La simulation numérique
Intégration numérique

Anis SAIED
Institut Préparatoire aux Etudes d'Ingénieurs de Nabeul (IPEIN),
Université de Carthage,
Tunisie
anis.saieed@gmail.com

Mise à jour la plus récente :2 mars 2017

Introduction et principe général

Dans ce TP, 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 segement de $[a_{i-1}, a_i]$, on remplace f par une fonction dont on sait calculer l'intégrale.

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.

1 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 constate prenant la même valeur que f à gauche de l'intervalle.

1.1 Définition de la suite associée à cette méthode

Sur Chaque segement 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 $(a_{i-1}, f(a_{i-1}))$.

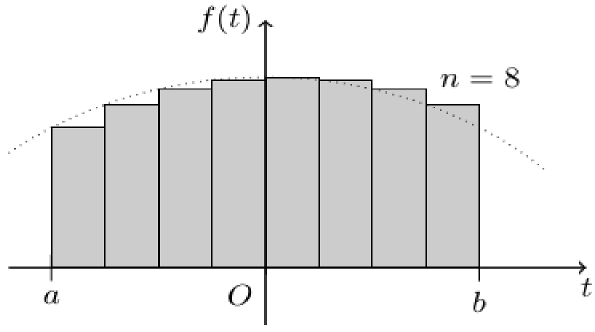

FIGURE 1 – Intégration numérique : méthode de rectangles

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

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

1.2 Algorithme

```python
def rectangle (f, a, b, n):
    """ valeur approchée de l'integrale 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
```

1.3 Exercice 1 :

Ecrire le script permettant de caculer numériquement $I = \displaystyle\int_1^5 x.ln(x)dx$

1.4 Solution :

On utilise la méthode des rectangles

1. On découpe l'intervalle $[a,b]=[1,5]$ en N parties et on pose $h=\dfrac{b-a}{n}$.
   On obtient ainsi, $N+1$ points $t_i = a + i\dfrac{b-a}{N} = a+ih$ 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 :
   $$\int_a^b f(x)dx \approx f(a).h + f(a+h).h + \ldots + f(a+(N-1).h).h$$
4. En simplifiant l'expression on obtient :
   $$\int_a^b f(x)dx \approx h\big(f(a)+f(a+h)+\ldots+f(a+(N-1).h)\big) = h\sum_{i=0}^{N-1} f(a+i.h)$$
   $$\int_a^b f(x)dx \approx h\sum_{i=0}^{N-1} f(t_i) \qquad avec \qquad t_i = a+ih$$
5. ```
   #On définit la fonction f
   def f(x):
           import numpy as np
           return x * np.log(x)

   #programme
   def rectangle (f, a, b, n):
       """ valeur approchée de l'integrale 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

   #Données
   a,b,n=1,5,1000

   #Résultats numériques
   I=rectangle(f,a,b,n)
   print("Valeur approchée de I = ",I)
   ```
6. Valeur approchée de $I \approx 14.1018816722$
   ```

2 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.

2.1 Définition de la suite associée à cette méthode

Sur chaque segement de $[a_{i-1}, a_i]$, on remplace f par la fonction affine qui coincide 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.

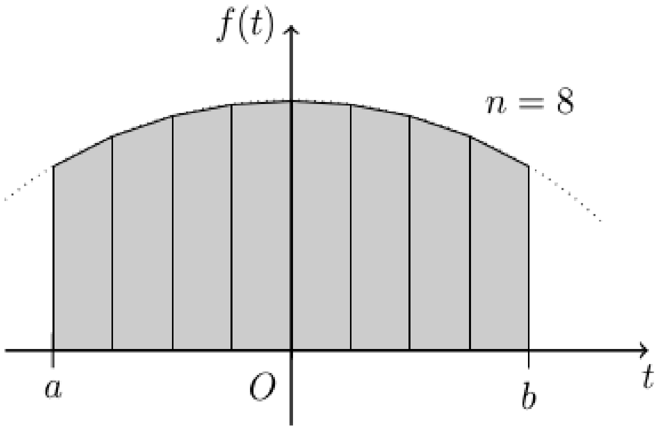

FIGURE 2 – Intégration numérique : méthode de trapèzes

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

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

2.2 Algorithme

```
def trapezes(f,a,b,n):
    h = (b-a)/n
    int_trapeze = 0.
    t = a
    for k in range(n):
        int_trapeze += h * (f(t) + f(t+h)) / 2
        t += h
    return int_trapeze
```

2.3 Exercice 2 :

Ecrire le script permettant de caculer numériquement $I = \int_1^5 x.ln(x)dx$

2.4 Solution :

On utilise la méthode des trapèzes

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

   On obtient ainsi, $N+1$ points $t_i = a + i\dfrac{b-a}{N} = a + ih$ avec $i \in [0, N]$.
2. On construit N trapèzes ;
   - L'aire du premier trapèze est égale à : $\dfrac{f(a)+f(a+h)}{2} \times h$
   - L'aire du second trapèze est égale à : $\dfrac{f(a+h)+f(a+2h)}{2} \times h$
   - ...
   - L'aire du dernier (n^{eme}) trapèze est égale à : $\dfrac{f(a+(N-1)h)+f(b)}{2} \times h$
3. On écrit la somme des aires :

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

4. En simplifiant l'expression on obtient :

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

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

#Résultats numériques
import numpy as np
print("I = ",trapezes(lambda x : x * np.log(x),1,5,1000))
```

6. Valeur approchée de $I \approx 14.1179760513$

3 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.