TP n°6 – Méthode d'Euler

Objectifs du TP

Vous avez découvert cette année ce qu'était une équation différentielle, et des méthodes de résolution explicites applicables dans certains cas très particuliers (équations différentielles linéaires d'ordre 1 ou d'ordre 2 à coefficients constants). Mais, comme vous l'avez constaté par la même occasion, la résolution d'une tel problème est souvent rendue possible par le fait que l'on se ramène d'une façon plus ou moins détournée au calcul d'une primitive. Or il n'existe pas de méthode générique pour déterminer explicitement une primitive d'une fonction continue, et "par conséquent" il n'existe pas de méthode générique pour résoudre explicitement et exactement une équation différentielle quelconque.

Ce dernier constat est fâcheux, au regard des nombreuses situations concrètes issues de la Physique, de la Chimie, des Sciences de l'Ingénieur où interviennent les équations différentielles. Cela étant, à défaut de les résoudre exactement, on peut souvent se contenter de les résoudre numériquement, c'est-à-dire d'en trouver des solutions approchées.

Partant de ce principe, le but de ce TP est de présenter une méthode de résolution numérique des équations différentielles : la méthode d'Euler. Celle-ci permettra donc de déterminer des solutions approchées de certaines équations différentielles, impossibles à résoudre par les méthodes classiques que vous avez rencontrées depuis le début de l'année.

Première Partie — La bibliothèque Matplotlib (Durée indicative : 45 minutes)

On aura souvent recours aux fonctions prédéfinies dans certaines bibliothèques logiciel. On peut citer **SciPy, NumPy** pour les calculs scientifiques, la manipulation des tableaux ..., ainsi que **Matplotlib** pour les représentations graphiques.
On propose ici de se familiariser avec cette dernière bibliothèque à partir d'exemples ; les précisions et détails seront à aller chercher sur les documentations en ligne.

Quelques graphiques avec Matplotlib

Exercice 1. — Représentation de la fonction cosinus

(1) Taper les lignes suivantes dans le fichier python en cours, puis l'exécuter :

```
import matplotlib.pyplot as pypl
import numpy as np
from math import pi
t =np.linspace(-pi,3*pi,1000)
y =np.cos(t)
pypl.plot(t,y)
pypl.show()
```

A partir de la courbe obtenue, réfléchir à la signification de chaque ligne.

(2) Dans les commandes de l'exercice précédent, insérer progressivement les lignes :

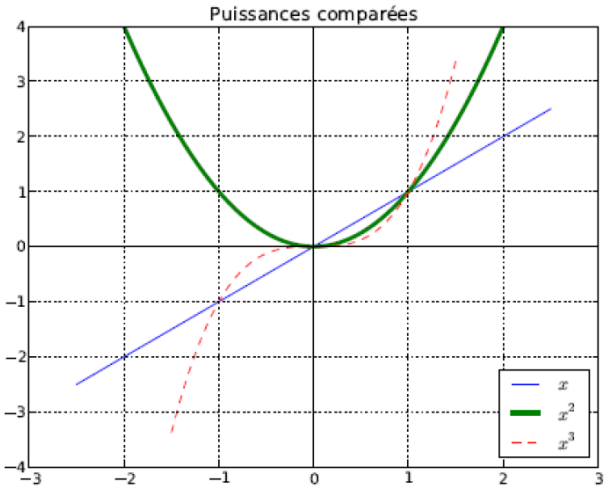
```
pypl.grid()
pypl.axhline(color='black')
pypl.axvline(color='black')
ys =np.sin(t)
pypl.plot(t,ys)
pypl.legend(['cosinus','sinus'],loc='upper left')
```

Observer le résultat produit par chaque commande.

