Conception et analyse d'algorithmes parallèles en temps pour l'accélération de simulations numériques d'équations d'évolution

Mohamed-Kamel Riahi

Université Pierre et Marie Curie – Paris 6 Laboratoire Jacques-Louis Lions – UMR 7598

Conception et analyse d'algorithmes parallèles en temps pour l'accélération de simulations numériques d'équations d'évolution

THÈSE DE DOCTORAT

présentée et soutenue publiquement le 10 Juillet 2012

pour l'obtention du

Doctorat de l'université Pierre et Marie Curie – Paris 6

Spécialité Mathématiques Appliquées

par

Mohamed Kamel Riahi

Composition du jury

Président :	Frédérique Hecht :	Professeur Lab. Jacques-Louis Lions Université Pierre et Marie Curie
Rapporteurs :	Victorita Dolean Maini	Maître de conférence HdR à l'université de Nice-Sofia Antipolis
	Stefan Vandewalle	Professeur de l'université Catholique, Leuven.
Examinateurs :	Yvon Maday	Directeur de thèse: Professeur Lab. Jacques-Louis Lions Université Pierre et Marie Curie
	Julien Salomon	Co-directeur de thèse: Maître de conférence HdR Lab. CEREMADE, Université Paris Dauphine.
	Gabriel Turinici	Professeur Lab. CEREMADE, Université Paris Dauphine.
	Adel Blouza	Maître de conférence HdR Laboratoire de Mathématiques Raphaël Salem, Université de Rouen.

École Doctorale de Sciences Mathématiques de Paris Centre UFR 929 - Mathématiques

Remerciements

Je tiens tout d'abord à exprimer ma profonde gratitude et ma sincère reconnaissance à Yvon Maday qui m'a encadré pendant cette thèse. Je le remercie pour sa grande disponibilité, sa patience, la confiance qu'il m'a donnée, ses précieux conseils et son optimisme toujours contagieux.

J'aimerais également exprimer ma sincère reconnaissance à Julien Salomon, mon co-directeur de thèse, qui m'a toujours soutenu depuis mon stage de Master. Je le remercie pour son encadrement et pour tout ce que j'ai pu apprendre auprès de lui, grâce à sa générosité, sa bonne humeur et surtout la confiance et la liberté qu'il m'a données à mes balbutiements.

Je remercie également les rapporteurs, M. Stefan Vandewalle et Mme Victorita Dolean-Maini d'avoir accepté de rapporter ma thèse et je les remercie du temps qu'ils ont consacré à la lecture de mon travail ainsi que les remarques constructives qu'ils m'ont faites.

Un très grand merci à Frédéric Hecht non seulement parce qu'il a accepté d'être un membre de mon jury, mais aussi pour avoir créé FreeFem++ et continuer à le développer. Ce logiciel m'a permis d'appliquer les théories mathématiques sur des problèmes réels très complexes.

Je tiens à remercier également très chaleureusement Gabriel Turinici qui fait partie de mon jury, et qui a encadré mon stage de fin d'études de Master Recherche au sein du projet MICMAC à l'INRIA-Rocquencourt, et qui m'a beaucoup appris durant ce stage.

Je tiens à remercier profondément Adel Blouza, pour tous ses divers conseils, ainsi que les discussions scientifiques qu'on a eu. Je le remercie aussi d'avoir accepté d'examiner mon travail et de faire partie de mon jury de thèse.

Je remercie également mes collaborateurs Anne-Marie Bourdon, Jean-Jacques Lautard et Dominique Sugny pour leurs précieux conseils et remarques qu'ils m'ont donnés. Je les remercie aussi pour leur disponibilité et l'aides apportée aux difficultés que j'ai confrontées au cours de mon travail.

Je remercie sincèrement Sidi Mahmoud Kaber et Faker Ben Belgacem de m'avoir fait confiance et de m'avoir donné récemment une place dans leur équipe de recherche sur les problèmes mal posés.

Un grand merci à A. Lehyaric pour ses précieux conseils en informatique scientifique et pour les discussions que nous avons eues sur le parallélisme des codes séquentiels et le post-traitement des données, notamment avec le Bash/Perl et Python.

Je remercie également mes collègues d'enseignement du cours d'Analyse Numérique et calcul scientifique à l'Université Paris Dauphine. Je pense en particulier à Matthieu Hillairet avec qui j'ai eu le grand plaisir de travailler. Je pense aussi à Ibtihel Ben-Garbia ma collègue et amie.

Merci à tous ceux qui ont pris le temps de relire quelques pages ou chapitres de ce manuscrit : Jean-Paul pour l'aide apportée à mes périphrases en Français, Khaled le sniper des « s » du pluriels, Ayman et Benjamin pour la mise en forme et leurs conseils précieux, sans oublier Etienne, Lise-Marie, William et Pierre.

Toute ma gratitude envers mes camarades doctorants, post doctorants et permanents du laboratoire Jacques-Louis Lions pour l'intérêt qu'ils m'ont porté durant ces années de

thèse et pour les échanges scientifiques que nous avons eus. Je pense en particulier à Rim, Moheddine, Riadh, Faouzia, Tej-eddin, Gailen, Mohamed, Jamel, Peipei, Ange, Abdalla. Andres,Tobias, Malik, Mamadou, Laurent, Imen, Lamjed, Saber, Sonia, Kirill, Alexandra, Pierre, Nicole, Marie, Olga, Hassan, Sepideh, Alexander, Nicolas, Laurent, Jean-Baptiste, et aussi à Salvatore, j'espère ne pas en avoir oublié !

Un grand merci à Edwige Godlewski la co-directeur du laboratoire Jacques-Louis Lions et à tous les membres du personnel administratif du Labo pour toute leur aide vraiment précieuse. En particulier Mesdames Danielle Boulic, Liliane Ruprecht, Salima Lounissi et Florence Saidani. Sans oublier bien sûr M. Christian David pour ses services rapides et de qualité, toujours accompagnés d'un grand sourire. Merci également à Khashayar Dadras pour son aide immense et son secours rapide afin que tout fonctionne parfaitement. Je n'oublie pas Pascal Joly et Philippe Parnaudeau pour leur aide, notamment sur la nouvelle machine SGI du laboratoire. Je pense aussi à Robert pour les discussions très intéressantes sur les relations entre les industriels et le monde de la recherche.

Une pensée particulière également à Tahar Boulmezaoud pour les discussions sympathiques et les divers conseils qu'il m'a donnés. Je pense aussi à Frédéric Coquel pour son aide, ses conseils et ses encouragements incessants. Merci également à Saloua Aouadi pour ses accueils chaleureux et ses encouragements à chaque fois que je l'ai rencontrée à la faculté des science de Tunis, à Hamdi Zorgati également, et à tous ceux qui m'ont soutenu tout le long de ma thèse.

Merci de tout cœur à **ma famille**, qui malgré la distance, m'a toujours encouragé. Je pense bien sûr en particulier à **mes parents** et tous les sacrifices qu'ils ont faits pour moi.
Enfin, je ne remercierai jamais assez mon épouse **Asma** pour son soutien permanent, la confiance solide qu'elle me donne et tous les moments de bonheur qu'elle m'apporte.

À mes très chers parents,
à ma chère épouse Asma, et à
la mémoire de ma chère grand mère CHARIFA.

Table des matières

Chapitre 1

Introduction Générale

Sommaire

1.1 Calcul intensif, décomposition et parallélisation des tâches

Durant les dernières décennies, les mathématiques appliquées et le calcul scientifique en particulier, ont connu un développement important et sont maintenant des outils usuels aussi bien dans les communautés (non mathématiques) d'application que dans le monde industriel. Le calcul scientifique, discipline regroupant la modélisation, l'analyse numérique et la simulation, est en évolution croissante vers des applications scientifiques très avancées ou proprement ingénieuses.
On trouve de multiples champs disciplinaires d'application se servant intensivement du calcul scientifique. On peut citer la biologie, l'astrophysique, la chimie quantique, la météorologie ou encore des branches de l'industrie telles que l'énergie, l'automobile ou l'aéronautique.

Les modèles mathématiques utilisés dans ces divers domaines d'applications sont souvent très complexes et font intervenir un grand nombre de paramètres lors de leurs formulations mathématiques. La simulation numérique de ces problèmes a pour objectif principal d'approcher le mieux possible le comportement physique du problème à travers des modèles discrétisés. Ces derniers forment des systèmes matriciels d'autant plus grands que le problème est complexe et que la précision demandée lors du traitement est importante.
Dans la majorité des simulations sur ordinateurs ordinaires, on ne parvient pas à résoudre des problèmes complexes en des temps raisonnables. On risque même de saturer ce type de machines par la simple demande de traitement des données qui nécessitent l'occupation d'une très grande place mémoire !

Avec les demandes actuelles, les machines ordinaires sont surexploitées dans la plus part des domaines d'applications du calcul scientifique. Elles montrent leurs limites malgré le développement et l'évolution de leurs vitesses très élevées de résolution qui se mesurent en *flops*[1].
Ce recours massif à l'informatique impose de disposer d'ordinateurs très puissants qui offrent un accès à une importante mémoire vive de stockage temporel lors de l'assimilation des données. Les ordinateurs parallèles, également appelés superordinateurs, disposent de la puissance et de l'espace mémoire nécessaires à certaines applications complexes. Ils offrent donc aux scientifiques un nouvel outil de calcul, avec des possibilités et avantages très largement supérieurs à ceux des ordinateurs conventionnels. Cependant, les algorithmes implémentés sur les ordinateurs parallèles, sont nettement différents de ceux que l'on met en œuvre sur des machines ordinaires séquentielles (uni-processeurs).
Pour ces raisons, et sans parler des contraintes économiques et techniques, les scientifiques et notamment les numériciens, sont invités à développer des algorithmes (parallèles) s'adaptant à ce type de nouveaux calculateurs massivement parallèles. Les algorithmes doivent privilégier l'accès simultané à plusieurs processeurs et aussi à plus de ressources mémoires.
Les méthodes développées à ce titre sont nombreuses. Elles sont principalement basées sur

1. Nombre d'opérations arithmétiques par seconde.

la notion de subdivision des tâches que l'on alloue à chacun des processeurs disponibles de la machine parallèle. C'est ainsi que l'efficacité des algorithmes parallèles est fortement liée au degré d'indépendance de chaque subdivision relativement aux autres. Il est préférable que la dépendance soit faible. En effet, ceci réduit les communications inter-processeurs qui peuvent être lourdes sur certaines architectures parallèles, notamment celles qui utilisent un réseau Ethernet.

Au niveau de la conception de l'algorithme parallèle, une attention particulière doit être portée à la communication inter-processeurs, inévitable dans les applications scientifiques. Dans un souci d'efficacité, cette communication inter-processeurs doit en effet s'effectuer en un temps strictement inférieur à celui de la différence entre le temps pris par la résolution séquentielle (sur ordinateur ordinaire uni-processeur) et celui de la résolution sur un des processeurs employés parallèlement. Sur ce point précis, il faut se poser la question suivante : la parallélisation de la résolution du problème est-elle nécessaire ?
En effet, il n'est pas question de sous-exploiter un superordinateur. Dans le cas où le problème n'est pas assez grand (au sens des données de simulation et de la taille du système matriciel relatif), une décomposition judicieuse du traitement permet de réduire la complexité sur chaque subdivision du problème initial. Cependant, une majeure partie de la tâche parallèle sera absorbée par les communications inter-processeurs. Ceci peut rendre la simulation sur des ordinateurs séquentiels plus rapide que sur une machine parallèle, car les vitesses de traitement de ces machines séquentielles ont de nos jours beaucoup augmentées. Par exemple, pour la résolution d'équations aux dérivées partielles (EDP) avec des techniques de décomposition de domaine, le domaine spatial du calcul est décomposé en sous-domaines de calculs de petite taille. L'EDP originale (qui est définie sur le grand domaine) est ensuite résolue sur chacun des sous-domaines, et il est alors nécessaire de définir de façon appropriée les conditions aux bords de chaque sous-domaine. Ceci permet des résolutions locales tout à fait indépendantes et simultanées.

La plupart des algorithmes de décomposition de domaine peuvent être interprétés comme une méthode de point fixe agissant sur ces états de bords inconnus afin de les découvrir. L'efficacité des meilleurs algorithmes de décomposition de domaine, basés sur une mise à jour des conditions de frontière internes, est telle qu'elle mène à la convergence prévue en un nombre d'itérations dépendant seulement marginalement du nombre de sous-domaines impliqués dans la décomposition totale (grâce à l'utilisation des préconditionneurs appropriés de grille grossière) et des paramètres de discrétisation (par exemple la taille du maillage spatial). Idéalement, les meilleures procédures de résolution des EDP à grande échelle par des techniques de parallélisation par décomposition de domaine, fournissent un taux de scalabilité [2] approximativement proche du nombre de sous-domaines.
Pour des problèmes d'évolution basés sur des EDP, la parallélisation par rapport à la direction temporelle n'a pas suscité beaucoup d'attention comparativement à la décomposition de domaine. Celle-ci permet pourtant de réduire significativement la complexité algorithmique de résolution de grands systèmes, où la décomposition de domaine ne suffit plus à accélérer la résolutions.

2. En Anglais "Speed-up" : c'est le rapport entre les temps d'exécution séquentiel et parallèle.

La modélisation des mécanismes agissant dans la nature aboutit souvent à des équations différentielles dépendantes du temps. Ces équations, appelées également équations d'évolution, décrivent l'état des systèmes qu'elles gouvernent à travers une dérivée en temps.
On trouve dans la littérature plusieurs contributions traitant la parallélisation de la résolutions des EDP et EDO. Nous renvoyons à l'ouvrage [14] qui traite d'une manière synthétique ce sujet, dont il présente et discute les diverses techniques. Nous renvoyons également à l'ouvrage [5] pour plus de détails sur les aspects informatiques liés aux techniques de parallélisation.
Considérons le problème de Cauchy bien posé suivant :

$$\dot{y}(t) = \mathcal{A}(t, y), \quad y(0) = y_0,$$

où $\mathcal{A}$ est un opérateur d'un espace de Hilbert E vers son dual E'.
Les techniques de parallélisation de cette équation peuvent être classées en trois catégories :

- Parallélisation à travers le système matriciel : cette classe de méthodes consiste à diviser le second membre en différentes parties, et à attribuer à chacun des processeurs disponibles la résolution de chacune de ces parties. C'est l'approche la plus efficace quand le système est de très grande taille. Cette technique est régulièrement utilisée en simulation moléculaire et gravitationnelle par exemple. Des techniques issues du traitement graphique (GPU) peuvent être employées pour fournir un équilibre approprié entre les charges des tâches de calcul sur chaque processeur. Une alternative plus récente dans la même catégorie est présentée dans [44].
- Parallélisation à travers la méthode : cette méthode, comme son nom l'indique d'ailleurs, est directement liée au schéma numérique mis en place pour résoudre l'équation. Son efficacité dépend fortement des caractéristiques du schéma, mais les recherches dans cette direction ont mené à concevoir des méthodes parallèles complètement nouvelles. Par exemple, la méthode de Runge-Kutta-partitionnée [22] offre des possibilités de parallélisation à travers la méthode de Runge-Kutta. Cependant cette méthode possède un taux faible de parallélisation. La recherche est encore en développement dans cette direction.
- Parallélisation à travers le temps : cette parallélisation consite à subdiviser l'intervalle de temps et à résoudre simultanément un problème défini sur chaque sous-intervalle. La principal difficulté est de fournir la bonne condition initiale sur chaque sous-intervalle. La plupart du temps, toute technique dans cette catégorie est une technique multi-tir [3] partant des travaux précurseurs [11] sur le parallélisation à travers le temps. Il en est ainsi des méthodes de relaxation de forme d'onde [36] et des approches multigrilles [25] par exemple.

La simulation de telles équations d'évolution semble par nature proprement séquentielle car la propagation de l'information sur la durée $t' - t$ est nécessaire dans la détermination de l'état à l'instant t'. Néanmoins, ceci n'est plus la seule voie permettant d'aboutir à des informations sur l'état du système à des temps avancés (dans son évolution). La publication de l'algorithme pararéel a donné une autre dimension à la simulation sur ordinateurs des équations d'évolution, une simulation tout à fait parallèle permettant une accélération significative des temps de résolution. Ce nouveau schéma parallèle a fait l'objet de

3. En anglais : multishooting

plusieurs études, portant notamment sur sa convergence et sa stabilité [49, 8, 18, 21, 20]. Plusieurs formulations et identifications de ce schéma pararéel ont été également développées : Farhat et ces collaborateurs dans [19] donnent un schéma analogue à l'algorithme pararéel, Gander et Vandewalle dans [21] identifient l'algorithme avec une méthode de multi-tir ainsi qu'avec une méthode multi-grille en temps. Une autre interprétation de l'algorithme est présentée dans [49] où Maday et Turinici interprètent le pararéel comme un pré-conditionneur dans le cadre du problème de contrôle virtuel optimal.

1.2 L'algorithme pararéel en temps : un algorithme efficace pour la parallélisation en temps

Depuis sa publication [41], l'algorithme pararéel a fait l'objet de plusieurs études. Les numériciens ont cherché à l'adapter numériquement et à l'appliquer à divers problèmes. Il offre en effet une capacité importante de parallélisation des problèmes d'évolution qui semblaient intrinsèquement séquentiels et non-parallélisables. Bien qu'il y ait eu des tentatives de parallélisation des problèmes d'évolutions à partir des années 90 [10, 11], celles-ci n'ont pas été exhaustives. L'algorithme pararéel tel qu'il a été présenté par Lions et al [41] et sous une version améliorée avec de nouveaux résultats de convergence [49], consiste en une méthode assez générale de parallélisation, des EDP et des EDO, pouvant être couplée avec d'autres méthodes itératives.
Cet algorithme a été testé sur plusieurs équations pour lesquelles il s'est avéré très performant et robuste. Par exemple, sur des problèmes paraboliques linéaires et non linéaires [65, 9] pour lesquels l'algorithme pararéel tire profit de l'aspect dissipatif de ce type d'équations : celui-ci rectifie certaines instabilités de la solution grossière. L'algorithme a aussi été testé sur d'autres type d'EDP pour des application en mathématiques financières telle que le calcul d'une option de vente Américaine [9]. Il a également prouvé son efficacité dans les domaines de la dynamique moléculaire [6] et de la cinétique en chimie [12].
L'algorithme pararéel a également montré son efficacité lors du couplage avec d'autres types de méthodes itératives en accélérant la résolution des équations sur lesquelles il a été appliqué. Dans cet optique, nous distinguons deux classes : le couplage avec des méthodes de décomposition de domaine et le couplage avec des problèmes de contrôle optimal. Nous citons, dans le cadre de la décomposition de domaine, les travaux précurseurs [49] dont l'objet est le couplage du pararéel avec une méthode de décomposition de domaine sans recouvrement. Aussi nous citons les travaux de Maday et Gueta dans le cadre d'une thèse de doctorat [43] et dans Aouadi, Maday et Gueta (à paraître) ainsi que les travaux de Minion [53].
Nous renvoyons à [45,47,62] pour le traitement du couplage du pararéel avec des problèmes de contrôle optimal.

1.3 Modèles traités dans la thèse

Dans cette thèse nous proposons des algorithmes parallèles dans trois domaines différents. Les trois parties correspondantes, sont complètement indépendantes.

1.3.1 Problème de contrôle optimal parabolique

Dans le cadre du projet de recherche ANR "PITAC" (Parallélisation Incluant le Temps pour Accélérer le Calcul), nous nous sommes intéressés à la parallélisation du problème de contrôle optimal d'un système régi par une EDP parabolique. Dans cette optique, nous nous inspirons de quelques techniques de parallélisation telles que le contrôle virtuel [38] (voir aussi [42, 23, 39, 40]) utilisé par Maday et Turinici [47] à travers l'introduction des sauts dans un terme quadratique dans la fonctionnelle de coût. Cette introduction des sauts a mené à l'utilisation de l'algorithme pararéel après avoir constaté son caractère préconditionneur vis-à-vis de l'équation d'optimalité d'une certaine fonctionnelle de coût. Une autre approche élégante est présentée dans [45] dans le cadre du contrôle de l'équation de Schrödinger. Cette méthode consiste à définir des points intermédiaires servant à la fois de conditions initiales et des cibles (comme c'est le cas d'ailleurs dans [47]), pour des sous-problèmes définis sur chacun des sous-intervalles de la décomposition. Ces deux dernières approches constituent les bases de la méthode de parallélisation en temps pour le contrôle optimal parabolique développé dans cette thèse. Nous considérons en effet une subdivision de l'intervalle de temps avec la définition suivante des conditions initiales λ et cibles ξ intermédiaires :

$$\lambda := (\lambda_n)_{n\geq 0} := y, \quad \text{et} \quad \xi := (\xi_n)_{n\geq 0} := y - p,$$

où y et p sont respectivement l'état direct et son adjoint dans le problème de minimisation. Ces définitions nous mènent aux sous-problèmes suivants :

$$\begin{cases} \partial_t y_n - \nu \Delta y_n = \mathcal{B} v_n & \text{sur } I_n \times \Omega \\ y(t_n^+) = \lambda_n \end{cases} \tag{1.1}$$

$$\begin{cases} -\partial_t p_n - \nu \Delta p_n = 0 & \text{sur } I_n \times \Omega \\ p_n(t_{n+1}^-) = y(t_{n+1}^-) - \xi_{n+1} \\ \alpha v_n + \mathcal{B}^* p_n = 0 & \text{sur } I_n \times \Omega, \end{cases} \tag{1.2}$$

où $\mathcal{B}^*$ est l'opérateur adjoint de $\mathcal{B}$. Ces sous-problèmes sont indépendants et peuvent être résolus simultanément.

1.3.2 Problème de neutronique : Simulation de la cinétique des populations de neutrons au sein d'un réacteur nucléaire

Dans le cadre du projet LRC Manon (Laboratoire de Recherche Conventionné Modélisation et approximation numérique orientées pour l'énergie nucléaire), et en collaboration avec des ingénieurs du CEA, nous nous sommes intéressés à l'application et l'adaptation de l'algorithme pararéel aux équations de la cinétique neutronique. Ces équations de grande taille doivent bénéficier d'une accélération afin de prévoir les cas accidentels au sein du réacteur nucléaire lors de la production d'électricité qui est basée sur une réaction en chaîne de fission des noyaux d'uranium. Nous simulons en trois dimensions un modèle réaliste avec deux groupes d'énergie de flux neutronique (indexés par g) et six groupes de concentrations de précurseurs (indexés par k). En outre, il faut prendre en compte la variation de la géométrie du domaine, due au mouvement vertical et continu des grappes

d'absorption. Celles-ci constituent un moyen de contrôle de la puissance produite au sein du cœur nucléaire. Le modèle s'écrit :

$$\begin{aligned}\frac{1}{v^{(g)}}\frac{\partial}{\partial t}\phi^{(g)}(\vec{r},t) = \quad & \mathrm{div}(d(\vec{r},t)\nabla\phi^{(g)}(\vec{r},t)) - \sigma_t^{(g)}\phi^{(g)}(\vec{r},t) \\ & + \textstyle\sum_{g'=1}^{\hat{g}} \sigma_s^{(g'\to g)}\phi^{(g')}(\vec{r},t) \\ & + \chi_p^{(g)}(\vec{r}) \textstyle\sum_{g'=1}^{\hat{g}}(1-\beta^{(g')})\nu^{(g)}(\vec{r})\sigma_f^{(g')}\phi^{(g')}(\vec{r},t) \\ & + \textstyle\sum_{k=1}^{\hat{k}} \chi_d^{(k,g)}(\vec{r})\mu^{(k)}c^{(k)}(\vec{r},t),\end{aligned}$$

$$\frac{\partial}{\partial t}c^{(k)}(\vec{r},t) = \mu^{(k)}c^{(k)}(\vec{r},t) + \sum_{g'=1}^{\hat{g}} \beta^{(k,g)}\nu^{(g')}(\vec{r},t)\sigma_f^{(g')}\phi^{(g')}(\vec{r},t).$$

1.3.3 Problème de mécanique quantique : contrôle optimal en résonance magnétique nucléaire

Dans le domaine de la spectroscopie par résonance magnétique nucléaire (RMN) les outils du contrôle optimal jouent un rôle fondamental dans l'obtention d'états physiques particuliers. Les outils mathématiques d'optimisation, typiquement les méthodes d'optimisation itératives, sont cruciaux pour affiner les expériences sur les instruments physiques. Dans le cadre d'une récente collaboration avec des physiciens, nous nous intéressons à un modèle de 5 spins couplés en phase liquide et qui sont ciblés par des champs magnétiques de contrôle. Ces champs de contrôle possèdent deux directions. Cela signifie que notre modèle quantique possède 10 composantes de contrôle. Ce modèle, riche en information, est assez complexe à résoudre. Sa résolution, ordinairement séquentielle, prend énormément de temps. Pour accélérer celle-ci, nous proposons une méthode de parallélisation [45] d'un problème de contrôle optimal associé à un algorithme monotone pour approcher la solution de l'état u des cinq-qbits qui vérifie l'équation de Schrödinger :

$$\partial_t u(t,x,y,z) = Hu(t,x,y,z)$$

Dans H interviennent 5 champs magnétiques ayant la forme suivante :

$$\omega_x(t)\vec{x}^{\varrho} + w_y(t)\vec{y}^{\varrho},$$

où ϱ désigne l'atome de l'échantillon parmi H, P, C, N et F. Les travaux à ce stade sont en cours de développent, cependant les résultats préliminaires d'ouverture montre la pertinence de l'algorithme.

1.4 Plan du manuscrit

Le travail présenté dans cette thèse propose ainsi quelques contributions à la parallélisation en temps de certains problèmes en liaison étroite avec les équations d'évolution et à l'algorithme pararéel.
Après l'introduction, nous présentons dans le **deuxième chapitre** une étude de la parallélisation en temps d'un problème de contrôle optimal d'un système régi par une équation

aux dérivées partielles parabolique avec un terme source. L'étude de parallélisation repose principalement sur l'introduction d'une nouvelle méthode des cibles et de conditions initiales intermédiaires, qui permet une définition de sous-problèmes de contrôle optimal totalement disjoints dont une résolution parallèle est tout à fait admissible. L'application (directe) de cette méthode fait l'objet de notre premier algorithme de contrôle optimal parallèle basé sur un calcul séquentiel des cibles et conditions initiales intermédiaires. Nous présentons cet algorithme et nous analysons sa convergence. Nous développons ensuite un deuxième algorithme basé sur un couplage entre la méthode précédente et l'algorithme pararéel. Dans cette approche, le calcul des cibles et des conditions initiales est effectué en parallèle. Ce couplage fournit une accélération considérable de la résolution séquentielle du problème initial. Nous présentons ce deuxième algorithme et donnons quelques résultats sur sa convergence. Une interprétation algébrique des deux algorithmes de contrôle optimal parallèle est également discutée dans ce chapitre.

Le **troisième chapitre** est consacré à l'étude de l'application de l'algorithme pararéel à l'équation de Boltzmann régissant la cinétique des neutrons au sein d'un réacteur nucléaire. Nous nous intéressons particulièrement à la simulation des accidents pour laquelle nous proposons un schéma pararéel permettant de simuler en parallèle, sur des machines à hautes performances, des configurations accidentelles d'explosion d'énergie pouvant survenir lors d'un emballement de la réaction en chaîne de fission. Certains schémas reposant sur des réductions de modèle sont aussi expliqués.

Nous présentons finalement, dans le **quatrième chapitre**, une méthode de parallélisation en temps des problèmes de contrôle optimal en résonance magnétique nucléaire (RMN). Une de ses premières applications sur un modèle nucléaire complexe de la RMN à l'état liquide a montré une accélération importante.

Chapitre 2

Méthode des cibles intermédiaires, théorie et mise en œuvre

Travail en collaboration avec Yvon Maday, Julien Salomon
dans le cadre du project PITAC
(Parallélisation Incluant le Temps pour Accélérer le Calcul).

Sommaire

Abstract

In this chapter, we develop a new method called *Intermediate target method* enabling the subdivision of an optimal control problem defined over a time interval I, to a series of optimal control subproblems defined in smaller time intervals. The new approach gives a possibility to parallelize the simulation of some problems using high performance computers. We first present the new method and give some related theoretical results. We then propose an adequate iterative coupling of that approach with the parareal in time algorithm. Our implementation, is performed using massively parallel machines. In order to show the efficiency and robustness of our algorithms, numerical results are presented at the end of the chapter.

Keywords: Parallel computation, optimal control of PDE, optimisation, iterative algorithm, parareal in time algorithm, domain decomposition methods

Résumé

Dans ce chapitre, nous développons une nouvelle méthode appelée *méthode des cibles intermédiaires* permettant de subdiviser un problème de contrôle optimal, défini sur un intervalle de temps I, en sous-problèmes de contrôle indépendants définis sur des sous-intervalles de petite taille. Cette nouvelle méthode rend possible un traitement parallèle du problème sur des super machines massivement parallèles. Dans un premier temps, nous présentons la méthode et nous donnons quelques résultats théoriques. Dans un second temps, nous proposons un couplage itératif entre notre méthode et l'algorithme pararéel en temps. La mise en œuvre numérique a été effectuée sur des ordinateurs à haute performance. Les résultats numériques sont présentés dans une dernière partie pour illustrer l'efficacité et la robustesse de nos algorithmes.

Mots-clés: Calcul parallèle, contrôle optimal des EDP, optimisation, algorithme itératif, l'algorithme pararéel en temps, méthodes de décomposition de domaine

2.1 Introduction

Dans ce chapitre, nous présentons une nouvelle méthode itérative permettant de décomposer un problème de contrôle optimal [45, 47, 50] en sous-problèmes indépendants et donc résolubles simultanément. Nous appliquons cette méthode, que nous appelons *méthode des cibles intermédiaires* à un problème de contrôle optimal d'un système gouverné par une équation aux dérivées partielles (EDP) parabolique avec un terme source. Notre méthode est basée sur la définition des cibles et de conditions initiales intermédiaires définissant à leur tour des sous-problèmes de contrôle optimal. Le problème considéré est posé sur un intervalle temporel $I := [0, T]$ et sur un domaine spatial Ω supposé borné. Notre procédure consiste à décomposer l'intervalle de temps et attribuer à chaque processeur disponible la résolution du sous-problème défini sur un sous-intervalle temporel.
Dans la literature il existe peu de méthodes de parallélisation qui étudient les problèmes de contrôle optimal parabolique. Nous citons ici [47, 50]. Dans la méthode introduite par Maday et Turinici [47], qui est principalement une méthode de gradient, le problème initial de contrôle optimal est étendu à un problème de contrôle parallèle par l'introduction d'une série de conditions initiales pour les problèmes de contrôle parallèles. Ces conditions initiales ainsi définies sont interprétées comme des contrôles virtuels [38]. Ensuite, le schéma pararéel en temps est utilisé dans un cadre de préconditionneur pour la méthode. Egalement, dans la méthode introduite par Mathew, Sarkis et Schaerer [50] le schéma pararéel en temps est utilisé comme un préconditionneur d'un schéma itératif qui résout un système de points critiques discret. Le couplage avec l'algorithme pararéel en temps a été bénéfique pour les deux méthodes de parallélisation. Il a, en effet, accéléré considérablement les simulations numériques.
Dans le présent chapitre, nous présentons nos travaux comme suit : dans un premier temps, la section 2.2 introduit le problème de contrôle optimal à résoudre. La section 2.3 fournit les outils de la parallélisation. Puis nous présentons à la section 2.4 deux algorithmes parallèles de contrôle optimal. Ensuite, à la section 2.5 nous donnons une interprétation matricielle des deux algorithmes ainsi que la méthode des cibles et conditions initiales intermédiaires elle-même. Enfin, nous présentons les tests et résultats numériques montrant la robustesse et l'efficacité des méthodes.

2.2 Présentation du problème

Le problème de contrôle étudié s'inscrit dans le cadre du contrôle distribué. Nous pensons que l'approche que nous développons s'étend sans difficulté au contrôle frontière des systèmes évolutifs de type parabolique ; ce type de problème est envisageable dans les proches perspectives. Nous nous limitons dans ce chapitre à l'étude du contrôle de la chaleur où le contrôle est appliqué sur une partie stricte Ω_c du domaine Ω. L'état à contrôler est un élément de l'espace $L^2(I; H_0^1(\Omega))$ où $H_0^1(\Omega) := \{u \in L^2(\Omega)/\nabla u \in L^2(\Omega), u = 0 \text{ sur } \partial\Omega\}$ tandis que le contrôle est un élément de l'espace $L^2(I; L^2(\Omega_c))$.

Par abus de notation, nous notons respectivement $\|.\|_2$ et $\langle ., . \rangle_2$ la norme et le produit scalaire sur $L^2(\Omega)$ et également sur $L^2(\Omega_c)$. Nous utilisons également la notation $\|.\|_v$ (respectivement $\|.\|_{v_n}$) ainsi que $\langle ., . \rangle_v$ (respectivement $\langle ., . \rangle_{v_n}$) pour représenter la norme et le produit scalaire dans l'espace Hilbertien $L^2(I, L^2(\Omega))$ (respectivement l'espace $L^2(I_n, L^2(\Omega))$).

Généralement, l'approche du contrôle optimal est une approche d'optimisation qui consiste à minimiser (ou maximiser selon la problématique) une fonctionnelle de coût quadratique de la forme :

$$\mathcal{J}(v) = \frac{1}{2}\|y(T) - y^{cible}\|_2^2 + \frac{\alpha}{2}\int_I \|v\|_2^2 dt, \tag{2.1}$$

où y^{cible} représente l'état cible du problème de contrôle, $\alpha \in \mathbb{R}_+^*$ pénalisant le terme intégral, et où l'état du système y vérifie une équation d'évolution de la forme

$$\begin{cases} \partial_t y - \nu\Delta y = \mathcal{B}v & \text{sur } I \times \Omega \\ y(0) = y_0 & \text{sur } \{0\} \times \Omega, \\ + \text{ conditions aux limites.} \end{cases} \tag{2.2}$$

Dans cette équation, Δ est l'opérateur Laplacien et ν est une constante positive. La condition initiale y_0 appartenant à $L^2(\Omega)$. La variable v est le contrôle agissant de façon linéaire sur le comportement y du système à travers l'opérateur $\mathcal{B}$ et la zone de contrôle Ω_c. Ce contrôle est un élément de l'espace $\mathcal{V} := L^2(I, L^2(\Omega))$ parfois appelé espace des contrôles admissibles. Formellement $\mathcal{B} \in \mathcal{L}(\mathcal{V}, L^2(I, L^2(\Omega_c)))$.
Le problème d'optimisation considéré est : trouver $v^\star$ tel que

$$v^\star = \arg\min_{v\in\mathcal{V}} \mathcal{J}(v). \tag{2.3}$$

Remarque 2.2.1. *Nous ne précisons pas le type de conditions aux limites utilisées. En fait, notre méthode reste valable pour différentes conditions aux limites. Sans perte de généralité, nous choisissons de traiter le cas des conditions de Dirichlet homogènes.*

Introduisons donc le *système d'optimalité classique séquentiel* (2.5) dont nous voudrons paralléliser la résolution. Comme nous sommes dans le cadre du contrôle linéaire, le coût (2.1) est strictement quadratique et strictement convexe. Il y a donc unicité de la solution. Pour calculer numériquement l'optimum, nous utilisons la méthode de descente de gradient. Pour appliquer ce type de méthode, nous devons bien sûr disposer du gradient de la fonctionnelle. Mais la dépendance de l'état du système y en le contrôle v rend difficile d'exprimer le gradient d'une manière explicite. Pour traiter cette difficulté, nous introduisons, comme il est classique de le faire, un nouvel état, appelé état adjoint (également appelé état dual) jouant le rôle d'un multiplicateur de Lagrange dans la définition du Lagrangien $\mathcal{L}g$ associé à notre problème :

$$\mathcal{L}g(v, y, p) := \frac{1}{2}\|y(T) - y^{cible}\|_2^2 + \frac{\alpha}{2}\int_I \|v\|_2^2 dt + \int_I \langle \partial_t y - \nu\Delta y - \mathcal{B}v, p\rangle_2 dt. \tag{2.4}$$

En dérivant le Lagrangien et en annulant ses dérivées par rapport aux trois variables nous obtenons :

$$\begin{cases} \dfrac{\partial \mathcal{L}g}{\partial v}(v, y, p) = 0, \\ \dfrac{\partial \mathcal{L}g}{\partial y}(v, y, p) = 0, \\ \dfrac{\partial \mathcal{L}g}{\partial p}(v, y, p) = 0. \end{cases} \tag{2.5}$$

Ce système conduit à

$$\begin{cases} \partial_t y - \nu \Delta y = \mathcal{B} v & \text{sur } I \times \Omega \\ y(0) = y_0 & \text{sur } \{0\} \times \Omega, \end{cases} \tag{2.6}$$

$$\begin{cases} -\partial_t p - \nu \Delta p = 0 & \text{sur } I \times \Omega \\ p(T) = y(T) - y^{cible} & \text{sur } \{T\} \times \Omega, \end{cases} \tag{2.7}$$

$$\alpha v + \mathcal{B}^* p = 0 \qquad \text{sur } I \times \Omega, \tag{2.8}$$

où $\mathcal{B}^*$ est l'opérateur adjoint de $\mathcal{B}$. L'état adjoint p défini en (2.7) nous permet de calculer explicitement le gradient $\nabla \mathcal{J}$ de la fonctionnelle de coût (2.1) puisque :

$$\nabla \mathcal{J}(v) = \alpha v + \mathcal{B}^* p.$$

2.3 Eléments de la parallélisation en temps

Notre méthode est basée sur une décomposition en sous-domaines du domaine temporel du contrôle. Elle fournit des outils permettant de construire des sous-fonctionnelles et des sous-problèmes de contrôle indépendants. Ces sous-problèmes peuvent alors être résolus simultanément.

2.3.1 Décomposition en temps et parallélisation

Nous subdivisons l'intervalle de temps I en $\hat{n}$ sous-intervalles $I_n = [t_n, t_{n+1}]$ de longueur identique :

$$I = \bigcup_{n=0}^{\hat{n}-1} I_n,$$

avec

$$0 = t_0 < t_1 < ... < t_{\hat{n}-1} < t_{\hat{n}} = T.$$

Ceci conduit à la définition de $\hat{n}$ sous-problèmes de la forme (2.9) ci-dessous.
Sur chaque intervalle I_n, nous cherchons à trouver un contrôle $v_n^\star$ dans l'ensemble $\mathcal{V}_n := L^2(I_n, L^2(\Omega_c))$ des contrôles admissibles qui vérifie :

$$v_n^\star = \arg \min_{v_n \in \mathcal{V}_n} \mathcal{J}_n(v_n), \tag{2.9}$$

avec

$$\mathcal{J}_n(v_n) := \frac{1}{2} \| y_n(t_{n+1}^-) - \xi_{n+1} \|_2^2 + \frac{\alpha}{2} \int_{I_n} \| v_n \|_2^2 dt, \tag{2.10}$$

où y_n est solution de l'EDP suivante :

$$\begin{cases} \partial_t y_n - \nu \Delta y_n = \mathcal{B} v_n & \text{sur } I_n \times \Omega \\ y_n(t_n^+) = \lambda_n & \text{sur } \{t_n^+\} \times \Omega, \end{cases} \tag{2.11}$$

où $(\xi_n)_{1 \le n \le \hat{n}}$ et $(\lambda_n)_{0 \le n \le \hat{n}-1}$ sont respectivement les cibles et les conditions initiales des sous-problèmes d'optimisation. Leurs définitions précises sont données par la suite.

Remarque 2.3.1. *Les sous-fonctionnelles de coût* (2.10) *sont de la même forme que la fonctionnelle de coût* (2.1)*. Ceci nous permet d'appliquer la même méthode pour minimiser la fonctionnelle et les sous-fonctionnelles.*

2.3.2 Système d'optimalité des sous-problèmes

De même que pour le problème initial (non-décomposé (2.3)), nous écrivons les systèmes d'optimalité associés aux sous-problèmes (2.9). Pour cela, nous en introduisons le Lagrangien $\mathcal{L}g_n$ et écrivons la condition d'optimalité selon :

$$\frac{\partial \mathcal{L}g_n}{\partial v_n} = \frac{\partial \mathcal{L}g_n}{\partial y_n} = \frac{\partial \mathcal{L}g_n}{\partial p_n} = 0.$$

qui s'écrit sous la forme :

$$\begin{cases} \partial_t y_n - \nu \Delta y_n = \mathcal{B} v_n & \text{sur } I_n \times \Omega \\ y_n(t_n^+) = \lambda_n & \text{sur } \{t_n^+\} \times \Omega, \end{cases} \tag{2.12}$$

$$\begin{cases} -\partial_t p_n - \nu \Delta p_n = 0 & \text{sur } I_n \times \Omega \\ p_n(t_{n+1}^-) = y(t_{n+1}) - \xi_{n+1} & \text{sur } \{t_{n+1}^-\} \times \Omega, \end{cases} \tag{2.13}$$

$$\alpha v_n + \mathcal{B}^* p_n = 0 \qquad \text{sur } I_n \times \Omega. \tag{2.14}$$

La solution de ce système est notée $v_n^\star$.
Nous enregistrons une série de points des conditions initiales de l'état direct $\lambda := \{\lambda_0, .., \lambda_{\hat{n}}\}$ et de conditions initiales pour l'état adjoint $\gamma := \{\gamma_1, .., \gamma_{\hat{n}}\}$ qui nous servirons pour définir la série des cibles $\xi := \{\xi_1, .., \xi_{\hat{n}}\}$:

$$\xi_n := \xi(t_n) \quad \text{avec} \quad \xi(t_n) := y(t_n) - p(t_n) \tag{2.15}$$

tandis que la série λ est définie par :

$$\lambda_n := y(t_n), \quad \text{pour } 0 \leq n \leq \hat{n} - 1 \tag{2.16}$$

et la série γ est définie par :

$$\gamma_n := p(t_n), \quad \text{pour } 1 \leq n \leq \hat{n}. \tag{2.17}$$

La trajectoire ξ n'est pas régie par une équation aux dérivées partielles, et elle vérifie $\xi(t_{\hat{n}}) = y^{cible}$ et la condition initiale de l'intervalle I_0 vérifie à son tour $\lambda_0 = y_0$.
Les sous-problèmes (2.9) sont totalement disjoints et peuvent donc être résolus simultanément et ne nécessitent aucune intercommunication. Pour chacune des unités de processeurs disponibles nous allouons le traitement des sous-problèmes (2.9) dont nous autorisons un nombre fixé d'itérations locales g_{max} (voir paragraphe page 30). Une communication entre le processeur maître et les processeurs esclaves est cependant nécessaire. Cette communication s'effectue par l'attribution à chaque sous-intervalle des points ξ_{n+1} et λ_n (voir définition ci-dessous). La donnée du couple (λ_n, ξ_{n+1}) est indispensable à la définition de la fonctionnelle (2.10) et de l'EDP (2.11). L'attribution de $\hat{n}$ couples de ce type à chacun des sous-domaines I_n est nécessaire lors de la communication entre le processeur maître, appelé *frontale* ou *cluster*, et les processeurs agents appelés *esclave*. La figure 2.1 illustre le traitement parallèle avec une décomposition de l'intervalle de temps en quatre sous-intervalles. L'attribution du couple λ_n, γ_n sur chaque sous-intervalle permet une évolution de l'état local parallèle direct (rouge) et de l'état local adjoint (bleu).

L'introduction de la trajectoire cible ξ est motivée par le résultat suivant :

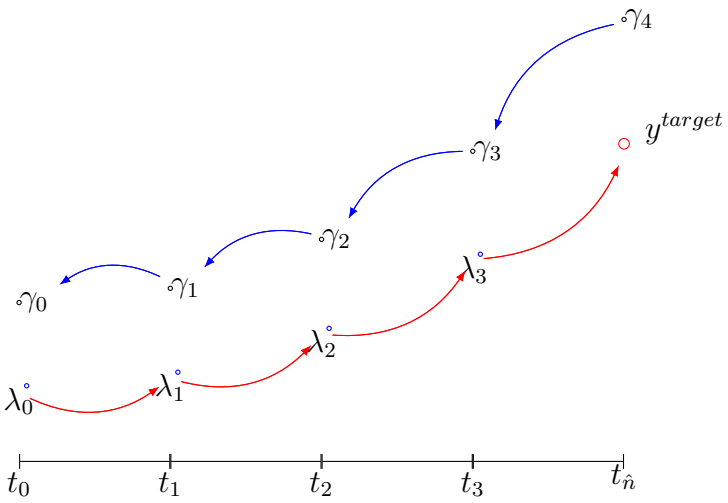


FIGURE 2.1 – Cas de $\hat{n} = 4$ sous-intervalles de décomposition.

Lemme 2.3.2. *Soit $\xi^\star$ la trajectoire cible définie en (2.15) avec $y = y^\star$ et $p = p^\star$, représentant respectivement l'état direct et l'état adjoint associés à $v^\star$. Soient $y_n^\star, p_n^\star, v_n^\star$ les solutions de (2.12)–(2.14), alors on a :*

$$v_n^\star = v_{|I_n}^\star.$$

Démonstration. Par unicité de la solution des sous-problèmes de contrôle locaux, il suffit de montrer que $v_{|I_n}^\star$ vérifie le système des points critiques (2.12)–(2.14). Nous remarquons d'abord que $y_{|I_n}^\star$ vérifie (2.12)avec $v_n = v_{|I_n}^\star$ et $\lambda_n = y^\star(t_n)$. D'après la définition (2.15), on a :

$$p^\star(t_{n+1}) = y^\star(t_{n+1}) - \xi^\star(t_{n+1}),$$

et donc $p_{|I_n}^\star$ vérifie (2.13). L'équation (2.14) est une conséquence de (2.13), d'où le résultat.□

Remarque 2.3.3. *Autrement dit, le lemme 2.3.2 donne une relation entre le minimum de la fonctionnelle de coût $\mathcal{J}$ et les minima des fonctionnelles de coût $\mathcal{J}_n$.*

2.4 Algorithmes de contrôle optimal parabolique parallèle

Dans cette section, nous présentons dans un premier temps la structure générale de notre méthode de parallélisation, puis nous donnons les algorithmes développés à partir de cette méthode et nous en discutons enfin la convergence.

