UMMTO, FGEI, Département Automatique Année Universitaire : 2019−2020
Spécialité : Master 2Automatique, Options : AII
Module : *Commande Optimale*

Travaux pratiques

TP 1. Résolution analytique des équations aux dérivées ordinaires

L'équation différentielle ordinaire prend la forme générale suivante

$$F\left(y^{(n)},\, y^{(n-1)},\, \cdots,\, y(t),\, t\right) = 0 \tag{1}$$

Pour déterminer la solution de cette équation pour $t_0 \leq t \leq t_f$, on doit disposer de n conditions terminales définies aux instants t_0 et t_f. Sans spécifier les conditions terminales, on peut déterminer une solution générale, lorsqu'on impose les conditions terminales, on détermine une solution particulière. La solution de l'équation différentielle ordinaire peut être déterminée soit analytique ou numériquement tout dépend de la complexité de l'équation.

Dans ce TP, on s'intéresse à la résolution des équations différentielles ordinaires analytiquement. Ainsi, on suppose que l'équation différentielle admet une solution qu'on peut déterminer analytiquement. Pour résoudre analytiquement une équation différentielle avec Matlab, on utilise la fonction **dsolve**. Cette fonction admet comme arguments, l'équation différentielle à résoudre et les conditions terminales. L'équation différentielle doit être déclarée sous forme d'une chaine de caractères (entre ' ') et dont l'opérateur de différentiation est déclaré par la lettre D (Attention en majuscule) comme suit

$$y^{(n)}(t) \rightarrow \text{Dny(t)} \tag{2}$$

Par exemple pour déclarer l'équation différentielle suivante

$$y^{(2)}(t) + y(t) - t = 0 \tag{3}$$

on écrit

$$\text{eq} = \text{'D2y(t)+y(t)-t=0'} \tag{4}$$

Pour les conditions aux limites sont déclarées de la même manière séparées pars des virgules. Par exemple

$$y(0) = 0,\, y^{(1)}(0) = 1 \rightarrow \text{'y(0)=0, Dy(0)=1'} \tag{5}$$
$$y(0) = 0,\, y(1) = 0 \rightarrow \text{'y(0)=0, y(1)=1'} \tag{6}$$

Ainsi, pour résoudre l'équation différentielle ordinaire précédente avec les première conditions aux limites, on utilise les instructions suivantes (on suppose que la boite à outils du calcul symbolique Symbolic est installée) :

```
syms y t
eq = 'D2y(t)+y(t)-t=0'
ci = 'y(0)=0, Dy(0)=1'
y = dsolve(eq, ci)
```

Sans les conditions terminales, on obtient la solution générale (avec des constantes)

Problème 1 Considérez le problème à valeurs initiales

$$4\,y^{(2)}(t) + 12\,y^{(1)}(t) + 9\,y(t) = 0, \tag{7}$$
$$y(0) = 1, \tag{8}$$
$$y^{(1)}(0) = -4 \tag{9}$$

(a) Résolvez le problème et tracez sa solution pour $0 \leq t \leq 5$,

(b) Déterminez la racine de la solution,

(c) Déterminez les coordonnées $(t,\, y)$ du minimum,

(d) Changez la deuxième condition initiale en $y^{(1)}(0) = b$ et trouvez la solution en fonction de b.

Problème 2

(a) Déterminer, analytiquement, la solution générale du système d'équations différentielles ordinaires suivant :

$$\dot{x}(t) = x(t) + y(t) \tag{10}$$
$$\dot{y}(t) = 4\,x(t) + y(t) \tag{11}$$

(b) Déduire et représenter la solution particulière correspondante pour $x(0) = 1$ et $y(0) = 0$.

Problème 3 Considérons le problème à deux limites suivant :

$$y^{(2)}(t) + 4\,y^{(1)}(t) + 4\,y(t) = 0, \tag{12}$$
$$y(-1) = 2, \tag{13}$$
$$y^{(1)}(-1) = 1 \tag{14}$$

(a) Convertir cette équation différentielle ordinaire en un système d'équations différentielles ordinaires du premier ordre en utilisant un changement de variables.

(b) Préciser les conditions terminales.

