AnalysisE72

March 18, 2024

1 Fit scattering data for polydisperse platelets

1.1 Instructions

About
With this notebook, you may fit scattering data of polydisperse platelets spanning a range of thicknesses L and radii R as is proposed in the manuscript "Morphological analysis of polydisperse nanoplatelets using SAXS" (2024) by L.S. van Hazendonk, R. Tuinier, E. Foschino, L. Matthews, H. Friedrich and M. Vis. Please cite the manuscript when using this script.

To run this script, you need the following: * *.csv file with your averaged and merged SAXS data of the* q*-range of interest (example file: "E72 SAXS data merged.csv")* (optional) Pre-calculated form factor table ("P11_table.csv"). Alternatively, you may calculate your own table spanning a range of thicknesses and radii of your own interest. Be aware that the calculation of this form factor table takes rather long depending on the number of data points in there.

Some tips for a better fit result: * When your fitting is slow or gives poor results, it might be worthwhile to change the bin size or to eliminate any noisy data ranges in the low- or high-q regions. * Increasing the resolution of the form factor table, i.e. decreasing the step size, generally improves the fit result.

Fitting time: The fitting with a pre-generated form factor table should not take long ($<$ 30 s for the example data file E72 from the manuscript (gibbsite) run on a 4-year old HP EliteBook 830 G6 with an Intel(R) Core(TM) i7-8665U CPU @ 1.90GHz processor, 32 GB RAM and a 64-bit Windows 11 operating system with an Anaconda Python distribution with Jupyter Notebook version 6.5.2). Be aware that the calculation of a new form factor table takes a number of hours depending on the number of data points in your table.

1.1.1 Preliminaries

[40]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import os
```

1.2 Read and bin data

[61]:
```
# Read data (csv)
columns = ["q", "I"]
input_data = pd.read_csv("E72 SAXS data merged.csv", names=columns, usecols=[0,␣
↪1]) # Replace with your data file name

#ln of q and I
log_data  = np.log(input_data)

bin_width = 0.05

if isinstance(bin_width, (int, float)):
    # Calculate bin edges
    bin_edges = np.arange(log_data["q"].min(), log_data["q"].max(), bin_width)

    # Bin the data based on q values
    log_data['bin'] = pd.cut(log_data['q'], bins=bin_edges)

    # Calculate mean values and standard deviation for each bin
    log_binned_data = log_data.groupby('bin', observed=True).agg({
        'q': ['mean'],
        'I': ['mean', 'std'],
        'bin': 'count'
    })

    # Rename columns for clarity
    log_binned_data.columns = ['ln(q)_center', 'mean_ln(I)', 'std_mean_ln(I)',␣
↪'count']

    # Add 1 column with std(I) in each bin (divide by sqrt(N))
    log_binned_data["std_mean_ln(I)"] /= np.sqrt(log_binned_data["count"])

    # Reset the index for a cleaner DataFrame
    log_binned_data.reset_index(inplace=True)

# Display the final DataFrame
log_binned_data.head()
```

[61]:

	bin	ln(q)_center	mean_ln(I)	std_mean_ln(I)	count
0	(-5.273, -5.223]	-5.244436	4.904700	NaN	1
1	(-5.223, -5.173]	-5.203084	4.885888	0.005718	2
2	(-5.173, -5.123]	-5.150361	4.869287	0.000069	2
3	(-5.123, -5.073]	-5.100279	4.867757	0.001773	2
4	(-5.073, -5.023]	-5.052586	4.861590	0.005385	2

[62]:
```
## take the exponent to return to the real values to compare with mathematica␣
 ↪script
q_values=np.exp(np.array(log_binned_data['ln(q)_center']))
mean_I_values=np.exp(np.array(log_binned_data['mean_ln(I)']))
std_mean_I_values=np.abs(mean_I_values)*(np.
 ↪array(log_binned_data['std_mean_ln(I)']))

binned_data = pd.DataFrame({
    'q_values': q_values,
    'mean_I_values': mean_I_values,
    'std_mean_I_values': std_mean_I_values
})

binned_data.head()
```

[62]:

	q_values	mean_I_values	std_mean_I_values
0	0.005277	134.922457	NaN
1	0.005500	132.407976	0.757113
2	0.005797	130.228030	0.009021
3	0.006095	130.028884	0.230476
4	0.006393	129.229508	0.695847

1.2.1 Compute slopes ('n')

[63]:
```
## Compute slopes in the binned ln data and in the ln data
# (here you lose 1 data point when doing the moving average)
# save it in dataframe called slopes_binned_data

slopes_log_binned_data = (np.array(log_binned_data["mean_ln(I)"][1:]) - np.
 ↪array(log_binned_data["mean_ln(I)"][:-1]))/(np.
 ↪array(log_binned_data["ln(q)_center"][1:]) - np.
 ↪array(log_binned_data["ln(q)_center"][:-1]))
q_log_binned_data = (np.array(log_binned_data["ln(q)_center"][1:]) + np.
 ↪array(log_binned_data["ln(q)_center"][:-1]))*0.5
std_slope  = np.sqrt((np.array(log_binned_data["std_mean_ln(I)"][1:])**2 + np.
 ↪array(log_binned_data["std_mean_ln(I)"][:-1])**2))/(np.
 ↪array(log_binned_data["ln(q)_center"][1:]) - np.
 ↪array(log_binned_data["ln(q)_center"][:-1]))

## Save in a dataframe
slopes_binned_data = pd.DataFrame({
    'ln_q': q_log_binned_data,
    'q' : np.exp(q_log_binned_data),
    'slope': slopes_log_binned_data,
    'std_slope': std_slope
})
```

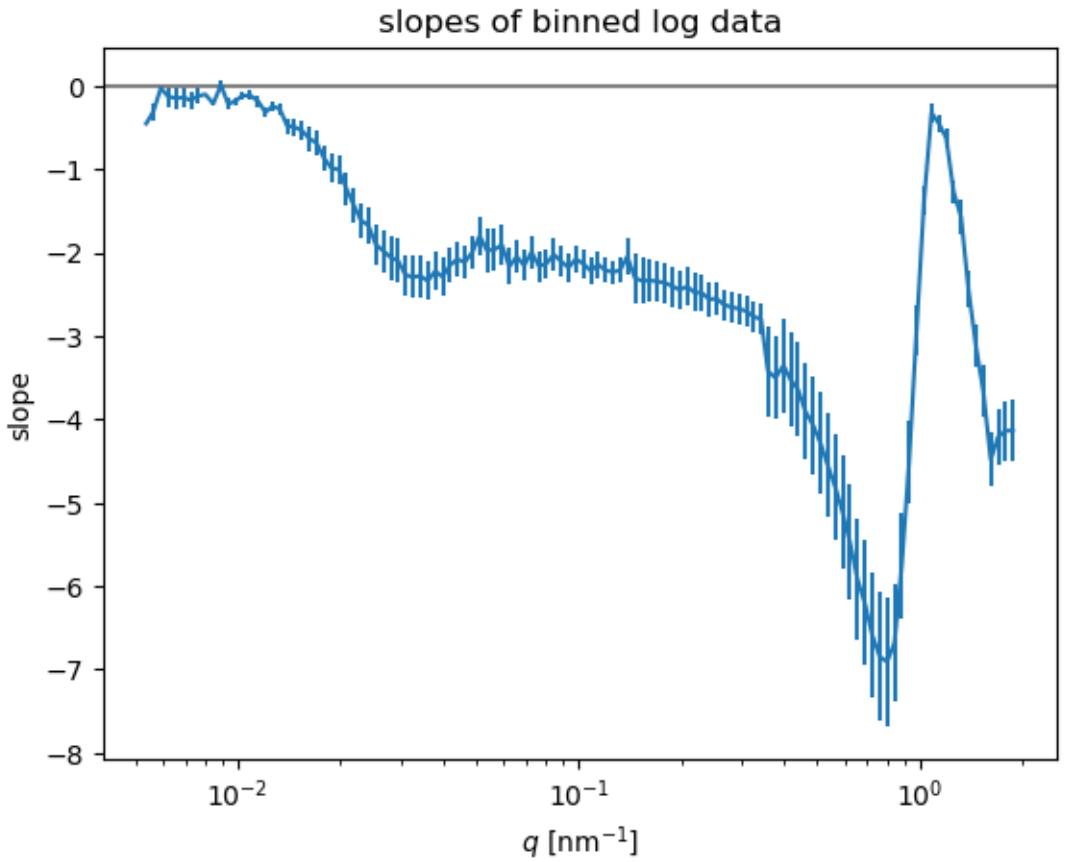
```
slopes_binned_data.head()
```

[63]:

	ln_q	q	slope	std_slope
0	-5.223760	0.005387	-0.454935	NaN
1	-5.176723	0.005646	-0.314868	0.108461
2	-5.125320	0.005944	-0.030558	0.035419
3	-5.076433	0.006242	-0.129300	0.118862
4	-5.024345	0.006576	-0.150815	0.142780

[64]:
```
plt.errorbar(x=np.exp(np.
  ↪array(slopes_binned_data["ln_q"])),y=slopes_binned_data["slope"],yerr=slopes_binned_data["s
  ↪)
plt.axhline(0,c="grey")
plt.xlabel(r"$q$ [nm$^{-1}]$")
plt.ylabel("slope")
plt.xscale("log")
plt.title("slopes of binned log data")
```

[64]: Text(0.5, 1.0, 'slopes of binned log data')



1.3 Fit the data

1.3.1 Set the values corresponding to your (imported) form factor table

[66]:
```
## Set the settings corresponding to your imported table
# (or adjust to your own preferences and regenerate the table in the cell below)
# The values below correspond to "P11_table.csv."

