```
import numpy as np
import matplotlib.pyplot as plt
import scipy.stats as ss
import scipy.optimize as op
import fejlpropagering as fejl

# Loader data og skaber plots

fig, ax = plt.subplots(figsize = (16,8))
deltas = np.array(np.loadtxt('tension.txt')[:,0])
tensions = np.loadtxt('tension.txt')[:,1]
sigma =  np.array(len(tensions)*[0.05])

# Vi fitter den reciprokke af tension

tens_1 = 1/tensions

# Konstanter

deltaAf = 0.01
deltas = deltas*deltaAf
masseP = 0.581
g = 9.82

# Fitte funktion

def tension(delta, *p):
    a = p[0]
    b = p[1]
    return b + a*delta

# Fittet laves

p0 = [0.2,0.2]
pop,cov = op.curve_fit(tension, deltas, tens_1, p0) #sigma = sigma, absolute_sigma = True)

# Afstanden til massemidtpunket beregnes.

Rw = (pop[0]*g*masseP)**-1

# Datapunkterne plottes med errorbars, fittet plottes med fejlpropageringen
# omkring.

ax.errorbar(deltas,1/tensions,sigma, linestyle ='',
            markersize = 5,  fmt = 's', capsize = 3,
            label = 'tensions',
            color = 'red')
ax.plot(deltas,tension(deltas,pop[0],pop[1]), label = 'tensionfit', color = 'black')

fejl.plot_propagation(deltas, tension, pop, cov, ax)

ax.set_title('Forhold mellem tension og $x$-forskydning', fontsize = 18)
ax.set_xlabel('$\Delta x$', fontsize = 16)
ax.set_ylabel('$T^{-1}$', fontsize = 16)
ax.legend()

# Usikkerheden pÃ¥ afstanden til massemidtpunktet beregnes

stds = np.sqrt(np.diag(cov))
Fw = masseP * g
fejl_RW = 1/(Fw*pop[0]**2)*stds[0]

fig.savefig('Plots/tension')
```

```
plt.show()
```