Une vision faiblement intrusive de la méthode LATIN-PGD en non-linéaire

Ronan Scanff

université PARIS-SACLAY

Une vision faiblement intrusive de la méthode LATIN-PGD en non-linéaire

A weakly-invasive LATIN-PGD method in the non-linear content

Thèse de doctorat de l'Université Paris-Saclay

École doctorale n° 579 : Sciences Mécaniques et Énergétiques, Matériaux et Géosciences (SMEMaG)
Spécialité de doctorat : Mécanique des solides
Graduate School : Sciences de l'ingénierie et des systèmes
Référent : ENS Paris-Saclay

Thèse préparée au sein de l'unité de recherche LMPS - Laboratoire de Mécanique Paris-Saclay (Université Paris-Saclay, CentraleSupélec, ENS Paris-Saclay, CNRS), sous la direction de **David Néron**, Professeur des Universités (ENS Paris-Saclay), la co-direction de **Pierre Ladevèze**, Professeur Émérite (ENS Paris-Saclay), et la co-supervision de **Philippe Barabinot**, Research Engineering Senior Manager (Siemens)

Thèse soutenue à Paris-Saclay, le 22 Mars 2022, par

Ronan Scanff

Composition du jury

Francisco Chinesta Professeur des Universités, ENSAM Paris	Président
David Dureisseix Professeur des Universités, INSA Lyon	Rapporteur
Pierre Kerfriden Assistant Professor (HDR), Mines ParisTech & Senior Lecturer, Cardiff University	Rapporteur
Marianne Beringhier Maîtresse de Conférences, Université de Poitiers - ISAE ENSMA	Examinatrice
Christian Rey Head of Computational Applied Mathematics, Safran	Examinateur
David Néron Professeur des Universités, ENS Paris-Saclay	Directeur de thèse

THÈSE DE DOCTORAT
NNT : 2022UPAST023

Remerciements

Après trois années riches et passionnantes, une page se tourne. J'en garderai, non pas sans une certaine nostalgie, de très bons souvenirs.

J'adresse tout d'abord mes remerciements à David Dureisseix et Pierre Kerfriden d'avoir accepté d'être mes rapporteurs et de m'avoir livré leurs remarques judicieuses à la relecture attentive du manuscrit. Je remercie plus généralement Francisco Chinesta, Marianne Beringhier et Christian Rey d'avoir accepté de prendre part à mon jury. Merci pour votre disponibilité à tous. Je suis honoré de l'attention que vous avez porté à l'égard de mon travail.

Mes remerciements vont ensuite à mes deux directeurs de thèse. Merci David de m'avoir mis le pied à l'étrier, pour ton accompagnement, ton sens de la pédagogie et ta bonne humeur sans limite. Merci Pierre pour tes idées, ta vision de la mécanique, tes conseils avisés et nos nombreuses discussions qui m'ont beaucoup appris. Merci à tous les deux de m'avoir fait confiance au lancement de ce projet.

Je tiens aussi à remercier l'ensemble des membres du laboratoire, qui lui confèrent cette atmosphère si chaleureuse et plaisante. Merci plus spéciquement à Clotilde, avec qui j'ai partagé ces 3 années, fidèle compagnon de la première heure, et fidèle pilier du café matinal, propice moment pour échanger sur nos thèses ou nos vies à côté. Merci à Agathe, avec qui nous formions – avec Clotilde – l'équipe bar : j'en garde de très bon souvenirs que ce soit pour l'organisation du repas de Noël, pour les courses de bûchettes et de café ou encore pour essayer de faire comprendre que la vaisselle ne peut pas se faire toute seule. Merci à mes autres compagnons de cordée, Jérémie, Guillaume mais aussi Willy, pour ton humour et ta force intérieure, et avec qui j'ai partagé plus d'un trajet commun jusqu'à cette contrée maintenant lointaine qu'est le plateau de Saclay. Merci également à la team ROMlab, Alexandre, Aurélia, Floriane et Thomas qui continuent de faire vivre ce magnifique code au quotidien. Merci Stéphane pour m'avoir transmis les prémices de ROMlab, et pour nos discussions interminables au sujet de la LATIN-PGD, quitte à finir par exaspérer nos amis lors de certaines soirées communes. Travailler avec toi a toujours été un réel bonheur. Merci plus généralement à tous les (ex-)doctorants, Marie, Richard, Renaud, Philippe, Florian, Xuyang, Maxence, Léonard, Achraf, Myriam, Matthieu, Héloïse, Julien, Claire, Roxane, Pascale, Aya, Ahmed, Perla, Livio,... Échanger avec chacun d'entre vous, au détour d'un couloir, au Crous, ou pendant une pause café, a toujours été un plaisir et une source d'enrichissement. Merci Philippe et Pierre de vous occuper aussi bien du CdC, en étant toujours présents pour répondre à la moindre question informatique. Merci PAB d'avoir toujours été là pour répondre à mes questions, plus ou moins tordues, et ce même bien avant la thèse. Merci Manu, Fédérica, Amélie et Karine pour vos conseils avisés sur ma présentation. Enfin, un grand merci à l'ensemble du CdG pour votre professionnalisme à toute épreuve qui fait que le laboratoire tourne comme une horloge. Merci Lydia d'avoir toujours une solution à tout, ton bureau adossé aux stations de travail me manque aussi. Si c'est aussi agréable de venir travailler au labo, dans ce cadre de travail idéal, c'est grâce à vous tous.

Merci également à l'ensemble de mes collègues Siemens que ce soit à Châtillon, Philippe, pour ton encadrement, ton soutien et les nombreux échanges que nous avons pu avoir lors des aller-retours en Belgique, Pascal, pour les incalculables parties de plongée sous-marine dans les méandres du code source, Bruno pour ta vision industrielle et pragmatique de Samcef, Anne-Charlotte pour ta passion de l'aviation à l'origine de multiples discussions ; ou à Liège, Frédéric, Jean-Pierre et Cédric pour le partage de vos innombrables connaissances sur le code.

Merci à la fameuse équipe jap', Maxime, Stéphane, Margaux, Clotilde, Jocelyn, Yassin et Louis, toujours partants pour un passer un bon moment ensemble, que ce soit autour d'un jeu, d'un verre ou d'une poutine mais pas que ! Tellement (trop) de choses pourraient être racontées à ce sujet, à commencer par la très belle semaine à Lacanau. Merci aussi à Julie, Wiwi et Eduardo mes amis « matheux » et compagnons de running en toutes circonstances. Merci Gabriel, mon plus vieil ami – littéralement –, d'être toujours présent aux moments importants.

Enfin, merci à ma famille, mes parents pour leur soutien toujours sans faille. Merci de m'avoir appris à trouver toutes les billes pour réussir. Une pensée aussi pour ma mamie, et mon papi, qui continue son chemin malgré l'adversité de ces dernières années.

Et pour finir, **merci** Chloé pour ton aide et ton sens de la logistique hors norme. Merci d'avoir accepté de partager ce chemin à mes côtés. Merci pour ce que tu es au quotidien tout simplement. À mon tour maintenant de te soutenir dans ta dernière ligne droite comme tu l'as si bien fait !

Table des matières

Introduction

La simulation numérique constitue l'un des piliers de l'industrie 4.0 dans le cadre de la transformation digitale des entreprises. Elle permet en effet aux industriels d'optimiser leurs processus de conception et de proposer des produits de meilleure qualité tout en réduisant leur temps de mise sur le marché, ce qui représente un atout indéniable de compétitivité. Au cœur des nouvelles pratiques, sont apparus, ces dernières années, les jumeaux numériques – Digital Twin – qui font l'objet d'un engouement prononcé. Ces derniers peuvent intervenir à toute étape du cycle de vie : pour particulariser les plans de maintenance prédictive, pour contrôler la qualité ou bien faciliter les étapes d'intégration lors de la fabrication, ou encore pour valider la conception de nouveaux produits dont les essais physiques restent compliqués voire impossibles à mettre en oeuvre. C'est le cas par exemple de l'hélicoptère Ingenuity (Figure 1) qui réalise actuellement ses missions sur la planète Mars. Avec une composition d'atmosphère radicalement différente, une pression atmosphérique moyenne de l'ordre de 600 Pa seulement et une plus faible gravité – de l'ordre de 38% de celle sur Terre –, les conditions de vol sur Mars sont complètement différentes de celles rencontrées sur Terre. Le Jet Propulsion Laboratory de la NASA a ainsi effectué des centaines de simulations afin de pouvoir prédire les conditions de vol sur Mars [Koning *et al.*, 2018].

FIGURE 1 – Modèle de l'hélicoptère Ingenuity pour la mission sur la planète Mars – Crédits NASA, 2021

Bien évidemment, l'avénement de ces nouveaux outils numériques coïncide avec la volonté de représenter de mieux en mieux la réalité physique, ce qui passe par la résolution de problèmes d'une complexité croissante. Ces problèmes bien souvent non-linéaires, multi-physiques, comprenant plusieurs millions de degrés de liberté sur des échelles de temps toujours plus fines, peuvent demander plusieurs heures voire jours de calculs. La Figure 2 illustre la simulation d'une valve aortique constituée de trois cuspides [1]. Cette simulation, réalisable avec le logiciel Simcenter STAR-CCM+™ qui modélise les interactions fluide-structure, est l'exemple même d'un problème fortement non-linéaire : les cuspides sont des structures extrêmement fines et flexibles, généralement associées à un comportement matériau hyper-élastique incompressible, qui, sous l'action du flux sanguin, se déforment pour permettre l'ouverture ou la fermeture de la valve. À ces non-linéarités matérielles s'ajoutent donc des non-linéarités géométriques fortes liées

1. terme dédié pour désigner les composantes de la valve cardiaque

aux grandes transformations, et où des processus d'instabilité de type *snap-through* entrent en jeu, mais également des non-linéarités de contact entre les différents cuspides lors des phases de refermeture de la valve, permettant ainsi d'empêcher le retour du fluide. Qui plus est, la dynamique relativement rapide des mouvements lors des phases d'instabilité impose d'avoir une discrétisation temporelle extrêmement fine – jusqu'à 10^{-5} seconde, pour un temps total de simulation de l'ordre d'une seconde – complexifiant davantage encore la résolution.

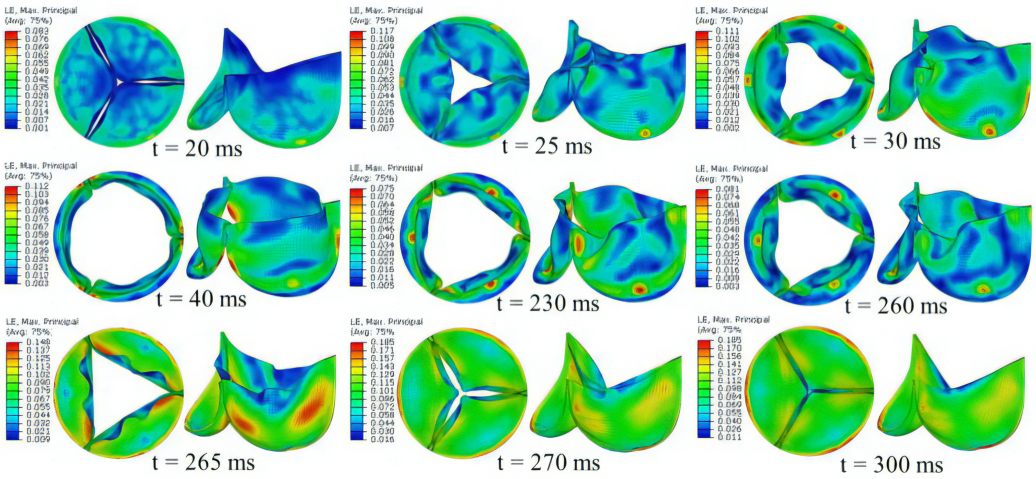


FIGURE 2 – Simulation du fonctionnement d'une valve aortique au cours d'un cycle – ouverture ($20\text{ ms} \leqslant t < 40\text{ ms}$), maintien ($40\text{ ms} \leqslant t \leqslant 230\text{ ms}$), refermeture ($230\text{ ms} < t \leqslant 300\text{ ms}$)

Ces simulations numériques non-linéaires déjà coûteuses le deviennent d'autant plus qu'elles sont généralement assorties d'une exploration d'espaces paramétriques de grandes dimensions durant les processus d'optimisation, se heurtant alors rapidement aux limites actuelles en matière de simulation. Ainsi, malgré des progrès fulgurants en termes de puissance de calcul – permettant notamment l'utilisation de *clusters* à architecture parallèle combinant plusieurs centaines voire plusieurs milliers de cœurs –, cette démultiplication des temps de calcul pousse dans le même temps à développer de nouvelles stratégies numériques de résolution plus performantes. De nombreuses méthodes ont été élaborées afin de mener une exploration plus intelligente des espaces paramétriques comme la méthode des plans d'expérience [Montgomery, 2017], basée sur des outils statistiques, ou encore les nombreuses méthodes d'optimisation [Nocedal et Wright, 2006] comme par exemple les méthodes à gradients.

De manière alternative, les méthodes de réduction de modèles (ROM) [Benner *et al.*, 2021] s'avèrent particulièrement efficaces pour la résolution d'équations aux dérivées partielles paramétrées et présentent donc un potentiel énorme pour le calcul hautes performances. Fondées sur l'idée que la complexité apparente du problème de grande dimension peut bien souvent se résumer à rechercher la solution dans un espace de dimension beaucoup plus petite avec une perte d'information négligeable, celles-ci visent à construire des approximations de rang faible à variables séparées en exploitant la redondance de l'information. Plusieurs voies ont été explorées pour construire de tels modèles réduits. Les méthodes Proper Orthogonal Decomposition (POD) [Chatterjee, 2000] et Reduced-Basis (RB) [Maday et Rønquist, 2002] se caractérisent par une première phase d'apprentissage issue d'une poignée de résolutions hautes fidélités. La méthode Proper Generalized Decomposition (PGD) [Chinesta *et al.*, 2014] propose quant à elle une démarche différente en venant directement construire à la volée l'approximation de rang faible par un processus de minimisation. Une fois construits, ces modèles réduits permettent d'évaluer un très grand nombre de scénarios en temps quasi-réel ce qui en fait de puissants outils d'aide à la décision.

La communauté autour des techniques de réduction de modèles est une communauté très active comme en témoigne le GdR AMORE[2] regroupant plus de 150 chercheurs répartis au sein de 22 laboratoires de recherche. Ce groupe de recherche à la croisée des chemins entre les mathématiques fondamentales, les sciences pour l'ingénieur et les techniques numériques avancées s'intéresse à diverses thématiques comme les modèles multi-paramétriques et non-linéaires, les modèles basés sur les données, les problèmes de certification, de vérification ou de validation, tout comme les formulations non-intrusives et le déploiement des méthodes ROM dans l'industrie. Dans ce dernier cas, on ne peut que constater le fait que les méthodes ROM restent encore bien souvent cantonnées au monde académique, ce qui motive les travaux présentés ici.

Pour devenir accessibles plus largement à l'ensemble de la communauté scientifique, ces méthodes doivent pouvoir se conformer aux habitudes d'utilisateurs experts (ou non) au sein des bureaux d'études, et donc être judicieusement intégrées dans les logiciels industriels. L'un des principaux verrous réside dans le caractère particulièrement intrusif de ces méthodes qui nécessitent des opérations et un format de stockage de données non conventionnels des logiciels industriels. De nombreux efforts ont déjà portés sur le développement de méthodes ROM non-intrusives. Dans un premier temps, ces efforts ont été principalement orientés sur des approches *a posteriori* POD ou RB [Audouze *et al.*, 2013, Casenave *et al.*, 2014]. Ce n'est que plus récemment que des approches PGD non-intrusives sont apparues [Ibáñez *et al.*, 2018, Borzacchiello *et al.*, 2019], toujours basées sur l'utilisation de *snapshots*, du fait de la formulation *a priori* – sous-jacente à la PGD – intrinsèquement plus intrusive. En parallèle, d'autres implémentations non-intrusives de la PGD ont été dédiées à des cas spécifiques d'applications [Courard *et al.*, 2016, Zou *et al.*, 2018, Leon *et al.*, 2019] et reposent sur le couplage en externe de logiciels industriels avec des codes maison développés en laboratoire, généralement sous Matlab® ou bien Python™ – à défaut d'accès aux codes sources des logiciels industriels.

L'idée d'introduire directement la PGD au sein même des logiciels industriels devient une question d'actualité, afin de ne plus être lié à une application spécifique, mais de disposer d'un cadre général valable pour tout type de non-linéarité, et d'ainsi dépasser les possibilités des codes prototypes développés en laboratoire. Les efforts de couplage sont alors portés en interne, au plus proche du logiciel industriel, quitte à être légèrement intrusif – dans le sens où un fort niveau d'expertise, et de connaissance du logiciel industriel, est requis pour mener à bien la tâche. Cette faible intrusivité est à mettre en regard de la robustesse qui demeure un point fondamental pour que les méthodes ROM fassent partie des outils des ingénieurs au quotidien. Dans ce contexte, la méthode LATIN-PGD [Ladevèze, 1985, Ladevèze, 1999] présente de nombreux atouts. En permettant de coupler l'utilisation de la PGD avec la résolution de problèmes mécaniques non-linéaires paramétrés dépendant du temps, la méthode LATIN-PGD permet d'aboutir à une réduction appréciable des temps de calcul [Relun *et al.*, 2013, Néron *et al.*, 2015], de construire efficacement des abaques virtuels [Courard *et al.*, 2016], mais également d'adresser tous les types de non-linéarités [Abdali *et al.*, 1996, Oumaziz *et al.*, 2017]. Ces avantages, ainsi que les nombreux travaux qui lui ont été consacrés, rendent la méthode attractive pour Siemens Digital Industries Software qui souhaite donc la voir intégrée dans son code par éléments finis Simcenter Samcef™ – faisant partie de la suite logiciel Simcenter 3D™. La méthode LATIN-PGD sera alors un outil supplémentaire pour la gestion des jumeaux numériques industriels [Chinesta *et al.*, 2020, Hartmann et van der Auweraer, 2020], et plus précisément pour la réalisation efficace de simulations ultra-hautes fidélités. Le volet données, autre composante fondamentale du jumeau numérique [Glaessgen et Stargel, 2012], ne sera pas abordé dans le cadre de ces travaux.

L'objectif de ces travaux de thèse est de proposer une nouvelle formulation faiblement intrusive de la méthode LATIN-PGD, au sens où elle puisse être aisément implémentable dans un logiciel éléments finis

2. GdR AMORE : https://insis.cnrs.fr/fr/les-groupements-de-recherche

industriel tel que Simcenter Samcef™. Mener des actions conjointes avec un éditeur de logiciel confère davantage de flexibilité quant à la possibilité d'intégration de nouvelles méthodes numériques mais de multiples contraintes persistent néanmoins. Avoir accès au cœur du logiciel – au code source – permet d'introduire les interfaces de couplage en interne, et d'ainsi assurer performance et robustesse. Le point clé est toutefois de ne pas modifier l'architecture globale du logiciel hôte, pour conserver la capitalisation des développements informatiques menés ces cinquante dernières années. Ces contraintes ont conduit au développement d'une variante faiblement intrusive de la méthode LATIN-PGD, qui diffère de la variante la plus couramment utilisée ces dernières années, basée sur la description à variables internes des matériaux [Cognard et Ladevèze, 1993, Ladevèze, 1999, Bhattacharyya *et al.*, 2018]. Elle peut toutefois être vue comme une extension des premières variantes fonctionnelles historiques [Ladevèze, 1985, Boisse *et al.*, 1990, Bussy *et al.*, 1990, Cognard et Ladevèze, 1991]. Alors que jusqu'à présent, la méthode LATIN-PGD a été majoritairement utilisée pour résoudre des applications spécifiques, l'approche abordée dans le cadre de ces travaux montre les possibilités offertes par la méthode avec un logiciel industriel sans qu'aucune application particulière ne soit visée. Cette approche générale est guidée par trois éléments principaux : la robustesse, la performance et la facilité d'utilisation vis-à-vis des utilisateurs. Ce travail est mené en collaboration avec Siemens Digital Industries Software qui finance ces travaux par l'intermédiaire d'une convention CIFRE décernée par l'ANRT (2018/0611).

Dans la suite du manuscrit, on détaille la méthodologie qui a été mise en place. Le manuscrit s'articule en trois grandes parties :

— La première partie s'attache à présenter des éléments bibliographiques portant sur la résolution de problèmes mécaniques non-linéaires au moyen de méthodes de réduction de modèles.

- Le Chapitre-1 fixe les différentes notations utilisées et précise le cadre de ces travaux en prenant soin de fixer les éventuelles limites. Bien que notre approche puisse *a priori* s'appliquer à tout type de non-linéarité – matérielle ou géométrique –, nous retenons plus spécifiquement le cas des non-linéarités dépendantes du temps sous hypothèses de quasi-staticité et de déplacements « modérés ». En particulier, les phénomènes d'instabilité et de flambage en grandes transformations ne sont pas considérés pour l'instant. La formulation du problème de référence étant écrite, nous rappelons ensuite brièvement la méthode Newton-Raphson ainsi que ses différentes variantes possibles [Zienkiewicz et Taylor, 2005]. C'est le solveur non-linéaire standard habituellement employé dans les logiciels éléments finis industriels. Une alternative réside dans l'application de la méthode LATIN-PGD [Ladevèze, 1985], une méthode de résolution non-linéaire non-incrémentale dont nous exposons les grands principes.
- Le Chapitre-2 commence par évoquer les spécificités des différentes méthodes ROM [Benner *et al.*, 2021] usuellement rencontrées en mécanique, avant de mettre l'accent sur le traitement des non-linéarités. Les méthodes ROM s'appliquent notamment très bien aux problèmes linéaires mais dès que des non-linéarités apparaissent, il faut trouver un moyen pour linéariser. Le solveur non-linéaire LATIN est une proposition – parmi d'autres – nativement conçue pour fonctionner avec la PGD où la variable temps joue alors un rôle particulier.
- Le Chapitre-3 recense les différents points de blocage qui freinent le développement des méthodes ROM dans l'industrie. Outre les opérations atypiques à réaliser, qui ne sont pas prévues en standard dans les codes éléments finis industriels, on est également amené à manipuler des quantités non-conventionnelles – sur l'espace, le temps et éventuellement d'autres paramètres. Ceci constitue le principal frein ayant motivé l'élaboration de méthodes ROM non-intrusives [Giraldi *et al.*, 2014]. Généralement, ces dernières se basent soit sur des approches de type *snapshots* [Ibáñez *et al.*, 2018, Borzacchiello *et al.*, 2019, Casenave *et al.*, 2020b] soit sur des stratégies de couplage logiciel [Courard *et al.*, 2016, Zou *et al.*, 2018, Tsiolakis *et al.*, 2020] – en externe. Nous proposons d'aller plus loin en implantant directement la PGD au sein d'un

logiciel industriel – en interne – par l'introduction du solveur LATIN non-incrémental dans une architecture incrémentale. Ce sera le seul point intrusif de notre approche.

— La deuxième partie se focalise plus spécifiquement sur la présentation de notre démarche, qui vise à faciliter la diffusion des méthodes ROM dans l'industrie.

- Le Chapitre-4 définit notre environnement de travail. Après une brève présentation du logiciel industriel Simcenter Samcef™, nous fixons les contraintes à satisfaire pour pouvoir s'intégrer aisément dans le logiciel industriel. En particulier, l'introduction de la méthode LATIN-PGD doit pouvoir se baser sur l'architecture incrémentale existante en faisant appel aux fonctions classiques du logiciel, sans ajout d'argument, et dans la même chronologie qu'habituellement. À cette fin, un premier démonstrateur semi-industriel ROMlab a été développé puis utilisé pour simuler les différentes contraintes d'intégration.
- Le Chapitre-5 présente le cœur de notre approche qui demeure basée sur la séparation entre les équations d'équilibre et les relations de comportement. Cette version faiblement intrusive de la méthode LATIN-PGD ne s'inscrit pas dans la lignée des variantes à variables internes [Ladevèze, 1999, Cognard *et al.*, 1999] mais s'exprime plutôt comme une extension des premières variantes fonctionnelles [Ladevèze, 1985, Boisse *et al.*, 1990, Cognard et Ladevèze, 1991]. La formulation passe par l'intermédiaire de grandeurs généralisées [Ladevèze, 2017] où les forces généralisées sont supposées être uniquement fonction de l'histoire des déplacements généralisés. La construction d'un tel opérateur permet d'englober une large variété de relations de comportement, ce qui définit une approche très générale. Dans le même temps, la construction de cet opérateur représente le seul point particulièrement intrusif de ce que nous proposons. Nous détaillons également les choix des deux directions de recherche ainsi que les différents critères mis en œuvre. Une analogie succincte avec la méthode Newton-Raphson est ensuite élaborée, ce qui permet de mieux appréhender les changements à apporter au logiciel industriel en vue de l'intégration du nouveau solveur LATIN-PGD non-linéaire. Pour finir, un premier cas-test académique est présenté pour illustrer le fonctionnement de la méthode et discerner l'influence du choix de la direction de recherche.
- Le Chapitre-6 complète la formulation proposée pour l'étendre aux non-linéarités géométriques. Le cas des grandes rotations est tout spécialement analysé. Nous montrons que le choix de la direction de recherche de descente doit être minutieusement établi pour assurer la convergence. Cette extension reste bien évidemment compatible avec la gestion des non-linéarités matériaux vue précédemment, ce qui permet d'envisager aisément des couplages entre non-linéarités de natures différentes comme l'illustre un second cas-test académique.
- Le Chapitre-7 s'intéresse finalement aux non-linéarités de contact. En se basant sur une formulation Lagrangien augmenté, nous enrichissons notre approche par l'ajout d'une direction de recherche supplémentaire pour le contact. Une fois encore, nous utilisons les fonctions existantes du logiciel pour la résolution des équations de contact, ce qui limite grandement le niveau d'intrusivité de la méthode proposée. Un troisième cas-test académique vient compléter l'analyse, ce qui permet finalement de valider la méthodologie mise en œuvre : par sa grande généralité et généricité, notre approche est susceptible de traiter tout type d'élément, tout type de relation de comportement, tout type de chargement, etc. – à savoir l'ensemble des capacités offertes classiquement par un logiciel industriel *généraliste* de manière standard.

— La troisième partie est enfin dédiée aux détails d'implémentation dans le logiciel industriel par éléments finis Simcenter Samcef™ ainsi qu'à la présentation de différents cas-tests industriels englobant différents types de non-linéarités.

- Le Chapitre-8 aborde les détails pratiques d'implémentation ainsi que les diverses étapes d'optimisation qui ont été menées au regard des performances. Sous forme de schémas, nous regroupons l'ensemble des algorithmes présentés afin d'offrir une vision claire et synthétique de

l'intégration faiblement intrusive proposée. Nous évoquons ensuite quelques points d'optimisation – à différents niveaux – comme la mise en place d'une structure parallèle MPI [3] pour la méthode LATIN-PGD.

- Le Chapitre-9 fournit finalement des illustrations sur des cas industriels, ce qui permet de mettre en lumière la performance de notre approche. Sur le cas d'une aube de réacteur d'avion possédant plus de cinq millions de degrés de liberté, nous montrons comment construire efficacement une base réduite issue d'une étude paramétrique par application de notre approche LATIN-PGD faiblement intrusive dans le logiciel industriel Simcenter Samcef™.

3. Message Passing Interface

Première partie

État de l’art des modèles réduits en mécanique non-linéaire

Chapitre 1

Résolution de problèmes mécaniques non-linéaires

Ce chapitre est consacré à la présentation des problèmes mécaniques non-linéaires dépendants du temps couramment rencontrés en environnement industriel. On introduit tout d'abord les différents types de non-linéarités abordés, ce qui permet d'aboutir à la formulation du problème de référence. On présente finalement un bref état de l'art des différentes méthodologies employées pour résoudre ces problèmes en distinguant deux types d'approches : les méthodes incrémentales et celles non-incrémentales.

Sommaire

1.1 Différentes sources de non-linéarités

Bien souvent, le modèle de l'élasticité linéaire sous hypothèse de petites perturbations s'avère trop rudimentaire pour pouvoir représenter la physique de manière adéquate. Il convient alors d'enrichir les modèles pour mieux prendre en compte les différents phénomènes non-linéaires dont nous détaillons dans la suite trois principales classes. Les phénomènes d'instabilité et de flambage ne seront pas considérés dans le cadre de ces travaux mais feront l'objet d'activités ultérieures.

1.1.1 Comportements matériaux

La première grande source de non-linéarités se situe au niveau de la description du comportement des matériaux, définissant de quelles manières la matière se déforme localement. Il existe une vaste pluralité de matériaux – à l'état naturel ou élaborés par l'homme – allant des matériaux granulaires aux matériaux composites en passant par les polymères ou encore les alliages métalliques, qui possèdent chacun leurs propres propriétés mécaniques, thermiques ou chimiques. L'objectif de la science des matériaux [Germain, 1973, Chaboche, 1986] réside dans la compréhension des phénomènes physiques sous-jacents afin de pouvoir proposer des modèles mathématiques susceptibles de représenter finement le comportement de ces divers matériaux. Il en résulte une très large variété de lois de comportement faisant l'objet d'une vaste littérature [Prager, 1949, Krempl, 1975, Valanis, 1978, Chaboche, 1989, Frederick et Armstrong, 2007, Lemaitre *et al.*, 2009] qui ne sera pas redéveloppée ici. Nous retiendrons simplement le fait que, basées sur des observations phénoménologiques, des lois de comportement sont construites dans le but de refléter le plus fidèlement possible la richesse physique à une certaine échelle – microscopique, mésoscopique ou macroscopique. Nous nous limitons dans ce qui suit à donner un exemple sur une loi élasto-visco-plastique. Cette loi sera en particulier utilisée Section-9.1.

Loi élasto-visco-plastique de type Lemaitre-Chaboche. Utilisée notamment pour les métaux et alliages à haute température, cette loi de comportement à écrouissage cinématique non-linéaire et écrouissage isotrope [Lemaitre et Chaboche, 1990] modélise les phénomènes de fluage et de visco-plasticité qui découlent des mécanismes de mouvements de dislocations des grains ainsi que des effets de glissement intercristallins [Lemaitre *et al.*, 2009]. Écrite dans un cadre de thermodynamique des processus irréversibles [Germain *et al.*, 1983], on stipule que l'état du matériau peut être entièrement décrit à partir de variables d'état observables, comme la déformation totale $\boldsymbol{\varepsilon}$, et de variables internes, comme la déformation inélastique $\boldsymbol{\varepsilon}^p$, la déformation plastique cumulée p ou la variable d'écrouissage $\boldsymbol{\alpha}$; ces dernières étant respectivement associées aux forces thermodynamiques R et $\boldsymbol{X}$. Ainsi, étant données ces variables primales et duales, on postule l'existence d'un potentiel thermodynamique – potentiel d'énergie libre spécifique ψ – dont découlent les lois d'état. En notant ρ la masse volumique du matériau, et en supposant une décomposition additive des déformations $\boldsymbol{\varepsilon} = \boldsymbol{\varepsilon}^e + \boldsymbol{\varepsilon}^p$ où $\boldsymbol{\varepsilon}^e$ représente la partie élastique, alors l'énergie libre spécifique est donnée par l'équation (1.1) sous condition isotherme où $\boldsymbol{\mathcal{K}}$ représente le tenseur de Hooke élastique reliant la contrainte $\boldsymbol{\sigma}$ à la déformation élastique :

$$\rho\psi(\boldsymbol{\varepsilon}^e, p, \boldsymbol{\alpha}) = \frac{1}{2}\boldsymbol{\varepsilon}^e : \boldsymbol{\mathcal{K}} : \boldsymbol{\varepsilon}^e + \frac{1}{3}C\boldsymbol{\alpha} : \boldsymbol{\alpha} + \left(Qp - \frac{Q}{b}\left[1 - e^{-bp}\right]\right) \tag{1.1}$$

avec C, b et Q trois paramètres matériaux scalaires. L'inégalité de Clausius-Duhem conduit à l'écriture des lois d'état (1.2) par considération des différentes dérivées partielles du potentiel d'énergie libre spécifique :

$$\textit{Équations d'état :} \quad \begin{cases} \boldsymbol{\sigma} = \rho\dfrac{\partial\psi}{\partial\boldsymbol{\varepsilon}^e} \\ R = \rho\dfrac{\partial\psi}{\partial p} \\ \boldsymbol{X} = \rho\dfrac{\partial\psi}{\partial\boldsymbol{\alpha}} \end{cases} \Rightarrow \quad \begin{cases} \boldsymbol{\sigma} = \boldsymbol{\mathcal{K}} : \boldsymbol{\varepsilon}^e \\ R = Q(1 - e^{-bp}) \\ \boldsymbol{X} = \dfrac{2}{3}C\boldsymbol{\alpha} \end{cases} \tag{1.2}$$

De plus, pour décrire les processus dissipatifs liés à l'évolution des différentes variables internes, il est nécessaire d'introduire un potentiel de dissipation ϕ ou, de manière équivalente, son dual ϕ^* suivant la transformation de Legendre-Fenchel. La formulation standard généralisée du modèle élasto-visco-plastique [Ladevèze, 1999] est alors associée au pseudo-potentiel dual de dissipation ϕ^* donné par l'équation (1.3) où $\langle\bullet\rangle_+$ désigne les crochets de Macaulay donnant la partie positive :

$$\phi^*(\boldsymbol{\sigma}, R, \boldsymbol{X}) = \frac{k}{n+1}\left\langle\frac{f(\boldsymbol{\sigma}, R, \boldsymbol{X})}{k}\right\rangle_+^{n+1} \tag{1.3}$$

avec k et n deux paramètres scalaires. Dans l'espace des contraintes déviatoriques principales, la surface seuil est un ellipsoïde d'élasticité évoluant en fonction de l'état actuel du matériau. Plus spécifiquement, l'équation de la surface seuil (1.4) délimitant le domaine élastique est définie par :

$$f = J_2(\boldsymbol{\sigma} - \boldsymbol{X}) + \frac{3\gamma}{4C}\boldsymbol{X} : \boldsymbol{X} - \sigma_0 - R(p) \qquad \begin{cases} f < 0 & \text{comportement élastique} \\ f \geqslant 0 & \text{comportement visco-plastique} \end{cases} \tag{1.4}$$

avec σ_0 la taille de la zone élastique initiale, γ un paramètre scalaire et $J_2(\bullet)$ le second invariant – contrainte équivalente de Von Mises. Finalement, l'application du second principe de la thermodynamique conduit à la formulation des équations d'évolution (1.5) où $\boldsymbol{s}_\bullet$ représente la partie déviatorique :

$$\textit{Équations d'évolution :} \quad \begin{cases} \dot{\boldsymbol{\varepsilon}}^p = \dfrac{\partial \phi^*}{\partial \boldsymbol{\sigma}} \\ -\dot{p} = \dfrac{\partial \phi^*}{\partial R} \\ -\dot{\boldsymbol{\alpha}} = \dfrac{\partial \phi^*}{\partial \boldsymbol{X}} \end{cases} \Rightarrow \quad \begin{cases} \dot{\boldsymbol{\varepsilon}}^p = \dfrac{3}{2}\dot{p}\dfrac{\boldsymbol{s}_{\boldsymbol{\sigma}} - \boldsymbol{s}_{\boldsymbol{X}}}{J_2(\boldsymbol{s}_{\boldsymbol{\sigma}} - \boldsymbol{s}_{\boldsymbol{X}})} \\ \dot{p} = \left\langle \dfrac{f}{k} \right\rangle_+^n \\ \dot{\boldsymbol{\alpha}} = \dot{\boldsymbol{\varepsilon}}^p - \gamma \dot{p} \boldsymbol{\alpha} \end{cases} \tag{1.5}$$

Pour résumer, en tenant compte des paramètres matériaux élastiques classiques (module d'Young E et coefficient de Poisson ν), nous avons donc une loi de comportement à neuf paramètres pouvant éventuellement dépendre de la température.

De manière générique, les lois de comportement définissent donc une relation entre un état de contrainte $\boldsymbol{\sigma}$ et un état de déformation $\boldsymbol{\varepsilon}$ à un instant t que l'on représentera symboliquement par un opérateur $\boldsymbol{\mathcal{H}}$. Suivant une approche fonctionnelle [Chaboche, 1986, Chaboche, 1989], on notera cet opérateur $\boldsymbol{\mathcal{H}}^f$ (1.6) de la forme :

$$\boldsymbol{\sigma}(t) = \boldsymbol{\mathcal{H}}^f\Big(t, \boldsymbol{\varepsilon}(\tau \leqslant t)\Big) \tag{1.6}$$

En effet, cet opérateur ne dépend pas seulement du temps mais aussi de toute l'histoire passée du chargement dans la mesure où des processus irréversibles tels que la plasticité peuvent intervenir. Les éventuelles variables internes demeurent alors cachées derrière la notation de l'opérateur. Abandonnant la vision fonctionnelle pour une vision à variables internes, un autre choix possible dans l'écriture pourrait consister à exhiber explicitement les variables internes au niveau de l'opérateur constitutif de la loi de comportement – que l'on notera alors $\boldsymbol{\mathcal{H}}^{vi}$ (1.7). En notant $\boldsymbol{\Xi}$ la collection des variables internes qui décrivent intégralement l'état courant du matériau à un instant t, englobant par là même l'intégralité de l'histoire du chargement, on aurait alors :

$$\boldsymbol{\sigma}(t) = \boldsymbol{\mathcal{H}}^{vi}\Big(t, \boldsymbol{\varepsilon}(t), \boldsymbol{\Xi}(t)\Big) \tag{1.7}$$

avec par exemple $\boldsymbol{\Xi} = \{\dot{\boldsymbol{\varepsilon}}^p, -\dot{p}, -\dot{\boldsymbol{\alpha}}\}$ pour la loi élasto-visco-plastique de type Lemaitre-Chaboche. Nous reviendrons par la suite sur ces deux visions en Section-1.3.3, respectivement fonctionnelle et à variables internes.

1.1.2 Grandes transformations

La deuxième grande source concerne les non-linéarités géométriques. Lorsque les déplacements, les rotations ou encore les déformations deviennent « grands », l'hypothèse des transformations infinitésimales n'est plus licite. Il convient alors de dissocier la configuration initiale $\mathcal{C}_0(\Omega_0, \partial\Omega_0)$ et sa configuration déformée $\mathcal{C}(\Omega, \partial\Omega)$ où $\partial\Omega$ désigne le bord d'un domaine spatial $\Omega \subset \mathbb{R}^3$ quelconque. En notant φ l'application bijective – ou transformation – qui à tout point $\boldsymbol{x}_0 \in \Omega_0$ dans la configuration de référence fait correspondre un unique point $\boldsymbol{x} \in \Omega$ dans la configuration déformée (cf. Figure 1.1), et $\boldsymbol{u}$ le champ

de déplacement correspondant, alors on peut définir $\mathbb{F}$ l'application linéaire tangente (1.8) aussi appelée tenseur gradient de la transformation :

$$\mathbb{F} = \frac{\partial \varphi}{\partial \boldsymbol{x}_0} = \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{x}_0} = \mathbb{I} + \mathbf{grad}_{\mathcal{C}_0} \boldsymbol{u} \tag{1.8}$$

ainsi que $\mathbb{E}$ le tenseur des déformations de Green-Lagrange (1.9) donné par :

$$\mathbb{E} = \frac{1}{2}\left({}^t\mathbb{F}\mathbb{F} - \mathbb{I}\right) \tag{1.9}$$

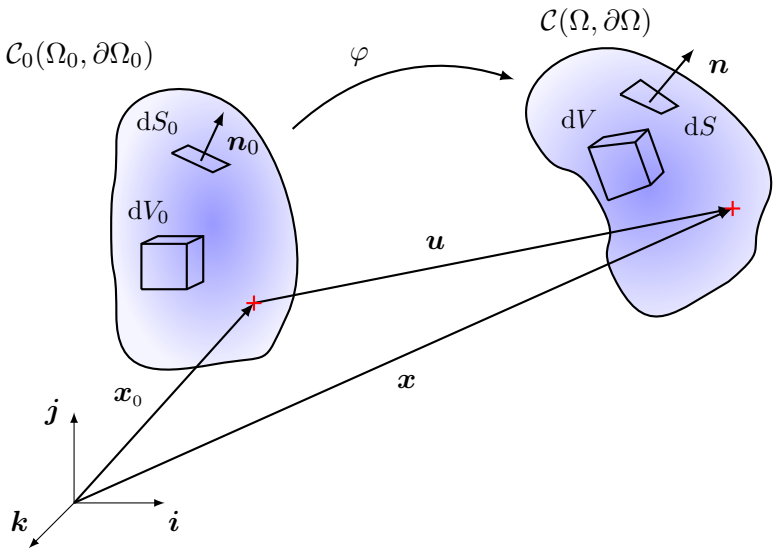


FIGURE 1.1 – Passage de la configuration initiale à la configuration déformée en transformation finie

Le jacobien de la transformation $J = \det \mathbb{F}$ représente la variation de volume entre la configuration initiale et la configuration déformée $\mathrm{d}V = J\,\mathrm{d}V_0$, au même titre qu'une surface élémentaire de normale $\boldsymbol{n}_0$ dans la configuration de référence se déforme en une surface de normale $\boldsymbol{n}$ dans la configuration actuelle par la formule $\boldsymbol{n}\,\mathrm{d}S = J\,{}^t\mathbb{F}^{-1}\boldsymbol{n}_0\,\mathrm{d}S_0$.

Alors qu'en petites perturbations il est naturel de travailler avec le tenseur (linéarisé) des petites déformations $\boldsymbol{\varepsilon}$, en grandes transformations, différents choix de variables sont possibles pour modéliser le problème. On parle alors soit de grandeurs lagrangiennes (définies sur $\mathcal{C}_0$) soit de grandeurs eulériennes (définies sur $\mathcal{C}$). En fonction du choix de la configuration de travail (initiale ou déformée), différentes mesures de déformation peuvent être introduites, auxquelles on associe systématiquement par dualité un tenseur des contraintes au sens de la conservation de la puissance des efforts intérieurs (1.10) où $\mathbb{D}$ représente le tenseur des taux de déformation eulérien :

$$\mathcal{P}_{int} = -\int_\Omega \boldsymbol{\sigma} : \mathbb{D}\,\mathrm{d}V \tag{1.10}$$

avec $\mathbb{D} = \left[\dot{\mathbb{F}}\mathbb{F}^{-1}\right]_{\mathrm{s}}$ et $\dot{\mathbb{E}} = {}^t\mathbb{F}\mathbb{D}\mathbb{F} = \left[{}^t\dot{\mathbb{F}}\mathbb{F}\right]_{\mathrm{s}}$ ce qui définit les opérations de *pull-back* et de *push-forward* [Marsden et Hughes, 1984] permettant respectivement de passer de la configuration déformée à la configuration de référence et réciproquement. Le Tableau 1.1 regroupe les mesures de déformation les plus couramment utilisées où on note :

- $\boldsymbol{\sigma}$ le tenseur des contraintes de Cauchy défini sur $\mathcal{C}$ la configuration déformée ;
- $\boldsymbol{\tau}$ le tenseur des contraintes de Kirchhoff défini sur $\mathcal{C}_0$ la configuration initiale ;

- $\mathbb{P}$ le premier tenseur des contraintes de Piola-Kirchhoff mixte et non-symétrique ;
- $\mathbb{S}$ le second tenseur des contraintes de Piola-Kirchhoff défini sur $\mathcal{C}_0$ la configuration initiale ;
- $\boldsymbol{\sigma}_{\mathbf{R}}$ le tenseur des contraintes corotationnel associé au tenseur $\dot{\boldsymbol{\Sigma}} = {}^t\mathbf{R}\mathbb{D}\mathbf{R}$ des taux de déformation tourné avec $\mathbf{R}$ l'opérateur de rotation issu de la décomposition corotationnelle.

TABLE 1.1 – Différentes mesures de déformation en transformation finie

Mesures de déformation	**Configuration**	**Relations d'équivalence**
$(\mathbb{D}, \boldsymbol{\sigma})$	Déformée $\mathcal{C}$	-
$(\mathbb{D}, \boldsymbol{\tau})$	Mixte	$\boldsymbol{\tau} = J\boldsymbol{\sigma}$
$(\dot{\mathbb{F}}, \mathbb{P})$	Mixte	$\mathbb{P} = \boldsymbol{\tau}\, {}^t\mathbb{F}^{-1}$
$(\dot{\mathbb{E}}, \mathbb{S})$	Référence $\mathcal{C}_0$	$\mathbb{S} = \mathbb{F}^{-1}\mathbb{P}$
$(\dot{\boldsymbol{\Sigma}}, \boldsymbol{\sigma}_{\mathbf{R}})$	Référence $\mathcal{C}_0$	$\boldsymbol{\sigma}_{\mathbf{R}} = J\, {}^t\mathbf{R}\boldsymbol{\sigma}\mathbf{R}$

Deux contraintes guident principalement le choix d'utiliser une mesure de déformation plutôt qu'une autre : la facilité d'implémentation numérique et l'écriture des lois de comportement matériaux. En effet, le tenseur gradient de la transformation $\mathbb{F}$ par exemple, qui permet de faire le lien entre les configurations $\mathcal{C}$ et $\mathcal{C}_0$, englobe et mélange à la fois les mouvements de corps rigides (notamment les grandes rotations), et les déformations locales de la matière. Il est donc compliqué de l'utiliser en l'état pour pouvoir décrire le milieu matériel et on lui préfère généralement soit le tenseur des déformations de Green-Lagrange $\mathbb{E}$, soit une décomposition permettant de séparer les différentes contributions. L'objectif *in fine* consiste à manipuler des quantités Ω_0-matérielles afin de respecter le principe d'objectivité.

Principe d'objectivité. Ce principe stipule que toute loi de comportement doit demeurer invariante par changement de référentiel d'observation [Lemaitre *et al.*, 2009], exigeant par là même de travailler avec des grandeurs intrinsèques. À ce titre, les scalaires, les tenseurs écrits dans la configuration de référence $\mathcal{C}_0$ ainsi que leur dérivée temporelle sont des grandeurs intrinsèques. En revanche, ce n'est pas le cas des quantités relatives à la configuration courante (déformée) $\mathcal{C}$ comme le tenseur des contraintes de Cauchy $\boldsymbol{\sigma}$ où il convient de surcroît d'employer d'autres dérivées temporelles objectives comme la dérivée de Lie ou la dérivée de Jaumann. La dérivation s'effectue alors dans un repère tourné lié à la matière encore appelé repère corotationnel.

Outre la décomposition polaire, la décomposition corotationnelle constitue l'autre principale décomposition du tenseur $\mathbb{F}$ dans la littérature. La décomposition polaire survient classiquement dans les codes éléments finis industriels. La décomposition corotationnelle, faisant quant à elle intervenir la rotation $\mathbf{R}$, définit un cadre de travail idéal pour la méthode LATIN en grandes transformations [Boucard *et al.*, 1997, Ladevèze, 1999] où les lois de comportement et les équations d'équilibre s'écrivent alors dans le même espace [Rougée, 1997]. L'un des avantages d'un tel formalisme est que l'on peut obtenir les lois de comportement en grandes transformations par simple extension des modèles développés en petites perturbations, en supposant de manière analogue une décomposition additive du taux de déformation tourné (1.11) en une partie élastique $\dot{\boldsymbol{\Sigma}}^e$ et une partie plastique $\dot{\boldsymbol{\Sigma}}^p$ sous la forme :

$$\dot{\boldsymbol{\Sigma}} = \dot{\boldsymbol{\Sigma}}^e + \dot{\boldsymbol{\Sigma}}^p \tag{1.11}$$

À noter également que cette formulation des lois de comportement basée sur la partition du taux de déformation [Dafalias, 1983] ne constitue pas une voie universelle pour la gestion de la plasticité en grandes transformations. Une autre manière classique de procéder consiste à effectuer une décomposition multiplicative [Lee et Liu, 1967, Lee, 1969] du tenseur gradient de la transformation (1.12) en une partie élastique

$\mathbb{F}^e$ et une partie plastique $\mathbb{F}^p$ telle que :

$$\mathbb{F} = \mathbb{F}^e\mathbb{F}^p \tag{1.12}$$

Ces deux modélisations ne sont pas équivalentes comme rappelé dans [Naghdi, 1990, Lemaitre *et al.*, 2009], ce qui ne permet pas de considérer la théorie de la plasticité en grandes transformations comme une théorie achevée.

1.1.3 Contact entre solides

La troisième grande source de non-linéarités se rapporte aux phénomènes de contacts. Ces derniers interviennent fréquemment dans les problèmes industriels que ce soit au niveau des processus de mise en forme [Abdali *et al.*, 1996], lors de la conception d'assemblages boulonnées [Verwaerde *et al.*, 2021], ou encore au niveau du calage des aubes de turbine par force centrifuge [Allara, 2009]. Étudiée dans un premier temps dans le cadre de l'élasticité linéaire [Duvaut et Lions, 1976], où une relation d'équivalence entre les conditions de Signorini et la formulation par inégalité variationnelle a été démontrée, la théorie du contact a fait depuis l'objet de multiples travaux [Wriggers et Zavarise, 2004, Yastrebov, 2013]. Une démonstration d'existence et d'unicité de la solution du problème de contact unilatéral sans frottement se trouve dans [Kikuchi et Oden, 1988]. En transformations finies, l'existence a également été établie [Ciarlet, 1988, Curnier *et al.*, 1992]. De manière générale, l'existence et l'unicité des problèmes de contact frottant restent des questions ouvertes [Klarbring, 1990] où seuls certains cas particuliers ont pu être démontrés [Cocu, 1984, Alart *et al.*, 1995], par le biais notamment de lois de contact régularisées.

Nous rappelons brièvement les lois de contact avec frottement de type Coulomb entre deux solides déformables Ω_1 et Ω_2 (Figure 1.2). Au niveau de la zone de contact Γ_c des conditions statiques, cinématiques et mixtes existent pour faire transiter les efforts. Une illustration des lois de contact unilatéral (1.13) et des lois de contact tangentiel en présence de frottement (1.14) est également donnée Figure 1.3. On note d_n la distance normale suivant la normale $\boldsymbol{n}$ au contact, $\boldsymbol{F}_c$ les efforts résultant du contact qui se décomposent en une partie normale F_n et une partie tangentielle $\boldsymbol{F}_t$, ainsi que $\Delta\dot{\boldsymbol{v}}_t$ la vitesse de glissement au niveau de l'interface. De plus, μ représente le coefficient de frottement dont la valeur dépend des matériaux considérés, de la rugosité des surfaces, de la lubrification, etc.

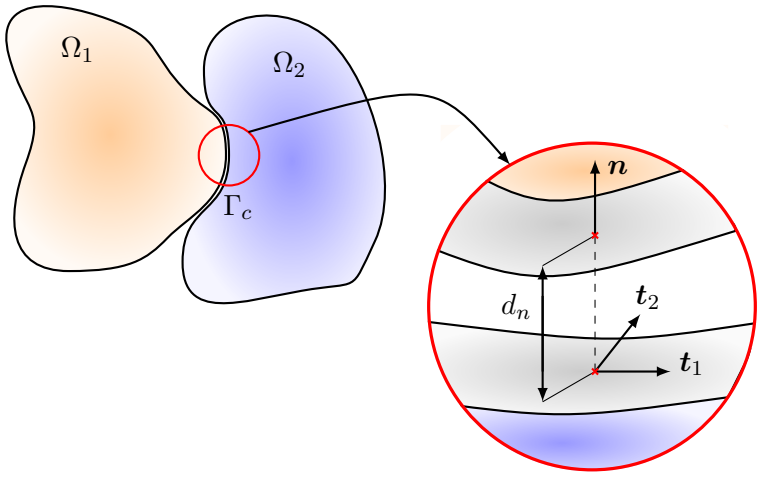


FIGURE 1.2 – Définition d'un problème de contact

Loi de contact normale.

$$\begin{cases} d_n \geqslant 0 & \text{Pas d'interpénétration} \\ F_n \leqslant 0 & \text{Pas d'adhésion} \\ d_n F_n = 0 & \text{Condition de complémentarité} \end{cases} \tag{1.13}$$

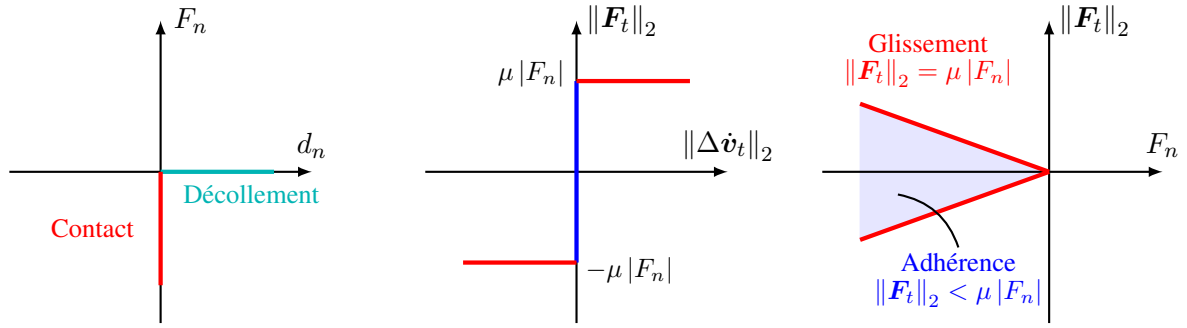
Loi de contact tangentiel.

$$\begin{cases} \text{si } \|\boldsymbol{F}_t\|_2 < \mu\,|F_n| \text{ alors } \Delta\dot{\boldsymbol{v}}_t = \boldsymbol{0} & \text{Adhérence} \\ \text{si } \|\boldsymbol{F}_t\|_2 = \mu\,|F_n| \text{ alors } \exists \beta_c > 0 & \text{Glissement} \\ \quad \text{tel que } \Delta\dot{\boldsymbol{v}}_t = -\beta_c \boldsymbol{F}_t \end{cases} \tag{1.14}$$

L'ensemble de ces équations non-linéaires peut être vu comme une relation de « comportement » non-linéaire et non régulière reliant les efforts et les déplacements au niveau de la zone de contact. Cette remarque sera particulièrement utile quand on développera la méthode LATIN au Chapitre-7. De manière analogue aux lois de comportement matériaux, nous définissons alors un opérateur $\boldsymbol{\mathcal{H}}_c$ (1.15) qui englobe de manière générique les lois de contact :

$$\boldsymbol{\mathcal{H}}_c\Big(\boldsymbol{v}, \boldsymbol{F}_c\Big) = \boldsymbol{0} \tag{1.15}$$

avec $\boldsymbol{v} = d_n \boldsymbol{n} + \boldsymbol{v}_t$. Cet opérateur sera notamment utilisé pour écrire le problème de référence, Section-1.2.



(a) Effort normal fonction du saut de déplacement

(b) Effort tangentiel fonction de la vitesse de glissement

(c) Cône de frottement

FIGURE 1.3 – Illustration des lois de contact frottant de type Coulomb

1.2 Problème de référence

Jusqu'ici, nous avons décrit les grands types de non-linéarités communément rencontrées. Ces dernières interviennent soit au niveau local pour décrire la déformation de la matière (non-linéarités matériaux), soit au niveau de la structure par des conditions aux limites (non-linéarités de contact) ou lors de l'actualisation de la configuration de travail (non-linéarités géométriques). Le dernier ingrédient pour la résolution réside dans l'équilibre de la structure. D'après le Principe des Puissances Virtuelles (1.16), la puissance des efforts d'accélération $\mathcal{P}_{acc}$ s'équilibre avec la puissance des efforts intérieurs $\mathcal{P}_{int}$ et celle des efforts extérieurs $\mathcal{P}_{ext}$ selon la relation :

$$\mathcal{P}_{int} + \mathcal{P}_{ext} = \mathcal{P}_{acc} \tag{1.16}$$

L'ajout des conditions aux limites et d'une condition initiale permet alors de définir intégralement le problème à résoudre.

Dans toute la suite, nous considérons une structure définie par son domaine $\Omega \subset \mathbb{R}^3$ soumise à différents types de sollicitations : des conditions aux limites de *Dirichlet* $\boldsymbol{u}_d$ sur le bord $\partial_D\Omega$, des conditions aux limites de *Neumann* sur le bord complémentaire $\partial_N\Omega$, des efforts volumiques $\boldsymbol{f}_d$ ainsi que des chargements thermiques $\boldsymbol{T}_d$ sur l'ensemble de la structure Ω (cf. Figure 1.4). Parmi les conditions de bord de *Neumann*, on distingue la contribution des efforts de contact $\boldsymbol{F}_c$ intervenant sur le bord $\partial_{N_1}\Omega$ du reste des contributions $\boldsymbol{F}_d$ sur la partie complémentaire $\partial_{N_2}\Omega$ avec $\partial_N\Omega = \big(\partial_{N_1}\Omega \cup \partial_{N_2}\Omega\big)$. L'étude quasi-statique en transformation finie se déroule sur un intervalle de temps $I \subset \mathbb{R}_+$ impliquant que l'ensemble des champs recherchés dépendent également d'une variable temporelle $t \in I$. On considère de plus la configuration initiale comme configuration de référence (approche Lagrangienne). On note $\boldsymbol{\mathcal{U}}$ l'ensemble des champs cinématiquement admissibles (1.17) et $\boldsymbol{\mathcal{U}_0}$ les champs cinématiques admissibles à zéro (1.18) définis sur $\Omega \times I$ par :

$$\boldsymbol{\mathcal{U}} \equiv \Big\{ \boldsymbol{u}(\boldsymbol{x},t) \mid \exists\, \widetilde{\boldsymbol{u}} : I \longrightarrow \big(H^1(\Omega)\big)^3 \text{ tel que } \widetilde{\boldsymbol{u}}(t)(\boldsymbol{x}) = \boldsymbol{u}(\boldsymbol{x},t) \text{ et } \boldsymbol{u} = \boldsymbol{u}_d \text{ sur } \partial_D\Omega \Big\} \tag{1.17}$$

$$\boldsymbol{\mathcal{U}_0} \equiv \Big\{ \boldsymbol{u}(\boldsymbol{x},t) \mid \exists\, \widetilde{\boldsymbol{u}} : I \longrightarrow \big(H^1_0(\Omega)\big)^3 \text{ tel que } \widetilde{\boldsymbol{u}}(t)(\boldsymbol{x}) = \boldsymbol{u}(\boldsymbol{x},t) \text{ et } \boldsymbol{u} = \boldsymbol{0} \text{ sur } \partial_D\Omega \Big\} \tag{1.18}$$

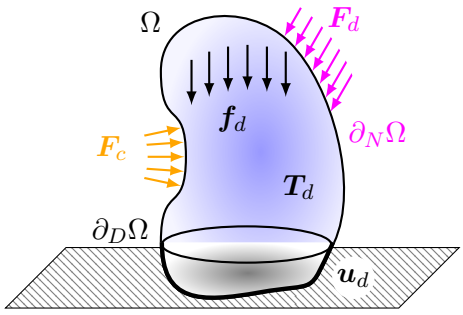


FIGURE 1.4 – Problème de référence

Formulation variationnelle. On cherche le champ de déplacement $\boldsymbol{u}$ sur $\Omega \times I$ tel que l'ensemble des équations suivantes (1.19 – 1.21) soient satisfaites :

- *Équations cinématiques* et *conditions initiales* :

$$\begin{cases} \boldsymbol{u} \in \boldsymbol{\mathcal{U}},\ \boldsymbol{u}|_{t=0} = \boldsymbol{0},\ \boldsymbol{v} = \boldsymbol{u}|_{\partial_{N_1}\Omega} \\ \mathbb{E} = \dfrac{1}{2}\Big({}^t\mathbb{F}\mathbb{F} - \mathbb{I}\Big) \quad \text{avec} \quad \mathbb{F} = \mathbb{I} + \mathbf{grad}_{\mathcal{C}_0}\boldsymbol{u} \end{cases} \tag{1.19}$$

- *Équation d'équilibre* :

$$\begin{aligned} &\forall \boldsymbol{w}^* \in \boldsymbol{\mathcal{U}_0}, \quad \text{tel que } \ \delta\mathbb{E}^* = D\mathbb{E}\cdot\boldsymbol{w}^* = \frac{1}{2}\Big[{}^t\mathbb{F}\,\mathbf{grad}_{\mathcal{C}_0}\boldsymbol{w}^* + {}^t\mathbf{grad}_{\mathcal{C}_0}\boldsymbol{w}^*\,\mathbb{F}\Big] \\ &\quad - \int_{\Omega_0} \mathbb{S} : \delta\mathbb{E}^*\ \mathrm{d}V_0 + \int_{\Omega_0} J\boldsymbol{f}_d\cdot\boldsymbol{w}^*\ \mathrm{d}V_0 + \int_{\partial_N\Omega_0} J\mathbb{F}^{-1}(\boldsymbol{F}_d + \boldsymbol{F}_c)\cdot\boldsymbol{w}^*\ \mathrm{d}S_0 = 0 \end{aligned} \tag{1.20}$$

- *Relations de comportement* et *de contact* :

$$\forall t \in I, \quad \begin{cases} \mathbb{S} = \boldsymbol{\mathcal{H}}_u\Big(t, \mathbb{E}(\tau \leqslant t)\Big) \\ \boldsymbol{\mathcal{H}}_c\Big(t, \boldsymbol{v}(t), \boldsymbol{F}_c(t)\Big) = \boldsymbol{0} \end{cases} \tag{1.21}$$

De manière générale, hormis dans certains cas spécifiques où une solution analytique existe, rechercher la solution du problème dans un espace de dimension infinie s'avère souvent délicat voire impossible. L'idée consiste alors à rechercher une approximation de la solution $\boldsymbol{u}_n$ dans un espace de dimension finie $\boldsymbol{\mathcal{U}}_n \subset \boldsymbol{\mathcal{U}}$ où $n = \dim(\boldsymbol{\mathcal{U}}_n)$ avec la propriété (1.22) de convergence :

$$\lim_{n \to +\infty} \boldsymbol{u}_n = \boldsymbol{u} \tag{1.22}$$

Issue de la formulation faible des équations, la méthode des éléments finis [Zienkiewicz et Taylor, 2005] est la méthode classiquement employée qui définit une manière de discrétiser la géométrie Ω pour se ramener à rechercher la solution dans un espace de dimension finie $\boldsymbol{\mathcal{V}} \equiv \mathbb{R}^n$ avec n le nombre de degrés de liberté. Les fonctions de forme permettent alors d'interpoler les champs dans tout l'espace par la simple connaissance des valeurs aux nœuds du maillage. Écrite sous la forme de quantités généralisées, nous donnons ci-après la formulation éléments finis du problème de référence où l'on note $\mathbf{u} \in \mathcal{I}_{\boldsymbol{\mathcal{V}}}$ les déplacements généralisés, $\mathbf{f}_{\text{int}} \in \mathcal{I}_{\boldsymbol{\mathcal{V}}}$ les efforts intérieurs généralisés et $\mathbf{f}_{\text{ext}} \in \mathcal{I}_{\boldsymbol{\mathcal{V}}}$ les efforts externes généralisés avec $\mathcal{I}_{\boldsymbol{\mathcal{V}}} \equiv L^2(I\,;\boldsymbol{\mathcal{V}})$. De plus, en présence de contacts, des conditions aux limites supplémentaires aux interfaces de contact apparaissent. Ces contraintes font notamment intervenir la trace du déplacement sur le bord $\mathbf{v} \in \mathcal{I}_{\boldsymbol{\mathcal{V}}_c}$ avec $\boldsymbol{\mathcal{V}}_c \equiv \mathbb{R}^{n_c}$, $\mathcal{I}_{\boldsymbol{\mathcal{V}}_c} \equiv L^2(I\,;\boldsymbol{\mathcal{V}}_c)$ et n_c le nombre de degrés de liberté associés aux surfaces de contact, ainsi que sa quantité duale $\boldsymbol{\xi} \in \mathcal{I}_{\boldsymbol{\mathcal{V}}_c}$ correspondant aux efforts de contact locaux.

Formulation éléments finis. On cherche le champ de déplacement $\mathbf{u} \in \mathcal{I}_{\boldsymbol{\mathcal{V}}}$ vérifiant les conditions initiales $\mathbf{u}|_{t=0} = \mathbf{0}$ tel que l'ensemble des équations suivantes (1.23 – 1.25) soient satisfaites :

- *Contraintes cinématiques* :

$$\forall t \in I, \quad \begin{cases} \boldsymbol{\mathcal{C}}_{\mathbf{u}}\Big(\mathbf{u}(t)\Big) = \mathbf{u}_d(t) \\ \boldsymbol{\mathcal{C}}_{\mathbf{v}}\Big(\mathbf{u}(t)\Big) = \mathbf{v}(t) \end{cases} \tag{1.23}$$

- *Équations d'équilibre* :

$$\forall t \in I, \quad \begin{cases} \mathbf{f}_{\text{int}}\big(t, \mathbf{u}(t)\big) - \mathbf{f}_{\text{ext}}(t) + \mathbf{f}_{\text{ctc}}\big(t, \boldsymbol{\xi}(t)\big) = \mathbf{0} \\ \mathbf{f}_{\text{ctc}}\big(t, \boldsymbol{\xi}(t)\big) = \boldsymbol{\mathcal{C}}_{\boldsymbol{\xi}}\Big(\boldsymbol{\xi}(t)\Big) \end{cases} \tag{1.24}$$

- *Relations de comportement* et *de contact* :

$$\forall t \in I, \quad \begin{cases} \mathbf{f}_{\text{int}}\big(t, \mathbf{u}(t)\big) = \boldsymbol{\mathcal{A}}_u\Big(t, \mathbf{u}(\tau \leqslant t)\Big) \\ \boldsymbol{\mathcal{A}}_c\Big(t, \mathbf{v}(t), \boldsymbol{\xi}(t)\Big) = \mathbf{0} \end{cases} \tag{1.25}$$

où les opérateurs $\boldsymbol{\mathcal{C}}_{\mathbf{u}}$, $\boldsymbol{\mathcal{C}}_{\mathbf{v}}$ et $\boldsymbol{\mathcal{C}}_{\boldsymbol{\xi}}$ sont linéaires, tandis que les opérateurs non-linéaires $\boldsymbol{\mathcal{A}}_u$ et $\boldsymbol{\mathcal{A}}_c$ rendent compte respectivement des relations de comportement et des relations de contact ; ce sont les pendants discrets des opérateurs continus non-linéaires $\boldsymbol{\mathcal{H}}_u$ et $\boldsymbol{\mathcal{H}}_c$ établis précédemment.

Bien évidemment, ce problème éléments finis non-linéaire est généralement résolu de nombreuses fois en environnement industriel en fonction de différents jeux de paramètres $\boldsymbol{\mu} \in \boldsymbol{\mathcal{D}}$ lors des processus d'optimisation. Par paramètres, nous entendons toutes données susceptibles de modifier la solution du problème de référence : données matériaux, données de chargement, modification de la géométrie, etc. Il résulte que chacune des quantités d'intérêt $\bullet^{(\mu)}$ dépend de $\boldsymbol{\mu}$, une collection de p paramètres $\boldsymbol{\mu} = \{\mu_i\}_{i \in [\![1,\, p]\!]}$. Dans la suite, par souci de concision d'écriture, nous n'écrirons pas explicitement la dépendance aux paramètres lorsque celle-ci n'est pas utile, mais nous garderons à l'esprit cet aspect paramétrique qui interviendra notamment lors des illustrations numériques.

