Computación científica LFIS-126

Profesor: Julio C. Marín

Departamento de Meteorología, Universidad de Valparaíso

Segundo Semestre

Temas a tratar en clase

- Integración numérica usando datos discretos y ptos no equidistantes
- Funciones para integrar en Python
- Integrales dobles

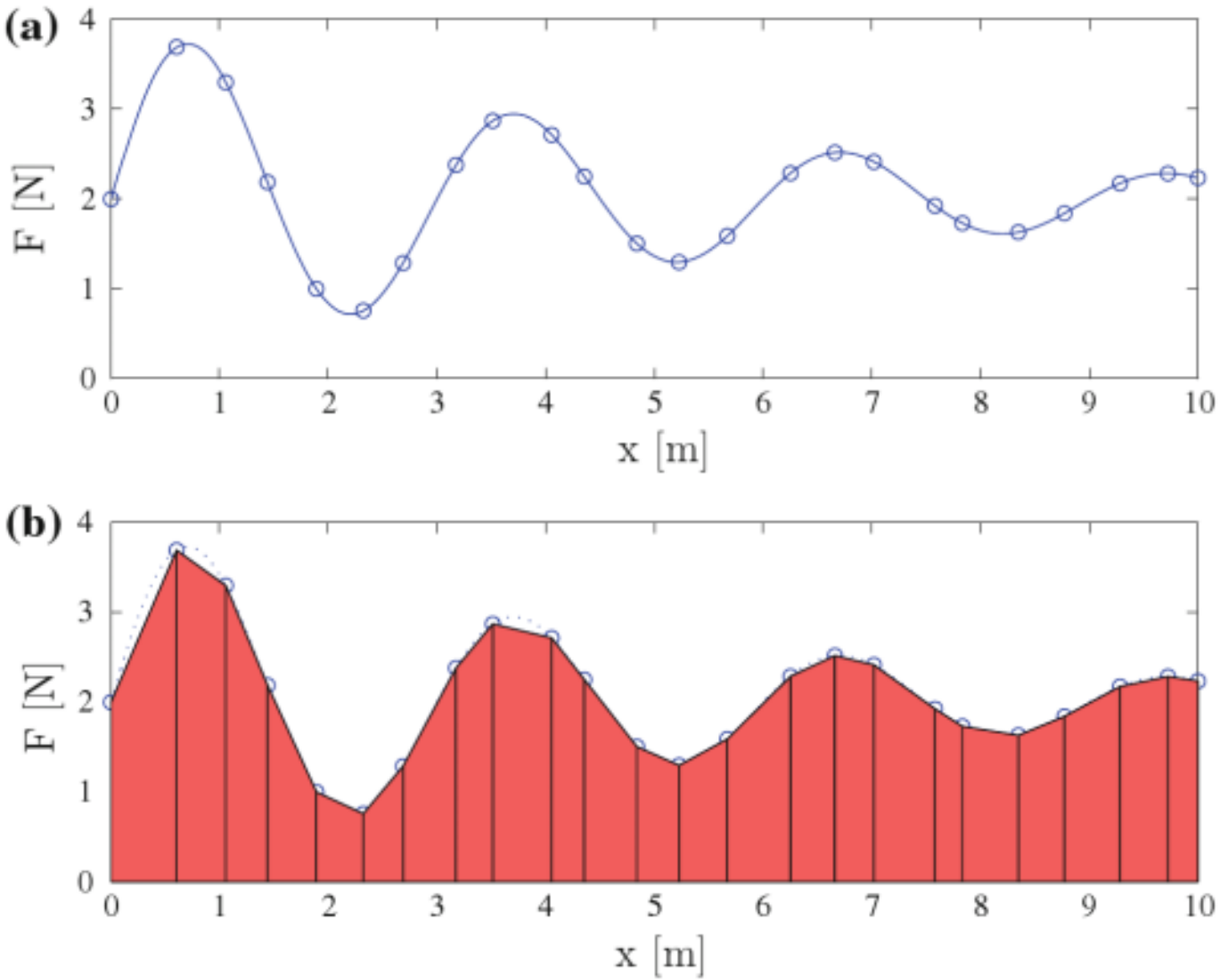
Integración de datos discretos

- Hay casos en que no tenemos valores de una función
- Tenemos resultados de cálculos o valores medidos $F(x_i)$ para valores x_i
- Datos pueden ser no equidistantes.
- Por ejemplo: Método trapezoidal puede aplicarse cuando $\triangle x_i = x_{i+1} - x_i$ varía.

$$I = \sum_{i=1}^{n} \frac{1}{2} \triangle x_i \left[F(x_{i+1}) + F(x_i) \right]$$

- Misma ec. que método trapezoidal compuesto solo que $\triangle x_i$ no es cte.

