AMS X02 : Méthodes numériques avancées et calcul haute performance pour la simulation de phénomènes complexes

Encadrants :

-Marc Massot

- Laurent Series

Méthodes IMEX et Résolution de systèmes linéaires

Hugo Negrel

Janvier 2023

Table des matières

1 Introduction

Ce troisième TP du cours AMS X02 est scindé en deux parties. En premier lieu, le sujet portera sur la méthode IMEX, qui consiste à décomposer une équation aux dérivées partielles en deux, une partie raide et une partie non raide. En deuxième partie, on s'intéressera à l'efficacité de la résolution d'un système linéaire.

2 La méthode IMEX

La méthode IMEX sera employé sur l'équation de NAGUMO, qui est la suivante :

$$\partial_t y - D\Delta y = ky^2(1-y) \quad \text{pour} - L \leq \mathrm{x} \leq L$$

En discrétisant l'opérateur laplacien, on arrive au système suivant :
$d_t U = AU + R(U)$
en sachant que $U(t) \in \mathcal{R}^N$ et que $R(U)_i = u_i^2(t)(1-u_i(t))$. L'opérateur A est raide dans la dynamique, contrairement à R. L'exercice principal est le test de plusieurs technique d'intégration, en particulier la méthode IMEX. En premier lieu, la figure 1 présente la solution quasi-exacte, calculée avec une méthode de runge-kutta d'ordre 2. Le pas de temps est dt = 0.001.

$$k_1 = AU^n + R(U^n) k_2 = A(U^n + dtk_1) + R(U^n + dtk_1) U^{n+1} = U^n + dt/2(k_1 + k_2)$$

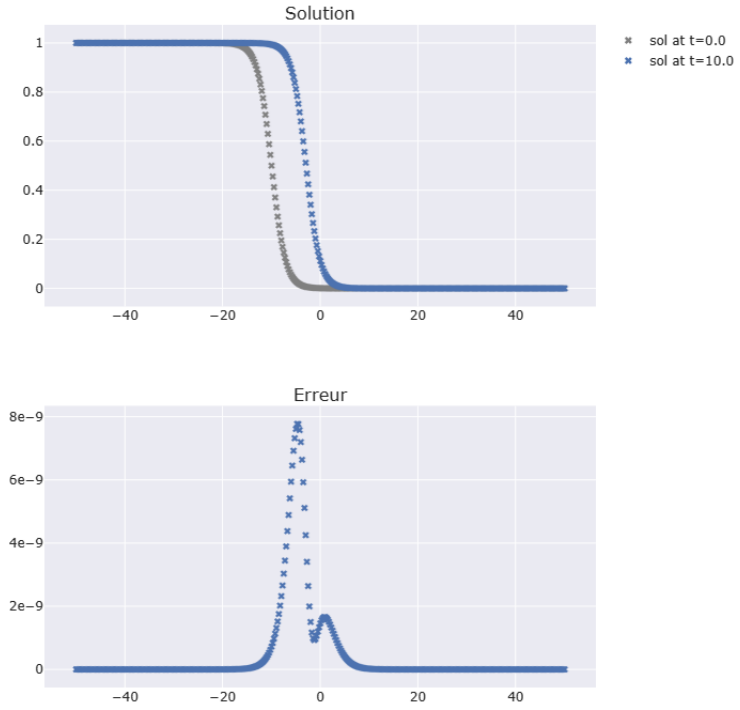

FIGURE 1 – Haut : Méthode de Runge-Kutta d'ordre 2. Bas : erreur

Il est intéressant de constater que, tout comme remarqué au TP2, l'erreur se trouve concentré là où la dérivée seconde est maximale, alors qu'elle est nulle sur le reste du graphe lorsque cette dernière est précisement nulle. On remarque de même qu'elle vaut 0 en $x = 0$, au point d'inflexion, où la dérivée seconde est nulle.

Pour cette méthode, le temps d'exécution s'éleve à 1.17s pour une erreur d'environ 1.4 10^{-9}.

La méthode IMEX consiste à utiliser plusieurs méthodes de Runge-Kutta successivement. En l'occurence, pour les méthodes d'ordre deux avec deux étages, on utilise le schéma suivant :

$$\begin{aligned}(1-\lambda\Delta t A)U_1 &= U^n\\ (1-\lambda\Delta t A)U_2 &= U^n + \Delta t(1-2\lambda)AU_1 + \Delta t R(U_1)\\ U^{n+1} &= U^n + \frac{\Delta t}{2}A(U_1+U_2) + \frac{\Delta t}{2}(R(U_1)+R(U_2))\end{aligned}$$

avec $\lambda = 1 - \sqrt{2}/2$.