# Parameters for qlist
qmin = -3
qmax = 1
qstep = 0.02

# Generate qlist
qlist = 10**np.arange(qmin, qmax + qstep, qstep)

# Calculate qlistslope
#qlistslope = np.exp(np.convolve(np.log(qlist), np.ones(2)/2, mode='valid'))

# Parameters for lengths
lengthmin = -1
lengthmax = 3
lengthstep = 0.02

# Generate lengths
lengths = 10**np.arange(lengthmin, lengthmax + lengthstep, lengthstep)

# Parameters for radii
radiusmin = 1
radiusmax = 4
radiusstep = 0.02

# Generate radii
radii = 10**np.arange(radiusmin, radiusmax + radiusstep, radiusstep)

# Print or use generated lists as needed
print("qlist:", len(qlist))
print("radii:", len(radii))
print("lengths:", len(lengths))
```

```
qlist: 201
radii: 151
lengths: 201
```

1.3.2 (Optional!) Compute a new form factor table

Beware: takes a number of hours depending on the range and step size of your L and R parameters. Skip this step if you want to import a form factor table.

[8]:
```
## compute the P11 and save it in a dataframe

# from scipy.integrate import quad
# from scipy.special import jn
# from itertools import product

# def P11(q, R, L):
#     integrand = lambda alpha: ((2 * jn(1, q * R * np.sin(alpha))) / (q * R *␣
 ↪np.sin(alpha)) * np.sin((q * L * np.cos(alpha)) / 2) / ((q * L * np.
 ↪cos(alpha)) / 2))**2 * np.sin(alpha)
#     result, error = quad(integrand, 0, np.pi/2)
#     return result

# data_all_lengths = np.zeros((len(qlist), len(radii),len(lengths)))
# combinations = np.zeros((len(qlist)*len(radii)*len(lengths),4))
# for k,q in enumerate(qlist):
#     for i, r in enumerate(radii):
#         for j, l in enumerate(lengths):
#             form_factor_value = P11(q, r, l)
#             data_all_lengths[k,i,j] = form_factor_value
#             combinations[k*len(lengths)*len(radii) + i*len(lengths)+j,0] = q
#             combinations[k*len(lengths)*len(radii) + i*len(lengths)+j,1] = r
#             combinations[k*len(lengths)*len(radii) + i*len(lengths)+j,2] = l
#             combinations[k*len(lengths)*len(radii) + i*len(lengths)+j,3] =␣
 ↪form_factor_value

# # Save data to the file as .npy
# #np.save(filename, data_all_lengths)
# # Creating a DataFrame
# columns = ['q', 'R', 'L', 'P11']
# df = pd.DataFrame(combinations, columns=columns)

# # # Save the DataFrame to a CSV file
# df.to_csv('P11_newtable.csv', index=False)
```

1.3.3 Import an existing form factor table

[67]:
```
### load the dataframe with the form factor
#read form factor from csv file

file_path = 'P11_table.csv'
df_old= pd.read_csv(file_path)

# Display the DataFrame
df_old
```

[67]:

	q	R	L	P11
0	0.001	10.0	0.100000	9.999833e-01
1	0.001	10.0	0.104713	9.999833e-01
2	0.001	10.0	0.109648	9.999833e-01
3	0.001	10.0	0.114815	9.999833e-01
4	0.001	10.0	0.120226	9.999833e-01
…	…	…	…	…
6100546	10.000	10000.0	831.763771	8.934717e-20
6100547	10.000	10000.0	870.963590	1.234475e-19
6100548	10.000	10000.0	912.010839	1.356450e-19
6100549	10.000	10000.0	954.992586	1.145389e-19
6100550	10.000	10000.0	1000.000000	7.192447e-20

[6100551 rows x 4 columns]

[68]:
```
df = df_old ## copy in case you make some mistakes while reading data etc
```

1.3.4 Reshape form factor table

Needed both in case of importing a table and generating a new one.

[69]:
```
data_all_lengths=np.array(df["P11"]).reshape(len(qlist),len(radii),len(lengths))
data_all_lengths.shape
```

[69]:
```
(201, 151, 201)
```

1.3.5 Define fit functions

[70]:
```
## Probability function

from scipy.stats import norm

def probfunc(mu_r, sigma_r, mu_l, sigma_l, r,l):
    pdf_r = norm.pdf(np.log(r), loc=mu_r, scale=sigma_r)
    pdf_l = norm.pdf(np.log(l), loc=mu_l, scale=sigma_l)
    den = l * r
    return (pdf_r * pdf_l) / den

##Example probability function
# mul = 1.0
# sigmal = 0.5
# mur = 2.0
# sigmar = 0.8
# x=np.linspace(0.07,7.0,100)
# plt.plot(x,probfunc(mul,sigmal,mur,sigmar,x,x))  # (same as with mathematica)
```

[71]:
```
## Fit function
```

```
# L = 0.717, ~L = 0.908, ~D = 5.33, ~D = 0.603.
#     L = 1.85, ~L = 0.193, ~D = 5.15, ~D = 0.147

def to_minimize(params):
    muR=params[0]
    sigmaR=params[1]
    muL=params[2]
    sigmaL=params[3]

    R, L = np.meshgrid(radii, lengths)
    term1 = L**2 * R**4 # volume term
    term2 = (10**(np.log10(R) + radiusstep/2) - 10**(np.log10(R) - radiusstep/
 ↪2)) # correction for bin width R
    term3 = (10**(np.log10(L) + lengthstep/2) - 10**(np.log10(L) - lengthstep/
 ↪2)) # correction for bin width L
    probabilities = term1 * term2 * term3 * probfunc(abs(muR), abs(sigmaR),␣
 ↪abs(muL), abs(sigmaL), R, L)

    sum_prob=np.sum(probabilities)
    normalized_probabilities = probabilities.T / sum_prob

    theo_I = np.sum(normalized_probabilities[None, :, :] * data_all_lengths,␣
 ↪axis=(1, 2))

    logtheoI = np.log(theo_I)
    logqlist=np.log(qlist)

    # compute slopes of theoretical values
    slopes_theo = (logtheoI[1:] - logtheoI[:-1])/(logqlist[1:] - logqlist[:-1])
    q_theo = (logqlist[1:] + logqlist[:-1])*0.5

    # calculate difference between theoretical and experimental slopes
    slopes_theo_val=[]
    q_vals = []
    ln_q = np.array(slopes_binned_data['ln_q'])
    start_index = np.abs(q_theo-ln_q[0]).argmin()
    stop_index = np.abs(q_theo-ln_q[-1]).argmin()
    for q_exp_value_to_match in ln_q:
        closest_qtheo_index = np.abs(q_theo[start_index:stop_index] -␣
 ↪q_exp_value_to_match).argmin()
        q_vals.append(q_theo[closest_qtheo_index+start_index])
        slopes_theo_val.append(slopes_theo[closest_qtheo_index+start_index])
    diff = slopes_theo_val - np.array(slopes_binned_data['slope'])
    squares=np.dot(diff,diff)
    #squares = np.linalg.norm(slopes_theo_val - slopes_binned_data['slope'])
    print(params, squares)
    return squares
```

1.3.6 Fit the data

[72]:
```
from scipy.optimize import minimize

# Initial guess
mu_r= np.log(250.)
sigma_r=0.5
mu_l=np.log(30.)
sigma_l=0.5

initial_guess = [ mu_r, sigma_r,mu_l, sigma_l]

# Constraint: sigmal and sigmar parameters must be greater than 0
#constraint0 = ({'type': 'ineq', 'fun': lambda params: params[1]})
#constraint1 = ({'type': 'ineq', 'fun': lambda params: params[3]})
#bounds = ([0.0, 0.01, 0.0, 0.01],[50.0,5.0, 50.0, 5.0])
#bounds = ([0.0, 0.01, 0.0, 0.01],[50.0,5.0, 50.0, 5.0])
#options = {'disp': True, 'eps': 0.1}

# Perform the minimization
import time # calculate the fitting time

st = time.time()
resultNEW = minimize(to_minimize, initial_guess,method='BFGS')
et = time.time()
elapsed_time = et - st
print('Execution time:', elapsed_time, 'seconds')
# Print the result
print(resultNEW)
```

```
[5.52146092 0.5        3.40119738 0.5       ] 302.4599256595935
[5.52146093 0.5        3.40119738 0.5       ] 302.4599258916411
[5.52146092 0.50000001 3.40119738 0.5       ] 302.459925773569
[5.52146092 0.5        3.4011974  0.5       ] 302.45992681264136
[5.52146092 0.5        3.40119738 0.50000001] 302.45992613196466
[5.33729502 0.40954269 2.48607427 0.1250999 ] 619.4567511058397
[5.33729503 0.40954269 2.48607427 0.1250999 ] 619.4567528059063
[5.33729502 0.40954271 2.48607427 0.1250999 ] 619.4567532025884
[5.33729502 0.40954269 2.48607428 0.1250999 ] 619.4567353076981
[5.33729502 0.40954269 2.48607427 0.12509991] 619.4566334288872
[5.45662078 0.46815228 3.07900577 0.36800719] 282.9282018387877
[5.45662079 0.46815228 3.07900577 0.36800719] 282.92820228210667
[5.45662078 0.46815229 3.07900577 0.36800719] 282.92820195770474
[5.45662078 0.46815228 3.07900578 0.36800719] 282.92820271579313
[5.45662078 0.46815228 3.07900577 0.36800721] 282.9282006306353
[5.30998273 0.41308207 2.57781292 0.43158077] 230.9929723957663
[5.30998275 0.41308207 2.57781292 0.43158077] 230.99297318677117
[5.30998273 0.41308208 2.57781292 0.43158077] 230.99297333884203
```

