National University of Computer and Emerging Sciences



Name: Muhammad Suleman

Roll #: 22F-3350

Section: BCS-4B

Assignment # 01-Part02

# Question 01

*import* numpy *as* np

*import* matplotlib.pyplot *as* plt

*from* scipy.interpolate *import* CubicSpline

*# Load the data from the file*

data = np.loadtxt('bps.dat')

*# Extract the columns you want to use for interpolation and plotting*

x = data[:, 0] *# First column*

y = data[:, 1] *# Second column*

*# Create a cubic spline interpolation object*

cubic\_spline = CubicSpline(x, y)

*# Generate points for interpolation*

x\_interp = np.linspace(x[0], x[-1], 1000)

y\_interp = cubic\_spline(x\_interp)

*# Plot the original data and interpolated curve*

plt.plot(x, y, 'o', label='Original Data')

plt.plot(x\_interp, y\_interp, label='Interpolated Curve')

plt.xlabel('X')

plt.ylabel('Y')

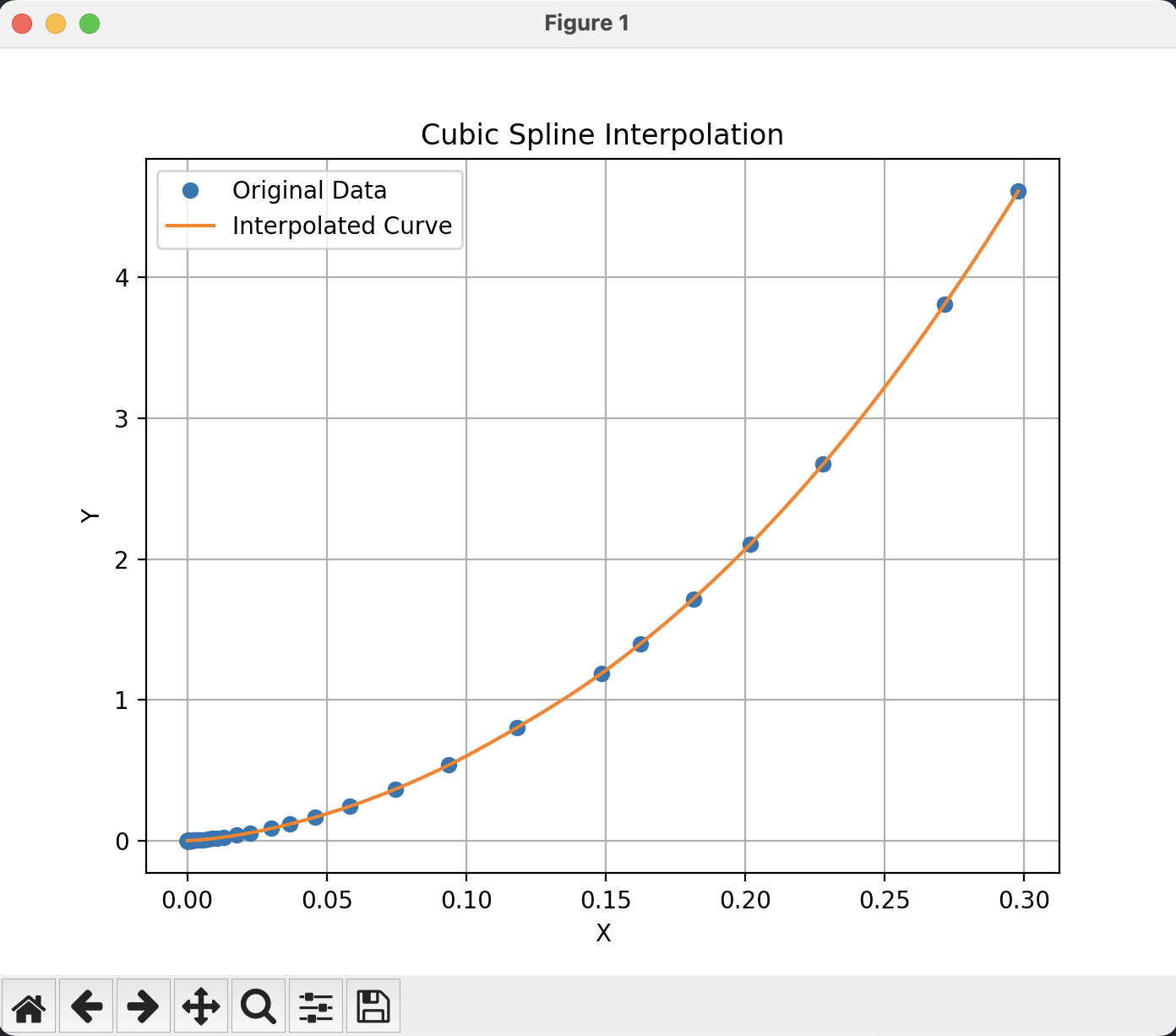
plt.title('Cubic Spline Interpolation')

plt.legend()

plt.grid(*True*)

plt.show()

