Versuch 223

Frauendorfer Andreas

5.5.2022

1 Einleitung

1.1 Grundlagen

1.1.1 Random-Walk

Um die Bewegung eines Teilchens in x-Richtung zu beschreiben benutzen wir das Random-Walk Prinzip. Dabei nehmen wir an, das Teilchen springt jedes Zeitintervall τ um δ nach links oder nach rechts, jeweils mit einer Wahrscheinlichkeit von 1/2. Soll nun das Teilchen in einem bestimmten Zeitraum t, in dem es $n = t/\tau$ Sprünge machtm, um m Sprünge ohne Beschränkung der Allgemeinheit m Schritte nach rechts gesprungen sein, dann müssen $(n+m)/2$ Schritte nach rechts und $(n-m)/2$ nach links stattgefunden haben. Die Wahrscheinlichkeit für jedes m zu einem n ergibt sich mit der Binomialverteilung zu:

$$P(m;n) = \frac{n!}{[\frac{1}{2}(n+m)]![\frac{1}{2}(n-m)]!}\left(\frac{1}{2}\right)^n \tag{1}$$

Mit t groß gegen τ lassen sich m! und n! mithilfe der Stirlingformel nähern.

$$P(m;n) = \sqrt{\frac{2}{\pi n}} e^{-\frac{m^2}{2n}} \tag{2}$$

Um die leicht messbaren Größen x und t zu verwenden, substituieren wir $n = t/\tau$ und $m = x/\delta$ und führen die Diffusionskonstante $D = \frac{\delta^2}{2\tau}$ ein.
Die Wahrscheinlichkeit, ein Partikel als nach der Zeit t im Bereich $[x, x + \Delta x]$ zu finden beträgt also:

$$P(x;t)\Delta x = \frac{\Delta x}{\sqrt{4\pi D t}} e^{-\frac{x^2}{4Dt}} \tag{3}$$

Dabei handelt es sich um eine zeitabhängie Gaußverteilung ohne mittlere Verrückung. Das mittlere Verschiebungsquadrat, also die Varianz ist:

$$< x^2 > = \sigma^2 = 2Dt \tag{4}$$

Zieht man daraus die Wurzel, ergibt sich der mittlere Abstand:

$$\sqrt{< x^2 >} = \sqrt{2Dt} \tag{5}$$

Die mittleren Verschiebungsquadrate lassen sich addieren, beziehungsweise die mittleren Abstände in mehreren Dimensionen lassen sich quadratisch addieren.

$$< r^2 > = < x^2 > + < y^2 > \tag{6}$$

Da die Brownsche Bewegung isotrop in allen Raumrichtungen ist, ergibt sich für 2 Raumrichtungen:

$$\sqrt{< r^2 >} = \sqrt{< x^2 > + < y^2 >} = \sqrt{2Dt + 2Dt} = \sqrt{4Dt} \tag{7}$$

Um nun die Boltzmannkostante zu berechnen benutzen wir einerseits die Definition des Diffusionskoeffizienten D mit der Boltzmannkonstante k_B der Temperatur T und dem Reibungskoeffizienten f:

$$D = \frac{k_B T}{f} \tag{8}$$

Nach dem Stoke‘schen Gesetz für kugelförmige Partikel mit Radius a ergibt sich der Reibungskoeffizient f eines Tropfens:

$$f = 6\pi\eta a \tag{9}$$

mit der Viskosität η. Setzt man nun Gleichung 9 in Gleichung 8 ein und Gleichung 8 in Gleichung 7, dann erhält man nach Umstellen die Boltzmannkonstante:

$$k_B = \frac{6\pi\eta a}{4T\bar{t}} < r^2 > \tag{10}$$

Da der Radius a, die Temperatur T und die Viskosität bekannt sind, muss nur noch das mittlere Verschiebungsquadrat berechnet werden, um die Boltzmannkonstante zu bestimmen.