```
[5.30998273 0.41308207 2.57781294 0.43158077] 230.99297408304233
[5.30998273 0.41308207 2.57781292 0.43158078] 230.9929703351399
[4.72343055 0.19280122 0.57304155 0.68587506] 159.6965329573099
[4.72343057 0.19280122 0.57304155 0.68587506] 159.69653369438433
[4.72343055 0.19280123 0.57304155 0.68587506] 159.69653361292762
[4.72343055 0.19280122 0.57304157 0.68587506] 159.69653231924448
[4.72343055 0.19280122 0.57304155 0.68587508] 159.69653126496246
[ 1.0000439  -3.57231267 -1.41955515  0.5486895 ] 263.9146021929713
[ 1.00004391 -3.57231267 -1.41955515  0.5486895 ] 263.9146022328025
[ 1.0000439  -3.57231266 -1.41955515  0.5486895 ] 263.9146020465139
[ 1.0000439  -3.57231267 -1.41955513  0.5486895 ] 263.9146026703475
[ 1.0000439  -3.57231267 -1.41955515  0.54868952] 263.9146006217419
[ 4.06699406 -0.47099182  0.22174492  0.66168912] 177.13640155084795
[ 4.06699407 -0.47099182  0.22174492  0.66168912] 177.13640181214794
[ 4.06699406 -0.47099181  0.22174492  0.66168912] 177.13640058902257
[ 4.06699406 -0.47099182  0.22174494  0.66168912] 177.1364008041799
[ 4.06699406 -0.47099182  0.22174492  0.66168914] 177.13639954050302
[4.53502807e+00 2.28735321e-03 4.72216634e-01 6.78933508e-01] 158.98109817526216
[4.53502809e+00 2.28735321e-03 4.72216634e-01 6.78933508e-01] 158.9810981501882
[4.53502807e+00 2.28736811e-03 4.72216634e-01 6.78933508e-01] 158.98109815192348
[4.53502807e+00 2.28735321e-03 4.72216649e-01 6.78933508e-01] 158.9810974960002
[4.53502807e+00 2.28735321e-03 4.72216634e-01 6.78933523e-01] 158.98109637911125
[ 4.61362717 -0.08478688  0.465202    0.6936099 ] 157.3492579191675
[ 4.61362719 -0.08478688  0.465202    0.6936099 ] 157.3492583020526
[ 4.61362717 -0.08478686  0.465202    0.6936099 ] 157.3492580461465
[ 4.61362717 -0.08478688  0.46520202  0.6936099 ] 157.34925725078682
[ 4.61362717 -0.08478688  0.465202    0.69360991] 157.34925615088392
[ 4.61037853 -0.08052819  0.5180795   0.70100326] 154.12449069217007
[ 4.61037855 -0.08052819  0.5180795   0.70100326] 154.12449106491295
[ 4.61037853 -0.08052818  0.5180795   0.70100326] 154.12449085694612
[ 4.61037853 -0.08052819  0.51807952  0.70100326] 154.1244900495446
[ 4.61037853 -0.08052819  0.5180795   0.70100328] 154.12448900687212
[ 4.59738398 -0.06349343  0.72958951  0.73057672] 142.88865355324583
[ 4.59738399 -0.06349343  0.72958951  0.73057672] 142.8886538978823
[ 4.59738398 -0.06349342  0.72958951  0.73057672] 142.8886538863475
[ 4.59738398 -0.06349343  0.72958953  0.73057672] 142.88865303801168
[ 4.59738398 -0.06349343  0.72958951  0.73057673] 142.8886522933766
[ 4.74554959 -0.41685568  1.19414276  0.83318515] 155.4109045729276
[ 4.7455496  -0.41685568  1.19414276  0.83318515] 155.41090544086828
[ 4.74554959 -0.41685566  1.19414276  0.83318515] 155.41090292161212
[ 4.74554959 -0.41685568  1.19414278  0.83318515] 155.41090451148256
[ 4.74554959 -0.41685568  1.19414276  0.83318517] 155.41090475449062
[ 4.64502313 -0.17710871  0.87895563  0.76356803] 138.18557421879987
[ 4.64502314 -0.17710871  0.87895563  0.76356803] 138.18557478730565
[ 4.64502313 -0.17710869  0.87895563  0.76356803] 138.18557372835892
[ 4.64502313 -0.17710871  0.87895564  0.76356803] 138.18557382796547
[ 4.64502313 -0.17710871  0.87895563  0.76356805] 138.1855733890105
[ 4.6118504  -0.14460599  1.14051442  0.7690376 ] 130.45989380860632
```

```
[ 4.61185042 -0.14460599  1.14051442  0.7690376 ] 130.45989427243313
[ 4.6118504  -0.14460597  1.14051442  0.7690376 ] 130.45989354702124
[ 4.6118504  -0.14460599  1.14051443  0.7690376 ] 130.45989357074393
[ 4.6118504  -0.14460599  1.14051442  0.76903761] 130.45989350982407
[ 4.52060196 -0.08160874  1.79999081  0.69914649] 123.08443652640115
[ 4.52060197 -0.08160874  1.79999081  0.69914649] 123.08443689316182
[ 4.52060196 -0.08160873  1.79999081  0.69914649] 123.08443671859197
[ 4.52060196 -0.08160874  1.79999083  0.69914649] 123.08443668908149
[ 4.52060196 -0.08160874  1.79999081  0.69914651] 123.08443758602473
[ 4.44973276 -0.04859937  2.10096477  0.58557402] 120.40638284172472
[ 4.44973277 -0.04859937  2.10096477  0.58557402] 120.4063821389139
[ 4.44973276 -0.04859936  2.10096477  0.58557402] 120.40638453017208
[ 4.44973276 -0.04859937  2.10096479  0.58557402] 120.4063835522326
[ 4.44973276 -0.04859937  2.10096477  0.58557403] 120.40638432001543
[ 4.49400164 -0.06921888  1.91295951  0.65651777] 120.76309021819134
[ 4.49400166 -0.06921888  1.91295951  0.65651777] 120.76309050775839
[ 4.49400164 -0.06921887  1.91295951  0.65651777] 120.7630902824018
[ 4.49400164 -0.06921888  1.91295953  0.65651777] 120.76309047698588
[ 4.49400164 -0.06921888  1.91295951  0.65651779] 120.7630915093562
[ 4.46705146 -0.05666606  2.02741406  0.61332837] 119.33150731800728
[ 4.46705148 -0.05666606  2.02741406  0.61332837] 119.33150718457492
[ 4.46705146 -0.05666604  2.02741406  0.61332837] 119.33150769091912
[ 4.46705146 -0.05666606  2.02741407  0.61332837] 119.3315077888485
[ 4.46705146 -0.05666606  2.02741406  0.61332838] 119.33150878188401
[ 4.4205536  -0.03844839  2.07791714  0.51103744] 114.94642600381232
[ 4.42055361 -0.03844839  2.07791714  0.51103744] 114.94642544280954
[ 4.4205536  -0.03844837  2.07791714  0.51103744] 114.94643044301318
[ 4.4205536  -0.03844839  2.07791716  0.51103744] 114.94642692513224
[ 4.4205536  -0.03844839  2.07791714  0.51103746] 114.94642773088698
[ 4.37537101 -0.04070608  2.12034158  0.40310689] 109.66356437649642
[ 4.37537103 -0.04070608  2.12034158  0.40310689] 109.66356301882666
[ 4.37537101 -0.04070607  2.12034158  0.40310689] 109.66356488968
[ 4.37537101 -0.04070608  2.12034159  0.40310689] 109.66356647371106
[ 4.37537101 -0.04070608  2.12034158  0.4031069 ] 109.66356522303501
[ 4.32762454 -0.03287846  2.04616707  0.31770375] 99.36566101399161
[ 4.32762456 -0.03287846  2.04616707  0.31770375] 99.36565914211072
[ 4.32762454 -0.03287845  2.04616707  0.31770375] 99.36566747007237
[ 4.32762454 -0.03287846  2.04616708  0.31770375] 99.36566505883914
[ 4.32762454 -0.03287846  2.04616707  0.31770377] 99.36566178552333
[ 4.13663865e+00 -1.56798797e-03  1.74946905e+00 -2.39087916e-02]
741.059403801559
[ 4.13663867e+00 -1.56798797e-03  1.74946905e+00 -2.39087916e-02]
741.0594038015594
[ 4.13663865e+00 -1.56797307e-03  1.74946905e+00 -2.39087916e-02]
741.0594038015594
[ 4.13663865e+00 -1.56798797e-03  1.74946906e+00 -2.39087916e-02]
741.0593697239945
[ 4.13663865e+00 -1.56798797e-03  1.74946905e+00 -2.39087767e-02]
```

```
741.0595097532266
[ 4.32608352 -0.03262583  2.04377308  0.31494736] 98.85975047680432
[ 4.32608354 -0.03262583  2.04377308  0.31494736] 98.85974879214714
[ 4.32608352 -0.03262581  2.04377308  0.31494736] 98.85975702663558
[ 4.32608352 -0.03262583  2.0437731   0.31494736] 98.8597546203014
[ 4.32608352 -0.03262583  2.04377308  0.31494738] 98.85975124610391
[ 4.32446113 -0.03235985  2.04125268  0.31204542] 98.28990890745575
[ 4.32446114 -0.03235985  2.04125268  0.31204542] 98.28990743124868
[ 4.32446113 -0.03235984  2.04125268  0.31204542] 98.28991549720959
[ 4.32446113 -0.03235985  2.0412527   0.31204542] 98.2899131571921
[ 4.32446113 -0.03235985  2.04125268  0.31204543] 98.28990967423579
[ 4.23054989 -0.01696392  1.89536087  0.14406831] 62.65115453208068
[ 4.2305499  -0.01696392  1.89536087  0.14406831] 62.6511546056447
[ 4.23054989 -0.0169639   1.89536087  0.14406831] 62.65115760724669
[ 4.23054989 -0.01696392  1.89536088  0.14406831] 62.65116281281804
[ 4.23054989 -0.01696392  1.89536087  0.14406833] 62.651132260341896
[ 4.30896114 -0.02981876  2.01717334  0.28432091] 90.65944845644138
[ 4.30896116 -0.02981876  2.01717334  0.28432091] 90.65944861706865
[ 4.30896114 -0.02981875  2.01717334  0.28432091] 90.65945265673841
[ 4.30896114 -0.02981876  2.01717335  0.28432091] 90.659453848203
[ 4.30896114 -0.02981876  2.01717334  0.28432093] 90.6594491973286
[ 4.25218958 -0.02051156  1.92897829  0.18277478] 48.717956272775645
[ 4.25218959 -0.02051156  1.92897829  0.18277478] 48.71795002670488
[ 4.25218958 -0.02051154  1.92897829  0.18277478] 48.717961280406755
[ 4.25218958 -0.02051156  1.9289783   0.18277478] 48.71796665326238
[ 4.25218958 -0.02051156  1.92897829  0.1827748 ] 48.71795201131186
[ 3.74233815  0.05531472  1.05212617 -0.68626691] 171.19314649393394
[ 3.74233816  0.05531472  1.05212617 -0.68626691] 171.19314528852317
[ 3.74233815  0.05531474  1.05212617 -0.68626691] 171.1931457117502
[ 3.74233815  0.05531472  1.05212619 -0.68626691] 171.19314603105124
[ 3.74233815  0.05531472  1.05212617 -0.68626689] 171.19314749093758
[ 4.19472441 -0.01196521  1.8301486   0.08482541] 295.3338919803517
[ 4.19472442 -0.01196521  1.8301486   0.08482541] 295.33389168903676
[ 4.19472441 -0.01196519  1.8301486   0.08482541] 295.3338929301089
[ 4.19472441 -0.01196521  1.83014862  0.08482541] 295.33390226359074
[ 4.19472441 -0.01196521  1.8301486   0.08482542] 295.33379478686106
[ 4.25042095 -0.02024852  1.92593656  0.17976015] 48.38301367411714
[ 4.25042096 -0.02024852  1.92593656  0.17976015] 48.383007255572664
[ 4.25042095 -0.02024851  1.92593656  0.17976015] 48.38302009556526
[ 4.25042095 -0.02024852  1.92593657  0.17976015] 48.383024059755286
[ 4.25042095 -0.02024852  1.92593656  0.17976016] 48.38300873370125
[ 4.25098703 -0.02044708  1.92510814  0.17906584] 47.71799523010754
[ 4.25098704 -0.02044708  1.92510814  0.17906584] 47.717988954460004
[ 4.25098703 -0.02044706  1.92510814  0.17906584] 47.718001056513465
[ 4.25098703 -0.02044708  1.92510816  0.17906584] 47.71800555512695
[ 4.25098703 -0.02044708  1.92510814  0.17906586] 47.7179901476066
[ 4.25325134 -0.02124129  1.92179446  0.17628861] 45.30650844962427
[ 4.25325135 -0.02124129  1.92179446  0.17628861] 45.306502973063864
```

