Méthodes numériques de résolution d'une ou d'un système d'équations différentielles

TD

Table des matières

Préambule :

Le présent TD a pour objectif de vous familiariser avec la méthode de résolution numérique utilisant le schéma Euler explicite et de découvrir d'autres méthodes. Vous disposez en annexe d'algorithmes typiques d'intégration d'équations différentielles.

1. Résolution numérique d'une équation différentielle d'ordre 1 : circuit RC :

Un condensateur de capacité C, préalablement déchargé, est alimenté par un générateur, de f.e.m. E et de résistance interne négligeable, à travers un élément résistif de valeur R.

La tension aux bornes de ce condensateur, notée $U_c(t)$ vérifie l'équation :

$$U_c(t) + R \cdot C \cdot \frac{dU_c(t)}{dt} = E$$

Avec : $E = 12\,V$; $R = 10\,k\Omega$; $C = 1000\,\mu F$.

Q1. Ecrire la mise en forme contextuelle du problème de Cauchy que l'on veut résoudre sur l'intervalle temporel $[0,60]$ en seconde.

Q2. En vous inspirant de l'exemple du cours : *Chute d'une bille d'acier dans une éprouvette remplie d'huile*, écrire sous python le script permettant de tracer l'évolution de la tension $U_c(t)$.

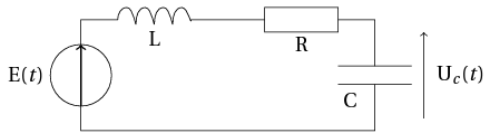
On prendra un incrément temporel de 0,1s.

Q3. Modifier le script en utilisant la fonction ***odeint*** du module ***integrate*** de la librairie *Scipy*.

Q4. Modifier le script afin d'afficher la réponse obtenue avec ***odeint*** lorsque l'on observe la décharge du condensateur en ayant initialement $U_c(0) = 12\,V$.

2. Résolution numérique d'une équation différentielle d'ordre 2 : circuit RLC :

On s'intéresse à la détermination de $U_c(t)$ dans un circuit de type « RLC série » tel que celui ci-dessous :



Pour lequel on peut établir aisément, grâce à la loi des mailles et aux caractéristiques des dipôles écrire :

$$E(t) = U_L(t) + U_R(t) + U_c(t)$$

$$E(t) = L \cdot C \cdot \frac{d^2U_c(t)}{dt^2} + R \cdot C \cdot \frac{dU_c(t)}{dt} + U_c(t) \quad (1)$$

On donne les valeurs des constantes : $E = 3\,V$, $R = 10\,\Omega$, $C = 1\,\mu F$, $L = 5\,mH$.

L'équation (1) peut se ramener en un système d'équations différentielles du premier ordre (voir exemple du cours : système masse-ressort) :

On introduit la variable $i(t) = C \cdot \frac{dU_c(t)}{dt}$

Q1. Transformer l'équation différentielle (1) en un système de 2 équations différentielles du premier ordre.

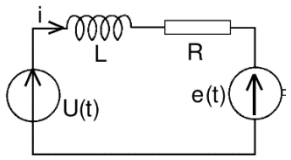
Q2. Ecrire la mise en forme contextuelle du problème de Cauchy que l'on veut résoudre sur l'intervalle temporel $[0; 5 \cdot 10^{-3}]$ en seconde. Les conditions initiales sont : $U_c(0) = 0$ et $i(0) = 0$

Q3. Toujours en vous inspirant de l'exemple du cours : *système masse-ressort*, écrire sous Python le script permettant de tracer l'évolution de la tension $U_c(t)$et celle du courant $i(t)$.

On prendra un nombre de points de calculs égal à $N = 5000$.

3. Système d'équations différentielles couplées d'ordre un. Moteur à courant continu.

Un moteur à courant continu est utilisé pour entrainer le bras rotatif d'un robot. Son alimentation en énergie électrique se fait par l'induit, qui peut être matérialisé par une résistance R en série avec une inductance L et une force contre-électromotrice $e(t)$.



