# PAP 2.1
# Versuch 223: "Brownsche Bewegung"

Jonas Schweer

9. März 2025

## Inhaltsverzeichnis

**1 Einleitung** **2**
- 1.1 Ziel des Versuchs . . . 2
- 1.2 Versuchsaufbau . . . 2
- 1.3 Physikalische Grundlagen . . . 3
  - 1.3.1 Random-Walk Modell . . . 3
  - 1.3.2 Einstein-Smoluchowski-Gleichung . . . 4
  - 1.3.3 Bestimmung der Boltzmannkonstante . . . 4

**2 Messprotokoll** **4**

**3 Auswertung** **6**
- 3.1 Graphische Darstellung . . . 6
- 3.2 Berechnung mit dem mittleren Verschiebungsquadrat . . . 7
- 3.3 Berechnung mit der kumulativen Verschiebung . . . 8

**4 Zusammenfassung und Diskussion** **9**

**5 Anhang** **11**

# 1 Einleitung

**Anmerkung:** Alle Bilder, welche in diesem Abschnitt verwendet wurden, sind aus dem Praktikumsskript übernommen worden.
**Quelle:** 'Physikalisches Praktikum PAP2.2 für Studierende der Physik B.Sc' - Jens Wagner, Februar 2025

## 1.1 Ziel des Versuchs

In diesem Versuch wollen wir uns mit dem Phänomen der Brownschen Bewegung beschäftigen, indem wir uns die Bewegung eines einzelnen Teilchens anschauen. Insbesondere wollen wir diese Bewegung nutzen, um die Boltzmannkonstante zu bestimmen.

## 1.2 Versuchsaufbau

Der Aufbau besteht aus einem Durchlichtmikroskop, dessen Bild durch eine Kamera an einen beistehenden PC gegeben wir. Mit einem Programm können wir nun automatisiert Bilder aufnehmen und anschließend auch die Position des beobachteten Teilchens verfolgen. Mit einem Objektmikrometer können wir die Skalierung des Bildes kalibrieren.

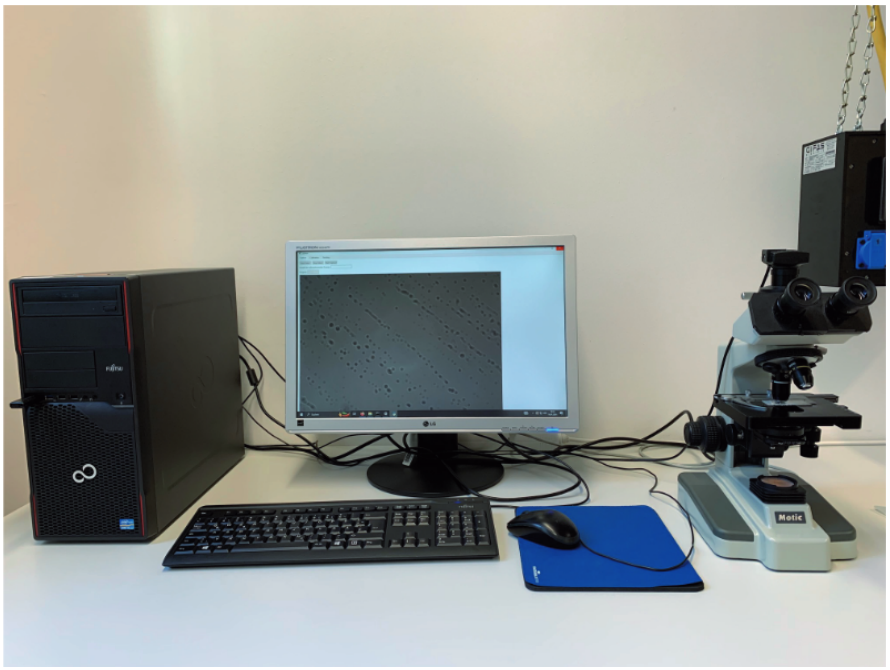


Abbildung 1: Versuchsaufbau

Das Präparat, welches wir unter das Mikroskop legen, bauen wir vor Ort selbst nach folgendem Schema:

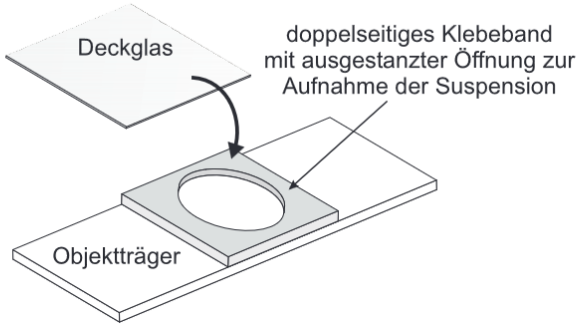



Abbildung 2: Das Präparat

## 1.3 Physikalische Grundlagen

### 1.3.1 Random-Walk Modell

Um uns vereinfacht die Statistik der Brownschen Bewegung herzuleiten, schauen wir uns den eindimensionalen 'Random-Walk' an:

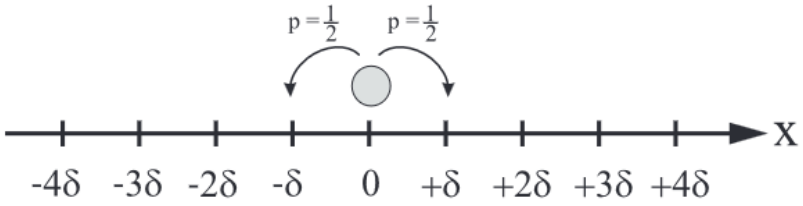



Abbildung 3: Darstellung des 1-D Random Walk

Wir nehmen an, dass das Teilchen alle $\tau$ Zeiteinheiten um Distanz $\delta$ entweder nach links oder nach rechts springt. Beide Richtungen haben Wahrscheinlichkeit $p = 0.5$, bei $t = 0$ startet das Teilchen bei $x = 0$. Bei Teilchensystemen gehen wir davon aus, dass das Teilchen bei jedem Stoß unabhängig von den anderen Teilchen ist. Wir wissen also, dass die Anzahl der Stöße durch $n = t/\tau$ gegeben ist, und das Teilchen sich an einer Position $x = m\delta$ mit $m \in \mathbb{Z}, |m| \leq n$. Die Wahrscheinlichkeit $P(m; n)$, dass das Teilchen nach $n$ Stößen an der Stelle $x = \delta m$ ist binomialverteilt und gegeben als:

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

Wird $n$ nun sehr groß ($\tau$ wird sehr klein), so kann man die Stirling-Formel als Näherung nutzen:

$$P(n;m) = \sqrt{\frac{2}{\pi n}} \exp\left[-\frac{m^2}{2n}\right] \tag{2}$$

### 1.3.2 Einstein-Smoluchowski-Gleichung

Nun wollen wir in Formel (2) die Größen $m$ und $n$ mit den messbaren Größen $x$ und $t$ ersetzen. Dabei führen wir direkt den Diffusionskoeffizienten $D$ ein. Für diesen gilt:

$$D = \frac{\delta^2}{2r} \tag{3}$$

Die Gleichung wir dann zu:

$$P(x;t)\Delta x = \frac{\Delta x}{\sqrt{4\pi Dt}} \exp\left[-\frac{x^2}{4Dt}\right] \tag{4}$$

Diese Gaußverteilung ist die Wahrscheinlichkeit, dass Teilchen im Bereich $[x, x + \Delta x]$ zu finden. Daraus folgt als Einstein-Smoluchowski-Gleichung:

$$\sqrt{\langle x^2 \rangle} = \sqrt{2Dt} \tag{5}$$

oder in 2 Dimensionen (so wie wir es in unserem Versuch brauchen):

$$\sqrt{\langle x^2 \rangle} = \sqrt{4Dt} \tag{6}$$

### 1.3.3 Bestimmung der Boltzmannkonstante

Nach Einstein ist der Diffusionskoeffizient gegeben als:

$$D = \frac{kT}{f} \tag{7}$$

Dabei ist $k$ die Boltzmannkonstante, $T$ die Temperatur und $f$ der Reibungskoeffizient. Mit dem Stokesschen Gesetz folgt für $f$: $f = 6\pi\eta a$ mit $\eta$ als Viskosität der Flüssigkeit und $a$ als Radius der kugelförmigen Partikel. Für den Diffusionskoeffizienten folgt also:

$$D = \frac{kT}{6\pi\eta a} \tag{8}$$

In Formel (6) eingesetzt folgt also:

$$\langle r^2 \rangle = \frac{4kT}{6\pi\eta a} t \Leftrightarrow k = \frac{6\pi\eta a}{4Tt} \langle r^2 \rangle \tag{9}$$

# 2 Messprotokoll

# Messprotokoll 223: Brownsche Bewegung

17.02.25

Harish Mahendran
Jonas Schwer

**Messaufbau:**

- Durchlichtmikroskop Motic B1 mit CCD-Kamera
- Kugelförmige Partikel suspendiert in Wasser
- PC mit Messprogramm, Jupyternotebook und Excel
- Thermometer
- Objektmikrometer

## Aufgabe 1: Herstellung der Probe

Wir stellen die Probe nach Anleitung her. Wir legen sie in das Mikroskop und stellen dessen Fokus so ein, dass wir die Partikel beobachten können. Außerdem notieren wir die Daten zum Durchmesser der Partikel auf der Flasche:

$$d = (755 \pm 30)\,\text{nm}$$

## Aufgabe 2: Aufnehmen der Bildreihe

Wir starten das Programm und nehmen eine Bildfolge von 150 Bildern auf. Nachdem das Programm die Bilder geschossen hat, notieren wir die Raumtemperatur.

$$T = (21{,}90 \pm 0{,}10)\,^\circ\text{C} = (295{,}05 \pm 0{,}10)\,\text{K}$$

## Aufgabe 3: Eichung des Ausschnittes

Mit Hilfe eines Mikrometermaßstabs wollen wir nun das Bild eichen. Dafür nutzen wir den "Calibration"-Tab innerhalb des Programms. Unsere Eichung lautet:

$$E = 0{,}0901\,\mu\text{m/pixel}$$

## Aufgabe 4: Vermessung der Partikelpositionen

Mit dem Tracking-Tab des Programms können wir nun anfangen, auf jedem Bild die Position des Partikels zu markieren. Sind wir fertig damit, speichern wir alle wichtigen Messergebnisse in einen Ordner zur Auswertung ab.

# 3 Auswertung

## 3.1 Graphische Darstellung

Zunächst wollen wir unsere gemessenen Daten graphisch darstellen. Dafür plotten wir zunächst unsere gemessenen x- und y-Koordinaten, um die 2-Dimensionale Bewegung unseres beobachteten Teilchens zu visualisieren:

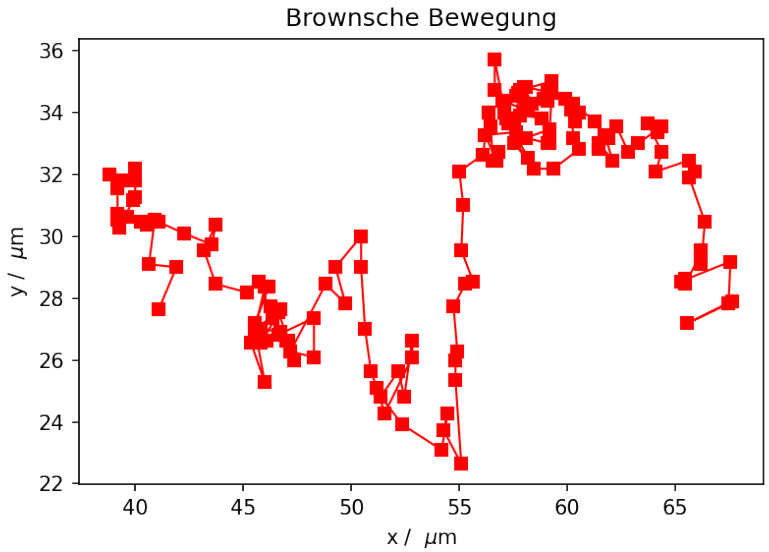



Abbildung 4: Zweidimensionale Bewegung unseres Teilchens

Des weiteren berechnen wir, wie weit sich das Teilchen pro Zeitschritt bewegt hat, und plotten die Häufigkeiten in ein Histogramm. Da diese Verteilung im Grenzfall zu einer Gaußverteilung werden sollte, nutzen wir die curve-fit Funktion in Python, um eine solche Kurve anzugleichen. Die Verteilung sieht dann letztendlich so aus:

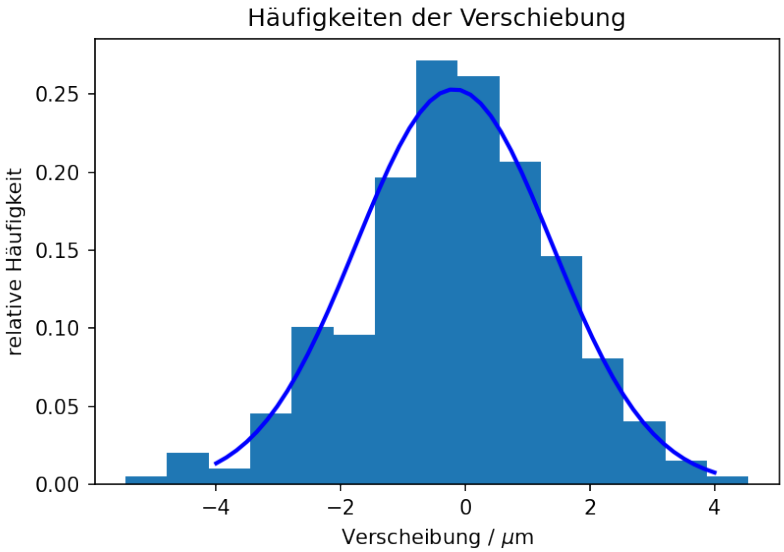



Abbildung 5: Histogramm der Verschiebungen mit Gaussfit

Die Parameter des Fits gibt uns die Funktion aus als:

$$\boxed{\mu = -0.18\mu\text{m}\,,\ \sigma = 1.58\mu\text{m}}$$

## 3.2 Berechnung mit dem mittleren Verschiebungsquadrat

Nachdem wir unsere Daten in Python geladen haben, berechnen wir automatisch das mittlere Verschiebungsquadrat $\langle r^2 \rangle$, sowie dessen Fehler. Der Wert lautet:

$$\boxed{\langle r^2 \rangle = (2.52 \pm 0.22)\mu\text{m}^2}$$

Da die Zeitkonstante laut Praktikumsanleitung $t = 1\text{s}$ mit vernachlässigbarem Fehler ist, genügt es nach Formel (6) bereits, die quadratische Abweichung mit ihrem zugehörigen Fehler zu vierteln, um den Diffusionskoeffizienten zu berechnen. Dieser ist also:

$$\boxed{D = (0.62 \pm 0.05)\frac{\mu\text{m}^2}{\text{s}}}$$

Um nun daraus die Boltzmannkonstante zu berechnen, nutzen wir die im Messprotokoll notierte Temperatur, sowie die Viskosität der Flüssigkeit, welche wir aus dem nachfolgenden Diagramm ablesen können:

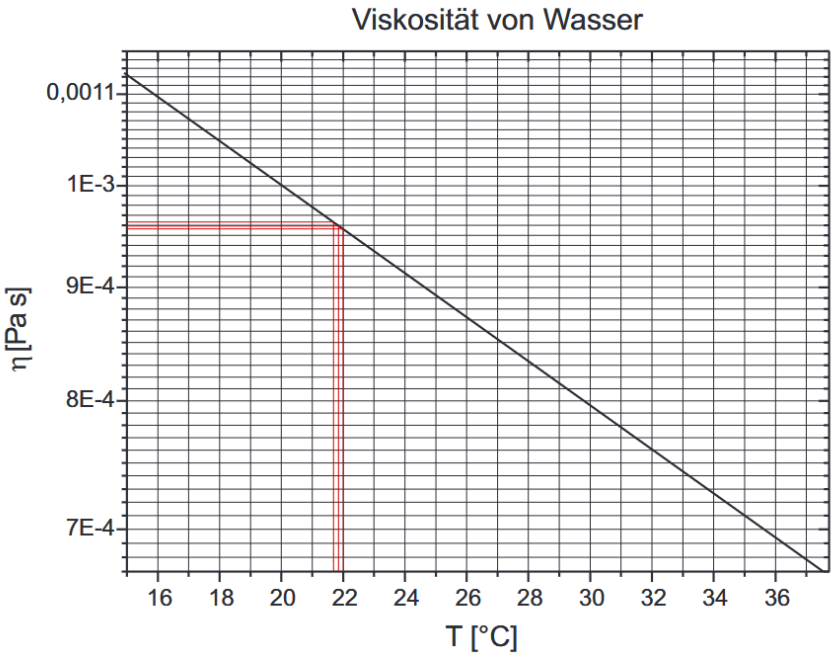



Abbildung 6: Viskosität des Wassers bei eingezeichneter Temperatur,
Quelle: 'Physikalisches Praktikum PAP2.2 für Studierende der Physik B.Sc' - Jens Wagner, Februar 2025