Messprotokoll Versuch 223 Boltzmann I

1 Versuchsaufbau

- allgemeinbekannt für das Mikroskop

2 Verwendete Materialien

- Durchlichtmikroskop Motic B1 mit CCD-Kamera
- Kugelförmige Partikel suspendiert in Wasser
- PC
- Thermometer
- Objektmikrometer

3 Versuchsdurchführung

```
T = (23.9 +/- 0.1)°C

Partikeldurchmesser: 755 \pm 30 nm

Für das Teilchenbewegungsdiagramm siehe Code

```

TBD

2 Auswertung

Im ersten Versuchsteil wurde die gemessenne **Brown'schen Bewegungen** der SiO_2-Partikel grafisch aufgetragen in Abbildung 1. Man sieht, dass unser Partikel am Ende seines Weges sich wenig bewegt hat, was daran liegent könnte, dass es sich verhangen hat. Dies wird in der Auswertung mit berücksichtigt.

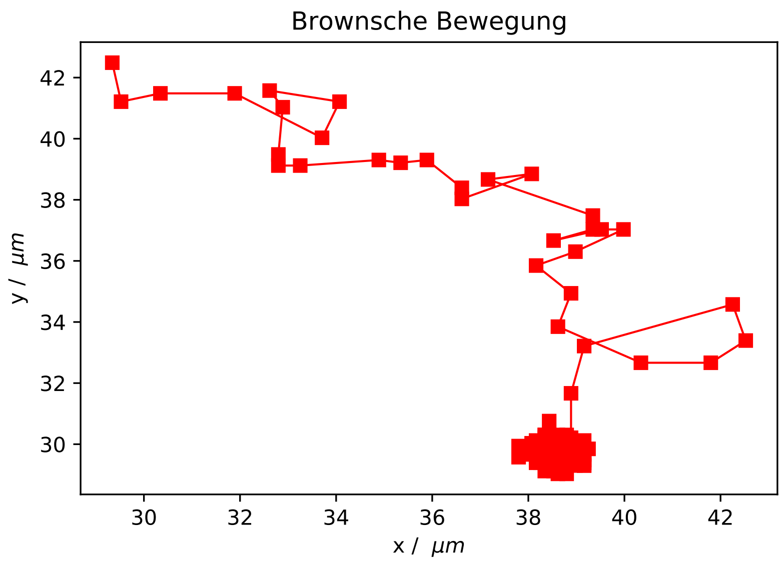


Figure 1: Brown'sche Bewegung

2.1 Berechnung des mittleren Verschiebungquadrates und dessen Fehler

Aus unseren erfassten Daten bestimmen wir mithilfe von Python die mittlere quadratische Verschiebung $< r^2 >$

$$< r^2 > = \ (0,62 \pm 0,10)\ [\mu m]^2 \tag{11}$$

und das mittlere Zeitintervall $\bar{t}$

$$\bar{t} \ = \ (1,013 \pm 0)s \tag{12}$$

Hier war der Fehler so klein, dass das Pythonobjekt float64 nicht mehr ausreichte um die nötigen Nachkommastellen abzubilden. Deshalb kann der Fehler aufgrund von Geringfügigkeit getrost vernachlässigt werden.

Aus Abbildung 12 im Skript entnehmen wir die Viskosität η von Wasser bei 21,3°C

$$\eta = (9,7 \pm 0,1) \times 10^{-4}\ Pa\ s \tag{13}$$

Der Radius a der SiO_2-Partikel beträgt

$$a = d/2 = (377,5 \pm 15)\ nm \tag{14}$$

Mit den oben gegebenen Daten lässt sich die **Boltzmannkonstante** k_B bestimmen.

$$k_B \ = \ (3,54 \pm 0,6) \times 10^{-24} \tfrac{J}{K} \tag{15}$$

Der Fehler Δk_B ergibt sich wie folgt aus der Gauß'schen Fehlerfortpflanzung

$$\Delta k_B = k_B \sqrt{(\tfrac{\Delta \eta}{\eta})^2 + (\tfrac{\Delta a}{a})^2 + (\tfrac{\Delta T}{T})^2 + (\tfrac{\Delta \bar{t}}{\bar{t}})^2 + (\tfrac{\Delta < r^2 >}{< r^2 >})^2} \tag{16}$$

