19

Équations différentielles (ordre 1)

◇ Les équations différentielles considérées devront être écrites sous la forme :

$$y' = F(y,t)$$

où F est une fonction des deux variables y et t. Par exemple pour l'équation différentielle :

$$y' + 2ty = 1$$

on définit $F(y,t) = 1 - 2ty$ ce qui s'écrit avec PYTHON :

```
def F(y,t):
    return 1-2*t*y
```

⚠ *Remarque.* Même si, comme c'est parfois le cas, on écrit l'équation différentielle sous la forme :

$$y'(t) + 2ty(t) = 1$$

il ne faut *surtout pas* définir la fonction **F** en écrivant :

```
def F(y,t):
    return 1-2*t*y(t)  # INCORRECT !
```

car une telle écriture est produira à coup sûr des erreurs. □

I. Avec Numpy/Scipy

◇ On considère un *problème de Cauchy* (équation différentielle avec condition initiale) de la forme :

$$\begin{cases} \dfrac{dy}{dt} = F(y,t) \\ y(t_0) = y_0 \end{cases}$$

On veut calculer des valeurs approchées $y_0, y_1, \dots, y_n$ de la solution y au temps $t_0, t_1, \dots, t_n$. Avec PYTHON, on utilise la fonction **odeint** du package **scipy.integrate**.

```
from math import *
from scipy.integrate import odeint
import numpy as np
import matplotlib.pyplot as plt
```

On appelle **odeint** sous la forme **odeint(F,y0,t)** où F est la fonction qui intervient dans l'équation différentielle, y_0 est la valeur initiale et t est le vecteur contenant les valeurs $t_0, \ldots, t_n$.

Exemple Python. On considère l'équation différentielle d'ordre 1 avec condition initiale :

$$\begin{cases} \dfrac{\mathrm{d}y}{\mathrm{d}t} = -ty \\ y(0) = 1 \end{cases}$$

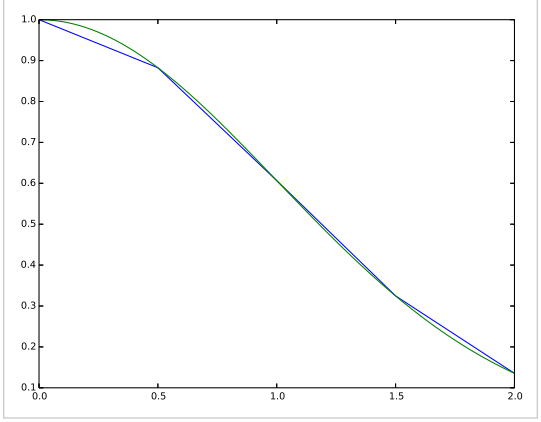
On veut calculer des valeurs approchées de y entre 0 et 2 pour 5 valeurs régulièrement espacées.

Remarque. Le tracé (on a superposé celui de la fonction $t \mapsto \exp(-t^2/2)$) :

```
plt.plot(t,y)
plt.show()
```

```
def F(y,t):
    return -t*y
t=np.linspace(0,2,5)
y=odeint(F,1,t)
print y
```

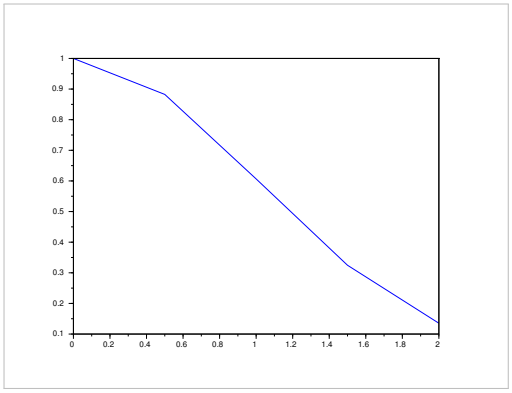
```
[[ 1. ] [ 0.88249698] [
0.60653074] [ 0.32465257] [
0.13533531]]
```



II. Avec Scilab

Exemple. On reprend le même problème de Cauchy. Avec SCILAB, on utilise la fonction **ode** que l'on appelle sous la forme **ode(y0,t0,t,F)**.

```
function ydot=F(t,y)
    ydot=-t*y
endfunction
t=linspace(0,2,5);
y=ode(1,0,t,F);
plot(t,y)
```



Exemples de résolution approchée d'éq. diff.

```
from math import *
from scipy.integrate import odeint
import numpy as np
import matplotlib.pyplot as plt
```

1 En mathématiques

Exemple. On veut résoudre sur l'intervalle $[0,5]$ le problème de Cauchy :

$$\begin{cases} y' + 2ty = 1 \\ y(0) = 1 \end{cases}$$

- La fonction F associée à l'équation différentielle :

```
def F(y,t):
    return 1-2*t*y
```

- Le vecteur t qui représente l'intervalle de temps considéré :

```
t=np.linspace(0,5,100)
```

