

Tarea06

November 27, 2025

Escuela Politecnica Nacional del Ecuador

Nombre: Luis Lema

Fecha: 26/11/2025

[Tarea 06] Unidad 03-A | Serie de Taylor y Polinomios de Lagrange

Repositorio:

CONJUNTO DE EJERCICIOS

Determine el orden de la mejor aproximación para los siguientes funciones, usando la Serie de Taylor y el Polinomio de Lagrange:

1. $\frac{1}{(25x^2+1)}, x_0 = 0$

2. $\arctan x, x_0 = 1$

Escriba las fórmulas de los diferentes polinomios

Grafique las diferentes aproximaciones

Series de Taylor

1. Para la primera función:

```
[4]: import matplotlib.pyplot as plt
import numpy as np
import sympy as sp

def f(x):
    return 1 / (25*x**2 + 1)

def f_taylor(x):
    return 1 - 25*x**2 + 625*x**4 - 15625*x**6

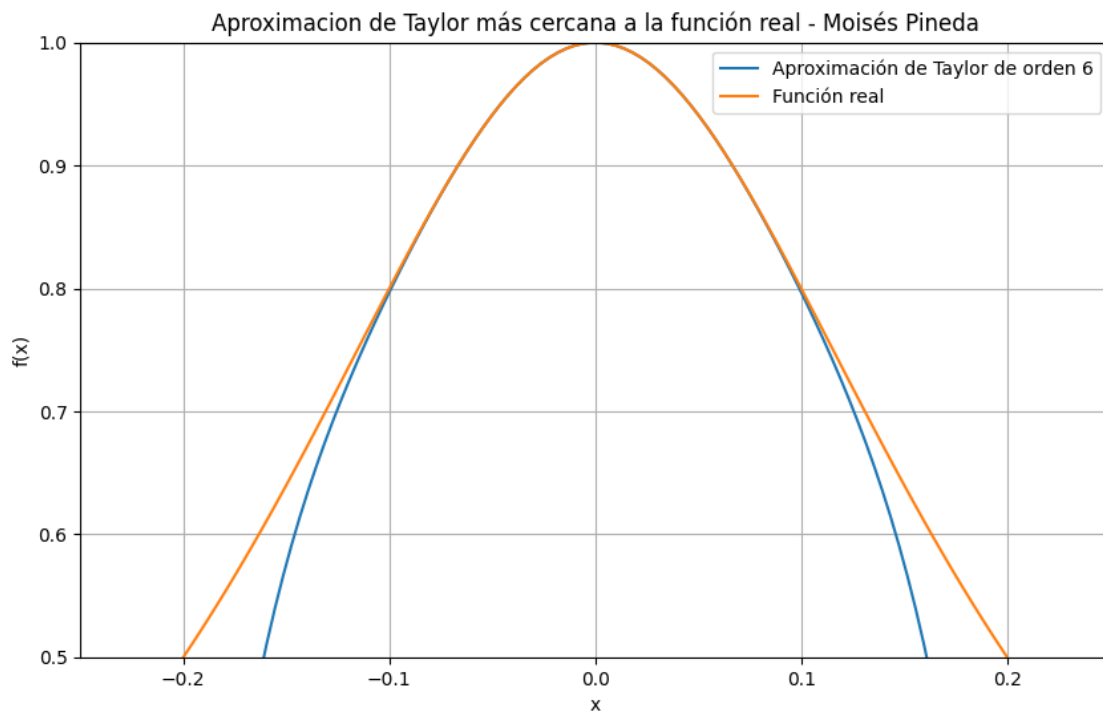
x = np.linspace(-0.2, 0.2, 100)
```

```

y = f_taylor(x)

plt.figure(figsize=(10, 6))
plt.title("Aproximacion de Taylor más cercana a la función real - Moisés Pineda")
plt.plot(x, y, label="Aproximación de Taylor de orden 6")
plt.plot(x, f(x), label="Función real")
plt.xlabel("x")
plt.ylabel("f(x)")
plt.grid()
plt.legend()
plt.xlim(-0.25, 0.25)
plt.ylim(0.5, 1)
plt.show()