2.4.1 Structure générale des la méthode des cibles et conditions initiales intermédiaires

Nous présentons maintenant la structure générale de la méthode des cibles et conditions initiales intermédiaires : considérons un contrôle initial v^0, supposons qu'à l'itération k, nous connaissions v^k. Le calcul de v^{k+1} est effectué comme suit :

(1) Calculer y^k, p^k par le contrôle v^k selon (2.6) et (2.7).

(2) Calculer les cibles et les conditions initiales intermédiaires $(\lambda_n^k)_{n\geq 0}$, $(\xi_n^k)_{n\geq 1}$ respectivement selon (2.16) et (2.15).

(3) Résoudre approximativement $\hat{n}$ sous-problèmes (2.9) en parallèle (pour $n = 0, ..., \hat{n} - 1$), pour chaque sous-intervalle nous effectuons g_{max} itérations d'une méthode d'optimisation. La solution correspondante sera noté $\tilde{v}_n^{k+1}$.

(4) Définir $\tilde{v}^{k+1}$ comme une concaténation des éléments $(\tilde{v}_n^{k+1})_{n=0,...,\hat{n}-1}$.

Afin de garantir le bon fonctionnement de cette méthode et d'en assurer la stabilité et la monotonie, nous relaxons les cibles et les conditions initiales intermédiaires par une interpolation adéquate du contrôle de la partie séquentielle (en mémoire) et du nouveau contrôle concaténé.
A partir de ces étapes basées sur les définitions (2.15)–(2.16), nous avons développé deux algorithmes pour simuler en parallèle le problème du contrôle optimal présenté initialement. On appelle ces deux algorithmes SITPOC (Serial Intermediate Target for Optimal Control) et PITPOC (Parallel Intermediate Target for Optimal Control) correspondant respectivement à :

- une approche des cibles et conditions initiales intermédiaires séquentielles.
- une approche des cibles et conditions initiales intermédiaires pararéelles.

Les deux algorithmes possèdent des structures semblables reposant sur les trois procédures principales suivantes :

- **Résolution en séquentiel** : elle varie entre les deux algorithmes, mais dans les deux cas, elle s'opère par l'optimisation d'une fonctionnelle de coût appropriée.
- **Résolution en parallèle** : elle est identique pour les deux algorithmes, elle consiste en une minimisation des sous-fonctionnelles de coût.
- **Echange d'informations et communication** : il s'agit de transfert d'informations dans les deux sens ; du processeur maître vers les processeurs agents, et inversement. On utilise ici la bibliothèque de calcul et de transfert de message parallèle MPI [4].

2.4.2 Première approche : cibles calculées séquentiellement

Dans cette sous-section nous présentons une première approche reposant sur ce qui a été introduit, où nous considérons une résolution tout à fait séquentielle de l'étape (1) afin de définir les conditions et cibles intermédiaires (2.15) dont le calcul est effectué à l'étape (2). Cette procédure fournit quelques particularités que nous présentons sous forme de remarques ci dessous.

Remarque 2.4.1. *Selon le lemme 2.3.2 et la procédure précédente, par une seule itération locale d'optimisation (i.e. $g_{max} = 1$), on dispose du gradient de $\mathcal{J}$ après l'étape (3) :*

$$\nabla \mathcal{J}_n = \nabla \mathcal{J}_{|I_n}.$$

4. MPI : *Message Passing Interface.*

En effet, rappelons que :

$$\nabla \mathcal{J}(t) = \alpha v(t) + \mathcal{B}^* p(t), \quad \forall t \in I. \tag{2.18}$$

Sur chaque sous-intervalle $I_{n=0,..,\hat{n}-1}$ on a aussi :

$$\nabla \mathcal{J}_n(s) = \alpha v_n(s) + \mathcal{B}^* p_n(s), \qquad \forall s \in I_n,$$

où la fonction cible relative est telle que

$$\xi_{n+1} = y(t_{n+1}) - p(t_{n+1}).$$

Représentons le semi-groupe relatif aux équations (2.2) et (2.11) par $\mathcal{E}_\tau(t; a, v)$ comme un propagateur de l'état direct partant de la condition initiale (a) à l'instant t contrôlée par v pour une durée égale à τ. Notons également $\mathcal{E}^*$ l'adjoint relatif au propagateur de l'état dual, sans variable de contrôle, tel que $\mathcal{E}^*_\tau(t, b)$ est le propagateur rétrograde de l'état adjoint b à partir du temps t pendant une durée égale à τ.
L'opérateur propagateur continu $\mathcal{E}$ est difficile à produire sur informatique, une approximation de celui-ci, notée $\mathcal{F}$ (respectivement $\mathcal{F}^*$), est donc mise en place. $\mathcal{F}$ (respectivement $\mathcal{F}^*$) garde toutes les propriétés de $\mathcal{E}$ (respectivement de $\mathcal{E}^*$). Suivant ces définitions nous avons :

$$\xi_{n+1} = \mathcal{F}_{t_{n+1}}(t = 0; \lambda_0; v_{|(0,t_n)}) - \mathcal{F}^*{}_{t_{\hat{n}} - t_{n+1}}(t_{\hat{n}}; p(t_{\hat{n}})).$$

Pour le choix des conditions initiales on prend $\lambda_{n,n=0,..,\hat{n}-1}$, telle que

$$\lambda_n = \mathcal{F}_{t_n}(t = 0; \lambda_0; v_{|(0,t_n)}). \tag{2.19}$$

On obtient donc

$$\xi_{n+1} = \mathcal{F}_{t_{n+1}-t_n}(t_n; \lambda_n; v_n) - \mathcal{F}^*{}_{t_{\hat{n}} - t_{n+1}}(t_{\hat{n}}; p(t_{\hat{n}})). \tag{2.20}$$

Or, pour tout s appartenant à I_n on a :

$$\begin{aligned} p_n(s) &= \mathcal{F}^*{}_{t_{n+1}-s}(t_{n+1}; (\mathcal{F}_{t_{n+1}-t_n}(t_n; \lambda_n; v_n) - \xi_{n+1})), \\ &= \mathcal{F}^*{}_{t_{n+1}-s}(t_{n+1}; \mathcal{F}^*{}_{t_{\hat{n}} - t_{n+1}}(t_{\hat{n}}; p(t_{\hat{n}}))), \\ &= \mathcal{F}^*{}_{t_{\hat{n}}-s}(t_{\hat{n}}; p(t_{\hat{n}})), \\ &= p(s). \end{aligned}$$

2.4.3 L'algorithme SITPOC : Serial Intermediate Target for Parallel Optimal Control

L'algorithme SITPOC est une version basique directement issue de l'approche des cibles intermédiaires. Nous développons ses étapes et nous étudions sa convergence.

Calculs séquentiel des cibles et conditions initiales intermédiaires

Avant de présenter les algorithmes, nous introduisons l'erreur relative qui va constituer notre critère d'arrêt :

$$Err^k = \frac{|\mathcal{J}(v^k) - \mathcal{J}(v^{k-1})|}{|\mathcal{J}(v^k)|}. \tag{2.21}$$

Nous présentons, à la page 21, l'algorithme calculant les cibles et conditions initiales d'une manière séquentielle. Nous précisons les définitions de quelques mots techniques utilisés dans l'algorithme.

- **Input** : ce sont les données que nécessite l'algorithme.
- **foreach** : utilisé pour des opérations récurrentes sur chacun des indices (i.e. il s'agit d'un calcul séquentiel).
- **forall** : utilisé pour des opérations simultanées (i.e. il s'agit d'un calcul parallèle).
- **Mpi_Send**(objet, destination) : c'est une commande (sub-routine de la bibliothèque MPI) pour la communication entre processeurs. Elle envoie le message "objet" depuis le processeur qui a exécuté la commande vers le processeur "destination". Cette commande est généralement accompagnée de la commande **MPI_Recv** sur le processeur destinataire.
- **Mpi_Recv**(object, source) : c'est une commande (sub-routine de la bibliothèque MPI) pour la communication entre processeurs. Elle reçoit le message "objet" depuis le processeur "source". Attention cette opération peut être bloquante, c'est à dire que l'exécution de la ligne de programme suivante ne se fait qu'après avoir reçu le message "objet".
- **Mpi_Broadcast**(source,objet) : une commande MPI de communication globale. Elle permet de distribuer le message "objet" sortant de la "source". Cette commande est de type globale et elle doit s'executer sur tout les processeurs en même temps.. Dans le cas d'utilisation (mise en œuvre) de communications bloquantes, il ne sera plus necessaire de faire un "Broadcast" de l'erreur "*Err*" afin d'arrêter la boucle "**repeat**"

Cette présentation de l'algorithme SITPOC explique en détail les étapes nécessaires qu'effectue ce dernier afin de pouvoir paralléliser la résolution du problème de contrôle optimal. Cet algorithme est adapté à l'implémentation sur des super calculateurs en trois parties principales. Une première partie de résolution séquentielle calcule les états direct et adjoint dont nous extrayons les cibles et conditions initiales intermédiaires (ξ_{n+1}^k etλ_n^k). Puis nous les communiquons selon leurs indices aux intervalles I_n pour construire les sous-fonctionnelles (2.10). Cette communication est faite par le processeur maître qui se charge d'envoyer ces informations vers chacun des processeurs esclaves. Après cette procédure de communication, tous les processeurs esclaves possèdent des fonctionnelles définies sur les intervalles I_n. La deuxième partie fait l'objet du cadre local de la résolution(voir sous-section 2.3.2). La troisième partie boucle l'algorithme pour réitérer les tâches de la première partie à l'itération suivante. Dans cette troisième partie, nous effectuons la concaténation des variables de contrôle locales et mettons à jour la variable globale du contrôle afin de minimiser le coût (2.1).

Analyse de convergence

Dans ce qui suit, nous donnons des preuves de convergence de notre approche des cibles intermédiaires permettant la parallélisation de problèmes de contrôle optimal.

Lemme 2.4.2. *L'opérateur Hessien : H_n est la restriction de l'opérateur H dont le contrôle correspondant est à support l'intervalle $[t_{\hat{n}-1}, t_{\hat{n}}]$.*

Algorithme 1 : SITPOC : Serial Intermediate Target for Parallele Optimal Control

Input : $\epsilon, \hat{n}(\#\, slave\, proc), y_0, y^T, \lambda_0^0 := y(0), v^0$;
$k \leftarrow 0; v^k \leftarrow v^0$;
repeat
  if *master processor* **then**
    Evolve $y^k := y(v^k)$ through (2.6);
    Evolve $p^k := p(v^k)$ through (2.7);
    Compute $(\lambda_n^k)_{0 \leq n \leq \hat{n}}$ with (2.16);
    Compute $(\gamma_n^k)_{\hat{n} \geq n \geq 1}$ with (2.17);
    foreach $n \in \{0, .., \hat{n}-1\}$ **do**
      $\xi_{n+1}^k \leftarrow \lambda_{n+1}^k - \gamma_{n+1}^k$;
      $v_n^k \leftarrow v_{|_{I_n}}^k$;
      Mpi_Send $(v_n^k, \lambda_n^k, \xi_{n+1}^k, processor(n))$;
    end
  else
    forall $slave\, processors(n) / \, n \in \{0, \ldots, \hat{n}-1\}$ **do**
      Mpi_Recv $(v_n^k, \lambda_n^k, \xi_{n+1}^k, master\, processor)$;
      $\tilde{v}_n^{k,1} \leftarrow v_n^k$;
      for $g = 1 : g_{max}$ **do**
        Evolve $\tilde{y}_n^{k,g} := y_n(\tilde{v}_n^{k,g})$ through (2.12);
        Evolve $\tilde{p}_n^{k,g} := p_n(\tilde{v}_n^{k,g})$ through (2.13);
        $\nabla \mathcal{J}_n(\tilde{v}_n^{k,g}) \leftarrow \alpha \tilde{v}_n^{k,g} + B^* \tilde{p}_n^{k,g}$;
        $\tilde{v}_n^{k,g+1} \leftarrow \tilde{v}_n^{k,g} - \rho_n^{k,g} \nabla \mathcal{J}_n(\tilde{v}_n^{k,g})$;
      end
      $\tilde{v}_n^{k+1} \leftarrow \tilde{v}_n^{k,g_{\max}+1}$;
      Mpi_Send $(\tilde{v}_n^{k+1}, master\, processor)$;
    end
  end
  if *master processor* **then**
    foreach $n \in \{0, .., \hat{n}-1\}$ **do**
      Mpi_Recv $(\tilde{v}_n^{k+1}, processor(n))$;
    end
    $\tilde{v}^{k+1} \leftarrow [\tilde{v}_0^{k+1}, .., \tilde{v}_n^{k+1}, .., \tilde{v}_{\hat{n}-1}^{k+1}]$;
    $\theta^k \leftarrow -\frac{\langle \nabla \mathcal{J}(v^k), \tilde{v}^{k+1} - v^k \rangle}{\langle \tilde{v}^{k+1} - v^k, H(\tilde{v}^{k+1} - v^k) \rangle}$;
    Update $v^{k+1} \leftarrow v^k + \theta^k (\tilde{v}^{k+1} - v^k)$;
    Evaluate Err according to (2.21) ;
  end
  $k \leftarrow k+1$;**Mpi_Broadcast** $(master\, processor, Err)$;
until $Err \leq \epsilon$;
Result : v^k is the optimal control

Démonstration. Soit une variation de contrôle $\delta v \in L^2([0,T];L^2(\Omega_c))$. Un calcul de différentielle standard donne :

$$\langle \delta v, H(\delta v)\rangle_v = \|\delta y(t_{\hat{n}})\|_2^2 + \alpha \int_I \|\delta v(t)\|_2^2 dt,$$

où δy est la solution de l'équation suivante :

$$\begin{cases} \partial_t \delta y - \nu \Delta \delta y = \mathcal{B}\delta v & \text{sur } I \times \Omega \\ \delta y(0) = 0 & \text{sur } \{0\} \times \Omega. \end{cases} \tag{2.22}$$

Etant donné $0 \leq n \leq \hat{n} - 1$, considérons maintenant la variation de contrôle δv_n à support inclus dans le sous-intervalle I_n, $\delta v_n \in L^2(I_n; L^2(\Omega_c))$. De la même manière que précédemment on trouve que :

$$\langle \delta v_n, H_n(\delta v_n)\rangle_{v_n} = \|\delta y_n(t_{n+1})\|_2^2 + \alpha \int_{I_n} \|\delta v_n(t)\|_2^2 dt,$$

où δy_n est la solution de l'équation suivante :

$$\begin{cases} \partial_t \delta y_n - \nu \Delta \delta y_n = \mathcal{B}\delta v_n & \text{sur } I_n \times \Omega \\ \delta y_n(t_n) = 0 & \text{sur } \{t_n\} \times \Omega. \end{cases} \tag{2.23}$$

Supposons maintenant que $\delta v = 0$ sur $[0, t_{\hat{n}-1}]$. La restriction de δy sur l'intervalle $[t_{\hat{n}-1}, t_{\hat{n}}]$ vaut $\delta y(t_{\hat{n}-1}) = 0$. Elle est donc à une translation temporelle près la solution de (2.23).□
Dans le lemme suivant, nous donnons une estimation de l'opérateur Hessien.

Lemme 2.4.3. *Soit $\delta v \in L^2(I; L^2(\Omega_c))$. Alors :*

$$\alpha \int_I \|\delta v(t)\|_2^2 dt \leq \langle \delta v, H(\delta v)\rangle_v \leq \beta \int_I \|\delta v(t)\|_2^2 dt, \tag{2.24}$$

où $\beta = \alpha + C/\sqrt{2}$, avec $C \equiv C(\Omega)$ la constante de Poincaré associée à $L^2(\Omega)$.

D'après le Lemme 2.4.2 H_n vérifie (2.24).

Démonstration. Grâce à (2.22) et à l'inégalité de Young, nous avons pour tout $t \in [0,T]$ et tout $\varepsilon > 0$:

$$\begin{aligned} \frac{1}{2}\frac{d}{dt}\|\delta y(t)\|_2^2 + \nu \|\nabla_x \delta y(t)\|_2^2 &= \langle \delta y(t), \mathcal{B}\delta v(t)\rangle_2 \\ &\leq \frac{1}{2}\left(\varepsilon \|\delta y(t)\|_2^2 + \frac{1}{\varepsilon}\|\mathcal{B}\delta v(t)\|_2^2\right), \end{aligned} \tag{2.25}$$

où ∇_x représente le gradient relatif à la variable "x" d'espace. Comme δy vérifie les conditions aux limites de Dirichlet imposées à l'équation aux dérivées partielles, nous pouvons appliquer l'inégalité de Poincaré dans (2.25) pour obtenir :

$$\|\delta y(t)\|_2 \leq C \|\nabla \delta y(t)\|_2 ,$$

où C est une constante positive. Par combinaison de cette estimation avec (2.25), nous en déduisons :

$$\frac{1}{2}\frac{d}{dt}\|\delta y(t)\|_2^2 \leq \left(\frac{\varepsilon}{2} - \frac{\nu}{C^2}\right)\|\delta y(t)\|_2^2 + \frac{1}{2\varepsilon}\|\mathcal{B}\delta v(t)\|_2^2.$$

Posons maintenant $\varepsilon = \frac{2\nu}{C^2}$, nous avons :

$$\begin{aligned}\frac{d}{dt}\|\delta y(t)\|_2^2 &\leq \frac{C^2}{2}\|\mathcal{B}\delta v(t)\|_2^2 \\ &\leq \frac{C^2}{2}\|\delta v(t)\|_2^2, \text{ car } \|\mathcal{B}\|_2 \leq 1,\end{aligned}$$

d'où le résultat, puisque $\|\delta y(0)\|_2^2 = 0$. □

Sous certaines hypothèses la convergence de l'algorithme SITPOC est garantie.

Théorème 2.4.4. *Supposons que la séquence de contrôle $(v^k)_{k\in\mathbb{N}}$ définie dans l'algorithme 2.4.3 vérifie :*

$$\mathcal{J}(v^k) \neq \mathcal{J}(v^\infty), \qquad \forall k \geq 0, \tag{2.26}$$
$$\|\nabla\mathcal{J}(v^k)\|_v \leq \eta\|v^{k+1} - v^k\|_v, \qquad \forall k \geq 0, \tag{2.27}$$
$$\mathcal{J}(v^k) - \mathcal{J}(v^{k+1}) \geq \kappa\|v^k - v^{k+1}\|_v^2, \qquad \forall k \geq 0, \tag{2.28}$$

pour $\eta > 0$ et $\kappa > 0$. Alors $(v^k)_{k\in\mathbb{N}}$ converge linéairement avec un taux de convergence $(1 - \frac{\kappa}{2(\gamma\eta)^2}) \in [0, 1)$ vers la solution du système d'optimalité local.

Notons que dans le cas où (2.26) est non vérifiée, alors il existe $k_0 \in \mathbb{N}$ tel que $v^{k_0} = v^\infty, \forall k \geq k_0$, et l'optimum est atteint en un nombre fini d'itérations.

Démonstration. Soit $\widetilde{\mathcal{J}}$ la fonction définie par :

$$\widetilde{\mathcal{J}}(v) = \mathcal{J}(v) - \mathcal{J}(v^\star).$$

Par définition de $v^\star$ nous avons :

$$\widetilde{\mathcal{J}}(v) = \frac{1}{2}\langle H(v - v^\star), v - v^\star\rangle_v \leq \frac{\beta}{2}\int_I \|v - v^\star\|_2^2 dt, \tag{2.29}$$

où H représente l'opérateur Hessien associé à la fonctionnelle objectif $\mathcal{J}$.
Comme $\mathcal{J}$ est strictement quadratique nous avons :

$$\nabla\mathcal{J}(v) = H(v - v^\star).$$

Par conséquent

$$\langle\nabla\mathcal{J}(v), v - v^\star\rangle_v = \langle H(v - v^\star), v - v^\star\rangle_v \geq \alpha\|v - v^\star\|_v^2,$$

d'où

$$\|v - v^\star\|_v \leq \frac{1}{\alpha}\|\nabla\mathcal{J}(v)\|_v. \tag{2.30}$$

La combinaison de (2.29) et de (2.30) implique :

$$\sqrt{\widetilde{\mathcal{J}}(v)} \leq \gamma\|\nabla\mathcal{J}(v)\|_v\,, \tag{2.31}$$

avec $\gamma = \frac{1}{\alpha}\sqrt{\frac{\beta}{2}}$.
Nous avons alors :

$$\begin{aligned}
\sqrt{\widetilde{\mathcal{J}}(v^k)} - \sqrt{\widetilde{\mathcal{J}}(v^{k+1})} &\geq \frac{1}{2\sqrt{\widetilde{\mathcal{J}}(v^k)}}\left(\mathcal{J}(v^k) - \mathcal{J}(v^{k+1})\right) \\
&\geq \frac{\kappa}{2\sqrt{\widetilde{\mathcal{J}}(v^k)}}\|v^k - v^{k+1}\|_v^2 \qquad (2.32) \\
&\geq \frac{\kappa}{2\gamma\|\nabla\mathcal{J}(v)\|_v}\|v^k - v^{k+1}\|_v^2 \qquad (2.33) \\
&\geq \frac{\kappa}{2\gamma\eta\|v^k - v^{k+1}\|_v}\|v^k - v^{k+1}\|_v^2 \qquad (2.34) \\
&\geq c\|v^k - v^{k+1}\|_v, \qquad (2.35)
\end{aligned}$$

où $c = \frac{\kappa}{2\gamma\eta}$. En effet, (2.32) vient de (2.28), (2.33) de (2.31) et (2.34) de (2.27). La convergence découle ensuite du fait que les suites $\left(\mathcal{J}(v^k)\right)_{k\in\mathbb{N}}$, et $\left(\sqrt{\widetilde{\mathcal{J}}(v^k)}\right)_{k\in\mathbb{N}}$ sont des suites de Cauchy.
Nous étudions maintenant le taux de convergence de l'algorithme. Nous définissons $r^k = \sum_{\ell=k}^{+\infty}\|v^{\ell+1} - v^\ell\|$. En sommant (2.35) à partir de $l = k$ jusqu'à l'infini, nous obtenons :

$$\sqrt{\widetilde{\mathcal{J}}(v^k)} \geq cr^k.$$

Avec (2.31) et (2.27), nous trouvons :

$$\eta\gamma(r^k - r^{k+1}) \geq cr^k. \qquad (2.36)$$

Notons que cette inégalité implique que $1 - \frac{c}{\eta\gamma} \geq 0$. Nous définissons

$$C := \frac{c}{\eta\gamma}.$$

Nous obtenons donc $0 < C \leq 1$. Grâce à (2.36) :

$$(1-C)^{-k}r^k \geq (1-C)^{-(k+1)}r^{k+1},$$

ce qui prouve le résultat. □

Nous donnons maintenant un exemple pour lequel les conditions (2.28)–(2.27) sont vérifiées.

Théorème 2.4.5. *Nous supposons que l'etape (3) de l'Algorithme 2.4.1 (à la page 17) est réalisée par une itération locale de la méthode de la descente (gradient), et qu'à l'étape k, l'algorithme est initialisé avec* $v_n^k := v_{|I_n}^k$. *Alors* (2.27)–(2.28) *sont vérifiées et l'algorithme converge vers la solution du système* (2.12)–(2.14).

Démonstration. L'étape d'optimisation (2) revient à :

$$\tilde{v}_n^{k+1} = v_n^k - \rho_n^k\nabla\mathcal{J}_n(v_n^k).$$

Puisque $\mathcal{J}_n$ est quadratique, nous obtenons :

$$\rho_n^k = \frac{\|\nabla \mathcal{J}_n(v_n^k)\|_{v_n}^2}{\langle \nabla \mathcal{J}_n(v_n^k), H_n(\nabla \mathcal{J}_n(v_n^k))\rangle_{v_n}}.$$

En outre, les lemmes 2.4.2 et 2.4.3 impliquent l'encadrement :

$$\frac{1}{\beta} \leq \rho_n^k \leq \frac{1}{\alpha}. \tag{2.37}$$

Nous pouvons obtenir des estimations similaires sur le paramètre de relaxation θ. Dans cette optique, notons tout d'abord que nous ne considérons qu'une seule itération de la méthode de descente de gradient : $\nabla \mathcal{J}_n(v_n^k) = \nabla \mathcal{J}(v^k)_{|I_n}$. Ainsi le paramètre de la relaxation θ vaut :

$$\begin{aligned} \theta^k &= -\frac{\langle \nabla \mathcal{J}(v^k), \tilde{v}^{k+1} - v^k\rangle_v}{\langle \tilde{v}^{k+1} - v^k, H(\tilde{v}^{k+1} - v^k)\rangle_v}, \\ &= -\frac{1}{\langle \tilde{v}^{k+1} - v^k, H(\tilde{v}^{k+1} - v^k)\rangle_v} \sum_{n=0}^{\hat{n}-1} \frac{1}{\rho_n} \|\tilde{v}_n^{k+1} - v_n^k\|_{v_n}^2. \end{aligned} \tag{2.38}$$

Nous en déduisons par (2.37) l'encadrement :

$$\frac{\alpha}{\beta} \leq |\theta^k| \leq \frac{\beta}{\alpha}, \tag{2.39}$$

vue que $-\frac{\beta}{\alpha} \leq \theta^k \leq -\frac{\alpha}{\beta}$.
Montrons maintenant (2.27). Nous avons :

$$\begin{aligned} \|v^{k+1} - v^k\|_v &= |\theta^k| \|\tilde{v}^{k+1} - v^k\|_v \\ &= |\theta^k| \sqrt{\sum_{n=0}^{\hat{n}-1} \|\tilde{v}_n^{k+1} - v_n^k\|_{v_n}^2} \\ &= |\theta^k| \sqrt{\sum_{n=0}^{\hat{n}-1} (\rho_n^k)^2 \|\nabla J_n(v_n^k)\|_{v_n}^2} \end{aligned}$$

Nous pouvons ainsi encadrer $\|v^{k+1} - v^k\|_v$ de la façon suivante :

$$\frac{\alpha}{\beta^2} \|\nabla J(v^k)\|_v \leq \|v^{k+1} - v^k\|_v \leq \frac{\beta}{\alpha^2} \|\nabla J(v^k)\|_v \tag{2.40}$$

ce qui établit (2.27), en posant $\eta = \frac{\beta^2}{\alpha}$.

Enfin, la variation de la fonctionnelle du coût entre deux itérations successives de

l'algorithme vaut :

$$
\begin{aligned}
J(v^k) - J(v^{k+1}) &= \langle \nabla J(v^k), v^k - v^{k+1} \rangle_v - \frac{1}{2} \langle HJ(v^k - v^{k+1}), v^k - v^{k+1} \rangle_v \\
&= -\theta^k \langle \nabla J(v^k), \tilde{v}^{k+1} - v^k \rangle_v \\
&\quad - \frac{1}{2} (\theta^k)^2 \langle HJ(\tilde{v}^{k+1} - v^k), \tilde{v}^{k+1} - v^k \rangle_v \text{ on utilisant la définition (2.38)} \\
&= \frac{1}{2} \frac{\left(\langle \nabla J(v^k), \tilde{v}^{k+1} - v^k \rangle_v \right)^2}{\langle HJ(\tilde{v}^{k+1} - v^k), \tilde{v}^{k+1} - v^k \rangle_v} = -\frac{\theta^k}{2} \langle \nabla J(v^k), \tilde{v}^{k+1} - v^k \rangle_v \\
&= \frac{|\theta^k|}{2} \langle \nabla J(v^k), \sum_{n=0}^{\hat{n}-1} \rho_n^k \nabla J_n(v_n^k) \rangle_v \\
&= \frac{|\theta^k|}{2} \sum_{n=0}^{\hat{n}-1} \rho_n^k \langle \nabla J(v^k), \nabla J_n(v_n^k) \rangle_v \\
&= \frac{|\theta^k|}{2} \sum_{n=0}^{\hat{n}-1} \rho_n^k \langle \nabla J_n(v_n^k), \nabla J_n(v_n^k) \rangle_{v_n} \\
&= \frac{|\theta^k|}{2} \sum_{n=0}^{\hat{n}-1} \rho_n^k \| \nabla J_n(v_n^k) \|_{v_n}^2 \\
&\geq \frac{1}{2} \frac{\alpha}{\beta^2} \| \nabla J(v^k) \|_v^2, \text{ on utilise (2.40)} \\
&\geq \frac{1}{2} \frac{\alpha}{\beta^2} \left(\frac{\alpha^2}{\beta} \right)^2 \| v^{k+1} - v^k \|_v^2 \\
&\geq \frac{1}{2} \frac{\alpha^5}{\beta^4} \| v^{k+1} - v^k \|_v^2. \qquad (2.41)
\end{aligned}
$$

Ce qui établie l'équation (2.28) on posant $\kappa = \frac{1}{2} \frac{\alpha^5}{\beta^4}$.

□

2.4.4 Deuxième approche : cibles calculées parallèlement

Dans l'approche précédente, chaque étape nécessite une propagation/résolution séquentielle sur tout l'intervalle de temps I. Nous présentons maintenant une approche plus adaptée aux machines parallèles. En effet, nous allégeons le traitement de la partie séquentielle de l'algorithme précédant. Nous signalons que pendant le traitement séquentiel effectué par le processeur maître les processeurs esclaves ne font aucune tâche, ce qui fait de cette durée un temps mort de traitement parallèle. La super machine parallèle sur laquelle nous exécutons le calcul se trouve alors sous-exploitée.
Afin d'utiliser au mieux ces ressources informatiques, nous considérons cette résolution séquentielle, et nous remplaçons le solveur propagateur fin $\mathcal{F}$ par un solveur propagateur grossier $\mathcal{G}$. En contrepartie, l'emploi d'un tel solveur propagateur dégradera d'une manière directe la définition des cibles intermédiaires, ce qui peut générer des instabilités numériques. Une technique permettant de résoudre ce problème, consiste à corriger l'erreur issue du solveur grossier par des solutions fines calculées en parallèle. La correction que

nous proposons est une combinaison entre les solutions grossière et fine qui sont en mémoire cache et entre la solution grossière de l'itération courante. Malgré cette correction des erreurs provenant du calcul séquentiel grossier, l'algorithme peut présenter des incohérences vis-à-vis de la monotonie. Une relaxation est donc mis en place à cet effet. Cette relaxation donne plus de flexibilité à l'algorithme et permet une mise à jour des variables itératives (v^k et λ_n^k) d'une manière qui garantie la monotonie de la minimisation. Nous introduisons une notation appropriée au niveau de l'algorithme pararéel (2.42) s'inscrivant dans le cadre du contrôle des équations d'évolutions. Nous utiliserons la notation de $\mathcal{F}_{\Delta T}$ pour désigner une propagation fine pendant une durée ΔT, respectivement $\mathcal{G}_{\Delta T}$ pour désigner une propagation grossière pendant une durée ΔT. Nous rappelons tout d'abord l'algorithme sans contrôle présenté ainsi dans [6, 9],

$$\lambda_n^{k+1} = \mathcal{G}_{\Delta T}(\lambda_{n-1}^{k+1}) + \mathcal{F}_{\Delta T}(\lambda_{n-1}^{k}) - \mathcal{G}_{\Delta T}(\lambda_{n-1}^{k}), \tag{2.42}$$

mettant à jour les conditions initiales des sous-problèmes dont les états sont définis par :

$$\begin{cases} \partial_t y_n - \nu \Delta y_n = \mathcal{B} v_n & \text{sur } I_n \times \Omega \\ y_n(t_n^+) = \lambda_n & \text{sur } \{t_n\} \times \Omega. \end{cases} \tag{2.43}$$

Nous adopterons la formulation suivante dans le cadre de notre problème :

$$\lambda_n^{k+1} = \mathcal{G}_{\Delta T}(\lambda_{n-1}^{k+1}; v_{n-1}^{k+1}) + \mathcal{F}_{\Delta T}(\lambda_{n-1}^{k}; v_{n-1}^{k}) - \mathcal{G}_{\Delta T}(\lambda_{n-1}^{k}; v_{n-1}^{k}), \tag{2.44}$$

où v^{k+1} est le contrôle courant, il commande le solveur $\mathcal{G}_{\Delta T}$ qui à son tour prédit l'état le long de l'intervalle temporel. Le reste de la formule est issu des calculs précédents, il est en mémoire cache.

Remarque 2.4.6. *Nous pouvons utiliser deux types de contrôles pour le solveur fin $\mathcal{F}_{\Delta T}$. Ces deux possibilités sont présentées ultérieurement.*

Dans la suite nous considérons une suite de fonctions coût $(\Phi(\theta)_k)_{k\geq 0}$ où θ est un paramètre réel. À chaque itération k nous allons chercher le minimum de cette fonction qui nous servira ensuite à la mise à jour du contrôle v^k et des conditions initiales $(\lambda_n^k)_{n\geq 0}^{k\geq 0}$. Le terme général de la suite $(\Phi_k)_{k\geq 0}$ est donc défini comme suit :

$$\Phi_k(\theta) := \frac{1}{2}\|\lambda_{\hat{n}}^{k+1}(\theta) - y^{cible}\|_2^2 + \frac{\alpha}{2}\int_I \|v^{k+1}(\theta)\|_2^2 dt. \tag{2.45}$$

La fonction $\Phi_k(\theta)$ atteint son minimum, noté Φ^{k+1}, en la valeur $\theta = \theta^k$. C'est-à-dire :

$$\Phi^{k+1} := \Phi_k(\theta^k). \tag{2.46}$$

La conception de cette fonction prend en considération un état final pararéel $\lambda_{\hat{n}}^{k+1}(\theta)$ (dont nous précisons la formulation de mise à jour (2.48) un peu plus tard). Il s'agit ici d'un état relaxé dont l'évolution est tout à fait parallèle.

Remarque 2.4.7. *Dans l'algorithme* PITPOC*, il est nécessaire de relaxer les conditions initiales $(\lambda_n)_{1\leq n\leq \hat{n}-1}$ (comme c'est le cas pour* SITPOC*) pour assurer la monotonie de $(\Phi_k(\theta^k))_{k\geq 0}$ (voir définition ci-après) dans le sens où :*

$$\Phi_k(\theta = 0) = \Phi_{k-1}(\theta^{k-1}) = \Phi^{k-1}.$$

Introduisons donc la relaxation θ pondérant les conditions initiales pararéelles (i.e. $(\lambda_n^k)_{n\geq 0}^{k\geq 0}$) pour garantir la monotonie de la fonction coût dépendante de l'itération courante k (que l'on définira ci-dessous en (2.45)).
Pour cela, considérons un opérateur de correction noté : $\mathcal{C}_{\Delta T}$ défini suivant une récurrence sur $k > 0$, tel que si on connaît $\lambda_n^{k-1}, \lambda_n^k, v_n^{k-1}, v_n^k$ et θ^{k-1} alors l'itéré suivant est défini par :

$$\mathcal{C}_{\Delta T}(\theta; \lambda_n^k; v_n^k) := (1-\theta)\mathcal{C}_{\Delta T}(\theta^{k-1}; \lambda_n^{k-1}; v_n^{k-1}) + \theta(\mathcal{F}-\mathcal{G})_{\Delta T}(\lambda_n^k; v_n^k). \tag{2.47}$$

Cette formule a un aspect de correction relaxée vis-à-vis des conditions initiales des sous-problèmes. Ces conditions seront définies par récurrence sur deux indices k et n, associés respectivement à l'itération courante et à l'intervalle courant :

$$\lambda_{n+1}^{k+1}(\theta) = \mathcal{G}_{\Delta T}(\lambda_n^{k+1}; v_n^{k+1}(\theta)) + \mathcal{C}_{\Delta T}(\theta; \lambda_n^k; v_n^k), \tag{2.48}$$

représentant le pararéel adapté à la structure itérative de notre méthode dont le contrôle $v_n^{k+1}(\theta)$ est mis à jour selon $v^{k+1}(\theta) = v^k + \theta(\tilde{v}^{k+1} - v^k)$. Il est important de considérer la notation suivante et celle-ci est valable pour toutes les variables dépendant de la relaxation θ :

$$v^{k+1} := v^{k+1}(\theta^k),$$

ainsi,

$$v^{k+1}(\theta = 0) := v^k := v^k(\theta^{k-1}).$$

Egalement,

$$\lambda_{n+1}^{k+1}(\theta = 0) = \lambda_{n+1}^k(\theta^k).$$

Remarque 2.4.8.
- *Nous soulignons que le terme initial λ_0^k de la suite récurrente $(\lambda_n^k)_{n\geq 0}^{k\geq 0}$ est un terme fixe indépendant de k.*
- *Nous avons une dépendance affine par rapport à θ de $\lambda_{n+1}^{k+1}(\theta)$, ce qui implique en particulier que*

$$\lambda_{n+1}^{k+1}(\theta) = \lambda_{n+1}^{k+1}(0) + \theta(\tilde{\lambda}_{n+1}^{k+1} - \lambda_{n+1}^k),$$

De plus,

$$\begin{aligned}
\lambda_{n+1}^{k+1}(\theta = 0) &= \mathcal{G}_{\Delta T}(\lambda_n^{k+1}; v_n^{k+1}(\theta = 0)) + \mathcal{C}_{\Delta T}((\theta = 0); \lambda_n^k; v_n^k) \\
&= \mathcal{G}_{\Delta T}(\lambda_n^{k+1}; v_n^k) + \mathcal{C}_{\Delta T}(\theta^{k-1}; \lambda_n^{k-1}; v_n^{k-1}) \\
&= \mathcal{G}_{\Delta T}(\lambda_n^{k+1}; v_n^k(\theta^{k-1})) + \mathcal{C}_{\Delta T}(\theta^{k-1}; \lambda_n^{k-1}; v_n^{k-1}) \\
&= \lambda_{n+1}^k(\theta^{k-1}) \\
&= \lambda_{n+1}^k.
\end{aligned}$$

- *Enfin, nous pouvons formuler une expression plus simple de $\lambda_{n+1}^{k+1}(\theta)$:*

$$\lambda_{n+1}^{k+1}(\theta) = \lambda_{n+1}^k + \theta(\tilde{\lambda}_{n+1}^{k+1} - \lambda_{n+1}^k), \tag{2.49}$$

avec $\tilde{\lambda}_{n+1}^{k+1} = \lambda_{n+1}^{k+1}(\theta = 1)$.

À chaque itération, et après avoir concaténé les contrôles parallèles obtenus sur les sous-intervalles par une procédure de gradient à pas fixe ou variable. Nous souhaitons que les trajectoires des états direct et adjoint deviennent continues, nous employons pour cela une approche itérative. Pour cette raison, nous introduisons dans la partie séquentielle de l'approche une fonction coût, dépendante du paramètre θ, à minimiser, dont il faut noter que l'état final $\lambda_{\hat{n}}$ du système pararéel est homologue à un état $y(T)$. Ceci est propagé suivant l'algorithme pararéel (2.49).

Remarque 2.4.9. *Les sous-fonctionnelles de coût* (2.10) *sont de même type que la fonctionnelle de coût globale* (2.1)*. En effet, elles sont quadratiques en deux contributions ; une intégrale pour l'énergie et une autre mesurant l'écart de la solution avec sa cible.*

Remarque 2.4.10. *Notre méthode repose plus précisément sur l'itération de l'algorithme pararéel qui est couplé avec l'algorithme de la sous-section 2.4.2 précédente.*

Equations et affectations Dans ce paragraphe, on présente les équations mises en place par l'approche afin d'affecter correctement et mettre à jour les variables servant à produire le contrôle adéquat. On présente ce qui se produit globalement dans le cadre séquentiel et ce qui se produit localement en parallèle.

Cadre global : Dans la partie globale (calcul séquentiel) chaque itération k de l'algorithme (2.4.5 voir ci-dessous) est une minimisation de la fonction (2.45). En plus de cela, le pararéel relaxé est mis en place afin de rapprocher l'état final $\lambda_{\hat{n}}^k$ du système parallèle à celui du système séquentiel.
La mise à jour des suites $v^{k+1} := v^{k+1}(\theta)$ et $\lambda^{k+1} := \lambda^{k+1}(\theta)$ est formalisée de la façon suivante :

$$v^{k+1} = v^k + \theta(\tilde{v}^{k+1} - v^k), \tag{2.50}$$

$$\lambda^{k+1} = \lambda^k + \theta(\tilde{\lambda}^{k+1} - \lambda^k), \tag{2.51}$$

où $\tilde{v}^k$ et $\tilde{\lambda}^k$ sont respectivement la concaténation sur l'indice n (du sous-intervalle de temps) du contrôle local $\tilde{v}_n^k$ et de la condition initiale locale $\tilde{\lambda}_n^k$ qui se mettent à jour selon l'algorithme pararéel en temps [49, 41] tel que :

$$\tilde{\lambda}_{n+1}^{k+1} = \mathcal{G}_{\Delta T}\big(\lambda_n^{k+1}; \tilde{v}_n^{k+1}\big) + \mathcal{F}_{\Delta T}\big(\lambda_n^k; \begin{Bmatrix} \tilde{v}_n^k \\ v_n^k \end{Bmatrix}\big) - \mathcal{G}_{\Delta T}\big(\lambda_n^k; \begin{Bmatrix} \tilde{v}_n^k \\ v_n^k \end{Bmatrix}\big). \tag{2.52}$$

Remarque 2.4.11. *Les accolades présentes dans la notation du solveur fin "*$\mathcal{F}_{\Delta T}\big(\lambda_n^k; \begin{Bmatrix} \tilde{v}_n^k \\ v_n^k \end{Bmatrix}\big)$*" représentent des choix admissibles de contrôle pour le solveur.*

Ayant calculé cette équation, la formule (2.51) fournit les conditions initiales correspondant à l'étape suivante.
Comme les conditions initiales $(\tilde{\lambda}_n^k)_{n\leq\hat{n}}^{k\leq\hat{k}}$ ont une structure pararéelle, il en est de même pour les états duaux $\gamma = (\gamma_n)_{n\geq 0}$ sur lesquels on itère d'une manière pararéelle similaire à celle des $\tilde{\lambda}$, grâce à la formule :

$$\gamma_{n+1}^{k+1} = \mathcal{G}_{\Delta T}^*(\gamma_{n+2}^{k+1}) + \mathcal{F}_{\Delta T}^*(\gamma_{n+2}^k) - \mathcal{G}_{\Delta T}^*(\gamma_{n+2}^k). \tag{2.53}$$

L'opérateur $\mathcal{F}_{\Delta T}^*$ (respectivement l'opérateur $\mathcal{G}_{\Delta T}^*$) représente l'opérateur adjoint de $\mathcal{F}_{\Delta T}$ (respectivement l'opérateur adjoint de $\mathcal{G}_{\Delta T}$) c'est à dire le propagateur rétrograde.

Cadre local : À ce niveau local (i.e. sur les intervalles I_n) nous optimisons d'une manière parallèle les fonctionnelles $\mathcal{J}_n$ (voir définition dans (2.10)). Nous minimisons localement par la méthode de gradient qui s'écrit sous la forme :

$$\tilde{v}_n^{k,g+1} = \tilde{v}_n^{k,g} - \rho_n^{k,g} \nabla \mathcal{J}_n(\tilde{v}_n^{k,g}),$$

où $1 \leq g \leq g_{\max}$ est l'indice des itérations locales (en parallèle). Si $g = 1$ on a :

$$\tilde{v}_n^{k,g} = v_n^k,$$

qui donne la mise à jour au niveau global, telle que :

$$\tilde{v}_n^{k+1} = \tilde{v}_n^{k,g_{\max}+1}.$$

2.4.5 L'algorithme PITPOC : Parareal Intermediate Target for Parallel Optimal Control

L'algorithme PITPOC diminue la complexité de calcul, surtout pour la partie séquentielle de l'algorithme SITPOC, par l'intermédiaire d'un couplage entre l'algorithme SITPOC et l'algorithme pararéel.

Calcul parallèle des cibles et conditions initiales intermédiaires

L'algorithme est présenté à la page 31. Pour connaître les définitions des mots clés voir page 20.