1.3 Méthodes de résolution non-linéaire

La recherche efficace de solutions de problèmes non-linéaires occupe une large place dans la communauté scientifique ; de nombreuses techniques numériques ont ainsi émergé. On retrouve toutes les méthodes d'optimisation sans contrainte, et avec contrainte d'égalité ou d'inégalité, ainsi que les méthodes basées sur la linéarisation [Nocedal et Wright, 2006, Wriggers, 2008]. Dans ce cadre, l'analyse convexe [Rockafellar, 2015] apporte des outils permettant de statuer sur la nature locale ou globale de la solution obtenue. L'idée générale pour la résolution de système d'équations non-linéaires consiste bien souvent à définir un schéma itératif convergeant pour se ramener à une résolution successive de systèmes linéaires. Dans le domaine de la mécanique non-linéaire, la méthode Newton-Raphson ou l'une de ses nombreuses variantes est habituellement mise en œuvre [Wriggers, 2008, Ibrahimbegovic, 2009, de Borst *et al.*, 2012, Bonnet *et al.*, 2014]. Une alternative consiste à employer le solveur non-linéaire LATIN [Ladevèze, 1985, Ladevèze, 1999] défini par l'intermédiaire d'une approche non-incrémentale.

1.3.1 Méthode Newton-Raphson

La méthode Newton-Raphson, dont les fondements ont été établis à la fin du 17^e siècle [Newton, 1736], est la méthode employée dans tous les logiciels éléments finis industriels. Prenant la forme d'un algorithme incrémental et itératif basé sur la linéarisation des équations, elle présente l'immense avantage d'avoir un taux de convergence quadratique au voisinage de la solution (comme rappelé dans [Isaacson et Keller, 2012]). Partant du problème non-linéaire d'évolution (1.19 – 1.21), une discrétisation temporelle de l'intervalle de temps I est mise en place par découpage en petits incréments. Cette discrétisation permet notamment de mettre la structure sous charge progressivement afin d'assurer la convergence. La méthode consiste alors à effectuer une boucle de résolution non-linéaire en chacun des n_t pas de temps $\{t_j\}_{j\in[\![1,n_t]\!]}$ ainsi définis, ce qui conduit à résoudre une succession de problèmes non-linéaires stationnaires. Puis, par linéarisation, on se ramène finalement à résoudre une série de problèmes linéaires stationnaires jusqu'à convergence.

Connaissant la solution au pas de temps t_{j-1}, on cherche la solution au pas de temps suivant t_j. On note $\boldsymbol{\mathcal{R}}$ le résidu d'équilibre (1.26), défini à partir de l'équation (1.20) par :

$$\begin{aligned}&\forall \boldsymbol{w}^* \in \boldsymbol{\mathcal{U}_0},\\ &\boldsymbol{\mathcal{R}}(\boldsymbol{u},\boldsymbol{w}^*) = -\int_{\Omega_0} \mathbb{S} : \delta\mathbb{E}^* \,\mathrm{d}V_0 + \int_{\Omega_0} J\boldsymbol{f}_d \cdot \boldsymbol{w}^* \,\mathrm{d}V_0 + \int_{\partial_N\Omega_0} J\mathbb{F}^{-1}(\boldsymbol{F}_d + \boldsymbol{F}_c)\cdot \boldsymbol{w}^* \,\mathrm{d}S_0\end{aligned} \tag{1.26}$$

où l'évaluation des termes manquants résulte de l'intégration locale des lois (1.21) définies précédemment. Alors, par annulation du développement en séries de Taylor au premier ordre du résidu d'équilibre autour de l'itéré précédent, et en notant $\boldsymbol{u}_j^{(k)}$ $(k \in \mathbb{N}^*)$ les approximations successives de la solution recherchée $\boldsymbol{u}_j$ à l'instant t_j, le schéma de résolution (1.27) consiste à trouver la correction $\Delta\boldsymbol{u}_j^{(k+1)}$ vérifiant :

$$\begin{cases} D\boldsymbol{\mathcal{R}}\left(\boldsymbol{u}_j^{(k)},\boldsymbol{w}^*\right)\cdot\Delta\boldsymbol{u}_j^{(k+1)} = -\boldsymbol{\mathcal{R}}\left(\boldsymbol{u}_j^{(k)},\boldsymbol{w}^*\right) & \forall \boldsymbol{w}^* \in \boldsymbol{\mathcal{U}_0} \\ \boldsymbol{u}_j^{(k+1)} = \boldsymbol{u}_j^{(k)} + \Delta\boldsymbol{u}_j^{(k+1)} \end{cases} \tag{1.27}$$

où $D\boldsymbol{\mathcal{R}}(\boldsymbol{u}_j^{(k)},\boldsymbol{w}^*)\cdot\Delta\boldsymbol{u}_j^{(k+1)}$ caractérise la dérivée directionnelle de $\boldsymbol{\mathcal{R}}$ autour de l'itéré $\boldsymbol{u}_j^{(k)}$ dans la direction $\Delta\boldsymbol{u}_j^{(k+1)}$. Dans [Wriggers, 2008, Ibrahimbegovic, 2009, de Borst *et al.*, 2012, Bonnet *et al.*, 2014], on pourra trouver de plus amples détails. L'algorithme 1 récapitule les grandes étapes de résolution de la méthode Newton-Raphson où η correspond à un critère d'arrêt fixé par l'utilisateur. L'évaluation de résidu à chaque itération demande notamment l'intégration des équations locales du problème (1.25) où diverses

techniques de résolution sont employées en fonction du type de non-linéarité considérée.

Une fois ramenée à la résolution successive de problèmes linéaires, n'importe quelle technique classique de résolution de système linéaire peut alors être utilisée : soit des solveurs directs comme MUMPS [1], soit des solveurs itératifs tels que les méthodes de projection ou les solveurs de Krylov [Ciarlet, 2007, Golub et Van Loan, 2013].

Algorithme 1 : Méthode Newton-Raphson

Initialisation : t_0, $\boldsymbol{u}_0$

/* Boucle sur les pas de temps $\{t_j\}_{j\in[\![1,n_t]\!]}$ */

for $j = 1 \,..\, n_t$ **do**

  Première estimation $\boldsymbol{u}_j^{(1)} = \boldsymbol{u}_{j-1}$

  Evaluation du résidu d'équilibre $\boldsymbol{\mathcal{R}}(\boldsymbol{u}_j^{(1)}, \boldsymbol{w}^*)$

  /* Boucle $\bullet^{(k)}$ de résolution non-linéaire ($k \in \mathbb{N}^*$) */

  while $\|\boldsymbol{\mathcal{R}}\| \geqslant \eta$ **do**

    Calcul et assemblage de la matrice tangente

    Détermination de la nouvelle correction $D\boldsymbol{\mathcal{R}}(\boldsymbol{u}_j^{(k)}, \boldsymbol{w}^*) \cdot \Delta\boldsymbol{u}_j^{(k+1)} = -\boldsymbol{\mathcal{R}}(\boldsymbol{u}_j^{(k)}, \boldsymbol{w}^*)$

    Mise à jour de l'itéré $\boldsymbol{u}_j^{(k+1)} = \boldsymbol{u}_j^{(k)} + \Delta\boldsymbol{u}_j^{(k+1)}$

    Evaluation du nouveau résidu d'équilibre $\boldsymbol{\mathcal{R}}(\boldsymbol{u}_j^{(k+1)}, \boldsymbol{w}^*)$

1.3.1.1 Spécificités des non-linéarités matériaux

Concernant les non-linéarités matériaux, il n'est pas rare d'avoir recours à un algorithme de Newton-Raphson *interne* pour intégrer localement en chaque point de Gauss de chaque élément les équations non-linéaires caractérisant le comportement telles que présentées en Section-1.1 pour le cas des lois élasto-visco-plastiques de type Lemaitre-Chaboche. Dans toute la suite, quand nous ferons référence à la méthode de Newton-Raphson, nous désignerons l'algorithme 1 dans son ensemble sans préciser la méthode d'intégration locale des lois de comportement. En effet, une des forces de la méthode LATIN-PGD faiblement intrusive qui sera présentée dans la deuxième partie de ce manuscrit est de pouvoir s'accommoder avec les techniques de résolution utilisées dans les logiciels industriels sans aucune modification.

1.3.1.2 Spécificités des non-linéarités liées aux contacts

Concernant les non-linéarités de contact, une étape essentielle des codes éléments finis réside dans la définition des opérateurs $\boldsymbol{\mathcal{C}}_{\mathbf{v}}$ et $\boldsymbol{\mathcal{C}}_{\boldsymbol{\xi}}$. Que le contact soit géré nœuds à nœuds, entre un groupe de nœuds et un groupe de faces, ou entre deux groupes de faces, il est nécessaire de savoir évaluer les éléments qui vont être reliés pour le contact. Ceci peut s'avérer être un véritable challenge au regard des performances en grandes transformations (grands glissements par exemple) où l'algorithme de recherche topologique peut devenir coûteux et représenter une part importante du coût total du calcul. Une fois la recherche du contact effectuée, on détermine la projection et on résout les équations locales décrites par l'opérateur $\boldsymbol{\mathcal{A}}_c$ élément de contact par élément de contact. Il en résulte des conditions aux limites qui sont généralement prises en compte par des méthodes de pénalisation, de Lagrangien ou encore de Lagrangien augmenté [Wriggers et Zavarise, 2004].

1. MUltifrontal Massively Parallel sparse direct Solver : http://mumps.enseeiht.fr

Pénalisation. Un coefficient de pénalisation p est rajouté dans la formulation énergétique pour forcer le système à se conformer aux contraintes induites par les conditions de contact. Le choix de ce coefficient de pondération est délicat : un coefficient trop faible ne permet pas de vérifier correctement les contraintes tandis qu'un coefficient trop fort peut amener des difficultés numériques liées à une dégradation du conditionnement. De manière générique, si $\mathcal{F}$ représente la fonctionnelle à minimiser sous la contrainte c alors on introduit une nouvelle fonctionnelle $\mathcal{G}$ (1.28) telle que :

$$\mathcal{G}(\mathbf{u}) = \mathcal{F}(\mathbf{u}) + \frac{1}{2} p c^2(\mathbf{u}) \tag{1.28}$$

Pour le cas du contact unilatéral par exemple, on passe ainsi d'un problème d'optimisation sous contrainte d'inégalité à un problème d'optimisation sans contrainte ce qui permet ensuite d'utiliser les algorithmes classiques d'optimisation sans contrainte [Nocedal et Wright, 2006].

Lagrangien augmenté. Un multiplicateur de Lagrange est introduit, ce qui permet de vérifier de manière exacte les contraintes du problème de contact. Un coefficient de pénalisation est généralement conservé pour stabiliser la résolution ce qui conduit à une formulation de type Lagrangien augmenté (1.29) donné par :

$$\mathcal{L}(\mathbf{u}, \lambda) = \mathcal{F}(\mathbf{u}) + \lambda c(\mathbf{u}) + \frac{1}{2} p c^2(\mathbf{u}) \tag{1.29}$$

La formulation Lagrangien augmenté a été largement étudiée dans le cadre des problèmes de contact, que ce soit sous hypothèse de petites perturbations [Kikuchi et Oden, 1988] ou en grandes transformations [Alart et Curnier, 1991, Simo et Laursen, 1992]. Cette formulation débouche sur une fonctionnelle de type point selle à minimiser [Pietrzak et Curnier, 1999].

La prise en compte des conditions aux limites par pénalisation ou par multiplicateurs de Lagrange correspond à résoudre le problème de minimisation sous contrainte d'égalité. Un processus itératif est donc nécessaire pour respecter les contraintes d'inégalité en venant mettre à jour les conditions de contact à chaque itération. Pour ce faire, plusieurs méthodes sont possibles : la méthode des statuts, la méthode d'Uzawa ou encore la méthode de Newton [Wriggers et Zavarise, 2004] en sont quelques illustrations.

Méthode des statuts. La méthode des statuts est relativement simple à mettre en œuvre [Wriggers, 1995]. Elle consiste à modifier les statuts des conditions de contact à chaque itération de l'algorithme en imposant des conditions de *Dirichlet* et de *Neumann*. Typiquement, l'algorithme comporte deux grandes étapes à chaque itération :

- la résolution de l'équilibre avec prise en compte des conditions aux limites induites par le contact ;
- la mise à jour des statuts des contacts en fonction des résultats de la résolution précédente. Par exemple, si la résolution conduit à une interpénétration $(d_n < 0)$ alors une condition de *Dirichlet* est imposée à l'itération suivante pour assurer la continuité des déplacements $(d_n = 0)$.

L'algorithme est considéré comme convergé lorsqu'aucun nœud n'a changé de statut entre deux itérations successives. Malgré sa simplicité, l'opérateur de raideur doit être actualisé à chaque itération ce qui peut s'avérer coûteux.

Méthode d'Uzawa. L'algorithme d'Uzawa [Bertsekas, 1984, Simo et Laursen, 1992] ajoute des rigidités de type ressort – proportionnelles à la valeur de l'interpénétration – pour pénaliser les nœuds interpénétrants et les ramener dans une situation acceptable. On se retrouve alors à itérer sur les efforts de contact. Pour le cas du contact unilatéral par exemple, connaissant les efforts de contact normaux $F_n^{(i)}$ à une itération, on vient résoudre l'équation d'équilibre (1.20) ce qui fournit une nouvelle valeur du saut de

déplacement $d_n^{(i)}$ permettant alors de réactualiser la valeur des efforts de contact suivant la formule (1.30) avec ρ un paramètre de la méthode :

$$F_n^{(i+1)} = -\langle F_n^{(i)} + \rho d_n^{(i)} \rangle_- \tag{1.30}$$

L'algorithme d'Uzawa ne faisant intervenir que des conditions de *Neumann*, il présente l'avantage de pouvoir mutualiser la factorisation de l'opérateur de rigidité sur l'ensemble des itérations lorsque l'on utilise un solveur direct.

Méthode de Newton. Cette méthode est par exemple mise en œuvre dans [Curnier et Alart, 1988, Alart et Curnier, 1991, Alart, 1997]. Les lois de contact n'étant pas toujours différentiables, il est possible de les régulariser pour retrouver une loi bi-univoque. L'algorithme de Newton fait alors intervenir le jacobien généralisé ce qui permet d'avoir une architecture commune de résolution pour intégrer simultanément les non-linéarités matériaux et les non-linéarités de contact entre autres. C'est la version standard en grandes transformations utilisée dans le logiciel Simcenter Samcef™ où la non-linéarité de type contact est gérée comme n'importe quelle autre non-linéarité (matérielle ou géométrique). Cette résolution du contact couplée se distingue donc d'une résolution découplée nécessitant l'introduction d'une boucle d'optimisation supplémentaire pour gérer spécifiquement le contact entre chaque itération de l'algorithme 1 de Newton-Raphson si d'autres non-linéarités entrent en considération. Dès que les non-linéarités géométriques sont considérées (la configuration de référence $\mathcal{C}_0$ ne coïncide plus avec la configuration déformée $\mathcal{C}$) alors un reprofilage de la matrice jacobienne est requis à chaque itération dans la mesure où les opérateurs $\boldsymbol{\mathcal{C}}_{\mathbf{v}}$ et $\boldsymbol{\mathcal{C}}_{\boldsymbol{\xi}}$ peuvent varier d'une itération à une autre.

1.3.1.3 Spécificités des non-linéarités géométriques

Concernant les non-linéarités géométriques, il faut prendre en compte la variation de géométrie, ce qui induit une rigidité – géométrique – supplémentaire [Zienkiewicz et Taylor, 2005, Ibrahimbegovic, 2009]. Sans perte de généralité, on se place dans le cadre de l'(hyper-)élasticité où on note $\mathbb{C}$ le tenseur d'élasticité qui découle d'un potentiel densité d'énergie interne et on ne considère pas les efforts de contact. De plus, pour simplifier les notations, on note $\bar{\boldsymbol{u}} = \boldsymbol{u}_j^{(k)}$ l'itéré du schéma de résolution – supposé connu – autour duquel on effectue la linéarisation. Ainsi, toutes les quantités $\bar{\bullet}$ sont connues et évaluées en $\bar{\boldsymbol{u}}$. L'opérateur de rigidité tangent $\mathbf{K}_{\mathrm{T}}$ éléments finis découle de la dérivée directionnelle $D\boldsymbol{\mathcal{R}}(\bar{\boldsymbol{u}}, \boldsymbol{w}^*) \cdot \Delta\boldsymbol{u}$, explicitée en (1.31), qui se décompose en deux termes [Wriggers, 2008] :

$$D\boldsymbol{\mathcal{R}}(\bar{\boldsymbol{u}}, \boldsymbol{w}^*) \cdot \Delta\boldsymbol{u} = \underbrace{\int_{\Omega_0} \delta\bar{\mathbb{E}}^* \cdot \bar{\mathbb{C}}[\Delta\bar{\mathbb{E}}] \, \mathrm{d}V_0}_{\text{terme de rigidité matériaux}} + \underbrace{\int_{\Omega_0} \mathbf{grad}_{\mathcal{C}_0} \Delta\boldsymbol{u} \, \bar{\mathbb{S}} \cdot \mathbf{grad}_{\mathcal{C}_0} \boldsymbol{w}^* \, \mathrm{d}V_0}_{\text{terme de rigidité géométrique}} \tag{1.31}$$

avec $\Delta\bar{\mathbb{E}} = \frac{1}{2}\left[{}^t\bar{\mathbb{F}} \, \mathbf{grad}_{\mathcal{C}_0} \Delta\boldsymbol{u} + {}^t\mathbf{grad}_{\mathcal{C}_0} \Delta\boldsymbol{u} \, \bar{\mathbb{F}}\right]$ l'incrément de déformation du tenseur de Green-Lagrange qui possède la même structure que $\delta\bar{\mathbb{E}}^* = \frac{1}{2}\left[{}^t\bar{\mathbb{F}} \, \mathbf{grad}_{\mathcal{C}_0} \boldsymbol{w}^* + {}^t\mathbf{grad}_{\mathcal{C}_0} \boldsymbol{w}^* \, \bar{\mathbb{F}}\right]$. À noter qu'en présence de forces suiveuses, un troisième terme vient s'ajouter que nous ne détaillons pas ici.

Sous le formalisme éléments finis, l'expression (1.31) coïncide avec la formule (1.32) présentant les matrices de rigidité matériaux $\mathbf{K}_m$ et géométrique $\mathbf{K}_g$ respectivement. En notant $\mathbf{B}^{[e]}$ l'opérateur éléments finis élémentaire reliant les déformations $\delta\mathbb{E}^{[e]}$ aux déplacements généralisés aux nœuds $\mathbf{u}^{[e]}$ sur un élément $\delta\mathbb{E}^{[e]} = \mathbf{B}^{[e]}\mathbf{u}^{[e]}$ alors la matrice tangente $\mathbf{K}_{\mathrm{T}}^{[e]}$ élémentaire (1.32) s'écrit :

$$\mathbf{K}_{\mathrm{T}}^{[e]} = \underbrace{\int_{\Omega_e} {}^t\mathbf{B}^{[e]} \mathbf{C}^{[e]} \mathbf{B}^{[e]} \, \mathrm{d}\Omega_e}_{\mathbf{K}_m^{[e]}} + \underbrace{\int_{\Omega_e} \frac{\partial \, {}^t\mathbf{B}^{[e]}}{\partial \mathbf{u}^{[e]}} \mathbb{S}^{[e]} \, \mathrm{d}\Omega_e}_{\mathbf{K}_g^{[e]}} \tag{1.32}$$

où $\mathbf{C}^{[e]}$ désigne la matrice tangente élémentaire constitutive de la loi matériau. Nous verrons notamment au Chapitre-6 que cette rigidité géométrique possède un fort impact sur le choix de la direction de recherche pour la méthode LATIN-PGD en grandes transformations.

De plus, en cas d'instabilité géométrique, des méthodes de continuation peuvent s'ajouter [Riks, 1972, Crisfield, 1981] pour passer les points critiques. Elles ne sont pas étudiées dans le cadre de ces travaux.

1.3.2 Variantes autour de la méthode Newton-Raphson

L'évaluation systématique de la direction tangente et la résolution du système linéaire à chaque itération peuvent s'avérer coûteuses dans la recherche successive des termes de la série convergente $\left(\Delta\mathbf{u}_j^{(k)}\right)_{k\in\mathbb{N}}$. De multiples variantes de la méthode Newton-Raphson ont ainsi été développées. L'idée générale consiste à utiliser une direction plus ou moins proche de la direction tangente mais plus simple à calculer quitte à effectuer quelques itérations supplémentaires – moins coûteuses. On présente dans ce qui suit quelques variantes [Matthies et Strang, 1979, Wriggers, 2008] couramment rencontrées dans les codes de calculs industriels.

1.3.2.1 Newton-Raphson modifié

Cette variante, la plus simple, consiste à calculer et à assembler la matrice tangente à la première itération puis à la conserver pour toutes les itérations suivantes. On peut alors garder en mémoire la factorisation de la matrice pour l'ensemble des itérations de la boucle de résolution non-linéaire, ce qui conduit généralement à des gains en temps de calculs appréciables dès que le nombre n de degrés de liberté devient grand. En effet, la factorisation de la matrice tangente possède une complexité algorithmique en $\mathcal{O}(n^3)$ alors que la résolution du système triangulaire résultant ne présente une complexité qu'en $\mathcal{O}(n^2)$. En contrepartie, le taux de convergence est dégradé : on passe d'une convergence quadratique à une convergence linéaire localement. Un compromis est donc généralement trouvé sur la fréquence de réactualisation de la matrice tangente en prenant en considération la vitesse de décroissance de l'erreur en résidu comparé au coût de résolution sur une itération.

1.3.2.2 Quasi-Newton

Cette seconde variante, dont l'idée générale consiste à approximer la direction tangente par la sécante, possède une convergence super-linéaire [Luenberger et Ye, 2016]. Se plaçant à un pas de temps t_j donné, on omet l'indice j dans ce paragraphe pour simplifier les notations. Plutôt que de résoudre le système linéaire (1.27) à partir de la vraie direction tangente $\mathbf{K}_\mathrm{T}$, on préfère utiliser la direction sécante $\mathbf{K}_\mathrm{S}$ définie à partir de la formule (1.33) où $\mathbf{G}^{(k)}$ représente le résidu évalué en $\mathbf{u}^{(k)}$ – équivalent de $\boldsymbol{\mathcal{R}}(\boldsymbol{u}^{(k)}, \boldsymbol{w}^*)$ – sous le formalisme éléments finis, $\boldsymbol{\delta}^{(k)} = \Delta\mathbf{u}^{(k)}$ et $\mathbf{g}^{(k)}$ est la variation du résidu sur deux itérations successives :

$$\mathbf{K}_\mathrm{S}^{(k)}\Big(\underbrace{\mathbf{u}^{(k)} - \mathbf{u}^{(k-1)}}_{\boldsymbol{\delta}^{(k)}}\Big) = \underbrace{\mathbf{G}^{(k-1)} - \mathbf{G}^{(k)}}_{\mathbf{g}^{(k)}} \tag{1.33}$$

Plus précisément, on cherche à évaluer à moindre coût l'inverse de la matrice $\mathbf{K}_\mathrm{S}^{(k)}$ lors de la résolution du système linéaire. Plusieurs algorithmes ont été développés à cet effet dont la méthode BFGS [Broyden, 1965] couramment utilisée dans les codes industriels.

Méthode BFGS. Cette méthode développée à l'origine dans le cadre de problèmes d'optimisation non-linéaires sans contrainte définit une procédure permettant de calculer aisément l'inverse $\left[\mathbf{K}_\mathrm{S}^{(k)}\right]^{-1}$ à partir de $\left[\mathbf{K}_\mathrm{S}^{(k-1)}\right]^{-1}$ la matrice inverse des itérations précédentes [Matthies et Strang, 1979]. L'update de la

matrice passe par une correction de rang faible (1.34) qui présente l'avantage de conserver les propriétés intéressantes – le côté symétrique défini-positif :

$$\left[\mathbf{K}_{\mathrm{S}}^{(k)}\right]^{-1} = \left(\mathbf{1} + \boldsymbol{a}^{(k)}\,{}^{t}\boldsymbol{b}^{(k)}\right)\left[\mathbf{K}_{\mathrm{S}}^{(k-1)}\right]^{-1}\left(\mathbf{1} + \boldsymbol{b}^{(k)}\,{}^{t}\boldsymbol{a}^{(k)}\right) \tag{1.34}$$

avec

$$\begin{cases} \boldsymbol{a}^{(k)} = \dfrac{1}{{}^{t}\mathbf{g}^{(k)}\boldsymbol{\delta}^{(k)}}\boldsymbol{\delta}^{(k)} \\ \boldsymbol{b}^{(k)} = \left[\dfrac{{}^{t}\boldsymbol{\delta}^{(k)}\mathbf{g}^{(k)}}{{}^{t}\boldsymbol{\delta}^{(k)}\mathbf{K}_{\mathrm{S}}^{(k-1)}\boldsymbol{\delta}^{(k)}}\right]^{1/2}\mathbf{G}^{(k-1)} - \mathbf{g}^{(k)} \end{cases} \tag{1.35}$$

On note en particulier que l'on a également la relation $\mathbf{K}_{\mathrm{S}}^{(k-1)}\boldsymbol{\delta}^{(k)} = -\mathbf{G}^{(k-1)}$ de l'itéré précédent. Dans la pratique, on n'a pas besoin de déterminer explicitement la nouvelle approximation de la matrice pour résoudre le système linéaire dans l'algorithme 1 de Newton-Raphson si on conserve en mémoire les différents vecteurs $\boldsymbol{a}^{(k)}$ et $\boldsymbol{b}^{(k)}$ au cours des itérations. Cette procédure s'avère en général efficace dès lors que le nombre de facteurs reste raisonnable ($0 < k \leqslant 15$) ce qui permet de considérer le temps associé aux différents produits scalaires de complexité $\mathcal{O}(n)$ négligeable devant la résolution du système triangulaire factorisé par montée-descente. Lorsque k devient trop grand, on peut alors se donner une nouvelle matrice initiale (et sa factorisation) et recommencer la procédure. L'algorithme 2 synthétise la procédure d'update par la méthode BFGS à une itération k donnée.

Algorithme 2 : Procédure d'update-BFGS à une itération k donnée

`/* Procédure permettant de calculer à moindre coût l'itéré` $\mathbf{u}^{(k+1)}$ `*/`

Data : On connait $\mathbf{K}_{\mathrm{S}}^{(0)} = \boldsymbol{L}\,{}^{t}\boldsymbol{L}$ (et sa factorisation) ainsi que $\mathbf{u}^{(k)}$ et $\{\boldsymbol{a}^{(k)}, \boldsymbol{b}^{(k)}\}$

Result : On cherche l'approximation $\left[\mathbf{K}_{\mathrm{S}}^{(k)}\right]^{-1}$ (1.34) de rang 2 pour résoudre $\mathbf{K}_{\mathrm{S}}^{(k)}\boldsymbol{\delta}^{(k+1)} = \mathbf{G}^{(k)}$

`/* Initialisation */`

$\boldsymbol{q}^{(k)} = \mathbf{G}^{(k)}$

`/* Résolution */`

for $i = k \,..\, 1$ **do**

  On calcule $\boldsymbol{q}^{(i-1)} = \left(\mathbf{1} + \boldsymbol{b}^{(i)}\,{}^{t}\boldsymbol{a}^{(i)}\right)\boldsymbol{q}^{(i)}$

On effectue la résolution $\left[\mathbf{K}_{\mathrm{S}}^{(0)}\right]^{-1}\boldsymbol{q}^{(0)} = \boldsymbol{c}^{(1)}$ connaissant la factorisation $\mathbf{K}_{\mathrm{S}}^{(0)} = \boldsymbol{L}\,{}^{t}\boldsymbol{L}$

for $i = 1 \,..\, k$ **do**

  On calcule $\boldsymbol{c}^{(i+1)} = \left(\mathbf{1} + \boldsymbol{a}^{(i)}\,{}^{t}\boldsymbol{b}^{(i)}\right)\boldsymbol{c}^{(i)}$

`/* Actualisation */`

On a finalement $\boldsymbol{c}^{(k+1)} = \boldsymbol{\delta}^{(k+1)}$ et on déduit $\mathbf{u}^{(k+1)} = \mathbf{u}^{(k)} + \boldsymbol{\delta}^{(k+1)}$

1.3.3 Méthode LATIN

La méthode LATIN [Ladevèze, 1985, Ladevèze, 1999] consiste à mener une approche globale et itérative simultanément sur l'espace et le temps. Alors que dans le cadre d'une résolution classique par un algorithme de Newton-Raphson, l'intervalle d'étude I est parcouru une unique fois afin d'orchestrer la résolution pas de temps par pas de temps, avec la méthode LATIN cet intervalle temporel est pris d'un seul tenant et étudié à de multiples reprises au cours de la résolution. La méthode LATIN est, de fait, une méthode itérative non-incrémentale qui, à chaque itération, produit une meilleure approximation de la solution sur l'ensemble de l'espace et du temps.

Cette méthode a été largement étudiée avec différents cas d'application : en présence de non-linéarités matériaux [Relun *et al.*, 2013, Bhattacharyya *et al.*, 2018], pour les problèmes d'assemblages [Blanzé *et al.*, 1996], de contact [Giacoma *et al.*, 2014, Ribeaucourt *et al.*, 2007] ou de décomposition de domaines [Champaney *et al.*, 1997], dans le cadre de la dynamique non-linéaire [Royer, 1990, Odièvre *et al.*, 2010], pour les problèmes multi-physiques [Dureisseix *et al.*, 2003, Néron et Dureisseix, 2008], ou encore en présence de grandes transformations [Boucard *et al.*, 1997, Bellenger et Bussy, 1998]. On pourra trouver en Annexe A un état de l'art des principaux développements menés autour de la méthode LATIN ces dernières années. Avant tout, la méthode LATIN doit être vue comme un solveur non-linéaire capable de traiter sous un même formalisme soit des problèmes d'évolution non-linéaires sur un domaine donné, soit des méthodes de décomposition de domaines – avec ou sans présence de non-linéarités dans les sous-domaines –, soit de jouer le rôle de couplage entre différentes physiques. Dans le cadre de ces travaux, on s'intéresse tout particulièrement au premier cas impliquant n'importe quel type de non-linéarité.

1.3.3.1 Principes fondamentaux

La méthode LATIN se base sur 3 principes fondamentaux [Ladevèze, 1999] :

- **Principe P1** : *la séparation des difficultés*. Les équations du problème sont séparées en deux groupes : un premier groupe d'équations locales en temps et en espace, éventuellement non-linéaires, noté $\boldsymbol{\Gamma}$ et un second groupe d'équations linéaires, éventuellement globales en espace noté $\mathbf{A}_d$;
- **Principe P2** : *un algorithme à deux directions de recherche*. À chaque itération, on construit alternativement une solution dans chacun des groupes $\boldsymbol{\Gamma}$ et $\mathbf{A}_d$: comme le montre la Figure 1.5, la première étape, dite *locale* (aussi appelée *étape de montée*) consiste à rechercher la solution $\widehat{\boldsymbol{s}}_\ell \in \boldsymbol{\Gamma}$ sachant $\boldsymbol{s}_\ell \in \mathbf{A}_d$ en exploitant la direction de recherche de montée $\boldsymbol{\Upsilon}^+$ tandis que la seconde étape, dite *globale* ou *linéaire* (aussi appelée *étape de descente*), consiste à rechercher la solution $\boldsymbol{s}_{\ell+1} \in \mathbf{A}_d$ connaissant $\widehat{\boldsymbol{s}}_\ell \in \boldsymbol{\Gamma}$ en utilisant la direction de recherche de descente $\boldsymbol{\Upsilon}^-$. Les deux espaces $\boldsymbol{\Upsilon}^+$ et $\boldsymbol{\Upsilon}^-$ additionnels sont caractérisés par deux opérateurs $\boldsymbol{\Theta}^+$ et $\boldsymbol{\Theta}^-$ respectivement. L'opérateur $\boldsymbol{\Theta}^-$ est choisi linéaire pour garantir une étape globale linéaire, tandis que l'autre opérateur $\boldsymbol{\Theta}^+$ peut potentiellement être pris non-linéaire mais doit rester local. Il faut noter que ces opérateurs sont les paramètres clés de la méthode LATIN. Des résultats de convergence ont été démontrés dans [Ladevèze, 1999] sous certaines conditions – en considérant les opérateurs $\boldsymbol{\Theta}^+$ et $\boldsymbol{\Theta}^-$ linéaires défini-positifs, strictement monotones et orthogonaux entre eux au sens d'une certaine norme ;
- **Principe P3** : *l'utilisation d'une représentation adaptée*. Le fait de raisonner sur l'ensemble de l'espace et du temps nous amène à manipuler des solutions qui peuvent s'avérer coûteuses en espace de stockage et en temps de calculs. Cependant, le fait de connaître une approximation spatio-temporelle complète à chaque itération nous permet de mettre en œuvre une technique de réduction de modèle par PGD [Benner *et al.*, 2021] au niveau de l'étape globale, d'où le nom de méthode LATIN-PGD. Pour être plus précis, les différents champs sont alors décrits comme (1.36) une somme de produits à variables séparées espace-temps :

$$\boldsymbol{s}_\ell(\boldsymbol{x},t) = \sum_{i=1}^{m_\ell} \lambda_i(t)\boldsymbol{\Phi}_i(\boldsymbol{x}) \tag{1.36}$$

  avec $\{\lambda_i\}_{1\leqslant i\leqslant m_\ell} \in \mathcal{I}_\mathbb{R} \equiv L^2(I)$ et $\{\boldsymbol{\Phi}_i\}_{1\leqslant i\leqslant m_\ell} \in \boldsymbol{\mathcal{T}} \equiv H_0^1(\Omega)$. Dans la suite, nous appellerons ainsi modes PGD les m_ℓ produits spatio-temporels qui forment ensemble ce que l'on appelle une base réduite. Nous reviendrons plus en détails sur les méthodes de réduction de modèles dans le Chapitre-2.

Un critère d'arrêt basé sur la distance entre les quantités $\widehat{\boldsymbol{s}}_\ell \in \boldsymbol{\Gamma}$ et $\boldsymbol{s}_{\ell+1} \in \mathbf{A}_d$ au sens d'une certaine norme est généralement utilisé. L'algorithme 3 décrit le fonctionnement de la méthode LATIN-PGD de manière générique où une étape de relaxation est introduite pour aider la convergence et donc accroître la robustesse.

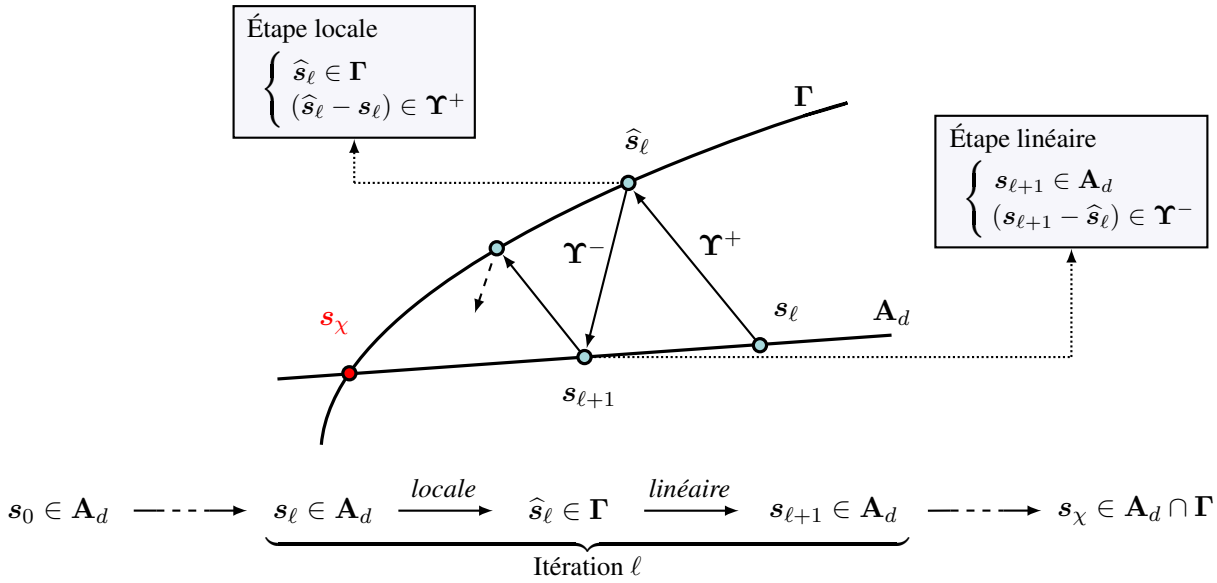


FIGURE 1.5 – Schéma itératif convergent méthode LATIN-PGD

Algorithme 3 : Méthode LATIN-PGD

Initialisation : $s_0 \in \mathbf{A}_d$

/* Boucle $\bullet_\ell$ sur les itérations LATIN */

while $\omega_\ell \geqslant \eta$ **do**

/* Étape locale */

Connaissant $s_\ell \in \mathbf{A}_d$, on cherche $\widehat{s}_\ell \in \Gamma$ pour chaque pas de temps tel que :

$$\begin{cases} \widehat{s}_\ell \in \Gamma \\ (\widehat{s}_\ell - s_\ell) \in \Upsilon^+ \end{cases}$$

/* Étape globale */

Connaissant $\widehat{s}_\ell \in \Gamma$, on cherche $s_{\ell+1} \in \mathbf{A}_d$ pour chaque pas de temps tel que :

$$\begin{cases} s_{\ell+1} \in \mathbf{A}_d \\ (s_{\ell+1} - \widehat{s}_\ell) \in \Upsilon^- \end{cases}$$

/* Relaxation */

$s_{\ell+1} \leftarrow \mu_r s_{\ell+1} + (1 - \mu_r) s_\ell$

/* Indicateur d'erreur */

On évalue l'erreur ω_ℓ

1.3.3.2 Les grandes variantes

Toujours basées sur les trois principes précédents, plusieurs variantes de la méthode LATIN-PGD ont été étudiées. Ces différentes variantes se distinguent par trois choix principaux : la définition des espaces $\mathbf{A}_d$ et Γ, les variables considérées dans s et le choix des directions de recherche. Historiquement, trois grandes variantes ont émergé : une variante fonctionnelle avec $s = (\varepsilon, \sigma)$ [Ladevèze, 1985, Bussy *et al.*, 1990, Boisse *et al.*, 1991], une variante à variables internes $s = (\Xi, \Xi^*)$ [Cognard et Ladevèze, 1993, Ladevèze, 1999, Relun *et al.*, 2013] où Ξ^* représente les quantités duales des différentes variables internes Ξ et une variante pour les problèmes de contact [Ladevèze, 1999, Giacoma *et al.*, 2014, Oumaziz *et al.*, 2017] – ou de décomposition de domaines. On pourra se référer à [Scanff *et al.*, 2021] pour plus

de détails. Dans le cadre de ces travaux, on retient une approche fonctionnelle qui présente sous certains aspects de fortes similarités avec l'algorithme de Newton-Raphson comme nous le verrons au Chapitre-5.

1.3.3.3 Stratégie multi-paramétrique

Une des propriétés intéressantes de la méthode LATIN-PGD est de pouvoir initialiser l'algorithme avec n'importe quel champ $\boldsymbol{s}_0 \in \mathbf{A}_d$. Ainsi, lors d'études paramétriques, il est intéressant d'exploiter les calculs précédents pour accélérer la résolution : en donnant dès l'initialisation une approximation proche de la solution, quelques itérations supplémentaires de l'algorithme suffisent alors généralement pour capter la solution complète. Cette méthodologie a déjà été employée dans [Boucard et Champaney, 2003, Relun, 2011, Laurent, 2013, Nachar, 2019]. Plusieurs choix sont possibles pour définir l'initialisation d'un nouveau calcul (pour une nouvelle valeur des paramètres) à partir des solutions déjà calculées :

- à partir du plus proche voisin dans l'espace paramétrique – au sens d'une norme euclidienne par exemple ;
- grâce à une combinaison pondérée des points les plus proches dans l'espace paramétrique ;
- par interpolation dans la variété de Grassmann [Amsallem et Farhat, 2008].

Il convient de préciser que le choix des points procurant un cheminement optimal dans l'espace paramétrique, au sens de la minimisation du nombre d'itérations, est loin d'être trivial. Nous utilisons simplement cette propriété avantageuse lors des applications présentées au Chapitre-9, aucun nouveau développement n'étant envisagé dans le cadre de ce manuscrit.

1.4 Études fréquentes rencontrées en environnement industriel

Les systèmes industriels sont bien souvent complexes à mettre en œuvre car devant répondre à une multitude de contraintes imposées par le cahier des charges. Ainsi, que ce soit au niveau de la conception, de l'optimisation, de la validation ou de la maintenance prédictive des structures, il est courant de vouloir tester différentes configurations impliquant divers paramètres $\boldsymbol{\mu} \in \boldsymbol{\mathcal{D}}$. Ces derniers peuvent être de natures très variées : données matériaux entachées d'incertitude, température, conditions aux limites mal connues, modification de géométrie, etc. Des exemples faisant intervenir ces paramètres seront donnés au Chapitre-9. Hormis la température peut-être, la définition de l'ensemble de ces paramètres demeure fortement liée au problème d'intérêt, et ainsi, il n'est généralement pas possible de définir un espace paramétrique $\boldsymbol{\mathcal{D}}$ universel. Dit autrement, chaque application spécifique possède son propre jeu de paramètres d'influence. Forts de cette remarque, nous traiterons donc différemment ces paramètres $\boldsymbol{\mu}$ des autres grandeurs que sont l'espace et le temps, qui interviennent naturellement dans tout problème non-linéaire quasi-statique. C'est d'ailleurs ce qui est naturellement fait dans les codes de calculs industriels : à chaque calcul est exhibée une solution spatiale à certains instants (archivés) pour un jeu de paramètres $\boldsymbol{\mu}$ fixés. Il y a donc autant de calculs à réaliser que de valeurs de paramètres à tester ce qui conduit inexorablement à la malédiction de la dimensionnalité.

Différentes techniques permettent de réduire cette complexité paramétrique exponentielle : les méthodes des plans d'expériences, les méta-modèles et toute la famille des méthodes de réduction de modèles [Benner *et al.*, 2021]. Des logiciels d'exploration d'espace paramétrique comme HEEDS™, généralement utilisés en environnement industriel, conduisent à une exploration plus parcimonieuse de ces espaces. Cette recherche optimisée, au sens d'un certain critère, permet notamment de générer des surfaces de réponse, des abaques virtuels, exploitables par la suite en temps réel. Cependant, chaque calcul réalisé lors du processus automatisé de cheminement dans l'espace de conception reste bien souvent coûteux à mener, ce qui amène de plus en plus à se tourner vers d'autres outils comme les méthodes de réduction de modèles. Dans le Chapitre-2, nous donnons plus de détails quant aux techniques de réduction de modèles usuellement rencontrées en mécanique non-linéaire.

Chapitre 2

Méthodes de réduction de modèles pour la mécanique

Ce chapitre rappelle les différentes méthodologies usuellement rencontrées en matière de réduction de modèles. On présente tout particulièrement la gestion des non-linéarités ainsi que l'intérêt des méthodes de réduction de modèles pour l'industrie.

Sommaire

L'objectif des méthodes de réduction de modèles (ROM) est de limiter la complexité calculatoire des modèles mathématiques lors des simulations numériques. La notion de *réduction de modèles* peut prêter à confusion puisqu'il ne s'agit pas de trouver un modèle de substitution moins coûteux au modèle 3D (modèle 2D voire 1D ou encore de suppléer le modèle mathématique par des données au moyen de techniques de *Machine Learning* par exemple), mais bien de conserver toute la richesse du modèle mathématique. Seule la représentation des inconnues change : on cherche simplement à exhiber la solution du modèle – inchangé, riche – sous un format réduit de rang faible. En exploitant la redondance de l'information inhérente à la physique, on peut alors compresser cette information pour ne préserver que l'essentiel du comportement, les effets dominants, limitant d'autant les temps de calculs associés. De plus, l'approximation introduite au niveau de la représentation des inconnues dépend directement de l'ordre de troncature. Ces méthodes sont notamment d'une grande aide lors d'études paramétriques pour vaincre la malédiction de la dimensionnalité. L'extension à un grand nombre de paramètres reste encore une question ouverte [Paillet *et al.*, 2018]. Dans ce qui suit, nous rappelons brièvement les différentes méthodes ROM couramment utilisées en mécanique.

2.1 Approximation de rang faible

Comme évoqué au Chapitre-1, en mécanique du solide, les champs $\boldsymbol{u}$ recherchés dépendent généralement de l'espace $\boldsymbol{x} \in \Omega$, du temps $t \in I$ et de divers paramètres influents $\boldsymbol{\mu} \in \boldsymbol{\mathcal{D}}$ qui intéressent l'ingénieur lors des phases de conception ou de maintenance prédictive. Classiquement, on cherche la solution $\boldsymbol{u}$ dans l'espace éléments finis $\boldsymbol{\mathcal{U}}_n$ de dimension n finie. Généralement, n est grand voire très grand pour les problèmes industriels : $n \equiv \mathcal{O}(10^7)$. Cette solution, exhibée par tous les logiciels industriels par éléments finis, peut être considérée comme solution de référence – solution haute fidélité. L'idée générale des méthodes de réduction de modèles consiste à rechercher un espace de représentation adéquat de plus petite dimension, sous forme à variables séparées (2.1) – aussi appelée forme canonique – composée de $m \ll n$ modes :

$$\boldsymbol{u}(\boldsymbol{x}, t\,;\boldsymbol{\mu}) \approx \boldsymbol{u}_m(\boldsymbol{x}, t\,;\boldsymbol{\mu}) = \sum_{i=1}^{m} \boldsymbol{\Lambda}_i(\boldsymbol{x})\lambda_i(t) \prod_{k=1}^{p} \alpha_i^k(\mu_k) \tag{2.1}$$

avec p le nombre de paramètres $\boldsymbol{\mu} = (\mu_1, \mu_2, \cdots, \mu_p)$ dans l'espace $\boldsymbol{\mathcal{D}}$. Du point de vue des notations, nous utilisons le « point virgule » pour faire la différence entre les variables spatio-temporelles qui sont systématiquement impliquées dans n'importe quelle simulation et les paramètres $\boldsymbol{\mu}$ qui peuvent éventuellement venir s'ajouter. Par ailleurs, le champ $\boldsymbol{u}$ n'est pas nécessairement un champ vectoriel (déplacement par exemple) : une telle décomposition peut également s'employer pour des champs scalaires (plasticité cumulée) ou encore des champs tensoriels (tenseur des contraintes) en travaillant composante par composante.

Outre le format canonique (2.1) retenu dans le cadre de ces travaux, d'autres représentations de rang faible sont possibles pour représenter les champs à plusieurs variables ($\boldsymbol{x}$, t, $\boldsymbol{\mu}$) : on peut par exemple citer les formats de Tucker [Tucker, 1966], Tucker hiérarchique [Hackbusch et Kühn, 2009], TT (*Tensor-Train*) [Oseledets, 2011] ou encore tenseur de Kruskal (k-tensor) [Kolda et Bader, 2009]. Toutefois, ces différents formats ne seront pas discutés ici et de plus amples détails se trouvent dans [Hackbusch, 2012, Benner *et al.*, 2017].

La décomposition sous forme à variables séparées fonctionne généralement bien lorsque les variables demeurent peu couplées entre elles. Un nombre de modes m réduit permet alors de représenter la solution avec une bonne précision. En revanche, ce n'est de manière générale pas le cas des problèmes de transport, de convection ou encore de propagation d'ondes par exemple où les variables d'espace et de temps demeurent intimement couplées. Des techniques de changement de variables peuvent alors parfois permettre de retrouver une forme facilement séparable [Cagniart *et al.*, 2019]. Du point de vue mathématique, la notion d'*épaisseur de Kolmogorov* permet d'appréhender la plus ou moins bonne séparabilité du champ $\boldsymbol{u} \in \boldsymbol{\mathcal{U}_0}$ sous forme à variables séparées. Définie suivant (2.2), la m-épaisseur de Kolmogorov mesure la possibilité d'approximer le sous-espace vectoriel de dimension infinie $\boldsymbol{\mathcal{U}_0} \subset \boldsymbol{\mathcal{U}}$ par un sous-espace vectoriel $\boldsymbol{\mathcal{M}}_m$ de dimension m finie :

$$d_m(\boldsymbol{\mathcal{U}_0}) = \inf_{\boldsymbol{\mathcal{M}}_m \subset \boldsymbol{\mathcal{U}}} \left\{ \sup_{\boldsymbol{u} \in \boldsymbol{\mathcal{U}_0}} \left[\inf_{\boldsymbol{u}_m \in \boldsymbol{\mathcal{M}}_m} \|\boldsymbol{u} - \boldsymbol{u}_m\|_{\boldsymbol{\mathcal{U}}} \right] \right\} \tag{2.2}$$

Une épaisseur de Kolmogorov petite signifie que l'espace $\boldsymbol{\mathcal{M}}_m$ fournit une assez bonne connaissance de l'espace entier $\boldsymbol{\mathcal{U}_0}$. On espère alors obtenir une épaisseur d_m petite pour m assez petit : c'est la base des méthodes de réduction de modèles. Ces dernières fonctionnent particulièrement bien lorsque d_m décroit relativement rapidement à mesure que la dimension m de l'espace $\boldsymbol{\mathcal{M}}_m$ augmente. De manière qualitative, cela signifie que chaque nouveau mode ajouté dans la base génératrice de l'espace $\boldsymbol{\mathcal{M}}_m$ apporte une grande partie de l'information manquante pour décrire correctement l'espace $\boldsymbol{\mathcal{U}_0}$. La notion de m assez petit est bien sûr à mettre en regard de la dimension n de l'espace éléments finis : si $(\boldsymbol{\varphi}_i)_{1 \leqslant i \leqslant n}$ désigne les fonctions de forme éléments finis classiques qui engendrent l'espace $\boldsymbol{\mathcal{U}}_n = \text{Vect}\{(\boldsymbol{\varphi}_i)_{1 \leqslant i \leqslant n}\}$

et $\boldsymbol{\mathcal{M}}_m = \text{Vect}\{(\boldsymbol{\Lambda}_i)_{1\leqslant i\leqslant m}\}$, on espère avoir $\boldsymbol{\mathcal{M}}_m$ proche de $\boldsymbol{\mathcal{U}_0}$ pour $m \ll n$. C'est ce qui va permettre de réduire efficacement les temps de calculs.

En pratique, la recherche du sous-espace vectoriel $\boldsymbol{\mathcal{M}}_m$ optimal est infaisable puisque l'on ne sait pas calculer l'épaisseur de Kolmogorov d_m, hormis quelques cas particuliers. L'application des méthodes de réduction de modèles passe alors par des algorithmes susceptibles de fournir un sous-espace vectoriel proche du sous-espace optimal, comme par exemple l'algorithme glouton ($greedy$) que nous verrons par la suite.

Toutes les méthodes de réduction de modèles permettent d'exhiber une décomposition sous la forme d'une somme finie de fonctions à variables séparées (2.1), décomposition qui n'est pas unique. Les différentes méthodes de réduction de modèles découlent donc des diverses manières de faire pour aboutir à une telle décomposition.

2.2 Analyse d'un champ connu : Analyse en Composantes Principales

L'Analyse en Composantes Principales (PCA) est un des piliers de l'analyse de données [Pearson, 1901, Jolliffe, 1986] utilisée notamment à l'origine en statistique multivariée. En fonction des domaines d'application, et des différentes communautés scientifiques sous-jacentes, cette méthode s'assimile volontiers à la Décomposition de Karhunen-Loève (KLD) [Karhunen, 1947, Loève, 1955] ou encore à la Décomposition en Valeurs Singulières (SVD) [Golub et Van Loan, 2013]. En dimension finie, la PCA et la SVD sont d'ailleurs complètement équivalentes. La SVD est une opération classique des bibliothèques d'algèbre linéaire comme LAPACK [1] par exemple. Cette opération est donc aisément accessible dans tous les codes industriels.

La SVD peut être vue comme une généralisation aux matrices rectangulaires d'une décomposition aux valeurs propres de matrices carrées. Soit $\boldsymbol{X} \in \mathcal{M}_{q,r}(\mathbb{R})$ une matrice rectangulaire à coefficients réels. Quitte à travailler avec la transposée, on peut supposer que $q \geqslant r$. Effectuer la SVD de $\boldsymbol{X}$ (2.3) consiste à trouver les matrices $\boldsymbol{U} \in \mathcal{O}_q(\mathbb{R})$ et $\boldsymbol{V} \in \mathcal{O}_r(\mathbb{R})$ orthogonales, ainsi que la matrice diagonale $\boldsymbol{\Sigma} \in \mathcal{M}_{q,r}(\mathbb{R})$ telles que :

$$\boldsymbol{X} = \boldsymbol{U}\boldsymbol{\Sigma}\,{}^t\boldsymbol{V} \tag{2.3}$$

où, quitte à effectuer une renumérotation, on peut supposer que les valeurs singulières sont rangées par ordre décroissant $\sigma_1 \geqslant \sigma_2 \geqslant \cdots \geqslant \sigma_r \geqslant 0$ avec $\boldsymbol{\Sigma} = \text{diag}((\sigma_i)_{1\leqslant i\leqslant r})$. La plupart du temps en mécanique, il est courant de manipuler des matrices où $q \gg r$, c'est-à-dire qu'une des dimensions est beaucoup plus grande devant l'autre – en considérant pour q la dimension spatiale $n = \mathcal{O}(10^7)$ et pour r la dimension temporelle $n_t = \mathcal{O}(10^2)$ par exemple. Il est alors plus commode de résoudre un problème aux valeurs propres de plus petites dimensions. En effet, du fait de l'orthogonalité des matrices $\boldsymbol{U}$ et $\boldsymbol{V}$, réaliser la SVD de la matrice $\boldsymbol{X}$ se révèle être complètement équivalent à trouver les valeurs propres $\{\sigma_i^2\}_{1\leqslant i\leqslant r}$ de la matrice carrée ${}^t\boldsymbol{X}\boldsymbol{X} \in \mathcal{M}_{r,r}(\mathbb{R})$.

Bien souvent, comme la décroissance des valeurs propres est relativement rapide, il semble raisonnable de tronquer la décomposition à partir d'un certain rang. Considérer l'approximation de rang k (2.4) revient à tenir compte des k premières valeurs propres uniquement :

$$\boldsymbol{X} \approx \boldsymbol{X}_k = \sum_{i=1}^{k} \sigma_i \boldsymbol{U}_i \otimes \boldsymbol{V}_i \tag{2.4}$$

1. Linear Algebra PACKage : http://www.netlib.org/lapack/

avec $\{\boldsymbol{U}_i\}_{1\leqslant i\leqslant q}$ les vecteurs singuliers à gauche et $\{\boldsymbol{V}_i\}_{1\leqslant i\leqslant r}$ les vecteurs singuliers à droite. La décomposition en valeurs singulières n'est alors plus exacte mais la majeure partie de l'information peut être conservée, l'erreur de troncature étant directement corrélée à la décroissance du spectre de ${}^t\boldsymbol{X}\boldsymbol{X}$, soit la décroissance des valeurs singulières. Pour fixer les idées, prenons l'illustration de l'application de la SVD à une image de (2800×4200) pixels. La Figure 2.1 fournit la résolution obtenue en considérant différents rangs d'approximation où les sous-figures 2.1d à 2.1f représentent la carte d'erreur résultante par rapport à l'image de référence, en négatif (un pixel blanc signifie une erreur nulle, tandis qu'un pixel noir correspond à une erreur maximale). La Figure 2.2 illustre la décroissance des valeurs singulières. L'erreur de troncature commise (2.5) peut s'exprimer sous la forme :

$$e_{\text{svd}}^k = \frac{\sum_{i=k+1}^{r} \sigma_i}{\sum_{i=1}^{r} \sigma_i} \tag{2.5}$$

On observe qu'une centaine de modes suffit pour avoir une représentation visuellement satisfaisante. Ce nombre, relativement élevé, s'explique par une décroissance assez lente des valeurs singulières, liée aux nombreuses discontinuités présentes dans l'image. Cependant, les champs usuellement rencontrés en mécanique sont de manière générale bien plus réguliers du fait de la physique régularisante. Ainsi, hormis quelques exceptions comme les problèmes de contact ou les problèmes avec apparition de fissures, un nombre de modes relativement faible suffit la plupart du temps pour représenter convenablement la solution.

(a) $k = 5$ (b) $k = 20$ (c) $k = 150$

(d) $k = 5$ (e) $k = 20$ (f) $k = 150$

FIGURE 2.1 – Différents rangs k d'approximation d'une image de (2800×4200) pixels et cartes d'erreurs correspondantes : (a) – (d) $k = 5$, (b) – (e) $k = 20$, (c) – (f) $k = 150$

La SVD permet donc une compression de l'information dans la mesure où l'image initiale composée de (2800×4200) pixels nécessite un stockage de 11.76 MB tandis qu'une SVD tronquée au 150^e rang permet de ne stocker que 1.14 MB sans perte essentielle d'information. C'est l'idée principale reprise par la Proper Orthogonal Decomposition (POD) que nous détaillons par la suite : une compression de données *offline* avant une utilisation *online* efficace.

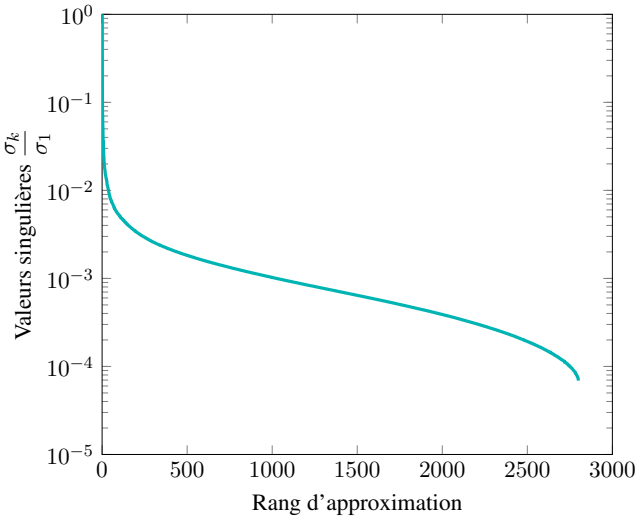


FIGURE 2.2 – Décroissance des valeurs singulières normées par rapport à σ_1

2.3 Résolution d'équations aux dérivées partielles paramétriques

Les méthodes de réduction de modèles [Benner *et al.*, 2021] ont déjà largement été utilisées pour la résolution des équations aux dérivées partielles paramétrées pour diverses applications en mécanique du solide [Kerfriden *et al.*, 2012, Ladevèze, 2014, Giacoma *et al.*, 2016, Fauque *et al.*, 2018], en biomécanique [González *et al.*, 2016, Quesada *et al.*, 2016, Zou *et al.*, 2018], en mécanique des fluides [Dumon *et al.*, 2011, Hesthaven et Ubbiali, 2018] ou encore pour les problèmes multi-physiques [Beringhier *et al.*, 2010, Néron et Ladevèze, 2010]. Parmi les méthodes les plus classiques rencontrées en mécanique, on trouve les méthodes Proper Orthogonal Decomposition (POD), Reduced-Basis (RB) et plus récemment Proper Generalized Decomposition (PGD). Usuellement, deux grandes catégories de méthodes se distinguent : les méthodes *a posteriori* (POD et RB) opposées aux méthodes *a priori* (PGD). Une première phase d'apprentissage *offline* – coûteuse – à partir de données, issues de l'exploration de l'espace paramétrique, caractérisent les méthodes *a posteriori*, tandis que l'exploitation du modèle réduit – peu coûteux – par projection des équations sur la base réduite n'intervient que dans une seconde phase dite *online*. A contrario, les méthodes *a priori* permettent de calculer directement la solution du problème d'intérêt sans nécessiter la résolution de problèmes connexes.

Bien que toutes ces méthodes puissent également intervenir dans le cadre des problèmes d'approximation – d'un champ connu –, nous nous intéressons plus spécifiquement à la résolution des équations aux dérivées partielles paramétriques, classiquement résolues par les logiciels éléments finis industriels.

2.3.1 Méthode Proper Orthogonal Decomposition (POD)

La méthode Proper Orthogonal Decomposition (POD) [Chatterjee, 2000, Rowley *et al.*, 2004], qui est parfois assimilée à la KLD ou à la PCA [Jolliffe, 1986], est largement utilisée en mécanique dont les premières applications se trouvent dès la fin des années soixante [Lumley, 1967]. En deux dimensions, la POD peut être vue comme une version continue de la SVD. Dans sa version discrète, et en considérant la norme $L^2(\boldsymbol{\mathcal{V}})$, la POD et la SVD sont équivalentes. On présente ci-après la POD-Galerkin dans sa version discrétisée éléments finis et on note $\langle \bullet, \bullet \rangle$ le produit scalaire dans $L^2(\boldsymbol{\mathcal{V}})$. L'idée consiste à chercher la solution $\mathbf{u}$ dans un espace $\boldsymbol{\mathcal{M}}_m = \mathrm{Vect}\{(\boldsymbol{\psi}_i)_{1 \leqslant i \leqslant m}\}$ de plus petite dimension (m) que l'espace éléments

finis $\boldsymbol{\mathcal{V}}$ (de dimension n) sous la forme (2.6) suivante :

$$\mathbf{u}(\boldsymbol{\xi}) = \sum_{i=1}^{m} a_i(\boldsymbol{\xi})\boldsymbol{\psi}_i \tag{2.6}$$

où par rapport à (2.1) on regroupe dans $\boldsymbol{\xi} \in I \times \boldsymbol{\mathcal{D}}$ l'ensemble des paramètres, y compris la variable temporelle.

Construction et analyse de données *offline*. La phase *offline* ou phase d'apprentissage consiste à construire la famille orthonormée génératrice de l'espace $\boldsymbol{\mathcal{M}}_m$. Pour ce faire, on se munit d'un ensemble de *snapshots* issus de données expérimentales, d'images ou de solutions haute fidélité calculées à partir d'un logiciel éléments finis par exemple. Chacun de ces *snapshots* correspond à une réalisation d'un point $(\boldsymbol{\xi}_j)_{1\leqslant j\leqslant \mathcal{N}}$ dans l'espace paramétrique avec $\mathcal{N}$ le nombre de tirages. C'est la phase coûteuse de la méthode. En concaténant l'ensemble des *snapshots*, on définit la matrice d'observation $\boldsymbol{X}$. Parmi l'ensemble de ces *snapshots*, on cherche ensuite à déceler la structure de rang faible sous-jacente, c'est-à-dire la meilleure approximation possible sous la forme (2.6) au sens d'une certaine norme – $L^2(\boldsymbol{\mathcal{V}})$ ici – en résolvant le problème de minimisation (2.7) suivant :

$$\{\boldsymbol{\psi}_i\}_{1\leqslant i\leqslant m} = \underset{\langle \boldsymbol{\psi}_i, \boldsymbol{\psi}_k\rangle = \delta_{ik}}{\arg\min} \sum_{j=1}^{\mathcal{N}} \left\| \mathbf{u}(\boldsymbol{\xi}_j) - \sum_{i=1}^{m} \langle \mathbf{u}(\boldsymbol{\xi}_j), \boldsymbol{\psi}_i \rangle \, \boldsymbol{\psi}_i \right\|^2_{L^2(\boldsymbol{\mathcal{V}})} \tag{2.7}$$

On peut montrer que la minimisation du problème (2.7) revient à maximiser une somme de quotients de Rayleigh et donc à résoudre un problème aux valeurs propres, de manière similaire à ce qui est fait lorsque l'on réalise une PCA ou une SVD, ce qui revient à :

- évaluer la matrice corrélation des *snapshots* $\boldsymbol{S} = {}^t\boldsymbol{X}\boldsymbol{X}$;
- résoudre le problème aux valeurs propres défini par $\boldsymbol{S}$ pour déterminer les couples $(\sigma_i^2, \boldsymbol{V}_i)_{1\leqslant i\leqslant \mathcal{N}}$ associés aux plus grandes valeurs propres ;
- trouver les vecteurs de base $\{\boldsymbol{\psi}_i\}_{1\leqslant i\leqslant m}$ par la formule d'équivalence (2.8) suivante :

$$\forall i \in [\![1, m]\!], \quad \boldsymbol{\psi}_i = \frac{1}{\sigma_i} \sum_{j=1}^{\mathcal{N}} (\boldsymbol{V}_i)_j \, \mathbf{u}(\boldsymbol{\xi}_j) \tag{2.8}$$

Utilisation du modèle réduit *online*. Cette seconde phase consiste à déterminer à la volée les coefficients $(a_i)_{1\leqslant i\leqslant m}$ de la décomposition (2.6) pour de nouveaux jeux de paramètres $\boldsymbol{\xi}$ non encore évalués. C'est la phase peu coûteuse en temps de calcul. On peut décider d'interpoler l'espace paramétrique (avec des fonctions de formes éléments finis par exemple) ou bien on peut projeter les équations du problème sur la base réduite afin d'obtenir un nouveau système d'équations bien moins coûteux à résoudre (de taille $(m \times m)$ au lieu de $(n \times n)$ avec généralement $m \ll n$). Bien que ce problème soit de petite taille, il demeure nécessaire d'assembler la matrice réduite qui généralement dépend de $\boldsymbol{\xi}$ (si données matériaux par exemple) en vue de l'inversion.

La méthode POD s'avère particulièrement efficace [Casenave *et al.*, 2020a, Quesada *et al.*, 2021] en permettant d'atteindre des gains en temps intéressants dès que l'utilisation du modèle réduit en phase *online* est réalisée de manière intensive ce qui permet de contrebalancer la première phase coûteuse d'apprentissage. Cependant, toute la difficulté réside dans le choix des *snapshots* : comment les choisir ? L'analyse de ces derniers par l'algorithmie POD permet de comprendre la structure de données sous-jacente, d'appréhender l'influence des paramètres sur la solution, et donc de construire une base réduite représentatrice des phénomènes physiques observés (avec une certaine précision). Tout manque d'information au travers des *snapshots* peut alors conduire à biaiser le modèle réduit.

2.3.2 Méthode Reduced-Basis (RB)

La méthode Reduced-Basis (RB) est tout comme la méthode POD une méthode de projection sur base réduite, avec apprentissage [Maday et Rønquist, 2004, Rozza, 2011]. Cependant, à la différence de la POD où les *snapshots* sont choisis plus ou moins au hasard, ces derniers sont choisis de manière efficace avec la méthode RB au moyen d'un algorithme glouton [Rozza *et al.*, 2007]. Une telle procédure permet de pallier un des inconvénients majeurs de la méthode POD, à savoir le choix des *snapshots* dont la pertinence affecte directement la qualité de la solution finale.

La technique se fonde sur un enrichissement progressif de la base réduite en choisissant à chaque fois le *snapshot* qui maximise l'erreur de projection de la solution $\mathbf{u}$ sur la base réduite déjà constituée, c'est-à-dire en rajoutant l'information manquante, qui est la plus mal représentée par la base réduite actuelle. En considérant un premier tirage aléatoire d'un point $\boldsymbol{\xi}^{(0)}$ dans l'espace paramétrique, on initialise l'algorithme. Puis, en supposant que l'on ait déjà évalué k points dans l'espace paramétrique, on cherche le point suivant $\boldsymbol{\xi}^{(k+1)}$ en choisissant le moins bien représenté par la base existante au sens d'une certaine norme (2.9) – énergétique par exemple :

$$\boldsymbol{\xi}^{(k+1)} = \underset{\boldsymbol{\xi} \in I \times \mathcal{D}}{\arg\max} \left\| \mathbf{u}(\boldsymbol{\xi}) - \Pi^k \mathbf{u}(\boldsymbol{\xi}) \right\| \tag{2.9}$$

avec $\Pi^k \mathbf{u}$ la projection de $\mathbf{u}$ sur la base déjà constituée. Généralement, la solution $\mathbf{u}$ sur l'ensemble de l'espace paramétrique n'est pas connue à l'avance. On a alors recours à un estimateur d'erreur pour évaluer la norme. Un choix possible pourrait être de considérer le résidu d'équilibre pour chacun des jeux de paramètres. Cet estimateur d'erreur est le point clé de la méthode : il doit être précis, garanti, robuste et peu coûteux en temps de calcul pour que la méthode RB puisse être performante.