(c) Résoudre le système d'équations différentielles obtenu

(d) Déduire et représenter la solution $y(t)$.

```
clc
clear

% PROBLEME 1
syms y t b

eq = '4*D2y(t)+12*Dy(t)+9*y(t)=0';
ci = 'y(0)=1,Dy(0)=-4';

% Question 1 :
% Solution analytique
y = dsolve(eq,ci)

% Représentation de la solution
T = 0:0.01:5;
yn = subs(y,t,T)
figure(1)
plot(T,yn)
grid

% Question 2
% Détermination de la racine y(t)=0
t0 = solve(y)

% Représentation de la solution et sa racine
figure(2)
plot(T,yn)
grid
hold on
plot(t0, subs(y,t,t0),'*r')
hold off

% Question 3
% Détermination du minimum de la solution y(t)
t_min = solve(diff(y,t))
% Représentation de la solution et son minimum
figure(3)
plot(T,yn)
grid
hold on
plot(t_min, subs(y,t,t_min),'*r')
hold off

% Question 4
% Changement de la deuxième condition
ci = 'y(0)=1,Dy(0)=b';
y = dsolve(eq,ci)
pause
```

```
% PROBLEME 2
clc
clear

syms x y t
eq1 = 'Dx(t) = x(t)+y(t)';
eq2 = 'Dy(t) = 4*x(t)+y(t)';
% Question 1
s = dsolve(eq1,eq2)
x = s.x
y = s.y

% Question 2
ci = 'x(0)=1, y(0)=0';
s = dsolve(eq1,eq2,ci)
x = s.x
sy = s.y

pause
% PROBLEME 3
clc
clear

syms x1 x2 t
eq1 = 'Dx1(t) = x2(t)';
eq2 = 'Dx2(t) = -4*x2(t)-4*x1(t)';
ci = 'x1(-1) = 2, x2(-1) = 1';
s = dsolve(eq1, eq2, ci);
x1 = s.x1
x2 = s.x2

% Solution y(t)
y = x1

% Représentation de la solution y
t = 0:0.01:5;
yn1 = subs(y);
figure(4)
plot(t, yn1)
grid

% Solution de l'équation différentielle
eq = 'D2y(t)+4*Dy(t)+4*y(t) = 0';
ci = 'y(-1) = 2, Dy(-1) = 1';
y = dsolve(eq, ci)
yn2 = subs(y)
```

```
figure(5)
plot(t, yn1, 'r', t, yn2, 'b*')
```

TP 2. Résolution analytique des conditions d'optimalité

Soit le problème de commande optimale formulé comme suit :

$$\min_{u(t)} J\left(u(t)\right) = \frac{1}{2}\int_0^2 u^2(t)\,dt \tag{15}$$

Sujet à :

$$\dot{x}_1(t) = x_2 \tag{16}$$
$$\dot{x}_2(t) = u(t) \tag{17}$$
$$x_1(0) = 1 \tag{18}$$
$$x_2(0) = 1 \tag{19}$$
$$x_1(2) = 0 \tag{20}$$
$$x_2(2) = 0 \tag{21}$$

1. On désire déterminer analytiquement la commande optimale $u^*(t)$ en utilisant les deux méthodes suivantes :

a. Calcul des variations,

b. Principe du minimum.

Partie 1. Calcul des variations

Pour résoudre le problème de commande optimale, on propose d'utiliser la méthode directe et la méthode de **Lagrange**. Ainsi, pour chaque méthode, on demande :

a. de donner les conditions d'optimalité,

b. d'écrire le programme qui permet :

- de résoudre analytiquement les conditions d'optimalité,
- de représenter les différentes trajectoires optimales, et
- d'évaluer la valeur du critère analytiquement et numériquement.

Partie 2. Principe du minimum

Pour résoudre le problème de commande optimale, on demande d'écrire le programme qui permet :

- de déterminer l'expression de la commande optimale et les conditions d'optimalité,
- de résoudre analytiquement les conditions d'optimalité,
- de représenter les différentes trajectoires optimales,
- et d'évaluer la valeur du critère analytiquement et numériquement.

2. Comparer les résultats obtenus par les différentes méthodes,

3. Comparer les différentes méthodes de point de vue programmation.

```
clc
clear

syms t

ode1 = 'Dp(t)=0';
ode2 = 'D2x2(t)+p(t)';
ode3 = 'Dx1(t)-x2(t)=0';
cl = 'x1(0)=1,x2(0)=1,x1(2)=0,x2(2)=0';
s = dsolve(ode1,ode2,ode3,cl);

x1 = s.x1
x2 = s.x2
u = diff(x2,t)

dt = 0.001;
T = 0:dt:2;

x1_num1 = subs(x1,t,T);
x2_num1 = subs(x2,t,T);
u_num1 = subs(u,t,T);

figure(1)
plot(T,x1_num1)
xlabel('t')
ylabel('x_1(t)')

figure(2)
plot(T,x2_num1)
xlabel('t')
ylabel('x_2(t)')

figure(3)
plot(T,u_num1)
xlabel('t')
ylabel('u(t)')

J1 = 1/2*int(u^2,t,0,2)
J_num1 = dt*trapz(1/2*u_num1.^2)


% clc
% clear
%
% syms t

ode1 = 'Dp1(t)=0';
ode2 = 'Dp2(t)+p1(t)=0';
```

```
ode3 = 'Dx1(t)-x2(t)=0';
ode4 = 'Dx2(t)-p2(t)=0';
cl = 'x1(0)=1,x2(0)=1,x1(2)=0,x2(2)=0';
s = dsolve(ode1,ode2,ode3,ode4,cl);

x1 = s.x1;
x2 = s.x2;
u  = s.p2;


dt = 0.001;
T = 0:dt:2;

x1_num2 = subs(x1,t,T);
x2_num2 = subs(x2,t,T);
u_num2 = subs(u,t,T);

figure(4)
plot(T,x1_num2)
xlabel('t')
ylabel('x_1(t)')

figure(5)
plot(T,x2_num2)
xlabel('t')
ylabel('x_2(t)')

figure(6)
plot(T,u_num2)
xlabel('t')
ylabel('u(t)')

J2 = 1/2*int(u^2,t,0,2)
J_num2 = dt*trapz(1/2*u_num2.^2)


figure(7)
plot(T,x1_num2,'r',T,x1_num1,'*b')
xlabel('t')
ylabel('x_1(t)')

figure(8)
plot(T,x2_num2,'r',T,x2_num1,'*b')
xlabel('t')
ylabel('x_2(t)')

figure(9)
plot(T,u_num2,'r',T,u_num1,'*b')
```

```
xlabel('t')
ylabel('u(t)')
```

