Calculs approchés d'intégrales

I. Notations

Le but de ce TD est d'étudier certaines méthodes de calcul numérique approché d'une intégrale.

Dans toute la suite, f désignera une fonction définie sur un intervalle $[a,b]$ $(a<b)$, à valeurs dans $\mathbb{R}$. On supposera f de classe $\mathscr{C}^n$, avec n suffisamment grand.

Pour tout $n\in\mathbb{N}^*$, on posera $h_n=\dfrac{b-a}{n}$, et, pour tout $k\in[\![0;n]\!]$, $a_k=a+k\dfrac{b-a}{n}$.

L'idée de base est de découper l'intégrale de f en n morceaux en écrivant

$$\int_a^b f(t)\,\mathrm{d}t=\sum_{k=0}^{n-1}\int_{a_k}^{a_{k+1}} f(t)\,\mathrm{d}t$$

puis d'approcher chaque intégrale $\displaystyle\int_{a_k}^{a_{k+1}} f(t)\,\mathrm{d}t$ en remplaçant f par une fonction polynomiale "proche" de f sur l'intervalle $[a_k,a_{k+1}]$.

II. Méthode des rectangles à gauche

Le principe de cette méthode consiste à remplacer f sur $[a_k,a_{k+1}]$ par une fonction constante (autrement dit, on approche f sur $[a,b]$ par une fonction en escalier). On prendra pour cette fonction constante la valeur de f en a_k (on aurait pu prendre la valeur en a_{k+1} ...).

Théorème 1:

On suppose f de classe $\mathscr{C}^1$ sur $[a,b]$. Pour tout $n\in\mathbb{N}^*$, posons $\mathrm{R}_n(f)=\dfrac{b-a}{n}\displaystyle\sum_{k=0}^{n-1} f(a_k)$.

Alors :

$$\left|\int_a^b f-\mathrm{R}_n(f)\right|\leqslant\frac{(b-a)^2}{2n}\left\|f'\right\|_\infty$$

☞ *Démonstration:*

$$\left|\int_{a_k}^{a_{k+1}} f(t)\,\mathrm{d}t-h_nf(a_k)\right|=\left|\int_{a_k}^{a_{k+1}}[f(t)-f(a_k)]\,\mathrm{d}t\right|\leqslant\int_{a_k}^{a_{k+1}}|f(t)-f(a_k)|\,\mathrm{d}t$$

puis d'après l'inégalité des accroissements finis : $|f(t)-f(a_k)|\leqslant|t-a_k|\left\|f'\right\|_\infty$. D'où :

$$\left|\int_{a_k}^{a_{k+1}} f(t)\,\mathrm{d}t-h_nf(a_k)\right|\leqslant\left\|f'\right\|_\infty\int_{a_k}^{a_{k+1}}(t-a_k)\,\mathrm{d}t=\left\|f'\right\|_\infty\frac{(a_{k+1}-a_k)^2}{2}=\left\|f'\right\|_\infty\frac{h_n^2}{2}$$

d'où en additionnant ces inégalités (et en utilisant l'inégalité triangulaire) :

$$\left|\int_a^b f-\mathrm{R}_n(f)\right|\leqslant\sum_{k=0}^{n-1}\left\|f'\right\|_\infty\frac{h_n^2}{2}=\frac{(b-a)^2}{2n}\left\|f'\right\|_\infty$$

1ère fonction

Écrire une fonction `Rectangle(f,a,b,n)` qui calcule $\mathrm{R}_n(f)$. Tester cette procédure avec la fonction $f\mapsto\dfrac{1}{t}$ sur $[1,2]$, et comparer avec la valeur exacte $(\ln 2)$.

III. Méthode des points milieux

Le principe est le même que dans la méthode précédente, mais on considère ici la valeur de f au point $m_k=\dfrac{a_k+a_{k+1}}{2}$. On approche donc $\displaystyle\int_a^b f$ par $\mathrm{M}_n(f)=\frac{b-a}{n}\sum_{k=0}^{n-1} f(m_k)$.