Étant donné que les *snapshots* ont été choisis suivant un cheminement optimal dans l'espace paramétrique, on ne cherche généralement pas à extraire les modes propres comme avec la POD, mais on conserve directement les vecteurs de base exhibés pour constituer le modèle réduit. Le cas échéant, on peut toutefois reconstituer une base orthonormée par un processus de type Gram-Schmidt. Une fois la base réduite construite, son utilisation en phase *online* ne diffère pas de la méthode POD au sein de laquelle les équations aux dérivées partielles sont projetées sur la base réduite dans l'optique de résoudre un grand nombre de fois un problème de plus petite dimension beaucoup plus rapidement.

2.3.3 Méthode Proper Generalized Decomposition (PGD)

La méthode Proper Generalized Decomposition (PGD) est une technique introduite initialement sous la dénomination d'*approximation radiale* par P. Ladevèze dans les années quatre-vingt [Ladevèze, 1989] dans le cadre de la méthode LATIN pour traiter les problèmes non-linéaires matériaux dépendants du temps [Ladevèze, 1999]. Ce n'est que dans les années deux-mille-dix, sous l'élan de P. Chinesta et de P. Ladevèze que la méthode prend le nom de Proper Generalized Decomposition (PGD) pour souligner son caractère général [Chinesta et Ladevèze, 2014], aussi bien dans l'approximation (espace-temps-paramètres) obtenue que dans la variété des applications accessibles. Depuis, la méthode PGD a fait l'objet de nombreux développements : pour la résolution de problèmes multi-paramétriques [Chinesta *et al.*, 2010, Heyberger *et al.*, 2013], multi-échelles [Ammar *et al.*, 2012] ou multi-physiques [Beringhier *et al.*, 2010, Néron et Ladevèze, 2010], dans le cadre de problèmes inverses [González *et al.*, 2012], pour la recherche de solutions multiples [Beringhier *et al.*, 2016], ou encore pour appréhender des problèmes de validation [Bouclier *et al.*, 2013] ou d'assimilation de données [Rubio *et al.*, 2019] en temps réel. On pourra se référer à [Chinesta *et al.*, 2011, Chinesta et Ladevèze, 2020] pour plus de détails sur l'étendue de ces nombreuses applications.

Contrairement aux méthodes précédentes, la méthode PGD ne s'intéresse pas à un système d'équations réduit mais directement à la solution paramétrée (2.1) elle-même. Aucune connaissance *a priori* sur la solution, au même titre qu'aucune donnée précalculée (*snapshots*), ne sont requises ; seule la connaissance des équations aux dérivées partielles du modèle sont nécessaires dans la construction à la volée du modèle réduit. Contrairement à la POD, les vecteurs de base de l'espace réduit $\boldsymbol{\mathcal{M}}_m$ ne sont pas nécessairement recherchés orthogonaux entre eux, bien qu'en général un processus d'orthonormalisation de type Gram-Schmidt soit mis en œuvre pour rendre la décomposition plus optimale. Alors que la SVD garantit l'obtention d'une base optimale d'approximation (à une certaine précision fixée), avec la PGD, cette propriété n'est pas d'office assurée. Certains travaux portent justement sur la construction de bases réduites plus optimales [Giacoma *et al.*, 2015, Alameddin *et al.*, 2019] avec la PGD c'est-à-dire possédant un nombre de modes équivalents à ce que donnerait une décomposition SVD.

Par la suite, pour simplifier la présentation, nous illustrons la méthodologie induite par la méthode PGD sur la résolution d'un système linéaire (2.10) où la variable temporelle t correspond au paramètre du modèle :

$$ {}^{t}\mathbf{w}(t)\Big(\mathbf{K}_{\mathrm{T}}(t)\mathbf{u}(t) - \mathbf{f}(t)\Big) = \mathbf{0}, \quad \forall t \in I, \quad \forall \mathbf{w} \in \mathcal{I}_{\boldsymbol{\mathcal{V}}} \tag{2.10} $$

Ce système découle classiquement des équations aux dérivées partielles – après discrétisation spatiale et une éventuelle linéarisation – comme évoqué au Chapitre-1. L'approximation PGD consiste à chercher le champ solution $\mathbf{u}$ sous la forme d'une décomposition à variables séparées (2.11) du type :

$$ \mathbf{u}(t) \approx \mathbf{u}_m(t) = \sum_{i=1}^{m} \lambda_i(t)\boldsymbol{\Lambda}_i \tag{2.11} $$

C'est en particulier cette décomposition spatio-temporelle qui sera implémentée dans Simcenter Samcef™ dans ces travaux de thèse. Les autres paramètres seront traités différemment pour assurer un maximum de généralité et de robustesse. Toutefois, la philosophie de la méthode PGD demeure inchangée pour son extension à plusieurs paramètres $\boldsymbol{\mu} \in \boldsymbol{\mathcal{D}}$ comme illustré dans [Prulière *et al.*, 2010].

La construction des modes $\{(\lambda_i, \boldsymbol{\Lambda}_i)\}_{1 \leqslant i \leqslant m} \in \{\mathcal{I}_{\mathbb{R}} \times \boldsymbol{\mathcal{V}}\}^m$ dans l'approximation (2.11) s'effectue au moyen d'un algorithme glouton. On suppose connue l'approximation (2.12) au rang k :

$$ \mathbf{u}_k(t) = \sum_{i=1}^{k} \lambda_i(t)\boldsymbol{\Lambda}_i \tag{2.12} $$

Deux grandes étapes guident la résolution. Une première étape consiste à mettre à jour les fonctions temporelles uniquement, sans générer de nouveau mode. Chaque couple étant déterminé séquentiellement de manière totalement indépendante, cette étape de mise à jour représente un moyen simple d'obtenir une meilleure approximation du champ $\mathbf{u}$, ce qui peut conduire à réduire sensiblement le nombre de couples nécessaires dans la décomposition pour approximer $\mathbf{u}$ à une précision fixée. La seconde étape consiste à générer un nouveau mode en cherchant à apporter une correction de rang 1 à la base existante (2.13) sous la forme :

$$ \mathbf{u}_{k+1}(t) = \mathbf{u}_k(t) + \delta\mathbf{u}(t) = \mathbf{u}_k(t) + \lambda(t)\boldsymbol{\Lambda} \tag{2.13} $$

Que ce soit pour l'une ou l'autre des deux étapes, lors de la mise en œuvre, plusieurs grandes approches existent, chacune présentant ses avantages et ses inconvénients.

Galerkin progressif. La formulation Garlerkin progressive correspond à la formulation d'origine [Ladevèze, 1999]. Bien qu'aucune preuve de convergence n'existe dans le cas général, mais seulement pour quelques types d'équations spécifiques, elle s'avère la plupart du temps extrêmement efficace – lorsqu'elle

fonctionne – pour bon nombre d'applications. La recherche d'un nouveau mode passe par la résolution du problème (2.14) suivant :

$$\begin{aligned}&\text{Trouver } (\lambda, \boldsymbol{\Lambda}) \in \mathcal{I}_{\mathbb{R}} \times \boldsymbol{\mathcal{V}} \text{ tels que}\\ &\int_I {}^t\Big[\delta\lambda(t)\boldsymbol{\Lambda} + \lambda(t)\delta\boldsymbol{\Lambda}\Big]\Big[\mathbf{K}_{\mathrm{T}}(t)\Big(\mathbf{u}_k(t) + \lambda(t)\boldsymbol{\Lambda}\Big) - \mathbf{f}(t)\Big]\,\mathrm{d}t \;= 0, \quad \forall(\delta\lambda, \delta\boldsymbol{\Lambda}) \in \mathcal{I}_{\mathbb{R}} \times \boldsymbol{\mathcal{V}}\end{aligned} \tag{2.14}$$

Minimisation du résidu. Plus récente, la formulation en résidu a été développée pour traiter les cas où la formulation de Galerkin faisait défaut [Nouy et Ladevèze, 2004]. Cette seconde approche s'avère plus robuste, et une preuve mathématique de convergence peut être formulée. Toutefois, bien qu'une décroissance monotone de l'erreur soit assurée, le taux de convergence peut être médiocre dans certains cas. À cela s'ajoute le fait que cette approche demeure généralement plus coûteuse à mettre en place. Ainsi, on privilégie usuellement la formulation Galerkin lorsque celle-ci fonctionne. La recherche d'un nouveau mode passe par la résolution du problème de minimisation (2.15) suivant :

$$\begin{aligned}&\text{Trouver } (\lambda, \boldsymbol{\Lambda}) \in \mathcal{I}_{\mathbb{R}} \times \boldsymbol{\mathcal{V}} \text{ tels que}\\ &(\lambda, \boldsymbol{\Lambda}) = \underset{\substack{\lambda_{k+1}\in\mathcal{I}_{\mathbb{R}}\\ \boldsymbol{\Lambda}_{k+1}\in\boldsymbol{\mathcal{V}}}}{\arg\min} \Big\|\mathbf{K}_{\mathrm{T}}(t)\Big(\mathbf{u}_k(t) + \lambda_{k+1}(t)\boldsymbol{\Lambda}_{k+1}\Big) - \mathbf{f}(t)\Big\|^2\end{aligned} \tag{2.15}$$

En fonction du choix de la norme utilisée, cette minimisation de l'erreur en résidu peut se retrouver être complètement équivalente à la formulation de Galerkin. C'est en particulier le cas ici en considérant la norme énergétique mais, dans le cas général, ce n'est pas toujours possible de trouver une telle norme.

Petrov-Galerkin progressif. Cette troisième formulation diffère de la formulation de Galerkin (2.14) dans le sens où les fonctions tests sont cette fois prises dans un espace différent de celui des fonctions inconnues recherchées. Développée notamment dans [Nouy, 2010], cette approche permet d'atteindre un meilleur taux de convergence pour certains types de problèmes – équations d'advection-diffusion par exemple. La recherche d'un nouveau mode passe par la résolution du problème (2.16) suivant :

$$\begin{aligned}&\text{Trouver } (\lambda, \boldsymbol{\Lambda}) \in \mathcal{I}_{\mathbb{R}} \times \boldsymbol{\mathcal{V}} \text{ tels que}\\ &\int_I {}^t\Big[\delta\bar{\lambda}(t)\bar{\boldsymbol{\Lambda}} + \bar{\lambda}(t)\delta\bar{\boldsymbol{\Lambda}}\Big]\Big[\mathbf{K}_{\mathrm{T}}(t)\Big(\mathbf{u}_k(t) + \lambda(t)\boldsymbol{\Lambda}\Big) - \mathbf{f}(t)\Big]\,\mathrm{d}t \;= 0, \quad \forall(\delta\bar{\lambda}, \delta\bar{\boldsymbol{\Lambda}}) \in \mathcal{I}_{\mathbb{R}} \times \boldsymbol{\mathcal{V}}\end{aligned} \tag{2.16}$$

où les fonctions tests $(\bar{\lambda}, \bar{\boldsymbol{\Lambda}}) \in \mathcal{I}_{\mathbb{R}} \times \boldsymbol{\mathcal{V}}$ sont choisies pour préserver une certaine équivalence énergétique suivant (2.17). Le critère additionnel s'écrit :

$$\begin{aligned}&\text{Trouver } (\bar{\lambda}, \bar{\boldsymbol{\Lambda}}) \in \mathcal{I}_{\mathbb{R}} \times \boldsymbol{\mathcal{V}} \text{ tels que}\\ &\begin{cases} \displaystyle\int_I {}^t(\lambda(t)\delta\boldsymbol{\Lambda})\mathbf{K}_{\mathrm{T}}(t)(\bar{\lambda}(t)\bar{\boldsymbol{\Lambda}})\,\mathrm{d}t = \int_I {}^t(\lambda(t)\delta\boldsymbol{\Lambda})\mathbf{K}_{\mathrm{T}}(t)(\lambda(t)\boldsymbol{\Lambda})\,\mathrm{d}t & \forall\delta\boldsymbol{\Lambda} \in \boldsymbol{\mathcal{V}}\\ \displaystyle\int_I {}^t(\delta\lambda(t)\boldsymbol{\Lambda})\mathbf{K}_{\mathrm{T}}(t)(\bar{\lambda}(t)\bar{\boldsymbol{\Lambda}})\,\mathrm{d}t = \int_I {}^t(\delta\lambda(t)\boldsymbol{\Lambda})\mathbf{K}_{\mathrm{T}}(t)(\lambda(t)\boldsymbol{\Lambda})\,\mathrm{d}t & \forall\delta\lambda \in \mathcal{I}_{\mathbb{R}}\end{cases}\end{aligned} \tag{2.17}$$

Les équations (2.16) et (2.17) peuvent également être vues comme les conditions de stationnarité d'une certaine fonctionnelle $\mathcal{L} : (\mathcal{I}_{\mathbb{R}} \otimes \boldsymbol{\mathcal{V}}) \times (\mathcal{I}_{\mathbb{R}} \otimes \boldsymbol{\mathcal{V}}) \to \mathbb{R}$, dont l'expression est donnée par la formule (2.18) suivante :

$$\mathcal{L}(\boldsymbol{w}, \bar{\boldsymbol{w}}) = \frac{1}{2}\int_I {}^t\boldsymbol{w}(t)\mathbf{K}_{\mathrm{T}}(t)\boldsymbol{w}(t)\,\mathrm{d}t - \int_I {}^t\bar{\boldsymbol{w}}(t)\Big[\mathbf{K}_{\mathrm{T}}(t)(\mathbf{u}_k(t) + \boldsymbol{w}(t)) - \mathbf{f}(t)\Big]\,\mathrm{d}t \tag{2.18}$$

Les quantités $(\lambda, \boldsymbol{\Lambda}, \bar{\lambda}, \bar{\boldsymbol{\Lambda}}) \in (\mathcal{I}_{\mathbb{R}} \times \boldsymbol{\mathcal{V}})^2$ découlent alors du problème (2.19) défini par :

$$(\lambda, \boldsymbol{\Lambda}, \bar{\lambda}, \bar{\boldsymbol{\Lambda}}) = \underset{\substack{\bar{\lambda}_{k+1}\in\mathcal{I}_{\mathbb{R}}\\ \bar{\boldsymbol{\Lambda}}_{k+1}\in\boldsymbol{\mathcal{V}}}}{\arg\max}\Big(\min_{\substack{\lambda_{k+1}\in\mathcal{I}_{\mathbb{R}}\\ \boldsymbol{\Lambda}_{k+1}\in\boldsymbol{\mathcal{V}}}} \mathcal{L}(\lambda_{k+1}\boldsymbol{\Lambda}_{k+1}, \bar{\lambda}_{k+1}\bar{\boldsymbol{\Lambda}}_{k+1})\Big) \tag{2.19}$$

En ce sens, cette formulation est aussi parfois appelée Minimax PGD.

Il est également possible de mener des approches hybrides : les fonctions spatiales peuvent être recherchées en suivant une approche Galerkin tandis que les fonctions temporelles le sont suivant une formulation en résidu [Bhattacharyya *et al.*, 2018]. Cette manière de procéder permet notamment de limiter la complexité calculatoire lors de la résolution du sous-problème spatial.

Dans tous les cas, cela conduit à la résolution simultanée d'au moins deux sous-problèmes couplés, résolubles par un algorithme de type point fixe. Dans la pratique, seules quelques itérations sont nécessaires pour la résolution du point fixe : une convergence stricte n'est pas de mise dans la mesure où la convergence s'effectue relativement rapidement et que l'ajout du couple suivant pourra venir corriger l'éventuelle erreur commise à cette étape.

2.4 Modèles réduits et non-linéarités

L'ensemble des méthodes ROM fonctionnent très bien pour résoudre des problèmes linéaires paramétrés, lorsqu'une dépendance affine aux paramètres existe. Cependant, la gestion de la perte d'affinité, ou plus généralement la présence de non-linéarités de quelque nature que ce soit, et le développement de méthodes ROM non-intrusives demeurent des challenges actuels qui ont fait très récemment – et font encore – l'objet de nombreux travaux. Dans la suite de ce paragraphe, nous rappelons succinctement les méthodes existantes pour la gestion des non-linéarités. Le caractère non-intrusif sera abordé au Chapitre-3.

En présence de non-linéarités, deux difficultés majeures apparaissent : (i) il faut trouver un moyen de linéariser les équations afin de se ramener à une succession de problèmes linéaires aisément résolubles ; (ii) il faut s'assurer que l'évaluation des termes non-linéaires soit indépendante de la dimension du modèle haute-fidélité sous peine de compromettre l'efficacité de l'utilisation du modèle réduit.

2.4.1 Linéarisations courantes

Conjointement aux méthodes POD et RB, la méthode de Newton-Raphson est usuellement choisie pour résoudre le cas échéant le système d'équations non-linéaires projetées sur la base réduite, comme illustré par de nombreux travaux [Kerfriden *et al.*, 2011, Radermacher et Reese, 2016, Hernández *et al.*, 2017, Casenave *et al.*, 2020b]. Concernant la résolution de problèmes non-linéaires avec la PGD, l'utilisation d'un algorithme de Newton-Raphson est largement moins répandue. Seuls de rares travaux récents existent à ce sujet comme [Hoareau et Deü, 2019] qui s'adresse à des problèmes d'interaction fluide-structure dans des réservoirs sous pression partiellement remplis de liquides avec prise en compte des forces suiveuses et des non-linéarités géométriques, ou [Ma et Shen, 2021] pour la résolution des équations de Allen-Cahn et Cahn-Hilliard qui régissent les phénomènes de changement de phase et de solidification.

D'autres choix ont été étudiés dans la littérature comme par exemple les Méthodes Asymptotiques Numériques (ANM) [Cochelin *et al.*, 1994]. En procédant par techniques d'homotopie et de perturbation, les méthodes asymptotiques numériques consistent à transformer le problème non-linéaire en une succession de problèmes linéaires par développement sous forme de séries de Taylor des différentes variables. On peut se référer à [Najah *et al.*, 1998] pour une brève revue sur ces méthodes. Dans [Chinesta *et al.*, 2013], les auteurs combinent les méthodes PGD et ANM dans la résolution d'équations thermiques non-linéaires où l'un des avantages réside dans la mutualisation de l'opérateur différentiel au cours des phases successives d'enrichissement. On retrouve également ce formalisme PGD-ANM pour traiter des problèmes d'hyper-élasticité [Niroomandi *et al.*, 2013].

La méthode LATIN, une alternative possible au schéma de Newton-Raphson en tant que solveur non-linéaire, est de fait beaucoup plus fréquemment utilisée avec la PGD [Ladevèze, 2016]. Cette vaste littérature, rappelée en Annexe A, s'explique notamment par le fait que la méthode PGD – sous le nom d'*approximation radiale* initialement – a été développée pour fonctionner spécifiquement avec la méthode LATIN. Le choix qui a été fait est de poursuivre dans cette lignée en introduisant le solveur non-linéaire LATIN-PGD dans le logiciel industriel Simcenter Samcef™ pour bénéficier des outils de réduction de modèle. De multiples contraintes doivent néanmoins être satisfaites comme nous le verrons au Chapitre-3.

2.4.2 Hyper-réduction

Les techniques d'hyper-réduction apportent une réduction supplémentaire. Quelle que soit la méthode ROM retenue, ces techniques d'hyper-réduction basées sur des processus d'interpolation plus ou moins complexes permettent de s'affranchir complètement des dimensions du problème initial dans la conduite des différentes opérations algébriques. Cette réduction supplémentaire intervient généralement pour traiter les opérations coûteuses sur les paramètres (intégrales par exemple) ou lors du traitement de problèmes non-linéaires. Le terme d'*hyper-réduction* provient historiquement de [Ryckelynck, 2005] et de nombreuses autres techniques ont par la suite été développées. Sans volonté d'exhaustivité, on peut par exemple citer les méthodes :

- Empirical Interpolation Method (EIM) [Barrault *et al.*, 2004, Grepl *et al.*, 2007] ;
- Best Point Interpolation Method (BPIM) [Nguyen *et al.*, 2008b] ;
- Discrete Empirical Interpolation Method (DEIM) [Chaturantabut et Sorensen, 2010] ;
- Gauss-Newton with Approximated Tensors (GNAT) [Carlberg *et al.*, 2013] ;
- Energy Conserving Sampling and Weighting (ECSW) [Farhat *et al.*, 2014] ;
- Generalized Empirical Interpolation Method (GEIM) [Casenave *et al.*, 2016] ;
- Missing Point Estimation (MPE) [Astrid *et al.*, 2008, Zimmermann et Willcox, 2016] ;
- Empirical Cubature Method (ECM) [Hernández *et al.*, 2017] ;
- Reference Point Method (RPM) [Capaldo *et al.*, 2017] ;
- Linear Program Empirical Quadrature Procedure (LP-EQP) [Yano et Patera, 2019].

Introduite dans [Ladevèze, 1997], et développée dans [Capaldo *et al.*, 2017], la méthode RPM fonctionne bien avec la méthode LATIN-PGD. La méthode consiste, pour chaque quantité d'intérêt, à stocker l'information en un certain nombre de points et d'instants de référence. La quantité d'intérêt pourra, par la suite, être reconstruite sur $\Omega \times I$ par une formule analytique ce qui permet de limiter grandement le coût de reconstruction. La méthode RPM diffère de la technique plus répandue, mais aussi plus sophistiquée, qu'est l'Empirical Interpolation Method introduite dans [Barrault *et al.*, 2004]. Toutefois, combinée à la méthode LATIN-PGD, la méthode RPM présente l'avantage de préserver la convergence quel que soit le nombre de points et d'instants de référence choisi [Ladevèze, 2016], en augmentant éventuellement le nombre de modes PGD calculés. Cette méthode n'a pas fait l'objet de nouveaux développements dans le cadre des travaux présentés ici, mais figure parmi les perspectives.

2.5 Intérêts des modèles réduits pour l'industrie

Les modèles réduits possèdent un fort potentiel attractif vis-à-vis des industriels pour de nombreuses raisons. Pour l'ingénieur en bureaux d'études, la conception de nouvelles structures toujours plus innovantes passe par la nécessité de tester toute une panoplie de configurations possibles – régies par un ensemble de paramètres $\boldsymbol{\mu} \in \boldsymbol{\mathcal{D}}$. Chacune de ces configurations demande du temps : pour la mise en données, pour le calcul, pour le post-traitement et l'exploitation des résultats. À cela s'ajoute le temps nécessaire pour dialoguer entre les différentes équipes que ce soit entre les différents experts techniques (dans le

cadre du travail collaboratif au profit du design de systèmes complexes pour la plupart multi-physiques) ou bien pour faire remonter une information pertinente aux personnes susceptibles d'acter les choix stratégiques du développement du produit. L'élaboration de modèles réduits permet d'accélérer grandement cette chaîne de valeur en permettant d'accélérer les temps de calculs, voire être dans la capacité de tester beaucoup plus de configurations dans un temps donné, mais également de générer des abaques virtuels débouchant sur des décisions rapides et simplifiées entre tous les acteurs.

L'aide à la conception de nouveaux produits n'est qu'une illustration possible des avantages des modèles réduits parmi d'autres. Un deuxième exemple pourrait être à destination de la maintenance prédictive des structures par le biais de jumeaux numériques. L'avénement des outils informatiques et la nécessité de procéder à un usage raisonné des ressources et de l'énergie ont conduit à de nouveaux paradigmes. Que ce soit pour raison de sécurité ou bien pour raison écologique, des modèles économiques complets se fondent dorénavant sur des programmes de maintenance plutôt que sur le remplacement systématique par un produit neuf dès qu'une avarie est constatée. À ce titre, les jumeaux numériques sont un outil en plein essor qui pourrait permettre de suivre l'état individualisé de chaque structure au cours de son cycle de vie.

Pour toutes ces raisons, les éditeurs de logiciels souhaitent être en mesure de fournir des solutions intégrées fiables et robustes auprès des nombreux clients utilisateurs. Cependant, de nombreux obstacles se dressent encore comme nous allons le voir au Chapitre-3.

Chapitre 3

Limites au rayonnement des méthodes ROM dans les codes industriels

Ce chapitre esquisse un panorama des différents verrous persistants quant à la diffusion des méthodes de réduction de modèles dans les codes industriels. On s'attache à analyser les solutions existantes avant de décrire la méthodologie retenue dans le cadre de ces travaux.

Sommaire

Bien que les méthodes de réduction de modèles aient été largement investiguées ces dernières années dans le monde de la recherche académique, leur déploiement industriel demeure encore relativement limité – malgré un fort engouement pour les bénéfices qu'elles peuvent apporter. En particulier, pour ne pas rester uniquement dans le giron de quelques experts, les méthodes ROM devraient être accessibles plus largement à l'ensemble des utilisateurs en bureaux d'études. Cependant, faute d'une intégration effective dans les logiciels industriels commerciaux par éléments finis, ces méthodes ne font pas encore partie des outils des ingénieurs au quotidien.

3.1 Points de blocage

De nombreux points de blocage freinent la diffusion des méthodes ROM dans les codes industriels. Outre la nécessité d'avoir une connaissance parfaite de l'architecture interne du logiciel, un des freins majeurs se situe au niveau de l'investissement (sur le plan humain et financier) à mettre en œuvre pour disposer d'une solution de réduction de modèles fiable et performante. Un logiciel tel que Simcenter Samcef™ résulte de plus de cinquante années de développements, ce qui en fait un outil certes très riche mais tout aussi complexe à manipuler. Il n'est alors pas concevable de toucher à l'architecture cœur du logiciel, sous peine de prendre le risque de détériorer la robustesse du code, de même qu'il n'est pas souhaitable de devoir repasser une dizaine d'années à valider et recertifier une nouvelle architecture. Il semble alors préférable de se tourner vers des approches *non-intrusives* [Giraldi *et al.*, 2014, Giraldi *et al.*, 2015] susceptibles de préserver les outils existants. Du point de vue technique, les méthodes ROM engendrent des opérations et des manipulations de données atypiques pour les principaux logiciels industriels.

3.1.1 Opérations et opérateurs atypiques

Comme nous l'avons vu au Chapitre-2, les méthodes ROM requièrent de manière générale des opérateurs d'intégration sur le temps I – voire sur l'ensemble de l'espace paramétrique $I \times \boldsymbol{\mathcal{D}}$ –, la résolution de problèmes de minimisation, ou encore des algorithmes de type point fixe. Ces opérateurs découlent d'opérations qui ne sont pas traitées en standard dans les logiciels industriels et demandent donc un effort d'intégration particulier. À cela s'ajoute également le choix des quadratures dans les méthodes d'hyper-réduction, en plus du processus d'interpolation sous-jacent permettant de faire le lien entre les quantités réduites, connues uniquement aux points de quadrature, et les quantités souhaitées sur l'ensemble du domaine d'étude.

3.1.2 Manipulation de données

Un autre point délicat réside dans la manipulation et le stockage des données : l'utilisation des méthodes ROM nécessite d'avoir accès à l'information partagée non plus uniquement sur l'espace mais également sur le temps, voire sur l'ensemble de l'espace paramétrique. À titre d'exemple, le système d'équations (2.10) impliqué dans la méthode PGD demande de stocker l'information sur l'ensemble de l'espace et l'ensemble de l'intervalle de temps avant de mener à bien la résolution. Ce stockage n'est pas aisé du fait de l'architecture souvent ancienne des logiciels dont les routines principales restent encore, pour la plupart, en grande partie codées dans des langages de programmation ne possédant pas d'allocation dynamique de la mémoire – tels que le Fortran 77. Aussi, du fait d'une structure incrémentale régie par l'algorithme classique de Newton-Raphson, les espaces mémoires (statiques) se limitent couramment à des tableaux de taille n, la dimension de l'espace $\boldsymbol{\mathcal{V}}$ éléments finis. Par ailleurs, il n'est pas souhaitable de stocker des tableaux de taille $n \times n_t$ complets au risque de faire exploser très rapidement les ressources mémoires nécessaires à la résolution. La compression de l'information sous forme à variables séparées, inhérente aux méthodes ROM, représente un point clé permettant d'introduire ces méthodes dans les logiciels industriels sans toutefois dégrader drastiquement les ressources mémoires.

3.2 Analyse des solutions existantes

Ces dernières années, de nombreux auteurs ont contribué au développement de méthodes ROM non-intrusives. La plupart des travaux se concentrent sur les méthodes *a posteriori* (POD ou RB) [Audouze *et al.*, 2013, Casenave *et al.*, 2014, Hammond *et al.*, 2019, Casenave *et al.*, 2020a, Vizzaccaro *et al.*, 2020, Kadeethum *et al.*, 2021] consistant à générer les *snapshots* en boîte noire – *black-box*. Concernant la PGD, intrinsèquement plus intrusive du fait de sa formulation *a priori*, les développements demeurent bien moins fréquents. Récemment, des efforts ont toutefois été produits pour édifier une formulation PGD moins intrusive, suivant principalement deux axes majeurs : l'utilisation en boîte noire reprenant l'idée des approches de type *snapshots* [Chinesta *et al.*, 2020] et l'interfaçage avec des codes industriels pour des applications spécifiques [Courard *et al.*, 2016].

3.2.1 Utilisation en boîte noire

L'idée d'utiliser une approche de type boîte noire pour explorer l'espace paramétrique $\boldsymbol{\mathcal{D}}$ permet de générer aisément les *snapshots* par n'importe quel logiciel industriel disponible pendant la phase d'apprentissage, tandis que l'extraction du modèle réduit et, par la suite, son utilisation peuvent être usuellement menées dans des codes *ad-hoc* externes. Le développement de ces derniers nécessite toutefois bien souvent un effort particulier – qui peut s'avérer fastidieux. Basées sur cette idée de *snapshots*, les méthodes PGD non-intrusives Sparse Subspace Learning (SSL) [Borzacchiello *et al.*, 2019, Leon *et al.*, 2019] et sparse-PGD (sPGD) [Ibáñez *et al.*, 2018, Ghnatios *et al.*, 2021a] ont été développées.

Ces approches fonctionnent généralement bien dès que l'on est en capacité de produire suffisamment de *snapshots* pour extraire une base réduite représentative. Cependant, en environnement industriel, le nombre de *snapshots* accessible demeure bien souvent limité du fait des temps de calculs prohibitifs de certains modèles. Ainsi, on s'autorise usuellement un certain budget temps pour générer ces *snapshots* et on espère que ce sera suffisant pour emmagasiner la majeure partie de l'information. À ce titre, les espaces paramétriques de grandes dimensions, souhaités par bon nombre de clients, se retrouvent souvent inaccessibles. C'est en ce sens qu'a été développée la méthode sPGD permettant de circonscrire l'évolution exponentielle dans l'exploration de l'espace paramétrique.

3.2.2 Interfaçage spécifique avec des codes industriels

Dans le même temps, un autre type d'approche a été étudié, pour des applications spécifiques. Pour élaborer une formulation PGD non-intrusive, l'idée repose sur l'utilisation externe de logiciels industriels dans la méthodologie de résolution – pour adresser le problème spatial coûteux dans l'algorithme du point fixe en particulier. Pour des problèmes d'optimisation topologique, on trouve dans [Courard *et al.*, 2016] l'idée préliminaire d'introduire la PGD dans le logiciel industriel Simcenter Samcef™ lui-même mais cette idée n'a pas été retenue à l'époque car jugée trop compliquée à mettre en œuvre au regard de l'architecture du code, comme évoqué en Section-3.1. Cela a donc donné lieu à l'encapsulation du logiciel Simcenter Samcef™ dans un code Matlab® externe consacré à la manipulation des différentes opérations impliquées dans la résolution PGD. Cette idée de faire appel à un logiciel industriel pour réaliser une brique de la résolution globale a plus tard été reprise dans [Zou *et al.*, 2018] en bio-mécanique, pour traiter des problèmes linéaires élastiques paramétrés avec Abaqus™, et plus récemment dans [Tsiolakis *et al.*, 2020] pour la résolution des équations de Navier-Stokes laminaires incompressibles paramétrées avec OpenFOAM®.

Ainsi, dans tous ces travaux, la PGD demeure externe au logiciel industriel et requiert un effort d'interfaçage avec d'autres logiciels tels que Matlab® ou Python™, plus propices aux développements parce que plus accessibles. Toutefois, l'idée d'introduire une version PGD non-intrusive directement dans un logiciel industriel commercial semble maintenant mature, de sorte que l'on ne se limite plus à des applications spécifiques, mais que l'on dispose d'un cadre beaucoup plus général et robuste, valable pour tous types de non-linéarités. La Section-3.3 suivante apporte un éclairage plus détaillé à ce sujet.

3.3 Modèles réduits : vers une généralisation dans les logiciels industriels

Ces dernières années, les méthodes ROM ont permis de répondre à bon nombre de besoins portant sur des applications spécifiques. Le niveau de maturité atteint et la montée en compétence sur ce type de méthodes de la part des acteurs du monde industriel font que les concepteurs de logiciels commerciaux souhaitent s'emparer de la question afin de pouvoir proposer des solutions fiables et robustes pour des pans entiers de secteurs industriels. En effet, que ce soit pour la construction ou l'exploitation de modèles réduits, les méthodes ROM ne peuvent se diffuser largement à l'échelle industrielle que si elles s'intègrent harmonieusement dans les processus industriels préétablis de longue date, et dont les contraintes économiques imposent une intégration la plus aisée possible dans les habitudes des bureaux d'études.

Par la même occasion, les paradigmes évoluent également : on passe de la stricte nécessité d'avoir des approches non-intrusives – ne pouvant faire autrement – à la volonté de développer des approches faiblement intrusives. En ayant accès aux codes sources de logiciels industriels, on pourrait apporter toutes les modifications nécessaires, mais les contraintes technico-économiques, dictées par l'architecture logicielle historique, entravent fortement ces possibilités. L'idée consiste à minimiser le nombre de changements à apporter dans le logiciel industriel lors de l'incorporation de ces nouvelles méthodes de réduction de

modèles afin de conserver la capitalisation de tous les développements passés.

Comme nous l'avons évoqué au Chapitre-2, dès que l'on traite des problèmes non-linéaires, il convient de se munir au préalable d'un schéma permettant de linéariser les équations. Dans le cadre de ces travaux, nous proposons d'introduire la méthode LATIN-PGD non-linéaire dans le logiciel industriel Simcenter Samcef™. Le choix d'implanter le solveur LATIN non-incrémental découle du constat suivant : dès que l'on s'intéresse à des problèmes d'évolution non-linéaires, c'est-à-dire impliquant la variable temporelle, alors il n'est possible de mettre en place la PGD – pour générer des modes à la volée – qu'en connaissant une première approximation de la solution sur tout l'intervalle de temps, ce qui n'est pas réalisable avec des solveurs incrémentaux classiques avant d'atteindre la convergence. Le choix d'introduire – en interne – ce nouveau solveur LATIN non-incrémental dans une architecture incrémentale constitue le seul point intrusif de notre approche, toutes les actions propres à la PGD pouvant être interfacées dans des parties bien séparées du code existant comme nous le verrons au Chapitre-8. Plus précisément, l'un des objectifs de cette approche consiste à définir une interface au sein du logiciel – comme une API [1] – afin que la méthode LATIN-PGD puisse être intégrée de manière transparente dans les logiciels éléments finis industriels avec des modifications mineures.

De plus, pour rester le plus généraliste possible, nous avons choisi de ne construire que des modes spatio-temporels via la méthode LATIN-PGD. L'espace et le temps sont de fait des quantités universelles intervenant dans toute simulation numérique, ce qui n'est pas le cas des autres paramètres $\boldsymbol{\mu} \in \boldsymbol{\mathcal{D}}$ pouvant varier au gré des modèles. Ces derniers sont ainsi traités via une approche *snapshots* classique à partir de la bibliothèque de modes spatio-temporels établie : à chaque nouveau jeu de paramètres $\boldsymbol{\mu}$ correspond une famille de modes espace-temps. À cette fin, une propriété intéressante de la méthode LATIN-PGD est mobilisée : celle de pouvoir initialiser l'algorithme avec n'importe quelle solution déjà calculée – pour un autre jeu de paramètres par exemple – comme évoqué en Section-1.3.3.3. Il est alors envisageable de remonter à une expression complète sous forme à variables séparées (2.1) en mettant en œuvre une seconde PGD paramétrique basée sur la concaténation de l'ensemble de cette bibliothèque de modes.

Par le passé, il y a déjà eu la volonté d'introduire la méthode LATIN-PGD dans des codes industriels. On peut par exemple citer le code prototype semi-industriel COFAST, basé sur l'approche CONTRAST, développé par L. Champaney [Champaney *et al.*, 1999] et dédié aux calculs d'assemblages de structures avec contacts frottants résolus avec la méthode LATIN mono-échelle dans Cast3M®. Ce logiciel de recherche avait notamment été utilisé par le CEA et EADS dans le cadre de contrats de recherche aérospatiaux. Toutefois, c'est la première fois que la méthode LATIN-PGD est intégrée à un aussi haut niveau dans un logiciel industriel généraliste comme Simcenter Samcef™ permettant d'adresser, non pas une application spécifique, mais n'importe quel type de non-linéarités, n'importe quel type d'éléments, et plus généralement l'ensemble des fonctionnalités déjà présentes dans le logiciel industriel.

⋆ ⋆
⋆

1. Application Programming Interface

Deuxième partie

Contributions à la diffusion des méthodes ROM dans l'industrie

Chapitre 4

Outils logiciels pour le développement des modèles réduits

Ce chapitre a vocation à présenter les outils logiciels qui ont été utilisés ou développés au cours de la thèse. On introduit en particulier le logiciel industriel Simcenter Samcef™ au cœur de ces travaux en mobilisant la méthode LATIN-PGD faiblement intrusive. Au préalable, un démonstrateur semi-industriel – ROMlab – a également été développé afin de pouvoir simuler en amont les différentes contraintes d'intégration auxquelles on serait susceptible de faire face.

Sommaire

4.1 Logiciel Simcenter Samcef™

Les premiers développements de SAMCEF [1] – aujourd'hui Simcenter Samcef™ – datent de 1965, au sein du Laboratoire des Techniques Aéronautiques et Spatiales de l'Université de Liège. Depuis 1986, c'est la société SAMTECH, basée en Belgique, qui lance la commercialisation du logiciel et poursuit les développements avant son rachat, en 2012, par Siemens.

4.1.1 Présentation du logiciel industriel

Le logiciel industriel Simcenter Samcef™ est le solveur thermo-mécanique non-linéaire appartenant à la grande famille de solutions Simcenter 3D™. Cette dernière est une plateforme qui fournit tout un ensemble de solutions de simulations CAE [2] regroupant, en un même lieu, les outils de pré-traitement, d'analyse et de post-traitement nécessaires, ainsi que des solutions d'exploration d'espace de conception et de gestion de données. Concrètement, l'élaboration de la géométrie, le maillage, l'ensemble de la mise

1. Système pour l'Analyse des Milieux Continus par Eléments Finis
2. Computer-Aided Engineering

en données du problème peuvent être réalisés grâce à l'interface graphique de Simcenter 3D™. Ces modèles peuvent ensuite être résolus à l'aide d'un ensemble de solveurs dédiés. Enfin, les résultats obtenus peuvent être visualisés directement dans l'interface graphique de Simcenter 3D™, ce qui permet de réaliser des simulations complètes, de bout en bout, dans un seul et même environnement.

Simcenter Samcef™ est un logiciel industriel généraliste par éléments finis capable d'adresser tous types de non-linéarités : matérielles, géométriques et conditions de contact comme présenté au Chapitre-1. Plus précisément, il est organisé en divers modules permettant de simuler différentes physiques allant des problèmes mécaniques non-linéaires, aux machines tournantes à grande vitesse, en passant par les mécanismes articulés ou encore la simulation thermique. Comme l'ensemble de ces modules font partie intégrante d'une seule et même famille – sous l'égide d'une architecture logicielle commune –, il est possible de passer facilement d'une analyse à une autre, de combiner des analyses ou encore de réaliser des co-simulations. Dans le cadre de ces travaux, le choix qui a été fait est de se concentrer plus spécifiquement sur le module dédié au solveur mécanique non-linéaire ; bien que notre approche, par sa généralité, puisse s'appliquer à d'autres modules (thermique par exemple) ou d'autres logiciels industriels, avec des modifications mineures. Dans la suite, nous présentons succinctement le fonctionnement général du logiciel Simcenter Samcef™ autour de l'algorithme de Newton-Raphson.

4.1.2 Grandes étapes de résolution

Au Chapitre-1 nous avons décrit l'algorithme de Newton-Raphson (Algorithme 1) usuellement employé dans la plupart des logiciels industriels traitant des problèmes non-linéaires. La Figure 4.1 reprend les mêmes étapes sous forme d'un schéma que nous compléterons au Chapitre-8 avec les détails pratiques d'implémentation de la méthode LATIN-PGD au sein du logiciel industriel. Partant de la mise en données d'un modèle (fichier *.dat*), on passe successivement par trois grandes étapes lors de la résolution :

- l'évaluation des différentes non-linéarités permettant de calculer le résidu d'équilibre $\mathcal{R}$ (1.26) ;
- la résolution du système linéarisé (1.27) permettant de trouver la correction à apporter ;
- le calcul d'indicateurs d'erreur permettant d'estimer l'état de convergence.

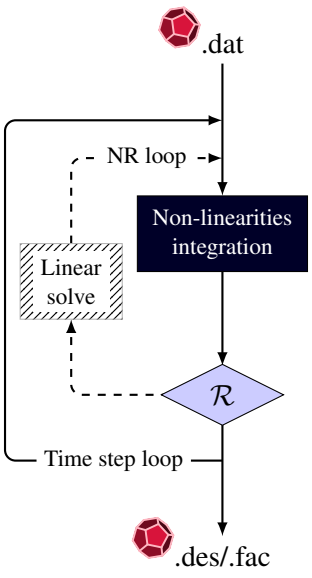


FIGURE 4.1 – Schématisation des grandes étapes de résolution de Simcenter Samcef™

Évaluation des non-linéarités. Cette étape englobe toute la richesse du logiciel éléments finis. En effet, c'est lors de celle-ci qu'est réalisée l'intégration de toutes les lois de comportement matériaux, que la gestion des différents types d'éléments s'opère (éléments 3D ou de contact par exemple), ou encore que les matrices de rigidité, caractérisant la raideur matérielle ou géométrique, sont évaluées et assemblées en vue de la résolution. Pour résumer, si on fournit un champ de déplacement en entrée de cette étape, pour un pas de temps donné, on est alors capable de récupérer en sortie les efforts internes et externes associés, ainsi que la raideur tangente assemblée – pour le pas de temps correspondant.

Résolution du système linéarisé. Cette étape concentre l'ensemble des méthodes de résolution disponibles pour la résolution de systèmes linéaires : méthodes directes – solveurs multi-frontaux, MUMPS – ou méthodes itératives. Ainsi, si on fournit un système linéaire en entrée, on est en mesure de déterminer à la sortie le vecteur solution.

Indicateurs d'erreur. Cette étape globalise les différents indicateurs d'erreur permettant d'appréhender l'état de convergence pour chaque itération k du pas de temps t_j courant. Plusieurs indicateurs sont utilisés dans Simcenter Samcef™ où $\|\bullet\|_1$ désigne la norme 1 sur $\mathbb{R}^n$ et où $\varepsilon_\bullet$ représente systématiquement une petite quantité assurant que le dénominateur ne s'annule pas :

- un indicateur en effort $\mathcal{E}_{\text{TESF}}$ (4.1) basé sur l'écart relatif en résidu d'équilibre ;

$$\mathcal{E}^{(k)}_{\text{TESF}}(t_j) = \frac{\left\|\mathbf{f}^{(k)}_{\text{int}}(t_j) - \mathbf{f}^{(k)}_{\text{ext}}(t_j)\right\|_1}{\left\|\mathbf{f}^{(k)}_{\text{int}}(t_j)\right\|_1 + \left\|\mathbf{f}^{(k)}_{\text{ext}}(t_j)\right\|_1 + \varepsilon_{\text{TESF}}} \qquad \forall k \in \mathbb{N}^* \tag{4.1}$$

- un indicateur d'erreur en déplacement $\mathcal{E}_{\text{TESQ}}$ (4.2) proportionnel à la correction $\Delta\mathbf{u}^{(k)}$ apportée à chaque itération $k \in \mathbb{N}^*$ de l'algorithme de Newton-Raphson ;

$$\mathcal{E}^{(k)}_{\text{TESQ}}(t_j) = \frac{\left\|\Delta\mathbf{u}^{(k)}(t_j)\right\|_1}{\left\|\mathbf{u}^{(k)}(t_j)\right\|_1 + \varepsilon_{\text{TESQ}}} \qquad \forall k \in \mathbb{N}^* \tag{4.2}$$

- un indicateur en énergie $\mathcal{E}_{\text{TESE}}$ (4.3) où w_{int} et w_{ext} désignent respectivement le travail des efforts intérieurs et extérieurs ;

$$\mathcal{E}^{(k)}_{\text{TESE}}(t_j) = \left(\frac{\left\|{}^t\Delta\mathbf{u}^{(k)}(t_j)\left(\mathbf{f}^{(k)}_{\text{int}}(t_j) - \mathbf{f}^{(k)}_{\text{ext}}(t_j)\right)\right\|_1}{\left\|w^{(k)}_{int}(t_j)\right\|_1 + \left\|w^{(k)}_{ext}(t_j)\right\|_1 + \varepsilon_{\text{TESE}}} \right)^{1/2} \qquad \forall k \in \mathbb{N}^* \tag{4.3}$$

- un indicateur d'erreur pour le contact $\mathcal{E}_{\text{TESC}}$ (4.4) relié à la correction des efforts de contact $\Delta\mathbf{f}_{\text{ctc}}$ calculée successivement au cours des itérations de convergence.

$$\mathcal{E}^{(k)}_{\text{TESC}}(t_j) = \frac{\left\|\Delta\mathbf{f}^{(k)}_{\text{ctc}}(t_j)\right\|_1}{\left\|\mathbf{f}^{(k)}_{\text{ctc}}(t_j)\right\|_1 + \varepsilon_{\text{TESC}}} \qquad \forall k \in \mathbb{N}^* \tag{4.4}$$

L'algorithme est alors considéré comme convergé pour le pas de temps t_j courant lorsque chaque indicateur possède une valeur inférieure à une certaine limite, fixée par l'utilisateur. Puis, une fois que tous les pas de temps ont été évalués, on aboutit à la génération d'un ensemble de fichiers résultats (*.des* et *.fac*) exploitable dans l'interface Simcenter 3D™ par exemple.

4.1.3 Contraintes inhérentes aux codes de calculs industriels

Par souci de robustesse, l'un des souhaits réside dans la non-modification de ces grandes routines, ce qui permet d'assurer qu'aucun artefact ne sera introduit dans le code. Il s'agit d'utiliser les mêmes routines que celles utilisées dans la version commerciale, sans modification, en utilisant les entrées et sorties standards prévues à cet effet – et non de ne reprendre que quelques parties. La solution ainsi développée doit rester compatible en tout point avec le logiciel existant, y compris être conforme aux différents protocoles de certification qui attestent de la qualité du code quotidiennement.

Un point délicat se situe notamment dans la gestion de l'aspect temporel puisque les différentes routines ont été conçues pour prendre comme arguments des vecteurs indépendants du temps – vision séquentielle incrémentale. L'utilisation de ces routines dans une architecture non-incrémentale, caractéristique de la méthode LATIN-PGD, demande donc de revenir ponctuellement à une architecture compatible avec une vision incrémentale de la résolution. De plus, au-delà des aspects purement algorithmiques de programmation, plusieurs questions se posent. À titre d'exemple, on peut se demander quel critère d'arrêt conserver afin d'obtenir des solutions comparables. Avec l'algorithme de Newton-Raphson, de nature incrémentale, on itère en chaque pas de temps jusqu'à ce que tous les indicateurs d'erreur (4.1 – 4.4) soient plus petits qu'une certaine valeur seuil : les itérations s'arrêtent dès que le critère est atteint. *A contrario*, avec la méthode LATIN-PGD non-incrémentale, on peut être amené à converger beaucoup plus finement certains pas de temps si on considère comme critère d'arrêt la norme infinie $\|\bullet\|_\infty$ sur $\mathcal{I}_\mathbb{R}$, évaluée à chaque itération de l'algorithme – sur l'ensemble des pas de temps.

Lors de la réflexion portant sur l'introduction de la méthode LATIN-PGD dans Simcenter Samcef™ de manière native et la moins intrusive possible, un démonstrateur semi-industriel a été construit pour tester les idées. Cette plateforme de démonstration a notamment permis de simuler un code industriel pour pouvoir expérimenter la méthode avant son incorporation effective dans le logiciel industriel.

4.2 Démonstrateur semi-industriel : ROMlab

Le code ROMlab a été largement utilisé et développé au cours de ces travaux en guise de démonstrateur semi-industriel. Après avoir brièvement rappelé l'historique de ce code *maison* développé intégralement au Laboratoire de Mécanique Paris-Saclay (LMPS) ces dernières années, nous donnons par la suite un bref aperçu de ses capacités.

4.2.1 Présentation du logiciel

Le démonstrateur ROMlab est principalement écrit sous Matlab®, bien que certaines parties du code aient été codées en C pour des raisons de performances. La mise en donnée du problème résulte de différentes routines permettant, à partir d'un maillage généré sous Gmsh [3] [Geuzaine et Remacle, 2009], de construire l'ensemble des opérateurs éléments finis classiques. Plusieurs méthodes sont alors disponibles pour mener à bien la résolution : l'algorithme de Newton-Raphson ou la méthode LATIN-PGD – variante fonctionnelle et variante à variables internes. Les différents résultats peuvent alors être exploités sous le logiciel libre ParaView [4] [Ahrens *et al.*, 2005].

Initiés à la base par D. Néron, les prémices de ROMlab remontent aux travaux de M. Vitse concernant la résolution multi-paramétrique de structures en béton armé [Vitse, 2016]. S'ensuivent des développements portant sur les problèmes élasto-visco-plastiques avec l'approche à variables internes de la méthode LATIN-PGD [Nachar, 2019], ainsi que sur l'optimisation du logiciel lui-même en exploitant au mieux les

3. Christophe Geuzaine et Jean-François Remacle (2021). Gmsh (https://gmsh.info)
4. ParaView : https://www.paraview.org

architectures parallèles multi-threads actuelles. Un exemple de développement autour de l'optimisation se retrouve dans la fonction *einsum* [5] développée par S. Nachar qui reprend dans les grandes lignes le fonctionnement de la routine de même nom incluse dans la bibliothèque numpy [6] en Python™. En parallèle, R. A. Cardoso [Cardoso, 2019] s'intéresse aux problèmes de *fretting fatigue* et S. Rodriguez [Rodriguez-Iturra *et al.*, 2021] développe certaines capacités pour les problèmes de dynamique basse fréquence (chargement sismique). Ces travaux de thèse ont également permis de mener certains développements, donnés en Section-4.2.2. Ce code est maintenant repris et développé par de nombreux doctorants : autour de problèmes de magnéto-statique avec décomposition de domaines par A. Ruda [Ruda *et al.*, 2021], pour des problèmes de dynamique en génie parasismique associé à la construction de courbes de fragilité par A. Daby-Seesaram [Daby-Seesaram *et al.*, 2021] et autour de problèmes multi-physiques thermo-mécaniques couplés par F. Wurtzer [Wurtzer, 2021]. Le schéma 4.2 rassemble succinctement l'ensemble des actions menées par les différents contributeurs au logiciel ROMlab tel qu'il est aujourd'hui.

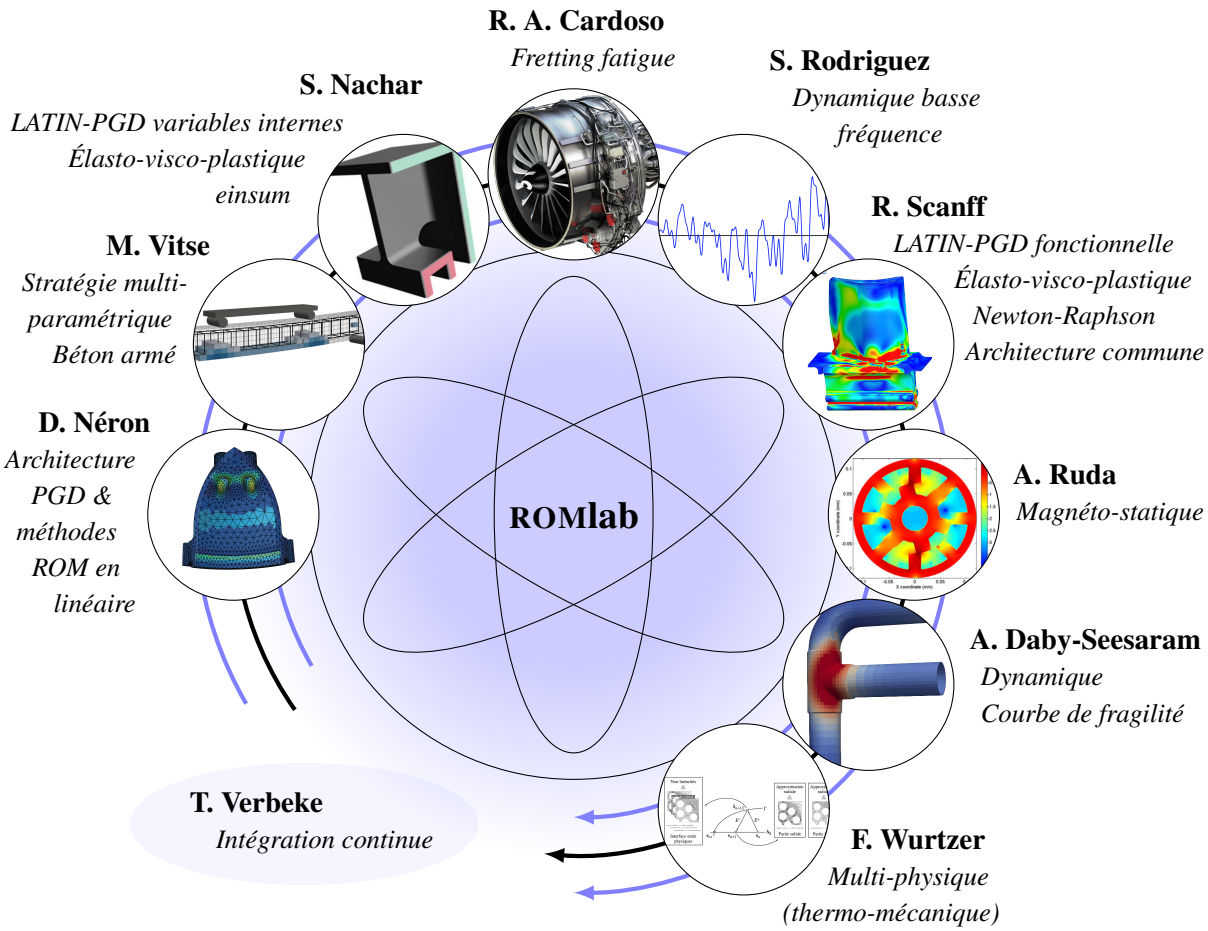


FIGURE 4.2 – Développements des différents contributeurs de ROMlab

4.2.2 Contributions à la communauté ROMlab

L'algorithme de Newton-Raphson a été intégralement codé pour la résolution de problèmes élasto-plastiques à écrouissage isotrope puis élasto-visco-plastiques à écrouissages cinématique et isotrope (loi de Lemaitre-Chaboche détaillée Section-1.1.1). Cette méthode peut servir de solution de référence pour

5. Stéphane Nachar (2021). einsum (https://github.com/SNachar/einsum), GitHub. Retrieved August 27, 2021.
6. numpy.einsum(args) : https://numpy.org/doc/stable/reference/generated/numpy.einsum.html

ROMlab et permet d'émuler le fonctionnement d'un code éléments finis industriels. La méthode LATIN-PGD, dans sa variante fonctionnelle, a alors été développée en se basant sur ces briques existantes. Différentes idées ont ainsi pu être testées pour préparer le terrain d'implémentation dans le logiciel industriel Simcenter Samcef™ proprement dit.

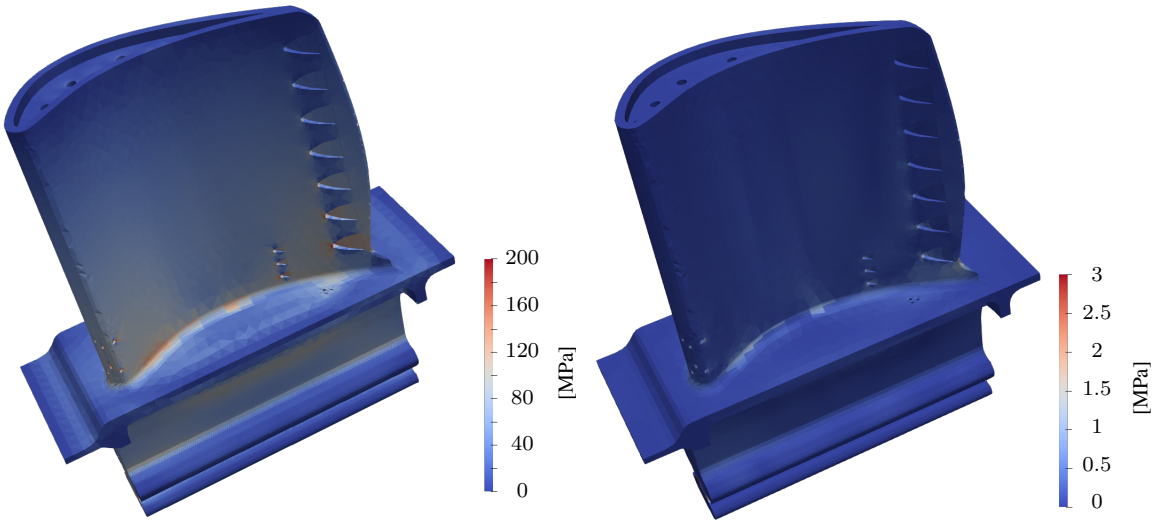
Un effort d'uniformisation des différents codes produits ces dernières années a également été mené dans l'optique de pérenniser au mieux les nombreux développements. Sous une architecture logicielle commune, il est désormais possible de résoudre des problèmes de mécanique non-linéaire (matériaux), de thermique, de magnéto-statique ainsi que des problèmes multi-physiques (thermo-mécanique). L'ensemble des outils de pré-traitement et de post-traitement se retrouvent mutualisés pour l'ensemble des utilisateurs. Parmi les méthodes de résolution non-linéaires, les méthodes Newton-Raphson et LATIN-PGD – dans ses variantes fonctionnelle et à variables internes – s'articulent sous un formalisme commun. Suite à une phase d'optimisation, ceci a notamment permis de mener une étude comparative entre les deux grandes variantes de la méthode LATIN-PGD sur des problèmes élasto-visco-plastiques [Scanff *et al.*, 2021]. Le démonstrateur semi-industriel ROMlab fait aujourd'hui l'objet d'un dépôt git (interne au LMPS) permettant de mener de front les différents développements actuels de manière conviviale et collaborative.

4.2.3 Illustration sur un cas-test

Afin d'illustrer une partie des capacités du démonstrateur semi-industriel ROMlab, on présente dans cette section la résolution d'un problème mécanique non-linéaire. Ce cas-test est simulé simultanément dans ROMlab et dans le logiciel industriel Simcenter Samcef™, ce qui permet de comparer les résultats obtenus en vue de la certification du démonstrateur ROMlab pour les non-linéarités matériaux.

Le cas-test représente l'aube de turbine d'un étage haute pression d'un réacteur d'avion et comptabilise $n = 1.7 \times 10^6$ degrés de liberté. Le matériau élasto-visco-plastique considéré suit une loi de Lemaitre-Chaboche telle qu'énoncée au Chapitre 1. À cela s'ajoute un chargement centrifuge sur l'ensemble de la structure en rotation. Cette géométrie sera notamment reprise au Chapitre 9 comme illustration pour la méthode LATIN-PGD implémentée nativement dans Simcenter Samcef™ de manière faiblement intrusive. Que ce soit avec le démonstrateur semi-industriel ROMlab ou le logiciel industriel Simcenter Samcef™, on adopte une stratégie de résolution incrémentale par un algorithme de Newton-Raphson dans cette partie. Les calculs sont réalisés sur une même machine pour s'affranchir des éventuelles différences liées aux architectures informatiques. Par l'intermédiaire du logiciel ParaView, la Figure 4.3 présente la carte d'erreur spatiale de la contrainte de Von Mises obtenue en comparant les deux solutions pour le pas de temps le plus chargé. On note ainsi une bonne concordance des solutions avec une erreur de l'ordre de 1 MPa dans la zone d'intérêt. La différence est légèrement plus importante par endroits, notamment au niveau des fortes discontinuités géométriques (trous du circuit de refroidissement), ce qui est principalement lié à des erreurs de discrétisation et d'interpolation.

Concernant les performances, alors qu'avec le démonstrateur ROMlab 1562 secondes sont nécessaires pour mener à bien la résolution, le logiciel industriel Simcenter Samcef™ demande 634 secondes pour aboutir à une solution de qualité équivalente. Le démonstrateur ROMlab est ainsi capable de réaliser des simulations complexes en un temps raisonnable, certes moins efficacement qu'un code industriel optimisé depuis de longues années mais dans un temps tout à fait compatible avec les cycles de développement de nouvelles méthodes en laboratoire. Un tel démonstrateur apporte notamment une grande souplesse pour tester rapidement diverses idées, ce qui permet par la suite de concentrer les efforts sur celles qui semblent les plus prometteuses.



(a) Contrainte de Von Mises

(b) Carte d'erreur de la contrainte de Von Mises

FIGURE 4.3 – Contrainte de Von Mises et carte d'erreur associée issue de la comparaison des résultats obtenus avec les logiciels ROMlab et Simcenter Samcef™ respectivement

REMARQUE 4.1 : La comparaison des temps de calculs exposée précédemment doit être relativisée dans la mesure où l'architecture parallèle émanant des deux logiciels est radicalement différente : ROMlab fonctionne exclusivement sur une architecture à mémoire partagée OpenMP (*multi-threading*) tandis que Simcenter Samcef™ possède une architecture hybride MPI-OpenMP pouvant faire intervenir des processus à mémoire distribuée et partagée. À cela s'ajoute que les langages de programmation sont également disjoints, ROMlab étant écrit majoritairement sous le langage Matlab® avec quelques routines codées en langage C tandis que Simcenter Samcef™ repose intégralement sur un langage Fortran compilé. De plus, alors que tous les calculs sont réalisés en mémoire vive (RAM) sous ROMlab, certaines opérations nécessitent des écritures sur disques au cours du calcul dans le logiciel Simcenter Samcef™, ce qui explique notamment la différence des temps de calculs constatés au niveau de l'intégration locale des éléments (cf. Tableau 4.1 fournissant la comparaison des temps de calculs des trois étapes majoritaires).

TABLE 4.1 – Comparaison des temps de calculs de certaines opérations

	Factorisation [s]	**Résolution** [s]	**Intégration locale** [s]
ROMlab	548.7	26.7	1.1
Simcenter Samcef™	119.5	0.98	2.5

Chapitre 5

Nouvelle version faiblement intrusive de la méthode LATIN-PGD

Ce chapitre est centré sur la formulation faiblement intrusive de la méthode LATIN-PGD destinée à être implémentée nativement dans un logiciel industriel. Basée sur une variante fonctionnelle par l'intermédiaire de grandeurs généralisées, cette formulation permet d'utiliser au mieux les opérateurs déjà disponibles de manière standard. Les détails d'implémentation pratiques, ainsi qu'une illustration sur un cas-test académique, sont fournis.

Sommaire

5.1 Formulation du problème mécanique non-linéaire

On considère dans ce chapitre que la seule source de non-linéarités se situe au niveau du comportement matériaux. La configuration initiale est prise comme référence et aucun effort de contact n'est considéré.

Reprenant les équations du problème de référence (1.23 – 1.25), le champ de déplacement $\mathbf{u} \in \mathcal{I}_{\mathcal{V}}$ est recherché tel que le système d'équations (5.1) soit satisfait. Le problème s'écrit :

$$\text{Trouver } \mathbf{u} \in \mathcal{I}_{\mathcal{V}} \text{ tel que :}$$
$$\mathbf{u}(t=0) = \mathbf{0} \qquad \textit{(conditions initiales)}$$
$$\forall t \in I, \quad \begin{cases} \boldsymbol{\mathcal{C}}_{\mathbf{u}}\Big(\mathbf{u}(t)\Big) = \mathbf{u}_d(t) & \textit{(contraintes cinématiques)} \\ \mathbf{f}_{\text{int}}\big(t, \mathbf{u}(t)\big) - \mathbf{f}_{\text{ext}}(t) = \mathbf{0} & \textit{(équation d'équilibre)} \\ \mathbf{f}_{\text{int}}\big(t, \mathbf{u}(t)\big) = \boldsymbol{\mathcal{A}}_u\Big(t, \mathbf{u}(\tau \leqslant t)\Big) & \textit{(relation de comportement)} \end{cases} \tag{5.1}$$

où l'opérateur $\boldsymbol{\mathcal{C}}_{\mathbf{u}}$ est linéaire. $\boldsymbol{\mathcal{A}}_u$ est un opérateur non-linéaire dépendant du temps, caractéristique du matériau considéré. Il est associé aux efforts intérieurs de cohésion et possède une forme très générale : nous supposons seulement que les forces généralisées $\mathbf{f}_{\text{int}}$ au temps t dépendent de l'histoire des déplacements généralisés $\mathbf{u}$ jusqu'à cet instant t.

Dans la suite, nous présentons la méthode LATIN-PGD sur une décomposition espace-temps : le traitement des paramètres $\boldsymbol{\mu} \in \boldsymbol{\mathcal{D}}$ supplémentaires sera détaillé au Chapitre-8.

5.2 Principes P1 et P2

L'application des deux premiers principes fondateurs de la méthode LATIN-PGD [Ladevèze, 1999] est détaillée dans cette section. Le principe P3 avec l'introduction de la représentation sous forme à variables séparées issue de la PGD sera quant à lui développé en Section-5.3.4 avec les détails d'implémentation.

