Chapitre de révision 5

Schémas et résolution d'E.D.P.

5.1 Schéma d'Euler-Cauchy explicite d'ordre 1

• La méthode d'Euler consiste à remplacer dans l'équation différentielle :

$$\boxed{f'(x) = \phi(x, f(x))}$$

le nombre dérivée $f'(x)$ en x, par : $\boxed{f'(x) \approx \dfrac{f(x+dx) - f(x)}{dx}}$.

Cette approximation donne le **schéma d'Euler explicite à l'ordre 1** :

$$\boxed{f(x+dx) \approx f(x) + dx \times f'(x)}$$

• Donnée une subdivision régulière $(x_k)_{0 \leqslant k \leqslant n}$ de l'intervalle $[a, b]$ d'intégration, dont les valeurs sont contenues dans le tableau :

```
X = np.linspace(a,b,n+1)
```

La solution approchée consistera en les valeurs prises par f aux points de (x_k),
contenues dans un tableau de longueur $n+1$:

```
Y = np.empty(n+1)
```

et complété, au sein d'une boucle `for` à l'aide du schéma d'Euler :

```
Y[0] = y0                                  C.I.
Y[k+1] = Y[k] + (b-a)/n*Phi(X[k],Y[k])     ∀ k ∈ [[0,n]]
```

5.1.1 Résolution numérique d'une E.D.P. d'ordre 1

• Le schéma d'Euler explicite à l'ordre 1 peut aussi être utilisé pour résoudre une **équation aux dérivées partielles** à l'ordre 1 (ce que ne permet pas la méthode d'Euler : il n'y a pas de méthode pour ramener une E.D.P. à une équation différentielle).

• Résolution approchée de l'équation aux dérivées partielles à l'ordre 1 :

$$F : [0,b] \times [0,T] \longrightarrow \mathbb{R} \quad \text{tel que}$$

$$\begin{cases} \boxed{\dfrac{\partial F}{\partial t}(x,t) = \dfrac{\partial F}{\partial x}(x,t)} & \forall\, x \in [0,b],\ \forall\, t \in [0,T] \\ F(x,0) = f(x) = x^2 & \forall\, x \in [0,b] \\ F(0,t) = g(t) = t^2 & \forall\, t \in [0,T] \end{cases}$$

On souhaite déterminer la valeur approchée de $x \mapsto F(x,t)$
Au temps t entre 0 et 15 s avec $x \in [0, 10]$.

En appliquant le schéma d'Euler explicite à l'ordre 1 :

$$\frac{\partial F}{\partial t}(x,t) \approx \frac{F(x,t+dt) - F(x,t)}{dt} \qquad \frac{\partial F}{\partial x}(x,t) \approx \frac{F(x,t) - F(x-dx,t)}{dx}$$

L'équation devient : $F(x,0) = x^2$ et $\forall\, x \in [0,b],\ \forall\, t \in [0,T]$:

$$\frac{F(x,t+dt) - F(x,t)}{dt} = \frac{F(x,t) - F(x-dx,t)}{dx}$$

$$\Longrightarrow \boxed{F(x,t+dt) = \frac{dt}{dx}(F(x,t) - F(x-dx,t)) + F(x,t)} \qquad (*)$$

• Méthode :

1. Un tableau `X` contient les points d'une subdivision régulière de l'intervalle $[0,b]$. On choisit initialement le nombre de points $N_x + 1$ de la subdivision, ça définit le pas $dx = b/N_x$.

2. Un tableau Y de même longueur contient initialement les valeurs de $x \mapsto F(x,0)$ aux points de `X`.

3. On choisit le nombre N_t+1 de temps considéré, ça définit le pas de temps $dt = T/N_t$.

4. A l'aide de l'équation $(*)$ on détermine le tableau `Y1` des valeurs de $x \mapsto F(x,t+dt)$ en fonction de celles `Y0` de $x \mapsto F(x,t)$:

```
for x in range(1,Nx+1):
    Y1[x] = dt/dx * (Y0[x]-Y0[x-1]) + Y0[x]
```