Proposition 2.4.12 (Monotonie du processus)**.**
La suite $(\Phi^k)_{k\geq 0}$ *définie dans l'équation* (2.46) *est décroissante :*

$$\Phi^k \geq \Phi^{k+1}, \quad \forall k \in \mathbb{N}$$

Démonstration. Rappelons tout d'abord les formules des dérivées première et seconde évaluées en zéro de la fonction $\Phi_k(\theta)$ associée à la k-ième itération.

$$\Phi_k'(0) = \alpha \int_I \langle \tilde{v}^{k+1} - v^k, v^k \rangle_2 dt + \langle \tilde{\lambda}_{\hat{n}}^{k+1} - \lambda_{\hat{n}}^k, \lambda_{\hat{n}}^k - y^{cible} \rangle_2, \tag{2.54}$$

$$\Phi_k''(0) = \alpha \int_I \|\tilde{v}^{k+1} - v^k\|_2^2 dt + \|\tilde{\lambda}_{\hat{n}}^{k+1} - \lambda_{\hat{n}}^k\|_2^2. \tag{2.55}$$

La fonction $\Phi_k(\theta)$ est quadratique en θ et atteint son minimum Φ^{k+1} au point θ^k qui vaut :

$$\theta^k = -\frac{\Phi_k'(0)}{\Phi_k''(0)}. \tag{2.56}$$

θ^k est donc solution de $\Phi_k'(\theta) = 0$. En effet, un développement limité à l'ordre deux au voisinage de zéro de la fonction $\Phi_k(\theta)$ permet d'écrire :

$$\Phi_k(\theta) = \Phi_k(0) + \theta \Phi_k'(0) + \frac{\theta^2}{2} \Phi_k''(0), \tag{2.57}$$

Algorithme 2 : PITPOC : Parareal Intermediaite Target for Parallel Optimal Control

Input : $\epsilon, \hat{n}(\#slave\,proc), y_0, y^T, \lambda_0^0 := y(0); k \leftarrow 0;$

repeat

  if *master processor* **then**

    if *k=0* **then**

      Evolve y^k through (2.6) using coarse time step;

      Evolve p^k through (2.7) using coarse time step;

      Compute $(\lambda_n^k)_{0\leq n\leq \hat{n}}$ with (2.16);

      Compute $(\gamma_n^k)_{\hat{n}\geq n\geq 1}$ with (2.17);

    else

      Update $v^k \leftarrow v^{k-1} + \theta^{k-1}(\tilde{v}^k - v^{k-1})$;

      Compute $\left(\lambda_n^k(\theta^{k-1})\right)_{0\leq n\leq \hat{n}}$, by (2.51);

      /* Hence, one gets $\lambda_{\hat{n}}^k$ which involves $p^k(T)$ */

      Compute $\left(\gamma_n^k\right)_{\hat{n}\geq n\geq 1}$, by (2.53);

    end

    foreach $n \in \{0, .., \hat{n}-1\}$ **do**

      $\xi_{n+1}^k \leftarrow \lambda_{n+1}^k - \gamma_{n+1}^k; v_n^k \leftarrow v_{|_{I_n}}^k$;

      Mpi_Send $(v_n^k, \lambda_n^k, \xi_{n+1}^k, processor(n))$;

    end

  else

    forall $slave\,processors(n)/n \in \{0, \ldots, \hat{n}-1\}$ **do**

      Mpi_Recv $(v_n^k, \lambda_n^k, \xi_{n+1}^k, master\,processor)$;

      $\tilde{v}_n^{k,1} \leftarrow v_n^k$;

      for $g = 1 : g_{max}$ **do**

        Evolve $\tilde{y}_n^{k,g} := y_n(\tilde{v}_n^{k,g})$ through (2.12);

        Evolve $\tilde{p}_n^{k,g} := p_n(\tilde{v}_n^{k,g})$ through (2.13);

        $\nabla \mathcal{J}_n(\tilde{v}_n^{k,g}) \leftarrow \alpha \tilde{v}_n^{k,g} + B^* \tilde{p}_n^{k,g}$;

        $\tilde{v}_n^{k,g+1} \leftarrow v_n^{k,g} - \rho_n^{k,g} \nabla \mathcal{J}_n(\tilde{v}_n^{k,g})$;

      end

      $\tilde{v}_n^{k+1} \leftarrow \tilde{v}^{k,g_{\max}+1}$;

      Mpi_Send $(\tilde{v}_n^{k+1}, \mathcal{F}_{\Delta T}(\lambda_n^k, \tilde{v}_n^{k,g_{\max}}), \mathcal{F}^*_{\Delta T}(\gamma_{n+1}^k), master\,processor)$;

    end

  end

  if *master processor* **then**

    foreach $n \in \{0, .., \hat{n}-1\}$ **do**

      Mpi_Recv $(\tilde{v}_n^{k+1}, \mathcal{F}_{\Delta T}(\lambda_n^k, \tilde{v}_n^{k,g_{\max}}), \mathcal{F}^*_{\Delta T}(\gamma_{n+1}^k), processor(n))$;

    end

    $\tilde{v}^{k+1} \leftarrow [\tilde{v}_0^{k+1}, .., \tilde{v}_n^{k+1}, .., \tilde{v}_{\hat{n}-1}^{k+1}]$;

    Compute $\tilde{\lambda}_{n+1}^{k+1}$ with (2.52);

    $\theta^k \leftarrow -\Phi_k'(0)/\Phi_k''(0)$; Evaluate Err according to (2.21);

  end

  $k \leftarrow k+1$;**Mpi_Broadcast** $(master\,processor, Err)$;

until $Err \leq \epsilon$;

Result : v^k is the optimal control

ce qui donne la formule (2.56) désignant le minimum d'une telle parabole.
En remplaçant le paramètre θ de l'équation (2.57) par θ^k on obtient :

$$\Phi^k - \Phi^{k+1} = \frac{1}{2}\frac{[\Phi_k'(0)]^2}{\Phi_k''(0)} = \frac{(\theta^k)^2}{2}\Phi_k''(0). \tag{2.58}$$

Cette quantité représente la variation de la suite $(\Phi^k)_{k\geq 0}$ calculée par l'algorithme. $\Phi_k''(0)$ est toujours positif, on aura donc $\Phi^k - \Phi^{k+1} \geq 0$ ce qui achève la démonstration. □

2.4.6 Complexité des algorithmes et stratégie de décomposition

La stratégie de décomposition varie considérablement d'une méthode parallèle à une autre. En effet, elle est principalement reliée à la complexité algorithmique et fatalement à l'informatique. Dans tous nos tests et stratégies de décomposition de domaine temporel, nous considérons des tâches parallèles équivalentes.

Remarque 2.4.13. *Suivant la linéarité du problème de contrôle optimal, il suffit de produire en parallèle les $\tilde{\lambda}^{k+1}$ pour ensuite avoir la relaxation optimale puis utiliser la formule* (2.51) *afin de produire les λ^{k+1}. Par ailleurs, la linéarité du contrôle donnée dans* (2.50) *aide à réduire un calcul important d'une évolution fine séquentielle, voire grossière séquentielle dans le cas du deuxième algorithme.*

La complexité $\mathcal{C}_{\mathcal{F}}$ (respectivement $\mathcal{C}_{\mathcal{G}}$) relative au solveur fin (respectivement grossier) sur un sous-intervalle de temps avec $\mathcal{C}_{\mathcal{F}} = \dfrac{\Delta T}{\tau_F}$ (respectivement : $\mathcal{C}_{\mathcal{G}} = \dfrac{\Delta T}{\tau_G}$), avec ΔT (respectivement : τ_F et τ_G) est la taille du sous-intervalle de temps I_n (respectivement : le pas de temps fin et grossier). On introduit le rapport r entre les pas de discrétisation temporelle τ_F et τ_G

$$r := \frac{\tau_G}{\tau_F}.$$

Le défi principal des superordinateurs calculateurs est la communication inter-processeurs. Cette tâche est déterminante dans le calcul parallèle. Les *clusters*[5] qui sont constitués (par exemple) de machines de type nouvelle génération quadri-cœur les nombres 4 et 8 des processeurs employés sont très déterminants de l'accélération des communications et donc de la résolution en général. En effet, si nous utilisons 4 cœurs alors la communication s'effectue au niveau des mémoires vive locales, de même pour 8 mais cette fois-ci avec une mémoire locale se situant un peu plus loin que celle du 4 cœurs.
Généralement les structures MPI des ordinateurs parallèles possèdent une synchronisation automatique, ce qui rend le calcul plus simple. Nous sommes donc partis sur cette voie au niveau de l'implémentation de notre algorithme pour les mises à jour des variables, notamment celle du contrôle.
L'efficacité de la parallélisation d'un calcul est liée à :

1. la manière de structurer un algorithme,
2. la machine avec laquelle on traite le problème,
3. la manière de coder.

5. C'est un assemblage réseau de machines puissantes qui forme un calculateur géant. On l'appelle aussi frontale.

Dans notre mise en œuvre nous avons exploité :

La linéarité du problème de contrôle selon $y(v+\delta v) = y(v)+y(\delta v)$, après avoir effectué une première itération d'optimisation nécessitant $3\hat{n}\,\mathcal{C}_{\mathcal{F}}$ opérations, cette équation sera opérationnelle.

La structure SPMD [6] automatise la synchronisation des données. Cela nous permet de réduire considérablement le transit des informations de plusieurs dizaines de réels contenues dans un vecteur, à un seul réel (qui est la relaxation θ).

D'un point de vue algorithmique, pour un nombre pré-déterminé $g_{\max}$ d'itérations locales de sous-optimisations, nous cumulons à une itération $k > 1$ les complexités suivantes respectivement pour la première et la deuxième approche.

$$\mathcal{C}_{\text{SITPOC}}(\hat{n}, g_{\max}) = \big((2k+1)\hat{n} + 2kg_{\max}\big)\mathcal{C}_{\mathcal{F}},$$
$$\mathcal{C}_{\text{PITPOC}}(\hat{n}, g_{\max}) = \big(\frac{2k+1}{r}\hat{n} + k(2g_{\max}+1)\big)\mathcal{C}_{\mathcal{F}}.$$

En effet, si $k = 1$ le terme $3\hat{n}\mathcal{C}_{\mathcal{F}}$ (respectivement $3\hat{n}\mathcal{C}_{\mathcal{G}}$) représente les trois évolutions lors de la première itération, le long de l'intervalle I, de l'état, de son adjoint et de sa perturbation [7] pour un calcul fin (respectivement grossier).
Notons que si $r \geq 2$, la formule de complexité de l'algorithme PITPOC sera :

$$\begin{aligned}\mathcal{C}_{\text{PITPOC}}(\hat{n}, g_{\max}) &= \big(\frac{2k+1}{r}\hat{n} + k(2g_{\max}+1)\big)\frac{T}{\hat{n}\tau_F} \\ &= \big(\frac{2k+1}{r} + \frac{k(2g_{\max}+1)}{\hat{n}}\big)\frac{T}{\tau_F}. \end{aligned} \tag{2.59}$$

2.5 Interprétation matricielle de la méthode et quelques extensions

Dans cette section, nous présentons le problème de contrôle optimal discret et nous proposons une interprétation matricielle de la méthode des cibles et conditions initiales intermédiaires. Nous établissons également un lien entre nos deux algorithmes de contrôle optimal parallèle d'un coté et la méthode de Jacobi par bloc d'un autre coté.

2.5.1 Discrétisation du problème de contrôle optimal

Rappelons la définition du coût quadratique associé au problème d'optimisation :

$$\mathcal{J}(v) := \frac{1}{2}\|y(T) - y^{cible}\|_2^2 + \frac{\alpha}{2}\int_0^T \|v(t)\|_2^2 dt.$$

La contrainte que vérifie l'état du système est une équation aux dérivées partielles parabolique :

$$\begin{cases} \partial_t y - \nu\Delta y = \mathcal{B}v, & \text{sur } [0,T]\times\Omega \\ y(0) = y_0, & \text{sur } \{t=0\}\times\Omega, \\ + \text{ Conditions aux bords.} \end{cases} \tag{2.60}$$

6. En Anglais, Single Program Multiple Data, aide à la reconnaissance des variables dans un seul programme par tout les processeurs.

7. C'est l'état contrôlé avec une perturbation δv de contrôle à partir de la condition initiale : la fonction zéro.

où y_0 appartient à $L^2(\Omega)$ et l'opérateur $\mathcal{B}$ est un opérateur de $\mathcal{L}\big(L^2(I, L^2(\Omega)), L^2(I, L^2(\Omega_c))\big)$. Présentons les outils nous permettant de discrétiser le problème de contrôle optimal. Maillons le domaine spatial Ω par une triangulation $\mathcal{T}_h(\Omega)$ afin d'approcher, par une méthode des éléments finis, les solutions de notre problème. Considérons l'élément fini $\mathbb{P}_1$ (polynôme de degré 1) pour l'approximation de l'état $y(t,.) \in \mathcal{U} := H^1(\Omega)$ et l'élément fini $\mathbb{P}_0$ (constantes par élément) pour l'approximation de la variable de contrôle $v(t,.) \in L^2(\Omega)$.

Soit donc l'ensemble $\{\psi_i\}_{i=1}^{\hat{s}}, \hat{s} > 1$, des fonctions de base de l'espace des éléments finis $\mathcal{U}_h \subset \mathcal{U}$ approchant les solutions de l'état $y(t,.)$ et l'ensemble $\{\phi_i\}_{i=1}^{\hat{q}}$ de fonctions de base de l'espace des éléments finis $V_h \subset L^2(\Omega)$ approchant la variable de contrôle $v(t,.)$.

Notons les matrices de masse $\mathcal{M}_h \in \mathbb{R}^{\hat{s}\times\hat{s}}$ et de rigidité $\mathcal{A}_h \in \mathbb{R}^{\hat{s}\times\hat{s}}$ respectivement par $(\mathcal{M}_h)_{ij} := \langle\psi_i, \psi_j\rangle$ et $(\mathcal{A}_h)_{ij} := \nu\langle\nabla\psi_i, \nabla\psi_j\rangle$. Notons la matrice de masse $\mathcal{R}_h \in \mathbb{R}^{\hat{q}\times\hat{q}}$ relative au contrôle définie par $(\mathcal{R}_h)_{ij} := \langle\phi_i, \phi_j\rangle$ et notons également la discrétisation de l'opérateur $\mathcal{B}$ par $\mathcal{B}_h \in \mathbb{R}^{\hat{s}\times\hat{q}}$; $(\mathcal{B}_h)_{ij} := \langle\psi_i, \mathcal{B}\phi_j\rangle$, où $\langle.,.\rangle$ est le produit scalaire de l'espace $L^2(\Omega)$. Notons que $\mathcal{B}_h$ est nulle pour des indices correspondant à des points hors du support Ω_c du contrôle. Par une méthode de pénalisation nous intégrons les conditions aux bords du domaine Ω dans la matrice de masse s'il s'agit des conditions de Dirichlet (e.g. le cas où $y(t,.) \in H_0^1(\Omega)$), autrement ces conditions figureront au second membre du système à résoudre.

Par convention nous notons $y \in \mathbb{R}^{\hat{s}}$ et $v \in \mathbb{R}^{\hat{q}}$ les représentations nodales, sur le maillage $\mathcal{T}_h(\Omega)$, de l'état $y(t,.) \in \mathcal{U}$ et du contrôle $v(t,.) \in L^2(\Omega)$, et afin de montrer leurs dépendances en temps nous notons $\bar{y}$ et $\bar{v}$ respectivement.

En écrivant la formulation variationnelle de l'équation de l'état du système (2.60), et en choisissant les fonctions tests égales aux fonctions de bases. Cette discrétisation spatiale prend alors la forme suivante :

$$\mathcal{M}_h\dot{\bar{y}} + \mathcal{A}_h\bar{y} = \mathcal{B}_h\bar{v}. \tag{2.61}$$

Pour la discrétisation en temps, nous considérons $\hat{l}$ nœuds de l'intervalle de temps I et nous utilisons la méthode d'Euler implicite. Alors le schéma en temps s'écrit :

$$(\mathcal{M}_h + \tau\mathcal{A}_h)y_{l+1} - \mathcal{M}_h y_l = \tau_F\mathcal{B}_h v_{l+1}, \tag{2.62}$$

où $0 \leq l \leq \hat{l} = \frac{T}{\tau_F}$ et τ_F est le pas de discrétisation en temps tel que $t_l = l \times \tau_F$, et $y_l \approx \bar{y}(t_l)$ le vecteur colonne de taille $\hat{s}$ de l'état à l'instant t_l avec $0 \leq l \leq \hat{l}$, également $v_l \approx \bar{v}(t_l)$ le vecteur colonne de taille $\hat{q}$ du contrôle à l'instant t_l.

La forme tensorielle temps-espace est représentée par des matrices par blocs. Soit

$$F := \mathcal{M}_h, \text{ et } F_{\tau_F} := \mathcal{M}_h + \tau_F\mathcal{A}_h$$

alors on a

$$F_{\tau_F}y_{l+1} = Fy_l + \tau_F\mathcal{B}_h v_{l+1}.$$

Posons $\mathbf{y} := [y_1^T, .., y_l^T, ..., y_{\hat{l}}^T]^T \in \mathbb{R}^{\hat{l}\hat{s}}$ et $\mathbf{v} := [v_1^T, .., v_l^T, ..., v_{\hat{l}}^T]^T \in \mathbb{R}^{\hat{l}\hat{q}}$ les vecteurs colonne représentant respectivement l'état direct et le contrôle sur tout le temps discret (avec cette notation un vecteur de dimension $\hat{l}\hat{s}$ (resp $\hat{l}\hat{q}$) est un grand vecteur de $\hat{l}$ sous-vecteurs de tailles $\hat{s}$ (resp $\hat{q}$)).

Dans ce qui suit et selon notre notation, une matrice par bloc dans $\mathbb{R}^{(ab)\times(cd)}$ possède a blocs lignes et c blocs colonnes où chaque blocs est une matrices de tailles $b \times d$.

L'état discret du système est alors régi par l'équation matricielle suivante :

$$\mathbf{E}\mathbf{y} + \mathbf{B}\mathbf{v} = \mathbf{C},$$

où $\mathbf{E} \in \mathbb{R}^{(\hat{ls})\times(\hat{ls})}$ et $\mathbf{B} \in \mathbb{R}^{(\hat{ls})\times(\hat{lq})}$, qui sont données par :

$$\mathbf{E} = \begin{bmatrix} F_{\tau_F} & 0 & 0 & 0 \\ -F & F_{\tau_F} & 0 & 0 \\ 0 & \ddots & \ddots & 0 \\ 0 & 0 & -F & F_{\tau_F} \end{bmatrix}, \text{ et } \mathbf{B} = -\tau_F \begin{bmatrix} \mathcal{B}_h & 0 & 0 & 0 \\ 0 & \mathcal{B}_h & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \mathcal{B}_h \end{bmatrix}$$

Le vecteur colonne $\mathbf{C} \in \mathbb{R}^{\hat{ls}}$ représente les conditions initiales du problème, il est défini par $\mathbf{C} := [(Fy_0)^T, \mathbf{0}^T ..., \mathbf{0}^T]^T$ où $y_0 \in \mathbb{R}^{\hat{s}}$ est le vecteur condition initiale et $\mathbf{0} = [0, \ldots, 0]^T$.

Afin d'approcher l'intégrale en temps dans la fonctionnelle de coût $\mathcal{J}$ (présentée au début de la section) nous utilisons la méthode des trapèzes. Nous pouvons donc donner une écriture matricielle de la fonctionnelle $\mathcal{J}$ qui prend la forme $\mathbf{J}^{h,\tau}$ suivante :

$$\mathbf{J}^{h,\tau}(\mathbf{v}, \mathbf{y}) := \frac{1}{2}(\mathbf{y} - \bar{\bar{\xi}}_{\hat{l}})^T \mathbf{K}(\mathbf{y} - \bar{\bar{\xi}}_{\hat{l}}) + \frac{1}{2}\mathbf{v}^T \mathbf{G}\mathbf{v}, \tag{2.63}$$

où la matrice $\mathbf{K} \in \mathbb{R}^{(\hat{ls})\times(\hat{ls})}$ est diagonale par blocs, définie par $\mathbf{K} := \text{Blockdiag}(\mathbf{0}, \ldots, \mathbf{0}, F)$, le vecteur cible $\bar{\bar{\xi}}_{\hat{l}} \in \mathbb{R}^{\hat{ls}}$ est un vecteur colonne défini par $\bar{\bar{\xi}}_{\hat{l}} := [\mathbf{0}^T, \ldots, \mathbf{0}^T, \xi_{\hat{l}}^T]^T$ où $\xi_{\hat{l}} \in \mathbb{R}^{\hat{s}}$ est la représentation nodale de la cible y^{cible}, et la matrice $\mathbf{G} \in \mathbb{R}^{(\hat{lq})\times(\hat{lq})}$ est définie par

$$\mathbf{G} = \tau_F \alpha \begin{bmatrix} \frac{1}{2}\mathcal{R}_h & 0 & 0 & 0 & 0 \\ 0 & \mathcal{R}_h & 0 & 0 & 0 \\ 0 & 0 & \ddots & 0 & 0 \\ 0 & 0 & 0 & \mathcal{R}_h & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2}\mathcal{R}_h \end{bmatrix}.$$

Remarque 2.5.1. *Par abus de notation, les symboles λ, ξ et γ correspondront à des représentations nodales, ce seront donc des vecteurs de dimensions finies plutôt que des fonctions continues.*

2.5.2 Système d'optimalité global discret

Afin de calculer les points critiques, nous introduisons le Lagrangien discret $\mathcal{L}_h^\tau$ défini par :

$$\mathcal{L}_h^\tau(\mathbf{y}, \mathbf{v}, \mathbf{p}) := \mathbf{J}^{h,\tau}(\mathbf{v}, \mathbf{y}) + \mathbf{p}^T(\mathbf{E}\mathbf{y} + \mathbf{B}\mathbf{v} - \mathbf{C}), \tag{2.64}$$

où $\mathbf{p}$ est le multiplicateur de Lagrange associé à la contrainte que vérifie l'état $\mathbf{y}$, ce nouvel état portera le nom d'état adjoint discret.
Les points critiques du Lagrangien (2.64) sont solution du système linéaire suivant :

$$\underbrace{\begin{bmatrix} \mathbf{E} & \mathbf{B} & \mathbf{0} \\ \mathbf{0} & \mathbf{G} & \mathbf{B^T} \\ \mathbf{K} & \mathbf{0} & \mathbf{E^T} \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} \mathbf{y} \\ \mathbf{v} \\ \mathbf{p} \end{bmatrix}}_{\mathbf{X}} = \underbrace{\begin{bmatrix} \mathbf{C} \\ \mathbf{0} \\ \mathbf{K}\bar{\bar{\xi}}_{\hat{l}} \end{bmatrix}}_{\mathbf{b}}. \tag{2.65}$$

Nous venons de développer la discrétisation du problème de contrôle optimal séquentiel sur l'intervalle de temps $[0,T]$. Dans ce qui suit nous développons la discrétisation du problème de contrôle optimal parallèle. Nous considérons donc les systèmes d'optimalité locaux.

2.5.3 Systèmes d'optimalité locaux discrets

Rappelons que $\hat{n}$ est le nombre des sous-intervalles I_n de taille $\Delta T := \frac{T}{\hat{n}}$, et que $T_n = n\Delta T$ pour $0 \leq n \leq \hat{n}$. Les sous-intervalles I_n sont à leurs tours décomposés, cette fois-ci, en $\hat{m}$ pas de temps τ_F (i.e. $\hat{l} = \hat{n}\hat{m}$). On a alors $\frac{\Delta T}{\hat{m}} = \tau_F$.
Dans ce qui suit, nous considérons un indice $j_n := \dfrac{T_n}{\tau_F}$ dont l'indice n parcourt $0, 1, \ldots, \hat{n}$. On a donc

$$j_n + \hat{m} = j_{n+1}.$$

Remarque 2.5.2. *Les symboles en gras indexés par " n" (et notamment les matrices par blocs) correspondent aux sous-intervalles I_n.*

On définit le vecteur colonne $\mathbf{y}_n := [y^T_{j_{n-1}+1}, .., y^T_{j_n}]^T \in \mathbb{R}^{\hat{m}\hat{s}}$ (i.e. un vecteur composé de $\hat{m}$ sous-vecteurs de tailles $\hat{s}$), pour $1 \leq n \leq \hat{n}$, de sorte que nous avons la réécriture par bloc du vecteur de l'état direct $\mathbf{y} = [\mathbf{y}^T_1, \ldots, \mathbf{y}^T_n, \ldots, \mathbf{y}^T_{\hat{n}}]^T \in \mathbb{R}^{\hat{l}\hat{s}}$, et également du vecteur de l'état adjoint $\mathbf{p} = [\mathbf{p}^T_1, \ldots, \mathbf{p}^T_n, \ldots, \mathbf{p}^T_{\hat{n}}]^T \in \mathbb{R}^{\hat{l}\hat{s}}$ avec $\mathbf{p}_n := [p^T_{j_{n-1}}, .., p^T_{j_n-1}]^T \in \mathbb{R}^{\hat{m}\hat{s}}$, pour $1 \leq n \leq \hat{n}$. On définit enfin l'ensemble des vecteurs colonnes correspondant aux cibles intermédiaires $(\xi_{j_n})_{1\leq n\leq \hat{n}} \in \mathbb{R}^{\hat{s}}$ par :

$$\xi_{j_n} := y_{j_n} - p_{j_n}. \tag{2.66}$$

Dont nous identifions : $\xi_{j_{\hat{n}}} = \xi_{\hat{l}}$.

Après avoir attribué des cibles et conditions initiales aux sous-intervalles de temps, nous sommes maintenant en mesure d'introduire les systèmes parallèles d'optimalité. Dans ce qui suit nous développons ces systèmes locaux discrets qui se résolvent simultanément.

Nous supposons données des vecteurs $\tilde{\mathbf{y}} := [\tilde{y}_1, .., \tilde{y}_{\hat{l}}]^T$ et $\tilde{\mathbf{p}} := [\tilde{p}_1, .., \tilde{p}_{\hat{l}}]^T$ de sorte à pouvoir définir les cibles selon (2.66) par $\bar{\xi}_{j_n} := \tilde{y}_{j_n} - \tilde{p}_{j_n}$ ainsi définir $(\bar{\xi}_{j_n})_{n=1,..\hat{n}}$ comme une suite de vecteurs colonnes de $\mathbb{R}^{\hat{m}\hat{s}}$ définis par $\bar{\xi}_n := [\mathbf{0}^T, \ldots, \mathbf{0}^T, \tilde{\xi}^T_{j_n}]^T$. Introduisons ces nouvelles cibles dans la définition du coût $\mathcal{J}$ (voir équation (2.10)) qui après discrétisation nous obtenons la fonctionnelle discrète $\mathbf{J}^{h,\tau}_n$ suivante pour tout $n = 1, \ldots, \hat{n}$:

$$\mathbf{J}^{h,\tau}_n(\mathbf{v}_n, \mathbf{y}_n) := \frac{1}{2}(\mathbf{y}_n - \bar{\xi}_n)^T \mathbf{K}_n (\mathbf{y}_n - \bar{\xi}_n) + \frac{1}{2}\mathbf{v}^T_n \mathbf{G}_n \mathbf{v}_n, \tag{2.67}$$

où $\mathbf{K}_n := \text{Blockdiag}(\mathbf{0}, \ldots, \mathbf{0}, F) \in \mathbb{R}^{(\hat{m}\hat{s})\times(\hat{m}\hat{s})}$ et $\mathbf{G}_n \in \mathbb{R}^{(\hat{m}\hat{q})\times(\hat{m}\hat{q})}$ est telle que :

$$\mathbf{G}_n = \tau_F \alpha \begin{bmatrix} \frac{1}{2}\mathcal{R}_h & 0 & 0 & 0 & 0 \\ 0 & \mathcal{R}_h & 0 & 0 & 0 \\ 0 & 0 & \ddots & 0 & 0 \\ 0 & 0 & 0 & \mathcal{R}_h & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2}\mathcal{R}_h \end{bmatrix},$$

$\mathbf{v}_n := [v_{j_{n-1}}, .., v_{j_n-1}]^T$ tel que $\mathbf{v} = [\mathbf{v}_1, .., \mathbf{v}_{\hat{n}}]^T$.
Le sous-système des points critiques est obtenu par l'introduction du Lagrangien discret $\mathcal{L}_n^{h,\tau}$ associé à la contrainte que forme l'équation de l'état local défini sur l'intervalle de temps I_n. Elle doit être résolue en parallèle :

$$\mathbf{E}_n\mathbf{y}_n + \mathbf{B}_n\mathbf{v}_n = \mathbf{C_n}, \tag{2.68}$$

où $\mathbf{C}_n := [(F\tilde{y}_{j_{n-1}})^T, \mathbf{0}^T, \ldots, \mathbf{0}^T]^T \in \mathbb{R}^{\hat{m}\hat{s}}$ est un vecteur colonne représentant les conditions initiales pour chaque problème local et le vecteur colonne $\tilde{y}_{j_{n-1}} \in \mathbb{R}^{\hat{s}}$ est un élément d'un vecteur colonne par blocs $\tilde{\mathbf{y}} \in \mathbb{R}^{\hat{l}\hat{s}}$ supposé donné. Tandis que la matices $\mathbf{E}_n := \text{Blocklowerdiag}(F_{\tau_F}, -F) \in \mathbb{R}^{(\hat{m}\hat{s})\times(\hat{m}\hat{s})}$;(i.e. $\mathbf{E}_{ii} = F_{\tau_F}$), la matrice $\mathbf{B}_n \in \mathbb{R}^{(\hat{m}\hat{s})\times(\hat{m}\hat{q})}$

$$\mathbf{B}_n = -\tau_F \begin{bmatrix} \mathcal{B}_h & 0 & 0 & 0 \\ 0 & \mathcal{B}_h & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \mathcal{B}_h \end{bmatrix}$$

Le Lagrangien discret est donc défini par :

$$\mathcal{L}_n^{h,\tau}(\mathbf{y}_n, \mathbf{v}_n, \mathbf{p}_n) := \mathbf{J}_n^{h,\tau}(\mathbf{v}_n, \mathbf{y}_n) + \mathbf{p}_n^T(\mathbf{E}_n\mathbf{y}_n + \mathbf{B}_n\mathbf{v}_n - \mathbf{C_n}). \tag{2.69}$$

L'état adjoint $\mathbf{p}_n$ est un multiplicateur de Lagrange pour la contrainte (2.68) vérifiée par l'état local (dont la résolution est réalisée en parallèle).

Le sous-système des points critiques est donc

$$\underbrace{\begin{bmatrix} \mathbf{E}_n & \mathbf{B}_n & \mathbf{0} \\ \mathbf{0} & \mathbf{G}_n & \mathbf{B}_n^T \\ \mathbf{K}_n & \mathbf{0} & \mathbf{E}_n^T \end{bmatrix}}_{\mathbf{A}_{nn}} \underbrace{\begin{bmatrix} \mathbf{y}_n \\ \mathbf{v}_n \\ \mathbf{p}_n \end{bmatrix}}_{X_n} = \underbrace{\begin{bmatrix} \mathbf{C}_n \\ \mathbf{0} \\ \mathbf{K}_n\bar{\xi}_n \end{bmatrix}}_{b_n}. \tag{2.70}$$

2.5.4 Représentation matricielle de la méthode des cibles intermédiaires

Dans cette sous-section nous illustrons d'un point de vue matriciel la propriété de décomposition utilisée par la méthode des cibes intermédiaires sur les problèmes de contrôle optimal. Nous introduisons pour cet effet le théorème suivant, qui nous sert par la suite à illustrer en termes matriciels les étapes des algorithmes présentés dans ce chapitre.

Théorème 2.5.3. *Pour toute valeur de $\hat{n} \geq 2$ le système $\mathbf{AX} = \mathbf{b}$ relatif au problème global de contrôle optimal (voir (2.65)) est décomposable en sous systèmes de taille plus petite, relatifs au sous problèmes de contrôle optimal, de type $\mathbf{A}_{nn}X_n = b_n$ (voir (2.70)) et reliés à travers les seconds membres $(b_n)_{n=1,..,\hat{n}}$. La somme des tailles de ces sous systèmes est égale à la taille du système global et la décomposition employée est arbitraire.*

Dans la preuve ci-dessous, nous ne détaillons pas toutes les dimensions des matrices par blocs issues de la décomposition afin de donner une structure arbitraire à cette décomposition. Cependant, nous supposons que les dimensions de ces matrices (e.g $\mathbf{E}_{ii}, i = 1, 2$) sont compatibles à la dimension de la matrice mère (i.e. $\mathbf{E}$).

Démonstration. On procède par récurrence. On montre tout d'abord que le système (2.65) est décomposable en deux sous systèmes disjoints. Puis on suppose que le résultat est vrai à un rang $\hat{n} > 2$ de sous-domaines et on montre que la formule est vrai à l'ordre $\hat{n}+1$.
Décomposition en $\hat{n} = 2$ sous-domaines temporels de tailles arbitraires :
On prend $\mathbf{y} = [\mathbf{y}_1^T, \mathbf{y}_2^T]^T$, $\mathbf{v} = [\mathbf{v}_1^T, \mathbf{v}_2^T]^T$ et $\mathbf{p} = [\mathbf{p}_1^T, \mathbf{p}_2^T]^T$.

La matrice globale à décomposer est la suivante :

$$\begin{bmatrix} \mathbf{E} & \mathbf{B} & \mathbf{0} \\ \mathbf{0} & \mathbf{G} & \mathbf{B}^T \\ \mathbf{K} & \mathbf{0} & \mathbf{E}^T \end{bmatrix}.$$

Nous rappelons que toutes les matrices par blocs sont diagonales par blocs à l'exception de $\mathbf{E}$ qui est bi-diagonale inférieur par blocs. Par conséquent, si nous découpons en deux cette matrice à la $\mathbf{i}$-ème ligne bloc et à la $\mathbf{j}$-ème colonne bloc, nous obtenons quatre matrices par blocs telle que

$$\begin{bmatrix} \mathbf{E}_{11} & \mathbf{0} \\ \mathbf{E}_{21} & \mathbf{E}_{22} \end{bmatrix}.$$

où $\mathbf{E}_{11} \in \mathbb{R}^{(\mathbf{i}\hat{s})\times(\mathbf{j}\hat{s})}$ et $\mathbf{E}_{22} \in \mathbb{R}^{((\hat{l}-\mathbf{i})\hat{s})\times((\hat{l}-\mathbf{j})\hat{s})}$ sont des matrices bi-diagonales par blocs de la même forme que $\mathbf{E}$. La matrice $\mathbf{E}_{21}$ est donc une matrice vérifiant $(\mathbf{E}_{21})_{lk} = 0$ pour tout bloc colonne k avec $0 \leq k \leq \mathbf{j}-1$ et tout bloc ligne l avec $\mathbf{i}+2 \leq l \leq \hat{l}$ et que $(\mathbf{E}_{21})_{(\mathbf{i}+1)\mathbf{j}} = -F$.
Après décomposition nous avons :

$$\left[\begin{array}{cc|cc|cc} \mathbf{E}_{11} & 0 & \mathbf{B}_{11} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{E}_{21} & \mathbf{E}_{22} & \mathbf{0} & \mathbf{B}_{22} & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{G}_{11} & \mathbf{0} & \mathbf{B}_{11}^T & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{G}_{22} & \mathbf{0} & \mathbf{B}_{22}^T \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{E}_{11}^T & \mathbf{E}_{21}^T \\ \mathbf{0} & \mathbf{K}_{22} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{E}_{22}^T \end{array}\right] \left[\begin{array}{c} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \hline \mathbf{v}_1 \\ \mathbf{v}_2 \\ \hline \mathbf{p}_1 \\ \mathbf{p}_2 \end{array}\right] = \left[\begin{array}{c} \mathbf{C} \\ \mathbf{0} \\ \hline \mathbf{0} \\ \mathbf{0} \\ \hline \mathbf{0} \\ \mathbf{K}_{22}\bar{\xi}_2 \end{array}\right]. \tag{2.71}$$

Par un réarrangement du vecteur inconnu, nous obtenons le système équivalent suivant :

$$\begin{bmatrix} \mathbf{E}_{11} & \mathbf{B}_{11} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{G}_{11} & \mathbf{B}_{11}^T & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{E}_{11}^T & \mathbf{0} & \mathbf{0} & \mathbf{E}_{21}^T \\ \mathbf{E}_{21} & \mathbf{0} & \mathbf{0} & \mathbf{E}_{22} & \mathbf{B}_{22} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{G}_{22} & \mathbf{B}_{22}^T \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{K}_{22} & \mathbf{0} & \mathbf{E}_{22}^T \end{bmatrix} \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{v}_1 \\ \mathbf{p}_1 \\ \mathbf{y}_2 \\ \mathbf{v}_2 \\ \mathbf{p}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{C} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{K}_{22}\bar{\xi}_2 \end{bmatrix}. \tag{2.72}$$

En ajoutant et en retranchant le produit matrice vecteur suivant $\mathbf{K}_{11}\mathbf{y}_1$ on obtient :

$$\boxed{\mathbf{K}_{11}\mathbf{y}_1} + \mathbf{E}_{11}^T\mathbf{p}_1 = \boxed{\mathbf{K}_{11}\mathbf{y}_1} - \mathbf{E}_{21}^T\mathbf{p}_2, \tag{2.73}$$

$$\mathbf{E}_{22}\mathbf{y}_2 + \mathbf{B}_{22}\mathbf{v}_2 = -\mathbf{E}_{21}\mathbf{y}_1, \tag{2.74}$$

où $\mathbf{K}_{11} := \text{Blockdiag}(\mathbf{0}, .., \mathbf{0}, F) \in \mathbb{R}^{(\mathbf{i}\hat{s})\times(\mathbf{j}\hat{s})}$. Alors, on obtient le système suivant

$$\left[\begin{array}{ccc|ccc} \mathbf{E}_{11} & \mathbf{B}_{11} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{G}_{11} & \mathbf{B}_{11}^T & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{K}_{11} & \mathbf{0} & \mathbf{E}_{11}^T & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{E}_{22} & \mathbf{B}_{22} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{G}_{22} & \mathbf{B}_{22}^T \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{K}_{22} & \mathbf{0} & \mathbf{E}_{22}^T \end{array}\right] \left[\begin{array}{c} \mathbf{y}_1 \\ \mathbf{v}_1 \\ \mathbf{p}_1 \\ \mathbf{y}_2 \\ \mathbf{v}_2 \\ \mathbf{p}_2 \end{array}\right] = \left[\begin{array}{c} \mathbf{C}_1 \\ \mathbf{0} \\ \mathbf{K}_{11}\mathbf{y}_1 - \mathbf{E}_{21}^T\mathbf{p}_2 \\ \hline -\mathbf{E}_{21}\mathbf{y}_1 \\ \mathbf{0} \\ \mathbf{K}_{22}\bar{\xi}_2 \end{array}\right]. \tag{2.75}$$

On note que la matrice $\mathbf{E}_{21}$ s'écrit

$$\mathbf{E}_{21} = \begin{bmatrix} 0 & \dots & -F \\ \vdots & \ddots & 0 \\ 0 & \dots & 0 \end{bmatrix}.$$

De plus on peut montrer simplement que $\mathbf{K}_{11}\mathbf{y}_1 = [0^T, \dots, 0^T, (Fy_{j_1})^T]^T$ et que $-\mathbf{E}_{21}^T\mathbf{p}_2 = [0^T, \dots, 0^T, -(Fp_{j_1})^T]^T$. Donc d'après la définition des cibles intermédiaires (2.66) on a :

$$\begin{aligned} \mathbf{K}_{11}\mathbf{y}_1 - \mathbf{E}_{21}^T\mathbf{p}_2 &= [0^T, \dots, 0^T, (F(y_{j_1} - p_{j_1}))^T]^T \\ &= [0^T, \dots, 0^T, (F\xi_{j_1})^T]^T \\ &= \mathbf{K}_{11}\bar{\xi}_1, \end{aligned}$$

et le vecteur $\bar{\xi}_1$ représente donc la cible relative au premier intervalle.

$$\left[\begin{array}{ccc|ccc} \mathbf{E}_{11} & \mathbf{B}_{11} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{G}_{11} & \mathbf{B}_{11}^T & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{K}_{11} & \mathbf{0} & \mathbf{E}_{11}^T & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{E}_{22} & \mathbf{B}_{22} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{G}_{22} & \mathbf{B}_{22}^T \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{K}_{22} & \mathbf{0} & \mathbf{E}_{22}^T \end{array}\right] \left[\begin{array}{c} \mathbf{y}_1 \\ \mathbf{v}_1 \\ \mathbf{p}_1 \\ \hline \mathbf{y}_2 \\ \mathbf{v}_2 \\ \mathbf{p}_2 \end{array}\right] = \left[\begin{array}{c} \mathbf{C}_1 \\ \mathbf{0} \\ \mathbf{K}_{11}\bar{\xi}_1 \\ \hline \mathbf{C}_2 \\ \mathbf{0} \\ \mathbf{K}_{22}\bar{\xi}_2 \end{array}\right], \tag{2.76}$$

où $\mathbf{C}_1 = [(Fy_0)^T, 0^T, .., 0^T]^T$ et $\mathbf{C}_2 = [-(\mathbf{E}_{21}\mathbf{y}_1)^T, 0^T, .., 0^T]^T = [(Fy_{j_1})^T, 0^T, .., 0^T]^T$.
Supposons maintenant que notre théorème est vrai à l'ordre $\hat{n}$, et montrons l'hérédité de la propriété. Le but est alors de donner une décomposition arbitraire du système global, associé au problème de contrôle sur [0,T] en $\hat{n}+1$ sous-systèmes de tailles S_n telles que $\sum_n S_n = S$ avec S est la taille du système global.
D'après l'hypothèse de récurrence, le système peut-etre décomposé en $\hat{n}$ systèmes de tailles $S_1, S_2, \dots, S_{\hat{n}-1}, S_{\hat{n}} + S_{\hat{n}+1}$.

Enfin, l'hypothèse au rang $\hat{n} = 2$ permet de décomposer le dernier sous-sytème de taille $S_{\hat{n}} + S_{\hat{n}+1}$ en deux sous-systèmes de tailles respectives $S_{\hat{n}}$ et $S_{\hat{n}+1}$. D'où le résultat. □

2.5.5 Interprétation matricielle de la résolution séquentielle d'un problème de contrôle optimal

La résolution séquentielle d'un problème de contrôle optimal, dans nos algorithmes, se fait par un schéma itératif. Ce schéma consiste à construire et à mettre à jour à chaque itération k un contrôle $\mathbf{v}^k$ par une méthode de descente de gradient de la fonctionnelle de coût (le cas où $k = 0$ le contrôle $\mathbf{v}^0$ est donné). Etant donné ce contrôle, à l'itération k le calcul du gradient est calculé via $\mathbf{v}^k$ qui alimente une propagation séquentielle de l'état direct $\mathbf{y}^k \equiv \mathbf{y}(\mathbf{v}^k)$, qui à son tour alimente à travers $\mathbf{y}_{j_{\hat{n}}}^k$ une propagation séquentielle rétrograde de l'état adjoint $\mathbf{p}^k \equiv \mathbf{p}(\mathbf{v}^k)$. Ensuite, nous formons une écriture explicite du gradient, tel que $\mathbf{G}\mathbf{v}^k + \mathbf{B}^\mathbf{T}\mathbf{p}^k$.

Si nous considérons le cas simple d'une méthode de descente de gradient à pas fixe θ, la mise à jour du contrôle est alors :

$$\mathbf{v}^{k+1} = \mathbf{v}^k - \theta\big(\mathbf{G}\mathbf{v}^k + \mathbf{B}^{\mathbf{T}}\mathbf{p}^k\big).$$

Toutes ces opérations peuvent donc être regroupées dans une représentation matricielle sous cette forme :

$$\underbrace{\begin{bmatrix} \mathbf{E} & \mathbf{B} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{K} & \mathbf{0} & \mathbf{E}^T \end{bmatrix}}_{\mathbf{M}} \begin{bmatrix} \mathbf{y} \\ \mathbf{v} \\ \mathbf{p} \end{bmatrix}^{k+1} = \begin{bmatrix} \mathbf{C} \\ \mathbf{0} \\ \mathbf{K}\bar{\bar{\xi}}_{\hat{n}} \end{bmatrix} - \underbrace{\begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \theta\mathbf{G} - \mathbf{I} & \theta\mathbf{B}^T \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}}_{\mathbf{N}} \begin{bmatrix} \mathbf{y} \\ \mathbf{v} \\ \mathbf{p} \end{bmatrix}^k, \quad \forall k \geq 0. \tag{2.77}$$

Le vecteurs état direct et adjont à l'itération zero (i.e. $\mathbf{y}^0$ et $\mathbf{p}^0$) sont donnés par le schéma ci-dessus avec une matrice $\mathbf{N}$ prise nulle !
Attention, nous n'inversons jamais de telle matrice par blocs (i.e. $\mathbf{M}$), c'est trop coûteux ! !
Cependant, par permutation des lignes le système devient triangulaire inférieur par bloc. Le calcul étant successif et dans l'ordre suivant :

$$\mathbf{v}^{k+1} = \mathbf{v}^k - \theta\Big(\mathbf{G}\mathbf{v}^k + \mathbf{B}^T\mathbf{p}^k\Big) \tag{2.78}$$

$$\mathbf{y}^{k+1} = \mathbf{E}^{-1}\Big(\mathbf{C} - \mathbf{B}\mathbf{v}^{k+1}\Big) \tag{2.79}$$

$$\mathbf{p}^{k+1} = \mathbf{E}^{-T}\Big(\mathbf{K}(\mathbf{y}^{k+1} - \bar{\bar{\xi}}_{\hat{n}})\Big). \tag{2.80}$$

Où $\mathbf{E}^{-1}$ et $\mathbf{E}^{-T}$ sont des matrices triangulaires inférieures et supérieures respectivement. Leurs résolutions se fait par la méthode de descente et de remonter respectivement. Ce qui donne le caractère séquentiel de la résolution.

Puisque le coût d'inverser ces matrices (i.e. $\mathbf{E}$ et $\mathbf{E}^{\mathbf{T}}$) est très élevé, nous allons proposer quelques opérations élémentaires sur le système linéaire $\mathbf{AX} = \mathbf{b}$ (issus de la discrétisation par éléments finis du problème de contrôle optimal sur l'intervalle de temps I) pour arriver à une structure diagonale par blocs de la matrice produite. Cette nouvelle matrice (notée $\hat{\mathbf{A}}$) possède des blocs diagonaux (notés $\mathbf{A}_{ii}$ voir (2.70)) qui sont des restrictions de la matrice $\mathbf{A}$ sur les sous-intervalles de temps I_n.

Nous décrivons dans ce qui suit comment ces opérations élémentaires de permutation appliquées sur le système $\mathbf{AX} = \mathbf{b}$ vont nous permettre de diviser la résolution, d'un problème de contrôle optimal global défini sur I, aux plusieurs résolution simultanées des sous-problèmes de contrôle optimal que nous définissions sur les I_n.

Remarque 2.5.4. *La résolution en parallèle des sous-systèmes* (2.70) *de contrôle optimal sur les sous-intervalles I_n se fait de la même manière que les trois résolutions* (2.78)–(2.80)*. Autrement dit, nous procédons dans l'ordre par :*

$$\mathbf{v}_n^{k+1} = \mathbf{v}_n^k - \rho_n\Big(\mathbf{G}_n\mathbf{v}_n^k + \mathbf{B}_n^T\mathbf{p}_n^k\Big) \tag{2.81}$$

$$\mathbf{y}_n^{k+1} = \mathbf{E}_n^{-1}\Big(\mathbf{C}_n - \mathbf{B}_n\mathbf{v}_n^{k+1}\Big) \tag{2.82}$$

$$\mathbf{p}_n^{k+1} = \mathbf{E}_n^{-T}\Big(\mathbf{K}_n(\mathbf{y}_n^{k+1} - \bar{\xi}_{n+1})\Big). \tag{2.83}$$

Avec ρ_n est (pour faire simple) le pas fixe de la descente locale.

