Développement de schémas de découplage pour la résolution de systèmes dynamiques sur architecture de calcul distribuée

Duc Toan Pham

Université Claude Bernard

N° attribué par la bibliothèque

THÈSE

pour obtenir le grade de

Docteur de Université Claude Bernard

Spécialité : **Modélisation et Calcul Scientifique**

préparée au laboratoire **Institut Camille Jordan - UMR 5208**

dans le cadre de l'École Doctorale **École Doctorale Info-Math**

présentée et soutenue publiquement
par

Toan PHAM

le 30 Septembre 2010

Titre:

Développement de schémas de découplage pour la résolution de systèmes dynamiques sur architecture de calcul distribuée

Directeur de thèse: **Damien TROMEUR-DERVOUT**
Co-directeur de thèse: **Fabienne OUDIN-DARDUN**

Jury

M. Jérôme HELIE,	Jury
M. Christophe PRUD'HOMME,	Jury
M. Marc GARBEY,	Rapporteur
M. Florian DEVUIST,	Rapporteur
M. Damien TROMEUR-DERVOUT,	Directeur de thèse
Mme. Fabienne OUDIN-DARDUN,	Co-directeur de thèse

Remerciements

Je tiens à remercier en tout premier Damien Tromeur-Dervout qui a dirigé cette thèse. Tout au long de ces trois années, il a su orienter mes recherches dans de nouvelles directions. Je tiens également à remercier Mme Fabienne Oudin-Dardun qui est ma codirectrice de thèse.

Je remercie les rapporteurs de cette thèse Florian De Vuyst et Marc Garbey pour l'intérêt qu'ils ont apporté à mon travail. Merci également à Christophe Prud'homme et Jérôme Helie ont accepté de faire partie du jury de cette thèse.

Je remercie mes parents et mon grand frère qui m'ont donnés leur soutient pendant mes années d'études et surtout pendant le début de mes études supérieures.

Je n'oublierai pas Chantal, Nhon, Jean-Claude et Dominique pour leur aide et leur chaleureux accueil dès mes premiers jours en France. Un grand merci à eux pour m'avoir accepté et considéré comme un membre de la famille, de m'avoir soutenu pendant les périodes difficiles.

Enfin, une pensée émue pour tous les collègues avec qui j'ai partagé une salle, un café, un repas, une sortie de ski pendant ces dernières années : Olivier, Farid, Daniel, Thomas, Jonathan, Anne, Romain, Patrice...et toute l'équipe du CDCSP.

Résumé

Nous nous intéressons dans ce mémoire à des méthodes de parallélisation par découplage du système dynamique. Plusieurs applications numériques de nos jours conduisent à des systèmes dynamiques de grande taille et nécessitent des méthodes de parallélisation en conséquence pour pouvoir être résolues sur les machines de calcul à plusieurs processeurs. Notre but est de trouver une méthode numérique à la fois consistante et stable pour réduire le temps de la résolution numérique.

La première approche consiste à découpler le système dynamique en sous-systèmes contenant des sous-ensembles de variables indépendants et à remplacer les termes de couplage par l'extrapolation polynomiale. Une telle méthode a été introduite sous le nom de schéma C(p, q, j), nous améliorons ce schéma en introduisant la possibilité à utiliser des pas de temps adaptatifs. Cependant, notre étude montre que cette méthode de découplage ne peut satisfaire les propriétés numériques que sous des conditions très strictes et ne peut donc pas s'appliquer aux problèmes raides présentant des couplages forts entre les sous-systèmes.

Afin de pouvoir répondre à cette problématique de découplage des systèmes fortement couplés, on introduit le deuxième axe de recherche, dont l'outil principal est la réduction d'ordre du modèle. L'idée est de remplacer le couplage entre les sous-ensembles de variables du système par leurs représentations sous forme réduite. Ces sous-systèmes peuvent être distribués sur une architecture de calcul parallèle. Notre analyse du schéma de découplage résultant nous conduit à définir un critère mathématique pour la mise à jour des bases réduites entre les sous-systèmes. La méthode de réduction d'ordre du modèle utilisée est fondée sur la décomposition orthogonale aux valeurs propres (POD). Cependant, ne disposant pas à priori des données requises pour la construction de la base réduite, nous proposons alors un algorithme de construction incrémentale de la base réduite permettant de représenter le maximum des dynamiques des solutions présentes dans l'intervalle de simulation.

Nous avons appliqué la méthode proposée sur les différents systèmes dynamiques tels que l'exemple provenant d'une EDP et celui provenant de l'équation de Navier Stokes. La méthode proposée montre l'avantage de l'utilisation de l'algorithme de découplage basé sur la réduction d'ordre. Les solutions numériques sont obtenues avec une bonne précision comparées à celle obtenue par une méthode de résolution classique tout en restant très performante selon le nombre de sous-systèmes définis.

Abstract

In this thesis, we are interested in parallelization algorithm for solving dynamical systems. Many industrial applications nowadays lead to large systems of huge number of variables. A such dynamical system requires parallel method in order to be solved on parallel computers. Our goal is to find a robust numerical method satisfying stability and consistency properties and suitable to be implemented in parallel machines.

The first method developed in this thesis consists in decoupling dynamical system into independent subsystems and using polynomial extrapolation for coupled terms between subsystems. Such a method is called $C(p, q, j)$. We have extended this numerical scheme to adaptive time steps. However, this method admits poor numerical properties and therefore cannot be applied in solving stiff systems with strong coupling terms.

When dealing with systems whose variables are strongly coupled, contrary to the technique of using extrapolation for coupled terms, one may suggest to use reduced order models to replace those terms and solve separately each independent subsystems. Thus, we introduced the second approach consisting in using order reduction technique in decoupling dynamical systems. The order reduction method uses the Proper Orthogonal Decomposition. Therefore, when constructing reduced order models, we do not have all the solutions required for the POD basis, then we developed a technique of updating the POD during the simulation process.

This method is applied successfully to solve different examples of dynamical systems : one example of stiff ODE provided from PDE and the other was the ODE system provided from the Nervier-Stokes equations. As a result, we have proposed a robust method of decoupling dynamical system based on reduced order technique. We have obtained good approximations to the reference solution with appropriated precision. Moreover, we obtained a great performance when solving the problem on parallel computers

Table des matières

Introduction

Ce travail s'inscrit dans la problématique de la résolution parallèle des équations différentielles ordinaires (EDO).

Pour $0 < T < +\infty$ et $t_0 \in \mathbb{R}$ données, on considère le problème dit de Cauchy sur $I = (t_0, t_0+T)$ qui consiste à trouver la fonction à variable réelle $y : I \to \mathbb{R}^n \in \mathcal{C}^1(I)$ solution de l'EDO à valeur scalaire ou vectorielle soumise à des conditions initiales données, suivant :

$$\left\{ \begin{array}{lcl} y'(t) & = & f(t, y(t)), \quad t \in I, \\ y(t_0) & = & y_0, \end{array} \right. \tag{1}$$

où $f(t, y(t))$ est continue par rapport à chacune de ses variables.

Les méthodes numériques pour approximer la solution du problème définissent une discrétisation de l'intervalle I. Afin de simplifier la lecture de ce manuscrit, on considère que le pas de discrétisation en temps $h > 0$ est constant. Posons alors $t_n = t_0 + nh$ où $n = 0, \ldots, N_h$ et $h = T/N_h$. Notons $I_n = [t_n, t_{n+1}]$ les sous-intervalles de discrétisation de I, u_j l'approximation au noeud t_j de la solution exacte $y_j := y(t_j)$. De même, notons $f(t_j, u_j)$ par f_j. Une méthode numérique d'intégration à un pas est définie par la donnée d'une fonction Φ et s'écrit pour tout $n \geq 0$,

$$u_{n+1} = u_n + h\Phi(t_n, u_n, u_{n+1}). \tag{2}$$

Les schémas à un pas les plus simples sont :

- le schéma d'Euler explicite : $\Phi(t_n, u_n, u_{n+1}) = f_n$,
- le schéma d'Euler implicite : $\Phi(t_n, u_n, u_{n+1}) = f_{n+1}$.

On peut également définir des schémas à pas multiple. Ces systèmes différentiels sont issus de la modélisation mathématiques des phénomènes physiques pour la conception de produits manufacturés. La prise en compte de différents phénomènes physiques conduit à l'accroissement du nombre d'inconnues dans ces systèmes qui sont le plus souvent raides, avec différentes échelles dans les solutions.

Nous nous intéressons ici à la résolution numérique des systèmes dynamiques de type raide. La raideur ("stiffness" en anglais) d'un système d'EDO a été introduite par Curtiss & al (voir [14] ou Lambert [33]). "*Un système d'EDO est dit raide si pour le résoudre numériquement avec une méthode admettant une région de stabilité absolue de taille finie et pour toute condition initiale auquelle le système admet une solution, cette méthode numérique doit utiliser des pas de discrétization extrêmement petits par rapport à la régularité de la solution exacte*". Cependant, différentes définitions de la raideur existent dans la littérature selon le contexte. Ce manuscrit s'appuie sur la définition suivante pour la raideur d'un système dynamique :

Définition 0.1. *Hairer (voir [28]) a défini la raideur d'un système d'EDO par le fait que les méthodes d'intégration explicites nécessitent pour des raisons de stabilité numérique des pas de temps trop petits pour obtenir la solution dans des temps raisonnables.*

Ces propriétés impliquent l'utilisation de schémas d'intégration en temps implicites, pour lesquels l'utilisation de pas de temps d'intégration raisonnables est possible du fait de la plus grande stabilité de ces schémas.

Contrairement aux méthodes d'intégration en temps explicites où la solution au pas de temps t_{n+1} est obtenue par une évaluation directe d'une fonction ayant pour arguments les pas de temps précédents, les méthodes d'intégration implicites nécessitent de trouver la valeur de la solution au pas de temps t_{n+1} par la résolution d'un problème non-linéaire. Pour ce faire, on utilise des méthodes de type Newton mettant en jeu la Jacobienne du système différentiel. Par conséquent, lorsque le nombre d'inconnues croît, le coût de résolution de ce système linéaire (différent à chaque pas de temps du fait des nonlinéarités du problème) devient prohibitif avec les capacités actuelles d'un seul processeur, et ne permet pas d'obtenir une solution pour des temps de calcul raisonnables.

Afin de pallier à ce problème, la parallélisation des systèmes d'EDO a fait l'objet de travaux intensifs depuis le début des années 90. Ces travaux sont souvent théoriques mais peu d'implémentations réelles, qui soient performantes, ont été mises en oeuvre. Ces méthodes d'intégration parallèles peuvent être regroupées en trois grandes familles de méthodes : la distribution locale de calcul, le partitionnement de l'ensemble des variables d'état et la décomposition du domaine en temps.

La distribution locale de calcul concerne principalement le partage des calculs élémentaires indépendants intervenant dans le schéma d'intégration (produit matrice-vecteur, calcul par bloc...) entre plusieurs processeurs (ou processus) de calcul. On peut faire le lien avec les algorithmes parallèles des systèmes algébriques linéaires (voir Bersekas [5, part I ch. 2]). Ce type de parallélisation est mise en oeuvre dans la suite SUNDIALs (voir [9]), utilisant des méthodes de Krylov où le produit matrice-vecteur est réalisé par distribution de la Jacobienne en bloc de lignes.

Dans [25], D. Guibert & al découpent le système en sous-systèmes à partir de considérations de partitionnement du graphe de la matrice d'adjacence associée à la matrice Jacobienne. Ceci leur permet d'utiliser les techniques classiques de décomposition de domaine de type complément de Schur primal pour résoudre le problème linéarisé dans la méthode de Newton. Cette approche nécessite toutefois de construire la matrice jacobienne du système entier à chaque pas de temps par différentiation automatique.

La deuxième approche de parallélisation en espace consiste à découpler le système global en partitionnant ce dernier en sous-ensembles de variables d'état (les sous-systèmes). Pour chaque sous-système, on cherche une solution locale approximant la solution cherchée avec un coût de calcul relativement faible. Par exemple, les travaux de Skelboe [44, 51], étudient le schéma de type semi-implicite (Backward Euler multirate methods, waveform relaxation methods) où chaque composante locale de la solution est résolue implicitement, les autres étant traitées explicitement.

La troisième famille de méthodes de parallélisation consiste à décomposer en temps le système différentiel. On résout alors simultanément le système sur les différents intervalles de temps (méthodes de tirs multiples, parareal) voir [39] ou plus récemment dans les travaux de Chartier [10], Guibert & al [24].

Les travaux présentés dans ce manuscript s'inscrivent dans la deuxième approche de découplage physique du système en sous-systèmes. Les principales contributions sont les suivantes :

- Dans un premier temps, nous proposons d'étendre la méthode C(p,q,j) développée dans M Garbey & Tromeur-Dervout [19] sur des schémas à pas de temps fixes, pour des schémas à pas variables. Nous proposons un algorithme d'échange d'informations entre les sous-systèmes qui se base sur une formule d'extrapolation par différences divisées. La difficulté de l'approche est de définir une fréquence d'échange d'informations entre les sous systèmes a priori, faute de disposer d'un indicateur numérique liant la dynamique des deux sous-systèmes.
- Le deuxième axe de recherche consiste à utiliser la réduction du modèle par la POD ("proper orthogonal decomposition" ou la décomposition orthogonale aux valeurs propres). Les modèles réduits prennent compte de la dynamique manquante entre les parties découplées dans les sous-systèmes.

Dans la suite on considère le problème raide (1) où la fonction y est à valeurs dans $\mathbb{R}^n$ et on s'intéresse aux méthodes numériques pour approximer la solution au problème.

Parmis les méthodes existentes, les méthodes à un pas sont connues pour leur simplicité d'écriture mais sont limitées en précision. On introduira donc des méthodes à pas multiple. Un schéma à pas multiple s'écrit sous la forme :

$$u_{n+1} = \sum_{j=0}^{p} a_j u_{n-j} + h \sum_{p}^{j=0} b_j f_{n-j} + h b_{-1} f_{n+1},\ n = p, p+1, \ldots \tag{3}$$

Ce schéma est à $p+1$ pas, avec $p \geq 0$. Pour $p = 0$ on retrouve le schéma à un pas. Il existe deux types de schémas à pas multiple linéaire : Adams et BDF ("backward differentiation formulae" ou schéma à differentiation rectrograde). Les schémas d'Adams sont divisés entre les méthodes Adams-Bashforth (explicites) et celles d' Adams-Moulton (implicites). Les schémas de type BDF sont implicites.

Un schéma de type BDF d'ordre p s'écrit sous la forme :

$$u_{n+1} = \sum_{j=-1}^{p} a_j u_{n-j} + h b_{-1} f_{n+1}, \tag{4}$$

où $b_{-1} \neq 0$. Pour obtenir u_{n+1} dans l'équation précédente lorsque f est non linéraire, une méthode de type Newton (ou Newton modifiée) pour rechercher les racines est nécéssaire. Dans ce cas, on fait intervenir la Jacobienne de f dans la résolution non-linéraire du système.

Les méthodes numériques développées dans ce travail s'appuient, pour traiter localement les problèmes de type raides avec les méthodes BDF, sur le solveur CVODE de sur la bibliothèque SUNDIALs (SUite of Nonlinear and DIffential/ALgebraic equation Solvers), voir [12, 46].

0.1 Méthodes développées

L'idée principale de notre travail consiste à partitionner le système de Cauchy en sous-systèmes disjoints. A titre d'exemple, on découple les composantes de la variable y

de l'équation (1) en deux parties $y = (y_1, y_2)^\top$

$$\begin{cases} y_1'(t) &= f_1(t, y_1, y_2) \\ y_2'(t) &= f_2(t, y_1, y_2) \end{cases} \tag{5}$$

Nous souhaitons dès lors résoudre chaque sous-système sur un processeur. Toutefois, la solution de l'autre système est nécessaire pour le calcul. Les méthodes parallèles développées consistent à définir des méthodes numériques basées sur les schémas présentés précédemment pour résoudre localement y_1 et y_2. De plus, on souhaite préserver les propriétés numériques de consistence et de stabilité pour obtenir une approximation précise dans un temps d'exécution raisonable.

La première approche consiste en une extension du schéma $C(p, q, j)$, où le système est décomposé en deux sous-systèmes selon le partitionnement des variables. Le premier système (respectivement le deuxième système) considère uniquement y_1 (respectivement y_2) comme inconnues. Le terme de couplage d'unsous-système est alors remplacé par une prédiction de celui-ci, obtenue à partir des solutions de l'autre sous-système. Ce type de schéma $C(p, q, j)$ initialement dévelopé avec des pas de temps constants a été étendu au cas où les pas de temps sont variables. De plus, une mise à jour par différences divisées du terme de couplage permet d'avoir un algorithme de communication asynchrone. Cependant, l'aspect de découplage du système par $C(p, q, j)$ a l'inconvénient de ne pas avoir une estimation a priori de la dynamique du terme découplé. Pour cette raison un autre axe de recherche basé sur la réduction d'ordre du modèle a été introduit dans cette thèse.

La simulation par réduction d'ordre du modèle est utilisée avec succès sur plusieurs types de problèmes scientifiques et d'applications industrielles (voir [2], [47–49]). Cette méthode de réduction du modèle (notamment par la POD) permet de définir une approximation du système physique complet par un système de dimension inférieure. Son utilisation a été validée dans différents domaines incluant : la dynamique des fluides, la cohérence des structures-turbulences ([4] [47–49], [35, 36], [45]), l'optimisation de forme (voir [38], [1, 34]), le contrôle d'optimale (voir [42], [32], [21, 22])...

Dans un contexte général, la contruction d'un modèle réduit par la POD résulte de la construction d'une base optimale représentative du modèle physique. Sirovich a introduit la méthode des clichés, voir [47–49], permettant d'obtenir la base POD. Cette méthode consiste à appliquer la POD sur l'ensemble des données (dénommées par les clichés) obtenu soit par une simulation directe, soit fourni par réalisations expérimentales. Ces clichés doivent former une famille génératrice dans l'espace réduit. Par conséquent, hormis la contrainte sur la précision de la résolution pour obtenir des clichés fiables, il faut disposer de suffisament des clichés pour couvrir tout l'intervalle de simulation. Cette construction du modèle réduit par POD permet d'analyser le système physique (ou optimiser les paramètres) avec un coût de calcul largement plus faible.

Notre objectif est de réduire le temps d'exécution de la simulation des systèmes dynamiques. Nous proposons alors d'utiliser la technique de réduction d'ordre du modèle pour paralléliser la résolution du système d'EDO écrit sous la forme d'un système partitionné.

Cependant, les clichés ne sont pas disponibles sur tout l'intervalle de simulation. En effet, dans ce cas le problème est déjà résolu et il n'y a pas d'intérêt à construire un modèle réduit pour en accélérer le calcul. De plus, les problèmes étant raides et non linéaires, on

ne dispose pas a priori de toutes les caractéristiques de la solution du problème sur un intervalle de temps réduit, afin de contruire le modèle réduit qui puissent être valide sur tout l'intervalle de simulation.

En revanche, on peut établir à partir des solutions sur un petit intervalle de temps, un modèle réduit qui sera valide sur un certain intervalle de temps. Ces solutions issues de la résolution du système global sont obtenues avec un coût faible puisque l'on résout le système complet sur quelques pas de temps initiaux. A partir des données initiales, on construit la matrice des clichés donnant le modèle réduit. L'idée de découplage consiste à remplacer une partie des variables du système par son modèle réduit. Le modèle réduit couplé avec une partie restante non réduite forment un système indépendant.

D'un point de vue pratique, les mêmes schémas numériques que ceux utilisés pour résoudre le système complet sont mis en oeuvre pour résoudre le système réduit. En effet l'espace des fonctions auquel appartient la solution sous forme réduite est un sous-espace vectoriel de l'espace de la solution exacte au problème. Toutes les propriétés du schéma numérique restent valides pour le schéma intégrant la projection sur l'espace réduit.

Toutefois, la durée de validité du système réduit est a priori limitée. Nous devons mettre à jour la base du système réduit afin de garantir que les sous-systèmes sont correctement résolus, c'est à dire qu'ils ont le même comportement que le système complet.

Une première stratégie consiste à échanger de manière régulière (à des points de rendez-vous) les bases réduites. Cependant, cette approche, qui est relativement simple à implémenter, présente plusieurs difficultés. Premièrement, nous ne disposons pas de critères a priori pour fixer la fréquence des échanges (définition des points de rendez-vous). Deuxièmement, nous voulons éviter de mettre à jour la base réduite lorsque cela n'est pas nécessaire. C'est pourquoi nous établissons un critère mathématique a posteriori pour la mise à jour des bases réduites en cours de simulation. Ce critère algébrique fondé sur l'erreur entre le système partitionné et le système complet, est affiné numériquement lorsque les composantes de la solution sont multi-échelles. On économise ainsi considérablement le volume de données à échanger entre les systèmes partitionnés. Néanmoins, la construction d'une nouvelle base réduite à un instant t_n nécessite la POD sur plusieurs clichés aux temps $t_n, t_{n-1}, ..., t_{n-q+1}$. La nouvelle base POD peut se calculer par deux différentes approches. Nous proposons deux approches différentes pour obtenir la base POD mise à jour. La première approche consiste simplement récalculer toute la base POD à partir des derniers clichés obtenus. La deuxième approche utilise les informations sur les anciennes bases réduites pour se mettre à jour.

En effet, nos tests numériques montrent que la base reduite est optimale dans le sens où elle peut rendre compte de toutes les dynamiques de la solution est celle construite en prenant des clichés sur tout l'intervalle de temps. Afin de combiner ces deux aspects, éviter de reconstruire un partie de l'information déjà présente et de capturer le plus de dynamiques possibles par le modèle réduit partitionné, nous avons mis au point une technique de calcul de POD incrémentale.

Les essais numériques montrent la robustesse de ces approches et l'obtention de gains significatifs en terme de précision de la solution obtenue et en temps d'exécution parallèle par rapport au temps séquentiel.

0.2 Structure de la thèse

Ce mémoire est organisé en cinq chapitres classés en deux parties. La première partie est dédiée à l'introduction des schémas numériques classiques et celui du découplage par $C(p, q, j)$. La deuxième partie présente les travaux concernant la parallélisation par la réduction d'ordre du modèle via la POD :

- **Chapitre 1** Le premier chapitre détaille les méthodes numériques ainsi que les concepts de base pour la résolution numériques du système dynamique. Les propriétés essentielles y sont rappelées.
- **Chapitre 2** Le schéma de découplage basé sur $C(p, q, j)$ est présenté dans ce chapitre. Après une présentation du schéma numérique, on introduit les propriétés numérique du schéma ainsi que les algorithmes. Les résultats numériques montrent l'intérêt de l'approche.
- **Chapitre 3** Dans le troisième chapitre, la décomposition orthogonale aux valeurs propres est présentée sur le plan théorique et pratique. Le calcul numérique est abordé et l'application de la POD pour la réduction d'ordre du modèle y est présentée.
- **Chapitre 4** Nous avons ensuite appliqué la méthode de réduction du modèle pour découpler le système dynamique. Dans ce chapitre, nous utilisons la méthode des clichées de manière locale pour construire le modèle réduit. L'analyse de l'erreur de la méthode ainsi que les algorithmes y sont présentés. Cependant, la méthode de la POD locale atteint ses limitations dans certains cas. Notre étude alors poursuit dans la direction d'évaluation d'une POD globale pour mieux présenter les dynamiques de chacun des sous-systèmes.
- **Chapitre 5** Nous avons introduit la méthode de la mise à jour du modèle réduit par l'algorithme de calcul incrémental de la POD. L'évaluation de la SVD incrémentale permet de construire une base POD réduite globale sur tout l'intervalle de simulation. Cette technique permet de surpasser les difficultés rencontrées au chapitre 4. Finalement, nous avons retrouvé une méthode de découplage efficace et fiable pour résoudre en parallèle le système dynamique.

Première partie

Schéma de découplage $C(p, q, j)$ basée sur l'extrapolation

Chapitre 1

Introduction aux méthodes numériques pour la résolution du système d'équations différentielles ordinaires

Ce chapitre présente les concepts fondamentaux des résolutions numériques d'équation aux dérivées ordinaires (EDO). Ces résultats sont rappelés afin de présenter une analyse des schémas de découplages qui seront introduits dans les chapitres suivants. Notamment dans le chapitre 2, pour le schéma de $C(p, q, j)$, ils nous permettront d'établir un critère sur le choix de paramètres p, q et j pour obtenir un schéma numérique de découplage consistent et stable.

Nous rappelons les notions de consistence et de stabilité pour des méthodes numériques à un pas sur le problème de Cauchy. Ces notions s'étendent pour l'analyse des méthodes à pas multiple au moyen d'études sur les équations de récurrences linéaires. Ce qui conduit aux deux résultats fondamentaux de [15] qui sont : la relation entre le nombre de pas et l'ordre de consistence (la *première barrière de Dahlquist*) et stabilité pour un schéma à pas multiple linéaire (la *deuxième barrière de Dahlquist*). La fin du chapitre va traiter le cas d'un système d'EDO.

1.1 Problème de Cauchy

Considérons le problème de Cauchy suivant :

$$\begin{cases} y'(t) &= f(t, y(t)), \quad t \in I, \\ y(t_0) &= y_0, \end{cases} \tag{1.1}$$

d'où I est un intervalle dans $\mathbb{R}$ contenant t_0. Si f est continue par rapport à t, la solution à (1.1) satisfait :

$$y(t) - y_0 = \int_{t_0}^{t} f(\tau, y(\tau)) d\tau \tag{1.2}$$

Abordons premièrement les notions de l'existence et l'unicité pour (1.1) :

- **L'existence locale et l'unicité**. Supposons que $f(t, y)$ est localement Lipschitzienne continue au point (t_0, y_0) par rapport à y, c'est à dire, il existe deux voisinages, $J \subset I$ contenant t_0 de largeur de r_J, et Σ autour de y_0 à largeur de r_Σ, et une constante $L > 0$ telle que :

$$|f(t, y_1) - f(t, y_2)| \leq L|y_1 - y_2| \quad \forall t \in J,\ \forall y_1, y_2 \in \Sigma \tag{1.3}$$

  Le problème de Cauchy admet une et une seule solution dans le boule voisinage de t_0 de rayon r_0 avec $0 < r_0 < min(r_J, r_\Sigma/M, 1/L)$, $M = max_{J\times\Sigma}|\partial f(t, y)/\partial y|$.
  La condition (1.3) est automatiquement vérifiée si f admet de dérivée continue par rapport à y. En conséquence, pour un tel cas il suffit de choisir L en pétant le maximum de $|\partial f(t, y)/\partial y|$, $(t, y) \in J \times \Sigma$.
- **L'existence et l'unicité globale**. Le problème admet une solution globale unique lorsque l'on peut prendre $J = I$ et $\Sigma = \mathbb{R}$ dans (1.3). C'est à dire f est uniformément Lipschitzienne continue par rapport à y.

Ensuite, la notion de stabilité est introduite pour le problème de Cauchy. Considérons le problème perturbé suivant :

$$z'(t) = f(t, z(t)) + \delta(t), \quad t \in I, \tag{1.4}$$

$$z(t_0) = y_0 + \delta_0 \tag{1.5}$$

où $\delta_0 \in \mathbb{R}$ et δ est une fonction continue définie sur I. On quantifie la sensibilité du problème de Cauchy par rapport à la perturbation par :

Définition 1.1. *Soit I un ensemble borné. Le problème de Cauchy est stable (ou **stable dans le sens de Liapunov**) sur I si, pour toutes perturbations $(\delta_0, \delta(t))$ satisfaisant $\delta_0 < \varepsilon$, $|\delta(t)| < \varepsilon\ \forall t \in I$ avec $\varepsilon > 0$ suffisament petit, alors :*

$$\exists C > 0 \text{ telle que } |y(t) - z(t)| < C\varepsilon \tag{1.6}$$

De plus, la constante C dépend en général des données t_0, y_0 et f, sauf ε.
*Si I n'est plus borné, on dit que le problème de Cauchy est asymtotiquement stable si, d'une part on a **la stabilité au sens de Liapunov** sur tout interval I borné, d'autre part la condition suivante :*

$$|y(t) - z(t)| \to 0,\ t \to +\infty \tag{1.7}$$

cette dernière condition implique que $\lim_{t\to\infty} |\delta(t)| = 0$.

Remarquons ensuite que problème de Cauchy est stable si f est uniformement Lipschitzienne en y (condition suffisante). En effet posons $w(t) = y(t) - z(t)$, on a :

$$w'(t) = f(t, z(t)) - f(t, y(t)) + \delta(t)$$

donc :

$$w(t) = \delta_0 + \int_{t_0}^{t} [f(s, z(s)) - f(s, y(s))]\, ds + \int_{t_0}^{t} \delta(s) ds, \quad \forall t \in I$$

d'après le lemme de Grönwall en utilisant la condition précédente sur f, on peut déduire que :

$$|w(t)| \leq (1 + |t - t_0|)\varepsilon e^{L|t-t_0|}, \quad \forall t \in I$$

d'où (1.7) avec $C = (1 + C_I)e^{LC_I}$ avec $C_I = \max_{t\in I} |t - t_0|$.

Lemme 1.1. ***(Lemme de Grönwall)*** *Soient $t \in (t_0, T)$, $\phi(t) \geq 0$ et $\varphi(t) \geq 0$ sont des fonctions continues, de plus $\varphi(t)$ non-négative. Soit $g(t)$ une fonction non-décroissante sur $[t_0, T]$. Si φ vérifie l'inégalité suivante :*

$$\phi(t) \leq g(t) + \int_{t_0}^{t} \varphi(\tau)\phi(\tau)d\tau, \quad \forall t \in [t_0, T]$$

alors

$$\phi(t) \leq g(t) \exp \int_{t_0}^{t} \varphi(\tau)d\tau, \quad \forall t \in [t_0, T]$$

Très peu EDOs non-linéaires admettent des solutions analytiques explicites. C'est pourquoi il est nécessaire de les résoudre numériquement par l'intermédiaire de schéma de discrétisation.

1.2 Méthodes numériques à un-pas

Considérons le problème de Cauchy (1.1) sur l'intervalle $I = [t_0, T] \subset \mathbb{R}$ non vide. I est ensuite divisé en N_h sous-intervalles de taille h pour la simplicité de la présentation. h est appelé le pas de discrétisation. On considère l'ensemble des noeuds de discrétisation $t_n = t_0 + nh,\ n = 0, 1, \ldots, N_h$ qui forment les sous-intervalles $I_n = [t_n, t_{n+1}] \subset I$. Soit u_j l'approximation de la solution exacte $y(t_j)$ aux noeud t_j. Pour abréger, on note y_j pour $y(t_j)$, f_j pour $f(t_j, y(t_j))$.

Définition 1.2. *Une méthode numérique appelée à un pas pour approximer la solution du problème de Cauchy si pour tout $n \geq 0$, u_{n+1} ne dépend que de u_n. Autrement la méthode est appelée méthode à pas multiple .*

Rappelons quelques méthodes à un pas :
- Euler explicite : $u_{n+1} = u_n + hf_n$
- Euler implicite : $u_{n+1} = u_n + hf_{n+1}$
- Crank-Nicolson : $u_{n+1} = u_n + \frac{h}{2}[f_n + f_{n+1}]$
- Heun : $u_{n+1} = u_n + \frac{h}{2}[f_n + f(t_{n+1}, u_n + hf_n)]$

Définition 1.3. ***(Méthode explicite et implicite)*** *Une méthode numérique est explicite si u_{n+1} est calculé directement en fonction des valeurs précédentes $u_{n+1-k},\ k \geq 1$, implicite si u_{n+1} dépend de elle même par f.*

1.3 Analyse des méthodes à un-pas

Les deux concepts cruciaux dans la convergence d'un schéma numérique sont la consistence et la stabilité.

Une méthode numérique à un pas est définie avec un pas de temps h et sa fonction d'incrément Φ comme suit :

$$u_{n+1} = u_n + h\Phi(t_n, u_n, f_n; h),\ 0 \leq n \leq N_h - 1,\ u_0 = y_0 \tag{1.8}$$

Φ s'appelle la fonction d'incrément.

1.3.1 Consistence

Posons $y_n = y(t_n)$ pour la simplicité d'écriture et écrivons l'équation (1.8) pour la solution exacte au point t_n :

$$y_{n+1} = y_n + h\Phi(t_n, y_n, f(t_n, y_n); h) + \varepsilon_{n+1} \tag{1.9}$$

où ε est le résidu au pas de temps t_{n+1}. Posons τ_{n+1} comme erreur de troncature locale au noeud t_{n+1} :

$$h\tau_{n+1}(h) = \varepsilon_{n+1} \tag{1.10}$$

Le terme l'erreur de troncature globale est défini par :

$$\tau(h) = \max_n |\tau_{n+1}(h)| \tag{1.11}$$

l'erreur de troncature globale dépend de la solution exacte y. Notons que dans le cas du schéma Euler progressif la fonction d'incrément est de la forme $\Phi(t_n, u_n, f_n; h) = f_n$, tandis que celle de la méthode de Heun prend la forme $\Phi(t_n, u_n, f_n; h) = \frac{1}{2}(f_n + f(t_n + h, u_n + hf_n))$.

Pour les méthodes à une pas considérées, leur foncions d'incrément vérifient :

$$\lim_{h\to 0} \Phi(t_n, y_n, f(t_n, y_n); h) = f(t_n, y_n),\ \forall t_n \geq t_0 \tag{1.12}$$

avec $y_{n+1} = y_n + hy'_n(t_n) + \mathcal{O}(h^2)$, on en déduit que :

$$\lim_{h\to 0} \tau_{n+1}(h) = 0 \tag{1.13}$$

La consistence d'une méthode numérique à un pas se trouve dans la définition suivante :

Définition 1.4. ***(Consistence d'une méthode numérique à un pas)*** *La méthode numérique définie dans (1.8) est dite consistante si son erreur de troncature locale vérifie (1.13). De plus la méthode est dite d'ordre p si $\forall t \in I$ si la solution exacte $y(t)$ vérifie :*

$$\tau(h) = \mathcal{O}(h^p),\ \textit{quand } h \to 0 \tag{1.14}$$

La méthode d'Euler explicite est d'ordre 1, la méthode de Heun est d'ordre 2.

1.3.2 Zéro-stabilité

La condition de zéro-stabilité pour un schéma numérique est semblable à **la condition de Liapunov** introduite pour le problème continu de Cauchy. Soit $u_n^{(h)}$ la solution numérique obtenue avec (1.8) avec un pas de discrétisation h et z_n la solution de :

$$\begin{cases} z_{n+1}^{(h)} &= z_n^{(h)} + h(\Phi(t_n, z_n^{(h)}, f(z_n^{(h)}); h) + \delta_{n+1}) \\ z_0^{(h)} &= y_0 + \delta_0 \end{cases} \tag{1.15}$$

Définition 1.5. ***Zéro-stabilité du schéma numérique à un pas*** *: la méthode numérique à un pas est dite zéro-stable s'ils existent $h_0 > 0$ et $C > 0$ tels que $\forall 0 < h \leq h_0$, $\forall \varepsilon > 0$ suffisamment petits, si $|\delta_n| \leq \varepsilon,\ n = 0, \ldots, N_h$, alors :*

$$|z_n^{(h)} - u_n^{(h)}| \leq C\varepsilon,\ n = 0, \ldots, N_h \tag{1.16}$$

La zéro stabilité est inhérente à la méthode numérique et indique le comportement du schéma (1.8) lorsque $h \to 0$. Elle représente l'insensibilité du schéma numérique aux petites perturbations des données. On constate que le problème de Cauchy est stable grâce à la condition de Lipschitz sur la fonction f. Alors que la zéro-stabilité d'un schéma numérique à un pas se vérifie grâce à la condition de Lipschitz sur la fonction d'incrément Φ, comme l'indique le théorème suivant.

Théorème 1.1. ***Zéro-stabilité*** *: Soit la méthode numérique à un pas (1.8) pour résoudre le problème de Cauchy. Si la fonction d'incrément Φ est lipchitzienne par rapport à la deuxième variable, c'est à dire s'ils existent $h_0 > 0$ et $\Lambda > 0$ tels que $\forall h \in]0, h_0]$, $n = 0, \ldots, N_h$:*

$$|\Phi(t_n, u_n^{(h)}, f(t_n, u_n^{(h)}); h) - \Phi(t_n, z_n^{(h)}, f(t_n, z_n^{(h)}); h)| \leq \Lambda |u_n^{(h)} - z_n^{(h)}| \tag{1.17}$$

alors la méthode (1.8) est zéro-stable.

La preuve du théorème repose sur l'extension du lemme de Grönwall 1.1 au cas des variables discrètes.

La zéro-stabilité pour le schéma d'Euler peut être vérifiée directement par la condition lipchitzienne de f. Pour l'étude de la zéro-stabilité des méthodes à pas-multiple, des conditions supplémentaires sur les racines des équations de récurrences linéaires sont nécessaires.

1.3.3 Convergence d'une méthode numérique

Définition 1.6. *Une méthode numérique est dite convergente si*

$$\lim_{h \to 0} \max_{1 \leq n \leq N_h} |u_n - y_n| = 0 \tag{1.18}$$

ou y_n est la solution exacte au problème de Cauchy , u_n est l'approximation provenant de la méthode numérique au noeuds t_n.

