IN4-Cours

Résolution d'une EDO

On apprend dans ce cours à résoudre une équation différentielle ordinaire d'ordre un (EDO1) numériquement, par la **méthode d'Euler**. Une EDO1 est une équation portant sur une fonction $y(t)$ de la forme

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

avec f quelconque.

Table des matières

1 Exemples

On rencontre très souvent des EDO1 en physique ou en chimie sous la forme des équations différentielles ordinaires linéaires à coefficients constants avec ou sans second membre constant. Par exemple, la tension u_C aux bornes du condensateur dans un circuit RC série vérifie

$$\frac{\mathrm{d}u_C}{\mathrm{d}t} + \frac{1}{RC}u_C = 0$$

Le champ de pression $P(z)$ dans le modèle de l'atmosphère isotherme vérifie

$$\frac{\mathrm{d}P}{\mathrm{d}z} + \frac{1}{H}P = 0$$

où z désigne l'altitude et H une hauteur caractéristique du problème. Un dernier exemple, la vitesse $\vec{v}$ d'une bille tombant dans un fluide visqueux vérifie

$$\frac{\mathrm{d}v}{\mathrm{d}t} + \frac{6\pi\eta R}{m}v = -g$$

si $\vec{v} = v(t)\vec{e_z}$ avec l'axe z vertical ascendant. Ici η est la viscosité du fluide, R le rayon de la bille et m sa masse. Ces trois exemples se résolvent analytiquement (la solution est à chaque fois exponentielle) et présentent donc peu d'intérêt pour une résolution numérique, mais cela n'est pas toujours le cas. Dans une situation à préciser, on pourrait par exemple avoir à résoudre une équation de la forme

$$y' + \cos(y^2) = 0$$

et là nous n'aurions pas d'autres choix que de chercher à la résoudre numériquement, puisqu'aucune solution analytique n'existe.

2 Principe de résolution par la méthode d'Euler explicite

Pour poser les notations, on cherche à résoudre l'équation portant sur la fonction $y(t)$, donnée par

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

Comme toutes les équations différentielles du premier ordre, il faut préciser la valeur en un point (souvent $t=0$, et on parle dans ce cas de **condition initiale**) pour que le système soit convenablement posé, c'est-à-dire que la solution soit unique. Dans ce cours on pose

$$y(t=0) = y_0$$

Le principe de la méthode d'Euler est de résoudre cette équation numériquement, en approchant la fonction $y(t)$ point par point successivement.

2.1 Discrétisation des abscisses

Il est impossible de déterminer $y(t)$ pour tout t (t est une variable continue donc elle peut prendre une infinité de valeurs...). On cherche donc en fait seulement à déterminer $y(t)$ en certaines abscisses t_n, et on choisit dans ce cours de les prendre espacées d'un pas Δt.

$$t_n = n\,\Delta t$$

2.2 Résolution pas à pas par différence finie

La condition initiale nous donne $y(t=0=t_0) = y_0$. La méthode d'Euler consiste à déterminer successivement $y(t_1)$ à partir de $y(t_0)$, puis $y(t_2)$ à partir de $y(t_1)$, puis $y(t_3)$ à partir de $y(t_2)$, et ainsi de suite...

Principe de la méthode d'Euler. Pour cela, on utilise la **méthode des différences finies** (voir le chapitre IN1) pour approcher la dérivée dans l'équation différentielle

$$y'(t) = f(y(t),t) \qquad \text{devient au temps } t_n \qquad \frac{y(t_{n+1}) - y(t_n)}{\Delta t} = f(y(t_n), t_n)$$

et donc

$$\boxed{y(t_{n+1}) = y(t_n) + f(y(t_n), t_n)\,\Delta t} \qquad \textbf{(Schéma d'Euler explicite)}$$

Cela donne une **relation de récurrence** pour calculer successivement tous les $y(t_n)$ à partir de $y(t_0)$.