```



2. Para la segunda función

```

[74]: def f_2(x):
        return np.arctan(x)

def f_2_taylor(x):
    return np.pi/4 + (x-1)/2 - (x-1)**2/4 + (x-1)**3/12 - (x-1)**5/40 +
    (x-1)**6/48

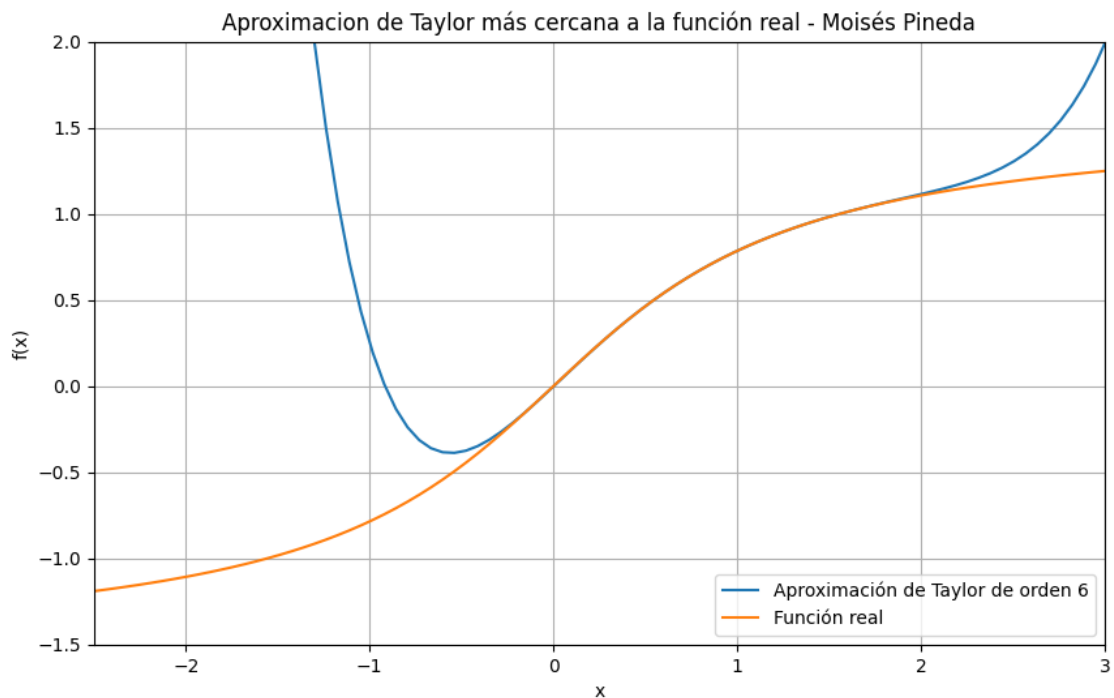
```

```

x = np.linspace(-np.pi, np.pi, 100)

y = f_2_taylor(x)
plt.figure(figsize=(10, 6))
plt.title("Aproximacion de Taylor más cercana a la función real - Moisés Pineda")
plt.plot(x, y, label="Aproximación de Taylor de orden 6")
plt.plot(x, f_2(x), label="Función real")
plt.xlabel("x")
plt.ylabel("f(x)")
plt.grid()
plt.legend()
plt.xlim(-2.5, 3)
plt.ylim(-1.5, 2)
plt.show()