```
[ 4.25325134 -0.02124127  1.92179446  0.17628861] 45.306512095928056
[ 4.25325134 -0.02124129  1.92179448  0.17628861] 45.30651851264509
[ 4.25325134 -0.02124129  1.92179446  0.17628862] 45.30650275477453
[ 4.24804893 -0.02181053  1.90932992  0.17516278] 39.61482080079502
[ 4.24804895 -0.02181053  1.90932992  0.17516278] 39.61481591583522
[ 4.24804893 -0.02181051  1.90932992  0.17516278] 39.61482661808437
[ 4.24804893 -0.02181053  1.90932994  0.17516278] 39.61482924839911
[ 4.24804893 -0.02181053  1.90932992  0.1751628 ] 39.6148157710515
[ 4.22365408 -0.028107    1.87688709  0.17114608] 30.027266230054742
[ 4.2236541  -0.028107    1.87688709  0.17114608] 30.027264471705884
[ 4.22365408 -0.02810698  1.87688709  0.17114608] 30.027268489710853
[ 4.22365408 -0.028107    1.8768871   0.17114608] 30.027269798943053
[ 4.22365408 -0.028107    1.87688709  0.1711461 ] 30.027260636837553
[ 4.21796527 -0.03102896  1.8596921   0.19216973] 25.532809044969174
[ 4.21796529 -0.03102896  1.8596921   0.19216973] 25.532807702530583
[ 4.21796527 -0.03102895  1.8596921   0.19216973] 25.532811195583882
[ 4.21796527 -0.03102896  1.85969212  0.19216973] 25.532809784302916
[ 4.21796527 -0.03102896  1.8596921   0.19216975] 25.532810718626866
[ 4.22974677 -0.03000905  1.86381442  0.1898371 ] 24.654456005132403
[ 4.22974678 -0.03000905  1.86381442  0.1898371 ] 24.654454555430224
[ 4.22974677 -0.03000903  1.86381442  0.1898371 ] 24.654458558155582
[ 4.22974677 -0.03000905  1.86381443  0.1898371 ] 24.65445721423199
[ 4.22974677 -0.03000905  1.86381442  0.18983712] 24.654457212825786
[ 4.24769429 -0.0293304   1.86649683  0.18536902] 22.690043370356204
[ 4.24769431 -0.0293304   1.86649683  0.18536902] 22.690041407309252
[ 4.24769429 -0.02933038  1.86649683  0.18536902] 22.690045211965025
[ 4.24769429 -0.0293304   1.86649685  0.18536902] 22.69004476813818
[ 4.24769429 -0.0293304   1.86649683  0.18536903] 22.690043424568834
[ 4.31948439 -0.02661579  1.87722649  0.16749666] 24.932472871736223
[ 4.31948441 -0.02661579  1.87722649  0.16749666] 24.932473160053146
[ 4.31948439 -0.02661578  1.87722649  0.16749666] 24.93247572404576
[ 4.31948439 -0.02661579  1.8772265   0.16749666] 24.93247538117638
[ 4.31948439 -0.02661579  1.87722649  0.16749667] 24.93246560131322
[ 4.2756104  -0.0282748   1.87066914  0.17841922] 21.21805266573535
[ 4.27561042 -0.0282748   1.87066914  0.17841922] 21.218052066322485
[ 4.2756104  -0.02827479  1.87066914  0.17841922] 21.218053986924637
[ 4.2756104  -0.0282748   1.87066915  0.17841922] 21.218054340738764
[ 4.2756104  -0.0282748   1.87066914  0.17841923] 21.21805041392701
[ 4.27763742 -0.02872041  1.86792815  0.18217352] 20.42679717191146
[ 4.27763743 -0.02872041  1.86792815  0.18217352] 20.42679658671062
[ 4.27763742 -0.02872039  1.86792815  0.18217352] 20.4267985883765
[ 4.27763742 -0.02872041  1.86792817  0.18217352] 20.42679841787721
[ 4.27763742 -0.02872041  1.86792815  0.18217353] 20.426796212992542
[ 4.2908337  -0.03052344  1.85923005  0.18499244] 19.304803979293766
[ 4.29083371 -0.03052344  1.85923005  0.18499244] 19.30480344697061
[ 4.2908337  -0.03052342  1.85923005  0.18499244] 19.304805338587855
[ 4.2908337  -0.03052344  1.85923006  0.18499244] 19.304803917449348
[ 4.2908337  -0.03052344  1.85923005  0.18499246] 19.304803784622543
```

```
[ 4.3520064  -0.03021608  1.85976786  0.19317686] 17.089828073208714
[ 4.35200641 -0.03021608  1.85976786  0.19317686] 17.089827061276402
[ 4.3520064  -0.03021606  1.85976786  0.19317686] 17.08982915554542
[ 4.3520064  -0.03021608  1.85976787  0.19317686] 17.089828085039596
[ 4.3520064  -0.03021608  1.85976786  0.19317687] 17.089829400773382
[ 4.59669721 -0.02898664  1.86191908  0.22591452] 23.694139666817566
[ 4.59669722 -0.02898664  1.86191908  0.22591452] 23.69413994497777
[ 4.59669721 -0.02898662  1.86191908  0.22591452] 23.694141369036785
[ 4.59669721 -0.02898664  1.8619191   0.22591452] 23.694140247473342
[ 4.59669721 -0.02898664  1.86191908  0.22591454] 23.694146029860356
[ 4.41868488 -0.02988105  1.86035407  0.2020979 ] 17.456161931210676
[ 4.4186849  -0.02988105  1.86035407  0.2020979 ] 17.456161940065186
[ 4.41868488 -0.02988104  1.86035407  0.2020979 ] 17.45616457361889
[ 4.41868488 -0.02988105  1.86035408  0.2020979 ] 17.456162304705693
[ 4.41868488 -0.02988105  1.86035407  0.20209792] 17.45616459265168
[ 4.38700325 -0.03245457  1.85390067  0.19316353] 15.904199750911896
[ 4.38700326 -0.03245457  1.85390067  0.19316353] 15.904199646820594
[ 4.38700325 -0.03245456  1.85390067  0.19316353] 15.904200403522086
[ 4.38700325 -0.03245457  1.85390069  0.19316353] 15.904199090239144
[ 4.38700325 -0.03245457  1.85390067  0.19316355] 15.904200389975632
[ 4.38991888 -0.0322656   1.8565253   0.1915578 ] 15.771014111862693
[ 4.3899189 -0.0322656  1.8565253  0.1915578] 15.771014121438576
[ 4.38991888 -0.03226559  1.8565253   0.1915578 ] 15.771014802911118
[ 4.38991888 -0.0322656   1.85652531  0.1915578 ] 15.771013770783034
[ 4.38991888 -0.0322656   1.8565253   0.19155782] 15.771014284155118
[ 4.38162615 -0.0323986   1.85966459  0.18950586] 15.775780011439634
[ 4.38162617 -0.0323986   1.85966459  0.18950586] 15.775779858622965
[ 4.38162615 -0.03239858  1.85966459  0.18950586] 15.775780643024099
[ 4.38162615 -0.0323986   1.8596646   0.18950586] 15.775779984368837
[ 4.38162615 -0.0323986   1.85966459  0.18950588] 15.775779707469098
[ 4.38595973 -0.0323291   1.85802407  0.19057816] 15.745452792337487
[ 4.38595974 -0.0323291   1.85802407  0.19057816] 15.745452722007242
[ 4.38595973 -0.03232908  1.85802407  0.19057816] 15.745453434299938
[ 4.38595973 -0.0323291   1.85802409  0.19057816] 15.74545259764225
[ 4.38595973 -0.0323291   1.85802407  0.19057817] 15.745452737044097
[ 4.38922034 -0.03278577  1.85934764  0.19038333] 15.712057292637724
[ 4.38922035 -0.03278577  1.85934764  0.19038333] 15.712057309754492
[ 4.38922034 -0.03278575  1.85934764  0.19038333] 15.712057969393877
[ 4.38922034 -0.03278577  1.85934765  0.19038333] 15.712057286234977
[ 4.38922034 -0.03278577  1.85934764  0.19038335] 15.712057154216254
[ 4.39115686 -0.03393863  1.85992398  0.19041945] 15.662707677798455
[ 4.39115687 -0.03393863  1.85992398  0.19041945] 15.662707714183071
[ 4.39115686 -0.03393861  1.85992398  0.19041945] 15.662708405247084
[ 4.39115686 -0.03393863  1.859924    0.19041945] 15.662707762799108
[ 4.39115686 -0.03393863  1.85992398  0.19041946] 15.662707540922167
[ 4.39890292 -0.03855005  1.86222938  0.1905639 ] 15.451299916895433
[ 4.39890294 -0.03855005  1.86222938  0.1905639 ] 15.451299903563688
[ 4.39890292 -0.03855004  1.86222938  0.1905639 ] 15.451300880766759
```

