

Méthode d'Euler

I - Principe de la méthode

Cette fiche méthode vous fournira la partie théorique nécessaire à la compréhension de la méthode de résolution numérique d'une équation différentielle, et servira de support pour le premier TP commun informatique - physique.

↪ Cliquez ou flashez le QR code ci-contre pour une présentation de la méthode d'Euler !



● Position du problème

Raisonnons sur un exemple simple : le tension u aux bornes d'un condensateur d'un circuit RC, régie par l'équation différentielle suivante :

$$\frac{du}{dt} + \frac{1}{\tau}u = \frac{E}{\tau} \quad (1)$$

avec $\tau = RC$ et E la force électromotrice du générateur de tension.

Comme il n'est pas possible de résoudre une équation différentielle "à tout instant" avec un ordinateur, il est nécessaire de passer par une résolution à N instants discrets : on dit que le pas de temps est **quantifié**. On appelle alors $t_0, t_1, \dots, t_n, \dots, t_{N-1}$ ces différents instants. On cherche alors une **solution approchée** de l'équation différentielle, qui soit la plus proche possible de la solution analytique trouvée par résolution directe de l'équation. Dans toute la suite, le pas de temps Δt sera pris constant : $t_n - t_{n-1} = \Delta t$

● Comment faire en pratique ?

Il faut transformer l'équation différentielle en une relation de récurrence permettant de calculer u_{n+1} (la tension à l'instant t_{n+1}) à partir uniquement de ce que l'on connaît, c'est-à-dire la tension aux instants passés : u_n, u_{n-1}, \dots , et les paramètres caractéristiques du système (E, τ, \dots) à tout instant et les paramètres caractéristiques du système. La "recette" pour passer de l'équation différentielle à la relation de récurrence est appelée schéma numérique.

Schéma d'Euler explicite

La dérivée est approximée par un taux de variation entre les instants t_n et t_{n+1} , et toutes les autres grandeurs sont prises à l'instant t_n . La dérivée d'une fonction s'écrit alors :

$$\frac{du}{dt} \approx \frac{u_{n+1} - u_n}{\Delta t}$$

Ainsi, l'équation différentielle du circuit RC (1) s'écrit :

$$\frac{u_{n+1} - u_n}{\Delta t} + \frac{1}{\tau}u_n = \frac{1}{\tau}E \quad \text{soit} \quad u_{n+1} = u_n + \frac{\Delta t}{\tau}(E - u_n)$$

Bien sûr, il existe des schémas numériques plus astucieux, plus précis et/ou plus rapides, d'où l'intérêt d'utiliser des fonctions issues de bibliothèques. Notons d'ailleurs qu'il existe des domaines de recherche comme la climatologie, la turbulence, l'astrophysique, la physique des plasmas, la météorologie, etc., dans lesquels une grosse partie, pour ne pas dire la majorité, du travail théorique consiste à construire des schémas numériques performants pour résoudre des équations "assez simples" à poser.

II - Implémentation en Python

Rappelons que pour un même algorithme, ici une même équation différentielle et un même schéma numérique, il existe souvent de nombreuses implémentations différentes. Deux exemples sont présentés ci-dessous, avec deux syntaxes différentes.

Le préambule ainsi que la partie dédiée à la représentation graphique ci-après sont communes aux deux implémentations présentées.

● Préambule commun

```

1 import numpy as np
2 import matplotlib.pyplot as plt      # utilisé pour la représentation graphique
3
4 R = 1e3      # R = 1 kiloohm
5 C = 1e-7     # C = 100nF
6 tau = R*C   # temps caractéristique
7 E = 5       # force électromotrice du générateur
8
9 dt = 2e-5   # pas de temps, en s
10 N = 100    # nombre de pas de temps tel que temps total = N*dt

```

● Premier exemple d'implémentation

```

1 t = [i*dt for i in range(N)]      # on crée un tableau contenant tous les temps
2 u = [None for i in range(N)]     # on crée un tableau vide de taille N pour la u
3 u[0] = 0                          # condition initiale u(t=0) = 0
4
5 for i in range (N-1):
6     u[i+1] = u[i] + dt/tau*(E - u[i])  #schéma explicite d'Euler

```

● Deuxième exemple d'implémentation

```

1 t = np.linspace(0, (N-1)*dt, N)  # on crée un tableau contenant tous les temps
2 u = np.empty_like (t)           # initialisée à un tableau vide de même forme que t
3 u[0] = 0                        # condition initiale u(t=0) = 0
4
5 for i in range (N-1):
6     u[i+1] = u[i] + dt/tau*(E - u[i])  #schéma explicite d'Euler

```

● Représentation graphique

```

1 plt.grid()
2 plt.xlabel("$t$ (ms)")
3 plt.ylabel("$u(t)$ (V)")
4 plt.plot(t,u, label = "Solution numérique par la méthode d'Euler")
5 plt.legend()
6 plt.show()

```

III - Résultats

Un schéma numérique ne propose qu'une résolution numérique **approchée** de l'équation différentielle. Il importe donc de s'interroger sur la pertinence de ce résultat, qui peut très nettement s'écarter du résultat théorique prévu par la résolution analytique de l'équation différentielle.