```



Polinomios de Lagrange

1. Para la primera función

```

[75]: def polinomio(x):
      return (-0.08*x**2 + 0.0064) / 0.0064

```

```

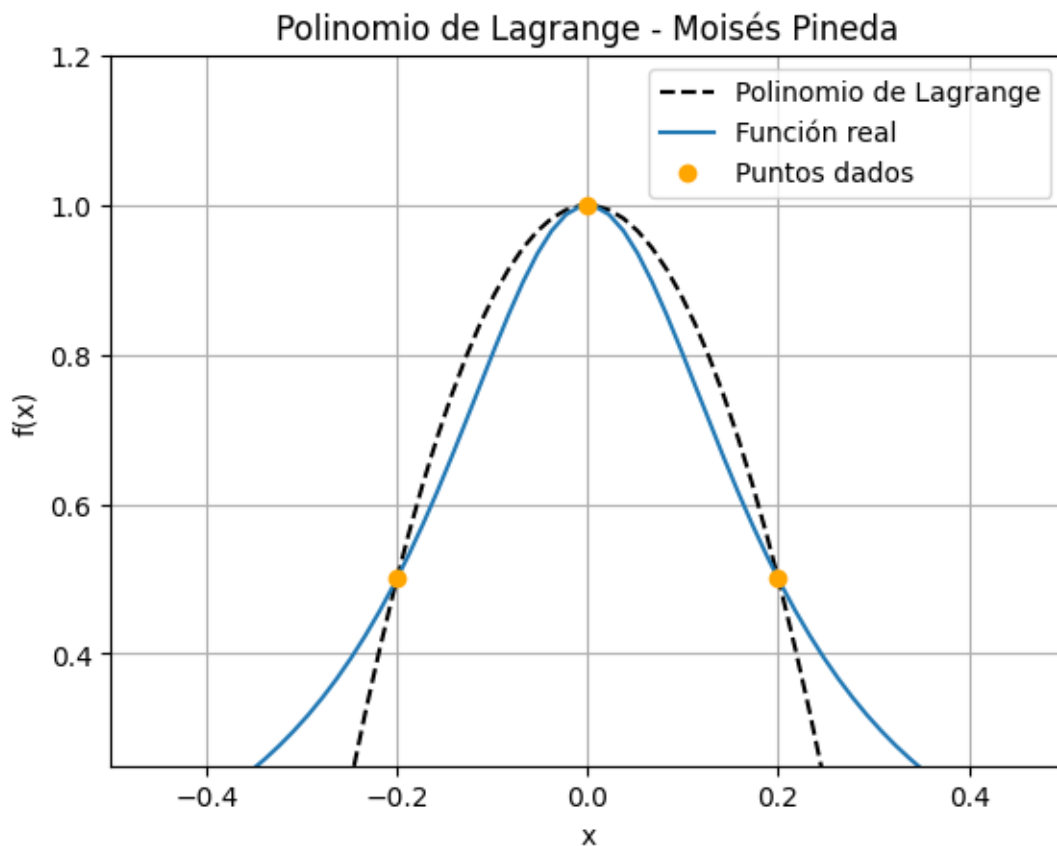
points2 = [(-1/5, 1/2), (0, 1), (1/5, 1/2)]

x_lagrange = np.linspace(-3, 3, 400)
y_lagrange = polinomio(x_lagrange)

y_real = f(x_lagrange)

x_coords, y_coords = zip(*points2)
plt.plot(x_lagrange, y_lagrange, label='Polinomio de Lagrange', linestyle='--',
        color='black')
plt.plot(x_lagrange, y_real, label='Función real')
plt.plot(x_coords, y_coords, 'o', label='Puntos dados', color='orange')
plt.legend()
plt.grid()
plt.title('Polinomio de Lagrange - Moisés Pineda')
plt.xlabel('x')
plt.ylabel('f(x)')
plt.xlim(-0.5, 0.5)
plt.ylim(0.25, 1.2)
plt.show()

```



2. Para la segunda función

```
[76]: def polinomio_2(x):  
    return (-1.107148718)*x*(x**2-3*x+2) / (-24) + (np.pi*x*(x**2-4))/(-12) \  
    + (1.107148718*x*(x**2+x-2)) / 8  
  
points_2 = [(0, 0), (-2, -1.107148718), (1, np.pi/4), (2, 1.107148718)]  
x_lagrange_2 = np.linspace(-3, 3, 100)  
y_lagrange_2 = polinomio_2(x_lagrange_2)  
  
x_coords_2, y_coords_2 = zip(*points_2)  
plt.plot(x_lagrange_2, y_lagrange_2, label='Polinomio de Lagrange',  
        color='black', linestyle='--')  
plt.plot(x, f_2(x_lagrange_2), label='Función real')  
plt.plot(x_coords_2, y_coords_2, 'o', label='Puntos dados', color='orange')  
plt.legend()  
plt.grid()  
plt.title('Polinomio de Lagrange - Moisés Pineda')  
plt.xlabel('x')  
plt.ylabel('f(x)')  
plt.xlim(-3, 3)  
plt.ylim(-2, 2)  
plt.show()
```