La loi des mailles donne :

$$u(t) = R \cdot i(t) + L \cdot \frac{di(t)}{dt} + e(t)$$

D'un point de vue mécanique, le rotor du moteur reçoit le couple moteur $C_m(t)$, supporte le couple dû à la charge mécanique $C_r(t)$ et le couple $f \cdot \omega(t)$ dû aux frottements fluides.

On note J le moment d'inertie de l'ensemble des pièces entrainées.

Le principe fondamental de la dynamique donne :

$$J \cdot \frac{d\omega(t)}{dt} = C_m(t) - C_r(t) - f \cdot \omega(t) \quad -$$

Dans une machine à courant continu :

- La force électromotrice dépend de la vitesse de rotation : $e(t) = K_e \cdot \omega(t)$;
- Le couple $C_m(t)$ est lié au courant $i(t)$ qui traverse l'induit : $C_m(t) = K_c \cdot i(t)$

On traduit le problème en introduisant le vecteur d'état $Y(t) : t \to (i(t), \omega(t))$.

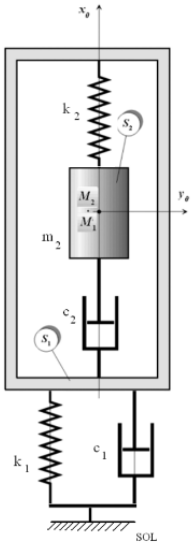
Q1. Ecrire la mise en forme contextuelle du problème de Cauchy que l'on veut résoudre sur l'intervalle temporel $[0; 3 \cdot 10^{-2}]$ en seconde. Les conditions initiales sont : $\omega(0) = 0$ et $i(0) = 0$

Q2. Compléter le script python : *mcc_maxpid_a_completer.py* afin d'afficher les évolutions temporelles de la vitesse angulaire et de l'intensité.

4. Système d'équations différentielles d'ordre 2 – Mass damper de Formule 1[1]

Pour limiter l'amplitude des vibrations verticales du train avant, et améliorer ainsi leur tenue de route, certaines Formules 1 furent équipées d'un système mécanique appelé mass damper, système aujourd'hui interdit.

Modélisation : un mass damper est constitué de deux solides S_1 et S_2 :



- S_1, le corps, est lié rigidement au châssis de la voiture ;
- S_2 de masse m_2, pouvant se déplacer verticalement par rapport au châssis du véhicule, est lié à S_1 par l'intermédiaire d'un ressort de raideur k_2 et d'un amortisseur visqueux de viscosité c_2.

Le châssis de la voiture limité à la partie avant de masse m_1, est lié au sol par un système d'amortisseurs et par les roues. Cet ensemble sera modélisé par un ressort de raideur k_1 et un amortisseur visqueux de viscosité c_2.

On considère deux points M_1 et M_2 appartenant respectivement à S_1 et S_2. On note $x_1(t)$ et $x_2(t)$ leurs coordonnées respectives dans un repère galiléen $(O;\overrightarrow{x_0},\overrightarrow{y_0},\overrightarrow{z_0})$.A l'état repos, M_1 et M_2 sont confondus. On a : $x_{1\,repos} = x_{2\,repos} = 0$

Par application du principe fondamental de la dynamique, on obtient les relations :

$$m_2 \cdot \frac{d^2x_2(t)}{dt^2} = c_2 \cdot \left(\frac{dx_1(t)}{dt} - \frac{dx_2(t)}{dt}\right) + k_2 \cdot \left(x_1(t) - x_2(t)\right) \quad (1)$$

$$m_1 \cdot \frac{d^2x_1(t)}{dt^2} = -c_1 \cdot \frac{dx_1(t)}{dt} - c_2 \cdot \left(\frac{dx_1(t)}{dt} - \frac{dx_2(t)}{dt}\right) - k_1 \cdot x_1(t) - k_2 \cdot \left(x_1(t) - x_2(t)\right) \quad (2)$$

On traduit le problème en introduisant le vecteur d'état $Y(t) : t \to \left(x_1(t), x_2(t), \frac{dx_1(t)}{dt}, \frac{dx_2(t)}{dt}\right)$.