Dans le cas de la méthode d'Euler, le seul paramètre choisi est le pas de temps Δt . La discussion de la pertinence des résultats dépend donc du choix de ce paramètre.

Précision de la méthode d'Euler

Pour que la méthode d'Euler conduise à un résultat précis, il faut que le pas de temps Δt soit très faible devant les constantes de temps caractéristiques du système étudié.

Pour la modélisation numérique d'un système du premier ordre, on retiendra :

$$\Delta t \ll \tau$$

Ci-dessous sont représentées les différentes simulations réalisées pour des pas de temps différents, allant de $0,1\tau$ à $0,8\tau$.

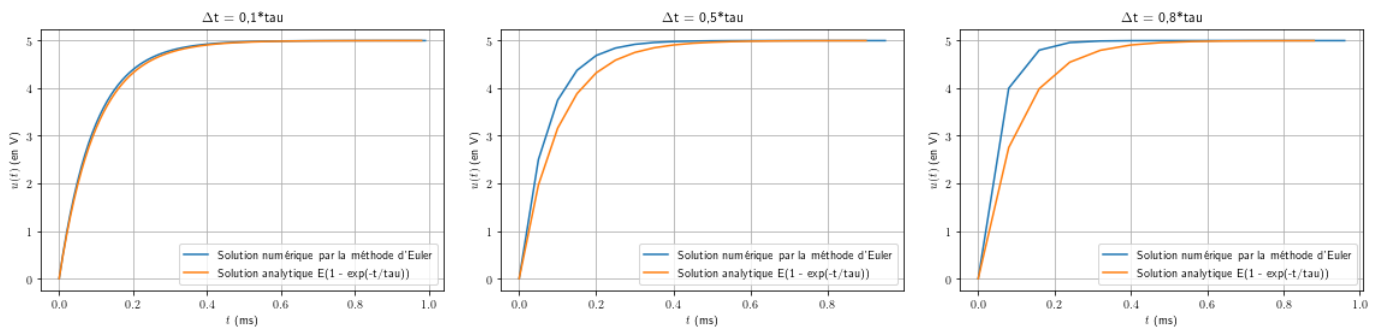


FIGURE 1 – Tracé de différentes solutions numériques pour $\Delta t = 0,1\tau$, $0,5\tau$ et $0,8\tau$. On remarque que lorsqu'on diminue le pas de temps, la solution numérique s'approche de la solution analytique.

IV - À vous de jouer !

Maintenant que nous avons présenté le principe de l'implémentation de la méthode d'Euler, je vous invite à vous entraîner à la résolution numérique d'équations différentielles du premier ordre sur les exemples suivants.

► Exemple n°1 : Pression dans la troposphère

Dans la troposphère, couche occupant les 11 premiers kilomètres de l'atmosphère, on peut montrer que le champ de pression $P(z)$ varie avec l'altitude z selon l'équation différentielle :

$$\frac{dP}{dz} = -P \frac{Mg}{R(T_0 - \alpha z)}$$

avec $M = 29.10^{-3} \text{ g.mol}^{-1}$ la masse molaire de l'air, $g = 9,81 \text{ m.s}^{-2}$ l'accélération de la pesanteur, $R = 8,3 \text{ J.mol}^{-1}.\text{K}^{-1}$ la constante des gaz parfaits, $T_0 = 288 \text{ K}$ la température moyenne à la surface de la Terre et $\alpha = 6.10^{-3} \text{ K.m}^{-1}$ le gradient thermique.

1. Écrire la relation de récurrence issue du schéma d'Euler appliqué à cette équation différentielle.
2. L'implémenter en Python pour trouver le champ de pression de 0 à 11 km d'altitude avec un pas de 1 m.

► Exemple n°2 : Vidange d'un réservoir

Par application des lois de la mécanique des fluides, on peut montrer que la hauteur d'eau h dans un réservoir se vidant par un petit orifice situé à sa base vérifie l'équation différentielle suivante, appelée relation de Torricelli :

$$\frac{dh}{dt} = -\alpha \sqrt{2gh}$$

avec g l'accélération de la pesanteur et α le rapport des sections de l'orifice de vidange et du réservoir.