Vergleicht man diesen Wert mit dem Literaturwert $k_{B,lit}$

$$k_{B,\ lit} = 1,38064852 \times 10^{-23} \tfrac{J}{K} \tag{17}$$

so erhält man für die σ-Umgebung folgenden Wert

$$\sigma_1 = \frac{|k_B - k_{B,\ lit}|}{\Delta k_B} \approx 17,69\sigma \tag{18}$$

Diese hoche Abweichung muss zum Einen daran liegen, dass wir die Fehler zu klein abgeschätzt haben und zum Anderen daran, dass sich unser verfolgtes Partikelchen anscheinend mit einem anderen Teilchen verbunden hat und somit andere Effekte neben der brownschen Bewegung eine Rolle spielen, die hier jedoch nicht beachtet werden.

Des Weiteren kann man nun auch die Diffusionskonstante D nach Gleichung (8) bestimmen.

$$D = (1,53 \pm 0,26) \times 10^{-13}\ \frac{m^2}{s} \tag{19}$$

Der Fehler ΔD berechnet sich mithilfe der Gauß'schen Fehlerfortpflanzung

$$\Delta D = D\sqrt{(\frac{\Delta\eta}{\eta})^2 + (\frac{\Delta a}{a})^2 + (\frac{\Delta T}{T})^2 + (\frac{\Delta k_B}{k_B})^2} \tag{20}$$

2.2 Kontrollverteilung

Da die **Brown'sche Bewegung** isotrop ist, können wir die Verschiebungen in x- und y-Richtung in ein gemeinsames Histogramm (Abbildung 2) eintragen

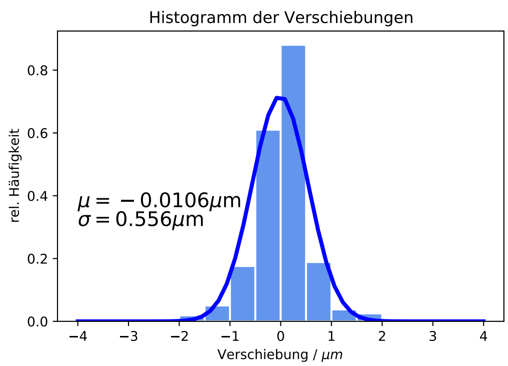


Figure 2: Verschiebungshistogramm

Aus dem obigen Histogramm lassen sich der Mittelwert μ und die Standardabweichung σ bestimmen.

$$\mu = 0,0106\ \mu m \tag{21}$$

$$\sigma = 0,556\ \mu m \tag{22}$$

2.3 Kumulative Verteilung der Verschiebungsquadrate

Nach der **Einstein-Smoluchowski-Gleichung** (5) ist das Verschiebungquadrat $< r^2 >$ proportional zur Zeit t, daher sollte sich bei einer kumulativen Darstellung der Verschiebung ein linearer Zusammenhang ergeben. Dieser ist in Abbildung 3 zu erkennen.

Die Steigung s des "linear fit" beträgt

$$s = (1,677 \pm 0,05)\ \frac{\mu m^2}{s} \tag{23}$$

Hierbei ist zu beachten, dass aufgrund der Tatsache, dass unser Partikel an etwas festgeklebt ist, nur der erste Abschnitt der Messung für die kumulative Verteilung aussagekräftig und nur aus diesem Teil wurde die Steigung extrahiert.