Théorème 2:

On suppose f de classe $\mathscr{C}^2$ sur $[a,b]$. Pour tout $n\in\mathbb{N}^*$, posons $M_n(f)=\dfrac{b-a}{n}\displaystyle\sum_{k=0}^{n-1} f\left(\frac{a_k+a_{k+1}}{2}\right)$.

Alors :

$$\left|\int_a^b f - M_n(f)\right| \leqslant \frac{(b-a)^3}{24n^2}\left\|f''\right\|_\infty$$

☞ *Démonstration:*

- On commence par démontrer le lemme suivant :
  Si g est de classe $\mathscr{C}^3$ sur $[a,b]$, il existe $c\in]a,b[$ tel que
  $$g(b)=g(a)+(b-a)g'\left(\frac{a+b}{2}\right)+\frac{(b-a)^3}{24}g'''(c)$$
  Dém : Soit A le réel tel que $g(b)=g(a)+(b-a)g'\left(\frac{a+b}{2}\right)+\frac{(b-a)^3}{24}A$ (A existe, il est solution d'une simple équation de degré 1). On considère alors $\varphi : \left\{\begin{array}{ccc}[a,b] & \longrightarrow & \mathbb{R}\\ x & \longmapsto & g(x)-g(a)-(x-a)g'\left(\frac{a+x}{2}\right)-\frac{(x-a)^3}{24}A\end{array}\right.$
  On a alors $\varphi'(x)=f'(x)-f'\left(\frac{a+x}{2}\right)-\frac{x-a}{2}f''\left(\frac{a+x}{2}\right)-\frac{(x-a)^2}{8}A$
  Puisque $\varphi(a)=\varphi(b)(=0)$, il existe $c\in]a,b[$ tel que $\varphi'(c)=0$ (Rolle), ce qui s'écrit
  $$f'(c)-f'\left(\frac{a+c}{2}\right)-\frac{c-a}{2}f''\left(\frac{a+c}{2}\right)=\frac{(c-a)^2}{8}A$$
  La formule de Taylor-Lagrange, appliquée à f' (qui est de classe $\mathscr{C}^2$) entre c et $\frac{a+c}{2}$ donne alors l'existence de d tel que
  $$f'(c)-f'\left(\frac{a+c}{2}\right)-\frac{c-a}{2}f''\left(\frac{a+c}{2}\right)=\frac{1}{2}\cdot\left(\frac{c-a}{2}\right)^2 f'''(d)$$
  ce qui donne $A=f'''(d)$.
- On applique alors le lemme précédent à une primitive de f sur chaque intervalle $[a_k,a_{k+1}]$:
  $$\exists c_k\in]a_k,a_{k+1}[\text{ tq }\int_{a_k}^{a_{k+1}} f(t)\,dt=(a_{k+1}-a_k)f\left(\frac{a_k+a_{k+1}}{2}\right)+\frac{(a_{k+1}-a_k)^3}{24}f''(c_k)$$
  donc
  $$\left|\int_{a_k}^{a_{k+1}} f(t)\,dt-h_n f\left(\frac{a_k+a_{k+1}}{2}\right)\right|\leqslant\frac{(b-a)^3}{24n^3}\left\|f''\right\|_\infty$$
  et on obtient le résultat voulu en additionnant ces inégalités.

2ème fonction

Écrire une fonction `Milieux(f,a,b,n)` qui calcule $M_n(f)$.
Comparer avec la précédente.

IV. Méthode des trapèzes

Le principe consiste ici à approcher f sur $[a_k,a_{k+1}]$ par la fonction affine φ qui coïncide avec f en a_k et a_{k+1}. Il est facile de montrer que $\displaystyle\int_{a_k}^{a_{k+1}}\varphi(t)\,dt$ est égale à $\dfrac{f(a_k)+f(a_{k+1})}{2}$ (faire un dessin, aire d'un trapèze...).

