Résolution numériques des équations différentielles - I

Objectifs :

- programmer les méthodes d'Euler, Runge-Kutta d'ordre 2 et Runge-Kutta d'ordre 4
- étudier expérimentalement l'ordre de ces trois méthodes

1 Implémentation des différents schémas d'intégration

On considère l'équation différentielle suivante :

$$\begin{cases} yy' = x \\ y(0) = 1 \end{cases} \tag{1}$$

1. Résoudre l'équation (1). Par la suite on notera g cette solution.
2. Programmer le schéma d'Euler explicite et l'appliquer à la résolution de l'équation (1) sur l'intervalle $[0, 0.3]$. Pour ce faire vous devrez créer trois fonctions matlab (ou scilab) :
   - une fonction `f(x,y)` admettant `x` et `y` comme paramètres et renvoyant la valeur de y'
   - une fonction `g(x)` admettant `x` comme paramètre et renvoyant $g(x)$
   - une fonction `euler_explicite(f,x0,y0,x1,n)` dont les paramètres sont ; `f` la fonction précédemment définie, `x0` et `x1` les bornes de l'intervalle de résolution, `y0` $= y(x_0)$ et `n+1` le nombre de points de discrétisation et qui renvoie deux vecteurs `x` et `y` de $n+1$ éléments contenant respectivement les $x_i = x_0 + \frac{x_1 - x_0}{n} i$ et les valeurs approchées de la solution, $y_i = y(x_i)$.

   Vous testerez votre code en comparant, sur un même graphique, la fonction $g(x)$ et la solution numérique obtenue.
3. Ecrire, en adaptant la fonction `euler_explicite(f,x0,y0,x1,n)`, deux nouvelles fonctions `RK2` et `RK4` correspondant aux schémas de Runge-Kutta d'ordre 2 et 4 (voir polycopié). Tracer, sur un même graphique, la fonction $g(x)$ ainsi que les solutions numériques obtenues pour chacune des trois méthodes programmées.

Indications 1. Vous complèterez les fonctions Scilab suivantes :

```
// Calcul de y' pour l'EDO y'=f(x,y)
// Paramètres d'entrée:
//    x valeur de la variable
//    y valeur de y(x)
// Valeur renvoyée:
//    f(x,y)
function [dy]=f(x,y)
  dy = ....;
endfunction

// Solution exacte de y'=f(x,y) avec y(x0)=y0
// Paramètre d'entrée:
//    x valeur de la variable
// Valeur renvoyée:
//    g(x) valeur de la solution exacte en x
function [sol]=g(x)
  sol = ....;
endfunction

// Schéma d'Euler explicite pour l'EDO y'=f(x,y)
// avec y(x0) = y0 sur l'intervalle [x0, x1]
// et pas de temps fixe h=(x1-x0)/n
// y(xn+1) = y(xn) + hf(xn, y(xn))
// Paramètres d'entrée:
//    f fonction f(x,y)
//    x0 point de départ
//    y0 condition initiale y0=y(x0)
//    x1 point d'arrivée
//    n nombre de points de discrétisation
// Valeurs renvoyées:
//    x vecteur contenant les points de
//      discrétisation xi
//    y vecteur contenant les valeurs de
//      la solution approchée yi=y(xi)
function [x,y]=euler_explicite(f,x0,y0,x1,n)
  // Initialisation des vecteurs x et y
  x=....;
  y=....;
  // Pas de temps
  h=....;
  // Calcul de y(xi)
```

```
  for i=1:n
    x(i+1)=....;
    y(i+1)=....;
  end
endfunction
```

2 Etude expérimentale de l'ordre des trois méthodes précédentes

On veut maintenant mettre en évidence l'ordre de convergence des trois schémas précédents. Pour cela, on va se placer dans un cas où l'on connait la solution exacte de l'équation différentielle.
Lors de la résolution numérique de l'équation différentielle :

$$y' = f(x, y) \tag{2}$$

les différents schémas numériques étudiés peuvent se mettre sous la forme :

$$y_{n+1} = y_n + \Phi(x_n, y_n, h)$$

La méthode associée à ce schéma est d'ordre au moins p si pour toute solution y de l'équation 2 on a :

$$e(h) = \max_n |y(x_{n+1}) - y(x_n) - \Phi(x_n, y(x_n), h)| = \mathcal{O}(h^{p+1}) \tag{3}$$

Pour différentes valeurs du pas de discrétisation $h = \frac{x_1 - x_0}{n}$, on calcule $e(h)$.

Appliquer cette procédure à l'exemple précédent, pour les trois schémas programmés pour $x \in [0, 0.3]$ en faisant varier n exponentiellement de 5 à 5.2^6 (n prendra successivement les valeurs 5, 5.2, 5.4,..., 5.2^6). Tracer $e(h)$ en fonction de h sur un graphique log-log. En déduire l'ordre de chacun des schémas en utilisant la fonction `polyfit`.