Also ist der Wert der Viskosität ungefähr:

$$\boxed{\eta = (9.60 \pm 0.03)\text{Pa s}}$$

Mit Formel (9) und einem Fehler nach Gauss von:

$$\Delta k = k\sqrt{\left(\frac{\Delta\eta}{\eta}\right)^2 + \left(\frac{\Delta T}{T}\right)^2 + \left(\frac{\Delta\langle r^2\rangle}{\langle r^2\rangle}\right)^2 + \left(\frac{\Delta a}{a}\right)^2} \tag{10}$$

ist die berechnete Boltzmannkonstante:

$$\boxed{k = (1.45 \pm 0.14) \cdot 10^{-23}\frac{\text{J}}{\text{K}}}$$

## 3.3 Berechnung mit der kumulativen Verschiebung

Nun wollen wir die kumulative Verschiebung gegen die Zeit plotten. Das dafür verwendete Python-Skript ist im Anhang wiederzufinden. Wir legen mit der curve-fit Funktion eine Ausgleichsgerade durch die Verschiebungen. Die Steigung $s$ dieser entspricht dann in Formel (9) dem Bruch $s = \frac{\langle r^2\rangle}{t}$.

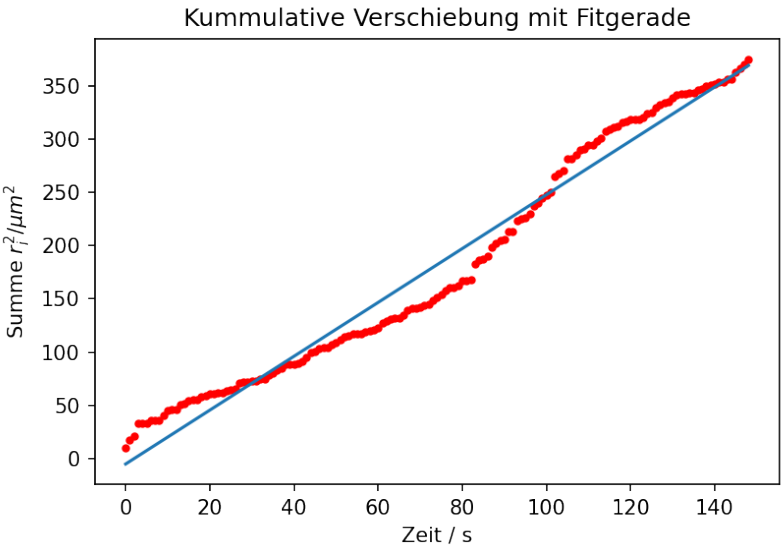



Abbildung 7: Kumulative Verschiebung mit Fitgerade

Die Steigung ist gegeben als:

$$\boxed{s = (2.53 \pm 0.03)\frac{\mu\mathrm{m}^2}{\mathrm{s}}}$$

Durch vierteln dieses Wertes kommen wir nun nach Formel (6) wieder auf den Diffusionskoeffizienten. Dieser lautet dann:

$$\boxed{D = (0.632 \pm 0.008)\frac{\mu\mathrm{m}^2}{\mathrm{s}}}$$

Setzen wir nun die Steigung in die Formel (9) ein, und nutzen den Fehler nach Gauss:

$$\Delta k = k\sqrt{\left(\frac{\Delta\eta}{\eta}\right)^2 + \left(\frac{\Delta a}{a}\right)^2 + \left(\frac{\Delta T}{T}\right)^2 + \left(\frac{\Delta s}{s}\right)^2} \tag{11}$$

ergibt unsere zweite Berechnung der Boltzmannkonstante:

$$\boxed{k = (1.46 \pm 0.06) \cdot 10^{-23}\frac{\mathrm{J}}{\mathrm{K}}}$$

# 4 Zusammenfassung und Diskussion

In diesem Versuch haben wir die Boltzmannkonstante durch Positionsmessung eines kugelförmigen Partikels bestimmt. Das haben wir einmal über die mittlere Änderung in einem Zeitintervall und einmal über die kumulative Änderung gemacht. Wir erhielten

so zwei Werte für den Diffusionskoeffizienten und zwei Werte für die Boltzmannkonstante. Diese lauteten:

$$\boxed{D_1 = (0.62 \pm 0.05)\frac{\mu\mathrm{m}^2}{\mathrm{s}}}$$
$$\boxed{D_2 = (0.632 \pm 0.008)\frac{\mu\mathrm{m}^2}{\mathrm{s}}}$$
$$\boxed{k_1 = (1.45 \pm 0.14) \cdot 10^{-23}\frac{\mathrm{J}}{\mathrm{K}}}$$
$$\boxed{k_2 = (1.46 \pm 0.06) \cdot 10^{-23}\frac{\mathrm{J}}{\mathrm{K}}}$$

Berechnen wir die Abweichung der Diffusionskoeffizienten voneinander unter Verwendung folgender Formel:

$$\sigma = \frac{|x - y|}{\sqrt{\Delta x^2 + \Delta y^2}} \tag{12}$$

wobei $x$ und $y$ unsere Messwerte sind, so erhalten wir:

$$\boxed{\sigma_D = 0.05}$$

Diese beiden Werte sind also sehr nah beieinander. Es ist also davon auszugehen, dass wir den Diffusionskoeffizienten nahezu richtig bestimmt haben und keine großen systematischen Fehler geschehen sind. Vergleichen wir nun die beiden Werte für die Boltzmannkonstante untereinander, sowie mit dem Literaturwert von $k = 1.380649 \cdot 10^{-23}\frac{\mathrm{J}}{\mathrm{K}}$ (Quelle: Wikipedia), so sind die Abweichungen:

$$\boxed{\sigma_{k_1,k_2} = 0.04}$$
$$\boxed{\sigma_{k_1,\mathrm{lit}} = 0.56}$$
$$\boxed{\sigma_{k_2,\mathrm{lit}} = 1.34}$$