On approche donc $\displaystyle\int_a^b f$ par $T_n(f)=\frac{b-a}{n}\sum_{k=0}^{n-1}\frac{f(a_k)+f(a_{k+1})}{2}$. Un calcul simple montre alors que :

$$T_n(f)=\frac{b-a}{n}\left[\frac{f(a)+f(b)}{2}+\sum_{k=1}^{n-1}f(a_k)\right]$$

Théorème 3:

On suppose f de classe $\mathscr{C}^2$ sur $[a,b]$. Pour tout $n\in\mathbb{N}^*$, posons $T_n(f)=\dfrac{b-a}{n}\displaystyle\sum_{k=0}^{n-1}\frac{f(a_k)+f(a_{k+1})}{2}$.

Alors :

$$\left|\int_a^b f - T_n(f)\right| \leqslant \frac{(b-a)^3}{12n^2}\left\|f''\right\|_\infty$$

☞ *Démonstration:*

- On commence par démontrer le lemme suivant :
  Si g est de classe $\mathcal{C}^3$ sur $[a,b]$, il existe $c\in]a,b[$ tel que
  $$g(b)=g(a)+\frac{b-a}{2}\big(g'(a)+g'(b)\big)-\frac{(b-a)^3}{12}g'''(c)$$
  Dem : Soit A le réel tel que $g(b)-g(a)-\dfrac{b-a}{2}\big(g'(a)+g'(b)\big)=\dfrac{(b-a)^3}{12}A$
  (A existe, il est solution d'une simple équation de degré 1). On considère alors
  $\varphi:\left\{\begin{array}{ccc}[a,b]&\longrightarrow&\mathbb{R}\\ x&\longmapsto&g(x)-g(a)-\dfrac{x-a}{2}(g'(a)+g'(x))-A\dfrac{(x-a)^3}{12}\end{array}\right.$.
  En appliquant deux fois le th. de Rolle, on montre qu'il existe c tel que $\varphi''(c)=0$, ce qui donne le résultat voulu.
- Pour en déduire l'inégalité, on procède de la même manière que dans le théorème précédent.

3ème fonction

Écrire une fonction `Trapeze(f,a,b,n)` qui calcule $T_n(f)$.
Comparer avec les précédentes.

V. Méthode de Simpson

Le principe consiste ici à approcher f sur $[a_k,a_{k+1}]$ par la fonction polynomiale de degré 2 qui coïncide avec f aux points a_k, m_k et a_{k+1}.

Prop 1: Formule des trois niveaux :

Pour tout polynôme $P\in\mathbb{R}_3[X]$ et tous réels a,b, on a :

$$\int_a^b P(t)\,dt=\frac{b-a}{6}\left[P(a)+4P\left(\frac{a+b}{2}\right)+P(b)\right]$$

☞ *Démonstration:*

Le changement de variable $u=t-\dfrac{a+b}{2}$ donne $\displaystyle\int_a^b P(t)\,dt=\int_{(a-b)/2}^{(b-a)/2}P\left(u+\frac{a+b}{2}\right)du$. En posant $Q(u)=P\left(u+\frac{a+b}{2}\right)$ la formule à démontrer s'écrit

$$\int_{(a-b)/2}^{(b-a)/2}Q(u)\,du=\frac{b-a}{6}\left[Q\left(\frac{a-b}{2}\right)+4Q(0)+Q\left(\frac{b-a}{2}\right)\right]$$

ou encore, en posant $h=\dfrac{b-a}{2}$:

$$\forall Q\in\mathbb{R}_3[X],\ \int_{-h}^{h}Q(u)\,du=\frac{h}{3}\left(Q(-h)+4Q(0)+Q(h)\right).$$

Par linéarité, il suffit de la démontrer pour les polynômes $1,X,X^2,X^3$ puisqu'ils forment une base de $\mathbb{K}[X]$, et c'est immédiat.

