وزَارة التعلِيم العَالِي و البحث العلمِي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Ecole Normale Supérieure Kouba, Alger
Département de Mathématiques

المدرسة العلِيَا لّلأَسَاتذة القبة، الجزَائِر
قسم الريَاضيَات

MÉMOIRE

Présenté pour obtenir le diplôme de

Magister

Spécialité : Analyse foncionnelle

Présenté par :

Mourad BOUSSAADA

Sujet de Mémoire :

Présentation de l'algorithme pararéel et son application.

Date de soutenance : 25/09/2012

Composition de jury :

M. Y.Atik, professeur à L'ENS-Kouba Président

M. A. Mokrane, professeur à L'ENS-Kouba Rapporteur

M. S.M. Kaber, professeur à L'UMPC-Paris Rapporteur

M. M.AMoussaoui, professeur à L'ENS-Kouba Examinateur

M. A.Choutri, Maître de conférences à L'ENS-Kouba Examinateur

Table des matières

Introduction

Pour répondre aux problèmes posés par la biologie, la physique, ou d'autres sciences, nous sommes régulièrement amenés à faire de la simulation numérique sur des modéles associés à des situations précises qui nous sont imposées par ces sciences, par exemple, l'étude de la trajectoire d'astéroïdes, de l'évolution du climat, ou des cycles stellaires...

En analyse numérique, nous sommes souvent confrontés au besoin d'effectuer des résolutions d'équations plus ou moins complexes par des méthodes approchées convergeant le plus rapidement possible, et dont l'exécution prend un temps relativement court. Le développement des technologies informatiques connaissant actuellement ses limites en termes de miniaturisation, la taille des processeurs ne peut plus diminuer drastiquement dans un futur proche, et leur rapidité risque donc de stagner. Il devient dès lors impossible pour une unité de calcul seule de résoudre dans un dèlai court une équation, dès que celle-ci fait appel à des méthodes coûteuses. Des procédés d'optimisation peuvent permettre un gain de temps, mais celui-ci devient rapidement négligeable. Il faut dès lors penser à de nouvelles méthodes pour " gagner du temps ". Une méthode simple dans le cas de programmes exécutant des tâches indépendantes est de distribuer ces tâches sur plusieurs unités de calcul, ce qui est facilement fait à l'aide d'ordinateurs à processeurs multi-coeurs.

Il arrive cependant qu'un programme soit à " mémoire temporelle ", c'est-à-dire que pour s'exécuter, une tâche ait besoin que la précédente ait été exécutée, et ainsi de suite. Ces

processus ne peuvent pas être découpés temporellement, car alors il faudrait attendre qu'une unité de calcul ait exécuté une portion de tâche pour que la suivante puisse effectuer la portion suivante, et aucun gain de temps ne pourrait être constaté. Cependant, il existe des méthodes de prédiction-correction, qui permettent à moindre coût de donner suffisamment d'informations aux unités pour qu'elles effectuent leurs tâches sans se préoccuper des autres machines, quitte à itérer plusieurs fois les dites tâches en tenant compte des résultats des itérations précédentes pour affiner leurs calculs.

Notre mémoire a porté sur une méthode, appelée méthode pararéelle, et effectuant ce genre d'approximations de prédiction, qui si elles font perdre un peu de précision par rapport à un algorithme de résolution type Euler, peuvent diviser par deux le temps de calcul total.

Pour commencer, on expose l'idée sur l'exemple simple d'une équation différentielle linéaire :

$$\begin{cases} \frac{dy}{dt} = -\mu y, & t \in [0,T] \\ y(t=0) = y_0 \end{cases} .$$

où μ est une constante réelle.

On considére le schéma d'Euler implicite

$$\begin{cases} \frac{Y^{n+1}-Y^n}{\Delta T} + \mu Y^{n+1} = 0, & t \in [0,T] \\ Y^0 = y_0 \end{cases} .$$

puis on utilise les valeurs précédemment calculées pour résoudre de façon exacte, sur chaque intervalle de temps $[T^n, T^{n+1}]$

$$\begin{cases} \frac{dy^n}{dt} = -\mu y^n, & t \in [T^n, T^{n+1}] \\ y^n(t=T^n) = Y^n. \end{cases} .$$

Nous effectuons donc deux discrétisations : une grossière pour calculer nos condtions initiales (CI) pour chaque intervalle, et une fine pour discrétiser chaque intervalle en sous-intervalles, comme le montre le schéma suivant :