2.3 Proposition d'implémentation

Proposition d'implémentation du schéma d'Euler.

Les arguments de la fonction sont la fonction f, les deux abscisses a et b entre lesquelles on souhaite approcher la solution et N le nombre de pas et la condition initiale $y(a) = y_0$. Les temps t_n vont de a inclu à b exclu. La fonction retourne les $y(t_n)$ sous forme de liste.

```
def euler(f, y0, a, b, N) :
    h = (b - a) / N                          # le pas de discrétisation
    y = [y0]                                 # la première valeur
    t = [ a + h * i for i in range(N) ]      # la liste des abscisses

    for i in range(1, N) :
        y.append( y[-1] + f(y[-1], t[i-1]) )      # on implémente la liste des y

    return y
```

Cette fonction est générale, mais en pratique on peut écrire une fonction particulière pour chaque équation rencontrée.

2.4 Exemple

On souhaite résoudre l'équation différentielle

$$y' + \frac{1}{\tau}\,y = 0 \qquad \text{avec} \qquad y(0) = y_0$$

pour t entre 0 et 5τ. Cela correspond à une fonction f valant

$$f(y,t) = -\frac{1}{\tau}\, y$$

indépendante de t. On implémente sa résolution numérique de la manière suivante

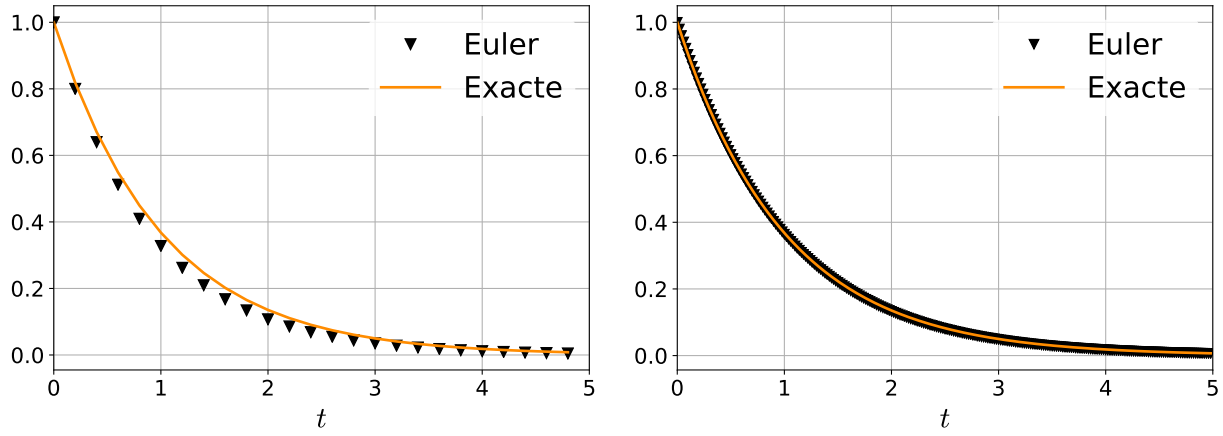
```
y0 = 1.
tau = 1.
N = 200
h = 5 * tau / N
y = [y0]

for i in range(N) :
    y.append( y[-1] - y[-1] / tau * h )
```

L'utilisation de la bibliothèque `pyplot` de `matplotlib` permet de tracer la solution approchée obtenue et de la comparer à la solution analytique

$$y(t) = y_0 \exp\left(-\frac{t}{\tau}\right)$$

Pour $N = 25$ points, on obtient la courbe suivante (à gauche) :



2.5 Erreur commise

On observe que les points obtenus par la méthode d'Euler sont proches mais dévient quand même un peu de la solution exacte.

Une manière d'améliorer ceci est de diminuer le pas de discrétisation Δt en augmentant le nombre de points N. On obtient alors la courbe suivante avec $N = 250$ points (au dessus à droite). Cette fois les points sont effectivement bien plus proches de la solution exacte. Cela s'explique par le fait que plus le pas Δt est petit, plus la méthode de différence finie estime bien la dérivée ! Voir le chapitre IN1.