En vertu de cette formule, si on approche f sur $[a_k,a_{k+1}]$ par la fonction polynomiale φ de degré 2 qui coïncide avec f aux points a_k, m_k et a_{k+1}, l'intégrale $\displaystyle\int_{a_k}^{a_{k+1}}f(t)\,dt$ sera approchée par $\displaystyle\int_{a_k}^{a_{k+1}}\varphi(t)\,dt=\frac{h_n}{6}(f(a_k)+4f(m_k)+f($

L'intégrale de f sur $[a,b]$ sera donc approchée par : $S_n(f)=\dfrac{b-a}{6n}\displaystyle\sum_{k=0}^{n-1}\left[f(a_k)+4f\left(\frac{a_k+a_{k+1}}{2}\right)+f(a_{k+1})\right]$.

La convergence de cette méthode est bien plus rapide que celle des précédentes méthodes, en vertu du théorème suivant :

Théorème 4:

On suppose f de classe $\mathscr{C}^4$ sur $[a,b]$. Pour tout $n \in \mathbb{N}^*$, posons $S_n(f) = \dfrac{b-a}{6n}\displaystyle\sum_{k=0}^{n-1}\left[f(a_k)+4f\left(\frac{a_k+a_{k+1}}{2}\right)+f(a_{k+1})\right]$. Alors :

$$\left|\int_a^b f - S_n(f)\right| \leqslant \frac{(b-a)^5}{720n^4}\left\|f^{(4)}\right\|_\infty .$$

(on peut en fait remplacer 720 par 2880, mais c'est plus difficile à démontrer)

☞ *Démonstration:*

On démontre le lemme suivant, par la même méthode que dans le théorème précédent :

Soit g de classe $\mathscr{C}^5$ sur $[a,b]$, à valeurs dans $\mathbb{R}$. Alors il existe $c \in]a,b[$ tel que :

$$g(b) = g(a) + \frac{b-a}{2}\big(g'(a)+g'(b)\big) - \frac{(b-a)^2}{12}\big(g''(b)-g''(a)\big) + \frac{(b-a)^5}{720}g^{(5)}(c).$$

et on conclut de la même façon.

4ème fonction

Écrire une fonction `Simpson(f,a,b,n)` qui calcule $S_n(f)$.
Comparer avec les précédentes.

Corrigé :