```
[ 4.39890292 -0.03855005  1.8622294   0.1905639 ] 15.451300349086964
[ 4.39890292 -0.03855005  1.86222938  0.19056392] 15.451299777960099
[ 4.42988719 -0.05699576  1.87145097  0.19114171] 14.369582758862204
[ 4.42988721 -0.05699576  1.87145097  0.19114171] 14.369582444619057
[ 4.42988719 -0.05699574  1.87145097  0.19114171] 14.369583636316737
[ 4.42988719 -0.05699576  1.87145098  0.19114171] 14.369584467831647
[ 4.42988719 -0.05699576  1.87145097  0.19114173] 14.369582650502212
[ 4.53821695 -0.11932721  1.89987022  0.19337197] 19.78954358634221
[ 4.53821696 -0.11932721  1.89987022  0.19337197] 19.789543823159505
[ 4.53821695 -0.1193272   1.89987022  0.19337197] 19.789543577774523
[ 4.53821695 -0.11932721  1.89987023  0.19337197] 19.789549057667728
[ 4.53821695 -0.11932721  1.89987022  0.19337198] 19.789543154264933
[ 4.4457795  -0.06613997  1.87562016  0.1914689 ] 14.171899336118791
[ 4.44577951 -0.06613997  1.87562016  0.1914689 ] 14.171899085479879
[ 4.4457795  -0.06613996  1.87562016  0.1914689 ] 14.171899929168344
[ 4.4457795  -0.06613997  1.87562017  0.1914689 ] 14.171901624548426
[ 4.4457795  -0.06613997  1.87562016  0.19146891] 14.17189926953389
[ 4.44628981 -0.06818265  1.87435379  0.19164828] 13.896397294234244
[ 4.44628983 -0.06818265  1.87435379  0.19164828] 13.896397049626534
[ 4.44628981 -0.06818263  1.87435379  0.19164828] 13.896397853372452
[ 4.44628981 -0.06818265  1.87435381  0.19164828] 13.896399425859105
[ 4.44628981 -0.06818265  1.87435379  0.19164829] 13.896397314880215
[ 4.44833108 -0.07635335  1.86928832  0.19236578] 12.980055343116344
[ 4.4483311  -0.07635335  1.86928832  0.19236578] 12.980055119883843
[ 4.44833108 -0.07635334  1.86928832  0.19236578] 12.980055794807994
[ 4.44833108 -0.07635335  1.86928834  0.19236578] 12.98005684612952
[ 4.44833108 -0.07635335  1.86928832  0.1923658 ] 12.9800556861962
[ 4.46535063 -0.10668653  1.85733251  0.19118643] 11.520980240961919
[ 4.46535064 -0.10668653  1.85733251  0.19118643] 11.520980144033196
[ 4.46535063 -0.10668652  1.85733251  0.19118643] 11.520980435164464
[ 4.46535063 -0.10668653  1.85733253  0.19118643] 11.520980272566673
[ 4.46535063 -0.10668653  1.85733251  0.19118645] 11.520980355789284
[ 4.48057317 -0.12188389  1.85586383  0.19093713] 11.332417061834295
[ 4.48057319 -0.12188389  1.85586383  0.19093713] 11.332417053866493
[ 4.48057317 -0.12188387  1.85586383  0.19093713] 11.33241714374055
[ 4.48057317 -0.12188389  1.85586385  0.19093713] 11.33241692455448
[ 4.48057317 -0.12188389  1.85586383  0.19093714] 11.33241707955155
[ 4.48731497 -0.12761938  1.85631693  0.1909213 ] 11.31046140713428
[ 4.48731498 -0.12761938  1.85631693  0.1909213 ] 11.310461435347918
[ 4.48731497 -0.12761936  1.85631693  0.1909213 ] 11.310461447168393
[ 4.48731497 -0.12761938  1.85631695  0.1909213 ] 11.31046133786355
[ 4.48731497 -0.12761938  1.85631693  0.19092132] 11.310461410892598
[ 4.48791491 -0.12850705  1.85648283  0.19087169] 11.308824548964155
[ 4.48791493 -0.12850705  1.85648283  0.19087169] 11.308824581419081
[ 4.48791491 -0.12850704  1.85648283  0.19087169] 11.308824583581659
[ 4.48791491 -0.12850705  1.85648284  0.19087169] 11.30882450196457
[ 4.48791491 -0.12850705  1.85648283  0.19087171] 11.308824537663966
[ 4.48799913 -0.12975713  1.85668553  0.19080105] 11.306005109384595
```

```
[ 4.48799915 -0.12975713  1.85668553  0.19080105] 11.306005145243628
[ 4.48799913 -0.12975711  1.85668553  0.19080105] 11.306005138044545
[ 4.48799913 -0.12975713  1.85668555  0.19080105] 11.306005088922
[ 4.48799913 -0.12975713  1.85668553  0.19080106] 11.306005078183828
[ 4.4872632  -0.13172013  1.85692393  0.19071575] 11.301050188181488
[ 4.48726322 -0.13172013  1.85692393  0.19071575] 11.301050226386675
[ 4.4872632  -0.13172012  1.85692393  0.19071575] 11.301050209381135
[ 4.4872632  -0.13172013  1.85692394  0.19071575] 11.30105019825671
[ 4.4872632  -0.13172013  1.85692393  0.19071576] 11.301050135043257
[ 4.48496055 -0.13499603  1.85720784  0.19061383] 11.292539969100922
[ 4.48496056 -0.13499603  1.85720784  0.19061383] 11.292540007607585
[ 4.48496055 -0.13499602  1.85720784  0.19061383] 11.292539980246614
[ 4.48496055 -0.13499603  1.85720785  0.19061383] 11.292540014585448
[ 4.48496055 -0.13499603  1.85720784  0.19061384] 11.292539893519269
[ 4.4793559  -0.14063942  1.85751675  0.1905013 ] 11.279218071682513
[ 4.47935591 -0.14063942  1.85751675  0.1905013 ] 11.279218105622409
[ 4.4793559  -0.1406394   1.85751675  0.1905013 ] 11.279218069469598
[ 4.4793559  -0.14063942  1.85751676  0.1905013 ] 11.27921815398304
[ 4.4793559  -0.14063942  1.85751675  0.19050132] 11.279217979685997
[ 4.46983827 -0.14768388  1.85761489  0.19045406] 11.266012615636367
[ 4.46983829 -0.14768388  1.85761489  0.19045406] 11.266012636635443
[ 4.46983827 -0.14768386  1.85761489  0.19045406] 11.266012603602753
[ 4.46983827 -0.14768388  1.85761491  0.19045406] 11.266012705508349
[ 4.46983827 -0.14768388  1.85761489  0.19045407] 11.266012536697914
[ 4.46385381 -0.15034375  1.8573621   0.19052216] 11.260481463936648
[ 4.46385383 -0.15034375  1.8573621   0.19052216] 11.26048147262555
[ 4.46385381 -0.15034374  1.8573621   0.19052216] 11.260481454653316
[ 4.46385381 -0.15034375  1.85736211  0.19052216] 11.260481517259713
[ 4.46385381 -0.15034375  1.8573621   0.19052217] 11.260481421336873
[ 4.46147033 -0.15031347  1.85706345  0.19061427] 11.25889209673844
[ 4.46147035 -0.15031347  1.85706345  0.19061427] 11.258892097515668
[ 4.46147033 -0.15031345  1.85706345  0.19061427] 11.258892093791422
[ 4.46147033 -0.15031347  1.85706346  0.19061427] 11.258892109229468
[ 4.46147033 -0.15031347  1.85706345  0.19061428] 11.25889208736919
[ 4.46174322 -0.14965178  1.85697886  0.19064579] 11.25876901750566
[ 4.46174324 -0.14965178  1.85697886  0.19064579] 11.258769017286554
[ 4.46174322 -0.14965176  1.85697886  0.19064579] 11.258769016942537
[ 4.46174322 -0.14965178  1.85697887  0.19064579] 11.258769018863156
[ 4.46174322 -0.14965178  1.85697886  0.19064581] 11.258769016517151
[ 4.4620332  -0.14937074  1.85696618  0.1906525 ] 11.2587601942263
[ 4.46203321 -0.14937074  1.85696618  0.1906525 ] 11.258760194160601
[ 4.4620332  -0.14937073  1.85696618  0.1906525 ] 11.25876019422013
[ 4.4620332  -0.14937074  1.8569662   0.1906525 ] 11.258760194023644
[ 4.4620332  -0.14937074  1.85696618  0.19065252] 11.258760194396913
[ 4.46207193 -0.14934721  1.85696736  0.19065238] 11.258760089687966
[ 4.46207194 -0.14934721  1.85696736  0.19065238] 11.258760089683397
[ 4.46207193 -0.1493472   1.85696736  0.19065238] 11.258760089690394
[ 4.46207193 -0.14934721  1.85696737  0.19065238] 11.25876008965629
```

