Energétique I **Méthodes Numériques II**
Sup'Galilée **Année 2018-2019**

Travaux pratiques - E.D.O.

Groupe 2
Durée : 10h00

Travail individuel et personnel

Table des matières

1 Tests Algorithmique et Matlab

Une archive compressée au format **zip**

`www.math.univ-paris13.fr/~cuvelier/docs/Enseignements/Energetique/MethNumII/18-19/TP1/G2/CodesFournis_Mosaiques.zip`

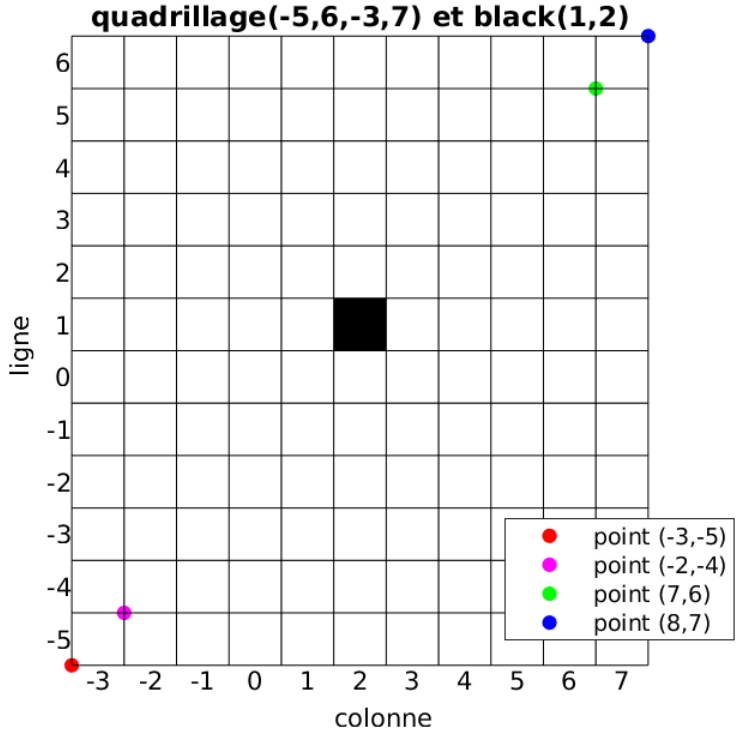
ou au format **tar.gz**

`www.math.univ-paris13.fr/~cuvelier/docs/Enseignements/Energetique/MethNumII/18-19/TP1/G2/CodesFournis_Mosaiques.tar.gz`

est disponible en ligne. Il faut télécharger l'archive et la décompresser dans un répertoire.

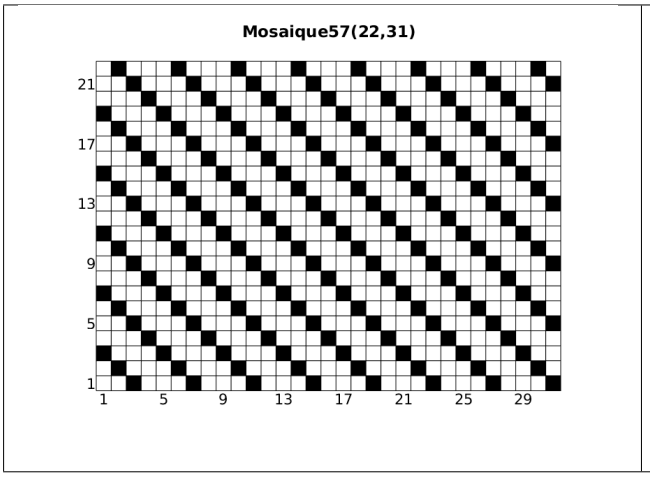
Cette archive contient , entre autres, la fonction `black` et le programme `Quadrillagefigure`. Dans le programme `Quadrillagefigure` l'appel à la fonction `Quadrillage` manquante a été mis en commentaire.

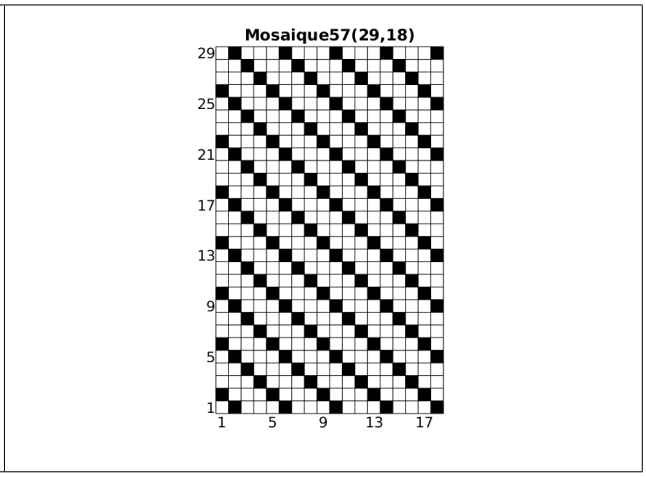
Q. 1 *Ecrire la fonction Matlab `Quadrillage(imin,imax,jmin,jmax)` permettant de générer un quadrillage pour les lignes `imin` à `imax` et les colonnes `jmin` à `jmax`. Voici un exemple avec la commande `Quadrillage(−5,6,−3,7)` représentant uniquement les traits noirs sur la figure :*