Remarques :

- Il est possible de sauvegarder dans divers formats (png, pdf, eps ...)
  pypl.savefig() sauvegarde le graphique. Par exemple **pypl.savefig("mongraphe.png")** sauve sous le nom «mongraphe.png» le graphique. Par défaut le format est png. Il est possible d'augmenter la résolution, la couleur de fond, l'orientation, la taille (A0, A1, lettertype, etc) et aussi le format de l'image. Si aucun format n'est spécifié, le format est celui de l'extension dans « nomfigure.ext »
- La commande **plt.clf()** efface la fenêtre graphique.

Exercice 2. — **Application** Essayer de produire un fichier pdf dont le contenu ressemble à celui ci-dessous :



Deuxième Partie — Euler (en dimension 1)

Exercice 3. — **(Pas trop dur pour commencer)**. Résolution du problème de Cauchy :

$$\forall\, t \in [0;\ 1]\,, \quad y'(t) = y(t) \quad \text{et} \quad y\,(0) = 1$$

1) Construire un programme demandant à l'utilisateur un entier n, et construisant la liste des abscisses t_0, t_1,..., t_n qui partagent l'intervalle $[0, 1]$ en n sous-intervalles de longueur $h = 1/n$. On pourra noter Val_t la liste de ces valeurs.

2) Construire la liste "Val_y" des valeurs y_k obtenues grâce à la méthode d'Euler décrite en cours.

3) Faire apparaître sur un graphique le nuage des points (t_k, y_k) ainsi obtenus, puis la ligne polygonale joignant ces points.

4) Facultatif mais non superflu : pendant que vous y êtes, vous pouvez également faire apparaître sur le même graphique la solution exacte du problème considéré dans cet exercice, puisque vous la connaissez. Cela vous permettra de mesurer la "distance" entre la solution exacte et la solution approchée obtenue grâce à la méthode d'Euler.

Exercice 4. — **(Décroissance radioactive)**. Il s'agit ici de résoudre le problème de Cauchy :

$$\forall\, t \in [0;\ T]\,, \quad \frac{dN\,(t)}{dt} + \frac{N\,(t)}{\tau} = 0 \quad \text{et} \quad N\,(0) = N_0 \ (\text{avec } N_0 \in \mathbb{R}^*).$$

1/ En vous inspirant de l'exercice précédent, écrire un programme résolvant ce problème par la méthode d'Euler. On pourra prendre $N_0 = 1000$ atomes de carbone 14, pour lequel $\tau = 6\,500$ ans, et on représentera l'évolution de cette population sur une durée T de 20 000 ans.

2/ Des mesures réalisées en 2018 sur un arbre fossilisé montrent qu'il ne contient plus que 20% du 14C qu'il contenait au moment de sa mort. Estimer l'année de la mort de cet arbre.

Exercice 5. — (Evolution d'une tension). Adapter le programme de l'exercice précédent à l'étude de l'évolution de la tension $u(t)$ lors de la charge d'un condensateur à travers un résistor, sous une tension $E = 10\ V$. La tension $u(t)$ est dans ce cas donnée par :

$$\forall\, t \in [0;\ T]\,, \quad \frac{du\,(t)}{dt} + \frac{u\,(t)}{RC} = \frac{E}{RC}$$

On prendra : $u\,(0) = 2,0\ V$; $R = 1,0.10^4\Omega$ et $C = 1,0.10^{-6}\ F$.

Exercice 6. — (Ski de vitesse).

Au cours d'une épreuve de kilomètre lancé, le skieur, doté d'un équipement spécial qui le fait plus ressembler à un astronaute qu'à un sportif (voir ci-contre), prend de la vitesse en descendant une forte pente dans une trajectoire rectiligne.