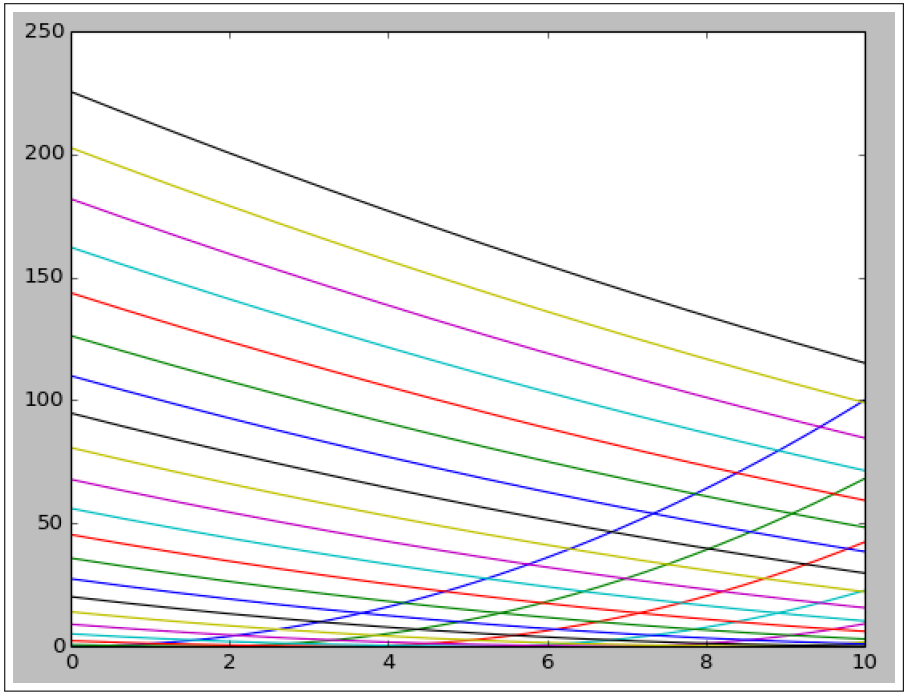
Code :

```
import numpy as np
import matplotlib.pyplot as plt

a, b = 0, 10
Tmax = 15
Nx = 200
Nt = 1000
X = np.linspace(a,b,Nx+1)
dx = (b-a)/Nx    # Pas spatial dx
dt = Tmax/Nt    # Pas temporel dt
f = lambda x: x**2    # Condition initiale à t=0
Y = f(X)
for t in range(Nt):
    Y0 = Y[:]
    Y = np.empty(Nx+1)
    Y[0] = (t*dt)**2    # Condition initiale à x=0 au
temps t
    for x in range(1,Nx+1):
        Y[x] = dt/dx * (Y0[x]-Y0[x-1]) + Y0[x]
```

Tracé tous les 50 pas temporel $\Delta t = 15s \times 10^{-3} \times 50 = 0,75s$:

```
plt.figure(1)
for k in range(0,Nt,50):
    plt.plot(X,Sol[k])
plt.show()
```



5.1.2 Tracé 3D

Pour un tracé 3D :

