

TRAVAUX PRATIQUES - E.D.O.

Groupe 2
Durée : 10h00

Travail individuel et personnel

Table des matières

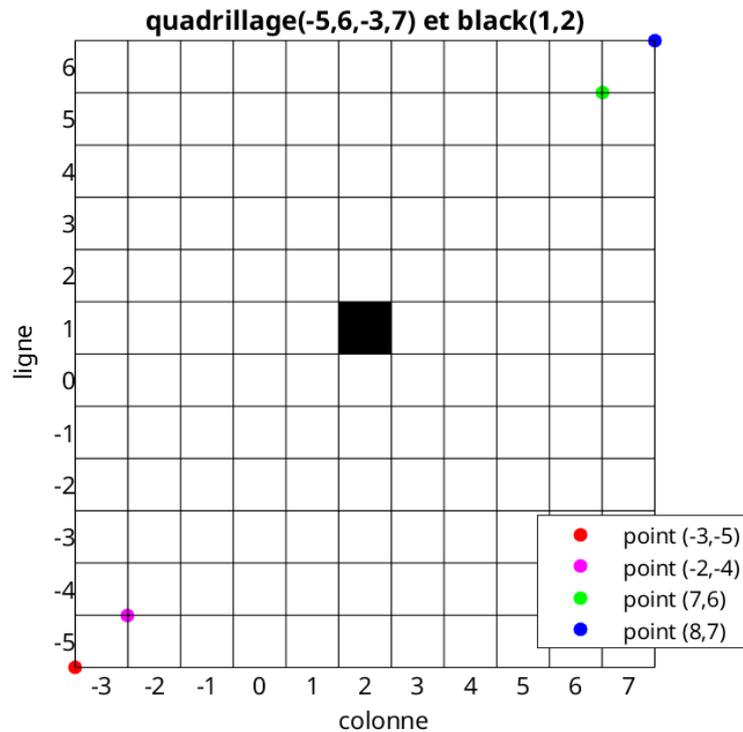
1 Tests Algorithmique et Matlab	1
2 Résolution numérique d'équations différentielles ordinaires	4
2.1 Schémas numériques pour la résolution d'un problème de Cauchy	4
2.1.1 Schéma d'Euler progressif (ordre 1)	4
2.1.2 Schéma de la tangente améliorée (ordre 2)	4
2.1.3 Schéma de Heun (ordre 2)	4
2.1.4 Schémas de Runge-Kutta	4
2.1.5 Méthodes d'Adams-Bashforth	5
2.1.6 Méthodes d'Adams-Moulton	6
2.1.7 Schéma de Nyström (ordre 3)	6
2.1.8 Schémas BDF (Backward-Difference Formulas)	6
2.1.9 Schéma de Milne (ordre 4)	6
2.1.10 Schémas de Milne-Simpson	6
2.2 Schéma prédicteur-correcteur	6
2.3 Travail à effectuer	7
3 Le système solaire	9
3.1 Position du problème et équations différentielles	9
3.2 Résolution numérique classique	9
3.3 Système à 3 corps : Soleil, Saturne et Jupiter	10
3.4 Système à 6 corps	11
3.5 Système à 7 corps	12
4 Annexes	12
4.1 Quelques E.D.O. du premier ordre	12
4.1.1 Exemple 1	12
4.1.2 Exemple 2	13
4.1.3 Exemple 3	13

1 Tests Algorithmique et Matlab

Une archive compressée au format **zip**, nommée `CodesFournis_Mosaiques.zip`, ou au format **tar.gz**, nommée `CodesFournis_Mosaiques.tar.gz`, est disponible sur le serveur Discord. Il faut télécharger l'archive et la décompresser dans un répertoire.

Cette archive contient , entre autres, la fonction `black` et le programme `Quadrillagefigure`. Dans le programme `Quadrillagefigure` l'appel à la fonction `Quadrillage` manquante a été mis en commentaire. Ce programme servira uniquement à déboguer/tester/valider la fonction `Quadrillage` que vous allez écrire.

Q. 1 *Ecrire la fonction Matlab `Quadrillage(imin,imax,jmin,jmax)` permettant de générer un quadrillage pour les lignes `imin` à `imax` et les colonnes `jmin` à `jmax`. Voici un exemple avec la commande `Quadrillage(-5,6,-3,7)` représentant uniquement les traits noirs sur la figure :*

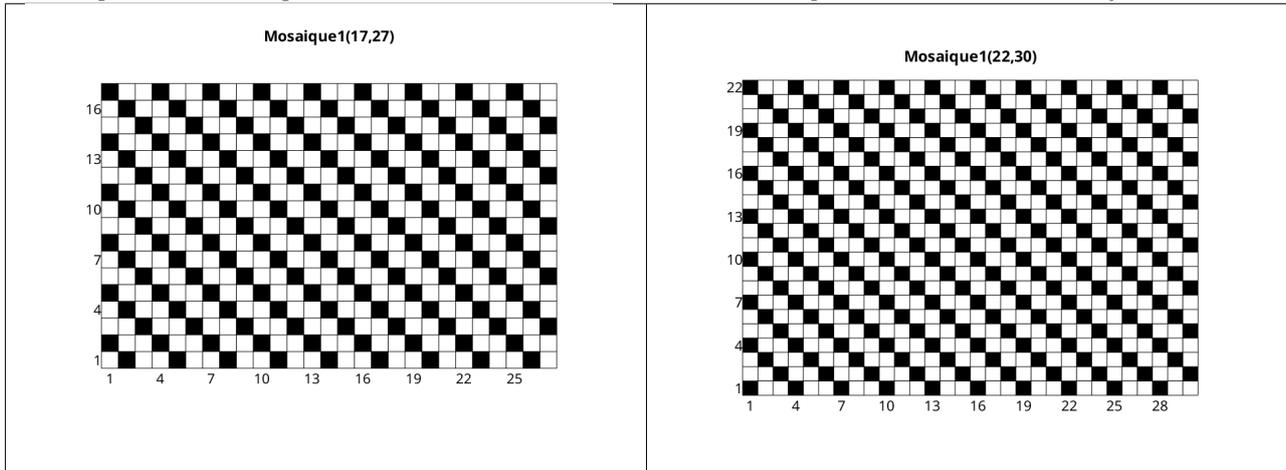


On peut noter que les coordonnées des points sont exprimées dans le plan classique xOy . On peut tester cette fonction avec le programme `Quadrillagefigure` fourni pour obtenir la figure précédente.