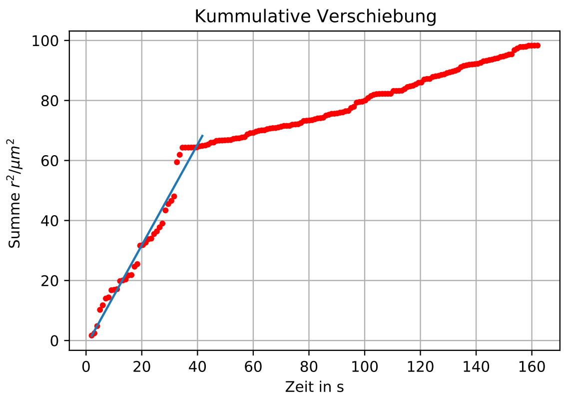


Figure 3: Kumulative Verschiebung

Mit diesem alternativen Wert für $< r^2 >$ lässt sich die Boltzmannkonstante k_B analog zu (15) und (16) berechnen:

$$k_B^* = (0,96 \pm 0,05) \ \times 10^{-23} \frac{J}{K} \tag{24}$$

Vergleicht man diesen Wert mit dem Literaturwert $k_{B,lit}$, so erhält man für die σ-Umgebung folgenden Wert

$$\sigma_2 = \frac{|k_B^* - k_{B,\ lit}|}{\Delta k_B^*} \approx 8,4\sigma \tag{25}$$

Im Vergleich zu dem anderen experimentellen Wert der **Boltzmannkonstante** k_B ergibt sich eine Abweichung von

$$\sigma_3 = \frac{|k_B^* - k_B|}{\sqrt{(\Delta k_B^*)^2 + (\Delta k_B)^2}} \approx 7,8\sigma \tag{26}$$

Auch die Diffusionskonstante D lässt sich erneut berechnen:

$$D^* = (4,14 \pm 0,26) \times 10^{-13} \ \frac{m^2}{s} \tag{27}$$

Vergleicht man die Diffusionskonstanten D und D* so erhält man eine Abweichung σ_D von

$$\sigma_D = \frac{|D^* - D|}{\sqrt{(\Delta D^*)^2 + (\Delta D)^2}} \approx 7,04\sigma \tag{28}$$

3 Diskussion

In diesem Versuch wurde die Brown'sche Bewegung von suspendierten SiO_2-Partikeln in Wasser beobachtet und gemessen. Aus den erhobenen Daten bestimmten wir die Boltzmannkonstante k_B auf zwei unterschiedliche Arten.

Unsere erste Methode ergab den Wert $k_B = (3,54 \pm 0,6) \times 10^{-24} \frac{J}{K}$ für die Boltzmannkonstante, welcher im 18 σ-Bereich zum Literaturwert von $k_{B,\ lit} = 1,38064852 \times 10^{-23} \frac{J}{K}$ liegt. Diese Abweichung ist sehr groß und liegt wahrscheinlich an unseren mäßig guten Messdaten. Andere Fehlerquellen sind hier Ströme die durch das Licht des Mikroskops ausgelöst werden oder sonstige Diffusionsströme in der Flüssigkeit welche durch bspw. eine schlechte Isolation durch das Klebeband oder Verschmutzungen in der Lösung, bzw. dem Präparat ausgelöst werden können.
Unser alternativer Wert für die Boltzmannkonstante $k_B^* = (0,96 \pm 0,05) \times 10^{-23} \frac{J}{K}$ liegt in einer 8,4σ-Umgebung zum Literaturwert und weicht damit ebenfalls signifikant ab, jedoch weniger als unser erstbestimmter Wert. Hier konnte schlimmeres dadurch verhindert werden, dass nur der Teil der Steigung verwendet wurde, als das Teilchen noch frei war. Die beiden Diffusionskonstanten $D = (4,49 \pm 0,52) \times 10^{-13}\ \frac{m^2}{s}$ und $D^* = (4,17 \pm 0,24) \times 10^{-13}\ \frac{m^2}{s}$ weichen auch signifikant voneinander ab (7,04σ), wobei hier auch die als zweites Ermittelte Diffusionskonstante wahrscheinlich eine kleiner Abweichung vom wahren Wert zeigt als die als erstes Emittelte.