L'application de la relation fondamentale de la dynamique prenant en compte le poids du skieur, la force de réaction de la piste et le frottement de l'air conduit alors à l'équation différentielle suivante :

$$\frac{dv\,(t)}{dt} = g\ \sin\alpha - \frac{k}{m}\ v^2(t)$$

avec :

- ☞ $g = 9,81\text{m.s}^{-2}$
- ☞ $\alpha = 30^0$
- ☞ $m = 70\text{kg}$
- ☞ $k = \dfrac{1}{2}\ \rho\ S\ Cx$ où :
  - ☛ $\rho = 1,008\text{kg.m}^{-3}$ est la masse volumique de l'air dans les conditions de l'épreuve ($P = 0,8\text{bar}$ et $T = 273\text{K}$).
  - ☛ $S = 0,55\text{m}^2$ est la surface de contact (c'est la surface projetée du coureur en contact avec l'air) ;
  - ☛ $Cx = 0,35$ est le coefficient de pénétration dans l'air du skieur.

Donner une valeur approchée de la vitesse du skieur au bout de 20 secondes (par exemple) de descente sachant que sa vitesse initiale est nulle.

Exercice 7. — (Généralisation).

Coder une fonction **euler(f,a,b,n,y0)** qui reçoit comme paramètres ceux que vous pouvez imaginer facilement d'après les paragraphes précédents, et qui construit la solution approchée de l'équation différentielle : $y'(t) = f\,(t, y(t))$.

Appliquer cette fonction aux 4 exercices précédents... et vérifier qu'elle fonctionne !

Troisième Partie — Euler (en dimension 2)

Système de Lotka-Volterra (prédateurs-proies)

Le modèle concerne deux populations dont les effectifs au temps t sont respectivement notés $x(t)$ et $y(t)$, la seconde (les prédateurs) se nourrissant de la première (les proies). On fait les hypothèses suivantes (forcément simplificatrices !) :

- Les proies $x(t)$ disposent de nourriture en quantité illimitée, seuls les prédateurs $y(t)$ s'opposent à leur croissance et en l'absence de prédateurs la population des proies a une croissance exponentielle.
- Le nombre de prédateurs est limité par la quantité de proies dont ils disposent pour se nourrir et en l'absence de proies, la population des prédateurs a une décroissance exponentielle.
- Le nombre de rencontres entre proies et prédateurs et à la fois proportionnel à $x(t)$ et $y(t)$, donc proportionnel au produit $x(t)y(t)$.
- Le taux de disparition des proies ainsi que le taux de croissance des prédateurs dûs à ces rencontres sont l'un et l'autre proportionnels au nombre de rencontres entres les deux populations.

Ce qui conduit au modèle suivant :

$$\textbf{(P)} : \quad \forall\, t \in [0;\, T]\,, \quad \begin{cases} x'(t) = \alpha_1 x\,(t) - \beta_1 x(t) y(t) \\ \\ y'(t) = -\alpha_2 y(t) + \beta_2 x(t) y(t) \end{cases} \quad \text{et} \quad \begin{cases} x\,(0) = x_0 \\ \\ y\,(0) = y_0 \end{cases} \quad \text{(populations initiales)}$$

où $\alpha_1 > 0$ est le taux de natalité (naturel) des proies, $\alpha_2 > 0$ le taux de mortalité (naturel) des prédateurs, $\beta_1 > 0$ et $\beta_2 > 0$ des coefficients d'interaction entre les deux populations.

Exercice 8. — A l'aide de la méthode décrite page 1, résoudre le problème (P) en prenant comme valeurs numériques :

- $\alpha_1 = 0,8\,;\ \alpha_2 = 0,2\,;\ \beta = 0,8\,;\ \beta_2 = 0,5\,;$
- T entre 10 et 100 (par exemple) ;
- $x_0 = 5$ et $y_0 = 3$, en ayant présent à l'esprit que ces deux valeurs ne correspondent pas exactement aux populations initiales, mais donnent plutôt le ratio initial entre nombre de proies et nombre de prédateurs. Rien ne vous empêche de prendre 5000 et 3000 à la place de 5 et 3... sauf la capacité de calcul de l'ordinateur, et la taille de l'erreur (incomparablement plus importante dans ce cas).

Puis observez les différences obtenues :

- En faisant augmenter n ;
- En modifiant les paramètres α_1, α_2...
- Les populations initiales.