TP d'informatique n°18

Intégration numérique

L'objectif du TP est de programmer et tester différentes méthodes pour approcher numériquement une intégrale.

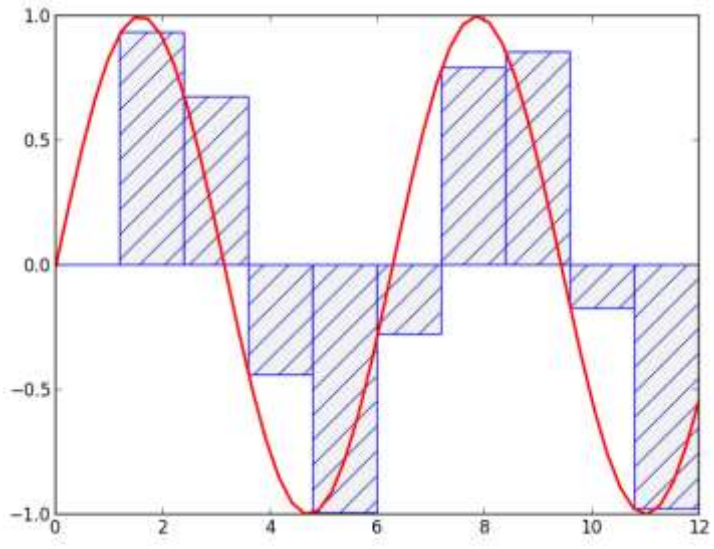
Les différentes bibliothèques utilisées seront :

- math pour les fonctions cos, exp, sin et log (ou numpy pour ces mêmes fonctions)
- scipy.integrate pour la fonction quad.
- time pour la fonction time.
- numpy pour les fonctions array, linspace et les fonctions usuelles chargé avec l'alias np.
- matplotlib.pyplot pour plot, show…… chargé avec l'alias plt.

I) Méthode des rectangles

Le théorème de Riemann dit que si f est une fonction continue sur un segment [a,b], alors on peut approcher l'aire située sous le graphe de f par la somme des aires des rectangles approchant f en n points uniformément répartis : $a_k = a + k\frac{b-a}{n}$ pour $k \in [\![0,n]\!]$:

$$R_n(f) = \frac{b-a}{n}\sum_{k=0}^{n-1} f\left(a + \frac{b-a}{n}k\right) \underset{n\to+\infty}{\longrightarrow} \int_a^b f(t)\,dt$$



Plus précisément, on montre que si f est de classe $\mathcal{C}^1$ sur [a,b] et si M_1 est un majorant de |f'|, alors : $\left|\int_{[a,b]} f - R_n(f)\right| \leqslant M_1 \frac{(b-a)^2}{2n}$

a) Ecrire une fonction prenant pour arguments une fonction f, deux bornes a et b et un nombre de pas n, et qui renvoie l'approximation $R_n(f)$ de l'intégrale de la fonction. La fonction commencera par :

```
def rectangles(f, a, b, n) :
    …
```

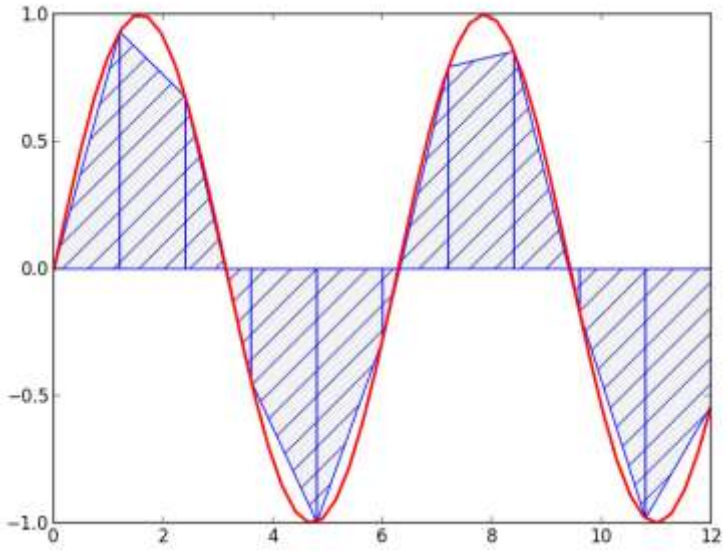
b) Tester la fonction précédente pour évaluer des intégrales connues :

- $t \mapsto t$ sur $[0,1]$
- $t \mapsto t^{10}$ sur $[0,1]$
- $t \mapsto \cos(t)$ sur $[0,\pi/2]$
- $t \mapsto e^t$ sur $[-3, 3]$

On commencera par définir en Python les 4 fonctions ; ensuite, on évaluera pour chacune de ces fonctions la différence entre la valeur exacte et la valeur approchée, pour différentes valeurs de n. On pourra enfin comparer au résultat fourni par la fonction quad de la bibliothèque scipy.integrate.

