Feuille 2: Méthodes d'Euler et point milieu

(1) Essayez sous Scilab les commandes suivantes, et expliquez leur rôle et leur syntaxe.

```
T= 0:0.1:1
T2=linspace(1,4,20)
X=0:0.1:2*%pi
sin(X)
```

(2) **Concaténation** C'est l'opération qui consiste à fabriquer des matrices à partir de matrices déjà consistuées. La syntaxe sous Scilab est simple pour ce genre d'opération. Essayez les commandes suivantes:

```
A=[1,2;6,7]  ,  B=[3,4,5;8,9,10]
C=[11,12]    ,  D=[13,14,15]
AA=[A,B;C,D] ,  BB=[B,A;C,D]
CC=[A,D;C,B]
T1=[T,1.1]   ,  T2=[T1;T1]
[X, sin(X)]  ,  [X ; sin(X)]
Y=[]  ,   Z= [ Y, 1] ,   Z=[Z,2,3],   Z=[Z,4,5]
Z=[Z;Z]
```

(3) **Représentation graphique avec Scilab**

Vérifiez que T et X sont toujours bien définis comme ci-dessus et essayez les commandes suivantes:

```
plot2d(X,cos(X))
clf()
plot2d(T',[exp(T)',exp(2*T)'])
clf()
plot2d(T',[exp(T)',exp(2*T)],2,-1)
plot2d(X,cos(X),-3,rect=[0,-2,2*%pi,2])
plot2d(X',[sin(X)', sin(2*X)', sin(3*X)'],[1,2,3],...
leg="sin(x)@sin(2x)@sin(3x)",rect=[0,-2,2*%pi,2])
```

Pour les options d'affichage, se réferer à la documentation distribuée.

Résolution numérique des EDO: méthode d'Euler

On cherche à résoudre l'EDO suivante:

$$\begin{cases} y' = f(y) \\ y(0) = y_0 \end{cases}$$

Donner la solution exacte de ce problème dans le cas où $f(x) = -x$.

Pour résoudre ce problème numériquement, du temps $t = 0$ à $t = 1$, on va utiliser la méthode suivante: On décompose l'intervalle $[0, 1]$ en n intervalles $[k/n, (k+1)/n]$ de taille $\Delta t = 1/n$. On construit une solution approchée du problème dont on donne uniquement la valeur y_k aux temps k/n. y_0 est donné par la condition initiale, et les valeurs de y_k par récurrence. En effet connaissant la valeur approchée au temp k/n, y_k, on peut calculer une valeur approchée au temps $(k+1)/n$, y_{k+1} comme ci-dessous:

$$\begin{aligned} y_{k+1} &= y_k + \int_{\frac{k}{n}}^{\frac{k+1}{n}} f(y(t))\,dt \\ y_{k+1} &\approx y_k + \int_{\frac{k}{n}}^{\frac{k+1}{n}} f(y_k)\,dt \\ y_{k+1} &\approx y_k + f(y_k)\Delta t \end{aligned} \tag{1}$$

C'est la **méthode d' Euler explicite**.

Attention: A partir de maintenant, on va conserver le code que l'on va écrire dans des fonctions réutilisables stockées dans un fichier `tp2.sci`. Pour cela on rappelle qu'on peut utiliser l'éditeur de Scilab ou un autre, et la commande `execnom_de_fichier` ou `getf` ou le bouton correspondant de l'éditeur pour charger ces commandes dans Scilab.

La fonction `euler_ex` calcule une solution approchée de $y' = -y$, entre les temps 0 et 1, une fois donnée la condition initiale y_0 et le nombre d'intervalles (c-a-d l'inverse de Δt).

```
// euler_ex: fonction calculant la solution de y'=-y avec Euler
// Explicite donnee initiale y_0 et n pas de temps jusqu'a t=1.

function X=euler_ex(y_0,n)
  dt= 1/n
  y=y_0                // condition initiale
  X=[]                 // X vecteur vide a t=0
  for i=1:n
    y= y - dt*y        // calcul de y au temps t+dt
    X=[X,y]            // ajout de la valeur de y(t+dt) au grand vecteur X
  end
endfunction
```

Remarque: Ne jamais oublier de mettre des commentaires pour expliquer votre code. Il sera ainsi plus facile à réutiliser.