5.2.1 Choix des variables et séparation des difficultés – Principe P1

Le premier principe P1 consiste à séparer les difficultés. Cette nouvelle version reste toujours basée sur la séparation entre les équations d'équilibre et les équations issues des relations de comportement. Nous adoptons dans le cadre de ces travaux une approche fonctionnelle de la méthode LATIN-PGD à savoir que l'état au temps $t \in \mathbb{R}_+$ est défini au moyen de quantités généralisées $\boldsymbol{s} = (\mathbf{u}, \mathbf{f}_{\text{int}}) \in \mathcal{I}_{\mathcal{V}} \times \mathcal{I}_{\mathcal{V}}$ – déplacements et forces généralisés respectivement. Le problème (5.1) peut se réécrire sous la forme (5.2 – 5.3) suivante où les variétés $\boldsymbol{\Gamma}$ et $\mathbf{A}_d$ sont définies :

$$\text{Trouver } \boldsymbol{s} = (\mathbf{u}, \mathbf{f}_{\text{int}}) \in \mathcal{I}_{\mathcal{V}} \times \mathcal{I}_{\mathcal{V}} \text{ tel que :}$$
$$(\boldsymbol{\Gamma}): \qquad \mathbf{f}_{\text{int}}\big(t, \mathbf{u}(t)\big) = \boldsymbol{\mathcal{A}}_u\Big(t, \mathbf{u}(\tau \leqslant t)\Big), \quad \forall t \in I \tag{5.2}$$
$$(\mathbf{A}_d): \quad \begin{cases} \mathbf{u}(t=0) = \mathbf{0} \\ \forall t \in I, \quad \begin{cases} \boldsymbol{\mathcal{C}}_{\mathbf{u}}\Big(\mathbf{u}(t)\Big) = \mathbf{u}_d(t) \\ \mathbf{f}_{\text{int}}\big(t, \mathbf{u}(t)\big) - \mathbf{f}_{\text{ext}}(t) = \mathbf{0} \end{cases} \end{cases} \tag{5.3}$$

La variété $\boldsymbol{\Gamma}$ regroupe l'ensemble des équations locales (éventuellement non-linéaires), tandis que la variété $\mathbf{A}_d$ rassemble toutes les équations linéaires (éventuellement globales) du problème. Une représentation géométrique de ces variétés est donnée Figure 5.1 où la variété $\mathbf{A}_d$ (linéaire) est représentée par une droite alors que la variété $\boldsymbol{\Gamma}$ (éventuellement non-linéaire) est décrite par une courbe. L'intersection $\boldsymbol{s}_\chi$ de ces deux variétés correspond à la solution exacte du problème de référence dont nous supposons l'existence et l'unicité.

REMARQUE 5.1 : Suivant une approche fonctionnelle, la variété $\boldsymbol{\Gamma}$ englobe ici l'ensemble des équations découlant des lois de comportement matériaux (équations d'état et équations d'évolution). Usuel-

lement, dans l'approche classique à variables internes, seules les lois d'évolution composent $\mathbf{\Gamma}$, les lois d'état – linéaires – étant assignées à la variété $\mathbf{A}_d$.

5.2.2 Choix des directions de recherche – Principe P2

Le second principe P2 repose sur la définition de deux directions de recherche pour fermer le système d'équations. Nous définissons ainsi deux espaces supplémentaires $\mathbf{\Upsilon}^+$ et $\mathbf{\Upsilon}^-$ permettant de construire alternativement, à chaque itération, une solution dans chacune des deux variétés, respectivement $\mathbf{\Gamma}$ et $\mathbf{A}_d$, comme illustré Figure 5.1. Les directions de recherche sont les paramètres majeurs de la méthode. Leur choix affecte directement la vitesse de convergence de l'algorithme mais également la facilité d'implémentation dans les logiciels industriels comme nous le verrons par la suite.

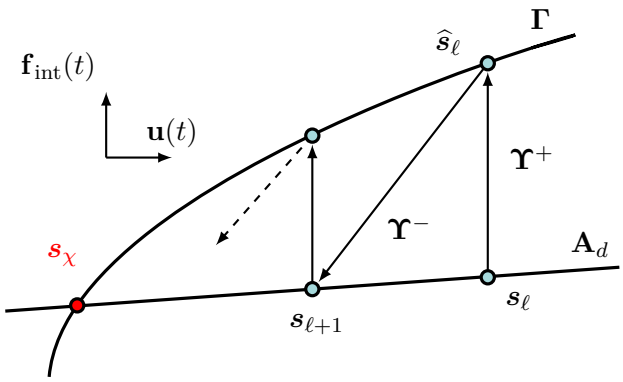


FIGURE 5.1 – Schéma itératif convergent méthode LATIN-PGD avec $\boldsymbol{s}_\ell = (\mathbf{u}_\ell, \mathbf{f}_{\text{int},\ell})$

L'initialisation de l'algorithme peut s'effectuer avec n'importe quel couple $\boldsymbol{s}_0 \in \mathbf{A}_d$ pourvu que les conditions d'admissibilité statiques et cinématiques soient respectées. Lorsqu'aucune information préalable n'est disponible, une initialisation élastique est usuellement retenue, celle-ci étant aisément accessible. Puis, au travers d'une succession d'étapes locales et globales, on construit les termes d'une série convergente $\left(\boldsymbol{s}_k(t)\right)_{k\in\mathbb{N}}$ de limite $\boldsymbol{s}_\chi(t)$. Plus précisément, si on suppose connu le couple $\boldsymbol{s}_\ell$ à une itération donnée de l'algorithme, le couple $\boldsymbol{s}_{\ell+1}$ est recherché de la manière suivante :

Étape locale. Étant donné $\boldsymbol{s}_\ell(t)$, nous cherchons un nouveau couple $\widehat{\boldsymbol{s}}_\ell(t) \in \mathbf{\Gamma}$ selon la direction de recherche $\mathbf{\Upsilon}^+$ en résolvant les équations non-linéaires du problème. À cette fin, une direction de recherche verticale (5.4) est exploitée $\left(\mathbf{\Theta}^+ \equiv \infty\right)$:

$$\forall t \in I, \quad \widehat{\mathbf{u}}_\ell(t) = \mathbf{u}_\ell(t) \tag{5.4}$$

Ce choix particulier que nous faisons, caractérisant l'espace $\mathbf{\Upsilon}^+$, conduit à découpler naturellement les quantités primales et duales au niveau de l'étape locale : la connaissance de la variable primale seule – déplacements généralisés $\mathbf{u}_\ell$ – suffit à résoudre l'ensemble du système d'équations. Pour obtenir les forces internes généralisées $\widehat{\mathbf{f}}_{\text{int},\ell}$ correspondantes, il faut alors intégrer les différentes relations de comportement grâce au logiciel industriel. Cette tâche n'est pas standard dans la mesure où l'ensemble de l'intervalle de temps I doit être adressé simultanément, mais reste néanmoins faiblement intrusive. C'est la pierre angulaire de la version proposée pour la méthode LATIN-PGD : l'opérateur $\boldsymbol{\mathcal{A}}_u$ doit être construit pour englober l'ensemble des capacités du logiciel industriel – tous les types d'éléments, toutes les relations de comportement, les algorithmes de résolution associés, etc.

Étape globale. Puis, connaissant $\widehat{\boldsymbol{s}}_\ell(t)$, nous cherchons un nouveau couple $\boldsymbol{s}_{\ell+1}(t) \in \mathbf{A}_d$ selon la direction de recherche $\mathbf{\Upsilon}^-$ en résolvant les équations linéaires du problème – équation d'équilibre. Pour ce

faire, une direction de recherche (5.5) caractérisée par un opérateur $\mathbf{\Theta}^-$, noté $\mathbf{H}$ pour simplifier par la suite, est sollicitée :

$$\forall t \in I, \quad \mathbf{H}\Big(\mathbf{u}_{\ell+1}(t) - \widehat{\mathbf{u}}_{\ell}(t)\Big) = \mathbf{f}_{\text{int},\,\ell+1}(t) - \widehat{\mathbf{f}}_{\text{int},\,\ell}(t) \tag{5.5}$$

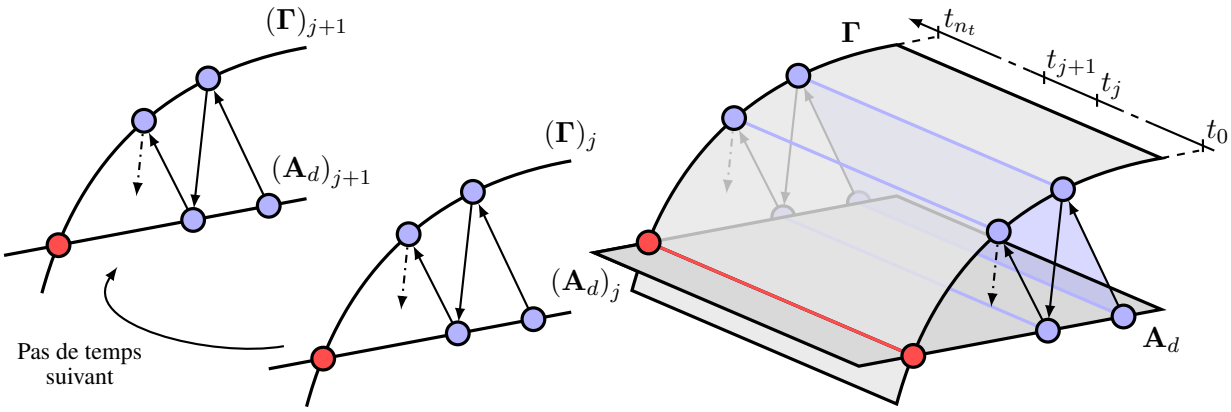
Le choix de l'opérateur $\mathbf{H}$ définissant l'espace $\mathbf{\Upsilon}^-$ est le paramètre majeur de la méthode. Là encore, plusieurs choix sont possibles pour définir cette direction de recherche, guidés d'une part par la facilité d'implémentation et, d'autre part, par la rapidité de convergence. Les différents choix possibles pour $\mathbf{H}$ seront détaillés par la suite (Section-5.4.2). Lors de cette étape linéaire, un point important se situe dans le calcul de la solution, mené par la méthode PGD. Cette méthode de réduction de modèle, incluse de fait dans la méthode LATIN-PGD, est la source de l'efficacité remarquable de la méthode, minimisant le nombre de systèmes linéaires à résoudre. Chacune des corrections est alors recherchée sous la forme à variables séparées (5.6) suivante :

$$\forall t \in I, \quad \Delta\mathbf{u}_{\ell+1}(t) = \mathbf{u}_{\ell+1}(t) - \mathbf{u}_{\ell}(t) = \sum_{i=1}^{m_\ell} \lambda_i(t)\mathbf{\Lambda}_i \quad \text{avec} \quad \begin{cases} \{\lambda_i\}_{1\leqslant i\leqslant m_\ell} \in \mathcal{I}_{\mathbb{R}} \\ \{\mathbf{\Lambda}_i\}_{1\leqslant i\leqslant m_\ell} \in \boldsymbol{\mathcal{V}} \end{cases} \tag{5.6}$$

Cet algorithme itératif continue tant que le nouveau couple $\boldsymbol{s}_{\ell+1}$ reste trop éloigné de la vraie solution $\boldsymbol{s}_\chi$ du problème avec une certaine tolérance. La vraie solution n'étant pas connue à l'avance, nous avons recours à un indicateur d'erreur qui doit être garanti, robuste et aisément calculable. Le critère retenu dans ces travaux est détaillé en Section-5.3.5.1.

5.2.3 Analogie avec le schéma de résolution par Newton-Raphson

L'un des avantages d'avoir effectué une telle séparation des difficultés dans la constitution des variétés $\mathbf{\Gamma}$ et $\mathbf{A}_d$ est de rester conforme à l'algorithme de Newton-Raphson qui régit le logiciel industriel. En considérant $(\mathbf{\Gamma})_j$ et $(\mathbf{A}_d)_j$ comme les restrictions respectives des variétés $\mathbf{\Gamma}$ et $\mathbf{A}_d$ au pas de temps t_j donné, il s'ensuit que sous cette forme – et sans technique de réduction de modèle – la méthode LATIN est équivalente à la méthode de Newton-Raphson par simple permutation des boucles sur les itérations et sur les pas de temps. La Figure 5.2 illustre d'une part (a) la stratégie incrémentale de Newton-Raphson où, à chaque pas de temps t_j, plusieurs itérations entre l'intégration des lois de comportement et la résolution de l'équilibre sont nécessaires pour atteindre la convergence, avant de passer au pas de temps t_{j+1} suivant ; et, d'autre part, (b) la stratégie non-incrémentale LATIN-PGD où les itérations entre les variétés $\mathbf{\Gamma}$ et $\mathbf{A}_d$ impliquent l'ensemble de l'intervalle de temps à chaque itération.



(a) Newton-Raphson : stratégie incrémentale (b) LATIN-PGD : stratégie non-incrémentale

FIGURE 5.2 – Interprétation de l'algorithme de Newton-Raphson au regard du formalisme induit par la méthode LATIN-PGD

Qui plus est, considérer une direction de recherche verticale au niveau de l'étape locale permet d'envisager une implémentation fonctionnelle et commode dans un logiciel industriel avec des modifications mineures. L'un des grands avantages d'une telle formalisation réside dans la manipulation du même type d'équations (relations de comportement) de manière semblable (direction de recherche verticale) au même endroit du code industriel. Seul l'ordonnancement des différentes opérations diffère dans la gestion de l'intervalle de temps.

5.3 Détails sur la mise en œuvre

On fournit ci-après quelques détails supplémentaires quant à la mise en œuvre de la méthodologie LATIN-PGD dans sa version faiblement intrusive destinée à être implémentée nativement dans un logiciel industriel commercial. Les considérations numériques de l'implémentation effective au sein du logiciel industriel Simcenter Samcef™ sont toutes rassemblées dans le Chapitre-8 pour plus de clarté.

5.3.1 Initialisation

L'initialisation de l'algorithme peut s'effectuer à partir de n'importe quelle approximation $\boldsymbol{s}_0 \in \mathbf{A}_d$ comme cela a déjà été évoqué précédemment. Parmi tous les choix possibles, nous en avons retenu deux qui se distinguent par leur accessibilité directe au logiciel industriel :

- une initialisation élastique (désactivation des non-linéarités matériaux) ;
- une initialisation non-linéaire partiellement convergée (en rehaussant les critères d'arrêt associés aux indicateurs $\mathcal{E}_{\text{TESF}}$, $\mathcal{E}_{\text{TESQ}}$ et $\mathcal{E}_{\text{TESE}}$ donnés respectivement en (4.1), (4.2) et (4.3)).

Dans chaque cas, des paramètres numériques sont en effet accessibles directement au niveau de la mise en données du problème, ce qui permet de générer aisément l'approximation initiale. Par technique SVD, il est ensuite possible d'extraire les modes cinématiquement admissibles et d'obtenir un stockage à moindre coût de cette solution. De plus, lors de la conduite d'études paramétriques, nous exploitons les solutions antérieurement calculées.

5.3.2 Structure générale de l'algorithme

De nombreuses discussions avec l'éditeur de logiciel Siemens Digital Industries Software nous ont conduit à imaginer une stratégie compatible avec la méthode LATIN-PGD d'une part, et avec les multiples contraintes industrielles d'autre part. La prise en compte de l'architecture non-incrémentale propre à la méthode LATIN-PGD avec sa boucle de convergence externe sur les itérations – atypique des codes industriels – a exigé de développer une structuration particulière et donc nécessité de légères modifications au niveau du code source de la part de l'éditeur de logiciels. C'est en ce sens que notre méthodologie s'inscrit dans le cadre des méthodes faiblement intrusives. De plus amples détails seront donnés par la suite et plus spécifiquement au Chapitre-8.

L'accès au code source, une très bonne maîtrise du logiciel industriel et l'ajout d'une poignée de lignes de code ont été des ingrédients essentiels sans quoi ces travaux n'auraient pas été possibles – sous cette forme tout du moins. Néanmoins, la nouvelle structure que nous avons mis en place a été pensée pour être la plus ergonomique possible : les quelques modifications nécessaires ont pu être réalisées en une seule fois et la délimitation d'un espace de travail adéquat a permis de circonscrire proprement les nouveaux développements du code préexistant, de manière à minimiser le niveau d'intrusivité, et par la même occasion les risques associés. Autrement dit, cette démarche très générale, conçue pour être la moins intrusive possible, permet d'exploiter pleinement les capacités du logiciel lors de l'étape locale tout en fournissant un cadre de travail indépendant pour la réalisation de l'étape globale. Toutefois, cette démarche entraine également de nouvelles contraintes comme nous le verrons par exemple au Chapitre-7 pour le traitement des problèmes de contact.

REMARQUE 5.2 : Une autre stratégie possible aurait été de réaliser la boucle globale sur les itérations LATIN de manière complètement externe au logiciel Simcenter Samcef™ – en appelant le logiciel autant de fois que nécessaire. Une telle façon de procéder aurait été encore moins intrusive mais beaucoup moins performante.

5.3.3 Étape locale

L'étape locale consiste à construire l'opérateur $\boldsymbol{\mathcal{A}}_u$ par intégration des lois de comportement matériaux sur l'ensemble de l'intervalle de temps. La résolution de ces équations non-linéaires nécessite la plupart du temps un schéma de résolution spécifique – algorithme de Newton-Raphson interne ou toute autre technique. L'idée qui a été retenue pour adresser la création de cet opérateur $\boldsymbol{\mathcal{A}}_u$ non-linéaire, qui n'apparait pas naturellement dans les codes industriels, consiste à s'appuyer sur les routines classiques préexistantes. Nous englobons ainsi toute la richesse du logiciel industriel dans la variété des capacités offertes – toutes lois de comportement, tous types d'élément, etc. En particulier, aucune modification n'est nécessaire dans la gestion des différentes fonctionnalités dans la mesure où cette étape est traitée en tant que *boîte noire* : nous fournissons simplement le champ de déplacement en entrée et nous récupérons en sortie les efforts internes généralisés, tout comme les matrices de raideur le cas échéant, sans se préoccuper du fonctionnement des algorithmes internes de résolution non-linéaire dissimulés derrière. Par exploitation de la boucle temporelle native du logiciel Simcenter Samcef™, nous réalisons cette intégration du comportement avec le temps. La construction de l'opérateur $\boldsymbol{\mathcal{A}}_u$ ne nécessite en définitive qu'une petite modification destinée à faire volontairement fi de toute résolution de systèmes linéaires au niveau de l'étape locale et entreprendre systématiquement la gestion du pas de temps suivant.

REMARQUE 5.3 : Pour assurer la convergence globale de la méthode LATIN-PGD faiblement intrusive, une condition nécessaire est que l'étape locale converge, et plus particulièrement l'algorithme de résolution dédié à l'intégration des lois de comportement non-linéaires.

5.3.4 Étape linéaire – Principe P3

La résolution de l'étape linéaire est réalisée via la méthode PGD espace-temps classique, par l'intermédiaire d'un algorithme glouton – aussi appelé *greedy*. À une itération ℓ donnée, on suppose connue l'approximation précédente c'est-à-dire les m_ℓ premiers modes $\{(\boldsymbol{\Lambda}_i, \lambda_i^\ell)\}_{1\leqslant i\leqslant m_\ell} \in \{\boldsymbol{\mathcal{V}} \times \mathcal{I}_{\mathbb{R}}\}^{m_\ell}$ de la décomposition à variables séparées (5.7) où les conditions aux limites non homogènes sont prises en compte au moyen d'un relèvement à partir de l'approximation $\mathbf{u}_0$ issue de l'initialisation :

$$\forall t \in I, \quad \mathbf{u}_\ell(t) = \sum_{i=1}^{m_\ell} \lambda_i^\ell(t)\boldsymbol{\Lambda}_i + \mathbf{u}_0(t) \tag{5.7}$$

On cherche alors une nouvelle approximation, de meilleure qualité, toujours sous forme à variables séparées par résolution du problème (5.8) concaténant l'ensemble des équations linéaires :

$$\text{Connaissant } \widehat{\boldsymbol{s}}_\ell = (\widehat{\mathbf{u}}_\ell, \widehat{\mathbf{f}}_{\text{int},\,\ell}) \in \boldsymbol{\Gamma} \text{ trouver } \boldsymbol{s}_{\ell+1} = (\mathbf{u}_{\ell+1}, \mathbf{f}_{\text{int},\,\ell+1}) \in \mathbf{A}_d \text{ tel que :}$$

$$\begin{aligned} (\mathbf{A}_d) : \quad & \begin{cases} \mathbf{u}_{\ell+1}(t=0) = \mathbf{0} \\ \forall t \in I, \quad \begin{cases} \boldsymbol{\mathcal{C}}_{\mathbf{u}}\Big(\mathbf{u}_{\ell+1}(t)\Big) = \mathbf{u}_d(t) \\ \mathbf{f}_{\text{int},\,\ell+1}(t) - \mathbf{f}_{\text{ext}}(t) = \mathbf{0} \end{cases} \end{cases} \\ (\boldsymbol{\Upsilon}^-) : \quad & \forall t \in I, \quad \begin{cases} \mathbf{H}\Delta\mathbf{u}_{\ell+1}(t) = \mathbf{f}_{\text{int},\,\ell+1}(t) - \widehat{\mathbf{f}}_{\text{int},\,\ell}(t) \\ \mathbf{u}_{\ell+1}(t) = \widehat{\mathbf{u}}_\ell(t) + \Delta\mathbf{u}_{\ell+1}(t) \end{cases} \end{aligned} \tag{5.8}$$

Résoudre ce problème sous la condition de rechercher les solutions dans un espace à variables séparées conduit naturellement à la définition d'un problème sur-contraint. Une façon de relaxer le problème

consiste alors à ne vérifier qu'en moyenne l'équation de la direction de recherche, ce qui conduit à l'écriture d'un problème de minimisation équivalent. On détaille dans la suite les deux grandes étapes inhérentes à l'application de la méthode PGD en considérant une certaine direction de recherche $\mathbf{H}$ générique. Le calcul de l'opérateur $\mathbf{H}$ est mené au niveau de l'étape locale précédente, en même temps que les efforts internes généralisés. Deux cas sont déclinés :

- le cas général où la direction de recherche $\mathbf{H}(t)$ est variable – dépendante du temps $t \in I$;
- le cas spécifique où la direction de recherche $\mathbf{H}$ est constante – indépendante du temps.

REMARQUE 5.4 : Du fait de la direction de recherche $\widehat{\mathbf{u}}_\ell = \mathbf{u}_\ell$ particulière employée, le champ $\widehat{\mathbf{u}}_\ell$ connu de l'étape locale précédente s'écrit également sous forme à variables séparées.

5.3.4.1 Étape de mise à jour (ou *update*)

Nous cherchons une actualisation $\{\lambda_i^{up}\}_{1\leqslant i\leqslant m_\ell}$ des fonctions temporelles seules – les modes spatiaux étant fixés. La meilleure approximation $\Delta\mathbf{u}_{\ell+1}$ (5.9) solution du problème (5.8) est recherchée sous la forme :

$$\Delta\mathbf{u}_{\ell+1}(t) = \sum_{i=1}^{m_\ell} (\underbrace{\lambda_i^{up}(t) - \lambda_i^{\ell}(t)}_{\beta_i(t)})\boldsymbol{\Lambda}_i \qquad \forall t \in I \tag{5.9}$$

En considérant que l'équation de la direction de recherche n'est vérifiée qu'en moyenne, cette étape préliminaire de mise à jour revient à résoudre le problème de minimisation (5.10) suivant :

$$\{\beta_i\}_{i\in[\![1,m_\ell]\!]} = \underset{\substack{\alpha_i \in \mathcal{I}_{\mathbb{R}} \\ i\in[\![1,m_\ell]\!]}}{\arg\min}\, J\Big(\sum_{i=1}^{m_\ell} \alpha_i(t)\boldsymbol{\Lambda}_i\Big) \tag{5.10}$$

où la fonctionnelle J est donnée par la formule (5.11) avec $\boldsymbol{\mathcal{Q}} = \mathbf{f}_{\text{ext}} - \widehat{\mathbf{f}}_{\text{int},\,\ell}$ pour simplifier les notations :

$$J(\mathbf{w}) = \frac{1}{2}\int_I {}^t\mathbf{w}(t)\Big[\mathbf{H}\mathbf{w}(t) - 2\boldsymbol{\mathcal{Q}}(t)\Big]\,\mathrm{d}t \tag{5.11}$$

Selon les propriétés de l'opérateur $\mathbf{H}$, le calcul peut s'avérer plus ou moins direct.

Direction de recherche $\mathbf{H}(t)$ variable – dépendante du temps. Dans le cas général, l'opérateur $\mathbf{H}$ dépend du temps. La résolution du problème de minimisation (5.10) conduit alors à n_t inversions de petits systèmes linéaires (5.12) – de taille $(m_\ell \times m_\ell)$ – dont le coût de calcul demeure généralement modéré puisque la résolution peut être aisément parallélisée en temps, et qu'un nombre de modes m_ℓ limité (une dizaine) est la plupart du temps suffisant pour représenter adéquatement la solution :

$$\forall t \in I,\ \forall j \in [\![1, m_\ell]\!], \quad {}^t\boldsymbol{\Lambda}_j\left[\mathbf{H}(t)\Big(\sum_{i=1}^{m_\ell}\beta_i(t)\boldsymbol{\Lambda}_i\Big) - \boldsymbol{\mathcal{Q}}(t)\right] = 0 \tag{5.12}$$

Il est également possible de réduire sensiblement la complexité des opérations algébriques en considérant un opérateur $\mathbf{H}$ plus spécifique, notamment constant.

Direction de recherche $\mathbf{H}$ constante – indépendante du temps. Dans le cas particulier où l'opérateur $\mathbf{H}$ est constant et indépendant du temps, l'actualisation des fonctions temporelles se réduit à une formule explicite. C'est par exemple le cas en considérant comme direction de recherche l'opérateur de Hooke élastique $\mathbf{K}$ – matrice de raideur assemblée. Sous hypothèse d'orthogonalité des modes spatiaux, au sens de l'énergie élastique associée au produit scalaire (5.13) suivant :

$$\langle \boldsymbol{v}, \boldsymbol{w}\rangle_{\mathbf{K}} = {}^t\boldsymbol{w}\mathbf{K}\boldsymbol{v} \qquad \forall(\boldsymbol{v}, \boldsymbol{w}) \in \boldsymbol{\mathcal{V}} \times \boldsymbol{\mathcal{V}} \tag{5.13}$$

alors l'équation (5.12) se résume à la formule explicite (5.14) donnée ci-dessous :

$$\forall i \in [\![1, m_\ell]\!],\ \forall t \in I, \quad \beta_i(t) = {}^t\boldsymbol{\Lambda}_i \boldsymbol{\mathcal{Q}}(t) \tag{5.14}$$

Une direction de recherche plus riche sous la forme d'un produit à variables séparées $\mathbf{H}(t) = g(t)\mathbf{K}$ constitue un second exemple. La formule explicite d'actualisation des fonctions temporelles est alors légèrement modifiée en (5.15) où la fonction $g(t)$ joue un rôle de pondération supplémentaire :

$$\forall i \in [\![1, m_\ell]\!],\ \forall t \in I, \quad \beta_i(t) = \frac{1}{g(t)} {}^t\boldsymbol{\Lambda}_i \boldsymbol{\mathcal{Q}}(t) \tag{5.15}$$

L'étape de mise à jour des fonctions temporelles représente ainsi un moyen peu coûteux d'utiliser au mieux la base réduite déjà constituée, ce qui permet de limiter le nombre de modes générés. On construit ainsi d'office une base plus optimale – au sens où moins de modes sont nécessaires pour représenter la même quantité d'information – plus rapidement, puisque l'ajout d'un nouveau mode est de loin l'étape la plus coûteuse comme nous allons le voir.

REMARQUE 5.5 : Bien que l'étape de mise à jour de la base existante puisse s'appliquer aussi bien aux fonctions spatiales que temporelles, seule la mise à jour des fonctions temporelles est effectuée dans le cadre de ces travaux. Cela permet de conserver une étape préliminaire peu coûteuse. Dans le cas général, l'actualisation des fonctions spatiales conduirait en effet à inverser un système linéaire de grande dimension – de taille $(n\, m_\ell) \times (n\, m_\ell)$ – ce qui peut rapidement devenir inaccessible. Un exemple d'actualisation des fonctions spatiales est donné dans [Giacoma *et al.*, 2015] pour les problèmes de contact frottant.

5.3.4.2 Génération de modes

Si, en un certain sens, l'étape de mise à jour s'avère inefficace dans l'amélioration de l'approximation sur base spatiale fixée, nous recherchons un nouveau couple $(\boldsymbol{\Lambda}, \lambda)$ pour enrichir l'espace réduit. Les critères associés au choix d'augmentation du nombre de modes seront discutés ultérieurement (Section-5.3.5). En considérant toujours que l'équation de la direction de recherche n'est vérifiée qu'en moyenne, ceci conduit à la résolution du problème de minimisation (5.16) donné ci-après :

$$(\boldsymbol{\Lambda}, \lambda) = \underset{\substack{\alpha \in \mathcal{I}_{\mathbb{R}} \\ \boldsymbol{\gamma} \in \boldsymbol{\mathcal{V}}}}{\arg\min}\ J\left(\alpha\boldsymbol{\gamma} + \sum_{i=1}^{m_\ell} \beta_i \boldsymbol{\Lambda}_i\right) \tag{5.16}$$

Nous aboutissons alors à un problème classique de type point fixe (5.17) que nous écrivons formellement :

$$\begin{cases} \boldsymbol{\Lambda} = \xi_{\boldsymbol{\Lambda}}(\lambda) \\ \lambda = \xi_\lambda(\boldsymbol{\Lambda}) \end{cases} \tag{5.17}$$

où les fonctions $\xi_{\boldsymbol{\Lambda}}(\lambda)$ et $\xi_\lambda(\boldsymbol{\Lambda})$ sont respectivement données par les formules (5.18) et (5.19) suivantes :

$$\xi_{\boldsymbol{\Lambda}}(\lambda) = \left[\int_I \lambda^2(t)\mathbf{H}\, \mathrm{d}t\right]^{-1} \int_I \lambda(t) \check{\boldsymbol{\mathcal{Q}}}(t)\, \mathrm{d}t \tag{5.18}$$

$$\xi_\lambda(\boldsymbol{\Lambda}) = \frac{{}^t\boldsymbol{\Lambda}\check{\boldsymbol{\mathcal{Q}}}}{{}^t\boldsymbol{\Lambda}\mathbf{H}\boldsymbol{\Lambda}} \tag{5.19}$$

avec $\check{\boldsymbol{\mathcal{Q}}} = \boldsymbol{\mathcal{Q}} - \mathbf{H}\sum_{i=1}^{m_\ell} \beta_i \boldsymbol{\Lambda}_i$ obtenu par déflation.

Dans la pratique, seules quelques itérations sont nécessaires pour la résolution du point fixe : la convergence s'effectue relativement rapidement et une convergence stricte n'est pas de mise dans la mesure où

l'ajout des couples suivants pourra venir corriger l'éventuelle erreur commise à cette étape. Dans la suite, on note n_p le nombre d'itérations de l'algorithme du point fixe ($n_p \approx 3$). La nouvelle approximation $\mathbf{u}_{\ell+1}$ s'exprime alors (5.20) comme :

$$\mathbf{u}_{\ell+1}(t) = \mathbf{u}_\ell(t) + \sum_{i=1}^{m_\ell} \beta_i(t)\mathbf{\Lambda}_i + \lambda(t)\mathbf{\Lambda} \qquad \forall t \in I \tag{5.20}$$

$$= \sum_{i=1}^{m_\ell} \big(\underbrace{\lambda_i^\ell(t) + \beta_i(t)}_{\check{\lambda}_i^{\ell+1}(t)} \big)\mathbf{\Lambda}_i + \lambda(t)\mathbf{\Lambda} \tag{5.21}$$

Nous décidons de n'ajouter qu'au plus un mode à chaque itération de l'algorithme LATIN-PGD, quitte à tolérer une certaine erreur d'approximation de l'équation caractérisant la direction de recherche sur les premières itérations. Du fait de la structure même de notre algorithme, les forces internes généralisées qui apparaissent au second membre sont *de facto* évaluées de manière imparfaite au début de l'algorithme. Il paraît donc peu raisonnable de résoudre les premières équations globales d'équilibre avec une précision extrêmement fine – en générant d'emblée un grand nombre de modes – alors que ces modes ne seront probablement pas les plus optimaux pour représenter la solution à convergence.

Qui plus est, chaque couple étant déterminé séquentiellement de manière totalement indépendante, il n'est pas impossible qu'au cours de l'algorithme certains couples trouvés soient dépendants de couples précédents, déjà exhibés à une itération antérieure. Il est alors possible de rendre la décomposition plus optimale en rajoutant une étape d'orthogonalisation – ce que nous faisons – de sorte que chaque nouvelle correction soit systématiquement recherchée dans un espace orthogonal à la base existante. Suivant un processus de type Gram-Schmidt, on suppose que la base spatiale $\{\mathbf{\Lambda}_i\}_{1\leqslant i\leqslant m_\ell}$ de l'itération ℓ précédente est orthonormée au sens de l'énergie élastique. En réécrivant l'équation (5.20) sous la forme (5.22) suivante :

$$\mathbf{u}_{\ell+1}(t) = \sum_{i=1}^{m_\ell} \Big(\underbrace{\check{\lambda}_i^{\ell+1}(t) + \lambda(t)\langle \mathbf{\Lambda}, \mathbf{\Lambda}_i \rangle_{\mathbf{K}}}_{\lambda_i^{\ell+1}(t)} \Big)\mathbf{\Lambda}_i + \lambda(t)\Big(\mathbf{\Lambda} - \sum_{i=1}^{m_\ell} \langle \mathbf{\Lambda}, \mathbf{\Lambda}_i \rangle_{\mathbf{K}} \mathbf{\Lambda}_i\Big) \quad \forall t \in I \tag{5.22}$$

et notant $\check{\mathbf{\Lambda}} = \mathbf{\Lambda} - \sum_{i=1}^{m_\ell} \langle \mathbf{\Lambda}, \mathbf{\Lambda}_i \rangle_{\mathbf{K}} \mathbf{\Lambda}_i$ puis $\mathbf{\Lambda}_{m_{\ell+1}} = \big\|\check{\mathbf{\Lambda}}\big\|_{\mathbf{K}}^{-1} \check{\mathbf{\Lambda}}$ alors la correction apportée (5.23) s'écrit :

$$\Delta\mathbf{u}_{\ell+1}(t) = \sum_{i=1}^{m_\ell} \Big(\beta_i(t) + \lambda(t)\langle \mathbf{\Lambda}, \mathbf{\Lambda}_i \rangle_{\mathbf{K}}\Big)\mathbf{\Lambda}_i + \underbrace{\lambda(t) \big\|\check{\mathbf{\Lambda}}\big\|_{\mathbf{K}}}_{\lambda_{m_{\ell+1}}^{\ell+1}(t)} \mathbf{\Lambda}_{m_{\ell+1}} \qquad \forall t \in I \tag{5.23}$$

où la nouvelle base spatiale $\{\mathbf{\Lambda}_i\}_{1\leqslant i\leqslant m_{\ell+1}}$ est orthonormée au sens de l'énergie élastique. Le nouveau mode obtenu n'est alors retenu que s'il possède un poids non négligeable par rapport à la base $\{\mathbf{\Lambda}_i\}_{1\leqslant i\leqslant m_\ell}$ préalablement établie, c'est-à-dire qu'il est capable d'apporter de nouvelles informations.

Finalement, nous appliquons une étape de relaxation (5.24), ce qui permet d'assurer la convergence de l'algorithme :

$$\forall t \in I, \quad \mathbf{u}_{\ell+1}(t) = \mathbf{u}_\ell(t) + \mu_r \Delta\mathbf{u}_{\ell+1}(t) \tag{5.24}$$

où μ_r représente le coefficient de relaxation. L'algorithme est d'autant plus stable que la valeur de la relaxation est faible. *A contrario*, un coefficient de relaxation élevé assure un meilleur taux de convergence. Il faut donc trouver un compromis entre stabilité et performance – avec usuellement $0.5 \leqslant \mu_r \leqslant 0.95$ selon les cas.

REMARQUE 5.6 : Cette étape de relaxation apporte plus de stabilité mais, dans le même temps, conduit naturellement à s'écarter de la direction de recherche souhaitée.

Impact du choix de la direction de recherche. Du point de vue des performances, à chaque itération du point fixe, il est nécessaire d'inverser un système linéaire de taille $(n \times n)$ lors de l'évaluation de la fonction $\xi_{\mathbf{\Lambda}}(\lambda)$. Ceci entraîne une étape de factorisation coûteuse – dont on note n_f la complexité algébrique – si on utilise un solveur direct. Ainsi, il est intéressant de chercher à mutualiser cette opération sur l'ensemble des itérations LATIN-PGD, ce qui est notamment possible en considérant une direction de recherche constante et indépendante du temps. Un tel choix occasionne la factorisation de l'opérateur une seule fois pour l'ensemble du calcul, ce qui permet de limiter grandement la complexité algébrique liée à la génération d'un nouveau mode qui se résume alors à l'inversion du système triangulaire résultant – dont on note n_s la complexité algébrique. Considérer une direction de recherche constante et indépendante du temps peut s'avérer particulièrement pertinent dès que l'on traite des problèmes industriels – n grand – bien que ce choix puisse ralentir la vitesse de convergence par rapport à un opérateur tangent. Typiquement, pour des opérateurs *full* on a $n_f \equiv \mathcal{O}(\frac{1}{3}n^3) \gg n_s \equiv \mathcal{O}(2n^2)$. Ce rapport de force reste vrai pour nos opérateurs *sparse* bien que la complexité de chacune des opérations soit inférieure compte tenu du fait que l'opérateur ne possède que n_z termes non nuls – avec $n^2 \gg n_z \gg n$. Le Tableau 5.1 fournit un résumé de la complexité sous-jacente des différentes étapes en fonction du choix retenu pour la direction de recherche – pour une itération ℓ donnée dont on suppose que m_ℓ modes ont déjà été construits.

TABLE 5.1 – Comparaison des complexités algébriques selon différentes directions de recherche avec $\varepsilon = 1$ pour la toute première résolution de $\xi_{\mathbf{\Lambda}}(\lambda)$ $(m_\ell = 0)$ et $\varepsilon = 0$ autrement

	Étape préliminaire	**Génération d'un nouveau mode**
$\mathbf{H}(t)$	$\mathcal{O}\Big(n_t n_z m_\ell (m_\ell + 1)\Big)$	$\mathcal{O}\Big(2n_z n_t (m_\ell + 2n_p)\Big) + n_p(n_f + n_s)$
$\mathbf{K}$	$\mathcal{O}\Big(2n_t n m_\ell\Big)$	$\mathcal{O}\Big(2n n_t (m_\ell + n_p + \frac{3}{2}) + 2n_z(n_p + m_\ell)\Big) + \varepsilon n_f + n_p n_s$

Du point de vue du stockage en mémoire, choisir un opérateur $\mathbf{H}$ dépendant du temps s'avère également beaucoup plus gourmand en ressources, comparativement à un opérateur constant indépendant du temps. Au cours du calcul, il faut alors disposer d'un espace mémoire de taille $n_t n_z$ qui croît exponentiellement à mesure que le problème industriel s'enrichit. La problématique du stockage en mémoire est prédominante en bureau d'étude où chaque calcul ne peut nécessairement pas être mené sur de gigantesques *clusters*, bien que les solutions SaaS [1] apportent plus de flexibilité. Une des contraintes émanant du développement de nouvelles méthodes numériques réside dans la dualité performance *versus* empreinte mémoire. À ce titre, cette version faiblement intrusive de la méthode LATIN-PGD, destinée à accroître les performances du logiciel industriel, ne doit pas, dans le même temps, dégrader sensiblement l'espace mémoire nécessaire à la réalisation d'un calcul. Nous donnerons des valeurs numériques en Section-5.4.2 à titre d'exemple. Plusieurs solutions existent pour limiter le stockage en mémoire vive (RAM) :

- le choix de l'opérateur lui-même, en considérant une direction de recherche indépendante du temps ;
- l'emplacement mémoire, en décidant de stocker ponctuellement une partie de l'information sur disque dur (SSD) plutôt qu'en mémoire vive (RAM), au risque cependant de dégrader fortement les performances ;
- l'usage d'une méthode d'hyper-réduction telle que la méthode RPM [Capaldo *et al.*, 2017].

Ainsi, à moins que le problème soit fortement non-linéaire, ou bien qu'un niveau élevé de plasticité soit atteint, – auquel cas un opérateur $\mathbf{H}$ tangent dépendant du temps semble plus judicieux – dans la plupart

1. Software as a Service

des cas un opérateur $\mathbf{H}$ constant peut s'avérer suffisant et particulièrement efficace pour réduire de manière significative la complexité des opérations algébriques à mener, avec qui plus est une gestion parcimonieuse de la mémoire. Comme nous le verrons par la suite, l'ensemble des études réalisées dans le cadre de ces travaux a suscité une large supériorité quant à l'utilisation d'un opérateur $\mathbf{H}$ constant.

5.3.5 Critères mis en place

Différents critères guident le fonctionnement général de l'algorithme. Ces critères sont de fait des paramètres supplémentaires de la méthode dont le choix découle de l'analyse de campagnes d'essais numériques : ils ont été choisis pour satisfaire au mieux les critères de performance et de robustesse.

5.3.5.1 Indicateur d'erreur global ω_ℓ

Un critère basé sur la norme énergétique moyennée sur l'intervalle de temps (5.25) a été retenu. Ce critère permet de mesurer, en un certain sens, la distance géométrique entre les espaces $\mathbf{\Gamma}$ et $\mathbf{A}_d$:

$$\omega_\ell = \left[\frac{\|\mathbf{u}_{\ell+1} - \widehat{\mathbf{u}}_\ell\|^2_{\mathbf{K}}}{\frac{1}{2}\|\mathbf{u}_{\ell+1}\|^2_{\mathbf{K}} + \frac{1}{2}\|\widehat{\mathbf{u}}_\ell\|^2_{\mathbf{K}}} \right]^{0.5} \quad \text{avec} \quad \|\mathbf{u}_\ell\|^2_{\mathbf{K}} = \langle \mathbf{u}_\ell, \mathbf{K}\mathbf{u}_\ell \rangle_{\mathcal{I}_\mathcal{V}} = \int_I {}^t\mathbf{u}_\ell(t)\mathbf{K}\mathbf{u}_\ell(t)\,\mathrm{d}t \tag{5.25}$$

Le calcul s'arrête dès que le seuil de convergence η est atteint à savoir $\omega_\ell \leqslant \eta$ avec $\eta = 10^{-4}$ par défaut. Cette valeur a été validée numériquement afin de garantir une solution de bonne qualité comme nous le verrons dans les applications. Il est également possible d'ajouter un critère en effort (5.26) qui s'appuie directement sur le critère en résidu d'équilibre $\mathcal{E}_{\text{TESF}}$ (4.1) natif du logiciel Simcenter Samcef™ :

$$\omega^{\mathbf{f}}_\ell = \frac{\|\mathcal{E}_{\text{TESF}}\|_1}{n_t} \tag{5.26}$$

Ce critère présente l'avantage d'être aisément accessible puisqu'il intervient classiquement dans le logiciel industriel.

5.3.5.2 Choix d'augmentation du nombre de modes

Chaque génération d'un nouveau mode conduit inexorablement à l'inversion d'un système linéaire de grande taille – d'ordre n, le nombre de degrés de liberté spatiaux – ce qui représente une opération coûteuse à réaliser. Le critère qui détermine la pertinence de l'étape de mise à jour conduisant, ou non, à l'élaboration d'un nouveau mode, pilote ainsi directement la performance de l'algorithme. Ce critère doit donc être choisi minutieusement pour assurer à la fois performance et robustesse, cette dernière étant d'une importance capitale en environnement industriel pour pouvoir considérer tous types de problèmes non-linéaires. Nous avons envisagé plusieurs critères chacun ayant ses avantages et ses inconvénients.

Critère purement algébrique. Le premier critère est basé sur les variations relatives des fonctions temporelles lors de l'étape de mise à jour et présente l'avantage d'être aisément calculable. L'étape d'augmentation du nombre de modes n'est mise en œuvre que si l'équation (5.27) est vérifiée :

$$\frac{\alpha_\ell(1+m_\ell)}{n_t m_\ell} \sum_{i=1}^{m_\ell} \sum_{j=1}^{n_t} \left| \frac{\beta_i(t_j)}{\lambda_i^{up}(t_j) + \lambda_i^\ell(t_j)} \right| \leqslant \eta_{c_1} \tag{5.27}$$

où η_{c_1} caractérise un certain seuil et α_ℓ représente un coefficient de pondération pouvant varier au gré des itérations. Ce coefficient modérateur procure une propriété intéressante à savoir une tendance à générer de moins en moins de modes au fur et à mesure que ℓ augmente. Cependant, ce critère présente certains défauts : il ne prévient pas des phénomènes de permutations circulaires, au même titre qu'une forte variation des fonctions temporelles n'implique pas nécessairement que l'équation de la direction de recherche

– qui n'est vérifiée qu'en moyenne – soit correctement décrite. Pour éviter les phases de stagnation, nous pouvons jouer sur le coefficient α_ℓ en diminuant sa valeur à mesure que les itérations sans génération d'un nouveau mode s'enchaînent, dans la limite d'un certain seuil (5 itérations par exemple) où la génération d'un nouveau mode est alors forcée.

Critère d'erreur en direction de recherche. Le second critère s'articule autour de l'analyse de l'équation caractérisant la direction de recherche – qui n'est vérifiée qu'en moyenne. L'étape d'augmentation du nombre de modes est réalisée uniquement lorsque la condition (5.28) est satisfaite :

$$\frac{\max\limits_{t\,\in\, I}\left(\max\limits_{\boldsymbol{\mathcal{V}}}\left|\mathbf{f}_{\text{ext}}(t)-\mathbf{H}\Delta\mathbf{u}_{\ell+1}(t)-\widehat{\mathbf{f}}_{\text{int},\,\ell}(t)\right|\right)}{\max\limits_{t\,\in\, I}\left(\max\limits_{\boldsymbol{\mathcal{V}}}\left|\widehat{\mathbf{f}}_{\text{int},\,\ell}(t)-\mathbf{f}_{\text{ext}}(t)\right|\right)}\geqslant\eta_{c_2} \tag{5.28}$$

où η_{c_2} symbolise un certain seuil de tolérance. Ce critère permet d'appréhender l'erreur relative commise entre la direction de recherche souhaitée et celle effectivement empruntée. Typiquement, au début de l'algorithme, l'espace réduit est généralement trop pauvre pour représenter convenablement la direction de recherche, ce qui conduit à un enrichissement par l'ajout de modes supplémentaires. Puis, au fil des itérations, l'étape de mise à jour devient de plus en plus efficace permettant de s'affranchir de la génération de nouveaux modes. La Figure 5.3 fournit une interprétation géométrique de ce critère : si à l'issue de l'étape de mise à jour, la direction de recherche empruntée s'avère être trop éloignée de la direction souhaitée, alors nous procédons à la génération d'un nouveau mode ce qui permet de mieux satisfaire l'équation de la direction de recherche en moyenne.

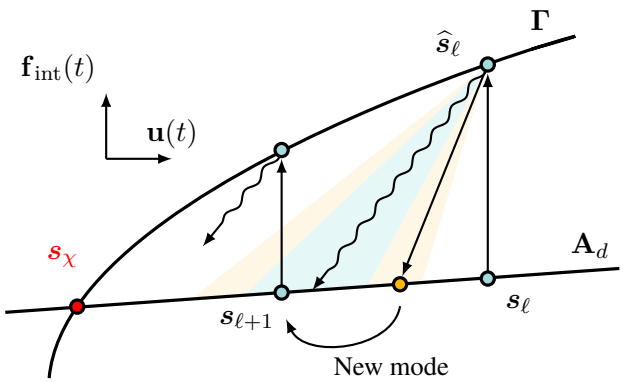


FIGURE 5.3 – Interprétation géométrique de la vérification de l'équation de la direction de recherche en moyenne

Critère basé sur les distances relatives aux espaces $\mathbf{A}_d$ et $\mathbf{\Gamma}$. Le troisième critère s'intéresse à l'évolution des distances entre les espaces $\mathbf{A}_d$ et $\mathbf{\Gamma}$ entre deux itérations successives. L'étape d'augmentation du nombre de modes est requise dès que l'inégalité (5.29) suivante est constatée :

$$\frac{d_\ell-d_{\ell+1}}{d_\ell}\leqslant\eta_{c_3} \tag{5.29}$$

où η_{c_3} fixe un certain seuil et la distance d_ℓ est donnée par la formule (5.30) ci-contre :

$$d_\ell=\frac{\|\boldsymbol{s}_\ell-\widehat{\boldsymbol{s}}_\ell\|_{\mathbf{K}}}{\|\boldsymbol{s}_\ell+\widehat{\boldsymbol{s}}_\ell\|_{\mathbf{K}}}\quad\text{avec}\quad\|\boldsymbol{s}_\ell\|_{\mathbf{K}}^2=\int_I{}^t\mathbf{u}_\ell(t)\mathbf{K}\mathbf{u}_\ell(t)+{}^t\mathbf{f}_{\text{int},\,\ell}(t)\mathbf{K}^{-1}\mathbf{f}_{\text{int},\,\ell}(t)\;\mathrm{d}t \tag{5.30}$$

Ce type de critère a déjà été utilisé dans [Heyberger *et al.*, 2012] par exemple avec $\eta_{c_3}=0.2$ pour des problèmes de thermique instationnaire.

5.4 Illustration sur un cas-test académique

Sous le prisme d'un premier cas-test, nous illustrons dans cette partie le fonctionnement de la méthode LATIN-PGD faiblement intrusive implémentée dans le logiciel industriel Simcenter Samcef™.

5.4.1 Description du cas-test

Ce premier cas-test académique correspond à la sollicitation en traction d'un barreau entaillé. Le chargement est appliqué à l'une des extrémités ($z = L$) tandis que l'autre extrémité reste encastrée ($z = 0$) comme illustré en Figure 5.4. Le maillage est composé d'éléments volumiques tétraédriques du second degré et recense $n = 9 \times 10^4$ dégrés de liberté. La source des non-linéarités se situe au niveau de la loi de comportement matériau supposée élasto-visco-plastique. La déformation plastique cumulée p est notamment gouvernée par une loi d'évolution de type Norton (5.31) écrite sous la forme :

$$\dot{p} = \left\langle \frac{J_2(\boldsymbol{\sigma}) - \sigma_0}{k p^{1/m}} \right\rangle_+^n \tag{5.31}$$

avec k, m et n trois paramètres scalaires, auxquels s'ajoute le seuil de plasticité σ_0 ainsi que les paramètres élastiques usuels pour un matériau supposé isotrope – module d'Young E et coefficient de Poisson ν. Le Tableau 5.2 recense les différentes valeurs retenues pour ces paramètres.

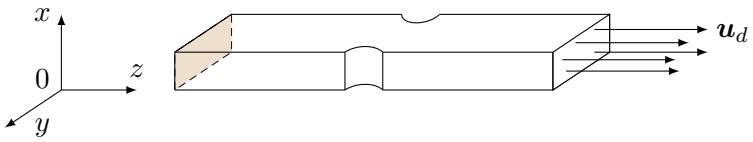


FIGURE 5.4 – Premier cas académique : barreau entaillé

TABLE 5.2 – Paramètres matériaux de la loi élasto-visco-plastique de type Norton

E [MPa]	ν [-]	σ_0 [MPa]	m [-]	n [-]	k [MPa.s$^{1/n}$]
2.0×10^5	0.28	10	12.1	15	713

5.4.2 Résultats numériques et analyses des performances

Ce cas-test étant relativement simple, il ne permet bien sûr pas d'analyser convenablement les performances au regard d'un éventuel gain en temps de calculs. Ce paragraphe s'intéresse plus particulièrement à l'influence des différents paramètres de la méthode LATIN-PGD sur le fonctionnement général de l'algorithme et sur le taux de convergence – en nombre d'itérations. Il convient de noter que les choix permettant de minimiser le nombre d'itérations ne conduisent pas nécessairement à une minimisation du temps global de simulation. Nous reviendrons sur ce point au Chapitre-9 lors de l'étude des cas industriels.

Influence du choix de la direction de recherche. L'un des paramètres majeurs de la méthode LATIN-PGD se situe dans le choix de la direction de recherche, c'est-à-dire dans le choix de l'opérateur $\mathbf{H}$ pouvant éventuellement dépendre du temps. Matérialisées sur la Figure 5.5, nous avons étudié trois directions particulières :

- la direction tangente $\mathbf{H}_\ell(t)$ dépendante du temps $t \in I$ évaluée à chaque itération LATIN-PGD ;
- la direction tangente $\mathbf{H}_0(t)$ dépendante du temps $t \in I$ calculée uniquement à la première itération LATIN-PGD, et conservée pour l'ensemble des itérations suivantes ;
- la direction de recherche élastique $\mathbf{K}$ constante et indépendante du temps.

À titre de comparaison, ont été également ajoutées les courbes de décroissance de l'indicateur d'erreur, d'une part pour la méthode LATIN pure (sans technique PGD mise en œuvre) et, d'autre part, en forçant la génération d'un mode à chaque itération sans effectuer d'étape de mise à jour au préalable – avec $\mathbf{H}_\ell$ comme direction de recherche. Sans surprise, avoir recours à une direction tangente $\mathbf{H}_\ell$ fournit la méthode la plus efficace au regard du nombre d'itérations.

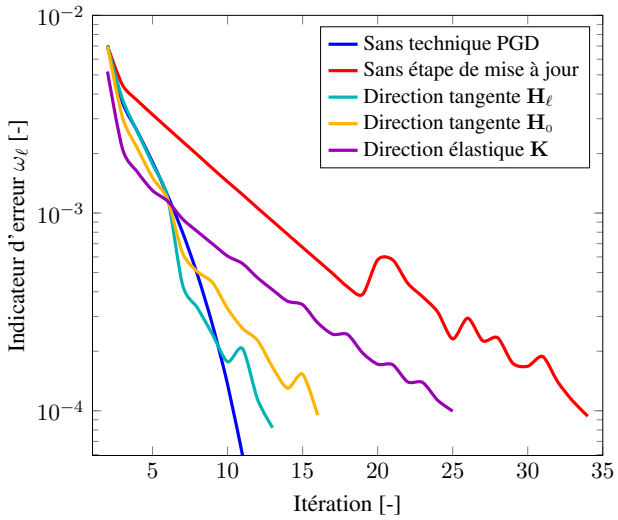


FIGURE 5.5 – Influence du choix de la direction de recherche sur le taux de convergence en fonction du nombre d'itérations

Le Tableau 5.3 expose quelques détails supplémentaires sur les différents calculs réalisés. Nous observons notamment une forte disparité des ressources en mémoire requises. Comparée aux ressources classiques demandées par le logiciel industriel dans sa version commerciale (algorithme de Newton-Raphson), l'introduction de la méthode LATIN-PGD nécessite d'apporter un supplément mémoire : un opérateur tangent entraîne une augmentation de près de 70% de la mémoire vive lorsqu'un opérateur indépendant du temps ne requière que 30% d'augmentation. Par ailleurs, comme nous l'avons déjà évoqué, ce premier cas-test académique demeure trop simple pour examiner les performances. Le propre de la méthode LATIN-PGD étant de réduire sensiblement le nombre de systèmes linéaires à résoudre, encore faut-il que l'étape globale soit largement prépondérante devant l'étape locale, ce qui n'est pas le cas sur cet exemple. On peut toutefois noter que même si deux fois plus d'itérations sont nécessaires pour atteindre la convergence entre les opérateurs $\mathbf{H}_\ell$ et $\mathbf{K}$, les temps de calculs demeurent similaires.

TABLE 5.3 – Caractéristiques relevées pour les différents calculs

		$\mathbf{H}_\ell$	$\mathbf{H}_0$	$\mathbf{K}$
Nombre d'itérations	[-]	13	16	25
Nombre de modes générés	[-]	6	8	10
***** Mémoire totale *****	[GB]	3.83	3.83	2.95
Ressources classiques	[GB]	2.27	2.27	2.27
Supplément LATIN-PGD	[GB]	1.56	1.56	0.68
***** Temps total *****	[s]	113	128	127
Prorata étape locale	[%]	71.3	69.4	89.2
Prorata étape globale	[%]	28.7	30.6	10.8

Influence du critère d'arrêt. Le calcul de l'indicateur d'erreur ω_ℓ permet d'estimer l'état de convergence. Afin d'avoir une comparaison plus explicite, nous avons juxtaposé sur la Figure 5.6 l'évolution de l'indicateur d'erreur et l'évolution de l'erreur vraie e_χ donnée ci-après (5.32) en fonction des itérations :

$$e_\chi = \max_{t \in I} \left[\frac{\max_{\boldsymbol{\nu}} |\mathbf{u}(t) - \mathbf{u}_\chi(t)|}{\frac{1}{n} \|\mathbf{u}_\chi(t)\|_1} \right] \tag{5.32}$$

où $\mathbf{u}_\chi$ représente la solution de référence donnée par le logiciel Simcenter Samcef™ dans sa version commerciale. On observe ainsi qu'une valeur $\omega_\ell \approx 10^{-4}$ de l'indicateur d'erreur coïncide avec une erreur vraie e_χ de l'ordre de 1%, occasionnant l'obtention d'une solution de bonne qualité.

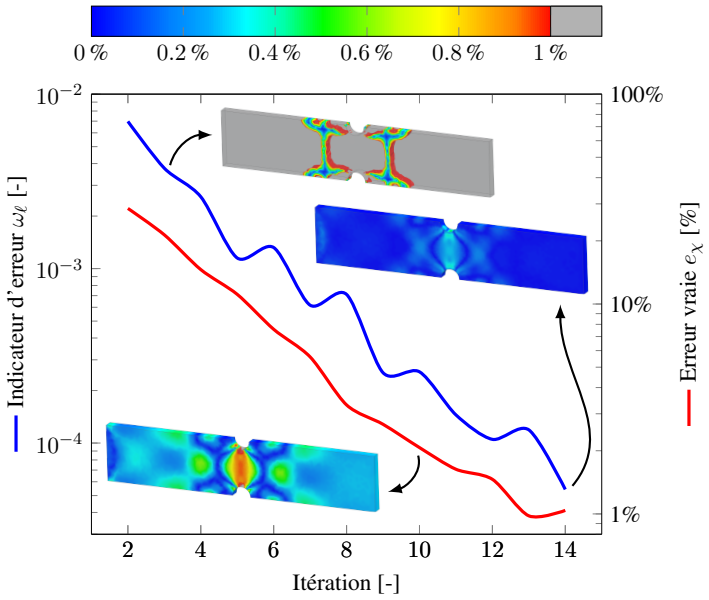


FIGURE 5.6 – Correspondance entre l'indicateur d'erreur ω_ℓ et l'erreur vraie e_χ en fonction des itérations avec illustrations spatiales des niveaux d'erreur relative $\frac{|\mathbf{u} - \mathbf{u}_\chi|}{|\mathbf{u}_\chi|}$ (en %) pour le pas de temps le plus chargé

En fonction du besoin, il est également possible de considérer des approximations plus grossières en rehaussant le critère d'arrêt. C'est une des forces de la stratégie non-incrémentale LATIN-PGD de pouvoir fournir très rapidement une approximation de la solution sur l'ensemble de l'espace et du temps. En effet, bien souvent en environnement industriel, la ressource du *temps* disponible, dictée par les impératifs économiques, n'est pas extensible. Ainsi, la méthode LATIN-PGD est susceptible d'apporter la meilleure approximation possible dans le budget *temps* imparti ce qui permet d'envisager des prises de décisions éclairées au moment opportun. Le cas échéant, il est ensuite possible d'accroître ce budget *temps* disponible dans l'optique d'affiner la qualité de l'approximation obtenue.

Chapitre 6

Extension de la méthode LATIN-PGD aux non-linéarités géométriques

Ce chapitre complète la formulation proposée pour l'étendre aux non-linéarités géométriques. On s'intéresse tout spécialement au cas des grandes rotations où l'on montre que la direction de recherche de descente doit être minutieusement choisie afin d'assurer la convergence. Cette extension reste en tout point compatible avec la gestion des non-linéarités matériaux vues au chapitre précédent, ce qui permet d'envisager des couplages entre non-linéarités de différentes natures.

Sommaire

Dans ce chapitre, seuls les éléments nouveaux ou différents par rapport au chapitre précédent de la méthode LATIN-PGD faiblement intrusive sont explicités. On pourra ainsi se référer au Chapitre-5 lorsque cela s'avèrera nécessaire. Sauf mention contraire, les mêmes notations sont conservées. On suppose de plus que l'on reste dans le cadre de déplacements « modérés », en dehors des zones d'instabilité.

6.1 Équations du problème en grandes transformations

En plus des éventuelles non-linéarités liées au comportement des matériaux, on considère aussi dorénavant la prise en compte des non-linéarités géométriques. La formulation du problème (5.1) à résoudre demeure inchangée, au détail près que le champ de déplacement $\mathbf{u}$ caractérise non plus uniquement la déformation de la matière mais également la variation de la géométrie. Suivant une approche Lagrangienne,

la configuration initiale est conservée comme configuration de référence.

Aucune modification dans l'énoncé des principes P1 et P2 de la méthode LATIN-PGD n'est nécessaire, la séparation des difficultés et la définition des espaces supplémentaires $\boldsymbol{\Upsilon}^+$ et $\boldsymbol{\Upsilon}^-$, caractérisant les directions de recherche, restant valides en grandes transformations. C'est d'ailleurs un point fondamental qui a été recherché afin d'assurer un maximum de robustesse et de généralité à la méthodologie proposée : sous un même formalisme fonctionnel, et sans avoir à modifier l'architecture déjà mise en œuvre, nous sommes alors capables d'adresser à la fois les non-linéarités matériaux, mais aussi les non-linéarités géométriques. Les quantités généralisées $\boldsymbol{s} = (\mathbf{u}, \mathbf{f}_{\text{int}}) \in \mathcal{I}_{\boldsymbol{\mathcal{V}}} \times \mathcal{I}_{\boldsymbol{\mathcal{V}}}$ – déplacements et forces généralisés respectivement – choisies comme inconnues du problème (5.2 – 5.3) sont universelles dans le sens où ce sont des grandeurs macroscopiques intervenant avec n'importe quel type de non-linéarités. Seules les variables locales derrière – variables internes, choix des mesures de déformation, etc. – varient en fonction de l'application mais demeurent cachées sous l'opérateur $\boldsymbol{\mathcal{A}}_u$ géré au niveau de l'étape locale, ce qui permet de s'appuyer pleinement sur l'ensemble des outils du logiciel industriel préexistants. Ainsi, l'unique modification à prévoir se situe dans l'adaptation de certains paramètres, typiquement le choix de l'opérateur $\mathbf{H}$ définissant la direction de recherche de descente, comme nous allons le voir par la suite.

6.2 Particularités du non-linéaire géométrique

On reprend ci-après succinctement les grandes étapes de la méthode LATIN-PGD dans sa version faiblement intrusive formulée précédemment en précisant les quelques particularités supplémentaires intervenant en non-linéaire géométrique.

6.2.1 Initialisation

L'initialisation de l'algorithme s'effectue toujours à partir de n'importe quelle approximation $\boldsymbol{s}_0 \in \mathbf{A}_d$. En fonction du type de non-linéarités impliquées, nous avons retenu plusieurs choix. En présence de non-linéarités géométriques uniquement, un choix aisément accessible consiste à utiliser une initialisation élastique – linéaire géométrique. Le paramètre numérique NLIS de Simcenter Samcef™ permet en effet, directement au niveau de la mise en données du problème, de définir si l'on souhaite tenir compte, ou non, des non-linéarités géométriques. Par ailleurs, lorsqu'aux non-linéarités géométriques s'ajoutent les non-linéarités matériaux, il est alors envisageable d'initialiser l'algorithme de plusieurs manières :

- par l'intermédiaire d'une approximation élastique – en linéaire géométrique ;
- grâce à une approximation prenant en compte les non-linéarités matériaux – en linéaire géométrique ;
- au moyen d'une approximation élastique – prenant en considération les non-linéarités géométriques.

Qui plus est, lors de la réalisation d'études paramétriques, n'importe quelle solution déjà calculée peut servir d'initialisation.

6.2.2 Étape locale

Nous nous appuyons une fois de plus sur l'intelligence du logiciel industriel qui possède déjà tous les outils nécessaires aux calculs des différentes quantités intervenant en non-linéaire géométrique. La méthodologie *boîte noire* mise en œuvre lors de l'étape locale, consistant à fournir le champ de déplacements généralisés en entrée pour récupérer en sortie les efforts internes généralisés, englobe de fait l'ensemble des opérations liées aux non-linéarités matériaux et géométriques. On rappelle que c'est l'un des points forts de cette version LATIN-PGD faiblement intrusive que de pouvoir réexploiter autant que possible

toute la richesse d'un logiciel industriel tel que Simcenter Samcef™. En particulier, nous explicitons ci-dessous quelques étapes incontournables intervenant en non-linéaire géométrique que nous réutilisons sans aucune modification.

Choix de la mesure de déformation. Lors de l'écriture des lois de comportement matériaux, définissant une relation entre un état de contrainte et un état de déformation, le choix de la mesure de déformation n'est pas unique. Par exemple, alors que les lois hyper-élastiques découlent d'une fonction densité d'énergie de déformation écrite à partir soit des invariants, soit des directions principales, du tenseur des déformations de Green-Lagrange $\mathbb{E}$, les lois plastiques sont généralement définies par l'intermédiaire du tenseur des déformations naturelles $\boldsymbol{\varepsilon}$ issu du tenseur des taux de déformation eulérien $\mathbb{D} \equiv \dot{\boldsymbol{\varepsilon}}$ afin de respecter l'incompressibilité en grandes déformations. Ainsi, chaque loi de comportement est écrite suivant une hypothèse sur la mesure de déformation utilisée dont les différents choix disponibles dans Simcenter Samcef™ sont rassemblés dans le Tableau 6.1 – où la notation $\bullet_{\mathbf{R}_p}$ désigne des quantités tournées, indépendantes de la rotation de la matière. Ces lois sont intégrées localement en chaque point de Gauss de chaque élément.