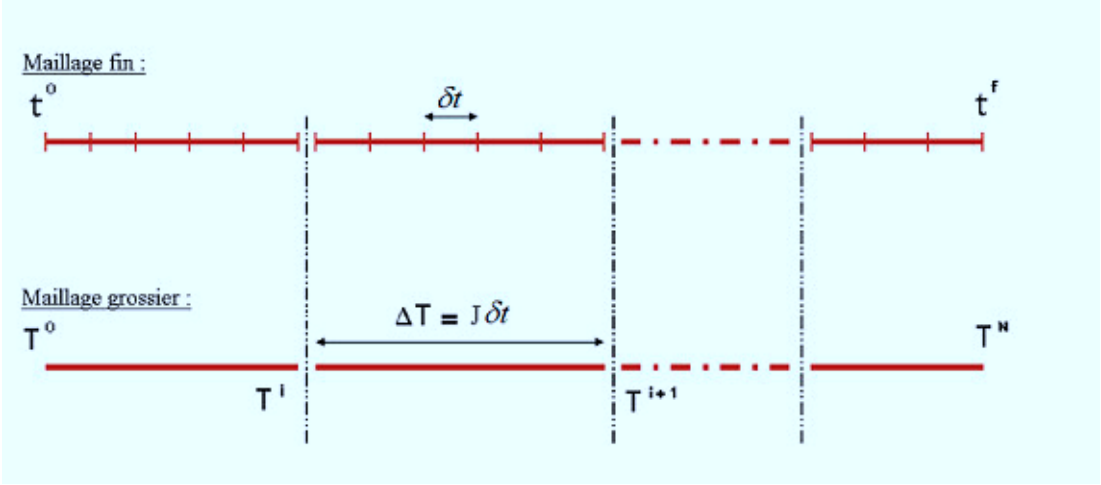

Figure 1 : Discrétisations grossière et fine en temps

Nous pouvons imaginer qu'aux instants T^n nous pourrons avoir des "sauts" dans notre solution, car juste un instant avant, nous aurons la solution calculée à l'intervalle précédant, et à cet instant, nous aurons la valeur initiale de l'intervalle, calculée de manière grossière. Nous voyons donc que notre schéma risque d'entrainer une discontinuité de notre solution calculée, comme le montre la figure 2 :

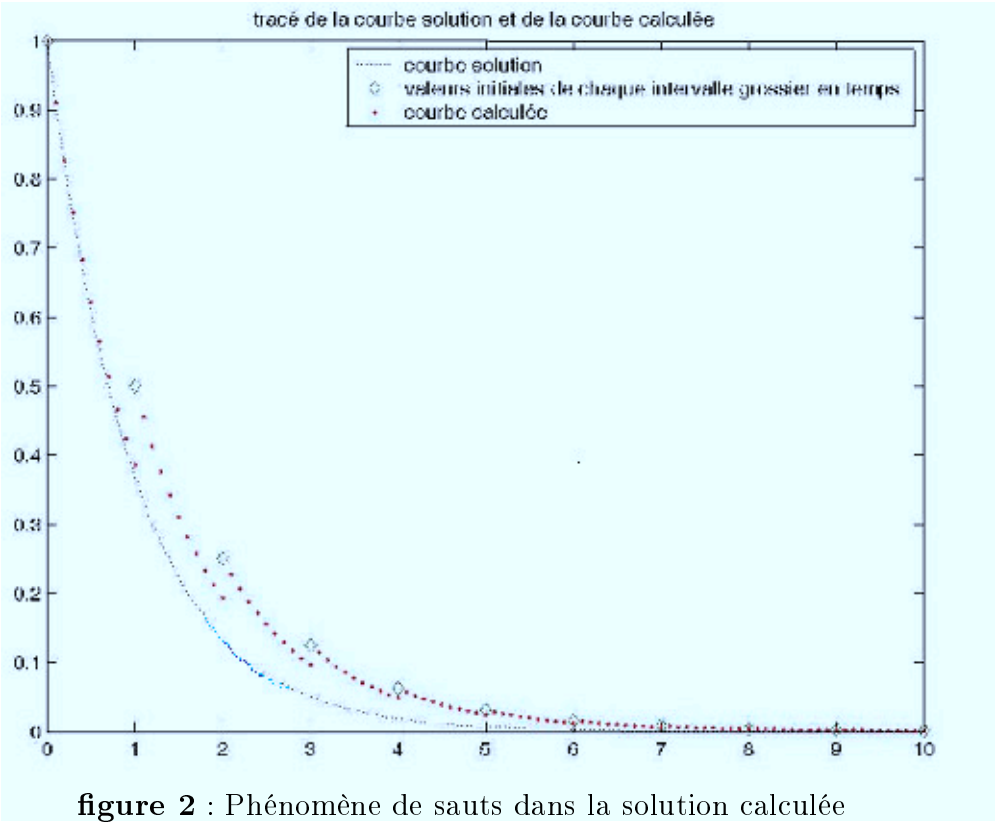

figure 2 : Phénomène de sauts dans la solution calculée

Sur cette figure, nous voyons bien que la courbe solution (en pointillés bleus) est approchée par la courbe calculée (en pointillés rouges) sur chaque intervalle de temps de la discrétisations groissière, indépendemment les uns des autres. Les valeurs initiales de chaque intervalle de temps sont représentées en losanges. Nous voyons donc que plus le choix des valeurs initiales est bon, plus la convergence est rapide. Pour résoudre ce problème, nous allons faire ce qu'on appelle la propagation des sauts, afin d'améliorer notre précision et obtenir une solution continue.

