

TP N°21 : Intégration numérique

1 Introduction

Dans cette partie, nous nous intéresserons aux algorithmes permettant le calcul numérique d'intégrales.

Le principe général commun aux méthodes que nous allons présenter est simple : découper l'intervalle d'intégration en petits morceaux, et approcher sur chacun de ces petits intervalles la courbe représentative de la fonction f par une courbe très simple pour laquelle le calcul d'aire est facile.

Étant donnés deux réels a et b tels que $a < b$ et une fonction f continue sur le segment $[a, b]$, on souhaite, à l'aide de suites, obtenir une valeur approchée de l'intégrale de f sur $[a, b]$.

On note $I(f) = \int_a^b f(t)dt$ l'intégrale à approcher.

Les valeurs de f peuvent provenir de la connaissance de la fonction, ou alors de la connaissance de valeurs de la fonction pour certaines valeurs de t (dans le cas, par exemple, de mesures régulières).

$\forall n \in \mathbb{N}^*$, on va partager le segment $[a, b]$ à l'aide d'une subdivision $(a_i)_{0 \leq i \leq n}$ à pas constant égal à $\frac{b-a}{n}$ en posant

$$\forall i \in [0, n], a_i = a + i \frac{b-a}{n}$$

Sur Chaque segment de $[a_{i-1}, a_i]$, on remplace f par une fonction dont on sait calculer l'intégrale.

Dans le cadre de notre cours, nous présentons deux méthodes :

- **La méthode des rectangles** où les fonctions utilisées pour approcher f sont constantes.
- **La méthode des trapèzes** où les fonctions utilisées pour approcher f sont affines.

2 La méthode des rectangles

Quoi de plus simple comme fonction qu'une fonction constante? Et quoi de plus simple comme aire à calculer qu'une aire de rectangle? La première méthode, que nous allons voir, consiste donc à approcher notre fonction sur chaque sous-intervalle par la fonction constante prenant la même valeur que f à gauche de l'intervalle.

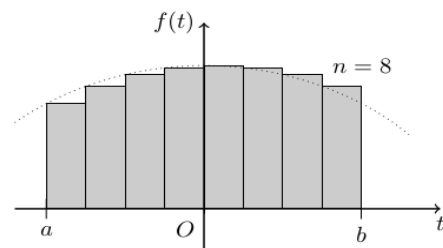
2.1 Définition de la suite associée à cette méthode

Sur Chaque segment de $[a_{i-1}, a_i]$, on remplace f par la fonction constante de valeur $f(a_{i-1})$.

L'intégrale de f est alors approchée par la somme $R_n(f)$ des aires algébriques des rectangles de sommets $(a_{i-1}, 0)$, $(a_i, 0)$, $(a_i, f(a_{i-1}))$ et $(a_{i-1}, f(a_{i-1}))$.

Chaque rectangle a un coté de mesure (positive) $a_i - a_{i-1} = \frac{b-a}{n}$ et un autre coté de mesure algébrique $f(a_{i-1})$ de telle sorte que

$$\forall n \in \mathbb{N}^*, R_n(f) = \frac{b-a}{n} \sum_{i=0}^{n-1} f(x_i)$$



Intégration numérique : méthode de rectangles

2.2 Algorithme

```
def rectangle (f, a, b, n):
    """ valeur approchée de l'intégrale de f sur [a,b]
    application de la methode des rectangles"""
    int_rect = 0
    h = (b-a)/n
    t = a
```

```
for k in range(n):
    int_rect += h*f(t)
    t += h
return int_rect
```

2.3 Exercices

Exercice 1

Écrire le script permettant de calculer numériquement $I = \int_1^5 x \ln(x) dx$

Solution : On utilise la méthode des rectangles

1. On découpe l'intervalle $[a, b] = [1, 5]$ en N parties et on pose $h = \frac{b-a}{n}$.

On obtient ainsi, $N + 1$ points $t_i = a + i \frac{b-a}{N} = a + ih$ avec $i \in [0, N]$.

2. On construit N rectangles;

- L'aire du premier rectangle est égale à : $f(a) \times h = f(a + 0 \times h) \times h$
- L'aire du second rectangle est égale à : $f(a + h) \times h = f(a + 1 \times h) \times h$
- ...
- L'aire du dernier (n^{eme}) rectangle est égale à : $f(a + (N - 1) \times h) \times h$

3. On écrit la somme des aires :

$$\int_a^b f(x) dx \approx f(a).h + f(a + h).h + \dots + f(a + (N - 1).h).h$$

4. En simplifiant l'expression on obtient :

$$\int_a^b f(x) dx \approx h (f(a) + f(a + h) + \dots + f(a + (N - 1).h)) = h \sum_{i=0}^{N-1} f(a + i.h)$$

$$\int_a^b f(x) dx \approx h \sum_{i=0}^{N-1} f(t_i) \quad \text{avec} \quad t_i = a + ih$$

```
5. 1  #On définit la fonction f
    2  def f(x):
    3      import numpy as np
    4      return x * np.log(x)
    5
    6  #programme
    7  def rectangle (f, a, b, n):
    8      """ valeur approchée de l'integrale de f sur [a,b]
    9      application de la methode des rectangles"""
   10      int_rect = 0
   11      h = (b-a) / n
   12      t = a
   13      for k in range(n):
   14          int_rect += h * f(t)
   15          t += h
   16      return int_rect
   17
   18  #Données
   19  a, b, n = 1, 5, 1000
   20
   21  #Résultats numériques
   22  I=rectangle(f, a, b, n)
   23  print("Valeur approchée de I = ", I)
```

6. Valeur approchée de $I \approx 14.1018816722$

3 La méthode des trapèzes

Le principe général est exactement le même que celui de la méthode des rectangles, mais on approche cette fois-ci la courbe sur le segment $[a_i, a_{i+1}]$ par le segment de droite reliant les deux points de la courbe d'abscisses a_i et a_{i+1} , ce qui revient à calculer une somme d'aires de trapèzes pour approcher l'intégrale.

3.1 Définition de la suite associée à cette méthode

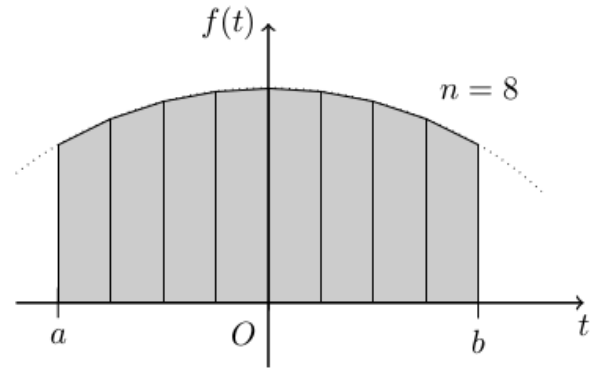
Sur chaque segment de $[a_{i-1}, a_i]$, on remplace f par la fonction affine qui coïncide avec f en a_{i-1} et en a_i .

L'intégrale de f est alors approchée par la somme $T_n(f)$ des aires algébriques des trapèzes de sommets $(a_{i-1}, 0)$, $(a_i, 0)$, $(a_i, f(a_i))$ et $(a_{i-1}, f(a_{i-1}))$.

Visuellement, l'impression est nettement meilleure que pour la méthode des rectangles.

Chaque trapèze a une aire algébrique qui vaut $\frac{b-a}{n} \frac{(f(a_i) + f(a_{i-1})))}{2}$ de telle sorte que

$$\forall n \in \mathbb{N}^*, T_n(f) = \frac{b-a}{2n} \sum_{i=1}^n (f(a_i) + f(a_{i-1}))$$



Intégration numérique : méthode de trapèzes

3.2 Algorithme

```
1 def trapezes(f, a, b, n):
2     h = (b - a) / n
3     int_trapeze = 0.
4     t = a
5     for k in range(n):
6         int_trapeze += h * (f(t) + f(t+h)) / 2
7         t += h
8     return int_trapeze
```

3.3 Exercices

Exercice 2

Écrire le script permettant de calculer numériquement $I = \int_1^5 x \ln(x) dx$

Solution : On utilise la méthode des trapèzes

1. On découpe l'intervalle $[a, b] = [1, 5]$ en N parties et on pose $h = \frac{b-a}{n}$.

On obtient ainsi, $N + 1$ points $t_i = a + i \frac{b-a}{N} = a + ih$ avec $i \in [0, N]$.

2. On construit N trapèzes;

- L'aire du premier trapèze est égale à : $\frac{f(a) + f(a+h)}{2} \times h$
- L'aire du second trapèze est égale à : $\frac{f(a+h) + f(a+2h)}{2} \times h$
- ...
- L'aire du dernier ($n^{\text{ème}}$) trapèze est égale à : $\frac{f(a + (N-1)h) + f(b)}{2} \times h$

3. On écrit la somme des aires :

$$\int_a^b f(x) dx \approx \frac{f(a) + f(a+h)}{2} \cdot h + \frac{f(a+h) + f(a+2h)}{2} \cdot h + \dots + \frac{f(a + (N-1)h) + f(b)}{2} \cdot h$$

4. En simplifiant l'expression on obtient :

$$\int_a^b f(x)dx \approx h \sum_{i=0}^{N-1} \left(\frac{f(t_i) + f(t_i + h)}{2} \right) \quad \text{avec} \quad t_i = a + ih$$

```

5. 1 def trapezes(f, a, b, n):
    2     h = (b - a) / n
    3     int_trapeze = 0.
    4     t = a
    5     for k in range(n):
    6         int_trapeze += h * (f(t) + f(t+h)) / 2
    7         t += h
    8     return int_trapeze
    9
10 #Résultats numériques
11 import numpy as np
12 f = lambda x : x * np.log(x)
13 a, b, n = 1, 5, 1000
14 print("I = ", trapezes(f, a, b, n))

```

6. Valeur approchée de $I \approx 14.1179760513$

4 Déjà disponible en Python

La fonction `scipy.integrate.trapz` permet de mettre en œuvre la méthode des trapèzes sans la programmer.

Si on peut/doit utiliser une fonction de bibliothèque, on privilégiera `scipy.integrate.quad`, plus rapide et performante que les méthodes du programme de notre classe.