TP 3. Résolution numérique des conditions d'optimalité

Résolution numérique d'un problème aux limites

Les conditions d'optimalité (équation d'Euler-Lagrange et équations d'Hamilton-Pontryagin) d'un problème de commande optimale sont des équations différentielles ordinaires munies de conditions aux limites. Le nombre de conditions aux limites dépend du nombre d'équations et de leurs ordres. Ces conditions sont spécifiées aux instants initial et final. Dans ce TP, on s'intéresse à la résolution numériques des problèmes aux limites (équations différentielles avec des conditions aux limites).

Matlab dispose de la fonction **bvp4c** qui permet de déterminer la solution d'un problème aux limites. Cette fonction nécessite la déclaration de l'équation différentielle à résoudre sous formes d'un système d'équations du premier ordre et des conditions aux limites. L'utilisation de cette fonction est illustrée dans le cas de l'exemple suivant :

Soit l'équation différentielle ordinaire

$$\ddot{x}(t) + |x(t)| = 0 \tag{22}$$

Pour les conditions aux limites, on considère les cas suivants :

(a) $x(-1) = 1.5$ et $x(1) = -2$ (Dirichlet),

(b) $x(-1) = -2$ et $\dot{x}(1) = 1.5$ (Dirichlet et Neumann),

(c) $x(-1) = -2$ et $x(1) + \dot{x}(1) = 1.5$ (Dirichlet et Robin).

En introduisant le changement de variables $x(t) = x_1(t)$ et $x_2(t) = \dot{x}(t)$, l'équation (22) peut s'écrire comme suit

$$\dot{x}_1(t) = x_2(t) \tag{23}$$
$$\dot{x}_2(t) = -|x_1(t)| \tag{24}$$

et les conditions aux limites s'écrivent

(a) $x_1(-1) = 1.5$ et $x_1(1) = -2$ (Dirichlet),

(b) $x_1(-1) = -2$ et $x_2(1) = 1.5$ (Dirichlet et Neumann),

(c) $x_1(-1) = -2$ et $x_1(1) + x_2(1) = 1.5$ (Dirichlet et Robin).

Pour résoudre ce problème aux limites, en utilisant la fonction **bvp4c**, considérons le premier cas de conditions aux limites. Dans le même dossier, on doit sauvegarder les programmes suivants :

equation.m

```
function dx = equation(t, x)

dx(1) = x(2);
dx(2) = -abs(x(1));

dx = dx(:);
```

conditionsLimitesCas_1.m

```
function y = conditionsLimitesCas_1(xa, xb)

y(1) = xa(1)-1.5;
y(2) = xb(1)+2;

y = y(:);
```

programme.m

```
clc
clear

% Domaine de la solution
t0 = -1;
dt = 0.001;
t_f = 1;

t = t0:dt:t_f;

% Estimées des solutions
sol_init = bvpinit(t,[1 0]);

% Résolution du problème aux limites
sol = bvp4c(@equation, @conditionsLimitesCas_1,sol_init);

% Récupération de la solution
t = sol.x;
x = sol.y;

% Affichage
% x_1(t)
figure(1)
plot(t,x(1,:));
xlabel('Temps')
ylabel('x(t)')

% x_2(t)
figure(2)
plot(t,x(2,:));
xlabel('Temps')
ylabel('dx(t)/dt')
```

conditionsLimitesCas_2.m

```
function y = conditionsLimitesCas_2(xa, xb)

y(1) = xa(1)+2;
y(2) = xb(2)-1.5;

y = y(:);
```

conditionsLimitesCas_3.m

```
function y = conditionsLimitesCas_3(xa, xb)

y(1) = xa(2) + 2;
y(2) = xb(1) + xb(2)-1.5

y = y(:);
```

Application à la résolution d'un problème de commande optimale

$$\min_{u(t)} J\left(u(t)\right) = \int_0^1 x^2(t) + u^2(t)\, dt \tag{25}$$

Sujet à :

$$\dot{x}(t) = -x(t) + u(t) \tag{26}$$

$$x(0) = 1 \tag{27}$$

$$x(1) = 0 \tag{28}$$

1. Résoudre numériquement le problème de commande optimale en utilisant

- Calcul des variations (méthode directe),
- Principe du minimum.

2. Reprendre le même problème avec les deux méthodes lorsque $x(1)$ est libre.