir des valeurs numériques de la solution, c'est-à-dire calculée conformément aux valeurs exactes de paramètres. Remarquons qu'un "post-processeur" classique assure déjà l'extraction de toutes les grandeurs usuelles et en plus, permet de les présenter sous la forme désirée. Il ne faut donc construire que des outils pour gérer l'ensemble des évaluations nécessaires. En réalité, cette opération consiste premièrement à générer les valeurs des paramètres, ensuite à assurer la synchronisation de toutes les évaluations des entités et des grandeurs, dites de base, enfin à calculer et stocker automatiquement la grandeur désirée.

Les implantations supplémentaires

Nous avons remarqué dans le paragraphe ci-dessus que l'extraction directe est envisageable pour définir les polynômes des grandeurs désirées et nécessitait, si elle était retenue, des outils spécifiques. Cependant, même avec l'approche choisie, des outils supplémentaires sont indispensables pour assurer les évaluations élémentaires dans le module de post-traitement.

En premier lieu, un interfaçage avec les fichiers de données est nécessaire. Notons que nous avons choisi d'enregistrer les résultats de calculs, c'est-à-dire les polynômes de la position des noeuds (chapitre 3) et des variables d'état (chapitre 4), sur les fichiers (Fig.5.1). Le contrôle et la synchronisation des données sont donc essentiels pour le fonctionnement normal de la procédure. Ensuite un estimateur des erreurs polynomiales est également souhaitable. Enfin, un "visualiseur" spécifique lié à des nouvelles représentations des résultats paramétrés (l'animation, l'enchaînement, etc.) est envisageable.

5.2. Tests effectués sur les exemples

Compte tenu de l'importance des implantations dans toutes les étapes de calculs, la validation de notre méthode nécessite un grand nombre de tests. Dans ce paragraphe, nous présenterons quelques résultats validant les calculs paramétrés. Ces exemples servent aussi à déterminer la performance de notre mise en oeuvre. En plus, nous voulons exprimer nos expériences acquises dans l'optimisation de calculs et préciser enfin quelques restrictions concernant la méthode avec certaines limitations propres à nos développements présents.

Nous nous sommes particulièrement intéressés aux temps de calculs. Dans ce domaine, les facteurs influants sont les suivants:

❶ Le nombre de paramètres actifs.

❷ L'ordre de dérivation de chaque paramètre et l'ordre de croissance des paramètres dans les régions déformées. Notons qu'une optimisation dans la définition des paramètres et éventuellement dans la description de la géométrie correspondante, est essentielle pour réduire le volume de calculs.

❸ L'influence des variations, c'est-à-dire le nombre d'éléments finis déformés lors d'une modification des paramètres ou aussi le nombre d'éléments à prendre en compte pour l'intégration paramétrée. L'amélioration peut être réalisée par les techniques particulières présentées dans (§3.5).

❹ Le nombre total de variables d'état, ce nombre influe sur le temps nécessaire pour résoudre les systèmes matriciels.

Dans chaque problème, un ou plusieurs facteurs ci-dessus sont décisifs et une optimisation éventuelle peut apporter les économies essentielles dans le temps de calcul. Par exemple dans le problème du contacteur ci-dessous (§5.3), le temps du calcul paramétré (d'ordre 10 pour l'entrefer) est réduit jusqu'au niveau de deux résolutions ordinaires lorsqu'on introduit des lignes supplémentaires afin de limiter la région déformable. Il est évident que le calcul paramétré est généralement beaucoup plus coûteux que ce cas particulier. Notre expérience montre que le temps de calcul est proportionnel au nombre d'éléments déformés (facteur ❸) et pratiquement en fonction exponentielle des autres facteurs.

5.2.1. Le traitement en multi-paramètre

L'intérêt du calcul paramétré réside, tout d'abord, dans la possibilité de traiter à la fois plusieurs paramètres. Un calcul de test a été réalisé sur TEAM problème 22 [TEAM22], qui possède 6 paramètres géométriques (voir leur définition dans l'annexe B). Dans ce problème nous avons essayé, entre autres, de réduire effectivement les ordres de dérivation pour limiter le temps de calculs (des heures de machines en plus pour quelques ordres supplémentaires). Après toutes les optimisations possibles, la résolution paramétrée est encore très coûteuse. Le coût de calculs est d'environ 11000 fois celui d'une résolution classique (Tab.5.1). Notons que le volume de données intermédiaires est aussi très important (50 MB) par rapport aux données du problème initial (300 KB). Par contre, le résultat final est relativement plus compact (d'environ 5 MB).

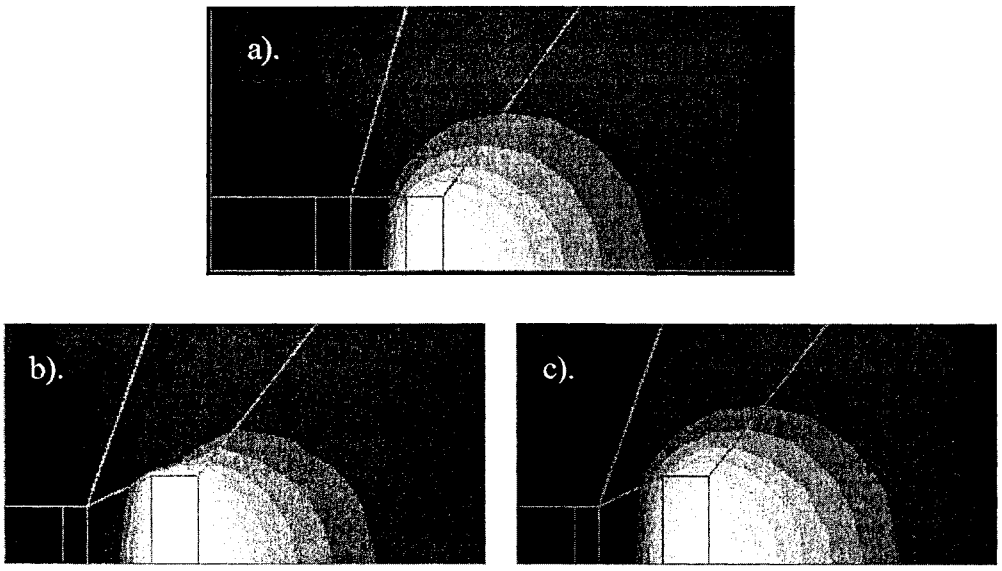


Fig.5.2. La solution pour TEAM problème 22 [FLUX3D]
a). Solution initiale;
b). Évaluation polynomiale; c). Re-analyse sans remaillage

Paramètres	R_i	d_{12}	$h_1/2$	$h_2/2$	d_1	d_2
Valeur Initiale	1.7	0.9	1.2	1.2	0.6	0.6
Valeur Évaluée	1.2	1.1	1.0	1.5	0.4	0.8

Table 5.1: La comparaison du temps de calculs

Nb. de Paramètres	Temps de résolution	
0	<1 minute	Résolution classique
3	10-20 minutes	Différence des combinaisons
4	50-75 minutes	Idem.
5	6 heures	Temps partiel de 6 param.
6	35 heures	(*)
Évaluation	2-3s	

(*) Ce calcul est réalisé en grande partie la nuit sur station HP782/c200 (RAM: 512 MB)

Il est évident que dans ce problème le contrôle de la validité des résultats est complexe. Il nous parait raisonnable d'appliquer un intervalle de variation assez limite afin d'assurer la précision nécessaire. Compte tenu du temps de calcul considérable, il vaut mieux limiter le nombre de paramètres géométriques à 3-4 pour chaque phase d'analyse, au-dessus de ce nombre le calcul est encore possible mais son coût est généralement trop important (Tab.5.1).

5.2.2. L'exemple d'une rotation

Dans le chapitre 3, nous avons présenté le dérivateur de maillages, qui, en particulier, permet la prise en compte de la rotation dans une structure géométrique. Notons que la rotation a été simulée par des polynômes d'ordre 5 à 7 ce qui est beaucoup plus élevé que les variations d'ordre 1 ou 2 dans les structures géométriques ordinaires. Par conséquent, l'intégration paramétrée sur les éléments est beaucoup plus coûteuse. Au niveau du problème global, le nombre de dérivées des variables d'état (nombre de résolutions successives) peut être limité pourtant au même ordre que celui du cas ordinaire.

En fait, nous avons testé deux types de rotation: la rotation de large échelle, non-destructive pour le maillage (Fig.5.3) et la rotation dans une machine tournante qui est assez limitée à cause de la cassure sur les éléments du maillage (Fig.5.4). Les résultats obtenus sont satisfaisants. Dans le premier problème, le temps de calculs paramétrés est d'environ 7-8 fois celui de la résolution classique et 3-4 fois pour le problème plus volumineux du deuxième exemple.