TABLE 6.1 – Différentes mesures de déformation disponibles dans Simcenter Samcef™

Mesures de déformation			**Contrainte associée**
Green-Lagrange	$\mathbb{E}$	$\mathbb{S}$	2^{nd} tenseur de Piola-Kirchhoff
Naturelles	$\boldsymbol{\gamma}_{\boldsymbol{\varepsilon}}$	$\boldsymbol{\sigma}_{\mathbf{R}_p}$	Cauchy
Naturelles	$\boldsymbol{\gamma}_{\boldsymbol{\varepsilon}}$	$\boldsymbol{\tau}_{\mathbf{R}_p}$	Kirchhoff
Biot	$\boldsymbol{\alpha}_{\boldsymbol{\varepsilon}}$	$\boldsymbol{h}$	Biot

Dans un souci de robustesse et de facilité d'implémentation, c'est le tenseur des déformations de Green-Lagrange $\mathbb{E}$ – couplé au second tenseur des contraintes de Piola-Kirchhoff $\mathbb{S}$ – qui est utilisé par défaut dans le logiciel Simcenter Samcef™ au niveau local, à l'échelle de l'élément, et ce quelle que soit la mesure de déformation sous-jacente constitutive de la loi de comportement. Ce choix permet de respecter le principe d'objectivité en grandes transformations dans la mesure où, d'une part, toutes les mesures de déformation peuvent se baser sur le tenseur symétrique des déformations pures $\mathbf{U}$ intervenant dans la décomposition polaire $\mathbb{F} = \mathbf{R}_p\mathbf{U}$ du tenseur gradient de la transformation, avec $\mathbf{R}_p$ l'opérateur orthogonal de rotation pure, et, d'autre part, le tenseur des déformations de Green-Lagrange $\mathbb{E}$ et le tenseur des déformations pures $\mathbf{U}$ possèdent les mêmes directions principales. On peut également montrer qu'il existe un lien (6.1) entre les déformations principales de $\mathbb{E}$ et celles de $\mathbf{U}$, notées respectivement $\left(\lambda_i^{\mathbb{E}}\right)_{1\leqslant i\leqslant 3}$ et $\left(\lambda_i^{\mathbf{U}}\right)_{1\leqslant i\leqslant 3}$ dans la suite :

$$\forall i \in [\![1,3]\!], \quad \lambda_i^{\mathbf{U}} = \sqrt{1+2\lambda_i^{\mathbb{E}}} \tag{6.1}$$

Ainsi, quelle que soit la mesure de déformation retenue pour l'écriture de la loi de comportement, la simple connaissance du tenseur des déformations de Green-Lagrange $\mathbb{E}$ est suffisante pour pouvoir remonter aux autres mesures de déformation – fonction de $\mathbf{U}$ excluvisement – lorsque cela s'avère nécessaire. Plus précisément, ces mesures de déformation appartiennent toutes à la même famille de tenseurs $\left(\mathbb{E}_n\right)_{n\in\mathbb{Z}^*}$ symétriques (6.2) du second ordre qui possèdent comme propriété de s'annuler pour tout mouvement de corps rigide :

$$\forall n \in \mathbb{Z}^*, \quad \mathbb{E}_n = \frac{1}{n}\Big(\mathbf{U}^n - \mathbb{I}\Big) \tag{6.2}$$

De plus, en définissant le logarithme d'un tenseur $\mathbf{U}$ symétrique défini positif de la manière (6.3) suivante, où $\left(\lambda_i^{\mathbf{U}}, \mathbf{V}_i^{\mathbf{U}}\right)_{1\leqslant i\leqslant 3}$ caractérisent les valeurs propres et les vecteurs propres de $\mathbf{U}$ – respectivement

déformations principales et directions principales de déformation :

$$\text{Si} \quad \mathbf{U} = \sum_{i=1}^{3} \lambda_i^{\mathbf{U}} \mathbf{V}_i^{\mathbf{U}} \otimes \mathbf{V}_i^{\mathbf{U}} \qquad \text{alors} \quad \ln(\mathbf{U}) = \sum_{i=1}^{3} \ln(\lambda_i^{\mathbf{U}}) \mathbf{V}_i^{\mathbf{U}} \otimes \mathbf{V}_i^{\mathbf{U}} \tag{6.3}$$

alors, par extension pour $n = 0$, on définit également $\mathbb{E}_0 = \ln(\mathbf{U})$. En reprenant les différentes mesures établies dans le Tableau 6.1, on aboutit aux formules (6.4) ci-dessous :

$$(\mathbb{E}_0) \quad \boldsymbol{\gamma}_{\varepsilon} = \ln(\mathbf{U}) \qquad (\mathbb{E}_1) \quad \boldsymbol{\alpha}_{\varepsilon} = \mathbf{U} - \mathbb{I} \qquad (\mathbb{E}_2) \quad \mathbb{E} = \frac{1}{2}({}^t\mathbf{U}\mathbf{U} - \mathbb{I}) \tag{6.4}$$

où la contrainte associée à chaque mesure de déformation est déterminée par équivalence énergétique.

REMARQUE 6.1 : La vitesse des déformations conjuguée du tenseur des contraintes de Cauchy tourné $\boldsymbol{\sigma}_{\mathbf{R}_p} = {}^t\mathbf{R}_p \boldsymbol{\sigma} \mathbf{R}_p = J^{-1}\mathbf{U}\mathbb{S}\mathbf{U}$, qui s'exprime sous la forme $\dot{\boldsymbol{\gamma}}_{\varepsilon} = \left[\dot{\mathbf{U}}\mathbf{U}^{-1}\right]_{\mathrm{S}}$ par dualité, n'est *a priori* pas intégrable analytiquement de manière exacte. Par conséquent, on cherche une approximation de cette mesure des déformations en postulant la loi $\boldsymbol{\gamma}_{\varepsilon} = \ln(\mathbf{U})$ dont on peut vérifier *a posteriori* que l'intégration est exacte dès que les contraintes et les déformations possèdent les mêmes directions principales dans un repère lié à la matière. Dans le cas général, par développement en séries, on montre que l'approximation est vérifiée jusqu'au deuxième ordre.

En résumé, en disposant pour chaque élément de la connaissance des déplacements généralisés $\mathbf{u}$ aux nœuds, on est alors capable d'évaluer aux différents points de Gauss de l'élément le tenseur des déformations de Green-Lagrange $\mathbb{E}$ et le tenseur gradient de la transformation $\mathbb{F}$. Ensuite, par intégration des lois de comportement matériaux, on détermine le tenseur des contraintes associé $\mathbb{S}$ en réalisant ponctuellement, le cas échéant, un changement de mesures de déformation via le tenseur des déformations pures $\mathbf{U}$ et sa décomposition en valeurs propres – les lois constitutives n'étant pas nécessairement écrites sous le même formalisme $(\mathbb{E}, \mathbb{S})$ de l'élément. Finalement, on remonte aux efforts internes généralisés $\mathbf{f}_{\text{int}}$ aux nœuds de l'élément par formules de quadrature. La Figure 6.1 récapitule ce cheminement dans un organigramme où la notation $\underset{\sim}{\mathbf{U}}$ sera explicitée dans le paragraphe qui suit.

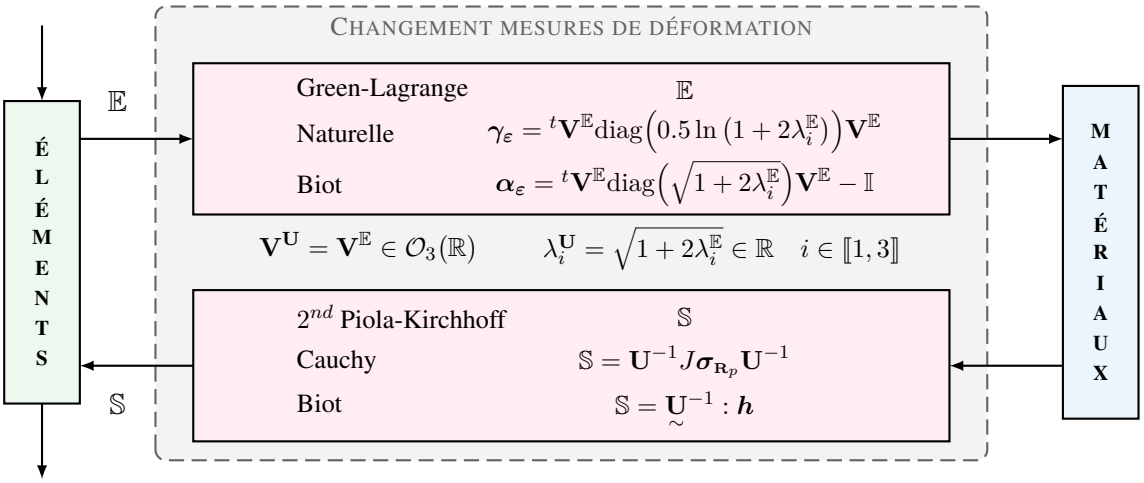


FIGURE 6.1 – Gestion des différentes mesures de déformation

REMARQUE 6.2 : Lorsque les déformations sont petites, le tenseur symétrique des déformations pures $\mathbf{U}$ est quasiment unitaire, ce qui rend les différentes mesures de déformation équivalentes.

Contribution de la raideur géométrique. Comme évoqué au Chapitre-1, la prise en compte des non-linéarités géométriques apporte une contribution supplémentaire à l'opérateur de rigidité tangent $\mathbf{K}_{\text{T}}$ éléments finis. L'impact de la variation de la géométrie intervient majoritairement à deux endroits : (i) au niveau du respect de la cohérence des différentes mesures de déformation lors de l'intégration des lois de comportement matériaux ; (ii) au niveau de la construction proprement dite de l'opérateur tangent dans le formalisme éléments finis grâce aux fonctions de forme de l'élément.

(i) Supposons par exemple que la loi de comportement définisse une relation $\boldsymbol{h} = \boldsymbol{\mathcal{K}_h} : \boldsymbol{\alpha_\varepsilon}$ entre les tenseurs des déformations et des contraintes de Biot. La raideur tangente cohérente $\boldsymbol{\mathcal{K}_h}$ est donc connue dans le formalisme de Biot. Cependant, au niveau de l'élément, c'est la mesure des déformations basée sur le tenseur des déformations de Green-Lagrange $\mathbb{E}$ qui nous intéresse, ce qui nécessite de déduire la loi tangente $\boldsymbol{\mathcal{K}}_{\mathbb{S}}$ qui relie le second tenseur des contraintes de Piola-Kirchhoff $\mathbb{S}$ au tenseur des déformations de Green-Lagrange $\mathbb{E}$ sous la forme $\mathbb{S} = \boldsymbol{\mathcal{K}}_{\mathbb{S}} : \mathbb{E}$. Sachant que la contrainte $\boldsymbol{h}$ associée à $\boldsymbol{\alpha_\varepsilon}$ est définie par la formule $\boldsymbol{h} = \left[\mathbb{S}\mathbf{U}\right]_{\text{S}}$, il s'ensuit la relation (6.5) qui lie les deux lois tangentes :

$$\boldsymbol{\mathcal{K}}_{\mathbb{S}} = \underset{\sim}{\mathbf{U}}^{-1} : \Big(\boldsymbol{\mathcal{K}_h} - \underset{\sim}{\mathbb{S}}\Big) : \underset{\sim}{\mathbf{U}}^{-1} \tag{6.5}$$

où les quantités $\underset{\sim}{\bullet}$ désignent à chaque fois l'unique tenseur symétrique d'ordre 4 vérifiant l'égalité (6.6) donnée ci-contre :

$$\underset{\sim}{\mathbf{U}} : \mathbb{S} = \frac{1}{2}\Big(\mathbf{U}\mathbb{S} + \mathbb{S}\mathbf{U}\Big) = \underset{\sim}{\mathbb{S}} : \mathbf{U} \tag{6.6}$$

Au sein de la relation (6.5), on distingue qu'à la rigidité classique $\boldsymbol{\mathcal{K}_h}$ vient s'ajouter un terme de rigidité géométrique $\underset{\sim}{\mathbb{S}}$ qui découle du changement de mesures de déformation. Des relations équivalentes peuvent être établies pour les autres mesures de déformation, que nous ne détaillons pas ici par souci de concision.

(ii) Comme nous l'avons déjà évoqué au Chapitre-1 (cf. Section-1.3.1.3), la variation de la géométrie impacte également l'élaboration de la matrice tangente élémentaire $\mathbf{K}_{\text{T}}^{[e]}$ où une rigidité géométrique $\mathbf{K}_g^{[e]}$ supplémentaire vient s'additionner à la raideur $\mathbf{K}_m^{[e]}$ classique issue du comportement matériaux. On rappelle la formule en (6.7) pour faciliter la lecture où la matrice élémentaire $\mathbf{C}^{[e]}$ désigne de fait le pendant du tenseur $\boldsymbol{\mathcal{K}}_{\mathbb{S}}$ symétrique d'ordre 4 sous un formalisme matriciel :

$$\mathbf{K}_{\text{T}}^{[e]} = \underbrace{\int_{\Omega_e} {}^t\mathbf{B}^{[e]}\mathbf{C}^{[e]}\mathbf{B}^{[e]}\ \mathrm{d}\Omega_e}_{\mathbf{K}_m^{[e]}} + \underbrace{\int_{\Omega_e} \frac{\partial\, {}^t\mathbf{B}^{[e]}}{\partial \mathbf{u}^{[e]}}\mathbb{S}^{[e]}\ \mathrm{d}\Omega_e}_{\mathbf{K}_g^{[e]}} \tag{6.7}$$

Ce second apport intervient cette fois-ci à l'échelle de l'élément.

Que ce soit lors de l'intégration des lois matériaux ou à l'échelle locale de l'élément, l'ensemble de ces contributions sont évaluées automatiquement par le logiciel Simcenter Samcef™ dont nous récupérons directement la raideur tangente assemblée globale $\mathbf{K}_{\text{T}}$ à l'issue de l'étape locale – éventuellement en chacun des pas de temps. Comme l'ensemble des opérations est réalisé localement en chaque point de Gauss de chaque élément, il n'est pas possible dans notre approche de type *boîte noire* faiblement intrusive de pouvoir discerner puis scinder les contributions. L'opérateur tangent assemblé est nécessairement récupéré d'un seul tenant.

REMARQUE 6.3 : L'implémentation numérique réalisée dans Simcenter Samcef™ n'est pas la seule manière de faire. Dans d'autres codes, en grandes déformations, on préfère actualiser la géométrie et travailler en vitesse de déformations. Ceci n'a pas d'impact sur la méthodologie LATIN-PGD faiblement intrusive proposée dès que l'on est capable de récupérer les efforts internes généralisés et les raideurs associées à l'issue de l'étape locale.

6.2.3 Étape linéaire

La résolution de l'étape linéaire s'opère par l'intermédiaire de la méthode PGD espace-temps classique où seulement quelques ajustements mineurs doivent être effectués lors de la prise en compte des non-linéarités géométriques.

6.2.3.1 Choix de la direction de recherche

Afin d'assurer la convergence de la méthode en grandes transformations, la direction de recherche doit être minutieusement choisie. En particulier, la raideur géométrique joue un rôle important en grandes rotations où elle introduit un terme de rappel sur la partie antisymétrique du gradient de la transformation. Si ce terme de rappel n'est pas du bon ordre de grandeur, le schéma de résolution peut devenir instable comme nous l'illustrerons sur le cas-test académique (cf. Section-6.3). Plusieurs choix de direction de recherche ont été testés : l'opérateur tangent $\mathbf{H}_\ell(t)$ complet – dépendant du temps – assurant un maximum de robustesse, mais également des versions à opérateur constant – indépendant du temps – permettant d'asseoir les performances de l'algorithme. L'opérateur constant est obtenu par combinaison linéaire d'opérateurs tangents à différents instants t, par technique SVD avec troncature au premier ordre, ou encore par l'intermédiaire d'une moyenne temporelle. Cependant, ces dernières versions – direction de recherche constante, indépendante du temps – n'ont pas montré de résultats aussi concluants que ceux qui ont été obtenus en présence de non-linéarités matériaux uniquement. La robustesse demeurant un point primordial dans l'élaboration de la stratégie proposée, nous n'avons pas retenu ces versions à opérateurs constants au profit de versions plus stables basées sur l'opérateur tangent $\mathbf{H}_\ell(t)$.

6.2.3.2 Résolution PGD

Bâties sur une direction de recherche tangente, deux versions ont été retenues en grandes transformations. Nous détaillons ci-après dans les grandes lignes chacune d'entre elles.

Version analogue au cas tangent $\mathbf{H}_\ell(t)$. Cette première version est rigoureusement la même que celle présentée au chapitre précédent en présence de non-linéarités matériaux – direction de recherche tangente $\mathbf{H}_\ell(t)$ dépendante du temps, évoluant au gré des itérations. La seule différence se situe au niveau du contenu de l'opérateur tangent $\mathbf{H}_\ell(t)$ qui englobe de fait une contribution supplémentaire liée aux non-linéarités géométriques – évaluées à l'étape locale. Cette contribution est calculée automatiquement par le logiciel industriel sans aucune modification particulière à apporter. On pourra se référer au Chapitre-5 pour davantage de détails.

Version avec conditionneur $\mathbf{H}_c$ constant. Cette seconde version vise à tirer profit d'un opérateur constant pour réduire la complexité algébrique des différentes opérations à mener. Cette version se base toujours sur une direction de recherche tangente $\mathbf{H}_\ell(t)$ mais l'étape d'augmentation du nombre de modes est réalisée par l'intermédiaire d'un conditionneur $\mathbf{H}_c$ constant et symétrique, ce qui permet de s'affranchir de l'étape coûteuse de factorisation à chaque nouveau mode généré. Plus précisément, à une itération ℓ donnée de l'algorithme, où on suppose que les m_ℓ premiers modes $\{(\mathbf{\Lambda}_i, \lambda_i^\ell)\}_{1\leqslant i\leqslant m_\ell} \in \{\boldsymbol{\mathcal{V}} \times \mathcal{I}_{\mathbb{R}}\}^{m_\ell}$ ont déjà été construits, on cherche une nouvelle approximation du problème (5.8) sous forme à variables séparées. L'équation de la direction de recherche (6.8) – en notant $\boldsymbol{\mathcal{Q}} = \mathbf{f}_{\text{ext}} - \widehat{\mathbf{f}}_{\text{int},\,\ell}$ le résidu d'équilibre :

$$\forall t \in I, \qquad \mathbf{H}_\ell(t)\Big(\mathbf{u}_{\ell+1}(t) - \mathbf{u}_\ell(t)\Big) = \boldsymbol{\mathcal{Q}}(t) \tag{6.8}$$

n'est invariablement vérifiée qu'en moyenne, ce qui conduit à la résolution de n_t petits systèmes linéaires (6.9) – de taille $(m_\ell \times m_\ell)$ – lors de l'étape préliminaire de mise à jour des fonctions temporelles :

$$\forall t \in I,\ \forall j \in [\![1, m_\ell]\!], \quad {}^t\mathbf{\Lambda}_j \left[\mathbf{H}_\ell(t)\Big(\sum_{i=1}^{m_\ell} \beta_i(t)\mathbf{\Lambda}_i\Big) - \boldsymbol{\mathcal{Q}}(t)\right] = 0 \tag{6.9}$$

Les nouvelles fonctions temporelles $\{\beta_i\}_{1\leqslant i\leqslant m_\ell}$ définissent alors classiquement la nouvelle approximation (6.10) sous la forme :

$$\mathbf{u}_{\ell+1}(t) = \mathbf{u}_\ell(t) + \sum_{i=1}^{m_\ell} \beta_i(t)\boldsymbol{\Lambda}_i \qquad \forall t \in I \tag{6.10}$$

À ce niveau-ci, il n'y a pas de différence par rapport à la première version. C'est la génération de modes qui marque le point de divergence. Plutôt que de chercher à générer un nouveau mode suivant la direction de recherche $\mathbf{H}_\ell(t)$ tangente, nous cherchons préférentiellement le premier terme de la $\mathbf{H}_c$-PGD associé au second membre $\check{\boldsymbol{\mathcal{Q}}}$ caractérisant le résidu (6.11) de l'étape de mise à jour précédente :

$$\check{\boldsymbol{\mathcal{Q}}}(t) = \boldsymbol{\mathcal{Q}}(t) - \mathbf{H}_\ell(t)\Big(\sum_{i=1}^{m_\ell} \beta_i(t)\boldsymbol{\Lambda}_i\Big) \qquad \forall t \in I \tag{6.11}$$

Autrement dit, nous nous donnons un conditionneur $\mathbf{H}_c$ constant et symétrique, puis le nouveau couple $(\boldsymbol{\Lambda}, \lambda)$ est recherché par résolution du problème de minimisation (6.12) ci-dessous :

$$(\boldsymbol{\Lambda}, \lambda) = \underset{\substack{\alpha\in\mathcal{I}_\mathbb{R}\\ \gamma\in\boldsymbol{\mathcal{V}}}}{\arg\min}\ \check{J}(\alpha\boldsymbol{\gamma}) \tag{6.12}$$

où la fonctionnelle modifiée $\check{J}$ est donnée par la formule (6.13) suivante :

$$\check{J}(\mathbf{w}) = \frac{1}{2}\int_I {}^t\mathbf{w}(t)\Big[\mathbf{H}_c\mathbf{w}(t) - 2\check{\boldsymbol{\mathcal{Q}}}(t)\Big]\,\mathrm{d}t \tag{6.13}$$

Ceci conduit à la résolution d'un problème de type point fixe. Une étape d'orthogonalisation des modes par rapport au conditionneur $\mathbf{H}_c$ est ensuite réalisée comme d'habitude, permettant de constituer la nouvelle base $\{(\boldsymbol{\Lambda}_i, \lambda_i^{\ell+1})\}_{1\leqslant i\leqslant m_\ell+1} \in \{\boldsymbol{\mathcal{V}} \times \mathcal{I}_\mathbb{R}\}^{m_\ell+1}$ orthonormée. Pour constituer le conditionneur $\mathbf{H}_c$, une large variété de choix est envisageable dans la mesure où les deux seules propriétés souhaitées – avoir un conditionneur constant et symétrique – ne sont pas très contraignantes. Sans exhaustivité, on peut par exemple opter pour la famille de conditionneurs (6.14) aisément accessible, élaborée à partir d'une combinaison linéaire des opérateurs tangents symétriques $\mathbf{H}_0(t)$:

$$\mathbf{H}_c = \sum_{j=1}^{n_t} \alpha_j \mathbf{H}_0(t_j) \tag{6.14}$$

Lorsque le n_t-uplet $(\alpha_j)_{1\leqslant j\leqslant n_t}$ prend la valeur $(1, 0, ..., 0)$ nous retrouvons le cas particulier de la rigidité élastique $\mathbf{H}_c = \mathbf{K}$. En outre, étant donné que l'équation de la direction de recherche n'est vérifiée qu'en moyenne et, qui plus est, suivant un conditionneur $\mathbf{H}_c$ autre que la direction tangente $\mathbf{H}_\ell(t)$ souhaitée, une nouvelle étape de mise à jour des fonctions temporelles *post*-génération est entreprise. Cette étape supplémentaire est en tout point similaire à l'étape préliminaire classique de mise à jour, au détail près qu'elle intervient, non plus en préambule de la génération, mais à l'issue de la génération d'un nouveau mode. Nous cherchons ainsi une nouvelle actualisation des fonctions temporelles – les modes spatiaux étant fixés – par résolution d'un problème de minimisation (6.15) connexe :

$$\{\check{\beta}_i\}_{i\in[\![1,m_\ell+1]\!]} = \underset{\substack{\alpha_i\,\in\,\mathcal{I}_\mathbb{R}\\ i\in[\![1,m_\ell+1]\!]}}{\arg\min}\ \check{\check{J}}\Big(\sum_{i=1}^{m_\ell+1} \alpha_i(t)\boldsymbol{\Lambda}_i\Big) \tag{6.15}$$

où en notant $\check{\check{\boldsymbol{\mathcal{Q}}}}$ le second membre actualisé (6.16) selon :

$$\check{\check{\boldsymbol{\mathcal{Q}}}}(t) = \boldsymbol{\mathcal{Q}}(t) - \mathbf{H}_\ell(t)\Big(\sum_{i=1}^{m_\ell}\Big(\lambda_i^{\ell+1}(t) - \lambda_i^\ell(t)\Big)\boldsymbol{\Lambda}_i + \lambda_{m_\ell+1}^{\ell+1}(t)\boldsymbol{\Lambda}_{m_\ell+1}\Big) \qquad \forall t \in I \tag{6.16}$$

la nouvelle fonctionnelle (6.17) s'écrit :

$$\check{\check{J}}(\mathbf{w}) = \frac{1}{2}\int_I {}^t\mathbf{w}(t)\Big[\mathbf{H}_\ell(t)\mathbf{w}(t) - 2\check{\check{\boldsymbol{\mathcal{Q}}}}(t)\Big]\,\mathrm{d}t \tag{6.17}$$

Finalement, la correction $\Delta\mathbf{u}_{\ell+1}$ recherchée (6.18) solution de l'étape linéaire (5.8) prend la forme :

$$\Delta\mathbf{u}_{\ell+1}(t) = \sum_{i=1}^{m_\ell}\Big(\lambda_i^{\ell+1}(t) - \lambda_i^\ell(t) + \check{\beta}_i(t)\Big)\boldsymbol{\Lambda}_i + \Big(\lambda_{m_\ell+1}^{\ell+1}(t) + \check{\beta}_{m_\ell+1}(t)\Big)\boldsymbol{\Lambda}_{m_\ell+1} \qquad \forall t\in I \tag{6.18}$$

Grâce au conditionneur $\mathbf{H}_c$, la génération d'un mode n'est pas trop chronophage ce qui permet d'envisager la création de plusieurs modes à chaque itération LATIN-PGD afin d'accroître le taux de convergence. Nous nous donnons alors un critère $z_{\mathbf{H}_c}$ supplémentaire – donné dans la Section-6.2.4 suivante – pour déterminer si l'on passe à l'itération suivante ou si, au contraire, il est plus intéressant de construire un nouveau mode en reprenant les étapes (6.12) puis (6.15) – génération et *post*-génération respectivement. L'algorithme 4 récapitule l'ensemble de la démarche.

Algorithme 4 : Méthode LATIN-PGD faiblement intrusive avec conditionneur $\mathbf{H}_c$ constant

Initialisation : $\boldsymbol{s}_0 \in \mathbf{A}_d$
/* Boucle $\bullet_\ell$ sur les itérations LATIN-PGD */
while $\omega_\ell \geqslant \eta$ **do**
  /* Étape locale $\boldsymbol{s}_\ell \in \mathbf{A}_d \longrightarrow \widehat{\boldsymbol{s}}_\ell \in \boldsymbol{\Gamma}$ */
  for $j = 1\,..\,n_t$ **do**
    Génération des efforts internes $\widehat{\mathbf{f}}_{\mathrm{int},\ell}(t_j,\mathbf{u}(t_j)) = \boldsymbol{\mathcal{A}}_u(t_j,\mathbf{u}(\tau\leqslant t_j))$ et externes $\mathbf{f}_{\mathrm{ext}}(t_j)$
    Évaluation de la raideur tangente $\mathbf{H}_\ell(t_j)$
  /* Étape globale $\widehat{\boldsymbol{s}}_\ell \in \boldsymbol{\Gamma} \longrightarrow \boldsymbol{s}_{\ell+1} \in \mathbf{A}_d$ */
  Étape de mise à jour (6.9)
  while $(z_{\mathbf{H}_c} = 1)$ .AND. $(m_{\ell+1}\leqslant m_{\mathrm{M}})$ .AND. $(m_{\ell+1} - m_\ell \leqslant m_{gen})$ .AND. $(\boldsymbol{\Lambda}_i\mathbf{H}_c\boldsymbol{\Lambda}_j \leqslant 10^{-10},\ \forall(i,j)\in[\![1,m_{\ell+1}]\!]^2,\ i\neq j)$ **do**
    Étape de génération de modes avec conditionneur $\mathbf{H}_c$ constant (6.12)
    Étape de mise à jour *post*-génération (6.15)
    Critère d'augmentation de modes supplémentaires (6.20)
  /* Relaxation */
  $\boldsymbol{s}_{\ell+1} \leftarrow \mu_r\boldsymbol{s}_{\ell+1} + (1-\mu_r)\boldsymbol{s}_\ell$
  /* Compression de la base réduite */
  if $(m_{\ell+1} > m_{\mathrm{M}})$ .OR. $(\exists(i,j)\in[\![1,m_{\ell+1}]\!]^2,\ i\neq j \mid \boldsymbol{\Lambda}_i\mathbf{H}_c\boldsymbol{\Lambda}_j > 10^{-10})$ **then**
    Compression par technique SVD
  /* Indicateur d'erreur */
  On évalue l'erreur ω_ℓ

REMARQUE 6.4 : Il convient de noter que cette nouvelle version, développée plus spécifiquement pour la gestion des non-linéarités géométriques en grandes transformations, n'est pas pour autant dévolue exclusivement à ce type de non-linéarités. Il est de fait tout à fait envisageable de mettre en œuvre cette approche en présence de non-linéarités matériaux uniquement.

6.2.4 Critères additionnels

Cette nouvelle version avec conditionneur nécessite la prise en compte de critères supplémentaires pour connaître le nombre de modes à générer à chaque itération, mais également pour limiter l'expansion de la base réduite en venant effectuer des phases de compression par technique SVD dans le but d'obtenir une décomposition plus optimale.

Choix d'augmentation du nombre de modes. L'un des critères possibles consiste à évaluer à chaque étape k le résidu d'équilibre $\boldsymbol{\mathcal{Y}}^{(k)}$ (6.19) suivant une certaine norme :

$$\boldsymbol{\mathcal{Y}}^{(k)}(t) = \boldsymbol{\mathcal{Q}}(t) - \mathbf{H}_\ell(t)\Delta\mathbf{u}_{\ell+1}^{(k)}(t) \qquad \forall t \in I \tag{6.19}$$

avec $\Delta\mathbf{u}_{\ell+1}^{(k)}$ la correction apportée à l'issue de chaque nouvelle étape *post*-génération. Nous décidons d'ajouter systématiquement un mode à chaque itération LATIN-PGD, puis l'ajout de modes supplémentaires est assujetti au respect de la condition $(z_{\mathbf{H}_c} = 1)$ donnée en (6.20) dans la limite d'un nombre maximum de modes générés m_{gen} atteint par itération :

$$z_{\mathbf{H}_c}\Big(\boldsymbol{\mathcal{Y}}^{(k)}(t),\, \boldsymbol{\mathcal{Y}}^{(k+1)}(t)\Big) = \begin{cases} 1 & \text{si } \dfrac{z_1^{(k)} - z_1^{(k+1)}}{z_1^{(k)}} \geqslant \eta_{\mathbf{H}_c}^{(1)} \text{ ou } \dfrac{z_2^{(k)} - z_2^{(k+1)}}{z_2^{(k)}} \geqslant \eta_{\mathbf{H}_c}^{(2)} \\ 0 & \text{sinon} \end{cases} \tag{6.20}$$

$$\text{avec} \quad z_1^{(k)} = \max_{t\,\in\, I}\Big(\max_{\boldsymbol{\mathcal{V}}}\Big|\boldsymbol{\mathcal{Y}}^{(k)}(t)\Big|\Big) \;\text{ et }\; z_2^{(k)} = \frac{1}{n_t}\sum_{j=1}^{n_t}\max_{\boldsymbol{\mathcal{V}}}\Big|\boldsymbol{\mathcal{Y}}^{(k)}(t_j)\Big|$$

où $\eta_{\mathbf{H}_c}^{(i)}$ représente une certaine tolérance – éventuellement variable – dont la valeur est fixée par l'utilisateur. Ce critère reflète la quantité d'information apportée par chaque nouveau mode comparé à l'apport du mode précédent. Autrement dit, nous continuons d'enrichir la base réduite aussi longtemps que la génération d'un nouveau mode apporte une information substantielle et pertinente pour faire décroître soit l'erreur maximale, soit l'erreur moyenne – dans la limite du nombre maximal m_{gen} de modes générés à chaque itération. Le choix d'augmentation du nombre de modes est ainsi achevé avec parcimonie dans la mesure où, les efforts internes et externes évoluant au fil des itérations, il n'est pas souhaitable de construire un grand nombre de modes à chaque itération.

Compression de la base par SVD. La génération de plusieurs modes à chaque itération LATIN-PGD peut rapidement conduire à une base de dimension conséquente dont on sait pertinemment qu'une grande partie de l'information y est redondante. Pour pallier ce constat, une compression de la base de modes obtenus par PGD est opérée par technique SVD à intervalles réguliers : soit lorsque le nombre maximum de modes admissibles m_{M} est atteint, soit lorsqu'une perte d'orthogonalité est constatée – signifiant que l'on n'est plus capable en l'état de produire un nouveau mode tel qu'une quantité suffisante d'informations puisse encore être décelée dans une nouvelle direction orthogonale non encore explorée.

6.2.5 Différentes méthodologies envisageables

La méthodologie mise en place jusqu'ici possède l'avantage d'être peu intrusive dans le sens où peu de modifications ont été nécessaires par rapport à la version présentée au Chapitre 5, et que les quelques modifications apportées se basent sur des opérateurs classiques du logiciel Simcenter Samcef™, aisément accessibles. C'est en ce sens que nous avons privilégié cette approche. Toutefois, partant du constat que la séparation des variables espace/temps n'est généralement pas optimale en grandes transformations, d'autres méthodologies sont envisageables [Ladevèze, 2017] – bien que plus compliquées à mener. En effet, la séparation des variables espace/temps ne fonctionne bien que dans un repère lié à la matière. Ainsi, plutôt que de chercher une décomposition à variables séparées du champ de déplacements généralisés $\mathbf{u}$, on peut proposer une autre représentation associée à la dérivée de Jaumann. On cherche alors le champ des vitesses $\dot{\mathbf{u}}$ sous la forme (6.21) suivante :

$$\dot{\mathbf{u}}(t) = \mathbf{R}(t)\mathbf{W}(t) \qquad \forall t \in I \tag{6.21}$$

avec $\mathbf{W}$ le champ des vitesses tournées et $\mathbf{R} \in \mathcal{O}_3(\mathbb{R})$ la rotation associée à la dérivée de Jaumann – fonction des variables d'espace et de temps – que l'on exprime par la formule remarquable de Rodrigues (6.22)

selon :

$$\mathbf{R} = \mathbb{I} + \frac{\sin\theta}{\theta}\mathbf{i}(\mathbf{\Omega}) + \frac{1-\cos\theta}{\theta^2}\mathbf{i}(\mathbf{\Omega})\mathbf{i}(\mathbf{\Omega}) \quad \text{avec} \quad \theta^2 = {}^t\mathbf{\Omega}\mathbf{\Omega} \tag{6.22}$$

où $\mathbf{i}$ symbolise l'opérateur du produit vectoriel. Chacune des quantités $\mathbf{W}$ et $\mathbf{\Omega}$ est ainsi recherchée sous la forme (6.23) à variables séparées :

$$\mathbf{W}(t) = \sum_{i=1}^{m_{\mathbf{W}}} \lambda_i^{\mathbf{W}}(t)\mathbf{\Lambda}_i^{\mathbf{W}} \qquad \mathbf{\Omega}(t) = \sum_{i=1}^{m_{\mathbf{\Omega}}} \lambda_i^{\mathbf{\Omega}}(t)\mathbf{\Lambda}_i^{\mathbf{\Omega}} \tag{6.23}$$

Par l'intermédiaire de la rotation $\mathbf{R}$, un nouveau référentiel est ainsi défini : le référentiel d'espace co-rotationnel dans lequel la matière au voisinage d'un point matériel se déforme sans tourner. Ce choix de représentation n'implique aucune modification lors de l'étape locale et seule l'étape globale nécessite d'être adaptée. Lors de cette dernière, on cherche alors à chaque itération ℓ une nouvelle approximation à variables séparées du champ des vitesses tournées $\mathbf{W}_{\ell+1}$ telle que la rotation soit associée à l'approximation résultant de l'itération précédente $\mathbf{R}_\ell$ – qui dépend de toute l'histoire de la déformation. La rotation $\mathbf{R}_\ell$ découle plus précisément de l'intégration (6.24) donnée ci-dessous :

$$\begin{cases} \dot{\mathbf{R}}_\ell {}^t\mathbf{R}_\ell = \left[\dot{\mathbb{F}}_\ell \mathbb{F}_\ell^{-1}\right]_{\text{AS}} \\ \mathbf{R}_\ell|_{t=0} = \mathbb{I} \end{cases} \tag{6.24}$$

ce qui nécessite d'avoir accès aux opérateurs éléments finis qui évaluent les gradients de la transformation, et plus spécifiquement la partie anti-symétrique. Ces opérateurs se situent généralement au niveau des routines sur les éléments et se retrouvent par là même peu accessibles dans une vision faiblement intrusive. À cela s'ajoute la résolution non classique de l'équation aux dérivées partielles (6.24), ce qui conduit à l'utilisation d'outils non standards des codes industriels. Cette variante n'a pas été privilégiée au regard de son intrusivité, bien qu'elle serait vraisemblablement plus optimale en termes de représentation des différents champs solutions.

6.3 Illustration sur un cas-test académique

Faisant suite au premier cas-test présenté au Chapitre-5, un second cas-test académique est exposé dans ce chapitre mettant en lumière les nouveautés liées à la prise en compte des non-linéarités géométriques.

6.3.1 Description du cas-test

Le second cas-test académique représente une structure en forme d'équerre sollicitée en flexion à l'une de ses extrémités tandis que l'autre extrémité est encastrée (cf. Figure 6.2). L'amplitude du chargement $\mathbf{u}_d$ est choisie telle que l'on sorte du cadre HPP – de manière à activer les non-linéarités géométriques au sein de la structure. Typiquement, nous considérons un chargement de 40 mm à titre d'exemple. Aux non-linéarités géométriques, il est également possible de juxtaposer des non-linéarités matériaux en considérant une loi élasto-plastique à écrouissage linéaire isotrope de fonction seuil $R(p) = \sigma_0 + hp$ avec σ_0 et h deux paramètres matériaux – caractérisant respectivement la limite d'élasticité et le module d'écrouissage. Le Tableau 6.2 recense les différentes valeurs retenues pour ces paramètres. Plus précisément, cette loi matériau est définie par les équations (6.25) suivantes :

$$\textit{Équations d'état :} \begin{cases} \boldsymbol{\sigma}_{\mathbf{R}_p} = \mathcal{K} : \boldsymbol{\gamma}_\varepsilon^e \\ R(p) = \sigma_0 + hp \\ \boldsymbol{\gamma}_\varepsilon^e = \boldsymbol{\gamma}_\varepsilon - \boldsymbol{\gamma}_\varepsilon^p \end{cases} \qquad \textit{Équations d'évolution :} \begin{cases} \dot{\boldsymbol{\gamma}}_\varepsilon^p = \dfrac{3}{2}\dot{p}\dfrac{\boldsymbol{s}_{\boldsymbol{\sigma}_{\mathbf{R}_p}}}{J_2(\boldsymbol{s}_{\boldsymbol{\sigma}_{\mathbf{R}_p}})} \\ \dot{p} \geqslant 0 \\ \dot{p} f = 0 \\ f \leqslant 0 \end{cases} \tag{6.25}$$

où l'équation de la surface seuil est donnée par $f(\boldsymbol{\sigma}_{\mathbf{R}_p}, p) = J_2(\boldsymbol{\sigma}_{\mathbf{R}_p}) - R(p)$. Ainsi, au fur et à mesure du chargement, l'équerre se plie pour faire intervenir des grandes rotations et, le cas échéant, une zone plastique localisée au niveau du coude géométrique.

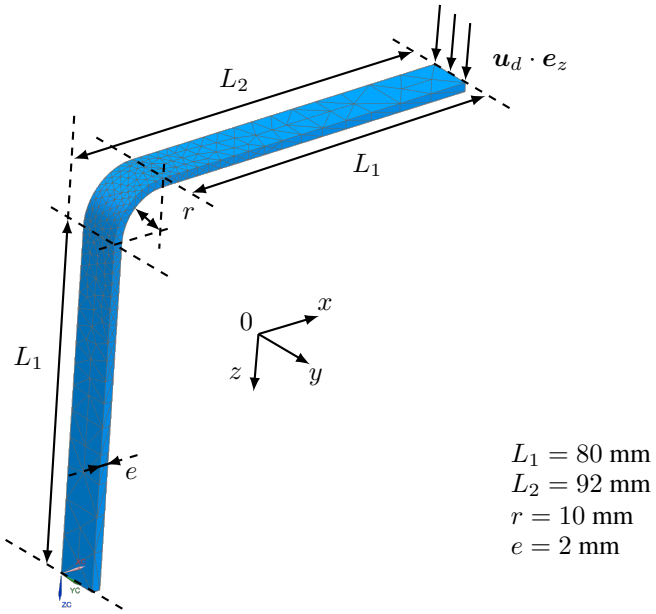


FIGURE 6.2 – Deuxième cas académique : structure en forme d'équerre

TABLE 6.2 – Paramètres matériaux de la loi élasto-plastique à écrouissage linéaire isotrope

E [GPa]	ν [-]	σ_0 [MPa]	h [MPa]
68	0.29	188.41	557.66

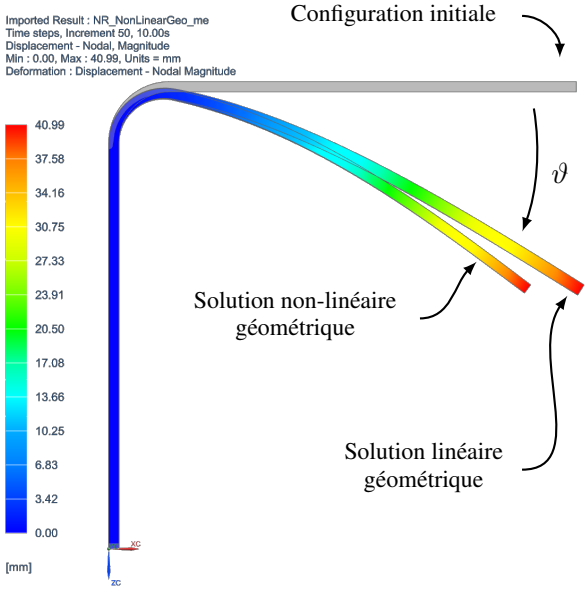
6.3.2 Résultats numériques

Dans la suite, on fournit quelques résultats issus de différentes simulations numériques. En particulier, on s'attache à mettre en exergue deux points spécifiques : d'une part, l'importance de circonscrire correctement la raideur géométrique dans la direction de recherche et, d'autre part, la pertinence de la version avec conditionneur $\mathbf{H}_c$ comparée à la version standard ne faisant intervenir que l'opérateur tangent $\mathbf{H}_\ell(t)$.

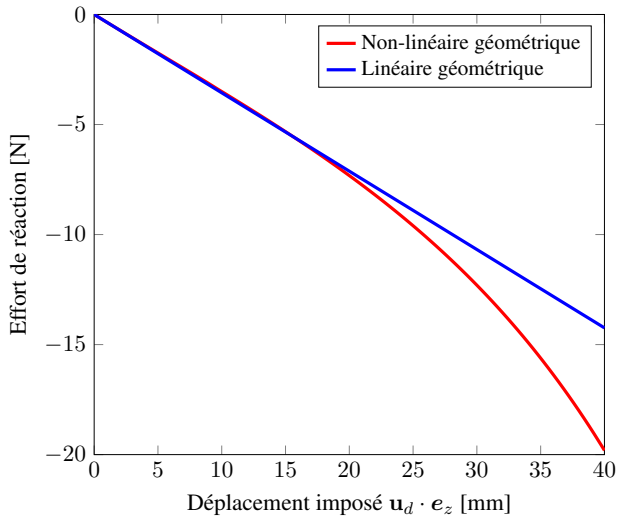
Impact de la raideur géométrique. Pour comprendre et illustrer les différents phénomènes sous-jacents, liés à la caractérisation de la raideur géométrique, nous traçons spécifiquement dans ce paragraphe la courbe donnant les efforts de réaction en fonction du déplacement suivant l'axe z (axe de sollicitation) pour un nœud de l'extrémité de la structure – là où le déplacement est imposé. La Figure 6.3 fournit l'allure du comportement de la structure avec, et sans, prise en compte des non-linéarités géométriques : la sous-figure 6.3a montre la déformée de la structure soumise à un chargement de 40 mm, tandis que l'on retrouve sur la sous-figure 6.3b le déplacement imposé $\mathbf{u}_d \cdot \boldsymbol{e}_z$ en abscisse et l'effort de réaction correspondant en ordonnée. On observe une perte de linéarité dès que le chargement devient suffisamment important – au-delà d'une quinzaine de millimètres, soit pour un angle d'inclinaison supérieur à dix degrés environ. Ceci s'apparente au moment où le développement limité au premier ordre des différentes fonctions n'est plus suffisant, nécessitant la prise en compte des termes d'ordre supérieur. La formule (6.26) fournit par exemple les premiers termes du développement en série de Taylor de la fonction tangente en 0 (de rayon de convergence $\frac{\pi}{2}$) :

$$\tan(\vartheta) = \sum_{n=1}^{3} |B_{2n}| \frac{4^n(4^n-1)}{(2n)!}\vartheta^{2n-1} + o(\vartheta^5) = \vartheta + \frac{1}{3}\vartheta^3 + \frac{2}{15}\vartheta^5 + o(\vartheta^5) \tag{6.26}$$

avec B_n le $n^{ième}$ nombre de Bernoulli. Pour des petites valeurs du déplacement imposé, on retrouve classiquement $\tan(\vartheta) = \left(L_1 + \frac{r}{\sqrt{2}}\right)^{-1} \mathbf{u}_d \cdot \boldsymbol{e}_z \approx \vartheta \approx \sin(\vartheta)$ au premier ordre si ϑ représente l'angle d'inclinaison entre la configuration initiale et la configuration déformée (cf. sous-figure 6.3a).



(a) Configurations initiale et déformée de la structure



(b) Effort de réaction en bout de structure en fonction du déplacement imposé

FIGURE 6.3 – Comparaisons avec, et sans, prise en compte des non-linéarités géométriques

REMARQUE 6.5 : On note en particulier que la prise en compte des non-linéarités géométriques conduit globalement à rigidifier la structure : plus le déplacement imposé augmente et plus l'effort de réaction est grand. La non-linéarité géométrique possède donc une nature différente des non-linéarités matériaux provenant de comportements plastiques (venant assouplir la structure) et s'apparente plutôt à des non-linéarités matériaux de type hyper-élastiques.

Sous l'hypothèse de linéarité géométrique, la déformation correspond à la partie symétrique du gradient. En grandes transformations, il convient de prendre également en compte la partie antisymétrique du gradient afin de décrire correctement la déformation effective de la structure. Cela se matérialise par une contribution supplémentaire – une rigidité géométrique – dans l'opérateur de la direction de recherche. Comme évoqué précédemment, cette contribution joue le rôle d'un terme de rappel sur la partie antisymétrique du gradient dont il est important de capter le bon ordre de grandeur pour assurer la convergence. Pour illustrer ces propos, nous réalisons une petite étude très simple : on considère le matériau comme étant élastique linéaire, on fournit la solution du problème non-linéaire géométrique en tant qu'initialisation de notre algorithme et on réalise les quelques premières itérations de la méthode LATIN (sans technique PGD). Connaissant parfaitement la solution du problème dès l'initialisation, nous avons donc également accès aux bonnes valeurs des rigidités – somme d'une rigidité élastique (matériau) et d'une rigidité géométrique. La Figure 6.4 donne l'évolution de l'effort de réaction en fonction du déplacement imposé à l'issue de la deuxième itération pour différents choix de la direction de recherche $\mathbf{H}$ – supposée constante :

- $\mathbf{H}_0(1)$: somme des raideurs élastique $\mathbf{K}$ et géométrique du premier pas de chargement ;

- $\mathbf{H}_0(\frac{n_t}{2})$: somme des raideurs élastique $\mathbf{K}$ et géométrique du pas de chargement milieu ;
- $\mathbf{H}_0(n_t)$: somme des raideurs élastique $\mathbf{K}$ et géométrique du dernier pas de chargement.

On observe alors que la stabilité du schéma du résolution est intimement liée à l'évaluation de la raideur géométrique : on reste stable – c'est-à-dire que l'on ne dévie pas de la solution – uniquement dans les zones où la raideur géométrique est correctement estimée. Considérer l'opérateur tangent complet $\mathbf{H}_0(t)$ semble donc préférable pour assurer la stabilité, bien que cela puisse engendrer un coût supplémentaire non négligeable lorsque le problème se complexifie. Nous présentons par la suite les résultats issus de la version avec conditionneur $\mathbf{H}_c$ constant, destinée à circonscrire la complexité algébrique dans la conduite des différentes opérations.

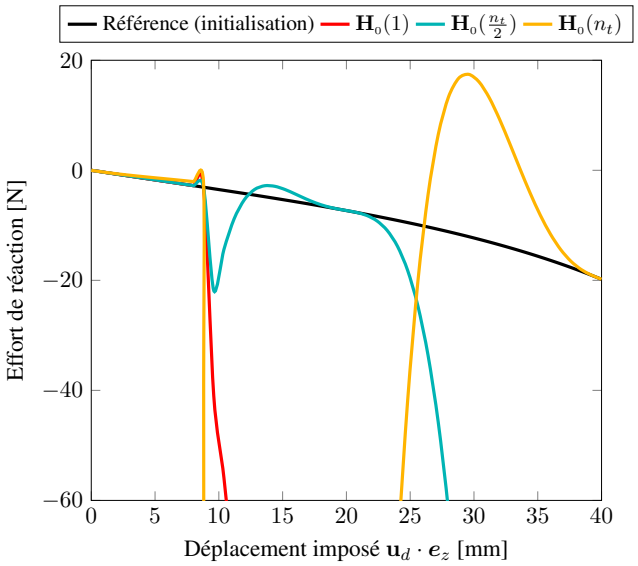


FIGURE 6.4 – Influence du choix de la direction de recherche sur la stabilité du schéma de résolution illustrée par l'évolution de l'effort de réaction en fonction du déplacement imposé à l'issue de la deuxième itération

REMARQUE 6.6 : Avec la méthode LATIN uniquement (sans technique PGD), tous les pas de temps sont traités indépendamment les uns à la suite des autres au niveau de l'étape linéaire.

Comparaison des deux versions $\mathbf{H}_\ell(t)$ *versus* $\mathbf{H}_c$. On étudie la version basée sur le conditionneur constant et symétrique $\mathbf{H}_c$. Plusieurs paramètres influencent grandement le taux de convergence de la méthode : le choix du conditionneur, son éventuelle actualisation au cours des itérations, ainsi que le nombre de modes générés à chaque itération. Nous choisissons comme conditionneur la combinaison $\mathbf{H}_c = \mathbf{H}_0(1) + \mathbf{H}_0(\frac{n_t}{2}) + \mathbf{H}_0(n_t)$ qui a semblé suffisamment robuste dans cette étude, quitte à redécouper l'intervalle de temps et mener une stratégie de résolution semi-incrémentale. L'idée de la résolution semi-incrémentale consiste à découper l'intervalle de temps I en plusieurs sous-intervalles de temps I_i (avec $I = \cup I_i$) et à faire converger successivement chacun de ces sous-intervalles avant de considérer la résolution de l'intervalle suivant. Ainsi, les modes générés s'accommodent mieux à chaque étape de résolution de ces sous-intervalles I_i ce qui permet d'atteindre une convergence relativement rapide par sous-intervalle. De plus, lors du calcul de l'intervalle suivant, on se base alors sur une meilleure appréciation de la rigidité géométrique, résultant de la résolution de l'intervalle précédent. La Figure 6.5 témoigne de l'influence respective de la fréquence d'actualisation du conditionneur $\mathbf{H}_c$ au cours des itérations et du nombre de modes générés – supposé identique à chaque itération. On observe en particulier qu'un optimum se dessine pour $m_{gen} = 3$ et une actualisation du conditionneur $\mathbf{H}_c$ toutes les dix itérations LATIN.

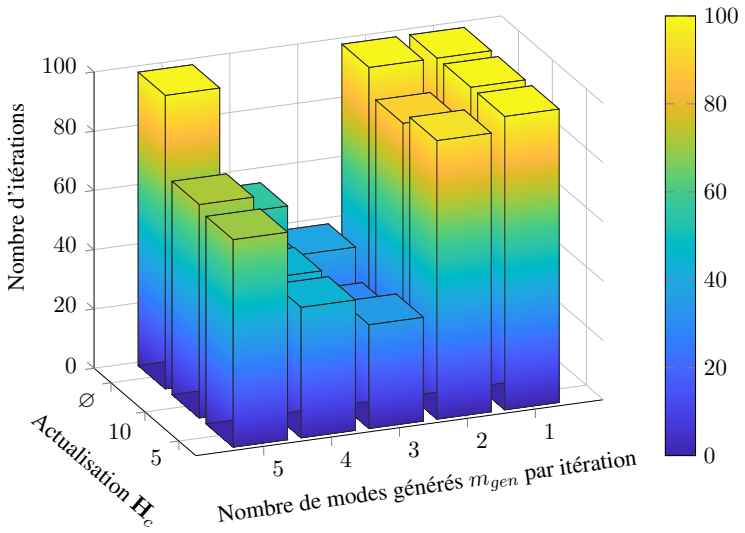


FIGURE 6.5 – Influences respectives sur le taux de convergence de la fréquence d'actualisation du conditionneur $\mathbf{H}_c$ au cours des itérations et du nombre de modes générés m_{gen} par itération

Produire toujours le même nombre de modes par itération est une possibilité mais il est également envisageable d'utiliser un critère basé sur l'erreur en direction de recherche (6.20) ce que nous employons à partir de maintenant – avec $\eta^{(i)}_{\mathbf{H}_c} = 5\%$, $i \in \{1, 2\}$ et $m_{gen} = 3$. La Figure 6.6 compare au cours des itérations l'évolution de l'erreur vraie e_χ entre les versions avec et sans conditionneur, ainsi qu'avec la méthode LATIN pure (sans technique PGD) suivant une direction de recherche tangente $\mathbf{H}_\ell(t)$. Sous un formalisme continu, e_χ est définie par la formule (6.27) suivante :

$$e_\chi = \max_{t \in I} \left[\frac{\boldsymbol{\delta u}(t)}{|\boldsymbol{u}_\chi(\boldsymbol{x}_0, t)|} \right]$$
$$\text{avec } \delta \boldsymbol{u}(t) = \max_{\boldsymbol{x} \in \Omega} |\boldsymbol{u}(\boldsymbol{x}, t) - \boldsymbol{u}_\chi(\boldsymbol{x}, t)| \text{ et } \boldsymbol{x}_0 = \arg\max_{\boldsymbol{x} \in \Omega} \Big(|\boldsymbol{u}(\boldsymbol{x}, t) - \boldsymbol{u}_\chi(\boldsymbol{x}, t)| \Big) \tag{6.27}$$

où $\boldsymbol{u}_\chi$ symbolise la solution du problème, donnée par le logiciel Simcenter Samcef™ dans sa version commerciale standard.

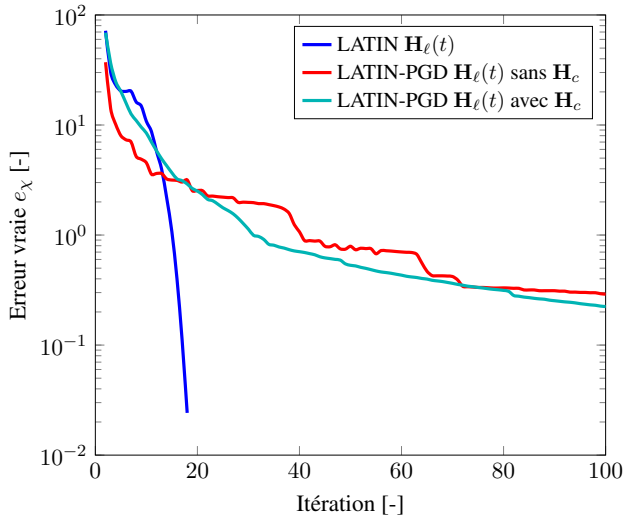


FIGURE 6.6 – Évolution de l'erreur vraie e_χ en fonction des itérations pour différentes versions

On observe en particulier un meilleur taux de convergence sur les premières itérations de l'algorithme, par rapport à celui de la version LATIN pure (sans technique PGD). En effet, l'introduction de la PGD conduit à la vérification en moyenne de l'équation de la direction de recherche impliquant une propagation plus rapide de l'information au début : tous les pas de temps ne sont alors plus traités indépendamment les uns des autres lors de l'étape globale linéaire. En revanche, une fois passées les premières itérations, le taux de convergence s'essouffle quelque peu et il faut plus d'itérations pour atteindre la convergence fine. Chaque nouveau mode construit peut être assimilé à un mode global – issu d'une moyenne temporelle – alors que la rigidité réelle de la structure au dernier pas de chargement dépend fortement des états convergés précédents et donc des rigidités effectives précédentes. Or, l'approximation aux premiers pas de temps n'étant encore que trop partiellement évaluée, l'estimation de la rigidité globale s'en retrouve dégradée. Qui plus est, on remarque un taux de convergence similaire entre les versions avec et sans emploi du conditionneur $\mathbf{H}_c$ ce qui légitime l'utilisation d'un conditionneur pour réduire la complexité algébrique, sans toutefois dégrader sensiblement le taux de convergence. Le Tableau 6.3 fournit une analyse succincte du nombre d'opérations macroscopiques à réaliser en fonction de la version retenue.

TABLE 6.3 – Complexité algébrique des différentes versions étudiées pour atteindre une erreur vraie e_χ inférieure à 1%

	Factorisation	**Résolution spatiale**	**Mise à jour**	**Étale locale**
LATIN $\mathbf{H}_\ell(t)$	750	750	0	750
LATIN-PGD $\mathbf{H}_\ell(t)$ sans $\mathbf{H}_c$	120	120	40	2000
LATIN-PGD $\mathbf{H}_\ell(t)$ avec $\mathbf{H}_c$	4	231	107	1500

Aux non-linéarités géométriques, il est également possible de rajouter les non-linéarités matériaux (plasticité). En reprenant la version avec conditionneur, on est alors capable de déceler une approximation de bonne qualité – erreur e_χ inférieure à 1% – en 247 itérations. La Figure 6.7 complète la sous-figure 6.3b précédente en rajoutant l'évolution correspondant à la prise en compte simultanée des non-linéarités géométriques et de comportements matériaux.

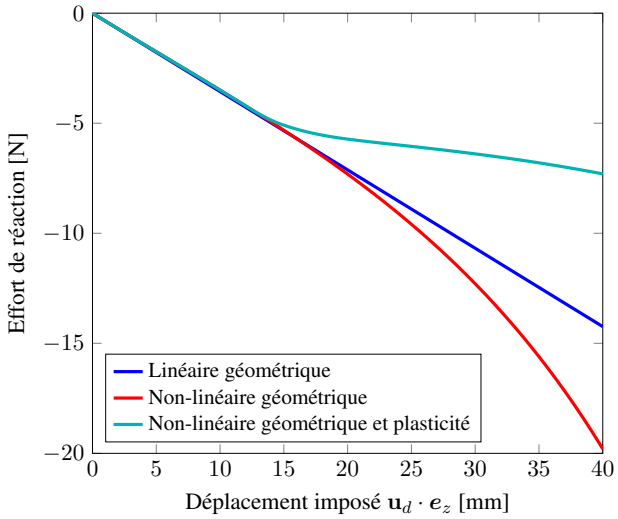


FIGURE 6.7 – Impact de la nature des non-linéarités sur l'évolution des courbes d'effort de réaction en fonction du déplacement imposé

Sans surprise, la considération simultanée de plusieurs types de non-linéarités engendre un problème de plus grande complexité, occasionnant un nombre d'itérations plus élevé pour atteindre la convergence.

Parmi les pistes d'amélioration, dans la lignée de la stratégie semi-incrémentale, une idée pourrait consister à réaliser une PGD pondérée : on donnerait plus de poids à certains instants lors de la construction des modes, de manière à privilégier les premiers pas de temps au début de l'algorithme pour accélérer la convergence.

Chapitre 7

Gestion des non-linéarités de contact avec la méthode LATIN-PGD faiblement intrusive

Ce chapitre étend la méthodologie proposée à la gestion des non-linéarités de type contact. Basées sur une formulation du Lagrangien augmentée, des directions de recherche supplémentaires sont introduites, reliant les inconnues liées au contact. On continue de s'appuyer au maximum sur les outils préexistants en limitant le niveau d'intrusivité.

Sommaire

On présente dans ce chapitre les nouveautés liées à la prise en compte des conditions de contact. La méthodologie proposée s'appuie sur les développements précédents de sorte que l'on reste compatible avec les autres types de non-linéarités. On s'intéresse en particulier au cas du contact entre deux solides déformables où l'on fait l'hypothèse, pour commencer, que les déplacements dans les zones de contact restent petits.

7.1 Équations du problème en présence de contact

Quel que soit le type de contact, celui-ci peut être adressé de différentes manières au sein du logiciel Simcenter Samcef™ : (i) soit au moyen d'un algorithme découplé annexe à l'algorithme classique de Newton-Raphson, (ii) soit directement de manière couplée par l'intermédiaire d'un algorithme de Newton-Raphson généralisé :

- (i) cette première manière de procéder repose sur la séparation des degrés de liberté en deux catégories : ceux associés aux conditions de contact et les autres. On résout alors, à chaque itération de

chaque pas de temps, le système d'équations issu de la condensation statique des inconnues libres de toute condition de contact. Les non-linéarités de type contact sont alors traitées séparément des autres formes de non-linéarités – matérielles ou géométriques ;

- (ii) cette seconde manière de procéder s'appuie sur l'introduction d'inconnues supplémentaires, caractérisant les contraintes cinématiques liées au contact, par l'intermédiaire de multiplicateurs de Lagrange. L'ensemble des inconnues du problème sont alors résolues d'un seul tenant, directement au niveau de l'algorithme de Newton-Raphson grâce au jacobien généralisé, et les non-linéarités de contact sont traitées comme toute autre forme de non-linéarités.

Motivés d'un côté par l'intérêt d'avoir tous les types de non-linéarités sous un même formalisme – analogue à une loi de comportement –, et d'un autre côté par le fait que (ii) est la méthode standard utilisée dans Simcenter Samcef™ en présence de non-linéarités géométriques importantes, nous avons décidé de privilégier cette seconde option pour le développement de la méthodologie LATIN-PGD faiblement intrusive. Aussi, les contraintes supplémentaires, qui apparaissent au niveau des différentes surfaces de contact, induisent une nouvelle paire de variables $(\boldsymbol{\xi}, \mathbf{v}) \in \mathcal{I}_{\boldsymbol{\mathcal{V}}_c} \times \mathcal{I}_{\boldsymbol{\mathcal{V}}_c}$ reliant les nœuds des surfaces en contact. Plus précisément, $\mathbf{v} \in \mathcal{I}_{\boldsymbol{\mathcal{V}}_c}$ correspond à la trace des déplacements généralisés sur la zone de contact et $\boldsymbol{\xi} \in \mathcal{I}_{\boldsymbol{\mathcal{V}}_c}$ désigne la quantité duale, à savoir les multiplicateurs de Lagrange supplémentaires représentant les efforts de contact. On rappelle alors les équations du problème de référence en (7.1 – 7.2) que nous séparons classiquement en deux groupes suivant le principe P1 de la méthode LATIN-PGD – séparation des difficultés :

Trouver $\boldsymbol{s} = (\mathbf{u}, \boldsymbol{\xi}, \mathbf{f}_{\text{int}}, \mathbf{v}) \in [\mathcal{I}_{\boldsymbol{\mathcal{V}}} \times \mathcal{I}_{\boldsymbol{\mathcal{V}}_c}]^2$ tel que :

$$(\boldsymbol{\Gamma}) : \quad \forall t \in I, \quad \begin{cases} \mathbf{f}_{\text{int}}(t, \mathbf{u}(t)) = \boldsymbol{\mathcal{A}}_u\Big(t, \mathbf{u}(\tau \leqslant t)\Big) \\ \boldsymbol{\mathcal{A}}_c\Big(t, \mathbf{v}(t), \boldsymbol{\xi}(t)\Big) = \mathbf{0} \end{cases} \tag{7.1}$$

$$(\mathbf{A}_d) : \quad \begin{cases} \mathbf{u}(t=0) = \mathbf{0} \\ \forall t \in I, \quad \begin{cases} \boldsymbol{\mathcal{C}}_{\mathbf{u}}\Big(\mathbf{u}(t)\Big) = \mathbf{u}_d(t) \\ \boldsymbol{\mathcal{C}}_{\mathbf{v}}\Big(\mathbf{u}(t)\Big) = \mathbf{v}(t) \\ \mathbf{f}_{\text{int}}(t, \mathbf{u}(t)) - \mathbf{f}_{\text{ext}}(t) + \mathbf{f}_{\text{ctc}}(t, \boldsymbol{\xi}(t)) = \mathbf{0} \\ \mathbf{f}_{\text{ctc}}(t, \boldsymbol{\xi}(t)) = \boldsymbol{\mathcal{C}}_{\boldsymbol{\xi}}\Big(\boldsymbol{\xi}(t)\Big) \end{cases} \end{cases} \tag{7.2}$$

On détaille par la suite le choix des directions de recherche, ainsi que les spécificités liées à l'application de la méthode LATIN-PGD faiblement intrusive dans le cadre des non-linéarités de type contact.

7.2 Spécificités de la méthode LATIN-PGD pour la gestion du contact

Historiquement, la variante de la méthode LATIN la plus couramment employée pour les problèmes de contact est celle dédiée à la décomposition de domaines (cf. Annexe A). L'idée très générale consiste à définir des interfaces entre les différents sous-domaines auxquelles on peut associer, par la suite, un comportement non-linéaire de type contact – comportement d'interface. Ces interfaces renferment leur propre jeu d'inconnues : déplacements et efforts d'interface. L'application des directions de recherche conduit alors généralement à introduire des raideurs d'interface dont les opérateurs s'apparentent classiquement à une matrice de masse éléments finis. Comme discuté dans [Oumaziz, 2017], la construction et l'assemblage de ces opérateurs ne sont pas toujours aisées dans les codes industriels, ce qui a poussé les auteurs à proposer une version non-intrusive de la méthode LATIN pour la décomposition de domaine par l'ajout d'une couche d'éléments, appelée *semelle*, aux interfaces. De cette manière, plutôt qu'une impédance d'interface complexe à déterminer, les conditions de Robin sont assimilées à un comportement de type ressort dont la rigidité découle d'un assemblage classique des codes éléments finis industriels.

Cette méthodologie a notamment été employée avec le logiciel libre *code_aster* développé par EDF. Dans notre cas, le logiciel industriel Simcenter Samcef™ dispose d'éléments spécifiques, appelés éléments de contact, sur lesquels nous allons tout naturellement nous appuyer pour développer notre méthodologie – Section-7.2.2.

REMARQUE 7.1 : Dans la littérature, la méthode PGD n'est que très rarement déployée, conjointement à la méthode LATIN, lors du traitement des phénomènes de contact – voir [Giacoma *et al.*, 2015] pour un exemple. Les efforts sont généralement plutôt portés sur les aspects de décomposition de domaines, alliant solveurs de Krylov performants et stratégies multi-échelles [Passieux *et al.*, 2010, Oumaziz *et al.*, 2021].

7.2.1 Initialisation

Aucune nouveauté n'intervient à ce stade, l'initialisation de l'algorithme étant toujours réalisée à partir de n'importe quelle approximation $\boldsymbol{s}_0 \in \mathbf{A}_d$. Un point particulier mérite toutefois que l'on y porte attention : des difficultés peuvent apparaître lorsque les conditions de contact figurent être les seules conditions aux limites cinématiques de certains domaines. Il convient alors d'éviter de générer des mouvements de corps rigides. Ce sera notamment le cas de la seconde illustration industrielle présentée au Chapitre-9 où l'aube n'est retenue dans son mouvement que par la condition de contact existante avec le disque. Ainsi, en fonction de la configuration, nous pouvons par exemple choisir pour l'initialisation :

- de résoudre le problème de contact uniquement, en désactivant les autres non-linéarités ;
- de définir un problème de substitution équivalent (possédant le même nombre de degrés de liberté) en suppléant les conditions de contact non-linéaires par d'autres conditions cinématiques plus simples – moyenne, collage, etc. – et de résoudre alors le problème élastique linéaire qui en résulte.

7.2.2 Étape locale

Entièrement basée sur les éléments de contact, la conduite de l'étape locale repose sur le logiciel existant. On présente tout d'abord succinctement ces types d'éléments, utilisés classiquement dans le logiciel Simcenter Samcef™. Au sein de notre approche fonctionnelle de la méthode LATIN-PGD faiblement intrusive, ce sont ces éléments de contact qui vont jouer le rôle d'interfaces.

REMARQUE 7.2 : Dans le logiciel Simcenter Samcef™, on s'appuie sur le traitement des équations de contact qui s'effectue à partir du champ des déplacements (contact de Signorini) – et non pas du champ des vitesses.

