Universidad de Sonora

División de Ciencias Exactas y Naturales

Licenciatura en Fìsica

Fìsica Computacional 1

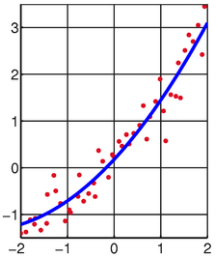
Actividad #4: Ajuste de mìnimos cuadrados

Jesùs Valenzuela Nieblas

20 de febrero de 2016

1. Introducciòn

Mínimos cuadrados es una técnica de análisis numérico enmarcada dentro de la optimización matemática, en la que, dados un conjunto de pares ordenados —variable independiente, variable dependiente— y una familia de funciones, se intenta encontrar la función continua, dentro de dicha familia, que mejor se aproxime a los datos (un "mejor ajuste"), de acuerdo con el criterio de mínimo error cuadrático.



En su forma más simple, intenta minimizar la suma de cuadrados de las diferencias en las ordenadas (llamadas residuos) entre los puntos generados por la función elegida y los correspondientes valores en los datos. Específicamente, se llama mínimos cuadrados promedio (LMS) cuando el número de datos medidos es 1 y se usa el método de descenso por gradiente para minimizar el residuo cuadrado. Se puede demostrar que LMS minimiza el residuo cuadrado esperado, con el mínimo de operaciones (por iteración), pero requiere un gran número de iteraciones para converger.

2. Actividad a realizar

En esta actividad se crearon dos programas que hacen un ajuste lineal y exponencial a un conjunto de datos sobre la temperatura del invierno en New York y la relaciòn presiòn-altitud respectivamente.

3. Ajuste Lineal

EL siguiente còdigo fue el utilizado para hacer el ajuste lineal sobre los datos obtenidos de las temperaturas de invierno de la ciudad de New York de los años 1990-1999.

3.1. Còdigo

```
import numpy as np
import matplotlib.pyplot as plt
from scipy import optimize

datos=np.loadtxt('s48.txt')
```

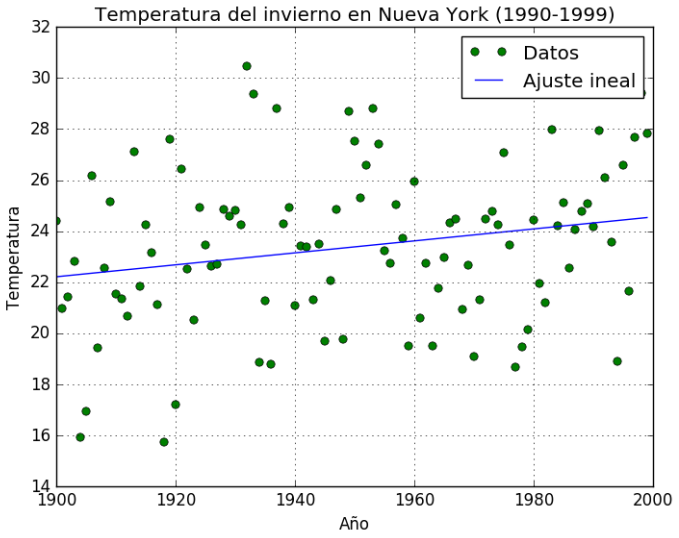
```
x=datos[:,0]
y=datos[:,1]

fitfunc = lambda p, x: p[0]*x + p[1]
errfunc = lambda p, x, y: fitfunc(p, x) - y
p0=[1,1]
p1, success=optimize.leastsq(errfunc, p0[:], args=(x,y))

time = np.linspace(x.min(), x.max(), 100)
plt.plot(x, y, "go", label="Datos")
plt.plot(time, fitfunc(p1, time), "b-", label="Ajuste ineal")

plt.title("Temperatura del invierno en Nueva York (1990-1999)")
plt.xlabel("Año")
plt.ylabel("Temperatura")
plt.grid()
plt.legend()
plt.show()
```

3.2. Resultado Obtenido



4. Ajuste Exponencial

El siguiente còdigo fue el utilizado para hacer un ajuste exponencial a los datos obtenidos de la relaciòn Presòn vs Altitud

4.1. Còdigo

```
import numpy as np
import matplotlib.pyplot as plt
from scipy import optimize

datos=np.loadtxt('s48.txt')
x=datos[:,0]
y=datos[:,1]

fitfunc = lambda p, x: p[0]*x + p[1]
errfunc = lambda p, x, y: fitfunc(p, x) - y
p0=[1,1]
p1, success=optimize.leastsq(errfunc, p0[:], args=(x,y))

time = np.linspace(x.min(), x.max(), 100)
plt.plot(x, y, "go", label="Datos")
plt.plot(time, fitfunc(p1, time), "b-", label="Ajuste ineal")

plt.title("Temperatura del invierno en Nueva York (1990-1999)")
plt.xlabel("Año")
plt.ylabel("Temperatura")
plt.grid()
plt.legend()
plt.show()
```

4.2. Resultado Obtenido