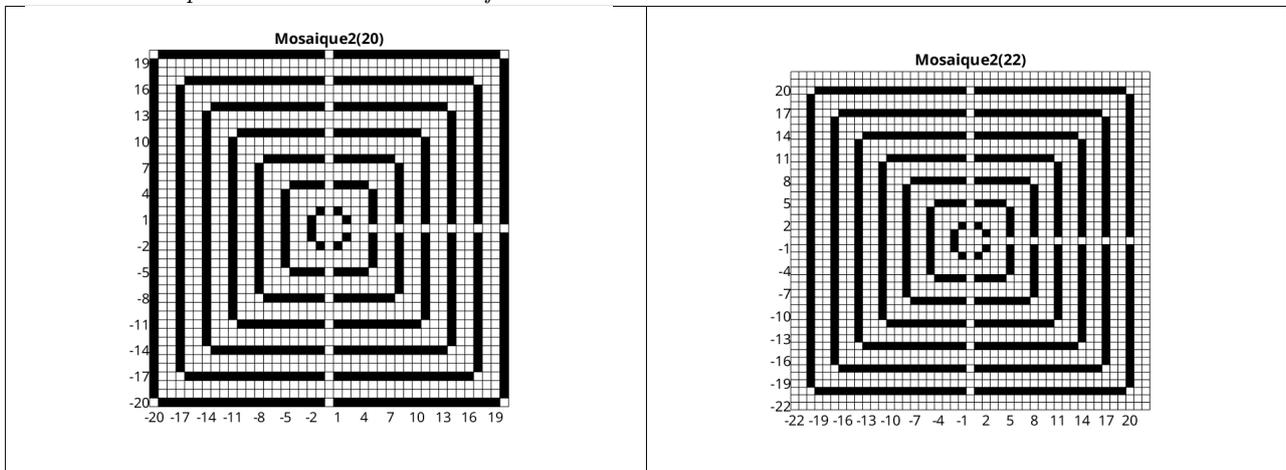
Le carré noir en ligne 1 et colonne 2 a été représenté à l'aide de la commande `black(1,2)`, la fonction `black` étant fournie.

On rappelle que pour représenter un segment entre les points $A_1 = (x_1, y_1)$ et $A_2 = (x_2, y_2)$, on peut utiliser sous Matlab, la commande `plot([x1 x2], [y1 y2])`. □

Q. 2 Ecrire la fonction `Mosaïque1(n,m)` permettant de créer une mosaïque sur le quadrillage `Quadrillage(1,n,1,m)` sachant que la case en ligne `n` et colonne `1` est noire. Voici deux exemples d'utilisation de cette fonction :



Q. 3 Ecrire la fonction `Mosaïque2(n)` permettant de créer une mosaïque sur le quadrillage `Quadrillage(-n,n,-n,n)`. Voici deux exemples d'utilisation de cette fonction :



A faire en 2h30 (temps indicatif)

- ◇ Créer une archive compressée nommée `<NOM>-TP1-Q1a3` contenant les fichiers `Quadrillage.m`, `black.m`, `Mosaïque1.m`, `Mosaïque2.m` et tout autre fichier permettant l'exécution des fonctions `Mosaïque1.m` et `Mosaïque2.m`. Ici `<NOM>` correspond évidemment à votre nom.
- ◇ Envoyer un mail à `cuvelier@math.univ-paris13.fr` ayant pour **sujet** "`<NOM> TP1 Q1a3`" et en fichier joint l'archive compressée créée précédemment.

2 Résolution numérique d'équations différentielles ordinaires

Pour une explication détaillée voir le polycopié fourni MethNumII_21janvier2025.pdf

2.1 Schémas numériques pour la résolution d'un problème de Cauchy

♥ Définition 2.1: problème de Cauchy

Soit \mathbf{f} l'application continue définie par

$$\begin{aligned} \mathbf{f} : [t^0, t^0 + T] \times \mathbb{R}^d &\longrightarrow \mathbb{R}^d \\ (t, \mathbf{y}) &\longmapsto \mathbf{f}(t, \mathbf{y}) \end{aligned}$$

avec $t^0 \in \mathbb{R}$ et $T \in]0, +\infty]$. Le **problème de Cauchy** revient à chercher une fonction \mathbf{y} définie par

$$\begin{aligned} \mathbf{y} : [t^0, t^0 + T] &\longrightarrow \mathbb{R}^d \\ t &\longmapsto \mathbf{y}(t) \end{aligned}$$

continue et dérivable, telle que

$$\mathbf{y}'(t) = \mathbf{f}(t, \mathbf{y}(t)), \quad \forall t \in [t^0, t^0 + T] \quad (2.1)$$

$$\mathbf{y}(t^0) = \mathbf{y}^{[0]} \in \mathbb{R}^d. \quad (2.2)$$

Dans tous les schémas qui suivent, on note t^n , $n \in \llbracket 0, N \rrbracket$, une discrétisation régulière de $[t^0, t^0 + T]$, $\mathbf{y}^{[n]} \approx \mathbf{y}(t^n)$ et $\mathbf{f}^{[n]} = \mathbf{f}(t^n, \mathbf{y}^{[n]})$.

