Licence 1, UE 1.2 — Institut Villebon-Charpak, Orsay
Mathématiques — 2016-2017

Partie informatique

1 Exploitation des données

On repart du tableau de valeurs de la question 9 de la partie physique mesurant la pseudo-période du cylindre immergé dans l'eau. Deux données intéressantes pour quantifier la pertinence d'une mesure sont la moyenne et l'écart type de l'échantillon de valeurs. L'écart type représente "l'écart par rapport à la moyenne". En d'autres termes plus il est grand, plus les valeurs prennent des valeurs éloignées de l'échantillon. Plus il est faible plus l'échantillon est regroupé autour de la moyenne. Soit N un entier naturel, en notant $(x_1, \ldots, x_N)$ un tableau de N valeurs, la moyenne est donnée par

$$\overline{x} = \frac{1}{N}\sum_{i=1}^{N} x_i$$

et l'écart type est donné par

$$\sigma_x = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \overline{x})^2}$$

1. Importer toutes les fonctions du module `numpy` en tapant `from numpy import *`. Importer le module `matplotlib.pyplot` en le renommant `plt` en tapant `import matplotlib.pyplot as plt`.
2. Créer le tableau de valeurs de la question 9 de la partie physique à l'aide d'une liste. Pour créer une liste de nombres contenant par exemple les nombres $1, 2, 3$, on écrit en python `L=[1,2,3]`. Vous pourrez pour les questions suivantes isoler la $i$ème valeur de la liste à l'aide de la commande `L[i-1]` (attention la première valeur de la liste est donnée par `L[0]`).
3. Ecrire un programme calculant et affichant la moyenne des valeurs de ce tableau à l'aide d'une boucle `for`.
4. Ecrire ensuite un programme calculant et affichant l'écart type de ce tableau puis l'incertitude type donnée par $\frac{\sigma_x}{\sqrt{N}}$.

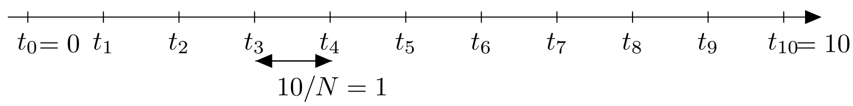
2 Approcher une équation différentielle

Nous avons exhibé dans la partie physique l'expression de la solution d'une équation différentielle du second ordre. Cependant pour la majorité des équations différentielles, il est impossible de déterminer l'expression de cette solution. A titre d'exemple, des équations "différentielles" régissent l'évolution de la température dans l'atmosphère terrestre. Leur complexité ne permet cependant pas de déterminer l'expression de la température. Ainsi, pour mettre à votre disposition des bulletins météorologiques, ces équations sont approchées par d'autres équations qui elles sont résolvables. L'objectif de cette partie est de comprendre comment on peut approcher une équation différentielle sur l'exemple de

$$\forall t \in \mathbb{R}, \quad y'(t) - y(t) = 0. \tag{1}$$

5. Montrer que $y : t \mapsto e^t$ est solution de cette équation.

Plaçons-nous sur l'intervalle $[0, 10]$ et divisons cet intervalle en $N+1$ points notés t_n pour n dans $[0, \dots, N]$. En voici une illustration pour $N = 10$.



On a alors

$$\forall n \in \{0, \dots, N\}, \quad t_n = \frac{10n}{N}$$

(en particulier $t_0 = 0$, $t_N = 10$) et l'écart entre deux points est $\Delta t = 1/N$. L'idée est alors de déterminer un ensemble $\{y_0, y_1, \dots, y_n, \dots, y_N\}$ de valeurs approchées de y en chacun des t_n. Vous allez me dire qu'on perd de l'information puisque d'une infinité de valeurs $y(t)$ on passe à un ensemble fini de valeurs. Certes mais l'idée est de faire augmenter n afin qu'il y ait de plus en plus de points dans l'ensemble et ainsi de précision.

Comment alors calculer les y_n ? L'idée est la suivante : sachant que la dérivée d'une fonction f dérivable en un point x_0 est définie comme limite du taux d'accroissement

$$\frac{f(x) - f(x_0)}{x - x_0} \underset{x \to x_0}{\longrightarrow} f'(x_0),$$

$y'(t_n)$ sera bien approximé par $\dfrac{y(t_{n+1}) - y(t_n)}{t_{n+1} - t_n}$ quand n augmentera. Comme pour tout n, $t_{n+1} - t_n = \Delta t$, l'équation $y' - y = 0$ peut alors s'approcher par

$$\forall n \in \{0, \dots, N\}, \quad \frac{y_{n+1} - y_n}{\Delta t} - y_n = 0,$$

où y_n est l'approximation de $y(t_n)$ ce qui se récrit

$$\forall n \in [0, \dots, N], \quad y_{n+1} = y_n + \Delta t y_n = (1 + \Delta t) y_n.$$

La méthode d'approximation obtenue est appelée ***méthode d'Euler***. Remarquons que si on connait y_0 alors la relation donne y_1 (car $y_1 = (1+\Delta t)y_0$) puis y_2 (car $y_2 = (1+\Delta t)y_1$) puis y_3 puis tous les y_n.

6. **(Facultatif)** Créer les deux tableaux suivants :

$$T_1 = [0, 1, 2, \dots, 8, 9, 10], \quad T_2 = [0, 0.1, 0.2, \dots, 9.8, 9.9, 10]$$

grâce à la fonction `arange` de `numpy` (à vous de voir comment elle s'utilise avec l'aide python ou sur internet).. Ces tableaux correspondent respectivement au tableau $[t_0, \dots, t_N]$ lorsque $N = 10$ et $N = 100$.

7. Ecrire une fonction `def=Euler(y0,N)` prenant en argument `y0` la condition initiale à $t = 0$ et `N` correspondant au N de l'énoncé. Cette fonction utilisera une boucle `for` pour placer dans un tableau de $N+1$ valeurs, les valeurs des y_n. Cette fonction retournera le tableau $[y_0, y_1, \ldots, y_N]$.
8. Exécuter la fonction pour la condition initiale $y_0 = 1$ et pour $N = 10$ puis 100. Vous créerez deux tableaux Y_1 et Y_2 correspondant aux deux tableaux retournés par la fonction `Euler`.
9. **(Facultatif)** Créer un tableau Y_3 des images de T_2 par la fonction exponentielle.
10. **(Facultatif)**Tracer à l'aide de la fonction `plot` sur un même graphique
    — Y_1 en fonction de T_1,
    — Y_2 en fonction de T_2,
    — Y_3 en fonction de T_2.

    Le troisième tracé sert à représenter la solution exacte afin de pouvoir lui comparer les deux solutions approchées. Commenter le résultat : que remarquez-vous lorsque le nombre N augmente ? Voici typiquement le genre d'allure que vous êtes censés obtenir, la courbe verte pleine étant la solution exacte de l'équation.