Concernant la convergence des méthodes numériques à un pas, on a le théorème suivant :

Théorème 1.2. ***(Convergence des méthodes à un pas)*** *Soit la méthode numérique à un pas (1.8) pour résoudre le problème de Cauchy (1.1). S'il existe h_0 pour que la fonction d'incrément $\Phi(t, u, f(t, u); h)$ est lipchitzienne par rapport à sa deuxième variable avec la constante de Lipschitz Λ sur D :*

$$D = \{(t, u, h) : \ t \in [0, T], \ u \in \mathbb{R}, \ h \in]0, h_0]\}$$

alors :

- *La méthode est stable*
- *La méthode est convergente si et seulement si elle est consistente :*

$$\Phi(t, y; 0) = f(t, y) \tag{1.19}$$

- *L'erreur de troncature globale définie dans (1.11) vérifie $\forall h \in]0, h_0]$:*

$$|y_n - u_n| \leq \frac{\tau(h)}{\Lambda} e^{nh\Lambda} \tag{1.20}$$

La preuve du théorème se trouve par exemple dans [27, chap. 3].

1.3.4 La stabilité absolue

La stabilité absolue est une propriété relative à la zéro-stabilité, lors on étudie l'influence du pas de discrétisation h et de l'intervalle I sur le comportement du schéma numérique. Une méthode numérique est absoluement stable si pour h fixé, u_n reste borné lorsque $n \to \infty$. Cette propriété caractérise le comportement asymptotique de u_n, tandis que la zero-stabilité indique, sur un intervalle borné, le comportement de u_n quand $h \to 0$.

L'étude de la stabilité absolue s'effectue sur le problème de Cauchy linéaire suivant (appelé aussi problème de test) :

$$\begin{cases} y'(t) &= \lambda y(t),\ t > 0 \\ y(0) &= 0 \end{cases} \tag{1.21}$$

où $\lambda \in \mathbb{C}$. Le problème (1.21) admet une solution $y(t) = e^{\lambda t}$. Le problème de test est stable si $\Re(\lambda) < 0$ ($\Re(\cdot)$ signifie la partie réelle d'un nombre complexe).

Définition 1.7. *Une méthode numérique est dite absolument stable si pour le problème test (1.21) :*

$$|u_n| \to 0, \quad \textit{quand } t_n \to \infty \tag{1.22}$$

Soit h le pas de discrétisation, la solution numérique u_n dépend de h et de λ. En effet, on définit la région de stabilité absolue d'une méthode numérique par un sous-ensemble du pland complexe $\mathcal{A} \subset \mathbb{C}$ par :

$$\mathcal{A} = \left\{ z = h\lambda \in \mathbb{C} : \lim_{t_n \to +\infty} |u_n| = 0 \right\} \tag{1.23}$$

$\mathcal{A}$ est donc l'ensemble des valeurs du produit $h\lambda$ telles qu'une méthode numérique pour approximer la solution de (1.21) tend vers zéros quand t_n tend vers l'infini.

Dans la suite on va présenter les régions de stabilité absolue pour les schémas vu ci-haut :

- Euler explicite : $\mathcal{A} = \left\{ h\lambda \in \mathbb{C} : \Re(h\lambda) < 0,\ 0 < h < -\dfrac{2\Re(\lambda)}{|\lambda|^2} \right\}$
- Euler implicite : $\mathcal{A} = \{ h\lambda \in \mathbb{C} : |1 - h\lambda| > 1 \}$
- Crank-Nicolson : $\mathcal{A} = \left\{ h\lambda \in \mathbb{C}^- \right\}$

En conséquence, une méthode numérique avec une grande région de stabilité absolue permet d'ulitiliser les grands pas de discrétization.

Définition 1.8. *Une méthode est A-stable si $\mathcal{A} \cap \mathbb{C}^- = \mathbb{C}^-$*

La méthode d'Euler rétrograde et Crank-Nicolson est A-stable, la méthode d'Euler progressif et les méthodes explicites sont conditionellement stable.

1.4 Méthode à pas multiple linéaire

Considérons les méthodes numériques linéraire à pas multiple.

Définition 1.9. *Une méthode linéaire à pas multiple de p-pas ($p \geq 1$) est une méthode telle que, $\forall n \geq p-1$, u_{n+1} dépend uniquement des u_{n+1-p} :*

$$u_{n+1} = \sum_{j=0}^{p} a_j u_{n-j} + h \sum_{j=0}^{p} b_j f_{n-j} + h b_{-1} f_{n+1} \tag{1.24}$$

Les méthodes à p-pas ont besoin de p premières valeurs de $u_0, \ldots, u_{p-1}$ pour démarrer. Les coefficients a_j et b_j caractérisent complétement une méthode à pas multiple. Si $b_{-1} = 0$, la méthode est explicite, sinon elle est implicite.

Comme dans la section précédente, on introduit la notion de la consistence du schéma à pas multiple :

Définition 1.10. ***L'erreur de troncature locale*** *pour la méthode à pas multiple $\tau_{n+1}(h)$ au pas de temps t_{n+1} est définie par :*

$$h\tau_{n+1}(h) = y_{n+1} - \left(\sum_{j=0}^{p} a_j y_{n-j} + h \sum_{j=-1}^{p} b_j y'_{n-j} \right), \; n \geq p \tag{1.25}$$

où on a utilisé la solution exacte y_{n-j} et y'_{n-j}, $j = -1, \ldots, p$ dans la formule (1.24).

la quantité $h\tau_{n+1}(h)$ est le résidu généré en t_{n+1} lorsque la solution exacte est entrée dans le schéma numérique. L'erreur de troncature globale est définie comme $\tau(h) = \max_n |\tau_n(h)|$ et on a :

Définition 1.11. *La méthode (1.24) est consistente si $\lim_{h \to 0} \tau(h) = 0$. De plus, si $\tau(h) = \mathcal{O}(h^q)$ pour $q \geq 1$ alors la méthode est dite d'ordre q.*

On peut retrouver quelques familles de méthodes à pas multiple : les méthodes de type d'Adams et de type BDF(backward differentiation formula). Notre étude traitant les problèmes de type raides, seuls les schémas BDF seront utilisés.

1.4.1 Méthodes d'Adams

Une méthode Adams provient de la forme intégrale (1.2) en utilisant l'intégration numérique de f sur l'intervalle t_n et t_{n+1}. Supposons que les pas de temps sont équidistant, le schéma d'Adams s'écrit :

$$u_{n+1} = u_n + h \sum_{j=-1}^{p} b_j f_{n-j} \tag{1.26}$$

où on distingue deux cas :

- $b_{-1} = 0$, la méthode est alors explicite (Adams-Bashforth).
- $b_{-1} \neq 0$, la méthode est implicite (Adams-Moulton).

dans la suite on présente quelques schémas de type d'Adams

Méthodes d'Adams-Bashforth

Les méthodes d'Adams-Bashforth :

– à deux pas :

$$u_{n+1} = u_n + \frac{h}{2}(3f_n - f_{n-1}) \tag{1.27}$$

– à trois pas :

$$u_{n+1} = u_n + \frac{h}{12}(23f_n - 16f_{n-1} + 5f_{n-2})$$

– à quatre pas :

$$u_{n+1} = u_n + \frac{h}{24}(55f_n - 59f_{n-1} + 37f_{n-2} - 9f_{n-3})$$

Une méthode d'Adams-Bashforth à p-pas utilise p noeuds et est d'ordre p.

Méthodes d'Adams-Moulton

La méthode d'Euler rétrograde est un cas particulier des méthodes d'Adams-Moulton (AM), les autres méthodes sont :

– AM à trois noeuds :

$$u_{n+1} = u_n + \frac{h}{12}(5f_{n+1} + 8f_n - f_{n-1}) \tag{1.28}$$

– AM à quatre noeuds :

$$u_{n+1} = u_n + \frac{h}{24}(9f_{n+1} + 19f_n - 5f_{n-1} + f_{n-2})$$

une méthode d'Adams-Moulton à p-pas utilise $p+1$ noeuds pour approximer la forme l'intégrale (1.2). Exceptée la méthodes d'euler rétrograde utilisant 2 noeuds et d'ordre 1. une méthode AM à p-pas est d'ordre $p+1$.

En générale, on n'utilise que les méthodes d'Adams à q-pas avec $q \leq 12$. Les schémas d'Adams sont utilisés pour les types de problème non-raide avec q varie de 1 à 12.

Méthode de différentiation rétrograde - BDF

Les méthode de différentiation rétrograde (BDF pour backward differentiation formulae) sont des méthodes à pas multiple implcites où la dérivée de y est approximée par celle du interpolant polynomial de degrée p aux $p+1$ noeuds $t_{n+1}, t_n, \ldots, t_{n-p}$ $(p > 0)$. Le schéma BDF est de la forme :

$$u_{n+1} = \sum_{j=0}^{p} a_j u_{n-j} + hb_{-1}f_{n+1},\ b_{-1} \neq 0 \tag{1.29}$$

les méthodes BDF sont zéros-stables sont celles avec $p \leq 5$, voir [13].

1.4.2 Consistence des méthodes à pas multiple

Dans cette partie la consistence et la stabilité des schémas à pas multiple sont présentés. Les démonstrations des théorèmes se trouvent par exemple dans [41] ou [27]. Les résultats d'analyse présentés dans cette section vont servir dans le chapitre 2 pour analyser le schéma $C(p,q,j)$. En effet, la notion consistence et stabilité du schéma $C(p,q,j)$ est très semblable qu'au cas d'un schéma classique.

La consistence d'un schéma à pas multiple peut être vérifiée avec certaines conditions algébriques concernant les coefficients du schéma numérique par le théorème suivant :

Théorème 1.3. *La méthode numérique (1.24) est consistente si et seulement si :*

$$\begin{array}{rcl} \sum_{j=0}^{p} a_j & = & 1 \\ -\sum_{j=0}^{p} j a_j + \sum_{j=-1}^{p} b_j & = & 1 \end{array} \tag{1.30}$$

Si de plus la solution exacte vérifie $y \in \mathcal{C}^{q+1}(I)$, $q \geq 1$, alors la méthode numérique est d'ordre q si et seulement si on a :

$$\sum_{j=0}^{p} (-j)^i a_j + i \sum_{j=-1}^{p} (-j)^{i-1} b_j = 1,\ i = 1, \ldots, q \tag{1.31}$$

Démonstration. La preuve du théorème repose sur le développement de Taylor au premier ordre de $y_{n-j},\ f_{n-j},\ j = 0, \ldots, p$ au temps t_n, où tous les $y_k,\ k \leq n$ sont des solutions exactes introduites dans le schéma numérique jusqu'au pas de temps t_n. On écrit :

$$\begin{array}{rcl} y_{n+1} & = & y_n + h y'_n + \dfrac{h^2}{2!} y''_n + \mathcal{O}(h^3) \\ y_{n-j} & = & y_n + (-jh) y'_n + \dfrac{(-jh)^2}{2!} y''_n + \mathcal{O}(h^3) \\ f_{n+1} & = & f_n + h f'_n + \dfrac{h^2}{2!} + \mathcal{O}(h^3) \\ f_{n-j} & = & f_n + (-jh) f'_n + \dfrac{(-jh)^2}{2!} + \mathcal{O}(h^3) \end{array}$$

et les remplaçons dans (1.30). L'erreur de consistence définie dans (1.25) est en effet en fonction de $\mathcal{O}(h)$ si et seulement si (1.30) sont vérifiés. La suite du théorème se déroule avec le développement de Taylor en augmentant l'ordre jusqu'à l'ordre voulu. □

1.4.3 La condition de racine

Une méthode à pas multiple linéaire utilisée pour résoudre le problème de test (1.21) s'écrit sous la forme :

$$u_{n+1} = \sum_{j=0}^{p} a_j u_{n-j} + h\lambda \sum_{j=0}^{p} b_j u_{n-j} \tag{1.32}$$

notamment on retrouve les méthodes d'Adams et BDF dans la section 1.4. Pour étudier la zéro-stabilité et la convergence des schémas à pas multiple, on fait intervenir son polynôme caractéristique suivant :

$$\Pi(r) = \rho(r) - h\lambda\sigma(r) \tag{1.33}$$

où :

$$\rho(r) = r^{p+1} - \sum_{j=0} a_j r^{p-j}, \quad \sigma(r) = b_{-1} r^{p+1} + \sum_{j=0}^{p} b_j r^{p-j}$$

qui sont le premier et deuxième polynôme caractéristique, $\Pi(r)$ est le polynôme caractéristique associé à la méthode (1.33). Une racine r du polynôme caractéristique est dépendante de $h\lambda$. Les racines du premier polynôme caractéristique sont des racines correspondantes à la valeur $h\lambda = 0$. Remarquons que si la première condition sur la consistence (1.30) est vérifiée, alors $r = 1$ est une racine de $\rho(r) = 0$.

Définition 1.12. ***(Condition de racine)*** *Soient la méthode numérique (1.32) et $\Pi(r)$ son polynôme caractéristique associé. Soient $r_1, \ldots, r_m$ les racines de :*

$$\Pi(r) = 0$$

La méthode numérique vérifie la condition de racine si :

$$\begin{array}{l} |r_j| \leq 1, \quad j = 0, \ldots, p; \\ si\ |r_j| = 1, alors\ que\ \rho'(r) \neq 0 \end{array} \tag{1.34}$$

La dernière équation dit que si 1 est une racine, alors qu'elle est simple.

1.4.4 Stabilité et convergence des méthodes à pas multiple

On va expliquer dans cette partie la relation entre la condition de racine introduite précédemment et la stabilité d'un schéma à pas multiple. On introduit la notion de zéro-stabilité d'un schéma à pas multiple :

Définition 1.13. ***(Zéro-stabilité d'une méthode à pas multiple)*** *La méthode à pas multiple est zéro-stable si $\exists h_0 > 0$, $\exists h \in]0, h_0]$, $\forall \varepsilon > 0$ suffisamment petit, si $|\delta_k| \leq \varepsilon$, $0 \leq k \leq N_h$, alors :*

$$|z_n^{(h)} - u_n^{(h)}| \leq C\varepsilon, \quad 0 \leq n \leq N_h \tag{1.35}$$

où N_h est le nombre total des noeuds de discrétisation et $z_n^{(h)}$ et $u_n^{(h)}$ sont respectivement solutions aux problèmes :

$$\left\{ \begin{array}{lcl} z_n^{(h)} & = & \sum_{j=0}^{p} a_j z_{n-j}^{(h)} + h \sum_{j=-1}^{p} b_j f(t_{n-j}, z_{n-j}^{(h)}) + h\delta_{n+1} \\ z_k^{(h)} & = & w_k^{(h)} + \delta_k, \quad k = 0, \ldots, p \end{array} \right. \tag{1.36}$$

$$\left\{ \begin{array}{lcl} u_n^{(h)} & = & \sum_{j=0}^{p} a_j u_{n-j}^{(h)} + h \sum_{j=-1}^{p} b_j f(t_{n-j}, u_{n-j}^{(h)}) \\ u_k^{(h)} & = & w_k^{(h)}, \quad k = 0, \ldots, p \end{array} \right. \tag{1.37}$$

où $w_0^{(h)} = y_0$ et $w_k^{(h)}$, $p > k > 0$ sont des valeurs initiales obtenues par une autre méthode numérique.

On a le théorème suivant concernant la zéro-stabilité et la condition de racine :

Théorème 1.4. *Si la méthode numérique à pas multiple (1.26) est consistente, alors il y a l'équivalence de la condition de racines et la zéro-stabilité*

La preuve du théorème se trouve dans [41]. On peut donc appliquer ce dernier pour quelques familles de méthodes vues précédemment

- Les méthodes à un pas consistent admettent seulement une racine égale à 1, elle est simple donc la méthode est zéro-stable.
- Les méthodes d'Adams où les polynômes caractéristique s'écrit sous forme $\Pi(r) = r^{p+1} - r^p$ admettent 1 comme racine simple et 0 racine d'ordre p. Elles sont zéro-stables.
- Les méthodes BDF sont zéro-stables sous condition que $p \leq 5$ (voir [27, chap. 3]).

Le théorème suivant caractérise la convergence de la méthode à pas multiple :

Théorème 1.5. ***(Convergence des méthodes à pas multiple)*** *Une méthode à pas multiple consistente est convergente si et seulement si elle vérifie la condition de racines et l'erreur sur les données initiales tend vers zéro quand $h \to 0$. Par ailleurs, la méthode converge en ordre q si elle est consistente d'ordre q et l'erreur initiale tend vers zéro à l'ordre q.*

Concernant l'ordre de convergence des méthodes, Dahlquist [15] établit :

Théorème 1.6. ***(Première barrière de Dalhquist)*** *Il n'existe pas de méthode zéro-stable, utilisant q pas d'ordre plus grand que $q+1$ si q est impair, et pas plus grand que $q+2$ si q est pair.*

1.4.5 La stabilité absolue d'une méthode numérique à pas multiple

Lorsque l'on aborde la notion de la stabilité absolue, une méthode à pas multiple appliquée pour le problème test (1.21) est absolument stable si elle vérifie les conditions de racines strictes (c'est à dire la condition de racine dans la définition 1.12 sauf sans la racine $r = 1$) pour tout $h \leq h_0$.

Parmi les méthodes satisfaisant la stabilité absolue, on choisit de préférence les méthodes d'où la région de stabilité absolue la plus large possible (les méthodes **A-stables**, voir section 1.3.4). Les méthodes **A-stables** sont les mieux adaptées pour résoudre les problèmes de type raides abordés à l'introduction de cette thèse. On s'intéresse aussi aux les méthodes de type ϑ-stable.

Définition 1.14. ***(Méthode ϑ-stable)*** *Une méthode **ϑ-stable** est celle qui admet une région de stabilité suivante :*

$$\mathcal{A} = \{h\lambda \in \mathbb{C} : -\vartheta < \pi - \arg(h\lambda) < \vartheta\}$$

Le résultat suivant établi pour une méthode à pas multiple, la relation entre le nombre de pas et la stabilité :

Théorème 1.7. ***(Deuxième barrière de Dalhquist)*** *: Une méthode à pas multiple explicite ne peut pas être ni **A-stable**, ni **ϑ-stable**. Par ailleurs, il n'existe pas de méthode **A-stable** d'ordre plus grand que 2. Finalement, pour tout $\vartheta \in]0, \pi/2[$, il n'existe que deux méthodes à 3 et 4-pas qui sont **ϑ-stables**.*

Ce dernier théorème a été introduit dans les travaux de thèse de [15]. Ces résultats présentés dans ce chapitre vont servir comme outils d'analyse des schémas de découplage.

1.5 Extention pour le système d'EDO

Considérons le système des ODEs suivant :

$$\begin{cases} \mathbf{y}' &= \mathbf{F}(t, \mathbf{y}),\ t \in I \\ \mathbf{y}(t_0) &= \mathbf{y_0} \end{cases} \tag{1.38}$$

où $I = [t_0, T]$, $\mathbf{y} \in \mathbb{R}^n$ et $\mathbf{F}$ est une fontion de $\mathbb{R} \times \mathbb{R}^n$ à valeurs dans $\mathbb{R}^n$. La condition Lipchitzienne pour une fonction à plusieurs variables s'écrit :

Propriété 1.1. *Soit $\mathbf{F} : \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}^n$ une fonction continue définie sur : $D = I \times \mathbb{R}^n$ d'òu I est fini. S'il existe une constante L telle que :*

$$\|\mathbf{F}(t, \mathbf{y}) - \mathbf{F}(t, \mathbf{z})\| \leq L\|\mathbf{y} - \mathbf{z}\|,\ \forall (t, \mathbf{y}), (t, \mathbf{z}) \in D \tag{1.39}$$

alors le problème (1.38) admet une solution unique $\mathbf{y}$, de plus cette solution est continue et différentiable par rapport à t sur I.

Remarque dans (1.39), la constante lipchitzienne dépend de la norme $\|\cdot\|$.

Considérons le cas d'un système dit linéaire et autonome suivant :

$$\mathbf{y}'(t) = A\mathbf{y}(t),\ A \in \mathbb{R}^{n \times n} \tag{1.40}$$

supposons que A admet n valeurs propres distinctes $\lambda_j,\ j = 1, \ldots, n$. La solution du système s'écrit :

$$y(t) = \sum_{i=0}^{n} C_j e^{\lambda_j t} \mathbf{v_j} \tag{1.41}$$

où les C_j sont des constantes dépendantes de la valeur initiale et $\{\mathbf{v_j}\}$ est la base de vecteur propre de A.

Concernant la stabilité absolue du système, il faut que toute les valeurs propres de A soient de partie réelle négatives. De plus, si A admet n valeurs propres distinctes, il existe une matrice Q non-singulière contenant n vecteurs propres telle que : $A = Qdiag(\lambda_j)_{1 \leq j \leq n} Q^{-1}$. Alors le système peut se transformer en introduisant la variable auxiliaire $z = Q^{-1}y$:

$$z' = diag(\lambda_j) z \tag{1.42}$$

Toutes méthodes numériques introduites au cas scalaire peuvent s'appliquer au cas d'un système odes. L'analyse pour le cas d'une EDO peuvent être étendue au système en considérant toutes les valeurs propres du système linéaire. Pour le système non-linéaire, la linéarisation permet d'étudier le comportement du système au premier ordre pour un intervalle de temps donné.

Ce chapitre introduit jusqu'ici les notions de la consistence, la stabilité et finalement la convergence d'une méthode numérique. L'analyse d'un schéma numérique pour le cas scalaire peut s'étendre au cas d'un système par la diagonalisation de ce dernier. Cependant, lorsque l'on découple le système par le schéma $C(p, p, j)$ dans le chapitre suivant, la structure diagonalisée du système n'est plus comme pour le cas présenté dans cette section. On propose alors à refaire l'étude de la convergence de ces méthodes.

Chapitre 2

Schéma $C(p, q, j)$ pour le découplage du système dynamique

Considérons un système d'équations différentielles :

$$y'(t) = f(t, y, z) \tag{2.1}$$
$$z'(t) = g(t, y, z) \tag{2.2}$$

Chaque équation de ce système définit un sous-système (ici un sous-système en y et en z). On propose d'utiliser une méthode numérique présentée dans le chapitre 1 pour trouver la solution numérique de chacun des sous-systèmes (2.1) et (2.2). On se place dans le contexte du traitement des problèmes de grande taille où le nombre d'inconnues (nombre de composantes de y respectivement de z) devient très important comparé à la capacité de calcul et de mémoire des calculateurs. On se propose de résoudre le système couplé (2.1)(2.2) à partir de la résolution découplée de chacun des sous-systèmes (2.1) respectivement (2.2) en considérant z respectivement y comme un paramètre du sous-système. Chaque sous-système est donc résolu de manière indépendante du moment où les paramètres sont à jour.

Pour le système couplé raide, on utilise de préférence une méthode implicite comme par exemple le schéma BDF à k-pas qui s'écrit sous la forme :

$$\sum_{i=0}^{k} \alpha_{n,i} u_{n+1-i} + h_n f(t_{n+1}, u_{n+1}, v_{n+1}) = 0 \tag{2.3}$$
$$\sum_{i=0}^{k} \alpha_{n,i} v_{n+1-i} + h_n g(t_{n+1}, u_{n+1}, v_{n+1}) = 0 \tag{2.4}$$

Le couplage du système se retrouve dans le schéma numérique parce qu'il y a la présence des termes v_{n+1} dans f et u_{n+1} dans g, contrairement à un schéma de type explicite qui est naturellement "découplé", le terme v_n dans f devenant un paramètre de la fonction.

Pour découpler la résolution du schéma implicite (2.3) et (2.4), on se propose de remplacer les termes de couplage par des approximations polynomiales. C'est sur ce principe que repose le schéma $C(p, q, j)$ que nous expliquerons dans la section suivante.

2.1 Découplage du système dynamique par $C(p, q, j)$ à pas de temps constant

Le schéma $C(p, q, j)$ introduit dans [19], consiste à découpler un système dynamique en sous-systèmes. Afin d'illustrer cette méthode, prenons l'exemple du schéma d'Euler implicite à pas de temps constant h (définition 1.2) pour sa simplicité d'écriture. On remplace v dans f et u dans g par leur extrapolations polynomiales v^* et u^* comme dan

$$u_{n+1} - u_n = hf(t_{n+1}, u_{n+1}, v^*_{n+1}) \tag{2.5}$$
$$v_{n+1} - v_n = hg(t_{n+1}, u^*_{n+1}, v_{n+1}) \tag{2.6}$$

le découplage consiste à trouver une approximation de v^*_{n+1} et u^*_{n+1}. Si $v^*_{n+1} = v_{n+1}$ et $u^*_{n+1} = u_{n+1}$, on retrouve le schéma couplé et implicite. Pour introduire le découplage, considérons u^*_{n+1} et v^*_{n+1} comme des approximations de u_{n+1} et v_{n+1} à partir de pas de temps précédents : par exemple, $u^*_{n+1} = (p+1)u_{n-p+1} - pu_{n-p}$ par extrapolation linéaire. On peut aussi introduire des extrapolations d'ordre supérieur comme celle d'ordre 3 qui utilise les 3 points correspondant aux temps t_{n-p+1}, t_{n-p} et t_{n-p-1} :

$$u^*_{n+1} = (p+1)(\frac{p}{2}+1)u_{n-p+1} - (p^2+2p)u_{n-p} + \frac{p^2+p}{2}u_{n-p-1} \tag{2.7}$$

En générale, le schéma $C(p, q, j)$ à pas de temps constant est défini avec :
- p : nombre de pas de délai
- q : nombre de pas de temps successivement extrapolé sur les même points
- j : nombre de points utilisés pour l'extrapolation

La figure (2.1) illustre le schéma de communication pour le schéma $C(5, 5, 2)$ entre deux sous-systèmes (code I et code II) pour les échanges des valeurs du temps t_{n-5+1} et t_{n-5} servant à calculer l'extrapolation dans l'autre sous-système. Dans son contexte original,

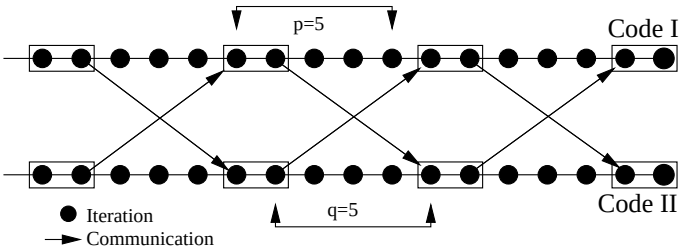


FIGURE 2.1 – Schéma de communication pour le schéma C(5, 5, 2)

le schéma $C(p, q, j)$ a été introduit pour la parallélisation sur les machines à calcul où le temps de communication devient important comparé au temps de la résolution locale de chaque sous-système. Le réseaux de communication entre les machines qui exécutaient chaque sous-système était lent. L'introduction d'un délai p pour le calcul de l'extrapolation permet de recouvrir le temps de communication par le calcul du sous-système. Le déséquilibre de l'architecture de calcul ne permettait pas de pouvoir traiter autre chose que des schémas à pas de temps régulier utilisant des communications synchrones. Le choix des paramètres p, q et j se faisant manuellement pour satisfaire le recouvrement des

communications par la charge de calcul, en essayant d'équilibrer la charge de calcul entre les sous-systèmes (voir [17])

Le contexte de calcul que nous visons est différent, on souhaite traiter des problèmes raides, par conséquent il faut introduire des pas de temps adaptatifs liés à la dynamique à chaque sous-système. D'autre part nous adressons à des architectures de calcul qui sont équilibrées en terme de latence et débit de communication et la puissance de calcul. Nous proposons donc une version améliorée du le schéma $C(p, q, j)$ par l'extension de l'extrapolation et par une gestion plus fine des communication entre les sous-systèmes (communication asynchrone). En effet une méthode de communication asynchrone assure la disponibilité des nouvelles informations aux sous-système, permettant d'accélérer le calcul.

2.2 Schéma $C(p, q, j)$ adaptatif à pas de temps variable

On propose d'étendre le schéma de découplage à pas de temps constant au cas de pas de temps adaptatif. La nouvelle technique consiste à effectuer extrapolation par un polynôme d'ordre j par la formule de différences divisées suivante :

$$u^*(t) = \sum_{k=0}^{j} u[t_0 \dots t_k](t - t_0) \dots (t - t_k)$$

avec les notations suivantes :

$$\begin{aligned} u[t_i] &= u(t_i) \\ u[t_i, \dots, t_{i+k}] &= \frac{u[t_i, \dots, t_{i+k-1}] - u[t_{i+1} \dots, t_{i+k}]}{t_i - t_{i+k}} \end{aligned}$$

les $u[t_i]$ sont les différences divisées de Newton qui se calculent par le tableau aux différences divisées suivant :

$$\begin{array}{llllll} t_0 & u[t_0] & & & & \\ \mathbf{t_1} & \mathbf{u[t_1]} & u[t_0, t_1] & & & \\ \mathbf{t_2} & \mathbf{u[t_2]} & \mathbf{u[t_1, t_2]} & u[t_0, t_1, t_2] & & \\ \vdots & \dots & & \dots & & \\ \mathbf{t_{j-1}} & \dots & & \dots & u[t_0, \dots, t_{j-1}] & \\ \mathbf{t_j} & \mathbf{u[t_j]} & & \dots & \mathbf{u[t_1, \dots, t_j]} & u[t_0, t_1, \dots, t_j] \end{array}$$

Cette technique d'extrapolation peut s'appliquer au pas de temps variable et permet au chaque sous-système d'avoir son propre pas de temps.

Montrons l'avantage de cette technique. En effet, le tableau des différences divisées permet de construire de manière incrémentale l'extrapolation (ou interpolation). Si au pas de temps t_{j+2}, on dispose des solutions aux points $t_0, \dots, t_{j+1}$, alors le polynôme d'interpolation de même ordre sur les points t_1 à t_{j+2} peut se construire en supprimant le point (t_0, v_0) et la ligne diagonale supérieure du tableau. Il suffit alors de calculer les

différences divisées de la ligne t_{j+1} :

$$\begin{array}{cccccc}
\mathbf{t_1} & \mathbf{u[t_1]} & & & & \\
\mathbf{t_2} & \mathbf{u[t_2]} & \mathbf{u[t_1,t_2]} & & & \\
\vdots & \dots & & \dots & & \\
\mathbf{t_j} & \dots & & & \mathbf{u[t_1,\dots,t_j]} & \\
t_{j+1} & u[t_{j+1}] & & \dots & u[t_2,\dots,t_{j+1}] & u[t_1,\dots,t_{j+1}]
\end{array}$$

D'un point de vue d'implémentation, lorsqu'une nouvelle valeur arrivant de l'autre sous-système, on décale le tableau de différences divisées d'un rang et on ne récalcule que la dernière ligne de ce dernier. Dans la pratique, cette technique est bien adaptée pour calculer les pas de temps adaptatifs.

Chaque sous-système ayant son propre pas de temps, la stratégie de communication synchrone n'est plus viable. Il est donc nécéssaire de mettre en oeuvre la gestion de communications asynchrones. Comme nous allons illustrer dans la section suivante.

La figure (2.2) montre un exemple de la méthode aux pas de temps adaptatifs.

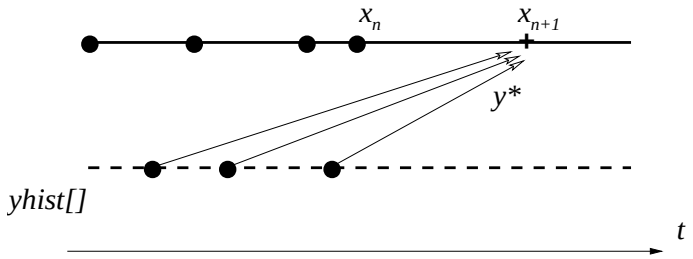


FIGURE 2.2 – Un schéma utilisant 3 points pour l'extrapolation

D'un point de vue d'implémentation informatique, la construction incrémentale du tableau de différences divisées est faite de manière optimisée. En effet, la mémoire allouée pour calculer l'extrapolation est minimisée à chaque instant.

1: **pour** Step **faire**
2: **tantque** *(Found a new message)* **faire**
3: ready to receive (by Irecv) and update if completed
4: **fin tantque**
5: Call SOLVER to get a new step
6: Send the last solution : make an Isend
7: **si** (History is out of date) **alors**
8: Wait for newer message
9: **finsi**
10: **fin pour**

En utilisant le découplage par la méthode $C(p,q,j)$, l'avantage est dans le coût de calcul :

- Gain en évaluation de la matrice Jacobienne
  - Coût de calcul pour une multiplication matrice-vecteur : n^3
  - Coût pour une Jacobienne "locale" : $n_y^3 + n_z^3$ ($n = n_y + n_z$)
- Gain en résolution non-linéaire dans les méthodes implicites

- Les schémas implicites mènent à la résolution non-linéaire de type $u_{n+1} = \mathcal{F}(u_{n+1})$. Cette résolution nécéssite une méthode itérative de type point-fixe ou la famille de méthode de Newton. La matrice Jacobienne locale est souvent mieux conditionnée que la matrice Jacobienne globale. Donc on peut espérer une meilleur performance pour la méthode itérative.
- L'extrapolation nécéssite $n(j-1)$ *flops* pour chaque mise-à-jour du système.

Dans algorithme du schéma $C(p, q, j)$ avec des communications asynchrones, les routines `ISend` et `IRecv` de la bibliothèque de communication de MPI sont utilisées. Le principe est le suivant : quand un sous-système avance d'un pas de temps, deux situations sont présents : soit un message correspondant à un pas de temps de l'autre système est disponible, soit aucun message a été reçuu. Dans le premier cas, on met à jour les différences divisées dans le tableau de Newton avec la nouvelle solution reçue. Autrement, on calcule l'extrapolation du prochain pas pour le sous-système courant à partir des données existantes dans le tableau de différences divisées. Cette communication asynchrone permettent d'éviter qu'un sous-système reste bloqué faute de disposer la solution la plus récente. Cependant, pour des raisons de stabilité, on limite le délai maximal. En effet dans le cas où le délai de communication devient assez important, un sous-système peut avoir de mauvaises extrapolations. Dans tel cas une mise à jour doit être forcée lorsque la limite est atteinte. Une stratégie consiste alternative consiste à redémarrer les sous-systèmes à des temps fixés. Cependant ces redémarrages doivent vérifier que la solution extrapolée d'un sous-système est très proche de la solution recherchée. Le premier contrôle consiste à vérifier que la solution extrapolées est proche de la solution calculée sur l'autre sous-système.

Nous nous proposons d'étudier les propriétés numériques du schéma de découplage par $C(p, q, j)$ dans la section suivante.

2.3 Analyse des schémas de découplage basés sur l'extrapolation

Les analyses sont appliquées pour le problème (2.1)(2.2) avec le schéma $C(p, q, j)$ à pas de temps constant utilisant une méthode numérique implicite à pas multiple comme intégrateur interne pour chaque sous système. Reprenons le schéma comme dans l'équation (2.3)(2.4) avec un pas de temps uniforme et en remplaçant les termes de découplage par son extrapolation polynomiale :

$$\sum_{j=0}^{k} \alpha_j u_{n+1-j} + h b_{-1} f(t_{n+1}, u_{n+1}, v^*_{n+1}) = 0 \tag{2.8}$$

$$\sum_{j=0}^{k} \alpha_j v_{n+1-j} + h b_{-1} g(t_{n+1}, u^*_{n+1}, v_{n+1}) = 0 \tag{2.9}$$

où les v^*_{n+1} et u^*_{n+1} sont des approximations polynomiales de v_{n+1} et u_{n+1}.

La notion sur la convergence de méthodes numériques de découplage similaires a été abordée sous le nom de "convergence des méthodes de relaxation asynchrone", "waveform

relaxation algorithm" comme dans [5], [44, 54]. Les résultats de convergence présentés dans ces travaux sont basés sur le critère de point fixe. La méthode itérative doit être définie par une fonction contractante. Dans ces méthodes, le découplage d'un système en sous-systèmes indépendants est réalisés par des approximations initiales des variables couplées. Cette prédiction initiales est corrigée au cours des itérations de point fixe. À chaque itération, ces approximations permettent de résoudre localement chaque sous-sytème, et les variables mises à jour servent à l'itération suivante des autres sous-systèmes, ceci jusqu'à convergence.

Néamoins ces techniques d'analyse de convergence des schémas découplés ne peuvent pas s'appliquer au schéma de découplage $C(p,q,j)$, car dans ce dernier, l'extrapolation n'est pas améliorée par un processus itératif. C'est pourquoi, nous nous proposons d'étudier dans la section suivante la consistence, la stabilité et la convergence de la méthode $C(p,q,j)$ en utilisant les outils du chapitre 1.

2.3.1 Consistence du schéma $C(p,q,j)$

Rappellons les résultats suivants pour étudier la consistence du schéma de découplage.