7.2.2.1 Éléments de contact

En fonction de la nature de la zone de contact, différents éléments sont mobilisés : on retrouve les contacts *nœuds / faces* et *faces / faces* classiques où à chaque fois soit le groupe de faces regroupe des surfaces 3D (triangles, quadrangles, etc.), soit il se réduit à des lignes en 2D. Dans la suite, on se concentre sur un élément de contact *nœud / face* à base triangle pour illustrer les propos. Du point de vue topologique, cet élément est crée entre chaque nœud potentiellement en contact avec sa face en regard. Par définition, chacun de ces éléments comporte donc 15 inconnues en 3D – 3 inconnues de déplacement pour le nœud, 9 inconnues de déplacement pour les 3 nœuds formant la surface de contact et 3 inconnues supplémentaires pour les multiplicateurs de Lagrange associés aux contraintes cinématiques. Un tel élément peut donc être vu comme un élément volumique pyramidale 3D (cf. Figure 7.1) dont le comportement interne régit les lois cinématiques reliant les différentes inconnues de déplacement impliquées. Plutôt que de travailler

avec les déplacements d'interface $\{\mathbf{v}^{(i)}\}_{1\leqslant i\leqslant 4}$ de chacun des nœuds, on préfère manipuler la distance $\boldsymbol{d}$ du contact dans la base locale $(\boldsymbol{n}, \boldsymbol{t}_1, \boldsymbol{t}_2)$. La projection orthogonale du nœud N_1 sur la face en P donne la distance normale d_n qui, à convergence, doit satisfaire au mieux l'égalité $(d_n = 0)$. Cette condition dicte les contraintes cinématiques à prendre en compte grâce aux multiplicateurs de Lagrange $\boldsymbol{\xi}$.

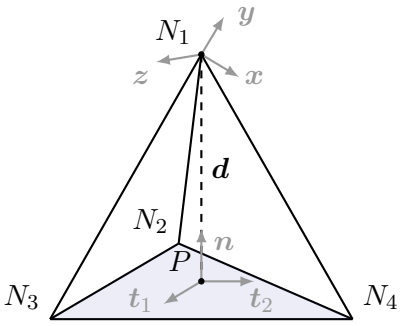


FIGURE 7.1 – Exemple d'un élément de contact 3D de typologie *nœuds / faces* connectant le nœud N_1 aux 3 autres nœuds N_2, N_3 et N_4 formant la surface en regard

REMARQUE 7.3 : Contrairement aux éléments volumiques classiques, les éléments de contact ne sont pas « figés » puisqu'au cours du calcul un même nœud peut successivement rentrer en contact avec différentes surfaces. C'est le cas notamment dès que les déplacements tangentiels deviennent importants au niveau de la zone de contact.

En définissant tout d'abord les quantités α_n et $\boldsymbol{\alpha}_t$, dépendantes de la trace des déplacements généralisés $\mathbf{v}$ aux nœuds de l'élément, par les relations (7.3) et (7.4) suivantes :

$$\alpha_n(\mathbf{v}) = k_c\xi_n + p_c d_n(\mathbf{v}) \tag{7.3}$$
$$\boldsymbol{\alpha}_t(\mathbf{v}) = k_c\boldsymbol{\xi}_t + p_c\delta\boldsymbol{d}_t(\mathbf{v}) \tag{7.4}$$

avec $\delta\boldsymbol{d}_t$ les variations de déplacements tangentiels durant le pas de chargement, on construit alors la fonctionnelle ψ de type point-selle (7.5) par la méthode du Lagrangien augmenté où $k_c \in \mathbb{R}_+$ et $p_c \in \mathbb{R}_+$ représentent respectivement un facteur d'échelle et un paramètre de régularisation :

$$\begin{aligned}\psi(\mathbf{v}, \xi_n, \boldsymbol{\xi}_t) = {} & k_c\xi_n d_n(\mathbf{v}) + \frac{p_c}{2}d_n^2(\mathbf{v}) - \frac{1}{2p_c}\langle\alpha_n(\mathbf{v})\rangle_+^2 \\ & + k_c\boldsymbol{\xi}_t\cdot\delta\boldsymbol{d}_t(\mathbf{v}) + \frac{p_c}{2}\delta\boldsymbol{d}_t(\mathbf{v})\cdot\delta\boldsymbol{d}_t(\mathbf{v}) - \frac{1}{2p_c}\langle\,\|\boldsymbol{\alpha}_t(\mathbf{v})\|_2 - \mu\,|\alpha_n(\mathbf{v})|\,\rangle_+^2\end{aligned} \tag{7.5}$$

Ces deux paramètres k_c et p_c sont soit fixés par l'utilisateur, soit déterminés automatiquement par le logiciel au cours de la résolution. Nous décrivons par cette formule le cas du contact isotrope où les coefficients de frottement, suivant les deux directions tangentielles, sont supposés identiques – on note $\mu = \mu_{t_1} = \mu_{t_2}$ pour simplifier. Des formes similaires sont proposées dans [Alart et Curnier, 1991, Pietrzak et Curnier, 1999] bien qu'une différence réside au niveau du facteur d'échelle k_c intervenant conjointement aux multiplicateurs de Lagrange. L'interprétation physique de ces derniers s'en retrouve modifiée : dans Simcenter Samcef™ les multiplicateurs de Lagrange $\boldsymbol{\xi}$ sont homogènes à un déplacement d'interface plutôt qu'à un effort d'interface. Les efforts de contact, ainsi que la matrice de rigidité élémentaire, découlent alors respectivement des dérivées partielles premières et secondes de ce potentiel ψ qui prend une forme

particulière – équations (7.6a – 7.6c) – en fonction des statuts des contacts (cf. Tableau 7.1) :

$$\psi \equiv \begin{cases} k_c \xi_n d_n + \dfrac{p_c}{2} d_n^2 - \dfrac{1}{2p_c}\alpha_n^2 - \dfrac{k_c^2}{2p_c}\boldsymbol{\xi}_t \cdot \boldsymbol{\xi}_t & \text{si } \alpha_n > 0 \quad (7.6a) \\ \left(k_c \boldsymbol{\xi} + \dfrac{p_c}{2}\boldsymbol{d}\right) \cdot \boldsymbol{d} & \text{si } \alpha_n \leqslant 0 \text{ et } \beta - |\alpha_n| \leqslant 0 \quad (7.6b) \\ \left(k_c \boldsymbol{\xi} + \dfrac{p_c}{2}\boldsymbol{d}\right) \cdot \boldsymbol{d} - \dfrac{1}{2p_c}\Big(\|\boldsymbol{\alpha}_t\|_2 - \mu\,|\alpha_n| \Big)^2 & \text{si } \alpha_n \leqslant 0 \text{ et } \beta - |\alpha_n| > 0 \quad (7.6c) \end{cases}$$

avec $\beta^2 = \left(\dfrac{\alpha_{t_1}}{\mu}\right)^2 + \left(\dfrac{\alpha_{t_2}}{\mu}\right)^2$ le seuil délimitant le passage du contact adhérant (*stick*) au contact glissant (*slip*).

TABLE 7.1 – Différents statuts du contact

	Contact inactif	**Contact actif**	
Contact normal	$\alpha_n > 0$	$\alpha_n \leqslant 0$	
Contact tangentiel	-	***stick*** $\beta - \lvert\alpha_n\rvert \leqslant 0$	***slip*** $\beta - \lvert\alpha_n\rvert > 0$

REMARQUE 7.4 : Il convient de noter la coïncidence appréciable au niveau de l'étape locale entre l'approche LATIN classiquement déployée dans la littérature [Giacoma *et al.*, 2015, Oumaziz *et al.*, 2017] et l'approche orchestrée nativement dans Simcenter Samcef™ – certes bien plus générale dans la mesure où les grandes transformations sont d'office prises en compte. La manipulation du même type de grandeur $(\boldsymbol{\xi}, \mathbf{v})$ par l'intermédiaire de quantités α_n et $\boldsymbol{\alpha}_t$ similaires représente un atout indéniable qui a permis l'élaboration de notre méthodologie.

7.2.2.2 Bref aperçu du cheminement de l'algorithme dédié au contact

L'un des nombreux avantages lié au fait de s'appuyer sur un logiciel industriel réside dans la possibilité de remobiliser l'ensemble des outils existants. La gestion du contact ne fait pas figure d'exception et on tire ainsi parti des nombreux développements, souvent complexes, déjà réalisés et optimisés depuis de longues années. À titre d'exemple, la prise en compte des non-linéarités de type contact nécessite un algorithme de recherche topologique performant pour associer chaque nœud potentiellement en contact avec sa face en regard, mais aussi le calcul des projections, dans toutes les configurations spatiales envisageables, incluant bien souvent des tests de proximité supplémentaires pour déceler si la projection obtenue est licite ou non – c'est-à-dire pas trop éloignée de la face –, ou encore l'ensemble des opérations liées à la cinématique et la sténique du contact. Plus spécifiquement, la recherche topologique du contact en grandes transformations peut aboutir à des situations délicates de *self-contact* qui, pour être prises en compte correctement, nécessitent la plupart du temps la mise en place d'algorithmes d'évaluation coûteux et demandent donc une forte optimisation. Nativement conçu pour fonctionner en grandes transformations, le logiciel Simcenter Samcef™ possède un algorithme performant dans ce domaine. À cela s'ajoute toute l'intelligence dans le cheminement des différents algorithmes dédiés au contact, émanant de l'expérience emmagasinée, ainsi que les multiples options disponibles pour faciliter la convergence – comme l'ajout d'une raideur de contact pour atténuer les discontinuités dans la loi de frottement ou encore la prise en compte des jeux initiaux par exemple. Dans le cadre de notre méthodologie LATIN-PGD faiblement intrusive, nous mettons à profit l'ensemble de ces capacités offertes sans aucune modification.

7.2.2.3 Choix de la direction de recherche

On suppose qu'on connait les quantités $(\mathbf{u}_\ell, \boldsymbol{\xi}_\ell)$ de l'étape linéaire précédente et on cherche les nouvelles quantités $(\widehat{\mathbf{u}}_\ell, \widehat{\boldsymbol{\xi}}_\ell, \widehat{\mathbf{f}}_{\text{int},\ell}, \widehat{\boldsymbol{d}}_\ell)$ par achèvement de l'étape locale. Nous conservons une direction de recherche verticale pour les déplacements (7.7), à savoir :

$$\forall t \in I, \quad \widehat{\mathbf{u}}_\ell(t) = \mathbf{u}_\ell(t) \tag{7.7}$$

tandis que les quantités locales $(\widehat{\boldsymbol{\xi}}_\ell, \widehat{\boldsymbol{d}}_\ell)$ et $(\boldsymbol{\xi}_\ell, \boldsymbol{d}_\ell)$ sont reliées par la direction de recherche (7.8) que nous choisissons d'adopter :

$$\forall t \in I, \quad p_c \widehat{\boldsymbol{d}}_\ell(t) - p_c \boldsymbol{d}_\ell(t) = k_c \boldsymbol{\xi}_\ell(t) - k_c \widehat{\boldsymbol{\xi}}_\ell(t) \tag{7.8}$$

L'intégration des lois locales au niveau de chaque élément contact permet finalement de clore le système d'équations où les efforts de contact locaux sont donnés par les formules (7.9) ci-dessous :

$$k_c \widehat{\xi}_{n,\ell} = -\langle \alpha_{n,\ell} \rangle_- \qquad \begin{cases} k_c \widehat{\boldsymbol{\xi}}_{t,\ell} = \boldsymbol{\alpha}_{t,\ell} & \text{si } \alpha_{n,\ell} \leqslant 0 \text{ et } \beta_\ell - |\alpha_{n,\ell}| \leqslant 0 \quad (\textit{stick}) \\ k_c \widehat{\boldsymbol{\xi}}_{t,\ell} = \boldsymbol{\alpha}_{t,\ell} \dfrac{|\alpha_{n,\ell}|}{\beta_\ell} & \text{si } \alpha_{n,\ell} \leqslant 0 \text{ et } \beta_\ell - |\alpha_{n,\ell}| > 0 \quad (\textit{slip}) \end{cases} \tag{7.9}$$

$$\alpha_{n,\ell} = k_c \xi_{n,\ell} + p_c d_{n,\ell} \qquad \boldsymbol{\alpha}_{t,\ell} = k_c \boldsymbol{\xi}_{t,\ell} + p_c \delta \boldsymbol{d}_{t,\ell} \qquad \beta_\ell = \frac{1}{\mu} \|\boldsymbol{\alpha}_{t,\ell}\|_2$$

La méthodologie a été pensée en parfaite adéquation avec le logiciel industriel Simcenter Samcef™ de sorte que l'on puisse réutiliser sans modification les différentes routines préexistantes. Aussi, la construction de l'opérateur $\boldsymbol{\mathcal{A}}_c$ s'effectue de manière analogue à celle de l'opérateur $\boldsymbol{\mathcal{A}}_u$ intervenant dans l'intégration des diverses lois matériaux. On retrouve, à l'issue de la génération des éléments, les efforts de contact calculés par le logiciel industriel sous la forme $\widehat{\mathbf{f}}_{\text{ctc},\ell}(t) = k_c {}^t\mathbf{B} \widehat{\boldsymbol{\xi}}_\ell(t)$ où ${}^t\mathbf{B}$ désigne l'opérateur de changement de base permettant de réaliser la projection des quantités locales sur les nœuds de l'élément. Les efforts internes généralisés $\widehat{\mathbf{f}}_{\text{int},\ell}$ sont quant à eux évalués comme d'habitude par intégration des lois de comportement matériaux lors de la construction de l'opérateur $\boldsymbol{\mathcal{A}}_u$ sur l'ensemble de l'intervalle de temps. Un point délicat mérite toutefois que l'on y prête attention : par l'intermédiaire du logiciel industriel, nous récupérons en réalité directement l'ensemble des efforts assemblés, c'est-à-dire qu'il n'est *a priori* pas possible de séparer les contributions entre les efforts internes $\widehat{\mathbf{f}}_{\text{int},\ell}$ et les efforts de contact $\widehat{\mathbf{f}}_{\text{ctc},\ell}$, provenant de l'intégration respective des lois matériaux et des lois de contact. Afin de limiter le niveau d'invasivité de la méthode LATIN-PGD, nous récupérons donc uniquement la contribution totale des efforts généralisés $\widehat{\mathbf{f}}_{\text{tot},\ell} = \widehat{\mathbf{f}}_{\text{int},\ell} + \widehat{\mathbf{f}}_{\text{ctc},\ell}$ de manière indivisible, ainsi que l'opérateur $\mathbf{B}$ et les paramètres k_c et p_c qui nous serviront lors de l'étape linéaire.

En résumé, l'étape locale consiste à injecter au début de chaque pas de temps les quantités $(\mathbf{u}_\ell, \boldsymbol{\xi}_\ell)$ connues de l'étape linéaire précédente, pour récupérer en définitive l'ensemble des efforts généralisés $\widehat{\mathbf{f}}_{\text{tot},\ell}$ provenant de toutes les contributions possibles, éventuellement l'opérateur de raideur – de manière indivisible également –, ainsi que quelques quantités supplémentaires liées au contact. Typiquement, connaissant les deux facteurs k_c et p_c, nous pouvons aisément remonter aux valeurs de $\widehat{\boldsymbol{d}}_\ell$ par l'intermédiaire de la direction de recherche (7.8) réécrite en (7.10) sous un format différent :

$$\begin{cases} p_c \widehat{d}_{n,\ell} = \alpha_{n,\ell} - k_c \widehat{\xi}_{n,\ell} = \langle \alpha_{n,\ell} \rangle_+ \\ p_c \delta \widehat{\boldsymbol{d}}_{t,\ell} = \boldsymbol{\alpha}_{t,\ell} - k_c \widehat{\boldsymbol{\xi}}_{t,\ell} \end{cases} \tag{7.10}$$

Pour ne pas intervenir de manière intempestive dans le code existant, ces calculs sont menés en amont de l'étape globale – bien qu'appartenant théoriquement à l'étape locale. Le fait d'avoir besoin de travailler sur des quantités locales, au niveau de l'élément, rend nécessairement cette méthodologie dédiée au contact légèrement plus intrusive par rapport aux cas non-linéaires matériaux ou géométriques.

REMARQUE 7.5 : La direction de recherche choisie (7.8) possède comme propriété particulière de satisfaire la relation $\delta\widehat{\boldsymbol{d}}_{t,\ell} = 0$ dans le cas du contact adhérant, tandis qu'une valeur non nulle est observée dans le cas du contact glissant : $\delta\widehat{\boldsymbol{d}}_{t,\ell} = \boldsymbol{\alpha}_{t,\ell}\Big(1 - \frac{|\alpha_{n,\ell}|}{\beta_\ell}\Big)$.

7.2.3 Étape linéaire

Faisant suite à l'étape locale précédente, et connaissant dorénavant les quantités $(\widehat{\mathbf{u}}_\ell, \widehat{\boldsymbol{\xi}}_\ell, \widehat{\mathbf{f}}_{\text{int},\ell}, \widehat{\boldsymbol{d}}_\ell)$, on cherche alors les nouvelles quantités $(\mathbf{u}_{\ell+1}, \boldsymbol{\xi}_{\ell+1})$ par résolution de l'étape linéaire. Nous choisissons en (7.11) une direction de recherche analogue à celle employée lors de l'étape locale, à savoir :

$$\forall t \in I, \quad k_c\boldsymbol{\xi}_{\ell+1}(t) - k_c\widehat{\boldsymbol{\xi}}_\ell(t) = p_c\boldsymbol{d}_{\ell+1}(t) - p_c\widehat{\boldsymbol{d}}_\ell(t) \tag{7.11}$$

où les coefficients k_c et p_c sont issus de l'étape locale, menée par les routines classiques de Simcenter Samcef™. Le système d'équations à résoudre se présente alors sous la forme (7.12) suivante, en notant $\mathbf{B}$ l'opérateur de projection de la restriction des déplacements généralisés aux nœuds de l'élément de contact :

$$\begin{aligned}
&(\mathbf{A}_d): \quad \begin{cases} \mathbf{u}_{\ell+1}(t=0) = \mathbf{0} \\ \forall t \in I, \quad \begin{cases} \boldsymbol{\mathcal{C}}_{\mathbf{u}}\Big(\mathbf{u}_{\ell+1}(t)\Big) = \mathbf{u}_d(t) \\ \boldsymbol{d}_{\ell+1} = \mathbf{B}\mathbf{u}_{\ell+1} \\ \mathbf{f}_{\text{int},\ell+1}(t) + \boldsymbol{\mathcal{C}}_{\boldsymbol{\xi}}\Big(\boldsymbol{\xi}_{\ell+1}(t)\Big) = \mathbf{f}_{\text{ext}}(t) \\ \boldsymbol{\mathcal{C}}_{\boldsymbol{\xi}}\Big(\boldsymbol{\xi}_{\ell+1}(t)\Big) = k_c{}^t\mathbf{B}\boldsymbol{\xi}_{\ell+1}(t) \end{cases} \end{cases} \\
&(\boldsymbol{\Upsilon}^-): \quad \forall t \in I, \quad \begin{cases} \mathbf{H}\Delta\mathbf{u}_{\ell+1}(t) = \mathbf{f}_{\text{int},\ell+1}(t) - \widehat{\mathbf{f}}_{\text{int},\ell}(t) \\ \mathbf{u}_{\ell+1}(t) = \widehat{\mathbf{u}}_\ell(t) + \Delta\mathbf{u}_{\ell+1}(t) \\ k_c\boldsymbol{\xi}_{\ell+1}(t) - k_c\widehat{\boldsymbol{\xi}}_\ell(t) = p_c\boldsymbol{d}_{\ell+1}(t) - p_c\widehat{\boldsymbol{d}}_\ell(t) \end{cases}
\end{aligned} \tag{7.12}$$

où nous conservons la direction de recherche $\mathbf{H}$ pour décrire $\boldsymbol{\Upsilon}^-$ globalement sur l'ensemble de l'espace. Il s'ensuit que la résolution peut s'effectuer de manière découplée, en cherchant tout d'abord la correction $\Delta\mathbf{u}_{\ell+1}$ sous forme à variables séparées telle que l'équation (7.13) soit vérifiée :

$$\forall t \in I, \quad \Big(\mathbf{H} + p_c{}^t\mathbf{B}\mathbf{B}\Big)\Delta\mathbf{u}_{\ell+1}(t) = \mathbf{f}_{\text{ext}}(t) - \Big(\underbrace{\widehat{\mathbf{f}}_{\text{int},\ell}(t) + k_c{}^t\mathbf{B}\widehat{\boldsymbol{\xi}}_\ell(t)}_{\widehat{\mathbf{f}}_{\text{tot},\ell}}\Big) + p_c{}^t\mathbf{B}\Big(\widehat{\boldsymbol{d}}_\ell(t) - \boldsymbol{d}_\ell(t)\Big) \tag{7.13}$$

puis la correction du multiplicateur de Lagrange $\Delta\boldsymbol{\xi}_{\ell+1}$ selon la direction de recherche (7.11) réécrite en (7.14) de manière équivalente :

$$\forall t \in I, \quad \Delta\boldsymbol{\xi}_{\ell+1}(t) = \widehat{\boldsymbol{\xi}}_\ell(t) + \frac{p_c}{k_c}\Big(\boldsymbol{d}_\ell(t) + \mathbf{B}\Delta\mathbf{u}_{\ell+1}(t) - \widehat{\boldsymbol{d}}_\ell(t)\Big) - \boldsymbol{\xi}_\ell(t) \tag{7.14}$$

La résolution de l'équation (7.13) est menée classiquement par technique PGD de manière analogue à ce qui a été présenté au Chapitre-5. Seuls l'opérateur de rigidité et le second membre sont légèrement modifiés : l'un avec l'ajout d'une raideur d'interface, l'autre avec l'apport d'une contribution supplémentaire aisément calculable. De plus, l'opérateur de la direction de recherche $\mathbf{H}$ peut être choisi constant – rigidité élastique $\mathbf{K}$ – ou tangent comme à l'accoutumée. Toutefois, en présence de contact, il est plus difficile de capitaliser sur la factorisation de cet opérateur dans la mesure où dès qu'une condition de contact change, l'opérateur $\mathbf{B}$ varie. Dans le cas général, ce dernier fluctue donc au gré des itérations LATIN et suivant

les pas de temps. Recourir à un opérateur $\mathbf{H}$ constant nécessite alors de mener quelques étapes supplémentaires au cours de la résolution, comme l'utilisation de la formule de Sherman-Morrison-Woodbury généralisée (7.15) entre autre, si l'on souhaite mutualiser la factorisation de l'opérateur :

$$\Big(\breve{\mathbf{H}} + p_c{}^t\breve{\mathbf{B}}\mathbf{I}_{n_c}\breve{\mathbf{B}}\Big)^{-1} = \breve{\mathbf{H}}^{-1} - \breve{\mathbf{H}}^{-1}p_c{}^t\breve{\mathbf{B}}\Big(\mathbf{I}_{n_c}^{-1} + \breve{\mathbf{B}}\breve{\mathbf{H}}^{-1}p_c{}^t\breve{\mathbf{B}}\Big)^{-1}\breve{\mathbf{B}}\breve{\mathbf{H}}^{-1} \tag{7.15}$$

où les différents termes seront explicités par la suite.

On se place à une itération ℓ donnée de l'algorithme où on suppose connue une approximation de $\Delta\mathbf{u}_{\ell+1}$ sous forme à variables séparées. La résolution de l'équation (7.13) par technique PGD s'effectue classiquement au moyen d'une étape préliminaire de mise à jour des fonctions temporelles, avant de générer le cas échéant un nouveau mode. En considérant toujours que l'équation de la direction de recherche n'est vérifiée qu'en moyenne, la recherche d'un nouveau mode passe par la résolution du problème de minimisation (7.16) donné ci-contre :

$$(\boldsymbol{\Lambda}, \lambda) = \underset{\substack{\alpha\in\mathcal{I}_{\mathbb{R}}\\ \boldsymbol{\gamma}\in\boldsymbol{\mathcal{V}}}}{\arg\min}\ \breve{J}\Big(\alpha\boldsymbol{\gamma} + \sum_{i=1}^{m_\ell}\beta_i\boldsymbol{\Lambda}_i\Big) \tag{7.16}$$

où la fonctionnelle $\breve{J}$ est définie par la formule (7.17) et $\{\beta_i\}_{1\leqslant i\leqslant m_\ell}$ représentent les fonctions temporelles issues de la première étape de mise à jour :

$$\breve{J}(\mathbf{w}) = \frac{1}{2}\int_I {}^t\mathbf{w}(t)\Big[\Big(\mathbf{H} + p_c{}^t\mathbf{B}\mathbf{B}\Big)\mathbf{w}(t) - 2\breve{\boldsymbol{\mathcal{Q}}}(t)\Big]\,\mathrm{d}t \tag{7.17}$$

avec $\breve{\boldsymbol{\mathcal{Q}}} = \mathbf{f}_{\text{ext}} - \widehat{\mathbf{f}}_{\text{tot},\ell} + p_c{}^t\mathbf{B}(\widehat{\boldsymbol{d}}_\ell - \boldsymbol{d}_\ell)$ le second membre de l'équation (7.13) pour simplifier les notations. Nous aboutissons alors à un problème classique de type point fixe (7.18) que nous écrivons formellement :

$$\begin{cases} \boldsymbol{\Lambda} = \Xi_{\boldsymbol{\Lambda}}(\lambda) \\ \lambda = \Xi_\lambda(\boldsymbol{\Lambda}) \end{cases} \tag{7.18}$$

où les fonctions $\Xi_{\boldsymbol{\Lambda}}(\lambda)$ et $\Xi_\lambda(\boldsymbol{\Lambda})$ sont respectivement données par les formules (7.19) et (7.20) suivantes :

$$\Xi_{\boldsymbol{\Lambda}}(\lambda) = \Big[\int_I \lambda(t)\Big(\mathbf{H} + p_c{}^t\mathbf{B}\mathbf{B}\Big)\lambda(t)\,\mathrm{d}t\Big]^{-1}\int_I \lambda(t)\breve{\breve{\boldsymbol{\mathcal{Q}}}}(t)\,\mathrm{d}t \tag{7.19}$$

$$\Xi_\lambda(\boldsymbol{\Lambda}) = \frac{{}^t\boldsymbol{\Lambda}\breve{\breve{\boldsymbol{\mathcal{Q}}}}}{{}^t\boldsymbol{\Lambda}\Big(\mathbf{H} + p_c{}^t\mathbf{B}\mathbf{B}\Big)\boldsymbol{\Lambda}} \tag{7.20}$$

avec $\breve{\breve{\boldsymbol{\mathcal{Q}}}} = \breve{\boldsymbol{\mathcal{Q}}} - \Big(\mathbf{H} + p_c{}^t\mathbf{B}\mathbf{B}\Big)\sum_{i=1}^{m_\ell}\beta_i\boldsymbol{\Lambda}_i$ obtenu par déflation.

En considérant alors un opérateur $\mathbf{H}$ constant – indépendant du temps –, il s'ensuit que l'équation (7.19) peut se mettre sous la forme (7.21) à laquelle nous pouvons ensuite appliquer la formule (7.15) de Sherman-Morrison-Woodbury généralisée :

$$\Xi_{\boldsymbol{\Lambda}}(\lambda) = \Big[\underbrace{\Big(\int_I \lambda^2(t)\,\mathrm{d}t\Big)\mathbf{H}}_{\breve{\mathbf{H}}} + \underbrace{p_c\int_I \lambda(t){}^t\mathbf{B}\mathbf{B}\lambda(t)\,\mathrm{d}t}_{p_c{}^t\breve{\mathbf{B}}\mathbf{I}_{n_c}\breve{\mathbf{B}}}\Big]^{-1}\int_I \lambda(t)\breve{\breve{\boldsymbol{\mathcal{Q}}}}(t)\,\mathrm{d}t \tag{7.21}$$

avec $\mathbf{I}_{n_c}$ l'opérateur identité de taille $(n_c \times n_c)$. Ainsi, plutôt que de devoir inverser un opérateur de taille $(n \times n)$ à chaque itération du point fixe, l'inversion n'est réalisée que sur un opérateur de taille bien plus

raisonnable – de taille $(n_c \times n_c)$ avec généralement $n_c \ll n$.

Par ailleurs, le choix des directions de recherche qui a été fait conduit à un découplage des résolutions, en permettant de trouver en premier lieu la correction du champ des déplacements généralisés avant d'en déduire la correction des multiplicateurs de Lagrange à appliquer. Cette manière de procéder fait ainsi abstraction du terme de couplage potentiel intervenant entre les inconnues $\Delta\mathbf{u}_{\ell+1}$ et $\Delta\boldsymbol{\xi}_{\ell+1}$ lors de la recherche de la correction des déplacements généralisés. Écrit sous forme algébrique, le système non-symétrique à résoudre peut se mettre sous la forme (7.22) suivante :

$$\begin{bmatrix} \mathbf{H} + p_c{}^t\mathbf{B}\mathbf{B} & \mathbf{0} \\ -p_c\mathbf{B} & k_c\,\mathbf{I}_{n_c} \end{bmatrix} \begin{bmatrix} \Delta\mathbf{u}_{\ell+1}(t) \\ \Delta\boldsymbol{\xi}_{\ell+1}(t) \end{bmatrix} = \begin{bmatrix} \mathbf{f}_{\text{ext}}(t) - \widehat{\mathbf{f}}_{\text{tot},\,\ell} + p_c{}^t\mathbf{B}\Big(\widehat{\boldsymbol{d}}_\ell(t) - \boldsymbol{d}_\ell(t)\Big) \\ k_c\widehat{\boldsymbol{\xi}}_\ell(t) + p_c\Big(\boldsymbol{d}_\ell(t) - \widehat{\boldsymbol{d}}_\ell(t)\Big) - k_c\boldsymbol{\xi}_\ell(t) \end{bmatrix} \tag{7.22}$$

Un choix différent des directions de recherche pourrait conduire à prendre en compte ces termes de couplages, intervenant naturellement dans le logiciel industriel Simcenter Samcef™.

7.2.4 Critère additionnel de vérification des efforts de contact

Par rapport au Chapitre-5, nous ajoutons un critère supplémentaire pour vérifier la bonne convergence des efforts de contact. Ce critère prend la forme (7.23) suivante en désignant par $\|\bullet\|_\infty$ la norme infinie sur $\boldsymbol{\mathcal{V}} \times I$:

$$\omega_\ell^{\boldsymbol{\xi}} = \left[\frac{2\left\|\boldsymbol{s}_\ell^{\boldsymbol{\xi}} - \widehat{\boldsymbol{s}}_\ell^{\boldsymbol{\xi}}\right\|_\infty}{\left\|\boldsymbol{s}_\ell^{\boldsymbol{\xi}}\right\|_\infty + \left\|\widehat{\boldsymbol{s}}_\ell^{\boldsymbol{\xi}}\right\|_\infty} \right]^{1/2} \tag{7.23}$$

où les quantités $\boldsymbol{s}_\ell^{\boldsymbol{\xi}}$ et $\widehat{\boldsymbol{s}}_\ell^{\boldsymbol{\xi}}$ sont données par les relations (7.24) définies respectivement à partir de la norme 2 – notée $\|\bullet\|_2$ – des fonctions multi-variées $\boldsymbol{s}_\ell^\diamond = (k_c\boldsymbol{\xi}_{\ell+1}, p_c\boldsymbol{d}_{\ell+1})$ et $\widehat{\boldsymbol{s}}_\ell^\diamond = (k_c\widehat{\boldsymbol{\xi}}_\ell, p_c\widehat{\boldsymbol{d}}_\ell)$:

$$\begin{cases} \boldsymbol{s}_\ell^{\boldsymbol{\xi}} = \|\boldsymbol{s}_\ell^\diamond\|_2 = k_c^2\boldsymbol{\xi}_{\ell+1}^2 + p_c^2\boldsymbol{d}_{\ell+1}^2 \\ \widehat{\boldsymbol{s}}_\ell^{\boldsymbol{\xi}} = \|\widehat{\boldsymbol{s}}_\ell^\diamond\|_2 = k_c^2\widehat{\boldsymbol{\xi}}_\ell^{\,2} + p_c^2\widehat{\boldsymbol{d}}_\ell^{\,2} \end{cases} \tag{7.24}$$

Un critère analogue peut être trouvé dans [Giacoma *et al.*, 2015] en séparant les contributions normales et tangentielles (7.25) sous la forme :

$$\omega_{n/t,\,\ell}^{\boldsymbol{\xi}} = \max\left[\frac{2\left\|s_{n,\,\ell}^{\boldsymbol{\xi}} - \widehat{s}_{n,\,\ell}^{\boldsymbol{\xi}}\right\|_\infty}{\left\|s_{n,\,\ell}^{\boldsymbol{\xi}}\right\|_\infty + \left\|\widehat{s}_{n,\,\ell}^{\boldsymbol{\xi}}\right\|_\infty} ; \frac{2\left\|\boldsymbol{s}_{t,\,\ell}^{\boldsymbol{\xi}} - \widehat{\boldsymbol{s}}_{t,\,\ell}^{\boldsymbol{\xi}}\right\|_\infty}{\left\|\boldsymbol{s}_{t,\,\ell}^{\boldsymbol{\xi}}\right\|_\infty + \left\|\widehat{\boldsymbol{s}}_{t,\,\ell}^{\boldsymbol{\xi}}\right\|_\infty} \right]^{1/2} \tag{7.25}$$

où les quantités $\boldsymbol{s}_{\bullet,\,\ell}^{\boldsymbol{\xi}}$ et $\widehat{\boldsymbol{s}}_{\bullet,\,\ell}^{\boldsymbol{\xi}}$ sont données par les relations (7.26) suivantes :

$$\begin{cases} s_{n,\,\ell}^{\boldsymbol{\xi}} = k_c^2\xi_{n,\,\ell+1}^2 + p_c^2 d_{n,\,\ell+1}^2 & \boldsymbol{s}_{t,\,\ell}^{\boldsymbol{\xi}} = k_c^2\boldsymbol{\xi}_{t,\,\ell+1}^2 + p_c^2\delta\boldsymbol{d}_{t,\,\ell+1}^2 \\ \widehat{s}_{n,\,\ell}^{\boldsymbol{\xi}} = k_c^2\widehat{\xi}_{n,\,\ell}^{\,2} + p_c^2\widehat{d}_{n,\,\ell}^{\,2} & \widehat{\boldsymbol{s}}_{t,\,\ell}^{\boldsymbol{\xi}} = k_c^2\widehat{\boldsymbol{\xi}}_{t,\,\ell}^{\,2} + p_c^2\delta\widehat{\boldsymbol{d}}_{t,\,\ell}^{\,2} \end{cases} \tag{7.26}$$

La convergence de l'algorithme est assurée lorsque la valeur seuil fixée par l'utilisateur pour chacun des critères est atteinte : $\omega_\ell \leqslant \eta$ globalement et $\omega_\ell^{\boldsymbol{\xi}} \leqslant \eta^{\boldsymbol{\xi}}$ pour le contact.

7.3 Illustration sur un cas-test académique

Un troisième cas-test académique est dévoilé dans ce chapitre afin de mieux cerner le fonctionnement de la méthode LATIN-PGD faiblement intrusive dans le logiciel Simcenter Samcef™ en présence de contact.

7.3.1 Description du cas-test

Le troisième cas-test académique est constitué de deux solides déformables en contact l'un avec l'autre (cf. Figure 7.2). Le disque, encastré au niveau de sa circonférence, se déforme sous l'action d'un indenteur cylindrique soumis à un déplacement imposé suivant z à son extrémité. Les conditions de contact font transiter les efforts entre l'indenteur, supposé linéaire élastique, et le disque composé d'un matériau élasto-visco-plastique de type Lemaitre-Chaboche. Le Tableau 7.2 recense les différentes valeurs retenues pour ce modèle. Un coefficient de frottement $\mu = 0.08$ est considéré dans la zone de contact. Le maillage, élaboré à partir d'éléments volumiques tétraédriques du premier ordre, compte 2.6×10^4 degrés de liberté dont 3×10^3 multiplicateurs de Lagrange associés aux conditions de contact. Avec une amplitude de chargement de 3 mm, nous considérons dans cet exemple les non-linéarités matérielles et géométriques ainsi que celles liées aux contacts – pour lesquelles l'opérateur $\mathbf{B}$ peut varier.

TABLE 7.2 – Paramètres matériaux des lois de comportement élastique (indenteur) et élasto-visco-plastique de type Lemaitre-Chaboche (disque)

	E [GPa]	ν [-]	σ_0 [MPa]	k [MPa]	n [-]	γ [-]	C [MPa]	Q [MPa]	b [-]
Indenteur	210	0.28	-	-	-	-	-	-	-
Disque	69	0.28	60	150	12	300	24800	80	10

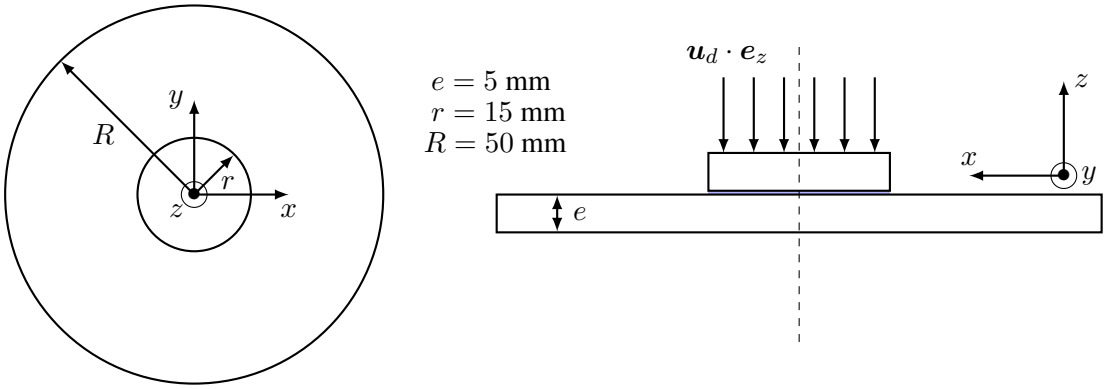
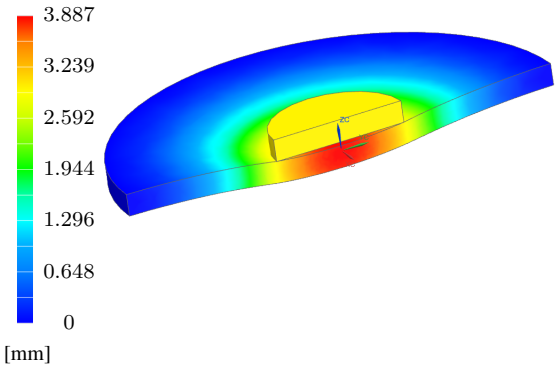


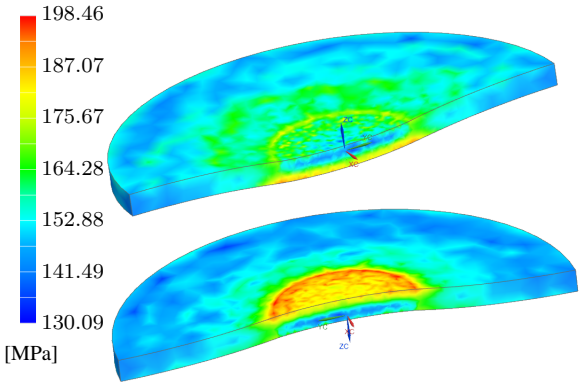
FIGURE 7.2 – Troisième cas académique : contact d'un indenteur sur un disque

7.3.2 Analyse des résultats

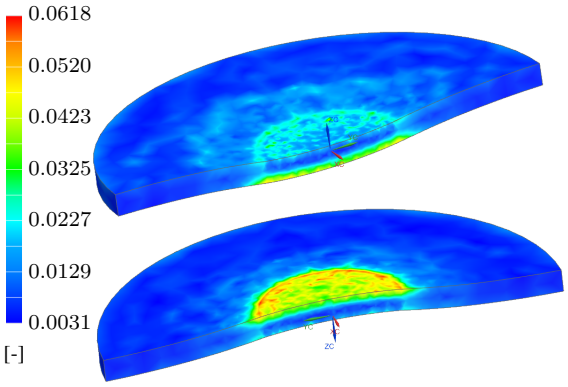
La Figure 7.3 présente une vue en coupe des champs de déplacements, de contraintes de Von Mises et de déformations plastiques cumulées correspondant au pas de temps le plus chargé. Le disque se déforme en son centre avec une localisation des contraintes et des déformations sur la face inférieure.



(a) Champ de déplacements



(b) Champ de contraintes de Von Mises (deux vues)



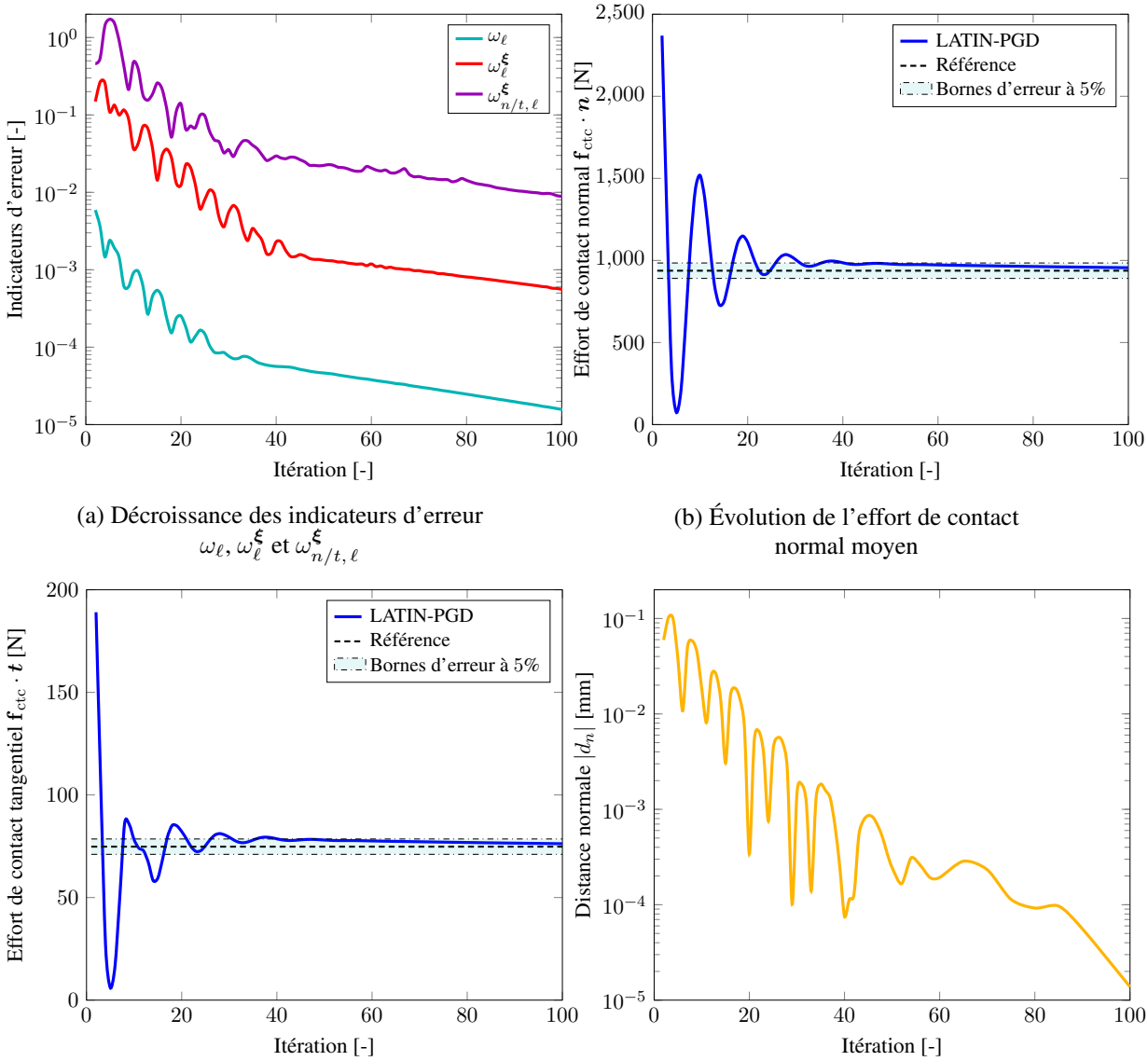
(c) Champ de déformations plastiques cumulées (deux vues)

FIGURE 7.3 – Vue en coupe de différents résultats au pas de temps le plus chargé : (a) champ de déplacements, (b) champ de contraintes de Von Mises et (c) champ de déformations plastiques cumulées

La Figure 7.4 apporte quelques détails supplémentaires quant à la convergence des différentes quantités liées au contact au cours des itérations LATIN-PGD. Un opérateur tangent $\mathbf{H}_\ell$ a été considéré pour la direction de recherche. On observe en particulier une convergence relativement rapide des efforts de contact – normal et tangentiel – avec quelques oscillations sur les premières itérations (cf. sous-figures 7.4b et 7.4c). Ces oscillations parasites s'atténuent rapidement pour obtenir une bonne estimation des efforts de contact – à 5% près – mais deux à trois fois plus d'itérations sont encore nécessaires pour atteindre la convergence fine – erreur inférieure à 1%. La distance normale de contact diminue progressivement à mesure que le nombre d'itérations augmente (cf. sous-figure 7.4d) – avec une fréquence d'oscillation deux fois supérieure du fait de la valeur absolue $|d_n|$ retenue.

Enfin, la sous-figure 7.4a illustre la décroissance des différents indicateurs d'erreur ω_ℓ, $\omega_\ell^{\boldsymbol{\xi}}$ et $\omega_{n/t,\ell}^{\boldsymbol{\xi}}$, ce qui permet de corroborer les analyses précédentes. L'indicateur d'erreur global ω_ℓ converge rapidement vers la valeur seuil fixée à $\eta = 10^{-4}$ correspondant au moment où l'atténuation des différentes oscillations est constatée ; l'approximation est donc globalement correcte – d'un point de vue énergétique. Mais des itérations supplémentaires sont ensuite nécessaires pour satisfaire le second critère $\omega_\ell^{\boldsymbol{\xi}}$ portant sur les quantités liées au contact. Le seuil, que nous fixons usuellement à $\eta^{\boldsymbol{\xi}} = 10^{-3}$, permet d'atteindre une approximation de bonne qualité. En séparant les contributions normales et tangentielles, l'indicateur $\omega_{n/t,\ell}^{\boldsymbol{\xi}}$ possède un taux de convergence similaire mais présente un offset. De manière générale, on pourra

retenir que ces différents indicateurs arborent deux pentes : une convergence relativement rapide au début, conduisant à une approximation globalement correcte, puis une convergence plus lente pour saisir une approximation haute fidélité.



(a) Décroissance des indicateurs d'erreur ω_ℓ, $\omega_\ell^{\boldsymbol{\xi}}$ et $\omega_{n/t,\ell}^{\boldsymbol{\xi}}$

(b) Évolution de l'effort de contact normal moyen

(c) Évolution de l'effort de contact tangentiel moyen

(d) Évolution de la distance normale de contact

FIGURE 7.4 – Évolution des quantités liées au contact et des indicateurs d'erreur en fonction des itérations LATIN-PGD

La méthodologie mise en place pour le contact a montré des résultats satisfaisants pour les cas de glissement modéré que nous avons testés. L'étude des cas à forts glissements, et le couplage avec les non-linéarités géométriques intervenant en grandes transformations, restent encore à étudier.

⋆ ⋆
⋆

Troisième partie

Implémentation numérique et illustrations sur quelques cas-tests

Chapitre 8

Implémentation pratique dans un code industriel

Ce chapitre évoque quelques détails pratiques d'implémentation dans le logiciel industriel Simcenter Samcef™. Un schéma arbore en particulier l'ensemble des étapes mentionnées dans les chapitres précédents afin d'offrir une vision claire et synthétique de la démarche faiblement intrusive proposée. Pour finir, on met l'accent sur les performances en précisant quelques points d'optimisation qui ont été regardés attentivement.

Sommaire

8.1 Détails sur l'implémentation

Une approche faiblement intrusive est adoptée pour incorporer la méthode LATIN-PGD dans le logiciel industriel Simcenter Samcef™ sans aucune interférence. Cette approche a été élaborée pour répondre aux différentes exigences industrielles.

8.1.1 Contraintes à satisfaire

Une des spécificités de notre approche réside dans la nature même de la collaboration instituée. En effet, la société Siemens Digital Industries Software, en tant qu'éditeur de logiciels, fournit des outils avancés en matière de simulations à une large communauté scientifique, y compris à d'autres acteurs industriels, et non uniquement à un partenaire spécifique, comme un fabricant de moteurs d'avion par exemple. Les questions de généralité et de robustesse des solutions proposées sont donc fondamentales. Autrement dit, on doit être capable de traiter tout type de problème incluant tout type d'élément, tout type de relation de comportement, tout type de chargement, etc., et ce quelque soit le secteur industriel dont est issue l'application.

N'étant pas un simple utilisateur de logiciels éléments finis industriels, nous avons aisément accès à l'intérieur de ces logiciels – au code source – ce qui soulève de nouvelles opportunités mais également de nouvelles contraintes. D'une part, l'ensemble des fonctionnalités et des routines du logiciel industriel deviennent accessibles ; mais d'autre part, et pour des raisons évidentes d'ergonomie et d'expérience utilisateur, on souhaite disposer d'une version unifiée incluant d'office tous les outils propres aux méthodes de réduction de modèles – et plus spécifiquement propres à la méthode LATIN-PGD développée. Par conséquent, on aspire non seulement à construire des modèles réduits mais aussi à les manipuler, voire les enrichir, directement au sein du logiciel industriel – ou par l'intermédiaire de son interface graphique. Dans cette optique, le recours à un code externe pour effectuer les différentes opérations semble être un frein, au même titre qu'il est impératif de ne pas bouleverser l'architecture du logiciel existant : la méthode LATIN-PGD doit pouvoir s'accommoder sans aucune entrave aux outils préexistants, d'où l'idée d'une l'implémentation native – au plus proche du code source – faiblement intrusive comme nous l'avions évoqué au Chapitre-3.

Une autre aspiration réside dans la possibilité d'utiliser les mêmes étapes de certification lors des phases de construction de chaque nouvelle version. Ces étapes de certification permettent d'assurer que la méthode LATIN-PGD fonctionne correctement, de manière fiable et robuste pour l'ensemble des cas possibles, mais surtout de vérifier qu'aucun effet indésirable n'est produit par interaction avec le logiciel commercial.

8.1.2 Méthodologie mise en œuvre

L'un des objectifs a été de définir une interface dédiée à la méthode LATIN-PGD afin que celle-ci puisse s'intégrer de manière transparente dans le logiciel Simcenter Samcef™. Notre choix s'est porté sur un module Fortran permettant d'allier les nouvelles capacités inhérentes à la méthode LATIN-PGD avec l'ensemble des fonctionnalités déjà présentes dans le logiciel hôte industriel. La Figure 8.1 fournit l'architecture générale de la méthodologie retenue dont on détaille par la suite les différents blocs.

Étape globale – nouveautés des modèles réduits. Cette étape, qui englobe la majeure partie des nouveaux algorithmes caractérisant la méthode PGD, est de fait une étape intrusive par essence – nécessitant des opérations et opérateurs atypiques des codes industriels standards comme abordé au Chapitre-3. Usuellement traitée en externe, cette étape est maintenue indépendante ce qui signifie qu'il n'y a aucune risque d'endommager le logiciel existant. Plutôt qu'en Fortran, cette partie aurait d'ailleurs très bien pu être externalisée dans un autre langage (en C, en Matlab® ou encore en Python™), mais pour des raisons de performances, il a semblé plus approprié de conserver un code Fortran compilé. Cette manière de procéder permet en effet d'exploiter pleinement les différentes opérations – SVD –, les différentes bibliothèques d'algèbre linéaire – BLAS ou LAPACK incluses nativement dans la librairie Intel® MKL – ou encore les différents solveurs – MUMPS – déjà intégrés et gérés efficacement en harmonie avec le compilateur. Cette partie du code étant conservée volontairement indépendante, sa potentielle intrusivité est d'office reléguée au niveau de l'interface.

Interface – lieu des échanges de données. L'interface est justement le lieu où transite – de manière exclusive – l'ensemble des informations, faisant le lien entre l'étape globale précédente et le logiciel industriel qui réalise l'intégralité de l'étape locale. Prenant la forme d'un module Fortran, cette interface a été conçue pour minimiser le nombre d'interactions avec le logiciel industriel. Elle englobe en son sein l'ensemble des données à véhiculer (grandeurs généralisées), stocke les modes PGD (sous un format réduit) et régit les différents protocoles de communication au travers de diverses opérations permettant de jongler entre les différents formats de données – complets (*full*) ou réduits. Qui plus est, l'interface concentre l'intégralité des paramètres de la méthode, tout en étant capable d'interagir en interne avec les

étapes locales et globales, mais aussi en externe avec n'importe quelle base de données préexistante. Ainsi, cette interface est en quelque sorte le noyau de la méthode LATIN-PGD, rassemblant en un seul et même endroit toute l'intelligence de la méthodologie développée.

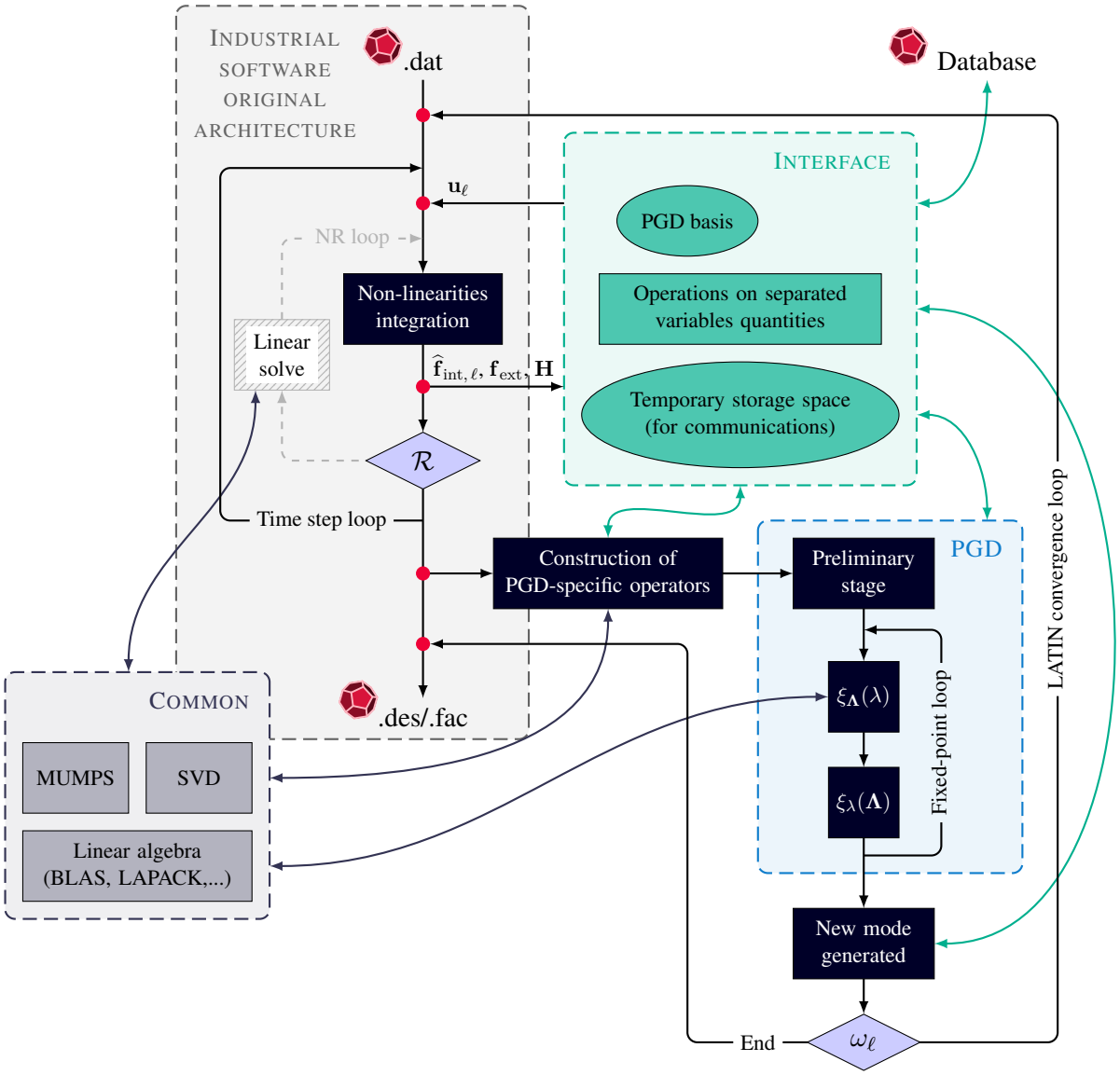


FIGURE 8.1 – Architecture de la méthodologie faiblement intrusive mise en place afin d'introduire la méthode LATIN-PGD dans un logiciel éléments finis industriel *généraliste* tel que Simcenter Samcef™

Étape locale – capitalisation sur le logiciel industriel. Pour finir, l'étape locale repose sur l'architecture incrémentale originelle du logiciel industriel de sorte que l'ensemble des capacités offertes classiquement par le logiciel industriel Simcenter Samcef™ soit conservé. L'étape locale, dont le principal objectif est de façonner l'opérateur $\mathcal{A}_u$ non-linéaire sur l'ensemble de l'intervalle de temps, a été pensée pour ne nécessiter aucune modification particulière dans l'architecture existante, le cheminement classique du logiciel industriel demeurant intact. En effet, par un simple jeu sur les critères d'arrêt de chacun des indicateurs d'erreur, il est possible de ne jamais opérer la boucle de convergence interne de l'algorithme de Newton-Raphson – correspondant à la recherche de la correction à apporter par la résolution du système linéarisé. Deux points seulement demandent une attention particulière au niveau de l'étape locale :

- le premier consiste à fournir, en chaque pas de temps, les quantités primales en amont de la gé-

nération des éléments pour pouvoir récupérer les quantités duales à l'issue. Cela est réalisé par l'intermédiaire de l'interface et ne nécessite qu'une opération de branchement de cette dernière au sein du code source du logiciel industriel. Bien que nous ayons pris l'initiative d'utiliser une interface spécifiquement conçue pour la méthode LATIN-PGD faiblement intrusive pour des raisons de performance, il aurait été tout à fait envisageable, par ailleurs, d'exploiter le superviseur de Simcenter Samcef™, précisément destiné à accomplir des couplages entre logiciels – pour réaliser des simulations multi-physiques par exemple. L'emploi de ce superviseur, disponible nativement dans Simcenter Samcef™ au même titre que d'autres coupleurs génériques, aurait permis de réduire encore davantage la voilure en minimisant le nombre de changements à apporter au code source ;

- le second point concerne l'architecture non-incrémentale, inhérente à la méthode LATIN-PGD, où il convient de surcroît d'ajouter une boucle supplémentaire sur les itérations LATIN englobant l'intégralité des différentes opérations susmentionnées. Ceci constitue finalement le seul point légèrement intrusif de notre approche. Le choix d'inclure cette boucle de manière interne au logiciel industriel permet de satisfaire au mieux les exigences en termes de performances. Un autre choix possible – tout à fait non-intrusif – aurait été de définir cette boucle de manière externe au logiciel industriel. Pour mener à bien la résolution, il conviendrait alors de faire appel non plus une et une seule fois au logiciel industriel, mais autant de fois que d'itérations LATIN nécessaires pour atteindre la convergence. Ces multiples appels dégraderaient sensiblement les performances dans la mesure où les phases d'initialisation et post-calculatoire représentent un temps non négligeable comme nous le verrons au Chapitre-9 (Section-9.1.4). C'est en ce sens que cette solution n'a pas été retenue.

REMARQUE 8.1 : Pour simplifier, le schéma (Figure 8.1) est représenté ici à l'image de la méthodologie développée au Chapitre-5 portant sur les non-linéarités matériaux – au plus un mode PGD généré par itération. Il demeure néanmoins aisément extensible aux autres cadres formulés, les seules modifications apparaissant au niveau de l'étape globale (version avec conditionneur $\mathbf{H}_c$, gestion des multiplicateurs de Lagrange, etc.).

8.2 Gestion des extra-paramètres

Lors de la présentation de la méthode LATIN-PGD au Chapitre-5, nous avions volontairement laissé en suspens la gestion des paramètres $\boldsymbol{\mu} \in \boldsymbol{\mathcal{D}}$ supplémentaires intervenant classiquement dans bon nombre de problèmes d'ingénierie. Revenant maintenant au traitement de ces problèmes paramétrés, nous exploitons les techniques non-intrusives développées notamment dans [Heyberger *et al.*, 2013, Relun *et al.*, 2013]. Nous avons implémenté la version la plus rustique à savoir que l'espace paramétrique $\boldsymbol{\mathcal{D}}$ est tout d'abord discrétisé selon une grille régulière, puis défini grâce à un processus d'interpolation, subordonné à la série de *snapshots* uniformément répartis sur l'espace paramétrique. Ces *snapshots*, associés à une certaine valeur des paramètres, sont calculés de proche en proche par application de la méthodologie LATIN-PGD faiblement intrusive développée jusqu'ici. Ceci ne s'avère guère coûteux dans la mesure où l'algorithme LATIN-PGD possède la propriété remarquable de pouvoir être initialisé à partir de n'importe quelle solution précédemment calculée. De manière plus précise, supposons que l'on cherche à calculer un nouveau point $(\boldsymbol{\mu} + \Delta\boldsymbol{\mu}) \in \boldsymbol{\mathcal{D}}$ de l'espace paramétrique. Connaissant la solution spatio-temporelle $\boldsymbol{s}^{(\boldsymbol{\mu})}$ pour le jeu de paramètres $\boldsymbol{\mu} \in \boldsymbol{\mathcal{D}}$ précédent, ainsi que les modes spatiaux (8.1) correspondants :

$$\left\{\boldsymbol{\Lambda}_i\right\}_{1 \leqslant i \leqslant m^{(\boldsymbol{\mu})}} \in \boldsymbol{\mathcal{V}} \tag{8.1}$$

alors la première itération de la méthode LATIN-PGD se déroule comme suit :

- **Étape locale** : étant donné $\boldsymbol{s}_0^{(\boldsymbol{\mu}+\Delta\boldsymbol{\mu})}(t)$, nous cherchons $\widehat{\boldsymbol{s}}_0^{(\boldsymbol{\mu}+\Delta\boldsymbol{\mu})}(t)$ tel que :

$$\mathbf{u}_0^{(\boldsymbol{\mu}+\Delta\boldsymbol{\mu})}(t) = \mathbf{u}^{(\boldsymbol{\mu})}(t) \qquad \forall t \in I \tag{8.2}$$

- **Étape linéaire** : étant donné $\widehat{\boldsymbol{s}}_0^{(\boldsymbol{\mu}+\Delta\boldsymbol{\mu})}(t)$, nous cherchons $\boldsymbol{s}_1^{(\boldsymbol{\mu}+\Delta\boldsymbol{\mu})}(t)$ où de l'étape préliminaire de mise à jour des fonctions temporelles, on obtient :

$$\mathbf{u}_1^{(\boldsymbol{\mu}+\Delta\boldsymbol{\mu})}(t) = \sum_{i=1}^{m^{(\boldsymbol{\mu})}} \lambda_i^{(\boldsymbol{\mu}+\Delta\boldsymbol{\mu})}(t)\boldsymbol{\Lambda}_i + \mathbf{u}_0^{(\boldsymbol{\mu})}(t) \tag{8.3}$$

puis l'algorithme continue avec $\ell \geqslant 1$ comme détaillé précédemment. Pour de nombreuses valeurs de paramètres, il n'est pas nécessaire d'ajouter de nouveaux modes à la base réduite. Après quelques itérations, on atteint la convergence – pilotée par certains indicateurs d'erreur – et l'algorithme s'arrête pour entamer la gestion du point suivant dans l'espace paramétrique $\boldsymbol{\mathcal{D}}$, ce qui représente un gage de performance. Ainsi, c'est l'interface qui joue pleinement son rôle dans la gestion des multiples reprises (*restart*) en permettant de faire le lien avec la base de données déjà constituée.

REMARQUE 8.2 : Lorsque l'espace paramétrique $\boldsymbol{\mathcal{D}}$ fait intervenir des variations au sein des conditions aux limites, et plus spécifiquement au niveau des conditions de *Dirichlet*, il n'est alors plus possible de réexploiter directement l'initialisation cinématiquement admissible $\boldsymbol{s}_0^{(\boldsymbol{\mu})} \in \mathbf{A}_d^{(\boldsymbol{\mu})}$ issue de la solution $\boldsymbol{s}^{(\boldsymbol{\mu})}$ précédente. Toutefois, ayant accès au logiciel industriel, nous pouvons aisément recalculer une initialisation admissible le cas échéant.

8.3 Quelques mots autour de l'optimisation

On rappelle que la robustesse, l'ergonomie ou l'expérience utilisateur et la performance sont les trois piliers fondamentaux qui guident l'élaboration de notre approche. Un effort d'optimisation, à différentes échelles, a ainsi été fourni pour ériger une approche la plus efficace possible. On précise dans la suite quelques points spécifiques qui ont fait l'objet d'une attention particulière, allant d'une vision macroscopique – haut niveau – à une vision plus microscopique – bas niveau, proche de l'architecture machine.

8.3.1 Vision macroscopique – paramètres & stockage mémoire

L'un des premiers leviers pour agir sur les performances consiste à trouver la meilleure adéquation possible entre les paramètres de la méthode et le problème traité. Dans cette optique, le choix de la direction de recherche joue un rôle fondamental comme nous l'avons déjà évoqué dans les chapitres précédents. Sans volonté d'exhaustivité, on peut par exemple citer le nombre d'itérations maximum dans l'algorithme du point fixe, le coefficient de relaxation ou encore les différents critères mis en œuvre en tant qu'autres paramètres numériques susceptibles d'affecter directement le taux de convergence de la méthode. Il convient toutefois de garder à l'esprit que, conjuguée à l'optimisation, la méthodologie développée se doit d'être accessible à un utilisateur non-expert et robuste – pour adresser une large variété de cas d'application. Ainsi, autant que possible, on doit disposer de valeurs par défaut *universelles* pour ces paramètres numériques. De plus, le nombre de ces paramètres accessibles à l'utilisateur doit être le plus réduit possible.

Un second point capital sur les performances concerne le stockage et l'optimisation de la mémoire. De manière analogue au solveur MUMPS, deux choix sont possibles : soit on décide de garder toute l'information en mémoire vive (RAM), soit on peut ponctuellement réaliser des écritures et lectures sur disques durs, ce qui permet de limiter l'empreinte mémoire – au détriment cependant des performances. Deux versions ont ainsi été développées : une version *in-core* plus performante, et une version *out-of-core* plus économe en ressources. Un point délicat concerne le stockage des différentes quantités et grandeurs généralisées dans la mesure où des approximations spatio-temporelles complètes interviennent à chaque itération de la méthode LATIN-PGD. Le logiciel industriel étant conçu historiquement pour travailler avec des tableaux de taille n et non pas de taille $(n \times n_t)$ – et encore moins avec des formats réduits –, c'est

encore une fois l'interface qui prend la charge de stocker l'information au bon format pour la délivrer le moment opportun au bon endroit. L'utilisation conjointe d'espaces mémoires préexistants et de quelques allocations dynamiques parcimonieuses au sein de l'interface ont permis de mener à bien la méthodologie, en tirant partie du fait que la solution s'écrit sous forme à variables séparées.

8.3.2 Vision mésoscopique – gestion du parallélisme hybride MPI-OpenMP

Un second volet de l'optimisation s'est porté sur la parallélisation du code. L'idée a été d'étendre l'architecture parallèle hybride MPI-OpenMP déjà existante nativement dans Simcenter Samcef™ à l'ensemble de l'interface et de l'étape globale. Choisir d'effectuer un calcul parallèle à mémoire distribuée MPI consiste à lancer simultanément plusieurs instances du logiciel industriel Simcenter Samcef™ (cf. Figure 8.2). Chacune de ces instances, ou processus, parcourt en parallèle le même algorithme mais ne traite qu'une partie des éléments – soit en séquentiel, soit en parallèle si plusieurs *threads* OpenMP sont mobilisés. Ensuite, des communications MPI sont nécessaires à certains moments pour distribuer ou rassembler l'information entre les différents processus. En particulier, certaines quantités sont conservées spécifiquement sur leur instance respective (matrices *sparse*) – chaque instance ne connait qu'une partie de la matrice –, tandis que d'autres quantités sont systématiquement partagées (modes PGD) sur l'ensemble des instances – chaque instance connait la même information.