Auch hier sind die Abweichungen wieder sehr gering. Es ist also auch hier zu vermuten, dass keine größeren systematischen Fehler aufgetreten sind. Vor allem die Abweichungen der beiden berechneten Werte ist unfassbar klein. Das ist allerdings auch zu erwarten, schließlich sind beide Messungen basierend auf den selben Distanzmessungen. Beide sind jedoch etwas zu hoch ausgefallen. Eine mögliche Ursache dafür ist eine sehr leichte Linksströmung, welche unsere Abstandsmessungen etwas erhöhen würde. Das würde auch erklären, weshalb unser Teilchen eher nach links gewandert ist, obwohl es ja eigentlich statistisch gesehen ungefähr an der selben Stelle hätte bleiben müssen. Allerdings scheint diese Strömung keinen großen unterschied gemacht zu haben, da wir trotzdem noch sehr gut in dem statistisch insignifikanten Bereich liegen.
Bei einer erneuten Durchführung des Versuches würde ich vor allem extra vorsichtig beim Präparieren der Probeflüssigkeit sein, und gegebenenfalls auch mehrere Teilchen gleichzeitig verfolgen, um so Faktoren wie eine möglicherweise präsente Strömung besser festzustellen und analysieren zu können.

# 5 Anhang

# Versuch 223

March 9, 2025

Versuch 223: Brownsche Bewegung

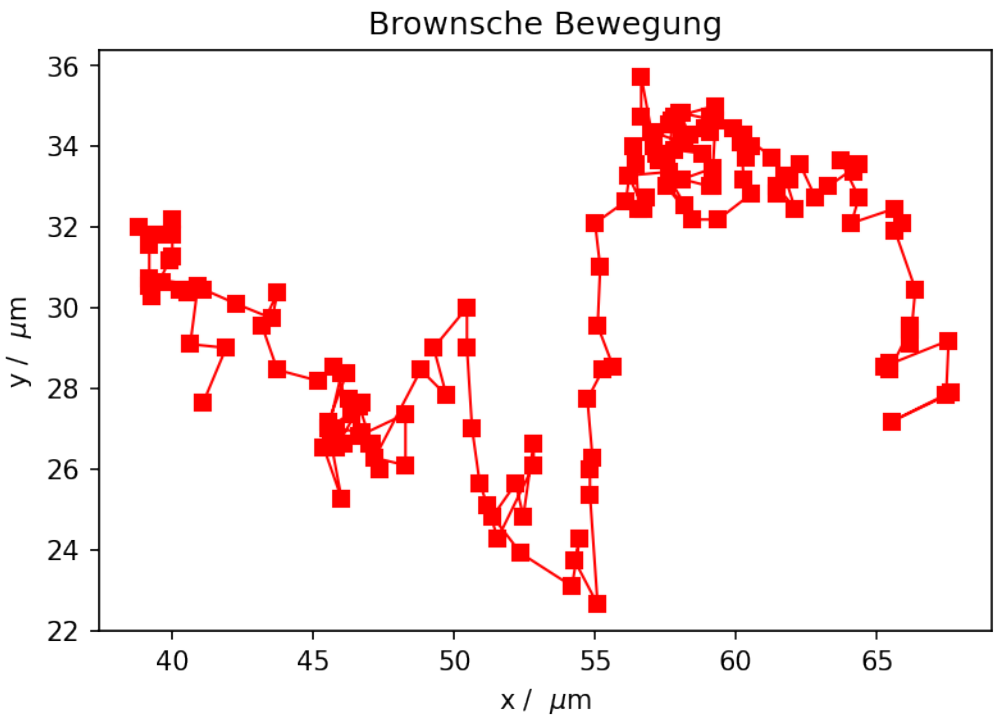
[1]:
```
#imports
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.mlab as mlab
import scipy.constants as c
from scipy.stats import norm
from scipy.optimize import curve_fit
```

[2]:
```
#Dateipfade
FILE_PATH = "../Input_Data/Versuch 223/position_data.txt"
OUTPUT_DIRECTORY = "../Plots/Versuch 223/"
```

[3]:
```
#Variablen aus Messprotokoll
d_particle = 755 * c.nano
d_particle_delta = 30 * c.nano
temp = 21.9 + 273.15
temp_delta = 0.1
calibration = 0.09091 * c.micro
```

[4]:
```
#Daten laden
t, x, y = np.loadtxt(FILE_PATH, delimiter=',', dtype = 'float', unpack = True)
```

[5]:
```
#Bewegung plotten
plt.figure(dpi=150)
plt.plot(x, y, marker = 's', color = 'red', linewidth = 1)
plt.xlabel('x / ' + ' $\mu$' + 'm')
plt.ylabel('y / ' + ' $\mu$' + 'm')
plt.title('Brownsche Bewegung')
plt.savefig(OUTPUT_DIRECTORY + 'Plot1.png')
plt.show()
```




[6]:
```
#quadratische Koordinatenänderungen berechnen
dx = np.array([])
dy = np.array([])

for i in range(0,len(t)-1):
    dx = np.append(dx, x[i+1] - x[i])
    dy = np.append(dy, y[i+1] - y[i])

r_squared = dx**2 + dy**2
```