2.1.1 Schéma d'Euler progressif (ordre 1)

$$\begin{cases} \mathbf{y}^{[n+1]} &= \mathbf{y}^{[n]} + h\mathbf{f}^{[n]}, \quad \forall n \in \llbracket 0, N-1 \rrbracket \\ \mathbf{y}^{[0]} &= \mathbf{y}(t^0) \end{cases} \quad (2.3)$$

2.1.2 Schéma de la tangente améliorée (ordre 2)

$$\begin{cases} \mathbf{y}^{[n+1]} &= \mathbf{y}^{[n]} + h\mathbf{f}(t^n + \frac{h}{2}, \mathbf{y}^{[n]} + \frac{h}{2}\mathbf{f}^{[n]}), \quad \forall n \in \llbracket 0, N-1 \rrbracket \\ \mathbf{y}^{[0]} &= \mathbf{y}(t^0) \end{cases} \quad (2.4)$$

2.1.3 Schéma de Heun (ordre 2)

$$\mathbf{y}^{[n+1]} = \mathbf{y}^{[n]} + \frac{h}{2} \left[\mathbf{f}^{[n]} + \mathbf{f} \left(t^n + h, \mathbf{y}^{[n]} + h\mathbf{f}^{[n]} \right) \right]. \quad (2.5)$$

2.1.4 Schémas de Runge-Kutta

- ordre 2 (version 1)

$$\begin{aligned} \mathbf{k}_1^{[n]} &= \mathbf{f}(t^n, \mathbf{y}^{[n]}) \\ \mathbf{k}_2^{[n]} &= \mathbf{f}(t^n + \frac{h}{2}, \mathbf{y}^{[n]} + \frac{h}{2}\mathbf{k}_1^{[n]}) \\ \mathbf{y}^{[n+1]} &= \mathbf{y}^{[n]} + h\mathbf{k}_2^{[n]}. \end{aligned} \quad (2.6)$$

- ordre 2 (version 2)

$$\begin{aligned} \mathbf{k}_1^{[n]} &= \mathbf{f}(t^n, \mathbf{y}^{[n]}) \\ \mathbf{k}_2^{[n]} &= \mathbf{f}(t^n + h, \mathbf{y}^{[n]} + h\mathbf{k}_1^{[n]}) \\ \mathbf{y}^{[n+1]} &= \mathbf{y}^{[n]} + \frac{h}{2}(\mathbf{k}_1^{[n]} + \mathbf{k}_2^{[n]}). \end{aligned} \quad (2.7)$$

- ordre 2 (version 3)

$$\begin{aligned} \mathbf{k}_1^{[n]} &= \mathbf{f}(t^n, \mathbf{y}^{[n]}) \\ \mathbf{k}_2^{[n]} &= \mathbf{f}(t^n + \frac{2h}{3}, \mathbf{y}^{[n]} + \frac{2h}{3}\mathbf{k}_1^{[n]}) \\ \mathbf{y}^{[n+1]} &= \mathbf{y}^{[n]} + \frac{h}{4}(\mathbf{k}_1^{[n]} + 3\mathbf{k}_2^{[n]}). \end{aligned} \quad (2.8)$$

- ordre 3 (version 1)

$$\begin{aligned} \mathbf{k}_1^{[n]} &= \mathbf{f}(t^n, \mathbf{y}^{[n]}) \\ \mathbf{k}_2^{[n]} &= \mathbf{f}(t^n + \frac{h}{3}, \mathbf{y}^{[n]} + \frac{h}{3}\mathbf{k}_1^{[n]}) \\ \mathbf{k}_3^{[n]} &= \mathbf{f}(t^n + \frac{2h}{3}, \mathbf{y}^{[n]} + \frac{2h}{3}\mathbf{k}_2^{[n]}) \\ \mathbf{y}^{[n+1]} &= \mathbf{y}^{[n]} + \frac{h}{4}(\mathbf{k}_1^{[n]} + 3\mathbf{k}_3^{[n]}). \end{aligned} \quad (2.9)$$

- ordre 3 (version 2)

$$\begin{aligned} \mathbf{k}_1^{[n]} &= \mathbf{f}(t^n, \mathbf{y}^{[n]}) \\ \mathbf{k}_2^{[n]} &= \mathbf{f}(t^n + \frac{2h}{3}, \mathbf{y}^{[n]} + \frac{2h}{3}\mathbf{k}_1^{[n]}) \\ \mathbf{k}_3^{[n]} &= \mathbf{f}(t^n + \frac{2h}{3}, \mathbf{y}^{[n]} + \frac{2h}{3}\mathbf{k}_2^{[n]}) \\ \mathbf{y}^{[n+1]} &= \mathbf{y}^{[n]} + \frac{h}{8}(2\mathbf{k}_1^{[n]} + 3\mathbf{k}_2^{[n]} + 3\mathbf{k}_3^{[n]}). \end{aligned} \quad (2.10)$$

- ordre 3 (version 3)

$$\begin{aligned} \mathbf{k}_1^{[n]} &= \mathbf{f}(t^n, \mathbf{y}^{[n]}) \\ \mathbf{k}_2^{[n]} &= \mathbf{f}(t^n + \frac{2h}{3}, \mathbf{y}^{[n]} + \frac{2h}{3}\mathbf{k}_1^{[n]}) \\ \mathbf{k}_3^{[n]} &= \mathbf{f}(t^n, \mathbf{y}^{[n]} - h\mathbf{k}_1^{[n]} + h\mathbf{k}_2^{[n]}) \\ \mathbf{y}^{[n+1]} &= \mathbf{y}^{[n]} + \frac{h}{4}(3\mathbf{k}_2^{[n]} + \mathbf{k}_3^{[n]}). \end{aligned} \quad (2.11)$$

- ordre 4 (version 1)

$$\begin{aligned} \mathbf{k}_1^{[n]} &= \mathbf{f}(t^n, \mathbf{y}^{[n]}) \\ \mathbf{k}_2^{[n]} &= \mathbf{f}(t^n + \frac{h}{2}, \mathbf{y}^{[n]} + \frac{h}{2}\mathbf{k}_1^{[n]}) \\ \mathbf{k}_3^{[n]} &= \mathbf{f}(t^n + \frac{h}{2}, \mathbf{y}^{[n]} + \frac{h}{2}\mathbf{k}_2^{[n]}) \\ \mathbf{k}_4^{[n]} &= \mathbf{f}(t^n + h, \mathbf{y}^{[n]} + h\mathbf{k}_3^{[n]}) \\ \mathbf{y}^{[n+1]} &= \mathbf{y}^{[n]} + \frac{h}{6}(\mathbf{k}_1^{[n]} + 2\mathbf{k}_2^{[n]} + 2\mathbf{k}_3^{[n]} + \mathbf{k}_4^{[n]}). \end{aligned} \quad (2.12)$$