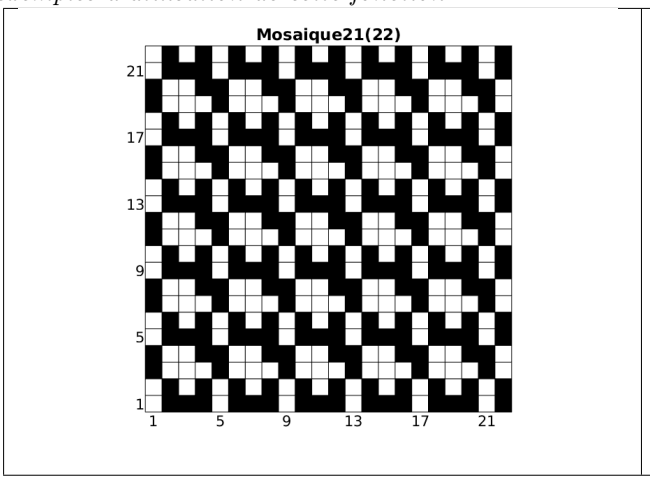
On peut noter que les coordonnées des points sont exprimées dans le plan classique xOy On peut tester cette fonction avec le programme `Quadrillagefigure` fourni pour obtenir la figure précédente.
Le carré noir en ligne 1 et colonne 2 a été représenté à l'aide de la commande `black(1,2)`, la fonction `black` étant fournie.
On rappelle que pour représenter un segment entre les points $A_1 = (x1, y1)$ et $A_2 = (x2, y2)$, on peut utiliser sous Matlab, la commande `plot([x1 x2],[y1 y2])`.

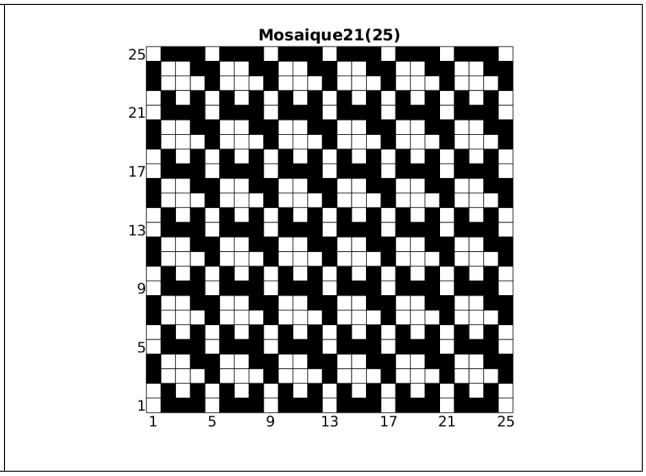
Q. 2 *Ecrire la fonction `Mosaique57(n,m)` permettant de créer une mosaïque sur le quadrillage `Quadrillage(1,n,1,m)` sachant que la case de position $(1, m)$ est noire. Voici deux exemples d'utilisation de cette fonction :*





Q. 3 *Ecrire la fonction* `Mosaique21(n)` *permettant de créer une mosaïque sur le quadrillage* `Quadrillage(1,n,1,n)` *sachant que la ligne* 1 *est composée de la séquence blanc,noir,noir,noir,blanc,noir,noir,noir, ... 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`, `Mosaique55.m`, `Mosaique24.m` et tout autre fichier permettant l'éxecution des fonctions `Mosaique55.m` et `Mosaique24.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 précédement.