```
[ 4.46207193 -0.14934721  1.85696736  0.19065239] 11.2587600897176
[ 4.46207368 -0.14934674  1.85696758  0.1906523 ] 11.258760089135325
[ 4.46207369 -0.14934674  1.85696758  0.1906523 ] 11.258760089135404
[ 4.46207368 -0.14934673  1.85696758  0.1906523 ] 11.258760089135315
[ 4.46207368 -0.14934674  1.8569676   0.1906523 ] 11.258760089133965
[ 4.46207368 -0.14934674  1.85696758  0.19065231] 11.258760089137446
[ 4.46207363 -0.14934678  1.85696759  0.19065229] 11.25876008913425
[ 4.46207364 -0.14934678  1.85696759  0.19065229] 11.258760089134322
[ 4.46207363 -0.14934676  1.85696759  0.19065229] 11.25876008913438
[ 4.46207363 -0.14934678  1.85696761  0.19065229] 11.258760089134345
[ 4.46207363 -0.14934678  1.85696759  0.1906523 ] 11.258760089134535
[ 4.46207347 -0.14934691  1.8569676   0.19065228] 11.258760089136391
[ 4.46207348 -0.14934691  1.8569676   0.19065228] 11.258760089136306
[ 4.46207347 -0.14934689  1.8569676   0.19065228] 11.258760089136123
[ 4.46207347 -0.14934691  1.85696761  0.19065228] 11.258760089137143
[ 4.46207347 -0.14934691  1.8569676   0.1906523 ] 11.258760089134627
[ 4.4620736  -0.1493468   1.85696759  0.19065229] 11.258760089134242
[ 4.46207361 -0.1493468   1.85696759  0.19065229] 11.258760089134501
[ 4.4620736  -0.14934679  1.85696759  0.19065229] 11.258760089134334
[ 4.4620736  -0.1493468   1.85696761  0.19065229] 11.25876008913444
[ 4.4620736  -0.1493468   1.85696759  0.1906523 ] 11.258760089134059
[ 4.46207357 -0.14934682  1.85696759  0.19065229] 11.258760089134348
[ 4.46207359 -0.14934682  1.85696759  0.19065229] 11.258760089134421
[ 4.46207357 -0.14934681  1.85696759  0.19065229] 11.258760089134496
[ 4.46207357 -0.14934682  1.85696761  0.19065229] 11.258760089134684
[ 4.46207357 -0.14934682  1.85696759  0.1906523 ] 11.258760089133967
[ 4.46207359 -0.1493468   1.85696759  0.19065229] 11.258760089134348
[ 4.46207361 -0.1493468   1.85696759  0.19065229] 11.258760089134519
[ 4.46207359 -0.14934679  1.85696759  0.19065229] 11.258760089134302
[ 4.46207359 -0.1493468   1.85696761  0.19065229] 11.258760089134489
[ 4.46207359 -0.1493468   1.85696759  0.1906523 ] 11.258760089134212
[ 4.4620736  -0.1493468   1.85696759  0.19065229] 11.258760089134281
[ 4.46207361 -0.1493468   1.85696759  0.19065229] 11.25876008913432
[ 4.4620736  -0.14934679  1.85696759  0.19065229] 11.258760089134249
[ 4.4620736  -0.1493468   1.85696761  0.19065229] 11.258760089134583
[ 4.4620736  -0.1493468   1.85696759  0.1906523 ] 11.258760089134183
[ 4.4620736  -0.1493468   1.85696759  0.19065229] 11.25876008913436
[ 4.46207361 -0.1493468   1.85696759  0.19065229] 11.25876008913438
[ 4.4620736  -0.14934679  1.85696759  0.19065229] 11.25876008913432
[ 4.4620736  -0.1493468   1.85696761  0.19065229] 11.258760089134585
[ 4.4620736  -0.1493468   1.85696759  0.1906523 ] 11.258760089134256
[ 4.4620736  -0.1493468   1.85696759  0.19065229] 11.258760089134242
[ 4.46207361 -0.1493468   1.85696759  0.19065229] 11.258760089134352
[ 4.4620736  -0.14934679  1.85696759  0.19065229] 11.258760089134231
[ 4.4620736  -0.1493468   1.85696761  0.19065229] 11.258760089134459
[ 4.4620736  -0.1493468   1.85696759  0.1906523 ] 11.258760089134135
[ 4.46207359 -0.1493468   1.85696759  0.19065229] 11.258760089134196
[ 4.46207361 -0.1493468   1.85696759  0.19065229] 11.258760089134489
```

```
[ 4.46207359 -0.14934679  1.85696759  0.19065229] 11.258760089134416
[ 4.46207359 -0.1493468   1.85696761  0.19065229] 11.258760089134583
[ 4.46207359 -0.1493468   1.85696759  0.1906523 ] 11.258760089134217
[ 4.46207358 -0.14934681  1.85696759  0.19065229] 11.258760089134288
[ 4.4620736  -0.14934681  1.85696759  0.19065229] 11.258760089134368
[ 4.46207358 -0.14934679  1.85696759  0.19065229] 11.258760089134242
[ 4.46207358 -0.14934681  1.85696761  0.19065229] 11.258760089134498
[ 4.46207358 -0.14934681  1.85696759  0.1906523 ] 11.258760089134068
[ 4.46207359 -0.1493468   1.85696759  0.19065229] 11.258760089134183
[ 4.46207361 -0.1493468   1.85696759  0.19065229] 11.258760089134373
[ 4.46207359 -0.14934679  1.85696759  0.19065229] 11.258760089134293
[ 4.46207359 -0.1493468   1.85696761  0.19065229] 11.258760089134505
[ 4.46207359 -0.1493468   1.85696759  0.1906523 ] 11.25876008913427
[ 4.46207359 -0.1493468   1.85696759  0.19065229] 11.258760089134267
[ 4.4620736  -0.1493468   1.85696759  0.19065229] 11.25876008913433
[ 4.46207359 -0.14934679  1.85696759  0.19065229] 11.25876008913437
[ 4.46207359 -0.1493468   1.85696761  0.19065229] 11.258760089134496
[ 4.46207359 -0.1493468   1.85696759  0.1906523 ] 11.25876008913417
[ 4.46207359 -0.1493468   1.85696759  0.19065229] 11.258760089134189
[ 4.46207361 -0.1493468   1.85696759  0.19065229] 11.25876008913435
[ 4.46207359 -0.14934679  1.85696759  0.19065229] 11.258760089134228
[ 4.46207359 -0.1493468   1.85696761  0.19065229] 11.25876008913444
[ 4.46207359 -0.1493468   1.85696759  0.1906523 ] 11.258760089134226
[ 4.46207359 -0.1493468   1.85696759  0.19065229] 11.258760089134265
[ 4.46207361 -0.1493468   1.85696759  0.19065229] 11.258760089134327
[ 4.46207359 -0.14934679  1.85696759  0.19065229] 11.258760089134203
[ 4.46207359 -0.1493468   1.85696761  0.19065229] 11.258760089134555
[ 4.46207359 -0.1493468   1.85696759  0.1906523 ] 11.258760089134212
[ 4.46207359 -0.1493468   1.85696759  0.19065229] 11.258760089134277
[ 4.46207361 -0.1493468   1.85696759  0.19065229] 11.258760089134507
[ 4.46207359 -0.14934679  1.85696759  0.19065229] 11.258760089134185
[ 4.46207359 -0.1493468   1.85696761  0.19065229] 11.258760089134505
[ 4.46207359 -0.1493468   1.85696759  0.1906523 ] 11.258760089134189
[ 4.46207359 -0.1493468   1.85696759  0.19065229] 11.258760089134183
[ 4.46207361 -0.1493468   1.85696759  0.19065229] 11.258760089134373
[ 4.46207359 -0.14934679  1.85696759  0.19065229] 11.258760089134293
[ 4.46207359 -0.1493468   1.85696761  0.19065229] 11.258760089134505
[ 4.46207359 -0.1493468   1.85696759  0.1906523 ] 11.25876008913427
[ 4.46207359 -0.1493468   1.85696759  0.19065229] 11.258760089134277
[ 4.46207361 -0.1493468   1.85696759  0.19065229] 11.258760089134471
[ 4.46207359 -0.14934679  1.85696759  0.19065229] 11.258760089134263
[ 4.46207359 -0.1493468   1.85696761  0.19065229] 11.25876008913447
[ 4.46207359 -0.1493468   1.85696759  0.1906523 ] 11.258760089134086
[ 4.46207359 -0.1493468   1.85696759  0.19065229] 11.258760089134183
[ 4.46207361 -0.1493468   1.85696759  0.19065229] 11.258760089134373
[ 4.46207359 -0.14934679  1.85696759  0.19065229] 11.258760089134293
[ 4.46207359 -0.1493468   1.85696761  0.19065229] 11.258760089134505
[ 4.46207359 -0.1493468   1.85696759  0.1906523 ] 11.25876008913427
```

