**Q1. Using different values of n\_refine what changes do we see -**

def create\_data(n):

"""Given an integer n, returns n data points

x and values y as a numpy.array."""

xmax = 10.

x = np.linspace(0, xmax, n)

y = - x\*\*2

#make x-data somewhat irregular

y += 1.5 \* np.random.normal(size=len(x))

#y = -x\*\*2 + 1.5 \* np.random.normal(size=len(x))

return x, y

#main program

n = 30

x, y = create\_data(n)

plt.plot(x, y, 'b.', label='data point')

plt.legend()

# using different values what changes do we see

n\_refine=10; #3, 15, 20

xfine = np.linspace(0.1, 9.9, n \* n\_refine)

#interpolate with piecewise constant function (p=0)

y0 = scipy.interpolate.interp1d(x, y, kind='nearest')

#interpolate with piecewise linear func (p=1)

y1 = scipy.interpolate.interp1d(x, y, kind='linear')

#interpolate with piecewise constant func (p=2)

y2 = scipy.interpolate.interp1d(x, y, kind='quadratic')

#interpolate with piecewise constant func (p=3)

y3=scipy.interpolate.interp1d(x, y, kind='cubic')

plt.plot(x, y, 'b.', label='data point')

plt.plot(xfine, y0(xfine), '-r', label='nearest')

plt.plot(xfine, y1(xfine), '-b', label='linear')

plt.plot(xfine, y2(xfine), '-g', label='quadratic')

plt.plot(xfine, y3(xfine), '--', label='cubic')

plt.legend()

plt.xlabel('x')

**Q2. The code below is for function f1 - Try the same with function f2**

# curve fitting

import numpy as np

from scipy.optimize import curve\_fit

import matplotlib.pyplot as plt

def f1(x, a, b, c):

"""Fit function y=f(x,p) with parameters p=(a,b,c). """

return a \* np.exp(- b \* x) + c

def f2(x, a, b, c):

"""Fit function y=f(x,p) with parameters p=(a,b,c). """

#return a \* np.exp(- b \* x^2) + c

return a \* np.sin(b\*x) + c

#create fake data

x = np.linspace(0, 4, 50)

y = f1(x, a=2.5, b=1.3, c=0.5)

#add noise

noise\_amp=0.25;

yi = y + noise\_amp \* np.random.normal(size=len(x))

plt.plot(x,y)

plt.plot(x,yi,'.')

**Q3. Try the above code of Q2 with different noise amplitude - 0.25, 0.01, 0.5 - for both f1 and f2**

**Share your insights.**

**Q4. OPTIONAL - WHEN SUBMITTING -** Sir will discuss in class

Summarize the reaction - how a, b, c values will change. Is there anything we can do to reduce the effect of noise? What happens if we go to previous finer grids?

#call curve fit function

popt, pcov = curve\_fit(f1, x, yi)

a, b, c = popt

print("Optimal parameters are a=%g, b=%g, and c=%g" % (a, b, c))