Zusammenfassend hätten es Sinn gemacht, die Messung zu wiederholen, allerdings war er auf dem Bildschirm nicht zu erkennen, ob das Teilchen festhängt oder sich nur in z-Richtung bewegt, weshalb und uns ist der Fehler erst beim Ausarbeiten aufgefallen ist. Außerdem hätten wir aufgrund einiger Schwierigkeiten bei der Probenpräparation kein besseres Resultat, wenn überhaupt eines erzielen können.
Da wir diesen Fehler als einordnen können und entsprechend versucht haben die relevante Information herauszuholen aus den Daten, können wir diesen Versuch insgesamt als gute Lektion betrachten.

Python Code Versuch 223

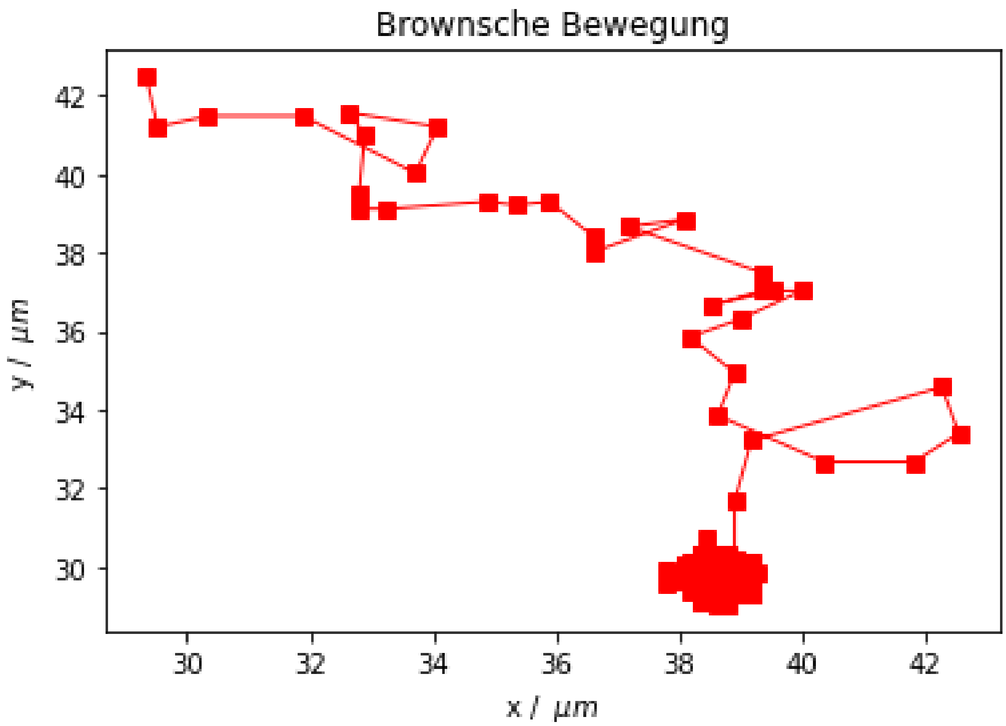
May 5, 2022

[49]:
```
%matplotlib inline



import matplotlib.pyplot as plt
import matplotlib.mlab as mlab
import numpy as np
from scipy.optimize import curve_fit
```

[50]:
```
def comma_to_float(valstr):
    return float(valstr.decode("utf-8").replace(',' , '.'))
```

[51]:
```
t,x,y = np.loadtxt('Messung.dat', skiprows = 1, usecols = (1,2,3), converters =
 ↪{1:comma_to_float, 2 : comma_to_float, 3:comma_to_float}, unpack = True)
#skippe erste spalte und convertiere kommmas in punkte
```

[52]:
```
# zeichne x,y diagramm als punkt-liniendiagramm

plt.plot(x,y, marker = 's', color = 'red', linewidth = 1)
plt.title("Brownsche Bewegung")
plt.xlabel('x / '+' $\mu '+' m$')
plt.ylabel('y / '+' $\mu '+' m$')
plt.savefig ('brown1.pdf', format = 'PDF')
```