Erreur de l'interpolation

On considère l'erreur de troncature concernant l'interpolation polynomiale. Soient $y(t)$ une fonction réelle à variable réelle, $\Pi_j^y(t)$ son polynôme d'interpolation de degré j interpolant y aux $j+1$ noeuds $t_0,\ldots,t_j$ sur l'intervalle $I=[t_0,t_j]\subset\mathbb{R}$. Si la dérivée $j+1$-ième de y existe, l'erreur d'interpolation vaut :

$$E_j^y(t)=\Pi_j^y(t)-y(t)=w_{j+1}(t)\frac{y^{(j+1)}(\xi)}{(j+1)!},\ \forall t\in\mathbb{R} \tag{2.10}$$

pour un ξ dans le plus petit intervalle ouvert I dans $\mathbb{R}$ contenant tous les points $\{t_0,\ldots,t_j,t\}$, où le polynôme des noeuds est défini par $w_{j+1}(t)=\prod_{i=0}^{j}(t-t_i)$. Si les t_i sont régulièrement espacés, posons $h=\frac{t_j-t_0}{j}$, l'erreur d'*interpolation* $\varepsilon_{y,I}$ sur l'intervalle I est majorée par :

$$\varepsilon_{y,I}(h)=\max_I|E_j^y(t)|\leq M\frac{1}{4(j+1)}h^{j+1},\ \forall t\in I \tag{2.11}$$

où on a posé $M=\max_I|f^{(j+1)}(\xi)|$. De manière générale :

$$\varepsilon_{y,I}(h)=\mathcal{O}(h^{j+1}),\ \forall t\in I \tag{2.12}$$

Ce dernier résultat s'obtient en majorant le polynôme $w_{j+1}(t)$ sur $[t_0,t_j]$ et en faisant la remarque que ce polynôme admet le maximum soit sur $]t_0,t_1[$ soit sur $]t_{j-1},t_j[$.

Par ailleurs, si y est extrapolée aux noeuds $t_{j+1},\ldots,t_{j+p}$, l'erreur de l'extrapolation ε_{y,I_i} est alors évaluée sur le nouvel intervalle $I_i=[t_0,t_i],\ i=j+1,\ldots j+p$:

$$\varepsilon_{y,I_i}(h)=\max_{I_i}|E_j^y(t)|=\mathcal{O}(h^{j-i+2}),\ i=1\ldots p \tag{2.13}$$

Cette dernière formule (2.13) peut être vérifiée par récurrence sur p. Pour $p = 1$, en supposant de plus que la dérivée $(j+1)$-ième de y existe sur $I_1 = [t_0, t_{j+1}]$, et en utilisant la même notation M pour : $M = \max_{I_1} |y^{(j+1)}(t)|$, alors pour tout $t \in I_1$ on peut écrire

$$\begin{aligned} |w_{j+1}(t)| &\leq |t_{j+1} - t_j||t_{j+1} - t_{j-1}| \dots |t_{j+1} - t_0| \\ &\leq h \cdot 2h \dots (n+1)h. \end{aligned}$$

donc :

$$\varepsilon_{y,I_1}(h) \leq Mh^{j+1} \tag{2.14}$$

La relation (2.13) est vraie pour $p = 1$. Remarquons que l'erreur d'extrapolation sur I_1 est de même ordre que l'erreur d'interpolation sur I à une constante près. Pour $p = 2$, on procède de la même manière que pour $p = 1$ mais sur l'intervalle I_2. La majoration du polynôme $w_{j+1}(t)$ sur I_2 donne

$$\varepsilon_{y,I_2} \leq 1!(t_{j+2} - t_0)Mh^j$$

d'où :

$$\varepsilon_{y,I_2}(h) = \mathcal{O}(h^j) \tag{2.15}$$

et ainsi de suite pour $I_3, I_4, \dots$.

Remarquons que plus l'extrapolation s'éloigne de l'intervalle initial, d'une part on perd l'ordre de précision, d'autre part la constante devient non contrôlable. Ce dernier est en facteur d'un factoriel et augmente avec la largeur de l'intervalle de l'extrapolation.

Un autre résultat utile à notre sujet est l'erreur suivante :

$$E_j^{y'}(t) = y'(t) - \Pi_j^{y'}(t)$$

qui est une erreur entre la dérivée et celle du polynôme d'interpolation. Elle est régi par (2.10). En dérivant la fonction w_{j+1}, vérifions que :

$$w'_{j+1}(t) = \sum_{m=0}^{j} \prod_{\substack{i=0 \\ i \neq m}}^{j} (t - t_i)$$

ce qui induit l'erreur sur les dérivées de l'interpolation aux noeuds suivante :

$$|E_j^{y'}(t_m)| = \left| \prod_{\substack{i=0 \\ i \neq m}}^{j} (t_m - t_i) \right| \frac{y^{j+1}(\xi)}{(j+1)!} = \mathcal{O}(h^j) \tag{2.16}$$

pour $m = 0, \dots, j$

Théorème de la consistence du schéma de découplage par extrapolation

On établit le théorème suivant concernant la consistence de la méthode de découplage associé à un schéma BDF comme intégrateur local.

Théorème 2.1. *Considérons la méthode $C(p,q,j)$ utilisant une extrapolation à $j+1$ points et une méthode numérique implicite de type BDF à l pas :*

$$u_{n+1} = \sum_{j=0}^{l} a_j u_{n-j} + hb_{-1} f(t_{n+1}, u_{n+1}, v^*_{n+1}) \tag{2.17}$$

$$v_{n+1} = \sum_{j=0}^{l} a_j v_{n-j} + hb_{-1} g(t_{n+1}, u^*_{n+1}, v_{n+1}) \tag{2.18}$$

où v^*_{n+1} et u^*_{n+1} sont des approximations polynomiales de degré j sur les noeuds $t_{n-p-j+1}$, $\ldots$, t_{n-p+1} ($j+1$ noeuds). Supposons que le schéma BDF utilisé est d'ordre $k \geq 1$. Alors le schéma $C(p,q,j) - BDF$ est consistent si $p = 1$. Dans les autre cas, il faut que j soit supérieur stricte qu'à p : $j > p$, et l'ordre consistence du schéma ainsi défini est $min(k, j+2-p)$.

Démonstration. On vérifie la consistence du schéma $C(p,q,j)$ en considérant l'erreur de troncature locale au pas de temps t_{n+1} comme dans la définition 1.10. Considérons (2.17) en premier, avec la solution exacte $(y(t), z(t))$ au problème (2.1)(2.2). En considérant l'erreur de troncature locale par

$$h\tau_u(h) = y_{n+1} - \sum_{i=0}^{l} a_i y_{n-i} - hb_{-1} f(t_{n+1}, y_{n+1}, z^*_{n+1}) \tag{2.19}$$

Pour écrire le développement de f, on utilise la **dérivée d'une fonction composée**, soit $f(x(t), y(t), z(t))$ une fonction à variable implicite réelle t. Sa dérivée par rapport à t s'écrit :

$$f = [\partial_1 f, \partial_2 f, \partial_3 f] \cdot [x', y', z']^\top$$

Par suite, on écrit le développement de Taylor pour :

$$\begin{aligned} y_{n+1} &= y_n + hy'_n + \frac{h^2}{2!} y''_n + \mathcal{O}(h^3) \\ y_{n-i} &= y_n + (-ih) y'_n + \frac{(-ih)^2}{2!} y''_n + \mathcal{O}(h^3) \end{aligned}$$

et en particulier :

$$f(t_{n+1}, y(t_{n+1}), z^*(t_{n+1})) = f(t_n, y_n, \Pi^z_j(t_n)) + h\left(\partial_1 f_n + \partial_2 f_n y'_n + \partial_3 f_n \Pi^{z\prime}_p(t_n)\right) + \mathcal{O}(h^2)$$

Remplaçons ces derniers résultat dans (2.19) en ignorant les termes jusqu'à l'ordre 2. Pour vérifier que $\tau_u(h) \to 0$ lorsque $h \to 0$. Le coefficient associé au terme y_n vaut 0 par la consistence du schéma BDF (théorème 1.3). Ensuite vérifions :

$$hy'_n + hy'_n \sum_{i=0}^{l} a_i i - hb_{-1} f(t_n, y_n, \Pi^z_j(t_n)) = 0 \tag{2.20}$$

Si $p = 1$ alors $\Pi_j^z(t_n) = z_n$ (l'interpolation est exacte sur les noeuds). Par suite on a : $f(t_n, y_n, \Pi_j^z(t_n)) = f_n = y'_n$. Sous la première condition du théorème 1.3, le schéma est *consistent* . Si $p > 1$, il faut que l'extrapolation au t_n est au moins d'ordre 1 :

$$\Pi_j^z(t_n) - z_n = Ch + \mathcal{O}(h^2) \tag{2.21}$$

cette dernière équation permet d'écrire :

$$f(t_n, y_n, \Pi_j^z(t_n)) = f(t_n, y_n, z_n) + Ch\partial_3 f_n \cdot \Pi_j^z(t_n)' + \mathcal{O}(h^2)$$

Pour que le schéma reste *consistent*, (2.21) impose que $j > p$ (d'après (2.13)).

La suite de la preuve pour l'ordre de consistence du schéma se déroule en annulant les termes d'ordre supérieur dans le développement de Taylor précédents. Par exemple, pour le terme d'ordre 2 faisant intervenir la dérivée seconde, en faisant l'appel à des formules comme celle dans (2.16).

Le raisonnement similaire peut s'appliquer pour l'expression de l'erreur de troncature locale pour (2.18).

Finalement, l'ordre de consistence est atteint par le minimum entre l'ordre de l'extrapolation (et ses dérivées correspondantes) et l'ordre du schéma BDF. □

Le dernier théorème nous indique que si on veut continuer à atteindre la précision du schéma numérique locale (BDF d'ordre k), selon le retard p, il faut utiliser au moins $p+2$ points pour l'extrapolation.

2.3.2 Zéro-Stabilité du schéma de découplage

La condition de zéro-stabilité introduite dans le chapitre 1 est étudiée pour le schéma $C(p,q,j)$. Le résultat de cette section présente essentiellement les travaux de [50].

La notion de zéro-stabilité d'un schéma de type (2.8)-(2.9) est généralisée sur la notion de la continuité de Lipchitz du problème de Cauchy :

$$\mathbf{y}' = f(t, \mathbf{y}), \quad \mathbf{y}(t) \in \mathbb{R}^n \tag{2.22}$$

Définition 2.1. ***(Condition de "Lipschitz" unilatérale ou one-sided Lipschitz-condition)*** *Un système EDO est monotoniquement stable (monotonically stable) si il vérifie la condition de Lipschitz unilatérale, pour toute u, v solution de (2.22) :*

$$\|u - v + \lambda\,(f(t,u) - f(t,v))\,\| \geq (1 + \lambda m(t))\|u - v\| \tag{2.23}$$

pour tout $t \geq t_0$, $\lambda \neq 0$ et $m(t)$ une fonction définie sur $\mathbb{R}$ positive, monotone et bornée.

C'est une alternative de la condition de Lipschitz dite "relaxée" car la constante de Lipschitz est remplacée par la fonction $m(t)$. Cette définition est aussi utilisée pour montrer la zéro-stabilité du schéma numérique Par exemple pour le schéma Euler implicite (BDF 1), (2.23) implique que :

$$\|u_n - v_n\| \leq M\|\delta_0\|$$

où M est une constante dépendante de $m(t)$ et du pas de temps h utilisé pour le schéma BDF1.

La définition précédente est généralisée au cas où le système d'EDO est partitionné en sous-systèmes, soit par exemple $\mathbf{y} = (y_1, y_2)^\top$:

$$\left\{ \begin{array}{rcl} y_1'(t) & = & f_1(t, \mathbf{y}(t)) \\ y_2'(t) & = & f_2(t, \mathbf{y}(t)) \end{array} \right. \tag{2.24}$$

Pour introduire la notion de stabilité d'un système partitionné, on utilise la notion de la norme logarithmique suivante :

Définition 2.2. *Norme Logarithmique- ou la dérivée logarithmique ou encore la mesure de la matrice : la norme logarithmique de l'operateur lineaire A est defini par*

$$\mu(A) = \lim_{h \to 0^+} \frac{\|I + hA\| - 1}{h} \tag{2.25}$$

La proposition suivante donne une condition suffisante pour la condition de stabilité monotonique au sens de la norme max pour un système partitionné :

Proposition 2.1. *En supposant les f_i continûment différentiables, alors le système partitionné (2.24) est* ***monotoniquement stable au sens de la norme max*** *s'il existe des normes $\|\cdot\|_i$ définiee sur chacun des sous-systèmes i, $i = 1, 2$ telles que :*

$$\|u_i - v_i + \lambda[f_i(t, \mathbf{u}) - f_i(t, \mathbf{v})]\|_i \geq \|u_i - v_i\|_i + \lambda \sum_{j=1}^{2} a_{ij}(t, \mathbf{u}, \mathbf{v}) \|u_j - v_j\|_j \tag{2.26}$$

pour tout $\mathbf{u} = (u_1, u_2)$ et $\mathbf{v} = (v_1, v_2)$ solutions au problème (2.24), pour tout $t \geq t_0$, $\lambda \leq 0$, les coefficients $a_{ij}(t, \mathbf{u}, \mathbf{v})$ peuvent être choisies par :

$$\left\{ \begin{array}{rcl} a_{ii}(t, \mathbf{u}, \mathbf{v}) & = & \mu_i \left(B_{ii} \right) \\ a_{ij}(t, \mathbf{u}, \mathbf{v}) & = & \|B_{ij}\|, \ i \neq j \\ B_{ij} & = & \int_0^1 \frac{\partial f_i}{\partial \mathbf{y_j}} \left(t, \theta \mathbf{u} + (1 - \theta)\mathbf{v} \right) d\theta \end{array} \right. \tag{2.27}$$

où μ_i est la norme logarithmique associée à la norme $\|\cdot\|_i$. Et la matrice $A = (a_{ij})_{1 \leq i \leq 2, 1 \leq j \leq 2}$ vérifie :

$$\mu_\infty[A] \leq 0$$

où μ_∞ est la norme max logarithmique.

La condition (2.27) est une **condition suffisante** pour la **stabilité monotonique** pour un système partitionné. Considérons ensuite le schéma dit Euler implicite découplé pour le système (2.24) :

$$\left\{ \begin{array}{rcl} u_{1,n+1} & = & u_{1,n} + h f_1(t, u_{1,n+1}, u^*_{2,n+1}) \\ u_{2,n+1} & = & u_{2,n} + h f_2(t, u^*_{1,n+1}, u_{2,n+1}) \\ u_{1,0} & = & y_1(t_0) \\ u_{2,0} & = & y_2(t_0) \end{array} \right. \tag{2.28}$$

ce schéma est un cas du schéma $C(p, q, j)$ avec BDF1 comme intégrateur interne. Skelboe [50] a montré le théorème suivant :

Théorème 2.2. *Soit le système partitionné(2.24) satisfaisant la condition de* ***stabilité mononotique au sens de la norme max*** *dans (2.26), pour toute* $\{u_{i,n}\}$ *obtenue avec (2.28) et* $\{v_{i,n}\}$ *obtenue avec :*

$$\begin{array}{lcl} v_{1,n+1} & = & v_{1,n} + h(f_1(t, v_{1,n+1}, v^*_{2,n+1}) + \delta_{1,n+1}) \\ v_{2,n+1} & = & v_{2,n} + h(f_2(t, v^*_{1,n+1}, v_{2,n+1}) + \delta_{2,n+1}) \\ v_{1,0} & = & y_1(t_0) + \delta_{1,0} \\ v_{2,0} & = & y_2(t_0) + \delta_{2,0} \end{array} \tag{2.29}$$

Si de plus on suppose que l'extrapolation des u^*_i *est une combinaison convexe (notamment une extrapolation linéaire ou celle avec* $u^*_{i,n+1} = u_{i,n}$*) alors :*

$$\sup_{i,n} \|u_{i,n} - v_{i,n}\| \leq \max_{\substack{1\leq i\leq 2 \\ n\geq 0}} \|\delta_{i,n}\| \tag{2.30}$$

Une application directe du théorème précédente est montré dans l'exemple d'un système linéaire :

$$\mathbf{y}'(t) = \mathbf{A}\mathbf{y}(t),\ \mathbf{y}(t_0) = \mathbf{y}_0,\ \mathbf{A} \in \mathbb{R}^n \tag{2.31}$$

et soit le système partitionné en p sous-systèmes :

$$y'_r(t) = \sum_{j=1}^{p} A_{rj} y_j(t) \tag{2.32}$$

Le système découplé utilisant le schéma Euler implicite et l'extrapolation de degré nul est **zéro-stable** si toute les valeurs propres de la matrice :

$$B = \begin{bmatrix} \mu(A_{11}) & & \|A_{ij}\| \\ & \ddots & \\ \|A_{ij}\| & & \mu(A_{qq}) \end{bmatrix} \tag{2.33}$$

sont situées dans le demi-plan négatif du plan complexe.

La zéro-stabilité du schéma $C(p,q,j)$ est vérifiée par une condition suffisante sur la **stabilité monotonique au sens de la norme max** pour le cas du schéma BDF1 avec une extrapolation d'ordre 0 ou linéaire. Il n'existe pas par ailleurs un résultat général concernant la zéro-stabilité pour un schéma d'ordre supérieur avec les extrapolations d'ordre plus grand que 1 comme remarqué dans [43]. Néamoins, [50, sec. 5] a conclu le résultat suivant :

Remarque 2.1. *Considérons un schéma implicite BDF à* k *pas (*$k > 1$*) utilisé avec le schéma de découplage comme dans (2.28). Si le schéma est d'ordre de précision* k'*, alors il faut appliquer ce schéma BDF pour au moins une fois sur le prédicteur (extrapolation des variables couplées* $u^*_{i,n+1}$*) et ensuite* $k' - 1$ *fois de correcteur pour préserver l'ordre de précision propre au schéma BDF utilisé. Par ailleurs, l'extrapolation des variables couplées d'ordre plus grand que* 1 *va détériorer fortement la stabilité du schéma utilisé.*

Dans la section suivante, nous allons abordons la notion de stabilité absolue du schéma $C(p,q,j)$.

2.3.3 La stabilité absolue du schéma $C(p,q,j)$

La stabilité absolue est étudiée sur le problème test suivant :

$$\begin{bmatrix} y(t) \\ z(t) \end{bmatrix}' = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} y(t) \\ z(t) \end{bmatrix} \tag{2.34}$$

Le schéma $C(p,q,j)$ utilisant un schéma BDF à k pas appliqué au système précédent prend la forme :

$$\begin{aligned} u_{n+1} &= \sum_{i=0}^{l} a_i u_{n-i} + hb_{-1}(a_{11}u_{n+1} + a_{12}v^*_{n+1}) \\ v_{n+1} &= \sum_{i=0}^{l} a_i v_{n-i} + hb_{-1}(a_{21}u^*_{n+1} + a_{22}v_{n+1}) \end{aligned} \tag{2.35}$$

On utilise une extrapolation d'ordre j sur les $j+1$ pas $t_{n-p+1-j}, \ldots, t_{n-p+1}$, par exemple :

$$v^*_{n+1} = \Pi^z_{j+1}(t_{n+1}) = \sum_{i=0}^{j} l_i(t_{n+1}) v_{n-p+1-i}$$

où les $l_i(t)$ sont des polynômes de base de Lagrange lorsque l'on écrit le polynôme interpolant v passant par les points considérés. Ensuite posons

$$U_i = \begin{pmatrix} u_i \\ v_i \end{pmatrix}$$

Soit $r = max\{k, j+p-1\}$, écrivons (2.35) sous forme :

$$\mathbf{U}_{n+1} \equiv \begin{pmatrix} U_{n+1} \\ \vdots \\ U_{n-r} \end{pmatrix} = \mathbf{A}\mathbf{U}_n \tag{2.36}$$

et la matrice $\mathbf{A} \in \mathbb{R}^{2r\times 2r}$ est définie par :

$$\mathbf{A} = \begin{pmatrix} \mathbf{0} & \mathbf{I} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \mathbf{0} \\ \mathbf{0} & \ldots & & \mathbf{0} & \mathbf{I} \\ \mathbf{M}_0 & \mathbf{M}_1 & \ldots & \ldots & \mathbf{M}_r \end{pmatrix} \tag{2.37}$$

la matrice $\mathbf{A}$ est de forme Frobenius par bloc, dans (2.37), les $\mathbf{0}$ sont des matrices des zéros de taille 2×2. $\mathbf{I}$ est la matrice d'identité de taille 2×2. Les matrices $\mathbf{M}_i \in \mathbb{R}^{2\times 2},\ i = 0, \ldots, r$ sont définies par :

$$\mathbf{M}_i = \begin{cases} a_i\mathbf{D}^{-1} & \text{si } 0 \le i & < p-1 \\ \mathbf{D}^{-1}(a_i\mathbf{I} + \mathbf{C}^j_{i-p+1}) & \text{si } p-1 \le i & \le \min\{p-1+j, k\} \\ \mathbf{D}^{-1}\mathbf{C}^j_{i-p+1} & \text{si } k+1 \le i & \le p-1+j \\ a_i\mathbf{D}^{-1} & \text{si } p+j \le i & \le k \end{cases} \tag{2.38}$$

avec :

$$\mathbf{D} = \begin{pmatrix} 1 - a_{11}hb_{-1} & 0 \\ 0 & 1 - a_{22}hb_{-1} \end{pmatrix} \tag{2.39}$$

$$\mathbf{C}_i^j = \begin{pmatrix} 0 & hb_{-1}a_{21}l_i(t_{n+1}) \\ hb_{-1}a_{12}l_i(t_{n+1}) & 0 \end{pmatrix},\ i = 0, \dots j \tag{2.40}$$

Le schéma (2.35) appliqué au problème (2.34) est absolument stable si la matrice $\mathbf{A}$ définie dans (2.37) admet un rayon spectral strictement plus petit que 1. Autrement, si λ_i est une valeur propre de $\mathbf{A}$, il faut que $|\lambda_i| < 1$, si $|\lambda_i| = 1$ alors qu'elle est simple. $\mathbf{A}$ admet son polynôme caractérisque suivant :

$$\rho(\lambda) = det(\mathbf{A} - \lambda\mathbf{I}) \tag{2.41}$$

Proposition 2.2. *Le polynôme caractéristique associé à la matrice* $\mathbf{A}$ *s'écrit sous la forme :*

$$\rho(\lambda) = \lambda^r det(P(\lambda)) \tag{2.42}$$

où :

$$P(\lambda) = \mathbf{M}_0 + \lambda\mathbf{M}_1 + \cdots + \lambda^r\mathbf{M}_r - \lambda^{r+1}\mathbf{I} \tag{2.43}$$

Démonstration. En considérant une décomposition LU par bloc de $\mathbf{A} - \lambda\mathbf{I}$. Sans perte de généralité sur r, prenons $r = 4$:

$$\begin{aligned} \mathbf{A} - \lambda\mathbf{I} &= \begin{pmatrix} -\lambda\mathbf{I} & \mathbf{I} & 0 & 0 & 0 \\ 0 & -\lambda\mathbf{I} & \mathbf{I} & 0 & 0 \\ 0 & 0 & -\lambda\mathbf{I} & \mathbf{I} & 0 \\ 0 & 0 & 0 & -\lambda\mathbf{I} & \mathbf{I} \\ \mathbf{M}_0 & \mathbf{M}_1 & \mathbf{M}_2 & \mathbf{M}_2 & \mathbf{M_4} - \lambda\mathbf{I} \end{pmatrix} \\ &= \begin{pmatrix} -\lambda\mathbf{I} & 0 & 0 & 0 & 0 \\ 0 & -\lambda\mathbf{I} & 0 & 0 & 0 \\ 0 & 0 & -\lambda\mathbf{I} & 0 & 0 \\ 0 & 0 & 0 & -\lambda\mathbf{I} & 0 \\ \mathbf{L}_{1,5} & \mathbf{L}_{2,5} & \mathbf{L}_{3,5} & \mathbf{L}_{4,5} & \mathbf{L}_{5,5} \end{pmatrix} \begin{pmatrix} \mathbf{I} & -\frac{1}{\lambda}\mathbf{I} & 0 & 0 & 0 \\ 0 & \mathbf{I} & -\frac{1}{\lambda}\mathbf{I} & 0 & 0 \\ 0 & 0 & \mathbf{I} & -\frac{1}{\lambda}\mathbf{I} & 0 \\ 0 & 0 & 0 & \mathbf{I} & -\frac{1}{\lambda}\mathbf{I} \\ 0 & 0 & 0 & 0 & \mathbf{I} \end{pmatrix} \\ &= \mathbf{LU} \end{aligned}$$

les blocs $L_{i,r}$ sont donnés par :

$$\begin{cases} L_{1,r+1} &= M_0 \\ L_{l,r+1} &= M_{l-1} + \dfrac{1}{\lambda}L_{l-1,r},\ l = 2 \dots r \\ L_{r+1,r+1} &= \dfrac{1}{\lambda^r}\left(\mathbf{M}_0 + \lambda\mathbf{M}_1 + \cdots + \lambda^r\mathbf{M}_r - \lambda^{r+1}\mathbf{I}\right) \end{cases} \tag{2.44}$$

et on remarque que le calcul du déterminant de $\mathbf{A} - \lambda\mathbf{I}$ revient à celui de la matrice triangulaire inférieure par bloc $\mathbf{L}$:

$$\begin{aligned} det(\mathbf{L}) &= (-\lambda)^{2r} det\left(\mathbf{L}_{r+1,r+1}\right) \\ &= det\left(\mathbf{M}_0 + \cdots + \lambda^r\mathbf{M}_r - \lambda^{r+1}\mathbf{I}\right) \end{aligned}$$

□

En effet, l'étude spectrale de la matrice $\mathbf{A}$ revient à un problème des polynômes matriciels définis dans (2.43). $P(\cdot) \in \mathbb{R}^{2\times 2}$ est défini comme un polynôme matriciel et problème associé consiste à trouver des valeurs de λ telles que la matrice P soit singulière (autrement dit une telle valeur λ est la valeur propre de $\mathbf{A}$), le problème est connu aussi sous le nom de problème polynomial à valeur propre (polynomial eigenvalues problem, voir [16]). Le polynôme caractéristique de $\mathbf{A}$ admet des racines identiques que celle de l'équation $det(P(\lambda)) = 0$. Par ailleurs, $det(P(\lambda))$ est de degré $2 \times (r+1)$. Nous propososons dans la suite une démarche étudier les racines d'un polynôme donné.

Critère de Jury

L'étude de la région de stabilité absolue vue jusqu'à présent consiste principalement à étudier les racines du polynôme caractéristique associé à la méthode numérique en question. Pour le cas des schémas vu dans le chapitre 1, les polynômes caractéristiques dépendent de $h\lambda \in \mathbb{C}$. Cependant, pour le schéma $C(p,q,j)$ appliqué au système test à deux composantes, son polynôme caractéristique dépend alors de tous les coefficients du système initial. L'étude des racines conduit à des inégalités forte complexes et qu'il n'existe pas de forme explicite pour la région de stabilité absolue recherchées. Nous mettons en garde que le polynôme caractéristique associé à une méthode

Néanmoins, pour un système où les coefficients sont donnés, on peut à posteriori étudier la stabilité absolue pour un schéma $C(p,q,j)$ utilisé.

Prenons un schéma *Euler implicite* utilisé avec l'interpolation linéaire à partir des données avec un délai de $p = 1$, en utilisant la proposition (2.42), on étudie le polynôme caractéristique suivant :

$$\rho(\lambda) = \lambda^4 + \alpha_1\lambda^3 + \alpha_2\lambda^2 + \alpha_3\lambda + a_4 \tag{2.45}$$

d'où :

$$\begin{array}{rcl} \alpha_1 & = & \frac{-2+ha_{11}+ha_{22}}{(-1+ha_{11})(-1+ha_{22})} \\ \alpha_2 & = & \frac{1-4\,h^2a_{12}\,a_{21}}{(-1+ha_{11})(-1+ha_{22})} \\ \alpha_3 & = & \frac{-4h^2a_{12}\,a_{21}}{(-1+ha_{11})(-1+ha_{22})} \\ \alpha_4 & = & \frac{-h^2a_{12}\,a_{21}}{(-1+ha_{11})(-1+ha_{22})} \end{array} \tag{2.46}$$

Le critere de Jury étudie la position des racines du polynôme caractéristique $\rho(z)$ à l'inférieur du cercle d'unité.

Le résultat présenté est une conséquence du *critère de Routh-Hurwitz (RH)* dans le [28, sec. I.13]. Le critère RH consiste à vérifier $Re(z_i) < 0$ où z_i est une racine du polynôme caractéristique. Cette vérification est basée sur deux résultats : le premier est le principe de Cauchy sur le module de la racine, le deuxième est l'application du théorème de Sturm. *Le critère de Jury* s'obtient en combinant le critère de RH avec la transformation $z - w$. La tranformation $z - w$ consiste à transformer le demi-plan complexe négatif sur le cercle d'unité. Le critère de Jury est une méthode élégante pour borner les racines du polynôme caractéristique.

Le calcul explicite ou par l'approche du rayon spectral (dans [19]) consiste à trouver explicitement les racines (en utilisant le calcul symbolique comme Maple) et majorer les

racines trouvées en module par 1. Le critère de Jury nécessite peu de calcul ($n-2$ lignes de déterminant de 2×2) et se ramène à des expressions algébriques.

Le critère de Jury est présenté comme suit. Soit $A(z)=a_0z^n+a_1z^{n-1}+\cdots+a_{n-1}z+a_n$ à valeur dans $\mathbb{C}$, on construit le tableau à $2n-3$ lignes :

`ligne`	z^n	z^{n-1}	z^{n-2}	...	...		...	z^1	z^0
1	a_n	a_{n-1}	a_{n-2}	...	a_{n-i}	a_{n-1-i}	...		a_0
2	a_0	a_1	a_2	...	a_i	a_{i+1}	...		a_n
3	b_{n-1}	b_{n-2}	...	...	b_{n-1-i}	b_{n-2-i}	...	b_0	
4	b_0	b_1		...	b_i	b_{i+1}	...	b_{n-1}	
5	c_{n-2}	c_{n-3}	...	...	c_{n-2-i}	c_{n-2-i}	...		
6	c_0	c_1		...	c_i	c_{i+1}	...		
	...	...	...						
$2n-3$	q_2	q_1	q_0						

Les deux premières lignes sont construites à partir des coefficients a_i du polynôme $A(z)$. Les lignes suivantes sont calculées de manière récursive :

$$b_i = \begin{vmatrix} a_n & a_{n-1-i} \\ a_0 & a_{i+1} \end{vmatrix} \tag{2.47}$$

$$c_i = \begin{vmatrix} b_{n-1} & b_{n-2-i} \\ b_0 & b_{i+1} \end{vmatrix} \tag{2.48}$$

En suite, les racines de $A(z)$ sont dans le cercle d'unité si et seulement si les conditions suivantes sont vérifiées :

$$\begin{aligned} A(1) &> 0 \\ (-1)^n A(-1) &> 0, \\ |a_n| &< a_0 \\ |b_{n-1}| &> |b_0| \\ &\dots \\ |q_2| &> |q_0| \end{aligned}$$

On condidère quelques exemples particuliers.

Cas du second ordre : le polynôme caractéristique est $A(z)=z^2+a_1z+a_2$. Les conditions de Jury se ramène à :

$$a_2 < 1 \tag{2.49}$$

$$A(1)=1+a_1+a_2 > 0 \tag{2.50}$$

$$A(-1)=1-a_1+a_2 > 0 \tag{2.51}$$

Cas du 3ème et 4ème ordre : le polynôme caractéristique est $A(z)=z^3+a_1z^2+a_2z+a_3$. On a :

$$\begin{aligned} |a_3| &< 1 \\ A(1)=1+a_1+a_2+a_3 &> 0 \\ A(-1)=-1+a_1-a_2+a_3 &< 0 \\ |b_2|>|b_0| \ : a_3^2-1 &< a_1a_3-a_2 \end{aligned}$$

Le polynôme d'ordre 4 : $A(z) = z^4 + a_1 z^3 + a_2 z^2 + a_3 z + a_4$ et on construit le tableau de Jury :

ligne	z^4	z^3	z^2	z^1	z^0
1	a_4	a_3	a_2	a_1	1
2	1	a_1	a_2	a_3	a_4
3	b_3	b_2	b_1	b_0	
4	b_0	b_1	b_2	b_3	
5	c_2	c_1	c_0		

Reprenons l'exemple du polynôme vu dans (2.45). Prenons par exemple $(a_{11}, a_{22}, a_{12}, a_{21}) = (-1, -2, -3, 1)$. Le calcul du critere de Jury consiste à vérifier les conditions suivantes :

- Dans le cas $p = 1$, on obtient une borne pour $h < 7.4686e - 2$.
- Dans le cas $p = 2$: $h < 1.0825e - 2$.
- Dans le cas $p = 3$ on obtient :$h < 3.557e - 3$

On résume dans le tableau suivant pour le cas du schéma Euler implicite avec l'interpolation linéaire

retard p	1	2	3
h	$7.46e-2$	$1.08e-2$	$3.55e-3$

On retrouve les resultats similaires avec le calcul du determinant et de valeurs propres comme dans [19].

2.4 CVODE - solver numérique pour les problèmes raides et non-raides

Dans le cadre de la thèse, les résultats numériques sont obtenus avec le solver CVODE (voir [12]). CVODE est une bibliothèque en language C pour résoudre numériquement les EDO. CVODE est capable de résoudre les problèmes raides ainsi que non-raides, en utilisant les méthodes Adams ou BDF à coefficients variables.

CVODE permet d'utiliser deux méthodes d'intégration :

- Adams (Adams-Moulton) méthodes d'ordre variable et pas de temps variable
- BDF d'ordre variable et pas de temps variable.

Ces méthodes sous la forme suivante :

$$\sum_{j=0}^{K_1} \alpha_{n,j} y_{n-j} + h_n \sum_{j=0}^{K_2} \beta_{n,j} y'_{n-j} = 0 \tag{2.52}$$

où $h_n = t - n - t_{n-1}$ est le pas de temps, $a_{n,0} = -1$. Pour traiter les problèmes non-raides, le schéma d'Adams-Moulton est utilisé avec $K_1 = 1$ et $K_2 = q$, $q \in [\![1, 12]\!]$. Pour traiter les problèmes de type raide, la formule BDF avec $K_1 = q$ et $K_2 = 0$, l'ordre q est tel que $q \in [\![1, 5]\!]$. Dans tous les deux cas de figure, le système non-linéaire :

$$G(y_n) \equiv y_n - h_n \beta_{n,0} f(t_n, y_n) - a_n = 0 \tag{2.53}$$

où on a posé $a_n \equiv \sum_{j>0} (\alpha_{n,j} y_{n-j} + h_n \beta_{n,j} y'_{n-j})$. Pour résoudre (2.53), CVODE propose deux choix de méthodes :

- Point-fixe : méthode itérative sans résolution du système linéaire.

– Newton : méthode itérative où interviennt la résolution d'un système algébrique linéaire à chaque itération.

La méthode de Newton (ou Newton modifiée) requiert une recherche de solution à l'équation :

$$M(y_{n(m+1)} - y_{n(m)}) = -G(y_{n(m)}) \tag{2.54}$$

où $M \approx I - h_n \beta_{n,0} \dfrac{\partial f}{\partial y}$. Plusieurs types de méthode de résolution sont proposés dans CVODE : les méthodes directes (LU) et les méthodes itératives (Krylov) avec l'option de pré-conditionnement.

À chaque pas d'intégration, le solver calcule une estimation de l'erreur locale pour ajuster le pas de temps et l'ordre du schéma d'intégration [8, 9]. Une autre propriété importante de CVODE est le choix d'utiliser un algorithme de détection de limite de stabilité pour les schémas BDF (STALD - STAbility Limit Detection).

Pour le problème de test $y'(t) = \lambda y(t)$ introduit dans la section 1.3.4, les schémas BDF d'ordre 1 et 2 est A-stable pour tous $\lambda \in \mathbb{C}^-$, c'est à dire sous telle condition ces schéma sont inconditionnelement stables. Cependant les méthodes BDF d'ordre 3 à 5-pour h fixé-ne sont stables que lorsque le produit $h\lambda$ se trouve dans le plan complexe exclus d'une portion limitée près de l'axe imgaginaire. Cette région d'instabilité s'élargie lorsque l'ordre du schéma augmente de 3 à 5 (les schémas d'ordre plus grand que 5 ne sont pas utilisés dans CVODE). Par conséquent, lorsqu'une méthode BDF de ces ordre est utilisée, si λ est assez près de l'axe imaginaire, le pas h doit être pris avec certain traitement particulier, voir détails dans [29].

L'algorithme STALD implémenté dans CVODE détecte la présence de la région de stabilité pour le pas de temps basée sur l'estimateur de l'erreur de troncature locale.