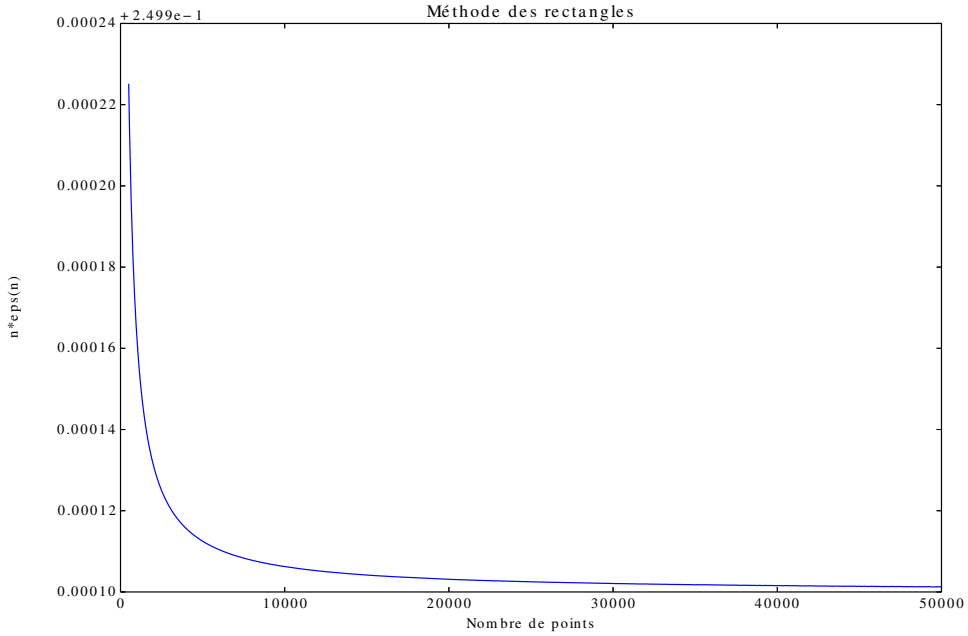
```python
import matplotlib.pyplot as plt
import math
from time import clock

# Cette version naîve de la méthode des rectangles fait bien trop
# de calculs inutiles
def Rectangle_Mauvais(f, a, b, n):
    S = 0
    for k in range(0,n):
        S = S + f(a + k*(b-a)/n)
    return S * (b-a)/n

# ici, le pas n'est calculé qu'une fois et on évite les multiplications
def Rectangle(f, a, b , n):
    pas = (b-a) / n
    S = 0
    ao = a
    for k in range(0,n):
        S += f(ao)
        ao += pas
    return pas * S

# on pourrait écrire pour profiter des fonctions Python:
# return pas * sum(f(x) for x in np.arange(a, b, pas))
# mais en essayant, c'est bien plus long!

def Milieux(f, a, b , n):
    pas = (b-a) / n
    S = 0
    ao = a + pas/2
    for k in range(0,n):
        S += f(ao)
        ao += pas
```