[7]:
```
#mittlere quadratische Abweichung und dessen Fehler berechnen
r_squared_mean = np.mean(r_squared)*c.micro**2
r_squared_mean_std = np.std(r_squared)/np.sqrt(len(r_squared))*c.micro**2

print("mittlere quadratische Abweichung: \n" +  str(r_squared_mean))
print("Fehler: \n" +  str(r_squared_mean_std))
print("Diffusionskoeffizient: \n" +  str(r_squared_mean/4))
print("Fehler: \n" +  str(r_squared_mean_std/4))
```

```
mittlere quadratische Abweichung:
2.51766086885235e-12
Fehler:
2.1609008060045914e-13
Diffusionskoeffizient:
```

```
6.294152172130875e-13
Fehler:
5.4022520150114784e-14
```

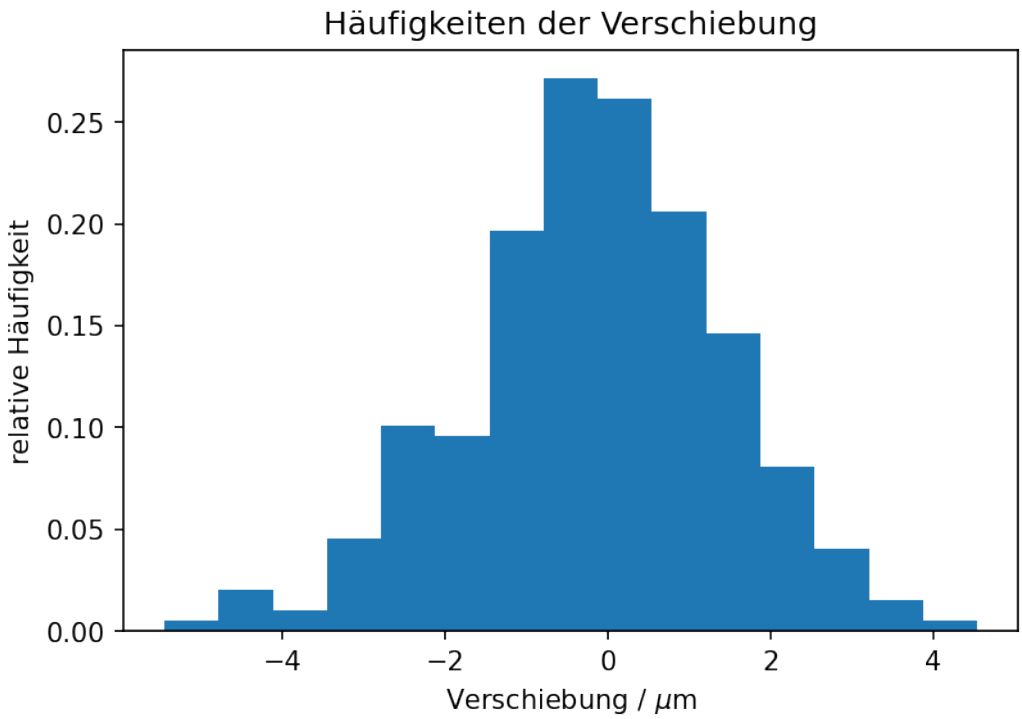
[8]:
```
#Boltzmannkonstante berechnen
eta = 9.6e-4
eta_delta = 0.03e-4
a = d_particle/2
a_delta = d_particle_delta/2
T = temp
T_delta = temp_delta

k1 = (6 * np.pi * r_squared_mean * eta * a)/(4 * T)
k1_delta = k1 * np.sqrt((eta_delta/eta)**2 +(a_delta/a)**2 +(T_delta/T)**2 +␣
↪(r_squared_mean_std/r_squared_mean)**2)
k1, k1_delta
```

[8]:
```
(1.4572394896134703e-23, 1.3790364964277638e-24)
```

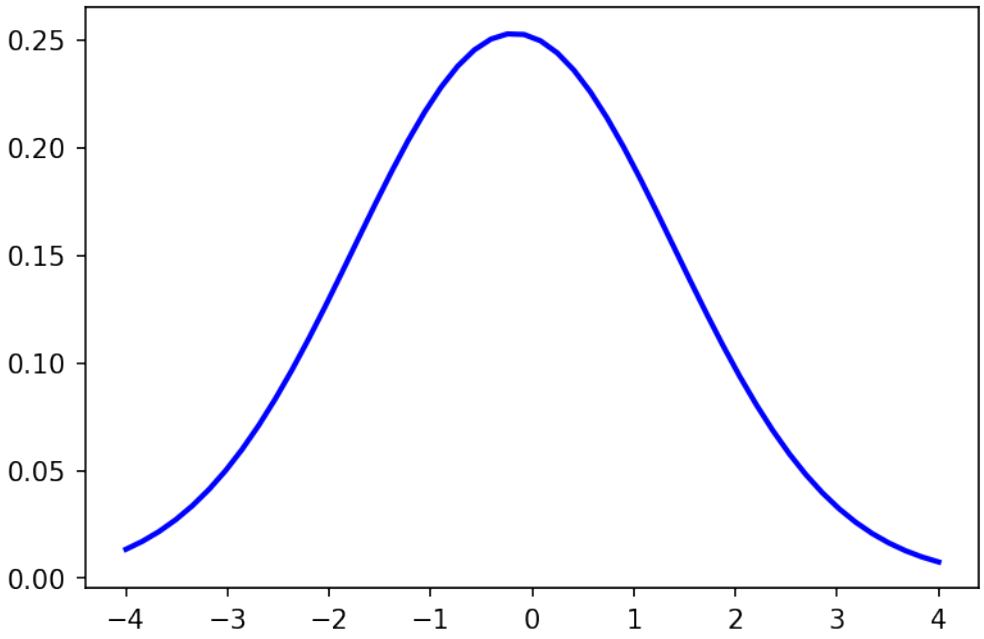
Kontrollverteilung

[9]:
```
#Histogramm plotten
all_data = np.append(dx,dy)
binwidth = 2/3
plt.figure(dpi=150)
plt.hist(all_data, bins = np.arange(min(all_data), max(all_data) + binwidth,␣
↪binwidth), density = True)
plt.xlabel('Verschiebung / $\mu$' + 'm')
plt.ylabel('relative Häufigkeit')
plt.title('Häufigkeiten der Verschiebung')
plt.savefig(OUTPUT_DIRECTORY + 'Plot2.png')
plt.show()
```