Table 5.2: Statistiques de calculs [FLUX3D]

Problème	(Fig.5.3)	(Fig.5.4)
Nombre d'éléments/noeuds	666/368	2050/1050
Éléments déformés	120	328
Résolution initiale	15s	60s
Résolution paramétré (*)	1' + 40"	2' + 1'30"
Angle de rotation valide	15° ± 10°	± 5°

(*) Le temps maximal tient compte de la dérivation du maillage et de l'intégration paramétrée

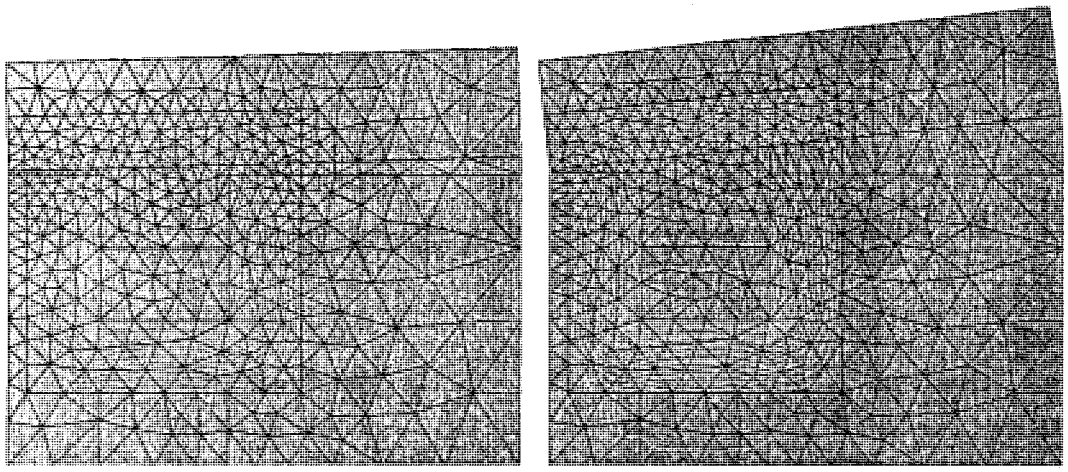


Fig.5.3. Le contacteur électrique - la rotation en état non destructive de maillages

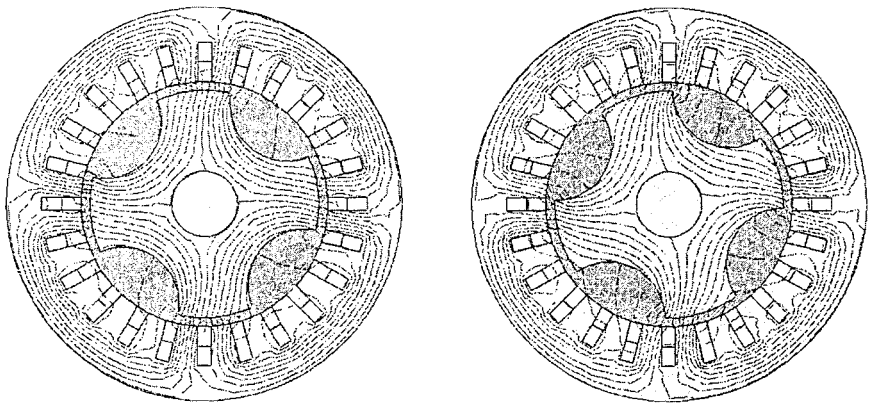
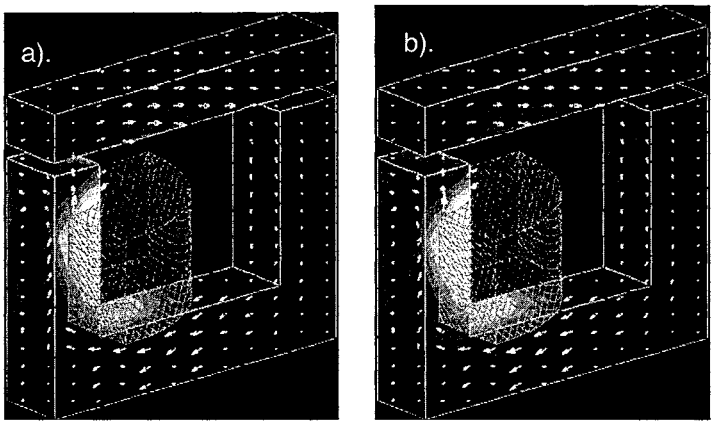


Fig.5.4. Machine tournante - la rotation incrémentielle

5.2.3. L'exemple 3D

Bien que tous les outils soient présents, nous n'avons pas eu le temps de construire et vérifier une grosse application 3D. Sur la (Fig.5.4) un simple exemple à un paramètre (l'entrefer ξ) est donc proposé. Nous nous sommes contentés, pour l'instant, de souligner que le solveur paramétré et sa bibliothèque des modèles de dérivation sont encore en état de développements. Quant au dérivateur de maillages, il est prêt pour traiter tous les types de maillages.



a).Initial ξ=2mm b). Perturbé ξ=3mm

Fig.5.5. Un simple exemple en 3D

Table 5.3: Statistiques de calculs [FLUX3D]

Nombre d'éléments/variables	6256/3660
Éléments déformés	1040
Résolution initiale	1'
Résolution paramétré	5'

5.3. La construction du modèle analytique équivalent

Avant l'apparition de l'ordinateur, le schéma équivalent et les formules empiriques étaient le moyen universel de décrire un dispositif réel. Leur utilisation était très simple, il suffisait d'un crayon et de l'expérience de l'ingénieur-concepteur.

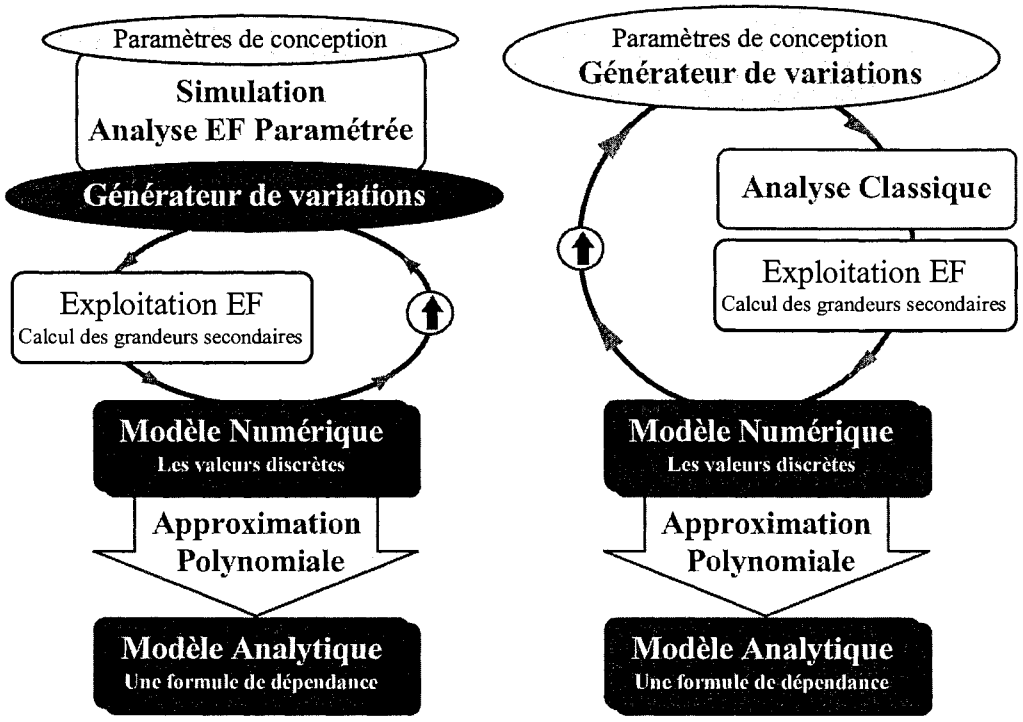
De nos jours, les simulations numériques sophistiquées, dont le calcul EF est un exemple, remplacent de plus en plus les calculs simplifiés. Les logiciels disponibles permettent de décrire précisément le comportement des dispositifs étudiés et de donner toutes les grandeurs et les caractéristiques désirées. Il existe cependant quelques problèmes:

◊ L'enchaînement de plusieurs calculs est toujours très complexe et coûteux.
Remarquons aussi que l'exploitation efficace des logiciels de calculs demande une longue formation et de l'expérience.

◊ Le volume de données numériques est considérable, il contient surtout fréquemment beaucoup d'informations inutiles pour une application précise: des milliers de variables d'état, la description complète du problème EF (maillage, formulation, etc.).

Bref, les méthodes numériques de type EF paraissent bien lourde à l'utilisation, comparées aux "schéma équivalents" bien plus simple à comprendre et surtout à manipuler, particulièrement dans une phase d'optimisation. L'idéal serait de définir un modèle analytique relativement compact utilisable partout à partir des informations fournies par un code numérique.

Dans cette partie, nous allons présenter une procédure pour définir un modèle analytique de dispositif électromagnétique en utilisant les résultats de calculs paramétrés. Pour extraire les informations, nous utilisons le module d'exploitation EF comme une "boîte noire" afin d'obtenir un ensemble de valeurs discrètes de la grandeur désirée (Fig.5.6a). Nous pouvons aussi utiliser un module d'analyse classique (Fig.5.6a). Ensuite un outil indépendant réalise les approximations nécessaires. Remarquons que le nombre de points d'évaluation peut être assez grand et un algorithme simple, comme l'approximation polynomiale par moindre carrée, est donc suffisant pour une bonne représentation. La possibilité de combiner les résultats de plusieurs analyses est un autre point fort de l'approximation externe.



a). Utilisation de résultats paramétrés b). Utilisation d'analyses classiques

Fig.5.6. Deux approches pour la construction du modèle analytique

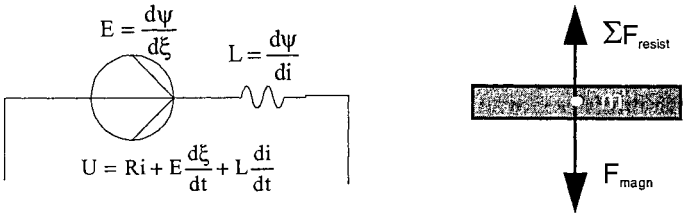
Remarquons que l'analyse paramétrée (Fig.5.6a) permet d'évaluer rapidement la grandeur désirée en plusieurs points et obtenir une représentation numérique assez fine de cette grandeur. Cette possibilité simplifie, en premier lieu, la fonction d'approximation qui suit. Elle peut donner, ensuite, un avantage net en cas des traitements complexes pour les grands problèmes avec le multiparamètrage et la construction de dépendances complexes (par exemple les fonctions objectifs dans l'optimisation globale [Salud97]).