2.5.6 Utilisation de la structure de la méthode de Jacobi pour paralléliser le problème de contrôle optimal

Nous commençons tout d'abord par rappeler la méthode de Jacobi par blocs. Nous décrivons ensuite l'étape de notre méthode là où nous nous servons de la structure du schéma de Jacobi afin de donner une formulation parallélisable de notre nouveau système. Nous expliquons également pourquoi notre méthode est plus robuste que celle de Jacobi appliquée à ce problème de contrôle optimal.

Méthode de Jacobi par blocs

Pour résoudre un système linéaire de type $\mathbf{AX} = \mathbf{b}$, la méthode de Jacobi décompose la matrice $\mathbf{A}$ comme la somme d'une matrice diagonale $\mathbf{D}$ et de deux matrices $\mathbf{U}$ et $\mathbf{L}$ respectivement triangulaire supérieure et triangulaire inférieure.

$$\mathbf{A} = \mathbf{L} + \mathbf{D} + \mathbf{U}.$$

Le système linéaire $\mathbf{AX} = \mathbf{b}$ est donc équivalent à $\mathbf{DX} = \mathbf{b} - (\mathbf{L} + \mathbf{U})\mathbf{X}$. La méthode de Jacobi propose alors le schéma itératif suivant :

$$\mathbf{X}^k := \mathbf{D}^{-1}\big(\mathbf{b} - (\mathbf{L} + \mathbf{U})\mathbf{X}^{k-1}\big). \tag{2.84}$$

Ce type de schéma itératif est parfaitement adaptable aux machines à haute performance avec une architecture parallèle. En effet, le caractère diagonal de la matrice $\mathbf{D}$ permet de réaliser l'inversion en parallèle, si bien sûr les blocs diagonaux sont inversibles. Le produit matrice vecteur $(\mathbf{L} + \mathbf{U})\mathbf{X}$ est parfois lourd à calculer : cela dépend du caractère creux ou non des matrices.

Application au problème de contrôle optimal parallèle

Dans cette sous section, nous considérons le système linéaire issus du problème de contrôle optimal après quelques opérations élémentaires de permutation. Soit donc le système linéaire $\hat{\mathbf{A}}\hat{\mathbf{X}} = \mathbf{b}$ suivant obtenu après

- une décomposition du système $\mathbf{AX} = \mathbf{b}$ (voir (2.65)) en $\hat{n}$ sous-systèmes (e.g. (2.76) où $\hat{n} = 2$).
- une opération (2.73) d'ajouter les matrices $\mathbf{K}_{ii} = 1, .., \hat{n}$ à la fois dans dans la matrice et dans le second membre, ce qui ne change pas le résultat puisque qu'il s'agissait d'ajouter et retrancher la même quantité.

Le système $\hat{\mathbf{A}}\hat{\mathbf{X}} = \mathbf{b}$ est donc :

$$\left[\begin{array}{ccc||c||ccc}
\mathbf{E}_{11} & \mathbf{B}_{11} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
\mathbf{0} & \mathbf{G}_{11} & \mathbf{B}_{11}^T & \dots & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
\boxed{\mathbf{K}_{11} - \mathbf{K}_{11}} & \mathbf{0} & \mathbf{E}_{11}^T & ..\mathbf{E}_{21}^T \dots & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
\hline\hline
\mathbf{E}_{21} & \mathbf{0} & \mathbf{0} & \ddots & \ddots & \dots & \vdots \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} & \dots & \mathbf{E}_{\hat{n}1}^T \\
\hline\hline
\mathbf{0} & \mathbf{0} & \mathbf{0} & \dots \mathbf{E}_{\hat{n}1} .. & \mathbf{E}_{\hat{n}\hat{n}} & \mathbf{B}_{\hat{n}\hat{n}} & \mathbf{0} \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} & \mathbf{G}_{\hat{n}\hat{n}} & \mathbf{B}_{\hat{n}\hat{n}}^T \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \ddots & \mathbf{K}_{\hat{n}\hat{n}} & \mathbf{0} & \mathbf{E}_{\hat{n}\hat{n}}^T
\end{array}\right]
\left[\begin{array}{c} \mathbf{y}_1 \\ \mathbf{v}_1 \\ \mathbf{p}_1 \\ \vdots \\ \vdots \\ \mathbf{y}_{\hat{n}} \\ \mathbf{v}_{\hat{n}} \\ \mathbf{p}_{\hat{n}} \end{array}\right]
= \left[\begin{array}{c} \mathbf{C} \\ \mathbf{0} \\ \mathbf{0} \\ \vdots \\ \vdots \\ \mathbf{0} \\ \vdots \\ \mathbf{K}_{\hat{n}\hat{n}}\bar{\xi}_{\hat{n}} \end{array}\right]. \tag{2.85}$$

Si nous appliquons la méthode de Jacobi sur ce système, nous obtenons donc le schéma itératif $\mathbf{D}\hat{\mathbf{X}}^k = \mathbf{b} - \big(\hat{\mathbf{L}} + \hat{\mathbf{U}}\big)\hat{\mathbf{X}}^{k-1}$, où $k \geq 1$, ou sous sa forme matricielle :

$$\left[\begin{array}{ccccc}
\mathbf{A}_{11} & & & & \\
& \ddots & & & \\
& & \mathbf{A}_{nn} & & \\
& & & \ddots & \\
& & & & \mathbf{A}_{\hat{n}\hat{n}}
\end{array}\right]
\left[\begin{array}{c} \hat{X}_1 \\ \vdots \\ \hat{X}_n \\ \vdots \\ \hat{X}_{\hat{n}} \end{array}\right]^k
= \left[\begin{array}{c} \mathbf{C}_1 \\ \vdots \\ \mathbf{0} \\ \vdots \\ \mathbf{K}_{\hat{n}\hat{n}}\bar{\xi}_{\hat{n}} \end{array}\right]
- \left[\begin{array}{ccccc}
\mathbf{0} & & & & \\
& \mathbf{0} & & \hat{\mathbf{U}} & \\
& & \mathbf{0} & & \\
& \hat{\mathbf{L}} & & \mathbf{0} & \\
& & & & \mathbf{0}
\end{array}\right]
\left[\begin{array}{c} \hat{X}_1 \\ \vdots \\ \hat{X}_n \\ \vdots \\ \hat{X}_{\hat{n}} \end{array}\right]^{k-1}, \tag{2.86}$$

où la matrice par blocs $\mathbf{D}$ correspond à Blockdiag$(\mathbf{A}_{ii})$, $i = 1, .., \hat{n}$ (voir (2.70)), $\hat{\mathbf{L}}$ et $\hat{\mathbf{U}}$ étant respectivement les matrices par blocs triangulaires inférieure et supérieure.

Ce schéma peut être implémenté de différentes façons parallèles, et ceci se base sur le calcul du produit matrice vecteur $(\hat{\mathbf{L}} + \hat{\mathbf{U}})\hat{\mathbf{X}}^{k-1}$. En outre, le fait d'ajouter et retrancher la matrice par blocs $\mathbf{K}_{ii}$ (mise en bleu dans la matrice (2.85)) définit les sous-problèmes de contrôle optimal qui se résolvent simultanément.

Bien que le schéma (2.86) de la méthode de Jacobi est tout à fait parallélisable, il est néanmoins pas robuste en terme de convergence. Cela vient du fait que la méthode de Jacobi inverse les blocs diagonales $\mathbf{A}_{ii}$ (définit à (2.70)) de la matrice $\mathbf{D}$ à chaque itération k. Cette inversion est premièrement assez coûteuse (même s'il s'agit des sous-système de tailles petites), deuxièmement cette méthode nécessite $\hat{n}$ itérations afin de pouvoir propager l'information du premier sous-intervalle de temps jusqu'au dernier. En outre, comme il s'agit d'un problème de contrôle itératif (i.e. le contrôle change d'une itération à l'autre) la propagation de l'information sera donc perturbée d'avantage, ce qui ralentisse la convergence de la méthode de Jacobi.

Il est claire qu'au niveau du schéma itératif (2.86) le second membre varie d'une itération à une autre. Notons ce second membre $\mathbf{R}(\hat{\mathbf{X}}^k) := \mathbf{b} - (\hat{\mathbf{L}} + \hat{\mathbf{U}})\hat{\mathbf{X}}^k$.

Remarque 2.5.5. *Nous signalons que le produit matrice vecteur :*

$$(\hat{\mathbf{L}} + \hat{\mathbf{U}})\hat{\mathbf{X}}^k$$

consiste à construire les cibles et les conditions intermédiaires pour les sous-problèmes locaux de contrôle optimal $\mathbf{A}_{ii}\hat{X}_n = b_n$ qui se résolvent simultanément.

En l'état de ce second membre $\mathbf{R}(\hat{\mathbf{X}}^k)$ et en tenant compte de la remarque 2.5.5 nous signalons que la convergence du schéma (2.86) est dépendante de l'initialisation des trajectoires des états direct et adjoint. En plus, si les sauts (calculés à l'interface entre chaque deux sous-intervalles voisins) des trajectoires respectives des états direct et adjoint sont considérables, la tâche de l'algorithme consistant à minimiser ces sauts devient de plus en plus lourde.

Notre méthode SITPOC propose donc d'améliorer le second membre $\mathbf{R}(\hat{\mathbf{X}}^k)$ à chaque itération k par une propagation le long de l'intervalle I ce qui permet une propagation net de l'information au cours d'une seule itération. Notons cette amélioration par $\tilde{\mathbf{R}}(\hat{\mathbf{X}}^k)$. En outre, la méthode SITPOC propose une approximation $\tilde{\mathbf{D}}^{-1} \approx \mathbf{D}^{-1}$. Chaque résolution bloc du problème en $\mathbf{D}$ correspondant en fait a un problème de contrôle local, on résout chacun de ces systèmes de façon indépendante, donc en parallèle de façon approchées par l'algorithme de gradient (2.81)–(2.83).

Dans une version plus améliorée de la méthode SITPOC, nous faisons appel dans la méthode PITPOC à l'algorithme pararéel en temps afin de prédire d'une manière légère, efficace et robuste les bonnes trajectoires des états direct et adjoint. L'algorithme pararéel en temps opère donc avec un nouveau second membre $\widetilde{\mathbf{R}}(\hat{\mathbf{X}}^k) \approx \tilde{\mathbf{R}}(\hat{\mathbf{X}}^k)$.

Nous décrivons brièvement une comparaisons entre la méthode de Jacobi et nos deux algorithmes :

la méthode Jacobi (2.86)

$$\hat{\mathbf{X}}^{k+1} = \mathbf{D}^{-1}\mathbf{R}(\hat{\mathbf{X}}^k).$$

La méthode SITPOC

$$\hat{\mathbf{X}}^{k+1} = \tilde{\mathbf{D}}^{-1}\tilde{\mathbf{R}}(\hat{\mathbf{X}}^k), \text{ avec } \tilde{\mathbf{D}}^{-1} \approx \mathbf{D}^{-1} \text{ et } \tilde{\mathbf{R}}(\hat{\mathbf{X}}^k) \text{ mieux que } \mathbf{R}(\hat{\mathbf{X}}^k).$$

La méthode PITPOC

$$\hat{\mathbf{X}}^{k+1} = \tilde{\mathbf{D}}^{-1}\widetilde{\mathbf{R}}(\hat{\mathbf{X}}^k), \text{ avec } \tilde{\mathbf{D}}^{-1} \approx \mathbf{D}^{-1} \text{ et } \widetilde{\mathbf{R}}(\hat{\mathbf{X}}^k) \approx \tilde{\mathbf{R}}(\hat{\mathbf{X}}^k).$$

Nous explicitons dans la suite les étapes de nos algorithmes.

2.5.7 Interprétation matricielle de l'algorithme SITPOC

Soit la matrice Π de permutation convertissant un vecteur inconnu $\mathbf{X}$ en $\hat{\mathbf{X}}$, où $\mathbf{X} = [\mathbf{y}^T, \mathbf{v}^T, \mathbf{p}^T]^T$ et $\hat{\mathbf{X}} = [\hat{X}_1^T, \ldots, \hat{X}_n^T, \ldots, \hat{X}_{\hat{n}}^T]^T$ avec $\hat{X}_n = [\mathbf{y}_n^T, \mathbf{v}_n^T, \mathbf{p}_n^T]^T$. On a

$$\Pi\mathbf{X} := \hat{\mathbf{X}} \quad \text{et} \quad \Pi^{-1}\hat{\mathbf{X}} := \mathbf{X}.$$

Après avoir appliqué la permutation sur le système linéaire $\mathbf{AX} = \mathbf{b}$ pour arriver à un système équivalent $\hat{\mathbf{A}}\hat{\mathbf{X}} = \mathbf{b}$,. Supposons connaître $\mathbf{X}^k$ nous considérons alors le système suivant :

$$\mathbf{D}\hat{\mathbf{X}} := \mathbf{b} - (\hat{\mathbf{L}} + \hat{\mathbf{U}})\Pi\mathbf{X}^k. \tag{2.87}$$

Finalement, dans la partie séquentielle de l'algorithme SITPOC nous **concaténons** les contrôles locaux pour obtenir le contrôle $\check{\mathbf{v}}$ que nous utilisons pour la mise à jour du contrôle séquentiel $\mathbf{v}$:

$$\mathbf{v}^{k+1} = \mathbf{v}^k + \theta^k(\check{\mathbf{v}}^{g_{\max}+1} - \mathbf{v}^k). \tag{2.88}$$

Par conséquent, nous sommes en mesure de réitérer avec les trois procédés (2.78)–(2.80).

D'une manière synthétique nous avons les étapes suivantes :

1. Résolution séquentielle de $\mathbf{MX} = \mathbf{b} - \mathbf{NX}^k$ suivant (2.78)–(2.80).
2. Permutation suivant : $\hat{\mathbf{X}}^1 = \Pi\mathbf{X}$,
3. Résolution parallèle de $\tilde{\mathbf{D}}\hat{\mathbf{X}}^2 = \mathbf{R}(\hat{\mathbf{X}}^1)$, avec $\big(\mathbf{R}(\hat{\mathbf{X}}^1) = \mathbf{b} - (\hat{\mathbf{L}} + \hat{\mathbf{U}})\hat{\mathbf{X}}^1$ et $\tilde{\mathbf{D}}^{-1} \approx \mathbf{D}^{-1}\big)$. $\tilde{\mathbf{D}}^{-1}$ est équivalent à quelques boucles sur la méthode de gradient (2.81)–(2.83) simultanément sur " n ".
4. Permutation inverse suivant : $\mathbf{X}^{k+1} = \Pi^{-1}\hat{\mathbf{X}}^2$ et récupération du contrôle.
5. Retour à l'étape 1.

Remarque 2.5.6 (**Concaténation**). *Uniquement les contrôles, solutions des sous-problèmes, sont concaténés. Cette concaténation a pour objectif d'approcher la solution du problème* (2.65). *Il ne s'agit donc pas de la concaténation de tout les sous-systèmes locaux tel que :*

$$\begin{bmatrix} \check{\mathbf{E}} & \check{\mathbf{B}} & \mathbf{0} \\ \mathbf{0} & \check{\mathbf{G}} & \check{\mathbf{B}}^T \\ \check{\mathbf{K}} & \mathbf{0} & \check{\mathbf{E}}^{\mathbf{T}} \end{bmatrix} \begin{bmatrix} \mathbf{y} \\ \mathbf{v} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \check{\mathbf{C}} \\ \mathbf{0} \\ \check{\mathbf{K}}\Xi \end{bmatrix} \tag{2.89}$$

où

$$\begin{aligned}
\check{\mathbf{K}} &:= \mathit{Blockdiag}(\mathbf{K}_n) \in \mathbb{R}^{(\hat{ls})\times(\hat{ls})}, \\
\check{\mathbf{E}} &:= \mathit{Blockdiag}(\mathbf{E}_n) \in \mathbb{R}^{(\hat{ls})\times(\hat{ls})}, \\
\check{\mathbf{B}} &:= \mathit{Blockdiag}(\mathbf{B}_n) \in \mathbb{R}^{(\hat{ls})\times(\hat{lq})}, \\
\check{\mathbf{G}} &:= \mathit{Blockdiag}(\mathbf{G}_n) \in \mathbb{R}^{(\hat{lq})\times(\hat{lq})}, \\
\check{\mathbf{C}} &:= [\mathbf{C}_1, \ldots, \mathbf{C}_{\hat{n}}]^T \in \mathbb{R}^{\hat{ls}}, \\
\Xi &:= [\bar{\xi}_1, \ldots, \bar{\xi}_{\hat{n}}]^T \in \mathbb{R}^{\hat{ls}}.
\end{aligned}$$

Cependant, à la ***convergence*** *de l'algorithme le système* (2.89) *est équivalent (****à une permutation*** Π ***près****) au système global des points critiques* (2.65).

2.5.8 Interprétation matricielle de l'algorithme PITPOC

Dans cette sous-section nous présentons les formulations matricielles de la première étape de l'algorithme PITPOC qui diffère de l'étape 1 de SITPOC. Cette étape consiste à calculer les cibles et conditions initiales avec l'algorithme pararéel en temps. Ce qui revient en terme matriciel à calculer le second membre $\widetilde{\mathbf{R}}(\hat{\mathbf{X}}^k)$.
Nous rappelons que dans la partie séquentielle de l'algorithme PITPOC nous optimisons le coût $\Phi_k(\theta)$ défini (2.45) à chaque itération par :

$$\Phi_k(\theta) := \frac{1}{2}\|\lambda_{\hat{n}}^{k+1}(\theta) - y^{cible}\|_2^2 + \frac{\alpha}{2}\int_I \|v^{k+1}(\theta)\|_2^2 dt,$$

où $\lambda_{\hat{n}}^{k+1}$ est entendu au sens de fonction.

Par abus de notations, nous identifions les représentations nodales des conditions initiales des états direct et adjoint avec leurs formulations continues. Dans ce qui suit, nous considérons un indice $\ell_n := \dfrac{T_n}{\tau_G}$ dont l'indice n parcourt $0, 1, \ldots, \hat{n}$. On a donc

$$\ell_n + \hat{\mathbf{m}} = \ell_{n+1} \text{ avec } \hat{\mathbf{m}}\hat{n} = \hat{\mathbf{l}}$$

Le coût discret $\bar{\Phi}_k$ en temps et en espace correspondant à Φ_k est défini par :

$$\bar{\Phi}_{k+1}(\theta) := \frac{1}{2}(\bar{\lambda}^{k+1}(\theta) - \bar{\xi}_{\hat{n}})^T \tilde{\mathbf{K}} (\bar{\lambda}^{k+1}(\theta) - \bar{\xi}_{\hat{n}}) + \frac{1}{2}\tilde{\mathbf{v}}^{k+1}(\theta)^T \tilde{\mathbf{G}} \tilde{\mathbf{v}}^{k+1}(\theta). \tag{2.90}$$

où $\tilde{\mathbf{K}} := \text{Blockdiag}(\mathbf{0}, \ldots, \mathbf{0}, F) \in \mathbb{R}^{(\hat{\mathbf{l}}\hat{s})\times(\hat{\mathbf{l}}\hat{s})}$, $\tilde{\mathbf{G}} \in \mathbb{R}^{(\hat{\mathbf{l}}\hat{q})\times(\hat{\mathbf{l}}\hat{q})}$ de même type que la matrice par blocs $\mathbf{G}$, le vecteur colonne $\bar{\lambda}^{k+1}(\theta) = [y_{j_0}, ..., y_{j_{\hat{n}}-1}, \lambda_{\hat{n}}^{k+1}(\theta)]^T \in \mathbb{R}^{\hat{\mathbf{l}}\hat{s}}$ où $\lambda_{\hat{n}}^{k+1}(\theta) \in \mathbb{R}^{\hat{s}}$ est l'état contrôlé, le vecteur colonne représentant la cible $\bar{\bar{\xi}}_{\hat{n}} = [0, ..., 0, \xi_{\hat{n}}]^T \in \mathbb{R}^{\hat{\mathbf{l}}\hat{s}}$ où le vecteur $\xi_{\hat{n}} \in \mathbb{R}^{\hat{s}}$, le contrôle $\tilde{\mathbf{v}}^{k+1}$ est un contrôle grossier, sa mise à jour est analogue à celle de $\mathbf{v}^{k+1}$ dans (2.88).
Le vecteur $\lambda_{\hat{n}}^k(\theta)$ correspond au terme final de la suite de vecteurs $(\lambda_n^k)_{n\geq 1}^{k\geq 0}$ calculée par :

$$\lambda_n^k(\theta) := \lambda_n^{k-1} + \theta\big(\tilde{\lambda}_n^{k-1} - \lambda_n^k\big), \tag{2.91}$$

où $\tilde{\lambda}_n^k$ est un état construit suivant le schéma pararéel (2.48) dont les solveurs sont utilisés avec les contrôle $\mathbf{v}(\theta = 1)$ (voir (2.88)) ceci revient à calculer la suite $(\tilde{\lambda}_n^k)_{n\geq 0}^{k\geq 0}$ selon le schéma (2.42).

$$\tilde{\lambda}_n^k := \tilde{y}_{\ell_n}^k + y_{j_n}^{k-1} - \tilde{y}_{\ell_n}^{k-1}, \tag{2.92}$$

où $\tilde{y}_{\ell_n}^k$ est le dernier élément de la suite de solutions du vecteur $\tilde{\mathbf{y}}_n^k = [(\tilde{y}_{\ell_{n-1}+1}^k)^T, .., (\tilde{y}_{\ell_n}^k)^T]^T$; vecteur colonne de taille $\hat{\mathbf{m}}\hat{s}$, calculé de manière séquentielle avec le solveur grossier. Par contre, le vecteur $y_{j_n}^{k-1}$ est le dernier élément de la suite de solutions du vecteur $\mathbf{y}_n^{k-1} = [(y_{j_{n-1}+1}^{k-1})^T, .., (y_{j_n}^{k-1})^T]^T$ calculé avec le solveur fin de manière parallèle. Les deux vecteurs $\tilde{\mathbf{y}}_n^k$ et $\mathbf{y}_n^{k-1}$ vérifient sur chaque sous-intervalle de temps $I_{n=1,\ldots,\hat{n}}$ les équations matricielles suivantes :

$$\tilde{\mathbf{E}}_n \tilde{\mathbf{y}}_n^k + \tilde{\mathbf{B}}_n \mathbf{v}_n^{k-1,g_{\max}+1} = [(F\lambda_n^k)^T, 0^T, \ldots, 0^T]^T, \text{(calcul séquentiel)}. \tag{2.93}$$

$$\mathbf{E}_n \mathbf{y}_n^{k-1} + \mathbf{B}_n \mathbf{v}_n^{k-1,g_{\max}} = [(F\lambda_n^{k-1})^T, 0^T, \ldots, 0^T]^T,\ 1 \leq g \leq g_{\max}\text{(calcul parallèle)}. \tag{2.94}$$

où $\mathbf{v}_n$ est pris comme une version grossière ; un vecteur de taille $\hat{\mathbf{m}}\hat{q}$, $\tilde{\mathbf{E}}_n \in \mathbb{R}^{(\hat{\mathbf{m}}\hat{s})\times(\hat{\mathbf{m}}\hat{s})}$ et $\tilde{\mathbf{B}}_n \in \mathbb{R}^{(\hat{\mathbf{m}}\hat{s})\times(\hat{\mathbf{m}}\hat{q})}$ deux matrices par blocs définis par :

$$\tilde{\mathbf{E}}_n = \begin{bmatrix} F_{\tau_G} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ -F & F_{\tau_G} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \ddots & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & -F & F_{\tau_G} \end{bmatrix}, \tilde{\mathbf{B}}_n = -\tau_F \begin{bmatrix} \mathcal{B}_h & 0 & 0 & 0 \\ 0 & \mathcal{B}_h & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \mathcal{B}_h \end{bmatrix}$$

Le calcul du gradient nécessite un état adjoint vectoriel γ_n^k adéquat qui s'écrit :

$$\gamma_n^k := \tilde{p}_{\ell_n}^k + p_{j_n}^{k-1} - \tilde{p}_{\ell_n}^{k-1}, \tag{2.95}$$

où $\tilde{p}^k_{\ell_n}$ est le premier élément de la suite de solutions du vecteur $\tilde{\mathbf{p}}^k_{n+1} = [(\tilde{p}^k_{\ell_n})^T, .., (\tilde{p}^k_{\ell_{n+1}-1})^T]^T$; vecteur colonne de taille $\hat{\mathbf{m}}\hat{s}$. De même le vecteur $p^{k-1}_{j_n}$ est le premier élément de la suite de solutions du vecteur $\mathbf{p}^{k-1}_{n+1} = [(p^{k-1}_{j_n})^T, .., (p^{k-1}_{j_{n+1}-1})^T]^T$. Les deux vecteurs $\tilde{\mathbf{p}}^k_{n+1}$ et $\mathbf{p}^{k-1}_{n+1}$ vérifient sur chaque sous-intervalle de temps I_n les équations matricielles suivantes :

$$\tilde{\mathbf{E}}^T_n \tilde{\mathbf{p}}^k_{n+1} + \tilde{\mathbf{K}}_n \tilde{\mathbf{y}}^k_{n+1} = [0^T, \ldots, 0^T, (F\gamma^k_{n+1})^T]^T, \text{(calcul séquentiel).} \tag{2.96}$$

$$\mathbf{E}^T_n \mathbf{p}^{k-1}_{n+1} + \mathbf{K}_n \mathbf{y}^{k-1}_{n+1} = [0^T, \ldots, 0^T, (F\gamma^{k-1}_{n+1})^T]^T, \text{(calcul parallèle).} \tag{2.97}$$

avec $\tilde{\mathbf{E}}^T_n$ la matrice transposée de $\tilde{\mathbf{E}}_n$ et $\tilde{\mathbf{K}}_n := \text{Blockdiag}(\mathbf{0}, \ldots, \mathbf{0}, F) \in \mathbb{R}^{(\hat{\mathbf{m}}\hat{s})\times(\hat{\mathbf{m}}\hat{s})}$.
Les cibles intermédiaires $(\xi^k_n)_{0\leq n\leq \hat{n}}$ sont donc définies d'une manière similaire à (2.66) comme suivant :

$$\xi^k_n = \lambda^k_n - \gamma^k_n. \tag{2.98}$$

Récapitulons maintenant les étapes permettant de calculer les états direct et adjoint par l'algorithme PITPOC.Supposons qu'on possède (par un calcul déjà effectué) les états direct et adjoint parallèles vérifiant respectivement les équations (2.94) et (2.97).

- Calculer grossièrement successivement les solutions $(\tilde{\mathbf{y}}^k_n)_{n\geq 1}$ sur chaque sous-intervalle I_n par :
  - a - Calculer l'état grossier $\tilde{\mathbf{y}}^k_n$ suivant (2.93).
  - b - Actualiser la condition initiale $\tilde{\lambda}^k_n$ pararéelle pour l'intervalle I_{n+1} par (2.92).
- Résoudre le problème adjoint (une fois l'état $\tilde{\lambda}^k_{\hat{n}}$ calculé) suivant :
  - a' - Calculer l'état grossier $\tilde{\mathbf{p}}^k_{n+1}$ d'une façon rétrograde suivant (2.96).
  - b' - Actualiser la condition finale (du problème rétrograde) $\tilde{\gamma}^k_n$ pour l'intervalle I_n par (2.95).
- Calculer le pas optimal θ^k minimisant (2.90).
- Mettre à jour les conditions initiales $(\lambda^k_n(\theta^k))_{1\leq n\leq \hat{n}}$ suivant (2.91).
- Calculer les cibles intermédiaires suivant la formule (2.98).

Nous venons donc d'expliquer la différence entre les algorithmes SITPOC et PITPOC qui est la résolution 1. (voir page 44) dans la partie séquentielle de la résolution.

2.6 Tests et résultats numériques

Les tests numériques ont été effectués avec le logiciel libre FreeFem++-mpi version parallèle [54] sur un super-calculateur massivement parallèle GNOME [8], de l'Université Pierre et Marie Curie. Il s'agit d'un puissant réseau de nœuds formulant le **iDataPlex** ; il possède 80 nœuds, chaque nœud ayant 2 processeurs quadricœurs Xeon Nehalem (à 8 processeurs), soit donc au total 640 cpus qui sont reliés via un réseau **Infiniband**.
Sur la frontale de ce réseau géant, nous avons soumis nos algorithmes qui simulent un problème de contrôle optimal d'un système régi par l'équation de la chaleur avec terme source suivante :

$$\partial_t y - \nu\Delta y = \mathcal{B}v,$$

le long d'une durée de temps $T = 6,4$ avec un pas de discrétisation temporel $\tau_F = 10^{-2}$. En particulier, nous choisissons le rapport de discrétisation temporel égal à $r = \Delta T/\tau_F$

8. `www.gnome.dsi.upmc.fr`

i.e. le pas de temps grossier est pris égal à la taille du sous-intervalle I_n.

À la figure 2.2 nous présentons deux types de configuration de réseau assurant les communications entre les processeurs qui exécutent des tâches parallèles. Notre méthode est basée sur la configuration de réseau master-slaves que nous présentons à gauche de la figure 2.2. Cette configuration montre la connexion directe de chacun des processeurs au processeur maître (le centre du réseau). Par contre, dans le deuxième réseau de processeurs, on trouve que chaque processeur ne possède qu'une seule connexion le reliant avec son voisin. Ceci pourrait être le cas pour le schéma (2.86) là où on ne fait pas de calcul séquentiel.

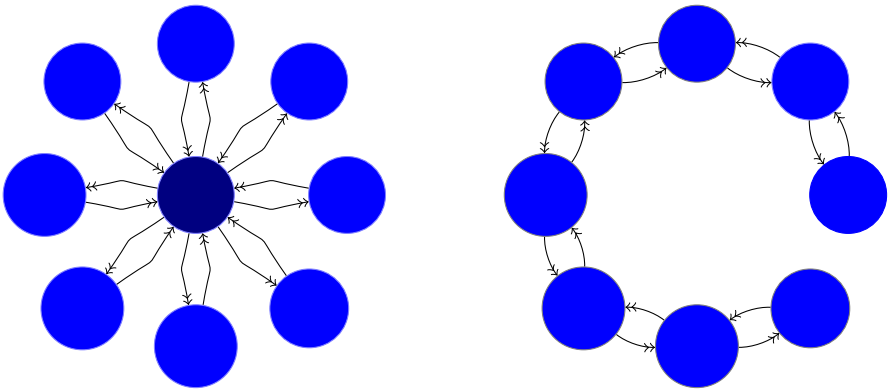


FIGURE 2.2 – Configuration de la communication : architecture master-slaves à gauche et architecture des clients à droite.

2.6.1 Bilan des simulations numériques

Nous présentons dans ce chapitre des résultats de simulation numérique permettant d'illustrer de manière quantitative l'intérêt de la parallélisation en temps, pour accélérer la procédure d'optimisation. La Figure 2.3 traite du comportement de la fonctionnelle $\mathcal{J}$ lors des itérations selon les différents algorithmes choisis et le nombre d'itérations utilisées. Nous comparons également ces algorithmes parallèlles par rapport au traitement en séquentiel classique. Nous remarquons, dans les deux algorithmes parallèlles, que la procédure de décomposition suivant l'approche des cibles intermédiaires fournit une accélération de convergence au sens ou un même niveau de décroissance de la fonctionnelle est atteint en un nombre moindre d'itérations. Nous remarquons également que l'algorithme SITPOC converge en un nombre plus faible d'itérations que l'algorithme PITPOC. En revanche le coût (évalué en nombre d'opérations ou en temps CPU) devient en faveur de l'algorithme PITPOC, comme on le voit dans les Figures 2.4–2.5 qui sont associées au même calcul que dans la Figure 2.3. Nous remplaçons ici le nombre d'itérations par le coût arithmétique (nombre d'opérations) effectué par une seule itération.
Ce type de courbe doit idéalement ressembler à des vraies courbes en temps d'exécution machine sur une machine virtuelle. Dans notre implémentation, nous allouons au processeur maître la résolution séquentielle et aussi l'une des tâches parallèles. C'est donc parmi tous les processeurs celui qui a le plus de tâches à effectuer.
Nous présentons à la Figure 2.4 la convergence de l'algorithme SITPOC en fonction du

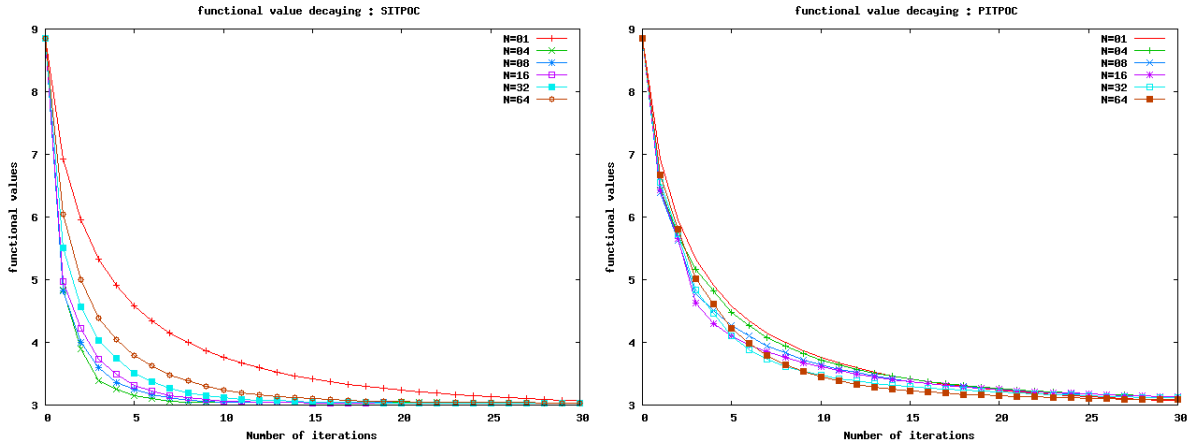


FIGURE 2.3 – Nombre d'itérations globale des algorithmes SITPOC (gauche) & PITPOC(droite).

coût arithmétique. Nous remarquons que ce coût ne change pratiquement pas selon le nombre de processeurs (égal au nombre de la décomposition) utilisés. Ceci s'explique par la domination de l'étape de propagation fine séquentielle sur l'intervalle I (qui est toujours fixe) par rapport aux tâches parallèles de calcul du contreôle dont le coût diminue environ en $\frac{1}{\hat{n}}$.

La Figure 2.5 représente le comportement de l'algorithme PITPOC. L'illustration du gain

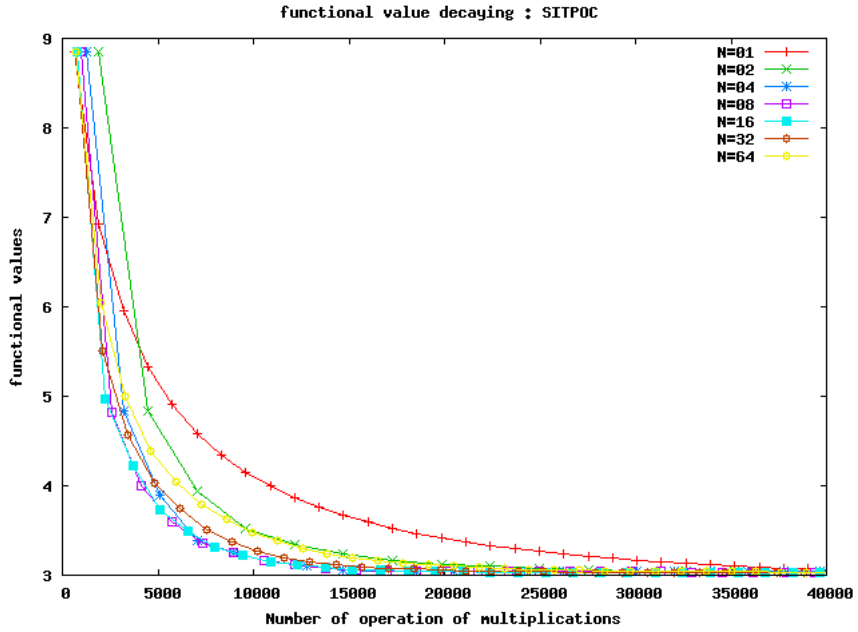


FIGURE 2.4 – Gain en nombre d'opérations de multiplication par itération de l'algorithme SITPOC.

en coût arithmétique est claire. La décroissance du coût est d'autant plus importante que le nombre $\hat{n}$ de sous-domaines est grand. Nous présentons enfin à la Figure 2.6 le comportement de la décroissance de la fonctionnelle de coût pour les deux algorithmes, en terme

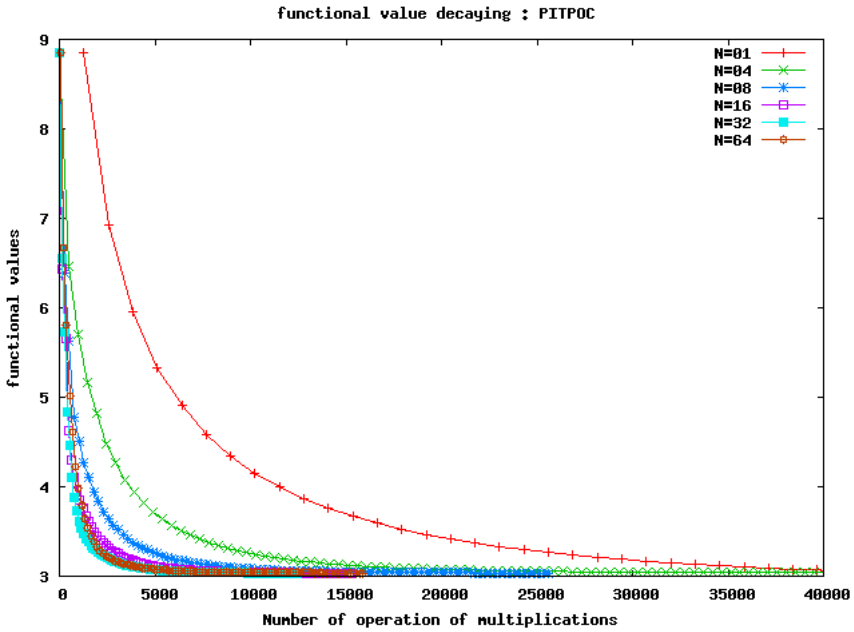


FIGURE 2.5 – Gain en nombre d'opérations de multiplication par itération de l'algorithme PITPOC.

de temps d'exécution véritable (wallclock) constaté sur une super-machine. Il apparaît que les deux algorithmes sont plus rapides que la résolution séquentielle grâce à leurs approches parallèles. Nous voyons également que PITPOC est bien une version améliorée de SITPOC. Pour ces résultats en temps "wallclock" de la convergence des deux algorithmes,

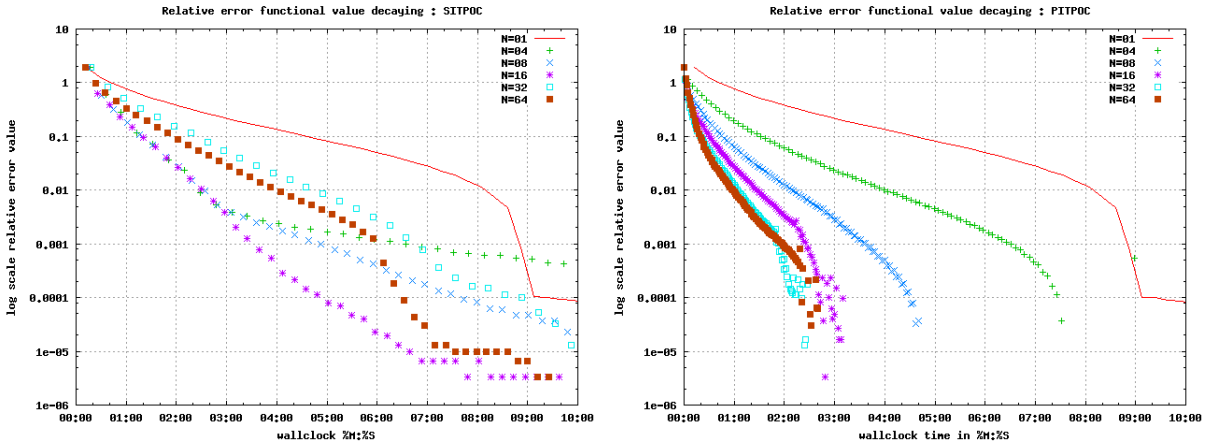


FIGURE 2.6 – Gain en temps réel de simulation machine : SITPOC (gauche) et PITPOC(droite).

si nous prenons le cas de $\hat{n} = 8$ décompositions qui sont attribuées aux 8 processeurs différents dans une architecture parallèle, nous voyons que SITPOC met environ 10 minutes pour atteindre la précision de 10^{-5} tandis que PITPOC met presque la moitié de ce temps.

2.6.2 Effet des itérations locales

Nous présentons ici le comportement de l'algorithme SITPOC vis-à-vis une variation du nombre des itérations locales $g_{\max}$. Nous rappelons que ces itérations locales correspondent aux minimisations des sous-fonctionnelles locales $\mathcal{J}_n$ par la méthode de gradient à pas optimal. Nous présentons à la figures 2.7 l'effet des itérations locales sur le comportement de l'algorithme SITPOC pour la minimisation de la fonctionnelle de coût non parallèle. Nous décomposons l'intervalle de temps en 4 sous-intervalles et nous présentons les tests à gauche de la figure 2.7, tandis que les tests relatifs à 16 subdivisions de l'intervalle de temps sont à droite de la même figure.

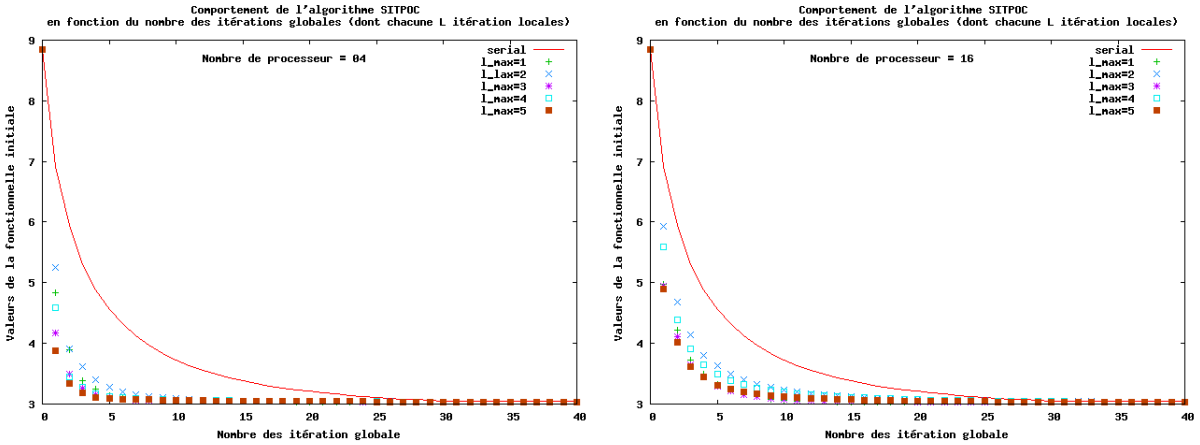


FIGURE 2.7 – L'algorithme SITPOC en 4 subdivisions à gauche et 16 subdivisions à droite : variation du nombre des itérations locales $l_{\max}$.

La bonne nouvelle est qu'il est préférable de faire 4 ou 5 itérations locales plutôt que 2. La mauvaise est que c'est encore mieux de n'en faire qu'une.

Pour inverser cette conclusion, il conviendrait, au moment de la concaténation des contrôles issus des itérations locales d'optimiser non pas sur un seul paramètre de relaxation, mais sur un multi-paramètre de $\mathbb{R}^{\hat{n}}$ pour optimiser la décroissance de la fonctionnelle de coût. Cette amélioration est en cours de mise en œuvre.

2.7 Conclusion du chapitre

Dans ce chapitre, nous avons présenté une méthode basée sur la définition des cibles et conditions initiales intermédiaires permettant de résoudre en parallèle un problème de contrôle parabolique. Ces méthodes consistent à décomposer l'intervalle de temps en subdivisions égales, et à définir des sous-problèmes indépendants de contrôle optimal, à l'aide d'attributions des cibles et des conditions initiales intermédiaires à chaque sous-intervalle de temps. Cette approche donne une autonomie aux sous-problèmes au niveau des sous-intervalles. En effet, elle leur fournit une indépendance totale entre eux, ce qui permet d'utiliser différentes méthodes de résolution en parallèle. Comme méthode d'optimisation, nous avons utilisé la méthode de gradient à pas optimal, mais nos algorithmes peuvent être utilisés avec d'autres procédures d'optimisation. La méthode des cibles intermédiaires a été élégamment couplée avec l'algorithme pararéel qui a participé à l'accélération du traitement sans nuire à sa stabilité numérique.

Chapitre 3

Parallélisation en temps de la résolution des équations de la cinétique neutronique

Ce travail est le fruit d'une collaboration avec Yvon Maday, Julien Salomon Anne-Marie Baudron et Jean-Jacques Lautard deux ingénieurs au CEA Saclay, dans le cadre du projet LRC Manon.

Sommaire

Abstract

This chapter studies the application of the parareal in time algorithm to time-parallelization of the Kinetic Neutronic Equation. We are especially interested in the simulation of the accidental phenomena within a nuclear reactor.
From neutronic point of view, these accidents are mathematically modeled by systems governed by linear partial differential equations. Our aim is to propose a time-parareal solver to achieve computing time acceleration, that helps to improve the nuclear accidents modelization.