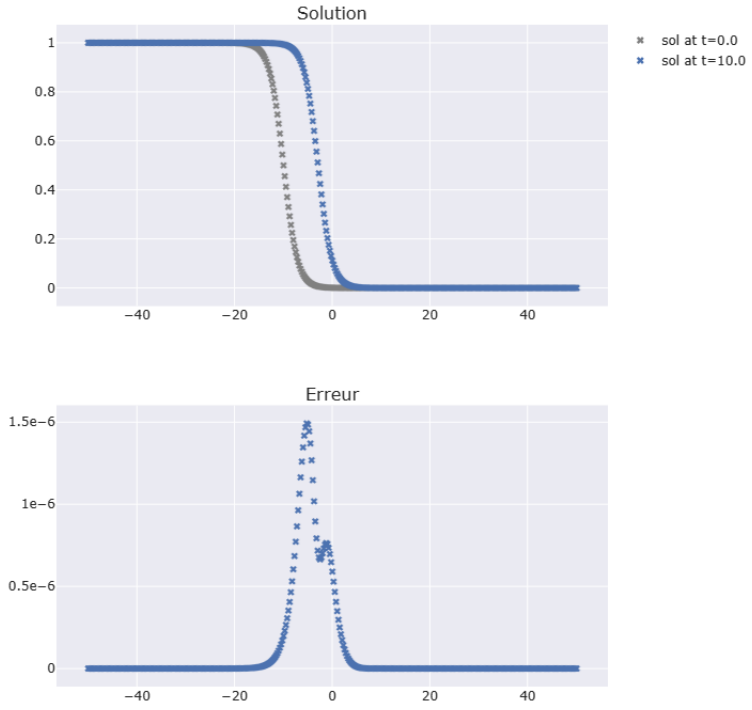

FIGURE 2 – Solution approché avec méthode IMEX : Runge kutta ordre 2 à deux étages(haut) et erreur avec la solution quasi exacte (bas)

On obtient un temps de résolution de 0.7s, avec une erreur de 3 10^{-7}. Même si on a réduit le temps de calcul de moitié environ, l'erreur devient cependant 100 fois plus importante.

Pour un 3 étages, on a :

$$\begin{aligned}U_1 &= U^n(1-\lambda\Delta t A)\\ U_2 &= U^n + \lambda\Delta t R(U_1)\\ (1-\lambda\Delta t A)U_3 &= U^n + \Delta t(1-2\lambda)AU_2 + (\lambda-1)\Delta t R(U_1) + 2(1-\lambda)\Delta t R(U_2)\\ U^{n+1} &= U^n + \frac{\Delta t}{2}A(U_2+U_3) + \frac{\Delta t}{2}(R(U_3)+R(U_2))\end{aligned}$$

On a cette fois que le temps de résolution est de 0.73s, mais pour cette fois une erreur de 3.4 10^{-8}. Par conséquent, pour le même temps de calcul, on a une erreur 10 fois moins importante. Le pas de temps utilisé est le même, on a dt = 0.01.

Enfin, avec la méthode de Strang, on obtient la figure 4. La méthode de Strang consiste à "splitter" l'opérateur d'évolution en deux opérateurs d'évolutions. Par conséquent,