- ordre 4 (version 2)

$$\begin{aligned} \mathbf{k}_1^{[n]} &= \mathbf{f}(t^n, \mathbf{y}^{[n]}) \\ \mathbf{k}_2^{[n]} &= \mathbf{f}(t^n + \frac{h}{3}, \mathbf{y}^{[n]} + \frac{h}{3}\mathbf{k}_1^{[n]}) \\ \mathbf{k}_3^{[n]} &= \mathbf{f}(t^n + \frac{2h}{3}, \mathbf{y}^{[n]} - \frac{h}{3}\mathbf{k}_1^{[n]} + h\mathbf{k}_2^{[n]}) \\ \mathbf{k}_4^{[n]} &= \mathbf{f}(t^n + h, \mathbf{y}^{[n]} + h\mathbf{k}_1^{[n]} - h\mathbf{k}_2^{[n]} + h\mathbf{k}_3^{[n]}) \\ \mathbf{y}^{[n+1]} &= \mathbf{y}^{[n]} + \frac{h}{8}(\mathbf{k}_1^{[n]} + 3\mathbf{k}_2^{[n]} + 3\mathbf{k}_3^{[n]} + \mathbf{k}_4^{[n]}). \end{aligned} \quad (2.13)$$

- ordre 4 (version 3)

$$\begin{aligned} \mathbf{k}_1^{[n]} &= \mathbf{f}(t^n, \mathbf{y}^{[n]}) \\ \mathbf{k}_2^{[n]} &= \mathbf{f}(t^n + \frac{h}{4}, \mathbf{y}^{[n]} + \frac{h}{4}\mathbf{k}_1^{[n]}) \\ \mathbf{k}_3^{[n]} &= \mathbf{f}(t^n + \frac{h}{2}, \mathbf{y}^{[n]} + \frac{h}{2}\mathbf{k}_2^{[n]}) \\ \mathbf{k}_4^{[n]} &= \mathbf{f}(t^n + h, \mathbf{y}^{[n]} + h\mathbf{k}_1^{[n]} - 2h\mathbf{k}_2^{[n]} + 2h\mathbf{k}_3^{[n]}) \\ \mathbf{y}^{[n+1]} &= \mathbf{y}^{[n]} + \frac{h}{6}(\mathbf{k}_1^{[n]} + 4\mathbf{k}_3^{[n]} + \mathbf{k}_4^{[n]}). \end{aligned} \quad (2.14)$$

2.1.5 Méthodes d'Adams-Bashforth

On note $\mathbf{f}^{[n]} = \mathbf{f}(t^n, \mathbf{y}^{[n]})$

$$\mathbf{y}^{[n+1]} = \mathbf{y}^{[n]} + \frac{h}{2} (3\mathbf{f}^{[n]} - \mathbf{f}^{[n-1]}). \quad (2.15)$$

$$\mathbf{y}^{[n+1]} = \mathbf{y}^{[n]} + \frac{h}{12} (23\mathbf{f}^{[n]} - 16\mathbf{f}^{[n-1]} + 5\mathbf{f}^{[n-2]}). \quad (2.16)$$

$$\mathbf{y}^{[n+1]} = \mathbf{y}^{[n]} + \frac{h}{24} (55\mathbf{f}^{[n]} - 59\mathbf{f}^{[n-1]} + 37\mathbf{f}^{[n-2]} - 9\mathbf{f}^{[n-3]}). \quad (2.17)$$

Ces 3 schémas sont **explicites** et leur ordre correspond au nombre de pas.

2.1.6 Méthodes d'Adams-Moulton

$$\mathbf{y}^{[n+1]} = \mathbf{y}^{[n]} + \frac{h}{2} \left(\mathbf{f}^{[n+1]} + \mathbf{f}^{[n]} \right). \quad (2.18)$$

$$\mathbf{y}^{[n+1]} = \mathbf{y}^{[n]} + \frac{h}{12} \left(5\mathbf{f}^{[n+1]} + 8\mathbf{f}^{[n]} - \mathbf{f}^{[n-1]} \right). \quad (2.19)$$

$$\mathbf{y}^{[n+1]} = \mathbf{y}^{[n]} + \frac{h}{24} \left(9\mathbf{f}^{[n+1]} + 19\mathbf{f}^{[n]} - 5\mathbf{f}^{[n-1]} + \mathbf{f}^{[n-2]} \right). \quad (2.20)$$

Ces 3 schémas sont **implicites** et leur ordre correspond au nombre de pas plus un.

2.1.7 Schéma de Nyström (ordre 3)

$$\mathbf{y}^{[n+1]} = \mathbf{y}^{[n-1]} + \frac{h}{3} \left(7\mathbf{f}(t^n, \mathbf{y}^{[n]}) - 2\mathbf{f}(t^{n-1}, \mathbf{y}^{[n-1]}) + \mathbf{f}(t^{n-2}, \mathbf{y}^{[n-2]}) \right) \quad (2.21)$$

2.1.8 Schémas BDF (Backward-Difference Formulas)

- ordre 1

$$\mathbf{y}^{[n+1]} = \mathbf{y}^{[n]} + h\mathbf{f}^{[n+1]} \quad (2.22)$$

- ordre 2

$$\mathbf{y}^{[n+1]} = \frac{4}{3}\mathbf{y}^{[n]} - \frac{1}{3}\mathbf{y}^{[n-1]} + \frac{2h}{3}\mathbf{f}^{[n+1]} \quad (2.23)$$

- ordre 3

$$\mathbf{y}^{[n+1]} = \frac{1}{11} \left(18\mathbf{y}^{[n]} - 9\mathbf{y}^{[n-1]} + 2\mathbf{y}^{[n-2]} + 6h\mathbf{f}^{[n+1]} \right) \quad (2.24)$$

- ordre 4

$$\mathbf{y}^{[n+1]} = \frac{1}{25} \left(48\mathbf{y}^{[n]} - 36\mathbf{y}^{[n-1]} + 16\mathbf{y}^{[n-2]} - 3\mathbf{y}^{[n-3]} + 12h\mathbf{f}^{[n+1]} \right) \quad (2.25)$$

