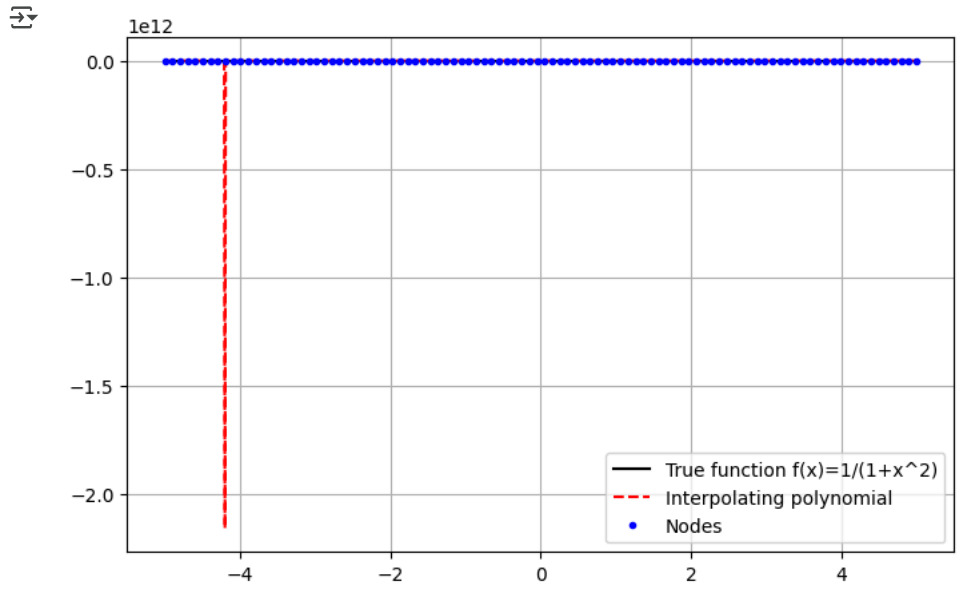
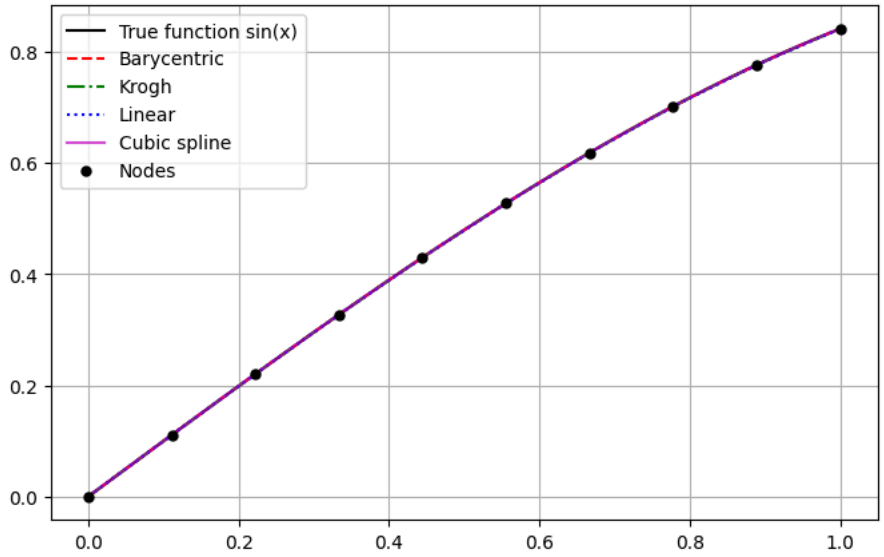
1. Consider $f(x) = \frac{1}{1+x^2}$ where $-5 \leq x \leq 5$ 取100個點

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.interpolate import BarycentricInterpolator, KroghInterpolator, interp1d

f = lambda x: 1/(1 + x**2)

# 範圍 [-5, 5]
x_plot = np.linspace(-5, 5, 1000)
y_true1 = f(x_plot)

# 插值點 (100 個等距節點)
x_nodes = np.linspace(-5, 5, 100)
y_nodes = f(x_nodes)

# 用 Barycentric 插值
interp_runge = BarycentricInterpolator(x_nodes, y_nodes)
y_interp1 = interp_runge(x_plot)

plt.figure(figsize=(8,5))
plt.plot(x_plot, y_true1, 'k', label="True function f(x)=1/(1+x^2)")
plt.plot(x_plot, y_interp1, 'r--', label="Interpolating polynomial")
plt.plot(x_nodes, y_nodes, 'bo', markersize=3, label="Nodes")
plt.legend()
plt.grid(True)
```



2. Consider $f(x) = \sin(x)$ where $x \in [0, 1]$ interpolate polynomial way 10 points with a different algorithm. scipy interpolate

```
f = np.sin
x_nodes2 = np.linspace(0, 1, 10)   # 10 個點
y_nodes2 = f(x_nodes2)

x_plot2 = np.linspace(0, 1, 500)
y_true2 = f(x_plot2)

# 幾種不同演算法：
interp_bary = BarycentricInterpolator(x_nodes2, y_nodes2)
interp_krogh = KroghInterpolator(x_nodes2, y_nodes2)
interp_linear = interp1d(x_nodes2, y_nodes2, kind='linear')
interp_cubic = interp1d(x_nodes2, y_nodes2, kind='cubic')

plt.figure(figsize=(8,5))
plt.plot(x_plot2, y_true2, 'k', label="True function sin(x)")
plt.plot(x_plot2, interp_bary(x_plot2), 'r--', label="Barycentric")
plt.plot(x_plot2, interp_krogh(x_plot2), 'g-.', label="Krogh")
plt.plot(x_plot2, interp_linear(x_plot2), 'b:', label="Linear")
plt.plot(x_plot2, interp_cubic(x_plot2), 'm', alpha=0.7, label="Cubic spline")
plt.plot(x_nodes2, y_nodes2, 'ko', markersize=5, label="Nodes")
plt.legend()
plt.grid(True)

plt.show()
```


True function sin(x)
Barycentric
Krogh
Linear
Cubic spline
Nodes
0.0
0.2
0.4
0.6
0.8
1.0