Ejercicio Mínimos Cuadrados

Anthony Contreras

Tabla de Contenidos

[Participación en clase 11] Ejercicio mínimos cuadrados

Resolución Prueba 02

- Nombre: Anthony Contreras
- Fecha: 26/01/2025
- Curso: Métodos Númericos GR1CC

Mínimos Cuadrados

- Interpole los siguientes conjuntos de datos con la función correspondiente. ## Link

Minimos

```
%load_ext autoreload
```

```
The autoreload extension is already loaded. To reload it, use:
  %reload_ext autoreload
```

Conjunto de Datos 1

```python
xs1 = [
    -5.0000,
    -3.8889,
    -2.7778,
    -1.6667,
    -0.5556,
    0.5556,
    1.6667,
    2.7778,
    3.8889,
    5.0000,
]
ys1 = [
    57.2441,
    33.0303,
    16.4817,
    7.0299,
    0.5498,
    0.7117,
    3.4185,
    12.1767,
    24.9167,
    44.2495,
]
```

```python
from src import ajustar_min_cuadrados
import numpy as np
import matplotlib.pyplot as plt
from sympy import symbols, Eq, solve

def der_parcial_2(xs: list, ys: list) -> tuple[float, float, float, float]:
    c_2 = sum(xi**2 for xi in xs)
    c_1 = sum(xs)
    c_0 = len(xs)
    c_ind = sum(ys)
    return (c_2, c_1, c_0, c_ind)


def der_parcial_1(xs: list, ys: list) -> tuple[float, float, float, float]:
    c_2 = sum(xi**3 for xi in xs)
    c_1 = sum(xi**2 for xi in xs)
    c_0 = sum(xs)
    c_ind = sum(xi * yi for xi, yi in zip(xs, ys))
    return (c_2, c_1, c_0, c_ind)


def der_parcial_0(xs: list, ys: list) -> tuple[float, float, float, float]:
    c_2 = sum(xi**4 for xi in xs)
    c_1 = sum(xi**3 for xi in xs)
    c_0 = sum(xi**2 for xi in xs)
    c_ind = sum(xi**2 * yi for xi, yi in zip(xs, ys))
    return (c_2, c_1, c_0, c_ind)

pars1 = ajustar_min_cuadrados(xs1, ys1, gradiente=[der_parcial_0, der_parcial_1, der_parcial_
x = symbols('x')
equation = Eq(pars1[0] * x**2 + pars1[1] * x + pars1[2], 2.25)
solutions = solve(equation, x)
for solution in solutions:
    print(f"x = {solution.evalf()}")
```

```
[01-26 20:20:09][INFO] Se ajustarán 3 parámetros.
[01-26 20:20:09][INFO]
[[ 1.01852593e+02  0.00000000e+00  1.00000000e+01  1.99808900e+02]
 [ 0.00000000e+00  1.01852593e+02  0.00000000e+00 -1.14413577e+02]
 [-2.27373675e-13  0.00000000e+00 -7.90113041e+01  5.04294087e+01]]
[01-26 20:20:09][INFO]
[[ 1.01852593e+02  0.00000000e+00  1.00000000e+01  1.99808900e+02]
 [ 0.00000000e+00  1.01852593e+02  0.00000000e+00 -1.14413577e+02]
 [-2.27373675e-13  0.00000000e+00 -7.90113041e+01  5.04294087e+01]]
x = -0.948805441087225
x = 1.50369543945971
```

Justamente tenia esta idea en el examen pero no me funciono, mi error fue en el der_parcial_2 sumaba en un ciclo para cada valor lo cual era erroneo.

```
# Datos
x1 = np.linspace(-5, 5, 100)
y1 = pars1[0] * x1**2 + pars1[1] * x1 + pars1[2]

# Graficar
plt.scatter(xs1, ys1, color='blue', label="Puntos", edgecolors='black', s=80)
plt.plot(x1, y1, 'r-', label="Interpolación cuadrática", linewidth=2)
```