Protokoll PAP2 Versuch 252: Aktivierung von Indium und von Silber mit thermischen Neutronen

Leonard Scheuer

Motivation

In diesem Versuch sollen Halbswertszeiten und Zerfallskonstanten von $^{116}\mathrm{In}$, $^{108}\mathrm{Ag}$ und $^{110}\mathrm{Ag}$ bestimmt werden. Dafür werden diese mit thermischen Neutronen aktiviert.

Grundlagen

Die genutzten Ausgangsstoffe $^{115}\mathrm{In}$, $^{107}\mathrm{Ag}$ und $^{109}\mathrm{Ag}$. Die Aktivierung findet in einer Neutronenquelle statt. Diese besteht aus einem α-Strahler und Berilliumspänen, sodass sich folgende Reaktion vollziehen kann:

$$^{9}\mathrm{Be} + \alpha \rightarrow ^{12}\mathrm{Ce} + n \tag{1}$$

Die Quelle ist eingelassen in einen Parafinblock, sodass die jetzt noch viel zu energetischen Neutronen mit Stößen mit Wasserstoffatomen abgebremst werden, wir erhalten die thermischen Neutronen, welche von den Präperaten aufgenommen werden können. In der Neutronenquelle gilt für die Aktivierung:

$$A(t) = A_\infty(1 - e^{-\lambda t}) \tag{2}$$

wobei A_∞ die Anzahl der Zerfälle/Aktivierungen im Gleichgewicht ist. Wird die Aktivierung beendet so grieft das Zerfallsgesetz:

$$A(t) = A_0 \cdot e^{-\lambda t} \tag{3}$$

Mit Halbwertszeit

$$T_{1/2} = \frac{\ln(2)}{\lambda} \tag{4}$$

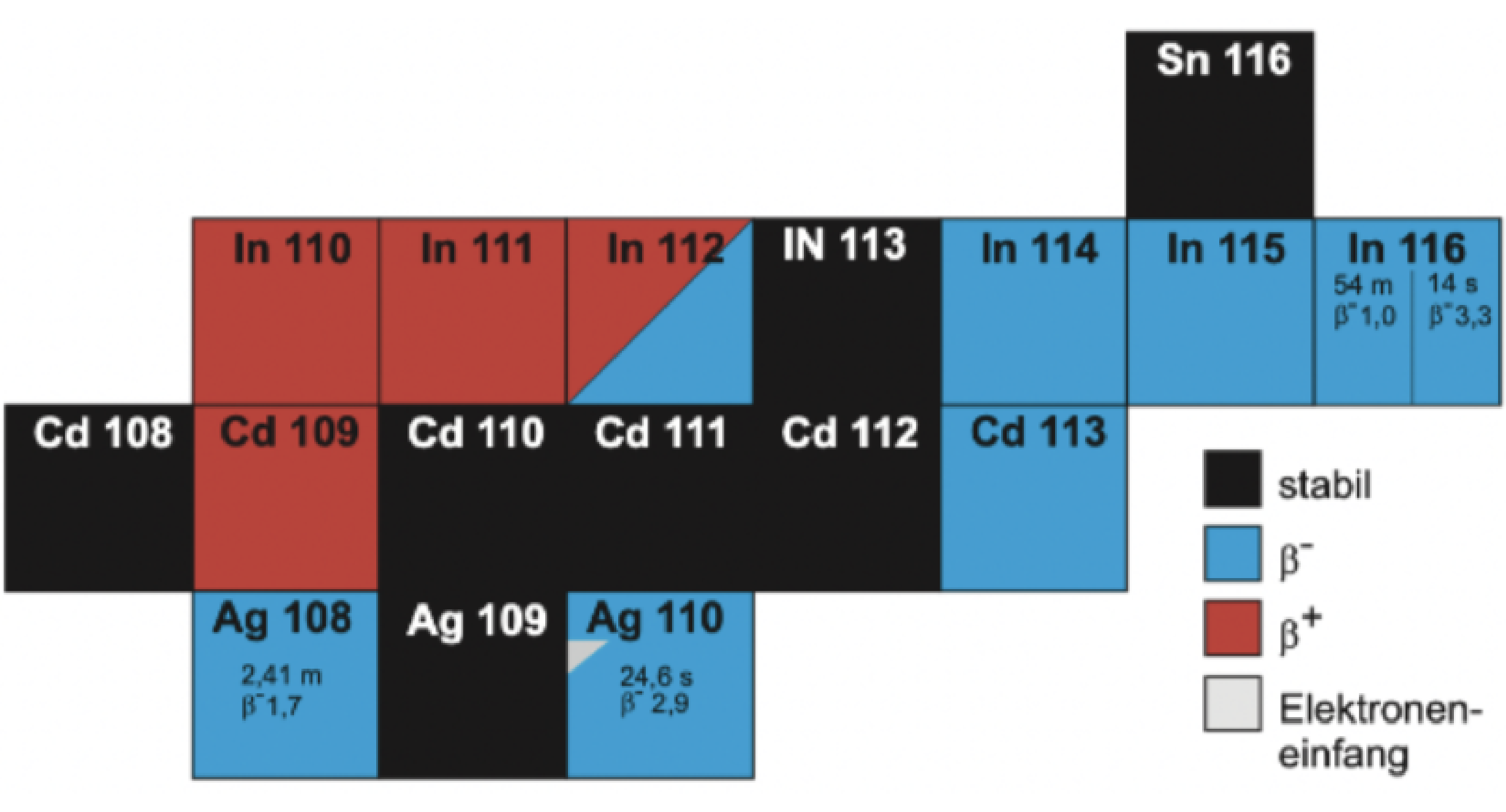


Abb. 1: Nuklidkarte (Quelle: Script)

Die zwei entstehenden Silberisotope haben sehr verschiedene Aktivierungszeiten, es ist daher möglich mit der Aktivierungszeit das Isotopenverhältnis zu wählen, s. Abb. 2.

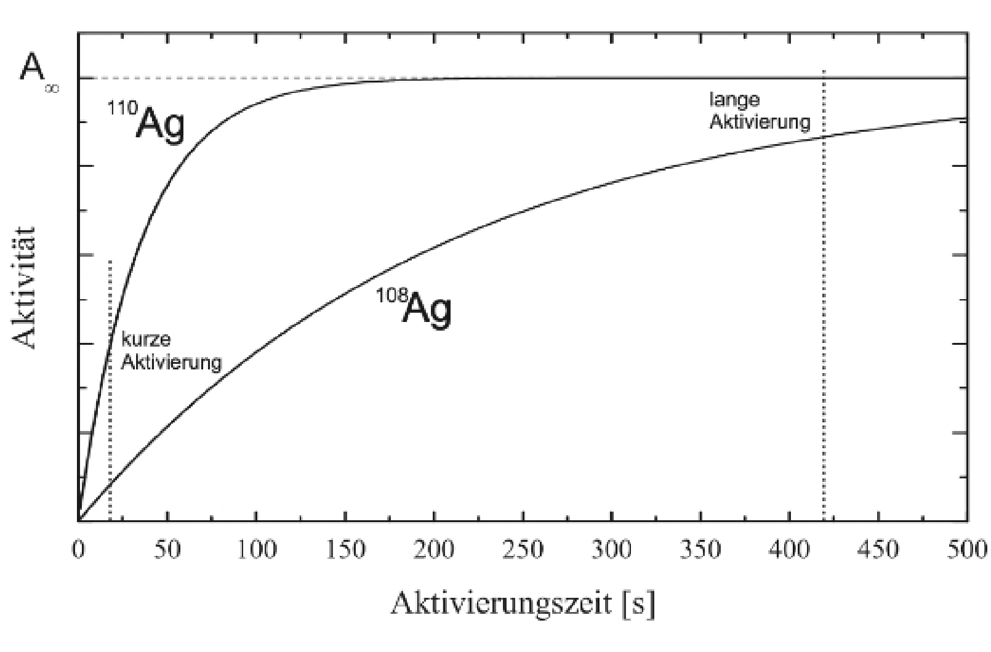


Abb. 2: Aktivierungsverhältnis (Quelle: Script)