5.3.1. Le modèle analytique d'un contacteur

Pour un contacteur, il est important de définir sa caractéristique électromécanique, c'est-à-dire de résoudre les équations suivantes (Fig.5.7):

(5.5) $m\frac{d^2x}{dt^2} = F_{magn} + F_{resist}$ est l'équation mécanique du système.

(5.6) $U = Ri + \frac{d\psi}{dt}$, est l'équation électrique.



a). Schéma électrique b). Structure mécanique

Fig.5.7. Schéma équivalent du contacteur

Dans ces équations, apparaissent la force magnétique et le flux à travers la bobine. Ces grandeurs dépendent de la position relative "x" (de la partie mobile), du courant I et de la configuration géométrique, c'est-à-dire de certains paramètres géométriques de construction.

En fait, en l'absence de saturation magnétique, le flux dépend directement du courant I et la force est en I^2. Par conséquent, dans la suite, nous étudierons seulement les dépendances géométriques.

5.3.2. La simulation numérique

Les calculs sont réalisés sur la structure géométrique (Fig.5.8), dans laquelle deux paramètres géométriques actifs sont: l'entrefer "ξ" qui représente "la position x" dans l'équation (5.5) et "la section λ" du circuit magnétique qui est un paramètre de conception (Fig.5.8). Nous avons réalisé, en effet, deux analyses paramétrées correspondant à deux valeurs de l'entrefer (ξ=1.6 et ξ=5), ceux qui permet d'élargir l'intervalle de variation de ce paramètre. Avec les lignes de limitation (voir §3.5), un maillage paramétré supporte bien les deux analyses, c'est-à-dire qu'on n'a pas besoin de remailler d'une analyse à l'autre.

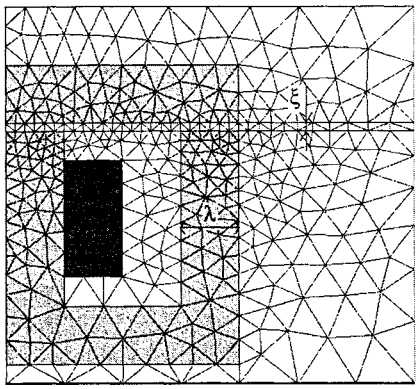


Fig. 5.8. Le contacteur électrique

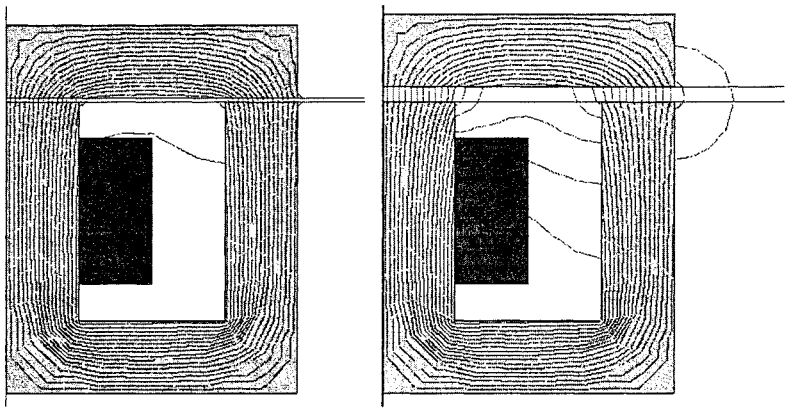


Fig. 5.9. Deux analyses effectuées

Table 5.4: Statistiques de calculs [FLUX3D]

Problème	ξ=1.6 mm	ξ=5 mm
Nombre d'éléments/noeuds	740/397	
Nombre de paramètres	2	
Éléments déformés	72	
Résolution initiale	15"	
Résolution paramétré	(*) 1' + 5'	30" + 5'
Évaluation de solution paramétrée	1" (instantanée)	

(*) Le temps maximal de dérivation du maillage et l'intégration paramétrée

5.3.3. L'approximation polynomiale

L'approximation est un des outils fondamentaux de l'analyse numérique, les méthodes d'approximations sont donc très nombreuses (moindres carrées, Lagrangien, Newton, linéaires, plan d'expériences etc.). En pratique, le choix d'une méthode dépend principalement du volume de données numériques et aussi du type de fonctions. Dans notre cas, on peut évaluer rapidement la grandeur désirée en plusieurs points et obtenir une représentation numérique assez fine de cette grandeur. En fait, une exploitation directe des tableaux numériques est envisageable, mais ici nous voulons construire une formule analytique de cette grandeur qui est donc plus compacte et aussi plus présentable que les tableaux des valeurs. En utilisant la méthode de dérivées d'ordre élevé dans la phase d'analyse, une approximation sur la base polynomiale est donc l'approche la plus naturelle que l'on puisse choisir. Dans ce cas, il faut construire les polynômes d'ordre défini de la grandeur désirée par rapport aux paramètres de conception **p** comme suit:

$$\textbf{(5.7)} \qquad F(p)=\Pi^N(p)=a_0+a_1.(p-p_0)+\ldots+a_N.(p-p_0)^N=\sum_{j=0}^{N}a_j\frac{(p-p_0)^j}{j!}$$

où N est l'ordre de polynôme et p_0, la valeur centrale de paramètre

En effet, il faut calculer les coefficients $\mathbf{a_j}$ de (5.7) à partir de l'ensemble des valeurs numériques $F(p_k)$ de la grandeur désirée. En utilisant la méthode de moindres carrées, cette recherche conduit à minimiser la fonction suivante:

$$\textbf{(5.8)} \qquad \Re(a_0,a_1,\ldots,a_N)=\sum_{k=1}^{m}\left(F(p_k)-\Pi(p_k)\right)^2$$

où **m** est le nombre total des valeurs numériques de cette grandeur

Il est évident que cette fonction sera minimale si sa dérivée

$$\frac{d\Re}{da_j}=0 \text{ pour tous les "}\mathbf{a_j}\text{".}$$

En utilisant ce critère, nous obtiendrons un système matriciel de N inconnues dont la résolution nous donnera tous les coefficients "inconnues $\mathbf{a_j}$". Dans le cas de multi-paramètres, nous avons utilisé, comme base, le polynôme complet par rapport à tous les paramètres, c'est-à-dire prise en compte toutes les dérivées croisées. Un exemple du polynôme de deux paramètres est donné sur le tableau (5.7).

En ce qui concerne l'implantation pratique, cette procédure d'approximation s'organise effectivement en trois étapes:

◊ D'abord un outil assistant sert à générer le fichier de commandes pour calculer automatiquement les grandeurs désirées (force et flux) dans un module d'exploitation classique (FLUX3D).

◊ Les résultats des calculs sont récupérés (vérifiés, pré-traités) et ensuite enregistrés en un format portable.

◊ Enfin, le module d'approximation est lancé pour construire le polynôme de cette grandeur. Ce module consiste en un outil indépendant qui a mis en oeuvre l'algorithme d'approximation par moindres carrées décrit ci-dessus.

Notons que plusieurs "options d'utilisateurs" sont disponibles dans chaque étape, ce qui permet de réaliser l'approximation pour les diverses études. En effet, nous avons étudié la dépendance de la force magnétique par rapport aux paramètres géométriques ("ξ" et "λ"). Sur un assez grand intervalle de variations, nous avons trouvé que l'approximation doit aller jusqu'à l'ordre 7 pour décrire la force en fonction de l'entrefer (Fig.5.10), par contre le 4ème ordre est suffisant pour un paramètre comme "la section magnétique" (Fig.5.11). Généralement, en présence de plusieurs paramètres le pré-calcul, dit à une dimension (Tab.5.1, Tab.5.2), est une bonne manière pour estimer l'ordre définitif du polynôme résultant (Tab.5.3). Remarquons enfin que la construction des polynômes par l'outil indépendant est assez rapide surtout en présence de données formattées.

Table **5.5**: Les coefficients d'approximation de polynôme: $\mathbf{F_{magn}}=\Pi^n(\xi)$

Polynôme	a1	a2	a3	a4	a5	a6	a7
$\Pi^4(\xi)$	-9.081338	2.569443	2.838251	-0.757203			
$\Pi^5(\xi)$	5.35183	4.400904	-1.651174	-0.976309	0.271964		
$\Pi^6(\xi)$	9.729225	-2.6981663	-3.404419	0.757738	0.401039	-0.09331	
$\Pi^7(\xi)$	1.93708	-4.6638	1.299527	1.33568	-0.29997	-0.12925	0.02967
a_0=-5.55	$F_{magn}=\Pi_N(\xi)=a_0+\sum_{i=1}^{N} a_i \frac{(\xi-\xi_0)^i}{i!}$						

Table **5.6**: Les coefficients d'approximation de polynôme: $\mathbf{F_{magn}}=\Pi^n(\lambda)$

Polynôme	a1	a2	a3	a4
$\Pi^2(\lambda)$	-2.1516694	-0.1154109	0	0
$\Pi^3(\lambda)$	-1.0353994	-0.1154109	-0.004241	0
$\Pi^4(\lambda)$	-1.03645688	-0.08731069	-0.00425711	-9.0962E-05
a_0=-5.55	$F_{magn}=\Pi_N(\lambda)=a_0+\sum_{j=1}^{N_d} a_j \frac{(\lambda-\lambda_0)^j}{j!}$			

Table **5.7**: Les coefficients d'approximation de polynôme: $\mathbf{F_{magn}}=\Pi^m(\xi,\lambda)$