1. Écrire la relation de récurrence issue du schéma d'Euler appliqué à cette équation différentielle.
2. L'implémenter en Python pour déterminer la hauteur d'eau restant dans le réservoir jusqu'à la vidange complète. On considérera le cas d'un réservoir d'eau de pluie prévu pour l'arrosage d'un potager. On prendra alors : $\alpha = 1,0.10^{-3}$, $h(t = 0) = 1,50 \text{ m}$, et un pas de temps $\Delta t = 0,1 \text{ s}$.

► Exemple n°3 : Chute d'un corps soumis à des frottements quadratiques

D'après le principe fondamental de la dynamique, un corps de masse m soumis au champ de pesanteur g et à une force de frottement $-kv^2$ a une vitesse donnée par l'équation différentielle suivante :

$$\frac{dv}{dt} + \frac{k}{m}v^2 = g$$

Cette équation est *non-linéaire* : elle ne présente a priori pas de solution analytique. On va donc estimer la solution grâce à la méthode d'Euler.

1. Écrire la relation de récurrence issue du schéma d'Euler appliqué à cette équation différentielle.
2. On considère que le voltigeur a une masse $m = 80 \text{ kg}$, et que le coefficient de frottement vaut $k = 1,6 \text{ USI}$. Représenter la vitesse du voltigeur en fonction du temps pour un pas de temps $\Delta t = 0,2 \text{ s}$.
3. Représenter l'altitude $z(t)$ du voltigeur en fonction du temps, sachant qu'il part d'une altitude $z_0 = 4000 \text{ m}$.

► Correction exemple n°1

Remarque : pour tous les exemples ci-dessous, on suppose avoir importé au préalable les bibliothèques **numpy** et **matplotlib** avec les alias classiques (np et plt).

1. L'altitude z_n est donnée par $z_n = n\Delta z$. L'équation différentielle porte ici sur l'altitude et non plus sur le temps, mais le schéma de résolution reste le même. Le schéma explicite d'Euler s'écrit :

$$\frac{P_{n+1} - P_n}{\Delta z} = -P_n \frac{Mg}{R(T_0 - \alpha z_n)} \quad \text{soit} \quad P_{n+1} = P_n \left(1 - \frac{Mg\Delta z}{R(T_0 - \alpha z_n)} \right)$$

2. Une proposition d'implémentation est donnée ci-dessous.

```

1 M = 29e-3 # en kg . mol -1
2 R = 8.3 # en J . mol -1. K -1
3 T0 = 288 # en K
4 alpha = 6e -3 # en K . m -1
5 g = 9.81 # en m . s -2
6 dz = 1 # pas de 1 m
7 N = 11000 # nbre de pas d ' espace : de 0 à 11000 m
8
9 z = [n*dz for n in range (N)] # altitude , en m
10 P = [None for n in range (N)] # pression , en Pa
11 P[0] = 1e5 # 1 bar au niveau du sol
12
13 for n in range (N-1) :
14     P [n+1] = (1 - (M*g*dz)/(R*(T0 - alpha*z [n]))) * P [n]
```

Soyez vigilants sur les nombreuses parenthèses, cela peut vous jouer des tours! Après implémentation, vous pouvez réinvestir la partie représentation graphique donnée dans la partie II. pour vérifier la cohérence de vos résultats.

► Correction exemple n°2

1. De la même manière que précédemment, on trouve :

$$\frac{h_{n+1} - h_n}{\Delta t} = -\alpha\sqrt{2gh_n} \quad \text{soit} \quad h_{n+1} = h_n - \alpha\Delta t\sqrt{2gh_n}$$

2. Devoir résoudre "jusqu'à" la vidange complète laisse entendre qu'il faut privilégier une implémentation avec une boucle **while** ("tant que..."). Une proposition d'implémentation pourrait être la suivante :

```

1 g = 9.81 # en m.s-2
2 alpha = 1e-3 # sans unité
3
4 # Initialisation des valeurs courantes :
5 tn = 0
6 hn = 1.5 # hauteur initiale 1m50
7 dt = 0.01 # pas de temps , en s
8
9 t = [tn] # listes ne contenant que la valeur initiale
10 h = [hn]
```

```
11  
12 while hn > 0:  
13     # Actualisation des valeurs courantes :  
14     tn += dt  
15     hn += - dt*alpha*np . sqrt (2*g*hn)  
16     # Nouveaux éléments ajoutés à chaque pas de temps :  
17     t.append(tn)  
18     h.append(hn)
```

► Correction exemple n°3

1. De la même manière que précédemment, on trouve :

$$\frac{v_{n+1} - v_n}{\Delta t} + \frac{k}{m}v_n^2 = g \quad \text{soit} \quad v_{n+1} = v_n + \Delta t \left(g - \frac{k}{m}v_n^2 \right)$$

2. Je vous laisse essayer d'implémenter ce schéma en Python, selon les méthodes d'implémentation présentées ci-dessus.