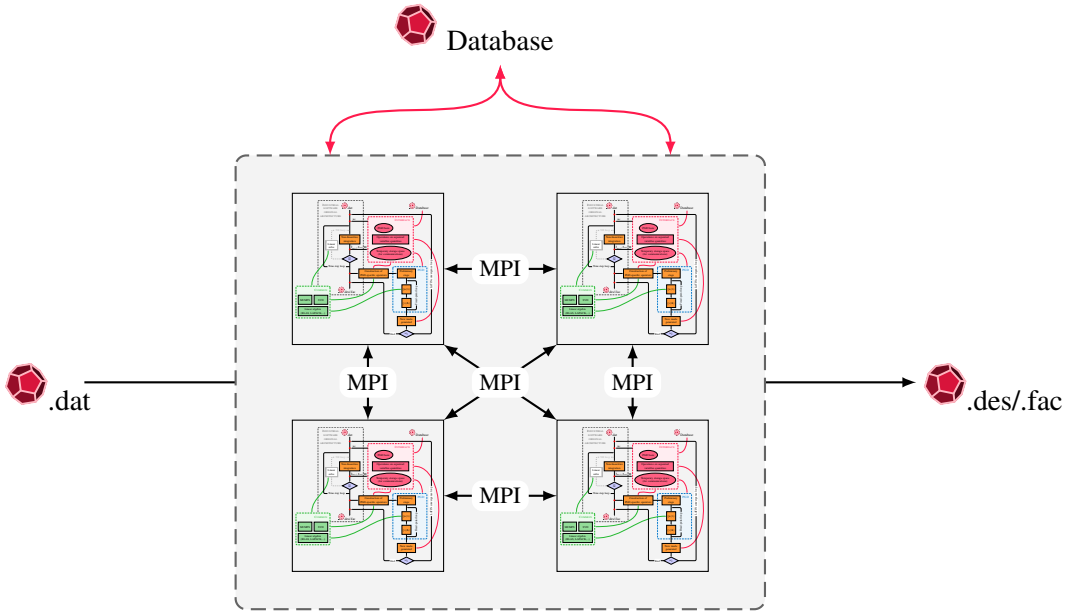


FIGURE 8.2 – Principe de l'architecture MPI parallèle classique

REMARQUE 8.3 : La mémoire étant distribuée entre les différentes instances, le calcul peut être réalisé indépendamment sur une même machine ou sur plusieurs. L'unique différence réside dans la nature des communications MPI où l'information transite soit en interne soit par le réseau via protocole *ssh* – InfiniBand.

8.3.3 Vision microscopique – code source & compilateur

Le troisième point d'intérêt se concentre sur le code lui-même dont l'écriture affecte directement les performances. En particulier, certaines fonctions élémentaires peuvent être appelées de très nombreuses fois. Une légère perte d'efficacité au niveau de ces appels répétés peut alors avoir un impact considérable

sur les performances globales de l'algorithme. À titre d'exemple, on donne ci-dessous quelques lignes en Fortran permettant de réaliser une multiplication matrice-vecteur de trois manières différentes :

- en utilisant la fonction intrinsèque Fortran prévue à cet effet ;
- en écrivant explicitement les boucles avec la notation d'Einstein $y_i = A_{ij}x_j$;
- en exploitant la bibliothèque BLAS de la librairie Intel® MKL.

S'ajoute ensuite la surcouche du compilateur où différentes options d'optimisation sont disponibles, notamment pour la gestion des boucles. Le Tableau 8.1 fournit les temps d'exécution associés à chacune des méthodes pour différentes tailles de tableaux, ainsi que le nombre moyen d'opérations en virgule flottante effectuées par seconde (Gflops).

```
! Déclaration des variables
INTEGER :: I,J,L
DOUBLE PRECISION :: A(L,L),X(L),Y(L)
! Méthode 1 : fonction intrinsèque
Y = MATMUL(A,X)
! Méthode 2 : notation d'Einstein
DO J = 1,L
   DO I = 1,L
      Y(I) = Y(I) + A(I,J)*X(J)
   ENDDO
ENDDO
! Méthode 3 : bibliothèque Intel MKL
CALL DGEMV('N',L,L,1.D00,A,L,X,1,0.D00,Y,1)
```

TABLE 8.1 – Temps d'exécution associés à chacune des méthodes

		Méthode 1	**Méthode 2**	**Méthode 3**
Temps [s]	$\mathtt{L} = 5 \times 10^4$	1.284	1.274	0.874
	$\mathtt{L} = 1 \times 10^5$	6.045	6.012	3.436
	$\mathtt{L} = 2 \times 10^5$	25.012	25.385	13.421
Gflops moyen	-	3.467	3.468	5.834

Ceci reste bien entendu un exemple très simple, à visée purement didactique. Il permet toutefois d'entrevoir l'immense avantage de pouvoir s'appuyer sur un logiciel industriel dont l'ensemble des opérations classiques ou élémentaires sont déjà optimisées depuis de très longues années.

Chapitre 9

Cas-tests industriels

Ce chapitre illustre le fonctionnement de la méthode LATIN-PGD faiblement intrusive au sein du logiciel Simcenter Samcef™ sur des cas-tests industriels. On présente des cas d'application variés qui mettent en lumière la robustesse et les performances de notre approche dans la conduite de problèmes paramétriques non-linéaires dépendant du temps.

Sommaire

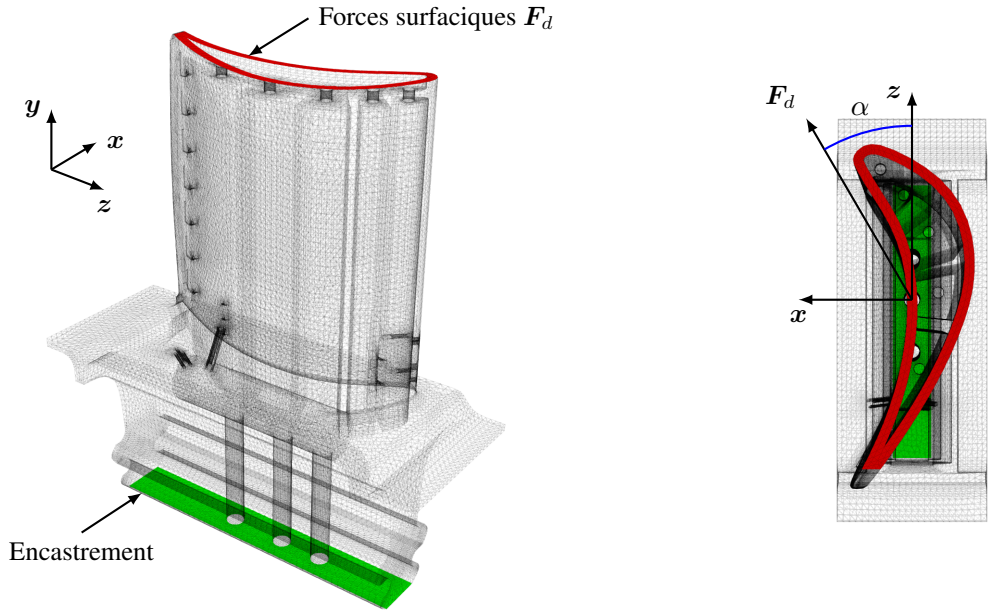
Une des nouveautés de ces travaux réside dans l'intégration de la méthode LATIN-PGD faiblement intrusive dans un logiciel industriel (Simcenter Samcef™), fournissant une solution de bout en bout efficace et robuste pour traiter un large panel d'applications industrielles avec des non-linéarités de différentes natures. À titre d'exemple, on fournit dans ce chapitre deux illustrations portant sur l'étude de solutions non-linéaires de problèmes multi-paramétriques.

9.1 Première illustration industrielle : aube de réacteur d'avion

L'ensemble des calculs ont été effectués sur un cluster de machines linux à architecture parallèle MPI multi-nœuds. Chaque nœud de calcul possède 24 cœurs et dispose de 192 GB de mémoire vive (RAM) – processeur Intel® Xeon® Gold 6126 CPU @ 2.60 GHz (19.25 Mo de mémoire cache L3). Parmi l'ensemble de ces ressources disponibles, 12 cœurs ont été utilisés pour cette application, parfois répartis sur 2 nœuds de calcul lorsque la mémoire disponible sur un seul nœud n'était pas suffisante. Le logiciel industriel Simcenter Samcef™ permet de générer les solutions soit à partir de la méthode classique, incrémentale, de Newton-Raphson, soit en employant la méthodologie LATIN-PGD faiblement intrusive mise en œuvre dans le cadre de ces travaux. Qui plus est, la version commerciale du logiciel Simcenter Samcef™ joue le rôle de référence pour effectuer les comparaisons. Les phases de pré et post-traitements sont réalisées par l'intermédiaire de l'interface Simcenter 3D™ ce qui permet de clore la boucle : la construction de modèles réduits peut s'effectuer via une solution bout en bout en environnement industriel.

9.1.1 Description du cas-test

Le premier cas d'illustration se concentre sur l'étude d'une aube d'un étage haute pression du moteur M88 équipant le Rafale dont la géométrie a été standardisée pour des questions de confidentialité. L'ensemble de la mise en données du problème est réalisée via le logiciel Simcenter 3D™. Le maillage, composé d'éléments tétraédriques de second degré, comptabilise $n = 5 \times 10^6$ degrés de liberté. L'aube en rotation autour de son axe subit des efforts centrifuges sur l'ensemble de sa structure, ainsi que des efforts surfaciques localisés sur la partie supérieure et paramétrés par un angle α comme représenté Figure 9.1. Le disque n'étant pas modélisé dans le cadre de cette étude, on suppose que l'aube est encastrée à sa base. Le matériau utilisé répond aux lois de Chaboche élasto-visco-plastiques explicitées au Chapitre 1 où les coefficients matériaux présentent une forte dépendance à la température. Une étude sur l'espace paramétrique $\boldsymbol{\mathcal{D}} = \mathcal{D}_\alpha \times \mathcal{D}_T$ à deux dimensions est réalisée avec $\alpha \in \mathcal{D}_\alpha \equiv [45, 170]$ (en degrés) l'angle de chargement et $T \in \mathcal{D}_T \equiv [850, 1010]$ la température (en degrés Celsius) supposée uniforme pour simplifier. On peut noter en particulier que ces deux paramètres influencent chacune des variétés $\boldsymbol{\Gamma}$ et $\mathbf{A}_d$ inhérentes à la méthode LATIN-PGD : chaque nouvelle valeur de la température T influe sur les 9 paramètres de la loi matériau donc affecte principalement $\boldsymbol{\Gamma}$ tandis que chaque nouvelle valeur de l'angle α impacte directement les efforts externes intervenant dans $\mathbf{A}_d$.

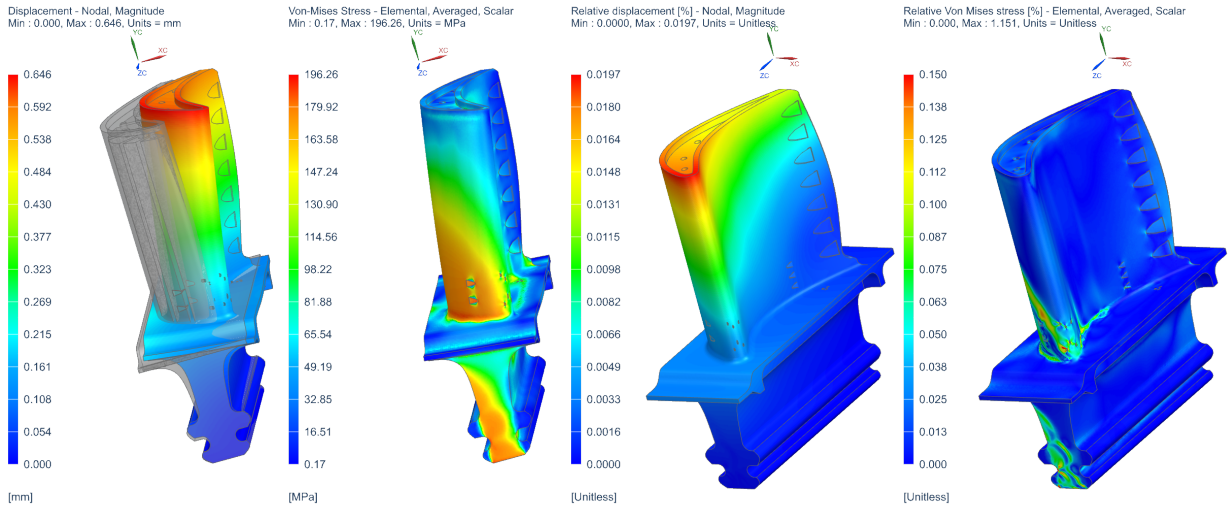


(a) Vue 3D (b) Vue suivant l'axe y

FIGURE 9.1 – Conditions aux limites appliquées à l'aube de réacteur d'avion

9.1.2 Analyse fine d'une résolution

La Figure 9.2 montre les résultats obtenus pour un point donné ($\alpha = 155$ ° et $T = 890$ °C) dans l'espace paramétrique $\boldsymbol{\mathcal{D}}$. De manière plus précise, les sous-figures 9.2a et 9.2b sont consacrées respectivement aux champs de déplacement, et de contrainte de Von Mises, résultants de l'application de la méthode LATIN-PGD faiblement intrusive. Des zones de concentration de contraintes apparaissent naturellement au niveau des connexions entre la partie supérieure et la partie inférieure de l'aube, ainsi qu'autour des différents trous de ventilation. Les sous-figures 9.2c et 9.2d donnent les cartes d'erreur correspondantes par comparaison avec la référence. Afin de mieux apprécier les éventuelles erreurs commises, ces cartes d'erreur sont tracées en quantités relatives. On note une bonne concordance des solutions haute-fidélités avec une erreur relative inférieure à 1% pour seulement 8 modes PGD composant la base réduite.



(a) Champ de déplacements (b) Champ de contraintes de Von Mises (c) Déplacements relatifs (d) Contraintes de Von Mises relatives

FIGURE 9.2 – Résultats obtenus avec la méthode LATIN-PGD faiblement intrusive dans Simcenter Samcef™ : (a) champ de déplacements et (b) champ de contraintes ; avec cartes d'erreur correspondantes par rapport à la référence : (c) déplacements relatifs et (d) contraintes relatives (en pourcents [%])

Optimisation de la direction de recherche. L'opérateur de la direction de recherche $\mathbf{H}$ est le paramètre principal de la méthode. En tant que paramètre, plusieurs choix sont possibles qui, comme nous l'avons déjà évoqué, affectent directement les performances et le taux de convergence de la méthode. Deux cas extrêmes peuvent être distingués. Le premier consiste à considérer l'opérateur tangent $\mathbf{H}_\ell(t)$ conduisant à un taux de convergence optimal par rapport au nombre d'itérations. Malheureusement, comme cet opérateur dépend du temps t, il peut conduire à un coût élevé de chaque itération. Le second cas extrême réside dans l'utilisation de la rigidité élastique $\mathbf{K}$ qui possède deux propriétés intéressantes : cet opérateur est indépendant du temps et ne varie pas au cours des itérations, ce qui induit des itérations à plus faible coût, au détriment cependant du taux de convergence. Au Chapitre-5, on avait déjà noté ce point mais le cas académique présenté n'avait pas permis d'assoir la supériorité de l'emploi d'une direction de recherche par rapport à une autre. Ce cas industriel apporte un nouvel éclairage à ce sujet comme l'illustre le Tableau 9.1 exposant le nombre total d'itérations et le temps de calcul pour atteindre un même niveau de précision.

De manière plus générale, l'ensemble des cas-tests industriels qui ont été testés dans le cadre des non-linéarités matériaux, avec éventuellement la présence de contact, donnent un avantage indéniable à la rigidité élastique. Seul le cas des grandes transformations – sous l'hypothèse de déplacements modérés – met en défaut cette direction de recherche élastique en nécessitant une direction plus élaborée comme évoqué au Chapitre-7. La Figure 9.3 fournit plus précisément le détail des temps de calculs pour chacune des étapes locale et globale en considérant pour direction de recherche l'opérateur d'élasticité $\mathbf{K}$. On observe notamment la propension de l'étape locale à devenir particulièrement gourmande en ressources – occupant le principal centre de coûts de calculs – lorsque l'on traite des cas représentatifs d'une complexité industrielle. L'étape locale étant aisément parallélisable en espace – sur l'ensemble des éléments et des points de Gauss –, au premier abord l'augmentation des ressources informatiques – nombre de cœurs – permettrait de circonscrire efficacement le poids associé à cette étape locale. Toutefois, cette assertion théorique reste à nuancer à la lumière des courbes qui seront présentées en Section-9.1.4 : la parallélisation n'est que rarement parfaite. Ceci est dû au fait que les calculs préliminaires et les opérations d'archivage de l'étape locale ne sont pas hautement parallélisables – nécessitant des écritures et lectures disque –, alors que, dans le même temps, ces opérations représentent plus de 20% des temps de calculs. Aussi, dans une

optique de recherche de meilleures performances, ce point mérite une attention particulière qui fera l'objet de certaines perspectives.

TABLE 9.1 – Comparaison des temps de calcul et du nombre d'itérations selon différentes directions de recherche

	Temps de calcul [s]								**Nombre d'itérations**
	Étape locale		*Étape globale*		*Compléments*		*Total*		-
$\mathbf{H}_\ell(t)$	1324	(32.9%)	2373	(59.0%)	326	(8.1%)	4023	(100%)	13
$\mathbf{K}$	2206	(73.9%)	354	(11.9%)	425	(14.2%)	2985	(100%)	26

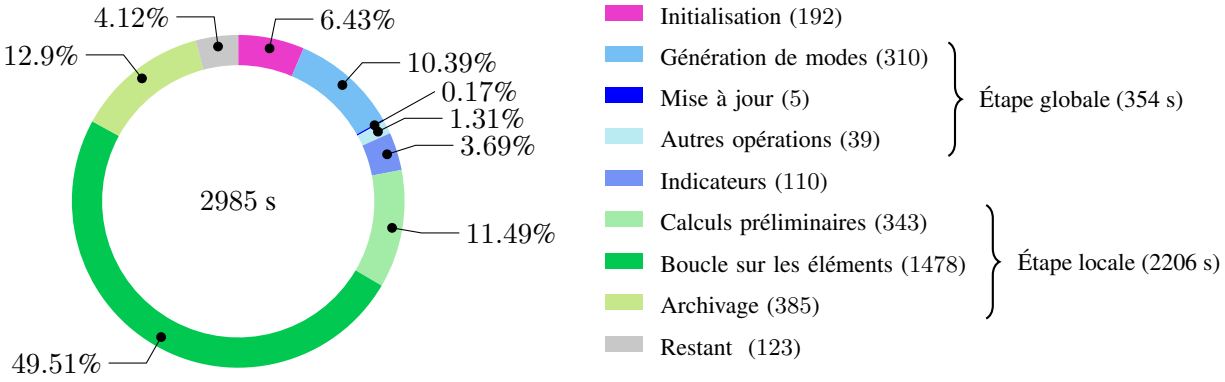


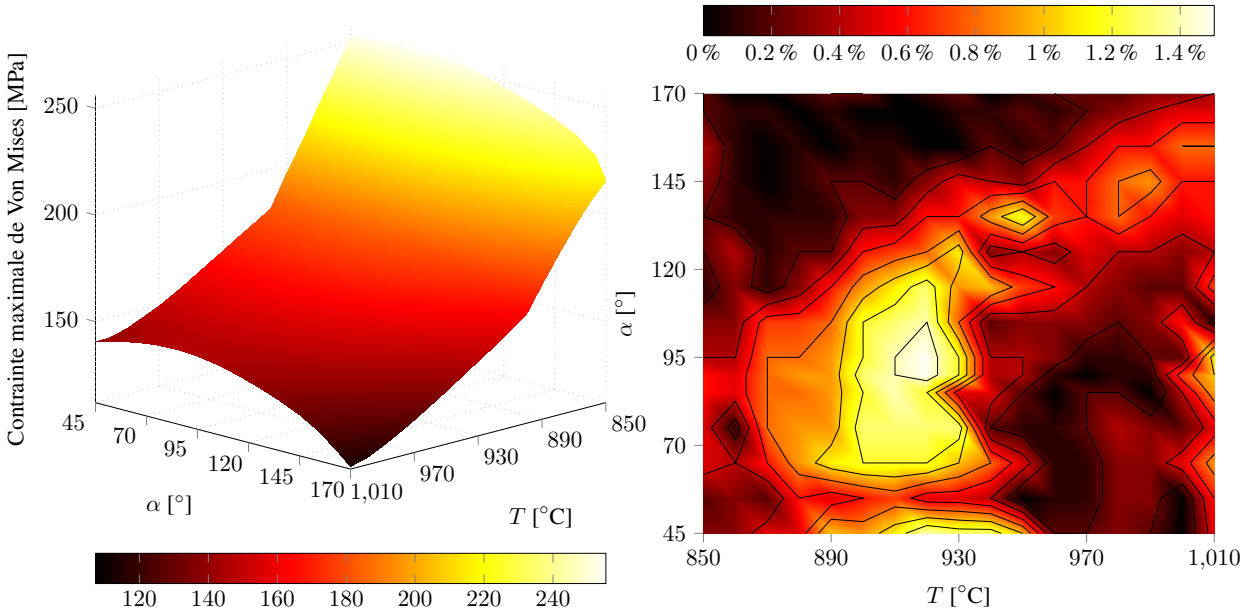
FIGURE 9.3 – Détails des temps de calculs [s]

9.1.3 Résultats de l'étude paramétrique

On présente ci-après les résultats, issus de l'étude paramétrique, qui mettent en lumière d'une part un gain en temps de calculs lors de la génération d'abaques virtuels et, d'autre part, les atouts que procurent un solveur multi-fidélité en environnement industriel. Il convient également de noter que la solution paramétrée obtenue n'est en aucun cas réduite à la simple connaissance d'une quantité d'intérêt : par la méthodologie faiblement intrusive mise en place, la solution spatio-temporelle complète est accessible facilement pour n'importe quelle valeur des paramètres.

Construction du modèle réduit et performance. L'abaque virtuel de la contrainte maximale de Von Mises sur l'ensemble de l'espace paramétrique $\mathcal{D}$ est donné en Figure 9.4. Ces graphiques résultent de plus de 250 calculs haute-fidélités obtenus par un balayage séquentiel et uniforme de l'espace paramétrique. On met en évidence une forte variation non-linéaire en fonction de l'évolution de la température et une variation plus modérée – mais existante – en fonction de l'angle de chargement α (cf. sous-figure 9.4a). De plus, nous avons une excellente approximation de la quantité d'intérêt puisque l'erreur est de l'ordre du pourcent sur l'ensemble de l'espace paramétrique $\mathcal{D}$ (cf. sous-figure 9.4b). Qui plus est, l'aspect performance de notre approche est mis en exergue sur la Figure 9.5 où l'on peut observer un gain d'un facteur 3. Ainsi, en plus d'être capable de générer des modèles réduits dans un environnement industriel, la méthodologie LATIN-PGD faiblement intrusive déployée permet une génération efficace, compatible avec les exigences industrielles. Il convient également de noter que l'accélération obtenue serait vraisemblablement encore plus impressionnante si on avait considéré plus de points ou plus de paramètres. En effet, les deux ingrédients principaux qui permettent d'atteindre ces performances sont d'une part l'encapsulation faiblement intrusive de la méthode PGD spatio-temporelle, et d'autre part, la propriété de la

méthode LATIN-PGD permettant d'initialiser un calcul par n'importe quel point proche dans l'espace paramétrique [Néron *et al.*, 2015]. Bien que non étudiés ici, d'autres développements autour des processus d'optimisation par utilisation de méta-modèles [Laurent *et al.*, 2013, Courrier *et al.*, 2014, Nachar *et al.*, 2020] permettraient d'accroître encore plus les performances. Les logiciels d'exploration d'espaces paramétriques tels que HEEDS™ sont déjà compatibles avec la plateforme Simcenter 3D™, et donc de fait également compatibles avec la méthodologie LATIN-PGD proposée. Cette fonctionnalité de couplage n'a toutefois pas été approfondie et mériterait de l'être pour de futurs travaux.



(a) Surface de réponse de la contrainte maximale de Von Mises sur l'espace paramétrique $\mathcal{D}$ obtenue avec la méthode LATIN-PGD faiblement intrusive

(b) Carte d'erreur de la contrainte maximale de Von Mises sur l'espace paramétrique $\mathcal{D}$ par rapport à la référence [%]

FIGURE 9.4 – Abaque virtuel de la contrainte maximale de Von Mises sur l'ensemble de l'espace paramétrique $\mathcal{D}$ résultant de 255 points tirés uniformément dans l'espace paramétrique

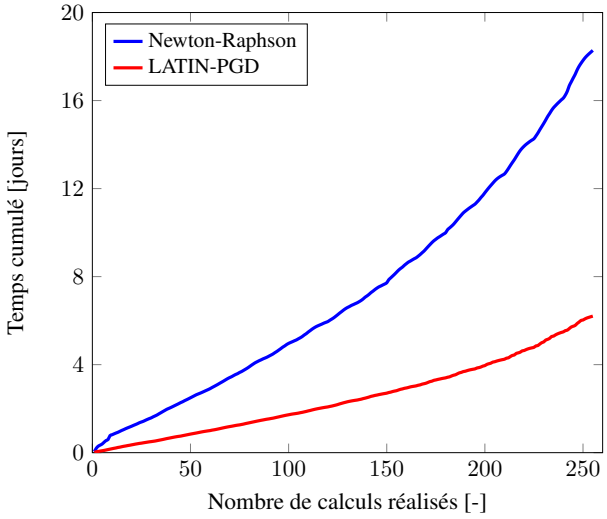
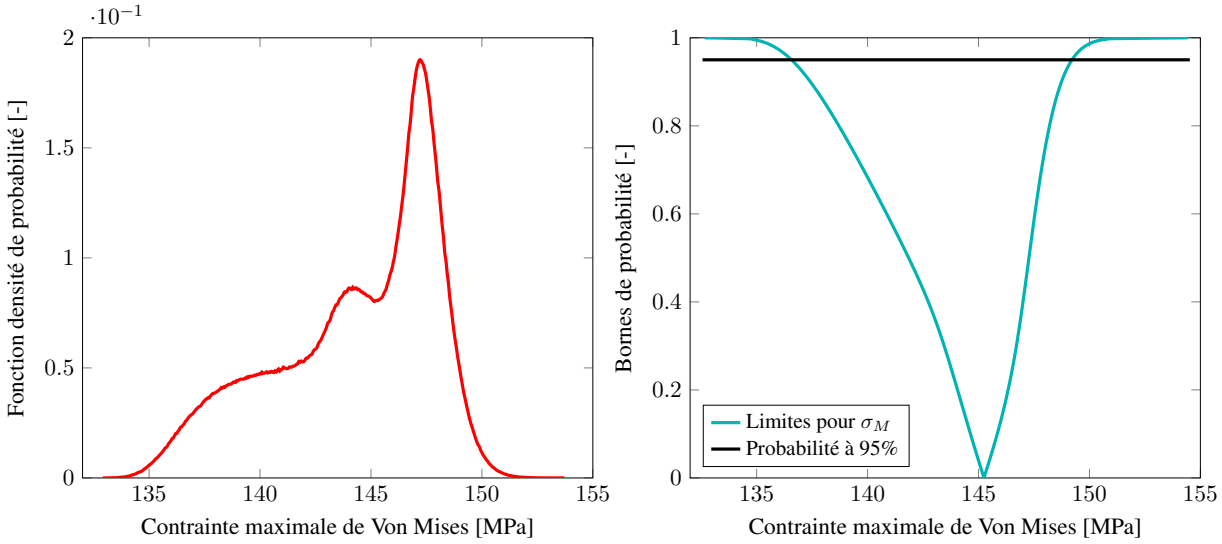


FIGURE 9.5 – Temps cumulé nécessaire pour la génération du modèle réduit en suivant soit l'utilisation du logiciel commercial dans sa version standard, soit la méthodologie LATIN-PGD faiblement intrusive développée

Étude probabiliste en temps réel. Une fois de tels abaques virtuels disponibles, il est alors aisé de réaliser des études stochastiques en temps réel (cf. Figure 9.6). Par exemple, en supposant une distribution de probabilité Gaussienne pour la température $T \sim \mathcal{N}(T_m, \sigma^2)$ (de moyenne T_m et d'écart-type σ) et une loi de densité de probabilité uniforme pour le second paramètre α, il est alors possible de réaliser plus de dix millions de tirages de Monte-Carlo en moins d'une seconde. La sous-figure 9.6a représente la fonction densité de probabilité résultante pour la contrainte maximale de Von Mises avec $T \sim \mathcal{N}(990, 25)$ et α uniformément distribué sur l'intervalle $[75, 140]$ (en degrés). Les incertitudes sur les paramètres peuvent donc être facilement appréhendées lors de la conception de nouvelles structures, ou bien même lors de phases de maintenance prédictive – si on imagine que ces données proviennent de différents capteurs reflétant l'état de la structure en fonctionnement. La sous-figure 9.6b fournit les bornes probabilistes associées à la contrainte maximale de Von Mises. Dans cet exemple en particulier, on recense une probabilité de 95% d'avoir $136.61 \leqslant \sigma_M^{95\%} \leqslant 149.19$ MPa.



(a) Fonction densité de probabilité associée à la contrainte maximale de Von Mises

(b) Bornes probabilistes associées à la contrainte maximale de Von Mises σ_M

FIGURE 9.6 – Illustration d'une étude stochastique avec $T \sim \mathcal{N}(990, 25)$ et α uniformément distribué sur l'intervalle $[75, 140]$

Solveur multi-fidélité. Une autre propriété intéressante de la méthode LATIN-PGD réside dans la description globale des principales caractéristiques de la solution dès les premières itérations – solveur non-incrémental. Ainsi, après chaque itération LATIN, on peut décider si la précision est suffisante au regard des qualités recherchées ou s'il est nécessaire de poursuivre la résolution jusqu'à l'obtention d'une solution haute-fidelité. La Figure 9.7 présente l'évolution de l'erreur sur la contrainte de Von Mises – pour une valeur donnée des paramètres au pas de temps le plus chargé – en fonction de ζ_ℓ, l'inverse du gain donné par la formule (9.1) caractérisant le rapport entre le temps Δt_r nécessaire à la résolution classique par le schéma de Newton-Raphson et le temps cumulé Δt_ℓ associé à la méthode LATIN-PGD faiblement intrusive au cours des itérations :

$$\zeta_\ell = \frac{\Delta t_\ell}{\Delta t_r} \tag{9.1}$$

Cela signifie que plus ζ_ℓ est petit, plus la méthode LATIN-PGD est performante. La courbe en pointillés noirs symbolise l'objectif à atteindre : avoir une erreur globale inférieure à 1% par rapport à la référence. La courbe rouge fournit l'erreur moyenne sur l'ensemble de l'espace Ω et les deux courbes bleues délimitent quant à elles les bornes d'erreur en dessous desquelles respectivement 95% ou 99% des éléments sont inclus. Enfin, les petites illustrations 3D révèlent la distribution spatiale de l'erreur en pourcentage

par rapport à la référence selon l'échelle de couleur qui figure à droite. Ainsi, on peut voir qu'en moyenne, une bonne approximation de la solution est rapidement atteinte puisque la courbe rouge croise la courbe en pointillés noirs autour de $\zeta_\ell = 0.2$ sur l'axe des abscisses, ce qui se traduit par un gain d'un facteur 5 en temps de calculs. En d'autres termes, les premiers modes PGD capturent les principales tendances de la solution et peuvent être considérés comme des modes globaux. Ensuite, si on a besoin d'obtenir une approximation plus fine de la solution – dans les zones spatiales encore trop mal représentées –, des modes supplémentaires peuvent être ajoutés pour diminuer l'erreur. Ces nouveaux modes ont un effet de plus en plus localisé au fur et à meure que la base réduite s'enrichit, et permettent ainsi une décroissance de l'erreur pour les quelques éléments qui ne satisfont pas encore les exigences souhaitées (courbes bleues). Par exemple, on peut noter que pour $\zeta_\ell = 0.4$ alors 95% des éléments ont une erreur inférieure à 1%. Disposer d'un solveur multi-fidélité peut s'avérer être un véritable atout en bureaux d'études pour le suivi en temps réel des simulations directement dans l'interface Simcenter 3D™ et ainsi ne pas laisser une simulation se poursuivre inutilement. À cela s'ajoute une tolérance accrue aux avaries informatiques : suite à une interruption (volontaire ou involontaire) du calcul, une approximation spatio-temporelle complète – issue de la dernière itération achevée – est connue grâce à la structure non-incrémentale de la méthode LATIN-PGD. Cette approximation peut *a posteriori* être analysée, voire affinée, et servir de base à un calcul ultérieur. *A contrario* avec la méthode de Newton-Raphson, seul le dernier pas de temps archivé serait accessible ce qui rend l'analyse plus délicate et nécessite la plupart du temps de relancer le calcul – en repartant éventuellement du dernier pas de temps archivé.

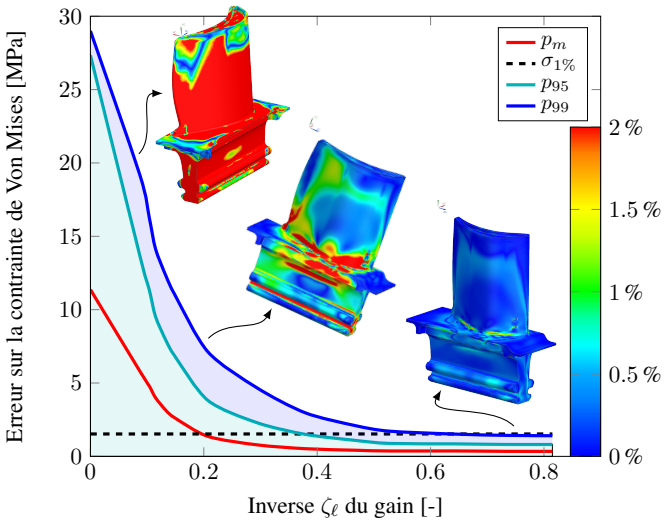


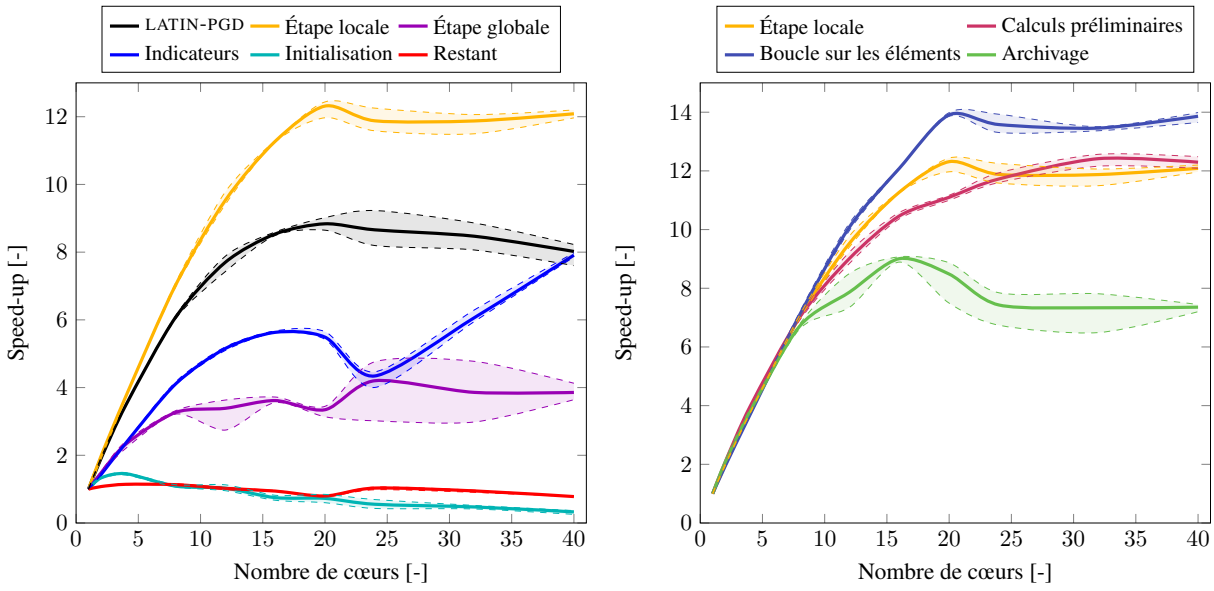
FIGURE 9.7 – Faisceaux de courbes montrant l'évolution de l'erreur sur la contrainte de Von Mises en fonction de l'inverse ζ_ℓ du gain au cours des itérations LATIN-PGD avec illustrations 3D de la répartition spatiale de l'erreur en certaines itérations

9.1.4 Compléments

On donne ici quelques compléments quant aux caractéristiques de la méthode LATIN-PGD faiblement intrusive incorporée nativement dans Simcenter Samcef™.

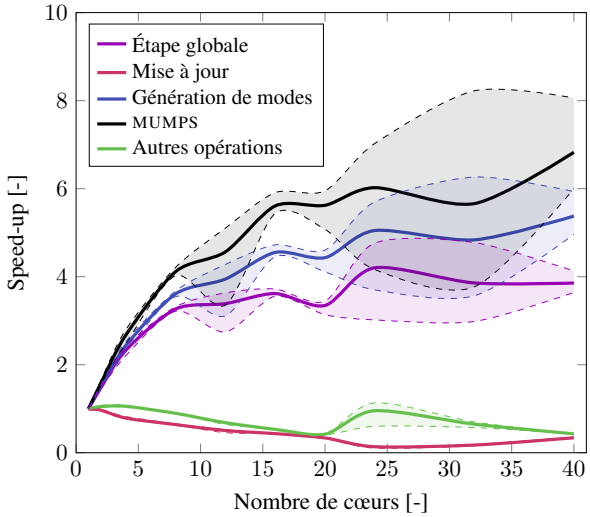
Étude du parallélisme. On mène une analyse plus approfondie des performances de la méthode déployée au regard des ressources *hardware* mobilisées – évolution du *speed-up* en fonction du nombre de cœurs (cf. Figure 9.8). L'ensemble de l'étude a été réalisée sur un unique nœud de calcul doté d'un processeur Intel® Xeon® Gold 6242R CPU @ 3.10 GHz (35.75 Mo de mémoire cache L3) possédant 40 cœurs et disposant de 754 GB de mémoire vive (RAM). Pour s'affranchir au mieux des différents artefacts indé-

pendants de la méthode LATIN-PGD proprement dite, chacun des calculs ont été exécutés suivant 5 occurrences. Pour chacune des courbes sont ainsi tracées la moyenne et la variabilité associée (sous-figures 9.8a à 9.8d). Globalement, on observe deux tendances : une nette amélioration du *speed-up* jusqu'à 20 cœurs suivie d'une phase de stagnation – couplée à une plus forte variabilité. Ceci est majoritairement dû à l'architecture informatique utilisée, le processeur exploitant la technologie Intel® Hyper-Threading.

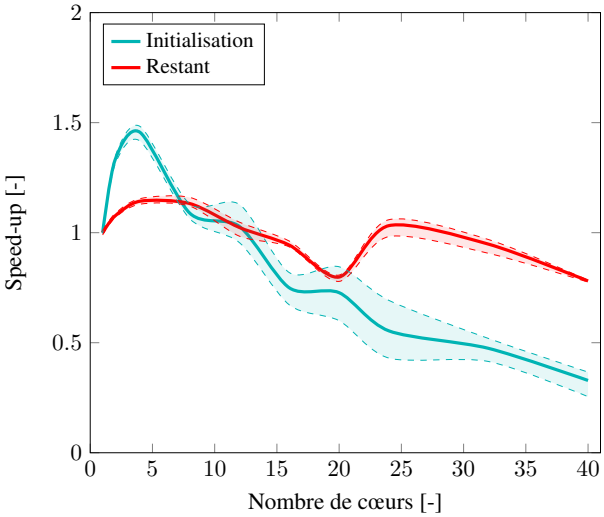


(a) Évolution du *speed-up* des grandes étapes de la méthode LATIN-PGD faiblement intrusive

(b) Évolution du *speed-up* des différentes composantes de l'étape locale



(c) Évolution du *speed-up* des différentes composantes de l'étape globale



(d) Évolution du *speed-up* des routines complémentaires

FIGURE 9.8 – Évolution du *speed-up* en fonction du nombre de cœurs utilisés sur un unique nœud de calcul doté d'un processeur Intel® Xeon® Gold 6242R CPU @ 3.10 GHz (35.75 Mo de mémoire cache L3) possédant 40 cœurs et disposant de 754 GB de mémoire vive (RAM)

Dans la suite, on s'intéresse donc plus spécifiquement à la première moitié des différentes courbes où un *speed-up* aux environs de 9 est constaté – par rapport au calcul séquentiel – pour la méthode LATIN-PGD avec 20 cœurs. Ces performances globales s'expliquent par l'agrégation des différents *speed-up* des différentes grandes étapes composant la méthode – au *prorata* des contributions respectives :

- l'étape locale, fortement parallélisable et, représentant la majeure partie du temps de calcul (cf. sous-figure 9.8b) ;
- l'étape globale, moins avantageusement parallélisable et, représentant le deuxième poste le plus coûteux en temps de calcul (cf. sous-figure 9.8c) ;
- l'initialisation et l'ensemble des autres opérations non répertoriées, excessivement peu parallélisables (cf. sous-figure 9.8d).

Plus précisément, l'étape locale souffre d'une opération d'archivage présentant un *speed-up* bien plus faible que celui apparenté à la boucle sur les éléments, ce qui pénalise le *speed-up* général de l'étape locale. Ce phénomène s'amplifie à mesure que le nombre de cœurs augmente : chaque processus MPI, associé à chaque cœur, demande simultanément des écritures et lectures sur un même et unique disque dur. Ceci peut engendrer un certain temps de latence piloté par la vitesse maximale d'échange d'informations avec le disque. Au niveau de l'étape globale maintenant, on constate que les diverses opérations autres que MUMPS ont tendance à entraver les performances globales. Une optimisation du code Fortran permettrait sans doute d'améliorer davantage le *speed-up* de l'étape globale – en tirant au maximum profit de la technologie Intel® AVX-512 incluse dans la librairie Intel® MKL par exemple, pour mener les diverses opérations sur les vecteurs, matrices, tenseurs, etc. Pour finir, l'initialisation inhibe également, dans une certaine mesure, le *speed-up* global de l'algorithme, cette phase étant de plus en plus lente à mesure que le nombre de cœurs augmente – du fait principalement de la synchronisation nécessaire entre tous les processus. Ainsi, alors que les quatre phases (dénommées respectivement initialisation, mise à jour, autres opérations et restant) ne représentent qu'à peine 1.5% du temps total en séquentiel, cette contribution grimpe à plus de 20% du temps total lorsque 20 cœurs sont sollicités. Ceci représente un temps substantiel clairement non optimisé puisque le *speed-up* de ces quatre phases est affiché à seulement 0.69 pour 20 cœurs : ne serait-ce que d'avoir un *speed-up* de 2 pour ces opérations permettrait d'accroître le *speed-up* global de la méthode LATIN-PGD de plus de 15%. De cette analyse, on gardera à l'esprit que l'allure de ces courbes est susceptible d'évoluer en fonction de la manière de coder d'une part, et des caractéristiques de la machine utilisée ainsi que son architecture d'autre part.

Jumeaux numériques – Digital Twins. La méthode LATIN-PGD faiblement intrusive, incluse nativement dans le logiciel Simcenter Samcef™, fournit l'ensemble de la solution spatio-temporelle complète pour chacune des valeurs discrètes $\boldsymbol{\mu} \in \boldsymbol{\mathcal{D}}$ dans l'espace paramétrique. Comme développé précédemment, on peut alors avoir accès à une quantité d'intérêt spécifique – contrainte de Von Mises par exemple – pour mener diverses études, esquisser des abaques virtuels, etc. Mais on peut également mettre à profit l'ensemble de l'information disponible en vue de constituer des jumeaux numériques. Par techniques d'interpolation multi-variées ou PGD ou encore HOSVD, il est possible de remonter à une quantité paramétrée globale (9.2) telle que :

$$\mathbf{u}(t, \alpha, T) = \sum_{k=1}^{m} \lambda_k(t)\omega_k(\alpha)\chi_k(T)\boldsymbol{\Lambda}_k \tag{9.2}$$

où dorénavant les (extra-)paramètres α et T deviennent des variables à part entière du champ des déplacements généralisés $\mathbf{u}$ solution. De telles quantités peuvent être visualisées en temps réel au niveau de l'interface graphique du logiciel Simcenter 3D™ et l'agrégation de données issues de capteurs permettrait d'enrichir encore ce modèle, afin de proposer des jumeaux numériques les plus fidèles possibles. Dans le cadre de ces travaux, nous n'avons pas eu l'occasion de travailler sur le volet données mais ceci constitue une voie à suivre dans le futur.

9.2 Seconde illustration industrielle : assemblage aube-disque

L'ensemble des calculs ont été menés sur une machine linux doté d'un processeur Intel® Xeon® CPU E5-2687W v4 @ 3.0 GHz (30 Mo de mémoire cache L3) possédant 24 cœurs et disposant de 252 GB de

mémoire vive (RAM). Une des particularités supplémentaires de ce cas d'illustration par rapport au cas précédent se situe dans la prise en compte des non-linéarités de contact.

9.2.1 Présentation du cas-test

Le second cas d'illustration se focalise sur l'étude d'un assemblage aube-disque d'une turbine à gaz impliquant des non-linéarités matériaux – plasticité et fluage – et des non-linéarités de contact à la jonction entre le disque et le pied d'aube. La structure est soumise à des forces centrifuges ainsi qu'à des forces de pression sur la surface de l'aube. Qui plus est, un gradient de température est appliqué suivant la coordonnée radiale. Au vu du chargement et de la périodicité de la géométrie caractérisant la machine tournante, des conditions de symétrie cyclique sont prescrites sur les parties latérales. Le maillage, composé d'éléments volumiques de différentes natures – tétraédriques ou hexaédriques – et d'éléments spécifiques dédiés aux interfaces entre les différentes parties du maillage – pour écrire les conditions cinématiques correspondantes –, regroupe un million de degrés de liberté parmi lesquels figurent 4×10^4 multiplicateurs de Lagrange. Dans ce qui suit, une étude à deux paramètres est menée où l'espace paramétrique $\boldsymbol{\mathcal{D}} = \mathcal{D}_E \times \mathcal{D}_{\alpha_{th}}$ se compose des valeurs du module d'Young $E \in \mathcal{D}_E \equiv [200, 260]$ MPa et du coefficient de dilatation thermique $\alpha_{th} \in \mathcal{D}_{\alpha_{th}} \equiv [1.38 \times 10^{-5}, 1.42 \times 10^{-5}]$ K^{-1} de l'aube. Liées aux différents processus de fabrication, les valeurs de ces paramètres sont supposées être insuffisamment connues. Une petite variation de la valeur nominale de ces coefficients affecte directement la valeur de la contrainte maximale de Von Mises dans la zone de contact, c'est-à-dire dans la zone d'apparition des phénomènes de fluage sur de longues périodes de temps, ce qui peut endommager la structure plus rapidement que prévu. Bien que d'autres paramètres puissent avoir une influence, nous nous concentrons uniquement sur l'espace paramétrique $\boldsymbol{\mathcal{D}} = \mathcal{D}_E \times \mathcal{D}_{\alpha_{th}}$ dans le cadre de cette étude.

9.2.2 Résultats

Soumis aux différents chargements, l'assemblage aube-disque d'une vingtaine de centimètres se dévrille légèrement. Sur la Figure 9.9, on peut apprécier les configurations initiale et déformée où un déplacement maximal de 3.86 mm est constaté à l'extrémité de la structure au dernier pas de chargement.

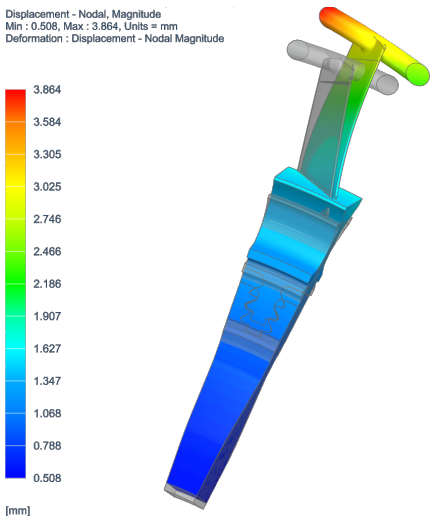


FIGURE 9.9 – Déformée de l'assemblage aube-disque soumis aux différents chargements et comparaison avec la configuration initiale de référence

On fournit également la carte de contrainte de Von Mises obtenue pour un point donné ($E = 220$ MPa et $\alpha_{th} = 14 \times 10^{-6}$ K^{-1}) dans l'espace paramétrique $\boldsymbol{\mathcal{D}}$ (cf. Figure 9.10) avec l'erreur relative par rapport à la référence – issue du logiciel Simcenter Samcef™ dans sa version commerciale. On observe une bonne

concordance des solutions avec une erreur relative inférieure au pourcent dans la zone d'intérêt pour 10 modes PGD composant la base réduite.

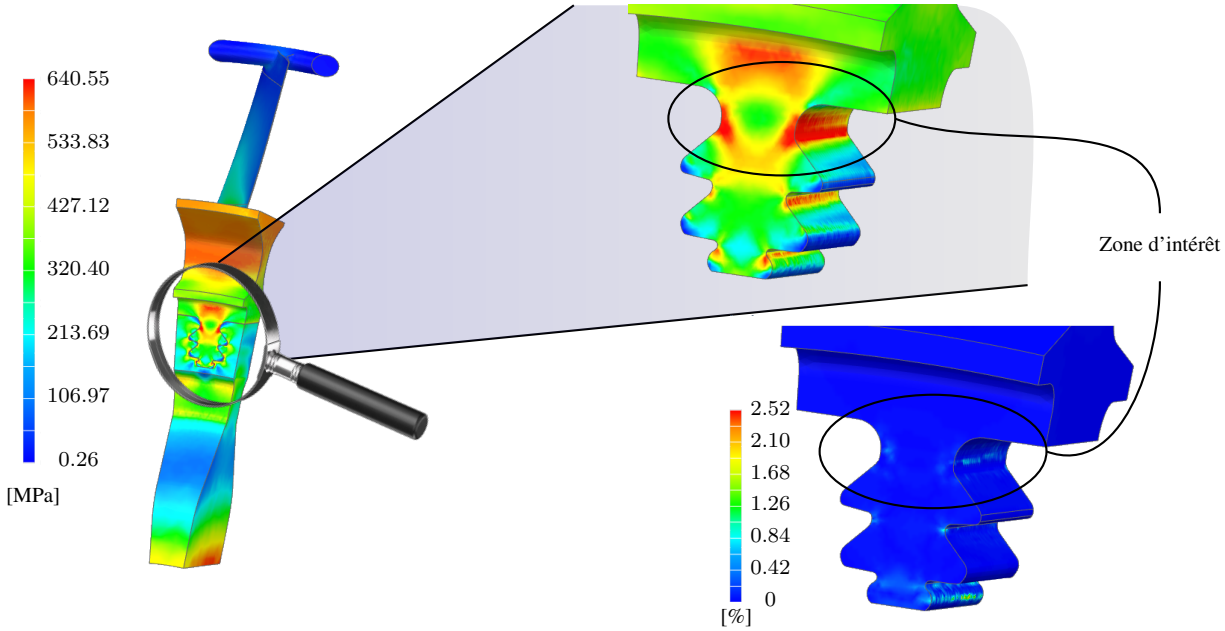


FIGURE 9.10 – Carte du champ des contraintes de Von Mises [MPa] avec zoom sur la zone d'intérêt et comparaison par rapport à la référence [%]

L'abaque virtuel de la surface de réponse de la contrainte maximale de Von Mises sur l'espace paramétrique $\mathcal{D}$ dans la zone de contact est donné Figure 9.11. Ce graphe met en évidence une forte variation non-linéaire par rapport au module d'Young, tandis qu'une tendance linéaire est constatée par rapport au second paramètre α_{th}. En utilisant la même méthodologie que précédemment, cet abaque résulte du calcul de 70 points dans l'espace paramétrique par un balayage séquentiel et uniforme des données.

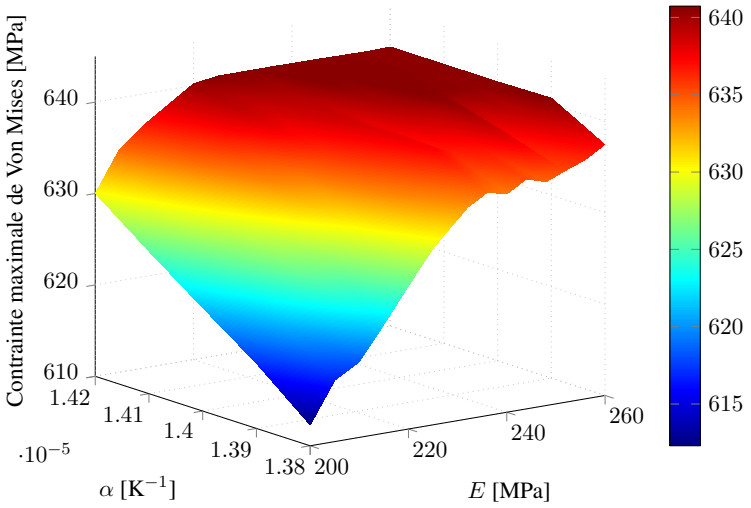


FIGURE 9.11 – Abaque virtuel de la contrainte maximale de Von Mises dans la zone de contact sur l'ensemble de l'espace paramétrique $\mathcal{D}$ résultant de 70 points uniformément répartis dans l'espace paramétrique

Grâce notamment à une réutilisation raisonnée des différents modes PGD générés, ainsi qu'à la structure même de l'algorithme LATIN-PGD – permettant de redémarrer un calcul à partir de n'importe quel point précédemment convergé proche dans l'espace paramétrique –, seulement 3 heures ont été nécessaires pour esquisser cet abaque virtuel. À titre de comparaison, plus de 21 heures auraient été requises avec une utilisation classique du logiciel Simcenter Samcef™ dans sa version commerciale, ce qui conduit à un gain d'un facteur 7 par rapport à cette référence. En outre, ce second cas-test met en évidence l'un des avantages lié au fait d'avoir incorporé nativement – de manière faiblement intrusive – la méthode LATIN-PGD dans un logiciel industriel. L'ensemble des capacités offertes par le logiciel hôte peuvent alors être aisément réexploitées : tous types d'éléments, tous types de lois de comportement, tous types de non-linéarités, etc. Ce cas-test recense par exemple 3 lois de comportement non-linéaires (fonction de la température) et 6 types d'éléments dont des éléments non-linéaires de contact.

⋆ ⋆

⋆

Conclusions & perspectives

Ces travaux portent sur le développement d'une méthodologie faiblement intrusive de la méthode LATIN-PGD destinée à être implémentée nativement dans n'importe quel logiciel industriel par éléments finis généraliste. Plus spécifiquement, c'est au sein du logiciel Simcenter Samcef™ propriété de Siemens Digital Industries Software que la démarche a été mise en œuvre, ce qui a permis d'aboutir à des gains, d'une part en termes de performance – d'un facteur allant de 3 à 7 sur les illustrations proposées –, et d'autre part en termes de robustesse – grâce à une méthodologie générique, compatible avec tous les types de non-linéarités incluses usuellement dans les logiciels industriels.

L'apport de ces travaux s'inscrit pleinement dans la volonté de pouvoir disposer d'outils propres aux méthodes ROM en environnement industriel, impliquant de fait l'utilisation de logiciels commerciaux par des ingénieurs non nécessairement experts de ces méthodes. L'originalité de l'approche tient à la relation privilégiée qui a été établie avec l'éditeur de logiciel Siemens Digital Industries Software, ce qui a donné lieu à cette démarche faiblement intrusive au sens où le nombre de modifications à apporter au logiciel industriel a été minimisé. Cela a nécessité un accès au code source, une excellente maîtrise et connaissance du logiciel industriel, ainsi qu'une poignée de modifications ponctuelles, apportées par l'éditeur de logiciel, pour rendre l'architecture du code Simcenter Samcef™ compatible avec la stratégie LATIN-PGD non-incrémentale. Ce cheminement se différencie des autres approches non-intrusives couramment rencontrées dans la littérature (cf. Section-3.2). En particulier, c'est la toute première fois que la méthode LATIN-PGD est intégrée nativement, et à un aussi haut niveau, dans un logiciel industriel généraliste, laissant espérer une meilleure diffusion des méthodes ROM à l'ensemble de la communauté scientifique à l'avenir, afin que celles-ci fassent partie au quotidien des outils des ingénieurs en bureaux d'études.

Dans le cadre de ces travaux, plusieurs formes de non-linéarités ont été considérées. Partant des non-linéarités matériaux pour définir une première esquisse de notre démarche, nous l'avons ensuite étendue aux non-linéarités géométriques et à la prise en compte des phénonènes de contacts. Nous nous sommes toutefois limités aux cas des déplacements modérés, en dehors des zones d'instabilité. À chaque fois, nous nous sommes appuyés sur l'ensemble des fonctionnalités existantes du logiciel industriel – pour la réalisation de l'étape locale notamment –, avant de compléter la formulation par les ingrédients propres à la méthode LATIN-PGD. Le fer de lance de la méthodologie proposée repose en effet sur sa capacité à pouvoir s'appuyer sur les multiples outils, développés et optimisés depuis de longues années, qui font la force des logiciels industriels dans la variété des cas qu'ils sont susceptibles d'adresser : tous types d'éléments, tous types de chargements, tous types de non-linéarités, tous types de lois de comportement, etc. Ceci représente une véritable opportunité permettant de concentrer les efforts sur de nouveaux développements, plutôt que de recoder des parties de codes non-linéaires complexes sans réelle valeur ajoutée par rapport à l'existant. Toutefois, cela a impliqué de devoir imaginer, concevoir, mettre en œuvre et optimiser les nouvelles fonctionnalités propres aux méthodes ROM, et plus spécifiquement propres à la méthode LATIN-PGD. Ces travaux ont donc donné naissance à de nombreux développements théoriques et logiciels : réflexion et mise en place de l'architecture LATIN, création d'une interface faisant le lien entre les étapes locales et globales, élaboration des outils et opérateurs propres à la stratégie PGD, création d'un module de post-traitement permettant de visualiser directement les modes générés sous l'interface gra-

phique Simcenter 3D™, etc.

Qui plus est, grâce à cette démarche, l'ensemble des améliorations à venir apportées au solveur natif Simcenter Samcef™ bénéficieront également à la nouvelle méthodologie LATIN-PGD déployée – comme par exemple l'accélération de la résolution de systèmes linéaires (que ce soit par MUMPS ou par méthodes itératives) ou encore l'ajout de nouvelles lois de comportement non-linéaires.

Pour récapituler, cette version faiblement intrusive de la méthode LATIN-PGD entièrement intégrée dans le logiciel Simcenter Samcef™ permet la construction rapide et efficace de modèles réduits en environnement industriel. Balayant un large spectre d'activités – allant des niveaux 3 à 7 de l'échelle TRL [1] –, ces travaux de thèse ont montré de premiers résultats prometteurs que Siemens Digital Industries Software souhaite consolider dans les prochaines années. À ce titre, de nombreuses perspectives sont envisagées.

Perspectives à court terme. Pour commencer, à court terme, il s'agit de pérenniser les développements qui ont été effectués, ainsi que de valider la méthode sur d'autres cas non encore explorés : matériaux composites ou endommageables, phénomènes de type *snap-back* [Vandoren *et al.*, 2013], ou encore gestion des grands nombres de cycles lors d'études de fatigue.

Fonctionnant en environnement industriel, il pourrait être également intéressant de jumeler la version LATIN-PGD faiblement intrusive développée à d'autres outils logiciels. Pour ne citer que deux exemples : (i) mobiliser des logiciels d'exploration d'espaces paramétriques tels que HEEDS™ – déjà capable d'interagir avec le logiciel Simcenter Samcef™ – permettrait de mener une exploration plus parcimonieuse et optimisée de ces espaces de grandes dimensions ; (ii) utiliser des outils de *morphing* de maillages, nativement inclus dans l'environnement Simcenter 3D™, permettrait de mener à bien des études paramétriques comportant des paramètres géométriques. Dans ces deux cas, des gains intéressants sont attendus en mobilisant notamment la propriété remarquable de la méthode LATIN-PGD qui consiste à pouvoir initialiser l'algorithme à partir de n'importe quelle approximation issue de calculs antérieurs.

Un autre point qui mériterait une attention particulière concerne la gestion automatique des pas de temps. Disponible dans le logiciel commercial, cette fonctionnalité n'a en effet pas encore été étendue au cadre de la méthode LATIN-PGD, ce qui représente un frein en termes de robustesse.

Perspectives à moyen terme. Pour continuer, à moyen terme, il serait intéressant de mettre en œuvre des stratégies d'hyper-réduction au niveau de l'étape locale. Comme nous l'avons constaté sur les illustrations industrielles, cette étape peut représenter la majeure partie du temps de calcul. Ces techniques permettraient en effet de s'affranchir des dimensions initiales du problème dans la conduite des différentes opérations algébriques, ce qui réduirait de fait le coût non négligeable de l'étape locale au fil des itérations. Parmi l'ensemble des techniques d'hyper-réduction envisageables (cf. Section-2.4.2), la méthode RPM [Capaldo *et al.*, 2017] pourrait être déployée. D'une part, cette dernière s'accommode bien avec la méthode LATIN-PGD, et d'autre part, son côté automatique basé sur les points et instants de référence semble adapté à une implémentation effective dans les logiciels industriels.

Faisant suite à l'utilisation conjointe de logiciels d'exploration d'espaces paramétriques tels que HEEDS™, des stratégies couplant méthodes de réduction de modèles et méta-modèles multi-fidélités permettraient d'accroître davantage les performances [Nachar *et al.*, 2020]. Un travail cette fois au niveau du logiciel HEEDS™ serait alors nécessaire pour tirer au mieux profit du caractère non-incrémental de la méthode LATIN-PGD naturellement génératrice d'approximations multi-fidélités, en ne considérant, le cas échéant,

1. Technology Readiness Level

des simulations que partiellement convergées.

De plus, la grande généralité de l'approche pourrait également permettre dans le futur, avec seulement quelques adaptations mineures, d'aborder la résolution d'autres physiques telles que la thermique ou la magnéto-statique. Le logiciel Simcenter Samcef™ regroupe de fait, sous une architecture commune, un solveur mécanique non-linéaire, mais également un solveur thermique instationnaire, ce qui permet d'envisager la simulation de problèmes multi-physiques. Ces capacités alliant simulations mécaniques et thermiques sont aujourd'hui à la base des outils utilisés pour modéliser les procédés de fabrication additive, extrêmement gourmands en temps de calcul. Les méthodes de réduction de modèles pourraient apporter une aide précieuse dans ce cadre [Quaranta *et al.*, 2020, Ghnatios *et al.*, 2021b].

Perspectives à plus long terme. Enfin, à plus long terme, il s'agirait de lever l'hypothèse des déplacements modérés, en dehors des zones d'instabilité, pour étendre la méthodologie aux cas du flambage ou encore du contact avec grands glissements.

Un autre axe de recherche s'inscrit dans la lignée générale des jumeaux numériques – *Digital Twins*. La méthode LATIN-PGD présente de nombreux avantages pour adresser efficacement la partie simulation haute fidélité [Barabinot *et al.*, 2021], mais d'autres travaux connexes seraient nécessaires pour enrichir le modèle avec des données. Ceci permettrait à terme la création d'un outil de maintenance prédictive fiable et robuste.