$\xi^7 \setminus \lambda^4$	0	1	2	3	4
0	**-5.55**	-1.03645688	-0.08731069	-0.00425711	-9.09624E-05
1	1.93708125	0.53154589	0.0539824	0.00208723	2.29986E-05
2	-4.6638	-0.40685649	-0.00758061	-0.00118034	-6.68861E-05
3	1.29952688	0.04851075	-0.00674582	0.00016289	2.48249E-05
4	1.33568438	0.08800939	-0.00185793	0.0001681	1.74013E-05
5	-0.29996944	-0.00684551	0.00217398	2.177E-07	-4.88352E-06
6	-0.1292535	-0.01205552	-0.00030576	-3.0215E-05	-1.57752E-06
7	0.02967469	0.00221081	-5.8957E-06	5.1269E-06	3.9648E-07
$F_{magn}=\Pi(\xi,\lambda)=\sum_{i=0}^{N_e}\sum_{j=0}^{N_d} a_{ij}\frac{(\xi-\xi_0)^i(\lambda-\lambda_0)^j}{i!\ j!}$					

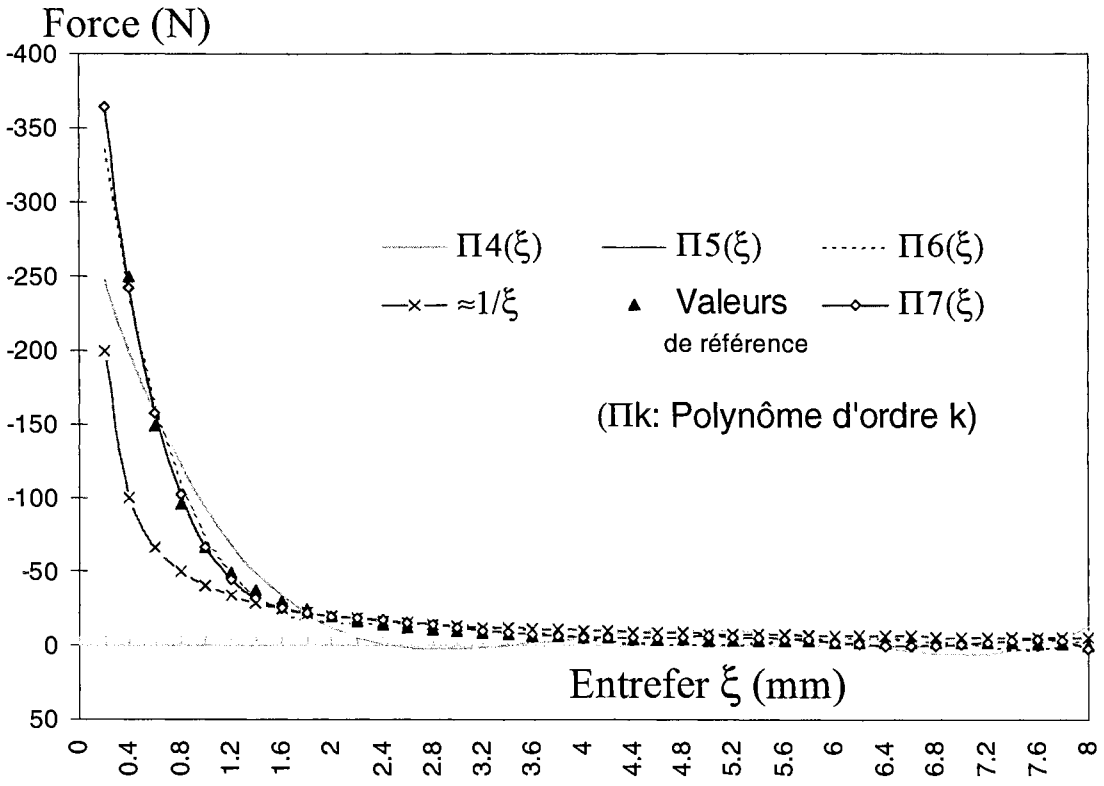


Fig.5.10. La force magnétique en fonction de l'entrefer

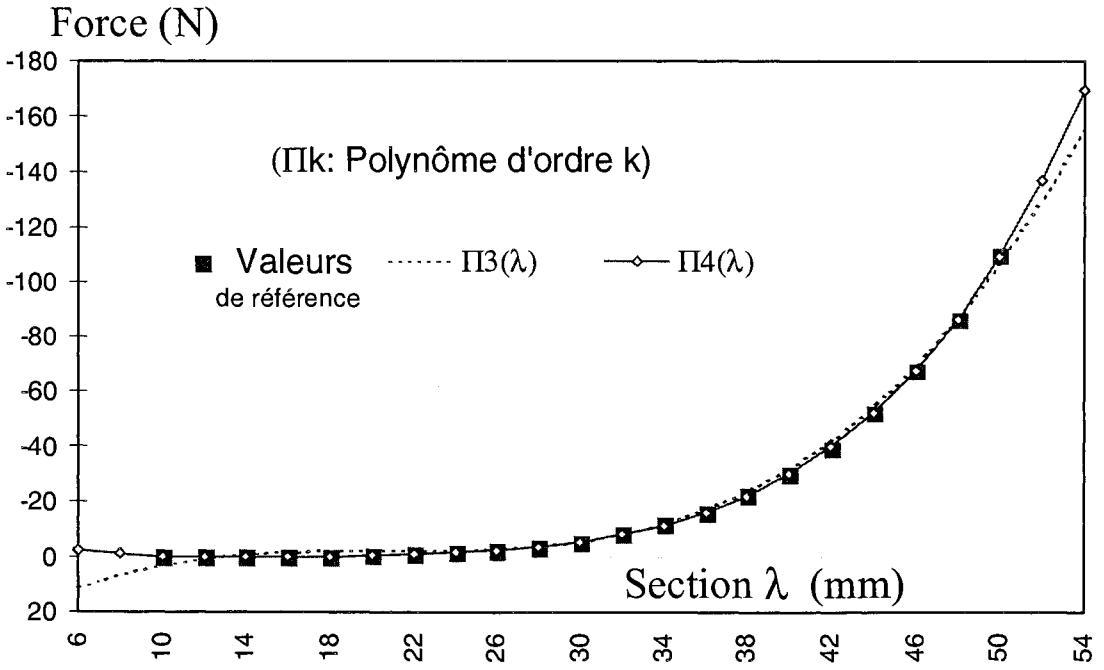


Fig.5.11. La force magnétique en fonction de la section
(la densité de courant et l'entrefer sont constants)

5.4. Réflexions sur les possibilités de la méthode des dérivées d'ordre élevé

Au LEG, nous avons réalisé des premières expériences sur la méthode de dérivées d'ordre élevé. Notre implantation de méthode de dérivées d'ordre élevé, en réalité, n'est pas encore complète et ne recouvre qu'une partie des problèmes de l'électromagnétisme. Entre autres, l'absence pour l'instant de traitements des paramètres physiques pénalise à certains niveaux les applications réalistes de ce calcul. Cependant, nous pensons que la poursuite de ces développements est méritée et nous voulons résumer ici quelques utilisations possibles des calculs paramétrés.

5.4.1. L'optimisation globale

Dans son travail, Lucas SALUDJIAN [Salud97] a couplé le calcul paramétré avec une méthode stochastique performante, l'algorithme génétique (Fig.5.12a). Les premiers résultats obtenus sont très encourageants, bien que la précision et la validité des calculs doivent être encore améliorés. En effet, une amélioration de la procédure d'optimisation est envisageable en s'appuyant sur le concept de modèles analytiques présenté ci-dessus (Fig.5.12b).

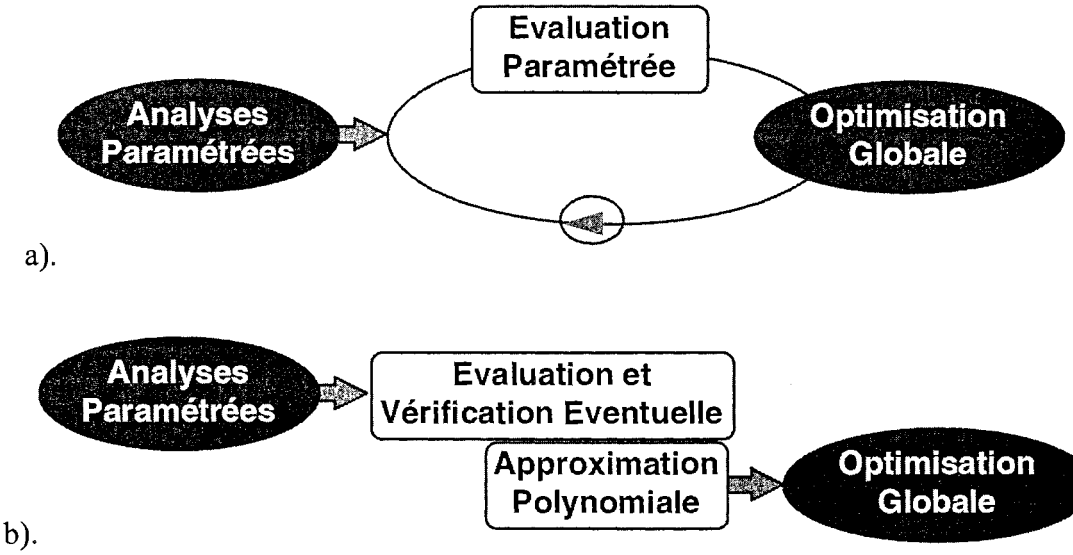


Fig.5.12. La chaîne d'optimisation globale

5.4.2. L'intérêt des résultats partiels

En étudiant certains exemples, nous avons remarqué que la plage de validité du maillage paramétré est souvent plus large que celle des résultats finaux de calcul, particulièrement, pour un maillage optimisé en vue de la dérivation (voir §3.5). Si cette propriété est générale, un enchaînement de calculs paramétrés sans remaillage est donc possible. Cette approche est applicable également pour répéter le calcul ordinaire, surtout en remarquant que le dérivateur semi-analytique permet de générer rapidement le maillage paramétré et de l'évaluer instantanément. En fait, un tel calcul est intéressant particulièrement dans la procédure d'optimisation par une méthode déterministe directe. Notons aussi que le calcul du gradient dans un problème non-linéaire est une autre possibilité intéressante. En effet, le calcul de la solution paramétrée en non-linéaire est peut-être trop difficile, par contre, un calcul du gradient et de quelques dérivées, est réalisable par la résolution du système linéaire lorsque les variables d'état sont connues au point de travail.