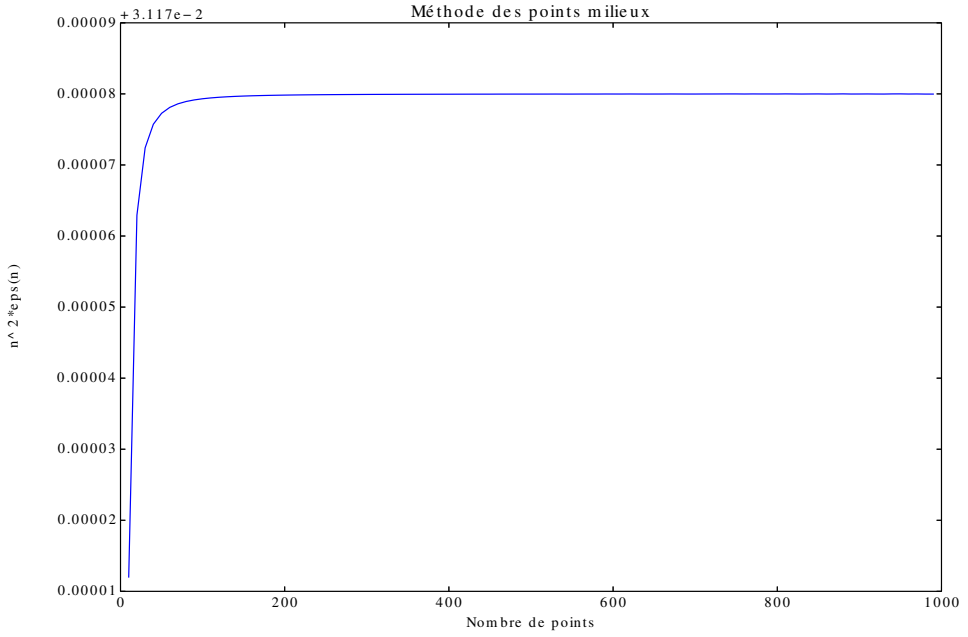
```
34      return pas * S
35
36  def Trapeze(f, a, b , n):
37      pas = (b-a) / n
38      S = 0
39      ao = a + pas
40      for k in range(1,n):
41          S += f(ao)
42          ao += pas
43      return pas * (S + 0.5*(f(a) + f(b)))
44
45  def Simpson(f, a, b, n):
46      pas = (b-a) / n
47      S1 = 0
48      S2 = 0
49      x = a
50      y = a + pas/2
51      for k in range(0, n):
52          S1 += f(x)
53          S2 += f(y)
54          x += pas
55          y += pas
56      return pas *( -f(a) + f(b) +2*S1 + 4*S2)/6
57
58  # Valeur 'exacte'
59  Valeur_Exacte = math.log(2)
60
61  # la méthode des rectangles d'abord
62  nb_iter = 2000000
63  debut = clock()
64  Ro = Rectangle_Mauvais(lambda t: 1/t, 1, 2, nb_iter)
65  Temps_Rectangle_Mauvais = clock() - debut
66  ERo = abs(Valeur_Exacte - Ro)
67
68  debut = clock()
69  R1 = Rectangle(lambda t: 1/t, 1, 2, nb_iter)
70  Temps_Rectangle = clock() - debut
71  ER1 = abs(Valeur_Exacte - R1)
72
73
74  print("{:.<30} {:.15f} {} {:.5e} {} {:.5f} \n".format("Rectangles_Moche "+ str(nb_iter)+" points:
75              Ro, "  Erreur ", ERo,"  Temps: ", Temps_Rectangle_Mauvais))
76
77  print("{:.<30} {:.15f} {} {:.5e} {} {:.5f} \n".format("Rectangles "+ str(nb_iter)+" points: ",\
78              R1, "  Erreur ", ER1,"  Temps: ", Temps_Rectangle))
79
80  # maintenant les méthodes en 1/n^2
81  nb_iter =  100000
82  debut = clock()
83  M1 = Milieux(lambda t: 1/t, 1, 2, nb_iter)
84  Temps_Milieux = clock() - debut
85  EM1 = abs(Valeur_Exacte - M1)
86
87  debut = clock()
88  T1 = Trapeze(lambda t: 1/t, 1, 2, nb_iter)
89  Temps_Trapeze = clock() - debut
90  ET1 = abs(Valeur_Exacte - T1)
91
92  print("{:.<30} {:.15f} {} {:.5e} {} {:.5f}\n".format("Trapezes "+ str(nb_iter)+" points: ", \
93                  T1,  "  Erreur ", ET1, "  Temps: ", Temps_Trapeze))
94  print("{:.<30} {:.15f} {} {:.5e} {} {:.5f} \n".format("Milieux "+ str(nb_iter)+" points: ",\
```