2.5 Validations numériques

2.5.1 Le prolbème HIRES

Notre premier cas de test présenté consiste dans le cas du problème HIRES(High Irradiance Response problem) provenant de la physiologie des plantes. Le problème HIRES se compose de 8 EDO suivantes :

$$\begin{aligned}
u_1' &= -1.71u_1 + 0.43u_2 + 8.32u_3 + 0.0007 \\
u_2' &= 1.71u_1 - 8.75u_2 \\
u_3' &= -10.03u_3 + 0.43u_4 + 0.035u_5 \\
u_4' &= 8.32u_2 + 1.71u_3 - 1.12u_4 \\
u_5' &= -1.745u_5 + 0.43u_6 + 0.43u_7 \\
u_6' &= -280.0u_6u_8 + 0.69u_4 + 1.71u_5 - 0.43u_6 + 0.69u_7 \\
u_7' &= 280.0u_6u_8 - 1.81u_7 \\
u_8' &= -280.0u_6u_8 + 1.81u_7
\end{aligned}$$

On découple le système par deux sous-systèmes indépendants de $\{u_1, u_2, u_4, u_6\}$ et de $\{u_3, u_5, u_7, u_8\}$.

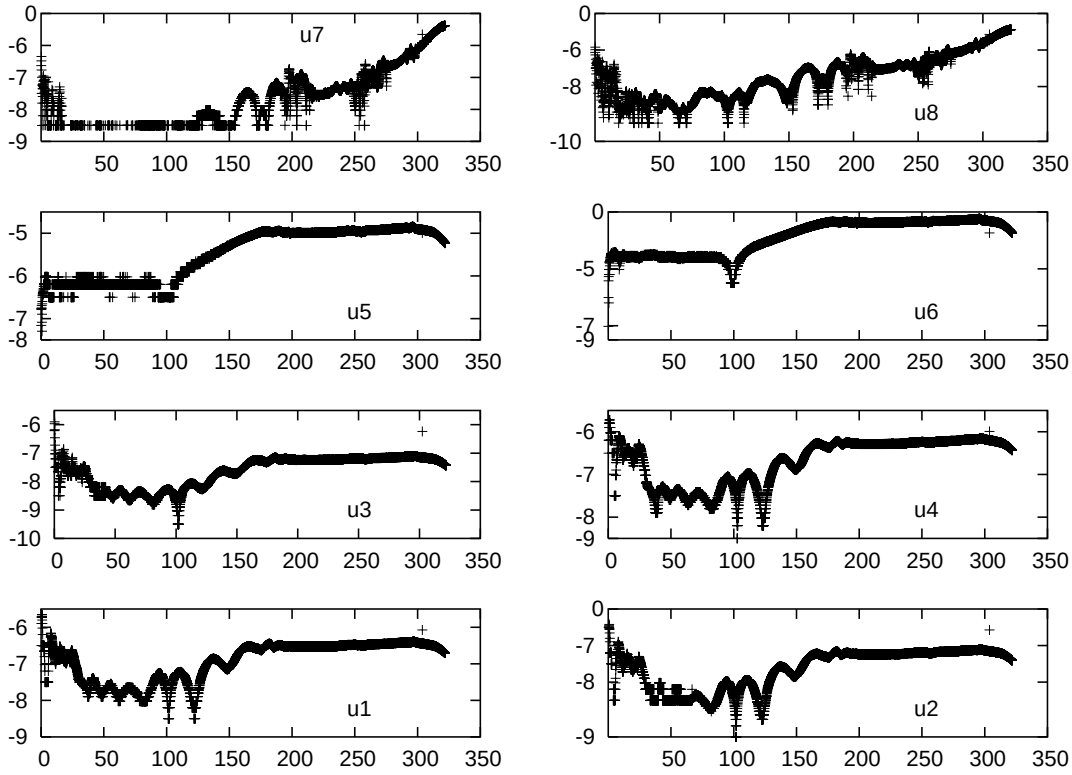


FIGURE 2.3 – Problème de HIRES : erreur relative comparée à la solution de référence en échelle logarithmique utilisant l'extrapolation d'ordre 5- solver's relative tolerance fixed at 10^{-8}

Dans la figure 2.3 : la solution de référence est comparée avec celle obtenue par le schéma $C(p, q, j)$ utilisant l'extrapolation d'ordre 5 à une solution de référence. On compare l'erreur relative sur un point donné est de l'ordre de 10^{-5}. La tolérance relative fixée au solveur local est fixée à 10^{-8}.

2.5.2 Le cas test EMEP

Le problème EMEP prend d'origine de la simulation chimique du problème EMEP MSC-W (Norwegian Meteorogical Institute-Oslo-Norway) sur le modèle réaction chimique d'ozone. Ce modèle de système dynamique comporte 140 réactions chimiques de 66 composantes. Le système obtenu est de l'échelle multiple et de type raide. Le solveur de référence utilisé utilise 10^{-9} comme tolérance relative.
Performance

- Sur l'intervalle de temps $[14400, 72000]$, utilisant le même tolérance relative
  - Solveur global : 3.71s
  - 2 solveurs : 3.05s

On considère le comportement de l'erreur relative sur les 6 variables d'échantillonnages selon différents ordres de l'extrapolation $j = 1$ à $j = 3$. Nous constatons que le résultat par l'extrapolation d'ordre $j = 4$ donne une solution fausse. Ceci peut être expliqué par l'instabilité de l'extrapolation. Nous remarquons que la meilleur solution obtenue est celle avec l'extrapolation d'ordre 3.

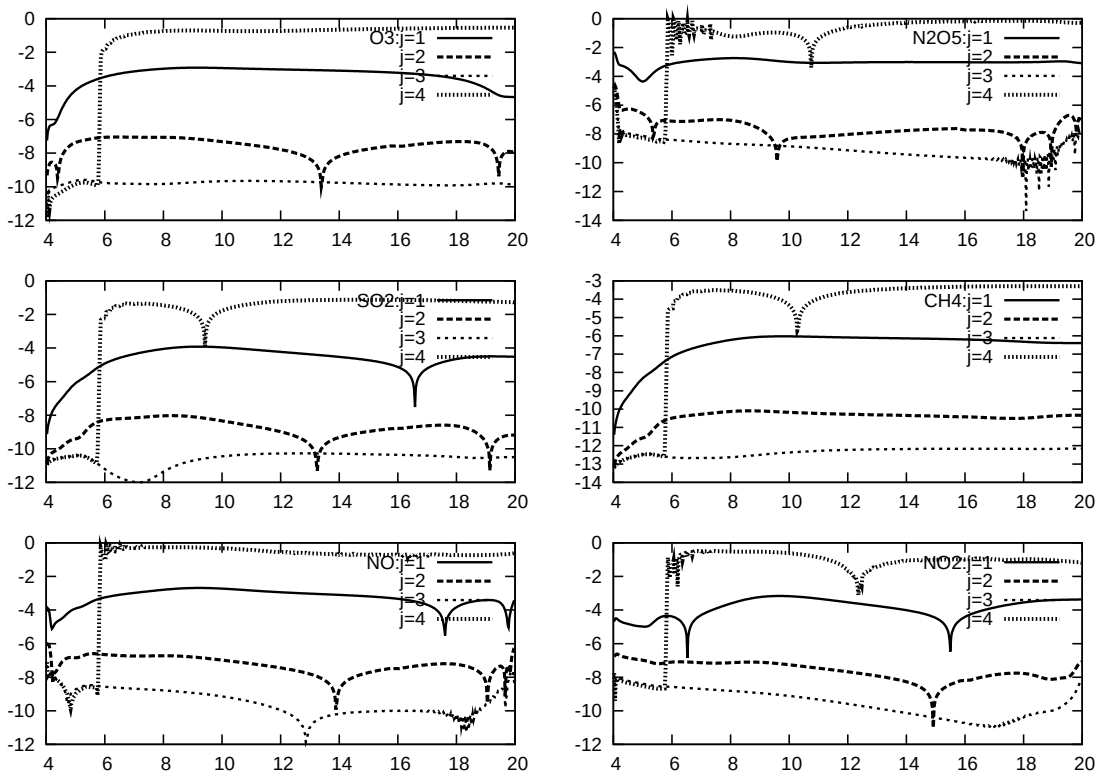


FIGURE 2.4 – Problème EMEP : erreur relative (en échelle logarithmique) comparée à la solution de référence avec une tolerance fixée à 10^{-9}

Deuxième partie

POD pour la résolution en parallèle des systèmes dynamiques

Chapitre 3

La décomposition orthogonale aux valeurs propres

3.1 Introduction à la POD et SVD

Les méthodes de résolution du système dynamique restent très coûteuses en terme de calcul pour de nombreux cas industriels. Souvent la dimension du problème à résoudre est trop importante par rapport aux capacités de calcul actuelles (vitesse d'horloge des processeurs, mémoire vive). Différentes approches ont donc vu le jour. Dans la premiere partie nous avions abordé les techniques de décomposition du système en sous-système, d'où les méthodes de résolution de type implicite découplés (Euler implicite découplé [44] , méthode dite "waveform relaxation" [44], Runge-Kutta multi-échelles [26], [43]). Dans ce qui suit, nous présentons les méthodes de réduction d'ordre du modèle appliquées à la résolution du système dynamique. Ces techniques sont largement utilisées dans plusieurs domaines d'application. La réduction d'ordre se divise en plusieurs sous familles de méthodes. Citons par exemple quelques méthodes de réduction d'ordre assez connues jusqu'à présent : la méthode de troncature "équilibrée" (*balanced truncation* method) et la réduction d'ordre par la POD. La première méthode malheureusement ne s'applique seulement que pour les systèmes aux entrées-sorties linéraires et est très coûteuse pour les problèmes comportant un grand nombre de variables (un système à plus de 10.000 variables devient incalculable). La méthode de réduction d'ordre par POD (avec la méthode des clichés "*snapshots method*") a l'avantage d'être utilisable pour les problèmes non-linéaires et admet un coût de calcul très avantageux par rapport à une résolution directe.

Les méthodes de découplage vue dans la première partie de la thèse, présentent l'avantage dans la facilité de mise en place pour le calcul parallèle. Par contre son inconvenient est l'absence d'une estimation de l'erreur du sous-système, exceptées certaines méthodes de type prédicteur-correcteur qui consistent à réitérer la résolution des sous-systèmes pour corriger l'écart introduit par le découplage (méthode waveform-relaxation), voir par exemple dans [44] ou [3].

Dans ce qui suit notre démarche consiste à combiner la réduction d'ordre du modèle avec le découplage des variables. Les méthodes de réduction d'ordre sont très avantageuses en terme de performance de calcul. Dans plusieurs cas, le modèle réduit dérivé

du système original admet une taille relativement faible comparée au système complet. Cependant, la difficulté réside dans la construction du modèle réduit. On suppose avoir suffisamment d'information (avoir une certaine dynamique suffisante sur le système complet) pour ensuite construire le système réduit. La nouvelle perspective consiste à combiner ces deux dernières techniques pour compenser l'inconvénient d'une par rapport à l'autre. Par conséquent, on utilise la réduction d'ordre du modèle pour construire les sous-systèmes en échangeant régulièrement l'information entre les sous-systèmes dans le but de garder de précision du modèle réduit.

3.2 Principe de la Décomposition Orthogonale Propre

Nous présentons la démarche de la POD appliquée dans un premier temps pour une fonction en générale (le cas continu). Ensuite nous allons voir comment appliquer numériquement la POD pour le cas discret.

3.2.1 La décomposition orthogonale propre d'une fonction

Considérons $\mathcal{X}$ un espace de Hilbert muni d'un produit scalaire $(\cdot,\cdot)_{\mathcal{X}}$, T est une constante positive. Soit $u \in \mathcal{L}^2([0,T];\mathcal{X})$ une fonction carrée intégrable de $t \in [0,T]$ à valeur dans $\mathcal{X}$. La démarche de rechercher la POD (appelé aussi analyse en composante principale, ou encore transformation de Karhunen-Loève) consiste à chercher la solution au problème d'optimisation suivant :

Prolème 3.1. *Trouver une base orthonormale $(\psi_i)_{i\geq 1}$ dans $\mathcal{X}$ telle que pour tous $l \geq 0$ et $(\varphi_k)_{k>1}$ quelconque dans $\mathcal{X}$, cette base doit satisfaire l'inégalité suivante :*

$$\int_0^T \left\| u(t) - \sum_{i=1}^{l} (u(t),\psi_i)_{\mathcal{X}}\ \psi_i \right\|_{\mathcal{X}}^2 dt \leq \int_0^T \left\| u(t) - \sum_{i=1}^{l} (u(t),\varphi_i)_{\mathcal{X}}\ \varphi_k \right\|_{\mathcal{X}}^2 dt \tag{3.1}$$

Une telle base $(\psi_i)_{i\geq 1}$ est appelée la base POD associée à u.. Le problème (3.1) peut s'écrire encore sous la forme d'un problème de minimisation :

$$\min_{(\psi_k)_{k\geq 1}} \int_0^T \left\| u(t) - \sum_{k=1}^{l} (u(t),\psi_k)_{\mathcal{X}}\ \psi_k \right\|_{\mathcal{X}}^2 dt \tag{3.2}$$

Dans le but de retrouver la base POD, nous introduisons l'opérateur $K(u) : \mathcal{X} \longrightarrow \mathcal{X}$:

$$K(u)\ :\ \varphi \longmapsto \frac{1}{T}\int_0^T (u(t),\varphi)_{\mathcal{X}}\ u(t)dt \tag{3.3}$$

et l'opérateur des corrélations temporelles :

$$\begin{aligned} \tilde{K}\ :\ \mathcal{L}^2([0,T]) &\longrightarrow \mathcal{L}^2([0,T]) \\ v(t) &\longmapsto \int_0^T \frac{1}{T}(u(s),u(t))_{\mathcal{X}}\ v(s)ds \end{aligned}$$

dans la dernière équation, notons le noyau de l'opérateur par : $\tilde{k}(s,t) = \frac{1}{T}(u(s), u(t))_{\mathcal{X}}$. Les opérateurs $K(u)$ et $\tilde{K}(u)$ sont auto-adjoints. Par ailleurs, $\tilde{K}$ est un opérateur de Hilbert-Schmidt, ceci provient du fait que le noyau $\tilde{k}$ est dans $\mathcal{L}^2([0,T])^2$ (voir [7, ch. 6]). Par conséquent, $\tilde{K}$ admet une famille dénombrable de valeurs propres positives ordonnées. Ces valeurs propres de l'opérateur $\tilde{K}$ seront ensuite notées dans l'ordre décroissant : $\tilde{\lambda}_1 \geq \tilde{\lambda}_2 \geq \cdots \geq \tilde{\lambda}_k \geq \cdots \geq 0$.

Lemme 3.1. *Les opérateurs K et $\tilde{K}$ ont les mêmes valeurs propres de mêmes multiplicités.*

Démonstration. En effet, si on note $\tilde{\lambda}$ comme une valeur propre de l'opérateur $\tilde{K}$ de multicité $r \geq 1$. Soit $(v_k)_{1\leq k\leq r} \in \mathcal{L}^2([0,T])$ l'ensemble des vecteurs propres orthonormaux associés, on définit l'ensemble $(\psi_k)_{1\leq k\leq r}$ par $\psi_k = \frac{1}{\sqrt{T\lambda_k}} \int_0^T u(t)v_k(t)dt$. Alors les vecteurs ψ_k sont aussi vecteurs propres de K. Inversement, pour λ et (ψ_k) valeurs et vecteurs propres de K, la valeur propre est la même pour $\tilde{K}$ et les vecteurs propres associé sont défini par $v_k = \frac{1}{\sqrt{\lambda T}}(u, \psi_k)_{\mathcal{X}},\ v_k \in \mathcal{L}^2([0,T])$. En outre, on suppose que $u \neq 0$ et la base des vecteurs propres (ψ_k) associés aux valeurs propres $\lambda_k \neq 0$ est appelée base POD de u. □

Définition 3.1. *Les valeurs propres non nulles de l'opérateur K classées dans l'ordre décroissant sont appelées les valeurs propres de la POD associée à la fonction u. L'ensemble des vecteurs propres $(\psi_k)_{k\geq 1},\ \psi_k \in \mathcal{X}$ orthonormés de K correspondants à ces valeurs propres sont des vecteurs propres de la base POD de u.*

La base des vecteurs propres de l'opérateur K est en effet une solution de (3.2). Par ailleurs, on a :

$$\frac{1}{T}\int_0^T \left\| u(t) - \sum_{k=1}^{M} (u(t), \psi_k)_{\mathcal{X}}\, \psi_k \right\|_{\mathcal{X}}^2 dt = \sum_{k=M+1}^{+\infty} \lambda_k \xrightarrow[M\to\infty]{} 0 \tag{3.4}$$

Une base POD est optimale pour représenter $u(t)$, on réécrit $u(t)$ dans la nouvelle base :

$$u(t) = \sum_{k=1}^{\infty} (u(t), \psi_k)_{\mathcal{X}} \psi_k \tag{3.5}$$

Remarque 3.1. *Dans les cas où $\mathcal{X} = [\mathcal{L}^2(\Omega)]^d$ où Ω un domaine borné dans $\mathbb{R}^n$, muni d'un produit scalaire classique. Supposons que u est assez régulière, alors l'opérateur K est appelé opérateur de corrélations spatiales :*

$$\begin{aligned} K(u) \circ \varphi &= \frac{1}{T}\int_0^T u(x,t) \int_\Omega u(\hat{x},t)^\top \varphi(\hat{x}) d\hat{x}\, dt \\ &= \int_\Omega k(\hat{x}, x)\varphi(\hat{x}) d\hat{x} \end{aligned} \tag{3.6}$$

avec $k(\hat{x}, x) = \frac{1}{T}\int_0^T u(x,t)u(\hat{x},t)^\top dt$ appelé le noyau de l'opérateur K.

Dans le cas où on a un ensemble de fonctions $(u_i(t))_{1\leq i\leq n},\ u_i(t) \in \mathcal{L}^2([0,T];\mathcal{X})$. On définit l'opérateur K par :

$$K\ :\ \varphi \longmapsto \sum_{i=1}^{n} \frac{1}{T}\int_0^T (u_i(t),\varphi)_{\mathcal{X}} u_i(t)dt \tag{3.7}$$

3.2.2 La POD discrète et la méthode des clichés

Le résultat précédent d'une POD continue peut s'étendre à la notion de la POD discrète si on regarde la fonction u comme une fonction constante par morceaux. Dans ce dernier cas on appelle la méthode des clichés.

Proposition 3.1. *On considère une constante $\delta_t > 0$ et une subdivision homogène de $[0,T]$: $0 = t_0 < \cdots < t_N = T,\ t_{i+1} - t_i = \delta_t$. Soit u une fonction continue par morceaux défini par $u(t) = u_{i+1} \in \mathcal{X},\ t \in [t_i, t_{i+1}]$. Les valeurs $(u_i)_{1\leq i\leq N}$ sont appelée les clichés. Alors une POD continue de u correspond à une POD discrète des N clichés u_i. Dans ce cas, la POD de u correspond à la méthode des clichés.*

Pour tout $\varphi \in \mathcal{X}$, l'opérateur K défini dans la section précédente devient : $K(u)\circ\varphi = \frac{1}{N}\sum_{i=1}^{N}(u_i,\varphi)_{\mathcal{X}}u_i$, où l'intégrale est remplacée par la somme discrète des u_i. Ensuite la méthode des clichés consiste à contruire la matrice de corrélation K_{cor} définie par :

$$K_{cor} = \left(\frac{1}{N}(u_i,u_j)_{\mathcal{X}}\right)_{1\leq i,j\leq N} \tag{3.8}$$

Notons $v_k = (v_{i,k})_{1\leq i\leq N}$ *in* $\mathbb{R}^N,\ k = 1,\ldots,N$, un ensemble des vecteurs propres orthonormés de la matrice K_{cor}. La base orthonormale des vecteurs propres de l'opérateur $K(u)$ est obtenue en posant :$\psi_i = \frac{1}{\sqrt{N\lambda_k}}\sum_1^N v_{i,k}u_i$. En conclusion, la méthode des clichés permets de calculer la POD de la fonction constante par morceaux approximant la fonction continue.

3.2.3 POD et SVD(Singular Value Decomposition)

Dans cette partie, on montre les relations entre la décomposition orthogonale propre et la décomposition en valeur singulière connue pour un opérateur linéaire dans l'espace de dimension fini. On considère la POD discrète d'une fonction dans un espace de Hilbert réel $\mathcal{X}$ de dimension finie. C'est à dire on cherche la POD à partir d'une base de donées numériques. En effet, la POD est équivalente à une décomposition en valeur singulière (SVD). Les résultats présentés dans cette partie se trouvent dans [53].

Soit $u(t)$ dans un espace de Hilbert de dimension finie L : $\mathcal{X}$ avec la base notée par $\{\mathcal{X}_i\}_{i=1\ldots L}$. Soit un ensemble des données $X = (u(t_j))_{j=1\ldots N}$ dans $\mathcal{X}^N$. Les $u(t_j)$ sont des clichés connus numériquement et peuvent s'exprimer dans la base de $\mathcal{X}$ par :

$$\forall j = 1\ldots N\ : u(t_j) = u_j = \sum_{i=1}^{L} y_{i,j}\mathcal{X}_i \tag{3.9}$$

où $y_{i,j}$ sont des coefficients de u dans la base de $\mathcal{X}$. On note y_j la représentation vectorielle de u_j : $y_j = (y_{1,j}, \ldots, y_{L,j})^\top \in \mathbb{R}^L$. On défini ainsi la matrice des clichés comme étant :

$$\mathbb{Y} = \begin{pmatrix} y_{1,1} & \cdots & y_{1,N} \\ \vdots & & \vdots \\ y_{L,1} & \cdots & y_{L,N} \end{pmatrix} \tag{3.10}$$

Le processus de rechercher la base optimale pour représenter les données $\mathbb{Y}$ peut se résumer sous la forme d'un problème de minimisation comme dans (3.1). On reformule ce dernier comme suit :

Prolème 3.2. *Trouver $l \leq P$ vecteurs orthonormaux $\{\psi_i\}_{i=1\ldots l}$ dans $\mathbb{R}^N$ minimisant :*

$$J(\psi_1, \ldots, \psi_l) = \sum_{j=1}^{n} \left\| y_j - \sum_{i=1}^{l} (y_j, \psi_i)_{\mathcal{X}} \psi_i \right\|^2, \tag{3.11}$$

sous condition que : $< \psi_i, \psi_k >_{\mathcal{X}} = \delta_{ij},\ 1 \leq i \leq l,\ 1 \leq j \leq i$, avec δ_{ij} vaut 1 si $i = j$ et 0 autrement.

Une solution au problème précédent peut s'obtenir en introduisant la fonctionnelle de Lagrange suivante :

$$L(\psi_1, \ldots, \psi_l, \lambda_{11}, \ldots, \lambda_{ll}) = J(\psi_1, \ldots, \psi_l) + \sum_{i,j=1}^{l} \lambda_{ij} (\psi_i^\top \psi_j - \delta_{ij}) \tag{3.12}$$

où :

$$\frac{\partial L}{\partial \psi_i}(\psi_1, \ldots, \psi_l, \lambda_{11}, \ldots, \lambda_{ll}) = 0 \in \mathbb{R}^N,\ \forall i = 1, \ldots, l \tag{3.13}$$

$$\frac{\partial L}{\partial \lambda_{ij}}(\psi_1, \ldots, \psi_l, \lambda_{11}, \ldots, \lambda_{ll}) = 0 \in \mathbb{R},\ \forall i, j = 1, \ldots, l \tag{3.14}$$

La condition d'optimalité (3.13)-(3.14) donne :

$$\sum_{j=1}^{n} y_j (y_j^\top \psi_i) = \lambda_{ii} \psi_i \text{ et} \lambda_{ij} = 0,\ i \neq j \tag{3.15}$$

$$\psi_i^\top \psi_j = 0 \text{ si } i \neq j, \text{ et 1 sinon}$$

Par suite, on remarque qu'en utilisant la notation dans (3.10), (3.15) est un problème aux valeurs propres :

$$\mathbb{Y}\mathbb{Y}^\top \psi_i = \lambda_i \psi_i,\ i = 1, \ldots, l \tag{3.16}$$

La condition d'optimalité du problème aux valeur propres est à la fois nécessaire et suffissante. Pour faire le lien avec la décomposition aux valeurs singulières. On rappelle la SVD d'une matrice comme suit :

Propriété 3.1. *Toute matrice peut se s'écrire sous une forme diagonale par une pré et post-multiplication avec des matrices unitaires appropriées. Soit $A \in \mathbb{R}^{m\times n}, m \geq n$ de rang $k \leq n$, ils existent deux matrices unitaires $\mathbb{U} = [u_1, \ldots, u_m] \in \mathbb{R}^{m\times m}$ et $\mathbb{V} = [v_1, \ldots, v_n] \in \mathbb{R}^{n\times n}$ telles que :*

$$\mathbb{U}^H A\mathbb{V} = \Sigma = diag(\sigma_1, \ldots, \sigma_m) \tag{3.17}$$

$\sigma_1 \geq \cdots \geq \sigma_k > \sigma_{k+1} = \cdots = \sigma_m = 0$. Cette décompostion est appelée décomposition en valeurs singulières (SVD) de la matrice A.

En appliquant la SVD pour la matrice des clichés $\mathbb{Y}$ dans l'équation (3.16), notons que $1 \leq i \leq N : \mathbb{Y}v_i = \sigma_i u_i,\ \mathbb{Y}^\top u_i = \sigma_i v_i$. La base POD peut alors s'obtenir en posant : $\psi_i = u_i$ et $\lambda_i = \sigma_i^2 > 0,\ i = 1, \ldots, k$.

En générale, le spectre obtenu par décomposition en valeur singulière de A est ordonné de manière décroissante et, de plus les valeurs singulières décroisent rapidement vers zéro. Pour reconstruire la matrice A, on tronque généralement la SVD à la k-ième plus grande valeur singulière ($k \leq m$) :

$$A \approx \sum_{i=1}^{k} \sigma_i u_i v_i^\top \tag{3.18}$$

Cette approximation admet la propriété suivante :

Proposition 3.2. ***L'optimalité au sens de la norme*** 2
Soit $A \in \mathbb{R}^{n\times m}$, trouver $X \in \mathbb{R}^{n\times m}$ de rang k, $k < rank(A)$, minimisant la la distance $\|A - X\|$. En effet, on a (voir [11] ou dans [31]) :

$$\min_{rank(X)\leq k} \|A - X\|_2 = \sigma_{k+1}(A) \tag{3.19}$$

un tel minimiseur $\hat{X}$ est obtenu par la troncature basée sur le spectre de la SVD aux k premières valeurs singulières : $\hat{X} = \sigma_1 u_1 v_1^\top + \sigma_2 u_2 v_2^\top + \ldots \sigma_k u_k v_k^\top$

où la distance entre X et A est égale au reste de la somme du spectre de la SVD de A.

Dans la pratique, le calcul de la POD discrète à partir d'une matrice des clichés $\mathbb{X} \in \mathbb{R}^{m\times n},\ m > n$ se ramène dans les cas suivants :

- Soit calculer la SVD de $\mathbb{X} \in \mathbb{R}^{m\times n} : \mathbb{X}v_i = \sigma_i u_i$.
- Soit trouver solutions du problème à valeurs propres de $\mathbb{X}^\top\mathbb{X} \in \mathbb{R}^{m\times m} : \mathbb{X}^\top\mathbb{X}v_i = \sigma_i^2 v_i$ et $u_i = \frac{1}{\sigma_i}\mathbb{X}v_i$.

3.2.4 POD discrète d'ordre supérieur

On a vu jusqu'à présent comment obtenir la POD discrète à partir du modèle continu en considérant la fonction traitée comme une fonction constante par morceaux (proprosition 3.1 et la relation entre POD et SVD)). Dans cette partie on va considérer comment obtenir la POD discrète en augmentant l'ordre d'approximation du modèle continu.

Soit $u \in \mathcal{L}^2([0, T], \mathcal{X})$ fonction définie dans $\mathcal{X}$ un espace de Hilbert, on suppose de connaître les valeurs de u sur une grille uniforme en temps $t_0 = 0, t_1 = t_0 + h, \ldots, t_N = T$

avec $h = \frac{T}{N}$. Notons $u(t_i)$ par u_i. Ensuite, on utilise l'approximation de u par une fonction linéaire par morceaux :

$$u(t) \approx v(t) \equiv \Pi_1^h(t) = \frac{t_{i+1} - t}{h} u_i - \frac{t_i - t}{h} u_{i+1}, \ \forall t \in [t_i, t_{i+1}] \tag{3.20}$$

Nous cherchons une approximation à $K(u) \circ \varphi$ par :

$$\begin{aligned}
K(u) \circ \varphi &= \frac{1}{T} \int_0^T (u(t), \varphi)_{\mathcal{X}} u(t) dt \\
&\approx \frac{1}{T} \sum_{i=0}^{N-1} \int_{t_i}^{t_{i+1}} \left(\frac{t_{i+1} - t}{h} u_i - \frac{t_i - t}{h} u_{i+1}, \varphi \right)_{\mathcal{X}} \left(\frac{t_{i+1} - t}{h} u_i - \frac{t_i - t}{h} u_{i+1} \right) dt \\
&= \frac{1}{Nh} \sum_{i=0}^{N-1} h \left[\frac{1}{3} (u_i, \varphi)_{\mathbb{X}} u_i + \frac{1}{3} (u_{i+1}, \varphi)_{\mathbb{X}} u_{i+1} + \frac{1}{6} (u_i, \varphi)_{\mathbb{X}} u_{i+1} + \frac{1}{6} (u_{i+1}, \varphi)_{\mathbb{X}} u_i \right] \\
&= \frac{1}{6N} \sum_{i=0}^{N-1} \left[2(u_i, \varphi)_{\mathbb{X}} u_i + 2(u_{i+1}, \varphi)_{\mathbb{X}} u_{i+1} + (u_i, \varphi)_{\mathbb{X}} u_{i+1} + (u_{i+1}, \varphi)_{\mathbb{X}} u_i \right]
\end{aligned}$$

On peut encore écrire sous forme matricielle, il existe en effet la matrice $S \in \mathbb{R}^{N \times N}$ telle que :

$$\forall \varphi \in \mathbb{X}, \quad K(u) \circ \varphi = (u_1, \ldots, u_N) S \begin{pmatrix} (u_1, \varphi)_{\mathbb{X}} \\ \vdots \\ (u_N, \varphi)_{\mathbb{X}} \end{pmatrix}$$

Par conséquent, la nouvelle base POD discrète "d'ordre un" proposée peut être obtenue en calculant une SVD généralisée de la matrice des clichés U.

En conlusion, Dans le cas où $\mathcal{X}$ est de dimension finie, toute base POD discrète peut se calculer à partir de la décomposition en valeurs singulières de la matrice des clichés.

Chapitre 4

Parallélisation par la méthode de réduction d'ordre POD

La méthode POD fournit une base orthogonale optimale (base POD) pour représenter dans un espace de dimension inférieure les inconnues du système dynamique. Cette base orthogonale est calculée à partir des données fournies soit par la résolution numérique du système dynamique soit par des données expérimentales. Ces données sont appelées les clichés du système dynamique d'où le nom "méthode des clichés" donné à ce processus de POD. L'efficacité de la base POD à représenter le système dynamique dépend de sa dimension et de la capacité des clichés à représenter toutes les dynamiques du système. C'est pourquoi, les clichés sont choisis de manière régulière et uniforme sur tout l'intervalle de temps considéré. De ce fait, la meilleure base POD est obtenue en résolvant le système dynamique sur tout l'intervalle de temps. Remarquons que ceci rend la méthode des clichés parfaitement adaptée aux problèmes de contrôle où plusieurs simulations sont requises pour trouver les meilleurs paramètres de contrôle du problème.

Notre objectif étant de réduire le temps de simulation, il ne nous est pas possible de faire la simulation sur tout l'intervalle de temps pour obtenir la base POD. Nous proposons une méthode pour paralléliser la résolution numérique du système dynamique qui se base sur une méthode des clichés localisée pour représenter les sous-systèmes sous formes réduites. Nous développons dans ce chapitre un critère mathématique pour définir la mise à jour de la base réduite locale. Nous donnons par ailleurs deux stratégies d'implémentation de cette parallélisation.

La section 4.1 présente les techniques de réduction d'ordre appliquée au système dynamique et au découplage. La section 4.2 présente les analyses sur l'erreur des méthodes proposées. Ensuite la section 4.3 propose les implémentations pratiques de la méthode. Finalement, les résultats numériques sont présentés dans la section 4.4.

4.1 POD pour la réduction d'ordre

4.1.1 La méthode POD dans la réduction d'ordre du modèle

Considérons le système dynamique à n inconnues :

$$x(t)' = \mathbf{f}(t, x(t)) \tag{4.1}$$
$$x(t_0) = x_0 \tag{4.2}$$

avec $t \in [0, T]$, $x_0 \in \mathbb{R}^n$ et $\mathbf{f} : \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}^n$. Notons l'espace contenant $x(t)$ par $\mathcal{X}$ (qui est un espace de Hilbert) . On résout numériquement (4.1) en m instances de temps données $t_1, \ldots, t_m$. La matrice des clichés $\mathbb{X} = [x(t_1), \ldots, x(t_m)]$ permet par la procédure POD d'obtenir une base orthonormale optimale pour représenter la solution x du problème continu (4.1). Cette base optimale peut s'obtenir à l'aide du calcul de la résolution des valeurs propres de $\mathbb{X}^\top\mathbb{X}$ (voir section 3.2.3) ou du calcul de la SVD de la matrice de clichés $\mathbb{X}$:

$$\mathbb{X} = \mathbb{U}\Sigma\mathbb{V}^T$$

À partir de cette décomposition, la base POD est formée avec des vecteurs singuliers définis dans $\mathbb{U} = [u_1, u_2, \ldots, u_m]$. Les clichés étant les solutions d'un problème d'évolution, il y a une certaine cohérence structurelle dans ces données, par conséquent, les valeurs singulières contenues dans la matrice diagonale Σ tendent rapidement vers zéros conduisant à ne garder dans la base POD que les k premiers vecteurs singuliers correspondants aux k premières plus grandes valeurs singulières significatives. Cette base POD est optimale pour représenter $\mathcal{X}$ dans le sens de la norme 2 usuelle. Si les clichés sont suffisament bien choisis pour représenter toutes les dynamiques du système, la base POD peut représenter la variable continue $x(t)$ sur tout l'intervalle de simulation. En effet, tout comme la procédure de projection de Galerkin en éléments finis, il suffit de projeter le système dynamique sur cette base POD pour obtenir le système dynamique de dimenstion inférieure. En introduisant $\xi \in \mathbb{R}^k$ défini par $\mathbf{x}(t) \approx \mathbb{U}_k\xi(t)$, le système de dimension réduite sur la variable réduite $\xi(t)$ satisfait :

$$\xi'(t) = \mathbb{U}_k^\top \mathbf{f}(\mathbb{U}_k\xi(t), t) \tag{4.3}$$
$$\xi(t_0) = \mathbb{U}_k^\top x_0 \tag{4.4}$$

La solution numérique au problème (4.1)-(4.2) peut s'obtenir en résolvant numériquement (4.3)-(4.4). Finalement en faisant une projection de la solution obtenue $\xi(t)$ sur l'espace initial $\mathbb{R}^n$, on obtient une solution numérique recherchée.

4.1.2 Méthode POD pour le découplage du système

Nous proposons ici une nouvelle technique utilisant la réduction d'ordre du modèle pour découpler un système dynamique en sous-systèmes. Considérons dans un premier temps un système d'EDO linéaire d'inconnue $x(t) = (x_1(t), x_2(t))^\top, (x_1(t), x_2(t)) \in \mathbb{R}^{n_1} \times \mathbb{R}^{n_2}$ $(n = n_1 + n_2)$:

$$x(t)' = \mathbf{f}(t, x(t)) = \begin{pmatrix} f(t, x_1(t), x_2(t)) \\ g(t, x_1(t), x_2(t)) \end{pmatrix}, \; x(t_0) = x_0 \tag{4.5}$$

Le système est divisé en deux parties distinctes. Résolvons (4.5) pour obtenir q premiers clichés :

$$\mathbb{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_q) \tag{4.6}$$

avec $\mathbf{x}_i = (x_1(t_i), x_2(t_i))^\top$, $1 \leq i \leq q$. Notons : $\mathbb{X} = \begin{pmatrix} \mathbb{X}_1 \\ \mathbb{X}_2 \end{pmatrix} = \begin{pmatrix} x_{1,1} & \ldots & x_{1,q} \\ x_{2,1} & \ldots & x_{2,q} \end{pmatrix}$, et calculons la SVD pour $\mathbb{X}_1$ et $\mathbb{X}_2$ séparément :

$$\mathbb{X}_1 = \mathbb{U}_1 \Sigma_1 \mathbb{V}_1^\top \tag{4.7}$$
$$\mathbb{X}_2 = \mathbb{U}_2 \Sigma_2 \mathbb{V}_2^\top \tag{4.8}$$

Ceci conduit à avoir deux systèmes séparés $(S1)$ et $(S2)$. $(S1)$ est composé du couple de variables y_1, α_2 où on note $y_1 \in \mathbb{R}^{n_1}$ pour représenter une approximation de x_1 et $\alpha_2 \in \mathbb{R}^q$ pour représenter la variable x_2 sous forme réduite. Dans le système (S_2), α_1 est la variable réduite correspondante à l'approximation de x_1 et y_2 est celle de x_2 :

$$(S1) \quad \begin{pmatrix} y_1' \\ \alpha_2' \end{pmatrix} = \begin{pmatrix} f_(t, y_1, \mathbb{U}_2\alpha_2) \\ \mathbb{U}_2^\top g(t, y_1, \mathbb{U}_2\alpha_2) \end{pmatrix} \tag{4.9}$$

$$(S2) \quad \begin{pmatrix} \alpha_1' \\ y_2' \end{pmatrix} = \begin{pmatrix} \mathbb{U}_1^\top f_(t, \mathbb{U}_1\alpha_1, y_2) \\ g(t, \mathbb{U}_1\alpha_1, y_2) \end{pmatrix} \tag{4.10}$$

Sous l'hypothèse que les bases $\mathbb{U}_1$ et $\mathbb{U}_2$ sont capables de représenter les variables pleines x_1 et x_2, alors (4.9) et (4.10) peuvent être résolue de manière indépendante. Notamment, lorsque les bases POD introduites dans chacun de ces systèmes sont construites à partir des données représentant toutes les dynamiques de x_1 et x_2 sur tout l'intervalle de temps $[0, T]$ donné. Alors $(S1)$ et $(S2)$ sont totalement indépendants.