Conclusion

En s'appuyant sur un "post-processeur" classique, nous avons étudié quelques exemples de calculs paramétrés et donc présenté des problèmes reliés à l'exploitation des polynômes représentant les résultats. Tout d'abord, les exemples traités ont montré que la méthode de dérivées d'ordre élevé est accessible et les premiers résultats obtenus sont plutôt prometteurs pour les diverses applications. Nous avons remarqué cependant que le contrôle de validité est assez difficile et le calcul peut-être très coûteux. Par conséquent son utilisation doit être très soignée, c'est-à-dire qu'il faut tirer le meilleur de toutes les optimisations possibles (voir les chapitres précédants). Entre autres, nous avons proposé les outils pour construire le modèle analytique d'un dispositif électromécanique. Cette approche est basée sur une approximation polynomiale des grandeurs désirées à partir des résultats d'analyses paramétrées.

Nous signalons enfin que notre étude est encore limitée, le développement mérite d'être poursuivi dans les deux parties suivantes:

◊ Compléter l'implantation des outils de calculs paramétrés,

◊ Étudier les applications possibles de résultats paramétrés.

NGUYEN Thanh Nam

CONCLUSION GÉNÉRALE

Conclusion

Perspectives

✣ ✣ ✣ ✣ ✣ ✣ ✣

CONCLUSION GÉNÉRALE ET PERSPECTIVES

Conclusion

Nous sommes partis de l'idée d'une analyse éléments finis paramétrée dans laquelle des résultats de calculs sont représentés sous la forme des développements de Taylor par rapport aux paramètres de conception. Il est évident qu'une telle représentation de résultats d'analyses est très attractive en synthèse de structures, pour la conception et pour l'optimisation.

Au cours de ce travail de thèse nous avons étudié les approches pratiques pour intégrer le calcul paramétré au sein de la méthode des éléments finis. En effet, notre enthousiasme a été modéré pour développer une analyse de sensibilité d'ordre élevé par rapport aux paramètres géométriques dans un grand logiciel de calculs en respectant l'ensemble des outils existants. L'originalité de cette implantation réside principalement dans une organisation de calculs qui est basée notamment sur les opérations symboliques rapides.

Nous avons développé et validé tout d'abord une méthode originale pour dériver le maillage EF, celle ci est une étape indispensable pour le traitement des paramètres géométriques. En fait, le maillage paramétré sert de base pour dériver le système matriciel associé au problème EF dans la phase de résolution. Ce maillage est nécessaire aussi en vue d'une exploitation paramétrée ou d'une résolution enchaînée. Nous avons mis en oeuvre effectivement une méthode semi-analytique qui se décompose en une dérivation symbolique des noeuds frontières suivi par une propagation directe de chaque dérivée sur l'ensemble des noeuds internes. C'est

une approche générale et très rapide qui permet, en outre, d'identifier le comportement précis de toutes les entités géométriques lors de variations des paramètres. Notons que le caractère analytique de la dérivation nous donne la possibilité de traiter les déformations complexes de la géométrie, en particulier, la rotation qui est souvent présente dans les structures électrotechniques.

Dans la phase suivante, les dérivées des variables d'état, c'est-à-dire les dérivées de solution EF, ont été calculées par la résolution successive du même système matriciel avec des seconds membres qui sont assemblés récursivement. Un langage de commandes a été introduit pour interpréter la partie d'assemblage de cette procédure "paramétrée". L'assemblage a été organisé effectivement en un calcul préparatoire au niveau des intégrales élémentaires et un assemblage récursif du second membre du système global. Le pré-calcul et l'enregistrement intermédiaire améliorent beaucoup l'efficacité des opérations, ils permettent, entre autres, d'accélérer sensiblement l'assemblage global. En effet, le concept du langage d'interprétation nous donne une grande souplesse pour manipuler la procédure de calculs et aussi des facilités pour une poursuite possible de développements.

Fort d'une méthode établie, nous avons présenté quelques exemples intéressants de calculs paramétrés et abordé aussi des problèmes éventuels reliés à l'exploitation, dite paramétrée. En étudiant l'organisation manoeuvre générale de l'exploitation des résultats paramétrés, nous avons proposé une procédure de construction du modèle analytique équivalent d'un dispositif électromécanique. Cette procédure se décompose effectivement en deux modules:

◊ Une implantation interne du logiciel EF, qui permet d'évaluer successivement tous les polynômes du problème et de calculer numériquement les grandeurs désirées.

◊ Un module indépendant qui assure les approximations nécessaires.

Au cours des chapitres, plusieurs problèmes techniques (liés surtout à l'informatique) ont été mis en évidence et résolus. La gestion de données, importantes pour l'efficacité des codes, nous a conduit, entre autres, à préconiser la programmation multi-langage très productive. Un gros travail a d'autre part été consacré sur l'optimisation des algorithmes afin d'accélérer la coûteuse procédure de calculs paramétrés.

Généralement, notre étude a montré que la mise en oeuvre de la méthode des dérivées d'ordre élevé est certes un gros investissement, mais qu'elle restait "accessible" même pour un grand

code de calcul très classique. Nous pensons donc avoir inspiré des bases utiles pour que l'analyse de sensibilité d'ordre élevé devienne plus réalisable. Cependant, la phase de validation doit être continuée et élargie.

Perspectives

En ce qui concerne la mise en oeuvre de notre méthode, la poursuite de développements est, selon nous, nécessaire dans les deux directions suivantes:

◊ Premièrement, l'ensemble des modèles d'intégration et d'assemblage paramétrés n'affecte pour l'instant, qu'une partie des problèmes électromagnétiques. En s'appuyant sur le langage existant, les modèles sont donc à compléter et aussi à optimiser.

◊ Deuxièmement, nous avons remarqué antérieurement que l'absence de traitements des paramètres physiques limite, à certains niveaux, les applications possibles. En utilisant les outils développés dans ce travail, une implantation en linéaire par rapport aux paramètres physiques est donc directement accessible. Pour le cas non-linéaire, l'utilisation du calcul symbolique et de l'interprétation par le langage de procédure, nous semble favorable pour générer la solution implicite, par exemple sous la forme de développement de Padé. Naturellement, ce traitement est très complexe et demande des améliorations importantes dans le concept du langage en créant des outils supplémentaires.

Au-delà des implantations informatiques de la méthode, l'étude des applications est l'autre grand axe de développements. Le couplage direct avec des algorithmes d'optimisation est souhaitable afin d'exploiter efficacement les résultats paramétrés. En effet, notre polynôme permet des évaluations rapides pour les nouvelles valeurs de paramètres, ouvrant ainsi la voie aux méthodes stochastiques. Enfin, la méthode des plans d'expériences pourrait être conjointement utilisée pour explorer les solutions sur des études multi-paramètres.

NGUYEN Thanh Nam

LEG

RÉFÉRENCES BIBLIOGRAPHIQUES

✧ ✧ ✧ ✧ ✧ ✧ ✧

RÉFÉRENCES BIBLIOGRAPHIQUES

[Alber88] ALBERTINI J.B., Contribution à la réalisation d'un logiciel de modélisation de phénomènes électromagnétiques en trois dimensions par la méthode des éléments finis: FLUX3D, *thèse de doctorat de l'INPG,* Grenoble 1988.

[AroCP91] ARORA J.S., CHAHAUDE A.I., PAENG J.K., "Multiplier methods for engineering optimization", *International Journal for Numerical Methods Enginneering*, no32, 1991, pp1485-1525.

[Baran91] BARANGER J., *Analyse numérique*, *Hermann*, 1991.

[Bossa89] BOSSAVIT A., "Edge elements for scatterings problems", *IEEE Transaction on Magnetism*, vol 25, no4, March 1989, pp.2816-2621.

[Bossa96] BOSSAVIT A., "Edge Elements for Magnetostatics", *International Journal of Numerical Modelling: Electronic Networks, Devices and Fields*, vol. 9, 1996, pp.19-34.

[ChaSi80] CHARI M.V.K. and SILVESTER P.P., *Finite Element in Electrical and Magnetic Field Problems*, *John Wiey & Sons*, New York 1980.

[CoPSN96] COULOMB J.L., PETIN P., SALUDJIAN L., NGUYEN T.N., R. PACAUT, "Sensitivity Analysis using High Order Derivatives", *Applied Comutational Electromagnetics Society Journal*, Vol 12 no2, 1997, pp.20-26.

[Coulo81] COULOMB J.L., Analyse Tridimensionnelle des Champs Electriques et Magnétiques par la méthode des élements finis, *Thèse de Doctorat d'Etat de l'INPG*, Grenoble 1981.

[Coulo83] COULOMB J.L., "A methodology for the determination of global electromecanical quantities from finite element analysis and its application to the evaluation of magnetic forces, torques and ", *IEEE Transaction on Magnetism*, vol MAG-19, no6, 1983, pp.2514-2519.

[CouSa86] COULOMB J.L. et SABONNADIÈRE J.C., *CAO en Electrotechnique*, *Hermès*, Paris 1986.

[DavRL85] DAVAT B., REN Z., LAJOLIE-MAZENC M., "The mouvement in field modeling", *IEEE Transaction on Magnetism*, vol 21, no6, 1985, pp.2296-2298.

[Demai91] DEMAILLY J.P., *Analyse numérique et Equations différentielles*, *Press Universitaire de Grenoble*, 1991.