2.1.9 Schéma de Milne (ordre 4)

$$\mathbf{y}^{[n+1]} = \mathbf{y}^{[n-3]} + \frac{4h}{3} \left(2\mathbf{f}^{[n]} - \mathbf{f}^{[n-1]} + 2\mathbf{f}^{[n-2]} \right) \quad (2.26)$$

2.1.10 Schémas de Milne-Simpson

- ordre 1

$$\mathbf{y}^{[n+1]} = \mathbf{y}^{[n-1]} + 2h\mathbf{f}^{[n+1]} \quad (2.27)$$

- ordre 2

$$\mathbf{y}^{[n+1]} = \mathbf{y}^{[n-1]} + 2h\mathbf{f}^{[n]} \quad (2.28)$$

- ordre 4

$$\mathbf{y}^{[n+1]} = \mathbf{y}^{[n-1]} + \frac{h}{3} \left(\mathbf{f}^{[n+1]} + 4\mathbf{f}^{[n]} + \mathbf{f}^{[n-1]} \right) \quad (2.29)$$

2.2 Schéma prédicteur-correcteur

Il s'agit là d'une des méthodes les plus employées. Une méthode de prédiction-correction procède en deux temps : à l'aide du schéma explicite, on calcule une valeur approchée de la solution au $n^{i\text{ème}}$ pas (notée $\bar{\mathbf{y}}^{(n+1)}$), puis on calcule $\mathbf{y}^{[n+1]}$ à l'aide du schéma implicite en substituant, dans l'expression de droite du schéma implicite, $\mathbf{y}^{[n+1]}$ par $\bar{\mathbf{y}}^{(n+1)}$. On obtient alors une valeur dite *corrigée*.

2.3 Travail à effectuer

Le but est de représenter graphiquement les erreurs données par plusieurs schémas et de retrouver numériquement leur ordre. Pour cela il faudra pouvoir connaître explicitement la solution du problème de Cauchy étudié. Voir l'annexe 4.1 pour plusieurs exemples de problèmes de Cauchy avec solutions.

- Q. 4** a. Ecrire les cinq fonctions Matlab suivantes correspondant à la résolution d'un problème de Cauchy :
- `[t,Y]=redEUP(f,a,b,yo,N)` : schéma d'Euler progressif (fichier `redEUP.m`).
 - `[t,Y]=redRK3(f,a,b,yo,N)` : schéma de Runge et Kutta d'ordre 3 (version 1) (fichier `redRK3.m`).
 - `[t,Y]=redRK4(f,a,b,yo,N)` : schéma de Runge et Kutta d'ordre 4 (version 2) (2.13) (fichier `redRK4.m`).
 - `[t,Y]=redAB3(f,a,b,yo,N)` : schéma d'Adams-Bashforth d'ordre 3 (fichier `redAB3.m`).
 - `[t,Y]=redPC3(f,a,b,yo,N)` : schéma de type prédiction-correction utilisant les schémas de Nyström d'ordre 3 (2.21) et d'Adams-Moulton d'ordre 3 (fichier `redPC3.m`).

Ici les paramètres f , a , b , y_0 correspondent respectivement aux $f, t^0, t^0 + T, \mathbf{y}^{[0]}$ du problème de Cauchy (2.1-2.2). Enfin, \mathbf{Y} est le tableau contenant les $\mathbf{y}^{[n]}$, $n \in \{0, \dots, N\}$ et \mathbf{t} est le tableau contenant les $n+1$ nombres t^n , $n \in \{0, \dots, N\}$

- b. Ecrire le programme principal (fichier `erreur.m`) permettant le calcul et le tracé des erreurs. Pour une méthode donnée le tracé de l'erreur correspond au tracé de l'ensemble des points $(t^n, \text{abs}(\mathbf{y}^{[n]} - \mathbf{y}(t^n)))$, $n \in \{0, \dots, N\}$.

Voir la figure 1 pour un exemple de tracé. Pour cette figure, la commande Matlab `subplot` a été utilisée.

- c. Ecrire le programme principal (fichier `ordre.m`) permettant de calculer numériquement l'ordre des 5 schémas et de les représenter.

Voir la figure 2 pour un exemple de tracé.

□

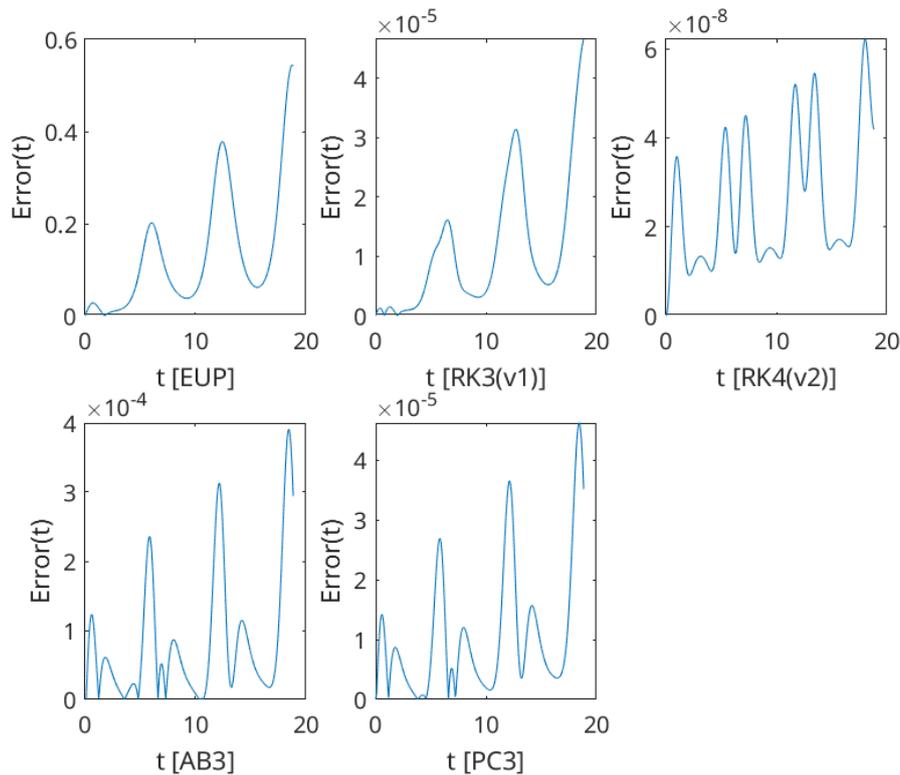


FIGURE 1 – Valeurs absolues des erreurs des 5 schémas représentées dans une unique figure