2 Résolution numérique d'équations différentielles ordinaires

Pour une explication détaillée voir le polycopié fourni `MethNumII_25fevrier2019.pdf`

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

Definition 2.1: problème de Cauchy

Soit $\boldsymbol{f}$ l'application continue définie par

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

avec $T \in]0,+\infty]$. Le **problème de Cauchy** revient à chercher une fonction $\boldsymbol{y}$ définie par

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

continue et dérivable, telle que

$$\begin{aligned} \boldsymbol{y}'(t) &= \boldsymbol{f}(t,\boldsymbol{y}(t)), \quad \forall t \in [t^0, t^0+T] && (2.1)\\ \boldsymbol{y}(t^0) &= \boldsymbol{y}^{[0]} \in \mathbb{R}^d. && (2.2) \end{aligned}$$

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

2.1.1 Schéma d'Euler progressif (ordre 1)

$$\left\{ \begin{array}{lcl} \boldsymbol{y}^{[n+1]} &=& \boldsymbol{y}^{[n]} + h\boldsymbol{f}^{[n]}, \ \forall n \in [\![0, N-1]\!] \\ \boldsymbol{y}^{[0]} &=& \boldsymbol{y}(t^0) \end{array} \right. \tag{2.3}$$

2.1.2 Schéma de la tangente améliorée (ordre 2)

$$\left\{ \begin{array}{lcl} \boldsymbol{y}^{[n+1]} &=& \boldsymbol{y}^{[n]} + h\boldsymbol{f}(t^n + \frac{h}{2}, \boldsymbol{y}^{[n]} + \frac{h}{2}\boldsymbol{f}^{[n]}), \ \forall n \in [\![0, N-1]\!] \\ \boldsymbol{y}^{[0]} &=& \boldsymbol{y}(t^0) \end{array} \right. \tag{2.4}$$

2.1.3 Schéma de Heun (ordre 2)

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

2.1.4 Shéma de Cavalieri-Simpson (ordre 2)

$$\boldsymbol{y}^{[n+1]} = \boldsymbol{y}^{[n-1]} + \frac{h}{3}\left(\boldsymbol{f}^{[n+1]} + 4\boldsymbol{f}^{[n]} + \boldsymbol{f}^{[n-1]}\right) \tag{2.6}$$

2.1.5 Shéma de Milne (ordre 4)

$$\boldsymbol{y}^{[n+1]} = \boldsymbol{y}^{[n-3]} + \frac{4h}{3}\left(2\boldsymbol{f}^{[n]} - \boldsymbol{f}^{[n-1]} + 2\boldsymbol{f}^{[n-2]}\right) \tag{2.7}$$

2.1.6 Schémas de Runge-Kutta

- ordre 3 (version 1)

$$\begin{array}{lcl} \boldsymbol{k}_1^{[n]} &=& \boldsymbol{f}(t^n, \boldsymbol{y}^{[n]}) \\ \boldsymbol{k}_2^{[n]} &=& \boldsymbol{f}(t^n + \frac{h}{3}, \boldsymbol{y}^{[n]} + \frac{h}{3}\boldsymbol{k}_1) \\ \boldsymbol{k}_3^{[n]} &=& \boldsymbol{f}(t^n + \frac{2h}{3}, \boldsymbol{y}^{[n]} + \frac{2h}{3}\boldsymbol{k}_2) \\ \hline \boldsymbol{y}^{[n+1]} &=& \boldsymbol{y}^{[n]} + \frac{h}{4}(\boldsymbol{k}_1 + 3\boldsymbol{k}_3)). \end{array} \tag{2.8}$$

- ordre 3 (version 2)

$$\begin{array}{lcl} \boldsymbol{k}_1^{[n]} & = & \boldsymbol{f}(t^n, \boldsymbol{y}^{[n]}) \\ \boldsymbol{k}_2^{[n]} & = & \boldsymbol{f}(t^n + \frac{2h}{3}, \boldsymbol{y}^{[n]} + \frac{2h}{3}\boldsymbol{k}_1) \\ \boldsymbol{k}_3^{[n]} & = & \boldsymbol{f}(t^n + \frac{2h}{3}, \boldsymbol{y}^{[n]} + \frac{2h}{3}\boldsymbol{k}_2) \\ \hline \boldsymbol{y}^{[n+1]} & = & \boldsymbol{y}^{[n]} + \frac{h}{8}(2\boldsymbol{k}_1 + 3\boldsymbol{k}_2 + 3\boldsymbol{k}_3)). \end{array} \tag{2.9}$$

- ordre 3 (version 3)

$$\begin{array}{lcl} \boldsymbol{k}_1^{[n]} & = & \boldsymbol{f}(t^n, \boldsymbol{y}^{[n]}) \\ \boldsymbol{k}_2^{[n]} & = & \boldsymbol{f}(t^n + \frac{2h}{3}, \boldsymbol{y}^{[n]} + \frac{2h}{3}\boldsymbol{k}_1) \\ \boldsymbol{k}_3^{[n]} & = & \boldsymbol{f}(t^n, \boldsymbol{y}^{[n]} - h\boldsymbol{k}_1 + h\boldsymbol{k}_2) \\ \hline \boldsymbol{y}^{[n+1]} & = & \boldsymbol{y}^{[n]} + \frac{h}{4}(3\boldsymbol{k}_2 + \boldsymbol{k}_3)). \end{array} \tag{2.10}$$

