8

Simulation d'un système en physique/chimie/SII

Mesures

t (s)	V (m/s)
0	0
0.2	0.7
0.4	1.7
0.6	2.3
0.8	2.6
1	2.8
1.2	3
1.4	3.1
1.6	3.1
1.8	3.1
2	3.1
2.2	3.1
2.4	3.1
2.6	3.2
2.8	3.2
3.1	3.1
3.5	3.2
3.9	3.2
4.3	3.1
4.7	3
5.1	3
5.5	3.1
5.9	3.1
6.3	3.1

⋄ On mesure sur un banc d'essai la vitesse d'une voiture radio-commandée, le tableau de mesures est donné ci-contre. On peut modéliser l'évolution de cette vitesse en fonction du temps par l'équation différentielle avec condition initiale :

$$\begin{cases} \tau \dfrac{\mathrm{d}v(t)}{\mathrm{d}t} + v(t) = K \cdot E_0 \\ v(0) = 0 \end{cases}$$

où τ, K et E_0 sont des constantes que l'on souhaite déterminer en utilisant les mesures effectuées ; τ est la constante de temps du système, E_0 est la tension qui alimente le moteur et K est le gain statique. Pour l'illustration, on prendra $\tau = 0.5$ s et $KE_0 = 5$ m/s.

⋄ But de la simulation :

- Pour des instants donnés $t_0, t_1, \ldots, t_N$, déterminer approximativement les valeurs de la vitesse v notées $v_0, \ldots, v_N$ à l'aide de l'équation différentielle ;
- On a $v_0 = 0$ et les autres valeurs $v_1, \ldots, v_N$ sont à calculer ;
- Pour pouvoir mener le calcul, il faut que t_{i+1} soit « proche » de t_i, c'est pourquoi on pose $t_i = i\Delta_t$ avec Δ_t fixé « assez petit » (c'est le pas de la simulation, on peut par exemple prendre ici $\Delta_t = 0.1$ s).

⋄ Avec PYTHON, v et t sont des listes : **t**$= [t_0, t_1, \ldots, t_N]$ et **v**$= [v_0, v_1, \ldots, v_N]$ avec $t[i] = i\Delta_t$ et **v**$[0] = 0$. On dispose également de deux listes **t_mes** et **v_mes** qui correspondent aux valeurs mesurées.

⚠ Bien comprendre les différentes notations :

- $v(t)$ est la vitesse à l'instant t donnée par l'équation différentielle ;
- $t_0, t_1, \ldots, t_{N-1}$ sont les instants où l'on va calculer numériquement cette vitesse ;
- **t[i]** représente, en PYTHON, ce que l'on note mathématiquement $t_i = i\Delta_t$;
- **v[i]** représente, en PYTHON, v_i (ou encore $v(t_i)$), donc la vitesse à l'instant t_i.

◇ Principe de l'approximation numérique. Comme $t_{i+1} - t_i$ est « petit » :

$$\frac{v(t_{i+1}) - v(t_i)}{t_{i+1} - t_i} \simeq v'(t_i) = \frac{1}{\tau}(KE_0 - v(t_i))$$

Traduction avec PYTHON : on *définit* $v[i+1]$ *de sorte que*

$$\frac{v[i+1] - v[i]}{t[i+1] - t[i]} = \frac{1}{\tau}(KE_0 - v_i)$$

On *calcule* donc $v[i+1]$ à partir de $v[i]$ en *posant* :

$$\forall i \in [\![0, N-1]\!],\ v[i+1] = v[i] + \frac{1}{\tau}(t[i+1] - t[i])(KE_0 - v[i])$$

Pour remplir la liste v, on répète cette instruction pour i compris entre 0 et $N-1$.

```
Delta_t = 0.1 # Paramètres de la simulation
N = 60        #
t = []                    # Construction des listes
v = []                    # t et v
for i in range(N+1):      #
    t.append(i*Delta_t) # Pour l'instant, tous les
    v.append(0)           # éléments de v sont nuls
tau = 0.5 # Valeurs numériques pour
KE0 = 5   # l'équation différentielle
for i in range(N): # 0<=i<=N-1
    v[i+1] = v[i]+1/tau*(t[i+1]-t[i])*(KE0-v[i])
```

◇ On représente graphiquement v en fonction de t pour les valeurs calculées. Sur le même graphique, on représente également les valeurs mesurées (enregistrées dans deux listes notées **t_mes** et **v_mes**) :

```
import matplotlib.pyplot as plt
plt.plot(t,v,'b-')
plt.plot(t_mes,v_mes,'ro')
```