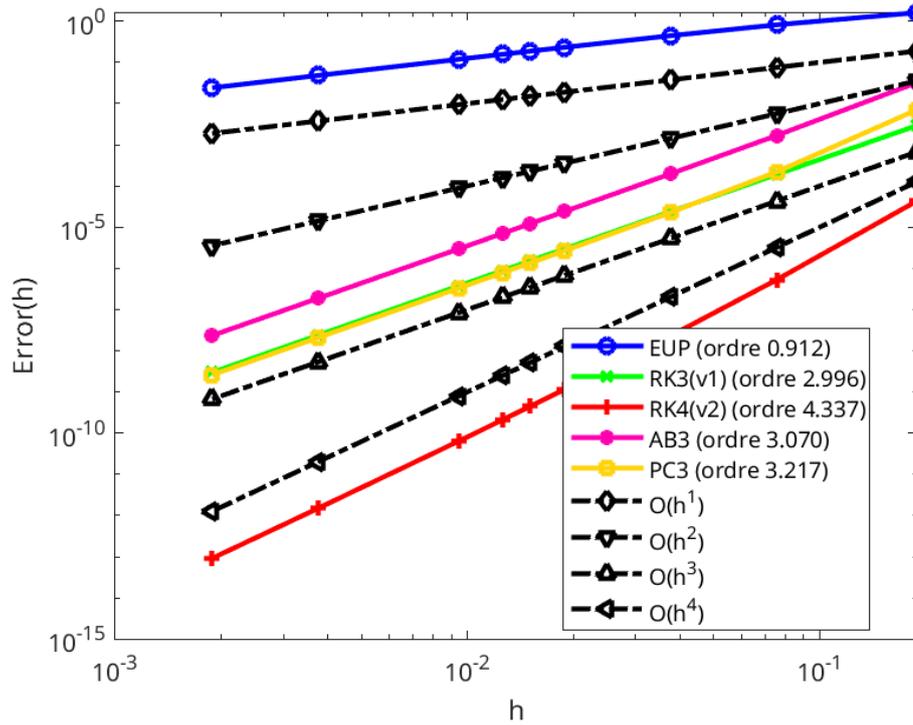


FIGURE 2 – Ordre des 5 schémas

A faire en 3h00 (temps indicatif)

- ◇ Créer une archive compressée nommée <NOM>-TP1-Q4 contenant les fichiers `redEUP.m`, `redRK3.m`, `redAB3.m`, `redRK4.m`, `redPC3.m`, `erreur.m`, `ordre.m` et toutes autres fonctions nécessaires à l'exécution des programmes `erreur.m` et `ordre.m`. Ici <NOM> correspond évidemment à votre nom.
- ◇ Envoyer un mail à cuvelier@math.univ-paris13.fr ayant pour **sujet** "<NOM> TP1 Q4" et en fichier joint l'archive compressée créée précédemment.

3 Le système solaire

Une archive compressée au format **zip**, nommée `CodesFournis_EDO.zip`, ou au format **tar.gz**, nommée `CodesFournis_EDO.tar.gz`, est disponible sur le serveur Discord. Il faut la télécharger et la décompresser dans un répertoire.

3.1 Position du problème et équations différentielles

Le but de cette partie est de prédire la position de planètes dans le système solaire à partir de positions et de vitesses initiales des planètes (voir fonction `SysSolData` fournie). Les unités choisies sont pour les masses, relatives au soleil, pour les distances, en unités astronomiques (1 U.A. = 149 597 870 km) et pour le temps, en jour terrestre.

En astrophysique, la masse solaire est l'unité de masse conventionnellement utilisée pour exprimer les masses des corps célestes massifs et des structures formées d'étoiles (amas, superamas, galaxies, etc.). Son symbole et sa valeur sont :

$$M_{\odot} = 1,9891 \times 10^{30} \text{ kg.}$$

La masse solaire peut aussi être déterminé par l'Unité astronomique (U.A.), l'année et le Constante gravitationnelle (G) :

$$M_{\odot} = \frac{4\pi^2 \times (1\text{UA})^3}{G \times (1\text{ans})^2}.$$

La constante gravitationnelle est alors

$$G = \frac{4\pi^2}{(365,25636567)^2} \approx 2.959130713485796 \times 10^{-4},$$

une année sidérale correspondant à 365,25636567 jours.

On note $\mathbf{q}_i(t)$ la position du i ème corps au temps t , $\dot{\mathbf{q}}_i(t)$ sa vitesse au temps t et m_i sa masse. Les données initiales pour le soleil (corps $i = 1$) sont $\mathbf{q}_1(0) = (0, 0, 0)^t$ et $\dot{\mathbf{q}}_1(0) = (0, 0, 0)^t$. Toutes les autres données sont fournies, récupérées sur JPL Solar System Dynamics, et sont disponibles dans le fichier `SysSolData.m` et correspondent aux positions de différentes planètes le 20 avril 2025 à 0h00m00s.

De manière classique, un problème à N corps se modélise par

$$\ddot{\mathbf{q}}_i(t) = -G \sum_{j=1, j \neq i}^N m_j \frac{\mathbf{q}_i(t) - \mathbf{q}_j(t)}{\|\mathbf{q}_i(t) - \mathbf{q}_j(t)\|^3}, \quad \forall i \in \llbracket 1, N \rrbracket.$$

et l'énergie du système ,correspondant à la somme des énergies cinétiques et des énergies potentielles gravitationnelle (voir wikipedia), est donnée par

$$\mathcal{E}(t) = \frac{1}{2} \sum_{i=1}^N \frac{1}{m_i} \|\dot{\mathbf{q}}_i(t)\|^2 - G \sum_{i=1}^N \sum_{j=1}^{i-1} \frac{m_i m_j}{\|\mathbf{q}_i(t) - \mathbf{q}_j(t)\|}.$$