Par contre, soulignons que plus le nombre de points N est grand, plus le programme prend du temps à s'exécuter : il est en effet en $O(N)$ du fait de la boucle for !

3 Autres méthodes de résolution

Il existe plein d'autres méthodes de résolution, dérivées du schéma d'Euler vu précédemment (qu'on appelle parfois **shéma d'Euler explicite**).

Mentionnons le **schéma d'Euler implicite**

$$\boxed{y(t_{n+1}) = y(t_n) + f(y(t_{n+1}), t_{n+1})\,\Delta t} \qquad \textbf{(Schéma d'Euler implicite)}$$

qui consiste à évaluer f à t_{n+1} plutôt qu'à t_n, ainsi que le **schéma de Crank-Nicholson**

$$\boxed{y(t_{n+1}) = y(t_n) + \frac{1}{2}\Big(f(y(t_{n+1}), t_{n+1}) + f(y(t_n), t_n)\Big)\,\Delta t} \qquad \textbf{(Schéma de Crank-Nicholson)}$$

qui prend une moyenne des valeurs de f à t_n et t_{n+1}.

Par ailleurs, il en existe d'autres basés sur des méthodes de différence finie variées. Mentionnons seulement le **schéma de saute-mouton** (*leapfrog scheme* en anglais) qui utilise l'approximation suivante pour la dérivée

$$y'(t) = \frac{y(t+\Delta t) - y(t-\Delta t)}{2\,\Delta t}$$

(voir le chapitre IN1). Le schéma de résolution est alors

$$\boxed{y(t_{n+1}) = y(t_{n-1}) + 2\,f(y(t_n), t_n)\,\Delta t} \qquad \textbf{(Schéma de saute-mouton)}$$

et on comprend que l'obtention de y en t_{n+1} ne part plus de $y(t_n)$ mais de $y(t_{n-1})$, d'où le nom de « saute-mouton ».

4 Stabilité des schémas d'Euler

Ayant donné l'idée générale des résolutions d'équations différentielles par différence finie, on peut souhaiter en étudier un exemple afin de commenter un peu plus les mathématiques sous-jacente. Concentrons-nous alors sur l'équation très répandue

$$y'(t) + \frac{1}{\tau}\,y(t) = 0 \qquad \text{dont la solution est} \qquad y(t) = y_0\,\exp\left(-\frac{t}{\tau}\right)$$

Et étudions deux schémas de résolutions numériques de cette équation : le schéma d'Euler explicite et le schéma d'Euler implicite. Nous souhaitons en fait montrer les différences entre les deux, en nous concentrant notamment sur la stabilité de ces schémas.

Définition. Un schéma est dit **stable** s'il converge bien vers la solution exacte lorsque $t \to \infty$.

4.1 Schéma d'Euler explicite

Adoptons les notations suivantes : Δt est le pas de temps. Le temps est discrétisé et les temps discrets sont donc les $t_k = k\,\Delta t$. Notons ensuite $y_k = y(t_k)$ les valeurs de y aux temps discrétisés.

Le schéma d'Euler se construit en approximant la dérivée par la différence finie

$$y'(t_k) \approx \frac{y(t+\Delta t) - y(t)}{\Delta t} = \frac{y(t_{k+1}) - y(t_k)}{\Delta} = \frac{y_{k+1} - y_k}{\Delta t}$$

L'équation que l'on cherche à résoudre est $y' = -y\,/\,\tau$, qui devient pour un schéma explicite où on va évaluer le membre de droite à l'instant t_k :

$$\frac{y_{k+1} - y_k}{\Delta t} = -\frac{y_k}{\tau}$$

Finalement, on se rend compte que pour cette équation simple le calcul peut être fait à la main ! On avance effectivement facilement :

$$y_{k+1} = \left(1 - \frac{\Delta t}{\tau}\right) y_k \qquad \text{et on reconnait une suite géométrique, donc} \qquad \boxed{y_n = \left(1 - \frac{\Delta t}{\tau}\right)^n y_0}$$