$$U_{n+1} = X_{t_n+\Delta t/2}Y_{t_n+\Delta t}X_{t_n+\Delta t/2}U_n$$

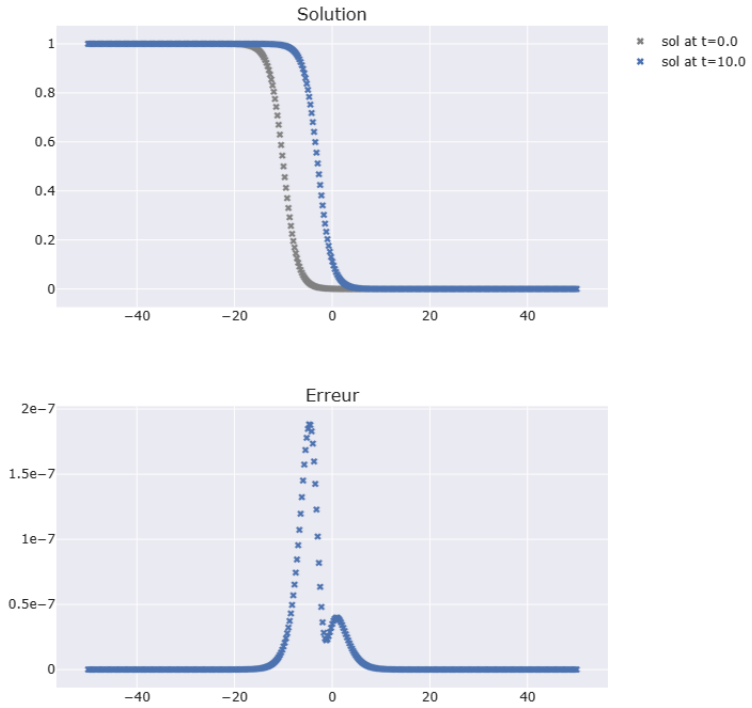

FIGURE 3 – Solution approché avec méthode IMEX : Runge kutta ordre 2 à trois étages(haut) et erreur avec la solution quasi exacte (bas)

On a une norme d'erreur de 5 10^{-8}, pour un temps d'éxecution de 1.5s et avec un pas de temps dt = 0.01s.

Si on passe à un nombre de points de 10001, au lieu de 1001 comme précedemment, on observe que naturellement, le pas de temps est 10 fois plus petit. Sur les tableaux de la figure 5 sont résumés les erreurs et temps de calculs pour chaqu'une des méthodes.

On observe que pour une erreur quasi-identique pour pas égale, la méthode de Runge-Kutta d'ordre 2 est celle qui met le moins de temps à s'éxecuter, et de loin.

Dans ce cas précis, il n'est donc absolument pas nécessaire de recourir à des techniques d'intégration plus sophistiqué. Les méthodes sont toutes stables, dans le sens où il y a a bien convergence de la solution numérique vers la solution exacte, d'après le théorème de Lax. Par ailleurs, on peut augmenter la raideur par le biais du coefficient de diffusivité D. En effet, on sait par analyse dimensionnelle que la vitesse de l'onde progressive a une vitesse d'environ $\sqrt{kD}/\sqrt{2}$ et un gradient $\approx \sqrt{k/D}$. Pour $k = D = 10$, le calcul avec Heun fait émerger des erreurs. Pour toutes les autres méthodes, cela marche aussi bien. Maintenant, pour $k = 50$, et $D = 1$, les solutions sont encore convergentes, même si l'erreur devient bien plus importante, pouvant atteindre 0.1 avec 1001 points. Les méthodes ont donc bien l'air stable. La figure 7 montre le résultat.

Pour un niveau de précision fixée, la méthode la plus avantageuse est certainement la méthode de Heun, car la plus économe en temps. En revanche, pour des problèmes particulièrement raide, on privilégira la méthode de Strang qui garde encore une erreur d'environ 10^{-4}, plus basse que les autres méthodes concurrentes.

3 Résolution de système linéaire

La résolution de l'équation de Poisson

$$-\Delta u = f$$

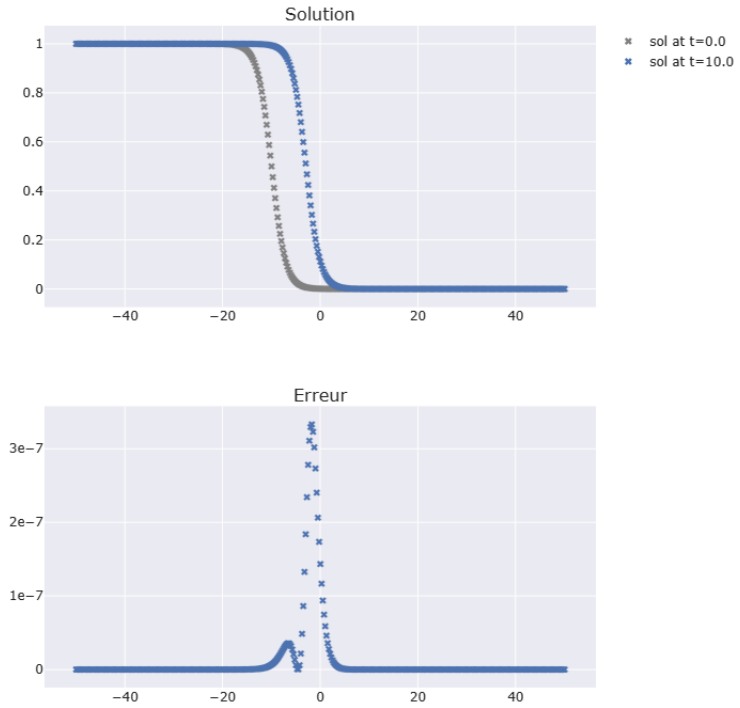

FIGURE 4 – Haut : Solution approché avec méthode strang. Bas : erreur comparé à la solution quasi exacte.