- ordre 4 (version 1)

$$\begin{array}{lcl} \boldsymbol{k}_1^{[n]} & = & \boldsymbol{f}(t^n, \boldsymbol{y}^{[n]}) \\ \boldsymbol{k}_2^{[n]} & = & \boldsymbol{f}(t^n + \frac{h}{2}, \boldsymbol{y}^{[n]} + \frac{h}{2}\boldsymbol{k}_1^{[n]}) \\ \boldsymbol{k}_3^{[n]} & = & \boldsymbol{f}(t^n + \frac{h}{2}, \boldsymbol{y}^{[n]} + \frac{h}{2}\boldsymbol{k}_2^{[n]}) \\ \boldsymbol{k}_4^{[n]} & = & \boldsymbol{f}(t^n + h, \boldsymbol{y}^{[n]} + h\boldsymbol{k}_3^{[n]}) \\ \hline \boldsymbol{y}^{[n+1]} & = & \boldsymbol{y}^{[n]} + \frac{h}{6}\big(\boldsymbol{k}_1^{[n]} + 2\boldsymbol{k}_2^{[n]} + 2\boldsymbol{k}_3^{[n]} + \boldsymbol{k}_4^{[n]}\big). \end{array} \tag{2.11}$$

- ordre 4 (version 2)

$$\begin{array}{lcl} \boldsymbol{k}_1^{[n]} & = & \boldsymbol{f}(t^n, \boldsymbol{y}^{[n]}) \\ \boldsymbol{k}_2^{[n]} & = & \boldsymbol{f}(t^n + \frac{h}{3}, \boldsymbol{y}^{[n]} + \frac{h}{3}\boldsymbol{k}_1^{[n]}) \\ \boldsymbol{k}_3^{[n]} & = & \boldsymbol{f}(t^n + \frac{2h}{3}, \boldsymbol{y}^{[n]} - \frac{h}{3}\boldsymbol{k}_1^{[n]} + h\boldsymbol{k}_2^{[n]}) \\ \boldsymbol{k}_4^{[n]} & = & \boldsymbol{f}(t^n + h, \boldsymbol{y}^{[n]} + h\boldsymbol{k}_1^{[n]} - h\boldsymbol{k}_2^{[n]} + h\boldsymbol{k}_3^{[n]}) \\ \hline \boldsymbol{y}^{[n+1]} & = & \boldsymbol{y}^{[n]} + \frac{h}{8}\big(\boldsymbol{k}_1^{[n]} + 3\boldsymbol{k}_2^{[n]} + 3\boldsymbol{k}_3^{[n]} + \boldsymbol{k}_4^{[n]}\big). \end{array} \tag{2.12}$$

2.1.7 Méthodes d'Adams-Bashforth

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

$$\boldsymbol{y}^{[n+1]} = \boldsymbol{y}^{[n]} + \frac{h}{2}\left(3\boldsymbol{f}^{[n]} - \boldsymbol{f}^{[n-1]}\right). \tag{2.13}$$

$$\boldsymbol{y}^{[n+1]} = \boldsymbol{y}^{[n]} + \frac{h}{12}\left(23\boldsymbol{f}^{[n]} - 16\boldsymbol{f}^{[n-1]} + 5\boldsymbol{f}^{[n-2]}\right). \tag{2.14}$$

$$\boldsymbol{y}^{[n+1]} = \boldsymbol{y}^{[n]} + \frac{h}{24}\left(55\boldsymbol{f}^{[n]} - 59\boldsymbol{f}^{[n-1]} + 37\boldsymbol{f}^{[n-2]} - 9\boldsymbol{f}^{[n-3]}\right). \tag{2.15}$$

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

2.1.8 Méthodes d'Adams-Moulton

$$\boldsymbol{y}^{[n+1]} = \boldsymbol{y}^{[n]} + \frac{h}{2}\left(\boldsymbol{f}^{[n+1]} + \boldsymbol{f}^{[n]}\right). \tag{2.16}$$

$$\boldsymbol{y}^{[n+1]} = \boldsymbol{y}^{[n]} + \frac{h}{12}\left(5\boldsymbol{f}^{[n+1]} + 8\boldsymbol{f}^{[n]} - \boldsymbol{f}^{[n-1]}\right). \tag{2.17}$$

$$\boldsymbol{y}^{[n+1]} = \boldsymbol{y}^{[n]} + \frac{h}{24}\left(9\boldsymbol{f}^{[n+1]} + 19\boldsymbol{f}^{[n]} - 5\boldsymbol{f}^{[n-1]} + \boldsymbol{f}^{[n-2]}\right). \tag{2.18}$$

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

2.1.9 Schéma de Nyström (ordre 3)

$$\boldsymbol{y}^{[n+1]} = \boldsymbol{y}^{[n-1]} + \frac{h}{3}\left(7\boldsymbol{f}(t^n, \boldsymbol{y}^{[n]}) - 2\boldsymbol{f}(t^{n-1}, \boldsymbol{y}^{[n-1]}) + \boldsymbol{f}(t^{n-2}, \boldsymbol{y}^{[n-2]})\right) \tag{2.19}$$

2.1.10 Schémas d'Euler rétrograde (Backward-Difference Methods)

- ordre 1
$$\boldsymbol{y}^{[n+1]} = \boldsymbol{y}^{[n]} + h\boldsymbol{f}^{[n+1]} \tag{2.20}$$
- ordre 2
$$\boldsymbol{y}^{[n+1]} = \frac{4}{3}\boldsymbol{y}^{[n]} - \frac{1}{3}\boldsymbol{y}^{[n-1]} + \frac{2h}{3}\boldsymbol{f}^{[n+1]} \tag{2.21}$$
- ordre 3
$$\boldsymbol{y}^{[n+1]} = \frac{1}{11}\left(18\boldsymbol{y}^{[n]} - 9\boldsymbol{y}^{[n-1]} + 2\boldsymbol{y}^{[n-2]} + 6h\boldsymbol{f}^{[n+1]}\right) \tag{2.22}$$
- ordre 4
$$\boldsymbol{y}^{[n+1]} = \frac{1}{25}\left(48\boldsymbol{y}^{[n]} - 36\boldsymbol{y}^{[n-1]} + 16\boldsymbol{y}^{[n-2]} - 3\boldsymbol{y}^{[n-3]} + 12h\boldsymbol{f}^{[n+1]}\right) \tag{2.23}$$

2.1.11 Shéma de Simpson (ordre 4)

$$\boldsymbol{y}^{[n+1]} = \boldsymbol{y}^{[n-1]} + \frac{h}{3}\left(\boldsymbol{f}^{[n+1]} + 4\boldsymbol{f}^{[n]} + \boldsymbol{f}^{[n-1]}\right) \tag{2.24}$$

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 : on fournit explicitement une valeur approchée de la solution au $n^{ième}$ pas (soit $\overline{\boldsymbol{y}}^{(n+1)}$), puis on calcule la valeur correspondante de $\boldsymbol{f}$ (soit $\overline{\boldsymbol{f}}^{(n+1)}$) et enfin, on subsitue cette valeur dans un schéma implicite (on obtient alors une valeur *corrigée*).

pour n variant de 0 à $N-1$ faire
Calculer une valeur approchée $\overline{\boldsymbol{y}}^{(n+1)}$ par un schéma explicite ;
Evaluer $\overline{\boldsymbol{f}}^{(n+1)} = \boldsymbol{f}(t^{n+1}, \overline{\boldsymbol{y}}^{(n+1)})$;
$\boldsymbol{y}^{[n+1]}$) à l'aide d'un schéma implicite en remplaçant l'inconnue par $\overline{\boldsymbol{y}}^{(n+1)}$);
finpour

2.3 Travail à effectuer

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

Q. 4 *1. 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]=redTGA(f,a,b,yo,N)` : schéma de la tangente améliorée* (2.4) *(fichier `redTGA.m`).*
- *`[t,Y]=redRK4(f,a,b,yo,N)` : schéma de Runge et Kutta d'ordre 4 (version 1)* (2.11) *(fichier `redRK4.m`).*
- *`[t,Y]=redMI4(f,a,b,yo,N)` : schéma de Milne d'ordre* 4 (2.7) *(fichier `redMI4.m`).*
- *`[t,Y]=redPC4(f,a,b,yo,N)` : schéma de type prédiction-correction utilisant les schémas de Milne d'ordre* 4 (2.7) *et Simpson d'ordre* 4 (2.24) *(fichier `redPC4.m`).*

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

2. 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 $(t^n, abs(\boldsymbol{y}^{[n]} - \boldsymbol{y}(t^n))),\ n \in \{0,\cdots,N\}$.
*Voir la figure 1 pour un exemple de tracé. Pour cette figure, la commande Matlab **subplot** a été utilisée.*

3. *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.