[DemWa90] DEMERDASH N.A., WANG R., "Theoretical and numerical difficulties in 3D vector-potential methods in finite element magnetostatic computation", *IEEE Transactions on Magnetics*, Vol. 26, pp. 1656-1658, 1990.

[DhaTo84] DHATT G., TOUZOT G., Une présentation de la méthode des éléments finis, *Collection Université de Compiègne*, Maloine S.A. Editeur, Paris, 1984.

[GitCS89] GITOSUSASTRO S., COULOMB J.L. et SABONNADIÈRE J.C., "Performance derivative calculations and optimisation process", *IEEE Transaction on Magnetics,* Vol.25 1988, 126p

[Gitos88] GITOSUSASTRO S., Analyse de sensibilité et optimisation par rapport aux paramètres de construction dans des systèmes électromagnétiques, *thèse de doctorat de l'INPG,* Grenoble 1988, 126p

[Golov97] GOLOVANOV Cristian, Développement de formulation 3D en potentiel vecteur magnétique: application à la simulation de dispositifs éléctromagnétiques en mouvement, *thèse de doctorat de l'INPG,* Grenoble 1997, 206p.

[GolTs93] GOLIAS N.A., TSIBOUKIS T.D., "Adaptive Refinement Stratégies in Three Dimensions", *IEEE Transaction on Magnetism*, vol 29, no2, March 1993, pp.1886-1889.

[Goupy88] GOUPY J., *La méthode des plans d'expériences*, *Dunod*, Paris 1988.

[GuaST90] GUAMIERI M., STELLA A., Trevisan F., "A methodological analysis of different formulations for solving inverse electromagnetic problems", *IEEE Transaction on Magnetism*, vol 26, no2, March 1990, pp.622-625.

[Guill94] GUILLAUME P., Dérivées d'ordre supérieur en conception optimale de forme, *Thèse de Doctorat de l'Université Paul Sabatier Toulouse*, 1994.

[GuiMa94] GUILAUME Ph. and MASMOUDI M., "Computation of high order derivatives in optimal shape design", *Numerische Mathematik*, vol.. 67, 1994, pp.231-250.

[HenGe93] HENOT B.I., GEORGE P.L., "Optimisation de maillages tridimensionnels", *INRIA, SDRC, STRUCOME*, 1993, pp 318-329.

[HooSC91] HOOLE S.R.H., SUBRAMANIAM S., SALDANHA R., COULOMB J.L., SABONNADIÈRE J.C. "Inverse Problem Methodology and Finite Elements in the Indentification of Cracks, Sources, Materials, and their Geometry in Inaccessible Location", *IEEE Transaction Magnetism* vol. 27, no3, May 1991, pp.3433-3443

[HooSu92] HOOLE S.R.H. and SUBRAMANIAM S., "Higher finite element derivatives for the quick synthesis of electromagnetic devices", *IEEE Transaction Magnetism* vol. 28, no2, March 1992, pp.1565-1568

[HooWS91] HOOLE S.R.H., WEEBER K. and SUBRAMANIAM S. "Fictious Minima of Object Functions, Finite Elements Meshes and Edge Elements in Electromagnetic Devices Synthesis", *IEEE Transaction Magnetism* vol. 27, no6, Sept 1991, pp.4146-4149.

[Hughe87] HUGHES T., *The Finite Element Method. Linear Static and Dynamic Finite Element Analysis*, *Prentice-Hall International*, New Jersey 1987.

[JonSt87] JONES R. and STEWART I., *The art of C programming*, *Springer-Verlag*, New York 1987.

[Kaded93] KADED K., Optimisation de forme de machines électriques à l'aide d'un logiciel éléments finis et de la méthode des pénalités intérieures étendues, *thèse de doctorat de l'INPG,* Grenoble 1993, 116p

[Koeni91] KOENIG A., *Pièges du Langage C*, *Addison Wesley France*, 1991.

[KokHa93] KOK C.S., and HAHN S.Y., "Design Sensitivity Analysis for Shape Optimization of 3-D Electromagnetic Devices", *IEEE Transaction on Magnetism*, vol 29, no2, March 1993, pp.1753-1757.

[KokMH95] KOK C.S., MOHAMED O.A. and HAHN S.Y., "Nonlinear Shape Design Sensitivity Analysis of Magnetostatic Problems using Boundary Element Method", *IEEE Transaction on Magnetism*, vol 31, no3, May 1995, pp.1944-1947.

[Kouyo85] KOUYOUMDJIAN A., Sensibilité paramétrique par la méthode des éléments finis en électromagnétisme, *thèse de doctorat de l'INPG,* Grenoble 1985, 134p.

[LasTh93] LASCAUX P., THÉODOR R., *Analyse numérique matricielle appliquée à l'art de l'ingénieur*, *Masson*, Paris 1993. Tom I,II.

[Lomba92] LOMBARD Patrick, Couplage des équations électriques et magnétiques, *thèse de doctorat de l'INPG,* Grenoble 1992.

[MarHe98] MARÉCHAL Y., HÉRAULT C., "Méthodes de Galerkin sans maillage pour la simulation en éléctromagnétique", *Revue Internationale de Génie électrique*, vol 1, 1998, pp.99-140.

[Masmo87] MASMOUDI M., Outils pour la conception optimale de formes, *thèse d'état de*

l'Université de Nice, 1987, 242p.

[Minou83] MINOUX M., *Programmation Mathématique, Théorie et Algorithmes*, Bordas et CNET-ENST, ISBN 2-04-015487-6, 1983, Tom 1.

[Moren93] MORENCAY, L. Représentation paramétrée et modélisation de systèmes physiques pour la conception optimale, *thèse de doctorat de l'Université de Technologie de Compiègne,* 1993, 239p.

[Ng-TN95] NGUYEN Thanh Nam, Déformation de maillages éléments finis paramétrés. Couplage des logiciels FLUX3D et ADOMESH, *rapport DEA LEG-ENSIEG INPG,* Grenoble 1995, 35p.

[NguCo98a] NGUYEN T.N., COULOMB J.L, "Design sensitivity analysis using high order derivatives, Problems versus geometric parameters", Proc. *4th International Workshop on EM Fields*, May 1998, Marseille, France, pp.105-110.

[NguCo98b] NGUYEN T.N., COULOMB J.L, "High order FE derivatives versus geometric parameters, Implantation on an existing code", Proc. IEEE CEEF'98, June 1998, Tucson, Arizona USA. In press.

[ParCH93] PARK I., COULOMB J.L. and HAHN S.Y., "Implementation of Continuum Sensitivty Analysis with Existing Finite Element Code", *IEEE Transaction on Magnetism*, vol 29, no2, March 1993, pp.1787-1790.

[ParLH91] PARK I., LEE B.T. and HAHN S.Y., "Sensitivity analysis based on analytic approach for shape optimization of electromagnetic devices: Interface problem of iron and air", *IEEE Transaction on Magnetism*, vol 27, no5, September 1991, pp.4142-4145.

[ParLKH94] PARK I., LEE H.B., KWAK I.G. and HAHN S.Y., "Design Sensitivity Analysis for Steady Stata Eddy Current Problems by Continuum Approach", *IEEE Transaction on Magnetism*, vol 30, no5, September 1994, pp.3411-3414.

[PerB94] PERRIN-BIT R., Modélisation des machines électriques tournantes par la méthode des éléments finis tridimensionnels: calcul des grandeurs magnétiques avec prise en compte du mouvement, *thèse de doctorat de l'INPG,* Grenoble 1994, 123p.

[Petin96] PÉTIN Pascal, Etude de sensibilité à l'aide des dérivées d'ordre élévé dans la méthode des éléments finis. Application à l'électromagnétisme, *thèse de doctorat de l'INPG,* Grenoble 1996, 209p.

[RamFr97] RAMIREZ J.A. & FREEMAN E.M., "Sensitivity analysis for the Automatic Shape Design", *IEEE Transaction on Magnetism*, vol 33, no2, March 1997, pp.1856-1859.

[SabCo86] SABONNADIÈRE J.C. et COULOMB J.L., *Eléments finis et CAO, Hermès,* Paris 1986.

[SalCFS91] SALDANNHA R.R., COULOMB J.L., Fogia A. and SABONNADIÈRE J.C., "A dual method for contrained optimization design in magnetostatic problems", *IEEE Transaction on Magnetism*, vol 27, no5, September 1991, pp.4136-4139.

[Salda92] SALDANHA R.R., Optimisation en électromagnétisme par application conjointe des méthodes de programmation non linéaire et de la méthode des éléments finis, *thèse de doctorat de l'INPG,* Grenoble 1992, 182p.

[Salud97] SALUDJIAN Lucas, Optimisation en électrotechnique par Algorithmes Génétiques, *thèse de doctorat de l'INPG,* Grenoble 1997, 169p.

[SilFe83] SILVESTER P.P., FERRARI R.L., *Finite Elements for Electrical Engineers, Cambridge University Press*, 1983, (2nd edition, 1990).

[StoBu94] STOER J., BULIRSCH R., "*Introduction to Numerical Analysis*", *Springer-Verlag*, New York 1994.

[TEAM22] Team Workshop Problem 22, SMES Optimization Benchmark,

http://www-igte.tu-graz.ac.at/team

[Terra85] TERRAIL Y.D., Modélisation géométrique et topologique en 3D pour l'application de la méthode élements finis en électromagnétique, *thèse de doctorat de l'INPG,* Grenoble 1985.

[Vande84] VANDERPLAATS G.N., *Numerical optimization techniques for engineering design : with applications, McGraw Hill*, 1984.

[VilWe97] VILLENEUVE D. and WEBB J.P., "Universal Matrices for High Order Finite Elements in Nolinear Magnetic Field Problems", *IEEE Transaction on Magnetism*, vol 33, no5, Sept. 1997, pp.4131-4133.