II) Méthode des trapèzes

Une autre idée pour donner une valeur approchée de $\int_a^b f(t)\,dt$ est d'approcher f en l'interpolant sur chaque intervalle $[a_k, a_{k+1}]$ par des fonctions affines (plutôt que des constantes comme dans le cas de la méthode des rectangles), ce qui conduit à considère l'aire des trapèzes $T_n(f) = \frac{b-a}{2n} \sum_{k=0}^{n-1} \left(f\left(a + \frac{b-a}{n} k\right) + f\left(a + \frac{b-a}{n}(k+1)\right)\right)$



a) (Question mathématique) **M**ontrer que : $T_n(f) = (R_n(f) + S_n(f)) / 2 = R_n(f) + \frac{b-a}{2n}(f(b) - f(a))$

On peut montrer que, si f est de classe $\mathcal{C}^2$ su [a,b] et si M_2 est un majorant de |f''|, alors :

$$\left| \int_{[a,b]} f - T_n(f) \right| \leqslant M_2 \frac{(b-a)^3}{12\,n^2}$$

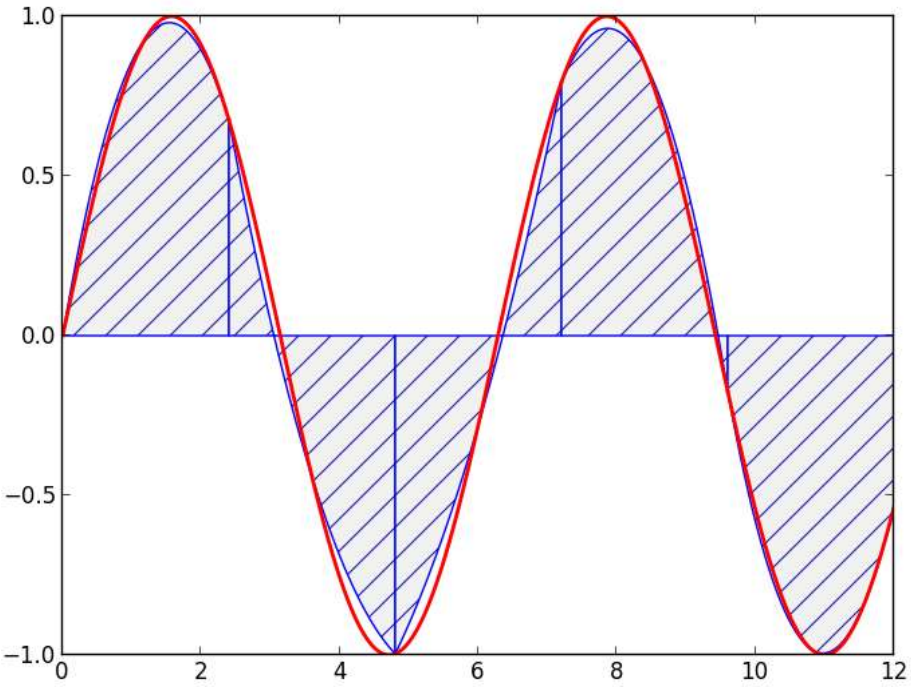
b) **E**crire une fonction prenant pour arguments une fonction f, deux bornes a et b et un nombre de pas n, et qui renvoie l'approximation $T_n(f)$ de l'intégrale de la fonction. La fonction commencera par :

```
def trapèzes(f, a, b, n) :
    …
```

III) Méthode de Simpson

Une dernière idée (pour ce TP…) pour donner une valeur approchée de $\int_a^b f(t)\,dt$ est d'approcher f sur chaque intervalle $[a_k, a_{k+1}]$ en l'interpolant par les valeurs de f en a_k , a_{k+1} et en $(a_k + a_{k+1}) / 2$. On approche donc f sur chaque intervalle par des fonctions polynomiales de degré $\leqslant 2$ et donc le graphe de f sur chaque intervalle $[a_k, a_{k+1}]$ par des paraboles.
On peut montrer que l'aire délimitée par ces paraboles est :

$$P_n(f) = \frac{b-a}{6\,n} \sum_{k=0}^{n-1} \left(f\left(a_k\right) + 4\, f\left(\frac{a_k + a_{k+1}}{2}\right) + f\left(a_{k+1}\right) \right) \quad \text{toujours avec} \quad a_k = a_0 + k\frac{b-a}{n}$$



On peut montrer également que, si f est de classe $\mathcal{C}^3$ su [a,b] et si M_3 est un majorant de $|f^{(3)}|$, alors : $\left| \int_{[a,b]} f - P_n(f) \right| \leqslant M_3 \frac{(b-a)^4}{192\,n^3}$

a) **E**crire une fonction prenant pour arguments une fonction f, deux bornes a et b et un nombre de pas n, et qui renvoie l'approximation $P_n(f)$ de l'intégrale de la fonction. La fonction commencera par :

```
def Simpson(f, a, b, n) :
    …
```

IV) Comparaison des méthodes

a) Comparer les qualités d'approximation et les temps de calcul nécessaires à l'évaluation de l'intégrale : $\int_0^{\frac{\pi}{2}} \cos(t)\,dt = 1$ pour les différentes méthodes et différentes valeurs de n.

- Rectangles :

n	10^2	10^4	10^6
Temps (secondes)			
$\lvert R_n - 1 \rvert$			

- Trapèzes :

n	10^2	10^4	10^6
Temps (secondes)			
$\lvert T_n - 1 \rvert$			

- Paraboles (Simpson) :

n	10^2	10^4	10^6
Temps (secondes)			
$\lvert P_n - 1 \rvert$			

Indication : pour chronométrer un calcul, on peut opérer ainsi :

```
t1 = time()
<mon_calcul>
t2 = time()
print(t2 – t1)
```

b) Calculer le nombre d'évaluations de f nécessaires pour calculer R_n , T_n avec la formule initiale, T_n avec la formule utilisant R_n, P_n directement, P_n en constatant que l'on peut l'exprimer en fonction de T_n