```
[ 4.46207359 -0.1493468   1.85696759  0.19065229] 11.25876008913422
[ 4.46207361 -0.1493468   1.85696759  0.19065229] 11.258760089134451
[ 4.46207359 -0.14934679  1.85696759  0.19065229] 11.258760089134267
[ 4.46207359 -0.1493468   1.85696761  0.19065229] 11.258760089134476
[ 4.46207359 -0.1493468   1.85696759  0.1906523 ] 11.258760089134277
[ 4.46207359 -0.1493468   1.85696759  0.19065229] 11.258760089134183
[ 4.46207361 -0.1493468   1.85696759  0.19065229] 11.258760089134373
[ 4.46207359 -0.14934679  1.85696759  0.19065229] 11.258760089134293
[ 4.46207359 -0.1493468   1.85696761  0.19065229] 11.258760089134505
[ 4.46207359 -0.1493468   1.85696759  0.1906523 ] 11.25876008913427
[ 4.46207359 -0.1493468   1.85696759  0.19065229] 11.258760089134318
[ 4.46207361 -0.1493468   1.85696759  0.19065229] 11.258760089134377
[ 4.46207359 -0.14934679  1.85696759  0.19065229] 11.258760089134212
[ 4.46207359 -0.1493468   1.85696761  0.19065229] 11.25876008913449
[ 4.46207359 -0.1493468   1.85696759  0.1906523 ] 11.258760089134054
[ 4.46207359 -0.1493468   1.85696759  0.19065229] 11.258760089134183
[ 4.46207361 -0.1493468   1.85696759  0.19065229] 11.258760089134373
[ 4.46207359 -0.14934679  1.85696759  0.19065229] 11.258760089134293
[ 4.46207359 -0.1493468   1.85696761  0.19065229] 11.258760089134505
[ 4.46207359 -0.1493468   1.85696759  0.1906523 ] 11.25876008913427
[ 4.46207359 -0.1493468   1.85696759  0.19065229] 11.258760089134292
[ 4.46207361 -0.1493468   1.85696759  0.19065229] 11.258760089134434
[ 4.46207359 -0.14934679  1.85696759  0.19065229] 11.258760089134313
[ 4.46207359 -0.1493468   1.85696761  0.19065229] 11.258760089134489
[ 4.46207359 -0.1493468   1.85696759  0.1906523 ] 11.258760089134187
[ 4.46207359 -0.1493468   1.85696759  0.19065229] 11.258760089134183
[ 4.46207361 -0.1493468   1.85696759  0.19065229] 11.258760089134373
[ 4.46207359 -0.14934679  1.85696759  0.19065229] 11.258760089134293
[ 4.46207359 -0.1493468   1.85696761  0.19065229] 11.258760089134505
[ 4.46207359 -0.1493468   1.85696759  0.1906523 ] 11.25876008913427
[ 4.46207359 -0.1493468   1.85696759  0.19065229] 11.258760089134194
[ 4.46207361 -0.1493468   1.85696759  0.19065229] 11.258760089134359
[ 4.46207359 -0.14934679  1.85696759  0.19065229] 11.258760089134418
[ 4.46207359 -0.1493468   1.85696761  0.19065229] 11.258760089134483
[ 4.46207359 -0.1493468   1.85696759  0.1906523 ] 11.258760089134277
[ 4.46207359 -0.1493468   1.85696759  0.19065229] 11.258760089134183
[ 4.46207361 -0.1493468   1.85696759  0.19065229] 11.258760089134373
[ 4.46207359 -0.14934679  1.85696759  0.19065229] 11.258760089134293
[ 4.46207359 -0.1493468   1.85696761  0.19065229] 11.258760089134505
[ 4.46207359 -0.1493468   1.85696759  0.1906523 ] 11.25876008913427
[ 4.46207359 -0.1493468   1.85696759  0.19065229] 11.258760089134356
[ 4.46207361 -0.1493468   1.85696759  0.19065229] 11.25876008913439
[ 4.46207359 -0.14934679  1.85696759  0.19065229] 11.258760089134423
[ 4.46207359 -0.1493468   1.85696761  0.19065229] 11.25876008913449
[ 4.46207359 -0.1493468   1.85696759  0.1906523 ] 11.258760089134341
[ 4.46207359 -0.1493468   1.85696759  0.19065229] 11.258760089134183
[ 4.46207361 -0.1493468   1.85696759  0.19065229] 11.258760089134373
[ 4.46207359 -0.14934679  1.85696759  0.19065229] 11.258760089134293
```

```
[ 4.46207359 -0.1493468   1.85696761  0.19065229] 11.258760089134505
[ 4.46207359 -0.1493468   1.85696759  0.1906523 ] 11.25876008913427
[ 4.46207359 -0.1493468   1.85696759  0.19065229] 11.258760089134196
[ 4.46207361 -0.1493468   1.85696759  0.19065229] 11.258760089134489
[ 4.46207359 -0.14934679  1.85696759  0.19065229] 11.258760089134416
[ 4.46207359 -0.1493468   1.85696761  0.19065229] 11.258760089134583
[ 4.46207359 -0.1493468   1.85696759  0.1906523 ] 11.258760089134217
[ 4.46207359 -0.1493468   1.85696759  0.19065229] 11.25876008913435
[ 4.46207359 -0.1493468   1.85696759  0.19065229] 11.2587600891343
[ 4.46207359 -0.1493468   1.85696759  0.19065229] 11.258760089134427
[ 4.46207359 -0.1493468   1.85696759  0.19065229] 11.258760089134324
[ 4.46207359 -0.1493468   1.85696759  0.19065229] 11.258760089134245
[ 4.46207359 -0.1493468   1.85696759  0.19065229] 11.258760089134284
[ 4.46207359 -0.1493468   1.85696759  0.19065229] 11.258760089134306
[ 4.46207359 -0.1493468   1.85696759  0.19065229] 11.258760089134302
[ 4.46207359 -0.1493468   1.85696759  0.19065229] 11.258760089134373
[ 4.46207359 -0.1493468   1.85696759  0.19065229] 11.258760089134341
[ 4.46207359 -0.1493468   1.85696759  0.19065229] 11.25876008913434
[ 4.46207359 -0.1493468   1.85696759  0.19065229] 11.258760089134281
Execution time: 23.107139348983765 seconds
  message: Desired error not necessarily achieved due to precision loss.
  success: False
   status: 2
      fun: 11.258760089134242
        x: [ 4.462e+00 -1.493e-01  1.857e+00  1.907e-01]
      nit: 46
      jac: [ 7.391e-06 -7.153e-07  1.454e-05 -7.153e-06]
 hess_inv: [[ 1.402e-02  5.881e-03 -3.616e-04  3.005e-04]
            [ 5.881e-03  3.715e-03 -1.704e-04  1.096e-04]
            [-3.616e-04 -1.704e-04  9.912e-05  1.757e-06]
            [ 3.005e-04  1.096e-04  1.757e-06  4.699e-05]]
     nfev: 512
     njev: 100
```

1.3.7 Display fit parameters

[73]:
```
## Fit parameters  L,~L,~D,~D
## Extract fit parameters from cells above
muL= abs(resultNEW['x'][2])
sigmaL = abs(resultNEW['x'][3])
muR = abs(resultNEW['x'][0])
sigmaR = abs(resultNEW['x'][1])

# Calculate expected value and standard deviation (not on log scale)
muLx = np.exp(muL + sigmaL**2/2)
sigmaLx = np.exp(muL) * np.sqrt(np.exp(sigmaL**2) * (np.exp(sigmaL**2) - 1))
muRx = np.exp(muR + sigmaR**2/2)
```

```
sigmaRx = np.exp(muR) * np.sqrt(np.exp(sigmaR**2) * (np.exp(sigmaR**2) - 1))

# convert radius to diameter
muD = muR + np.log(2)
sigmaD = sigmaR
muDx = 2*muRx
sigmaDx = 2*sigmaRx

params = [muL, sigmaL, muR, sigmaR, muLx, sigmaLx, muRx, sigmaRx, muD, sigmaD,␣
 ↪muDx, sigmaDx]

print(params)
```

```
[1.8569675935593666, 0.19065228830921502, 4.462073596724697, 0.1493468003407359,
6.521743453769023, 1.2547700759996905, 87.63897596618733, 13.161924485054966,
5.155220777284643, 0.1493468003407359, 175.27795193237466, 26.323848970109932]
```

1.4 Plots

1.4.1 Calculate theoretical intensities and slopes based on fit parameters

[74]:
```
## Calculate probabilities based on the fit parameters
R,L = np.meshgrid(radii, lengths)
term1 = L**2 * R**4
term2 = (10**(np.log10(R) + radiusstep/2) - 10**(np.log10(R) - radiusstep/2))
term3 = (10**(np.log10(L) + lengthstep/2) - 10**(np.log10(L) - lengthstep/2))
probabilities = term1 * term2 * term3 * probfunc(abs(muR), abs(sigmaR),␣
 ↪abs(muL), abs(sigmaL), R, L)
sum_prob=np.sum(probabilities)
normalized_probabilities = probabilities.T / sum_prob

theo_I = np.sum(normalized_probabilities[None,:, :] * data_all_lengths,␣
 ↪axis=(1, 2))

logtheoI = np.log(theo_I)
logqlist=np.log(qlist)

## Compute slopes of theoretical values
slopes_theo = (logtheoI[1:] - logtheoI[:-1])/(logqlist[1:] - logqlist[:-1])
q_theo = (logqlist[1:] + logqlist[:-1])*0.5

slopes_theo_val=[]
q_vals = []
ln_q = np.array(slopes_binned_data['ln_q'])
start_index = np.abs(q_theo-ln_q[0]).argmin()
stop_index = np.abs(q_theo-ln_q[-1]).argmin()
for q_exp_value_to_match in ln_q:
```

```
    closest_qtheo_index = np.abs(q_theo[start_index:stop_index] -␣
 ↪q_exp_value_to_match).argmin()
    q_vals.append(q_theo[closest_qtheo_index+start_index])
    slopes_theo_val.append(slopes_theo[closest_qtheo_index+start_index])
```

1.4.2 Choose output folder

Only necessary if you want to export your plots.

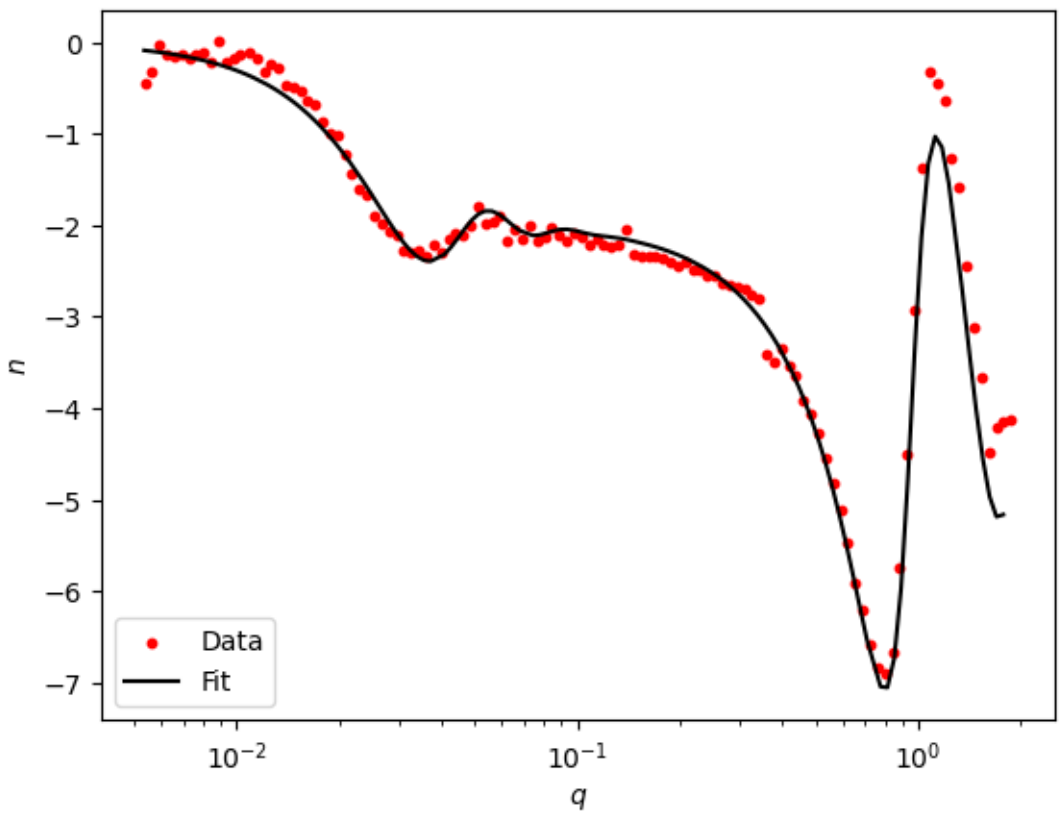
[75]:
```
# outpath = r"C:\Users\Analysis\Python\Figures" # Example path
# outpath = r"C:\Users\s168562\OneDrive - TU␣
 ↪Eindhoven\PhD\4_Colloidal_interactions\Manuscript SAXS\Data & Code\SAXS␣
 ↪analysis Python\Working example (E72)"

