1. Méthode de Runge, première amélioration

On reprend le problème 3 de la série précédente, concernant l'oscillateur amorti.
On a vu que l'évolution est régie par l'équation différentielle :

$a(t) = -\frac{k}{m} \cdot x(t) - \frac{c}{m} \cdot V(t)$ et

$V'(t) = a(t)$ et

$x'(t) = V(t)$ avec $V(0) = 0$ [m/s] et $x(0) = 0.10$ [m]

La **méthode d'Euler** consiste à remplacer :

V'(t) par $\frac{V(t+dt)-V(t)}{dt}$ et x'(t) par $\frac{x(t+dt)-x(t)}{dt}$.

L'équation différentielle est remplacé par la méthode numérique :

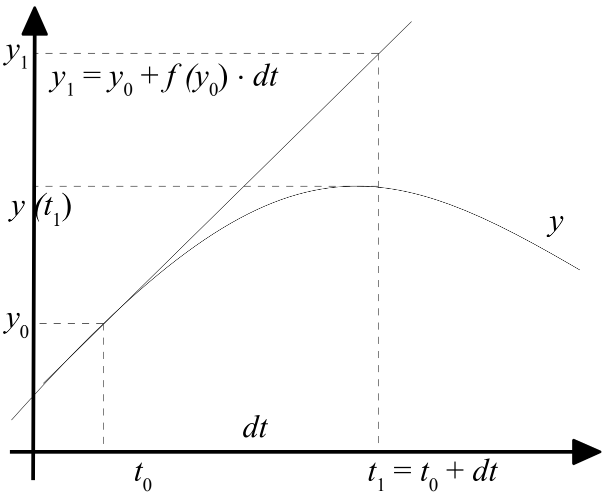
$a_t = (-k \cdot x(j \cdot dt) - c \cdot V(j \cdot dt)) / m$ et

$V((j+1) \cdot dt) = V(j \cdot dt) + dt \cdot a_t$ et

$x((j+1) \cdot dt) = x(j \cdot dt) + dt \cdot V(j \cdot dt)$, avec $j = 0, 1, 2, 3$, etc. ($t = j \cdot dt$)

La différence entre l'approximation numérique et la solution exacte est proportionnelle à la valeur numérique choisie pour *dt*.
On dit que la méthode d'Euler est **d'ordre 1.**

Illustration graphique de la méthode d'Euler pour résoudre : $\boldsymbol{y'(t) = f(y(t)),\quad y(t_0) = y_0}$

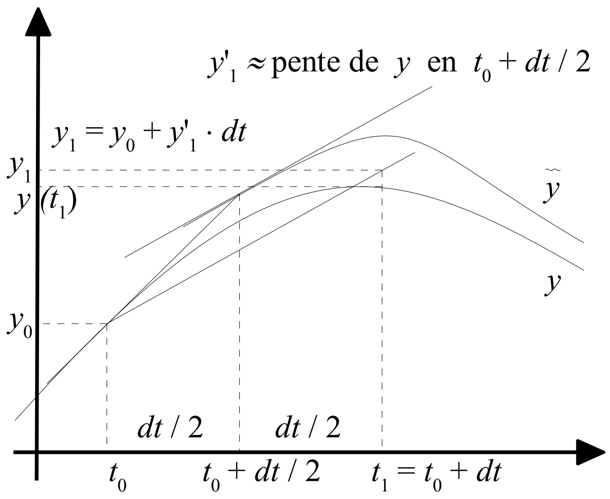


$y(t_1)$ est la solution exacte,
y_1 est l'approximation numérique obtenue par la méthode d'Euler.

On remarque que l'approximation est rapidement mauvaise, si *dt* n'est pas assez petit.

1. suite La **méthode de Runge** est une amélioration de la méthode d'Euler.

Illustration graphique de la méthode de Runge pour résoudre : $\boldsymbol{y'(t) = f(y(t)),\quad y(t_0) = y_0}$



y(t₁) est la solution exacte,
y_1 est l'approximation numérique obtenue par la méthode de Runge.

On remarque que l'approximation est bien meilleure que celle obtenue avec la méthode d'Euler.