Cependant, comme les bases réduites ont été construites sur un nombre limité de clichés calculés initialement sur le système complet, elles ne sont généralement pas représentatives de toutes les dynamiques des solutions x_1 et x_2 sur tout l'intervalle de simulation. Par conséquent, des mises à jour régulières de ces bases réduites sont nécessaires. Une mise à jour consiste à réévaluer la base réduite $\mathbb{U}_i$ à partir de la variable "pleine" y_i. Par exemple, une mise à jour pour le système $S1$ au temps t_n signifie que l'on recalcule $\mathbb{U}_2$ à partir des clichés de la variable "pleine" y_2 provenant du système $S2$ et que l'on donne la nouvelle condition initiale $\alpha_2(t_n)$ correspondante à cette mise à jour. Dans la section suivante, on analyse l'erreur sur la solution issue de ce processus afin de déterminer un critère mathématique pour déclencher la mise à jour de chacun des sous-systèmes.

4.2 Analyse de l'erreur de la réduction d'ordre par POD

Plusieurs études sur l'erreur due à la réduction d'ordre du modèle ont été proposées dans la littérature. Par exemple dans [52], la linéarisation du problème non-linéraire au voisinage du temps initial permet d'obtenir une estimation de l'erreur. Cependant, cette estimation de l'erreur du modèle réduit est une estimation locale. Elle n'est valable que si l'on peut considérer la matrice Jacobienne comme constante sur un certain nombre de pas de temps. L'estimation globale de l'erreur sur tout l'intervalle de temps a été introduite dans [37] sous la forme de l'équation adjointe au système initial. La résolution de

ce système adjoint est très coûteuse et a la même complexité que le système initial. L'auteur Homescu (voir [30]) propose alors une approche statistique (*small sample statistical methods*) pour résoudre ce système adjoint à un coût relativement inférieur.

Dans l'approche de découplage du système en sous-systèmes via la réduction d'ordre, seulement quelques clichés initiaux sont requis pour construire le système réduit. Nous ne pouvons pas disposer de toutes les solutions du système dynamique complet. Dans le but d'obtenir une meilleure performance de la résolution, il ne nous est pas possible de résoudre le système adjoint. Donc nous allons établir simplement une majoration de l'erreur qui servira comme critère pour la mise à jour de la base réduite.

4.2.1 Consistence, Stabilité et Convergence des méthodes numériques appliquées aux modèles réduits

Considérons le système d'EDO sous forme réduite (4.3)(4.4) ou le sous-système découplé comme celui dans (4.9) ou (4.10). Nous avons la propriété suivante :

Propriété 4.1. *Considérons le système réduit comme dans (4.3)(4.4) à résoudre numériquement avec une méthode numérique classique (voir chapitre 1). Soit $x(t)$ une solution au problème initial. Supposons que la base réduite POD par la méthode de clichés est calculée sur tout l'intervalle $[0, T]$, c'est à dire que cette base POD satisfait :*

$$\|(\mathbb{U}\mathbb{U}^\top - I)x(t_i)\| < \sum_{i>k} \sigma_i \tag{4.11}$$

pour tout t_i dans la grille des clichés, k est le nombre de valeurs singulières significatives dans le spectre de la SVD de la matrice des clichés des x. Soit $\tilde{\xi}(t)$ une solution au système réduit, on obtient $\tilde{x}(t) = \mathbb{U}\tilde{\xi}(t)$ comme une solution obtenue par la solution réduite, alors :

$$\|\tilde{x}(t_i) - x(t_i)\| < \sum_{i>k} \sigma_i \tag{4.12}$$

De plus, si la matrice des clichés est formée sur une grille de temps uniforme de pas de temps $h = t_{i+1} - t_i$, $\forall i$, alors :

$$\max_t \|\tilde{x}(t) - x(t)\| = \left(\sum_{i>k} \sigma_i\right) \mathcal{O}(h) \tag{4.13}$$

Démonstration. Nous vérifions aisément (4.12) avec (4.11). Par ailleurs, (4.13) est une conséquence de (3.4) sachant que l'on a utilisé les fonctions constantes par morceaux (méthode des clichés) pour approximer l'intégrale dans cette dernière. Cette approximation d'ordre 1 nous permet d'écrire que l'erreur est en $\mathcal{O}(h)$. □

Les méthodes numériques pour résoudre le système dynamique complet s'appliquent naturellement sur celui sous forme réduite. En effet, le système réduit est une projection du sytème d'origine sur la base des vecteurs singuliers. Cette réduction d'ordre peut être vue comme une transformation unitaire (ou changement de base) du système initial. Par conséquent, toutes les analyses des schémas numériques présentées dans le chapitre 1 s'appliquent aux systèmes de taille réduite.

4.2.2 Equation de l'erreur pour la réduction d'ordre du modèle

Nous établissons une estimation de l'erreur de la solution du système réduit par rapport à celle du système complet. Par simplicité, considérons le système linéaire et autonome :

$$x'(t) = Ax(t),\ x(t_0) = x_0 \tag{4.14}$$

Le système réduit associé s'écrit :

$$\xi'(t) = \mathbb{U}^\top A\mathbb{U}\xi(t),\ \xi(t_0) = \mathbb{U}^\top x_0 \tag{4.15}$$

$\mathbb{U}$ étant la base POD calculée à partir d'un certain nombre de clichés du système global. Soit $x(t)$ la solution exacte du problème (4.14), et soit $y(t)$ la projection dans l'espace initial de la solution du système (4.15) : $y(t) = \mathbb{U}\xi(t)$. On définit l'erreur de la réduction d'ordre comme :

$$e(t) = y(t) - x(t) \tag{4.16}$$

En soustrayant (4.14) et (4.15) on obtient le système d'edo sur l'erreur suivant :

$$\begin{aligned} e'(t) &= \mathbb{U}\xi'(t) - x'(t) \\ &= \mathbb{U}\mathbb{U}^\top A\mathbb{U}\xi(t) - Ax(t) \\ &= \mathbb{U}\mathbb{U}^\top A\mathbb{U}\xi(t) - \mathbb{U}\mathbb{U}^\top Ax(t) + \mathbb{U}\mathbb{U}^\top Ax(t) - Ax(t) \\ &= \mathbb{U}\mathbb{U}^\top Ae(t) + (\mathbb{U}\mathbb{U}^\top - I)Ax(t) \end{aligned} \tag{4.17}$$

Pour le cas d'un système linéaire non autonome, l'erreur satisfait également (4.17) car $\mathbb{U}^\top\mathbb{U} = I$. Le système (4.17) montre que le système réduit représente le système complet à partir du moment que $e(t_0) = 0$ et que $Ax(t)$ reste dans le noyau de $\mathbb{U}^\top\mathbb{U} - I$.

Dans le cas d'un système non-linéaire (équation (4.1)(4.2)), comme dans [52], la linéarisation au premier ordre de $y(t)$ donne :

$$\begin{aligned} e'(t) &= \mathbb{U}\mathbb{U}^\top f(t, y(t)) - f(t, x(t)) \\ &= (\mathbb{U}\mathbb{U}^\top - I)f(t, x(t)) + \mathbb{U}\mathbb{U}^\top(f(t, y(t)) - f(t, x(t))) \\ &= (\mathbb{U}\mathbb{U}^\top - I)f(t, x(t)) + \frac{\partial f}{\partial x}(t, y(t))(y(t) - x(t)) + \mathcal{O}(\|e(t)\|^2) \\ &= (\mathbb{U}\mathbb{U}^\top - I)f(t, x(t)) + J_f(t, x(t))e(t) + \mathcal{O}(\|e(t)\|^2) \end{aligned} \tag{4.18}$$

où J_f est la matrice Jacobienne de f. De même, l'erreur de réduction d'ordre reste petit quand $e(t_0) = 0$ et que $f(t, x(t))$ appartient au noyau de $(\mathbb{U}\mathbb{U}^\top - I)$.

4.2.3 Erreur sur la solution par le schéma de découplage

Considérons comme dans la section 4.1.2 la POD pour le système découplé. Dans un premier temps pour le système linéaire autonome :

$$\begin{pmatrix} x_1' \\ x_2' \end{pmatrix} = A \begin{pmatrix} x_1' \\ x_2' \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \tag{4.19}$$

$A_{ij} \in \mathbb{R}^{n_i \times n_j}$. En utilisant la POD pour découpler le système précédent, on obtient deux systèmes indépendants :

$$\begin{pmatrix} y_1' \\ \alpha_2' \end{pmatrix} = \mathcal{U}_2^\top A \mathcal{U}_2 \begin{pmatrix} y_1 \\ \alpha_2 \end{pmatrix} \tag{4.20}$$

$$\begin{pmatrix} \alpha_1' \\ y_2' \end{pmatrix} = \mathcal{U}_1^\top A \mathcal{U}_1 \begin{pmatrix} \alpha_1 \\ y_2 \end{pmatrix} \tag{4.21}$$

où $\mathcal{U}_2 = \left(\begin{smallmatrix} I & 0 \\ 0 & \mathbb{U}_2 \end{smallmatrix}\right)$, $\mathcal{U}_1 = \left(\begin{smallmatrix} \mathbb{U}_1 & 0 \\ 0 & I \end{smallmatrix}\right)$, $\mathbb{U}_1$ (respectivement $\mathbb{U}_2$) contient la bases POD calculée à partir des SVD sur des clichés de y_1 (respectivement y_2). Nous cherchons dans la suite l'expression de l'erreur par rapport au système initial.

Notons la solution au problème (4.20) par $(\hat{y}_1, \hat{y}_2)^\top = (\hat{y}_1, \mathbb{U}_2\alpha_2)^\top$. Soit $(x_1, x_2)^\top$ la solution exacte au problème (4.19)- (4.20). L'erreur entre la solution du premier sous-système et celle du système d'origine est notée par $e_1(t) = \hat{y}_1(t) - x_1(t)$ et $e_2(t) = \hat{y}_2 - x_2 = \mathbb{U}_2\alpha_2 - x_2$. On peut vérifier que e_1 et e_2 satisfont :

$$e_1'(t) = A_{11}e_1(t) + A_{12}e_2(t) \tag{4.22}$$

$$e_2'(t) = \mathbb{U}_2\mathbb{U}_2^\top A_{21}x_1(t) - A_{21}x_1(t) + \mathbb{U}_2\mathbb{U}_2^\top A_{22}\mathbb{U}_2\alpha_2(t) - A_{22}x_2(t) \tag{4.23}$$

(4.23) peut s'écrire sous la forme :

$$\begin{aligned} e_2'(t) &= (\mathbb{U}_2\mathbb{U}_2^\top - I)(A_{21}x_1(t) + A_{22}x_2(t)) + \mathbb{U}_2\mathbb{U}_2^\top A_{21}e_1(t) + \mathbb{U}_2\mathbb{U}_2^\top A_{22}e_2(t) \\ &= (\mathbb{U}_2\mathbb{U}_2^\top - I)x_2'(t) + \mathbb{U}_2\mathbb{U}_2^\top A_{21}e_1(t) + \mathbb{U}_2\mathbb{U}_2^\top A_{22}e_2(t) \end{aligned}$$

finalement :

$$e_1'(t) = A_{11}e_1(t) + A_{12}e_2(t) \tag{4.24}$$

$$e_2'(t) = \mathbb{U}_2\mathbb{U}_2^\top(A_{21}e_1(t) + A_{22}e_2(t)) + (\mathbb{U}_2\mathbb{U}_2^\top - I)x_2'(t) \tag{4.25}$$

Dans le cas où on résout un système non-linéaire comme dans l'équation (4.5), en développant les termes non linéaires au premier ordre, on obtient :

$$\begin{aligned} e_1' &= J_f(y_1)e_1 + J_f(y_2)e_2 + \mathcal{O}(\|\mathbf{x} - \mathbf{y}\|^2) \\ e_2' &= (\mathbb{U}_2\mathbb{U}_2^\top - I)g(x_1, x_2, t) + \mathbb{U}_2\mathbb{U}_2^\top(g(y_1, y_2, t) - g(x_1, x_2, t)) \\ &= (\mathbb{U}_2\mathbb{U}_2^\top - I)g(x_1, x_2, t) + \mathbb{U}_2\mathbb{U}_2^\top(J_g(y_1)e_1 + J_g(y_2)e_2) + \mathcal{O}(\|\mathbf{x} - \mathbf{y}\|^2) \end{aligned}$$

où $\mathbf{x} = (x_1, x_2)^\top$, $\mathbf{y} = (y_1, y_2)^\top$, J_f (respectivement J_g) est la matrice Jacobienne de f(resp. g). En ignorant les termes d'ordre supérieur à un, l'équation gouvernant l'erreur est :

$$e_1'(t) = J_f(y_1)e_1(t) + J_f(y_2)e_2(t) \tag{4.26}$$

$$e_2'(t) = \mathbb{U}_2\mathbb{U}_2^\top(J_g(x_1)e_1(t) + J_g(x_2)e_2(t)) + (\mathbb{U}_2\mathbb{U}_2^\top - I)x_2'(t) \tag{4.27}$$

La même démarche s'applique si on considère l'erreur entre le deuxième sous-système et le système complet. On a alors la proposition suivante :

Proposition 4.1. *L'erreur jusqu'à d'ordre* 1 *de la méthode de découplage par la POD sur chacun des sous-systèmes satisfait le système l'EDO suivant*

$$\mathbf{e}_i'(t) = M_i \mathbf{e}_i(t) + \epsilon_i(t),\ i = 1, 2 \tag{4.28}$$

où $\mathbf{e}_i(t) = (e_{1,i}(t), e_{2,i}(t))^\top,\ i = 1, 2$ *selon les différents sous-systèmes* $(S1)$ *et* $(S2)$*, les* $e_{1..2,i}$ *sont définis dans (4.26)(4.27). Les matrices sont définies par :*

$$M_1 = \begin{pmatrix} J_f(y_1) & J_f(y_2) \\ \mathbb{U}_2\mathbb{U}_2^\top J_g(y_1) & \mathbb{U}_2\mathbb{U}_2^\top J_g(y_2) \end{pmatrix} \tag{4.29}$$

$$M_2 = \begin{pmatrix} \mathbb{U}_1\mathbb{U}_1^\top J_f(y_1) & \mathbb{U}_1\mathbb{U}_1^\top J_f(y_2) \\ J_g(y_1) & J_g(y_2) \end{pmatrix} \tag{4.30}$$

et

$$\epsilon_1(t) = \begin{pmatrix} 0_{n_1} \\ (\mathbb{U}_2\mathbb{U}_2^\top - I)g \end{pmatrix} \tag{4.31}$$

$$\epsilon_2(t) = \begin{pmatrix} (\mathbb{U}_1\mathbb{U}_1^\top - I)f \\ 0_{n_2} \end{pmatrix} \tag{4.32}$$

0_n *est le vecteur nul de taille* n*,* n_1 *est la taille de* y_1 *et* n_2 *la taille de* y_2*.*

Cette proposition montre que si l'erreur de réduction d'ordre dans le cas du sous-système découplé reste "petit" lorsque $\mathbf{e}_i(t_0) = 0$ et le terme $g(t, x_1, x_2)$ (respectivement $f(t, x_1, x_2)$) pour $(S1)$ (resp. $(S2)$) appartient au noyau de $\mathbb{U}_2\mathbb{U}_2^\top - I$ (respectivement f pour $(\mathbb{U}_1\mathbb{U}_1^\top - I)$), alors l'erreur de réduction d'ordre pour le système découplé reste négligeable. Les estimations (4.31)(4.32) restent petites lorsque les second membres de chacun des sous-systèmes sont tous simultanément dans les espaces générés par les bases réduites correspondantes.

Remarque 4.1. ***(Principe de la mise à jour basée sur le résidu)*** *D'après l'équation (4.28), l'erreur de la réduction d'ordre peut se répandre implicitement entre les variables dans un sous-système aux celles de l'autre sous-système. Une ré-initialisation (ou mise à jour) basée sur le résidu d'un sous-système doit en conséquence entrainer une mise à jour globale entre tous les sous-systèmes.*

4.2.4 Estimation de l'erreur dans la réduction d'ordre du modèle

Considérons le système d'EDO linéaire sous leur forme générique dans (4.28), sa solution peut s'écrire :

$$e(t) = \int_{t_0}^{t} \exp M(t-\tau)\epsilon(\tau)d\tau + \exp(Mt) \cdot e(t_0) \tag{4.33}$$

la solution $e(t)$ peut être écrite sous la forme $e(t) = F(T, M)\epsilon(t) + G(T, M)e(t_0)$, où $F(T, M) : \mathcal{L}_2([0, T], \mathbb{R}^n) \to \mathcal{L}_2([0, T], \mathbb{R}^n)$ et $G(T; A) : \mathbb{R}^n \to \mathcal{L}_2([0, T], \mathbb{R}^n)$ sont des opérateurs linéaires. En général, pour borner F et G, il faut estimer les normes exponentielles correspondantes pour obtenir une estimation de l'erreur sous la forme :

$$\|e\| \leq \|F(T, M)\|\|\epsilon\| + \|G(T, M)\|\|e_0\| \tag{4.34}$$

Nous remarquons que cette norme croît de manière exponentielle avec T et le taux d'accroissement dépend d'une part de la partie réelle la plus grande de toutes les valeurs propres de M et d'autre part de la non-normalité de la matrice M. On peut par exemple consulter [20] sur l'estimation de l'exponentielle d'une matrice. Cependant, cette estimation en norme exponentielle n'a pas d'intérêt pratique. En effet, il est très difficile d'obtenir une borne raisonnable pour l'estimation d'erreur.

Pour le cas de l'estimation de l'erreur commise du sous-système dans la section précédente, nous cherchons simplement à trouver une estimation grossière de l'erreur pour pouvoir corriger le sous-système de taille réduite par des mises à jour de la base POD.

Remarque 4.2. *La solution à l'équation (4.33) - celle gouvernant l'erreur du modèle réduit - reste bornée quand la source $\epsilon(t)$ reste suffisament petite. C'est à dire si la norme de $(\mathbb{U}_2\mathbb{U}_2^\top - \mathcal{I})y_2'(t)$ reste petite. En effet, l'erreur de la réduction d'ordre est due à la non-orthogonalité de $y_2'(t)$ à l'espace engendré par les vecteurs de la base réduite définie définis par les colonnes $\mathbb{U}_2$.Notons que $\|(\mathbb{U}_2\mathbb{U}_2^\top - \mathcal{I})y_2'(t)\|$ reste petit quand $y_2'(t) = g(y_1, y_2, t) \in Ker(\mathbb{U}_2\mathbb{U}_2^\top - I)$.*

Définition 4.1. *(**Résidu sur le découplage par POD**) : Le résidu de découplage pour chaque instant de t est donné par :*

$$\epsilon_1(t) = \|(\mathbb{U}_1\mathbb{U}_1^\top - I)y_1'(t)\| \tag{4.35}$$
$$\epsilon_2(t) = \|(\mathbb{U}_2\mathbb{U}_2^\top - I)y_2'(t)\| \tag{4.36}$$

Remarquons qu'une mise à jour (ou l'échange de la base POD) entre les sous-systèmes consiste à réinitialiser chaque partie réduite avec la nouvelle base POD évaluée sur l'autre système.

L'analyse de l'erreur pour le schéma de découplage mène à un calcul du résidu donné par la définition 4.1. Pour chaque sous-système, définissant un ensemble de variables réduites, on obtient un résidu associé. Le déclenchement de la mise à jour des bases réduites est réalisé lorsque ces valeurs de résidus dépassent un seuil de tolérance fixé. Ceci nous permet d'avoir une mise à jour adaptative entre les sous-systèmes.

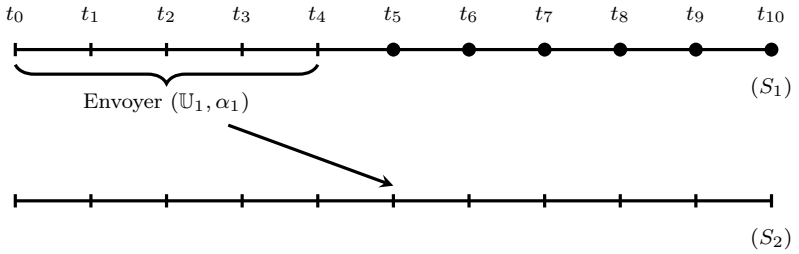
Lorsque le système dynamique complet fait intervenir des échelles fortement différentes entre les composantes, ce seuil de tolérance doit être affiné pour en tenir compte. Plutôt que de définir pour chaque ensemble de variables différentes tolérances, nous normalisons le résidu avec la première valeur singulière tronquée σ_{k_i+1} du i-ième système réduit et la norme du second membre pour définir un nouveau résidu normalisé $\tilde{\epsilon_i}$ donné par :

$$\tilde{\epsilon_i} = \frac{\sigma_{k_i+1}}{\|f_i\|}\epsilon_i = \frac{\sigma_{k_i+1}}{\|f_i\|}\|(\mathbb{U}_i\mathbb{U}_i^\top - I)f_i\| \tag{4.37}$$

Cette normalisation du résidu permet d'utiliser une unique tolérance sur tous les différents sous-systèmes sous forme réduite.

4.3 Implémentation de l'algorithme de découplage par la méthode POD

Cette section présente des algorithmes pour découpler un système dynamique en sous-systèmes par la réduction d'ordre présenté précédemment. Soit x la solution du système dynamique sous forme pleine, nous découplons x en P sous ensembles disjoints $y_1, \ldots, y_P$. Ensuite, les P sous-systèmes indépendants $(S_1), \ldots, (S_P)$ sont formés de la manière suivante : chaque sous-système (S_I) est composé de sa propre variable y_I sous forme "pleine" couplée avec tous les termes $\alpha_J,\ J \neq I$, les α_J sont des formes réduites représentante les parties y_J. Pour simplifier, on note $u_I, I = 1, \ldots, P$ la variable du système (S_I) : $u_I = (\alpha_1, \ldots, \alpha_{I-1}, y_I, \alpha_{I+1}, \alpha_P)^\top$. Dans un premier temps, on décrit l'algorithme 1 avec les points de rendez-vous pour échanger la base. Ensuite, l'algorithme 2 implémente la version d'échange de base POD selon le résidu défini dans la section précédente.



La première approche consiste à fixer des points de synchronisation pour échanger les bases entre les sous-systèmes. Fixons un ensemble des $T_j : 0 < T_1 < \cdots < T_N < T$. Nous définissons un intervalle d'échange $[T_j, T_{j+1}]$ pendant lequel chaque système procède indépendament des autres. Un échange globale entre les sous-systèmes se fait à chaque moment T_j. La première version pour découplage est la suivante :

Algorithme 1 Echanges sur les points fixés

Initialisation de q clichés du système complet jusqu'à T_1
Calculer la SVD de chaque sous ensemble des clichés séparé : $\mathbb{X}_I$.
Initialisation de $u_{I,I}(T_1)$ et calculer la condition initiale pour $u_{I,J}(T_1)$
pour T_j : j=1 à J **faire**
  pour t_{it}=T_j à T_{j+1} **faire**
    Prendre un pas $u_I(t_{it})$
    Mettre à jour (par décalage) la matrice des clichés $\mathbb{X}_I$ avec la variable locale $u_{I,I}$
  fin pour
  Calculer $\mathbb{U}_I$ de la SVD de $\mathbb{X}_I$
  Envoyer la base $\mathbb{U}_I$ et la variable réduite α_I à tous les autres solveur S_J.
  Recevoir tous les nouvelles bases des S_J
  Mettre à jour $u_{I,J},\ J \neq I$
fin pour

Cet algorithme est illustré dans la figure 4.1. Cet algorithme effectue des échanges forcés de manière régulière pour garantir la mise à jour entre les sous-systèmes.

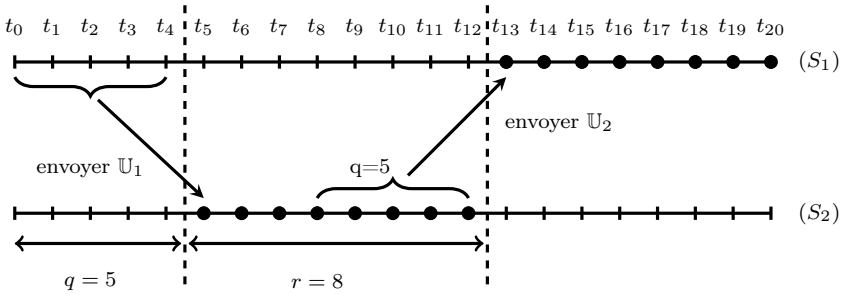


FIGURE 4.1 – Shéma d'échange de base POD : $q = 5$ et $r = 8$

On peut en effet améliorer ce dernier algorithme d'échange de base POD en se basant sur l'étude du résidu. L'échange forcé sur les points T_j ne tient pas en compte de l'erreur commise entre la partie réduite et celle qu'elle représente. Les échanges peuvent être soit inutiles soit insuffisants. L'algorithme suivant décrit une stratégie où les échanges se font de manière adaptative :

Algorithme 2 Echanges de base POD basé sur le résidu

Initialisation de q clichés jusqu'à T_1
Calculer la base POD pour chacun de système séparé
Initialisation de $u_{I,I}(T_1)$ la condition initiale $u_I(T_1)$
pour t_{it}=T_1 à T_{stop} **faire**
  Faire un pas $u_I(t_{it})$
  Mettre à jour (par décalage) $u_{I,I}$ dans $\mathbb{X}_{I,I}$ - la matrice des clichés
  Calculer le résidus $\epsilon_{I,i}, \forall i \neq I$ par la formule (4.31)
  si $\epsilon_{I,i} > tol$, $i \neq I$ **alors**
    Communiquer à (S_i) pour la base $\mathbb{U}_i$ de (S_i)
    Mettre une réception non-bloquante pour $\mathbb{U}_i, \alpha_i(t_{it})$ de tous les (S_i)
    Envoyer (non bloquant) $\mathbb{U}_I$ et $\alpha_I(t_{it})$ à tous les (S_i).
  finsi
  Terminer toutes les communications (si elles existent)
  si Une nouvelle base est reçue **alors**
    Réinitialiser la condition initiale au moment t_{it}
  finsi
fin pour

Ainsi ce deuxième algorithme vérifie à chaque pas de temps le résidu pour déclencher la mise à jour des variables sous forme réduite. Dans un premier temps, nous nous contentons de calculer la nouvelle base sur les derniers clichés du modèle d'origine. Dans le chapitre suivant, une nouvelle perspective de l'évaluation de la base POD (POD incrémentale) sera développée.

4.4 Applications numériques

La méthode de découplage basée sur la réduction d'ordre du modèle est appliquée pour résoudre un système d'EDO de type raide. Notre premier cas test consiste à résoudre un problème d'EDO provenant de l'EDP DUK. Le deuxième cas test est celui de l'équation de Navier-Stokes en formulation courant-tourbilon ω-ψ.

La machine utilisée est une SGI altix 350 avec des processeurs de IA64 de vitesse d'horloge à 1.5Ghz et 6Mo de cache niveau 3 (L3-cache). La communication des données entre les processeurs est réalisée avec les routines de la bibliothèque MPI (Message Passing Interface) basée sur le réseaux de type Numalink entre processeurs internes. Nous présentons la performance de la méthode proposée comparée à une résolution classique, ensuite nous considérons l'aspect de précision de la solution obtenue.

4.4.1 Cas test de DUK

Prenons l'exemple du problème advection-diffusion cinétique de deux composantes couplées (DUK *diurnal kinetics-advection-diffusion PDE* en anglais). Ce dernier consiste à un système d'EDP évolutif, où la variable d'espace discrétisée par la méthode de ligne. Le système obtenu est alors un système d'EDO où on propose de résoudre par les méthodes numériques proposées.

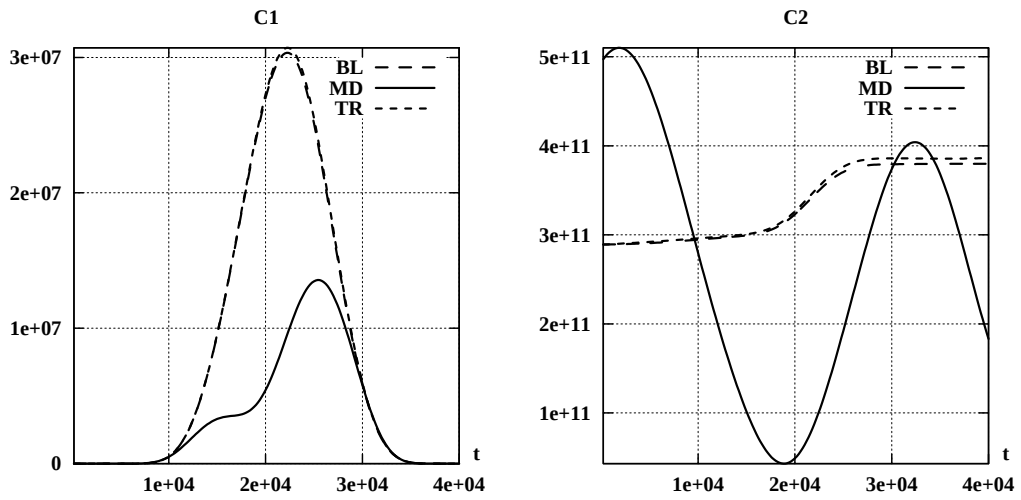


FIGURE 4.2 – La solution de référence c^1 et c^2 aux points d'échantillonage sur la grille d'espace. Notation BL : bas à gauche, MD : millieu, TR : haut droite. Erreur relative est fixée à $1 \cdot 10^{-5}$

L'example du problème DUK s'écrit :

$$\frac{\partial c^i}{\partial t} = K_h \frac{\partial^2 c^i}{\partial x^2} + V\frac{\partial c^i}{\partial x} + \frac{\partial}{\partial y}K_v(y)\frac{\partial c^i}{\partial y} + R^i(c^1, c^2, t) \quad (i = 1, 2) \tag{4.38}$$

où $t \in [0, 4.0 \cdot 10^4]$ et les R^i sont :

$$\begin{aligned} R^1(c^1, c^2, t) &= -q_1 c^1 c^3 - q_2 c^1 c^2 + 2q_3(t)c^3 + q_4(t)c^2, \\ R^2(c^1, c^2, t) &= q_1 c^1 c^3 - q_2 c^1 c^2 - q_4(t)c^2 \end{aligned}$$

$0 \leq x \leq 20$, $30 \leq y \leq 50$ (en unité km). Les constantes et paramètres sont définis comme suit : $K_h = 4.0 \cdot 10^{-6}$,$V = 10^{-3}$, $K_v = 10^{-8} exp(y/5)$, $q_1 = 1.63 \cdot 10^{-16}$, $q_2 = 4.66 \cdot 10^{-16}$, $c^3 = 3.7 \cdot 10^{16}$,

$$q_i(t) = \begin{Bmatrix} exp(-a_i/sin(\omega t)), & \text{for } sin(\omega t) > 0 \\ 0, & \text{for } sin(\omega t) \leq 0 \end{Bmatrix} \ (i = 3, 4)$$

et $\omega = \pi/43200$, $a_3 = 22.62$, $a_4 = 7.601$. Les conditions de bord imposées sont de type Neumann homogène. Chaque direction d'espace x(y respectivement) est discrétisée avec 10 intervalles uniformes. Ce dernier conduit à un système d'EDO de taille 200. L'intérêt de cet exemple est d'augmenter la taille du système d'EDO en affinant la grille d'espace.

La figure 4.2 présente la solution de référence issue d'une résolution classique (par le solveur CVODE) où les composantes c^1 et c^2 ne varient pas dans la même échelle de valeurs.

Cas de la réduction d'ordre sur le système global

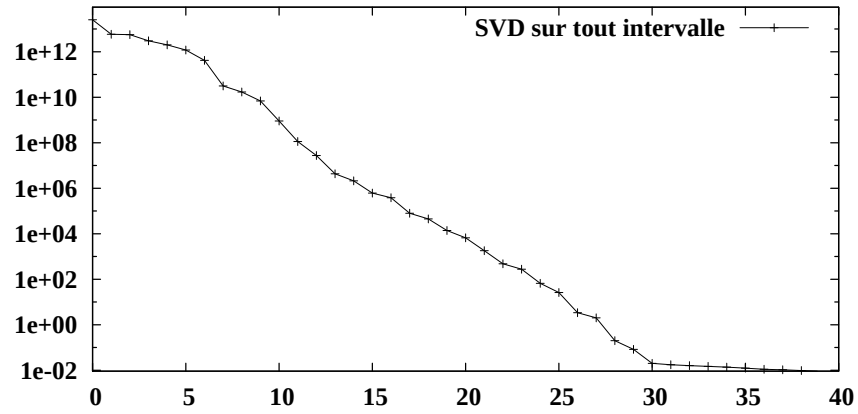


FIGURE 4.3 – Spectre de la SVD sur les clichés uniformément répartis dans $[0, 4 \cdot 10^5]$

Dans le but de comparer la solution de la méthode de découplage par le POD. Considérons d'abord une solution de la méthode de réduction d'ordre simple, i.e. le modèle réduit est contruit pour représenter tout le système complet. Construisons un modèle réduit du système complet basé sur les clichés calculés. Ces clichés sont pris sur tout l'intervalle du

temps où une solution fiable du système est calculée. Cet exemple n'a pas de signification réelle pour la cause suivante : dans notre contexte de résolution d'EDO, on n'a pas vraiment besoin de construire le modèle réduit POD lorsque l'on a résolu tout le système. Néamoins, cela illustre un cas idéal de la réduction d'ordre "optimale" du système complet. Un tel système réduit serait optimal, avec le nombre minimum de variables pour simuler le système d'origine. Cela nous donne une idée d'une borne inférieure du coût de calcul idéal pour le système réduit. Pour le cas test figuré, on utilise des clichés séparés de manière régulière les uns des autres dans l'intervalle $[0, 400.000]$, le système initial est de taille 200. La base POD calculée sur ces clichés donne un spectre avec 31 modes significatifs. Cette base définit un système réduit de 31 variables, on peut affirmer qu'au mieux, on aura un système de dimension 31.

En dehors de son manque d'intérêt d'un point de vue pratique au niveau de l'implémentation, cette construction de la base réduite a de même certaines limitations au point de vue numérique. Considérons le spectre des 31 valeurs singulières obtenu. C.f figure 4.3, remarquons que la plus petite valeur propre significative est de l'ordre de 10^{-1}. Nous constatons que le modèle réduit est de précision de l'ordre de 10^{-1} en norme 2 au mieux (voir l'équation (3.4). Cette erreur de 10^{-1} est beaucoup trop forte pour la composant c^1 qui est de l'ordre 10^{-4} en valeur absolue (sur certain intervalle de simulation). Par conséquent, le modèle réduit construit ne permet pas de tenir en compte réellement la dynamique de cette variable. Donc, nous considérons le découplage par POD en découplant c^1 de c^2 pour calculer indépendamment les bases réduites correspondantes.

Solution par la méthode de découplage par POD

Nous utiliserons les notations suivantes dans la suite : la POD est calculée sur les q pas de temps par la SVD de la matrice des clichés correspondants. Une fois cette POD calculée, tronquons le spectre des valeurs singulières par les k premières plus grandes valeurs. Il existe par ailleurs plusieurs critères sur cette troncature. En générale, les $n - k$ valeurs ignorées dans le spectre de la SVD représente du bruit dans les modes considérés. Dans la première approche présentée dans ce chapitre, toute base POD est calculée sur les q derniers clichés disponibles de chacun de sous-système. Selon l'algorithme présenté dans la section 4.3, une fois les bases POD échangées, elles sont utilisées pendant r pas de manière indépendante. Ainsi de suite, les solutions calculées sur ces r pas fournissent les derniers clichés pour construire la nouvelle base POD, puis on échange les bases POD entre les sous-systèmes. Un tel cycle est appliqué jusqu'au temps final de la simulation. Ceci décrit le premier algorithme avec les échanges de bases réduites à des instants de temps fixés. Le deuxième algorithme de la section 4.3 qui serait présenté ensuite, utilise le critère sur le résidu et calcule la base réduite en utilisant les q derniers clichés disponibles au fut et à mesure de la simulation.