Deux commentaires nous viennent : déjà, si on fixe un temps total $T = t_N = N\,\Delta t$ avec N le nombre de points, c'est-à-dire $n \in [0, N]$, alors $\Delta t = T\,/\,N$ et on peut écrire

$$y(T) = y_N = \left(1 - \frac{T}{N\,\tau}\right)^N y_0 = y_0\,\exp\left(N\,\ln\left(1 - \frac{T}{N\,\tau}\right)\right)$$

et lorsqu'on fait tendre $N \to \infty$, puisque $\ln(1-x) \approx -x$ pour $x \to 0$, on obtient

$$y(T) \to y_\infty = y_0\,\exp\left(-\frac{T}{\tau}\right)$$

qui est la solution exacte !

Conclusion 1. Le schéma d'Euler explicite tend vers la solution exacte lorsque le nombre de points tend vers l'infini $N \to \infty$.

Nous l'avions intuité précédemment (plus il y a de points, plus la solution numérique est précise), voilà ce résultat démontré.

Ensuite, deuxième commentaire : si au contraire on fixe Δt et qu'on fait tendre N vers l'infini (donc T aussi), alors comment se comporte y_N ? Physiquement, la solution exacte est exponentiellement décroissante, donc elle tend vers 0. Mais numériquement

$$y_N = y_0 \left(1 - \frac{\Delta t}{\tau}\right)^N \underset{N\to\infty}{\longrightarrow} \begin{cases} 0 & \text{si } |1 - \Delta t\,/\,\tau| < 1 \\ \infty & \text{sinon} \end{cases}$$

En fait, si

$$\left|1 - \frac{\Delta t}{\tau}\right| > 1 \qquad \text{soit} \qquad \Delta t > 2\,\tau$$

alors le schéma d'Euler explicite est instable ! Il tend vers l'infini au lieu de tendre vers 0 !