# Plot preliminaries
# plt.style.use('default') # default, ggplot, seaborn-*, dark_background,␣
 ↪fivethirtyeight
# plt.rcParams['font.size'] = 8 # set plot font size matplotlib
# cm = 1/2.54  # convert centimeters to inches
```

1.4.3 Plot the slope fit

[ ]:
```
```

[76]:
```
# plt.figure(figsize= [5.5*cm, 3.8*cm])
plt.figure()
ax = plt.axes()
# plt.
 ↪scatter(slopes_binned_data['ln_q'],slopes_binned_data['slope'],label="Data",s=10,␣
 ↪color = "red")
# plt.plot(q_vals,slopes_theo_val,label="Fit", color = "black")
plt.
 ↪scatter(slopes_binned_data['q'],slopes_binned_data['slope'],label="Data",s=10,␣
 ↪color = "red")
plt.plot(np.exp(q_vals),slopes_theo_val,label="Fit", color = "black")
plt.ylabel("$n$")
plt.xlabel("$q$")
ax.set_xscale('log')
plt.legend()
# figurepath = os.path.join(outpath, "SlopeFit.pdf")
# plt.savefig(figurepath, dpi = 300, bbox_inches='tight', transparent = True)
```

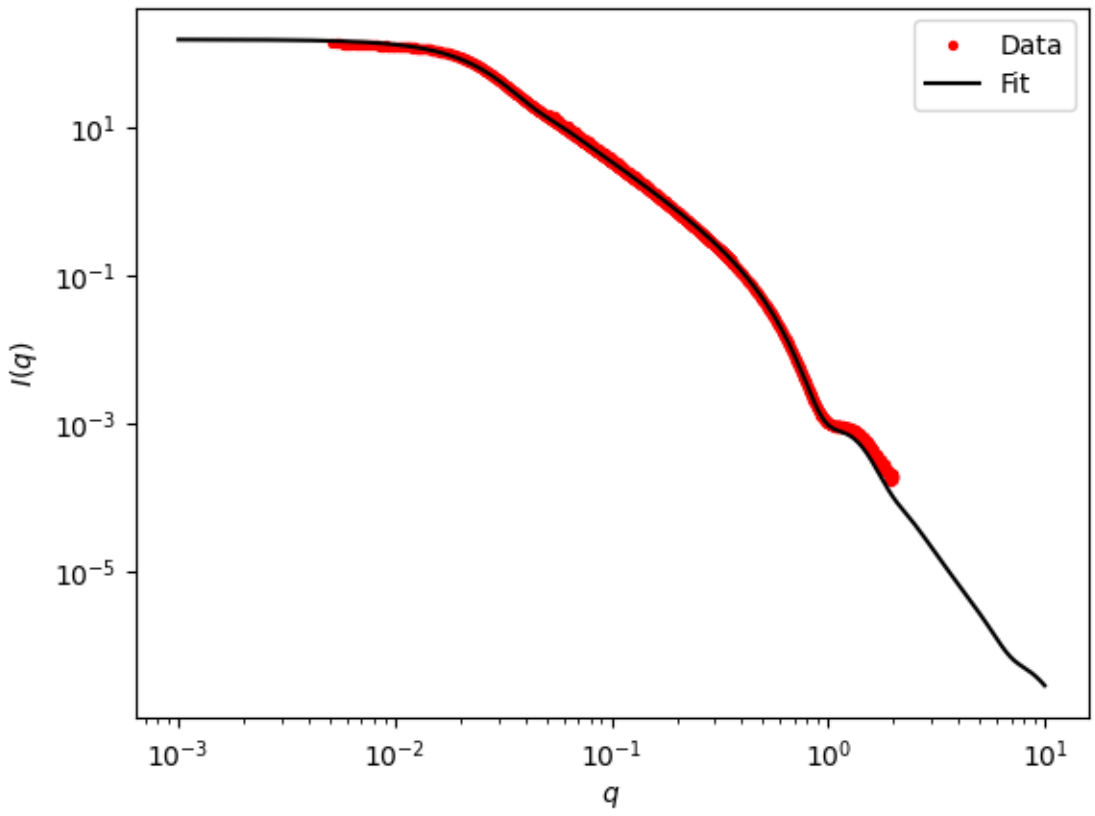
[76]:
```
<matplotlib.legend.Legend at 0x1fc8aa98eb0>
```



1.4.4 Plot the fit on top of the intensity curve

[78]:
```
rescale = 150 # fudge factor to match the calculated intensity with the␣
 ↪experimental intensity

plt.figure()
plt.scatter(input_data['q'],input_data['I'],label="Data",s=8, color = "red")
plt.loglog(qlist,rescale*theo_I,label="Fit", color = "black")
plt.ylabel("$I(q)$")
plt.xlabel("$q$")
plt.legend()
# figurepath = os.path.join(outpath, "IntensityFit.pdf")
# plt.savefig(figurepath, dpi = 300, bbox_inches='tight', transparent = True)
```

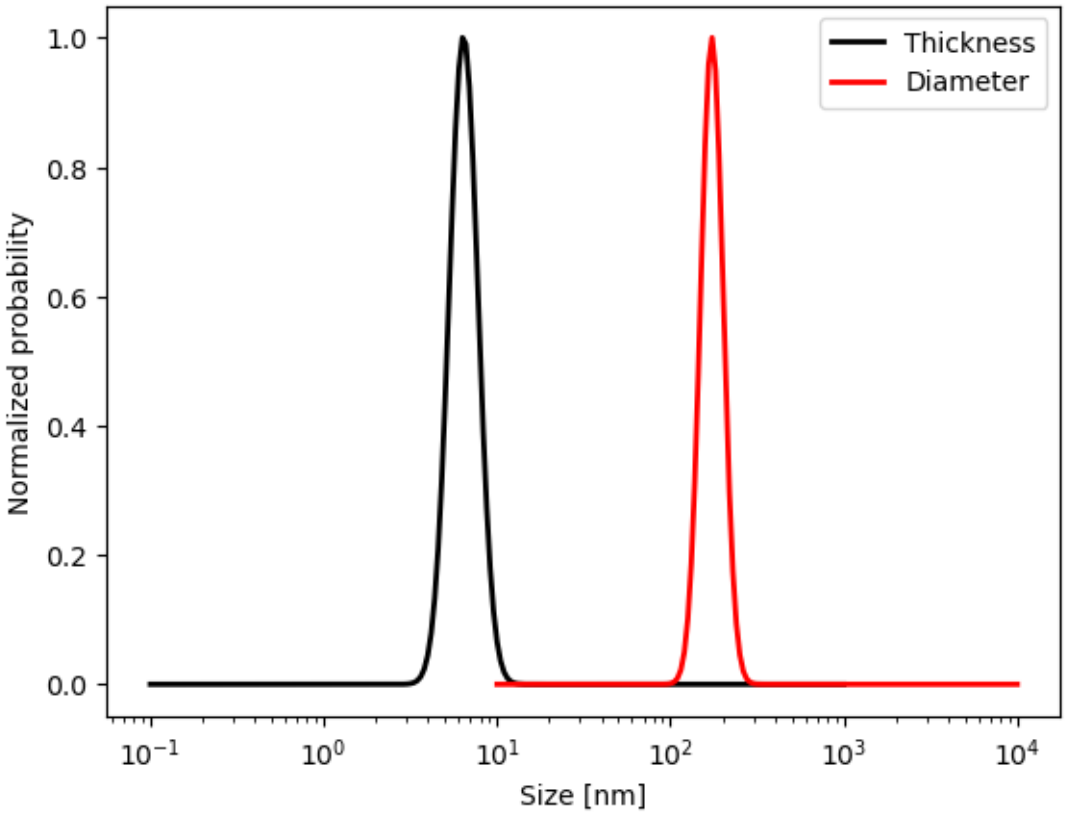
[78]:
```
<matplotlib.legend.Legend at 0x1fc8bf9d510>
```



1.4.5 Plot the size distributions

[79]:
```
pdf_r = norm.pdf(np.log(radii), loc=muD, scale=sigmaR)
pdf_l = norm.pdf(np.log(lengths), loc=muL, scale=sigmaL)

# plt.figure(figsize= [5.5*cm, 3.8*cm])
plt.figure()
ax = plt.axes()
plt.plot(lengths,pdf_l/pdf_l.max(), label="Thickness", color = "black",␣
 ↪linewidth = 2)
plt.plot(radii,pdf_r/pdf_r.max(), label="Diameter", color = "red", linewidth =␣
 ↪2)
ax.set_xscale('log')
plt.xlabel(r'Size [nm]')
plt.ylabel(r'Normalized probability')
plt.legend()
# figurepath = os.path.join(outpath, "Distributions.pdf")
# plt.savefig(figurepath, dpi = 300, bbox_inches='tight', transparent = True)
```

[79]:
```
<matplotlib.legend.Legend at 0x1fc91755a50>
```


Thickness
Diameter
Normalized probability
1.0
0.8
0.6
0.4
0.2
0.0
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}
10^{4}
Size [nm]