

Computación científica LFIS-126

Profesor: Julio C. Marín

Departamento de Meteorología, Universidad de Valparaíso

Segundo Semestre

- Funciones para calcular derivadas en Python
- Diferenciación de datos discretos
- Diferenciación simbólica
- Ejercicios

- Diferencias finitas adelantadas

$$f'(x_i) \cong \frac{f(x_{i+1}) - f(x_i)}{(x_{i+1} - x_i)}$$

- Diferencias finitas atrasadas

$$f'(x_i) \cong \frac{f(x_i) - f(x_{i-1})}{(x_i - x_{i-1})}$$

- Diferencias finitas centradas

$$f'(x_i) = \frac{f(x_{i+1}) - f(x_{i-1}))}{2(x_{i+1} - x_i)}$$

Escriba una función llamada derivada que tenga como argumentos de entrada f , a , h , y mt y devuelva $f'(a)$ usando diferencias finitas atrasadas, centradas o adelantadas según se le diga en el argumento "mt" usando un intervalo h .

- 1 Calcule la derivada de la función $f = \sin^2(\frac{\pi}{4})$ usando $h = 0.1$.
- 2 Haga un plot de la función y de su derivada en el rango $(0 - 2\pi)$.

- Fórmulas anteriores para estimar derivadas son para datos uniformemente espaciados.
- Si datos se obtienen de observaciones no espaciadas uniformemente, no se pueden usar fórmulas anteriores.
- Una solución es ajustar un polinomio en intervalo de ptos vecinos que rodean pto donde se quiere evaluar derivada.
- Polinomio no requiere que ptos estén equidistantes.
- Polinomio puede derivarse analíticamente para obtener fórmula para estimar derivada.

`numpy.gradient`: Usa diferencias finitas centradas (2do orden) en puntos internos y diferencias finitas adelantadas o atrasadas en puntos de borde

- `import numpy as np`
- `x = np.linspace(0,10,1000); dx = x[1] - x[0]`
- `y = x**2 + 1`
- `dydx = np.gradient(y, dx)`

<http://docs.scipy.org/doc/numpy/reference/generated/numpy.gradient.html>

Use la función derivada del ejercicio anterior o programe una función que calcule la primera derivada usando el método de diferencias finitas centradas.

- 1 Usando la función creada, calcule la derivada de la función $f = 2 \sin(x/3)$ en el intervalo $(0 - 2\pi)$ usando $h = 0.1$.
- 2 Repita el cálculo de 1. pero usando la función `gradient` de python en el intervalo $(0 - 2\pi)$, usando $h = 0.1$.
- 3 Haga un plot de las primeras derivadas obtenidas con los dos métodos en el rango $(0 - 2\pi)$ donde tengan datos para comparar ambos cálculos.

Diferenciación en Python

`numpy.gradient`: Usa diferencias finitas centradas (2do orden) en puntos internos y diferencias finitas adelantadas o atrasadas en puntos de borde

```
>>> f = np.array([1, 2, 4, 7, 11, 16], dtype=float)
>>> np.gradient(f)
array([1. , 1.5, 2.5, 3.5, 4.5, 5. ])
>>> np.gradient(f, 2)
array([0.5 , 0.75, 1.25, 1.75, 2.25, 2.5 ])
```

<http://docs.scipy.org/doc/numpy/reference/generated/numpy.gradient.html>

Diferenciación de datos discretos

`numpy.gradient`: Usa diferencias finitas centradas (2do orden) en ptos internos y diferencias finitas adelantadas o atrasadas en ptos de borde

```
>>> np.gradient(np.array([[1, 2, 6], [3, 4, 5]], dtype=np.float))  
[array([[ 2.,  2., -1.],  
        [ 2.,  2., -1.]]), array([[ 1. ,  2.5,  4. ],  
        [ 1. ,  1. ,  1. ]])]
```

<http://docs.scipy.org/doc/numpy/reference/generated/numpy.gradient.html>

Ejercicio: Calcule el gradiente de temperatura $\frac{dT}{dz}$ a la altura $z = 7$ m usando los siguientes datos de altura (z) y temperatura (T):

$T = [25.1, 24.0, 22.3, 21.6, 21.1, 20.5, 20, 19.8, 19.2, 18.1]$,
 $z = [0, 1.1, 2.3, 5.2, 6.2, 8.7, 10.5, 12.5, 13.3, 15]$

Ejercicio: Calcule el gradiente de temperatura $\frac{dT}{dz}$ a la altura $z = 7$ m usando los siguientes datos de altura (z) y temperatura (T):

$T = [25.1, 24.0, 22.3, 21.6, 21.1, 20.5, 20, 19.8, 19.2, 18.1]$,

$z = [0, 1.1, 2.3, 5.2, 6.2, 8.7, 10.5, 12.5, 13.3, 15]$

- `import numpy as np`
- `from scipy.interpolate import interp1d`
- `zz = np.arange(0, 15, 0.1)`
- `T_int = interp1d(z, T); T_new = T_int(zz)`
- `dtdz = np.gradient(T_new, 0.1);`
- `dtdz_7m = dtdz[zz == 7.0]`