## Output



# Question 02

*import* sympy

*def* equations():

M, r, q = sympy.symbols('M r q', real=*True*, positive=*True*)

x, y, z = sympy.symbols('x y z', real=*True*)

eq1 = 1 / r + 2 \* M \* q \*\* 2 / r \*\* 3 - x \*\* 2 \* r \*\* 2

eq2 = (1 / r + 2 \* M \* q \*\* 2 / r \*\* 3) \* (y \*\* 2 / (x + z \* r) \*\* 2)

eq3 = sympy.diff((1 / r + 2 \* M \* q \*\* 2 / r \*\* 3) \* (y \*\* 2 / (x + z \* r) \*\* 2), r)

*return* eq1, eq2, eq3

eq1, eq2, eq3 = equations()

*# Define x, y, z*

x, y, z = sympy.symbols('x y z', real=*True*)

sol = sympy.solve([eq1, eq2, eq3], [x, y, z])

*for* solution *in* sol:

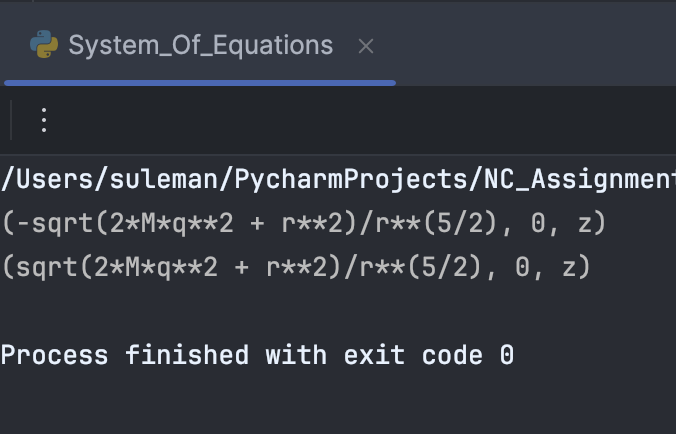
print(sympy.simplify(solution))

*# Answer*

*# (-sqrt(2\*M\*q\*\*2 + r\*\*2)/r\*\*(5/2), 0, z)*

*# (sqrt(2\*M\*q\*\*2 + r\*\*2)/r\*\*(5/2), 0, z)*

## Output



# Question 03

*import* numpy

*def* newton\_backward\_interpolation(x, y, x\_new):

n = len(x)

*# Create a table to store the divided differences*

divided\_differences = numpy.zeros((n, n))

*# Fill in the first column of the divided differences table*

divided\_differences[:, 0] = y

*# Calculate the remaining divided differences*

*for* j *in* range(1, n):

*for* i *in* range(n - j):

divided\_differences[i, j] = (divided\_differences[i + 1, j - 1] - divided\_differences[i, j - 1]) / (

x[i + j] - x[i])

*# Calculate the interpolated value*

y\_new = divided\_differences[0, 0]

*for* j *in* range(1, n):

term = divided\_differences[0, j]

*for* k *in* range(j):

term \*= (x\_new - x[k])

y\_new += term

*return* y\_new

years = numpy.array([1941, 1951, 1961, 1971, 1981, 1991])

population = numpy.array([12, 15, 20, 27, 39, 51])

estimated\_year1 = 1976

estimated\_year2 = 1978

interpolated\_population2 = newton\_backward\_interpolation(years, population, estimated\_year2)

interpolated\_population1 = newton\_backward\_interpolation(years, population, estimated\_year1)

result = interpolated\_population2 - interpolated\_population1

print(f"Interpolated population at year {estimated\_year1} is: {interpolated\_population1:.4f}")

print(f"Interpolated population at year {estimated\_year2} is: {interpolated\_population2:.4f}")

print(f"Interpolated population from year {estimated\_year1} to {estimated\_year2} is: {result:.4f}")

## Output