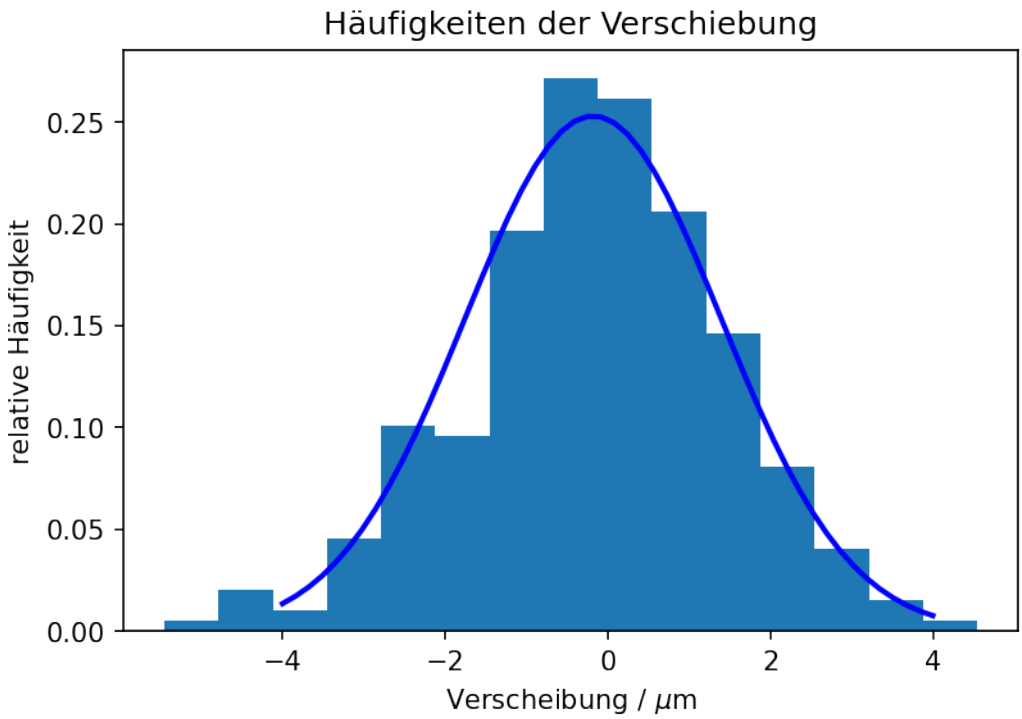

[10]:
```
#Gauß hinzufügen
mu = np.mean(all_data)
sigma = np.std(all_data)
x = np.linspace(-4,4)
gauss = norm.pdf(x, mu, sigma)

plt.figure(dpi=150)
plt.plot(x, gauss, 'b-', linewidth = 2)
plt.savefig(OUTPUT_DIRECTORY + 'Plot3.png')
plt.show()
mu,sigma
```



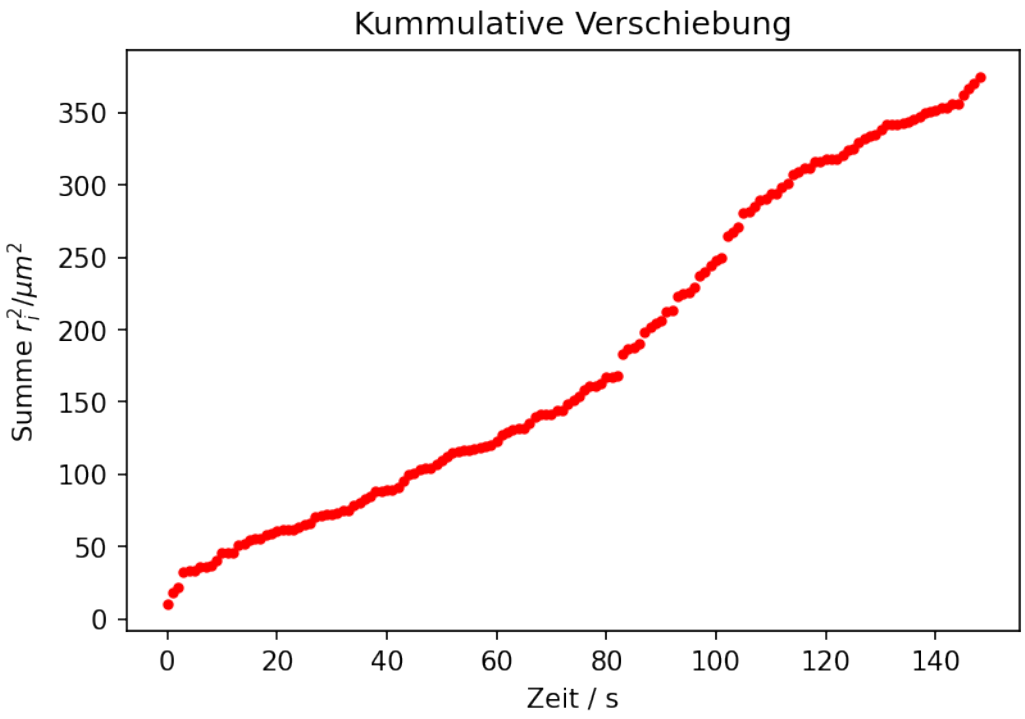

[10]: (-0.17998959731543623, 1.57647220518174)