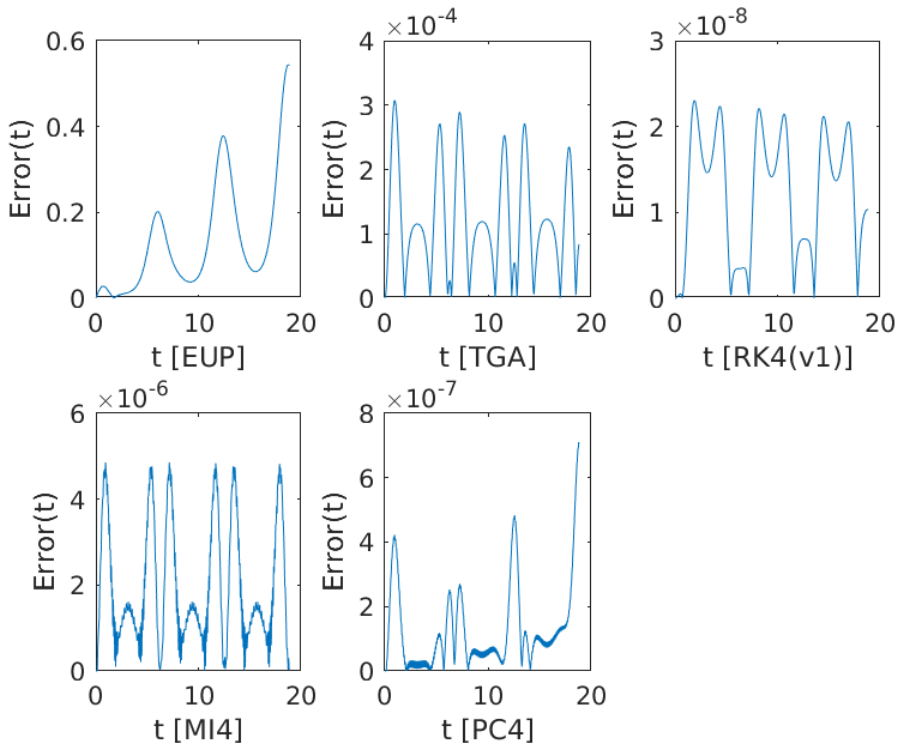


FIGURE 1 – Valeurs absolues des erreurs des 5 schémas représentées dans une unique figure

A faire en 3h00 (temps indicatif)

- ◇ Créer une archive compressée nommée `<NOM>-TP1-Q4` contenant les fichiers `redEUP.m`, `redTGA.m`, `redMI4.m`, `redRK4.m`, `redPC4.m`, `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 précédement.

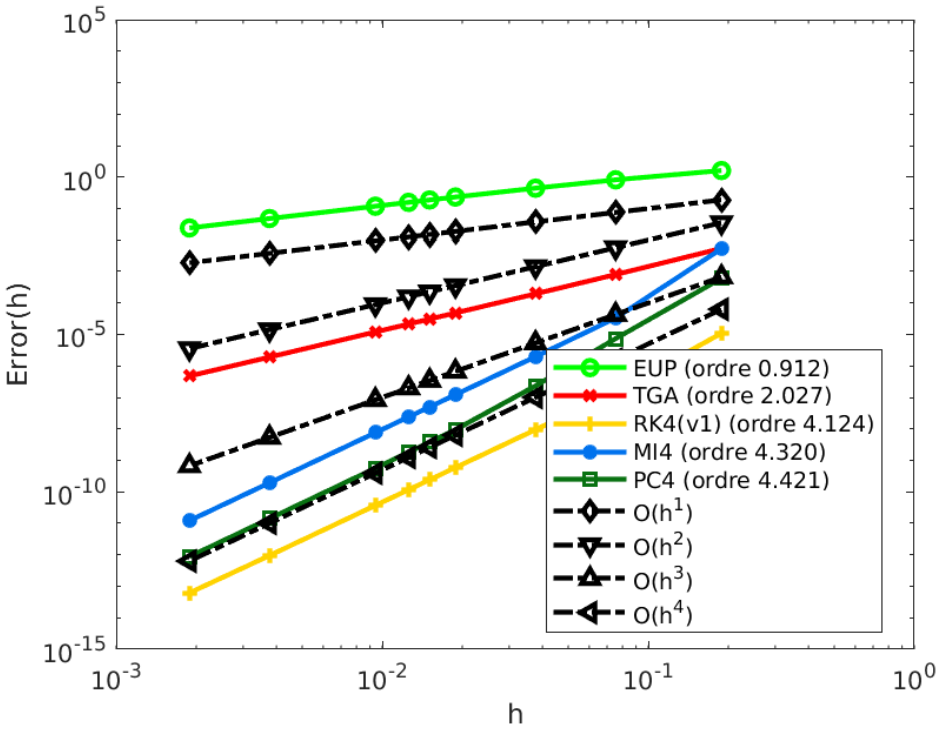


FIGURE 2 – Ordre des 5 schémas

3 Le système solaire

Une archive compressée au format **zip**

`www.math.univ-paris13.fr/~cuvelier/docs/Enseignements/Energetique/MethNumII/18-19/TP1/G2/CodesFournis_EDO.zip`