Cependant, cette approche de calcul de la base POD n'est pas satisfaisante dans certains cas. Comme on verra dans le cas de l'équation de Navier-Stokes, il arrive que la base réduite construite à partir des q derniers clichés ne soit pas significative pour représenter la variable complète. En effet cette méthode rencontre des difficultés dans la résolution du système d'EDO issu de l'équation de Navier-Stokes. Pour cette raison on présentera dans le chapitre 5 une nouvelle technique de mise à jour de SVD pour la

construction de la base POD.

Troncature du spectre de la POD

La troncature des valeurs singulières repose sur les deux critères suivants : $(\sigma_k - \sigma_{k-1}) < RTOL \cdot \sigma_{k-1}$ et $\sigma_k < ATOL \cdot \sigma_0$. Dans un premier temps, on applique séparément chaque critère de troncature pour construire le modèle réduit. Utilisons $q = 40$ pas pour construire la matrice des clichés. Ensuite, l'algorithme d'échange de la base POD est implémenté avec des échanges régulière fixées : la base POD est échangée globalement à chaque $r = 90$ pas. Le pas de temps vaut 10^{-1} (en seconde "s"). Le solveur numérique utilise le schéma Euler implicite à pas constant. Le solveur CVODE présenté dans le chapitre 1)-section 2.4 fait appel à la résolution de Newton en utilisant une décomposition LU.

TABLE 4.1 – Erreur relative comparée à la solution de référence de précision fixée 10^{-5}. 3 méthodes numériques sont présentées : la première est donnée par le schéma d'Euler implicite sur le système global, les autres sont données par la méthode de découplage par POD avec les différentes troncatures de la base POD

Nombre procs	1	2	
Cas	BDF	POD : $\lambda_k < 10^{-2}\lambda_1$	POD : $\frac{\lambda_k - \lambda_{k+1}}{\lambda_k} < 10^{-1}$
Taille	200	102	109
Temps écoulé (s)	258.75	151.3	171.4
Précision	$1.8245\ 10^{-4}$	$1.64\ 10^{-2}$	$1.8245\ 10^{-4}$

Le tableau 4.1 donne le temps d'exécution des différentes méthodes de résolution, l'erreur relative est celle comparée avec la solution de référence donnée avec une précision de 10^{-5}. Le premier cas est celle où on a utilisé le schéma Euler implicite à pas de temps constant 10^{-1}. Le deuxième et le troisième cas sont celles utilisant le découplage en deux sous-systèmes avec $q = 30$ et $r = 80$ avec le pas de temps constant à 10^{-1} et se distinguent de la manière de tronquer la base POD. Le deuxième cas utilise le critère $\lambda_k < 10^{-2}\lambda_1$, le troisième cas utilise celui de : $\frac{\lambda_k - \lambda_{k+1}}{\lambda_k} < 10^{-1}$. Comparons l'influence de la troncature sur la précision de la solution obtenue. La solution du premier cas admet une erreur relative de 10^{-4}, celle donnée par le cas 2 est de 10^{-2} due à la taille limitée du système réduit. Dans le troisième cas, l'ordre de précision est le même comparé à la solution provenant d'un schéma classique. La solution par le modèle réduit de seulement 2 modes n'est pas suffisante pour représenter le modèle complet.

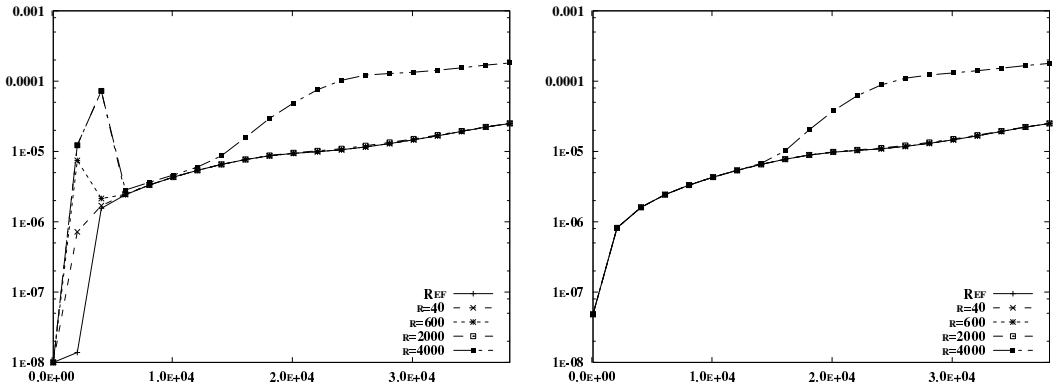
En conclusion, la troncature de la base réduite est cruciale dans la précision du modèle réduit. D'une part il faut éviter de prendre trop de modes non-significatifs, d'autre part il faut prendre en compte suffisament de modes pour représenter le modèle réduit. Dans la suite, nous nous proposons d'utiliser de préférence la troncature suivante :

$$\sigma_k < max\{10^{-15}\sigma_1, 10^{-6}\} \tag{4.39}$$

Cette troncature garantit une bonne prise en compte du modèle réduit dans la plupart de cas de figure. La première valeur dans (4.39) est une tolérance relative à la première valeur singulière, la deuxième est une tolérance absolue.

Algorithme avec échanges de la base POD fixées

Nous présentons les résultats numériques issus de l'implémentation du premier algorithme où les échanges de la base réduite sont fixées sur des instants donnés. L'avantage de cette approche est dans la simplicité d'implémentation de l'algorithme. En effet nous fixons des points de rencontre de manière assez régulière pour garantir une bonne prise en compte des nouvelles solutions obtenues. Par contre, l'inconvénient est que ces échanges peuvent parfois être excessifs. La figure 4.4 montre l'erreur relative comparée à une solution numérique de référence sur chacune des parties réduites correspondantes aux composantes c^1 et c^2. La première courbe en trait plein représente l'erreur relative entre deux solutions "classiques", une avec un pas de temps 10^{-1} et l'autre avec $5.0\ 10^{-2}$. Cette courbe sert de référence pour les méthodes de découplage par POD considérées. Les courbes suivantes représentent les solutions obtenues avec les différents paramètres. r est le nombre de pas indépendant de chaque sous-systèmes avant les mises à jour régulière. Les différentes valeurs de r considérées sont : $r = 40, 600, 2000, 4000$, ces fréquences d'échange de base réduite correspondent aux nombres d'échanges globalement effectués suivant : $R = 10000, 666, 200, 100$. Avec la mise à jour extrêmement régulière ($r = 40$), la solution obtenue est de même précision avec le cas de référence. En bassant la fréquence de mise à jour, nous constatons que les solutions obtenues sont moins précises. Par exemple, la solution donnée par la méthode où les mises à jour effectuées tous les $r = 4000$ pas - dont globalement 100 échanges effectués - conduisent à une baisse de précision d'un décimal comparée à la solution "classique". Ici la précision atteinte du schéma est au mieux à l'ordre de $\mathcal{O}(10^{-4})$ avec le nombre d'échanges total qui vaut 10.000 échanges.



(a) (b)

FIGURE 4.4 – Erreur relative en norme 2 de la solution de l'algorithme 1, la figure 4.4a est pour la composante c^1, la figure 4.4b est pour la composante c^2. $q = 40$, $\delta t = 10^{-1}$, algorithme d'échange de la base POD sur les points fixés selon l'algorithme 1

Algorithme avec échanges adaptatifs de la base POD

Considérons ensuite le deuxième algorithme où l'échange de la base réduite repose sur le critère de résidu défini dans (4.37). Ce résidu est comparé à une tolérance pour

déclencher la mise à jour des sous-systèmes.

La figure 4.5 présente les temps des mises à jour. Chaque saut correspond à une mise à jour entre les sous-systèmes. Par conséquent, une petite tolérance déclenche beaucoup plus d'échanges. Pour les tolérances fixées à 10^{-2} et 10^{-4}, le nombre total d'échanges varie de 10 à 212 échanges. Remarquons qu'il y a nettement moins d'échanges comparés à la première méthode où les échanges sont fixés à l'avance.

La figure 4.6 donnent les erreurs relatives des solutions considérées avec la solution de référence. Comme précédemment, nous comparons ces solutions avec la solution dite "classique" avec le même pas de temps. Pour une tolérance du résidus de 10^{-2}, nous obtenons une solution de même précision que celles obtenue dans la section précédente. Cette tolérance déclenchent au total 58 mises à jour de la base réduite. Une meilleure solution s'obtient en fixant une tolérance de 10^{-4}, ceci déclenche au total 212 mises à jour de la base réduite pour obtenir une précision sur la solution d'environ 10^{-5} (à comparer avec la précision de 10^{-4} obtenue par la stratégie à l'échange fixe utilisant 10000 mises à jour). A l'extrême, une tolérance à 1.0 conduit seulement à 10 échanges sur tout l'intervalle de simulation mais donne une solution fausse avec une erreur relative de plus de 10% (voir figure 4.6).

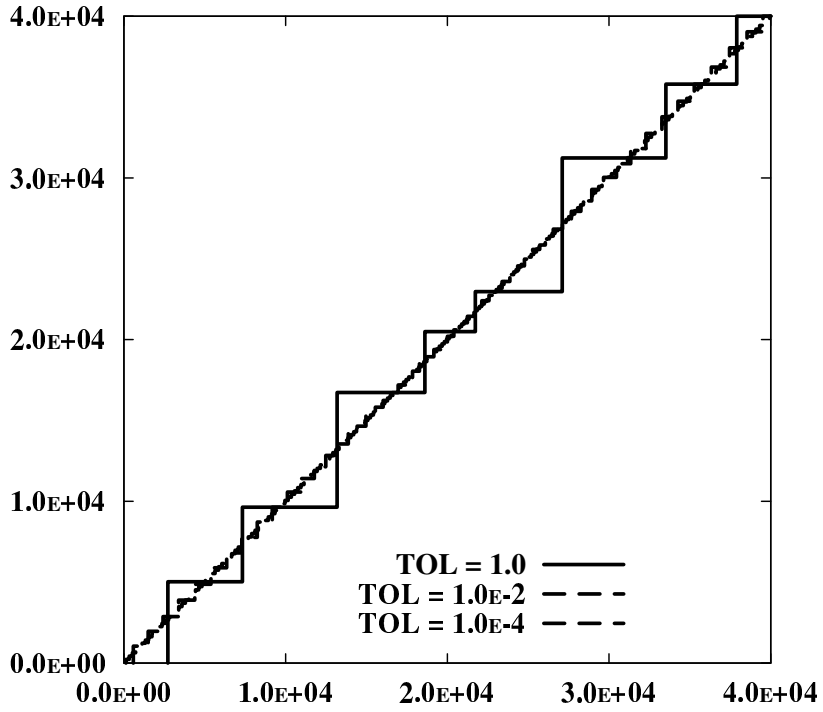
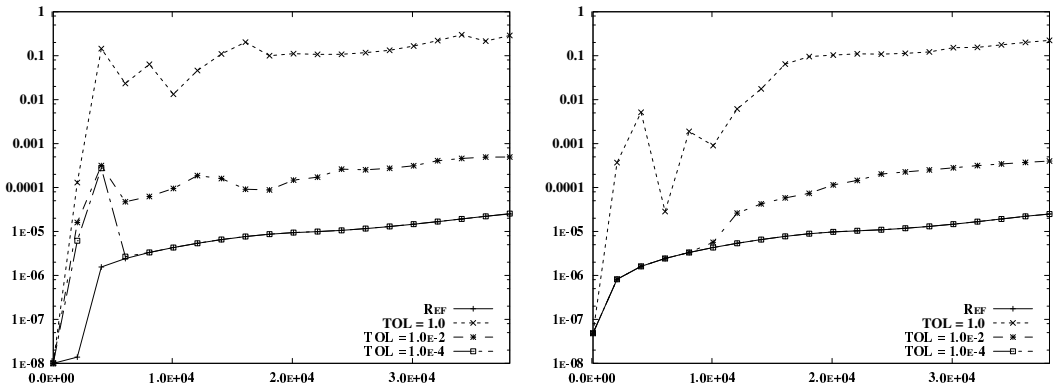


FIGURE 4.5 – L'échange de la base POD selon le résidus - algorithme 2 avec les différentes valeurs de tolérance pour le résidu. Le nombre d'échange au total pour chaque tolérance différente : cas 1 : 10 échanges, cas 2 : 58 échanges, cas 3 : 212 échanges



(a) (b)

FIGURE 4.6 – Erreur relative de la solution de l'algorithme 2 : (a) sur la composante c^1, (b) sur la composante c^2

Analyse des résidus des algorithmes

Le résidu est analysé pour chacun des algorithmes d'échange de la base présentés. Nous présentons dans la figure 4.7 les résultats obtenus.

Les figures 4.7a-4.7b donnent les résultats obtenus par l'algorithme 1 où l'échange des bases réduites se font tous les 2000 pas. Celles dans 4.7c-4.7d et 4.7e-4.7f représentent les résulats obtenus par l'algorithme 2 avec les tolérances fixées à 10^{-4} et 1.0. Ces résidus sont en concordance avec les précisions des solutions dans les dernières sections. Cependant, nous remarquons que la base POD basée sur les $q = 40$ derniers clichés n'est pas forcément un bon choix. En effet on remarque que l'erreur commise entre solution découplée-réduite et la solution de référence est assez grande au début de la simulation, cela est due à la POD locale qui ne contient pas assez de dynamique du système.

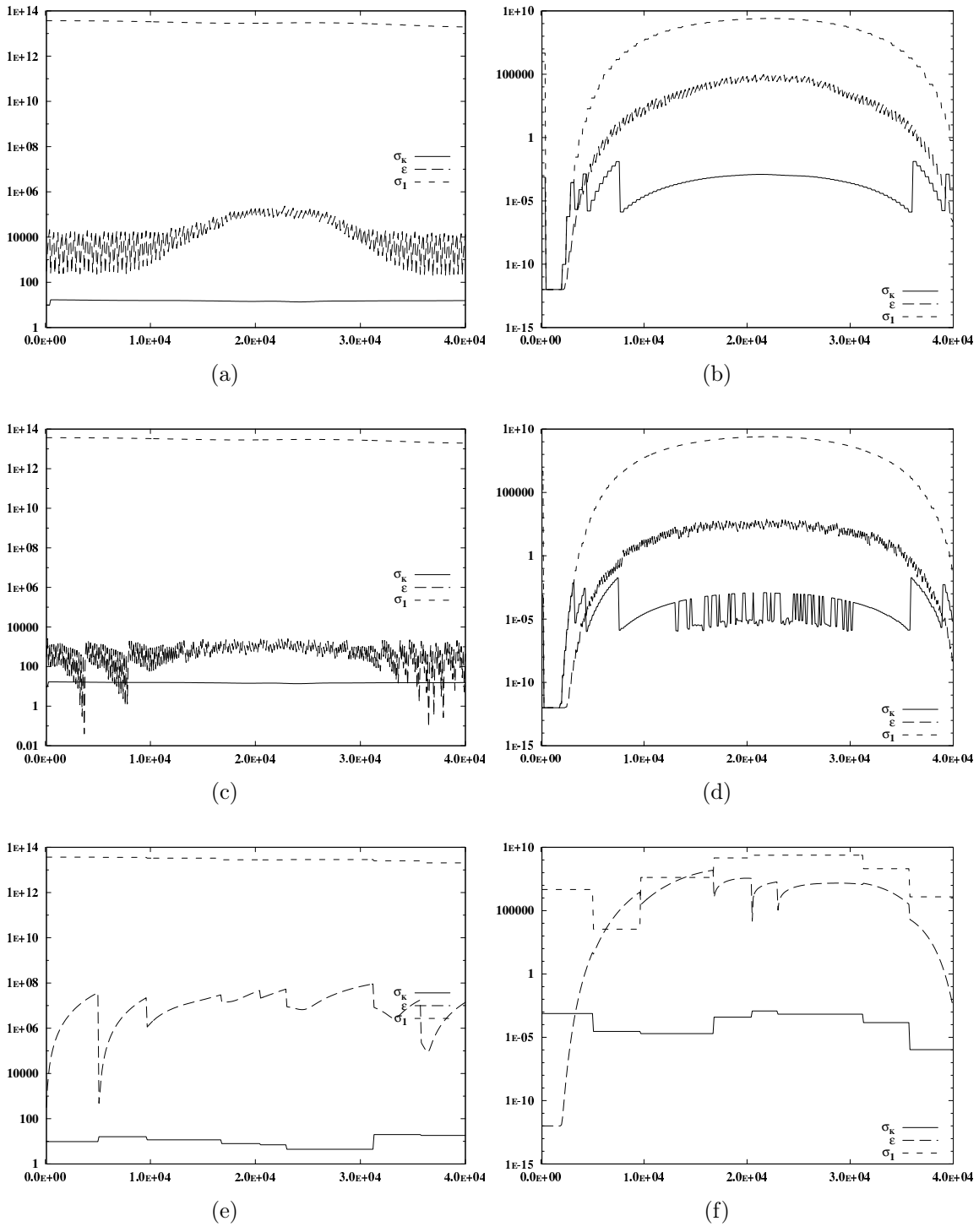


FIGURE 4.7 – Le résidu (ε), la plus grande(σ_1) et la plus petite(σ_k) des valeurs singulières pour les différentes méthodes. Les figure de la colonne à gauche représentent les valeurs à la composante c^2, celle de droite correspond à la composante c^1. Les figures (a) et (b) sont obtenues avec la méthode où les échanges se passent tous les $r = 2000$ pas. Les figures (c) et (d) sont avec les échanges basés sur le résidu comparé à une tolérance de 10^{-4}, les deux dernières figures (e) et (f) correspondent à la méthode où l'échange est basé sur le résidu comparé à la tolérance 1.0

Performance de la méthode de découplage par POD

Nous comparons finalement le temps d'exécution pour les algorithmes présentés. Dans le cas où la grille en espace est de 10×10, donnant à un système d'EDO de taille 200, le système découplé-réduit admet 140 variables, d'où une réduction du temps d'exécution de seulement 25%. Cette taille de 140 inconnues est une limite supérieure car nous avons pris un nombre maximum de variables possibles de la base réduite de taille 40 (40 clichés utilisés). Dans le cas où l'on fixe dynamiquement la taille en tenant compte de la troncature de la base réduite (comme dans la section 4.4.1), nous pouvons encore réduire le temps d'exécution. Nous proposons ensuite d'augmenter la taille du système pour montrer l'avantage de la méthode proposée : avec un système de taille 800 variables, le temps d'exécution est réduit d'un facteur de plus de 2 (c.f table 4.2).

TABLE 4.2 – Comparaison de la solution découplée par POD sur les différentes tailles du problème. Le premier cas correspond à une grille 10×10 ($MX = MY = 10$), le deuxième à une grille 20×20 ($MX = MY = 20$).

Nb procs	1	2	
système	S	$S1$	$S2$
Nb. de pas	400000	400000	400000
Taille du système	$10 \times 10 \times 2 = 200$	140	140
Temps écoulé (s)	258.75	200.57	200.57
Taille du sytème	$20 \times 20 \times 2 = 800$	440	440
Temps écoulé (s)	6494.51	2752.12	2752.13

4.4.2 Cas de test Navier-Stokes Incompressible

Nous présentons maintenant des résultats de la méthode POD sur l'équation de Navier-Stokes. L'équation de Navier-Stokes dérive de l'équation de conservation de masse, d'énergie et de moment sous forme équations aux dérivées partielles. Les équations utilisées dans cette partie se trouve par exemple dans [18, 40]. On rappelle l'équation de Navier-Stokes pour un fluide incompressible :

$$\frac{\partial \rho}{\partial t} + \mathbf{V} \cdot \nabla \rho = 0 \tag{4.40}$$

$$\rho \left(\frac{\partial \mathbf{V}}{\partial t} + (\mathbf{V} \cdot \nabla)\mathbf{V} \right) + \nabla p - \mu \nabla^2 \mathbf{V} = \mathbf{f}_e \tag{4.41}$$

$$\nabla^2 p = g(u, v, w) \tag{4.42}$$

où t est la variable de temps, ρ la densité, $\mathbf{V}$ la vitesse, E est l'énergie totale,p est la pression, $\mathbf{f}_e$ est la force extérieure par unité de volume. g est une fonction des composantes (u, v, w) de la vitesse pour déterminer la pression p. Les équations (4.40)-(4.42) sont dites sous formulation des variables primitives (pression-vitesse). Il est courant de les réécrire sous la formulation fonction de courant-tourbillon. En introduisant la variable vectorielle de vorticité :

$$\omega = \nabla \times \mathbf{V} \tag{4.43}$$

On obtient :

$$\frac{\partial \omega}{\partial t} + (\mathbf{V} \cdot \nabla)\omega - -\nu \nabla^2 \omega = \frac{1}{\rho} \nabla \times \mathbf{f}_e \tag{4.44}$$

$$\nabla^2 \Psi + \omega = 0 \tag{4.45}$$

où $\nu = \mu/\rho$ et Ψ est le vecteur des fonctions de courant. Par définition la fonction de courant prend la forme :

$$\mathbf{V} = \nabla \times \Psi \tag{4.46}$$

Pour le cas de l'écoulement plan bi-dimensionnel, la formulation courant-tourbillon devient très intéressante car Ψ n'a qu'une composante :

$$\mathbf{V} = \nabla \times (\Psi \mathbf{k}) \tag{4.47}$$

$\mathbf{k}$ est la normale unitaire au plan de l'écoulement. Dans ce cas bi-dimensionnel, la formulation de courant-tourbillon devient :

$$\begin{cases} \omega_t + \psi_x \omega_y - \psi_y \omega_x - \frac{1}{Re} \Delta \omega = 0 \\ \Delta \psi = -\omega \\ \psi = 0, \psi_\nu = v(t) \text{ sur } \partial\Omega \end{cases} \tag{4.48}$$

où Re est le nombre de Reynolds défini par $Re = \frac{1}{\rho}$.

On utilise ensuite une discrétisation de type BDF sur la variable du temps pour obtenir :

$$\begin{cases} \omega^{n+1} + \Delta t(\psi_x^n \omega_y^{n+1} - \psi_y^n \omega_x^{n+1}) - \frac{\Delta t}{Re} \Delta \omega^{n+1} = \omega^n \\ \Delta \psi^{n+1} = -\omega^{n+1} \end{cases} \tag{4.49}$$

On va considérer la méthode de lignes (ou l'approche par EDO) pour discrétiser le problème précédent. L'équation de Navier-Stokes peut s'écrire sous forme du problème de Cauchy suivant :

$$\frac{d}{dt} \begin{pmatrix} \omega \\ \psi \end{pmatrix} (t) = f(t, \omega, \psi), \ \begin{pmatrix} \omega \\ \psi \end{pmatrix} (0) = \begin{pmatrix} \omega_0 \\ \psi_0 \end{pmatrix} \tag{4.50}$$

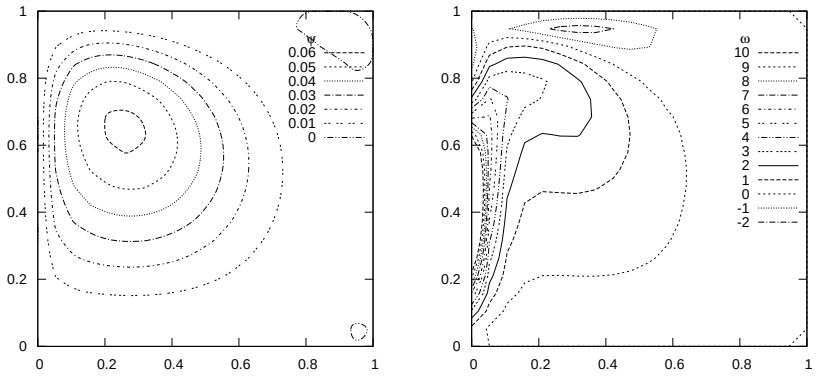
ensuite, la discrétisation en espace de $f(t, \omega, \psi)$ mène au système d'EDO suivant :

$$\frac{d}{dt} \begin{pmatrix} \omega_{i,j} \\ \psi_{i,j} \end{pmatrix} = f(t, \omega_{i[\pm 1],j[\pm 1]}, \psi_{i[\pm 1],j[\pm 1]}) \tag{4.51}$$

L'équation (4.51) peut être résolue comme une EDO classique. Selon les différents nombres de Reynold, on compare la solution obtenue avec la résolution classique du système EDO et celle de la méthode POD pour découpler le système.

Pour un nombre de Reynold $Re = 400$, le domaine de l'espace est un carré d'unité $\Omega = [0, 1] \times [0, 1]$. Ω est ensuite discrétisé en 20×20 points selon chaque direction donnant un système à 800 inconnus au système EDO.

Le pas de temps est uniforme $\delta_t = 10^{-2}$ sur un intervalle de temps $[0, 40]$ (s). L'état stationnaire est atteint pour $t > 12$.



(a) (b)

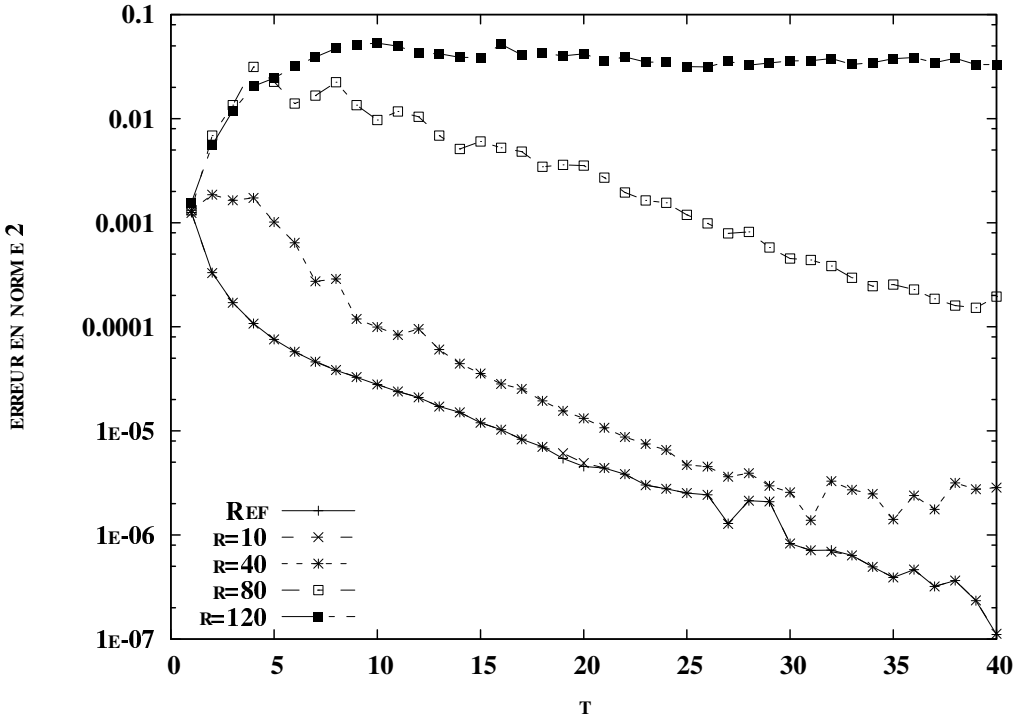
FIGURE 4.8 – ω-ψ à l'état stationnaire, $t = 40$

Résultats numériques

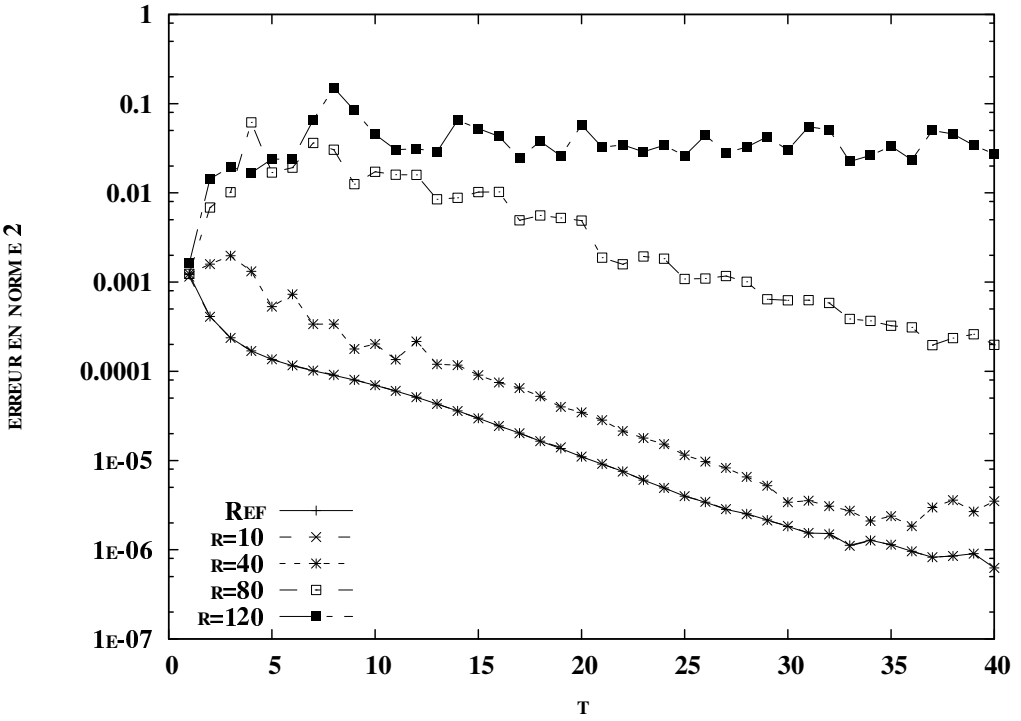
En utilisant le premier algorithme d'échange de la base POD, on obtient les solutions dont les erreurs relatives sont présentées dans la **figure 4.9**. Le pas de temps utilisé est de 10^{-2} et la base POD réduite est construite à partir des $q = 70$ derniers pas de temps. Les solutions sont comparées avec une solution de référence avec un pas de temps de 0.005. La première courbe qui sert de référence, est l'erreur relative d'une solution de référence avec un même pas de temps avec les solutions découplées-réduites à 10^{-2}. Contrairement au cas de DUK, seuls les échanges extrêmement réguliers ($r = 10$ par exemple) permettent avoir un bon résultat (les erreurs relatives à l'ordre 10^{-1} implique que la solution n'est pas acceptable). Cela est dû à la base POD locale, en effet le système réduit qui ne contient pas assez d'information pour représenter le système d'origine. D'où l'idée d'un calcul de manière incrémentale vise à enrichir la base uniquement avec les clichés qui apportent une nouvelle dynamique au système.

4.4.3 Conclusion

Les algorithmes 1 et 2 avec la stratégie de construction de la base POD basée sur les derniers clichés calculés permettent d'avoir des résultats convainquant pour la résolution en parallèle des système dynamiques. Cependant, cette stratégie de calcul local de la base réduite ne permet de simuler les sous-systèmes indépendants que lorsque l'on est dans un intervalle de temps suffisament petit. Cela implique que les échanges de la base réduite doivent être faits de manière excessive. Par conséquent, nous nous proposons dans la suite d'étendre la notion de la base POD locale par la base POD globale. Cette base POD globale dans le contexte actuel ne peut pas être calculée avant la fin de la simulation. Le chapitre suivant présente ainsi une technique pour calculer de manière incrémentale cette base POD réduite.



(a)



(b)

FIGURE 4.9 – L'erreur relative au cour du temps t sur ω-ψ avec les différents r, les solutions sont avec un pas de temps 10^{-2}. La solution de référence est celle avec un pas de temps 0.005. L'erreur relative sur ψ est donnée dans la figure (a), celle sur ω est donnée dans la figure (b)

Chapitre 5

Construction incrémentale de la POD

5.1 Construction de la base POD incrémentale appliquée à la contruction du modèle réduit

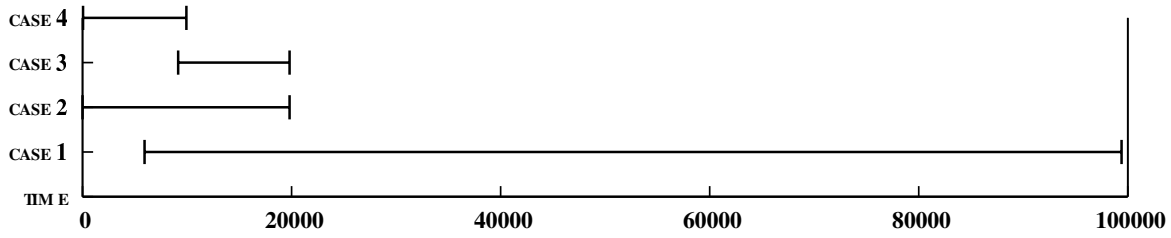
Nous poursuivons notre étude par la construction de la base réduite POD au fur et à mesure lorsque chaque sous-système avance dans le temps. Contrairement à l'approche initiale où la base POD est calculée (et recalculée) sur les q derniers pas de temps donnés, nous utilisons ici des procédures de mise à jour de la SVD afin d'obtenir une SVD des clichés sur tout l'intervalle de temps considéré. Pour cela, on s'inspire des travaux présentés dans [6] et [23].

5.1.1 Motivation de la construction incrémentale de la SVD

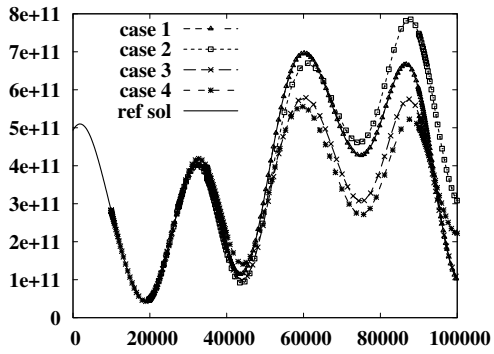
Prenons un exemple illustrant l'importance de la construction de la base réduite de manière incrémentale. Cet exemple numérique est pris de l'exemple numérique de la section 4.4.1, nous l'introduisons dans cette partie à titre d'illustration pour motiver notre algorithme de SVD incrémentale.

Considérons la relation entre le choix des clichés et la solution par le modèle réduit. Soit le système dynamique (4.5) et on s'intéresse à la solution numérique par le modèle réduit découplé comme dans (4.9). Pour construire (4.9), supposons que l'on dispose toutes solutions sur l'intervalle de temps donné (ces solutions sont obtenues par une simulation directe du système global utilisant une méthode numérique de référence). Selon les différents intervalles de temps où sont pris les clichés utilisés pour contruire la base réduite, nous résolvons (4.9) à partir du temps initial et nous comparons les solutions obtenues. La comparaison est présentée dans **la figure 5.1**.

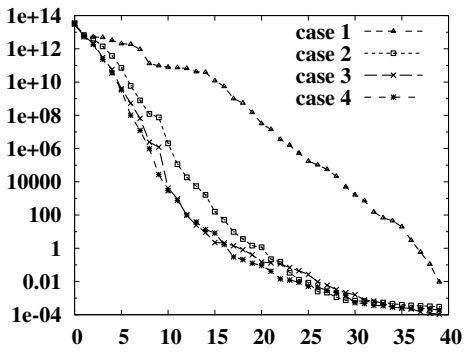
La figure 5.1a donne la position où les clichés sont pris pour construire la base réduite. Le nombre de clichés est fixé à 40 pas équidistants pour les 4 cas de figure, mais l'intervalle de construction des clichés diffère d'un cas à l'autre. La figure 5.1c donne le spectre de la SVD correspondante aux clichés pris sur les 4 cas différents. On retrouve que plus l'intervalle des clichés est large, plus on augmente le nombres de modes importants dans le spectre. Par exemple, avec une troncature à 10^{-1}, le cas 4 n'a environ que 16 modes significatifs, le cas 1 en admet 40. En effet le nombre de modes significatifs décroit



(a)



(b)



(c)

FIGURE 5.1 – Les intervalles différentes pour la matrices des clichés (a) - Les solutions obtenues par POD (b) - les spectres obtenus (c)

lorsque l'intervalle des clichés devient petit. Le manque d'information du cas 1 au cas 4 se traduit principalement dans le spectre obtenu, plus on possède d'informations (ou de dynamique) dans la matrice des clichés, plus on dispose de modes significatifs dans les modèles réduits. Par conséquent, la figure 5.1b donne la solution obtenue en résolvant le système réduit décrit dans (4.9) avec les différents modèles réduits. On peut en conclure que la meilleure base réduite pour découpler le système est celle qui possède plus de modes significatifs.

En conclusion, la contruction de la base POD sur les q derniers pas de temps résolus ne prend pas suffisament en compte la dynamique du système donné. On souhaite à priori avoir le plus d'informations (ou de clichés) possibles pour construire la base réduite. Contrairement aux exemples donnés ici, on ne possède pas de clichés sur toute l'intervalle de temps, mais les clichés ne sont disponibles qu'au fur et à mesure de la simulation. Dans cette perspective, nous nous proposons d'étudier l'algorithme de mise à jour de la SVD pour compléter les algorithmes présentés jusqu'à présent.