Integración de datos discretos



Integración de datos discretos

Problema: Calcule la integral para los siguientes datos:

$$I = \sum_{i=1}^{n} \frac{1}{2} \triangle x_i \left[F(x_{i+1}) + F(x_i) \right]$$

x	$f(x)$	x	$f(x)$
0.00	0.200000	0.44	2.842985
0.12	1.309729	0.54	3.507297
0.22	1.305241	0.64	3.181929
0.32	1.743393	0.70	2.363000
0.36	2.074903	0.80	0.232000
0.40	2.456000		

Datos de función $f(x)$ espaciados no uniformemente. Valor exacto de integral es I =1.640533.

$$f(x) = 0.2 + 25x - 200x^2 + 675x^3 - 900x^4 + 400x^5$$

Solución:

Aplicando fórmula anterior queda:

$$I = 0.12\frac{0.2 + 1.309729}{2} + 0.10\frac{1.309729 + 1.305241}{2}$$

$$+ \cdots + 0.10\frac{2.363 + 0.232}{2} = 1.594801$$

Integración de datos discretos

- def f(x):
- return 0.2 + 25*x - 200*x**2 + 675*x**3 - 900*x**4 + 400*x**5
- x1 = [0, 0.12, 0.22, 0.32, 0.36, 0.4, 0.44, 0.54, 0.64, 0.7, 0.8]
- y1 = [f(x) for x in x1]
- Int = 0
- for ii in range(len(x1)-1):
- Int += 0.5*(x1[ii+1] - x1[ii])*(f(x1[ii+1]) + f(x1[ii]))
- print("El valor de la integral = ", Int)
- print("El error relativo porcentual = ", (1.640533-Int)/1.640533*100, "%")

Integración de datos discretos

scipy.integrate

- import scipy.integrate as inte
- import numpy as np
- x = np.array([0, 0.12, 0.22, 0.32, 0.36, 0.4, 0.44, 0.54, 0.64, 0.7, 0.8])
- def f(x):
- return 0.2 + 25*x - 200*x**2 + 675*x**3 - 900*x**4 + 400*x**5
- fx = f(x)
- #### Usando método trapezoidal #####
- Itrap = inte.trapz(fx, x) ==> 1.594800899999999
- Er = 2.8%

Integración de datos discretos

scipy.integrate o numpy

- import scipy.integrate as inte o import numpy as np

```
>>> np.trapz([1,2,3])
4.0
>>> np.trapz([1,2,3], x=[4,6,8])
8.0
>>> np.trapz([1,2,3], dx=2)
8.0
>>> a = np.arange(6).reshape(2, 3)
>>> a
array([[0, 1, 2],
       [3, 4, 5]])
>>> np.trapz(a, axis=0)
array([1.5, 2.5, 3.5])
>>> np.trapz(a, axis=1)
array([2.,  8.])
```

https://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.trapz.html

Usando método de Simpson

- import scipy.integrate as inte
- x = np.array([0, 0.12, 0.22, 0.32, 0.36, 0.4, 0.44, 0.54, 0.64, 0.7, 0.8])
- def f(x):
- return 0.2 + 25*x - 200*x**2 + 675*x**3 - 900*x**4 + 400*x**5
- fx = f(x)
- Vr = 1.640533
- Isimp = inte.simps(fx, x)
- print(Isimp) ### ==> 1.6352173289999998
- Er = abs(Vr - Isimp)/Vr*100 ## ==> 0.32%

Función puede usarse con datos discretos y para una función

https://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.simps.html

Integrales de funciones usando funciones de Python

- def f(x):

  return 0.2 + 25*x - 200*x**2 + 675*x**3 - 900*x**4 + 400*x**5

- Vr = 1.640533; xx = np.arange(0,0.85, 0.05); fx = f(xx)

- #### Método trapezoide ####
- Itrap = inte.trapz(fx, xx); Er_Tr = abs(Vr - Itrap)/Vr*100
- print("Error M. Trapezoidal = ", Er_Tr, "%")