Conclusion 2. Le schéma d'Euler explicite n'est stable que si

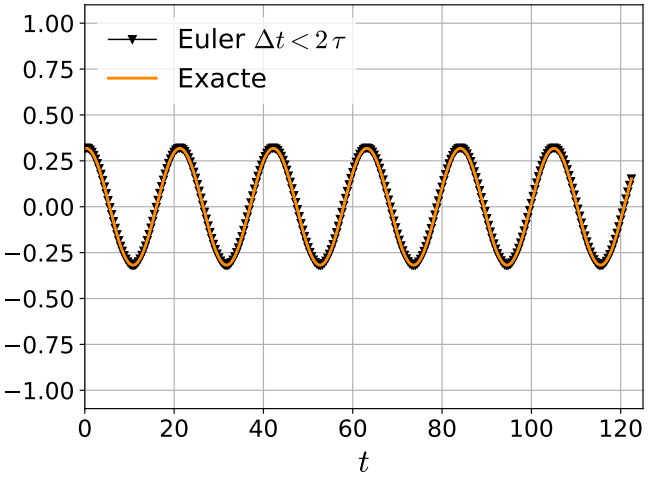
$$\boxed{\Delta t < 2\,\tau}$$

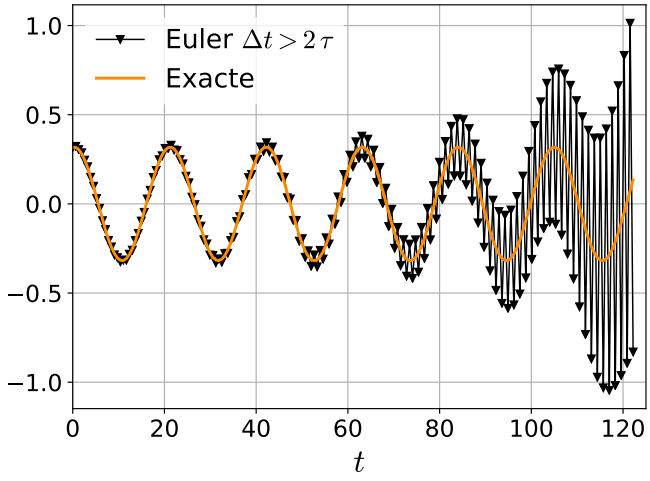
Notons que notre étude est propre à l'exemple de $y' = -y\,/\,\tau$ et que le résultat n'a pas le statut de généralité.

Cependant, on peut montrer et on retiendra que le **schéma d'Euler explicite est conditionnellement stable**, c'est-à-dire qu'il existe un pas de temps maximal à ne pas dépasser pour que le schéma soit stable (c'est-à-dire qu'il converge vers la solution exacte).

Illustration sur une autre équation. Donnons les résultats obtenus par le schéma d'Euler explicite dans les deux cas stable $\Delta t < 2\,\tau$ (à gauche) et instable $\Delta t > 2\,\tau$ (à droite) pour l'équation ci-dessous :

$$y' = -\frac{y}{\tau} + \cos \omega\, t$$





L'instabilité que provoque le schéma d'Euler explicite est clairement visible. Par contre, ce n'est en fait pas le cas du schéma d'Euler implicite ! D'où son intérêt ! Étudions-le.

4.2 Schéma d'Euler implicite

Mêmes notations, mêmes calculs. Cette fois, le schéma d'Euler implicite consiste à évaluer le membre de droite de $y' = -y\,/\,\tau$ à l'instant t_{k+1} (au lieu de t_k pour le schéma explicite). Alors en utilisant une différence finie on obtient la relation

$$\frac{y_{k+1} - y_k}{\Delta t} = -\frac{1}{\tau}\, y_{k+1}$$

soit cette fois

$$y_{k+1} = \frac{y_k}{1 + \dfrac{\Delta t}{\tau}} \qquad \text{donc} \qquad \boxed{y_n = y_0 \left(1 + \frac{\Delta t}{\tau}\right)^{-n}}$$

Faisons alors les deux mêmes commentaires : déjà, pour T fixé, donc $\Delta t = T\,/\,N$, on a

$$y_N = y_0 \left(1 + \frac{T}{N\,\tau}\right)^{-N} = y_0 \exp\left(-N \ln\left(1 + \frac{T}{N\,\tau}\right)\right) \underset{N\to\infty}{\longrightarrow} y_0 \exp\left(-\frac{T}{\tau}\right)$$

Conclusion 1. Le schéma d'Euler implicite converge bien lui aussi vers la solution exacte lorsque le nombre de points tend vers l'infini $N \to \infty$.

et à l'inverse, si Δt est fixé et qu'on regarde la limite de y_N pour $N \to \infty$, alors

