```
from matplotlib.pyplot import *
import matplotlib.pyplot as plt
from scipy.optimize import curve_fit
from scipy.interpolate import PchipInterpolator
from scipy.signal import savgol_filter
import numpy as np

# data entry for neutron and proton in relation to the stage
x = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21,
              22, 23, 24, 25, 26])

# neutron free state in the electron charges for each stage
y22 = np.array([0.1, - 0.65, 0.2, 0.35, 0.15, 0.1, - 0.65, 0.15, 0.15, 0, 0.3, 0, -0.55,
0.0,
                0.5, 0.2, 0, - 0.55, - 0.55, 0, 0.7, 0, 0, -0.25, 0, 0, 0])

# proton free state in the electron charges for each stage
z22 = np.array([0.2, 0.35, 0.1, - 0.65, 0.15, 0.2, 0.35, 0.15, 0.15, 0, 0, 0.65, 0.2,
-0.50, 0,
                0.1, 0.35, 0.2, 0.2, 0.35, 0.1, 0, 0.15, 0.2, 0, 0, 0])

#  It is used to create an evenly spaced sequence in a specified interval
xx = np.linspace(x.min(),x.max(), 1000)
fig, axs = plt.subplots(1, 1, figsize=(16, 11))

# monotonic cubic interpolation
# The interpolant uses monotonic cubic splines to find the value of new points
itp1 = PchipInterpolator(x,y22)
itp2 = PchipInterpolator(x,z22)

# The order of the polynomial used to fit the samples
window_size, poly_order = 57, 2

# Apply a Savitzky-Golay filter to an array
y22y22_sg = savgol_filter(itp1(xx), window_size, poly_order)
z22z22_sg = savgol_filter(itp2(xx), window_size, poly_order)

# Graphic images
axs.plot(x, y22, 'gs', label= 'Cycle of charge distribution in Qe in a free neutron over
shells\
taking into account the stages n11, n12, n13 annihilation')

axs.plot(xx, y22y22_sg, 'green', label= "Smoothed curve")

axs.plot(x, z22, 'bs', label= 'Cycle of charge distribution in Qe in a free proton over
shells\
taking into account the stages p11, p12, p13 annihilation')

axs.plot(xx, z22z22_sg, 'b', label= "Smoothed curve")

# fit to a global function
def func(x, A, B, x0, sigma):
    return abs(A)+B*np.tanh((x-x0)/sigma)

fit, _ = curve_fit(func, x, y22)
y22y22_fit = func(xx, *fit)

fit, _ = curve_fit(func, x, z22)
z22z22_fit = func(xx, *fit)

axs.plot(xx, y22y22_fit, 'g--', label=r"$f(xn) = |A| + B \tanh\left(\frac{x-x_0}
{\sigma}\right)$")

axs.plot(xx, z22z22_fit, 'b--', label=r"$f(xp) = |A| + B \tanh\left(\frac{x-x_0}
{\sigma}\right)$")

# phase separation
plt.axvline(x=8, color='r', linestyle='--', lw=2)
plt.axvline(x=17, color='r', linestyle='--', lw=2)
plt.axvline(x=26, color='r', linestyle='--', lw=2)

# axis labels
plt.ylabel('The amount of charge in the charges of an electron', fontsize=12)
plt.xlabel('Shell number,                         End of the first stage n11, p11
\
End of the second stage n12, p12                                End of cycle n13, p13',
fontsize=12)

# stage numbering in each phase
xticks(range(27), ['0', '1', '2', '3', '4', '5', '6', '7', '8', '8', '7', '6', '5', '4',
'3', '2', '1',
                   '0', '0', '1', '2', '3', '4', '5', '6', '7', '8'], fontsize=12)
yticks(fontsize=12)

plt.legend(loc='upper left', fontsize=12)

# saving in pdf format
savefig('Cycle of charge distribution over shells in Qe in a free neutron and proton,
taking into account the stages \
annihilation.pdf')
```