PROJET ANALYSE NUMERIQUE

Intégration Numérique

Classe : 2DNI Isitcom

On appelle formule composite l'expression caractérisant cette estimation.

Notons k l'indice des n sous-intervalles, $h = (b-a)/n$ la longueur de chacun d'eux, $x_k = a + kh$ la borne inférieure et $m_k = a + (k + 1/2)h$ le point milieu, ceci pour k entre 0 et $n-1$. Voici quelques formules composites :

Méthode des réctangles :

$$I(f) = \frac{(b-a)}{n} \sum_{k=0}^{n-1} f(x_k)$$

Méthode du point milieu :

$$I(f) = \frac{(b-a)}{n} \sum_{k=0}^{n-1} f(m_k)$$

Méthode des trapèzes :

$$I(f) = \frac{(b-a)}{n} \left(\frac{f(a)+f(b)}{2} + \sum_{k=1}^{n-1} f(a+kh) \right)$$

Méthode de Simpson :

$$I(f) = \frac{h}{6} \left(f(a) + f(b) + 2\sum_{k=1}^{n-1} f(x_k) + 4\sum_{k=0}^{n-1} f(m_k) \right)$$

Problème :

On considère les 4 fonctions classiques suivantes :

$$f_1(x) = \cos(x) \qquad f_2(x) = \sin(x)$$
$$f_3(x) = x^2 - x + 5 \qquad f_4(x) = \frac{1}{1+x^2}$$

Le but de ce Projet est de comparer les 4 méthode d'intégrations numériques.

1) Compléter les codes Python Dans les classes RectangleM , Trapeszoidal et Simpson en commentant les réponses dans le codes

2) Utiliser un codes Python pour calculer une valeur approché des intégrales

$$\int_0^1 f_i(t)dt, \qquad i = 1, 2, 3, 4$$

Pour $n = 5$,$n = 25$,$n = 50$ et $n = 100$.

3) Calculer les valeurs exactes des $\displaystyle\int_0^1 f_i(t)dt, \qquad i = 1, 2, 3, 4.$

4) Donner les erreurs numériques pour chaque méthodes pour les différentes valeurs de n.

5) Conclure

```python
class RectangleG(object):
    def __init__(self, a, b, n, f):
        self.a = a
        self.b = b
        self.x = np.linspace(a, b, n+1)
        self.f = f
        self.n = n
    def integrate(self,f):
        x=self.x
        y=f(x)
        h = float(x[1] - x[0])
        s = sum(y[0:-1])
        return h * s
    def Graph(self,f,resolution=1001):
        xl = self.x
        yl = f(xl)
        xlist_fine=np.linspace(self.a, self.b, resolution)
        for i in range(self.n):
            x_rect = [xl[i], xl[i], xl[i+1], xl[i+1], xl[i]] # abscisses des sommets
            y_rect = [0   , yl[i], yl[i]  , 0     , 0   ] # ordonnees des sommets
            plot(x_rect, y_rect,"r")
        yflist_fine = f(xlist_fine)
        plt.plot(xlist_fine, yflist_fine)
        #plt.plot(xl, yl,"bo-")
        plt.xlabel('x')
        plt.ylabel('f(x)')
        plt.title(' Methode des rectangles gauches ')
        text(0.5*(self.a+self.b), f(self.b), '$I_n$ =%12.4f' % (self.integrate(f)), fontsize=15)
```

```python
class RectangleM(object):
    def __init__(self, a, b, n, f):
        self.a = a
        self.b = b
        self.x = np.linspace(a, b, n+1)
        self.f = f
        self.n = n
    def integrate(self,f):
        x=self.x
        y=f(x)
        ###les points milieux m????!!!
        #h = A complèter
        #s = A compléter
        return h * s
    def Graph(self,f,resolution=1001):
        xl = self.x
        yl = f(xl)
        xlist_fine=np.linspace(self.a, self.b, resolution)
        ##### dessiner les rectangles points milieux
        for i in range(self.n):
            #A completer # abscisses des sommets
            #A compléter # ordonnees des sommets
            plot(x_rect, y_rect,"r")
        yflist_fine = f(xlist_fine)
        plt.plot(xlist_fine, yflist_fine)
        #  plt.plot(xl, yl,"bo-")
        plt.xlabel('x')
        plt.ylabel('f(x)')
        plt.title(' Methode des points milieux ')
        text(0.5*(self.a+self.b), f(self.b), '$I_n$ =%12.4f' % (self.integrate(f)), fontsize=15)
```

```python
class Trapezoidal(object):
    def __init__(self, a, b, n, f):
        self.a = a
        self.b = b
        self.x = np.linspace(a, b, n+1)
        self.f = f
        self.n = n
    def integrate(self,f):
        x=self.x
        y=f(x)
        #h = A compléter
        #s = A compléter
        return h * s / 2.0
    def Graph(self,f,resolution=1001):
        xl = self.x
        yl = f(xl)
        xlist_fine=np.linspace(self.a, self.b, resolution)
        for i in range(self.n):
            #A compléter # abscisses des sommets
            #A compléter # ordonnees des sommets
            plot(x_rect, y_rect,"r")
        yflist_fine = f(xlist_fine)
        plt.plot(xlist_fine, yflist_fine)
        plt.title(' Methode des Trapèses ')
        #plt.plot(xl, yl,"bo-")
        plt.xlabel('x')
        plt.ylabel('f(x)')
        text(0.5*(self.a+self.b), f(self.b), '$I_n$ =%12.4f' % (self.integrate(f)), fontsize=15)
```

```python
class Simpson(object):
    def __init__(self, a, b, n, f):
        self.a = a
        self.b = b
        self.x = np.linspace(a, b, n+1)
        self.f = f
        self.n = n

    def integrate(self,f):
        x=self.x
        y=f(x)
        h = float(x[2] - x[1])
        n = len(x) - 1
        ### A Compléter
        ###
        ##s =
        return h * s / 3.0
    def Graph(self,f,resolution=1001):
        xl = self.x
        yl = f(xl)
        xlist_fine=np.linspace(self.a, self.b, resolution)
        for i in range(self.n):
            xx = np.linspace(xl[i], xl[i+1], resolution)
            m = (xl[i]+xl[i+1])/2
            a = xl[i]
            b = xl[i+1]
            l0 = (xx-m)/(a-m)*(xx-b)/(a-b)
            l1 = (xx-a)/(m-a)*(xx-b)/(m-b)
            l2 = (xx-a)/(b-a)*(xx-m)/(b-m)
            P = f(a)*l0 + f(m)*l1 + f(b)*l2
            plot(xx,P,'r')
        yflist_fine = f(xlist_fine)
        plt.plot(xlist_fine, yflist_fine,'g')
        plt.plot(xl, yl,'bo')
        plt.xlabel('x')
        plt.ylabel('f(x)')
        plt.title(' Methode de Simpson  ')
        text(0.5*(self.a+self.b), f(self.b), '$I_n$ =%12.4f' % (self.integrate(f)), fontsize=15)
```