La **méthode de Runge** est une amélioration de la méthode d'Euler, qui consiste à remplacer :

$V'\left(t+\frac{dt}{2}\right)$ par $\frac{V(t+dt)-V(t)}{dt}$ et $x'\left(t+\frac{dt}{2}\right)$ par $\frac{x(t+dt)-x(t)}{dt}$.

Le problème étant d'obtenir une bonne approximation de : $V'\left(t+\frac{dt}{2}\right)$ et de : $x'\left(t+\frac{dt}{2}\right)$.

Pour cela, on utilise : $V'\left(t+\frac{dt}{2}\right) = -\frac{k}{m}\cdot x\left(t+\frac{dt}{2}\right)-\frac{c}{m}\cdot V\left(t+\frac{dt}{2}\right)$ et $x'\left(t+\frac{dt}{2}\right) = V\left(t+\frac{dt}{2}\right)$

avec $x\left(t+\frac{dt}{2}\right)$ et $V\left(t+\frac{dt}{2}\right)$, les valeurs obtenues par la méthode d'Euler.

En résumé, on utilise la méthode d'Euler pour obtenir une première approximation, que l'on utilise dans un deuxième temps, pour obtenir une meilleure approximation.

<u>Cela donne concrètement :</u>

$a_0 = -\frac{k}{m}\cdot x(j\cdot dt)-\frac{c}{m}\cdot V(j\cdot dt)$; $V_{1/2} = V(j\cdot dt)+\frac{dt}{2}\cdot a_0$; (approxime $V\left(t+\frac{dt}{2}\right)$)

$x_{1/2} = x(j\cdot dt)+\frac{dt}{2}\cdot V(j\cdot dt)$; (approxime $x\left(t+\frac{dt}{2}\right)$)

$a_{1/2} = (-k\cdot x_{1/2}-c\cdot V_{1/2})/m$; $V((j+1)\cdot dt) = V(j\cdot dt)+dt\cdot a_{1/2}$;

$x((j+1)\cdot dt) = x(j\cdot dt)+dt\cdot V_{1/2}$, avec $j = 0, 1, 2, 3$, etc. ($t = j\cdot dt$)

Il y a deux fois plus de calculs, mais l'approximation est bien meilleure que celle d'Euler.
La différence entre l'approximation numérique et la solution exacte est proportionnelle au carré de la valeur numérique choisie pour *dt*. On dit que la méthode de Runge est **d'ordre 2.**
Lorsque *dt* est petit, son carré est beaucoup plus petit.

Écrivez une fonction ex3b(), qui utilise la méthode de Runge et remplace la fonction "ex3()" de l'exercice 3 de la série précédente. Continuez comme dans l'exercice 3 de la série précédente. Observez l'amélioration de la précision.

2. La méthode de Runge-Kutta

C'est une grosse amélioration de la méthode de Runge. Mais on ne peut plus l'illustrer graphiquement et il faut écrire l'équation différentielle de façon plus abstraite.

On veut résoudre numériquement une équation différentielle : $y'(t) = f(y(t)), \quad y(t_0) = y_0$

La **méthode de Runge-Kutta** décrite ci-dessous est d'ordre 4, c'est-à-dire que l'erreur est proportionnelle à dt^4.
Par exemple, un pas dt de 0,01 donne une erreur proportionnelle à 0,00000001.

Le schéma (un peu simplifié) de la **méthode de Runge-Kutta** est le suivant : (c.f. CRM page 97)

$$\boxed{\begin{array}{l} k_1 = f(y_0) \\ k_2 = f\left(y_0 + k_1 \cdot \frac{dt}{2}\right) \\ k_3 = f\left(y_0 + k_2 \cdot \frac{dt}{2}\right) \\ k_4 = f(y_0 + k_3 \cdot dt) \\ y_1 = y_0 + (k_1 + 2 \cdot k_2 + 2 \cdot k_3 + k_4) \cdot \frac{dt}{6} \end{array}}$$

Dans le cas de l'oscillateur amorti on a :

$y(t) = \begin{pmatrix} y_1(t) \\ y_2(t) \end{pmatrix} = \begin{pmatrix} x(t) \\ V(t) \end{pmatrix}$ est un vecteur et

$f(y) = \begin{pmatrix} y_2 \\ -\frac{k}{m} \cdot y_1 - \frac{c}{m} \cdot y_2 \end{pmatrix}$ est un vecteur égale à $y'(t) = \begin{pmatrix} y'_1(t) \\ y'_2(t) \end{pmatrix}$.