3.2 Résolution numérique classique

Q. 5 [Sur feuille]

- a. Ecrire de manière détaillée le problème de Cauchy associé à un problème à 3 corps (Soleil, Saturne et Jupiter) sachant que la fonction vectorielle inconnue \mathbf{y} est définie par

$$\mathbf{y}(t) = \begin{pmatrix} \mathbf{q}_1(t) \\ \mathbf{q}_2(t) \\ \mathbf{q}_3(t) \\ \dot{\mathbf{q}}_1(t) \\ \dot{\mathbf{q}}_2(t) \\ \dot{\mathbf{q}}_3(t) \end{pmatrix} \in \mathbb{R}^{18}$$

- b. Quelles sont les données du problème de Cauchy obtenu ? (avec leur type détaillé : entier, réel, complexe, vecteur, matrice, fonction, ...)

c. Quelles sont les inconnues du problème de Cauchy obtenu ? (avec leur type détaillé : entier, réel, complexe, vecteur, matrice, fonction, ...)

□

A faire en 0h30 (temps indicatif)

- ◇ Envoyer un mail à cuvelier@math.univ-paris13.fr ayant pour **sujet** "<NOM> TP1 Q5" et en fichier joint un PDF contenant un scan de ce que vous avez écrit.

3.3 Système à 3 corps : Soleil, Saturne et Jupiter

Pour récupérer les données initiales de ce système, on peut utiliser les commandes Matlab :

```
1 SysSol=SysSolData();
2 SysSol.planetes=SysSol.planetes(1:3);
```

En effet, les données pour le Soleil, Saturne et Jupiter sont respectivement stockées dans `SysSol.planetes(1)`, `SysSol.planetes(2)` et `SysSol.planetes(3)`.

Les deux fonctions fournies `plotSysSol` et `PlotPlanets` (nécessite `PlotPlanet`) permettent de représenter, après calculs, respectivement l'orbite des planètes de la structure `SysSol` (voir fonction `SysSolData`) et les planètes à un instant donné. Leur syntaxe est la suivante `plotSysSol(T,Y,SysSol)` et `PlotPlanets(SysSol,q)` avec

- `(T,Y)` tableaux retournés lors de la résolution du problème de Cauchy associé,
- `SysSol` structure contenant divers renseignements sur le système solaire,
- `q` positions des planètes.

Q. 6 [Matlab]

- Ecrire la fonction Matlab `fSysSol3` (fichier `fSysSol3.m`) correspondant à la fonction f du problème de Cauchy associé au problème à 3 corps (Soleil, Saturne et Jupiter).
- Ecrire le programme `prgSysSol3` (fichier `prgSysSol3.m`) qui résout le problème de Cauchy par `redEUP` et `redRK4` puis représente, pour chaque méthode, les orbites des 3 planètes ainsi que leurs positions à l'instant final avec comme données `T=24000` (jours) et `N=8000` (voir figure 3) On représentera aussi l'énergie du système pour chaque méthode.

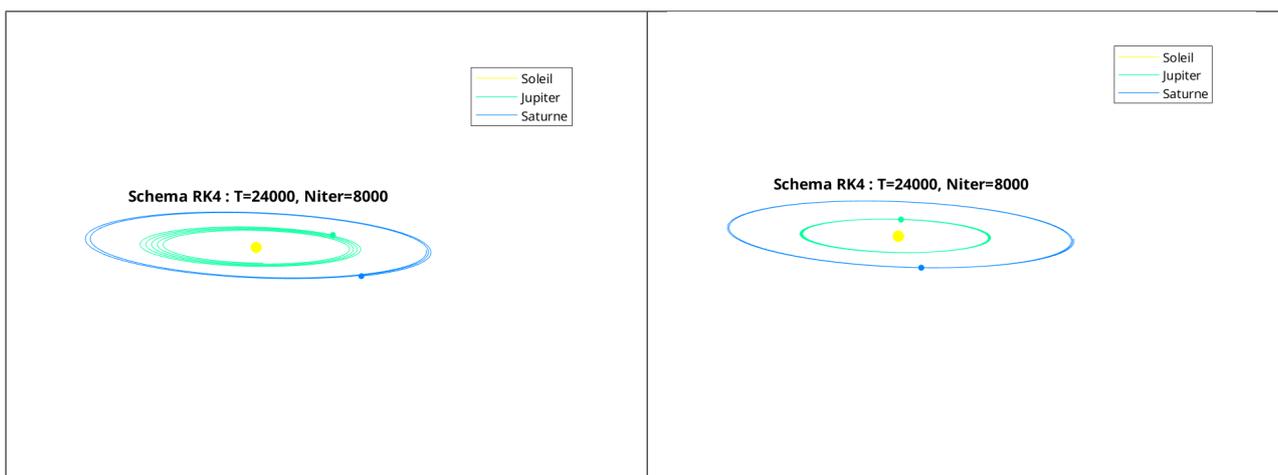


FIGURE 3 – Orbites Soleil-Saturne-Jupiter

□

— A faire en 1h30 (temps indicatif) —

- ◇ Créer une archive compressée nommée <NOM>-TP1-Q6 contenant l'ensemble des fichiers nécessaires à l'exécution de [prgSysSol3](#). Ici <NOM> correspond évidemment à votre nom.
- ◇ Envoyer un mail à cuvelier@math.univ-paris13.fr ayant pour **sujet** "<NOM> TP1 Q6" et en fichier joint l'archive compressée créée précédemment.

3.4 Système à 6 corps

On étudie ici le système composé des 6 planètes Soleil, Saturne, Jupiter, Uranus, Neptune et Pluton.

Q. 7 [Sur feuille]

- a. *Ecrire de manière détaillée le problème de Cauchy associé à un problème à N corps sachant que la fonction vectorielle inconnue \mathbf{y} est définie par*

$$\mathbf{y}(t) = \begin{pmatrix} \mathbf{q}_1(t) \\ \vdots \\ \mathbf{q}_N(t) \\ \dot{\mathbf{q}}_1(t) \\ \vdots \\ \dot{\mathbf{q}}_N(t) \end{pmatrix} \in \mathbb{R}^{6N}$$

- b. *Quelles sont les données du problème de Cauchy obtenu ? (avec leur type détaillé : entier, réel, complexe, vecteur, matrice, fonction, ...)*
- c. *Quelles sont les inconnues du problème de Cauchy obtenu ? (avec leur type détaillé : entier, réel, complexe, vecteur, matrice, fonction, ...)*

□

— A faire en 0h30 (temps indicatif) —

- ◇ Envoyer un mail à cuvelier@math.univ-paris13.fr ayant pour **sujet** "<NOM> TP1 Q7" et en fichier joint un PDF contenant un scan de ce que vous avez écrit.

