```
"""
Registration : xxxx
Description  : Curve Fitting using numpy and scipy
Author       : AKB
"""

import numpy as np
import matplotlib.pyplot as plt
from scipy.optimize import curve_fit
import warnings
warnings.filterwarnings("ignore")

# Parameter values
a = 2.5; b = 1.2; c = 1; var = 1.005;
N = 5;                          # number of point in x
x  = np.linspace(0,  N,   50)  # define x-axis
xp = np.linspace(-1, N+1, 100) # define x-axis on which interpolation is performed

# define y = f(x)
def func(x, a, b, c):
    #return a*np.exp(-b*x) + c
    return a*pow(x,3)-b*pow(x,2)+c*x

y = func(x, a, b, c)
y = y + var*np.random.randn(x.size) # Add noise to y for better visibility of Fitting
plt.plot(x,y,'r-',lw=2,ms=5,label=r'$y = '+str(a)+'e^{-'+str(b)+'x}+'+str(c)+'+\mathcal{N}
(0,'+str(var)+')$')  # Plot the Raw Data

# use scipy optimize to construct the polynomial object by fitting
popt, pcov = curve_fit(func, x, y)
plt.plot(x, func(x, *popt), 'x', lw=1, ms=6, c='olive', label=r'Fit: $y = %ge^{-%gx}+%g$'
%tuple(popt))
#plt.plot(x, func(x, *popt), 'o-.', lw=2, ms=10, c='olive', label='fit: a=%g, b=%g, c=%g'
%tuple(popt))

# optimize within the region 2<=a<=3, 0.5<=b<=1.5, 0.5<=c<=1.5 for educated fit
popt, pcov = curve_fit(func, x, y, bounds=(0, [2.5, 1.2, 1.0]))
plt.plot(x, func(x, *popt), 'go', lw=1, ms=3, label=r'Optimized Fit: $y = %ge^{-%gx}+%g$'
%tuple(popt))

# use numpy to construct the polynomial object by fitting
p3  = np.poly1d(np.polyfit(x, y, 3))
p12 = np.poly1d(np.polyfit(x, y, 12))
plt.plot(xp, p3(xp),  'd', lw=1, ms=3, c='b', label=r'$3^{rd}$ Order Polynomial')
plt.plot(xp, p12(xp), '>', lw=1, ms=3, c='m', label=r'$12^{th}$ Order Polynomial')
plt.legend(loc='best', prop={'size':14})
plt.title('Curve Fitting', size=14)
plt.xlabel('X Data', size = 16)
plt.xticks(size = 16)
plt.ylabel('Y Data', size=16)
plt.yticks(size = 16)
plt.xlim(-1, N+1); #plt.ylim(-1, 5)
plt.grid()
plt.show()
```