$y(0) = \begin{pmatrix} y_1(0) \\ y_2(0) \end{pmatrix} = \begin{pmatrix} 0.01\,[m] \\ 0\,[m/s] \end{pmatrix}$

Correspond au système suivant :

$V'(t) = -\frac{k}{m} \cdot x(t) - \frac{c}{m} \cdot V(t)$ et

$x'(t) = V(t)$ avec $V(0) = 0$ [m/s] et $x(0) = 0.10$ [m]

2.1 Écrivez une "function z=f(y)", qui pour le vecteur y donné, retourne le vecteur z défini par : z = f(y). La fonction *f* est celle du problème de l'exercice précédent.
Donc z(1) = y(2) et z(2) = ... *à compléter.*

2.2 Saurez-vous écrire une fonction ex3c() qui implémente la méthode de Runge-Kutta ?

3. La méthode générale de Scilab

Vu que la résolution numérique de système d'équations différentielles est fréquente, Scilab a implémenté une fonction générale de résolution de système d'équations différentielles. De plus la méthode de Scilab est meilleure que celle de Runge-Kutta, qui était déjà très bonne.

La fonction de Scilab s'appelle **ode(...)** qui est l'abréviation de "Ordinary Differential Equation"

Voici un exemple d'utilisation de la fonction **ode(...)**

```
function ex3d()
//============
  // Définition de l'équation différentielle :  [y(1) ; y(2)]' = [y(2) ; -k/m * y(1) -c/m * y(2)]
  function z=f(t, y)
  //================
  // t représente le temps  (non utilisé ici)
  // y(1)  représente la position
  // y(2)  représente la vitesse
  // z(1)  représente la dérivée de la position par rapport au temps
  // z(2)  représente la dérivée de la vitesse par rapport au temps
  m = 0.80;  // masse du corps
  k = 2.0;
  c = 0.30;
  z(1) = y(2);      // car  position' = vitesse
  z(2) = -k/m * y(1) -c/m * y(2);  // car  vitesse' = -k/m * position - c/m * vitesse
  endfunction

// données :
t0    =  0;   // temps initial
tFin  = 20;   // temps final
y0    = [0.1; 0];  // position = 0.1 [m] ; vitesse initiale = 0 [m/s]], vecteur COLONNE
N  = 201;      // nombre de points d'évaluation
at = linspace(t0, tFin, N);  // vecteur LIGNE

// résolution numérique en appelant la fonction  ode  de SciLab
ay = ode( y0,..     // condition initiale
          t0,..     // t_initial,
          at,..     // série des t(j) pour lesquels  y(j)  sera calculé
          1e-4,..   // tolérance d'erreur relative
          1e-5,..   // tolérance d'erreur absolue
          f..       // fonction définissant l'équation différentielle
          );
// L'algorithme essaye d'avoir une erreur inférieure au
// maximum entre l'erreur absolue et l'erreur relative.

// graphique de la solution numérique
scf(4);
clf();
plot(at, ay(1,:), 'g');  // 'g' = green, pour les positions
plot(at, ay(2,:), 'r');  // 'r' = red , pour les vitesses
endfunction
```

3.1 Écrivez le programme ci-dessus.
Testez-le.

3.2 Comment faudrait-il le modifier si la force de frottement était proportionnelle au carré de la vitesse ?

3.3 S'il y avait deux corps se déplaçant selon une seule direction, quelle serait la dimension du vecteur y dans la fonction f ?

3.4 S'il y avait deux corps se déplaçant selon trois dimensions, quelle serait la dimension du vecteur y dans la fonction f ?

3.5 S'il y avait trois corps se déplaçant selon trois dimensions, quelle serait la dimension du vecteur y dans la fonction f ?