Q. 8 [Matlab]

- a. *Ecrire la fonction Matlab [fSysSol6](#) (fichier [fSysSol6.m](#)) correspondant à la fonction \mathbf{f} du problème de Cauchy associé au problème à 6 corps (Soleil, Saturne, Jupiter, Uranus, Neptune et Pluton).*
- b. *Ecrire le programme [prgSysSol6](#) (fichier [prgSysSol6.m](#)) qui résout le problème de Cauchy par [redEUP](#) et [redRK4](#) puis représente, pour chaque méthode, les orbites des 6 planètes ainsi que leurs positions à l'instant final avec comme données $T=100000$ (jours) et $N=10000$ (voir figure 4).*

□

— A faire en 1h30 (temps indicatif) —

- ◇ Créer une archive compressée nommée <NOM>-TP1-Q8 contenant l'ensemble des fichiers nécessaires à l'exécution de [prgSysSol6](#). Ici <NOM> correspond évidemment à votre nom.
- ◇ Envoyer un mail à cuvelier@math.univ-paris13.fr ayant pour **sujet** "<NOM> TP1 Q8" et en fichier joint l'archive compressée créée précédemment.

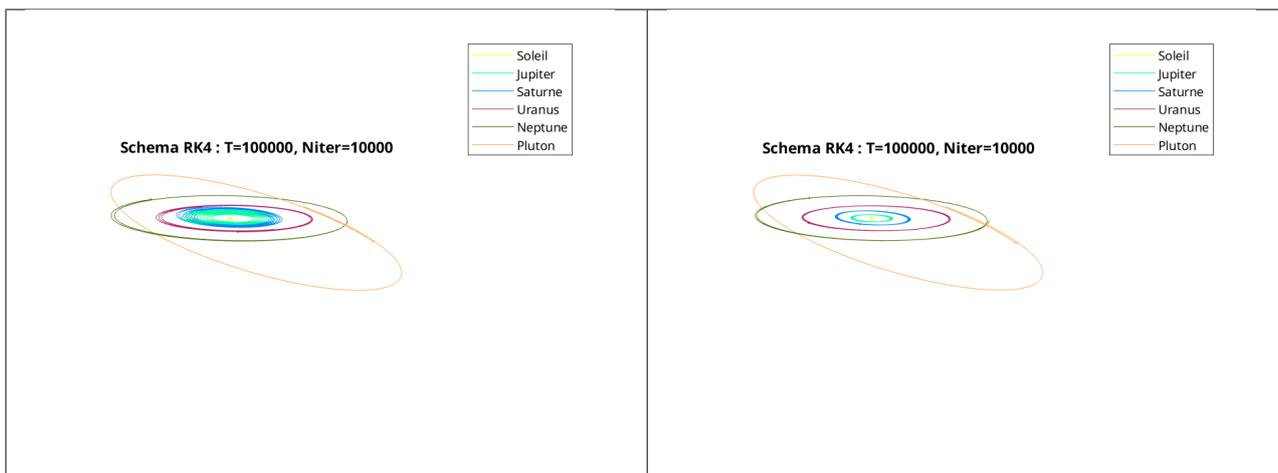


FIGURE 4 – Orbites Soleil, Saturne, Jupiter, Uranus, Neptune et Pluton

3.5 Système à 7 corps

On étudie ici le système composé des 7 planetes Soleil, Saturne, Jupiter, Uranus, Neptune, Pluton et Terre.

Q. 9 [Matlab]

- Ecrire la fonction Matlab `fSysSol7` (fichier `fSysSol7.m`) correspondant à la fonction f du problème de Cauchy associé au problème à 7 corps (Soleil, Saturne, Jupiter, Uranus, Neptune, Pluton et Terre). On pourra utiliser une variable globale pour les paramètres physiques (voir fichiers `SysSolDataHairer.m` ou `SysSolData.m`).
- Ecrire le programme `prgSysSol7` (fichier `prgSysSol7.m`) qui résoud le problème de Cauchy par `redEUP` et `redRK4` puis représente, pour chaque méthode, les orbites des 7 planètes ainsi que leurs positions à l'instant final avec comme données $T=100000$ (jours) et $N=10000$. On représentera aussi l'énergie du système pour chaque méthode.

□

A faire en 1h30 (temps indicatif)

- ◇ Créer une archive compressée nommée `<NOM>-TP1-Q9` contenant l'ensemble des fichiers nécessaire à l'exécution de `prgSysSol7`. Ici `<NOM>` correspond évidemment à votre nom.
- ◇ Envoyer un mail à `cuvelier@math.univ-paris13.fr` ayant pour **sujet** "`<NOM> TP1 Q9`" et en fichier joint l'archive compressée créée précédemment.

4 Annexes

4.1 Quelques E.D.O. du premier ordre

4.1.1 Exemple 1

Soit $\alpha \in \mathbb{R}$. L'E.D.O.

$$\begin{cases} y'(t) &= \cos(t), \quad \forall t \geq 0, \\ y(0) &= \alpha, \end{cases}$$

a pour solution $y(t) = \sin(t) + \alpha$.

4.1.2 Exemple 2

Soit $\beta \in \mathbb{R}$. L'E.D.O.

$$\begin{cases} y'(t) &= \sin(t), \forall t \geq 0, \\ y(0) &= \beta, \end{cases}$$

a pour solution $y(t) = -\cos(t) + 1 + \beta$.

4.1.3 Exemple 3

L'E.D.O.

$$\begin{cases} y'(t) &= -y(t) \sin(t), \forall t \geq 0, \\ y(0) &= e = \exp(1), \end{cases}$$

a pour solution $y(t) = \exp(\cos(t))$.