TP n°19 : Intégration numérique

L'objectif de ce TP est de (re)programmer quelques formules de quadrature et de comparer leur efficacité. On propose ensuite deux exercices qui permettent de mettre en application les méthodes d'intégration numérique pour résoudre des problèmes issus de la physique et de l'ingénierie.

1 Comparaison des méthodes de quadrature

On rappelle qu'on peut approcher l'intégrale d'une fonction f sur un intervalle $[a,b]$ en subdivisant régulièrement le segment $[a,b]$ en n sous-intervalles de taille $h = \dfrac{b-a}{n}$ et en remplaçant f sur chaque sous-segment par un polynôme interpolateur d'un degré ℓ. On rappelle aussi que pour une subdivision $a = \sigma_0 < \sigma_1 < \cdots < \sigma_n = b$, on a $\sigma_k = \sigma_{k-1} + h = a + kh$.

La méthode des rectangles à gauche, à droite et du point milieu correspond à une interpolation par une constante ($\ell = 0$) ; la méthode des trapèzes correspond à une interpolation par une droite affine ($\ell = 1$).

✔ Encore une fois, il faut absolument essayer de *retrouver* les formules du cours, au brouillon, dessin à l'appui.

La méthode de SIMPSON, qui n'est pas au programme, correspond à une interpolation par une parabole ($\ell = 2$). On a alors :

$$\begin{aligned}\int_a^b f(t)\,\mathrm{d}t &\approx h\sum_{k=0}^{n-1}\left(\frac{1}{6}f(a_k)+\frac{2}{3}f\left(\frac{a_k+a_{k+1}}{2}\right)+\frac{1}{6}f(a_{k+1})\right)\\ &\approx \frac{h}{6}\sum_{k=0}^{n-1}\left(f(a_k)+4f(a_k+h/2)+f(a_k+h)\right)\\ &\approx \frac{h}{3}\left(\frac{f(a)+f(b)}{2}+2f(a+h/2)+\sum_{k=1}^{n-1}\left(f(a_k)+2f(a_k+h/2)\right)\right)\end{aligned}$$

Enfin, la bibliothèque `scipy` fournit une fonction `quad` permettant le calcul approché d'une intégrale. Elle se trouve dans le sous-module `intergrate`. Il faudra donc, pour l'utiliser, ajouter :

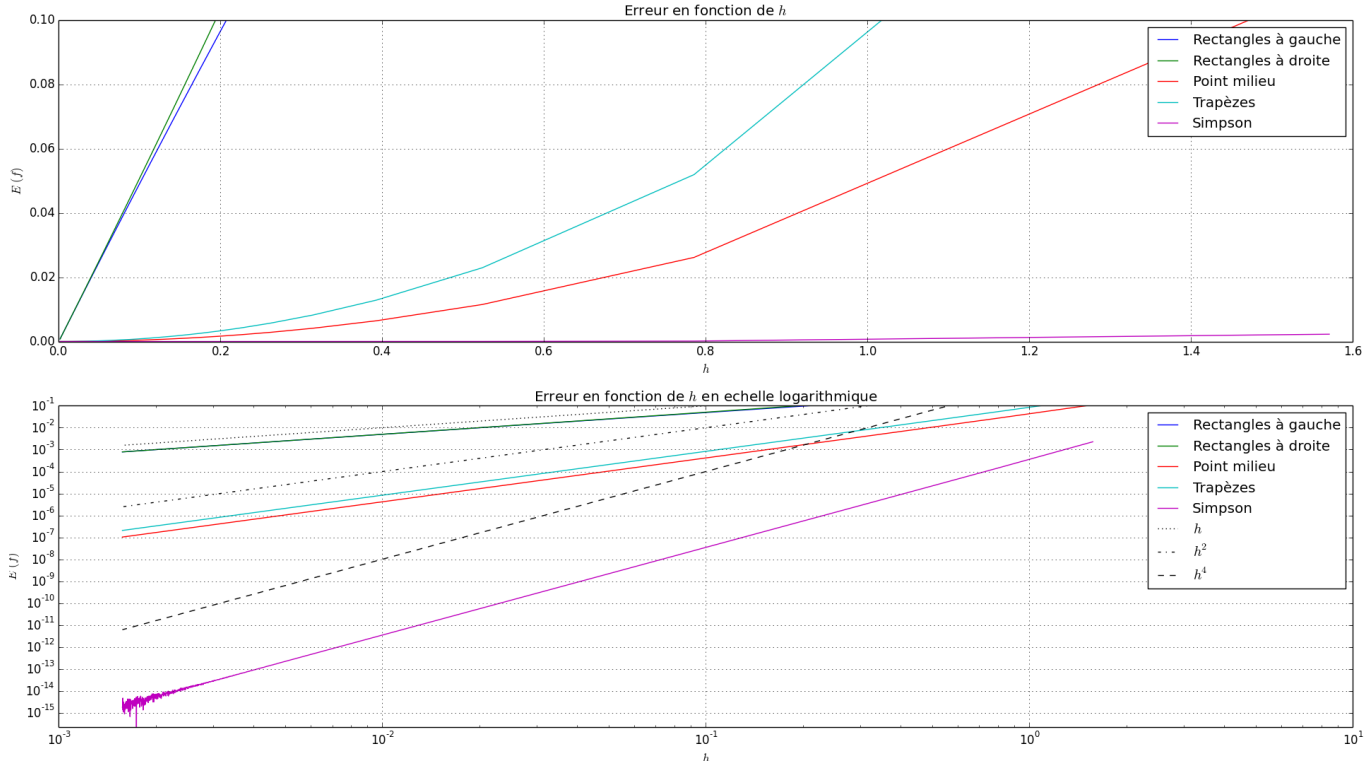
PYTHON
```
import numpy as np
import matplotlib.pyplot as plt
import scipy.integrate as itg
```