[11]:
```
plt.figure(dpi=150)
plt.hist(all_data, bins = np.arange(min(all_data), max(all_data) + binwidth,␣
 ↪binwidth), density = True)
plt.xlabel('Verscheibung / $\mu$' + 'm')
plt.ylabel('relative Häufigkeit')
plt.title('Häufigkeiten der Verschiebung')
plt.plot(x, gauss, 'b-', linewidth = 2)
plt.savefig(OUTPUT_DIRECTORY + 'Plot4.png')
plt.show()
```




Kumulative Verteilung der Verschiebungsquadrate

[12]:
```
r_kumm = np.cumsum(r_squared)
plt.figure(dpi=150)
plt.plot(t[:-1], r_kumm, marker = '.', color = 'red', linewidth = 0)
plt.xlabel('Zeit / s')
plt.ylabel('Summe $r_i^2 /\mu m^2$')
plt.title('Kummulative Verschiebung')
plt.savefig(OUTPUT_DIRECTORY + 'Plot5.png')
plt.show()
```

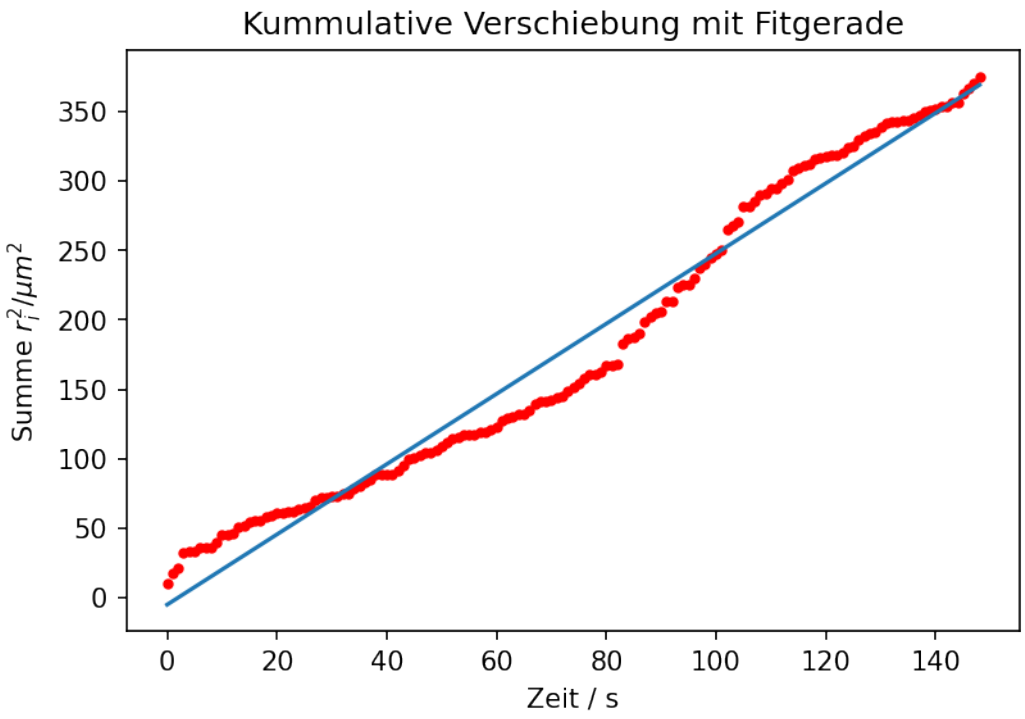



[13]:
```
#fitfunktion definieren
def lin(x,a,b):
    return a*x+b
```

[14]:
```
#fit durchführen
popt, pcov = curve_fit(lin, t[:-1], r_kumm)
popt[0], np.sqrt(pcov[0][0]), popt[0]/4, np.sqrt(pcov[0][0])/4
```

[14]:
```
(2.527932154055179,
 0.03438405314418733,
 0.6319830385137948,
 0.008596013286046833)
```

[15]:
```
plt.figure(dpi=150)
plt.plot(t[:-1], r_kumm, marker = '.', color = 'red', linewidth = 0)
plt.plot(t[:-1], lin(t[:-1], *popt))
plt.xlabel('Zeit / s')
plt.ylabel('Summe $r_i^2 /\mu m^2$')
plt.title('Kummulative Verschiebung mit Fitgerade')
plt.savefig(OUTPUT_DIRECTORY + 'Plot6.png')
```




[16]:
```
#Bestimmung der Boltzmannkonstante
k2 = 6*np.pi*eta*a*popt[0]*c.micro**2/(4*T)
k2_delta = k1 * np.sqrt((eta_delta/eta)**2 +(a_delta/a)**2 +(T_delta/T)**2 +␣
↪(np.sqrt(pcov[0][0])/popt[0])**2)
k2,k2_delta
```

[16]:
```
(1.463184580388729e-23, 6.137320375356281e-25)
```

Abweichungen bestimmen

[17]:
```
def calcDiff(x, deltax, y, deltay):
    num = np.abs(x-y)
    denom = np.sqrt((deltax)**2 + (deltay)**2)
    return(num/denom)

print(calcDiff(popt[0]*1e-12/4, np.sqrt(pcov[0][0])*1e-12/4, r_squared_mean/4,␣
↪r_squared_mean_std/4))
print(calcDiff(k1,k1_delta,c.k,0),calcDiff(k2,k2_delta,c.
↪k,0),calcDiff(k1,k1_delta,k2,k2_delta))
```

```
0.046941880109559006
0.5553913171396777 1.3448145988946796 0.039386072655445845
```