[53]:
```
#berechnung mittlere Verschiebungsquadrat und Fehler
#warum muss ich leere array vorher erzeugen? wegen append!
dt = np.array([])
dx = np.array([])
dy = np.array([])

i = 0
while i < len(t)-1:
    dt = np.append(dt, t[i+1]-t[i])
    dx=np.append(dx,x[i+1]-x[i])
    dy=np.append(dy,y[i+1]-y[i])
    i = i + 1
r_squared=dx**2+dy**2  #das ist mitt versch quad, array!

#Fehler
r_squared_mean = np.mean(r_squared)
print("r_squared_mean= " ,r_squared_mean)

r_squared_mean_std=np.std(r_squared)/np.sqrt(len(r_squared))  #man muss noch␣
↪durch die wurzeln teilen, da Abweichung des Mittelwerts vom Wahren wert!!
print("r_squared_mean_std= " ,r_squared_mean_std)
dt_mean=np.mean(dt)
print("dt_mean= ", dt_mean)
```

```
dt_mean_std = np.std(dt_mean)/np.sqrt(len(dt))
print("dt_mean_std=",dt_mean_std)
```

```
r_squared_mean=  0.618279490566038
r_squared_mean_std=  0.09794195971751934
dt_mean=  1.0130817610062892
dt_mean_std= 0.0
```

1 Berechnung Boltzmann kosntante

[54]:
```
eta = 9.7*10**(-4)
d_eta = 0.1 *10**(-4)

a = 377.5 *10**(-9)
d_a = 15 *10**(-9)

T = 297.05 #in K
d_T = 0.1



k = (6*np.pi* eta * a*r_squared_mean*10**(-12))/(4*T*dt_mean)   #*10**(-12) da␣
↪r_squared mean in micrometer quadrat!
d_k = k*np.sqrt((d_eta/eta)**2 + (d_a/a)**2 + (d_T/T)**2 + (r_squared_mean_std/
↪r_squared_mean)**2)

#Literaturwert Boltzmannkonstante
k_b = 1.38064852 * 10**(-23)


print(k, d_k)
```

```
3.5451989260286594e-24 5.801483518792721e-25
```

[55]:
```
#Diffusionskoeffizient D

D = (k*T)/(6*np.pi*eta*a)
d_D = D*np.sqrt((d_eta/eta)**2 + (d_a/a)**2 + (d_T/T)**2 + (d_k/k)**2)
print(D, d_D)
```

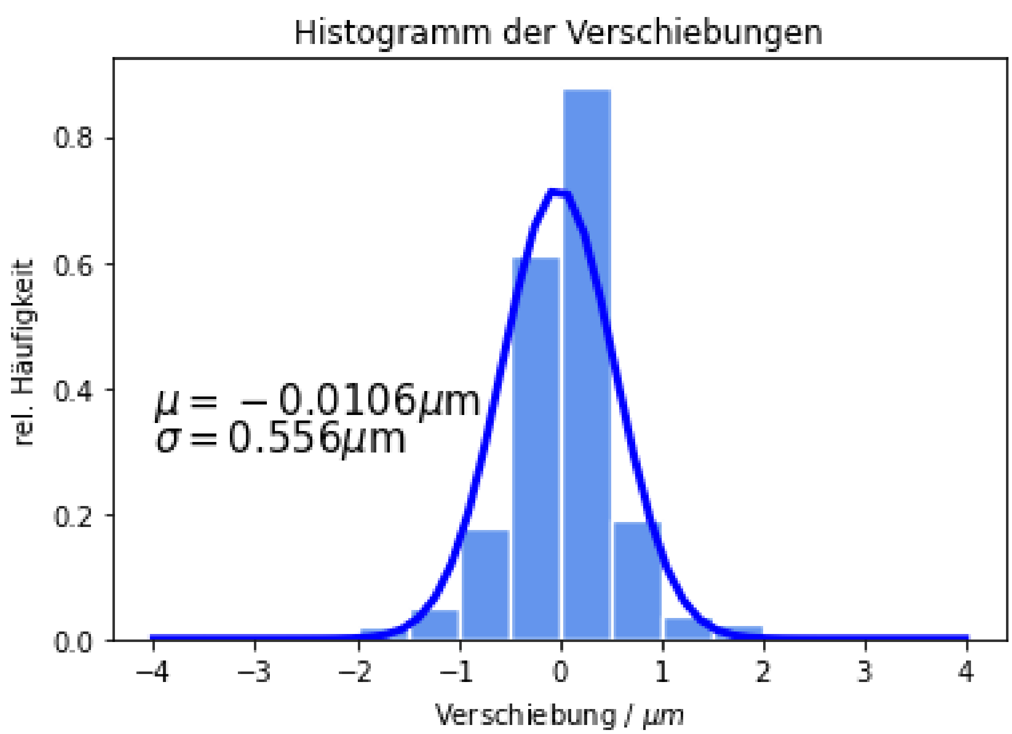
```
1.5257393686367029e-13 2.574136652991754e-14
```