On propose maintenant une procedure iterative pour ameliorer la precision de ce schema.

On pose donc $Y_1^n = Y^n$ et $y_1^n = y^n$, définis sur l'intervalle $[T^n, T^{n+1}]$. Puis, supposant connus (Y_k^n et y_k^n sur $[T^n, T^{n+1}]$)

(i) On introduit les sauts $S_k^n = y_k^{n-1}(T^{n-1}) - Y_k^n$ pour $n = 1, ..., N-1$ et avec $S_k^0 = 0$

(ii) On propage les sauts

$$\begin{cases} \frac{\delta_k^{n+1} - \delta_k^n}{\Delta T} + \mu \delta_k^{n+1} = \frac{S_k^n}{\Delta T} \\ \delta_k^0 = 0 \end{cases} .$$

(iii) On pose ensuite $Y_{k+1}^n = y_k^{n-1}(T^{n-1}) + \delta_k^n$ et on résout de façon exacte et en parallèle le système suivant :

$$\begin{cases} \frac{dy_{k+1}^n}{dt} = -\mu y_{k+1}^n, \qquad t \in [T^n, T^{n+1}] \\ y_{k+1}^n(t = T^n) = Y_{k+1}^n \end{cases} .$$

Après avoir propagé les sauts, nous obtenons alors la figure 3 :

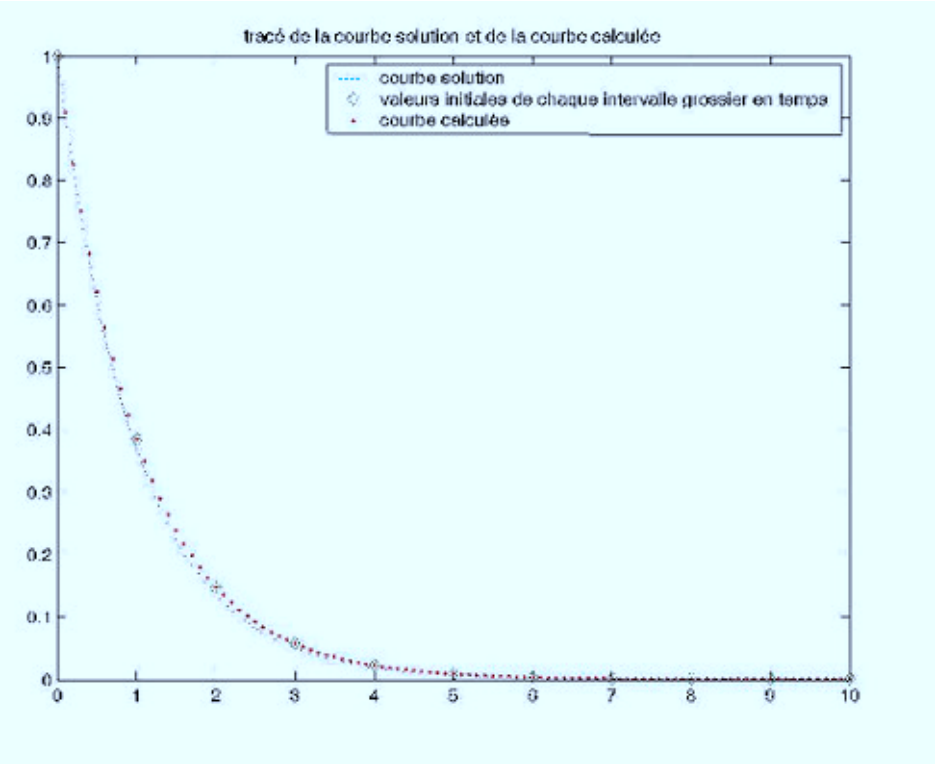

figure 3 : "Ecrasement" des sauts dans la solution calculée.

Nous voyons donc bien que les sauts sont "écrasés" et que les valeurs initiales coïncident avec la courbe.

Ce problème de propagation des sauts ralentit le calcul, car même si on parallélise notre calcul, il faut après perdre du temps pour le réajuster en propageant les sauts. Cela implique que notre méthode ne sera vraiment utile qu'avec des choix judiceux de conditons intiales, pour ne pas perdre trop de temps dans la convergence.

Notre but, tout au long de ce mémoire est de montrer l'utilité de cette méthode par rapport aux méthodes habituelles de résolution d'EDP par un schéma en temps, c'est-à-dire de comparer le schéma "**Pararéel**" aux schémas tels que Euler explicite, Runge-Kutta...

Plan du mémoire

Ce mémoire est subdivisé en trois chapitres.

• Dans le premier chapitre, nous présentons l'algorithme pararéel et quelques propriétés fon-

damentales.

• Dans le deuxième chapitre, nous démontrons la convergence et la stabilité de l'algorithme.

• Dans le dernier chapitre, nous appliquons cette méthode à un exemple issu de la biologie et à un autre issu de la météorologie.