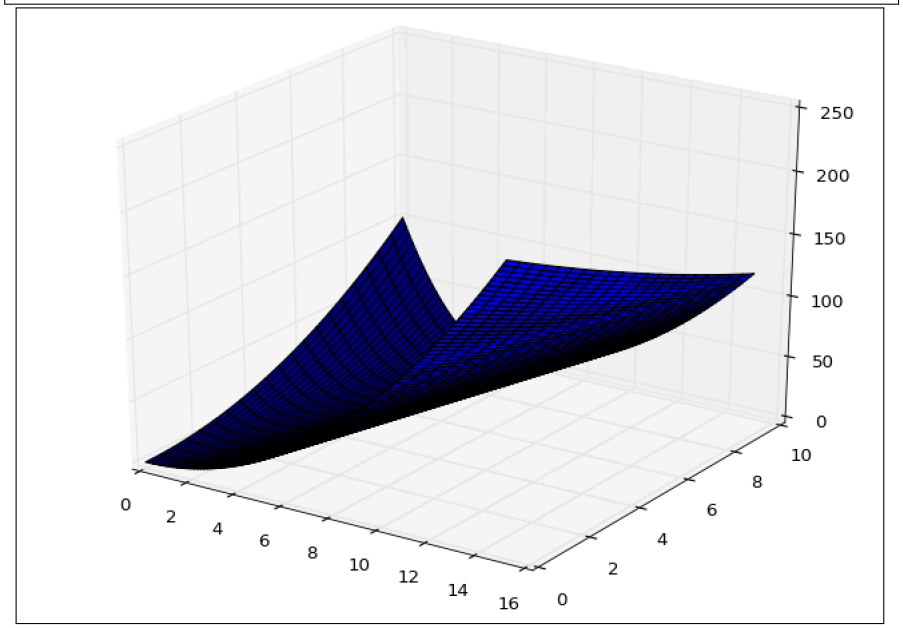
I- Remplir un tableau 2-dimensionel :

```
import numpy as np
import matplotlib.pyplot as plt
a, b = 0, 10
Tmax = 15
Nx = 200
Nt = 1000
```

```
X = np.linspace(a,b,Nx+1)
dx = (b-a)/Nx # Pas spatial dx
dt = Tmax/Nt # Pas temporel dt
f = lambda x: x**2 # Condition initiale à t=0
Z = np.empty((Nx+1,Nt+1))
Z[:,0] = f(X)
for t in range(Nt):
    Z[0,t+1] = (t*dt)**2
    for x in range(1,Nx+1):
        Z[x,t+1] = dt/dx * (Z[x,t]-Z[x-1,t]) + Z[x,t]
```

II- Importer `Axes3D` du sous-module `mpl_toolkits.mplot3d` :

```
from mpl_toolkits.mplot3d import Axes3D
ax = Axes3D(plt.figure())
T = np.linspace(0,Tmax,Nt+1)
T,X = np.meshgrid(T,X)
ax.plot_surface(T, X ,Z)    # T, X, Z : 3 tableaux 2d
plt.show()
```



5.2 Schéma d'Euler explicite d'ordre 2

• On peut résoudre une équation différentielle à l'ordre 2 en utilisant des schémas explicites d'approximation des dérivées premières et secondes.

• Schéma d'Euler à l'ordre 1 :

$$\boxed{f'(x) \approx \frac{f(x+dx)-f(x)}{dx}}$$

• Schéma d'Euler à l'ordre 2 :

$$\boxed{f''(x) \approx \frac{f(x+dx)-2f(x)+f(x-dx)}{dx^2}}$$

• On les utilise aussi pour les EDP d'ordre 2.

• Schéma d'Euler à l'ordre 2 :

$$\boxed{f''(x) \approx \frac{f(x+dx)-2f(x)+f(x-dx)}{dx^2}}$$

Comment s'obtient-il ? Pour une application de classe C^2

Grâce au développement de Taylor-Young à l'ordre 2 de la fonction f :

$$f(x+dx) = f(x) + dx.f'(x) + \frac{1}{2}f''(x)dx^2 + o(dx^2)$$
$$f(x-dx) = f(x) - dx.f'(x) + \frac{1}{2}f''(x)dx^2 + o(dx^2)$$

$$\Longrightarrow f(x+dx) + f(x-dx) = 2f(x) + f''(x)dx^2 + o(dx^2)$$
$$\Longrightarrow f''(x) = \frac{f(x+dx)+f(x-dx)-2f(x)}{dx^2} + \epsilon(dx)$$

avec $\lim\limits_{dx\to 0} \epsilon(dx) = 0$.

lorsque f est de classe C^3 l'approximation est en $o(dx)$.

5.2.1 Résolution d'une Equa.Diff. d'ordre 2

A résoudre sur $[0, 12]$:

$$y'' = \cos(t) \qquad \text{avec } y(0) = -1 \text{ et } y'(0) = 0$$

On insère les schémas d'Euler à l'ordre 1 et 2 dans l'équation et dans une condition initiale :

$$y(t+dt) + y(t-dt) - 2y(t) = dt^2 \cos(t)$$
$$\text{avec} \quad y(0) = -1 \text{ et } \frac{y(dt) - y(0)}{dt} = 0$$

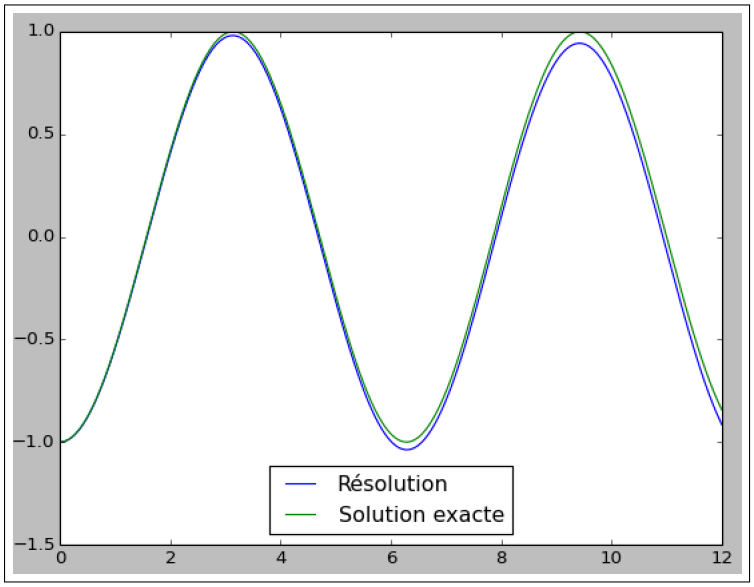
Soit :

$$\boxed{y(t+dt) = 2y(t) - y(t-dt) + dt^2 \cos(t) \quad \text{avec} \quad y(0) = y(dt) = -1}$$

```
Nt = 1000    # choix du nombre de points
T = np.linspace(0,12,Nt+1)
Y = np.empty(Nt+1)
Y[0] = Y[1] = -1    # Conditions initiales
dt = 12 / Nt
for k in range(2, Nt+1):
    Y[k] = 2 * Y[k-1] - Y[k-2] + dt**2 * np.cos(T[k-1])
```

- Tracé :

```
plt.plot(T,Y)
plt.plot(T,-np.cos(T))
plt.legend(('Résolution','Solution exacte'), 'lower
center')
plt.show()
```



L'écart entre les solutions exactes et approchées s'amplifie lorsque t augmente.

5.3 Exemple : l'équation de la chaleur

- *Diffusion de la chaleur le long d'une barre métallique.*

Une barre métallique de longueur L a un coefficient de diffusion K. L'une des extrémités de la barre est reliée à une source de chaleur de température T_1 et l'autre à une source de chaleur de température T_2 avec $T_1 < T_2$. Soit $T(x,t)$ la température de la barre au point d'abscisse x au temps t.
L'équation de la chaleur est l'équation aux dérivées partielles d'ordre 2 :

$$\boxed{\frac{\partial T}{\partial t} = K \frac{\partial^2 T}{\partial x^2}}$$

On se donne $K = 10^{-5} m^2.s^{-1}$, $l = 1m$, $T_1 = 20°C$ et $T_2 = 200°C$. Initialement la température est de T_1 en tout point de la barre.

- On ne peut pas utiliser `odeint`. On peut appliquer les schémas d'Euler :

$\boxed{\frac{\partial T}{\partial t}(x,t) \approx \frac{T(x,t+dt)-T(x,t)}{dt}}$ et $\boxed{\frac{\partial^2 T}{\partial x^2} \approx \frac{T(x+dx,t)+T(x-dx,t)-2T(x,t)}{(dx)^2}}$.

pour exprimer $T(x,t+dt)$ (température au point x au temps $t+dt$) en fonction de $T(x,t)$, $T(x+dx,t)$, $T(x-dx,t)$ (températures aux points $x-dx$, x, $x+dx$ au temps t) et des pas de discrétisation dx et dt.

