CENTRE DE MATHÉMATIQUES
INSA DE LYON

Résolution d'une équation différentielle ordinaire avec MATLAB [1]

Soient $I = [t_0, t_1]$ un intervalle de $\mathbb{R}$ et a, b, f des applications de I dans $\mathbb{R}$ continues. Considérons l'équation différentielle ordinaire du second ordre :

$$(\mathcal{E}) \qquad y''(x) + a(x)y'(x) + b(x)y(x) = f(x) \qquad \forall x \in I$$

avec pour conditions initiales : $y(t_0) = \alpha_0$, $y'(t_0) = \alpha_1$.

1. Mise sous forme d'un système différentiel du premier ordre

MATLAB ne sait résoudre que des systèmes différentiels du premier ordre. Une étape préliminaire à la résolution par MATLAB de l'équation différentielle est donc la mise de celle-ci sous forme d'un système différentiel du premier ordre. Pour ce faire, désignons par y_1 la fonction inconnue y et par y_2 sa dérivée y'. On a bien entendu :

$$y_1'(x) = y_2(x) \qquad \text{pour tout } x \in I. \tag{1}$$

Par ailleurs, l'équation $(\mathcal{E})$ s'écrit encore :

$$\begin{aligned} y_2'(x) \;=\; y''(x) &= f(x) - a(x)y'(x) - b(x)y(x) \\ &= f(x) - a(x)y_2(x) - b(x)y_1(x). \end{aligned} \tag{2}$$

Introduisons la fonction inconnue $Y : x \in I \longmapsto \begin{pmatrix} y_1(x) \\ y_2(x) \end{pmatrix} \in \mathbb{R}^2$. Compte-tenu des équations (1) et (2) on a

$$Y'(x) = \begin{pmatrix} y_1'(x) \\ y_2'(x) \end{pmatrix} = \begin{pmatrix} y_2(x) \\ f(x) - a(x)y_2(x) - b(x)y_1(x) \end{pmatrix} = F(x, Y(x)) \tag{3}$$

où F est la fonction de $\mathbb{R}^3$ dans $\mathbb{R}^2$ définie par

$$F(x, y_1, y_2) = \begin{pmatrix} y_2 \\ f(x) - a(x)y_2 - b(x)y_1 \end{pmatrix}.$$

Ainsi,

$$Y'(x) = \begin{pmatrix} F_1(x, y_1, y_2) \\ F_2(x, y_1, y_2) \end{pmatrix}$$

avec $F_1 : (x, y_1, y_2) \longmapsto y_2$ et $F_2 : (x, y_1, y_2) \longmapsto f(x) - a(x)y_2 - b(x)y_1$.

[1] Note rédigée par S. Balac, Centre de Mathématiques Bat. Léonard de Vinci, INSA de Lyon F-69621 Villeurbanne cedex.

De la même manière, une équation différentielle d'ordre 3 peut s'écrire sous forme d'un système différentiel du premier ordre de dimension 3 et plus généralement, toute équation différentielle d'ordre N est équivalente à un système différentiel du premier ordre de dimension N. Voir par exemple :
[1] Demailly J.P, Analyse numérique et équations différentielles, PUG, 1996, page 134,
[2] Crouzeix M., Mignot A.L, Analyse numérique des équations différentielles, Masson, 1984, page 59.

2. Créer la fonction MATLAB décrivant le système différentiel

La seconde étape consiste à écrire une fonction MATLAB décrivant le système différentiel. La fonction doit être de la forme

```
function dY = edofct(x,Y)
```

où `edofct` est le nom (arbitraire) de la fonction MATLAB codant la fonction mathématique F. Le résultat `dY` est un vecteur colonne contenant les composantes F_1 et F_2 de la fonction F.

REMARQUE : Même si F ne dépend pas explicitement de x, la présence des deux paramètres d'entrée de la fonction est obligatoire.

Dans le cas de notre exemple, on définit cette fonction de la manière suivante.

EXEMPLE

```
function dY=linedo(x,Y)
% definition de l'edo d'ordre n comme systeme du 1er ordre de la forme :
% Y' = F(x,Y(x))
% F tant une application de R^{n+1} dans R^n
dY(1,:) = Y(2);
dY(2,:) = f(x)-a(x)*Y(2)-b(x)*Y(1);
```

Les fonctions a, b et f sont définies dans des fichiers séparés.

```
function val=a(x)
val = 1;

function val=b(x)
val = 1;

function val=f(x)
val = sin(x);
```

3. Résolution du système différentiel

La troisième étape consiste à choisir le solveur MATLAB devant résoudre le problème. Il existe deux types de solveur : les solveurs pour problèmes classiques et ceux pour problèmes dits « raides ». Les solveurs pour problèmes classiques sont les suivants :

ode45 utilise une méthode de Runge-Kutta explicite à un pas[2]. C'est le solveur à utiliser en premier choix pour la plupart des problèmes.

ode23 utilise également une méthode de Runge-Kutta explicite à un pas. Il peut être plus efficace que `ode45` dans certains cas.

[2] Voir [1] p. 221 ou [2] p. 107.

ode113 utilise une méthode de Adams-Bashforth-Moulton[3]. C'est un solveur multi-pas.

Les solveurs pour problèmes « raides » sont au nombre de quatre : `ode15s`, `ode23s`, `ode23t` et `ode23tb`.
La syntaxe d'appel du solveur est :

```
[t,Y] = odexx(@edofct,[t0 t1],Y0)
```

où
- `odexx` désigne l'un des solveurs listés ci-dessus ;
- `edofct` est le nom de la fonction MATLAB décrivant la fonction F associée au système différentiel (voir la partie précédente) ;
- `[t0 t1]` est l'intervalle de temps sur lequel on cherche à calculer la solution ;
- `Y0` est le vecteur colonne contenant les données initiales : $Y_0 = (y(t_0), y'(t_0))$.

Les paramètres de sortie du solveur sont :
- `t` : un vecteur colonne contenant les noeuds de l'intervalle $[t_0, t_1]$ où a été calculée la solution ;
- `Y` : une matrice contenant les valeurs de la solution et de ses dérivées aux noeuds de l'intervalle $[t_0, t_1]$ spécifiées dans le vecteur `t`. La i^{e} ligne de `Y` contient les valeurs de la dérivée $i-1^{\text{e}}$ de la solution aux noeuds de l'intervalle $[t_0, t_1]$. La première ligne contient la solution y de l'equation différentielle de départ $(\mathcal{E})$.

REMARQUE : Noter la présence du caractère `@` devant `edofct` dans les paramètres de la fonction.

[3] Voir [1] p. 244 ou [2] p. 120.

```
Y0=[0; 0]; %conditions initiales Y0=[y(t0),y'(t0)]
[t,Y] = ode45(@linedo,[0 20],Y0); %solveur
%La solution exacte
Ye=exp(-t/2).*sin(sqrt(3)*t/2)/sqrt(3)+ exp(-t/2).*cos(sqrt(3)*t/2)-cos(t);
%Représentation grapghique de la solution calculée par matlab
%et de la solution exacte
plot(t,Y(:,1),'*',t,Ye,'--')
```