[Webb93] WEBB J.P., "Edge elements and What They can do for You", *IEEE Transaction on Magnetism*, vol 29, no2, March 1993, pp.1460-1463.

[Weebe92] WEEBER Konrad, Paramétrisation, efficacité et formulations pour l'optimisation de la forme de dispositifs électromagnétiques avec des éléments finis, *thèse de doctorat de l'INPG,* Grenoble 1992, 200p.

WEEBER Konrad, Parametrization, Efficiency and formulations for the shape optimisation of electromagnetic device with finite elements, *Doctorate thesis of INPG*, Grenoble 1992, 200p.

[WeHo92a] WEEBER K., HOOLE S.H.R. Ratnajeevan S., "The subregion method in magnetic field analysis and design optimisation", *IEEE Transaction on Magnetics*, vol 28, no2, 1992, pp.1561-1564.

[Yoda95] YODA K., "Direct Design Optimization Using Approximate Algebra", *IEEE Transaction on Magnetism*, vol 31, no3, May 1995, pp.1952-1955.

[Zgain96] ZGAINSKI François-Xavier, Un pré-processeur pour l'éléctromagnétisme, électro-mécanique et l'électro-acoustique, *thèse de doctorat de l'INPG,*

Grenoble 1996, 217p.

[ZieWo89] ZIENKIEWICZ O.C., WOOD W.L., *The finite element method, Mc Graw Hill*, New York 1989.

Informatique

[ADIFOR] The ADIFOR 2.0 System for the Automatic Differentiation of Fortran 77 programs, *Center for Rechearch on Parallel Computation, Rice University, USA.*

http://www.mcs.anl.gov/

[ADOC] ADOC Aide d'utilisation, 1996, *CADOE S.A.*, Villeubanne, France.

[ADOL-C] GRIEWANK A., JUEDES D., UTKE J., A Package for the Automatic Differentiation C++, Sept. 1996,

http://ww.math.tu-dresden.de/~adol-c

[ADOMESH] ADOMESH Aide d'utilisation, 1996, *CADOE S.A.*, Villeubanne, France

[ASA] INGBER Lester, Adaptive simulated anneling, *Lester Ingber Research*, McLean, USA.

http://www.ingber.com/

[C-HP] *HP-C/HP-UX Reference Manual HP 9000 Serie 700/800 Computer, Hewlet Packard, 1993.*

[Coul95i] COULOMB J.L., Le Langage HE V.2, Août 1995, Note interne, LEG-ENSIEG/INPG, France

[F2D] Flux2D Notice d'utilisation, version 7.1, 1994, *CEDRAT S.A., France et Magsoft Co., USA.*

[F2Dp] Analyse paramétrée des dispositifs électriques et magnétiques, Flux_Param version 1.00, Mode d'emploi, Fevrier 1996, *CEDRAT-Recherche*, ZIRST 4301 - Meylan, France

[F3D] Flux3D Notice d'utilisation, version 2.3, 1994, *CEDRAT S.A., France et Magsoft Co., USA.*

[F77-HP] FORTRAN Domain Programmation - Reference Manual, *Hewlet Packard, 1991.*

[NURBS] HEWITT W.T. and YIP D., The NURBS procedures library, *Manchester Computing Centre*, 1992.

Annexe A

Les commandes de simulations

Nous décrivons dans cette annexes les commandes de la procédure d'assemblage (chapitre 4). Les principales syntaxes sont données donc pour mieux comprendre la signification de chaque commande.

Notons que les commentaires sont situés sur la ligne de commande après le signe "//".

Commandes du précalcul des intégrales

CFF	Assignation des coefficients pour l'opération SUM (Déactivés lorsque GS est active)
CST	Multiplication par un constant **CST** opérand cst
FIN	Terminer **FIN** **ASM** // Lancer l'assemblage
GS	Opérations reliées à l'intégration de Gauss #GS : Initialiser (recupérer) les coefficients (poids, valeurs et gradients de fonctionsde forme) pour l'opération SUM **#GS** Nbp Nbf // Nombre des points et Nombre des fonctions de forme GSS : Activer la procédure (les coefficients) **GSS** idn // Un code pour boucler sur différentes parties des coefficients GS# : Retourner au paquet des coefficients "normaux"
IF	Les tests #IF : Déterminer la condition **#IF** id p1 [p2 p3 ...] // id est un type d'opération logique, p_j, les données IF@ : Signaler le branchement **IF@** 1 // Branchement si verifié **IF@** 0 // Branchement si non vérifié **IF#** // Le point de fermeture du IF@

INI	INI : Déclarer les identificateurs du tableau d'entrée **INI** 2 3 // Pour {x1,y1},{x2,y2},{x3,y3} **INI** 3 2 u // Pour {u1,v1,w1},{u2,v2,w2}
INV	Inversion d'un polynôme **INV** opd
MEM	Compactage de la mémoire, tableau de résultats **MEM** 0 x // Remplacer le tableau d'entrée **MEM** n1 n2 // Suprimer les élements entre n1 et n2
MLT	Multiplication de deux polynômes **MLT** opd1 opd2
PBL	Déclaration du type d'élément et la formation associée **PBL** Elem. Form. **PBL** Elem. ALL // Défault pour toutes les formulation
SUM	Addition multi-modale des polynômes **SUM** opd1 opd2 [...]

Commandes de l'assemblage récursif

ADD	Addition (vectorielle ou matricielle) **ADD** opd1 opd2 // Résultat est stocké dans opd1
CFF	Assigner un nombre à un coefficient pour l'usage général
CFS	Affecter les coefficients à un vecteur ou à une matrice **CFS** Xx cff // Xx(k)= Xx(k)*cff(k)
DRV	Assignation spécifique à partir du tableau des polynômes **DRV** Xx n1 n2 [...] // Xx est un vecteur ou une matrice
IF	**IF@** 0 // Commencer la partie d'assemblage, dit initial (une fois) // La dérivée du vecteur source **IF#** // Le point de "FinSi"
MLT	Multiplication **MLT** opd1 opd2 // Résultat dans opd1 **MLT** opd1 opd2 opd3 // Résultat dans opd3
PBL	Déclaration de la formation associée **PBL** Form.
SET	Assignation directe d'un vecteur ou d'une matrice **SET** Xx cst // Xx(k)=cst **SET** Xx Yy // Xx(k)=Yy(k)

ANNEXES

Annexe B

Le TEAM WorkShop Problème n°22

B.1. Le SMES

Les SMES (Superconducting Magnetic Energy Storage) sont des dispositifs qui stockent de grandes quantités d'énergie sous forme magnétique. Ils sont constitués de matériaux supraconducteurs qui permettent d'avoir une résistance électrique nulle moyennant de très basses températures. Schématiquement, le SMES peut être vu comme une bobine supraconductrice attachée à un dispositif qui charge et décharge la bobine (Fig. B.1).

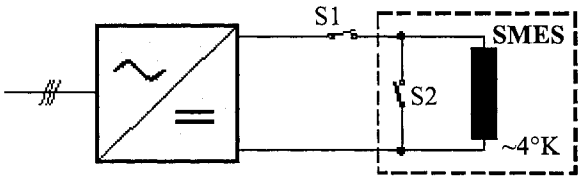


Fig.B.1. Schéma de principe d'un SMES

La bobine se charge lorsque l'interrupteur S1 est fermé. Une fois la bobine chargée, on ferme S2 puis on ouvre l'interrupteur S1 : le courant circule alors dans la bobine sans aucune perte puisque le SMES est à l'état supraconducteur. Le champ magnétique créé alors par le courant permet de conserver directement de l'énergie sous forme magnétique sans aucune conversion. Dans les années 80, de nombreux projets ont vu le jour, impliquant l'utilisation de SMES constitués de bobines de plusieurs centaines de mètres de diamètre (Fig. B.6) mais aujourd'hui les recherches sont plutôt axées sur des SMES de dimensions "raisonnables".

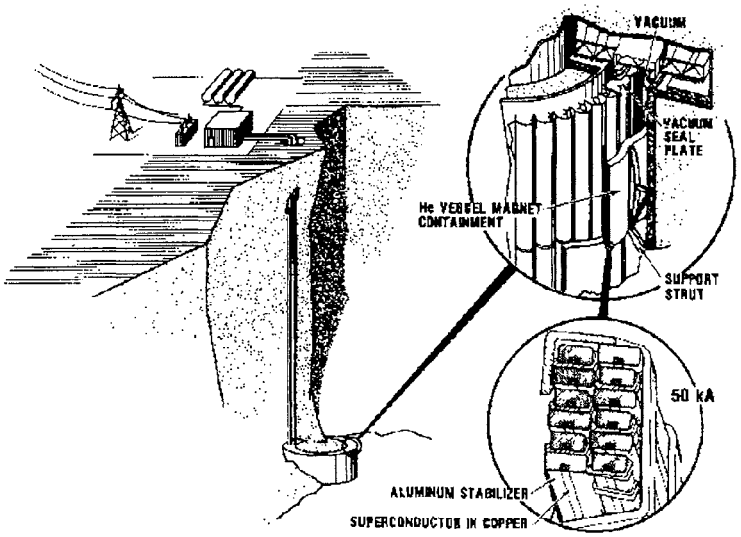


Fig.B.2. *Vision d'artiste pour un projet de SMES de 5000 MWh*

Deux types de bobines peuvent être utilisées dans les SMES: le solénoïde et le tore (Fig. B.3). L'intérêt d'une bobine toroïdale est que le champ magnétique en dehors du dispositif est nul et même si ceci n'est valable que dans le cas d'un enroulement parfait, il semble que cela soit une solution plus prometteuse que le solénoïde. Une autre solution pour diminuer le champ à l'extérieur du dispositif consiste à utiliser deux solénoïdes parcourus par des courants de sens opposés. Cette dernière solution a été retenu dans le problème TEAM workshop *problem 22.*

