```
## Se rullende_legeme.py for at fÃ¥ en forstÃ¥else af hvordan scriptet
## virker. Dette script fungere pÃ¥ prÃ¦cis samme metode.

import matplotlib.pyplot as plt
import numpy as np
import scipy.optimize as scp
import scipy.special as ss
import os

exec(open('Kalibrering/kalibrering.py').read())
exec(open('../Scripts/Statistik.py').read())
exec(open('../Scripts/data_renser.py').read())

fig, ax = plt.subplots(2, 2, figsize = (20, 10))

ax = ax.ravel()
# Importer data

def fit(t,*p):
    a = p[0]
    t0 = p[1]
    c = p[2]
    return np.heaviside(t-t0,1)*(1/2*a*(t-t0)**2)+c

def plot_data(data, ax, labels, title, kali, grader):

    theta = grader*(2*np.pi/360)
    sol1 = Data(data)
    x = func(sol1.points, *kali)
    t = sol1.t*1000

    mask = sol1.rinse([[-1, 0.1], [0.7, 0.4]])

    ax.scatter(t[~mask], x[~mask], color = 'blue', label = 'outliers')
    ax.scatter(t[mask], x[mask], color = 'red', label = 'data points', alpha = 0.25)

    guess_params = [1,-0.17,0.05]

###
    popt,pcov = scp.curve_fit(fit, t[mask], x[mask],
                            guess_params, bounds = ((-10, -0.5, -10),(10, 0.5, 10)))
###


    t_fit = np.linspace(t[mask][0],t[mask][-1],1000)
    ax.plot(t_fit, fit(t_fit, *popt), color = 'k', linewidth = 2,
            label = 'fitted function')

    var_a = round(np.sqrt(np.diag(pcov)[0]), 2)
    eksp_a = round(popt[0], 3)
    eksp_gnid =  (np.sin(theta) - popt[0]/9.82)/np.cos(theta)

    error = propagation_function(t_fit, fit, list(popt), pcov)
    ax.fill_between(t_fit, fit(t_fit, *popt) + error,
                    fit(t_fit, *popt) - error, alpha = 0.3)

    ax.set_ylim(-0.2,0.7)
    ax.set_ylabel('x/m')
    ax.set_xlabel('t/s')
    ax.set_title(title)
    ax.legend()

    t_a = 9.82*(np.sin(theta) - np.cos(theta)*0.19)
    print("Eksperimentel gnidningskoefficient = {} $\pm$ {}".format(eksp_a, var_a))


plot_data("Kobber12", ax[0], labels = None, title = 'Kobber 12grader',
          kali = kali, grader = 12)

plot_data("Kobber13", ax[1], labels = None, title = 'Kobber 13grader',
          kali = kali, grader = 13)

plot_data("Kobber", ax[2], labels = None, title = 'Kobber 15grader',
          kali = kali, grader = 15)

plot_data("Kobber19", ax[3], labels = None, title = 'Kobber 19grader',
          kali = kali, grader = 19)


plt.show()

    ###############

fig.savefig('Plots/glider_vinkel.png')
```