Error M. Trapezoidal = 0.608521742629 %

- #### Método de Simpson ####
- Isimp = inte.simps(fx, xx); Er_Simp = abs(Vr - Isimp)/Vr*100
- print("Error M. Simpson = ", Er_Simp, "%")

Error M. Simpson = 0.00404340134166 %

Método Romberg

- def f(x):

  return 0.2 + 25*x - 200*x**2 + 675*x**3 - 900*x**4 + 400*x**5

- Vr = 1.640533
- #### Método de Romberg ####
- IRomb = inte.romberg(f, 0, 0.8, show=True)
- Er_Romb = abs(Vr - IRomb)/Vr*100
- print("Error M. Romberg = ", Er_Romb, "%")

Error M. Romberg = 2.03185998859e-05 %

https://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.romberg.html

Quadratura Gaussiana

- def f(x):

  return 0.2 + 25*x - 200*x**2 + 675*x**3 - 900*x**4 + 400*x**5

- Iqua = inte.quadrature(f, 0, 0.8)
- Vr = 1.640533; Er_qua = abs(Vr - Iqua[0])/Vr*100
- print("Error M. Q. Gaussiana = ", Er_qua, "%")

Error M. Q. Gaussiana = 2.03185995205e-05 %

- Entrega tuple con 2 valores:
- 1- Valor de integral y 2- Dif. entre 2 últimas integrales

https://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.quadrature.html

Integrales dobles de funciones

- Función nquad realiza integración en múltiples variables.
- Supongamos que queremos calcular:

$$\int_0^2 \int_0^1 x \cdot y^2 dydx$$

- import scipy.integrate as integ
- def f(y, x):

  return x*y**2

- Int2d = integ.nquad(f, [[0, 1], [0, 2]])

(0.6666666666666667, 7.401486830834377e-15)

- Entrega tuple con 2 valores:
- 1- Valor de integral y 2- Máx. error en estimación

https://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.nquad.html
https://docs.scipy.org/doc/scipy/reference/tutorial/integrate.html

Ejercicios

- El archivo perfil_humedad.txt contiene datos de razón de mezcla de vapor de agua (q_v) y presión (p), calcule el agua precipitable (PWV) con la siguiente fórmula:

$$PWV = \frac{1}{g} \int_{p_0}^{p_1} q_v dp$$

donde $g = 9.81 ms^{-2}$ es la aceleración de la gravedad, p_o es la presión en 100000 Pa y p_1 es la presión en 10000 Pa.

Ejercicios

- El archivo perfil_humedad.txt contiene datos de razón de mezcla de vapor de agua (q_v) y presión (p), calcule el agua precipitable (PWV) con la siguiente fórmula:

$$PWV = \frac{1}{g}\int_{p_0}^{p_1} q_v dp$$

donde $g = 9.81 ms^{-2}$ es la aceleración de la gravedad, p_o es la presión en 100000 Pa y p_1 es la presión en 10000 Pa.

- import scipy.integrate as integ
- import numpy as np
- data = np.loadtxt('perfil_humedad.txt', float)
- pwv1 = -integ.trapz(data[:,0], data[:,1])/9.81
- pwv2 = -integ.simps(data[:,0], data[:,1])/9.81

Ejercicio 5.4 del libro

La difracción de la luz en los telescopios limita la calidad de las observaciones astronómicas. Cuando esa luz, con λ, pasa a través de apertura circular del telescopio (asumiendo radio unitario) y se enfoca en un plano focal, se introduce un patrón de difracción circular cuya intensidad está dada por:

$$I(r) = \left(\frac{J_1(kr)}{kr} \right)^2$$

r: distancia en plano focal desde centro del patrón de difracción, $k = 2\pi/\lambda$, y $J_1(x)$ es una función de Bessel. Las funciones de Bessel están dadas por:

$$J_m(x) = \frac{1}{\pi} \int_0^\pi \cos(m\theta - x \sin\theta) d\theta$$

donde m es un entero positivo y $x \geq 0$.
a) Escriba una función de Python $J(m, x)$ que calcule el valor de $J_m(x)$ usando el método de Simpson con $N = 1000$ puntos. Use la función en un programa que plotee las funciones de Bessel J_0, J_1 y J_2 en un mismo gráfico como función de x entre $x = 0$ y $x = 20$.