Résumé

Ce chapitre est consacré à l'étude du comportement de l'algorithme pararéel pour la résolution numérique des équations de la cinétique neutronique. Nous nous intéressons spécialement à la simulation des phénomènes d'accidents dans les réacteurs nucléaires. Du point de vue neutronique, ces accidents sont modélisés mathématiquement par des systèmes d'équations aux dérivées partielles, et notre but est de proposer une méthode basée sur une parallélisation en temps afin d'accélérer la simulation.

FIGURE 3.1 – Assemblage du combustible.

3.1 Introduction

L'électricité assure de nos jours un confort dont on ne peut que difficilement se priver. En France, les centrales nucléaires forment une grande partie du parc de production de l'électrique. " Grâce à son parc de 58 réacteurs, la France atteint un taux d'indépendance énergétique proche de 50%, lui garantissant une grande stabilité d'approvisionnement."[9]. Les études pour améliorer la production de l'électricité constituent des défis aussi bien scientifiques qu'industriels. Les recherches mathématiques portent tout aussi bien sur la modélisation que sur la résolution numérique des modèles proposés. Dans ces conditions, la simulation numérique des phénomènes physiques est devenue un facteur indispensable dans la prise de décision, d'autant plus que les expériences physiques sont coûteuses ou mêmes dangereuses. Notre étude sur la cinétique neutronique s'intéresse à la simulation d'accidents dans un réacteur nucléaire. Ces accidents proviennent de l'augmentation de l'énergie au cours d'une réaction en chaîne de fission atomique, qui peut entraîner une fission du combustible et donc de grands dégâts. Afin d'approcher correctement la physique réelle, on est amené à considérer un nombre très important de paramètres dans les équations.
Faute de disponibilité de mémoire vive dans les ordinateurs séquentiels de nos jours, il est pertinent de proposer des méthodes parallèles qui résolvent ces systèmes, souvent de grandes tailles, avec des ordinateurs massivement parallèles.
Nous commençons par quelques remarques sur la physique des réacteurs. Un réacteur nucléaire produit de l'énergie graçe à la fission atomique, en particulier celles mettant en jeu des noyaux d'uranium. Au cours d'une fission, les noyaux se cassent suite à un choc avec un neutron et libèrent de l'énergie sous forme de chaleur, qui échauffe un liquide ou un gaz caloporteur qui produit ensuite de la vapeur. De la même manière que d'autres types de centrales électriques, cette vapeur active alors une turbine, qui est reliée à un alternateur et produit l'électricité. Les neutrons libres entrent alors en collision avec les noyaux du combustible provoquant ainsi l'éjection de nouveaux neutrons, qui à leur tour peuvent participer, par des chocs, à d'autres fissions. Cette réaction exothermique est appelée *réaction en chaîne de fission* et constitue le principal moteur de la production d'énergie.
Il est indispensable de contrôler la réaction en chaîne de fission. En effet, une réaction de

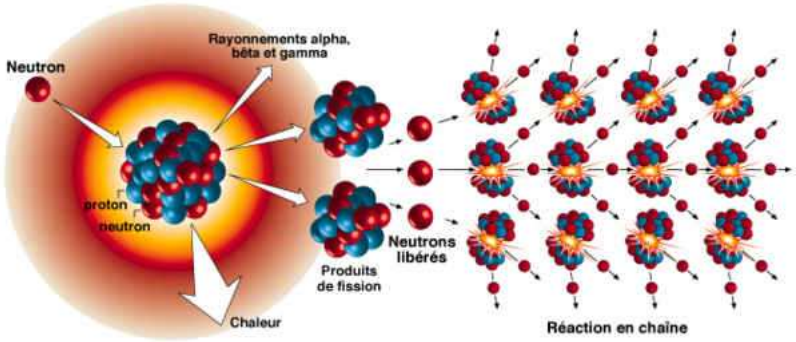


FIGURE 3.2 – Schéma d'une réaction en chaîne de fission.

9. SITE WEB DU COMMISSARIAT À L'ÉNERGIE ATOMIQUE ET AUX ÉNERGIES ALTERNATIVES `http://www.cea.fr`

fission non contrôlée peut conduire à une catastrophe conduisant à une fusion du cœur. De manière générale, le taux de production énergétique de la réaction en chaîne caractérise la réactivité du *cœur*[10] (voir figure 3.3). Si le coeur est trop fissile, la réaction en chaîne s'emballe et la production d'énergie du *cœur* est à son tour très importante. Afin de contrôler ce comportement, il est indispensable d'enfoncer dans le réacteur des *grappes* de contrôle constituées de matériaux servant à absorber les neutrons. Grâce à la surface de contact avec le combustible, la présence des grappes diminue ainsi le flux neutronique par absorption de neutrons.
Si on note v la vitesse des neutrons et ρ leur densité volumique (nombre de neutrons par unité de volume), la population des neutrons peut être représentée par le *flux neutronique* $\Phi = \rho v$, et d'une façon plus détaillée par :

- un repère spatial $\vec{r}$ à trois variables tel que : $\vec{r} \equiv (x, y, z)$,
- la vitesse v et leurs directions angulaires $\vec{\omega}$,
- le temps t précisant l'instant d'observation,
- l'ensemble des probabilités de choc des neutrons avec différents noyaux cibles. Les grandeurs physiques caractérisant ces probabilités sont appelées les *sections efficaces* microscopiques qui caractérisent une cible isolée, et les sections efficaces macroscopiques qui caractérisent un matériau contenant un ensemble de cibles.

La présentation de notre étude de la cinétique neutronique sera faite suivant quatre grands axes. Dans un premier temps, nous présentons le modèle neutronique basé sur l'équation de transport neutronique d'un groupe d'énergie (mono-cinétique). Une présentation multi-groupe est aussi introduite. Dans un second temps, nous proposons une unification des équations du modèle et discrétisons par la méthode des éléments finis. L'écriture unifiée permettra alors de présenter notre méthode de parallélisation. Ensuite, nous présentons l'algorithme pararéel [41] et donnons quelques propriétés. Enfin, nous présentons les résultats numériques obtenus sur quelques cas tests déjà étudiés.

3.2 Modélisation de la cinétique des neutrons

Dans cette section, nous présentons les équations mathématiques décrivant l'évolution des neutrons au sein du cœur nucléaire. Ces équations portent sur le flux neutronique Φ et font intervenir des grandeurs physiques, les sections efficaces, caractérisant les probabilités d'interaction des neutrons avec les noyaux environnants. Ces grandeurs sont données sous forme de coefficients $\sigma_t, \sigma_s, \sigma_f, \nu$ et χ constants par milieu physique :

σ_t : est la section efficace totale qui représente le taux de disparition des neutrons par choc soit par absorption (par des atomes) soit par transfert vers une autre vitesse et/ou direction.

σ_s : est la section de *scattering*[11], ou de transfert. Ce paramètre modélise une interaction de la matière conduisant à la modification de la direction et/ou de la vitesse des neutrons. La section conduisant à un changement entre un état “a” et un état “b” noté :

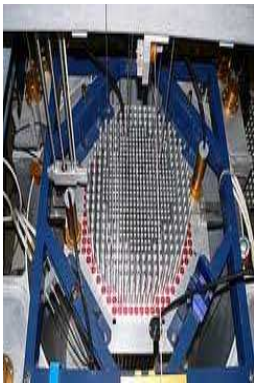
$$\text{“a”} \rightharpoonup \text{“b”}.$$

10. Le *cœur du réacteur* : c'est le noyau du réacteur là où s'effectue la réaction en chaîne, on utilise le terme de cœur .
11. Scatering : est une type de diffusion différente de la diffusion spatiale.

σ_f : est la section efficace de fission.

ν : représente la moyenne des neutrons émis par fission.

χ : est le *spectre* de neutrons émis par fission. Le spectre définit la répartition énergétique des neutrons après une fission.



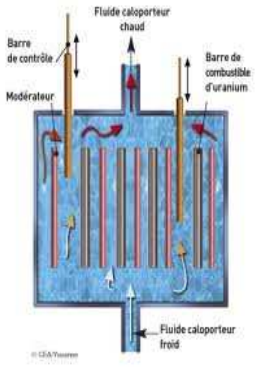


FIGURE 3.3 – Le *cœur* nucléaire à l'assemblage (à gauche), à l'état de fonctionner (au milieu) et sous forme de schéma (à droite).

3.2.1 Équation de Boltzmann sur le flux neutronique

L'équation de Boltzmann sur le flux neutronique Φ décrit le bilan cinétique de la population des neutrons. Il apparaît deux opérateurs dans cette équation aux dérivées partielles en temps et espace :

- l'opérateur de transport modélisant la diffusion de flux dans l'espace ainsi que le cheminement des neutrons en ligne droite sans interaction avec la matière.
- l'opérateur de collision exprimant les neutrons sortant de collision en fonction de ceux y entrant.

L'opérateur de transport peut s'écrire sous deux formes, une forme intégrale et une autre forme intégro-différentielle. Les deux écritures conduisent respectivement à deux types de résolutions numériques. Une résolution probabiliste de type Monte-Carlo, coûteuse en temps de calcul et une approche déterministe, cette dernière description correspond à l'équation :

$$\begin{aligned}\frac{1}{v}\frac{\partial}{\partial t}\varphi(\vec{r},v,\vec{\omega},t) &= -\nabla.[\vec{\omega}\varphi(\vec{r},v,\vec{\omega},t)] - \sigma_t(\vec{r},v,\vec{\omega},t)\varphi(\vec{r},v,\vec{\omega},t)\\ &+\int_0^\infty\int_{4\pi}\sigma_s(\vec{r},(v',\vec{\omega}')\rightharpoonup(v,\vec{\omega}),t)\varphi(\vec{r},v,\vec{\omega},t)d^2\omega' dv'\\ &+\frac{1}{4\pi}\chi(\vec{r},v,\vec{\omega},t)\int_0^\infty\int_{4\pi}\nu(\vec{r},v',\vec{\omega'},t)\sigma_f(\vec{r},v,\vec{\omega},t)\varphi(\vec{r},v,\vec{\omega},t)d^2\omega' dv'\\ &+S_e(\vec{r},v,\vec{\omega},t).\end{aligned} \tag{3.1}$$

C'est cette formulation que nous utilisons dans notre travail. De plus nous ne prenons pas en compte la source extérieure S_e. Dans (3.1), l'opérateur différentiel ∇ est le gradient qu'on note en coordonnées cartésiennes par :

$$\nabla. = \frac{\partial}{\partial\vec{r}} = (\frac{\partial}{\partial x},\frac{\partial}{\partial y},\frac{\partial}{\partial z}).$$

La variable $\vec{r}$ appartient au domaine d'étude Ω qui est un domaine borné inclus dans $\mathbb{R}^3$ modélisant le cœur nucléaire avec ses différents milieux (y compris le réflecteur constituant la couche extérieure).
Les sections efficaces sont des éléments de l'espace $L^\infty(\Omega \times [0,T])$ que nous notons désormais $\mathbb{F}$. Au bord du domaine les condition aux limites expriment qu'aucun neutron ne pénètre dans le domaine du coeur.

Momentanément nous ne précisons pas les conditions initiales utilisées (voir sous-section 3.3.2). Ces dernières sont conçues par un algorithme de puissance 3 produisant les flux propres, qui dépendent de k_{eff} (voir ci-après). Le flux propre ainsi défini, permet à son tour de calculer les conditions initiales des groupes de précurseurs.

3.2.2 Équation de la diffusion

On introduit le flux scalaire $\phi = \frac{1}{4\pi}\int \varphi dw$, et le courant $\vec{p} = \frac{1}{4\pi}\int \vec{w}\varphi dw$. Le modèle approché de la diffusion repose sur la loi de Fick $\vec{p} = -D\nabla\phi$ par intégration de (3.1) par rapport à la variable angulaire où D est le coefficient de diffusion (pour plus de détails voir [15]), dans notre étude ce coefficient vaut $\frac{1}{3\sigma_t}$. On en déduit une formulation multi-groupe avec $\hat{g}$ groupes d'énergies de flux neutroniques :

$$\begin{aligned}\frac{1}{v^{(g)}}\frac{\partial}{\partial t}\phi^{(g)}(\vec{r},t) = \quad & \mathrm{div}(D(\vec{r},t)^{(g)}\nabla\phi^{(g)})(\vec{r},t) - \sigma_t^{(g)}(x)\phi^{(g)}(\vec{r},t) + \textstyle\sum_{g=1}^{\hat{g}} \sigma_s^{(g\rightharpoonup g')}\phi^{(g')}(\vec{r},t) \\ & +\chi^{(g)} \textstyle\sum_{g'=1}^{\hat{g}} \nu^{(g')}\sigma_f^{(g')}\phi^{(g')}(\vec{r},t), \qquad\qquad \forall g \leq \hat{g},\end{aligned} \tag{3.2}$$

où

- $\phi^{(g)}$ est le flux du groupe d'énergie g parmi $\hat{g}$.
- $D^{(g)}$ est le coefficient de diffusion du groupe d'énergie g parmi $\hat{g}$.
- $\sigma_t^{(g)}$ est la section totale du groupe d'énergie g parmi $\hat{g}$.
- $\sigma_s^{(g\rightharpoonup g')}$ est le section de scattering du groupe d'énergie g vers le groupe d'énergie g'.
- $\chi^{(g)}$ est le spectre de fission des neutrons du groupe d'énergie g parmi $\hat{g}$.
- $\nu^{(g)}$ est le nombre moyen des neutrons du groupe d'énergie g parmi $\hat{g}$ émis par fission.
- $\sigma_f^{(g)}$ est la section de fission du groupe d'énergie g parmi $\hat{g}$.

Remarque 3.2.1. *Nous renvoyons le lecteur à [15] pour des compléments sur le passage entre l'écriture continue de l'équation de Boltzmann* (3.1) *vers l'écriture multi-groupe de type* (3.2)*. À cette équation on associe des conditions aux limites $\phi = 0$ sur $\partial\Omega$.*

Par ailleurs, deux types de conditions aux limites du domaine seront considérées. Sur le bord du domaine Γ_{ext} (cœur nucléaire : réflecteur inclus) on impose sur le flux des conditions de Dirichlet homogènes :

$$\Phi(\vec{r},t) = 0, \quad \forall \vec{r} \in \Gamma_{ext},$$

qui traduisent le cas limite d'un matériau très absorbant. Par ailleurs, dans le cas de l'étude d'un quart de cœur (dans ce cas le domaine possède de la symétrie modélisant une réflexion de flux nul à l'interface Γ_{sym}) le courant normal du flux est donc considéré nul au bord :

$$\langle \nabla.\Phi(\vec{r},t), \vec{n}(\vec{r})\rangle_{L^2(\mathbb{R}^3)} = 0, \quad \forall \vec{r} \in \Gamma_{sym},$$

ce qui constitue une condition de Neumann pour $\Phi(\vec{r},t)$ sur cette interface, où le vecteur $\vec{n}(\vec{r})$ lui est normal.

3.2.3 Groupes de précurseurs

Lors de la réaction de fission, le noyau de l'atome du combustible se divise en deux nucléides plus légers, souvent appelés produits de la réaction de fission, en émettant simultanément des neutrons (deux ou trois neutrons appelés neutrons *prompts*[12] ou instantanés). Étant excédentaires en neutrons, les noyaux produits sont instables. Pour devenir stables, ils se transforment grâce à une émission β^- (désintégration/décroissance radioactive) qui transforme les neutrons en protons.
Dans certains cas, ces fragments (les produits) de fission expulsent leurs neutrons hors de leurs noyaux. Ces neutrons émis sont dits *neutrons retardés* et les produits de fission émettant des neutrons sont dits *précurseurs*. Les neutrons expulsés sont dits retardés car ils ne sont pas le produit direct d'une fission, et leurs naissances sont décalées par rapport à celle de la réaction en chaîne d'un laps de temps entre une seconde et une minute, plus grand que la durée de vie d'un neutron (de l'ordre de la micro-seconde).
Introduisons enfin la notion de concentration de précurseurs qui est regroupée selon les différents processus émetteurs. Nous associons deux groupes de paramètres à chaque groupe d'énergie :

- La portion $\beta^{(k)}$ (voir [16, 34]) des neutrons retardés issus du groupe de précurseurs k parmi l'ensemble de ceux émis par la fission. Cette valeur est répartie selon le groupe d'énergie n dans lequel les neutrons retardés ont été émis.
- La constante de décroissance radioactive ou de désintégration $\mu^{(k)}$ du groupe de précurseurs k en seconde^{-1}.

3.2.4 Équation de bilan multi-groupe

Notons $c^{(k)}$ la concentration du précurseur de groupe k parmis $\hat{k}$ groupes. L'équation de bilan neutronique fait intervenir les groupes de précurseurs dans l'équation régissant le flux neutronique. Il s'agit d'un couplage au niveau de la source de fission, où les neutrons retardés participent au flux neutronique :

$$\begin{aligned}\frac{1}{v^{(g)}}\frac{\partial}{\partial t}\phi^{(g)}(\vec{r},t) = \quad & \operatorname{div}(d(\vec{r},t)\nabla\phi^{(g)}(\vec{r},t)) - \sigma_t^{(g)}\phi^{(g)}(\vec{r},t) \\ & + \textstyle\sum_{g'=1}^{\hat{g}} \sigma_s^{(g'\to g)}\phi^{(g')}(\vec{r},t) \\ & + \chi_p^{(g)}(\vec{r}) \textstyle\sum_{g'=1}^{\hat{g}}(1-\beta^{(g')})\nu^{(g)}(\vec{r})\sigma_f^{(g')}\phi^{(g')}(\vec{r},t) \\ & + \textstyle\sum_{k=1}^{\hat{k}} \chi_d^{(k,g)}(\vec{r})\mu^{(k)}c^{(k)}(\vec{r},t),\end{aligned}$$

où $\chi_p^{(g)}$ est le spectre de fission [58] des neutrons prompts et retardés du groupe d'énergie g tandis que $\chi_d^{(k,g)}$ est le spectre des neutrons prompts associés au groupe de précurseurs k.
L'opérateur différentiel est : $\operatorname{div}(.) := \partial_x(.) + \partial_y(.) + \partial_z(.)$.

12. Les neutrons sont dits prompts lorsque leur apparition est rapide comme produit direct de la réaction de fission, "prompt" étant le contraire de "retardé".

L'évolution des concentrations de groupes de précurseurs est régie par l'équation différentielle ordinaire :

$$\frac{\partial}{\partial t}c^{(k)}(\vec{r},t) = \mu^{(k)}c^{(k)}(\vec{r},t) + \sum_{g'=1}^{\hat{g}} \beta^{(k,g)}\nu^{(g')}(\vec{r},t)\sigma_f^{(g')}\phi^{(g')}(\vec{r},t). \tag{3.3}$$

3.2.5 Unification des équations de bilan multi-groupe

Soit $\phi := [\phi^{(1)}, \phi^{(2)}, ..., \phi^{(\hat{g})}]^T$ le vecteur représentant le flux de tous les groupes d'énergies, et $c := [c^{(1)}, c^{(2)}, ..., c^{(\hat{k})}]^T$ le vecteur représentant la concentration de tous les groupes de précurseurs.
L'équation régissant le flux ϕ est :

$$V^{-1}\frac{\partial}{\partial t}\phi(\vec{r},t) = \mathrm{div}(\bar{d}(\vec{r},t)\phi(\vec{r},t)) - [\Sigma](\vec{r},t)\phi(\vec{r},t) + Rc(\vec{r},t), \tag{3.4}$$

où le vecteur $V^{-1} \in \mathbb{R}^{\hat{g}\hat{g}}$ est défini par :

$$V^{-1} := \mathrm{Diag}(\frac{1}{v^{(1)}}, .., \frac{1}{v^{(g)}}, .., \frac{1}{v^{(\hat{g})}}),$$

et $\bar{d} \in \mathbb{F}^{\hat{g}\hat{g}}$ est défini par Blocdiag(d). Par "Diag($\bullet$)", nous désignons la matrice diagonale dont les coefficients diagonaux sont présentés sous formes d'arguments (i.e. ($\bullet$)). De même, "Blocdiag" désigne ici une matrice diagonale par bloc. Le paramètre des sections physiques $[\Sigma] \in \mathbb{F}^{\hat{g}\hat{g}}$ est :

$$[\Sigma](\vec{r},t) := \begin{cases} [\Sigma]_{i,i}(\vec{r},t) & := \sigma_t^{(i)}(\vec{r},t) - \sigma_s^{(i\to i)}(\vec{r},t) + \nu^{(i)}(1-\beta^{(i)})\chi_p^{(i)}\sigma_f^{(i)}(\vec{r},t), \\ [\Sigma]_{\substack{i,j \\ i\neq j}}(\vec{r},t) & := -\sigma_s^{(i\to j)}(\vec{r},t) + \chi_p^{(i)}(1-\beta^{(j)})\nu^{(j)}(\vec{r},t)\sigma_f^{(j)}(\vec{r},t), \end{cases} \tag{3.5}$$

où $R \in \mathbb{F}^{\hat{k}\hat{g}}$ est définie par $(R)_{i,j} := \chi_d^{(j,i)}(\vec{r},t)\mu^{(j)} \in \mathbb{F}$. L'équation régissant les concentrations des groupes de précurseurs est :

$$\frac{\partial}{\partial t}c(\vec{r},t) = [\mu]c(\vec{r},t) + [\Sigma^f]\phi(\vec{r},t), \tag{3.6}$$

où $[\mu]$ est une matrice diagonale appartenant à $(L^\infty(\Omega))^{\hat{k}\hat{k}}$ avec $[\mu]_{i,i} = \mu^k$. L'élément $[\Sigma^f] \in \mathbb{F}^{\hat{k}\hat{g}}$ est une matrice pleine telle que $[\Sigma^f]_{i,j} = \beta^{(i,j)}\nu^{(j)}(\vec{r},t)\sigma^{(j)} \in \mathbb{F}$.

3.3 Méthode de résolution

Notre approche pour la résolution consiste à regrouper toutes les équations de différents groupes d'énergie portant d'une part sur le flux neutronique et d'autre part sur les concentrations de précurseurs. Par commodité de présentation, nous donnons tout d'abord les équations régissant les différents groupes d'énergie (pour le flux et pour les concentrations de précurseurs) sous une forme unifiée dans le cadre continu. Dans un deuxième temps, nous procéderons à la discrétisation en temps et espace. Nous présenterons enfin

la structure matricielle utilisée. Cette procédure d'unification facilite la compréhension de l'application de l'algorithme pararéel sur le modèle neutronique car elle permet de regrouper toutes les solutions recherchées sous forme vectorielle. Ainsi un unique solveur sera donc suffisant pour la propagation.

3.3.1 Discrétisation spatiale par éléments finis

Les flux neutroniques modélisés par les équations (3.2)-(3.4) sont des éléments de l'espace de Hilbert $L^2([0,T];H_0^1(\Omega))$ où $H_0^1(\Omega)$ est un espace de Hilbert défini par :

$$H_0^1(\Omega)) := \{u \in L^2(\Omega),\ \text{tel que } \nabla u \in L^2(\Omega), u = 0 \text{ sur } \partial\Omega\}.$$

Afin de réaliser la discrétisation en espace, nous utilisons la méthode des éléments finis associée à une formulation faible des équations du flux neutronique et des concentrations des précurseurs. Pour cela, nous notons $\langle ., . \rangle$ le produit scalaire Hilbertien. L'espace des fonctions tests pour le flux est $H_0^1(\Omega)$ et $L^2(\Omega)$ pour les concentrations des précurseurs. Considérons une triangulation uniforme $\mathcal{T}_h$ de l'espace $\Omega \subset \mathbb{R}^3$, nous approchons les fonctions flux et concentration de précurseurs par des éléments finis conformes $\mathbb{P}_1$ constituant l'espace des polynômes de degrés 1 et à trois variables d'espace. Soit donc la famille de fonctions $\{\psi_i'\}_{i=1}^{\hat{s}_1}$ (respectivement $\{\psi_i''\}_{i=1}^{\hat{s}_2}$), une base standard de l'espace discret $\mathcal{Y}'_h \subset H_0^1(\Omega)$ (respectivement $\mathcal{Y}''_h \subset L^2(\Omega)$) dans lequel nous représentons vectoriellement le flux ϕ (respectivement les concentrations des précurseurs c) par $\mathbf{\Phi}$ (respectivement par $\mathbf{c}$).
Soit $\hat{s} := \hat{g}\hat{s}_1 + \hat{k}\hat{s}_2$ et les espaces $\mathcal{Y}'$ et $\mathcal{Y}''$ sont pris égaux et définis par

$$\mathcal{Y}' := \{u_h \in C^0(\Omega), u_{h|\mathbf{K}} \in \mathbb{P}_1, \forall \mathbf{K} \in \mathcal{T}_h\}.$$

Remarque 3.3.1. *La physique réelle du modèle que nous traitons admet une géométrie dynamique puisque les grappes de contrôle d'énergie sont en mouvement au fur et à mesure que la réaction se déroule. Comme les sections efficaces sont constantes dans chaque milieu, il est plus judicieux de lier leur dépendance en temps au mouvement des grappes. Pour cette raison, ces coefficients seront nécessairement des fonctions bornées appartenant à* $L^\infty(\Omega) := \{u : \Omega \to \mathbb{R}, \exists c > 0, |u(\vec{r})| < c, \forall \vec{r} \in \Omega\}$ *dont on considérera une représentation* $\mathbb{P}_0$.

Remarque 3.3.2. *Pour le modèle déterministe des équations de la cinétique, les sections efficaces discretisées en groupe sont supposées constantes sur chacune des mailles géométriques (appelées milieux). La notation des représentations nodales de ces sections efficaces est accompagné d'un chapeau (i.e. : "$\widehat{\ }$ ") pour indiquer une section définie sur tout le domaine indépendamment des groupes d'énergie (ou de milieu d'occupation).*

Nous procédons maintenant à une discrétisation par éléments finis pour les équations de la cinétique neutronique. Considérons $\dot{\mathbf{\Phi}}(t)$ la représentation vectorielle (nodale) de la dérivée en temps du vecteur flux neutronique, et $\dot{\mathbf{c}}(t)$ la représentation vectorielle de la dérivée en temps de la concentration des précurseurs. L'équation discrète multi-groupe unifiée régissant le flux neutronique est la suivante :

$$\mathcal{M}_h^v \dot{\mathbf{\Phi}}(t) = \mathcal{A}_h^d \mathbf{\Phi}(t) - \mathbf{\Sigma}_h \mathbf{\Phi}(t) + \mathcal{R}_h \mathbf{c}(t), \tag{3.7}$$

où $\mathcal{M}_h^v \in \mathbb{R}^{(\hat{s}_1\hat{g})\times(\hat{s}_1\hat{g})}$ est définie par $\mathcal{M}_h^v := \text{Blockdiag}(M_h^v)$, avec $M_h^v \in \mathbb{R}^{\hat{s}_1\hat{s}_1}$,
et $\mathcal{A}_h^d \in \mathbb{R}^{(\hat{s}_1\hat{g})\times(\hat{s}_1\hat{g})}$ est définie par $\mathcal{A}_h^d := \text{Blockdiag}(A_h^d)$, la matrice $\mathbf{\Sigma}_h \in \mathbb{R}^{(\hat{s}_1\hat{g})\times(\hat{s}_1\hat{g})}$ et la

matrice $\mathcal{R}_h \in \mathbb{R}^{(\hat{s}_1\hat{g})\times(\hat{s}_1\hat{k})}$ est définie telle que $\mathcal{R}_h := \text{Blockdiag}(R_h)$.
Notons que $(M_h^v)_{i,j} := VM_h$, tandis que $M_h := \langle\psi_i, \psi_j\rangle$, $(A_h^d)_{i,j} := \langle\bar{D}_i\nabla\psi_i, \nabla\psi_j\rangle$, où $\bar{D}_i$ est la représentation nodale du coefficient de diffusion D (constante par élément en $\mathbb{P}_0$) relative à l'élément en question, $(R_h)_{i,j} := \langle(\widehat{\chi_d\mu})_i\psi_i, \psi_j\rangle$. La matrice $\mathbf{\Sigma}_h$ des sections efficaces est définie par :

$$\mathbf{\Sigma}_h := \mathbf{\Sigma}_h^t - \mathbf{\Sigma}_h^s + (Id - \widehat{[\beta]})\mathbf{\Sigma}_h^f,$$

où,

$$\begin{aligned}
\mathbf{\Sigma}_h^t &:= \text{Blockdiag}(\Sigma_h^t) \in \mathbb{R}^{(\hat{s}_1\hat{g})\times(\hat{s}_1\hat{g})},\\
\mathbf{\Sigma}_h^s &:= \text{Blockdiag}(\Sigma_h^s) \in \mathbb{R}^{(\hat{s}_1\hat{g})\times(\hat{s}_1\hat{g})},\\
\mathbf{\Sigma}_h^f &:= \text{Blockdiag}(\Sigma_h^f) \in \mathbb{R}^{(\hat{s}_1\hat{g})\times(\hat{s}_1\hat{g})},\\
\widehat{[\beta]} &:= \text{Blockdiag}(\beta^{(i)} I_{\mathbb{R}^{\hat{s}_1\hat{s}_1}}) \in \mathbb{R}^{(\hat{s}_1\hat{g})\times(\hat{s}_1\hat{g})},
\end{aligned}$$

avec

$$\begin{aligned}
(\Sigma_h^t)_{i,j} &:= \langle(\widehat{\sigma}_t)_i\psi_i, \psi_j\rangle \in \mathbb{R}^{\hat{s}_1\hat{s}_1},\\
(\Sigma_h^s)_{i,j} &:= \langle(\widehat{\bar{\sigma}}_s)_i\psi_i, \psi_j\rangle \in \mathbb{R}^{\hat{s}_1\hat{s}_1},\\
(\Sigma_h^f)_{i,j} &:= \langle(\widehat{\chi_p\nu\sigma_f})_i\psi_i, \psi_j\rangle \in \mathbb{R}^{\hat{s}_1\hat{s}_1},
\end{aligned}$$

où les sections $\widehat{\sigma}_t$, $\widehat{\sigma}_s$ et $\widehat{\chi_p\nu\sigma_f}$ sont les représentations nodales en élément fini $\mathbb{P}_0$, et où $(\widehat{\sigma}_t)_i$, $(\widehat{\bar{\sigma}}_s)_i$ et $(\widehat{\chi_p\nu\sigma_f})_i$ font référence au tétraèdre relatif à l'indice i. L'équation discrète régissant la concentration totale $\mathbf{c}$ des précurseurs est :

$$\mathcal{M}_h\dot{\mathbf{c}}(t) = \mathcal{N}_h\mathbf{c}(t) + \widetilde{[\beta]}\mathbf{\Sigma}_h^f\mathbf{\Phi}(t), \tag{3.8}$$

où $\mathcal{M}_h \in \mathbb{R}^{(\hat{s}_2\hat{k})\times(\hat{s}_2\hat{k})}$, $\mathcal{M}_h := \text{Diag}(M_h)$, et $\mathcal{N}_h := \text{Blockdiag}(N_h^k) \in \mathbb{R}^{(\hat{s}_2\hat{k})\times(\hat{s}_2\hat{k})}$ avec $(N_h^k)_{i,j} := (\mu_i^{(k)}\psi_i, \psi_j)$, et $\widetilde{[\beta]} \in \mathbb{R}^{(\hat{s}_2\hat{k})\times(\hat{s}_2\hat{g})}$, est telle que $(\widetilde{[\beta]})_{k,g} := (\bar{\beta}^{(g)}) \in \mathbb{R}^{\hat{s}_2\hat{s}_2}$ où $(\bar{\beta}^{(g)})_{i,j} := \beta^{(g,k)} I_{\mathbb{R}^{\hat{s}_2\hat{s}_2}}$.

3.3.2 Construction des états initiaux

Dans cette section, nous présentons et expliquons les outils de construction des états initiaux relatifs aux flux et aux concentrations de précurseurs.

Équation stationnaire de la cinétique neutronique

L'étude des équations stationnaires de la neutronique est nécessaire pour définir l'état initial pour le calcul cinétique. Nous utilisons dans cette section la notion de *réactivité* qui porte sur les états du cœur nucléaire. En particulier, nous aurons recours à la notion d'*état stable* qui nécessite l'introduction d'un problème aux valeurs propres. Ce problème aux valeurs propres donne lieu à ce qu'on appelle la *réactivité efficace* caractérisée par le facteur k_{eff} qui désigne la plus grande valeur propre généralisée des deux matrices (de

production $\widehat{F} := \mathbf{\Sigma}_h^f$ (c'est la matrice qui caractérise la production des neutrons par la fission) et la matrice de disparition (absorption + fuites) $\widehat{A}$, voir (3.13)).

$$\widehat{A}\Phi = \frac{1}{k_{eff}}\widehat{F}\Phi.$$

Nous proposons enfin un algorithme permettant de calculer le coefficient k_{eff}.
Le réacteur nucléaire peut avoir trois états d'énergie (*sous-critique, critique et sur-critique*). Ces trois états sont liés à la densité des neutrons diffusant dans le cœur nucléaire. Elles sont principalement liées à la production neutronique par la réaction en chaîne et la disparition des neutrons par absorption. Le réacteur est dit *critique* si le bilan neutronique est nul. Ceci signifie qu'à chaque instant le nombre de neutrons produits est égal au nombre de neutrons disparus. Le bilan neutronique à l'intérieur d'un cœur à l'équilibre vérifie alors :

$$\text{Production}_{fission} + \text{Source}_{extérieure} = \text{Absorption} + \text{Fuites}. \tag{3.9}$$

Les neutrons injectés dans le système constituent la source d'amorçage de la réaction, l'absorption peut être importante, la fuite représente les neutrons quittant le système.
Le facteur multiplicatif dans un milieu sans fuite, noté k_∞ est défini par :

$$k_\infty := \frac{\text{nombre de neutrons à l'instant } t}{\text{nombre de neutrons à l 'instant } t\text{-1}}. \tag{3.10}$$

Dans un milieu borné, le facteur de multiplication effectif noté k_{eff} est le suivant :

$$k_{eff} = \frac{k_\infty}{1 + \text{Fuites}}. \tag{3.11}$$

Ce coefficient devrait être très proche de l'unité pour garantir un état d'équilibre. Ainsi, si $k_{eff} < 1$ l'état du cœur est dit sous-critique, si $k_{eff} > 1$ l'état est sur-critique et si $k_{eff} = 1$ l'état du cœur est critique.
Afin de déterminer un tel état, il est indispensable d'évaluer l'état stationnaire propre sans source extérieure (i.e. la seule source considérée est la fission).
L'équation de Boltzmann stationnaire admet des solutions uniquement dans le cas critique. Dans ce cadre particulier, nous considérons une source fission :

$$\chi^{(g)}\Sigma_{m=1}^{\hat{g}}\nu^{(g')}\sigma_f^{(g')}\phi^{(g')}(\vec{r},t),$$

et les termes de l'absorption (diffusion et scattering,..) :

$$-\text{div}(D(\vec{r},t)\nabla\phi^{(g)})(\vec{r},t) + \sigma_t^{(g)}(x)\phi^{(g)}(\vec{r},t) - \sum_{g'=1}^{\hat{g}}\sigma_s^{(g\rightarrow g')}\phi^{(g')}(\vec{r},t).$$

Les données de la modélisation ne sont pas forcément des données caractérisant un état critique et le cas échéant, nous forçons notre système à devenir critique en ajoutant le facteur k_{eff} dans l'équation de Boltzmann stationnaire. Cette équation devient alors :

$$\begin{aligned} &-\text{div}(d(\vec{r},t)\nabla\phi^{(g)})(\vec{r},t) + \sigma_t^{(g)}(x)\phi^{(g)}(\vec{r},t) - \textstyle\sum_{g'=1}^{\hat{g}}\sigma_s^{(g\rightarrow g')}\phi^{(g')}(\vec{r},t) \\ &= \frac{1}{k_{eff}}\chi^{(g)}\textstyle\sum_{g'=1}^{\hat{g}}\nu^{(g')}\sigma_f^{(g')}\phi^{(g')}(\vec{r},t). \end{aligned} \tag{3.12}$$

On s'est ainsi ramené à un problème aux valeurs propres consistant à chercher la valeur k_{eff} pour laquelle l'équation bilan (3.11) critique a lieu.

Algorithme de calcul d'un état critique du réacteur nucléaire

Pour calculer l'état critique d'un réacteur, c'est à dire résoudre l'équation aux valeurs propres (3.12), nous utilisons la méthode de la puissance qui est un algorithme itératif permettant de calculer la valeur propre de module maximal. Dans le cadre de la neutronique, la méthode de la puissance résout un problème dont le second membre est la source de fission.
Nous allons maintenant étudier précisément les matrices impliquées dans ce problème. Les matrices disparition (absorption+fuites) sont décrites par les matrices :

$$\widehat{\mathrm{A}} := \mathcal{A}_h^d - \mathbf{\Sigma}_h^t + \mathbf{\Sigma}_h^s, \tag{3.13}$$

et la matrice relative à la source de fission est :

$$\widehat{\mathrm{F}} := \mathbf{\Sigma}_h^f.$$

Grâce à ces matrices, nous pouvons calculer la valeur propre généralisée k_{eff} du couple $(\widehat{\mathrm{A}}, \widehat{\mathrm{F}})$, vérifiant :

$$\widehat{\mathrm{A}}\mathbf{\Phi} = \frac{1}{k_{eff}}\widehat{\mathrm{F}}\mathbf{\Phi},$$

où $\mathbf{\Phi}$ est le flux propre associé au k_{eff}. L'algorithme ci-dessous permet de calculer cette valeur propre de plus grand module, la norme $\|.\|_2$ étant associée au produit scalaire $\langle ., . \rangle_{\mathbb{R}^{\hat{g}\hat{s}_1}}$.

Algorithme 3 : Algorithme de la puissance, problème à source

Input : Initial flow vector $\mathbf{\Phi^0}$, tolerance $\tilde{\epsilon}$;
Initialize the source term $\mathcal{S}^0 = \widehat{\mathrm{F}}\mathbf{\Phi}^0; l \leftarrow 0$;
repeat

1. Solve iteratively : $\widehat{\mathrm{A}}\mathbf{\Phi}^{l+1} = \mathcal{S}^l$;
2. Normalize $\tilde{\mathbf{\Phi}}^{l+1} := \mathbf{\Phi}^{l+1}/\|\mathbf{\Phi}^{l+1}\|_2$;
3. Evaluate the relative error $Err := \|\tilde{\mathbf{\Phi}}^{l+1} - \tilde{\mathbf{\Phi}}^l\|_2/\|\tilde{\mathbf{\Phi}}^{l+1}\|_2$;
4. Update the source : $\mathcal{S}^{l+1} = \widehat{\mathrm{F}}\tilde{\mathbf{\Phi}}^{l+1}$;

$l \leftarrow l + 1$;
until $Err \leq \tilde{\epsilon}$;
Result : $\mathbf{\Phi} := \mathbf{\Phi}^l \quad \& \quad k_{eff} := \|\mathbf{\Phi}^l\|_2$;

Conditions initiales pour les concentrations des groupes de précurseurs

Dans le cas où nous ne disposons pas de flux initial, le problème devient un problème aux valeurs propres. Une fois que nous disposons de la condition initiale $\Phi^{(g,\star)}$ (i.e. flux propre du groupe (g)), nous pouvons déterminer la concentration des précurseurs $c^{(k)}(\vec{r}, t = 0)$ à l'instant $t = 0$ par :

$$c^{(k)}(\vec{r}, 0) := \frac{1}{\mu^{(k)}}\Sigma_{g'=1}^{\hat{g}}\beta^{(k,g')}\nu\sigma_f^{(g')}\mathbf{\Phi}^{(g')},$$

en écrivant les équations stationnaires issues des équations (3.3)-(3.6).

3.3.3 Discrétisation temporelle par θ-schéma

La solution $\mathbf{\Phi}(t_{i+1}) =: \mathbf{\Phi}_{i+1}$ est construite par récurrence à partir d'une solution antérieure $\mathbf{\Phi}(t_i) =: \mathbf{\Phi}_i$ par le θ-schéma :

$$\begin{aligned}\mathcal{M}_h^v\mathbf{\Phi}_{i+1} - \mathcal{M}_h^v\mathbf{\Phi}_i = \tau\theta(\mathcal{A}_h^d\mathbf{\Phi}_{i+1} - \mathbf{\Sigma}_h\mathbf{\Phi}_{i+1} + \mathcal{R}_h\mathbf{c}_{i+1}) \\ + \tau(1-\theta)(\mathcal{A}_h^d\mathbf{\Phi}_i - \mathbf{\Sigma}_h\mathbf{\Phi}_i + \mathcal{R}_h\mathbf{c}_i),\end{aligned} \tag{3.14}$$

et,

$$\mathcal{M}_h^k\mathbf{c}_{i+1} - \mathcal{M}_h^k\mathbf{c}_i = \tau\theta(\mathcal{N}_h\mathbf{c}_{i+1} + \widetilde{[\beta]}\mathbf{\Sigma}_h^f\mathbf{\Phi}_{i+1}) + \tau(1-\theta)(\mathcal{N}_h\mathbf{c}_i + \widetilde{[\beta]}\mathbf{\Sigma}_h^f\mathbf{\Phi}_i), \tag{3.15}$$

où $\theta \in [0,1]$ et τ est le pas de discrétisation en temps.
Soit le vecteur inconnu $X_i := [\mathbf{\Phi}_i^T, \mathbf{c}_i^T]^T$ représentant une solution à l'instant $t = t_i$. Le système linéaire permettant de calculer la solution à l'instant $t = t_{i+1}$ à partir de la solution à l'instant $t = t_i$ est alors :

$$\begin{bmatrix} \mathbf{\Phi}_{i+1} \\ \mathbf{c}_{i+1} \end{bmatrix} = \begin{bmatrix} D_{II} & D_{IJ} \\ D_{JI} & D_{JJ} \end{bmatrix}^{-1} \begin{bmatrix} B_{II} & B_{IJ} \\ B_{JI} & B_{JJ} \end{bmatrix} \begin{bmatrix} \mathbf{\Phi}_i \\ \mathbf{c}_i \end{bmatrix}, \tag{3.16}$$

où

$$\begin{aligned} D_{II} &:= \mathcal{M}_h^v + \tau\theta\big(\mathbf{\Sigma}_h - \mathcal{A}_h^d\big), & B_{II} &:= \mathcal{M}_h^v + \tau(1-\theta)\big(\mathcal{A}_h^d - \mathbf{\Sigma}_h\big), \\ D_{IJ} &:= -\tau\theta\mathcal{R}_h, & B_{IJ} &:= \tau(1-\theta)\mathcal{R}_h, \\ D_{JI} &:= -\tau\theta\widetilde{[\beta]}\mathbf{\Sigma}_h^f, & B_{JI} &:= \tau(1-\theta)\widetilde{[\beta]}\mathbf{\Sigma}_h^f, \\ D_{JJ} &:= \mathcal{M}_h^k - \tau\theta\mathcal{N}_h, & B_{JJ} &:= \mathcal{M}_h^k + \tau(1-\theta)\mathcal{N}_h. \end{aligned}$$

Ce dernier système peut s'écrire de manière synthétique :

$$X_{i+1} = EX_i, \tag{3.17}$$

où la matrice par blocs E s'écrit :

$$E := \begin{bmatrix} D_{II} & D_{IJ} \\ D_{JI} & D_{JJ} \end{bmatrix}^{-1} \begin{bmatrix} B_{II} & B_{IJ} \\ B_{JI} & B_{JJ} \end{bmatrix}.$$

3.3.4 Description concise du solveur en temps

Après avoir construit notre maillage 3d, nous construisons notre solveur-neutronique séquentiel, pour un seul pas de temps, suivant les trois étapes principales suivantes :

- Étape 1 : localiser les grappes selon leur chronologie par groupe, et recalculer les sections efficaces.
- Étape 2 : construire et assembler par blocs les matrices du problème.
- Étape 3 : résoudre le système par l'algorithme GMRES [60].

Dans ce qui suit, nous expliquons plus en détails certaines caractéristiques du solveur, plus précisément la construction du maillage et la prise en compte de la dynamique de la géométrie. Nous présentons également un autre solveur, utilisé au CEA [13], le solveur MINOS [4].

13. Commissariat à l'Energie Atomique et à l'énergie renouvelable (France)

Construction du maillage
Le maillage doit identifier différents milieux (combustibles à différents groupes d'énergie, grappes à deux groupes et le réflecteur constituant la couche extérieure du cœur nucléaire), particulièrement pour les grappes de contrôle qui bougent verticalement d'une manière continue au cours du temps et en fonction de l'avancement de la réaction en chaîne de fission.

Le maillage tridimensionnel représente un quart du coeur nucléaire vu de face, le reste se déduisant par symétrie.

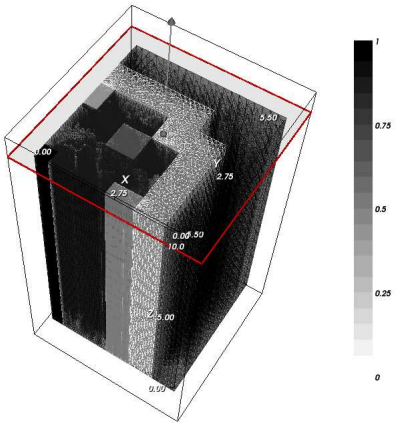


FIGURE 3.4 – Maillage en trois dimensions d'un quart du cœur nucléaire.

Mouvement continu des grappes et assemblage des matrices de production
Dans ce paragraphe, nous étudions deux procédures ; la première utilisant une adaptation du nombre des couches (mailles) en hauteur selon le pas de temps choisi (cette méthode est utilisée par les ingénieurs de CEA) et une autre méthode, qui est la notre, dans laquelle nous utilisons une interpolation des fonctions indicatrices.

Solveur du cœur MINOS
L'approximation utilisé dans le solveur MINOS : le pas de temps τ et le pas d'espace de hauteur dz sont liés. Cette relation a pour effet de faire coïncider le pas de temps avec le passage des grappes sur les surfaces des cubes. On peut également utiliser une méthode d'interpolation semblable à celle que nous allons développer dans notre solveur.

Notre Solveur
Notre solveur utilise des éléments tétraédriques générés par le logiciel de calcul scientifique FreeFem++ [55]. Nous tenons compte du mouvement linéaire et continu des grappes par une interpolation. Le coefficient d'interpolation caractérise les parties de l'éléments finis (tétraèdre) occupées par les grappes et par le combustible. Ces indicateurs nous servent à distinguer les coefficients physiques constants dans chaque milieu (voir tables (3.1)-(3.2)), et qui peuvent donc être correctement approchés par des éléments finis $\mathbb{P}_0$. Dans la mise en œuvre de notre solveur, la construction de ces matrices passe par une construction de fonctions indicatrices qui identifient les milieux, par la règle intuitive :

$$\text{Présence des grappes} \Longrightarrow \text{Absence du combustible.} \tag{3.18}$$

Dans le cadre d'approximation $\mathbb{P}_0$ sur chaque tétraèdre les fonctions indicatrices des milieux valent "1" ou "0". Cependant, cette modélisation n'est pas assez sophistiquée car elle produit des discontinuités au niveau de la courbe de la puissance. Elle assimile le milieu constituant l'interface entre les grappes d'absorption et le combustible à une couche infiniment fine. Supposons qu'au temps t_1 les grappes commencent à être au même niveau qu'un tétraèdre et qu'au temps t_2 on passe au tétraèdre suivant. Pendant une durée $t_2 - t_1$ les grappes peuvent êtres localisées dans des tétraèdres spécifiques (voir figure 3.6).
Notre approche consiste à mesurer d'abord le taux d'enfoncement des grappes, puis à affecter aux fonctions indicatrices de ces milieux (grappes et combustible) un coefficient entre zéro et un. Cette interpolation prend en compte l'évolution continue des grappes. Une fois que les fonctions indicatrices des grappes sont assemblées, celles du combustible sont déduites selon la règle (3.18).

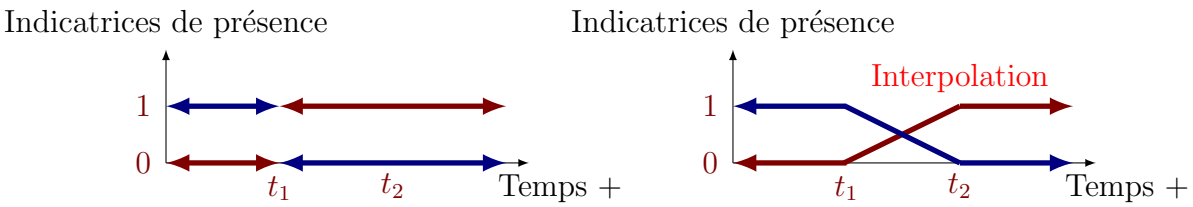


FIGURE 3.5 – Indicatrices des taux de présence des grappes (courbe rouge) et du combustible (courbe bleu) dans un tétraèdre.

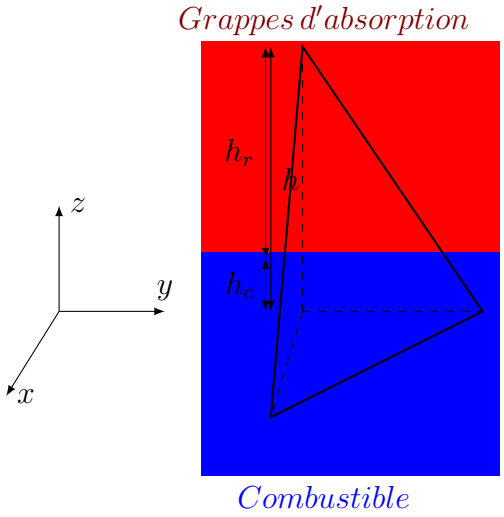


FIGURE 3.6 – Coefficients de la fonction indicatrice 3d pour les grappes absorbantes.

3.4 Parallélisation en temps de la résolution

Pour accélérer la résolution du système (3.17) nous appliquons le schéma pararéel présenté dans [41]. Nous développons maintenant les outils et techniques nécessaires à sa réalisation.

3.4.1 Décomposition de l'intervalle de temps

L'intervalle de temps $[0, T]$ est subdivisé en $\hat{n}$ segments égaux :

$$[0, T] := \cup_{n=0}^{n=\hat{n}-1}[T_n, T_{n+1}],$$

où

$$[T_n, T_{n+1}] := \cup_{i=0}^{i=l-1}[T_{n,i}, T_{n,i+1}],$$

où $l = \dfrac{\Delta T}{\tau}$ avec ΔT est la taille d'un sous intervalle de temps et τ est le pas de temps de la discrétisation temporelle, avec $T_{n,0} = T_n$ et $T_{n,l} = T_{n+1}$. Nous notons $\mathbf{F}_\triangle$ l'opérateur propagateur

$$\mathbf{F}_\triangle := E^l.$$

Remarque 3.4.1. *La notation $F_\triangle = E^l$ est abusive et ne signifie pas que le propagateur $F_\triangle$ est égal à la puissance l-ième de la matrice E. En effet, la matrice bloc E n'est pas constante au cours du temps puisque nous sommes dans le cadre d'un système dynamique.*

Soit $\{\mathbf{X}_n\}_{n\geq 0}$ la suite des solutions de (3.17) aux instants $t = T_n, \ldots, T_n$, on a $\mathbf{X}_{n+1} := X_{n,l}$.
Les éléments de cette suite satisfont la relation :

$$\begin{bmatrix} \mathbf{Id} & & & & \\ -\mathbf{F}_\triangle & \mathbf{Id} & & & \\ & \ddots & \ddots & & \\ & & & -\mathbf{F}_\triangle & \mathbf{Id} \end{bmatrix} \begin{bmatrix} \mathbf{X}_0 \\ \mathbf{X}_1 \\ \vdots \\ \mathbf{X}_{\hat{n}-1} \end{bmatrix} = \begin{bmatrix} \mathbf{X}_0 \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix}. \tag{3.19}$$

3.4.2 Application du schéma pararéel aux équations de la neutronique

Nous avons vu que l'algorithme pararéel peut s'écrire sous forme itérative, qu'il est parallélisable en temps et qu'il assure une convergence exacte vers la solution fine après $\hat{n}$ itérations, où $\hat{n}$ est le nombre des sous intervalles de la décomposition. Cette convergence exacte est due à l'aspect triangle de Pascal, une loi régissant la mise à jour des variables de la suite $(\mathbf{X}_n^p)_{n\geq 0}^{p\geq 0}$:

$$\mathbf{X}_{n+1}^{p+1} = \mathbf{G}_\triangle \mathbf{X}_n^{p+1} + \mathbf{F}_\triangle \mathbf{X}_n^p - \mathbf{G}_\triangle \mathbf{X}_n^p. \tag{3.20}$$

où $\mathbf{G}_\triangle$ est une dégradation du propagateur fin $\mathbf{F}_\triangle$. Cette dégradation peut rendre instable le schéma (3.20)(voir [8, 18, 65]). Comme la partie grossière est résolue séquentiellement, il est préférable de trouver une dégradation du modèle pour que le propagateur grossier $\mathbf{G}_\triangle$ soit le moins coûteux possible tout en restant stable et prenant toujours en compte la physique du modèle. Dans ce qui suit, nous présentons des réductions du modèle neutronique qui permettent de dégrader la propagation fine mais aussi d'assurer une réduction de la complexité de l'algorithme.

3.5 Tests, algorithme parallèle, résultats numériques et discussion

Avant d'entamer nos tests numériques, nous présentons tout d'abord à la figure 3.7 la géométrie (radiale) du cœur ainsi que les milieux d'occupation pour les combustibles et les grappes d'absorptions. La figure montre également les positionnements des grappes à l'instant $t = 0$, ainsi que les volumes des combustibles.

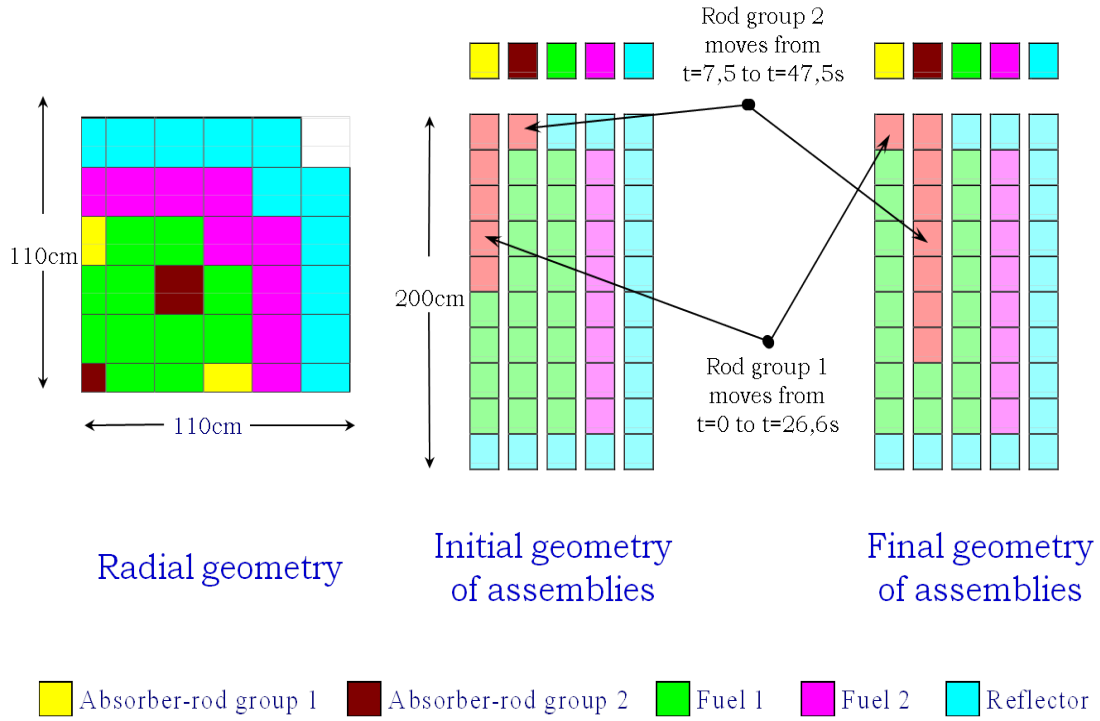


FIGURE 3.7 – Géométrie du cœur et positionnement initial des grappes d'absorption.

Pour nos tests sur le modèle de la neutronique, nous considérons une réduction du modèle. Cette réduction portera principalement sur l'aspect dynamique des grappes de contrôle qui ont une évolution continue en temps dans la direction verticale par rapport au cœur du réacteur.

3.5.1 Simulation de la production d'énergie

Dans cette partie, nous simulons une physique réelle qui prend en compte le mouvement des grappes d'absorption (des deux groupes) dans le cœur. Au temps $t = 0sec$ le premier groupe de barre est à position initiale $z = 100$cm tandis que le deuxième groupe est à position plus haute $z = 180cm$.
Le mouvement linéaire des grappes d'absorption est comme suit :
Groupe 1 : $(t = 0 \ |z = 100$ cm$) \nearrow (t = 26,5|z = 180$ cm$)$, avec une vitesse de 3 cm/sec
Groupe 2 : $(t = 7,5|z = 180$ cm$) \searrow (t = 47,5|z = 60$ cm$)$, avec une vitesse de 3 cm/sec
Les séries de courbes présentées à la figure 3.8 correspondent au variations de la réactivité k_{eff} du modèle en fonction du temps ou de l'évolution des grappes. À chaque pas de

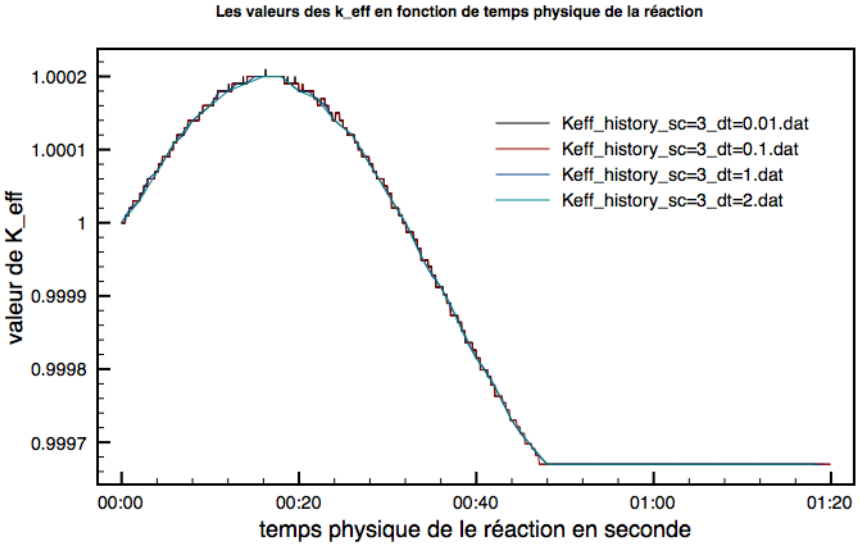


FIGURE 3.8 – Variation des k_{eff} en fonction du temps.

temps, et après avoir assemblé les matrices de réactivité $\widehat{\mathrm{A}}$ et $\widehat{\mathrm{F}}$ nous faisons appel à l'**algorithme** 3 pour résoudre le problème aux valeurs propres associé et calculer la valeur propre généralisée k_{eff}.

3.5.2 Réduction du modèle physique du réacteur autour des scénarios sur la dynamique des grappes

Dans cette section, nous présentons les scénarios dynamique et statique dont les définitions sont fortement liées aux grappes de contrôle, et spécialement à leurs niveaux d'enfoncement dans le cœur. Nous montrons qu'une simplification de modèle peut être un atout pour accélérer la résolution par l'algorithme pararéel.
Nous notons *modèle complet* respectivement *modèle réduit* notre modèle de cinétique neutronique en présence respectivement en absence des groupes de concentrations de précurseurs. Une étude approfondie de la réduction du modèle est fondamentale afin de permettre une dégradation du solveur fin et déterminer les caractéristiques d'un solveur stable. Cette dégradation peut être réalisée soit par un dé-raffinement du maillage (spatio-temporel), soit par une réduction du nombre des itérations d'inversion soit en agrandissant la tolérance lors de la résolution du système linéaire ou par simplification de la physique du modèle (si cela est possible). Dans le cadre de la cinétique neutronique, plusieurs paramètres permettent une simplification du modèle tout en gardant des résultats de simulation assez proches de ceux obtenus avec un modèle complet. A cet effet, on peut par exemple diminuer le nombre des groupes d'énergies pour le flux neutronique, ou bien encore le nombre des groupes des concentrations des précurseurs. Nous pouvons aussi approcher la productivité du cœur par un simple changement de la réactivité.
Nous détaillons dans la suite l'aspect dynamique de la physique du modèle et nous proposons une réduction de cette dynamique. En effet, cette réduction nous permet de rendre le calcul moins coûteux et avec de bons résultats comme nous le montrons dans la partie

expérimentale.

Scénario dynamique

La cinétique dans ce cadre est caractérisée par un mouvement linéaire des grappes de contrôle au cours du temps avec une vitesse égale à 3 cm/sec. Ces grappes dans leur configuration initiale fournissent un état critique du réacteur (i.e. k_{eff}=1), cette configuration initiale positionne les groupes des grappes (le groupe 1 est placé à une hauteur égale à $z = 100$ cm et le groupe 2 est placé à une hauteur égale à $z = 180$ cm). Au cours du temps le réacteur se met progressivement à produire des neutrons par fission, car les grappes d'absorption, spécialement le groupe 1, remontent ce qui fait que le taux d'absorption diminue. Cela réduit la surface de contact des grappes avec les combustibles. Pendant une durée d'environ 20 sec, la surface de contact entre les grappes et le combustible est plus petite que celle assurant un état critique du réacteur. Ce dernier va donc produire de plus en plus d'énergie jusqu'à ce que cette surface de contact assure un second état critique. Ceci est le résultat d'un enfoncement du deuxième groupe des grappes partant de $z = 180$ cm jusqu'à $z = 60$ cm pour une durée de temps égale à 40 sec débutant à 7.5 sec. Pendant la dynamique des grappes d'absorption, l'énergie nucléaire (en norme de flux neutronique) atteint son pic et décroît pour atteindre la deuxième phase critique, tout en continuant à décroître jusqu'à l'extinction du réacteur.
Nous commençons dans la suite par la réduction de la physique dynamique du modèle par une fixation intelligente des grappes.

Scénario statique

Dans ce scénario les grappes d'absorption sont fixées. Nous pouvons approcher le scénario réel en choisissant correctement les positions des grappes. Dans le cadre du scénario dynamique, il y a deux phases principales du réacteur qui sont caractérisées par deux états du cœur : sous-critique et sur-critique. Une reproduction de ces phases (relatives aux états du cœur) est tout à fait possible au sens où nous pouvons arbitrairement contrôler l'état du cœur.
Dans ce qui suit nous proposons deux approches pour cette reproduction.

Positionnement spécial des grappes Il suffit de choisir deux positions des deux groupes de grappes, de telle sorte que la surface de contact soit strictement inférieure à celle d'un état critique, soit par exemple

	$t \in [0, 20]$	$t \in]20, 80]$
Groupe 1	z=160 cm	z=180 cm
Groupe 2	z=180 cm	z=60 cm

FIGURE 3.9 – Changement de l'état du réacteur par un changement des positions des grappes.

Changement de la réactivité Cette approche ne change pas la configuration initiale des grappes d'absorption. Néanmoins nous sommes en mesure de reproduire le phénomène

assuré par les positionnements (voir figure (3.9)). En effet, ces positions correspondent à une certaine réactivité, il suffit donc de reproduire cette réactivité correspondante à une valeur donnée de k_{eff} aux temps précis relatifs aux positions des grappes, pour reproduire le même scénario.

	$t \in [0, 20]$	$t \in]20, 80]$
réactivité	10008.e-5	9998.e-5

FIGURE 3.10 – Changement de l'état du réacteur par un changement de sa réactivité.

Remarque 3.5.1. *Dans le cadre du scénario statique, il est préférable pour la mise en oeuvre d'utiliser la seconde méthode, car celle-ci ne demande pas plusieurs mises à jour des coefficients, contrairement à la première méthode. En outre, cette méthode facilite la mise au point et accélère l'exécution de l'algorithme résultant sans aucun changement de résultats par rapport à la première approche*

Remarque 3.5.2. *Avec le scénario statique il est difficile de reproduire les résultats du scénario réel. Ceci est clair au moins dans le scénario statique car nous perdons déjà la continuité de la dynamique qui a un impact direct sur l'énergie résultant de ce modèle. Néanmoins, avec le scénario statique et avec la deuxième méthode, nous pouvons jouer sur la réactivité pour atteindre le même pic que celui du scénario réel. La version statique relative au positionnement des grappes ne propose pas cette possibilité.*

3.5.3 Pararéel pour la neutronique

Ayant le flux propre (critique), obtenu de l'algorithme 3 nous pouvons entamer une cinétique neutronique dont les instances sont décrites par l'algorithme 4 (voir page 75).
Pour la mise en œuvre et le test numérique des méthodes précédentes, nous avons utilisé des données de différentes sections efficaces fournies par les ingénieurs du CEA. Les simulations numériques ont été effectuées sur une machine parallèle *SGI* du laboratoire Jacques-Louis Lions, qui possède 64 coeurs cadencés à 2.0 GHz, 256 Go de mémoire partagée et un réseau de communication Numalink (15 GB/s) offrant une des meilleures bandes passantes (supérieure à l'Infiniband 4xQDR), avec une faible latence. Nous avons donc exploité cette machine massivement parallèle avec la bibliothèque MPI [64] et le logiciel de calcul scientifique FreeFem++. Nos données de physique sont présentées à la table 3.1 et complétées par les tables 3.2.
Ce paragraphe présente les différentes sections. Le flux neutronique $\mathbf{\Phi}$ concerne dans nos cas tests deux groupes d'énergies qui se trouvent dans chacun des quatre milieux du domaine combustible A, combustible B, grappes d'absorption et réflecteur. Deux groupes d'énergie cohabitent dans chaque milieu. Les neutrons correspondants sont en interaction permanente. Ces neutrons peuvent changer d'un groupe à un autre par un ralentissement (chocs) ou un simple changement de direction (diffusion). Les sections efficaces de changement d'énergie de neutrons sont présentées sous forme de quatre tables à la table 3.1. Chaque table présente la section dans un milieu spécifique. Les données neutroniques relatives à la section de scattering sont présentées dans les tables 3.2.

Algorithme 4 : Algorithme pararéel de la cinétique neutronique

Input : $\hat{n} := \#slave\,proc$ that gives rise to I_n from I, fine time step τ_F and coarse one τ_G

Input : $\mathbf{X}_0^0 = [(\Phi^\star)^T, (\mathbf{c}^\star)^T]^T$ as initial conditions, ϵ a tolerance of the algorithm ;

Input : a solver $\mathcal{A}$, a data vector $\mathbf{X}$;

Routine$(\mathcal{A}, \mathbf{X})$

1) Positioning the absorber rods with respect to the dynamic chronology;

2) Constructing matrices related to equations (3.14)-(3.15)-(3.16);

3) Serial propagation of $\mathbf{X}$ using $\mathcal{A}$ with respect to its screenplay [14] (the result is denoted by $\mathcal{A}\mathbf{X}$);

end Routine ;

$p \longleftarrow 0$;

repeat

  if *master processor* **then**

    foreach $n \in \{0, .., \hat{n} - 1\}$ **do**

      1) **Call : Routine**$(\mathbf{G}_\triangle, \mathbf{X}_n^p)$ (i.e. coarse-serial propagation);

      2)

        if $p = 0$ **then**

          repeat return to 1 **with** $\mathbf{X}_{n+1}^p$ **until** $n = \hat{n} - 1$

        else

          Construct $(\mathbf{X}_n^p)_{n \geq 1}$ with respect to relationship (3.20);

        end

      3) **Mpi_Send** $(\mathbf{X}_n^p, processor(n))$;

    end

  else

    forall $slave\,processor(n)/n \in \{0, \ldots, \hat{n} - 1\}$ **do**

      Mpi_Recv $(\mathbf{X}_n^p, master\,processor)$;

      Call : Routine$(\mathbf{F}_\triangle, \mathbf{X}_n^p)$ (i.e. fine-parallel propagation);

      Mpi_Send $(\mathbf{F}_\triangle \mathbf{X}_n^p, processor(n))$;

    end

  end

  if *master processor* **then**

    foreach $n \in \{0, .., \hat{n} - 1\}$ **do**

      Mpi_Recv $(\mathbf{F}_\triangle \mathbf{X}_n^p, processor(n))$;

      Evaluate $\epsilon_n^p = ||\mathbf{F}_\triangle \mathbf{X}_n^p - \mathbf{X}_{n+1}^p||_{l^2(\mathbb{R}^{8\hat{s}})}$;

    end

  end

  $p \leftarrow p + 1$;

  Mpi_Broadcast $(master\,processor, \epsilon_n^p)$;

until $\max_n \epsilon_n^p \leq \epsilon$;

Données physique	Milieux			
Sections	Combustible A		Combustible B	
	groupe1	groupe2	groupe1	groupe2
σ_t	0.23409670	0.93552546	0.23381787	0.95082160
σ_f	0.006477691	0.1127328	0.007503284	0.1378004
	Grappes		Réflecteur	
	groupe1	groupe2	groupe1	groupe2
σ_t	0.23409670	0.93552546	0.20397003	1.26261670
σ_f	0.006477691	0.1127328	.0	.0
vitesse	1.25e+7	2.5e+5	1.25e+7	2.5e+5.

TABLE 3.1 – Données neutroniques pour le coeur du benchmark 3D.

Combustible A

$\sigma_s^{(g\rightarrow g')}$	groupe(1)	groupe(2)
groupe(1)	.20613914	.01755550
g(roupe2)	.0	.84786329

Combustible B

$\sigma_s^{(g\rightarrow g')}$	groupe(1)	groupe(2)
groupe(1)	.20564756	.01717768
groupe(2)	.85156526	.0

Grappes d'absorption

$\sigma_s^{(g\rightarrow g')}$	groupe(1)	groupe(2)
groupe(1)	.20558914	.01755550
groupe(2)	.84406329	.0

Réflecteur

$\sigma_s^{(g\rightarrow g')}$	groupe(1)	groupe(2)
groupe(1)	.17371253	.02759693
groupe(2)	1.21325319	.0

TABLE 3.2 – Données de la section "scattering" pour les deux groupes d'énergies.

3.5.4 Le solveur séquentiel

Nous étudions dans ce paragraphe la stabilité et l'ordre de convergence du θ-schéma pour la discrétisation temporelle des équations (3.7)-(3.8). Les tests effectués sur les différents modèles ont montré que le schéma explicite correspondant à une valeur nulle de θ est inconditionnellement instable, tandis que le schéma de Crank-Nicolson relatif à une valeur de $\theta = \frac{1}{2}$ présente des oscillations si les groupes de concentrations de précurseurs sont présents, que ce soit lors de la phase sur-critique [15] caractérisée par la montée de la puissance du cœur, ou de la phase sous-critique [16] caractérisée par une diminution de la puissance. Il apparaît que l'unique valeur pour laquelle le θ-schéma est inconditionnellement stable et ne présente pas d'oscillation est le schéma implicite avec $\theta = 1$. Dans la figure 3.11 nous calculons numériquement l'ordre d'approximation du schéma de discrétisation en temps du schéma d'Euler implicite ($\theta = 1$). Ces courbes représentent deux pentes décrivant l'ordre de convergence du schéma pour les deux phases de l'état du cœur sur-critique et sous-critique. Nous remarquons que l'ordre est indépendant des phases.
Nous présentons maintenant dans ce qui suit quelques graphes montrant le comportement de la norme du flux neutronique dans le coeur nucléaire. Tout d'abord, nous simulons un modèle neutronique complet, avec six groupes de précurseurs et deux groupes d'énergie de flux neutronique. Nous éliminons ensuite les groupes de précurseurs et examinons l'impact de leur présence sur le comportement de la puissance. Enfin, nous présentons, un peu plus

15. Phase d'emballement de la valeur de l'intégrale du flux neutronique sur cœur nucléaire.
16. Phase d'extinction du cœur nucléaire.

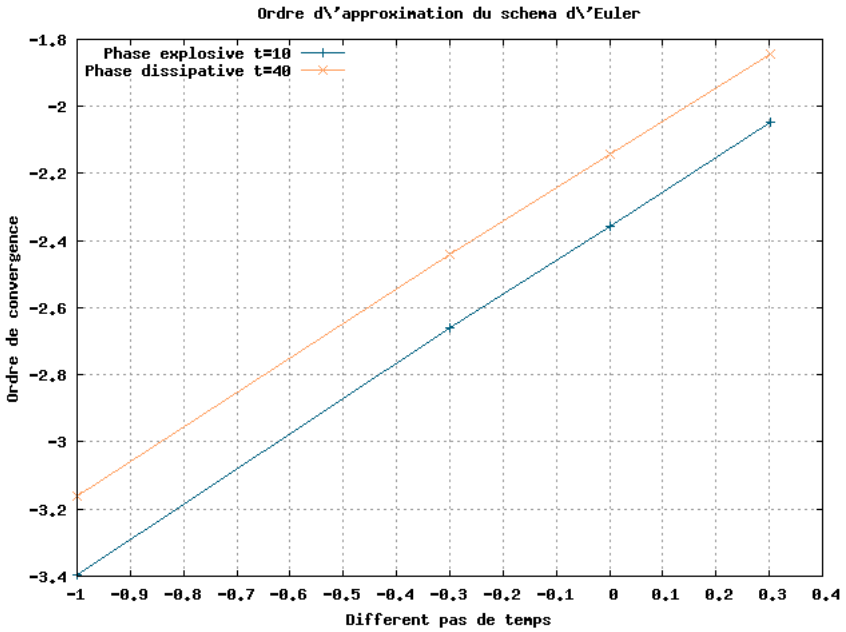
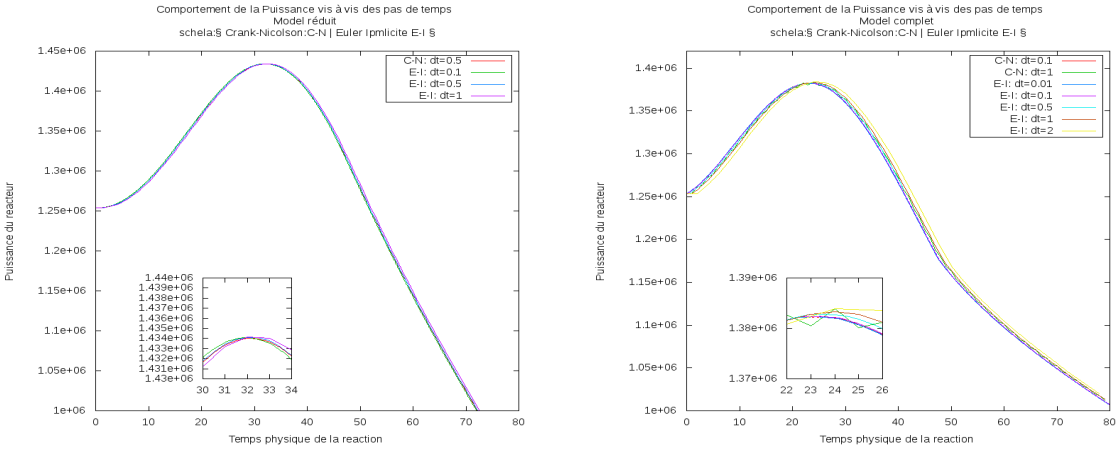


FIGURE 3.11 – Ordre de convergence du schéma d'Euler implicite, pour le scénario dynamique du modèle complet (six groupes de précurseurs) : échelle logarithmique décimale sur les axes "x" et "y".

FIGURE 3.12 – Valeurs de la norme du flux moyen de réacteur avec le scénario *dynamique.*

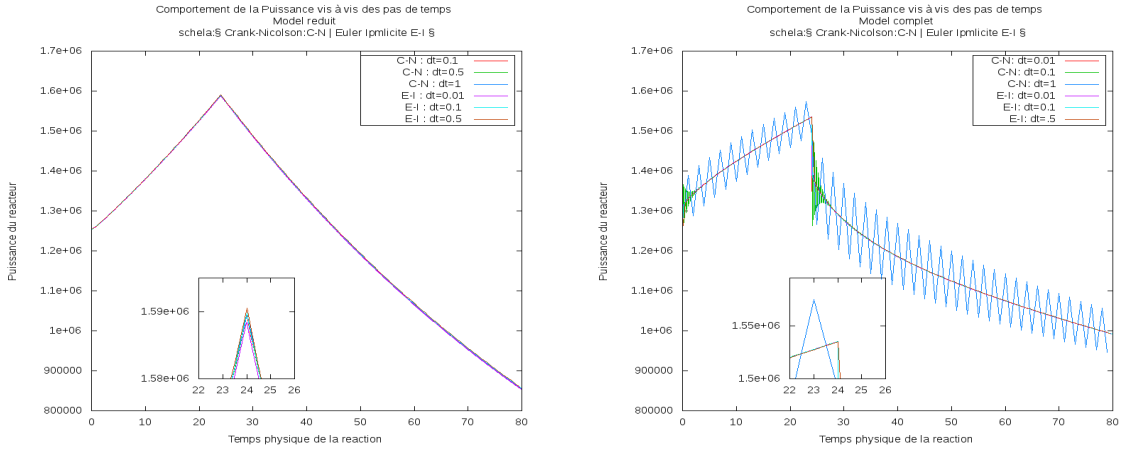


loin sous forme de graphes, le comportement de la norme du flux dans le coeur nucléaire, vis-à-vis du pas de discrétisation en temps.

3.5.5 Critère d'arrêt de l'algorithme pararéel

Dans cette sous-section nous donnons une stratégie consistant à arrêter la simulation si une certaine tolérance est violée. Nous définissons donc un critère d'arrêt pour l'algorithme pararéel.
Suivant une discrétisation en temps du premier ordre, comme c'est le cas pour nos résultats numériques, l'ordre de convergence de l'algorithme pararéel peut s'ecrire comme (pour plus

FIGURE 3.13 – Valeurs de la norme de flux moyen neutronique avec le scénario *statique.*



de détails sur l'ordre de convergence on renvoie à [41]) :

$$\|\mathbf{X}_n^p - X(t_n)\|_{L^2(\mathbb{R}^{\hat{s}})} \approx o(\Delta T^p + \tau),$$

où $X(t_n)$ représente la solution, produite par un solveur fin à l'instant t_n, du flux neutronique avec les concentrations des précurseurs (le cas échéant).
L'algorithme pararéel peut fournir une erreur de l'ordre de 2τ après certaines itérations.

$$\begin{aligned}\|\mathbf{X}_n^p - X_n\|_{L^2(\mathbb{R}^{\hat{s}})} &\leq \|\mathbf{X}_n^p - E^{n\times l}\mathbf{X}_0\|_{L^2(\mathbb{R}^{\hat{s}})} + \|E^{n\times l}\mathbf{X}_0 - X_n\|_{L^2(\mathbb{R}^{\hat{s}})} \\ &\leq o(\Delta T^p) + o(\tau) \\ &\leq 2o(\tau) = o(\tau), \text{ si } \Delta T^p \leq \tau.\end{aligned}$$

Le critère d'arrêt utilisé est alors une tolérance ϵ de l'ordre de τ comme étant l'erreur qu'effectue le solveur séquentiel fin par rapport à la solution de référence. De ce fait, il est garanti que la solution pararéelle et la solution exacte appartiennent à une même boule centrée en la solution fine et de rayon égale à $o(\tau)$. Dans le pire cas, l'écart entre la solution pararéelle et la solution exacte est au plus le diamètre de cette boule (i.e. $2o(\tau)$). Alors l'algorithme pararéel est arrêté à une itération $p^\star$:

$$p^\star := p(\epsilon).$$

Nous nous intéressons maintenant à la complexité de l'algorithme pararéel, c'est-à-dire au nombre de résolutions de systèmes linaires au cours des itérations. Supposons que pendant la durée d'un intervalle de temps le propagateur fin effectue n_F résolutions du système (3.17), et que le propagateur grossier effectue n_G résolutions. La complexité de l'algorithme pararéel à l'itération p est :

$$\mathcal{C}(p) = (\hat{n}n_G + n_F)p + \hat{n}n_G.$$

De plus si nous choisissons les pas de temps fins τ_F et grossiers n_G tels que $\tau_G = s\tau_F$, et en utilisant le fait que $n_F = \frac{\Delta T}{\tau_F}$ et $n_G = \frac{\Delta T}{\tau_G}$ nous avons alors :

$$\mathcal{C}(p) = \frac{T}{\tau_G}((1 + \frac{s}{\hat{n}})p + 1).$$

D'après la formule ci-dessus, nous constatons que la complexité de l'algorithme pararéel diminue si le nombre de processeurs disponibles augmente. Néanmoins, cela n'implique généralement pas que l'algorithme convergera plus vite.

3.5.6 Comportement de l'algorithme pararéel vis-à-vis de la norme du flux moyen du cœur

Dans cette sous-section nous présentons une série de courbes (voir Figures 3.14 et 3.15) décrivant l'état de la norme du flux neutronique dans le réacteur en temps réel de la réaction. Ces courbes sont accompagnées par des courbes décrivant le comportement de la différence quadratique entre la solution à l'instant t_n^- et la condition initiale à l'instant t_n^+ de l'intervalle suivant.

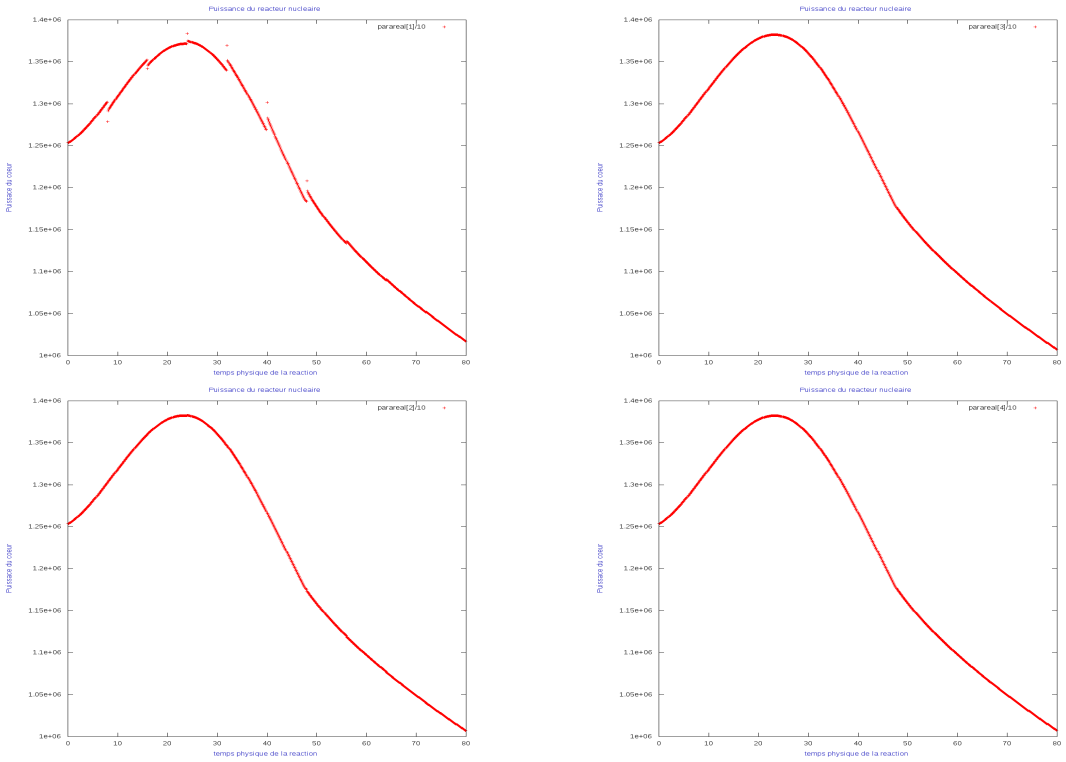


FIGURE 3.14 – Comportement du pararéel (itérations 1-2-3-4) sur la norme du flux neutronique du cœur avec le modèle complet pour le scénario dynamique, $\tau_F = 10^{-1}$, $\tau_G = 4$ et $\hat{n} = 10$.

3.5.7 Etude numérique de la convergence de l'algorithme

Dans cette sous-section nous montrons la convergence numérique des erreurs des solutions calculées avec l'algorithme pararéel et la solution séquentielle (voir Tables 3.3-3.4-3.5-3.6) dont nous varions la physique que nous traitons. L'erreur considérée dans ces

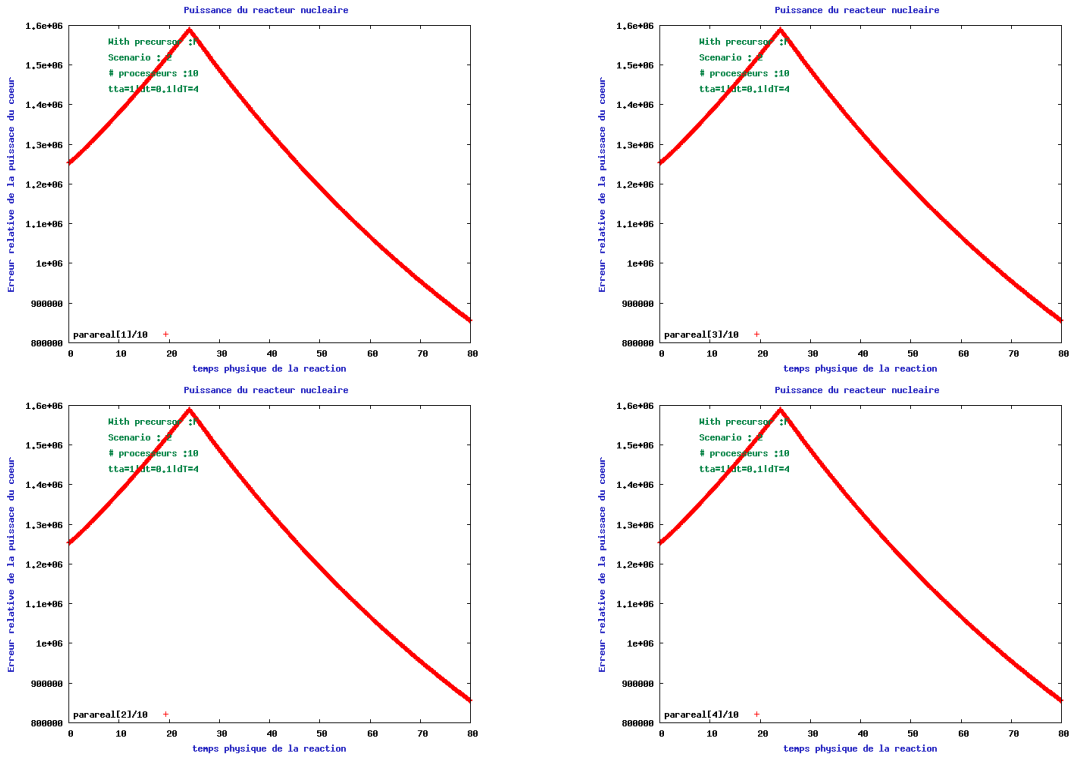


FIGURE 3.15 – Comportement du pararéel appliqué à la neutronique (itérations 1-4) sur la norme du flux moyen dans le cœur avec le modèle complet et un scénario statique, $\tau_F = 10^0$, $\tau_G = 4$ et $\hat{n} = 10$.

courbes est la maximale parmi les ϵ_n^p, où :

$$\epsilon_n^p := \frac{\|\mathbf{X}_n^p - X_n\|_{L^2(\mathbb{R}^s)}}{\|X_n\|_{L^2(\mathbb{R}^s)}}.$$

Nous présentons tout d'abord les courbes relatives au scénario dynamique du modèle complet. Dans un second temps, nous présentons les courbes d'erreur obtenues pour le scénario statique d'un modèle réduit.
En tenons compte des résultats sur l'ordre de convergence de notre schéma de discrétisation en temps qui sont présentés dans la Figure 3.11 dont la solution de référence à été approximativement reproduite par une discrétisation très fine $\tau_F = 10^{-2}$. Nous nous sommes donc intéressé à des erreurs du même ordre de grandeur que celle-ci. Il faut donc considérer une convergence de l'algorithme pararéel à une erreur de 10^{-2} près ou bien de 10^{-3} près. Néanmoins, nous présentons des résultats de convergence plus poussés (voir Fugure 3.5.7–3.5.8–3.21) pour démontrer la convergence vers la solution numérique séquentielle.

Remarque 3.5.3. *Nous soulignons que la courbes d'erreur présenté à la Figure 3.5.7 (respectivement 3.5.8) correspond aux résultats du comportement de la norme de flux moyen intégré présenté à la Figure 3.14 (respectivement 3.15)*

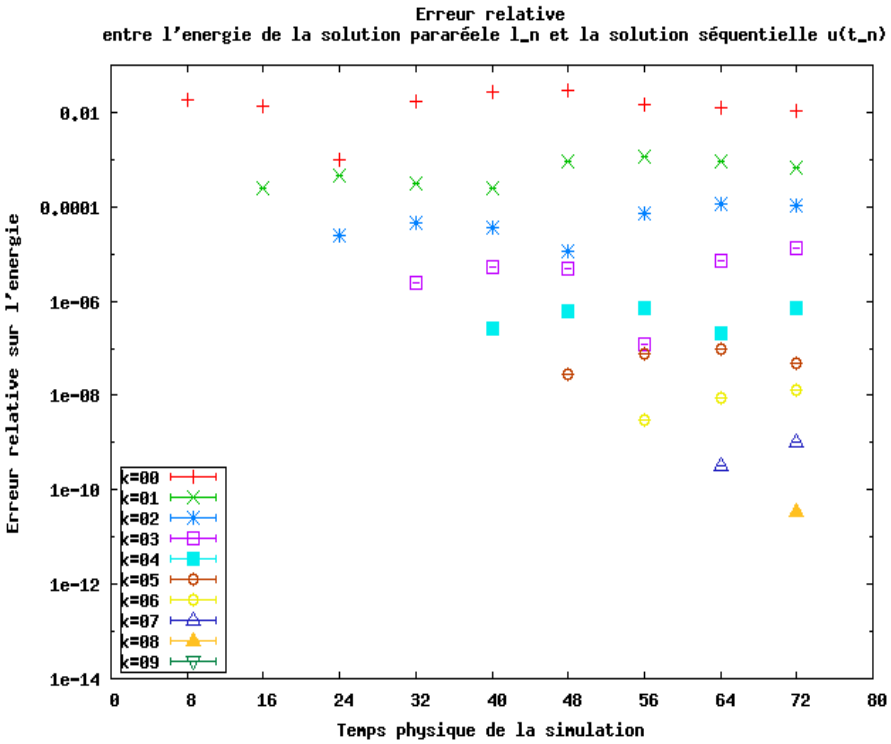


FIGURE 3.16 – Evolution de l'erreur relative entre les énergies pararéelles et fines de reference. Evolution avec un modèle complet (6 groupes de précurseurs) avec un scénario dynamique, $\tau_F = 10^{-1}$, $\tau_G = 4$ et $\hat{n} = 10$.

3.5.8 Le solveur grossier en scénario statique

Nous présentons dans cette sous-section une dégradation du modèle permettant une accélération de la résolution sans nuire à la convergence de l'algorithme. Cette dégradation est effectuée pour le solveur grossier auquel nous mettant en un scénario statique. Ce scénario comme expliqué dans la section 3.5.2 utilise une variation de la réactivité du système sur deux intervalles de temps physique de la réaction. La procédure employée nous permet d'accélérer le calcul et de ne pas recalculer la matrice de production à tout les pas de temps. En effet, uniquement une multiplication de la matrice de production (déjà en mémoire cache) par un coefficient suffit.
Nous étudions à la figure 3.20 (page 85) l'influence de la taille du pas de temps grossier τ_G sur les résultats pararéels d'un modèle complet de la cinétique neutronique avec un scénario dynamique, un pas de temps fin $\tau_F = 10^{-1}$ et 10 sous intervalles. Nous comparons ensuite les exécutions en wallclock (relative à la super-machine) avec la mise à disposition de 10 processeurs traitant d'une façon parallèle les sous-problèmes. Pour avoir un ordre d'idée, nous signalons que le calcul séquentiel (execution avec un seul processeur) dure **03h :06mn :57s**, et que les temps affichés à la figure 3.20 (page 85) correspondent au temps wallclock des erreurs représentées.
Bien que les complexités soient assez différentes selon les deux cas envisagés, on constate une accélération de celle qui a plus de complexité. Ce gain vient principalement du fait que le pas de temps correspondant à la courbe la plus rapide (en rouge) est le minimum des deux.

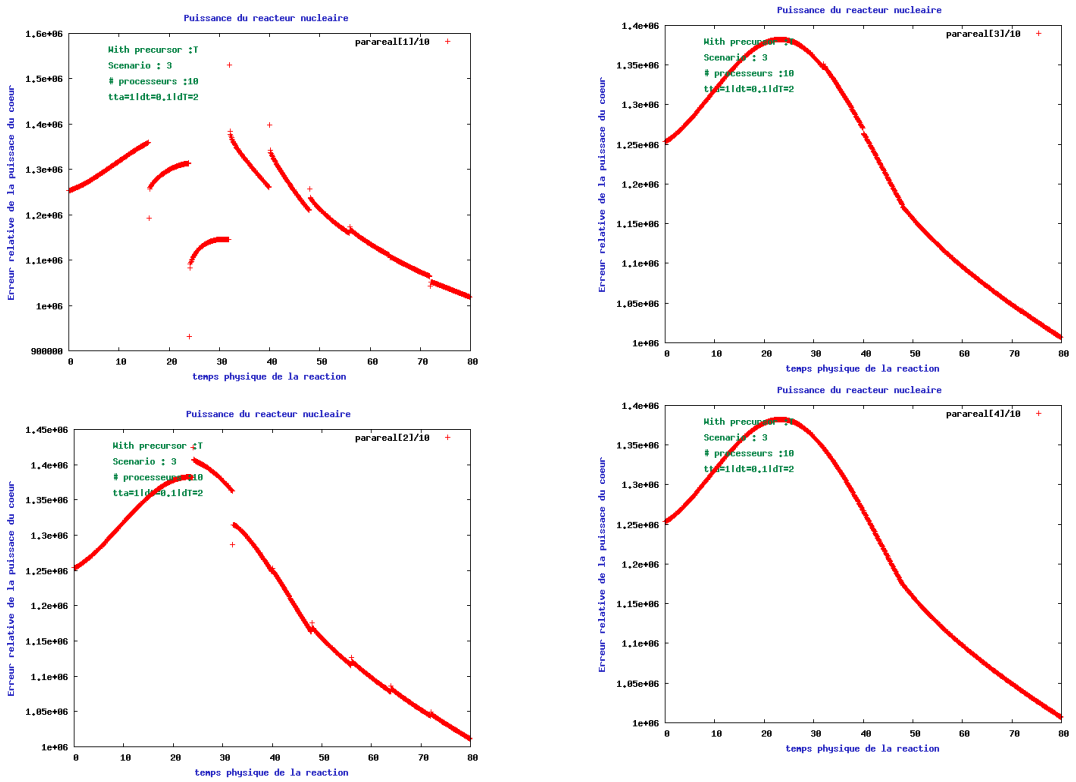


FIGURE 3.17 – Comportement de la norme de flux neutronique (itérations 1-2-3-4) avec un modèle complet et un scénario statique pour le calcul grossier et un scénario dynamique pour le calcul fin, $\tau_F = 10^{-1}$, $\tau_G = 2$ et $\hat{n} = 10$.

Ce comportement d'accélération de la convergence apparaît clairement sur les Tables 3.3–3.6. En effet, si une itération pararéelle est fixée dans une Table, on constate que l'erreur des sauts est d'autant plus décroissante que le pas de temps grossier τ_G se rapproche du pas fin τ_F employé parallèlement.

Nous présentons à la figure 3.21 l'accélération de la résolution en fonction du nombre de subdivisions par sous-domaines temporels. La résolution sur chaque sous-domaine est confiée à un processeur de la machine massivement parallèle. Comme nous nous intéressons aux erreurs de convergence de l'ordre de 10^{-2} ou de l'ordre de 10^{-3} nous constatons un gain en temps cpu de traitement parallèle de la cinétique neutronique par rapport à un calcul séquentiel. En outre, le fait de passer de 8 processeurs à 16 processeurs permet de passer de $2h$: 45mn à $1h$: 15mn de traitement. Nous remarquons une division par deux (approximative) du temps cpu, qui est en adéquation avec une efficacité locale d'une machine à 8 cœurs par processeur. Signalons également que si le nombre des subdivisions $\hat{n}$ augmente alors le nombre des processeurs agents en parallèle augmente également. La communication devient donc de plus en plus importante. Ceci explique le fait que la courbe d'erreur en temps wallclock (cpu) relative à l'utilisation de 40 processeurs est moins bonne que celle pour 20 processeurs. La machine sur laquelle nous avons lancé ces tests numériques a donc atteint sa saturation de speed-up.

Remarque 3.5.4. *L'algorithme pararéel est implémenté avec une architecture maître-esclave pour laquelle nous avons deux types de communications : une communication de*

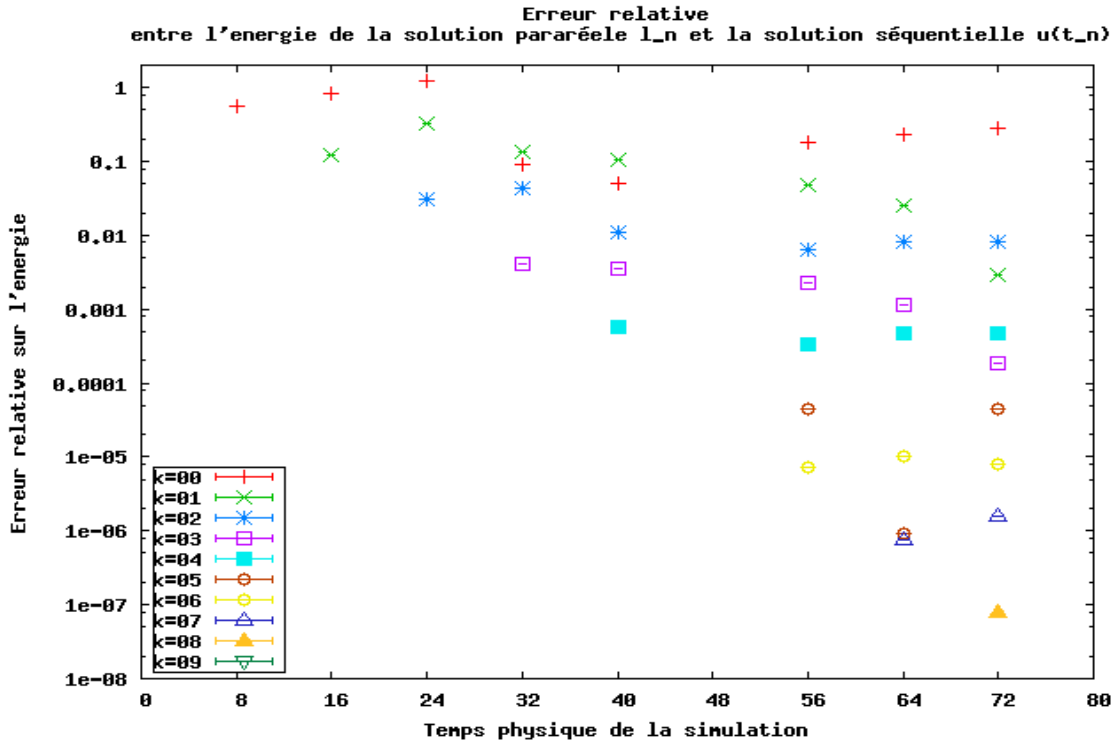


FIGURE 3.18 – Convergence du pararéel appliqué à la neutronique (itérations 00-09), un modèle complet avec un scénario dynamique pour le calcul fin et utilisant un scénario statique pour le calcul grossier. $\tau_F = 10^{-1}$, $\tau_G = 2$ et $\hat{n} = 10$.

distribution et une communication de collecte. Dans la communication distribution, le processeur maître envoie l'information vers tous ses processeurs agents. Par contre dans la communication collecte, le maître lui-même reçoit et collecte les informations depuis ses agents. Dans les deux cas, il s'agit de la même quantité d'information qui passe dans les deux sens, sauf si le programmeur dégrade le maillage pour le solveur grossier. En effet, si un déraffinement de maillage accompagne une dégradation du modèle utilisé pour le solveur grossier alors l'information passant dans la communication distribution est plus grossière que l'autre car dans ce type de communication nous communiquons les solutions grossières correspondantes aux conditions initiales grossières de l'algorithme pararéel. Le second type de communication est consacré à la correction de l'erreur grossière, qui nécessite une information fine qui doit être communiquée par les processeurs agents (sous-domaine) au processeur maître, ce qui rend cette communication importante et influançant directement le temps du passage de l'information. Ceci est clair, en particulier, si on augmente le nombre de processeurs agents ou le nombre de sous-domaines.

3.6 Conclusion

Bien que la physique du modèle soit une montée de puissance exponentielle au cours des vingt premières secondes, le pararéel a montré une stabilité pour différents choix de pas de temps grossier strictement plus grands que le pas fin utilisé en parallèle. Ce comportement

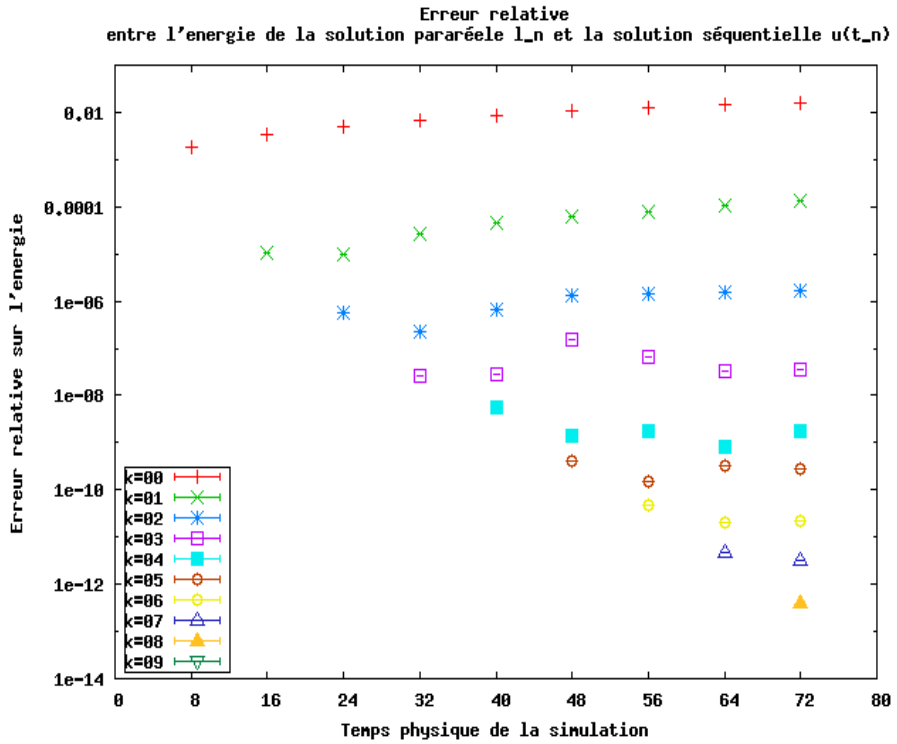


FIGURE 3.19 – Evolution de l'erreur relative, entre les énergies pararéelles et fines de référence, avec un modèle réduit (absence des groupes de précurseurs) et un scénario statique, $\tau_F = 10^{-1}$, $\tau_G = 4$ et $\hat{n} = 10$.

stable a bien aidé l'algorithme à assurer un taux de convergence du même ordre qu'un schéma séquentiel vers une solution exacte, avec une durée de temps strictement plus petite.

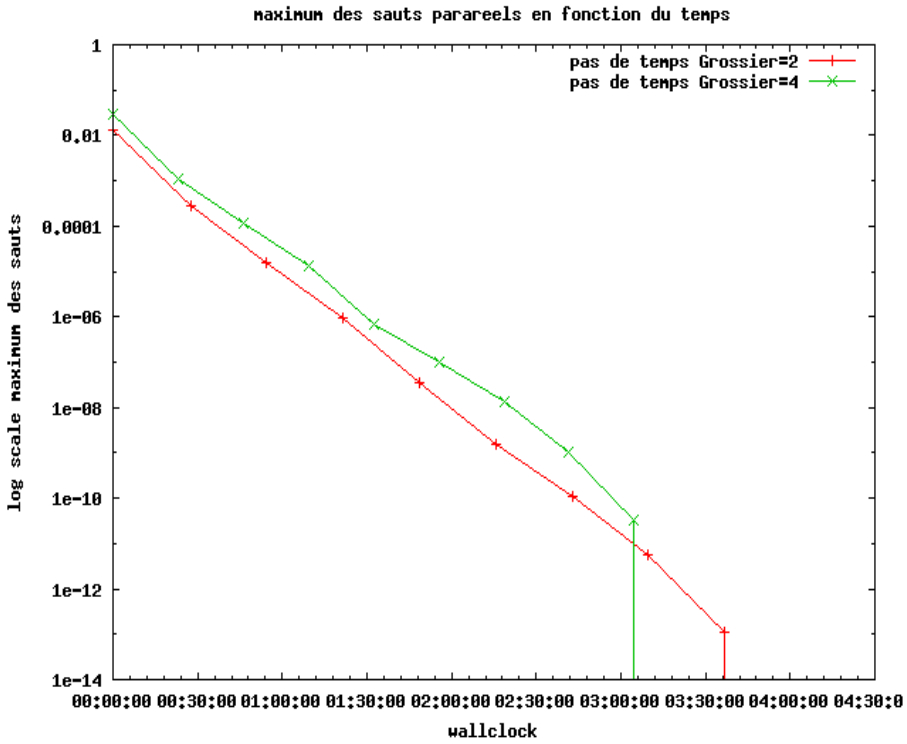


FIGURE 3.20 – Temps réel de simulation CPU (Wallclock sous format h :m :s), modèle complet avec un scénario dynamique selon une variation du pas de temps grossier τ_G, $\tau_F = 10^{-1}$, $\tau_G = \{2(rouge), 4(verte)\}$.

TABLE 3.3 – Itérations 1 et 2 pararéelles pour le modèle complet en scénario dynamique.

τ_F	τ_G	$\max_{n\geq 0} \epsilon_n^1$
0.01	2	1.42e-02
0.01	4	2.93e-02
0.01	8	6.09e-02
0.1	0.5	0.29e-02
0.1	1	0.65e-02
0.1	2	1.35e-02
0.1	4	2.88e-02
0.1	8	6.04e-02
0.5	2	1.13e-02
0.5	4	2.68e-02
0.5	8	5.83e-02
1	2	0.75e-02
1	4	2.29e-02
1	8	5.43e-02

τ_F	τ_G	$\max_{n\geq 0} \epsilon_n^2$
0.01	2	3.04e-04
0.01	4	1.16e-03
0.01	8	4.11e-03
0.1	0.5	1.31e-05
0.1	1	6.48e-05
0.1	2	0.28e-04
0.1	4	1.11-03
0.1	8	4.02e-03
0.5	2	1.71e-04
0.5	4	0.88e-03
0.5	8	3.55e-03
1	2	7.42e-05
1	4	0.63e-04
1	8	3.01e-03

TABLE 3.4 – Itérations 3 et 4 pararéelles pour le modèle complet en scénario dynamique.

τ_F	τ_G	$\max_{n\geq 0} \epsilon_n^3$
0.01	2	3.04e-04
0.01	4	1.16e-03
0.01	8	4.11e-03
0.1	0.5	1.31e-05
0.1	1	6.48e-05
0.1	2	2.81e-04
0.1	4	1.11e-03
0.1	8	4.02e-03
0.5	2	1.71e-04
0.5	4	8.80e-04
0.5	8	3.55e-03
1	2	7.42e-05
1	4	6.30e-04
1	8	3.01e-03

τ_F	τ_G	$\max_{n\geq 0} \epsilon_n^4$
0.01	2	1.73e-05
0.01	4	1.22e-04
0.01	8	7.85e-04
0.1	0.5	1.41e-07
0.1	1	1.67-06
0.1	2	1.51e-05
0.1	4	1.13e-04
0.1	8	7.53e-04
0.5	2	7.02e-06
0.5	4	7.80e-05
0.5	8	6.17e-04
1	2	1.94e-06
1	4	4.64e-05
1	8	4.77e-04

TABLE 3.5 – Itérations 1 et 2 pararéelles pour le modèle réduit en scénario dynamique.

τ_F	τ_G	$\max_{n\geq 0} \epsilon_n^1$
0.1	2	1.39e-02
0.1	4	2.78e-02
0.1	8	5.34e-02
0.5	2	1.09e-02
0.5	4	2.48e-02
0.5	8	5.03e-02
1	2	7.22e-03
1	4	2.10e-02
1	8	4.65e-02

τ_F	τ_G	$\max_{n\geq 0} \epsilon_n^2$
0.1	2	5.19e-05
0.1	4	1.88e-04
0.1	8	5.64e-04
0.5	2	3.17e-05
0.5	4	1.47e-04
0.5	8	4.91e-04
1	2	1.33e-05
1	4	1.03e-04
1	8	4.06e-04

TABLE 3.6 – Itérations 3 et 4 pararéelles pour le modèle réduit en scénario dynamique.

τ_F	τ_G	$\max_{n\geq 0} \epsilon_n^3$
0.1	2	1.91e-07
0.1	4	1.74e-06
0.1	8	1.63e-05
0.5	2	1.01e-07
0.5	4	1.30e-06
0.5	8	1.21e-05
1	2	3.18e-08
1	4	8.35e-07
1	8	9.42e-06

τ_F	τ_G	$\max_{n\geq 0} \epsilon_n^4$
0.1	2	5.71e-09
0.1	4	1.52e-07
0.1	8	2.53e-06
0.5	2	2.69e-09
0.5	4	7.39e-08
0.5	8	2.17e-06
1	2	6.96e-10
1	4	5.50e-08
1	8	1.86e-06

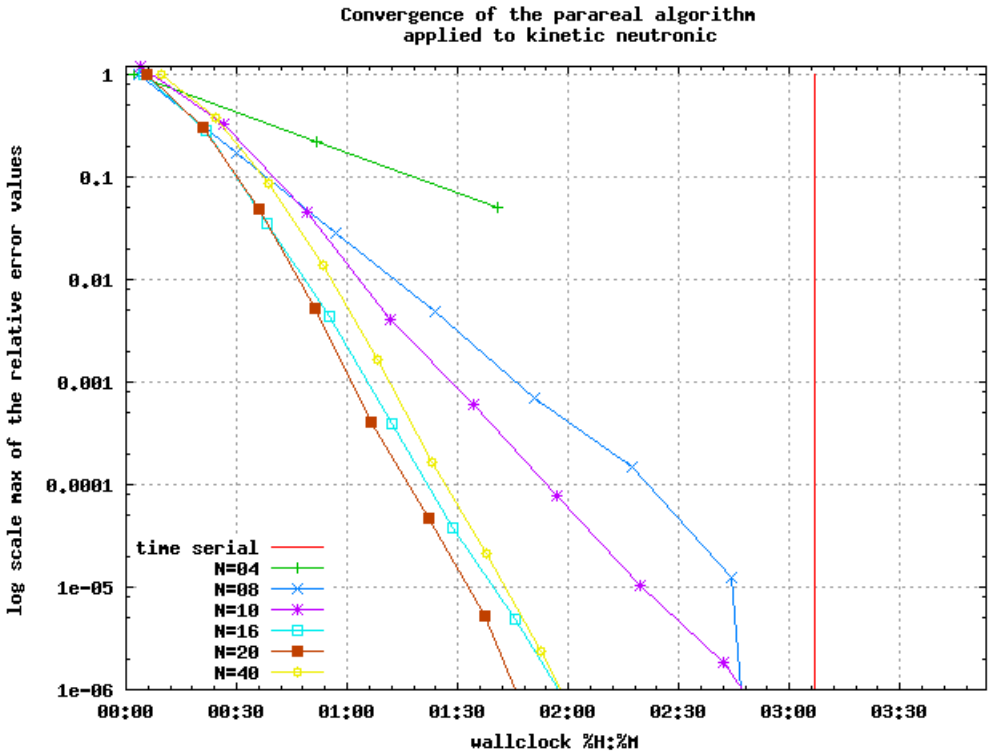


FIGURE 3.21 – Temps réel de simulation CPU (Wallclock sous format h :m) pour les erreurs commises par l'algorithme pararéel en temps appliqué selon le nombre de processeurs (ou sous-intervalles) utilisés à un modèle dans la physique est dégradée pour le solveur grossier. $\tau_F = 10^{-1}, \tau_G = 2$ et $\hat{n} \in \{1, 4, 8, 10, 16, 20, 40\}$.

Chapitre 4

Un algorithme de parallélisation en temps pour le contrôle optimal en résonance magnétique nucléaire

Travail en collaboration avec Dominique Sugny (Physicien) et Julien Salomon

Sommaire

Abstract

The past decade has demonstrated increasing interests in using optimal control based methods within coherent quantum controllable systems. The versatility of such methods has been demonstrated in the field of nuclear magnetic resonance (NMR), where the experiments are coherent with the theory when dealing with the control of spin dynamic. This has led to new design principles as well as powerful new experimental methods within magnetic resonance imaging, liquid-state and solid-state NMR spectroscopy. For this development to continue and expand, it is important to constantly improve the underlying numerical algorithms to provide numerical solutions which are optimally compatible with implementation on current instrumentation.
Addressing such an aim, we present here a smoothing parallel monotonically convergent algorithm for pulse sequence design in magnetic resonance.

Résumé

La décennie passée a vu se développer un intérêt croissant des physiciens pour l'application des méthodes du contrôle optimal aux systèmes quantiques. L'efficacité de telles méthodes a été démontrée dans le domaine de la résonance magnétique nucléaire (RMN), où la théorie et la pratique sont en parfaite cohérence dans le cadre de la dynamique de spins. Ceci a permis de développer de nouveaux principes de conception ainsi que de nouvelles méthodes expérimentales appliquées à l'imagerie par RMN à l'état liquide, et à la spectroscopie.
Nous présentons ici un algorithme monotone parallèlisé en temps pour la conception d'une séquence de pulsations à la résonance magnétique. Cet algorithme inclut une sous-étape de filtrage fréquentiel qui permet d'obtenir des champs magnétiques simplement implémentables dans les expériences.

4.1 Introduction

La théorie du contrôle optimal fournit des outils efficaces pour le contrôle de processus en dynamique quantique.

Pendant la dernière décennie, les méthodes basées sur le contrôle optimal ont été développées pour effectuer de nouvelles expériences en optique spectroscopique [13, 61], pour étudier certains processus quantiques [3, 66] pour améliorer la spectroscopique en RMN [24,67] et l'imagerie par RMN [30,56]. De telles applications ont été non seulement utiles pour les disciplines spécifiques qui profitent de nouvelles procédures efficaces de conception et des méthodes expérimentales améliorées, mais elles ont aussi stimulé des investigations mathématiques pour ce qui concerne la modélisation [2, 17].
Dans leur grande majorité, les méthodes de contrôle optimal appliqué à la résonance magnétique ont utilisé des méthodes de type gradient. Par exemple, en conception d'impulsion, un algorithme (GRAPE) a été présenté par Khaneja et ses collaborateurs [31] pour des applications en RMN. Il a été récemment développé et distribué pour utilisation générale par Nielsen et ses collaborateurs [28,67] à travers le logiciel libre de simulation RMN SIMPSON [7,69].
En association avec des algorithmes de gradient conjugué, cette méthode de GRAPE améliorée [28,31,67,68] s'avère être très efficace. Comme l'on démontré de nombreuses applications en RMN avec une sensibilité expérimentale améliorée. Ces algorithmes sont très robustes par rapport aux variations des paramètres du système de spin. Dans le but de réduire des effets indésirables de processus dissipatifs et également permettre au algorithmes de prendre en compte les faibles radiofréquences des expérimentations, de nombreux développements sont faits et ils ont conduit à des méthodes réduisant les risques du chauffage des échantillons.

Plus récemment, il a été démontré [52] que les algorithmes monotones, qui généralisent l'algorithme de calcul de contrôle optimal de Krotov [33], représentent une alternative intéressante par rapport aux méthodes de gradient [37, 48, 70].

Notre travail repose initialement sur cet algorithme, que l'on applique au contrôle de l'état quantique d'un système quantique isolé. Les propriétés importantes de notre algorithme sont la convergence rapide, la simplicité algorithmique, et l'indépendance de la convergence par rapport à la discrétisation en temps. À l'origine, la méthode que nous prenons comme base dans ce chapitre a été appliquée sur des problèmes d'ingénierie et d'économie [33, 57].

En dépit de leur utilisation croissante, il est évident que les méthodes courantes posent des défis sérieux lors de la réalisation pratique, ce qui implique une prise en compte de certaines contraintes lors des calculs. Ceci s'applique surtout pour les applications mettant en jeu des systèmes de spins de grande taille ou pour des échantillons de poudre en spectroscopie RMN par exemple. En considérant les nombreux contrôles optimaux par séquences d'impulsions proposés jusqu'à présent, il s'avère que beaucoup montrent des oscillations très rapide en phase et en amplitude des champs de contrôle par radiofréquence (voir par exemple [32] et [63]– [67]). Ces oscillations peuvent compliquer l'implémentation sur l'instrumentation disponible qui imposent des limitations sur la vitesse et l'exactitude de la commutation de phase et d'amplitude. En outre, il s'avère que l'algorithme [31] GRAPE montre une très forte dépendance par rapport à la condition initiale de la séquence d'impulsion, ainsi que par rapport à la discrétisation en temps employée (i.e. le nombre

de variables d'impulsions et leurs durées).
D'un autre côté, les algorithmes basés sur la méthode de Krotov rencontrent en général des problèmes d'instabilité numérique. Le choix pertinent des paramètres de l'algorithme s'avère parfois difficile [52].
En résumé, il reste encore un important manque d'intuition et d'expériences. La répétition des calculs d'optimisation avec un ensemble très grand de conditions initiales différentes apparaît encore comme nécessaire. Il existe toujours un fort besoin concernant les applications de la RMN, par exemple dans le formalisme à opérateur de densité en cas de mélange statique et dans le cas d'un formalisme d'un état quantique dans un système isolé.

4.2 Motivations

Nous introduisons brièvement quelques notions sur les systèmes d'information quantique qui, grâce aux outils de contrôle optimal, se manipulent à travers de série d'impulsions radiofréquence.

4.2.1 Spin nucléaire

La résonance magnétique nucléaire est un phénomène physique lié à l'état rotatoire de certains noyaux atomiques exposés à un champ magnétique. Ce phénomène quantique est exploitable par spectroscopie de résonance magnétique nucléaire (Spectroscopie RMN). Le noyau nucléaire est observable par une résonance magnétique s'il présente des propriétés intrinsèques liées à l'existence d'un nombre de spins n_s. Ce nombre strictement positif indique le nombre d'orientations éventuelles du noyau $(2n_s + 1)$ lors de son adressage par un champ magnétique $\vec{B}$. Nous nous intéressons dans ce chapitre à des atomes ayant un nombre de masse impair dont ils présentent un spin $\frac{1}{2}$ (i.e. ils présentent deux états de spin $\pm\frac{1}{2}$ ou encore $|0\rangle$ et $|1\rangle$). Expérimentalement, le spin a été mis en évidence par les physiciens Stern et Gerlach en 1922 par une expérience célèbre. Celle-ci consiste à envoyer des atomes d'argent dans un champ magnétique intense. L'effet observé est une séparation du faisceau atomique en deux. Une interprétation théorique de cette observation est faite par Paul Dirac. Il a conduit à introduire la notion du spin nucléaire afin d'expliquer ces résultats. Cette notion décrit donc une propriété intrinsèque de la matière, qui possède des propriétés analogues à celle du moment cinétique quantique. Le spin est représenté par un opérateur vectoriel Hermitien $\hat{S} := (S_x, S_y, S_z)$ dont les composantes correspondent aux axes du repère d'observation. Elles vérifient les relations de commutations caractéristiques d'un moment cinétique :

$$[S_i, S_j] = ih\varepsilon_{ijk}S_k, \quad \text{et} \quad [S^2, S_i] = 0,$$

où ε_{ijk} est le symbole de Levi-Civita [59] et $S^2 = \sum_{i=x,y,z} S_i^2$.
Pour l'opérateur de spin, il existe une base propre notée $|s, n_s\rangle$, où s et n_s prennent des valeurs d'un entier ou de demi-entier de nombre. n_s peut prendre $2s + 1$ valeurs parmi $[-s, s]$. L'opérateur spin S vérifie :

$$\begin{cases} S^2|s, n_s & = s(s+1)h^2|s, n_s\rangle, \\ S_z|s, n_s & = n_s h|s, n_s\rangle. \end{cases}$$

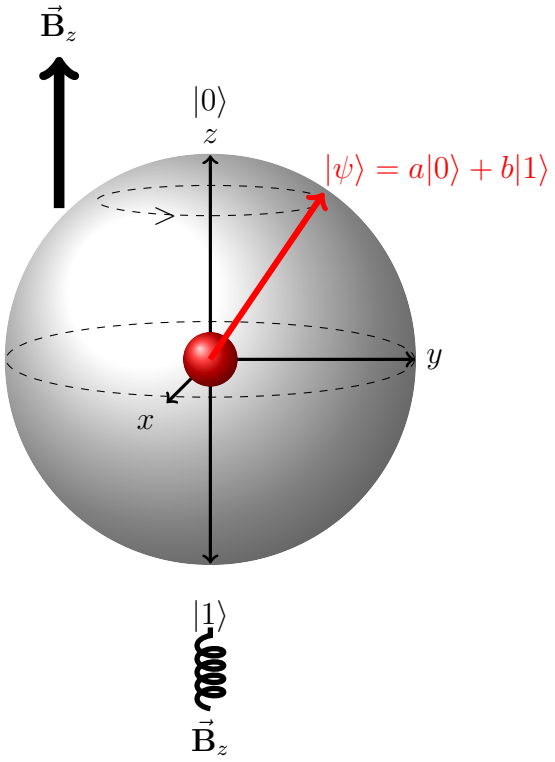


FIGURE 4.1 – Représentation d'un qbit par la sphère de Bloch.

En présence d'un champ magnétique $\vec{\mathbf{B}}_0$, le spin commence à tourner autour de ce champ avec une fréquence donnée ω_0, appelée *fréquence de Larmor* (proportionnelle à l'intensité du champ $|\vec{\mathbf{B}}_0|$). Le phénomène de résonance dépend fortement de l'amplitude. En particulier, les noyaux de spin $\frac{1}{2}$ de chaque molécule deviennent un système à deux niveaux assimilable à des *qbits* (voir ci-après). Le spin, dans ce cas, peut s'aligner soit dans la même direction que le champ magnétique et il aura une énergie minimale $|0\rangle$, soit dans le sens opposé que celui du champ magnétique et aura une énergie maximale $|1\rangle$.

4.2.2 Information quantique

Les méthodes de contrôle optimal en résonance magnétique nucléaire aboutissent au calcul d'un champ de contrôle, sous forme de séquence d'impulsions radiofréquences, permettant un transfert d'état-à-état d'un spin nucléaire. Cette manipulation d'état quantique est en fait une manipulation d'une information (spin) donnée par interaction de la matière avec un champ magnétique. On considère ici un état quantique comme une information quantique. Les ordinateurs quantiques reposent sur ces informations caractérisant des propriétés intrinsèques de la matière. Les calculateurs conventionnels possèdent des mémoires qui utilisent des bits (binaire) généralement désignés par « zéro » et « un ». Tout calcul manipule ces deux quantités binaires. Au contraire, le principe de superposition de l'information quantique permet à un qbit (quantum bytes) de se trouver dans un état générique de la forme :

$$|\psi\rangle = a|0\rangle + b|1\rangle,$$

où a et b sont deux nombres complexes vérifiant $|a|^2 + |b|^2 = 1$. L'ensemble des états accessibles par un qbit de spin $\frac{1}{2}$, forme un espace de Hilbert de dimension 2, dont $\{|0\rangle, |1\rangle\}$ est une base.
Plus généralement, un système à n-qbits dont l'état est noté par $|\psi\rangle$ peut se trouver dans un espace de dimension 2^n dont les état quantiques accessibles sont les produit de superposition des 2^n états formés par le produit des états de base $|0\rangle$ et $|1\rangle$ distincts :

$$|\psi\rangle = \sum_{i=1}^{2^n} \iota_i \underbrace{|\psi_i\rangle \otimes |\psi_i\rangle \otimes \ldots |\psi_i\rangle}_{n-\text{fois}},$$

où $\otimes$ est le produit tensoriel, et les $(\iota_i)_{1 \leq i \leq 2^n}$ sont des scalaires qui satisfont la relation de normalisation $\sum_{i=1}^{2^n} |\iota_i|^2 = 1$. Par exemple, si un qbit est dans une superposition d'état $cos(\theta_a)|0\rangle + sin(\theta_a)|1\rangle$, deux qbits sont (par exemple) dans une superposition d'état

$$\iota_1|00\rangle + \iota_2|01\rangle + \iota_3|10\rangle + \iota_4|11\rangle,$$

avec $|0\rangle \otimes |1\rangle = |01\rangle$ et $\iota_1^2 + \iota_2^2 + \iota_3^2 + \iota_4^2 = 1$.

4.2.3 Contrôle optimal en résonance magnétique nucléaire et manipulation de l'information quantique

Le contrôle optimal en résonance magnétique nucléaire consiste en la manipulation externe de l'état quantique de certaines molécules par l'intermédiaire d'une séquence de pulsations radiofréquences d'un champ magnétique $\vec{\mathbf{B}}$. Ce champs de contrôle tourne dans le plan xy et a la forme suivante :

$$\vec{\mathbf{B}} = w_x(t)\vec{x} + w_y(t)\vec{y} + \mathbf{B}_0\vec{z},$$

$$\begin{cases} w_x(t) = \alpha_x(t)cos(\omega_0 t + \phi), \\ w_y(t) = \alpha_y(t)sin(\omega_0 t + \phi), \\ \mathbf{B}_0 = |\vec{\mathbf{B}}_0|, \end{cases}$$

où ϕ est la phase du champ résonant sur le plan xy, $\alpha_x(t)$ et $\alpha_y(t)$ sont les amplitudes du champs respectivement sur les directions $\vec{x}$ et $\vec{y}$.
Afin de rendre la représentation de $\vec{\mathbf{B}}$ indépendante du temps, il est commode de se placer dans un référentiel tournant autour de l'axe z à la vitesse de la fréquence ω_0. Mais, pour bien modéliser, il faut tenir compte d'une nouvelle composante due au champ d'entraînement.
Dans le cadre du contrôle optimal on se donne un état quantique cible u_{cible} et on fait varier les amplitudes ($\alpha_x(t)$ et $\alpha_y(t)$) du champ $\vec{\mathbf{B}}$ afin d'approcher l'état quantique contrôlé (vérifiant l'équation de Schrödinger ; voir ci-après) de cet état cible. Pour cela, nous considérons un problème de minimisation dont nous définissons une fonctionnelle à deux termes quadratique :

- Un terme mesurant l'écart entre l'état quantique contrôlé par le champ $\vec{\mathbf{B}}$.
- Un terme pénalisant les paramètres de contrôle du champ résonant $\vec{\mathbf{B}}$, telle que l'intensité.

4.3 Le problème de contrôle

L'installation la plus typique pour la conception d'une séquence de pulsations de contrôle en spectroscopie par RMN repose sur une séquence d'impulsions systématiques en radiofréquence. Ces dernières, dans un système de spin donné, ou bien accomplissent le transfert le plus efficace de la concordance ou bien polarisent le système partant d'un état initial de spin donné u_0 à un état cible de spin u_{cible} (souvent désigné comme un transfert d'état-à-état). Les impulsions radiofréquences peuvent également permettre la synthèse d'un opérateur Hamiltonian effectif (ou moyen) [26,27] amplifiant ou supprimant certaines parties des interactions nucléaires internes de rotation pour adapter le Hamiltonien à une évolution sous des interactions désirées.
Dans un premier temps, nous présentons la modélisation mathématique du problème considéré dans ce chapitre. Introduisons tout d'abord quelques notations. Soit Ω l'espace de configurations possibles du système étudié. Le formalisme de la mécanique quantique conduit à décrire l'état du système au temps t à l'aide d'une fonction $u(\cdot, t)$ appartenant à $L^2(\Omega, \mathbb{C})$ (noté simplement $L^2(\Omega)$ dans la suite). Le carré du module de cette fonction $|u(x,t)|^2$ représente la probabilité que l'état du système soit dans la configuration x à l'instant t. En particulier

$$\int_\Omega |u(x,t)|^2 dx = 1. \tag{4.1}$$

On note $\langle \cdot, \cdot \rangle$ le produit hermitien associé à $L^2(\Omega)$. Étant donné un nombre complexe z, on note respectivement $\Re(z)$ et $\Im(z)$ ses parties réelle et imaginaire. On note i le nombre imaginaire pur ($i^2 = -1$).

4.3.1 Fonctionnelle de coût

Étant donné un temps de contrôle $T > 0$, un réel $\alpha > 0$ et un état cible $u_{cible} \in L^2(\Omega)$, on s'intéresse ici à la minimisation de la fonctionnelle de coût :

$$J(c) = \|u(T) - u_{cible}\|_2^2 + \alpha \int_0^T \|c(t)\|_2^2 dt, \tag{4.2}$$

où c est un contrôle à valeurs dans $\mathbb{R}^P$ avec $P \in \mathbb{N}^*$, dont la norme euclidienne est notée $\|\cdot\|_2$, et où u est l'état à contrôler selon l'équation de Schrödinger suivante :

$$i\partial_t u(t) = H(c)u(t). \tag{4.3}$$

L'état initial est noté $u_0 := u(t = 0)$.
Nous décrirons plus précisément l'opérateur hamiltonnien H à la section 4.6.1. Pour l'instant, nous utiliserons seulement le fait qu'il est symétrique, et donc en particulier que iH est hermitien. Une première conséquence de cette caractéristique est la préservation de la norme de u, déjà évoquée en (4.1). En effet, on a :

$$\frac{d}{dt}\|u(\cdot,t)\|_2^2 = 2\Re\left(i\langle u(\cdot,t), Hu(\cdot,t)\rangle\right) = 0.$$

4.3.2 Système d'optimalité

Avant d'écrire le système d'optimalité associé à la minimisation de J, notons qu'en vertu de la propriété (4.1), cette dernière peut se réécrire :

$$J(c) = 2 - 2\Re\left(\langle u(T), u_{cible}\rangle\right) + \alpha \int_0^T \|c(t)\|_2^2 dt.$$

On introduit alors l'adjoint $b(t) \in L^2(\Omega)$, dont la dynamique est la même que celle de l'état $u(t)$:

$$i\partial_t b(t) = H(c)b(t), \tag{4.4}$$

avec la condition finale $b(t = T) = u_{cible}$. Cet adjoint correspond au multiplicateur de Lagrange associé à la contrainte (4.3) dans le Lagrangien :

$$\mathcal{L}(c) = 2 - 2\Re\left(\langle u(T), u_{cible}\rangle\right) + \alpha \int_0^T \|c(t)\|_2^2 dt + 2\Re \int_0^T \langle b(t), \left(\partial_t + iH(c)\right) u(t)\rangle.$$

Le système d'optimalité est alors composé des équations (4.3–4.4), de leurs conditions initiales et finales ainsi que d'une équation supplémentaire portant sur le contrôle optimal :

$$\alpha c(t) - \Im\left(\langle b(t), H'(c)u(t)\rangle\right) = 0, \tag{4.5}$$

où $H'(c)$ désigne la dérivée de H par rapport à c.

4.4 Parallélisation en temps de la résolution

De manière à paralléliser en temps la résolution des équations du sytème d'optimalité, nous reprenons, en l'adaptant, la méthode des cibles intermédiaires présentée au chapitre 2.

4.4.1 Cibles intermédiaires et sous-problèmes

Le fait que l'état et son adjoint aient la même dynamique nous conduit à définir la trajectoire cible

$$\lambda(t) = \frac{T-t}{T} u(t) + \frac{t}{T} b(t). \tag{4.6}$$

Notons que de même qu'au chapitre 2, cette trajectoire ne suit pas la dynamique de l'état et de l'adjoint mais elle vérifie $\Lambda(T) = u_{cible}$.
Étant donné $N \in \mathbb{N}$, décomposons maintenant l'intervalle $[0, T]$ en une union disjointe selon la formule $[0, T] = \cup_{n=0}^{N-1}[t_n, t_{n+1}]$, avec $0 = t_0 < t_1 < ... < t_N = T$. Grâce à la trajectoire cible, nous pouvons considérer sur chaque intervalle de cette partition le sous-problème de minimisation associé à la fonctionnelle J_n suivante :

$$J_n(\lambda, c_n) = \|u_n(t_{n+1}) - \lambda(t_{n+1})\|_2^2 + \alpha_n \int_{t_n}^{t_{n+1}} \|c_n(t)\|_2^2 dt.$$

La dynamique de l'état est ici régie par :

$$i\partial_t u_n(t) = H(c)u_n(t),$$

et l'état initial est donné par $u_n(t = t_n) = \lambda(t_n)$. Le coefficient de pénalisation est défini par $\alpha_n = \frac{t_{n+1}-t_n}{T}$. Comme c'était le cas au chapitre 2, les sous-problèmes ont la même structure que le problème initial.
On peut également montrer un lemme de consistance, analogue au lemme 2.3.2 à la page 17.
Dans la figure 4.2 nous présentons les trajectoires de l'état $u(t)$ (en bas en noir) et de son

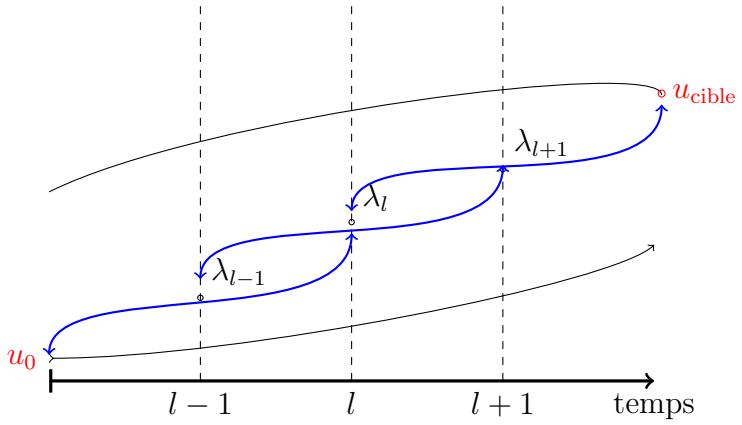


FIGURE 4.2 – Décomposition de l'intervalle de temps et calcul parallèle.

adjoint $b(t)$ (en haut en noir), la série des points $(\lambda_l)_{0\leq l\leq L}$ à la fois cibles et conditions initiales sont calculés selon (4.6). Les courbes rouges représentent les états locaux (direct et adjoint). Ces états sont calculés en parallèle.

Lemme 4.4.1. *Supposons que $c^\star$ soit solution du système d'optimalité (4.3–4.4) de la fonctionnelle J définie par (4.2). Alors, les restrictions de c aux intervalles $[t_n, t_{n+1}]$ sont solutions des systèmes d'optimalités correspondant aux fonctionnelles J_n, avec λ la trajectoire cible calculée avec les états $u^\star$ et $b^\star$ associés à $c^\star$.*

Démonstration. Supposons que la trajectoire cible $\lambda =: \lambda^\star$ soit construite à partir des trajectoires $u^\star$ et $b^\star$ associées à un point critique $c^\star$ de la fonctionnelle J. Montrons alors que les optima $c_n^\star$ des fonctionnelles $J_n(\lambda^\star, c_n^*)$ vérifient la même équation que $c^\star$. On a :

$$\alpha c_n^\star(t) - \Im\left(\langle b_n(t), H'(c_n^\star)u_n(t)\rangle\right) = 0.$$

Or les états u_n et b_n sont calculés à partir de $\lambda^\star$ de telle sorte que :

$$\begin{aligned} u_n &= \frac{T-t_n}{T}u^\star(t) + \frac{t_n}{T}b^\star(t), \\ b_n &= \frac{T-t_{n+1}}{T}u^\star(t) + \frac{t_{n+1}}{T}b^\star(t). \end{aligned}$$

Par conséquent :

$$\Im\langle b_n(t), H'(c_n^\star)u_n(t)\rangle = \frac{t_{n+1}-t_n}{T}\Im\langle b^\star(t), H'(c_n^\star)u^\star(t)\rangle.$$

D'après la définition de α', c_n vérifie donc bien l'équation (4.5), restreinte à l'intervalle $[t_n, t_{n+1}]$.

4.4.2 Structure de l'algorithme

Supposons que l'on dispose d'un algorithme $\mathcal{A}$ de résolution, approximative ou non, des problèmes de type $\min_c J(c)$, avec J de la forme (4.2). L'algorithme itératif de parallélisation de A que nous proposons est alors le suivant. Étant donné, à une étape k, un contrôle c^k, calculer c^{k+1} par les instances :

(1) calculer les trajectoires sur $[0,T]$ de l'état $u^k(t)$ et de l'adjoint $b^k(t)$ associées à c^k par (4.3) et (4.4),

(2) en déduire la trajectoire cible $\lambda^k(t)$ par la formule (4.6),

(3) calculer en appliquant $\mathcal{A}$, sur chacun des sous-intervalles $[t_n, t_{n+1}]$ une approximation de la solution c_n^{k+1} du problème $\min_{c_n} J_n(\lambda^k, c_n)$,

(4) définir c^{k+1} comme la concaténation des contrôles c_n^{k+1}.

4.4.3 Un résultat de convergence

Dans [46], un résultat de convergence est obtenu à partir de cet algorithme. Dans notre cas, ce résultat s'énonce comme suit :

Théorème 4.4.2. *Étant donné un contrôle initial c^0, on suppose que l'algorithme $\mathcal{A}$ est itératif et monotone dans le sens où il fait décroître strictement les valeurs des fonctionnelles J auxquelles il est appliqué tant que l'optimum n'est pas atteint. Alors la suite c^k définie par la procédure de parallélisation précédente converge vers un point critique de la fonctionnelle qui est définie par* (4.2).

4.5 Méthodes numériques

Nous détaillons maintenant les différentes procédures que nous utilisons pour mettre en œuvre la procédure de parallélisation décrite à la section 4.4.2.

4.5.1 Fonctionnelle discrète et calcul numérique de la dynamique

Étant donné $L \in \mathbb{N}^*$, on considère une discrétisation en temps de $[0,T]$ uniforme associée au pas de temps $\Delta T = T/L$. La fonctionnelle discrète $J_{\Delta T}$ associée à (4.2) est définie par :

$$J_{\Delta T}(c) = \|u_L - u_{cible}\|_2^2 + \alpha \Delta T \sum_{\ell=0}^{L-1} \|c_\ell\|_2^2,$$

où les suites $(u_\ell)_{\ell=0}^{L}$ et $(c_\ell)_{\ell=0}^{L-1}$ sont respectivement des discrétisations de $u(t)$ et $c(t)$ aux points $t_\ell = \ell \Delta T$ reliées par la formule de récurrence :

$$u_{\ell+1} = \exp(-i\Delta T H(c_\ell))u_\ell, \tag{4.7}$$

où exp est la fonction exponentielle, ici appliquée à des opérateurs. La condition initiale est donnée par $u_{\ell=0} = u_0$. La formule (4.7) correspond à une méthode de résolution

approximative de (4.3) qui présente la caractéristique intéressante en pratique d'assurer la conservation de la norme. L'adjoint associé à cette discrétisation est donné par une itération analogue, mais rétrograde :

$$b_\ell = \exp(i\Delta T H(c_\ell))b_{\ell+1}, \tag{4.8}$$

muni de la condition finale $b_L = u_{cible}$.

4.5.2 Optimisation par schémas monotones

L'algorithme d'optimisation que nous utilisons est un algorithme dit "monotone" [61, 51]. Nous le décrivons ici sous sa forme discrétisée en temps, qui est celle qui est implémentée en machine. Celui-ci repose sur la remarque suivante. Étant donnés deux contrôles c et c', la variation correspondante des valeurs de la fonctionnelle $J_{\Delta t}$ peut s'écrire sous la forme :

$$\begin{array}{rl} J_{\Delta t}(c') - J_{\Delta t}(c) & = 2\Re \sum_{\ell=0}^{L} \langle b_{\ell+1}, (\exp(-i\Delta T H(c'_\ell)) - \exp(-i\Delta T H(c_\ell)))\, u'_\ell \rangle \\ & \quad +\alpha \Delta T \sum_{\ell=0}^{L-1} \|c'_\ell\|_2^2 - \|c_\ell\|_2^2. \end{array} \tag{4.9}$$

Ici, l'état u' est associé au contrôle c' par l'équation (4.7) tandis que l'état adjoint b correspond à la solution de (4.8) associée au contrôle c. La démonstration de ce résultat repose sur le calcul suivant :

$$\begin{aligned} J_{\Delta t}(c') - J_{\Delta t}(c) &= -2\Re\left(\langle u'_L - u_L, u_{cible}\rangle\right) + \alpha\Delta T \sum_{\ell=0}^{L-1} \|c'_\ell\|_2^2 - \|c_\ell\|_2^2 \\ &= -2\Re\left(\langle u'_L - u_L, b_L\rangle\right) + \alpha\Delta T \sum_{\ell=0}^{L-1} \|c'_\ell\|_2^2 - \|c_\ell\|_2^2, \\ &= -2\Re\left(\Delta T \sum_{\ell=0}^{L-1} \langle u'_{\ell+1} - u_{\ell+1}, b_{\ell+1}\rangle - \langle u'_\ell - u_\ell, b_\ell\rangle\right) \\ &\quad +\alpha\Delta T \sum_{\ell=0}^{L-1} \|c'_\ell\|_2^2 - \|c_\ell\|_2^2, \\ &= -2\Re\left(\Delta T \sum_{\ell=0}^{L-1} \langle u'_{\ell+1} - u_{\ell+1}, b_{\ell+1}\rangle - \langle u'_\ell - u_\ell, \exp(i\Delta T H(c_\ell) b_{\ell+1}\rangle\right) \\ &\quad +\alpha\Delta T \sum_{\ell=0}^{L-1} \|c'_\ell\|_2^2 - \|c_\ell\|_2^2, \\ &= -2\Re\left(\Delta T \sum_{\ell=0}^{L-1} \langle (\exp(-i\Delta T H(c'_\ell) - \exp(-i\Delta T H(c_\ell)) u'_\ell, b_{\ell+1}\rangle\right) \\ &\quad +\alpha\Delta T \sum_{\ell=0}^{L-1} \|c'_\ell\|_2^2 - \|c_\ell\|_2^2. \end{aligned}$$

Dans ce cadre, l'algorithme monotone est donné par les instances suivantes. Étant donné un contrôle c^k à une étape k, le calcul de c^{k+1} est achevé par les étapes suivantes :

1. calculer itérativement selon la formule (4.8) la suite $(b_\ell^k)_{\ell=0}^L$,
2. calculer itérativement la suite $(u_\ell^{k+1})_{\ell=0}^L$ de la manière suivante. À chaque pas de temps t_ℓ un état u_ℓ^{k+1} étant donné, calculer c_ℓ^{k+1} et $u_{\ell+1}^{k+1}$ par les instances :
   (a) calculer
   $$c_\ell^{k+1} = \text{argmin}_x(\varphi_\ell(x)),$$
   avec
   $$\begin{aligned}\varphi_\ell(x) = \; & -2\Re\big(\Delta T\langle b_{\ell+1}^k, \big(\exp(-i\Delta T H(x)) - \exp(-i\Delta T H(c_\ell^k))\big)\, u_\ell^{k+1}\rangle\big) \\ & +\alpha\Delta T(\|x\|_2^2 - \|c_\ell^k\|_2^2),\end{aligned}$$
   (b) calculer $u_{\ell+1}^{k+1}$ par la formule (4.7) en utilisant le contrôle c_ℓ^{k+1}.

4.5.3 Filtrage

Pour obtenir des contrôles réalisables en pratique, il convient dans certains cas d'imposer des contraintes sur le spectre fréquentiel des contrôles calculés. La méthode que nous suivons pour accomplir cet objectif est celle proposée dans [35]. Elle consiste à insérer après l'étape (4) de l'algorithme de parallélisation 5 la sous-boucle :

$$c^{k+1} = \theta^k \mathcal{F}(\tilde{c}^{k+1}) + (1-\theta^k)\tilde{c}^{k+1},$$

où $\tilde{c}^{k+1}$ est le champ calculé à l'étape 4 de l'algorithme 5 de parallélisation, $\mathcal{F}(\tilde{c}^{k+1})$ correspond à un filtrage séquentiel de ce même champ, θ est déterminé itérativement de telle sorte que θ soit aussi proche possible de la valeur $\theta^k = 1$ tout en assurant la monotonie de l'algorithme, i.e. $J(c^{k+1}) \leq J(c^k)$. Notons qu'en utilisant un schéma monotone tel que celui décrit à la section 4.5.2, la monotonie est garantie pour $\theta^k = 0$, c'est-à-dire $c^{k+1} = \tilde{c}^{k+1}$. Dans ce cas, on peut donc initialiser θ^k à la valeur 1, puis multiplier cette valeur par un réel $\nu \in]0,1[$, jusqu'à ce que la monotonie soit assurée.

Remarque 4.5.1. *L'opération de filtrage du contrôle concaténé fait l'objet d'une boucle while ayant pour test d'arrêt une vérification simple de la monotonie de la fonctionnelle de coût J. Cette opération est malheureusement séquentielle, les autres méthodes de filtrage parallèle sont en cours de développement. Néanmoins, une parallélisation de la boucle while est possible. En effet, supposons que nous allouons les tâches parallèles à L processeurs. Au niveau de la boucle de filtrage nous allouons à chaque processeur p le filtrage et le test de la monotonie avec la pondération $\theta^k = \theta^{k-1}/(2p)$ où p est le processeur numéro p. Une fois cette étape finie, les valeurs des fonctionnelles $J(c^{k+1})$ seront comparées au niveau du processeur maître qui prendra la meilleure. Dans le cas où le filtrage à cette étape ne suffit pas pour trouver une monotonie et sortir de la boucle while, le calcul reprend avec $\theta^k = \theta^{k-1}/(2p[2L])$ où $[2L]$ est modulo $2L$.*

4.5.4 Algorithme parallèle pour le contrôle optimal en RMN

Maintenant que les ingrédients de notre travail ont été exposé, nous somme en mesure de décrire l'algorithme 5 que nous proposons pour la résolution en parallèle (mode MPI) de la résonance magnétique.
Nous utilisons dans notre implémentation deux routines de Matlab que nous notons comme suit :

– Expm : désigne l'exponentielle matricielle, voir ,pour plus de détails sur cette routine, `http://www.mathworks.fr/help/techdoc/ref/expm.html`.
– Fminnuc : désigne une fonction calculant le minimum d'une fonction donnée, tel que $\min_x f(x)$. Voir, pour plus de détails sur cette routine, `http://www.mathworks.fr/help/toolbox/optim/ug/fminunc.html`.
– Pour la définition des mots clés relatifs à l'utilisation de MPI (**Mpi_Send, Mpi_Recv, Mpi_Broadcast**) nous renvoyons le lecteur à la page 20.

Remarque 4.5.2. *Dans l'algorithme 5, nous considérons N sous-intervalles de temps de contrôle, également N processeurs esclaves et un processeur maître. L'algorithme fait L petit pas de temps égal à τ (i.e. $\tau = \frac{T}{L}$) pour parcourir l'intervalle global. Le calcul qui est effectué en parallèle fait n_p pas de temps local (e.g. $n_p = \frac{L}{N}$). Afin de bien indexer les variables qui dépendent du temps (e.g. t_ℓ)*

$$0 < t_0 < \ldots < t_\ell < \ldots < t_L = T,$$

nous notons $u_\ell :\approx u(t_\ell)$. Egalement, $u_{\ell+j}$ (respectivement $b_{\ell+j}$) la solution de l'état u (respectivement de l'adjoint b) à l'instant $t_\ell + j\tau$.

L'algorithme est présenté ci-dessous.

Remarque 4.5.3. *Au niveau de l'algorithme 5 nous interprétons la monotonie de notre algorithme par une minimisation de $-J$ ce qui reviens à une maximisation de la nouvelle fonctionnelle (aussi notée J dans le cadre de cet algorithme).*
Nous notons également qu'au niveau du calcul séquentiel de l'algorithme 5 à partir de la deuxième itération, nous ne calculons plus toute la trajectoire pour les états direct et adjoint, en effet, l'algorithme nécessite désormais les valeurs des états en certain nombre de points t_n dont le propagateur direct P (respectivement l'adjoint P^*) est mis en place pour assurer cette tâche.

4.6 Résultats numériques

Présentons enfin un test numérique qui montre l'efficacité de notre algorithme.

4.6.1 Le problème physique

Nous nous intéressons pour l'expérience numérique à un système quantique à *cinq-qbits* (quantum-bytes) de spin $\frac{1}{2}$ des atomes N, C, H, F et P de la molécule d'*Acide phosphorique, P-(fluorométhyl) -, ester éthylique* $C5H12FO3P$. La dynamique de spins est régie par l'opérateur Hamiltonien :

$$H = H_0 + H_1, \tag{4.10}$$

où H_0 est le Hamiltonien libre et H_1 est l'opérateur régissant l'interaction des spins en présence de champs magnétiques transverses. L'adressage des champs magnétiques peut

Algorithme 5 : Parallel optimal control in RMN spectroscopy

Data : $L, N(\#\, slave\, proc), n_p = L/N, (c_\ell^0)_{\ell=0}^{L-1}$, Tolerence ϵ; $u_0^0, b_L^0, \lambda_0^0 \leftarrow u_0^0$;
$k \longleftarrow 0$;
repeat
  if *master processor* **then**
    if $k = 0$ **then**
      for $\ell = 1$ **to** L **do** (forward computation) u_ℓ^k according to (4.7);
      for $\ell = 1$ **to** L **do** (backward computation) $b_{L-\ell}^k$ according to (4.8);
    else
      for $n = 1$ **to** N **do**
        $u_{n.n_p}^k = P_n u_{(n-1).n_p}^k$; $b_{(N-n).n_p}^k = P_n^* b_{(N-n+1).n_p}^k$;//Where P^* is the adjoint
      end
    end
    for $n = 1$ **to** N **do**
      $\lambda_n^k \leftarrow \frac{N-n}{N} u_{n.n_p}^k + \frac{n}{N} b_{n.n_p}^k$;
      Mpi_Send $(\lambda_{n-1}^k, \lambda_n^k, \left(c_{(n-1).n_p+j}^k\right)_{j=0}^{n_p-1}, \left(b_{(n-1).n_p+j}^k\right)_{j=1}^{n_p}, processor(n))$;
    end
  else
    forall $slave\, processor(n) / \, n \in \{1, \ldots, N\}$ **do**
      Mpi_Recv $(\lambda_{n-1}^k, \lambda_n^k, \left(c_{(n-1).n_p+j}^k\right)_{j=0}^{n_p-1}, \left(b_{(n-1).n_p+j}^k\right)_{j=1}^{n_p}, master\, processor)$;
      $\tilde{u}_0^{k+1} \leftarrow \lambda_{n-1}^k$; $P_n \leftarrow \mathbb{I}_d$;
      for $j = 0$ **to** $n_p - 1$ **do**

$$\begin{aligned}
\varphi_j(x) \quad &\leftarrow \quad -2\Re\Bigg(\Delta T\Big\langle b_{(n-1).n_p+j+1}^k, \big(\mathrm{Expm}(-i\Delta T H(x)) \\
&\qquad\qquad -\mathrm{Expm}(-i\Delta T H(c_{(n-1).n_p+j}^k))\big)\tilde{u}_j^{k+1}\Big\rangle\Bigg) \\
&\qquad +\alpha\Delta T(\|x\|_2^2 - \|c_{(n-1).n_p+j}^k\|_2^2); \\
\tilde{c}_{(n-1).n_p+j}^{k+1} \quad &\leftarrow \quad \textsc{Fminnuc}(\varphi_j(x)); \\
P_n \quad &\leftarrow \quad P_n \times \mathrm{Expm}\Big(-i\Delta T\big(\tilde{c}_{(n-1).n_p+j}^{k+1}\big)\Big) \\
\tilde{u}_{j+1}^{k+1} \quad &\leftarrow \quad \mathrm{Expm}\Big(-i\Delta T H\big(\tilde{c}_{(n-1).n_p+j}^{k+1}\big)\Big)\tilde{u}_j^{k+1};
\end{aligned}$$

      end
      Mpi_Send $\left(\left(\tilde{c}_{(n-1).n_p+j}^k\right)_{j=1}^{n_p}, P_n, master\, processor\right)$
    end
  end
  if *master processor* **then**
    for $n = 1$ **to** N **do** **Mpi_Recv** $\left(\left(\tilde{c}_{(n-1).n_p+j}^k\right)_{j=1}^{n_p}, P_n, processor(n)\right)$;
    $\tilde{c}^{k+1} := \left(\tilde{c}_\ell^{k+1}\right)_{\ell=1}^{L}$; $\theta \leftarrow 1$;
    repeat $c^{k+1} \leftarrow \theta\mathcal{F}(\tilde{c}^{k+1}) + (1-\theta)\tilde{c}^{k+1}$; $\theta \leftarrow \frac{\theta}{2}$; **until** $J(c^{k+1}) > J(c^k)$
    $Err \leftarrow |J(c^{k+1}) - J(c^k)|/|J(c^k)|$;
  end
  $k \leftarrow k + 1$; **Mpi_Broadcast** $(master\, processor, Err)$;
until $Err \leq \epsilon$;

s'effectuer individuellement. Le terme H_0 s'écrit comme suit :

$$H_0 = 2\pi\big(\quad \Upsilon_{CH}\sigma_z^C\sigma_z^H, \Upsilon_{CF}\sigma_z^C\sigma_z^F, \\ \Upsilon_{CN}\sigma_z^C\sigma_z^N, \Upsilon_{HF}\sigma_z^H\sigma_z^F, \\ \Upsilon_{HP}\sigma_z^H\sigma_z^P, \Upsilon_{FP}\sigma_z^F\sigma_z^P, \\ \Upsilon_{FN}\sigma_z^F\sigma_z^N, \Upsilon_{NP}\sigma_z^N\sigma_z^P \quad \big),$$

avec $\Upsilon_{CH} = 4.1\text{Hz}$, $\Upsilon_{CF} = 19,3\text{Hz}$, $\Upsilon_{CN} = 18,1\text{Hz}$, $\Upsilon_{HF} = 46\text{Hz}$, $\Upsilon_{HP} = 1\text{Hz}$, $\Upsilon_{FP} = 73,1\text{Hz}$, $\Upsilon_{FN} = 2\text{Hz}$ et $\Upsilon_{NP} = 1.9\text{Hz}$. Le terme d'interaction H_1 est donné par la somme de cinq termes tels que :

$$H_1 := H_1^H + H_1^P + H_1^N + H_1^C + H_1^F,$$

où

$$H_1^\varrho = w_x^\varrho(t)\vec{x}^\varrho + w_y^\varrho(t)\vec{y}^\varrho, \qquad \text{avec } \varrho \in \{H, P, C, N, F\},$$

avec w_x^ϱ et w_y^ϱ les deux variables de contrôle correspondantes au spin ϱ.

4.6.2 Mise en œuvre de l'algorithme

Le code numérique a été implémenté avec une version parallèle (MPI) de Matlab MATLABMPI [29]. Les tests numériques sont lancés sur une supermachine du laboratoire Jacques-Louis Lions, elle possède 24 processeurs qui se partagent la mémoire.
En pratique, l'étape 2a du schéma monotone est réalisée par une fonction d'optimisation standard. Plus précisément, nous utilisons la fonction `fminunc` de la toolbox 'optimisation' de Matlab [1].
L'implémentation des formulations discrètes des équations (4.3) et (4.4) a été réalisée par splitting d'opérateur. Cette méthode donne lieu à l'approximation :

$$\exp(-i\Delta T H(c_\ell)) \approx \exp(-i\frac{\Delta T}{2}H_0)\exp(-i\Delta T H_1)\exp(-i\frac{\Delta T}{2}H_0),$$

où H est donné par (4.10).

4.6.3 Apport de la méthode

L'implémentation de l'algorithme 5 sur une machine haute performance a montré une accélération remarquable de temps de traitement. À la Figure 4.3 nous présentons la maximisation de la fonctionnelle de coût J par rapport au temps de calcul wallclock de la machine. Ces résultats d'ouverture montrent un grand intérêt de la parallélisation pour ce type de problème de contrôle optimal en RMN. Nous remarquons qu'avec un calcul séquentiel, la réalisation de l'étape 2a de l'algorithme 5 prend plus de temps que dans le cas où l'on utilise une parallélisation. Nous suspectons que cela soit dû au caractère préconditionneur de la parallélisation par cibles intermédiaires.
Les champs de contrôle produits par la parallélisation sont en outre *lisses* (voir Figure 4.5). Nous ne savons pour l'instant pas interpréter ce phénomène, qui ne présente pas de hautes fréquences pouvant nuire à la maximisation du coût.

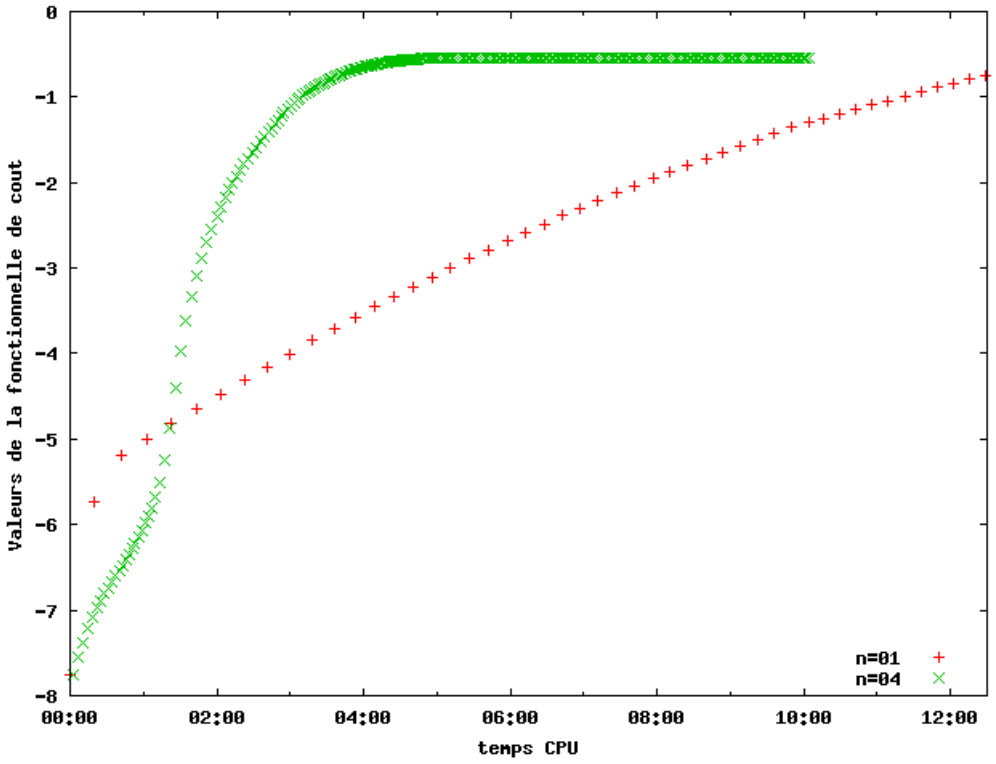


FIGURE 4.3 – Accélération de la résolution en temps Wallclock, interprétation par maximisation de la fonctionnelle de coût $-J$ avec 4 processeurs en parallèle (courbe verte) et en séquentiel (courbe rouge) 1 processeur.

Remarque 4.6.1. *Les champs sont obtenus avec une procédure de filtrage fort. Les champs calculés séquentiellement (sans parallélisation) ne permettent pas d'assurer la monotonie de la fonctionnelle de coût J après filtrage. Ces champs contiennent donc dans ce cas des plus hautes fréquences.*

4.7 Conclusion

Nous avons présenté un algorithme parallèle pour produire des champs magnétiques efficaces en résonance magnétique nucléaire. Cet algorithme, dont nous avons prouvé la monotonie, est basé sur une méthode de Krotov. La méthode peut certainement bénéficier d'une optimisation algorithmique encore plus importante, ce qui permettrait une réduction de la complexité. La recherche dans cette direction est en cours de développement. Nous développons également un autre schéma numérique de discrétisation basé sur le schéma de Crank-Nicolson qui, comme le schéma présent préserve la norme L^2 des états, mais dont la complexité numérique est beaucoup plus faible.
La méthode présentée, ainsi que ses développements à venir devraient permettre de traiter efficacement des problèmes en RMN à l'état solide et à l'état liquide, la RMI et les développements et manipulations d'information pour des ordinateurs quantiques. L'intérêt de tels modèles est en effet que les champs de contrôle obtenus par le calcul sont directement réalisables expérimentalement.

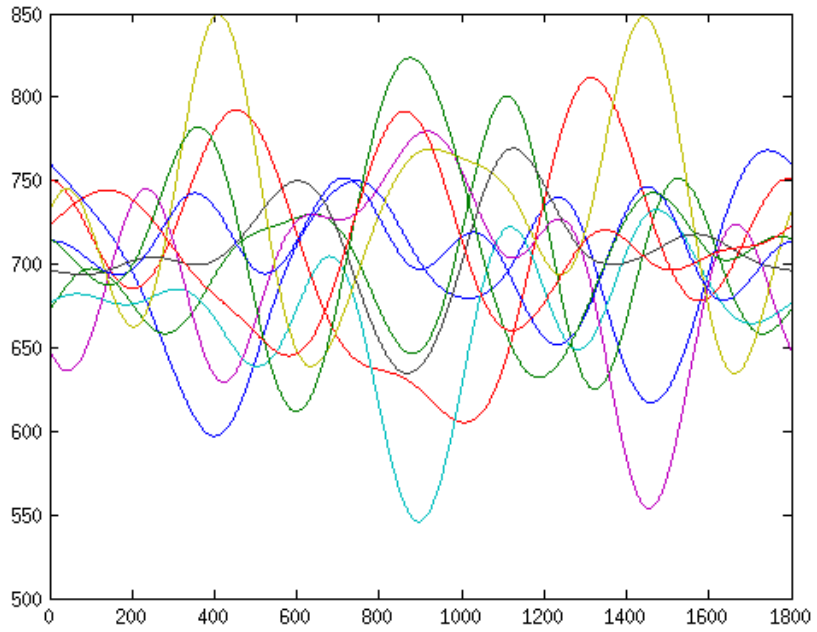


FIGURE 4.4 – Exemple de champs de contrôle parallèle en RMN obtenu avec un filtrage fort.

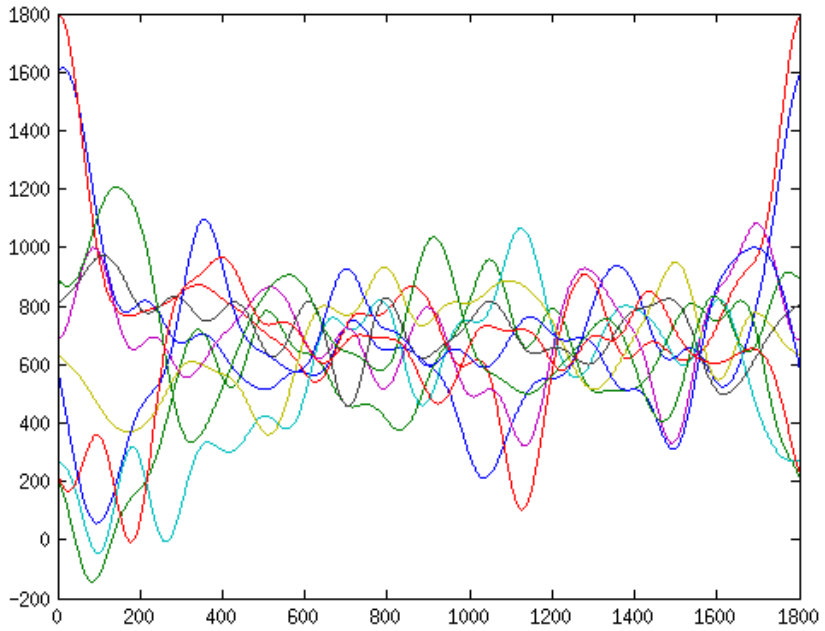


FIGURE 4.5 – Exemple de champs de contrôle séquentiel (sans parallélisation) après filtrage fort.

Bibliographie

[1] Matlab, logiciels de calcul scientifique et technique. version R2010b. USA,`http://www.mathworks.fr/company/?s_cid=global_nav`.

[2] Claudio Altafini. Controllability of quantum mechanical systems by root space decomposition of [fraktur su](n). *Journal of Mathematical Physics*, 43(5) :2051–2062, 2002.

[3] Claudio Altafini. Feedback control of spin systems. *Quantum Information Processing*, 6 :9–36, 2007. 10.1007/s11128-006-0038-x.

[4] Jean-Jacques Lautard Anne-Marie Baudron. Minos : a spn solver for core calculation. *Nuclear Science and Engineering*, 7(2) :250–263, 2007.

[5] Greg Astfalk, editor. *Applications on Advanced Architecture Computers*. Society for Industrial and Applied Mathematics, Philadephia, PA, 1996.

[6] L. Baffico, S. Bernard, Y. Maday, G. Turinici, and G. Zérah. Parallel-in-time molecular-dynamics simulations. *Phys. Rev. E*, 66 :057701, Nov 2002.

[7] Mads Bak, Jimmy T Rasmussen, and Niels Chr Nielsen. Simpson : A general simulation program for solid-state nmr spectroscopy. *Journal of Magnetic Resonance*, 147(2) :296 – 330, 2000.

[8] Guillaume Bal. On the convergence and the stability of the parareal algorithm to solve partial differential equations. In *Domain decomposition methods in science and engineering*, volume 40 of *Lect. Notes Comput. Sci. Eng.*, pages 425–432. Springer, Berlin, 2005.

[9] Guillaume Bal and Yvon Maday. A "parareal" time discretization for non-linear PDE's with application to the pricing of an American put. In *Recent developments in domain decomposition methods (Zürich, 2001)*, volume 23 of *Lect. Notes Comput. Sci. Eng.*, pages 189–202. Springer, Berlin, 2002.

[10] A. Bellen. Parallelism across the steps for difference and differential equations. In *Numerical methods for ordinary differential equations (L'Aquila, 1987)*, volume 1386 of *Lecture Notes in Math.*, pages 22–35. Springer, Berlin, 1989.

[11] Alfredo Bellen and Marino Zennaro. Parallel algorithms for initial value problems for difference and differential equations. *J. Comput. Appl. Math.*, 25(3) :341–350, 1989.

[12] Adel Blouza, Laurent Boudin, and Sidi Mahmoud Kaber. Parallel in time algorithms with reduction methods for solving chemical kinetics. *Commun. Appl. Math. Comput. Sci.*, 5(2) :241–263, 2010.

[13] J. Broeckhove and L. Lathouwers. Quantum molecular dynamics and angular momentum projection. In *Numerical grid methods and their application to Schrödinger's*

equation (Corte, 1992), volume 412 of *NATO Adv. Sci. Inst. Ser. C Math. Phys. Sci.*, pages 49–56. Kluwer Acad. Publ., Dordrecht, 1993.

[14] Kevin Burrage. *Parallel and sequential methods for ordinary differential equations*. Numerical Mathematics and Scientific Computation. The Clarendon Press Oxford University Press, New York, 1995. Oxford Science Publications.

[15] Steve CHAUVET. *Méthode multi-èchelle pour la résolution des équations de la cinétique neutronique*. PhD thesis, Université de Nantes, 2008.

[16] L. J. Curtis. A diagrammatic mnemonic for calculation of cascading level populations. 36(12) :1123–1125, 1968.

[17] Domenico D'Alessandro. *Introduction to quantum control and dynamics*. Chapman & Hall/CRC Applied Mathematics and Nonlinear Science Series. Chapman & Hall/CRC, Boca Raton, FL, 2008.

[18] Daoud S. Daoud. Stability of the parareal time discretization for parabolic inverse problems. In *Domain decomposition methods in science and engineering XVI*, volume 55 of *Lect. Notes Comput. Sci. Eng.*, pages 275–282. Springer, Berlin, 2007.

[19] Charbel Farhat, Julien Cortial, Climène Dastillung, and Henri Bavestrello. Time-parallel implicit integrators for the near-real-time prediction of linear structural dynamic responses. *Internat. J. Numer. Methods Engrg.*, 67(5) :697–724, 2006.

[20] Martin J. Gander and Ernst Hairer. Nonlinear convergence analysis for the parareal algorithm. In *Domain decomposition methods in science and engineering XVII*, volume 60 of *Lect. Notes Comput. Sci. Eng.*, pages 45–56. Springer, Berlin, 2008.

[21] Martin J. Gander and Stefan Vandewalle. Analysis of the parareal time-parallel time-integration method. *SIAM J. Sci. Comput.*, 29(2) :556–578 (electronic), 2007.

[22] C. W. Gear. Parallel methods for ordinary differential equations. *Calcolo*, 25(1-2) :1–20 (1989), 1988.

[23] P. Gervasio, J.-L. Lions, and A. Quarteroni. Heterogeneous coupling by virtual control methods. *Numer. Math.*, 90(2) :241–264, 2001.

[24] S. J. Glaser, T. Schulte-Herbruggen, M. Sieveking, O. Schedletzky, N. C. Nielsen, O. W. SØ rensen, and C. Griesinger. Unitary control in quantum ensembles : Maximizing signal intensity in coherent spectroscopy. *Science*, 280(5362) :421–424, 1998.

[25] Wolfgang Hackbusch. Parabolic multigrid methods. In *Computing methods in applied sciences and engineering, VI (Versailles, 1983)*, pages 189–197. North-Holland, Amsterdam, 1984.

[26] U. Haeberlen and J. S. Waugh. Coherent averaging effects in magnetic resonance. *Phys. Rev.*, 175 :453–467, Nov 1968.

[27] M. Hohwy and N. C. Nielsen. Systematic design and evaluation of multiple-pulse experiments in nuclear magnetic resonance spectroscopy using a semi-continuous baker ?campbell ?hausdorff expansion. 109(10) :3780–3791, 1998.

[28] Cindie T. Kehlet, Astrid C. Sivertsen, Morten Bjerring, Timo O. Reiss, Navin Khaneja, Steffen J. Glaser, and Niels Chr. Nielsen. Improving solid-state nmr dipolar recoupling by optimal control. *Journal of the American Chemical Society*, 126(33) :10202–10203, 2004. PMID : 15315406.

[29] Jeremy Kepner. *Parallel MATLAB for Multicore and Multinode Computers*. Society for Industrial and Applied Mathematics, Philadephia, PA, 2009.

[30] Navin Khaneja. Switched control of electron nuclear spin systems. *Phys. Rev. A*, 76 :032326, Sep 2007.

[31] Navin Khaneja, Timo Reiss, Cindie Kehlet, Uggen Thomas Schulte-Herbr, and Steffen J. Glaser. Optimal control of coupled spin dynamics : design of nmr pulse sequences by gradient ascent algorithms. *Journal of Magnetic Resonance*, 172(2) :296 – 305, 2005.

[32] Kyryl Kobzar, Burkhard Luy, Navin Khaneja, and Steffen J. Glaser. Pattern pulses : design of arbitrary excitation profiles as a function of pulse amplitude and offset. *Journal of Magnetic Resonance*, 173(2) :229 – 235, 2005.

[33] Vadim F. Krotov. *Global methods in optimal control theory*, volume 195 of *Monographs and Textbooks in Pure and Applied Mathematics*. Marcel Dekker Inc., New York, 1996.

[34] Franz N. D. Kurie, J. R. Richardson, and H. C. Paxton. The radiations emitted from artificially produced radioactive substances. i. the upper limits and shapes of the β-ray spectra from several elements. *Phys. Rev.*, 49 :368–381, Mar 1936.

[35] M. Lapert, R. Tehini, G. Turinici, and D. Sugny. Monotonically convergent optimal control theory of quantum systems with spectral constraints on the control field. *Phys. Rev. A*, 79 :063411, Jun 2009.

[36] A.E. ; Sangiovanni-Vincentelli A.L Lelarasmee, E. ; Ruehli. The waveform relaxation method for time-domain analysis of large scale integrated circuits. In *, VI (Versailles, 1983)*, pages 189–197. North-Holland, Amsterdam, 1982.

[37] J. C. Light. Discrete variable representation in quantum dynamics. In J. Broeckhove and L. Lathouwers, editors, *Time–Dependent Quantum Molecular Dynamics*, pages 185–199, New York, 1992. Plenum Press.

[38] J. L. Lions. Virtual and effective control for distributed systems and decomposition of everything. *J. Anal. Math.*, 80 :257–297, 2000.

[39] J. L. Lions. Decomposition of energy space and virtual control for parabolic systems. In *Domain decomposition methods in sciences and engineering (Chiba, 1999)*, pages 41–53 (electronic). DDM.org, Augsburg, 2001.

[40] Jacques-Louis Lions. Least regret control, virtual control and decomposition methods. *M2AN Math. Model. Numer. Anal.*, 34(2) :409–418, 2000. Special issue for R. Temam's 60th birthday.

[41] Jacques-Louis Lions, Yvon Maday, and Gabriel Turinici. Résolution d'EDP par un schéma en temps "pararéel". *C. R. Acad. Sci. Paris Sér. I Math.*, 332(7) :661–668, 2001.

[42] Jacques-Louis Lions and Olivier Pironneau. Virtual control, replicas and decomposition of operators. *C. R. Acad. Sci. Paris Sér. I Math.*, 330(1) :47–54, 2000.

[43] Yvon Maday and Guettat Rim. *Méthode de parallélisation en temps : Application aux méthodes de décomposition de domaine*. PhD thesis.

[44] Yvon Maday and Einar M. Rønquist. Parallelization in time through tensor-product space-time solvers. *C. R. Math. Acad. Sci. Paris*, 346(1-2) :113–118, 2008.

[45] Yvon Maday, Julien Salomon, and Gabriel Turinici. Monotonic parareal control for quantum systems. *SIAM J. Numer. Anal.*, 45(6) :2468–2482 (electronic), 2007.

[46] Yvon Maday, Julien Salomon, and Gabriel Turinici. Monotonic parareal control for quantum systems. *SIAM Journal on Numerical Analysis*, 45(6) :2468–2482, 2007.

[47] Yvon Maday and Gabriel Turinici. A parareal in time procedure for the control of partial differential equations. *C. R. Math. Acad. Sci. Paris*, 335(4) :387–392, 2002.

[48] Yvon Maday and Gabriel Turinici. New formulations of monotonically convergent quantum control algorithms. *The Journal of Chemical Physics*, 118(18) :8191–8196, 2003.

[49] Yvon Maday and Gabriel Turinici. The parareal in time iterative solver : a further direction to parallel implementation. In *Domain decomposition methods in science and engineering*, volume 40 of *Lect. Notes Comput. Sci. Eng.*, pages 441–448. Springer, Berlin, 2005.

[50] Tarek P. Mathew, Marcus Sarkis, and Christian E. Schaerer. Analysis of block parareal preconditioners for parabolic optimal control problems. *SIAM J. Sci. Comput.*, 32(3) :1180–1200, 2010.

[51] Ivan I. Maximov, Julien Salomon, Gabriel Turinici, and Niels Chr. Nielsen. A smoothing monotonic convergent optimal control algorithm for nuclear magnetic resonance pulse sequence design. *The Journal of Chemical Physics*, 132(8) :084107, 2010.

[52] Ivan I. Maximov, Zdeněk Tošner, and Niels Chr. Nielsen. Optimal control design of nmr and dynamic nuclear polarization experiments using monotonically convergent algorithms. *The Journal of Chemical Physics*, 128(18) :184505, 2008.

[53] Michael L. Minion. A hybrid parareal spectral deferred corrections method. *Commun. Appl. Math. Comput. Sci.*, 5(2) :265–301, 2010.

[54] F. Hecht O. Pironneau and K. Ohtsuka. Freefem++-mpi. In *, Université Pierre et Marie Curie (Paris,)*. Free-Soft, Laboratoire Jacques-Louis Lions.

[55] Olivier Pironneau, Frédéric Hecht, and Jacques Morice. freeFEM++, www.freefem.org/.

[56] N. Pomplun, B. Heitmann, N. Khaneja, and S. J. Glaser. Optimization of electron, nuclear polarization transfer. *Applied Magnetic Resonance*, 34 :331–346, 2008. 10.1007/s00723-008-0124-6.

[57] L. S. Pontryagin, V. G. Boltyanskii, R. V. Gamkrelidze, and E. F. Mishchenko. *The mathematical theory of optimal processes*. Translated from the Russian by K. N. Trirogoff ; edited by L.W. Neustadt. Interscience Publishers John Wiley & Sons, Inc. New York-London, 1962.

[58] P. Reuss. *Précis de neutronique*. Génie atomique. EDP SCIENCES, 2003.

[59] M. M. G. Ricci and T. Levi-Civita. Méthodes de calcul différentiel absolu et leurs applications. *Mathematische Annalen*, 54 :125–201, 1900. 10.1007/BF01454201.

[60] Youcef Saad and Martin H. Schultz. GMRES : a generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Statist. Comput.*, 7(3) :856–869, 1986.

[61] Julien Salomon and Gabriel Turinici. On the relationship between the local tracking procedures and monotonic schemes in quantum optimal control. *The Journal of Chemical Physics*, 124(7) :074102, 2006.

[62] Marcus Sarkis, Christian E. Schaerer, and Tarek Mathew. Block diagonal parareal preconditioner for parabolic optimal control problems. In *Domain decomposition methods in science and engineering XVII*, volume 60 of *Lect. Notes Comput. Sci. Eng.*, pages 409–416. Springer, Berlin, 2008.

[63] Thomas E Skinner, Timo O Reiss, Burkhard Luy, Navin Khaneja, and Steffen J Glaser. Application of optimal control theory to the design of broadband excitation pulses for high-resolution nmr. *Journal of Magnetic Resonance*, 163(1) :8 – 15, 2003.

[64] Marc Snir, Steve Otto, Steven Huss-Lederman, David Walker, and Jack Dongarra. *MPI-The Complete Reference, Volume 1 : The MPI Core*. MIT Press, Cambridge, MA, USA, 2nd. (revised) edition, 1998.

[65] Gunnar Andreas Staff and Einar M. Rønquist. Stability of the parareal algorithm. In *Domain decomposition methods in science and engineering*, volume 40 of *Lect. Notes Comput. Sci. Eng.*, pages 449–456. Springer, Berlin, 2005.

[66] Carmen M. Tesch, Lukas Kurtz, and Regina de Vivie-Riedle. Applying optimal control theory for elements of quantum computation in molecular systems. *Chemical Physics Letters*, 343(5-6) :633 – 641, 2001.

[67] Zdenfok Tošner, Thomas Vosegaard, Cindie Kehlet, Navin Khaneja, Steffen J. Glaser, and Niels Chr. Nielsen. Optimal control in nmr spectroscopy : Numerical implementation in simpson. *Journal of Magnetic Resonance*, 197(2) :120 – 134, 2009.

[68] Zdeněk Tošner, Steffen J. Glaser, Navin Khaneja, and Niels Chr. Nielsen. Effective hamiltonians by optimal control : Solid-state nmr double-quantum planar and isotropic dipolar recoupling. 125(18) :184502, 2006.

[69] Thomas Vosegaard, Anders Malmendal, and Niels C. Nielsen. The flexibility of simpson and simmol for numerical simulations in solid-and liquid-state nmr spectroscopy. *Monatshefte f-r Chemie / Chemical Monthly*, 133 :1555–1574, 2002. 10.1007/s00706-002-0519-2.

[70] Wusheng Zhu and Herschel Rabitz. A rapid monotonically convergent iteration algorithm for quantum optimal control over the expectation value of a positive definite operator. *The Journal of Chemical Physics*, 109(2) :385–391, 1998.

Résumé

Cette thèse présente des algorithmes permettant la parallélisation dans la direction temporelle de la simulation des systèmes régis par des équations aux dérivées partielles issues de trois domaines d'application proposant des modèles complexes :

1. Contrôle optimal d'équations paraboliques :
Nous développons deux algorithmes parallèles (SITPOC et PITPOC). Ces deux algorithmes sont basés sur une méthode générale de décomposition en temps des problèmes de contrôle optimal. Un résultat de convergence est obtenu pour SITPOC ainsi que des interprétations matricielles des deux algorithmes.

2. Cinétique de la population de neutrons dans un réacteur nucléaire :
Nous concevons un solveur parallélisé en temps regroupant toutes les variables issues de la modélisation neutronique d'un réacteur. Notre méthode est adaptée aux différents scénarios possibles réduisant la physique. Les résultats numériques sont comparables à ceux obtenus par le code MINOS du CEA. La parallélisation repose sur un schéma de type pararéel pour la résolution en temps. Nous considérons plusieurs modèles physiques de la cinétique des neutrons que nous traitons par notre algorithme dans lequel le solveur grossier utilisé correspond à une simulation par réduction du modèle physique. Cette réduction est bénéfique puisqu'elle permet une accélération importante du traitement en temps machine.

3. Contrôle par résonance magnétique nucléaire et information quantique :
Ce chapitre présente un travail d'ouverture sur une méthode parallèle en temps pour la résolution d'un problème de contrôle optimal en résonance magnétique nucléaire. Notre méthode produit une accélération importante par rapport à l'algorithme de référence non parallèle. De plus, les champs de contrôle calculés par notre méthode sont lisses, ce qui permet une mise en œuvre expérimentale plus simple dans les instruments. Les tests numériques prouvent l'efficacité de notre approche. Sur des exemples académiques et sans faire usage de techniques de programmation avancées, nous obtenons des accélérations de résolution significatives.

Abstract

This thesis presents algorithms allowing parallelization in the time direction in the simulation of systems, which are governed by partial differential equations. These methods are applied in three application fields, and complex models.

1. Parabolic Optimal Control :
We develop two parallel algorithms (SITPOC and PITPOC). These two algorithms are based on a general method of time domain decomposition of optimal control problems. A convergence result for SITPOC is obtained. Moreover, a matrix interpretation for both algorithms is also given.

2. Kinetics of the population of neutrons in a nuclear reactor :
We design a parallel in time solver gathering all the variables associated with a nuclear reactor. Our method is adapted to various possible scenarios used in model reduction. The results of this solver are comparable with those obtained by the MINOS code of the CEA. The time parallelization is based on a parareal in time scheme for the time resolution. We consider several physical models of the kinetics of the neutrons. We simulate these models with the parareal in time algorithm in which the coarse solver corresponds to a reduced physical model. This reduction allows an important acceleration of the computational time. **3. Pulse sequence design in nuclear magnetic resonance and quantum information :**
This chapter presents preliminary work on a time-parallel method for the resolution of an optimal control problem related to magnetic nuclear resonance. Our method produces an important acceleration compared with the nonparallel version. Moreover, the control fields computed with our method are smooth, which allow a simpler experimental implementation. Numerical tests prove the efficiency of our approach. On academic examples and without optimizing the code, we obtain significant improvements.