Université de Franche-Comté
Licence de Mathématiques, Semestre 6, 2018-19

DISCRÉTISATION DES EDPS
Quelques conseils généraux de programmation

Pour illustrer ces quelques principes, on s'appuiera sur une partie du TP1 :

on veut résoudre numériquement le problème de Cauchy

$$(\mathrm{P}) \quad \begin{cases} y'(t) = f(t, y(t)) \text{ pour } t \in (0, T) \\ y(0) = y_0 \end{cases}$$

en utilisant la méthode d'Euler explicite : $y_{n+1} = y_n + hf(t_n, y_n)$.

1 Programme principal, fonction

Lorsqu'on programme une tâche un peu complexe, on distingue le *Programme principal* (PP) des *fonctions*. La différence essentielle entre les deux est que le (PP) exécute la tâche principale désirée (résoudre (P)), tandis qu'une *fonction* éxécute une tâche élémentaire (par exemple discrétiser le segment $(0, T)$).
Ainsi,

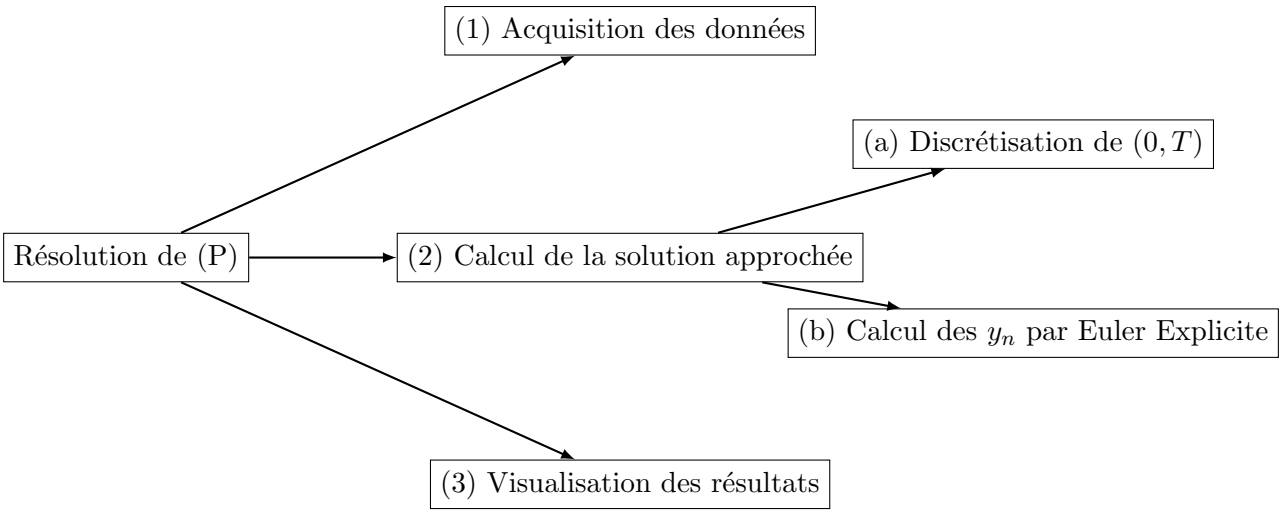
1. un (PP) utilisera des *fonctions* mais pas le contraire (une *fonction* peut utiliser d'autres *fonctions*) ;
2. une *fonction* pourra être utilisée par différents (PP) mais un (PP) ne peut pas être utilisé par un autre (PP) ou une *fonction* ;
3. une *fonction* a des variables d'entrée et des variables de sortie mais pas un (PP)

Le point 2. est essentiel à intégrer dans votre programmation :
une *fonction* doit pouvoir être réutilisée par d'autres (PP) **sans devoir être modifiée** à chaque fois. Les variables de la *fonction* sont là pour cela et doivent être choisies en conséquence.

SOUS SCILAB :
- le (PP) se met dans un fichier dont l'extension est " `.sce`" (par exemple `TP1-Etudes.sce`)
- les fonctions se mettent dans un fichier dont l'extension est " `.sci`" (par exemple `ApproximationsEDP.sci`).