Allez voir la documentation pour son fonctionnement : `help(itg.quad)` ou bien

http://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.quad.html

1. Implémenter en PYTHON les fonctions calculant une approximation de $\int_a^b f(t)\,\mathrm{d}t$ qui prennent en arguments une fonction `f`, les bornes de l'intervalle `a` et `b` et la taille `n` de la subdivision pour les méthodes :
   - des rectangles à gauche
   - des rectangles à droite
   - du point milieu
   - des trapèzes
   - de SIMPSON
2. Tester les différentes méthodes pour $\int_0^{\pi/2} \cos t\,\mathrm{d}t$ et comparer avec `itg.quad`.
3. Tracer sur un même graphe l'erreur correspondant à chaque formule de quadrature en fonction de $h = \dfrac{b-a}{n}$.
4. Faire ensuite un tracé logarithmique pour observer l'ordre des différentes méthodes.

Exemple de résultat :



2 Accélération d'une suspension de moto

Par la suite on pourra travailler avec des listes PYTHON ou bien des tableaux NUMPY. Lorsque l'on dit « la liste des ... » cela peut donc, au choix, désigner une liste PYTHON ou un tableau NUMPY contenant les valeurs indiquées.

5. Le fichier `acquisition_moto.txt` contient l'acquisition d'un accéléromètre placé sur la suspension d'une maquette de moto simulant le passage d'une marche (trottoir, etc.). Chaque ligne contient un flottant correspondant au temps (en secondes) et à l'accélération (en $g = 9{,}81\ \mathrm{m{\cdot}s^{-2}}$), séparés par une tabulation.
6. Écrire une fonction `lire_fichier(nom_fichier)` renvoyant le couple `(t, acc)` où `t` est la liste des temps et `acc` la liste des accélérations correspondantes à partir du fichier dont le nom se trouve au chemin d'accès `nom_fichier`.
7. Écrire une fonction `position(nom_fichier)` qui renvoie le triplet `(t, vit, pos)` où `t` est la liste des temps, et `vit` et `pos`, les listes des vitesses et des positions, respectivement, correspondant aux temps et accélérations contenus dans les listes `t` et `acc`.