Q1. Transformer les équations différentielles (1) et (2) en un système de 2 fois 2 équations différentielles du premier ordre.

[1] D'après C. Mathiotte

Q2. Ecrire la mise en forme contextuelle du problème de Cauchy que l'on veut résoudre sur l'intervalle temporel $[0; 1]$ en seconde. On suppose que la voiture sort d'un vibreur impliquant les conditions initiales :

$$x_1(0) = x_{10} = 0.05\,;\; x_2(0) = x_{20} = 0$$
$$\frac{dx_1(0)}{dt} = v_{10} = 0\,;\; \frac{dx_2(0)}{dt} = v_{20} = 0$$

Q3. Toujours en vous inspirant de l'exemple du cours : *système masse-ressort*, écrire sous Python le script permettant de tracer sur une même figure l'évolution des positions du châssis $x_1(t)$ et de la masse $x_2(t)$.

On prendra : un nombre de points de calculs égal à $N = 5000$, les différentes constantes sont définies dans le fichier de travail mass_damper.py

Q4. On ajoutera à la figure le tracé du déplacement du châssis dépourvu du système mass damper dont l'équation de mouvement est :

$$m_1 \cdot \frac{d^2x_s(t)}{dt^2} = -c_1 \cdot \frac{dx_s(t)}{dt} - k_1 \cdot x_s(t)$$

On choisira les mêmes conditions initiales que pour le système équipé de mass damper, à savoir :

$$x_s(0) = x_{s0} = 0.05\,;\; \frac{dx_s(0)}{dt} = v_{s0} = 0$$

Annexe

Algorithmes typiques d'intégration des équations différentielles

1. Algorithme typique schéma Euler explicite :

```
#---Importation des bibliothèques nécessaires :
numpy,matplotlib.pyplot, scipy.integrate

#------Initialisation------#
#définition des valeurs initiales
#nombre de points de calcul (N)
#temps initial et temps final (t0, tf)

#---Déclaration de la ou des fonctions différentielles à intégrer---#
def F(V,t) :
   #constantes de l'expérience
   #équation(s) différentielle(s)

   return résultat

#---Mise en œuvre d'Euler explicite---#
#---Déclaration de la fonction Euler explicite---#
def euler(F,t0,tf,N):
        # définition et initialisation de la liste ou des listes des ordonnées de la courbes (valeurs_Y)
        # définition et initialisation de la liste temporelle (valeurs_t)
        #initialisation de l'incrémentation de la liste ou des listes des ordonnées
        #initialisation de l'incrémentation de la liste temporelle
        dt=(tf-t0)/(N) # pas temporel d'intégration

        for i in range(N-1) :# création de deux listes de N valeurs , valeur initiale comprise
        …
        …
        return des deux listes : valeurs_t, valeurs_Y

#---Appel de la fonction Euler---#
valeurs_t,valeurs_V =euler(F,t0,tf,N)

#------Affichage------#
…
```

2. Algorithme typique utilisant la fonction ***odeint*** du module ***integrate*** de la librairie ***Scipy :***

#---Importation des bibliothèques nécessaires :
numpy,matplotlib.pyplot, scipy.integrate

#------Initialisation------#
#définition des valeurs initiales
#nombre de points de calcul (*N*)
#temps initial et temps final (*t0*, *tf*)

#---Déclaration de la ou des fonctions différentielles à intégrer---#
def F(V,t) :
  #constantes de l'expérience
  #équation(s) différentielle(s)
  return résultat

#---Mise en œuvre odeint---#
#création des valeurs temporelles *valeurs_t* de type *np.array* en utilisant *np.linspace*
#création des *valeurs_Y* avec *odeint()*

#------Affichage------#
…