[56]:
```
#Gut dann gibts halt keine Gaußkurvendrüber
```

[57]:
```
print(gauss)
```

```
<function gauss at 0x00000235A4BE8B80>
```

[60]:
```
all_data=np.append(dx,dy)
plot3 = plt.xlabel('Verschiebung / $\mu m$')
plot3 = plt.ylabel('rel. Häufigkeit')
plot3 = plt.hist(all_data, density=1, bins = np.arange(-4,4,0.5),
color='cornflowerblue', edgecolor='white',lw = 1.4)
plot3 = plt.title("Histogramm der Verschiebungen")
mu = np.mean(all_data)
sigma = np.std(all_data)

def gauss(x, mu, sigma): #A: Flaeche der Gaussfunktion
    return 1/(np.sqrt(2*np.pi)*sigma)*np.exp(-(x-mu)**2/2/sigma**2)


#gauss = mlab.normpdf(np.linspace(-4,4), mu , sigma) #tja das hat leider nicht␣
↪funktioniert, ist das etwas systematisches?

x = np.linspace(-4,4)

plot3 = plt.plot(x, gauss(x,mu, sigma), 'b-', linewidth = 3)

plt.text(-4, 0.360, f'$\mu = {mu:.03} \mu$m', size = 15)

plt.text(-4, 0.300, f'$\sigma = {sigma:.03} \mu$m' , size = 15)

plt.savefig('brown2.pdf')
```