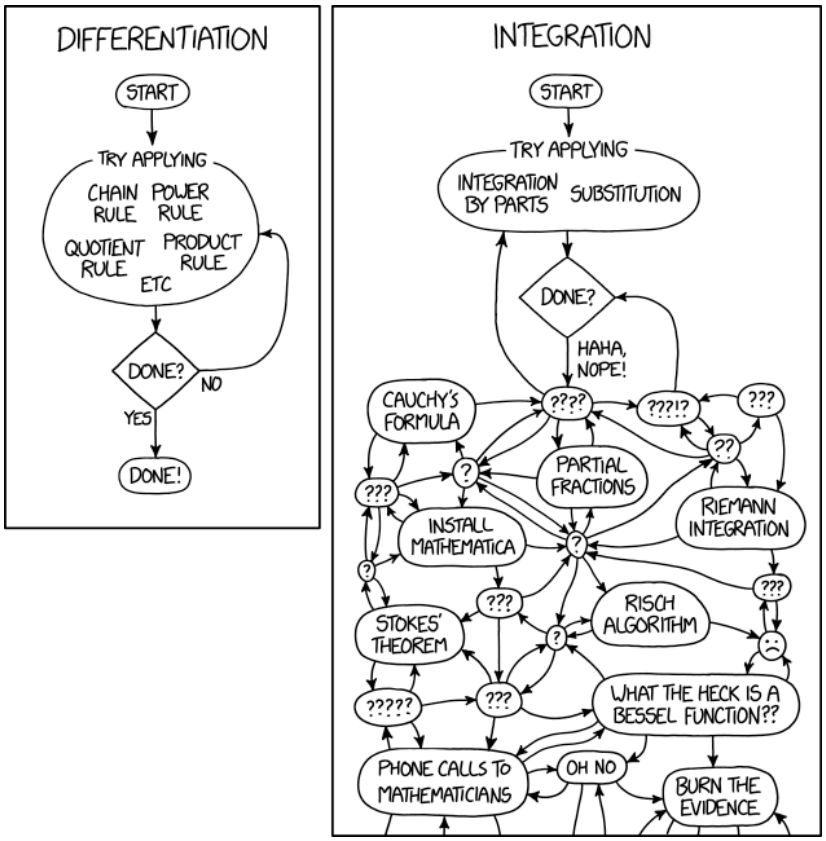
Les vitesses et positions seront calculées par intégration numérique.

8. Tracer la courbe de position en fonction du temps et commenter. On pourra comparer le résultat obtenu avec différentes méthodes d'intégration numérique et avec une solution théorique donnée par

$$x(t) = v_f\left(1 - \frac{\mathrm{e}^{-z\omega_0 t}}{\sqrt{1-z^2}}\sin\left(\omega_0\sqrt{1-z^2}\,t - \arctan\left(-\frac{\sqrt{1-z^2}}{z}\right)\right)\right)$$

où $v_f = 0{,}137\ \mathrm{m/s}$, $z = 0{,}22$, $\omega_0 = 8\ \mathrm{rad/s}$.

9. Résoudre le problème (étalonnage de l'accéléromètre ?) en appliquant un offset sur l'accélération de l'ordre de 0,073.

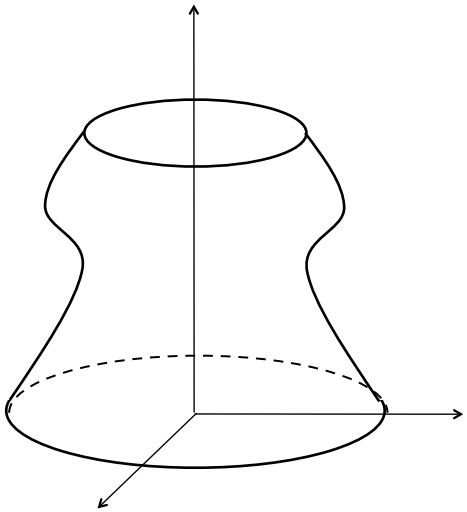


https://xkcd.com/644/

3 Moment d'inertie d'une pièce de révolution

Le moment d'inertie d'un solide S axisymétrique par rapport à son axe de révolution (Oz), de masse volumique ρ, de hauteur$H$ et dont le contour est décrit par une fonction $R(z)$, est donné par l'intégrale :

$$I_{Oz} = \int_0^{2\pi}\int_0^{H}\int_0^{R(z)} \rho \,.\, r^2 \,.\, r \,.\, dr \,.\, dz \,.\, d\theta = 2\pi \,.\, \rho \,.\, \int_0^{H}\int_0^{R(z)} r^3 \,.\, dr \,.\, dz = \frac{\pi \,.\, \rho}{2} \int_0^{H} R(z)^4 \,.\, dz$$



Pour un cylindre de rayon R_0 :$R(z) = R_0$

Pour une sphère de rayon R_0 :$R(z) = \sqrt{{R_0}^2 - \left(z - R_0\right)^2}$et $H = 2R_0$

Pour un cône de petit rayon R_1et de grand rayon R_2 :$R(z) = R_1 + z \,.\, \frac{R_2 - R_1}{H}$

10. Écrire une fonction, prenant en argument le hauteur H, la fonction R, la masse volumique rho et le nombre n de points d'intégration qui renvoie la valeur du moment d'inertie I_{Oz}.
11. Vérifier le programme pour un cylindre en acier de rayon $100mm$ et de hauteur $400mm$. Pour l'acier $\rho = 7{,}8 \,.\, 10^3 kg \,.\, m^{-3}$.