5.1.2 Algorithme QR pour l'orthogonalisation des matrices

La technique de mise à jour de la SVD proposée utilise les algorithmes d'orthogonalisation des matrices comme la décomposition QR. Cette section a pour objectif présenter

cette factorisation QR.

Soit $A \in \mathbb{R}^{m\times n}$ - une matrice à coefficients réels. Nous supposons que $m > n$. Par ailleurs, on suppose que A est de rang complet par rapport à ses colonnes, i.e $rang(A) = n$. La décomposition QR de la matrice A est donnée par :

$$A = QR \tag{5.1}$$

où $Q \in \mathbb{R}^{m\times m}$ est orthogonale et $R \in \mathbb{R}^{m\times n}$ est de forme triangulaire supérieure. Sous l'hypothèse que A est de rang complet par colonnes, les n-premières colonnes de Q forment une base orthonormale. La technique de calcul peut se faire par plusieurs types d'algorithmes : Householder QR, Givens QR ou encore Gram-Schmidt (et Gram-Schmidt modifié).

En ce qui nous intéresse dans la mise à jour de la SVD, on a besoin d'une réorthogonalisation de la base des vecteurs propres par une décomposition QR. On va présenter dans cette section quelques techniques usuels, leurs avantages et inconvénients (voir [20, chap.5]). Il existe plusieurs types de factorisations QR, notamment celles qui utilisent :

- Transformation de Householder (simple, et Householder par block)
- Rotation de Givens
- Transformation de Givens rapide (fast Givens transformation)
- Gram Schmidt et Gram Schmidt modifié

Les détails d'implémentation de ces factorizations se trouvent dans la plupart des ouvrages d'analyse numérique comme outils de base du calcul matriciel. L'algorithme d'orthogonalisation par Gram Schmidt modifié a été recommandé par [6] dans la procédure de la mise à jour de la SVD. Par ailleurs, la factorization par Householder est la plus utilisée en pratique (routine `xGEQRF` de Lapack). Afin de déterminer le meilleur choix entre la méthode de Householder et Gram-Schidt modifié. Nous allons les décrire en donnant leurs avantages et leurs inconvénients.

Calcul de la factorisation QR par Householder

Définissons la matrice de Householder P associée à un vecteur $v \in \mathbb{R}^n$, $v \neq 0$:

$$P = I - \frac{2}{v^\top v} v v^\top \tag{5.2}$$

le vecteur v est appelé vecteur de Householder. Si un vecteur x est multiplié par P, on dit alors qu'il est reflété dans l'hyperplan généré par $span\{v\}^\top$. La matrice de Householder P est symétrique et orthogonale. La transformation de Householder est une transformation élémentaire. Elle sert pour annuler une(ou plusieurs) composantes à un vecteur. Par exemple, pour éliminer certains coefficients du vecteur non nul $x \in \mathbb{R}^n$, c'est à dire que l'on cherche le vecteur v pour que la transformation Px soit un multiple du vecteur de la base canonique $e_1 = (1, 0, \ldots, 0)^\top \in \mathbb{R}^n$. Remarquons que si $Px \in span\{e_1\}$, alors $v \in span\{x, e_1\}$. Posons $v = x + \alpha e_1$ pour obtenir :

$$v^\top v = x^\top x + 2\alpha x_1 + \alpha^2$$

par suite :

$$Px = \left(1 - 2\frac{x^\top x + \alpha x_1}{x^\top x + 2\alpha x_1 + \alpha^2}\right) x - 2\alpha \frac{v^\top x}{v^\top v} e_1$$

Si on pose $\alpha = \pm\|x\|_2$, alors $v = x \pm \|x\|_2 e_1$, on obtient donc :

$$Px = \mp\|x\|_2 e_1$$

Ainsi, la détermination de la matrice de Householder consiste à calculer le vecteur de Householder v. Tout d'abord posons la question sur le signe plus ou moins dans la définition de Px. Le fait de poser $v_1 = x_1 - \|x\|_2$ donne que produit Px est un multiple de e_1 avec une constante positive.

Soit $x \in \mathbb{R}^n$, l'algorithme suivant calcule $v \in \mathbb{R}^n$ avec $v_1 = 1$ et $\beta \in \mathbb{R}$ telle que la matrice $P = I_n - \beta v v^\top$ soit orthogonale et $Px = \|x\|_2 e_1$:

Algorithme 3 Algorithme de vecteur de Householder - calcul de (v, β)

function $[v, \beta] = house(x)$
$n =$ **length**(x)
$\sigma = x(2:n)^\top x(2:n)$
$v = [1, x(2:n)]^\top$
si $\sigma = 0$ **alors**
  $\beta = 0$
sinon
  $\mu = \sqrt{x(1)^2 + \sigma}$
  si $x(1) <= 0$ **alors**
    $v(1) = x(1) - \mu$
  sinon
    $v(1) = -\sigma/(x(1) + \mu)$
  finsi
  $\beta = 2v(1)^2/(\sigma + v(1)^2)$
  $v = v/v(1)$
finsi
end function

Une propriété importante qui rend la matrice de Householder très favorable dans le calcul la décomposition QR est sa propriété sur l'erreur arrondie. Le vecteur $\hat{v}$ calculé numériquement à partir d'un vecteur x donné est proche du vecteur v à la précision de machine.

Considérons ensuite comment utiliser la matrice de Householder dans la factorisation QR d'une matrice A. Pour une matrice A de taille $m \times n$, soit a_i la i-ème colonne de A, une étappe de QR par Householder consiste à trouver la matrice de Householder $H_i = diag(I_i, \tilde{H}_i)$ telle que le produit $H_i A$ éliminine la partie sous-diagonale de la i-ième colonne de la matrice A. Après avoir déterminé n matrices de Householder, la décomposition QR est obtenue par $Q = H_1 \cdots H_n$ et $R = H_n H_{n-1} \cdots H_1 A$.

Algorithme 4 QR par Householder

pour $j = 1 : n$ **faire**
  $[v, \beta] = \mathbf{house}(A(j : m, j))$
  $A(j : m, j : n) = (I_{m-j+1} - \beta v v^\top) A(j : m, j : n)$
  si $j < m$ **alors**
    $A(j + 1 : m, j) = v(2 : m - j + 1)$
  finsi
fin pour

A la sortie de l'algorithme, le contenu de la matrice A sera remplacé par la matrice R triangulaire supérieure (diagonale incluse), la partie triangulaire inférieure contient les vecteurs de Householder. La matrice Q est stockée sous la forme de factorisation des vecteurs de Householder.

Pour conclure, la décomposition de QR par Householder utilise $4(m^2n - mn^2 + n^3/3)$ flops si la matrice Q est explicitement calculé. L'algorithme où Q est sous forme d'une factorisation compacte avec des vecteurs de Householder utilise $2n^2(m - n/3)$ flops.

Factorisation partielle QR par Gram Schmidt modifié

On applique la méthode de Gram-Schmidt pour calculer la factorisation QR d'une matrice $A \in \mathbb{R}^{m\times n}$ où $rang(A) = n$ - on dit A est de rang complet par rapport à ses colonnes. On a le théorème suivant :

Théorème 5.1. *Si $A = QR$ est une factorisation QR et que $rang(A) = n$. Si de plus $A = [a_1, \ldots, a_n]$ et $Q = [q_1, \ldots, q_m]$ sont les partitionnements de A et Q respectivement, alors :*

$$span\{a_1, \ldots, a_k\} = span\{q_1, \ldots, q_k\},\ k = 1 : n$$

en particulier

$$A = Q_1 R_1 \tag{5.3}$$

avec $Q_1 = Q(1 : m, 1 : n)$ et $R_1 = R(1 : n, 1 : n)$.De plus, Q_1 et R_1 sont uniques.

La preuve du théorème se trouve dans [20].

Pour calculer la factorisation QR de A, l'algorithme de Gram-Schmidt consiste à identifier les matrices dans (5.3) colonne par colonne. Remarquons que :

$$a_k = \sum_{i=1}^{k} r_{ik} q_i \in span\{q_1, \ldots, q_k\} \tag{5.4}$$

Si $rang(A) = n$, on a :

$$q_k = \frac{a_k - \sum_{i=1}^{k-1} r_{ij} q_i}{r_{kk}} \tag{5.5}$$

alors q_k peut se calculer par la normalisaton de :

$$z_k = a_k - \sum_{i=1}^{k-1} r_{ik} q_i$$

On a l'algorithme de Gram Schmidt classique (CGS pour Classical Gram-Schmidt) suivant :

Algorithme 5 Algorithme de CGS

$R(1,1) = \|A(:,1)\|_2$
$Q(:,1) = A(:,1)/R(1,1)$
pour $k = 2 : n$ **faire**
  $R(1:k-1,k) = Q(1:m,1:k-1)^\top A(1:m,k)$
  $z = A(1:m,k) - Q(1:m,1:k-1)R(1:k-1,k)$
  $R(k,k) = \|z\|_2$
  $Q(1:m,k) = z/R(k,k)$
fin pour

La méthode CGM est numériquement incorrecte due aux erreurs arrondies. On propose une version dite Gram-Schmidt modifié (MGS pour modified Gram-Schmidt) :

Algorithme 6 Algorithme de MGS

pour $k = 1 : n$ **faire**
  $R(k,k) = \|A(1:m,k)\|_2$
  $Q(1:m,k) = A(1:m,k)/R(k,k)$
  pour $j = k+1 : m$ **faire**
    $R(k,j) = Q(1:m,k)^\top A(1:m,j)$
    $A(1:m,j) = A(1:m,j) - Q(1:m,k)R(k,j)$
  fin pour
fin pour

Au niveau de l'implémentation, il n'est pas possible d'écraser A avec Q_1 et R_1 ensemble. Il faut en conséquence remplacer A par Q_1 et réserver l'espace nécéssaire pour R_1.

Remarque 5.1. *Une propriété importante de la méthode MGS est dans la partitionnement des colonnes. En effet si on suppose que certaines des colonnes de A sont déjà orthonormales. La méthode MGS peut s'appliquer de manière partielle pour ne prendre en compte que la partie orthonormée et continuer à calculer le reste de la matrice. Cette propriété est très intéressante dans le calcul de la SVD incrémentale. Par contre, l'hypothèse de l'utilisation de MGS est très stricte, il faut que la matrice soit de rang plein par rapport à ses colonnes. On va voir dans la suite de cette partie ,que la mise à jour de la SVD ne nous mène pas forcément dans le cadre d'application de MGS.*

L'effort d'évaluation de la factorisation de QR par MGS est de $2mn^2$ flops.

5.1.3 Mise à jour de la décomposition en valeurs singulières

La décomposition en valeurs singulières mène à une bi-factorisation de la matrice de clichés $A \in \mathbb{R}^{m\times n}$, on peut éventuellement supposer que $m \geq n$, la SVD de A s'écrit

$$A = U\Sigma V^\top \tag{5.6}$$

où $U \in \mathbb{R}^{m\times n}$ et $V \in \mathbb{R}^{n\times n}$ sont des matrices orthogonales contenant des vecteurs singuliers, $\Sigma = diag(\sigma_1, \ldots, \sigma_n)$ contenant des valeurs singulières classées dans l'ordre décroissant. On peut tronquer le spectre des valeurs singulières à k premières valeurs $\sigma_k, k \leq n$. Cette troncature permet de reconstruire la matrice $A_{(k)}$ qui est une approximation de rang k de la matrice A. Cette approximation est optimale au sens de la norme de Frobenius définie par $\|A\|_F = \left(\sum_{i,j} \|a_{ij}\|^2\right)^{\frac{1}{2}}$. c'est à dire $A_{(k)}$ minimise la norme de Frobenius $\|A - A_{(k)}\|_F$ parmi toute les matrices de rang k.

$$A_{(k)} = U_{(k)}\Sigma_{(k)}V_{(k)}^{\top} \tag{5.7}$$

où $A_{(k)} \in \mathbb{R}^{m\times n}$, $U_{(k)} \in \mathbb{R}^{m\times k}$, $\Sigma_{(k)} \in \mathbb{R}^{k\times k}$, $V_{(k)} \in \mathbb{R}^{n\times k}$.

On souhaite mettre à jour la SVD de la matrice A avec les nouvelles colonnes (qui représentent des nouveaux clichés) $A_u \in \mathbb{R}^{m\times u}$. Chaque colone de la matrice A_u contient un nouveau cliché.

Posons $L = U^{\top}A_u$ la projection de C sur la base orthonormale définie par les vecteurs colonnes de U. Soit $H = (I - UU^{\top})A_u = A_u - UL$ - la composante orthogonale de A_u au sous-espace engendré par U. Soit J une base orthonormale de H et posons $K = J^{\top}H$ la projection de A_u sur l'espace orthogonal à U. À titre d'exemple, on considère JK comme une décomposition de type QR de H. On a :

$$\begin{aligned} \begin{bmatrix} U & J \end{bmatrix} \begin{bmatrix} \Sigma & L \\ 0 & K \end{bmatrix} \begin{bmatrix} V & 0 \\ 0 & I \end{bmatrix}^{\top} &= \begin{bmatrix} U\Sigma V^{\top} & UL + JK \end{bmatrix} \\ &= \begin{bmatrix} A & A_u \end{bmatrix} \end{aligned} \tag{5.8}$$

où on a utilisé $JK = H = (I - UU^{\top})A_u = A_u - UL$. Remarquons qu'à droite de l'équation (5.8), les matrices sont orthogonales. Notons :

$$Q = \begin{bmatrix} \Sigma & L \\ 0 & K \end{bmatrix} \tag{5.9}$$

la matrice du milieu - notée par Q - est de forme diagonale sauf sur les u dernières colonnes. Pour mettre à jour la nouvelle SVD, diagonalisons cette matrice Q :

$$Q = U'\Sigma' V'^{\top} \tag{5.10}$$

La nouvelle SVD peut s'obtenir par :

$$[M\ C] = U''\Sigma'' V''^{\top} \tag{5.11}$$

où :

$$U'' = [U\ J]U',\ \Sigma'' = \Sigma',\ V'' = \begin{bmatrix} V & 0 \\ 0 & I \end{bmatrix} V' \tag{5.12}$$

En effet, une mise à jour de la SVD commence avec la décomposition du nouveau vecteur d'observations en deux parties, l'une est incluse dans l'ancienne base SVD tandis que l'autre est dans son orthogonal. La composante incluse va ajouter des rotations supplémentaires à l'ancienne base, tandis que la composante orthogonale va contribuer dans spectre de l'ancienne SVD. On peut ainsi résumer l'algorithme de mise à jour de la SVD par l'ajout des colonnes en trois étapes :

- Orthogonaliser l'ensemble des vecteurs de A_u par rapport à l'ancienne base de vecteur propre : $[U\ A_u] = [U\ J]R$. Cette étape est faite avec une décomposition de type QR.
- Former la matrice Q définie dans (5.9) à partir de l'ancienne diagonale et la matrice R calculée précédemment. Ensuite diagonaliser Q comme dans (5.10)
- Multiplier les matrices obtenues dans (5.12) pour obtenir la nouvelle SVD.

La procédure de mise à jour de la SVD prend $\mathcal{O}((m+n)r^2 + mc^2)$ d'ordre de temps d'exécution. L'évaluation de la SVD de la matrice d'observations prend $\mathcal{O}((m+c)n^2 + (m+c)^2n + n^3)$ en ordre de temps.

5.1.4 Comparaison de performance des méthodes de calcul de la SVD

Considérons dans la suite les coûts de calcul de la SVD des méthodes proposées jusqu'à présent. Les méthodes de mise à jour de la SVD comparées sont celles utilisant Gram-Schmidt modifié, Householder et celle qui évalue globalement la SVD.

Rappelons les coûts de la SVD d'une matrice de réelles de taille $m \times n$ avec $m > n$:

- Calcul de U, S, V^T SVD : $6mn^2 + 20n^3$ (R-SVD, voir [20, chap. 5])

A titre d'illustration, nous nous proposons l'exemple suivant : soit une matrice A de taille $M \times N$ et on propose d'obtenir U et S de sa SVD. Par exemple, dans le spectre de S qui contient $min(M, N)$ valeurs singulière, il n'y a que $K < min(M, N)$ valeurs singulières significatives. Considérons trois méthodes pour obtenir cette SVD :

- Calcul direct de la SVD sur A (*fSVD pour "full SVD"*)
- Calcul incrémentale utilisant Gram-Schmidt modifié en partant d'une SVD de $A(1 : M, 1 : q)$ et mises à jour successives de cette SVD par $A(1 : M, j),\ j = q + 1 \ldots N$, par exemple on prendra $q = 80$. Notons cette méthode par *ISVDMGS*.
- La même procédure de la deuxième méthode mais en utilisant Householder pour la factorisation de QR. Cette méthode est notée par *ISVDHouse*.

A chaque mise à jour de la SVD, en ajoutant des colonnes les unes après les autres, $2M$ flops sont requis pour la décomposition de QR utilisant l'algorithme MGS partiel. Il y a $4M^2$ pour Householder partiel (étape 1 dans la ISVD). On dépense $6Mk^2 + 20k^3$ flops pour la diagonalisation (étape 2 de la ISVD), en générale cette étape prend $\mathcal{O}(k^3)$ flops. k est le nombre de vecteurs après la troncature actuelle dans la SVD. En générale $k \leq K$ mais on prend ici $k = K$ pour une majoration du coût de calcul. Au total il y a $N - q$ mises à jour ISVD pour atteindre la SVD de A. La complexité de chacun des ces trois méthodes est alors :

$$
\begin{aligned}
fsvd &= 6MN^2 + 20N^3 && (5.13)\\
isvdmgs &\approx 6Mq^2 + 20q^3 + (N-q)(6MK^2 + 20K^3 + 2M) && (5.14)\\
isvdHouse &\approx 6Mq^2 + 20q^3 + (N-q)(6MK^2 + 20K^3 + 4M^2) && (5.15)
\end{aligned}
$$

Par exemple, `M=100;N=800;K=20;q=10;`, on obtient : $fsvd = 1,0624\ 10^{10}$; $isvdmgs = 3.16238\ 10^8$; $isvdHouse = 2.4768\ 10^8$.

5.1.5 Discussion du choix de l'algorithme de factorisation de QR

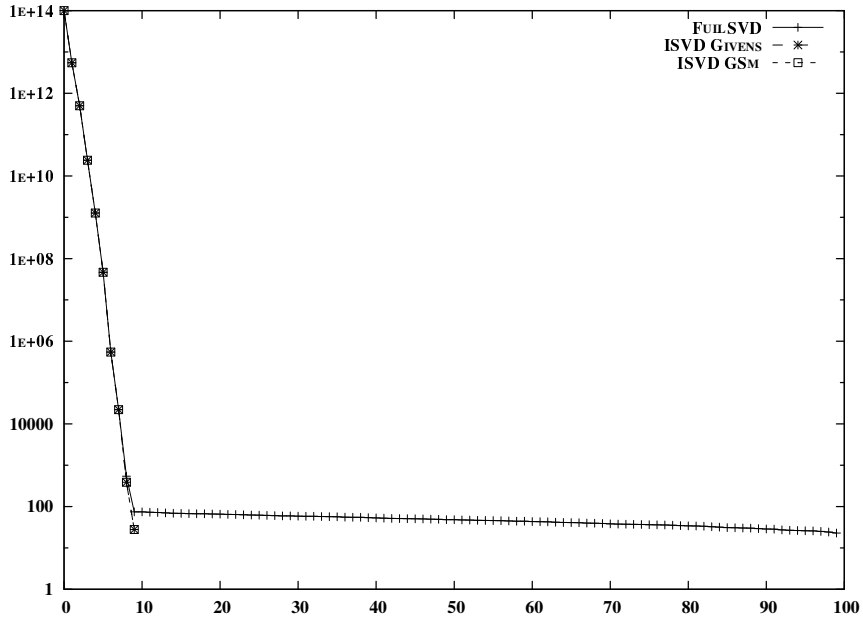


FIGURE 5.2 – Les SVD obtenus par 3 méthodes différentes : une calculée sur toute la matrice pleine, une par la SVD incrémentale utilisant l'algorithme QR de Householder, une par la SVD incrémentale utilisant l'algorithme de Gram-Schmidt modifié

La procédure de mise à jour de la SVD nécéssite en conséquence la ré-factorisation QR à la première étape de réorthogonalisation de la matrice $[U, A_u] = QR$. La première méthode est celle utilisant la méthode MGS partielle dans la section 5.1.2, et la deuxième méthode est la factorisation QR par Householder (par le routine intermédiare `xGEQRF` de Lapack). Remarquons dans 5.1 que MGS est beaucoup moins coûteux (partielle MGS), mais inappliquable dans le cas de mise à jour de la SVD. En effet, si $[A_u]$ contient des nouveaux clichés linérairement dépendants de U. Alors la matrice $[U, A_u]$ n'est pas de rang complet par rapport à ces colonnes. Les vecteurs orthonormaux obtenus par la méthode de MGS présentent dans ce cas des pertes d'optimalité. La mesure de l'optimalité de l'espace réduit est définie par la distance de l'espace des vecteurs initiaux et celui généré par la base POD obtenue. Rappelons la propriété d'optimalité d'une POD dans la proprosition 3.2 et vérifions cette propriété pour les bases réduites obtenues par les différentes méthodes de calcul de la SVD.

L'optimalité de l'espace réduit est caractérisée par :

Propriété 5.1. *Soit $U \in \mathbb{R}^{m\times k}$ "l'espace réduit" pour les observations $X = [x_1, \ldots, x_m] \in \mathbb{R}^{m\times n}$ avec $n > k$ (en générale avec une SVD, U est l'espace généré par des colonnes de vecteurs singuliers). On définit l'optimalité au sens de la norme 2 de cet espace pour représenter les données dans X par :*

$$\varepsilon_U = \max_i(\|UU^\top x_i - x_i\|_2) \tag{5.16}$$

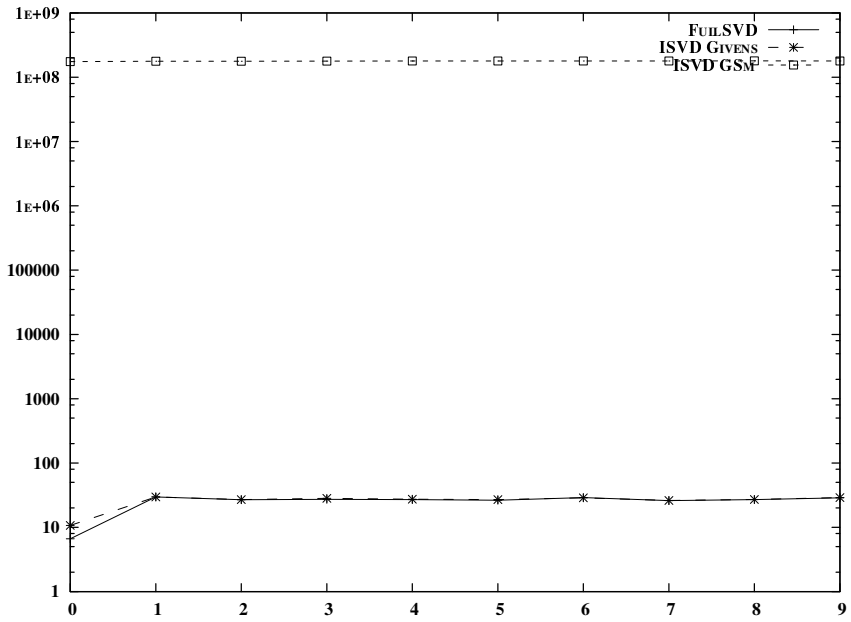


FIGURE 5.3 – Mesure de l'optimalité de la base réduite, la base obtenue par MGS n'est pas optimale. L'algorithme de mise à jour de la SVD avec Householder donne quand à lui la même optimalité que la SVD évaluée de manière globale

Si U est la base des k premiers vecteurs singuliers provenant de la SVD alors l'optimalité ε_U satisfait :

$$\varepsilon_U \leq \sum_{i=k+1}^{n} \sigma_i \tag{5.17}$$

Dans la suite, on présente les résultats de la mise en application de l'algorithme de mise à jour incrémentale de la SVD. Prenons l'exemple d'une matrice $X \in \mathbb{R}^{100\times 300}$, $X = [x_1, \ldots, x_{300}]$ représentant une collection de solutions d'un système de 100 EDO(problème DUK de la section 4.4.1) sur 300 pas de temps équidistants : $t_1 = 0, \ldots, t_{300} = 400$. Le test numérique consite à vérifier que la SVD incrémentale est capable de retrouver toutes les dynamiques présentes dans ces clichés. Comme base de référence, on calcule directement la SVD des 300 clichés inclus dans X. Ensuite on calcule la SVD de ces 300 clichés en partant de la SVD des 40 premiers clichés et en mettant à jour cette SVD par l'ajout colonne après colonne jusqu'au dernier cliché. Cette SVD incrémentale utilise soit MGS soit Householder pour la décomposition de QR.

Tous les algorithmes de mise à jour de la SVD présentés donnent des spectres tout à fait admissibles (sauf sur les petites valeurs singulières tronquées). Mais l'espace généré par des colonnes des vecteurs singuliers n'est pas optimal pour le cas de MGS. Dans la figure 5.3, l'optimalité de la base réduite obtenue avec MGS est largement au dessus de la troncature (à 10^2 comme dans la figure 5.2). Cette perte d'optimalité est due au rang non complet de la matrice. Pour cette raison, on privilégiera l'utilisation de l'algorithme de Householder pour le calcul de la SVD incrémentale.

5.2 Méthodes de découplage par POD utilisant la POD incrémentale

A partir des nouveaux algorithmes de la mise à jour de la base POD, nous proposons dans cette section des nouvelles approches pour le découplage du système dynamique. Selon les algorithmes 1 et 2 de la section 4.3, nous allons intégrer la méthode d'évaluation incrémentale de la SVD dans les algorithmes proposés.

La version améliorée de l'algorithme 1 est présentée comme suit :

Algorithme 7 Echanges sur les points fixés, avec la POD incrémentale

Initialisation de q clichés du système complet jusqu'à T_1
Calculer la SVD de chaque sous ensemble des clichés séparé : $\mathbb{X}_I$.
Initialisation de $u_{I,I}(T_1)$ et calculer la condition initiale pour $u_{I,J}(T_1)$
pour T_j : j=1 à J **faire**
  pour t_{it}=T_j à T_{j+1} **faire**
    Prendre un pas $u_I(t_{it})$
    Mettre à jour la base POD réduite avec la variable locale $u_{I,I}$ (utilisant l'algorithme incrémental SVD)
  fin pour
  Envoyer la base $\mathbb{U}_I$ et la variable réduite α_I à tous les autres solveur S_J.
  Recevoir tous les nouvelles bases des S_J
  Mettre à jour $u_{I,J},\ J \neq I$
fin pour

Nous proposons également la version finale de l'algorithme 2 avec la SVD incrémentale :

Algorithme 8 Echanges de base POD basé sur le résidu, avec la POD incrémentale

Initialisation de q clichés jusqu'à T_1
Calculer la base POD pour chacun de système séparé
Initialisation de $u_{I,I}(T_1)$ la condition initiale $u_I(T_1)$
pour t_{it}=T_1 à T_{stop} **faire**
  Faire un pas $u_I(t_{it})$
  Mettre à jour la base POD réduite avec $u_{I,I}$ (par la SVD incrémentale)
  Calculer le résidus $\epsilon_{I,i}, \forall i \neq I$ par la formule (4.31)
  si $\epsilon_{I,i} > tol$, $i \neq I$ **alors**
    Communiquer à (S_i) pour la base $\mathbb{U}_i$ de (S_i)
    Mettre une réception non-bloquante pour $\mathbb{U}_i, \alpha_i(t_{it})$ de tous les (S_i)
    Envoyer (non bloquant) $\mathbb{U}_I$ et $\alpha_I(t_{it})$ à tous les (S_i).
  finsi
  Terminer toutes les communications (si elles existent)
  si Une nouvelle base est reçue **alors**
    Réinitialiser la condition initiale au moment t_{it}
  finsi
fin pour

Nous allons présenter dans la section suivante les résultats numériques obtenus avec la SVD incrémentale.

5.3 Résultats numériques

Reprenons le cas test numérique sur l'équation de Navier-Stoke en formulation $\omega - \psi$ décrit dans le chapitre 4. L'objectif du test suivant est de valider la SVD incrémental pour construire la base POD des systèmes découplés. Nous étudions l'apport cet algorithme incrémental dans l'algorithme de découplage 7 basé sur l'échange régulier des bases réduites, ainsi que dans l'algorithme de découplage 8 qui utilise le critère sur le résidu pour échanger les bases.

Nous prenons comme solution de référence celle obtenue par le système d'EDO couplé (4.51) avec un pas de temps pris à 10^{-4}. Les autres paramètres de la simulation sont ceux de la section 4.4.2 : $Re = 400$, $NX = 20, NY = 20$.

5.3.1 Evolution au cours du temps de la SVD incrémentale pour chacune des composantes ω-ψ de la solution couplée.

Ce premier résultat numérique nous permet de montrer que la SVD incrémentale de à même de capturer l'ensemble des dynamiques présentes dans chacune des composantes de la solution.

Ce test considère la solution couplée avec un pas de temps 10^{-2}. Nous sauvegardons les 4000 clichés de la solution correspondants à tout l'intervalle de simulation $[0, 40]$. Les bases POD réduites pour ω et ψ sont initialisées par le calcul des SVD des matrices des 40 clichés initiaux. La base POD réduite est mise à jour par l'ajout successif de nouveaux clichés. L'objectif est de voir l'évolution des bases POD associées à chacune des deux composantes $\omega - \psi$ calculées par cet algorithme de SVD incrémentale.

La **figure 5.4** présente le spectre de chacune des composantes pour $t = 1$, $t = 7$ et $t = 40$, ainsi que le nombre de modes significatifs présents dans la SVD incrémentale au cours du temps. Le critère de troncature pour le spectre est le suivant :

$$\sigma_k > \max\{10^{-11}\sigma_1, 10^{-6}\} \tag{5.18}$$

Les figures 5.4c et 5.4d montrent que les modes significatifs pour chacune des composantes augmentent progressivement jusqu'à atteindre une limite de 25 pour ψ et de 70 pour ω pour $t > 12$.

Les figures 5.4e et 5.4f montrent cependant que la plus grande valeur singulière continue à évoluer au cour du temps. Par conséquent, on a bien une adaptation de la base réduite par rapport au phénomène physique sous-jacent.

Les figures 5.4a et 5.4b montrent les propriétés d'entrelacement des valeurs singulières mises à jour. Ce spectre des valeurs singulières tends vers une limite qui correspond à l'état stationnaire du système atteint pour $t = 40$.

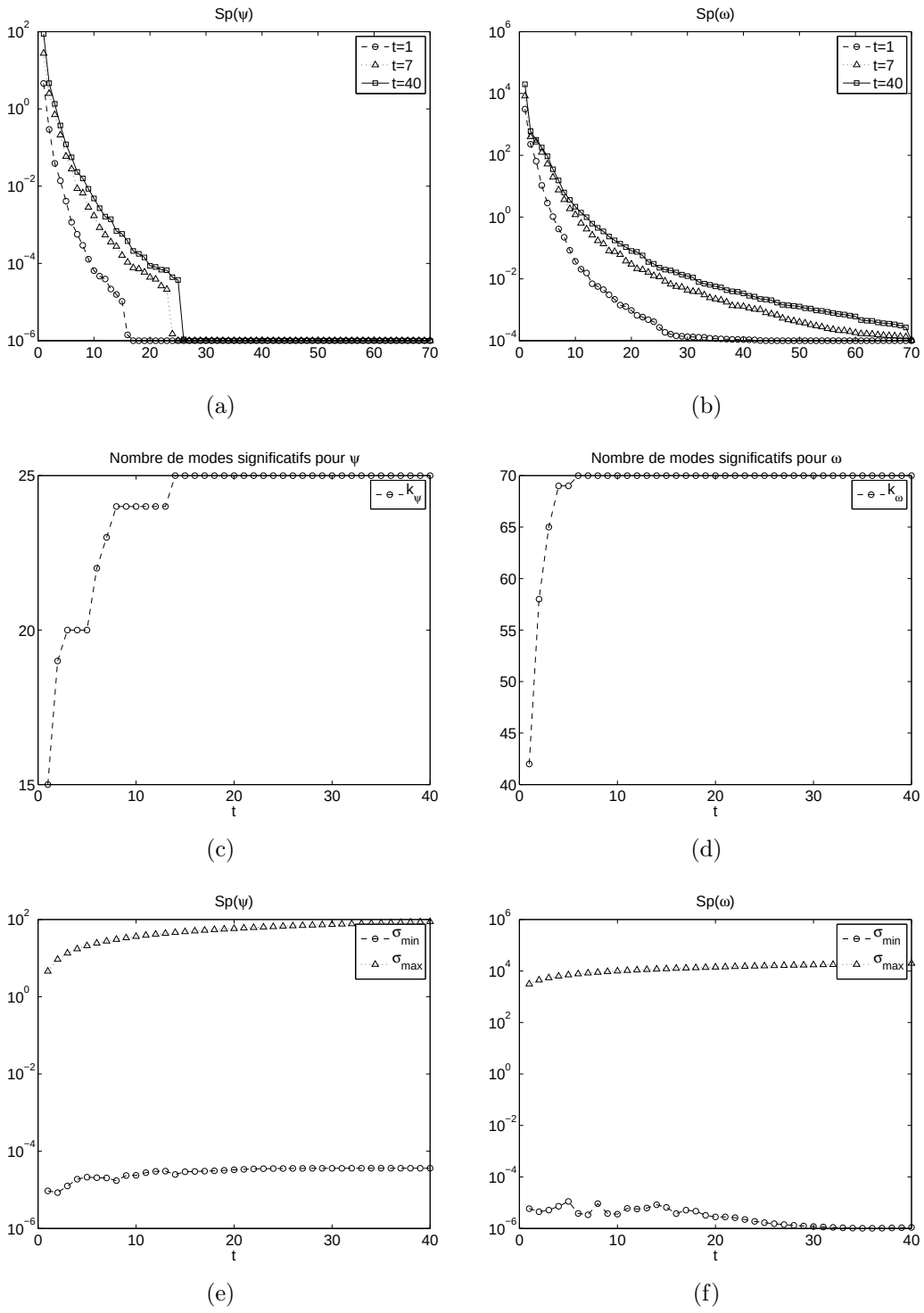


FIGURE 5.4 – SVD incrémentale : on compare le spectre contruit de manière incrémentale aux différents temps données. A partir de $t > 12$, le spectre n'évolue plus car le système atteint son état stationnaire. L'avancement en temps de la solution n'apporte plus aucune information au système réduit. Les figures à gauche (a), (c) et (e) donnent les informations du spectre calculé sur la composante ψ, celles de droite (b), (d) et (f) présentent le spectre calculé sur la composante ω

5.3.2 Simulation du système découplé-réduit avec la POD incrémentale

Nous nous intéressons désormais à une étude numériques des capacités à représenter correctement la solution couplée pour le système découplé-réduit où la POD est calculée de manière incrémentale.

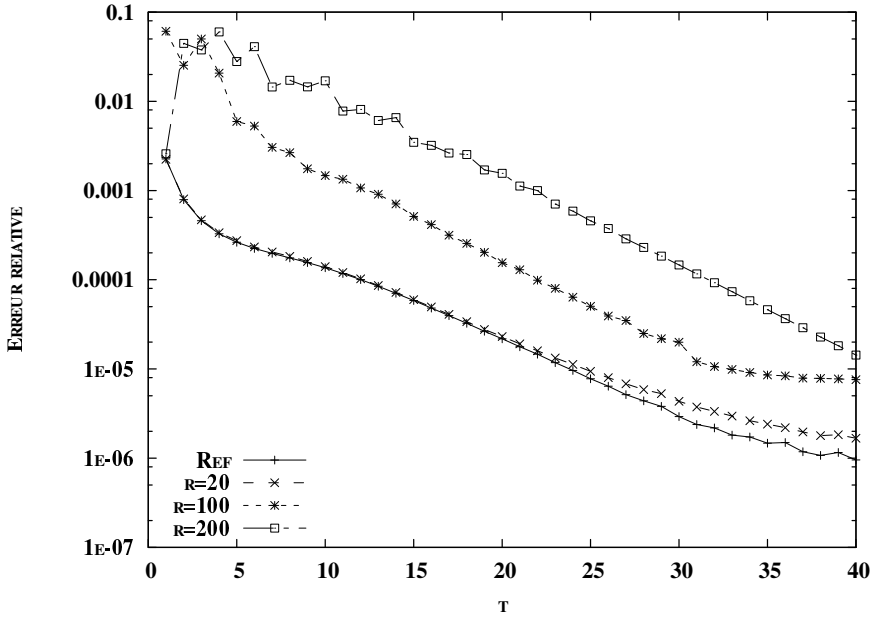
Résultat obtenu par l'algorithme 7 avec les échanges fixes

Les figures 5.5 et 5.6 représentent les erreurs relatives entre la solution découplée-réduite où la base POD est calculée de manière incrémentale et la solution de référence calculée avec un pas de temps de 10^{-4}. La courbe servant de référence est celle indiquant l'erreur relative entre deux solutions couplées calculées respectivement avec un pas de temps de 10^{-2} et de 10^{-4}. L'objectif de cette analyse pour les solutions découplées-réduites (ici avec pas de temps de 10^{-2}) est de retrouver au mieux la même précision que la solution couplée (avec un pas de temps 10^{-2}).

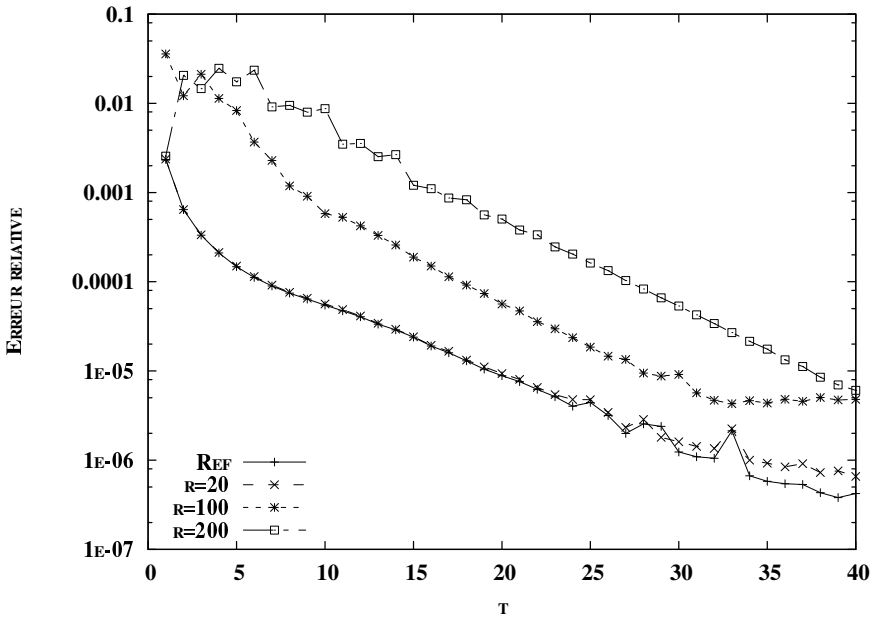
Nous comparons les résultats obtenus avec différentes fréquences des échanges $r = \{20, 100, 200\}$. Nous fixons par ailleurs la taille maximale du système réduit à $q = 70$, ceci conduit à des sous-systèmes de taille fixée à 470 variables. Ces résultats montrent que pour $r = 20$, on obtient une solution de même précision que la solution de référence avec un même pas de temps de 10^{-2}. La perte de précision en fin de simulation n'est pas significative comparée à l'erreur de consistence du schéma. Contrairement aux résultats obtenus pour la POD locale du chapitre 4 (figure 4.9), où une mauvaise solution est obtenue pour des valeurs de $r = 80$ ou plus, la méthode utilisant la SVD incrémentale donne une solution beaucoup plus précise. Notamment la solution obtenue avec $r = 200$ est plus précise que celle obtenue avec $r = 120$ dans la figure 4.9.

Cependant, dans les cas où la fréquence des échanges est faible ($r = 100$ et $r = 200$), l'erreur relative dans l'état transitoire du système dépasse 5%, ce qui peut ne pas être acceptable d'un point de vue fonctionnel, d'où l'intérêt d'avoir un déclenchement de la mise à jour basé sur le résidu (voir l'algorithme 8).

Considérons les solutions obtenues avec l'algorithme 1 avec un pas de temps de 10^{-3} au lieu de 10^{-2}. L'erreur relative de référence est celle obtenue entre deux solutions couplées calculées respectivement avec un pas de temps à 10^{-3} et 10^{-4}. **La figure 5.6** représente l'erreur relative de la solution découplée réduite (pas de temps 10^{-3}) par rapport à la solution couplée de référence à pas de temps 10^{-4}. Ces résultats montrent que la solution donnée par la méthode où les échanges des bases réduites se font tous les $r = 20$ pas atteint la même précision que celle donnée par une méthode de résolution classique.

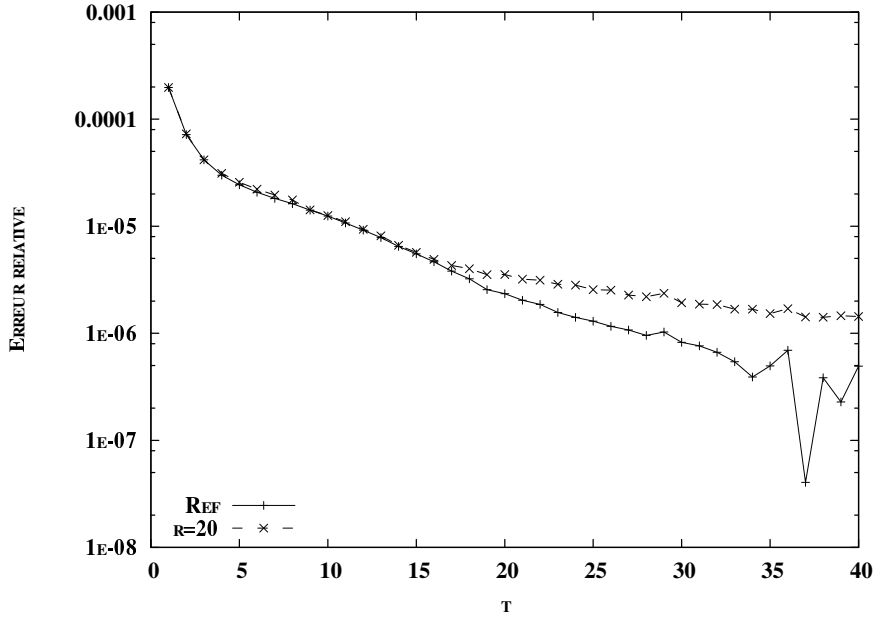


(a)

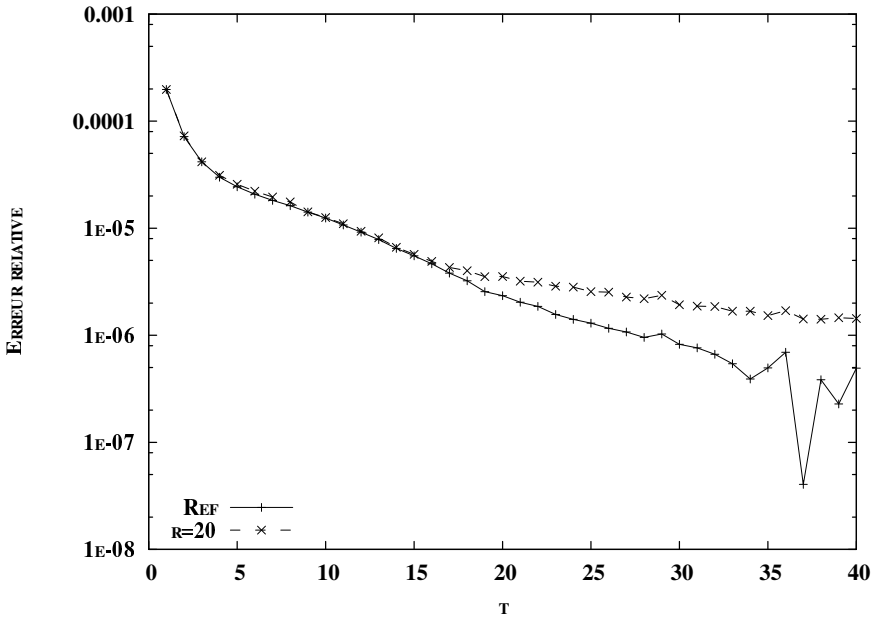


(b)

FIGURE 5.5 – Navier-Stokes, erreur relative au cour du temps en norme 2 comparée à la solution de référence : la figure (a) correspond aux erreurs calculées sur la composante ω, la figure (b) correspond à celles calculées sur ψ. La première courbe en continu représente l'erreur relative entre deux solutions de référence avec un pas de temps respectivement de 10^{-2} et de 10^{-4}. Les solutions découplées-réduites considérées sont celles où la fréquence d'échange est de tous les $r = 20$, $r = 100$ et $r = 200$ pas.



(a)



(b)

FIGURE 5.6 – Navier-Stokes, erreur relative au cour du temps en norme 2 comparée à la solution de référence : la figure (a) correspond aux erreurs en calculées sur la composante ω, la figure (b) correspond à celles calculées sur ψ. La première courbe en continu représente l'erreur relative entre deux solutions de référence avec un pas de temps respectivement de 10^{-3} et 10^{-4}. Les solutions découplées-réduites considérées sont celles où la fréquence d'échanges est de tous les $r = 20$ pas.

Résultats de l'algorithme 8 avec les échanges basés sur le critère de résidu

Considérons la solution obtenue avec l'algorithme 8 où les échanges sont déclenchés sur le critère du résidu comme dans (4.37). Les pas d'escalier dans **la figure 5.7** représentent les instants de temps où les échanges sont déclenchés, ceci pour trois valeurs de tolérances de résidu : cas 1 - $TOL = 10^{-1}$, cas 2 - 10^{-2} et cas 3 - 10^{-3}. Le nombre total des échanges sur l'intervalle de simulation pour chaque cas vaut respectivement : 17, 38 et 212 échanges. Pour le cas 3, on fait légèrement plus d'échange que le cas $r = 20$ de l'algorithme 7. Par contre, le cas 2 nous conduit à un nombre des échanges plus faible pour atteindre une précision similaire au cas $r = 20$.

Contrairement aux résultats du chapitre précédent où la base POD est évaluée sur les q derniers clichés, la base POD globale calculée de manière incrémentale tient en compte toute la dynamique sur tout l'intervalle de temps de la solution considéré. L'intérêt de la technique d'évaluation la POD de manière incrémentale est de capturer le maximum d'information sur la dynamique du système. En effet lorsque le système arrive à son état stationnaire, il n'y a pratiquement plus aucune dynamique manquante dans le modèle réduit et par conséquent les sous-systèmes n'échangent plus de bases réduites.

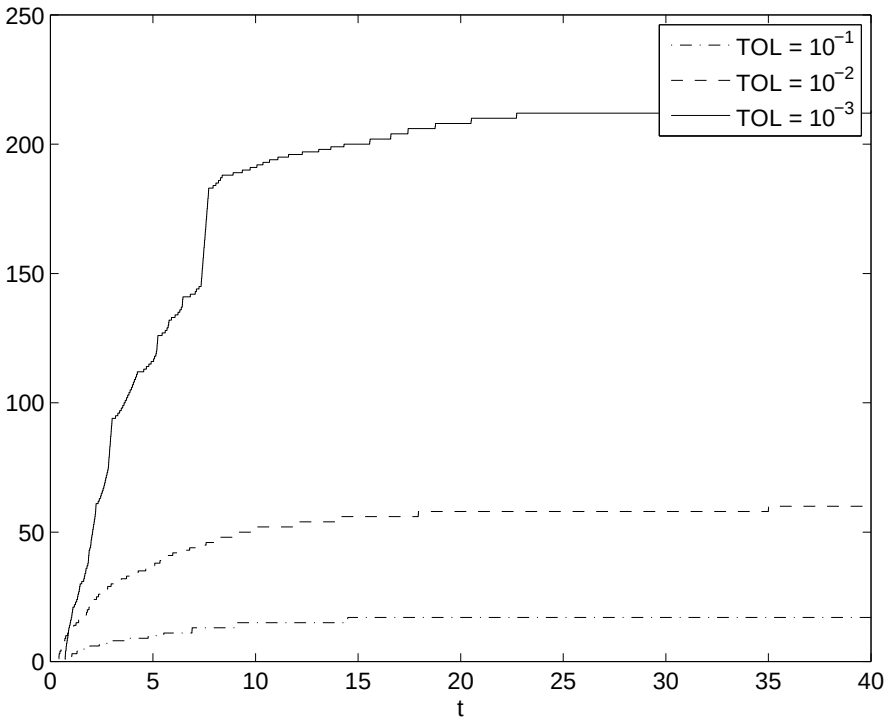
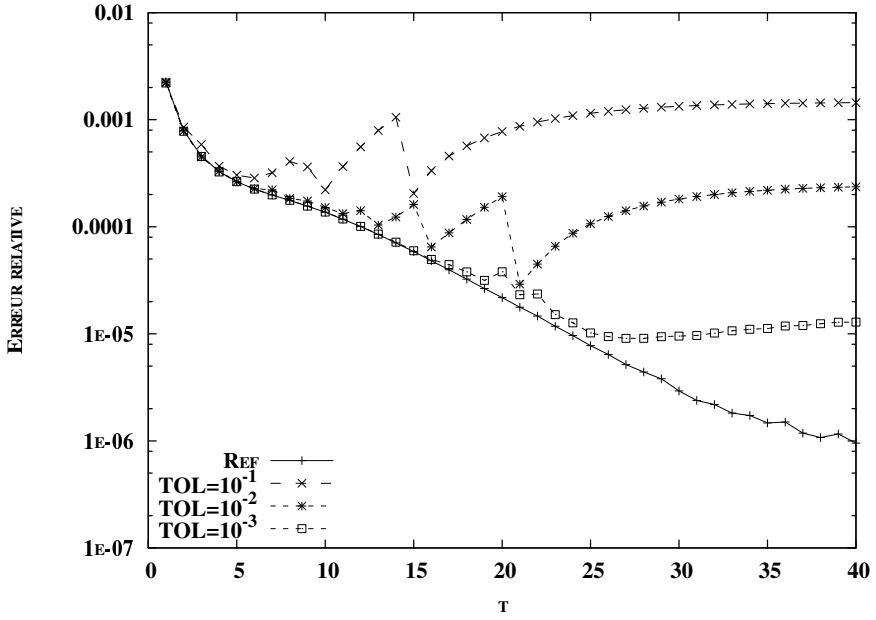


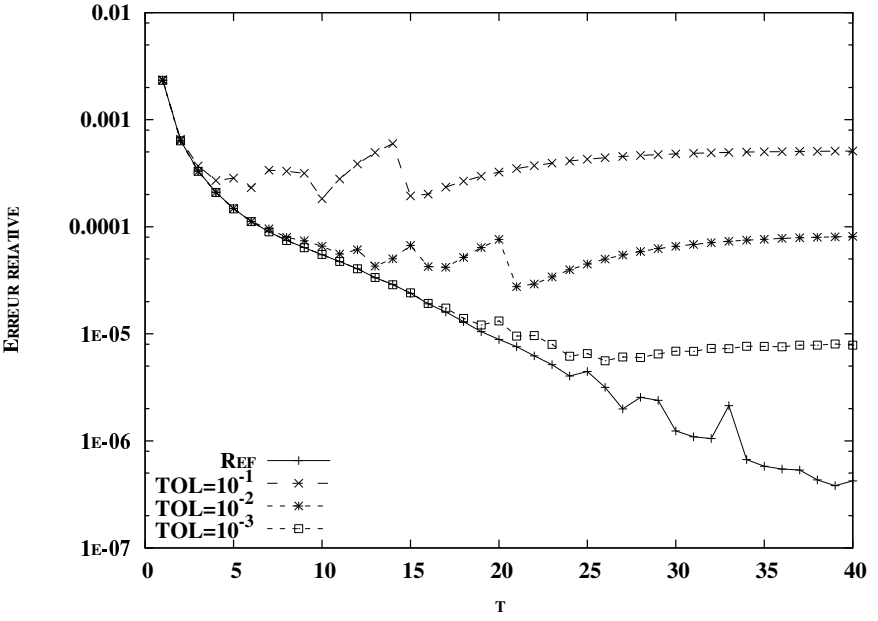
FIGURE 5.7 – Nombre d'échanges au cours du temps entre les sous-systèmes. L'algorithme utilise le critère sur le résidus pour échanger les bases POD, les valeurs de tolérances considérées sont à : 10^{-1}, 10^{-2} et 10^{-3}

La figure 5.8 compare les erreurs relatives sur les solutions découplées-réduites avec les différentes valeurs de tolérances. Comme précédemment, cette erreur relative est à

comparer avec l'erreur relative entre les deux solutions couplées de référence avec un pas de temps de 10^{-2} et de 10^{-4}. La figure 5.8a donne l'erreur relative évaluée sur la composante ψ, celle de 5.8b donne l'erreur relative sur la composante ω. On remarque bien que l'erreur relative reste bornée sur les composantes en accord avec la tolérance fixée pour le résidu. Dans ces trois cas, la solution obtenue admet une bonne précision par rapport à l'ordre de consistence de la méthode numérique utilisée avec un nombre d'échanges équivalent ou inférieur à celui utilisé dans l'algorithme 7,



(a)



(b)

FIGURE 5.8 – Erreur relative comparée à la solution de référence. La première courbe en continue représente l'erreur relative entre deux solutions de référence à pas de temps respectivement 10^{-2} et 10^{-4}. Les solutions découplées considérées sont celles où l'échange repose sur le résidus avec différentes tolérances.

Performance de la méthode proposée

Le tableau 5.1 réunit les résultats en terme de performance de l'algorithme de découplage par POD incrémentale pour l'algorithme 7 et l'algorithme 8. Ici, le problème de Navier-Stokes conduit à système d'EDO couplé de taille N variables. Le découplage en P sous-systèmes conduit à des sous-systèmes dont la partie pleine est de de N/P variables couplée avec les autres $P-1$ parties sous forme réduite de taille maximale q. Par simplicité, tous les découplages considérés sont faits de manière naturelle par séparation de la variable globale en sous-ensembles égaux des variables. Le tableau 5.1 présente les différents cas d'expériences. La colonne "Algorithme" donne l'information sur quel type de méthode utilisée : pleine signifie que l'on a résolu le système couplé de manière globale, (q, r) signifie que l'on a utilisé la base réduite de taille maximale q et les bases réduites sont échangées tous les r pas. Puis les colonnes de temps d'exécution donne le temps écoulé pour terminer la simulation, Temps (SVD) signifie le temps consacré à l'évaluation de la SVD de manière incrémentale. Finalement, la colonne "précision" donne la précision finale de chaque solution comparée à la solution couplée de référence calculée avec un pas de temps de 10^{-4}. Cette dernière est notée par $y(t_i),\ t_i \in [0, T = 40]$. Soit $y^*(t_i)$ la solution numérique obtenue avec la méthode proposée, cette précision est l'erreur relative maximale sur tout l'intervalle de temps considéré :

$$e = \max_{t_i} \frac{\|y(t_i) - y^*(t_i)\|_2}{\|y(t_i)\|_2} \tag{5.19}$$

Considérons le système d'EDO de taille 800, ce dernier provient d'une discrétisation de 20×20 en espace. La précision atteinte pour tous les cas présentés dans le tableau 5.1 est de l'ordre de $2.2\ 10^{-3}$ sauf le cas où ou a découplé en 8 sous-systèmes avec une base POD de taille maximale 10. La perte de précision de la solution obtenue est due au faible nombre de modes (10) disponible pour le système réduit. Pour la même précision de référence, le test sur plusieurs processeurs donne des bonnes performances pour résoudre le système dynamique. Les méthodes où $r = 20$ (les bases sont échangées tous les 20 pas) admettent des temps d'exécution bien inférieurs à la méthode classique. Les résultats montrent que la performance est super-linéaire jusqu'à 4 sous-systèmes. Cependant, rappelons-nous que le schéma numérique pour résoudre un pas de temps est un schéma de type BDF implicite. Le calcul d'un pas de temps nécessite en effet la résolution de type Newton d'un système non-linéaire(voir la section 2.4). Cette résolution non-linéaire fait appel à une résolution directe de décomposition de type LU à chaque pas de temps : pour un système de taille N ce coût est évalué à l'ordre de $\mathcal{O}(N^3)$ opérations. Pour la simplicité de l'implémentation, cette résolution LU est réalisée sur les matrices pleines. Par conséquent, la résolution d'un sous-système découplé-réduit est beaucoup moins coûteuse que celle du système initial.

Notre test de performance montré ici a de même une limitation. En effet, les systèmes d'EDO provenant de la méthode de ligne d'un système EDP admettent des jacobiennes de type creuse, le passage en modèle réduit n'hérite pas de cette structure creuse. Néamoins, nous montrons dans le cadre de la résolution du système dynamique générale que cette performance de la méthode de découplage réduit proposée doit être considérée comme un gain réel.

TABLE 5.1 – Comparaison de la solution découplée par POD pour les différents nombres de sous-systèmes dans le cas d'un système d'EDO de taille $20 \times 20 \times 2 = 800$ variables et celui de taille $40 \times 40 \times 2 = 3200$. Colonne R signifie le nombre total d'échanges effectués, colonnes T est le temps total écoulé pour la simulation, T (SVD) est le temps nécéssaire pour évaluer les SVD incrémentale, e est l'erreur relative définie dans (5.19)

NP	**Taille**	**Algorithme**	R	**T**(s)	**T-SVD**(s)	e
1	800	"Pleine"	–	77s	–	$2.2\ 10^{-3}$
2	440	$(q = 40, r = 20)$	200	28.6s	6.1s	$2.2\ 10^{-3}$
4	320	$(q = 40, r = 20)$	200	15.3s	2.7s	$2.2\ 10^{-3}$
8	380	$(q = 40, r = 20)$	200	19.1s	1.5s	$2.2\ 10^{-3}$
4	260	$(q = 20, r = 20)$	200	8.9s	0.7s	$2.34\ 10^{-3}$
8	170	$(q = 10, r = 20)$	200	5.1s	0.07s	$3.748\ 10^{-3}$
2	430	$(q = 30, r = 20)$	200	22.0s	1.69s	$2.2\ 10^{-3}$
4	290	$(q = 30, r = 20)$	200	11.7s	1.3s	$2.2\ 10^{-3}$
8	310	$(q = 30, r = 20)$	200	14.1s	1.17s	$2.2\ 10^{-3}$
2	440	$\varepsilon_i > 10^{-2}$	63	27.0s	5.5s	$2.2\ 10^{-3}$
4	320	$\varepsilon_i > 10^{-2}$	249	27.6s	2.27s	$2.2\ 10^{-3}$
8	380	$\varepsilon_i > 10^{-2}$	272	45.1s	1.6s	$2.2\ 10^{-3}$
1	3200	"Pleine"	–	4892s	–	$9.50\ 10^{-4}$
2	1670	$(q = 70, r = 20)$	200	2517s	153s	$9.91\ 10^{-4}$
4	1010	$(q = 70, r = 20)$	200	496s	36.6s	$1.2\ 10^{-3}$
8	890	$(q = 70, r = 20)$	200	410s	22s	$5.13\ 10^{-4}$

Conclusions et Perspectives

Ce travail de développement de méthodes numériques parallèles de découplage d'un système dynamique raide en sous-systèmes nous a amené à étudier et améliorer le schéma $C(p, q, j)$ comme introduit dans [19] puis à proposer le découplage par réduction d'ordre par POD [47].

Tout d'abord, le schéma $C(p, q, j)$ a été amélioré par l'extension du module de l'extrapolation d'ordre variable qui tient en compte de l'avancement de la solution en temps avec un coût de calcul optimisé basé sur le tableau des différences divisées. Ensuite, l'utilisation de pas variable dans les sous-systèmes nécessite de faire appel à des communications asynchrones entre les sous-systèmes. Ces communications asynchrones permettent de recouvrir le temps de communication par le temps du calcul local du sous-sytème. Cependant, notre étude montre que cette méthode $C(p, q, j)$ rencontre des limitations en terme de stabilité numérique (voir section 2.3.2) dû à l'instabilité de l'extrapolation. Notre conclusion est que cette méthode ne peut s'appliquer que dans les cas où les sous-systèmes sont faiblement couplés.

Le principal inconvénient du schéma $C(p, q, j)$ est que les interactions des dynamiques entre les sous-systèmes ne sont pas corectement prises en comptes dans le cas de systèmes fortement couplés. En effet en séparant le système initial en sous-systèmes indépendants, le fait d'extrapoler les termes de couplage ne suffit pas pour représenter les dynamiques du système. Nous avons proposé la méthode de réduction d'ordre pour remplacer ce couplage de la dynamique entre les sous-systèmes. La méthode utilisée est la réduction d'ordre par la POD. Nous avons présenté la méthode de découplage basée sur la base POD. Cette base réduite est alors basée sur les derniers clichés disponibles au cours de la simulation, puis elle est communiquée pour mettre à jour les sous-systèmes indépendants. Nous avons développé deux algorithmes concernant deux approches différentes pour échanger (mettre à jour) les bases réduites. L'un consiste à échanger de manière régulière les bases POD les plus récentes, l'autre consiste à définir un résidu sur le modèle réduit pour déclencher les échanges des bases réduites. Le deuxième algorithme montre que l'on peut éviter les échanges non-nécessaires. Cependant, lorsque l'on a à résoudre des systèmes avec un fort couplage des dynamiques, comme dans le deuxième exemple de l'équation Navier-Stokes en formulation fonction de courant et tourbillon, la technique d'évaluation de la POD sur les clichés locales ne s'applique pour un très petit nombre de pas indépendants et rencontre des difficultés à donner une solution avec précision.

Nous avons complété l'étude sur la méthode de découplage par POD en présentant l'algorithme d'évaluation de la SVD de manière incrémentale. En effet, la base POD globale - celle qui est calculée sur les clichés sur tout l'intervalle de simulation - peut s'obtenir en faisant un calcul incrémental de la SVD. Les bases POD sont alors améliorées au cours

de la simulation en parallèle. Les résultats numériques présentés montrent l'avantage de la SVD calculée de manière incrémentale appliquée à la méthode proposée. Nous avons obtenu finalement une méthode de parallélisation permettant de réduire considérablement le temps de simulation comparée à une méthode de résolution classique.

Cependant, les tests numériques conduits jusqu'à présent dans la technique de découplage par POD utilisent les schémas à pas fixe. Nous n'avons pas traiter la base POD réduite adaptée pour les schémas à pas de temps variable. Les perspectives de la thèse consisteront à une version de la POD évaluée à partir des clichés sur les pas de temps variables. Une autre perspective de poursuite de ces travaux serait de définir un critère mathématique a priori pour guider le choix du découplage du système dynamique en sous-systèmes dynamiques.

Par ailleurs, Les tests numériques fait dans ce travail sont restés relativement modestes en terme de nombres d'inconnues et du nombre de processeurs utilisés. C'est ce type de situation à laquelle les ingénieurs de conception seront confrontés sur leur portable multicoeurs à court terme. Cependant, nous pensons que l'algorithme de découplage par POD peut s'appliquer sur des machines massivement parallèles pour traiter des systèmes dynamiques à plusieurs millions d'inconnues à partir du moment où les sous-systèmes peuvent être représentés correctement avec les sous-systèmes réduits avec un bon rapport de reduction entre la dimension de la base POD et le nombre d'inconnues du sous-système complet.

Bibliographie

[1] P. A.LeGresley. *Application of Proper Orthogonal Decomposition(POD) to Design Decomposition Methods*. PhD thesis, Department of Aeronautics and Astronautics-Stanford University, October 2005.

[2] Athanasios C. Antoulas and Dan C. Sorensen. Approximation of large-scale dynamical systems : an overview. *Int. J. Appl. Math. Comput. Sci.*, 11(5) :1093–1121, 2001. Numerical analysis and systems theory (Perpignan, 2000).

[3] J. M. Bahi, K. Rhofir, and J.-C. Miellou. Parallel solution of linear DAEs by multisplitting waveform relaxation methods. In *Proceedings of the Eighth Conference of the International Linear Algebra Society (Barcelona, 1999)*, volume 332/334, pages 181–196, 2001.

[4] Gal Berkooz, Philip Holmes, and John L. Lumley. The proper orthogonal decomposition in the analysis of turbulent flows. In *Annual review of fluid mechanics, Vol. 25*, pages 539–575. Annual Reviews, Palo Alto, CA, 1993.

[5] DP Bertsekas and JN Tsitsiklis. *Parallel and Distributed Computation*. Old Tappan, NJ (USA) ; Prentice Hall Inc., 1989.

[6] Matthew Brand. Incremental singular value decomposition of uncertain data with missing values. In *In ECCV*, pages 707–720, 2002.

[7] Haïm Brezis. *Analyse fonctionnelle*. Collection Mathématiques Appliquées pour la Maîtrise. [Collection of Applied Mathematics for the Master's Degree]. Masson, Paris, 1983. Théorie et applications. [Theory and applications].

[8] Peter N. Brown, George D. Byrne, and Alan C. Hindmarsh. VODE : a variable-coefficient ODE solver. *SIAM J. Sci. Statist. Comput.*, 10(5) :1038–1051, 1989.

[9] George D. Byrne and Alan C. Hindmarsh. Stiff ODE solvers : a review of current and coming attractions. *J. Comput. Phys.*, 70(1) :1–62, 1987.

[10] P. Chartier and B. Philippe. A parallel shooting technique for solving dissipative ODEs. *Computing*, 51(3-4) :209–236, 1993.

[11] Moody T. Chu, Robert E. Funderlic, and Gene H. Golub. A rank-one reduction formula and its applications to matrix factorizations. *SIAM Rev.*, 37(4) :512–530, 1995.

[12] S.D. Cohen and A.C. Hindmarsh. CVODE, a stiff/nonstiff ODE solver in C. *Computers in Physics*, 10(2) :138–143, 1996.

[13] Colin W. Cryer. On the instability of high order backward-difference multistep methods. *Nordisk Tidskr. Informations behandling (BIT)*, 12 :17–25, 1972.

[14] C. F. Curtiss and J. O. Hirschfelder. Integration of stiff equations. *Proc. Nat. Acad. Sci. U. S. A.*, 38 :235–243, 1952.

[15] G. Dahlquist. *Stability and error bounds in numerical integration of ordinary differential equations*. PhD thesis, Royal Institute of Technology, Stockholm, 1958.

[16] James W. Demmel. *Applied numerical linear algebra*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1997.

[17] G. Edjlali, M. Garbey, and D. Tromeur-Dervout. Interoperability parallel programs approach to simulate 3d frontal polymerization processes. *Parallel Computing*, 25(9) :1161 – 1191, 1999.

[18] J. H. Ferziger and M. Perić. *Computational methods for fluid dynamics*. Springer-Verlag, Berlin, revised edition, 1999.

[19] M. Garbey and D. Tromeur-Dervout. A parallel adaptive coupling algorithm for systems of differential equations. *J. Comput. Phys.*, 161(2) :401–427, 2000.

[20] Gene H. Golub and Charles F. Van Loan. *Matrix computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, MD, third edition, 1996.

[21] W. R. Graham, J. Peraire, and K. Y. Tang. Optimal control of vortex shedding using low-order models. I. Open-loop model development. *Internat. J. Numer. Methods Engrg.*, 44(7) :945–972, 1999.

[22] W. R. Graham, J. Peraire, and K. Y. Tang. Optimal control of vortex shedding using low-order models. II. Model-based control. *Internat. J. Numer. Methods Engrg.*, 44(7) :973–990, 1999.

[23] M. Gu and S.C. Eisenstat. A stable and fast algorithm for updating the singular value decomposition. *Research Report YALE DCR/RR-966*, 1994.

[24] D. Guibert and D. Tromeur-Dervout. Parallel adaptive time domain decomposition for stiff systems of ODEs/DAEs. *Comput. & Structures*, 85(9) :553–562, 2007.

[25] David Guibert and Damien Tromeur-Dervout. A Schur complement method for DAE/ODE systems in multi-domain mechanical design. In *Domain decomposition methods in science and engineering XVII*, volume 60 of *Lect. Notes Comput. Sci. Eng.*, pages 535–541. Springer, Berlin, 2008.

[26] M. Günther, KværnøA., and P. Rentrop. Multirate partitioned Runge-Kutta methods. *BIT*, 41(3) :504–514, 2001.

[27] E. Hairer, S. P. Nørsett, and G. Wanner. *Solving ordinary differential equations. I*, volume 8 of *Springer Series in Computational Mathematics*. Springer-Verlag, Berlin, second edition, 1993. Nonstiff problems.

[28] E. Hairer and G. Wanner. *Solving ordinary differential equations. II*, volume 14 of *Springer Series in Computational Mathematics*. Springer-Verlag, Berlin, second edition, 1996. Stiff and differential-algebraic problems.

[29] Alan C. Hindmarsh. Detecting stability barriers in BDF solvers. In *Computational ordinary differential equations (London, 1989)*, volume 39 of *Inst. Math. Appl. Conf. Ser. New Ser.*, pages 87–96. Oxford Univ. Press, New York, 1992.

[30] Chris Homescu, Linda R. Petzold, and Radu Serban. Error estimation for reduced-order models of dynamical systems. *SIAM Rev.*, 49(2) :277–299, 2007.

[31] Lawrence Hubert, Jacqueline Meulman, and Willem Heiser. Two purposes for matrix factorization : a historical appraisal. *SIAM Rev.*, 42(1) :68–82 (electronic), 2000.

[32] Karl Kunisch and Stefan Volkwein. Proper orthogonal decomposition for optimality systems. *M2AN Math. Model. Numer. Anal.*, 42(1) :1–23, 2008.

[33] J. D. Lambert. *Numerical methods for ordinary differential systems*. John Wiley & Sons Ltd., Chichester, 1991. The initial value problem.

[34] Patrick A. Legresley and Juan J. Alonso. Dynamic domain decomposition and error correction for reduced order models. In *In AIAA 41st Aerospace Sciences Meeting*, 2003.

[35] Cristóbal López and Emilio Hernández-García. Low-dimensional dynamical system model for observed coherent structures in ocean satellite data. *Phys. A*, 328(1-2) :233–250, 2003.

[36] Xia Ma and George Em Karniadakis. A low-dimensional model for simulating three-dimensional cylinder flow. *J. Fluid Mech.*, 458 :181–190, 2002.

[37] M. Meyer and HG Matthies. Efficient model reduction in non-linear dynamics using the Karhunen-Loève expansion and dual-weighted-residual methods. *Computational Mechanics*, 31(1) :179–191, 2003.

[38] B. Mohammadi and O. Pironneau. *Applied shape optimization for fluids*. Numerical Mathematics and Scientific Computation. The Clarendon Press Oxford University Press, New York, 2001. Oxford Science Publications.

[39] M. R. Osborne. On shooting methods for boundary value problems. *J. Math. Anal. Appl.*, 27 :417–433, 1969.

[40] Roger Peyret and Thomas D. Taylor. *Computational methods for fluid flow*. Springer Series in Computational Physics. Springer-Verlag, New York, second edition, 1985.

[41] Alfio Quarteroni, Riccardo Sacco, and Fausto Saleri. *Numerical mathematics*, volume 37 of *Texts in Applied Mathematics*. Springer-Verlag, Berlin, second edition, 2007.

[42] S. S. Ravindran. A reduced-order approach for optimal control of fluids using proper orthogonal decomposition. *Internat. J. Numer. Methods Fluids*, 34(5) :425–448, 2000.

[43] M. Reichelt, F. Odeh, and J. White. *A*-stability of multirate integration methods, with application to parallel semiconductor device simulation. In *Parallel processing for scientific computing, Vol. I, II (Norfolk, VA, 1993)*, pages 246–253. SIAM, Philadelphia, PA, 1993.

[44] Jørgen Sand and Stig Skelboe. Stability of backward Euler multirate methods and convergence of waveform relaxation. *BIT*, 32(2) :350–366, 1992.

[45] Tapan K. Sengupta and S. Dey. Proper orthogonal decomposition of direct numerical simulation data of by-pass transition. *Computers and Structures*, 82(31-32) :2693 – 2703, 2004. Nonlinear Dynamics of Continuous Systems.

[46] R. Serban and AC Hindmarsh. CVODES, the sensitivity-enabled ode solver in SUNDIALS. In *Proceedings of the 5th International Conference on Multibody Systems, Nonlinear Dynamics and Control, Long Beach, CA*, 2005.

[47] Lawrence Sirovich. Turbulence and the dynamics of coherent structures. I. Coherent structures. *Quart. Appl. Math.*, 45(3) :561–571, 1987.

[48] Lawrence Sirovich. Turbulence and the dynamics of coherent structures. II. Symmetries and transformations. *Quart. Appl. Math.*, 45(3) :573–582, 1987.

[49] Lawrence Sirovich. Turbulence and the dynamics of coherent structures. III. Dynamics and scaling. *Quart. Appl. Math.*, 45(3) :583–590, 1987.

[50] Stig Skelboe. Methods for parallel integration of stiff systems of ODEs. *BIT*, 32(4) :689–701, 1992.

[51] Stig Skelboe. Accuracy of decoupled implicit integration formulas. *SIAM J. Sci. Comput.*, 21(6) :2206–2224 (electronic), 2000.

[52] Senol Utku, Jose L. M. Clemente, and Moktar Salama. Errors in reduction methods. *Comput. & Structures*, 21(6) :1153–1157, 1985.

[53] S. Volkwein, Universität Graz, and Technische Universität Graz. *Proper orthogonal decomposition and singular value decomposition*. Karl-Franzens-Universität Graz & Technische Universität Graz, 1999.

[54] J. White, F. Odch, and Yorktown Cights. A connection between the convergence properties of waveform relaxation and the a-stability of multirate integration methods. In *in Proceedings of the NASECODE VII Conference, Copper*, pages 73–76. Press, 1991.