(5) Représenter graphiquement la solution approchée avec condition initiale 1 pour $n = 30$. Afficher sur le même graphique la solution exacte. Répéter cela pour plusieurs valeurs de n (entre 3 et 500, pas plus). Qu'observe-t-on?

(6) Construire une matrice B, 2×10 colonnes, qui contient sur la première ligne les indices n de 10 à 100 par pas de 10, et sur la seconde ligne la valeur de $(y_n - e^{-1})$ correspondante. Représenter graphiquement $y_n - e^{-1}$ en fonction de n, et $\ln(|y_n - e^{-1}|)$ en fonction de $\ln(n)$. Noter la pente de la courbe dans le dernier cas. Quel est le comportement de l'erreur en fonction de n?

(7) Définir avec Scilab la fonction f suivante:

$$f(x) = -x + \frac{x^3}{6}\,. \tag{2}$$

Construire une nouvelle fonction `euler_ex2` qui demande la condition initiale y_0, le pas de temps dt, et le temps final T_f, et qui renvoie un vecteur contenant une solution approchée de $y' = f(y)$

par la méthode d'Euler entre $t = 0$ et $t = T_f$. (**Attention:** on demande que cela fonctionne encore si on change la fonction f). Essayez la avec la fonction ci-dessus.

Cette méthode donne de bons résultats, mais seulement si le pas de temps utilisé est petit. Pour diminuer le temps de calcul, on cherche des méthode qui vont donner une bonne approximation avec des pas de temps plus grand. La suite du Tp est consacrée à l'étude de deux nouvelles méthodes.

La **méthode d'Euler implicite** consiste à utiliser l'approximation

$$y_{k+1} \approx y_k + f(y_{k+1})\Delta t \tag{3}$$

en lieu et place de (1). Cela ne permet pas toujours de calculer explicitement la valeur de y_{k+1}, mais dans le cas $f(x) = -x$, ceci est possible.

(7) Construire une fonction `euler_imp` similaire à la fonction `euler_ex`, mais utilisant la méthode d'Euler implicite. L'utiliser pour représenter sur le même graphique la solution exacte, et les solutions approchées par les méthodes explicite et implicite (pour le même pas de temps). Que remarque-t-on? Reprendre la question 6 pour cette nouvelle fonction.

Dans le cas d'une fonction f quelconque, on utilise dans (3) une approximation de y_{k+1} (meilleur que y_k), par exemple celle obtenue avec la méthode d'Euler explicite.

$$\begin{aligned} z_{k+1} &= y_k + f(y_k)\,dt \\ y_{k+1} &= y_k + f(z_{k+1})\,dt \end{aligned}$$

(8) Utiliser cette méthode pour programmer une fonction `euler_imp2 } similaire {\'a} la fonction {\verb e`
Essayer-la.

La **méthode du point milieu** consiste à utiliser l'approximation

$$y_{k+1} \approx y_k + f\left(\frac{y_{k+1} + y_k}{2}\right)\Delta t \tag{4}$$

en lieu et place de (1). Cela ne permet pas toujours de calculer explicitement la valeur de y_{k+1}, mais dans le cas $f(x) = x$, ceci est possible.

(9) Construire une fonction `euler_imp` similaire à la fonction `euler_ex`, mais utilisant la méthode d'Euler implicite. L'utiliser pour représenter sur le même graphique la solution exacte, et les solutions approchées par les méthodes explicite et implicite (pour le même pas de temps). Que remarque-t-on? Reprendre la question 6 pour cette nouvelle fonction.

Dans le cas d'une fonction f quelconque, on utilise dans (4) une approximation de $(y_{k+1} + y_k)/2$ (meilleur que y_k), par exemple celle obtenue avec la méthode d'Euler explicite.

$$\begin{aligned} z_{k+1} &= y_k + f(y_k)\,\frac{dt}{2} \\ y_{k+1} &= y_k + f(z_{k+1})\,dt \end{aligned}$$

(10) Utiliser cette méthode pour programmer une fonction `euler_mp2` similaire à la fonction `euler_mp2`. Essayer-la.

(11) **Comparaison entre les 3 méthodes:** dans le cas $f(x) = x$, comparer sur une même graphique ces trois méthodes avec la solution exacte. Quelle est la méthode qui approche le mieux la solution. Pouvez-vous expliquer cela?

(12) Reprendre cette comparaison dans le cadre d'une fonction f quelconque.