• L'équation de la chaleur se discrétise pour dx et dt des constantes suffisamment petites en :

$$\frac{T(x,t+dt)-T(x,t)}{dt} \approx K\frac{T(x+dx,t)+T(x-dx,t)-2T(x,t)}{(dx)^2}$$

$$\boxed{T(x,t+dt) \approx T(x,t) + \frac{K dt}{(dx)^2}\left(T(x+dx,t)+T(x-dx,t)-2T(x,t)\right)}$$

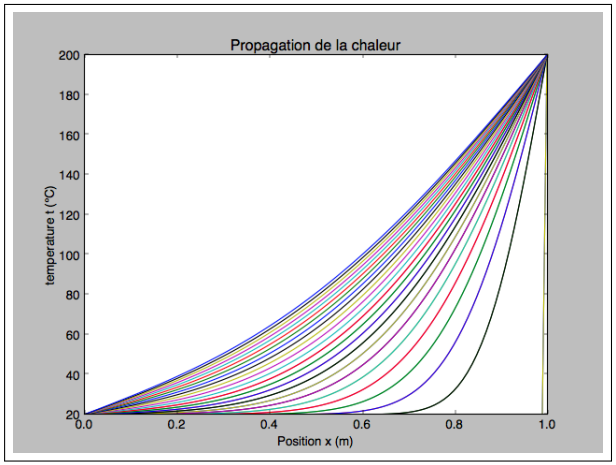
A un instant t la température au différentes positions le long de la barre (subdivision régulière de pas dx de la barre) est contenu dans un tableau unidimensionnel.

Initialement la température est de T_1 en tout point de la barre.

Au temps $t+dt$ la température vaut T_1 au premier point, T_2 au dernier point, et s'obtient en tout autre point x de la barre grâce à la relation ci-dessus à partir des températures au temps t aux positions x, $x-dx$ et $x+dx$. Le tableau des températures au temps $t+dt$ se calcule à partir du tableau au temps t.

On met ainsi à jour la température le long de la barre des temps t = 0 à t_{max} par pas dt de 1 seconde. Toutes les 600 secondes (=10 minutes) on trace la courbe des températures le long de la barre.

```
T1 = 20
T2 = 200
K = 1e-5
tmax = 4*3600    # 4 h = 4*3600 s
n=100
x = np.linspace(0,1,n)
T = T1*np.ones(n)
dx = 1./(n-1) ; dt =1
c = K*dt/(dx**2)
for i in range(tmax):    # Evolution temporelle
    t = np.empty(n)        # Températures à t+dt... à
remplir
    for k in range(1,n-1):    # Mise à jour
        t[k] = T[k]+c*(T[k+1]+T[k-1]-2*T[k])
    t[0]=T1 ; t[n-1]=T2    # Température aux 2 extrémités
    T = t                  # Température à t + dt
    if (i%600 == 0):       # Toutes les 10 minutes = 600
s...
        plt.plot(x,T)    # ... Tracer la courbe de
température
plt.xlabel('Position x (m)'); plt.ylabel('temperature t
(°C)')
plt.title('Propagation de la chaleur')
```



5.3.1 Tracé 3D

```
T1=20
T2=200
K=1e-5
tmax = 4*3600
Nt = tmax
Nx = 100
dx=1/Nx
dt=tmax/Nt
Z = np.empty((Nx+1,Nt+1))
Z[:,0] = T1*np.ones(Nx+1)
c= K*dt/dx**2
for t in range(Nt):
    Z[0,t+1], Z[Nx,t+1] = T1, T2
    for x in range(1,Nx):
        Z[x,t+1] = Z[x,t]+c*(Z[x+1,t]+Z[x-1,t]-2*Z[x,t])
```

On remplit un tableau bidimensionnel des valeurs de la température au dessus d'un maillage de $[0,1] \times [0, t_{max}]$.

```
from mpl_toolkits.mplot3d import Axes3D
ax = Axes3D(plt.figure())
X=np.linspace(0,1,Nx+1)
T=np.linspace(0,tmax,Nt+1)
T,X = np.meshgrid(T,X)
ax.plot_surface(T,X,Z)
plt.show()
```