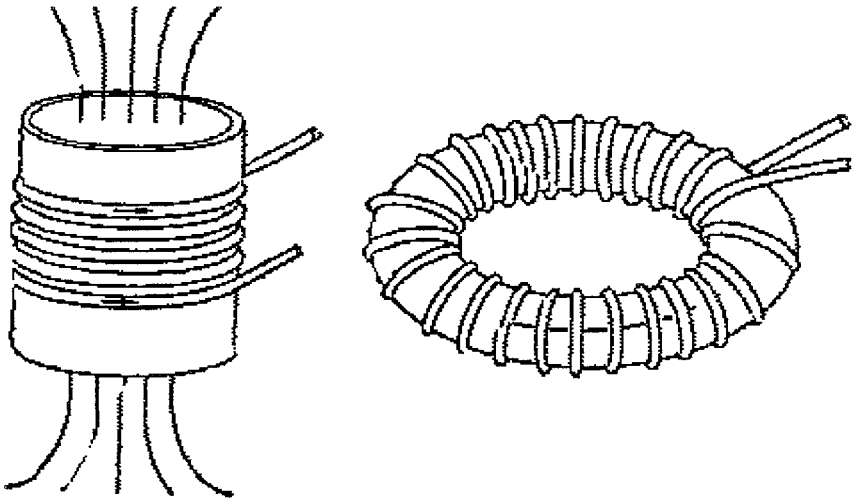


Fig.B.3. Bobines solénoïdale et toroïdale pour le SMES

B.2. La description du problème

Le problème TEAM workshop n°22 [TEAM22] est constitué de deux bobines supraconductrices et possède 8 degrés de liberté (Fig. B.8). L'optimisation consiste à trouver les "bonnes valeurs" de paramètres géométriques (R_1, R_2, h_1, h_2, d_1, d_2) et physiques (J_1, J_2) de façon à satisfaire les deux objectifs suivants:

◊ L'énergie stockée sous forme magnétique dans le dispositif doit être égale à 180MJ

◊ L'induction, le long de deux lignes situées à 10 mètres du dispositif (ligne **a** et **b**), doit être la plus faible possible.

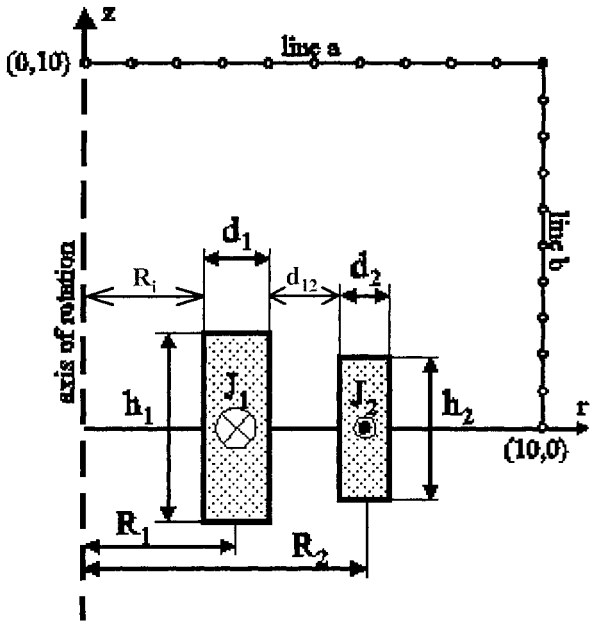


Fig.B.4. La géométrie et les paramètres du SMES

Table B.1: Contraintes de domaines sur les paramètres du SMES.

	$R_1 \rightarrow R_i$ [m](*)		$R_2 \rightarrow d_{12}$ [m](*)		$h_1/2$ [m]	$h_2/2$ [m]	d_1 [m]	d_2 [m]	J_1 [MA/m2]	J_2 [MA/m2]
min.	1.0	0.6	1.8	0	0.1	0.1	0.1	0.1	10.0	-30.0
max.	4.0	3.95	5.0	3.9	1.8	1.8	0.8	0.8	30.0	-10.0

(*) $R_i=R_1-d_1/2$ et $d_{12}=(R_2-R_1)-(d_1+d_2)/2$

Les domaines de variations des paramètres sont présentés dans la table B.1. Deux nouveaux paramètres R_i et d12 remplacent R_1 et R_2 afin d'assurer que les deux bobines ne se chevaucheront pas. Pour conserver le caractère supraconducteur des bobines il est nécessaire que les grandeurs densité de courant, température et induction soient inférieures à certaines valeurs critiques. Celles-ci sont liées entre elles et forment une surface critique dans l'espace induction, température et densité de courant. Ainsi, dans le processus d'optimisation cette condition ("quench condition" en anglais) se traduira, pour chaque bobine, par une contrainte sur la densité de courant et la valeur maximum de l'induction à l'intérieur de la bobine. Cette courbe "critique" est en général approximée par une droite au dessus de laquelle le matériau perd ses propriétés supraconductrices (Fig. B.5). Cette contrainte physique sur les bobine peut être approximée par l'inégalité: $|\mathbf{J}| \leq (6.4\,|\mathbf{B}| + 54.0)$ A/mm^2

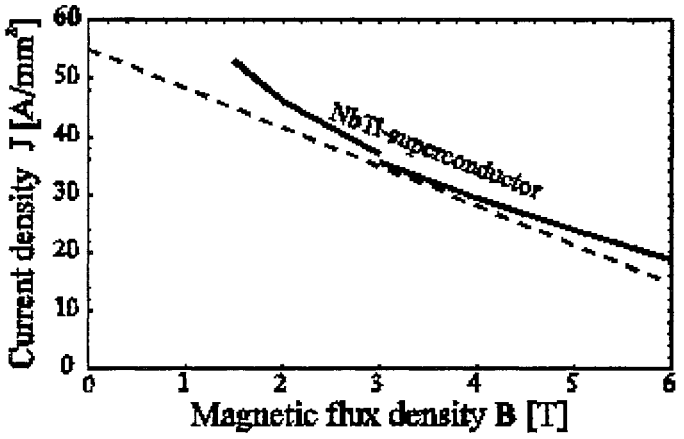


Fig.B.5. Courbe critique d'un supraconducteur industriel

Problème d'optimisation:

La fonction objective se calcule par l'équation: $OF = \dfrac{B^2_{stray}}{B^2_{norm}} + \dfrac{|Energy - E_{ref}|}{E_{ref}}$

avec $\mathbf{E}_{ref}$=180 [MJ], $\mathbf{B}_{norm}$= 2.0E-4 [T] et $\mathbf{B}^2_{stray} = \dfrac{1}{22}\sum_{i=1}^{22} \mathbf{B}^2_{stray\,i}$

où $\mathbf{B}_{stray_i}$ désigne l'induction évaluée en 22 points répartis uniformément sur les lignes **a** et **b** (Fig.B.4). Ce problème est particulièrement intéressant dans le cadre de l'optimisation stochastique puisqu'il possède de nombreux optima locaux [TEAM22, Salud97].

Résumé — Au cours de ce travail, une implantation de la méthode d'analyse de sensibilité d'ordre élevé a été réalisée au sein de la méthode des éléments finis (MEF). Tout d'abord nous avons posé la base théorique des développements: la MEF classique et surtout la méthode de dérivées d'ordre élevé qui engendre les résultats sous la forme de développements de Taylor par rapport aux paramètres de conception. En s'appuyant sur la structure typique d'un code de calculs EF, nous avons développé ensuite les modules d'un calcul "paramétré": la paramétrisation géométrique, la dérivation du maillage et la résolution paramétrée. L'originalité de cette implantation consiste dans une nouvelle organisation des calculs de dérivées qui sont basés notamment sur les opérations symboliques. Pour valider des étapes de calculs, plusieurs exemples ont été présentés. Enfin, les applications possibles des résultats paramétrés sont évoquées, entre autres, une procédure pour définir le modèle analytique équivalent de dispositifs électromagnétiques a été proposée.

Notons que plusieurs problèmes de réalisation informatique ont été mise en évidence et résolus, ce qui représente un gros investissement en programmation. Des nombreuses perspectives sont ouvertes par cette approche, le travail mériterait donc d'être poursuivi.

Title: **Parametrized FE Analysis Based on High Order Derivative, Applications in Electromagnetic**

Abstract — In this work, a procedure of high order sensitivity analysis was implemented within the finite element method (FEM). Firstly, we presented the fundamental of this approach: the classical FEM and the essential high order derivative method which emulates the results in form of the Taylor's developments that report to the sensitivity parameters. Next, basing on a typical FE analysis, we have developed the modules of a parametrized analysis: the geometric parametrization, the mesh derivation and the parametrized solving stage. The originality of this implementation consists firstly of using the symbolic operations in a new organisation of the derivatives calculus. A number of the examples was done for validating all the created tools. In the end, the possible applications of the parametrized results were evoked; among those, we propose the procedure that will allow us to define the equivalent analytical model of electromagnetic device.

The numerous problems related to the software achievement are taken on evidence and solved, which represent the heavy investment in programming. Several clear perspectives are opened, however, by this approach and the pursuits of the developments will achieve their merit.

Mots clés français	Index terms (* IEEE'98 list not included)
Méthode des éléments finis	Finite Element Method
Analyse de sensibilité	Sensitivity Analysis*
Dérivée d'ordre élevé	High order Derivative*
Programmation	Programming
Calcul symbolique	Symbolic computation*
Paramétrisation	Parametrization
Géométrique modèle, Maillage	Geometric model, Mesh
Langage de Commande	Command language
Électromagnétisme	Electromagnetism
Approximation polynomiale	Polynomial Approximation*