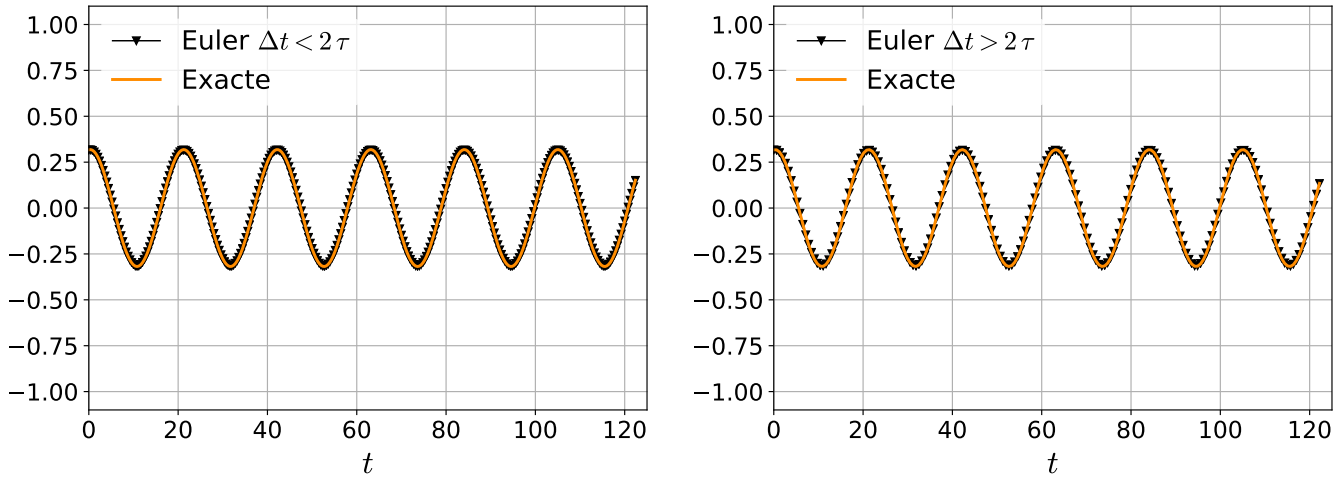
$$y_N = \frac{y_0}{\left(1 + \dfrac{\Delta t}{\tau}\right)^N} \to 0$$

et cela sans aucune condition sur Δt !

Conclusion 2. Le **schéma d'Euler implicite est donc inconditionnellement stable**, contrairement au schéma d'Euler explicite.

Illustration. Donnons les résultats obtenus par le schéma d'Euler implicite dans les deux cas $\Delta t < 2\,\tau$ (à gauche) et $\Delta t > 2\,\tau$ (à droite) pour la même équation d'illustration que précédemment

$$y' = -\frac{y}{\tau} + \cos \omega\, t$$



Aucune instabilité n'est constatée. Le schéma d'Euler implicite est donc bien meilleur en ce sens. Notez cependant que l'exemple que nous avons choisi est particulièrement simple. Pour un exemple plus compliqué (typiquement une équation d'ordre $n > 1$) un schéma implicite requiert souvent une inversion matricielle, plus lourde en calculs... Il n'y a pas que des avantages !

Pour ceux qui serait intéressés, on donne le programme ci-dessous pour obtenir les courbes montrées.

```
import numpy as np
from matplotlib import pyplot as plt
plt.rc('mathtext', fontset="cm")

tau = 0.32
w = 0.3
T = 123.
Dt = 0.4    #  si Dt < 2 tau, sinon mettre Dt = 0.65
N = int(T / Dt)
t = np.array([ i * Dt for i in range(N) ])
alpha = tau / (1 + tau**2 * w**2)
y0 = alpha
y = [y0]

def exacte(u) :
    yexa = (y0 - alpha) * np.exp(-u/tau) + alpha * ( np.cos(w*u) + w * tau * np.sin(w*u) )
    return yexa

# Pour le schema explicite
for i in range(1, N) :
    y.append( y[-1] * (1 - Dt / tau) + np.cos(w * t[i-1]) * Dt )
```

```
# Pour le schema implicite
#for i in range(1, N) :
    y.append( ( y[-1] + np.cos(w * t[i]) * Dt ) / (1 + Dt / tau) )

plt.xlim([0.0,125.])
plt.ylim([-1.1,1.1])
plt.grid()
plt.xlabel(r"$t$", fontsize=22)
plt.plot(t, y, color='black', marker='v', markersize=5, label=r"Euler $\Delta t < 2\, \tau$")
plt.plot(t, exacte(t), color='darkorange', linewidth=2, label=r"Exacte")
plt.xticks(fontsize=16)
plt.yticks(fontsize=16)
legend = plt.legend(loc = 2, numpoints = 1, handlelength=1.8, frameon=1, fontsize=18)
frame = legend.get_frame()
frame.set_facecolor('white')
frame.set_edgecolor('white')
plt.tight_layout()
plt.show()
```

5 Pour aller plus loin : principe de résolution d'EDO d'ordre n

Ayant étudié l'équation simple $y' = -y/\tau$ (d'ordre 1), on peut chercher à s'attaquer à des équations (un peu) plus compliquées. Notamment, on peut se demander comment résoudre une équation d'ordre supérieur à 1.