Colonne1	Méthode numérique	Méthode de Runge et Kutta d'ordre 2 (Heun)	IMEX - RK d'ordre 2 à 2 étages	IMEX - RK d'ordre 2 à 3 étages	Méthode de splitting (Strang)
nombre de point	1001	1.4108303455e-07	2.9823618619e-07	3.4118278407e-08	4.9488970790e-08
	10001	1.4067949619e-09	2.9808053138e-09	3.4044594893e-10	4.9584407542e-10
	100001	1.4723639783e-11	3.0573726165e-11	2.9480085674e-12	6.1551874339e-12

FIGURE 5 – Tableau des erreurs en fonction du nombre de points

Colonne1	Méthode numérique	Méthode de Runge et Kutta d'ordre 2 (Heun)	IMEX - RK d'ordre 2 à 2 étages	IMEX - RK d'ordre 2 à 3 étages	Méthode de splitting (Strang)
nombre de point	1001	0.0674278736114502 s	0.495386838912963 87 s	0.6231625080108643 s	1.6658499240875244 s
	10001	0.0674278736114502 s	4.970597505569458 s	5.964402198791504 s	9.026012897491455 s
	100001	4.486247539520264 s	46.03328204154968 s	60.0381920337677 s	90.1246771812439 s

FIGURE 6 – Tableau du temps d'éxecution en fonction du nombre de points

est équivalent après discrétisation au système suivant :

$$Au = b$$

Pour trouver u, il est nullement nécessaire d'inverser A, il suffit de résoudre le système linéaire avec un pivot de Gauss. Cependant, dû à la nature flottante des opérations informatiques ainsi qu'à un potentiel mauvais conditionnement de la matrice A, cette opération peut s'avérer peu précise. On s'intéressera au laplacien en 1D, 2D et 3D, la forme de la matrice A dépend de quel cas on traite.

Outre le conditionnement de la matrice qui est en $\prime(N_x^2)$, la recherche du bon pivot s'avère cruciale pour la résolution pour la précision calcul de la machine. Pour avoir une bonne précision machine, il faut trouver le pivot le plus grand possible en valeur absolu. Par conséquent, il faut alors permuter les différentes lignes de A pour se trouver dans la bonne configuration, c'est la résolution par méthode direct.

Cela saute aux yeux que la méthode direct devient bien moins efficace en 3 dimensions qu'en 1 dimension. C'est tout-à-fait l'inverse pour la méthode du gradient conjugué. Cependant, par construction, on voit bien que la méthode direct reste la plus précise pour la résolution, surtout en dimension 3. Par ailleurs, le conditionnement de A est le plus élevé en dimension 1 et le moins élevé en dimension 3. Par conséquent, on voit que la méthode du gradient conjugué est sensible au conditionnement de la matrice, contrairement à la méthode direct qui en est