    Le moment d'inertie d'un cylindre de masse M et de rayon R vaut : $I_{Oz} = M \,.\, R^2/2$
12. Faire de même pour une sphère en acier de rayon $100mm$.

Le moment d'inertie d'une sphère de masse M et de rayon R vaut : $I_{Oz} = 2.M \,.\, R^2/5$

13. Faire de même pour un cône de petit rayon $50mm$, de grand rayon $100mm$ et de hauteur $400mm$ en acier.

Le moment d'inertie d'un cône de petit rayon R_1et de grand rayon R_2est :$I_{Oz} = \frac{\pi \,.\, \rho \,.\, H}{10} \,.\, \frac{R_2^5 - R_1^5}{R_2 - R_1}$

4 Flux massique

On souhaite calculer le flux d'air massique total qui traverse une éolienne. Cette valeur est un indicateur de la puissance maximale récupérable. Le flux est défini par l'intégrale :

$$\phi = \int_0^{2\pi}\int_0^{R} F(r,\theta)\,.\,r\,.\,dr\,.\,d\theta = \int_0^{2\pi}\int_0^{R} v(H + r\,.\,sin\theta)\,.\,\rho(H + r\,.\,sin\theta)\,.\,r\,.\,dr\,.\,d\theta$$

Données :

$H = 120m$
hauteur du mât

$R = 30m$
longueur d'une pale

$v1 = 5m\,.\,s^{-1}$

$z1 = 9{,}54m$

$z0 = 0{,}04m$

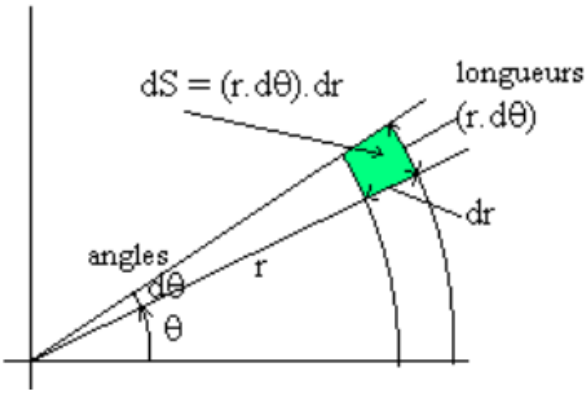
$a = 1{,}247015kg\,.\,m^{-3}$
$b = 1{,}04\,.\,10^{-4}m^{-1}$

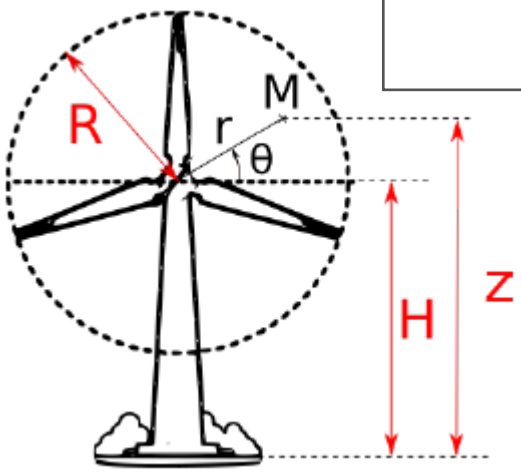
Le flux massique d'air dépend essentiellement de 2 phénomènes :

- la variation de la vitesse du vent en fonction de l'altitude :

$$v(z) = v_1 \bullet \frac{ln\frac{z}{z_0}}{ln\frac{z_1}{z_0}}$$

- la variation de la masse volumique en fonction de l'altitude (hauteur de l'air par rapport au sol). Le modèle ci-dessous a été établi à partir de relevés météorologiques :$\rho(z) = a\,.\,e^{-b.z}$





En coordonnées polaires : $dS = r\,.\,dr\,.\,d\theta$

14. Écrire deux fonctions, prenant en argument z, qui renvoient les valeurs de $v(z)$ et $\rho(z)$.
15. Écrire une fonction, prenant en arguments (r,θ) qui renvoie la valeur de $F(r,\theta)$ et utilisant les fonctions précédentes.
16. Écrire une fonction, prenant en arguments la longueur de pale R, la hauteur de mat H et le nombre n de subdivisions des intervalles d'intégration et qui renvoie la valeur du flux massique. On pourra réfléchir à la manière dont on peut étendre la méthode des rectangles (ou tout autre méthode) à la dimension 2.
17. Comparer avec la fonction `dblquad` de la bibliothèque `scipy.integrate`. On pourra consulter la documentation pour comprendre son fonctionnement.