Méthode. Pour résoudre une équation différentielle d'ordre n, on se ramène à n équations d'ordre 1.

Expliquons l'astuce sur un exemple, par exemple l'équation d'un oscillateur harmonique amorti

$$y'' + \frac{\omega_0}{Q}\, y' + \omega_0^2\, y = 0$$

Il suffit de définir un jeu de deux variables contenant la dérivée première comme une variable nouvelle, par exemple

$$\begin{cases} u &= y \\ v &= y' \end{cases}$$

alors évidemment $y'' = v'$ et $u' = v$. On peut donc écrire, à l'aide de l'équation de l'oscillateur harmonique amorti,

$$\begin{cases} u' &= v \\ v' &= -\dfrac{\omega_0}{Q}\, v - \omega_0^2\, u \end{cases}$$

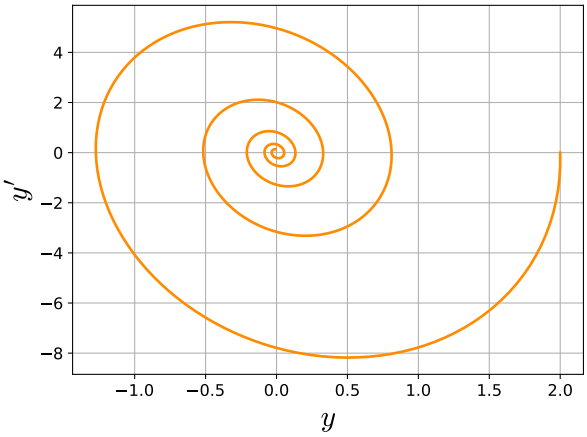
Il s'agit bien de deux équations du premier ordre (mais couplées), équivalentes à l'équation initiale. On peut alors les résoudre par un schéma d'Euler explicite par exemple (en espérant les notations elles-aussi explicites !)

$$\begin{cases} \dfrac{u_{k+1} - u_k}{\Delta t} &= v_k \\ \dfrac{v_{k+1} - v_k}{\Delta t} &= -\dfrac{\omega_0}{Q}\, v_k - \omega_0^2\, u_k \end{cases} \quad \text{soit} \quad \begin{cases} u_{k+1} &= u_k + v_k\, \Delta t \\ v_{k+1} &= v_k - \dfrac{\omega_0}{Q}\, v_k\, \Delta t - \omega_0^2\, u_k\, \Delta t \end{cases}$$

Puis évidemment, on obtient la solution y de l'équation du 2ème ordre par la donnée de $u = y$.

Remarque. Notez que dans ce cas d'une équation d'ordre 2, on apprécie généralement tracer la solution dans un **portrait de phase** (y, y') soit ici (u, v). Pour l'oscillateur harmonique amorti, on obtient la figure ci-contre avec les paramètres $Q = 3$, $\omega_0 = 5$, $\Delta t = 0.01$ et un nombre de points $N = 600$. Les conditions initiales sont $y(0) = 2$ et $y'(0) = 0$. L'implémentation de la méthode d'Euler s'écrit selon

```
u, v = [2.], [0.]
for i in range(1, N) :
   u.append( u[-1] + v[-1] * Dt )
   v.append( v[-1] - w0 / Q * v[-1] * Dt
             - w0**2 * u[-2] * Dt)
```



6 Pour aller plus loin : résolution d'une EDP

Compliquons encore davantage le propos en considérant non pas une équation différentielle ordinaire, mais une équation aux dérivées partielles. Prenons l'exemple de l'équation de diffusion thermique

$$\frac{\partial T}{\partial t} = D\,\frac{\partial^2 T}{\partial x^2}$$

où $T(x,t)$ est un champ (de température).

Le champ dépendant cette fois de deux variables, il faut discrétiser les deux. On définit le pas de temps Δt et le pas d'espace Δx. Les temps discrétisés sont alors les $t_k = k\,\Delta t$ et les positions discrétisées sont les $x_i = i\,\Delta x$. Notons alors $T_i^k = T(x_i, t_k)$.

Ensuite, on évalue les dérivées par des différences finies (voir fiche IN1)

$$\frac{\partial T}{\partial t}(x,t) \approx \frac{T(x,t+\Delta t) - T(x,\,t)}{\Delta t} \qquad \text{et} \qquad \frac{\partial^2 T}{\partial x^2}(x,t) \approx \frac{T(x+\Delta x,t) + T(x-\Delta x,t) - 2\,T(x,t)}{(\Delta x)^2}$$

soit numériquement

$$\frac{\partial T}{\partial t}(x,t) \approx \frac{T_i^{k+1} - T_i^k}{\Delta t} \qquad \text{et} \qquad \frac{\partial^2 T}{\partial x^2}(x,t) \approx \frac{T_{i+1}^k + T_{i-1}^k - 2\,T_i^k}{(\Delta x)^2}$$

alors finalement un schéma d'Euler explicite pour résoudre l'équation de la diffusion est

$$\frac{T_i^{k+1} - T_i^k}{\Delta t} = D\,\frac{T_{i+1}^k + T_{i-1}^k - 2\,T_i^k}{(\Delta x)^2}$$

La relation de récurrence à utiliser pour déterminer l'évolution temporelle du champ de température est ainsi

$$T_i^{k+1} = T_i^k + \frac{D\,\Delta t}{(\Delta x)^2}\left(T_{i+1}^k + T_{i-1}^k - 2\,T_i^k\right)$$

Remarque. Le schéma est explicite car on exprime T_i^{k+1} en fonction des températures à l'instant précédent k uniquement.

Remarque. Le schéma proposé n'est pas valable sur les bords de l'espace, en $i = 0$ et $i = N$, puisque T_{N+1}^k et T_{-1}^k n'existent pas. Pour pallier cette difficulté, on change spécifiquement de schéma de résolution sur ces deux points. On peut montrer aisément par des développements limités que la dérivée seconde s'approxime également par

$$\frac{\partial^2 T}{\partial x^2}(x,t) \approx \frac{T(x+2\,\Delta x,t) + T(x,t) - 2\,T(x+\Delta x,t)}{(\Delta x)^2} \qquad \text{et} \qquad \frac{\partial^2 T}{\partial x^2}(x,t) \approx \frac{T(x,t) + T(x-2\,\Delta x,t) - 2\,T(x-\Delta x,t)}{(\Delta x)^2}$$

soit numériquement

$$\frac{\partial^2 T}{\partial x^2}(x,t) \approx \frac{T_{i+2}^k + T_i^k - 2\,T_{i+1}^k}{(\Delta x)^2} \qquad \text{et} \qquad \frac{\partial^2 T}{\partial x^2}(x,t) \approx \frac{T_i^k + T_{i-2}^k - 2\,T_{i-1}^k}{(\Delta x)^2}$$

La formule de gauche peut être utilisée pour le bord $i = 0$ et celle de droite pour le bord $i = N$.

Cela a été fait en TP pour rappel. Voir le programme python associé à cet algorithme dans la capacité numérique CN5.

Exercice 1. Écrire un programme qui utilise le schéma de Crank-Nicholson pour résoudre l'équation

$$y' = \cos(y) + \sin(t)$$

On utilisera le cours pour cela, le schéma de Crank-Nicholson n'est pas à connaître par cœur.