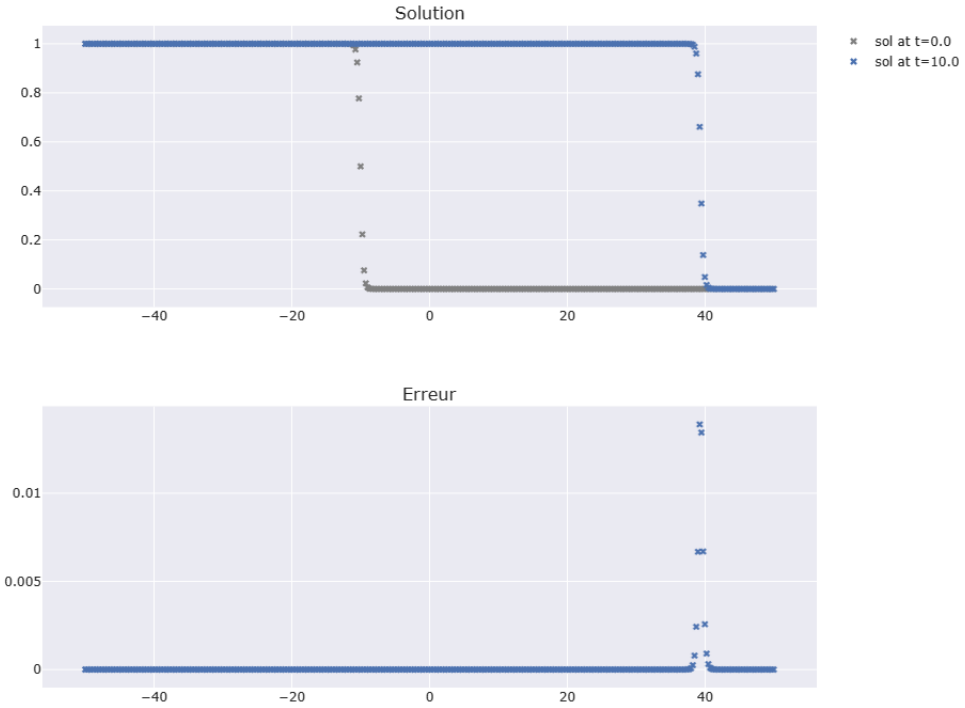

FIGURE 7 – Haut : Solution approché avec IMEX 3 étages, $k = 50$ et $D = 1$. Bas : erreur comparé à la solution quasi exacte.

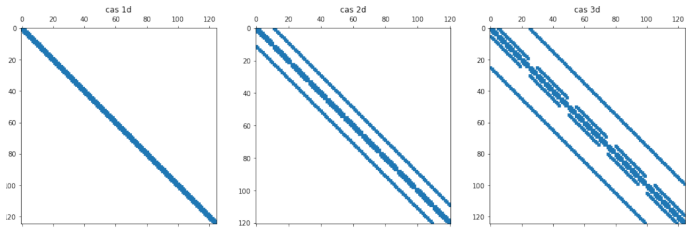

FIGURE 8 – Structure de A selon que l'on soit en 1D, 2D ou 3D

Méthode gradient conjugué	1D	2D	3D
Temps d'éxecution	150.2585825920105 s	1.7257790565490723 s	0.4354124069213867 s
erreur relative	1.6284427528922828e-07	9.970802356723043e-06	9.683279786771778e-06

FIGURE 9 – Tableau des erreurs et temps d'éxecution par gradient conjugué

Méthode direct	1D	2D	3D
Temps d'éxecution	0.1640331745147705 s	2.7101047039031982 s	403.6903831958771 s
erreur relative	2.038112937392189e-07	3.3900660711809027e-12	2.539670978104994e-13

FIGURE 10 – Tableau du temps d'éxecution et erreurs pour méthode direct

insensible, ce qui fait sens puisque la méthode direct est faite pour lutter contre les arrondis dû à la représentation machine des nombres et non au conditionnement de la matrice, qui est une propriété strictement mathématique. La structure en bande diagonale de A en 3D fait que l'algorithme de tri nécessaire pour trouver le meilleurs pivots prend bien plus de temps. En ce sens, il devient moins optimal. D'un autre côté, lorsque la méthode A est symétrique définie positive, une autre méthode direct est utilisé qui ne requiert pas de recherche de pivot est la décomposition

de Cholesky. Cependant, en 3 dimensions, c'est cette décomposition qui prend du temps.

En conclusion, la méthode direct, indépendante du conditionnement de la matrice, sera utilisé pour la dimension 1. En revanche, en 3D, c'est une autre histoire, l'algorithme de tri prend énormément de temps, mais n'affecte pas la précision de la méthode. C'est l'exacte inverse pour la méthode du gradient conjugué. La convergence de l'algorithme dépend fortement du conditionnement de la matrice. On retiendra en général que la méthode direct est plus adaptée lorsque la matrice A possède un mauvais conditionnement, tandis que les méthodes itératives, tels que le gradient conjuguée, seront préférées lorsque l'on se trouve en grande dimension. Une remarque intéressante, que je n'aurais pas trouvé intuitive à premier abord, est que le conditionnement de la matrice de discrétisation du Laplacien baisse avec la dimension du problème. C'est un argument supplémentaire quant à l'utilisation de méthode itérative dans le cas de grande dimension. La question qui reste en suspens est : pourquoi les méthodes itératives marchent mal, tandis que les méthodes directes marchent bien lorsque $\kappa(A) \gg 1$. Une première réponse pourrait se trouver dans le fait que le taux de convergence est souvent relié au conditionnement. Dans le cas du gradient conjuguée, on a par exemple :

$$||x_k - x^\star|| \leq 2 \left(\frac{\kappa(A) - 1}{\kappa(A) + 1} \right)^k ||x_0 - x^\star||$$

Dans le cas de méthodes directs, cela peut sans doute diminuer la précision de la solution calculée, mais n'affectera pas le temps de calcul lui même.