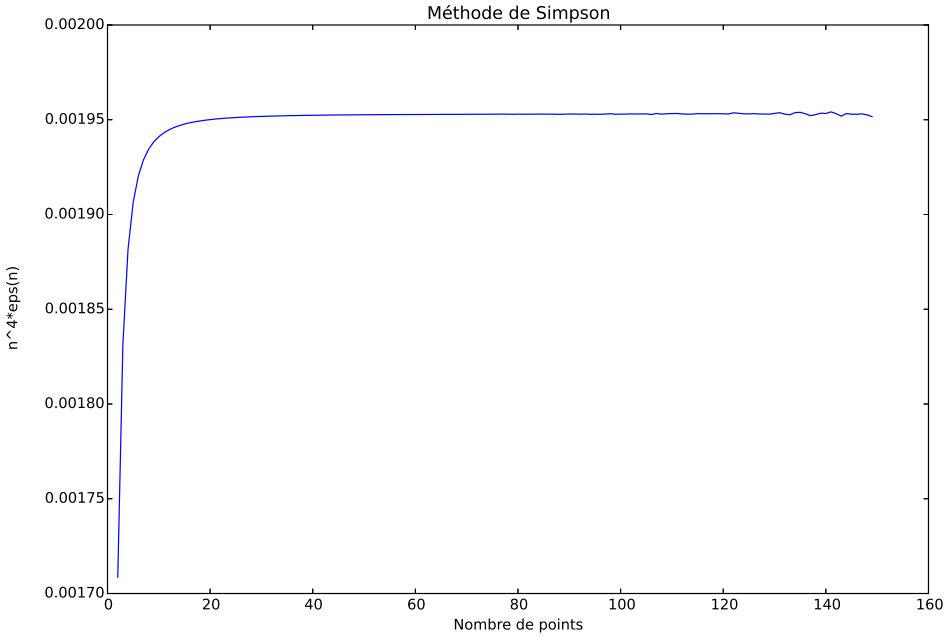
```
                M1, "  Erreur ", EM1, "  Temps: ", Temps_Milieux))

# Et enfin la méthode de Simpson
# on ne compare plus le temps ici
nb_iter = 500
debut = clock()
S1 = Simpson(lambda t: 1/t, 1, 2, nb_iter)
Temps_Simpson = clock() - debut
ES1 = abs(Valeur_Exacte - S1)

print("{:.<30} {:.15f} {} {:.5e} {} {:.5f} \n".format("Simpson "+ str(nb_iter)+" points: ",\
            S1, "  Erreur ", ES1,"  Temps: ", Temps_Simpson))

print("{:.<30} {}".format("Valeur exacte ", Valeur_Exacte))

# Tracé des n^i*eps(n) où eps(n) est l'erreur en fonction de n
# avec i=1 2 ou 4 selon la méthode
# afin de verifier que l'on a bien eps(n) = O(1/n^i)
# ne pas prendre de trop grandes valeurs de n car alors les erreurs d'arrondi l'emportent sue l'er

X1 = range(500, 50000, 50)
Y1 = []
for n in X1:
    Y1.append(n*abs(Valeur_Exacte - Rectangle(lambda t: 1/t, 1, 2, n)))

plt.figure(1, figsize = (12,8))
plt.plot(X1, Y1)
plt.xlabel("Nombre de points")
plt.ylabel("n*eps(n)", labelpad = 30)
plt.title('Méthode des rectangles')

X2 = range(10, 1000, 10)
Y2 = []
for n in X2:
    Y2.append(n*n*abs(Valeur_Exacte - Milieux(lambda t: 1/t, 1, 2, n)))

plt.figure(2, figsize = (12,8))
plt.plot(X2, Y2)
plt.xlabel("Nombre de points")
plt.ylabel("n^2*eps(n)", labelpad = 30)
plt.title('Méthode des points milieux')

X3 = range(2, 150)
Y3 = []
for n in X3:
    Y3.append(n*n*n*n*abs(Valeur_Exacte - Simpson(lambda t: 1/t, 1, 2, n)))

plt.figure(3, figsize = (12,8))
plt.plot(X3, Y3)
plt.xlabel("Nombre de points")
plt.ylabel("n^4*eps(n)", labelpad = 20)
plt.title('Méthode de Simpson')
#plt.show()
plt.close()
```

```
Rectangles_Moche 2000000 points:  0.693147305559908   Erreur  1.25000e-07   Temps:  0.54065
Rectangles 2000000 points: ... 0.693147305532963   Erreur  1.24973e-07   Temps:  0.38990
Trapezes 100000 points: ..... 0.693147180564929   Erreur  4.98324e-12   Temps:  0.01921
Milieux 100000 points: ..... 0.693147180555560   Erreur  4.38538e-12   Temps:  0.01940
Simpson 500 points: ..... 0.693147180559976   Erreur  3.07532e-14   Temps:  0.00020
Valeur exacte ..... 0.693147180559453
```

Les graphiques ci-dessous montrent que l'erreur est bien un $O\left(\frac{1}{n^k}\right)$ avec $k = 1, 2, 4$.






Méthode de Simpson
n^4*eps(n)
Nombre de points
0.00200
0.00195
0.00190
0.00185
0.00180
0.00175
0.00170
0
20
40
60
80
100
120
140
160

★ ★ ★ ★
★ ★ ★
★ ★
★