ou au format **tar.gz**

`www.math.univ-paris13.fr/~cuvelier/docs/Enseignements/Energetique/MethNumII/18-19/TP1/G2/CodesFournis_EDO.tar.gz`

est disponible en ligne. Il faut télécharger l'archive 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. Les unités choisies sont : les masses sont relatives au soleil, les distances sont en unités astronomiques (1 U.A. = 149 597 870 km) et 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 $\boldsymbol{q}_i(t)$ la position du ième corps au temps $t.$, $\dot{\boldsymbol{q}}_i(t)$ sa vitesse au temps t et m_i sa masse. Les données initiales pour le soleil (corps $i=1$) sont $\boldsymbol{q}_1 = (0,0,0)^t$ et $\dot{\boldsymbol{q}}_1 = (0,0,0)^t$. Toutes les autres données sont fournies dans [1] page 14, regroupées dans le fichier `SysSolDataHairer.m` et correspondent aux positions de différentes planètes le 5 septembre 1994 à 0h00m00s. D'autres données, récupérées sur JPL Solar System Dynamics sont disponibles dans le fichier `SysSolData.m` et correspondent aux positions de différentes planètes le 6 mai 2019 à 0h00m00s.

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

$$\ddot{\boldsymbol{q}}_i(t) = -G \sum_{j=1,\ j\neq i}^{N} m_j \frac{\boldsymbol{q}_i(t) - \boldsymbol{q}_j(t)}{\|\boldsymbol{q}_i(t) - \boldsymbol{q}_j(t)\|^3},\ \forall i \in [\![1, N]\!].$$

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

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

3.2 Résolution numérique classique

Q. 5 (Sur feuille) *1. 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 $\boldsymbol{y}$ est définie par*

$$\boldsymbol{y}(t) = \left(\begin{array}{c} \boldsymbol{q}_1(t) \\ \vdots \\ \boldsymbol{q}_3(t) \\ \hline \dot{\boldsymbol{q}}_1(t) \\ \vdots \\ \dot{\boldsymbol{q}}_3(t) \end{array}\right) \in \mathbb{R}^{18}$$

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

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

3.3 Système à 3 corps : Soleil, Saturne et Jupiter

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

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

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

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

Q. 6 (Matlab) *1. Ecrire la fonction Matlab `fSysSol3` (fichier `fSysSol3.m`) correspondant à la fonction $\boldsymbol{f}$ du problème de Cauchy associé au problème à 3 corps (Soleil, Saturne et Jupiter).*

2. *Ecrire le programme* `prgSysSol3` *(fichier* `prgSysSol3.m`*) qui résoud 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'energie du système pour chaque méthode.*

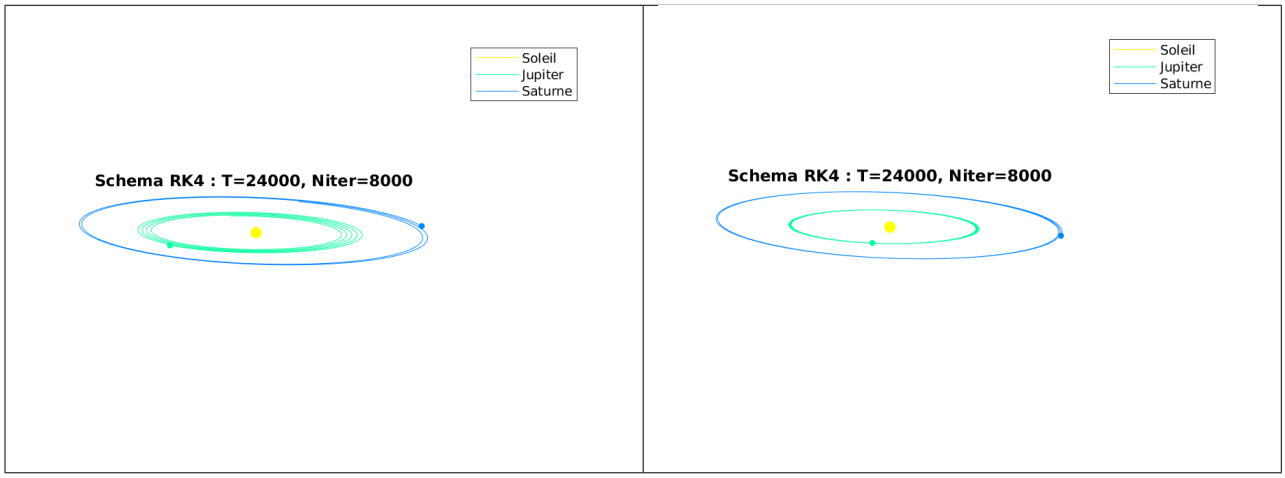


FIGURE 3 – Orbites Soleil-Saturne-Jupiter

A faire en 1h30 (temps indicatif)

- ◇ Créer une archive compressée nommée `<NOM>-TP1-Q6` contenant l'ensembles des fichiers nécessaire 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 précédement.