Kontrollverteilung

[ ]:

```

```

[ ]:

```

```

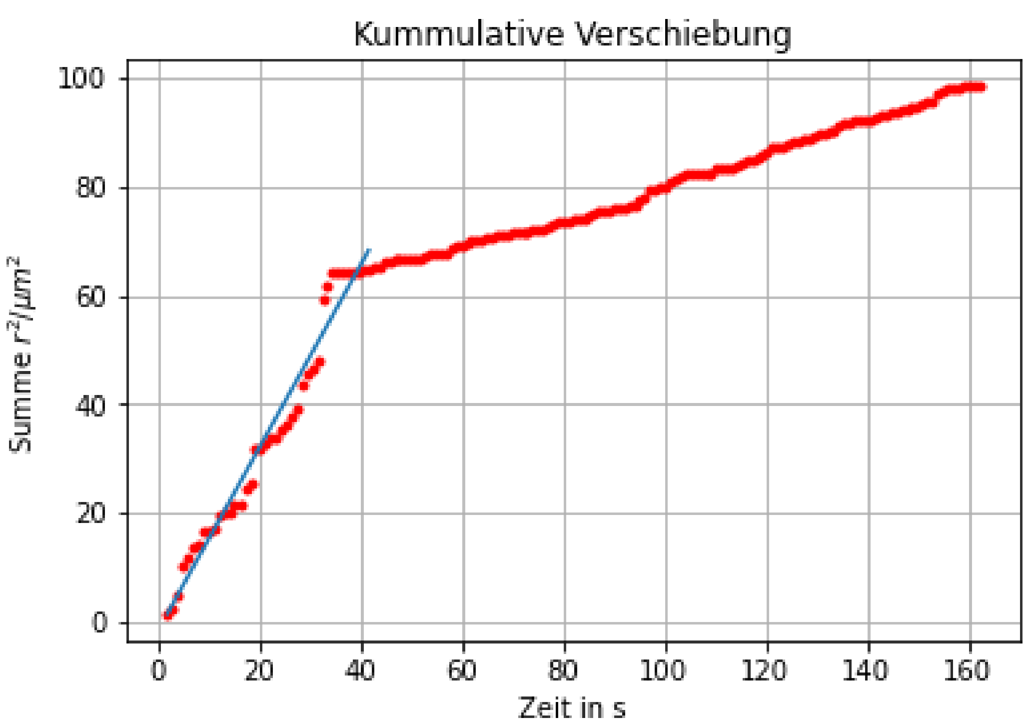
[62]:

```
# KUMULATIV§ Verschieungsquadrate

r_kumm=np.cumsum(r_squared)
plt.plot(t[:-1], r_kumm, marker='.', color='red', linewidth=0)
plt.xlabel('Zeit / s')
plt.ylabel('Summe $r_i^2 /\mu m^2$') #anführungszeichen wahrsclh nicht notwenig
plt.title('Kummulative Verschiebung')
#plt.savefig('brown3.pdf', format='PDF')

#gerade fitten
from scipy.optimize import curve_fit

def linear (x,a,b):
    return a*x+b
popt, pcov = curve_fit(linear, t[:40], r_kumm[:40])    #hier hab ich die ersten␣
↪23 t werte raus genommen, da hier der batzen anfängt!

plt.plot(t[:40], linear(t[:40],*popt)) #fügt funktion in diagramm ein!
plt.text(5, 255, f'm = {popt[0]:.4} $\mu m^2/s$', size = 15)
plt.text(5, 230, f'b = {popt[1]:.4} $\mu m^2$', size = 15)
plt.text(5, 280, f'Fit-Funktion: y = a*x+b', size = 15)
plt.title("Kummulative Verschiebung")
plt.xlabel("Zeit in s")
plt.ylabel("Summe $r^2 / \mu m^2$ ")
plt.grid(True)



plt.savefig('brown3.pdf')
```

Fit-Funktion: y = a*x+b

m = 1.677 $\mu m^2/s$

b = -1.795 μm^2



[63]:
```
#berechne wieder Boltzmann, Diffusionskonstante siehe oben     Diesmal mit␣
↪steigung m!
# finden man unten in skirpt
```

[64]:
```
m = popt[0]
d_m = np.sqrt(np.diag(pcov[0,[0]]))
d_m = np.sqrt(pcov[0,[0]])


print(a)
print(m)
print(d_m)
```

```
3.7750000000000004e-07
1.6773290384991242
[0.04786401]
```

[65]:
```
km = (6*np.pi* eta * a* m*10**(-12))/(4*T*dt_mean)
d_km = km*np.sqrt((d_eta/eta)**2 + (d_a/a)**2 + (d_T/T)**2 + (d_m/m)**2)

print(km, d_km)
```

```
9.617762187841232e-24 [4.80846516e-25]
```

[66]:
```
#berechne Diffusionskonsante

Dm = (km*T)/(6*np.pi*eta*a)
d_Dm = Dm *np.sqrt((d_eta/eta)**2 + (d_a/a)**2 + (d_T/T)**2 + (d_km/km)**2)

print(Dm, d_Dm)
```

```
4.139174899449975e-13 [2.67764636e-14]
```

[ ]:

[ ]:

[ ]: