

TRAVAUX PRATIQUES - E.D.O.

Groupe 3
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	6
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	7
2.3 Travail à effectuer	7
3 Le pendule pesant	10
3.1 Position du problème et équations différentielles	10
3.2 Résolution numérique	10
3.3 Portrait de phase du pendule	11
4 Annexes	13
4.1 listings	13
4.2 Quelques E.D.O. du premier ordre	14
4.2.1 Exemple 1	14
4.2.2 Exemple 2	14
4.2.3 Exemple 3	14

1 Tests Algorithmique et Matlab

Une archive compressée au format **zip**

www.math.univ-paris13.fr/~cuvelier/docs/Enseignements/Energetique/MethNumII/21-22/TP1/G3/CodesFournis_Mosaiques.zip

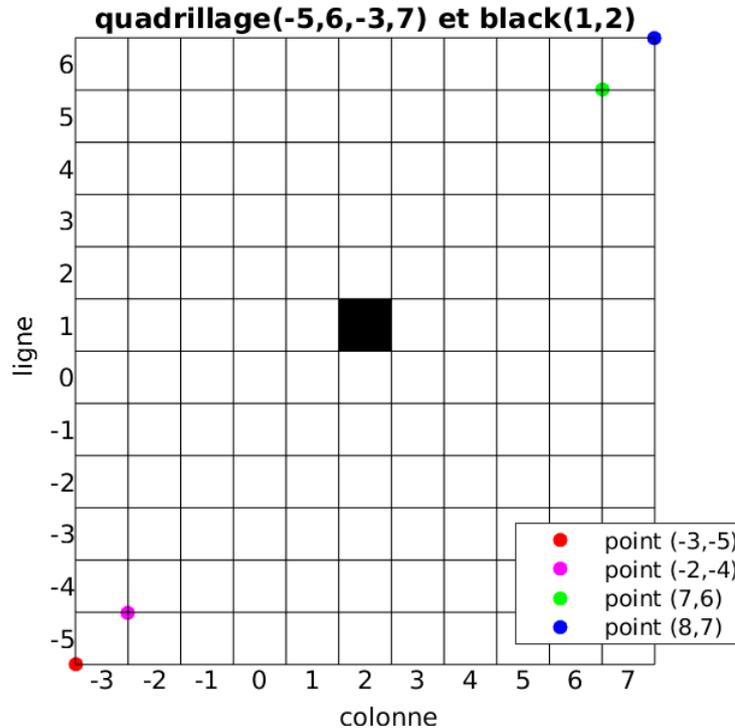
ou au format **tar.gz**

www.math.univ-paris13.fr/~cuvelier/docs/Enseignements/Energetique/MethNumII/21-22/TP1/G3/CodesFournis_Mosaiques.tar.gz

est disponible en ligne (page web). 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 :*



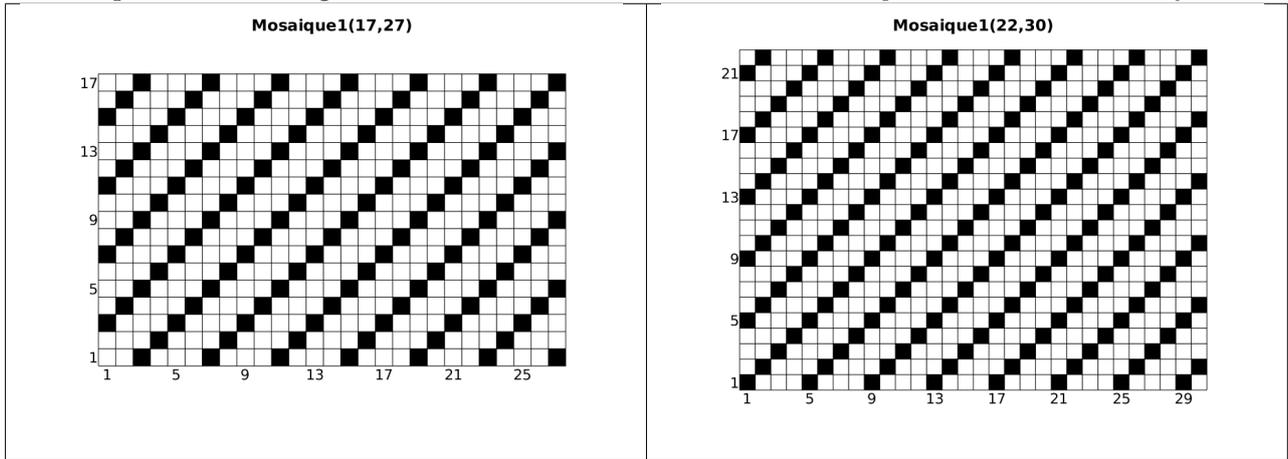
On peut noter que les coordonnées des points sont exprimées dans le plan classique xOy .

On peut tester cette fonction avec le programme `Quadrillagefigure` fourni pour obtenir la figure précédente.

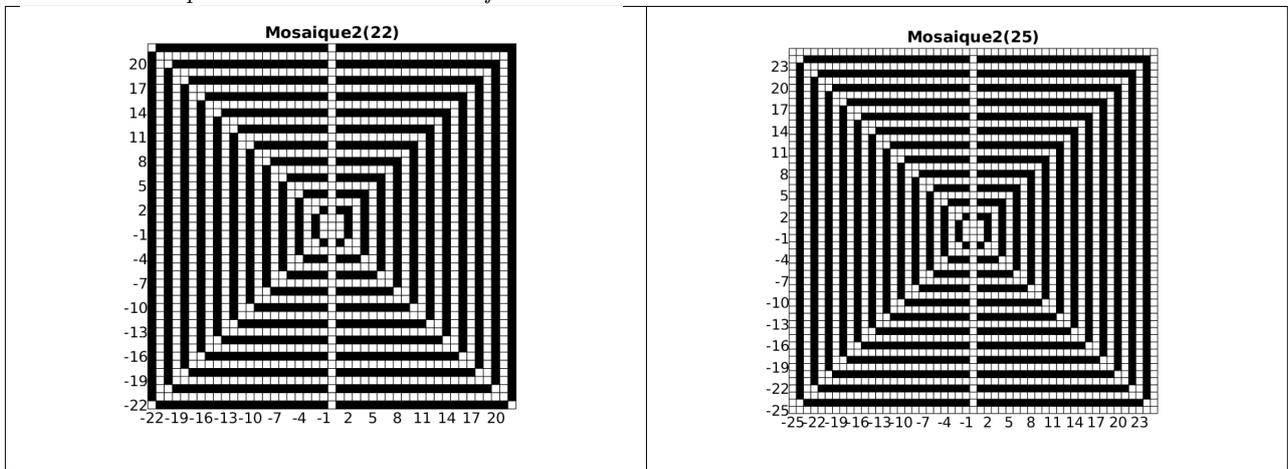
Le carré noir en ligne 1 et colonne 2 a été représenté à l'aide de la commande `black(1,2)`, la fonction `black` étant fournie.

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

Q. 2 Ecrire la fonction `Mosaique1(n,m)` permettant de créer une mosaïque sur le quadrillage `Quadrillage(1,n,1,m)` sachant que la case en ligne n et colonne m est noire. Voici deux exemples d'utilisation de cette fonction :



Q. 3 Ecrire la fonction `Mosaique2(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`, `Mosaique1.m`, `Mosaique2.m` et tout autre fichier permettant l'exécution des fonctions `Mosaique1.m` et `Mosaique2.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_25fevrier2020.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

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

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

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

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)

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

- ordre 3 (version 1)

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

- ordre 3 (version 2)

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

- ordre 3 (version 3)

$$\begin{aligned}
\mathbf{k}_1^{[n]} &= \mathbf{f}(t^n, \mathbf{y}^{[n]}) \\
\mathbf{k}_2^{[n]} &= \mathbf{f}\left(t^n + \frac{h}{2}, \mathbf{y}^{[n]} + \frac{h}{2}\mathbf{k}_1^{[n]}\right) \\
\mathbf{k}_3^{[n]} &= \mathbf{f}\left(t^{n+1}, \mathbf{y}^{[n]} - h\mathbf{k}_1^{[n]} + 2h\mathbf{k}_2^{[n]}\right) \\
\mathbf{y}^{[n+1]} &= \mathbf{y}^{[n]} + \frac{h}{6}(\mathbf{k}_1^{[n]} + 4\mathbf{k}_2^{[n]} + \mathbf{k}_3^{[n]}).
\end{aligned} \tag{2.11}$$

- ordre 3 (version 4)

$$\begin{aligned}
\mathbf{k}_1^{[n]} &= \mathbf{f}(t^n, \mathbf{y}^{[n]}) \\
\mathbf{k}_2^{[n]} &= \mathbf{f}\left(t^n + \frac{1}{4}h, \mathbf{y}^{[n]} + \frac{1}{4}h\mathbf{k}_1^{[n]}\right) \\
\mathbf{k}_3^{[n]} &= \mathbf{f}\left(t^n + h, \mathbf{y}^{[n]} + h\left(-\frac{7}{5}\mathbf{k}_1^{[n]} + \frac{12}{5}\mathbf{k}_2^{[n]}\right)\right) \\
\mathbf{y}^{[n+1]} &= \mathbf{y}^{[n]} + h\left(-\frac{1}{6}\mathbf{k}_1^{[n]} + \frac{8}{9}\mathbf{k}_2^{[n]} + \frac{5}{18}\mathbf{k}_3^{[n]}\right).
\end{aligned} \tag{2.12}$$

- ordre 4 (version 1)

$$\begin{aligned}
\mathbf{k}_1^{[n]} &= \mathbf{f}(t^n, \mathbf{y}^{[n]}) \\
\mathbf{k}_2^{[n]} &= \mathbf{f}\left(t^n + \frac{h}{2}, \mathbf{y}^{[n]} + \frac{h}{2}\mathbf{k}_1^{[n]}\right) \\
\mathbf{k}_3^{[n]} &= \mathbf{f}\left(t^n + \frac{h}{2}, \mathbf{y}^{[n]} + \frac{h}{2}\mathbf{k}_2^{[n]}\right) \\
\mathbf{k}_4^{[n]} &= \mathbf{f}\left(t^n + h, \mathbf{y}^{[n]} + h\mathbf{k}_3^{[n]}\right) \\
\mathbf{y}^{[n+1]} &= \mathbf{y}^{[n]} + \frac{h}{6}(\mathbf{k}_1^{[n]} + 2\mathbf{k}_2^{[n]} + 2\mathbf{k}_3^{[n]} + \mathbf{k}_4^{[n]}).
\end{aligned} \tag{2.13}$$

- ordre 4 (version 2)

$$\begin{aligned}
\mathbf{k}_1^{[n]} &= \mathbf{f}(t^n, \mathbf{y}^{[n]}) \\
\mathbf{k}_2^{[n]} &= \mathbf{f}\left(t^n + \frac{h}{3}, \mathbf{y}^{[n]} + \frac{h}{3}\mathbf{k}_1^{[n]}\right) \\
\mathbf{k}_3^{[n]} &= \mathbf{f}\left(t^n + \frac{2h}{3}, \mathbf{y}^{[n]} - \frac{h}{3}\mathbf{k}_1^{[n]} + h\mathbf{k}_2^{[n]}\right) \\
\mathbf{k}_4^{[n]} &= \mathbf{f}\left(t^n + h, \mathbf{y}^{[n]} + h\mathbf{k}_1^{[n]} - h\mathbf{k}_2^{[n]} + h\mathbf{k}_3^{[n]}\right) \\
\mathbf{y}^{[n+1]} &= \mathbf{y}^{[n]} + \frac{h}{8}(\mathbf{k}_1^{[n]} + 3\mathbf{k}_2^{[n]} + 3\mathbf{k}_3^{[n]} + \mathbf{k}_4^{[n]}).
\end{aligned} \tag{2.14}$$

- ordre 4 (version 3)

$$\begin{aligned}
\mathbf{k}_1^{[n]} &= \mathbf{f}(t^n, \mathbf{y}^{[n]}) \\
\mathbf{k}_2^{[n]} &= \mathbf{f}\left(t^n + \frac{h}{4}, \mathbf{y}^{[n]} + \frac{h}{4}\mathbf{k}_1^{[n]}\right) \\
\mathbf{k}_3^{[n]} &= \mathbf{f}\left(t^n + \frac{h}{2}, \mathbf{y}^{[n]} + \frac{h}{2}\mathbf{k}_2^{[n]}\right) \\
\mathbf{k}_4^{[n]} &= \mathbf{f}\left(t^n + h, \mathbf{y}^{[n]} + h\mathbf{k}_1^{[n]} - 2h\mathbf{k}_2^{[n]} + 2h\mathbf{k}_3^{[n]}\right) \\
\mathbf{y}^{[n+1]} &= \mathbf{y}^{[n]} + \frac{h}{6}(\mathbf{k}_1^{[n]} + 4\mathbf{k}_3^{[n]} + \mathbf{k}_4^{[n]}).
\end{aligned} \tag{2.15}$$

2.1.5 Méthodes d'Adams-Bashforth

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

$$\mathbf{y}^{[n+1]} = \mathbf{y}^{[n]} + \frac{h}{2} \left(3\mathbf{f}^{[n]} - \mathbf{f}^{[n-1]} \right). \quad (2.16)$$

$$\mathbf{y}^{[n+1]} = \mathbf{y}^{[n]} + \frac{h}{12} \left(23\mathbf{f}^{[n]} - 16\mathbf{f}^{[n-1]} + 5\mathbf{f}^{[n-2]} \right). \quad (2.17)$$

$$\mathbf{y}^{[n+1]} = \mathbf{y}^{[n]} + \frac{h}{24} \left(55\mathbf{f}^{[n]} - 59\mathbf{f}^{[n-1]} + 37\mathbf{f}^{[n-2]} - 9\mathbf{f}^{[n-3]} \right). \quad (2.18)$$

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

2.1.6 Méthodes d'Adams-Moulton

$$\mathbf{y}^{[n+1]} = \mathbf{y}^{[n]} + \frac{h}{2} \left(\mathbf{f}^{[n+1]} + \mathbf{f}^{[n]} \right). \quad (2.19)$$

$$\mathbf{y}^{[n+1]} = \mathbf{y}^{[n]} + \frac{h}{12} \left(5\mathbf{f}^{[n+1]} + 8\mathbf{f}^{[n]} - \mathbf{f}^{[n-1]} \right). \quad (2.20)$$

$$\mathbf{y}^{[n+1]} = \mathbf{y}^{[n]} + \frac{h}{24} \left(9\mathbf{f}^{[n+1]} + 19\mathbf{f}^{[n]} - 5\mathbf{f}^{[n-1]} + \mathbf{f}^{[n-2]} \right). \quad (2.21)$$

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

2.1.7 Schéma de Nyström (ordre 3)

$$\mathbf{y}^{[n+1]} = \mathbf{y}^{[n-1]} + \frac{h}{3} \left(7\mathbf{f}(t^n, \mathbf{y}^{[n]}) - 2\mathbf{f}(t^{n-1}, \mathbf{y}^{[n-1]}) + \mathbf{f}(t^{n-2}, \mathbf{y}^{[n-2]}) \right) \quad (2.22)$$

2.1.8 Schémas BDF (Backward-Difference Formulas)

- ordre 1

$$\mathbf{y}^{[n+1]} = \mathbf{y}^{[n]} + h\mathbf{f}^{[n+1]} \quad (2.23)$$

- ordre 2

$$\mathbf{y}^{[n+1]} = \frac{4}{3}\mathbf{y}^{[n]} - \frac{1}{3}\mathbf{y}^{[n-1]} + \frac{2h}{3}\mathbf{f}^{[n+1]} \quad (2.24)$$

- ordre 3

$$\mathbf{y}^{[n+1]} = \frac{1}{11} \left(18\mathbf{y}^{[n]} - 9\mathbf{y}^{[n-1]} + 2\mathbf{y}^{[n-2]} + 6h\mathbf{f}^{[n+1]} \right) \quad (2.25)$$

- ordre 4

$$\mathbf{y}^{[n+1]} = \frac{1}{25} \left(48\mathbf{y}^{[n]} - 36\mathbf{y}^{[n-1]} + 16\mathbf{y}^{[n-2]} - 3\mathbf{y}^{[n-3]} + 12h\mathbf{f}^{[n+1]} \right) \quad (2.26)$$

2.1.9 Schéma de Milne (ordre 4)

$$\mathbf{y}^{[n+1]} = \mathbf{y}^{[n-3]} + \frac{4h}{3} \left(2\mathbf{f}^{[n]} - \mathbf{f}^{[n-1]} + 2\mathbf{f}^{[n-2]} \right) \quad (2.27)$$

2.1.10 Schémas de Milne-Simpson

- ordre 1

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

- ordre 2

$$\mathbf{y}^{[n+1]} = \mathbf{y}^{[n-1]} + 2h\mathbf{f}^{[n]} \quad (2.29)$$

- ordre 4

$$\mathbf{y}^{[n+1]} = \mathbf{y}^{[n-1]} + \frac{h}{3} \left(\mathbf{f}^{[n+1]} + 4\mathbf{f}^{[n]} + \mathbf{f}^{[n-1]} \right) \quad (2.30)$$

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 $n^{\text{ième}}$ 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 4.2 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]=redTGA(f,a,b,yo,N)` : schéma de la tangente améliorée (fichier `redTGA.m`).
 - `[t,Y]=redRK3(f,a,b,yo,N)` : schéma de Runge et Kutta d'ordre 3 (version 2) (2.10) (fichier `redRK3.m`).
 - `[t,Y]=redRK4(f,a,b,yo,N)` : schéma de Runge et Kutta d'ordre 4 (version 1) (2.13) (fichier `redRK4.m`).
 - `[t,Y]=redAB4(f,a,b,yo,N)` : schéma d'Adams-Bashforth d'ordre 4 (fichier `redAB4.m`).
 - `[t,Y]=redPC3(f,a,b,yo,N)` : schéma de type prédiction-correction utilisant les schémas d'Adams-Bashforth d'ordre 3 et BDF d'ordre 3 (2.25) (fichier `redPC3.m`).

Ici les paramètres \mathbf{f} , \mathbf{a} , \mathbf{b} , \mathbf{y}_0 correspondent respectivement aux \mathbf{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 $\mathbf{y}^{[n]}$, $n \in \{0, \dots, \mathbf{N}\}$ et \mathbf{t} est le tableau contenant les $n+1$ nombres t^n , $n \in \{0, \dots, \mathbf{N}\}$

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 $(t^n, \text{abs}(\mathbf{y}^{[n]} - \mathbf{y}(t^n)))$, $n \in \{0, \dots, \mathbf{N}\}$.*

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 sous la forme de la figure 2 (à reproduire).*

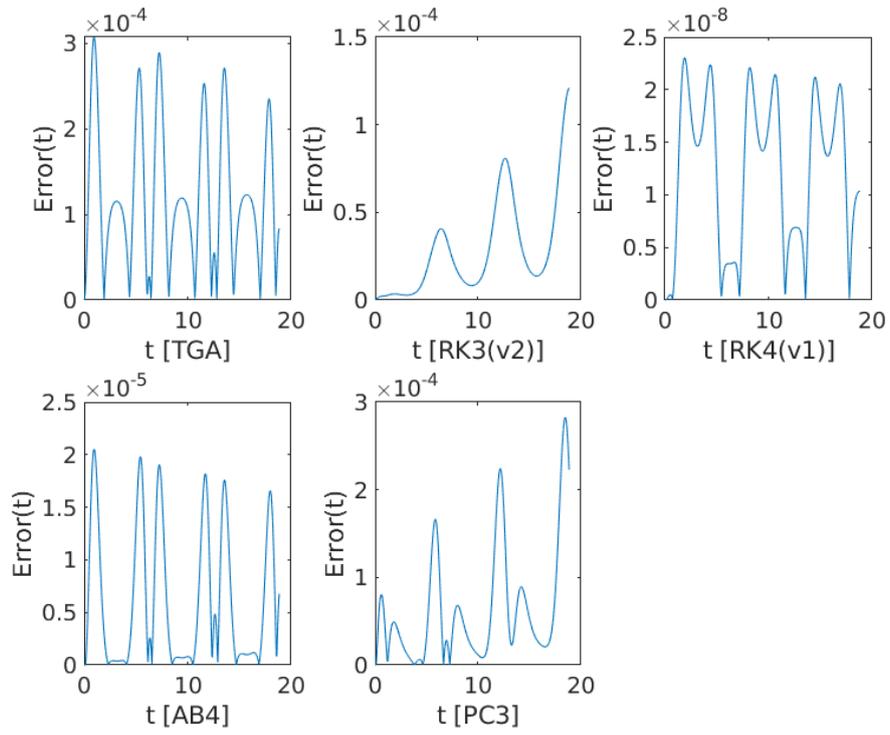


FIGURE 1 – Valeurs absolues des erreurs des 5 schémas représentées dans une unique figure

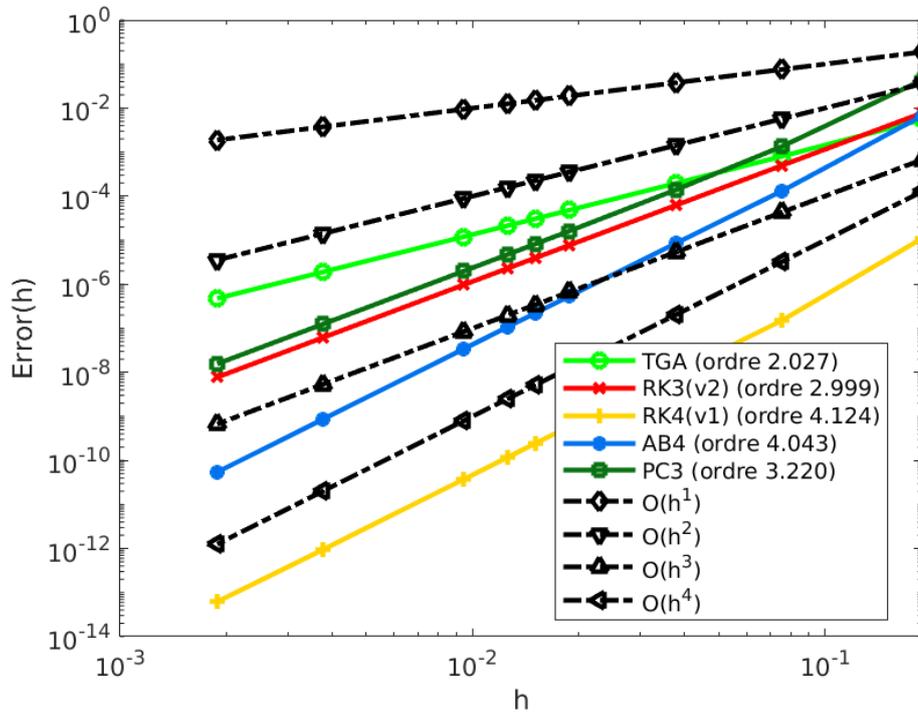


FIGURE 2 – Ordre des 5 schémas

A faire en 3h00 (temps indicatif)

- ◇ Créer une archive compressée nommée $\langle \text{NOM} \rangle$ -TP1-Q4 contenant les fichiers `redTGA.m`, `redRK3.m`, `redPC3.m`, `redRK4.m`, `redAB4.m`, `erreur.m`, `ordre.m` et toutes autres fonctions nécessaires à l'exécution des programmes `erreur.m` et `ordre.m`. Ici $\langle \text{NOM} \rangle$ correspond évidemment à votre nom.
- ◇ Envoyer un mail à `cuvelier@math.univ-paris13.fr` ayant pour **sujet** " $\langle \text{NOM} \rangle$ TP1 Q4" et en fichier joint l'archive compressée créée précédemment.

3 Le pendule pesant

3.1 Position du problème et équations différentielles

On considère un pendule de masse M , fixé à une tige rigide de longueur L et de masse négligeable, dans un milieu visqueux dont le coefficient de viscosité vaut k . On note θ l'angle formé par le pendule et l'axe verticale : il vérifie l'équation différentielle suivante (principe fondamental de la dynamique) :

$$\theta''(t) = -\frac{g}{L} \sin(\theta(t)) - \frac{k}{ML^2} \theta'(t), \quad \forall t \geq 0 \quad (3.1)$$

avec les conditions initiales

$$\theta(0) = \theta_0 \text{ et } \theta'(0) = \theta'_0. \quad (3.2)$$

θ_0 est l'angle initial en radian et θ'_0 la vitesse angulaire initiale en radian/seconde.

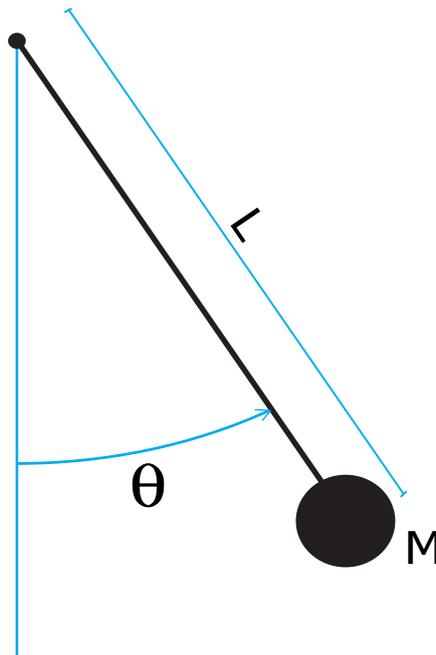


FIGURE 3 – Pendule pesant

On peut prendre, par exemple, $M = 1\text{kg}$, $L = 1\text{m}$, $g = 9.8\text{m.s}^{-2}$ et $k = 0.5\text{USI}$.

L'E.D.O. (3.1-3.2) ne peut être résolue de manière exacte. On se propose d'utiliser la méthode de Runge-Kutta d'ordre 4 pour l'étude des différents types de mouvements possibles, suivant les conditions de l'expérience (dans le vide $k = 0$, dans l'air $k = 0.1$, dans l'eau $k = 0.5, \dots$) et les conditions initiales imposées (pendule lancé, lâché, ...). On souhaite ensuite tracer les deux courbes discrètes $\theta(t)$ et $\theta'(t)$.

3.2 Résolution numérique

- Q. 5 (Matlab)**
1. Ecrire la fonction Matlab `fCauchy` (fichier `fCauchy.m`) correspondant à la fonction f du problème de Cauchy associé à (3.1-3.2). (On pourra utiliser des variables globales pour les paramètres physiques M , k , L et g . Voir l'aide sur `global` de Matlab. Toutefois il y a mieux en utilisant les fonctions anonymes générées avec `@`)
 2. Reprendre le fonction `redRK4.m` (déjà écrite dans le cas 1d) et la modifier si besoin pour l'adapter à la résolution de problèmes de Cauchy correspondant à des systèmes de d E.D.O. du premier ordre, $d > 1$.

3. Ecrire le programme `prg4` (fichier `prg4.m`) permettant de représenter la position et la vitesse du pendule au cours du temps.
4. En s'aidant du programme `PenduleMovie.m` et de la fonction `PlotPendule.m` (voir annexe 4.1), écrire le programme `PenduleVideo.m` permettant de réaliser une vidéo représentant le pendule en mouvement au cours du temps.

A faire en 2h00 (temps indicatif)

- ◊ Créer une archive compressée nommée `<NOM>-TP1-Q5` contenant les fichiers `fCauchy.m`, `prg4.m`, `redrk4.m` et `PenduleMovie.m`, ainsi que tout autre code nécessaire à l'exécution de `prg4` et `PenduleMovie`. Ici `<NOM>` correspond évidemment à votre nom.
- ◊ Envoyer un mail à `cuvelier@math.univ-paris13.fr` ayant pour **objet** "`<NOM> TP1 Q5`" et en fichier joint l'archive compressée créée précédemment.

3.3 Portrait de phase du pendule

Le portrait de phase du pendule (M , k et L fixé) est la représentation des courbes paramétrées $(x(t), y(t)) = (\theta(t), \theta'(t))$ calculée avec un grand nombre de données initiales (voir les figures 4 et 5)

- Q. 6 (Oral)** Soit le portrait de phase d'un pendule sans viscosité représenté en figure 4. A partir de celui-ci,
1. décrire le mouvement du pendule lorsque $(\theta(0); \theta'(0)) = A = (5; 0)$,
 2. décrire le mouvement du pendule lorsque $(\theta(0); \theta'(0)) = B = (\pi; 1)$,
 3. décrire le mouvement du pendule lorsque $(\theta(0); \theta'(0)) = C = (\pi; -1)$.
- Q. 7 (Oral)** Soit le portrait de phase d'un pendule avec viscosité représenté en figure 5. A partir de celui-ci,
1. décrire le mouvement du pendule lorsque $(\theta(0); \theta'(0)) = E = (0; 11)$,
 2. décrire le mouvement du pendule lorsque $(\theta(0); \theta'(0)) = F = (6\pi; -10)$.
- Q. 8 (Matlab)**
1. Ecrire le programme `PPSansViscosite.m` permettant de représenter le portrait de phase du pendule de paramètres $M = L = 1$ et $k = 0$ (figure 4).
 2. Ecrire le programme `PPAvecViscosite.m` permettant de représenter le portrait de phase du pendule de paramètres $M = L = 1$ et $k = 0.5$ (figure 5).

A faire en 1h00 (temps indicatif)

- ◊ Créer une archive compressée nommée `<NOM>-TP1-Q8` contenant les fichiers `PPSansViscosite.m` et `PPAvecViscosite.m`, ainsi que tout autre code nécessaire à leur exécution. Ici `<NOM>` correspond évidemment à votre nom.
- ◊ Envoyer un mail à `cuvelier@math.univ-paris13.fr` ayant pour **objet** "`<NOM> TP1 Q8`" et en fichier joint l'archive compressée créée précédemment.

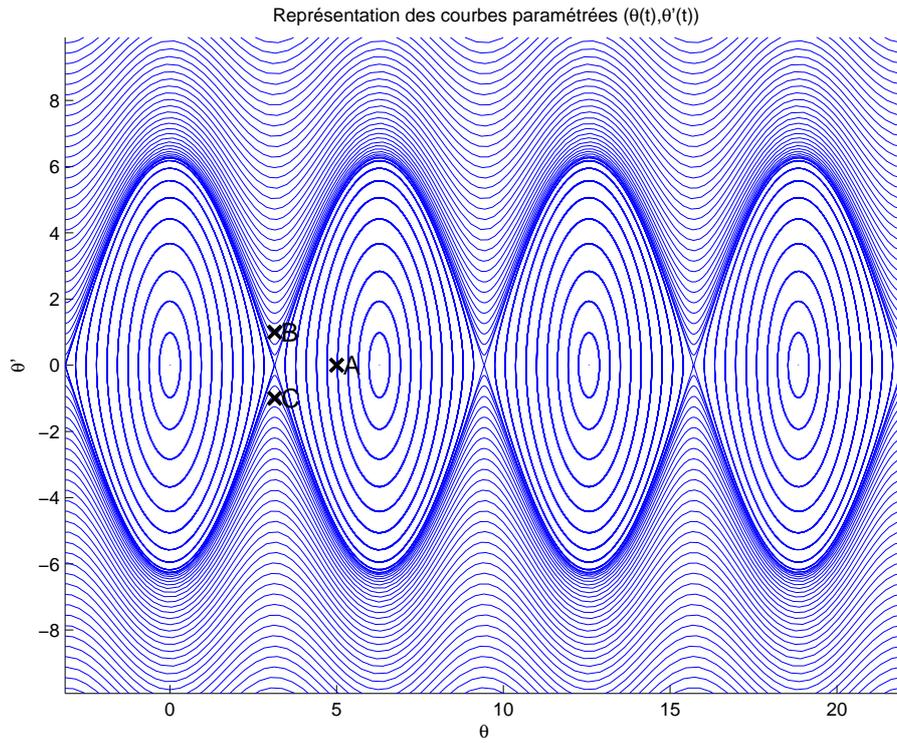


FIGURE 4 – Pendule pesant sans viscosité - Portrait de phase

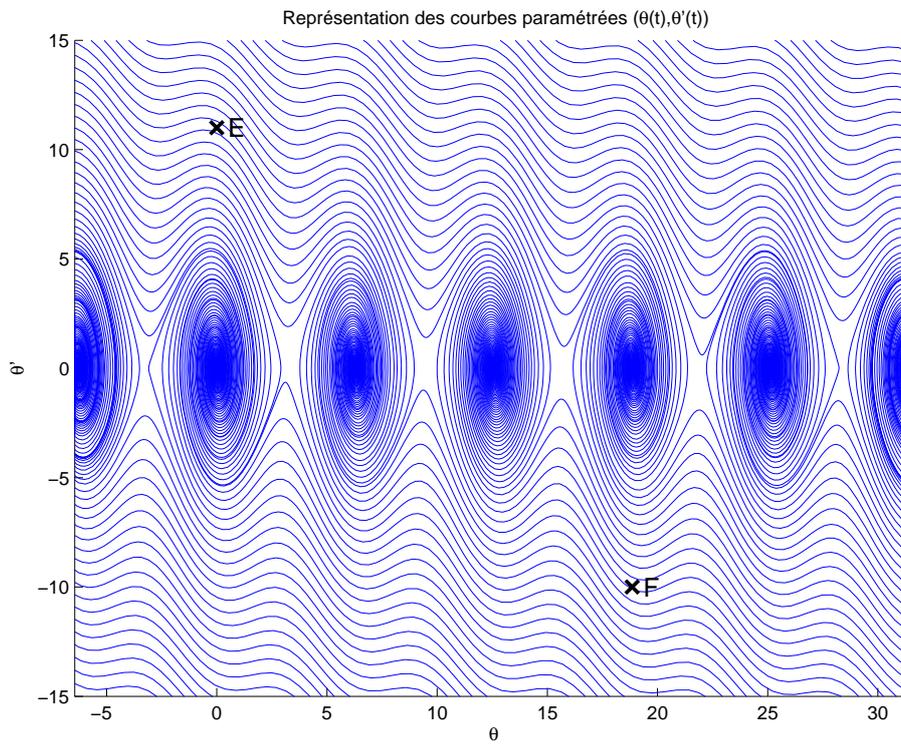


FIGURE 5 – Pendule pesant avec viscosité - Portrait de phase

4 Annexes

4.1 listings

Listing 1 – PenduleMovie.m

```
1 % Creation d'un fichier avi
2 % Teste avec version 2012b
3 % Utilise la fonction : PlotPendule.
4 clear all
5 close all
6
7 FrameRate=30;Step=10;
8 T=5;          % temps final
9 N=T*FrameRate*Step;
10 L=3;R=0.2; % Parametres graphiques du pendule
11
12 % pour simuler la rotation : physiquement faux (sauf sans viscosite et en apenseteur←
    !)
13 theta=@(t) pi*t;
14 t=0:T/N:T;
15
16 n=length(t);
17 figure(1);
18
19 VideoFile='VideoTest.avi';
20
21 %
22 writerObj = VideoWriter(VideoFile,'Motion JPEG AVI');
23 set(writerObj,'FrameRate',FrameRate);
24 open(writerObj);
25
26 for j=1:Step:n
27     cla
28     PlotPendule(L,R,theta(t(j)));
29     title(sprintf('\theta(%f)=%f',t(j),theta(t(j))))
30
31     axis on
32     drawnow
33     writeVideo(writerObj,getframe(gcf));
34 end
35
36 close(writerObj);
37 fprintf('Video file created : %s\n',VideoFile)
```

Listing 2 – PlotPendule.m

```
1 function PlotPendule(L,R,theta)
2 % Representation d'un pendule compose d'une tige de longueur L
3 % et d'une masse representee par un disque de rayon R.
4 % theta est la position angulaire du pendule.
5 hold on
6 % axe de rotation
7 plot(0,0,'ro')
8
9 % extremite de la tige
10 C=[L*cos(theta-pi/2);L*sin(theta-pi/2)];
```

```

11 plot([0, C(1)], [0, C(2)], 'r')
12
13 % Representation de la masse
14 t=0:pi/100:2*pi;
15 % Coordonnee du cercle
16 XC=C(1)+R*cos(t);
17 YC=C(2)+R*sin(t);
18 fill(XC, YC, 'r');
19 LL=L+2*R;
20 axis equal
21 axis([-LL LL -LL LL])
22 axis off
23 hold off

```

4.2 Quelques E.D.O. du premier ordre

4.2.1 Exemple 1

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

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

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

4.2.2 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 $y(t) = -\cos(t) + 1 + \beta$.

4.2.3 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 $y(t) = \exp(\cos(t))$.