3.4 Système à 6 corps

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

Q. 7 (Sur feuille) *1. Ecrire de manière détaillée le problème de Cauchy associé à un problème à N corps sachant que la fonction vectorielle inconnue $\boldsymbol{y}$ est définie par*

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

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

Q. 8 (Matlab) *1. Ecrire la fonction Matlab* `fSysSol6` *(fichier* `fSysSol6.m`*) correspondant à la fonction $\boldsymbol{f}$ du problème de Cauchy associé au problème à 6 corps (Soleil, Saturne, Jupiter, Uranus, Neptune et Pluton).*

2. *Ecrire le programme* `prgSysSol6` *(fichier* `prgSysSol6.m`*) qui résoud 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) On représentera aussi l'energie du système pour chaque méthode.*

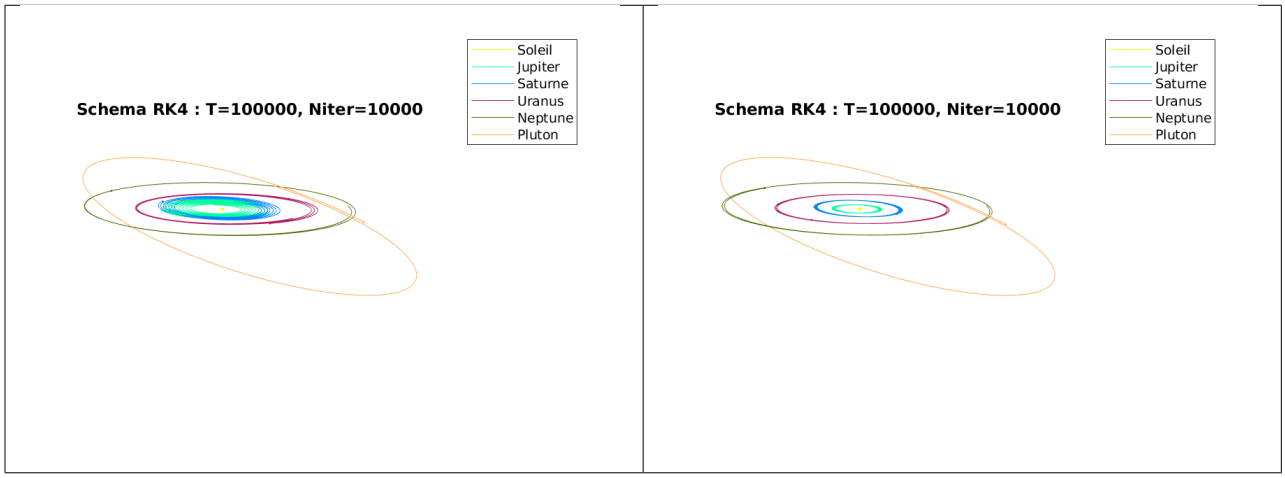


FIGURE 4 – Orbites Soleil, Saturne, Jupiter, Uranus, Neptune et Pluton

A faire en 1h30 (temps indicatif)

- ◇ Créer une archive compressée nommée `<NOM>-TP1-Q8` contenant l'ensembles des fichiers nécessaire 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 précédement.

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) *1. Ecrire la fonction Matlab* `fSysSol7` *(fichier* `fSysSol7.m`*) correspondant à la fonction* $\boldsymbol{f}$ *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`*).*

2. 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'energie du système pour chaque méthode.*

A faire en 1h30 (temps indicatif)

- ◇ Créer une archive compressée nommée `<NOM>-TP1-Q9` contenant l'ensembles 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 précédement.

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.

$$\left\{ \begin{array}{rcl} y'(t) & = & \cos(t),\ \forall t \geqslant 0, \\ y(0) & = & \alpha, \end{array} \right.$$

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

4.1.2 Exemple 2

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

$$\left\{ \begin{array}{rcl} y'(t) & = & \sin(t),\ \forall t \geqslant 0, \\ y(0) & = & \beta, \end{array} \right.$$

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

4.1.3 Exemple 3

L'E.D.O.

$$\left\{ \begin{array}{rcl} y'(t) & = & -y(t)\sin(t),\ \forall t \geqslant 0, \\ y(0) & = & e = \exp(1), \end{array} \right.$$

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

Références

[1] C. Lubich E. Hairer and G. Wanner. *Geometrical Numerical Integration.* Springer, 2006.