- Le calcul du vecteur y avec la commande **odeint** :

```
y=odeint(F,1,t)
```

- Représentation graphique :

```
plt.clf()
plt.plot(t,y,'b-')
plt.xlabel("x")
plt.ylabel("y")
```

2 En chimie

Exemple. On considère la réaction :

$$Fe^{2+} + Co^{3+} = Fe^{3+} + Co^{2+}$$

C'est une réaction d'ordre 2 et l'évolution de la concentration en ions Fe^{2+}, notée C, est décrite par l'équation différentielle :

$$\frac{dC}{dt} = -kC^2$$

On donne $k = 80.2$ mol^{-1}L$\cdot$s^{-1} et $C_0 = 5.0\cdot10^{-4}$ mol$\cdot$L^{-1}. On donne ci-contre des mesures expérimentales de C en fonction du temps.
On donne :

t (s)	C (mol$\cdot$L^{-1})
20	$2.78\cdot10^{-4}$
40	$1.92\cdot10^{-4}$
60	$1.47\cdot10^{-4}$
80	$1.19\cdot10^{-4}$
100	$1.0\cdot10^{-4}$
120	$0.86\cdot10^{-4}$

```
t_mes=[20,40,60,80,100,120]
C_mes=[2.78e-4,1.92e-4,1.47e-4,1.19e-4,1.0e-4,0.86e-4]
```

- Définir les constantes de l'énoncé :

```
k=80.2
C0=5e-4
```

- La fonction F associée à l'équation différentielle :

```
def F(C,t):
    return -k*C**2
```

- Le vecteur t qui représente l'intervalle de temps considéré :

```
t=np.linspace(0,120,100)
```

- Le calcul du vecteur C avec la commande **`odeint`** :

```
C=odeint(F,C0,t)
```

- Représentation graphique :

```
plt.clf()
plt.plot(t,C,'b-')
plt.plot(t_mes,C_mes,'k+')
plt.xlabel("t (s)")
plt.ylabel("C (mol/L)")
```

3 En SII

Exemple. On alimente un moteur initialement à l'arrêt par une tension constante $U = 10$ V. La vitesse de rotation w du moteur est solution de l'équation différentielle :

$$J\frac{\mathrm{d}w}{\mathrm{d}t} + \left(\frac{K_e K_c}{R} + f\right) w(t) = \frac{K_c}{R} U \qquad \text{avec } w(0) = 0$$

où :

- J est l'inertie du moteur, $J = 5.5 \cdot 10^{-5}$ kg$\cdot$m^2 ;
- R est la résistance électrique du système, $R = 5.2\ \Omega$;
- K_c et K_e sont des constantes, $K_c = 0.24$ N$\cdot$m$\cdot$A^{-1} et $K_e = 0.24$ V$\cdot$rad$^{-1}\cdot$s ;
- f est un coefficient caractérisant les forces de frottement, $f = 0.05$ N$\cdot$m$\cdot$s$\cdot$rad^{-1}.

Déterminer et représenter l'évolution de la vitesse sur l'intervalle de temps $[0, 10^{-2}]$ (en secondes).

- Les constantes de l'énoncé :

```
U=10
J=5.5e-5
R=5.2
Kc=0.24
Ke=0.24
f=0.05
```

- La fonction F associée à l'équation différentielle :

```
def F(w,t):
    return (Kc/R*U-(Ke*Kc/R+f)*w)/J
```

- Le vecteur t qui représente l'intervalle de temps considéré :

```
t=np.linspace(0,1e-2,100)
```

- Le calcul du vecteur w avec la commande **odeint** :

```
w=odeint(F,0,t)
```

- Représentation graphique :

```
plt.clf()
plt.plot(t,w,'b-')
plt.xlabel("t (s)")
plt.ylabel("w (rad/s)")
```

4 En biologie

Exemple. On veut représenter l'évolution d'une population N modélisée par l'équation différentielle :

$$\frac{\mathrm{d}N}{\mathrm{d}t} = rN\left(1 - \frac{N}{N_{\max}}\right)$$

Représenter graphiquement les deux solutions obtenues pour $N(0) = N_{\max}/10$ puis $N(0) = N_{\max}/2$ sur l'intervalle $[0,10]$.

- Les constantes de l'énoncé :

```
r=0.5
Nmax=1000
```

- La fonction F associée à l'équation différentielle :

```
def F(N,t):
    return r*N*(1-float(N)/Nmax)
```

- Le vecteur t qui représente l'intervalle de temps considéré :

```
t=np.linspace(0,10,100)
```

- Le calcul des vecteurs qui correspondent aux deux conditions initiales :

```
N1=odeint(F,Nmax/10,t)
N2=odeint(F,Nmax/2,t)
```

- Représentation graphique :

```
plt.clf()
plt.plot(t,N1,'b-')
plt.plot(t,N2,'r-')
plt.xlabel("t")
plt.ylabel("N")
plt.legend(["N0=Nmax/10","N0=Nmax/2"],loc="lower right")
```