Annexe A

Contributions non-exhaustives aux développements de la méthode LATIN

Scope	LATIN version	Search directions Θ^+	Search directions Θ^-	References	Features	MS	MP	PGD
Large deformation	$\mathcal{F}$	∞	$T_{\widehat{s}}(\Gamma)$	[Bussy *et al.*, 1990]	Plasticity			✓
		$\alpha\mathcal{K}$	$\alpha\mathcal{K}$	[Liu *et al.*, 1996]	Forming process			✓
		∞	$\alpha\mathcal{K}$	[Abdali *et al.*, 1996]	Sheet cutting			✓
				[Bellenger et Bussy, 1998]	Visco-plasticity, damage			✓
				[Bellenger et Bussy, 2001]	Creep damage			✓
	$\mathcal{IV}$	∞	$\alpha\mathcal{K}$	[Boucard *et al.*, 1997]	Beam buckling			✓
				[Aubard *et al.*, 2002]	Elastomers damage			
Elasto-visco-plasticity, large number of cycles, fatigue, damage, quasi-brittle materials	$\mathcal{F}$	$T_{\widehat{s}}(\Gamma)$	$T_{\widehat{s}}(\Gamma)$	[Ladevèze, 1985]	Reference work			✓
				[Boisse *et al.*, 1990]				✓
				[Boisse *et al.*, 1991]				✓
				[Cognard et Ladevèze, 1991]	$\mathcal{IV}$ premises			✓
				[Hu et Thomson, 1996]				✓
		∞	$\alpha\mathcal{K}$	[Vandoren *et al.*, 2013]	Snap-backs			
	$\mathcal{IV}$	$T_{\widehat{s}}(\Gamma)$	$T_{\widehat{s}}(\Gamma)$	[Arzt *et al.*, 1992]				✓
				[Cognard et Ladevèze, 1993]				✓
				[Ladevèze, 1999]	Reference work			✓
				[Ladevèze et Perego, 2000]	Generalized variables			
		∞	$T_{\widehat{s}}(\Gamma)$	[Boucard et Ladevèze, 1999]	Buckling		✓	✓
				[Cognard *et al.*, 1999]	Thermo-mechanical			✓
				[Pelle et Ryckelynck, 2000]	Verification			✓
				[Relun *et al.*, 2013]				✓
				[Néron *et al.*, 2015]			✓	✓
				[Bhattacharyya *et al.*, 2018]	Cyclic damage			✓
		$\widehat{\varepsilon} = \varepsilon_\ell$	$T_{\widehat{s}}(\Gamma)$	[Nachar, 2019]	Multi-fidelity kriging		✓	✓
		$\alpha\mathcal{K}$	$\alpha\mathcal{K}$	[Vitse *et al.*, 2019]	Concrete damage		✓	✓
	$\mathcal{DDM}$			[Ladevèze, 1999]				
				[Passieux *et al.*, 2010]	MS in space & time	✓		✓
Thermal	$\mathcal{F}$	$T_{\widehat{s}}(\Gamma)$	$T_{\widehat{s}}(\Gamma)$	[Heyberger *et al.*, 2013]	Optimal MP		✓	✓
Non-linear dynamics, fast dynamics, shocks, wave propagation	$\mathcal{F}$	$\mathcal{K}$	$\mathcal{K}$	[Royer, 1990]				✓
		∞	$\mathcal{K}$	[Jourdan et Bussy, 2000]	Sheet cutting			✓
		∞	$T_{\widehat{s}}(\Gamma)$	[Givoli *et al.*, 2017]	Causality principle			
	$\mathcal{IV}$	∞	$T_{\widehat{s}}(\Gamma)$	[Gaignebet, 1996]	Visco-plasticity			✓
	$\mathcal{DDM}$			[Lemoussu *et al.*, 2002]	Assembly			
				[Boucard *et al.*, 2003]				
				[Sen Gupta *et al.*, 2006]	Composite damage			
				[Odièvre *et al.*, 2010]		✓	✓	
				[Boucard *et al.*, 2011]	Frictional contact	✓	✓	
Composite damage	$\mathcal{F}$	∞	$\mathcal{K}$	[Allix *et al.*, 1989]				✓
				[Allix et Ladevèze, 1992]				✓
				[Guinard *et al.*, 2002]				
	$\mathcal{IV}$	∞	$T_{\widehat{s}}(\Gamma)$	[Douchin et Ladevèze, 2001]				
	$\mathcal{DDM}$			[Violeau *et al.*, 2009]		✓		
				[Kerfriden *et al.*, 2009]		✓		
				[Trovalet *et al.*, 2009]		✓		
				[Roulet *et al.*, 2013]	Assembly	✓	✓	
				[Saavedra *et al.*, 2017]	Buckling	✓		

Inverse problems, identification	$\mathcal{IV}$	∞	$\mathrm{T}_{\widehat{s}}(\Gamma)$	[Allix et Vidal, 2002] [Nguyen *et al.*, 2008a]	Visco-plasticity, damage		✓
Assembly, frictional contact, parallelism, cracking	$\mathcal{F}$			[Dolbow *et al.*, 2001]	X-FEM		
				[Gravouil *et al.*, 2011]	Stabilized X-FEM		
				[Trollé *et al.*, 2012]	Frictional cracks		
				[Giacoma *et al.*, 2014]	POD & MG-FAS	✓	
				[Giacoma *et al.*, 2015]	Quasi-optimality		✓
	$\mathcal{DDM}$			[Ladevèze et Lorong, 1991]			
				[Blanzé *et al.*, 1992]	2D - axisymmetrical		
				[Cognard *et al.*, 1996]			
				[Champaney *et al.*, 1999]	3D		
				[Ladevèze et Dureisseix, 1999]		✓	
				[Blanzé *et al.*, 2000]			
				[Ladevèze *et al.*, 2001]		✓	
				[Loiseau *et al.*, 2002]	Composite	✓	
				[Boucard et Champaney, 2003]			✓
				[Nouy et Ladevèze, 2004]	Visco-plasticity, damage	✓	
				[Boucard *et al.*, 2007]	Optimization		
				[Guidault *et al.*, 2008]	X-FEM	✓	
				[Alart et Dureisseix, 2008]	Discrete systems	✓	
				[Caignot *et al.*, 2010]	Damping	✓	
				[Capaldo, 2015]	Hyper-reduction RPM	✓	✓
				[Oumaziz *et al.*, 2017]	Non-intrusive	✓	
				[Claus et Kerfriden, 2018]	CutFEM		
Multi-physics	$\mathcal{DDM}$			[Dureisseix *et al.*, 2003]	Poro-elasticity	✓	✓
				[Néron et Dureisseix, 2008]	Thermo-poro-elasticity	✓	✓

MS : multi-scale **MP** : multi-parametric $\mathrm{T}_{\widehat{s}}(\Gamma)$: search direction related to *tangential* operator
$\mathcal{F}$: functional variant $\mathcal{IV}$: internal variables variant $\mathcal{DDM}$: domain decomposition variant

Search directions for $\mathcal{F}$ variant :

$$\begin{aligned}(\widehat{\boldsymbol{\sigma}}_\ell - \boldsymbol{\sigma}_\ell) + \boldsymbol{\Theta}^+(\widehat{\boldsymbol{\varepsilon}}_\ell - \boldsymbol{\varepsilon}_\ell) = \mathbf{0}\\ (\boldsymbol{\sigma}_{\ell+1} - \widehat{\boldsymbol{\sigma}}_\ell) - \boldsymbol{\Theta}^-(\boldsymbol{\varepsilon}_{\ell+1} - \widehat{\boldsymbol{\varepsilon}}_\ell) = \mathbf{0}\end{aligned} \tag{A.1}$$

Search directions for $\mathcal{IV}$ variant :

$$\begin{aligned}(\widehat{\boldsymbol{\Xi}}_\ell - \boldsymbol{\Xi}_\ell) + \boldsymbol{\Theta}^+(\widehat{\boldsymbol{\Xi}}^*_\ell - \boldsymbol{\Xi}^*_\ell) = \mathbf{0}\\ (\boldsymbol{\Xi}_{\ell+1} - \widehat{\boldsymbol{\Xi}}_\ell) - \boldsymbol{\Theta}^-(\boldsymbol{\Xi}^*_{\ell+1} - \widehat{\boldsymbol{\Xi}}^*_\ell) = \mathbf{0}\end{aligned} \tag{A.2}$$

Bibliographie

[Abdali *et al.*, 1996] ABDALI, A., BENKRID, K. et BUSSY, P. (1996). Simulation of sheet cutting by the large time increment method. *Journal of Materials Processing Technology*, 60(1):255–260. 3, 16, 130

[Ahrens *et al.*, 2005] AHRENS, J., GEVECI, B. et LAW, C. (2005). ParaView : An End-User Tool for Large Data Visualization. *Visualization Handbook*. 52

[Alameddin *et al.*, 2019] ALAMEDDIN, S., FAU, A., NÉRON, D., LADEVÈZE, P. et NACKENHORST, U. (2019). Toward Optimality of Proper Generalised Decomposition Bases. *Mathematical and Computational Applications*, 24(1):30. 36

[Alart, 1997] ALART, P. (1997). Méthode de Newton généralisée en mécanique du contact. *Journal de Mathématiques Pures et Appliquées*, 76(1):83–108. 23

[Alart et Curnier, 1991] ALART, P. et CURNIER, A. (1991). A mixed formulation for frictional contact problems prone to Newton like solution methods. *Computer Methods in Applied Mechanics and Engineering*, 92(3):353–375. 22, 23, 92

[Alart et Dureisseix, 2008] ALART, P. et DUREISSEIX, D. (2008). A scalable multiscale LATIN method adapted to nonsmooth discrete media. *Computer Methods in Applied Mechanics and Engineering*, 197(5):319–331. 131

[Alart *et al.*, 1995] ALART, P., LEBON, F., QUITTAU, F. et REY, K. (1995). Frictional Contact Problem in Elastostatics : Revisiting the Uniqueness Condition. *In* RAOUS, M., JEAN, M. et MOREAU, J. J., éditeurs : *Contact Mechanics*, pages 63–70. Springer US, Boston, MA. 16

[Allara, 2009] ALLARA, M. (2009). A model for the characterization of friction contacts in turbine blades. *Journal of Sound and Vibration*, 320(3):527–544. 16

[Allix et Ladevèze, 1992] ALLIX, O. et LADEVÈZE, P. (1992). Interlaminar interface modelling for the prediction of delamination. *Composite Structures*, 22(4):235–242. 130

[Allix *et al.*, 1989] ALLIX, O., LADEVÈZE, P., GILLETTA, D. et OHAYON, R. (1989). A damage prediction method for composite structures. *International Journal for Numerical Methods in Engineering*, 27(2):271–283. 130

[Allix et Vidal, 2002] ALLIX, O. et VIDAL, P. (2002). A new multi-solution approach suitable for structural identification problems. *Computer Methods in Applied Mechanics and Engineering*, 191(25):2727–2758. 131

[Ammar *et al.*, 2012] AMMAR, A., CHINESTA, F., CUETO, E. et DOBLARÉ, M. (2012). Proper generalized decomposition of time-multiscale models. *International Journal for Numerical Methods in Engineering*, 90(5):569–596. 35

[Amsallem et Farhat, 2008] AMSALLEM, D. et FARHAT, C. (2008). Interpolation Method for Adapting Reduced-Order Models and Application to Aeroelasticity. *AIAA Journal*, 46(7):1803–1813. 28

[Arzt *et al.*, 1992] ARZT, M., COGNARD, J.-Y. et LADEVÈZE, P. (1992). An efficient computational method for complex loading histories. *In* OWEN, D., ONÃTE, E. et HINTON, E., éditeurs : *Proceedings of 3rd International Conference on Computational Plasticity*, pages 225–236, Swansea, UK. Pineridge Press. 130

[Astrid *et al.*, 2008] ASTRID, P., WEILAND, S., WILLCOX, K. et BACKX, T. (2008). Missing Point Estimation in Models Described by Proper Orthogonal Decomposition. *IEEE Transactions on Automatic Control*, 53(10):2237–2251. 39

[Aubard *et al.*, 2002] AUBARD, X., BOUCARD, P.-A., LADEVÈZE, P. et MICHEL, S. (2002). Modeling and simulation of damage in elastomer structures at high strains. *Computers & Structures*, 80(27):2289–2298. 130

[Audouze *et al.*, 2013] AUDOUZE, C., VUYST, F. D. et NAIR, P. B. (2013). Nonintrusive reduced-order modeling of parametrized time-dependent partial differential equations. *Numerical Methods for Partial Differential Equations*, 29(5):1587–1628. 3, 42

[Barabinot *et al.*, 2021] BARABINOT, P., SCANFF, R., LADEVÈZE, P., NÉRON, D. et CAUVILLE, B. (2021). Industrial Digital Twins based on the non-linear LATIN-PGD. *Advanced Modeling and Simulation in Engineering Sciences*, 8(1):22. 129

[Barrault *et al.*, 2004] BARRAULT, M., MADAY, Y., NGUYEN, N. C. et PATERA, A. T. (2004). An 'empirical interpolation' method : Application to efficient reduced-basis discretization of partial differential equations. *Comptes Rendus Mathématique*, 339(9):667–672. 39

[Bellenger et Bussy, 1998] BELLENGER, E. et BUSSY, P. (1998). Plastic and viscoplastic damage models with numerical treatment for metal forming processes. *Journal of Materials Processing Technology*, 80–81:591–596. 26, 130

[Bellenger et Bussy, 2001] BELLENGER, E. et BUSSY, P. (2001). Phenomenological modeling and numerical simulation of different modes of creep damage evolution. *International Journal of Solids and Structures*, 38(4):577–604. 130

[Benner *et al.*, 2017] BENNER, P., COHEN, A., OHLBERGER, M. et WILLCOX, K., éditeurs (2017). *Model Reduction and Approximation : Theory and Algorithms*. Numéro 15 de Computational Science and Engineering. Society for Industrial and Applied Mathematics, Philadelphia. 30

[Benner *et al.*, 2021] BENNER, P., GRIVET-TALOCIA, S., QUARTERONI, A., ROZZA, G., SCHILDERS, W. et SILVEIRA, L. M., éditeurs (2021). *Model Order Reduction : Snapshot-Based Methods and Algorithms*, volume 2. De Gruyter, Berlin, Boston. 2, 4, 26, 28, 33

[Beringhier *et al.*, 2010] BERINGHIER, M., GUEGUEN, M. et GRANDIDIER, J. C. (2010). Solution of Strongly Coupled Multiphysics Problems Using Space-Time Separated Representations—Application to Thermoviscoelasticity. *Archives of Computational Methods in Engineering*, 17(4):393–401. 33, 35

[Beringhier *et al.*, 2016] BERINGHIER, M., LEYGUE, A. et CHINESTA, F. (2016). Parametric nonlinear PDEs with multiple solutions : A PGD approach. *Discrete & Continuous Dynamical Systems - S*, 9(2):383. 35

[Bertsekas, 1984] BERTSEKAS, D. P. (1984). *Constrained Optimization and Lagrange Multiplier Methods*. Academic Press. 22

[Bhattacharyya *et al.*, 2018] BHATTACHARYYA, M., FAU, A., NACKENHORST, U., NÉRON, D. et LADEVÈZE, P. (2018). A LATIN-based model reduction approach for the simulation of cycling damage. *Computational Mechanics*, 62(4):725–743. 4, 26, 38, 130

[Blanzé *et al.*, 1996] BLANZÉ, C., CHAMPANEY, L., COGNARD, J.-Y. et LADEVÈZE, P. (1996). A modular approach to structure assembly computations : Application to contact problems. *Engineering Computations*, 13(1):15–32. 26

[Blanzé *et al.*, 2000] BLANZÉ, C., CHAMPANEY, L. et VEDRINE, P. (2000). Contact problems in the design of a superconducting quadrupole prototype. *Engineering Computations*, 17(2):136–153. 131

[Blanzé *et al.*, 1992] BLANZÉ, C., DANWÉ, R., LADEVÈZE, P. et MAUREL, P. (1992). A new simplified method for the analysis of 2D structures. *In XXVIIIth International Congress of Theorical and Applied Mechanics*. 131

[Boisse *et al.*, 1990] BOISSE, P., BUSSY, P. et LADEVÈZE, P. (1990). A new approach in non-linear mechanics : The large time increment method. *International Journal for Numerical Methods in Engineering*, 29(3):647–663. 4, 5, 130

[Boisse *et al.*, 1991] BOISSE, P., LADEVÈZE, P., POSS, M. et ROUGÉE, P. (1991). A new large time increment algorithm for anisotropic plasticity. *International Journal of Plasticity*, 7(1):65–77. 27, 130

[Bonnet *et al.*, 2014] BONNET, M., FRANGI, A. et REY, C. (2014). *The Finite Element Method in Solid Mechanics*. McGraw Hill Education, New York. 20

[Borzacchiello *et al.*, 2019] BORZACCHIELLO, D., AGUADO, J. V. et CHINESTA, F. (2019). Non-intrusive Sparse Subspace Learning for Parametrized Problems. *Archives of Computational Methods in Engineering*, 26(2):303–326. 3, 4, 42

[Boucard *et al.*, 2007] BOUCARD, P.-A., BUYTET, S. et GUIDAULT, P.-A. (2007). Une stratégie multi-échelle pour l'étude paramétrique de détails géométriques au sein de structures en contacts multiples. *European Journal of Computational Mechanics*, 16(8):1011–1036. 131

[Boucard et Champaney, 2003] BOUCARD, P.-A. et CHAMPANEY, L. (2003). A suitable computational strategy for the parametric analysis of problems with multiple contact. *International Journal for Numerical Methods in Engineering*, 57(9):1259–1281. 28, 131

[Boucard *et al.*, 2003] BOUCARD, P.-A., DÉRUMAUX, M., LADEVÈZE, P. et ROUX, P. (2003). - Macro-meso models for joint submitted to pyrotechnic shock. *In* BATHE, K. J., éditeur : *Computational Fluid and Solid Mechanics 2003*, pages 139–142. Elsevier Science Ltd, Oxford. 130

[Boucard et Ladevèze, 1999] BOUCARD, P.-A. et LADEVÈZE, P. (1999). Une application de la méthode LATIN au calcul multirésolution de structures non linéaires. *Revue Européenne des Éléments Finis*, 8(8):903–920. 130

[Boucard *et al.*, 1997] BOUCARD, P.-A., LADEVÈZE, P., POSS, M. et ROUGÉE, P. (1997). A nonincremental approach for large displacement problems. *Computers & Structures*, 64(1):499–508. 15, 26, 130

[Boucard *et al.*, 2011] BOUCARD, P.-A., ODIÈVRE, D. et GATUINGT, F. (2011). A parallel and multiscale strategy for the parametric study of transient dynamic problems with friction. *International Journal for Numerical Methods in Engineering*, 88(7):657–672. 130

[Bouclier *et al.*, 2013] BOUCLIER, R., LOUF, F. et CHAMOIN, L. (2013). Real-time validation of mechanical models coupling PGD and constitutive relation error. *Computational Mechanics*, 52(4):861–883. 35

[Broyden, 1965] BROYDEN, C. G. (1965). A Class of Methods for Solving Nonlinear Simultaneous Equations. *Mathematics of Computation*, 19(92):577–593. 24

[Bussy *et al.*, 1990] BUSSY, P., ROUGÉE, P. et VAUCHEZ, P. (1990). The large time increment method for numerical simulation of metal forming processes. *In NUMETA*, pages 102–109. Elsevier Science Ltd, New York. 4, 27, 130

[Cagniart *et al.*, 2019] CAGNIART, N., MADAY, Y. et STAMM, B. (2019). Model Order Reduction for Problems with Large Convection Effects. *In* CHETVERUSHKIN, B. N., FITZGIBBON, W., KUZNETSOV, Y., NEITTAANMÄKI, P., PERIAUX, J. et PIRONNEAU, O., éditeurs : *Contributions to Partial Differential Equations and Applications*, Computational Methods in Applied Sciences, pages 131–150. Springer International Publishing, Cham. 30

[Caignot *et al.*, 2010] CAIGNOT, A., LADEVÈZE, P., NÉRON, D. et DURAND, J.-F. (2010). Virtual testing for the prediction of damping in joints. *Engineering Computations*, 27(5):621–644. 131

[Capaldo, 2015] CAPALDO, M. (2015). *A New Approximation Framework for PGD-based Nonlinear Solvers*. Thèse de doctorat, Ecole Normale Supérieure de Cachan, Université Paris-Saclay. 131

[Capaldo *et al.*, 2017] CAPALDO, M., GUIDAULT, P.-A., NÉRON, D. et LADEVÈZE, P. (2017). The Reference Point Method, a "hyperreduction" technique : Application to PGD-based nonlinear model reduction. *Computer Methods in Applied Mechanics and Engineering*, 322:483–514. 39, 66, 128

[Cardoso, 2019] CARDOSO, R. A. (2019). *Études numériques sur la modélisation du fretting fatigue*. Thèse de doctorat, Université Paris-Saclay ; University of Brasilia. 53

[Carlberg *et al.*, 2013] CARLBERG, K., FARHAT, C., CORTIAL, J. et AMSALLEM, D. (2013). The GNAT method for nonlinear model reduction : Effective implementation and application to computational fluid dynamics and turbulent flows. *Journal of Computational Physics*, 242:623–647. 39

[Casenave *et al.*, 2020a] CASENAVE, F., AKKARI, N., BORDEU, F., REY, C. et RYCKELYNCK, D. (2020a). A nonintrusive distributed reduced-order modeling framework for nonlinear structural mechanics—Application to elastoviscoplastic computations. *International Journal for Numerical Methods in Engineering*, 121(1):32–53. 34, 42

[Casenave *et al.*, 2014] CASENAVE, F., ERN, A. et LELIÈVRE, T. (2014). A nonintrusive reduced basis method applied to aeroacoustic simulations. *Advances in Computational Mathematics*, 41(5):961–986. 3, 42

[Casenave *et al.*, 2016] CASENAVE, F., ERN, A. et LELIÈVRE, T. (2016). Variants of the Empirical Interpolation Method : Symmetric formulation, choice of norms and rectangular extension. *Applied Mathematics Letters*, 56:23–28. 39

[Casenave *et al.*, 2020b] CASENAVE, F., GARIAH, A., REY, C. et FEYEL, F. (2020b). A nonintrusive reduced order model for nonlinear transient thermal problems with nonparametrized variability. *Advanced Modeling and Simulation in Engineering Sciences*, 7(1):22. 4, 38

[Chaboche, 1986] CHABOCHE, J.-L. (1986). Time-independent constitutive theories for cyclic plasticity. *International Journal of Plasticity*, 2(2):149–188. 12, 13

[Chaboche, 1989] CHABOCHE, J.-L. (1989). Constitutive equations for cyclic plasticity and cyclic viscoplasticity. *International Journal of Plasticity*, 5(3):247–302. 12, 13

[Champaney *et al.*, 1997] CHAMPANEY, L., COGNARD, J. Y., DUREISSEIX, D. et LADEVÈZE, P. (1997). Large scale applications on parallel computers of a mixed domain decomposition method. *Computational Mechanics*, 19(4):253–263. 26

[Champaney *et al.*, 1999] CHAMPANEY, L., COGNARD, J. Y. et LADEVÈZE, P. (1999). Modular analysis of assemblages of three-dimensional structures with unilateral contact conditions. *Computers & Structures*, 73(1):249–266. 44, 131

[Chatterjee, 2000] CHATTERJEE, A. (2000). An introduction to the proper orthogonal decomposition. *Current Science*, 78(7):808–817. 2, 33

[Chaturantabut et Sorensen, 2010] CHATURANTABUT, S. et SORENSEN, D. C. (2010). Nonlinear Model Reduction via Discrete Empirical Interpolation. *SIAM Journal on Scientific Computing*, 32(5):2737–2764. 39

[Chinesta *et al.*, 2010] CHINESTA, F., AMMAR, A. et CUETO, E. (2010). Recent Advances and New Challenges in the Use of the Proper Generalized Decomposition for Solving Multidimensional Models. *Archives of Computational Methods in Engineering*, 17(4):327–350. 35

[Chinesta *et al.*, 2020] CHINESTA, F., CUETO, E., ABISSET-CHAVANNE, E., DUVAL, J.-L. et KHALDI, F. E. (2020). Virtual, Digital and Hybrid Twins : A New Paradigm in Data-Based Engineering and Engineered Data. *Archives of Computational Methods in Engineering*, 27(1):105–134. 3, 42

[Chinesta *et al.*, 2014] CHINESTA, F., KEUNINGS, R. et LEYGUE, A. (2014). *The Proper Generalized Decomposition for Advanced Numerical Simulations*. SpringerBriefs in Applied Sciences and Technology. Springer Science & Business Media, Cham. 2

[Chinesta et Ladevèze, 2014] CHINESTA, F. et LADEVÈZE, P., éditeurs (2014). *Separated Representations and PGD-based Model Reduction : Fundamentals and Applications*. Numéro 554 de Courses and Lectures / International Centre for Mechanical Sciences. Springer, Wien. 35

[Chinesta et Ladevèze, 2020] CHINESTA, F. et LADEVÈZE, P. (2020). Proper generalized decomposition. *In Model Order Reduction : Snapshot-Based Methods and Algorithms*, volume 2, chapitre Volume 2 Snapshot-Based Methods and Algorithms, pages 97–138. De Gruyter. 35

[Chinesta *et al.*, 2011] CHINESTA, F., LADEVÈZE, P. et CUETO, E. (2011). A Short Review on Model Order Reduction Based on Proper Generalized Decomposition. *Archives of Computational Methods in Engineering*, 18(4):395–404. 35

[Chinesta *et al.*, 2013] CHINESTA, F., LEYGUE, A., BERINGHIER, M., TUAN NGUYEN, L., GRANDIDIER, J.-C., SCHREFLER, B. et PESAVENTO, F. (2013). Towards a framework for non-linear thermal models in shell domains. *International Journal of Numerical Methods for Heat & Fluid Flow*, 23(1):55–73. 38

[Ciarlet, 1988] CIARLET, P. G. (1988). *Mathematical Elasticity : Three-dimensional Elasticity*, volume 1. North-Holland. 16

[Ciarlet, 2007] CIARLET, P. G. (2007). *Introduction à l'analyse numérique matricielle et à l'optimisation*. DUNOD. 21

[Claus et Kerfriden, 2018] CLAUS, S. et KERFRIDEN, P. (2018). A stable and optimally convergent LaTIn-CutFEM algorithm for multiple unilateral contact problems. *International Journal for Numerical Methods in Engineering*, 113(6):938–966. 131

[Cochelin *et al.*, 1994] COCHELIN, B., DAMIL, N. et POTIER-FERRY, M. (1994). Asymptotic–numerical methods and Pade approximants for non-linear elastic structures. *International Journal for Numerical Methods in Engineering*, 37(7):1187–1213. 38

[Cocu, 1984] COCU, M. (1984). Existence of solutions of Signorini problems with friction. *International Journal of Engineering Science*, 22(5):567–575. 16

[Cognard *et al.*, 1999] COGNARD, J., LADEVÈZE, P. et TALBOT, P. (1999). A large time increment approach for thermo-mechanical problems. *Advances in Engineering Software*, 30(9-11):583–593. 5, 130

[Cognard *et al.*, 1996] COGNARD, J.-Y., DUREISSEIX, D., LADEVÈZE, P. et LORONG, P. (1996). Expérimentation d'une approche parallèle en calcul des structures. *Revue Européenne des Éléments Finis*, 5(2):197–220. 131

[Cognard et Ladevèze, 1991] COGNARD, J.-Y. et LADEVÈZE, P. (1991). The Large Time Increment Method Applied to Cyclic Loadings. *In* ŻYCZKOWSKI, M., éditeur : *Creep in Structures*, pages 555–562. Springer Berlin Heidelberg, Berlin, Heidelberg. 4, 5, 130

[Cognard et Ladevèze, 1993] COGNARD, J.-Y. et LADEVÈZE, P. (1993). A large time increment approach for cyclic viscoplasticity. *International Journal of Plasticity*, 9(2):141–157. 4, 27, 130

[Courard *et al.*, 2016] COURARD, A., NÉRON, D., LADEVÈZE, P. et BALLERE, L. (2016). Integration of PGD-virtual charts into an engineering design process. *Computational Mechanics*, 57(4):637–651. 3, 4, 42, 43

[Courrier *et al.*, 2014] COURRIER, N., BOUCARD, P.-A. et SOULIER, B. (2014). The use of partially converged simulations in building surrogate models. *Advances in Engineering Software*, 67:186–197. 117

[Crisfield, 1981] CRISFIELD, M. A. (1981). A fast incremental/iterative solution procedure that handles "snap-through". *Computers & Structures*, 13(1):55–62. 24

[Curnier et Alart, 1988] CURNIER, A. et ALART, P. (1988). A generalized Newton method for contact problems with friction. *Journal de Mécanique Théorique et Appliquée*. 23

[Curnier *et al.*, 1992] CURNIER, A., HE, Q.-C. et TELEGA, J. J. (1992). Formulation of unilateral contact between two elastic bodies undergoing finite derformations. *Formulation of unilateral contact between two elastic bodies undergoing finite derformations*, 314(1):1–6. 16

[Daby-Seesaram *et al.*, 2021] DABY-SEESARAM, A., FAU, A., CHARBONNEL, P. E. et NÉRON, D. (2021). Model-order reduction for nonlinear dynamics including nonlinearities induced by damage. *In VI ECCOMAS YIC Proceedings*. 53

[Dafalias, 1983] DAFALIAS, Y. F. (1983). Corotational Rates for Kinematic Hardening at Large Plastic Deformations. *Journal of Applied Mechanics*, 50(3):561–565. 15

[de Borst *et al.*, 2012] de BORST, R., CRISFIELD, M. A., REMMERS, J. J. C. et VERHOOSEL, C. V. (2012). *Nonlinear Finite Element Analysis of Solids and Structures*. John Wiley & Sons, United Kingdom. 20

[Dolbow *et al.*, 2001] DOLBOW, J., MOËS, N. et BELYTSCHKO, T. (2001). An extended finite element method for modeling crack growth with frictional contact. *Computer Methods in Applied Mechanics and Engineering*, 190(51):6825–6846. 131

[Douchin et Ladevèze, 2001] DOUCHIN, B. et LADEVÈZE, P. (2001). Mise en œuvre numérique d'un mésomodèle d'endommagement des stratifiés. *Revue Européenne des Éléments Finis*, 10(2-4):473–487. 130

[Dumon *et al.*, 2011] DUMON, A., ALLERY, C. et AMMAR, A. (2011). Proper general decomposition (PGD) for the resolution of Navier–Stokes equations. *Journal of Computational Physics*, 230(4):1387–1407. 33

[Dureisseix *et al.*, 2003] DUREISSEIX, D., LADEVÈZE, P., NÉRON, D. et SCHREFLER, B. (2003). A Multi-Time-Scale Strategy for Multiphysics Problems : Application to Poroelasticity. *International Journal for Multiscale Computational Engineering*, 1(4). 26, 131

[Duvaut et Lions, 1976] DUVAUT, G. et LIONS, J. L. (1976). *Inequalities in Mechanics and Physics*. Springer-Verlag, Berlin, Heidelberg. 16

[Farhat *et al.*, 2014] FARHAT, C., AVERY, P., CHAPMAN, T. et CORTIAL, J. (2014). Dimensional reduction of nonlinear finite element dynamic models with finite rotations and energy-based mesh sampling and weighting for computational efficiency. *International Journal for Numerical Methods in Engineering*, 98(9):625–662. 39

[Fauque *et al.*, 2018] FAUQUE, J., RAMIÈRE, I. et RYCKELYNCK, D. (2018). Hybrid hyper-reduced modeling for contact mechanics problems. *International Journal for Numerical Methods in Engineering*, 115(1):117–139. 33

[Frederick et Armstrong, 2007] FREDERICK, C. O. et ARMSTRONG, P. J. (2007). A mathematical representation of the multiaxial Bauschinger effect. *Materials at High Temperatures*, 24(1):1–26. 12

[Gaignebet, 1996] GAIGNEBET, Y. (1996). *Approche Non Incrémentale Des Calculs de Chocs Pour Des Structures Viscoplastiques*. Thèse de doctorat, Ecole Normale Supérieure de Cachan. 130

[Germain, 1973] GERMAIN, P. (1973). *Cours de Mécanique Des Milieux Continus*, volume 1. Masson. 12

[Germain *et al.*, 1983] GERMAIN, P., SUQUET, P. et NGUYEN, Q. S. (1983). Continuum thermodynamics. *ASME Transactions Series E Journal of Applied Mechanics*, 50:1010–1020. 12

[Geuzaine et Remacle, 2009] GEUZAINE, C. et REMACLE, J.-F. (2009). Gmsh : A 3-D finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering*, 79(11):1309–1331. 52

[Ghnatios *et al.*, 2021a] GHNATIOS, C., CUETO, E., FALCO, A., DUVAL, J.-L. et CHINESTA, F. (2021a). Spurious-free interpolations for non-intrusive PGD-based parametric solutions : Application to composites forming processes. *International Journal of Material Forming*, 14(1):83–95. 42

[Ghnatios *et al.*, 2021b] GHNATIOS, C., RAI, K. E., HASCOET, N., PIRES, P.-A., DUVAL, J.-L., LAMBARRI, J., HASCOET, J.-Y. et CHINESTA, F. (2021b). Reduced order modeling of selective laser melting : From calibration to parametric part distortion. *International Journal of Material Forming*. 129

[Giacoma *et al.*, 2016] GIACOMA, A., DUREISSEIX, D. et GRAVOUIL, A. (2016). An efficient quasi-optimal space-time PGD application to frictional contact mechanics. *Advanced Modeling and Simulation in Engineering Sciences*, 3(1):1–17. 33

[Giacoma *et al.*, 2014] GIACOMA, A., DUREISSEIX, D., GRAVOUIL, A. et ROCHETTE, M. (2014). A multiscale large time increment/FAS algorithm with time-space model reduction for frictional contact problems. *International Journal for Numerical Methods in Engineering*, 97(3):207–230. 26, 27, 131

[Giacoma *et al.*, 2015] GIACOMA, A., DUREISSEIX, D., GRAVOUIL, A. et ROCHETTE, M. (2015). Toward an optimal a priori reduced basis strategy for frictional contact problems with LATIN solver. *Computer Methods in Applied Mechanics and Engineering*, 283:1357–1381. 36, 64, 91, 93, 97, 131

[Giraldi *et al.*, 2014] GIRALDI, L., LITVINENKO, A., LIU, D., MATTHIES, H. G. et NOUY, A. (2014). To Be or Not to Be Intrusive ? The Solution of Parametric and Stochastic Equations—the "Plain Vanilla" Galerkin Case. *SIAM Journal on Scientific Computing*, 36(6):A2720–A2744. 4, 41

[Giraldi *et al.*, 2015] GIRALDI, L., LIU, D., MATTHIES, H. G. et NOUY, A. (2015). To Be or Not to be Intrusive ? The Solution of Parametric and Stochastic Equations—Proper Generalized Decomposition. *SIAM Journal on Scientific Computing*, 37(1):A347–A368. 41

[Givoli *et al.*, 2017] GIVOLI, D., BHARALI, R. et SLUYS, L. J. (2017). LATIN : A new view and an extension to wave propagation in nonlinear media. *International Journal for Numerical Methods in Engineering*, 112(2):125–156. 130

[Glaessgen et Stargel, 2012] GLAESSGEN, E. et STARGEL, D. (2012). The Digital Twin Paradigm for Future NASA and U.S. Air Force Vehicles. *In 53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference*. American Institute of Aeronautics and Astronautics. 3

[Golub et Van Loan, 2013] GOLUB, G. H. et VAN LOAN, C. F. (2013). *Matrix Computations*. JHU Press. 21, 31

[González *et al.*, 2016] GONZÁLEZ, D., CUETO, E. et CHINESTA, F. (2016). Computational Patient Avatars for Surgery Planning. *Annals of Biomedical Engineering*, 44(1):35–45. 33

[González *et al.*, 2012] GONZÁLEZ, D., MASSON, F., POULHAON, F., LEYGUE, A., CUETO, E. et CHINESTA, F. (2012). Proper Generalized Decomposition based dynamic data driven inverse identification. *Mathematics and Computers in Simulation*, 82(9):1677–1695. 35

[Gravouil *et al.*, 2011] GRAVOUIL, A., PIERRES, E. et BAIETTO, M. C. (2011). Stabilized global–local X-FEM for 3D non-planar frictional crack using relevant meshes. *International Journal for Numerical Methods in Engineering*, 88(13):1449–1475. 131

[Grepl *et al.*, 2007] GREPL, M. A., MADAY, Y., NGUYEN, N. C. et PATERA, A. T. (2007). Efficient reduced-basis treatment of nonaffine and nonlinear partial differential equations. *ESAIM : Mathematical Modelling and Numerical Analysis*, 41(3):575–605. 39

[Guidault *et al.*, 2008] GUIDAULT, P.-A., ALLIX, O., CHAMPANEY, L. et CORNUAULT, C. (2008). A multiscale extended finite element method for crack propagation. *Computer Methods in Applied Mechanics and Engineering*, 197(5):381–399. 131

[Guinard *et al.*, 2002] GUINARD, S., ALLIX, O., GUÉDRA-DEGEORGES, D. et VINET, A. (2002). A 3D damage analysis of low-velocity impacts on laminated composites. *Composites Science and Technology*, 62(4):585–589. 130

[Hackbusch, 2012] HACKBUSCH, W. (2012). *Tensor Spaces and Numerical Tensor Calculus*. Springer Series in Computational Mathematics. Springer-Verlag, Berlin Heidelberg. 30

[Hackbusch et Kühn, 2009] HACKBUSCH, W. et KÜHN, S. (2009). A New Scheme for the Tensor Representation. *Journal of Fourier Analysis and Applications*, 15(5):706–722. 30

[Hammond *et al.*, 2019] HAMMOND, J. K., CHAKIR, R., BOURQUIN, F. et MADAY, Y. (2019). PBDW : A non-intrusive Reduced Basis Data Assimilation method and its application to an urban dispersion modeling framework. *Applied Mathematical Modelling*, 76:1–25. 42

[Hartmann et van der Auweraer, 2020] HARTMANN, D. et VAN DER AUWERAER, H. (2020). Digital Twins. *arXiv :2001.09747 [cs]*. 3

[Hernández *et al.*, 2017] HERNÁNDEZ, J. A., CAICEDO, M. A. et FERRER, A. (2017). Dimensional hyper-reduction of nonlinear finite element models via empirical cubature. *Computer Methods in Applied Mechanics and Engineering*, 313:687–722. 38, 39

[Hesthaven et Ubbiali, 2018] HESTHAVEN, J. S. et UBBIALI, S. (2018). Non-intrusive reduced order modeling of nonlinear problems using neural networks. *Journal of Computational Physics*, 363:55–78. 33

[Heyberger *et al.*, 2012] HEYBERGER, C., BOUCARD, P.-A. et NÉRON, D. (2012). Multiparametric analysis within the proper generalized decomposition framework. *Computational Mechanics*, 49(3):277–289. 68

[Heyberger *et al.*, 2013] HEYBERGER, C., BOUCARD, P.-A. et NÉRON, D. (2013). A rational strategy for the resolution of parametrized problems in the PGD framework. *Computer Methods in Applied Mechanics and Engineering*, 259:40–49. 35, 108, 130

[Hoareau et Deü, 2019] HOAREAU, C. et DEÜ, J.-F. (2019). Formulation d'un problème non-linéaire avec forces suiveuses via une approche éléments finis de type PGD. *In CSMA 2019 - 14ème Colloque National en Calcul des Structures*, Giens, Var, France. 38

[Hu et Thomson, 1996] HU, W. et THOMSON, P. F. (1996). An evaluation of a large time increment method. *Computers & Structures*, 58(3):633–637. 130

[Ibáñez *et al.*, 2018] IBÁÑEZ, R., ABISSET-CHAVANNE, E., AMMAR, A., GONZÁLEZ, D., CUETO, E., HUERTA, A., DUVAL, J.-L. et CHINESTA, F. (2018). A Multidimensional Data-Driven Sparse Identification Technique : The Sparse Proper Generalized Decomposition. *Complexity*. 3, 4, 42

[Ibrahimbegovic, 2009] IBRAHIMBEGOVIC, A. (2009). *Nonlinear Solid Mechanics : Theoretical Formulations and Finite Element Solution Methods*, volume 160 de *Solid Mechanics and Its Applications*. Springer Netherlands, Dordrecht. 20, 23

[Isaacson et Keller, 2012] ISAACSON, E. et KELLER, H. B. (2012). *Analysis of Numerical Methods*. Courier Corporation. 20

[Jolliffe, 1986] JOLLIFFE, I. T. (1986). *Principal Component Analysis*. Springer New York, New York, NY. 31, 33

[Jourdan et Bussy, 2000] JOURDAN, F. et BUSSY, P. (2000). Large time increment method in dynamic regularization : Sheet cutting simulations. *Computer Methods in Applied Mechanics and Engineering*, 190(8):1245–1259. 130

[Kadeethum *et al.*, 2021] KADEETHUM, T., BALLARIN, F. et BOUKLAS, N. (2021). Non-intrusive reduced order modeling of poroelasticity of heterogeneous media based on a discontinuous Galerkin approximation. *arXiv :2101.11810 [cs, math]*. 42

[Karhunen, 1947] KARHUNEN, K. (1947). Under Lineare Methoden in der Wahr Scheinlichkeitsrechnung. *Annales Academiae Scientiarun Fennicae Series A1 : Mathematia Physica*, 47. 31

[Kerfriden *et al.*, 2009] KERFRIDEN, P., ALLIX, O. et GOSSELET, P. (2009). A three-scale domain decomposition method for the 3D analysis of debonding in laminates. *Computational Mechanics*, 44(3):343–362. 130

[Kerfriden *et al.*, 2011] KERFRIDEN, P., GOSSELET, P., ADHIKARI, S. et BORDAS, S. P. A. (2011). Bridging proper orthogonal decomposition methods and augmented Newton–Krylov algorithms : An adaptive model order reduction for highly nonlinear mechanical problems. *Computer Methods in Applied Mechanics and Engineering*, 200(5):850–866. 38

[Kerfriden *et al.*, 2012] KERFRIDEN, P., PASSIEUX, J. C. et BORDAS, S. P. A. (2012). Local/global model order reduction strategy for the simulation of quasi-brittle fracture. *International Journal for Numerical Methods in Engineering*, 89(2):154–179. 33

[Kikuchi et Oden, 1988] KIKUCHI, N. et ODEN, J. T. (1988). Contact problems in elasticity : A study of variational inequalities and finite element methods. *Society for Industrial and Applied Mathematics*. 16, 22

[Klarbring, 1990] KLARBRING, A. (1990). Examples of non-uniqueness and non-existence of solutions to quasi-static contact problems with friction. *Ingenieur-Archiv*, 60(8):529–541. 16

[Kolda et Bader, 2009] KOLDA, T. G. et BADER, B. W. (2009). Tensor Decompositions and Applications. *SIAM Review*, 51(3):455–500. 30

[Koning *et al.*, 2018] KONING, W. J., JOHNSON, W. et ALLAN, B. G. (2018). Generation of Mars Helicopter Rotor Model for Comprehensive Analyses. *AHS Aeromechanics Design for Transformative Vertical Flight*. 1

[Krempl, 1975] KREMPL, E. (1975). On the interaction of rate and history dependence in structural metals. *Acta Mechanica*, 22(1):53–90. 12

[Ladevèze, 1985] LADEVÈZE, P. (1985). Sur une famille d'algorithmes en mécanique des structures. *Comptes-rendus des séances de l'Académie des sciences. Série 2, Mécanique-physique, chimie, sciences de l'univers, sciences de la terre*, 300(2):41–44. 3, 4, 5, 20, 25, 27, 130

[Ladevèze, 1989] LADEVÈZE, P. (1989). La méthode à grand incrément de temps pour l'analyse de structures à comportement non linéaire décrit par variables internes. *Comptes Rendus de l'Académie des Sciences*, 309(2): 1095–1999. 35

[Ladevèze, 1997] LADEVÈZE, P. (1997). A computational technique for the integrals over the time-space domain in connection with the LATIN method. Rapport interne LMT 193, Ecole Normale Supérieure Paris-Saclay. 39

[Ladevèze, 1999] LADEVÈZE, P. (1999). *Nonlinear Computational Structural Mechanics : New Approaches and Non-Incremental Methods of Calculation.* Springer, New York. 3, 4, 5, 12, 15, 20, 25, 26, 27, 35, 36, 58, 130

[Ladevèze, 2014] LADEVÈZE, P. (2014). PGD in linear and nonlinear Computational Solid Mechanics. *In* CHINESTA, F. et LADEVÈZE, P., éditeurs : *Separated Representations and PGD-Based Model Reduction : Fundamentals and Applications*, CISM International Centre for Mechanical Sciences, pages 91–152. Springer, Vienna. 33

[Ladevèze, 2016] LADEVÈZE, P. (2016). On reduced models in nonlinear solid mechanics. *European Journal of Mechanics - A/Solids*, 60:227–237. 39

[Ladevèze, 2017] LADEVÈZE, P. (2017). On a new non-intrusive version of the LATIN method for quasi-static loads (in French). Rapport interne LMT, Ecole Normale Supérieure Paris-Saclay. 5, 81

[Ladevèze et Dureisseix, 1999] LADEVÈZE, P. et DUREISSEIX, D. (1999). Une nouvelle stratégie de calcul micro/macro en mécanique des structures. *Comptes Rendus de l'Académie des Sciences - Series IIB - Mechanics-Physics-Astronomy*, 327(12):1237–1244. 131

[Ladevèze *et al.*, 2001] LADEVÈZE, P., LOISEAU, O. et DUREISSEIX, D. (2001). A micro–macro and parallel computational strategy for highly heterogeneous structures. *International Journal for Numerical Methods in Engineering*, 52(1-2):121–138. 131

[Ladevèze et Lorong, 1991] LADEVÈZE, P. et LORONG, P. (1991). A large time increment approach with domain decomposition for mechanical non linear problem. *In Proceedings of the 10th International Conference on Computing Methods in Applied Sciences and Engineering on Computing Methods in Applied Sciences and Engineering*, pages 569–578, Paris, France. Nova Science Publishers, Inc. 131

[Ladevèze et Perego, 2000] LADEVÈZE, P. et PEREGO, U. (2000). Duality preserving discretization of the large time increment methods. *Computer Methods in Applied Mechanics and Engineering*, 189(1):205–232. 130

[Laurent, 2013] LAURENT, L. (2013). *Stratégie multiparamétrique et métamodèles pour l'optimisation multiniveaux de structures*. Thèse de doctorat, Ecole Normale Supérieure de Cachan. 28

[Laurent *et al.*, 2013] LAURENT, L., BOUCARD, P.-A. et SOULIER, B. (2013). A dedicated multiparametric strategy for the fast construction of a cokriging metamodel. *Computers & Structures*, 124:61–73. 117

[Lee, 1969] LEE, E. H. (1969). Elastic-Plastic Deformation at Finite Strains. *Journal of Applied Mechanics*, 36(1):1–6. 15

[Lee et Liu, 1967] LEE, E. H. et LIU, D. T. (1967). Finite-Strain Elastic—Plastic Theory with Application to Plane-Wave Analysis. *Journal of Applied Physics*, 38(1):19–27. 15

[Lemaitre et Chaboche, 1990] LEMAITRE, J. et CHABOCHE, J.-L. (1990). *Mechanics of Solid Materials*. Cambridge University Press, Cambridge. 12

[Lemaitre *et al.*, 2009] LEMAITRE, J., CHABOCHE, J.-L., BENALLAL, A. et DESMORAT, R. (2009). *Mécanique des matériaux solides - 3e édition*. Dunod, Paris, France. 12, 15, 16

[Lemoussu *et al.*, 2002] LEMOUSSU, H., BOUCARD, P.-A. et LADEVÈZE, P. (2002). A 3D shock computational strategy for real assembly and shock attenuator. *Advances in Engineering Software*, 33(7):517–526. 130

[Leon *et al.*, 2019] LEON, A., MUELLER, S., DE LUCA, P., SAID, R., DUVAL, J.-L. et CHINESTA, F. (2019). Non-intrusive proper generalized decomposition involving space and parameters : Application to the mechanical modeling of 3D woven fabrics. *Advanced Modeling and Simulation in Engineering Sciences*, 6(1):1–20. 3, 42

[Liu *et al.*, 1996] LIU, B., DAPENG, C. et YU, L. (1996). A non-incremental time-space algorithm for numerical simulation of forming process. *Applied Mathematics and Mechanics*, 17(11):1021–1029. 130

[Loève, 1955] LOÈVE, M. (1955). *Probability Theory : Foundations, Random Sequences*. Van Nostrand. 31

[Loiseau *et al.*, 2002] LOISEAU, O., LADEVÈZE, P. et DUREISSEIX, D. (2002). Sur une stratégie de calcul multiéchelle pour l'analyse des structures composites. *Revue Européenne des Éléments Finis*, 11(2-4):349–362. 131

[Luenberger et Ye, 2016] LUENBERGER, D. G. et YE, Y. (2016). *Linear and Nonlinear Programming*. Numéro 228 de International Series in Operations Research & Management Science. Springer International Publishing : Imprint : Springer, Cham, fourth édition. 24

[Lumley, 1967] LUMLEY, J. L. (1967). The structure of inhomogeneous turbulent flows. *Atmospheric Turbulence and Radio Wave Propagation*. 33

[Ma et Shen, 2021] MA, W. et SHEN, Y. (2021). A mixed formulation of proper generalized decomposition for solving the Allen-Cahn and Cahn-Hilliard equations. *Finite Elements in Analysis and Design*, 194:103560. 38

[Maday et Rønquist, 2002] MADAY, Y. et RØNQUIST, E. M. (2002). A reduced-basis element method. *Comptes Rendus Mathematique*, 335(2):195–200. 2

[Maday et Rønquist, 2004] MADAY, Y. et RØNQUIST, E. M. (2004). The Reduced Basis Element Method : Application to a Thermal Fin Problem. *SIAM Journal on Scientific Computing*, 26(1):240–258. 35

[Marsden et Hughes, 1984] MARSDEN, J. E. et HUGHES, T. J. R. (1984). *Mathematical Foundations of Elasticity*. Courier Corporation. Dover Publications, inc., New York. 14

[Matthies et Strang, 1979] MATTHIES, H. et STRANG, G. (1979). The solution of nonlinear finite element equations. *International Journal for Numerical Methods in Engineering*, 14(11):1613–1626. 24

[Montgomery, 2017] MONTGOMERY, D. C. (2017). *Design and Analysis of Experiments*. John Wiley & Sons, Hoboken, NJ, ninth édition. 2

[Nachar, 2019] NACHAR, S. (2019). *Optimisation de structures viscoplastiques par couplage entre métamodèle multi-fidélité et modèles réduits*. Thèse de doctorat, Ecole Normale Supérieure Paris-Saclay, Université Paris-Saclay. 28, 52, 130

[Nachar *et al.*, 2020] NACHAR, S., BOUCARD, P.-A., NÉRON, D., NACKENHORST, U. et FAU, A. (2020). Multifidelity Metamodels Nourished by Reduced Order Models. *In* WRIGGERS, P., ALLIX, O. et WEISSENFELS, C., éditeurs : *Virtual Design and Validation*, Lecture Notes in Applied and Computational Mechanics, pages 61–79. Springer International Publishing, Cham. 117, 128

[Naghdi, 1990] NAGHDI, P. M. (1990). A critical review of the state of finite plasticity. *Journal of Applied Mathematics and Physics*, 41(3):315–394. 16

[Najah *et al.*, 1998] NAJAH, A., COCHELIN, B., DAMIL, N. et POTIER-FERRY, M. (1998). A critical review of asymptotic numerical methods. *Archives of Computational Methods in Engineering*, 5(1):31–50. 38

[Néron *et al.*, 2015] NÉRON, D., BOUCARD, P.-A. et RELUN, N. (2015). Time-space PGD for the rapid solution of 3D nonlinear parametrized problems in the many-query context. *International Journal for Numerical Methods in Engineering*, 103(4):275–292. 3, 117, 130

[Néron et Dureisseix, 2008] NÉRON, D. et DUREISSEIX, D. (2008). A computational strategy for thermo-poroelastic structures with a time–space interface coupling. *International Journal for Numerical Methods in Engineering*, 75(9):1053–1084. 26, 131

[Néron et Ladevèze, 2010] NÉRON, D. et LADEVÈZE, P. (2010). Proper Generalized Decomposition for Multiscale and Multiphysics Problems. *Archives of Computational Methods in Engineering*, 17(4):351–372. 33, 35

[Newton, 1736] NEWTON, S. I. (1736). *The Method of Fluxions and Infinite Series*. 20

[Nguyen *et al.*, 2008a] NGUYEN, H.-M., ALLIX, O. et FEISSEL, P. (2008a). A robust identification strategy for rate-dependent models in dynamics. *Inverse Problems*, 24(6):065006. 131

[Nguyen *et al.*, 2008b] NGUYEN, N. C., PATERA, A. T. et PERAIRE, J. (2008b). A 'best points' interpolation method for efficient approximation of parametrized functions. *International Journal for Numerical Methods in Engineering*, 73(4):521–543. 39

[Niroomandi *et al.*, 2013] NIROOMANDI, S., ALFARO, I., GONZÁLEZ, D., CUETO, E. et CHINESTA, F. (2013). Model order reduction in hyperelasticity : A proper generalized decomposition approach. *International Journal for Numerical Methods in Engineering*, 96(3):129–149. 38

[Nocedal et Wright, 2006] NOCEDAL, J. et WRIGHT, S. J. (2006). *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering. Springer New York. 2, 20, 22

[Nouy, 2010] NOUY, A. (2010). A priori model reduction through Proper Generalized Decomposition for solving time-dependent partial differential equations. *Computer Methods in Applied Mechanics and Engineering*, 199(23):1603–1626. 37

[Nouy et Ladevèze, 2004] NOUY, A. et LADEVÈZE, P. (2004). Multiscale Computational Strategy With Time and Space Homogenization : A Radial-Type Approximation Technique for Solving Microproblems. *International Journal for Multiscale Computational Engineering*, 2(4). 37, 131

[Odièvre *et al.*, 2010] ODIÈVRE, D., BOUCARD, P.-A. et GATUINGT, F. (2010). A parallel, multiscale domain decomposition method for the transient dynamic analysis of assemblies with friction. *Computer Methods in Applied Mechanics and Engineering*, 199(21):1297–1306. 26, 130

[Oseledets, 2011] OSELEDETS, I. V. (2011). Tensor-Train Decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–2317. 30

[Oumaziz, 2017] OUMAZIZ, P. (2017). *Une méthode de décomposition de domaine mixte non-intrusive pour le calcul parallèle d'assemblages*. Thèse de doctorat, Ecole Normale Supérieure de Cachan, Université Paris-Saclay. 90

[Oumaziz *et al.*, 2017] OUMAZIZ, P., GOSSELET, P., BOUCARD, P.-A. et GUINARD, S. (2017). A non-invasive implementation of a mixed domain decomposition method for frictional contact problems. *Computational Mechanics*, 60(5):797–812. 3, 27, 93, 131

[Oumaziz *et al.*, 2021] OUMAZIZ, P., GOSSELET, P., SAAVEDRA, K. et TARDIEU, N. (2021). Analysis, improvement and limits of the multiscale Latin method. *Computer Methods in Applied Mechanics and Engineering*, 384:113955. 91

[Paillet *et al.*, 2018] PAILLET, C., NÉRON, D. et LADEVÈZE, P. (2018). A door to model reduction in high-dimensional parameter space. *Comptes Rendus Mécanique*, 346(7):524–531. 29

[Passieux *et al.*, 2010] PASSIEUX, J.-C., LADEVÈZE, P. et NÉRON, D. (2010). A scalable time–space multiscale domain decomposition method : Adaptive time scale separation. *Computational Mechanics*, 46(4):621–633. 91, 130

[Pearson, 1901] PEARSON, K. (1901). LIII. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572. 31

[Pelle et Ryckelynck, 2000] PELLE, J. P. et RYCKELYNCK, D. (2000). An efficient adaptive strategy to master the global quality of viscoplastic analysis. *Computers & Structures*, 78(1):169–183. 130

[Pietrzak et Curnier, 1999] PIETRZAK, G. et CURNIER, A. (1999). Large deformation frictional contact mechanics : Continuum formulation and augmented Lagrangian treatment. *Computer Methods in Applied Mechanics and Engineering*, 177(3):351–381. 22, 92

[Prager, 1949] PRAGER, W. (1949). Recent Developments in the Mathematical Theory of Plasticity. *Journal of Applied Physics*, 20(3):235–241. 12

[Prulière *et al.*, 2010] PRULIÈRE, E., CHINESTA, F. et AMMAR, A. (2010). On the deterministic solution of multidimensional parametric models using the Proper Generalized Decomposition. *Mathematics and Computers in Simulation*, 81(4):791–810. 36

[Quaranta *et al.*, 2020] QUARANTA, G., HAUG, E., DUVAL, J.-L. et CHINESTA, F. (2020). Parametric evaluation of part distortion in additive manufacturing processes. *International Journal of Material Forming*, 13(1):29–41. 129

[Quesada *et al.*, 2016] QUESADA, C., GONZÁLEZ, D., ALFARO, I., CUETO, E. et CHINESTA, F. (2016). Computational vademecums for real-time simulation of surgical cutting in haptic environments. *International Journal for Numerical Methods in Engineering*, 108(10):1230–1247. 33

[Quesada *et al.*, 2021] QUESADA, C., VILLON, P. et SALSAC, A.-V. (2021). Real-time prediction of the deformation of microcapsules using Proper Orthogonal Decomposition. *Journal of Fluids and Structures*, 101:103193. 34

[Radermacher et Reese, 2016] RADERMACHER, A. et REESE, S. (2016). POD-based model reduction with empirical interpolation applied to nonlinear elasticity. *International Journal for Numerical Methods in Engineering*, 107(6):477–495. 38

[Relun, 2011] RELUN, N. (2011). *Stratégie multiparamétrique pour la conception robuste en fatigue*. Thèse de doctorat, Ecole Normale Supérieure de Cachan, Université Paris-Saclay. 28

[Relun *et al.*, 2013] RELUN, N., NÉRON, D. et BOUCARD, P.-A. (2013). A model reduction technique based on the PGD for elastic-viscoplastic computational analysis. *Computational Mechanics*, 51(1):83–92. 3, 26, 27, 108, 130

[Ribeaucourt *et al.*, 2007] RIBEAUCOURT, R., BAIETTO-DUBOURG, M. C. et GRAVOUIL, A. (2007). A new fatigue frictional contact crack propagation model with the coupled X-FEM/LATIN method. *Computer Methods in Applied Mechanics and Engineering*, 196(33):3230–3247. 26

[Riks, 1972] RIKS, E. (1972). The Application of Newton's Method to the Problem of Elastic Stability. *Journal of Applied Mechanics*, 39(4):1060–1065. 24

[Rockafellar, 2015] ROCKAFELLAR, R. T. (2015). *Convex Analysis*. Princeton University Press. 20

[Rodriguez-Iturra *et al.*, 2021] RODRIGUEZ-ITURRA, S., NÉRON, D., LADEVÈZE, P., CHARBONNEL, P. E. et NAHAS, G. (2021). Latin-PGD multi-scale In time for complex fatigue problems. *In 14th World Congress on Computational Mechanics (WCCM XIV) - ECCOMAS Congress 2020*. 53

[Rougée, 1997] ROUGÉE, P. (1997). *Mécanique des grandes transformations*. Springer Science & Business Media. 15

[Roulet *et al.*, 2013] ROULET, V., BOUCARD, P.-A. et CHAMPANEY, L. (2013). An efficient computational strategy for composite laminates assemblies including variability. *International Journal of Solids and Structures*, 50(18): 2749–2757. 130

[Rowley *et al.*, 2004] ROWLEY, C. W., COLONIUS, T. et MURRAY, R. M. (2004). Model reduction for compressible flows using POD and Galerkin projection. *Physica D : Nonlinear Phenomena*, 189(1):115–129. 33

[Royer, 1990] ROYER, C. (1990). *Une Approche Des Problèmes de Dynamique Non-Lineaires Par La Méthode à Grand Incrément de Temps*. Thèse de doctorat, Université Pierre et Marie Curie - Paris 6. 26, 130

[Rozza, 2011] ROZZA, G. (2011). Reduced Basis Approximation and Error Bounds for Potential Flows in Parametrized Geometries. *Communications in Computational Physics*, 9(1):1–48. 35

[Rozza *et al.*, 2007] ROZZA, G., HUYNH, D. B. P. et PATERA, A. T. (2007). Reduced basis approximation and a posteriori error estimation for affinely parametrized elliptic coercive partial differential equations. *Archives of Computational Methods in Engineering*, 15(3):229–275. 35

[Rubio *et al.*, 2019] RUBIO, P.-B., CHAMOIN, L. et LOUF, F. (2019). Real-time Bayesian data assimilation with data selection, correction of model bias, and on-the-fly uncertainty propagation. *Comptes Rendus Mécanique*, 347(11):762–779. 35

[Ruda *et al.*, 2021] RUDA, A., LOUF, F., BOUCARD, P.-A. et MININGER, X. (2021). First approach of a mixed domain decomposition method for magneto-static simulation of rotating machines. *In 12th International Symposium on Electric and Magnetic Fields*, online, France. 53

[Ryckelynck, 2005] RYCKELYNCK, D. (2005). A priori hyperreduction method : An adaptive approach. *Journal of Computational Physics*, 202(1):346–366. 39

[Saavedra *et al.*, 2017] SAAVEDRA, K., ALLIX, O., GOSSELET, P., HINOJOSA, J. et VIARD, A. (2017). An enhanced nonlinear multi-scale strategy for the simulation of buckling and delamination on 3D composite plates. *Computer Methods in Applied Mechanics and Engineering*, 317:952–969. 130

[Scanff *et al.*, 2021] SCANFF, R., NACHAR, S., BOUCARD, P.-A. et NÉRON, D. (2021). A Study on the LATIN-PGD Method : Analysis of Some Variants in the Light of the Latest Developments. *Archives of Computational Methods in Engineering*, 28(5):3457–3473. 27, 54

[Sen Gupta *et al.*, 2006] SEN GUPTA, J., ALLIX, O., BOUCARD, P.-A. et FANGET, A. (2006). Mesodynamics of a 3D C/C : A dedicated numerical strategy. *Computers & Structures*, 84(19):1177–1189. 130

[Simo et Laursen, 1992] SIMO, J. C. et LAURSEN, T. A. (1992). An augmented lagrangian treatment of contact problems involving friction. *Computers & Structures*, 42(1):97–116. 22

[Trollé *et al.*, 2012] TROLLÉ, B., GRAVOUIL, A., BAIETTO, M. C. et NGUYEN-TAJAN, T. M. L. (2012). Optimization of a stabilized X-FEM formulation for frictional cracks. *Finite Elements in Analysis and Design*, 59:18–27. 131

[Trovalet *et al.*, 2009] TROVALET, M., LADEVÈZE, P. et LUBINEAU, G. (2009). A micro model for analysis of laminated composites, improvement and illustrations (in French). *In* LAMON, P. O. e. J., éditeur : *JNC 16*, pages 1–9, France. AMAC. 130

[Tsiolakis *et al.*, 2020] TSIOLAKIS, V., GIACOMINI, M., SEVILLA, R., OTHMER, C. et HUERTA, A. (2020). Nonintrusive proper generalised decomposition for parametrised incompressible flow problems in OpenFOAM. *Computer Physics Communications*, 249:107013. 4, 43

[Tucker, 1966] TUCKER, L. R. (1966). Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311. 30

[Valanis, 1978] VALANIS, K. C. (1978). Fundamental Consequences of a New Intrinsic Time Measure. Plasticity as a Limit of the Endochronic Theory. Rapport technique G-224/DME-78-01, Iowa University, Iowa City. 12

[Vandoren *et al.*, 2013] VANDOREN, B., DE PROFT, K., SIMONE, A. et SLUYS, L. (2013). A novel constrained LArge Time INcrement method for modelling quasi-brittle failure. *Computer Methods in Applied Mechanics and Engineering*, 265:148–162. 128, 130

[Verwaerde *et al.*, 2021] VERWAERDE, R., GUIDAULT, P.-A. et BOUCARD, P.-A. (2021). A non-linear finite element connector model with friction and plasticity for the simulation of bolted assemblies. *Finite Elements in Analysis and Design*, 195:103586. 16

[Violeau *et al.*, 2009] VIOLEAU, D., LADEVÈZE, P. et LUBINEAU, G. (2009). Micromodel-based simulations for laminated composites. *Composites Science and Technology*, 69(9):1364–1371. 130

[Vitse, 2016] VITSE, M. (2016). *Model-Order Reduction for the Parametric Analysis of Damage in Reinforced Concrete Structures*. Thèse de doctorat, Ecole Normale Supérieure de Cachan, Université Paris-Saclay. 52

[Vitse *et al.*, 2019] VITSE, M., NÉRON, D. et BOUCARD, P.-A. (2019). Dealing with a nonlinear material behavior and its variability through PGD models : Application to reinforced concrete structures. *Finite Elements in Analysis and Design*, 153:22–37. 130

[Vizzaccaro *et al.*, 2020] VIZZACCARO, A., GIVOIS, A., LONGOBARDI, P., SHEN, Y., DEÜ, J.-F., SALLES, L., TOUZÉ, C. et THOMAS, O. (2020). Non-intrusive reduced order modelling for the dynamics of geometrically nonlinear flat structures using three-dimensional finite elements. *Computational Mechanics*. 42

[Wriggers, 1995] WRIGGERS, P. (1995). Finite element algorithms for contact problems. *Archives of Computational Methods in Engineering*, 2(4):1–49. 22

[Wriggers, 2008] WRIGGERS, P. (2008). *Nonlinear Finite Element Methods*. Springer Science & Business Media. 20, 23, 24

[Wriggers et Zavarise, 2004] WRIGGERS, P. et ZAVARISE, G. (2004). Computational Contact Mechanics. *In Encyclopedia of Computational Mechanics*, chapitre 6. American Cancer Society. 16, 21, 22

[Wurtzer, 2021] WURTZER, F. (2021). Stratégie de couplage physiques/données pour la construction d'un jumeau numérique. Mémoire de Master, Ecole Normale Supérieure Paris-Saclay. 53

[Yano et Patera, 2019] YANO, M. et PATERA, A. T. (2019). An LP empirical quadrature procedure for reduced basis treatment of parametrized nonlinear PDEs. *Computer Methods in Applied Mechanics and Engineering*, 344:1104–1123. 39

[Yastrebov, 2013] YASTREBOV, V. A. (2013). *Numerical Methods in Contact Mechanics*. Numerical Methods in Engineering Series. ISTE ; Wiley, London : Hoboken, NJ. 16

[Zienkiewicz et Taylor, 2005] ZIENKIEWICZ, O. C. et TAYLOR, R. L. (2005). *The Finite Element Method for Solid and Structural Mechanics*. Elsevier, New York. 4, 19, 23

[Zimmermann et Willcox, 2016] ZIMMERMANN, R. et WILLCOX, K. (2016). An Accelerated Greedy Missing Point Estimation Procedure. *SIAM Journal on Scientific Computing*, 38(5):A2827–A2850. 39

[Zou *et al.*, 2018] ZOU, X., CONTI, M., DÍEZ, P. et AURICCHIO, F. (2018). A nonintrusive proper generalized decomposition scheme with application in biomechanics. *International Journal for Numerical Methods in Engineering*, 113(2):230–251. 3, 4, 33, 43

université PARIS-SACLAY

ÉCOLE DOCTORALE
Sciences mécaniques et énergétiques, matériaux et géosciences (SMEMAG)

Titre : Une vision faiblement intrusive de la méthode LATIN-PGD en non-linéaire

Mots clés : Non-linéaire, Faiblement intrusif, Modèles réduits, LATIN-PGD, Logiciel industriel

Résumé : Portées par une digitalisation omniprésente et grandissante, comme en témoigne l'intérêt porté aux jumeaux numériques, les méthodes de réduction de modèles (ROM) sont de plus en plus convoitées par l'ensemble du secteur industriel, qui souhaite bénéficier d'outils novateurs et de techniques robustes pour réduire drastiquement les temps de calcul. Cependant, le caractère particulièrement intrusif de ces méthodes, ainsi que leur robustesse, représentent encore deux obstacles majeurs qui entravent la diffusion des méthodes ROM dans les logiciels industriels. Cette thèse propose une version faiblement intrusive de la méthode LATIN-PGD destinée à construire et enrichir des modèles réduits nativement au sein de n'importe quel logiciel industriel par éléments finis généraliste. En considérant des problèmes non-linéaires paramétrés dépendant du temps en mécanique du solide, et sous hypothèse de quasi-staticité, nous proposons une démarche générique permettant de combiner la PGD avec toutes les non-linéarités usuellement rencontrées dans les logiciels commerciaux – matériaux, géométriques et contact. Cette manière de procéder conduit à la conception d'outils unifiés – pour la construction de modèles réduits non-linéaires – tous intégrés au sein d'un seul et unique logiciel certifié, en cohérence avec les besoins des ingénieurs des bureaux d'études au quotidien. Grâce à la première implémentation réalisée au sein du logiciel pilote Simcenter Samcef™ de Siemens Digital Industries Software, nous soulignons les atouts de cette approche sans jamais redévelopper aucune partie non-linéaire du logiciel. Enfin, sur la base de deux cas-tests industriels, présentant diverses non-linéarités et plusieurs paramètres, nous mettons en lumière des gains intéressants en termes de performance.

Title : A weakly-invasive LATIN-PGD method in the non-linear content

Keywords : Non-linear, Weakly-invasive, Reduced-order models, LATIN-PGD, Industrial software

Abstract : Driven by a high digitalization context with Digital Twins, reduced-order modeling (ROM) methods are becoming increasingly coveted by the industrial sector, keen to benefit from new tools and robust techniques that allow faster computations. However, the spread of these ROM methods within commercial finite element software is currently hindered by two main obstacles : non-intrusiveness and robustness. This arises directly from the nature of the associated algorithms involving many unconventional operations and thus preventing from getting robust and reliable tools all integrated into one certified product. This thesis introduces a weakly-invasive reformulation of the LATIN-PGD method designed to compute and improve reduced-order models directly into any general-purpose industrial finite element software. Under the quasi-static assumption, we assume any time-dependent non-linear parametrized problems in solid mechanics. The guiding thought we propose to pursue is to combine PGD with all facilities offered by such software, implying *de facto* the ability to handle any non-linearities – materials, contacts, large transformations. This way of proceeding provides unified tools – for the construction of reduced-order models in non-linear context – all integrated into one certified product in consistency to have end-to-end processes. Thanks to the first implementation conducted within Simcenter Samcef™ pilot software owned by Siemens Digital Industries Software, we emphasize the assets of this approach without ever redeveloping any non-linear part of the software. Finally, attractive performance gains are achieved on some non-linear time-dependent industrial test-cases involving some parameters.