2 Tâche principale, tâches secondaires, tâches élémentaires

On commence par décomposer la tâche principale en tâches secondaires puis chaque tâche secondaire encore trop complexe en tâches tertiaires et ainsi de suite jusqu'à obtenir des tâches élémentaires. Une tâche élémentaire étant réalisable par une seule *fonction*.
Pour la résolution de (P) par la méthode d'Euler explicite on obtient le schéma suivant :



3 Programmation

On commence par identifier les variables d'entrée et les variables de sortie qui seront nécessaires à chaque tâche élémentaire donnée par la décomposition précédente :

(1) *Acquisition des données*
Entrées : N entier, T réel, Y0 (vecteur) réel, f fonction
Sorties : aucune

(2) *Calcul de la solution approchée*

(a) *Discrétisation de* $(0, T)$
Entrées : N entier, T réel
Sorties : h réel (pas de la discrétisation)
tps : vecteur réel (temps de la discrétisation)

(b) *Calcul des* y_n *par Euler Explicite*
Entrées : N entier
h réel (pas de la discrétisation)
tps : vecteur réel (temps de la discrétisation)
f : fonction
Sortie : Yn : vecteur réel (solution approchée par Euler Explicite)

(3) *Visualisation des résultats*
Entrées : h réel (pas de la discrétisation)
tps : vecteur réel (temps de la discrétisation)
Yn : vecteur réel (solution approchée par Euler Explicite)
Sortie : aucune (graphique)

Conclusion :
Les tâches ne produisant aucune sortie seront mises directement dans le programme principal.
Les tâches produisant des sorties seront traitées par des fonctions.
Il y a ainsi 3 fonctions à écrire : la fonction f, la fonction servant à discrétiser $(0, T)$ et la fonction calculant la solution approchée par Euler Explicite.

4 Résolution de (P) par Euler Explicite : code Scilab possible

Programme principal :

```
////////////////////////////////////////////////////////////////////////
//                    Fichier TP1-Etudes.sce
////////////////////////////////////////////////////////////////////////
clear
exec('ApproximationsEDP.sci')
//
// Donnees du probleme
// -------------------
T0 = 0    // Temps initial
T = 2     // Temps final
N = 10    // Nb de sous-intervalles
Y0=1      // Donnee initiale
//
// Initialisations (pas toujours necessaire)
// ---------------
tps=zeros(5*N) // Temps de la discretisation
```

```
Yn=zeros(5*N)  // Solution approchee
//
// Discretisation de (T0,T)
// ------------------------
[h,tps] = maillage(T0,T,N)
//
// Rsolution (solutions approchees pour h=T/N)
// ----------
Yn=EulerExplicite(N,h,Y0,f1,tps)
//
// Reprsentation graphique
// -----------------------
plot(tps(1:N+1),Yn(1:(N+1)), tps(1:N+1),Yex )
title('Approximation par Euler implicite')
xlabel('Temps (t)')
ylabel('Solution y(t)')
legend(['Sol. approchee';'Sol. exacte'],[0,8])
```

Le fichier des fonctions :

```
////////////////////////////////////////////////////////////////////////
//                     Fichier ApproximationsEDP.sci
////////////////////////////////////////////////////////////////////////
//
// Terme source de l'EDO
function yp = f1(t,y)
    yp = y
endfunction
//---------------------------------
function [h,tps] = maillage(T0,TF,N)
// Discretise uniformement le segment [T0,TF] en N sous intervalles (ie N+1 noeuds)
    h = (TF-T0)/N  // le pas
    tps = T0:h:TF   // les noeuds
endfunction
//---------------------------------
function Y = EulerExplicite(N,h,Y0,f,tps)
// Euler explicite pour dy=f(t,y), Y0 donnee de Cauchy
    aux = Y0
    Y = [Y0]
    for i=2:N+1
        aux = aux + h*f(tps(i-1),aux)
        Y=[Y,aux]
    end
endfunction
```