Material

- Geiger-Müller Zählrohr mit Betriebsgerät
- Externer Impulszähler
- PC mit Drucker
- Neutronenquelle
- Präparatehalterung
- Indium- und Silberbleche

Durchführung

Untergrundmessung

Die Zählrohrspannung wird auf 500-550V gestellt, Torzeit 10 Sekunden. Untergrund wird über 8 Minuten gemessen.

Silber

Nach mindestens 7 Minuten Aktivierung wird das Silberblech vor dem Zählrohr fixiert und die Zählrate über 400 Sekunden gemessen.

Indium

Mit Messintervall von 120s wird über 50 Minuten der Zerfall des Indiumisotops nach vorheriger Aktivierung gemessen.

Messdaten

Die Daten sind in Dateien und in der Auswertung visualisiert.

Auswertung

Bestimmung des Untergrunds

In [1]:

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
from scipy.optimize import curve_fit
from scipy.stats import chi2

unterg = np.loadtxt('untergrund.dat', usecols=[1])

mittelw_unterg = np.mean(4 * unterg)
fehler_unterg = np.std(4 * unterg) / np.sqrt(len(unterg))

print('Mittelwert Untergrund: ({0:.1f} ± {1:.1f}) Ereignisse / 10s'.format(mittelw_unterg, fehler_unterg))
```

```
Mittelwert Untergrund: (14.7 ± 1.2) Ereignisse / 10s
```

Bestimmung der Halbwertszeit von Silber

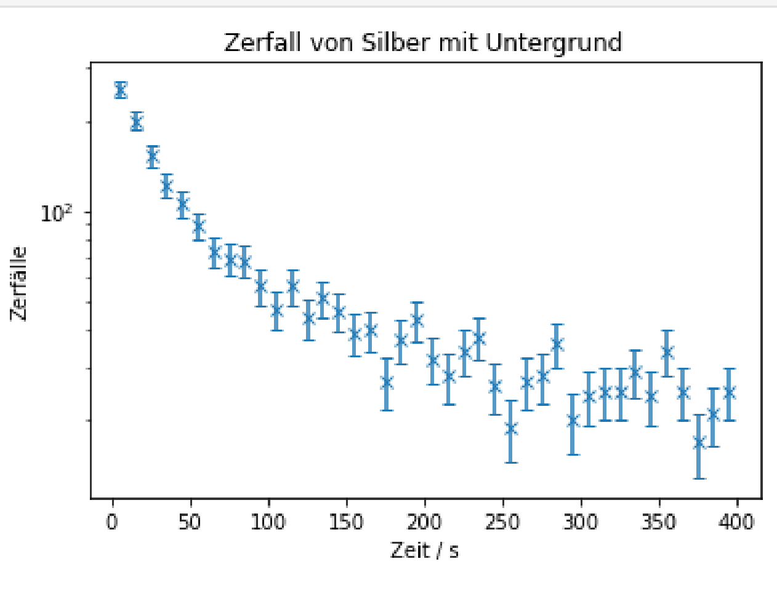
In [2]:

```
n1 = np.loadtxt('silber_1.dat', usecols = [1])
n2 = np.loadtxt('silber_2.dat', usecols = [1])
n3 = np.loadtxt('silber_3.dat', usecols = [1])
n4 = np.loadtxt('silber_4.dat', usecols = [1])

N = n1 + n2 + n3 + n4
Fehler_N = np.sqrt(N)

t = np.arange(5,405,10)

plt.errorbar(t, N, Fehler_N, linestyle='None', fmt = 'x', capsize = 3)
plt.xlabel('Zeit / s')
plt.ylabel('Zerfälle')
plt.title('Zerfall von Silber mit Untergrund')
plt.yscale('log')
```



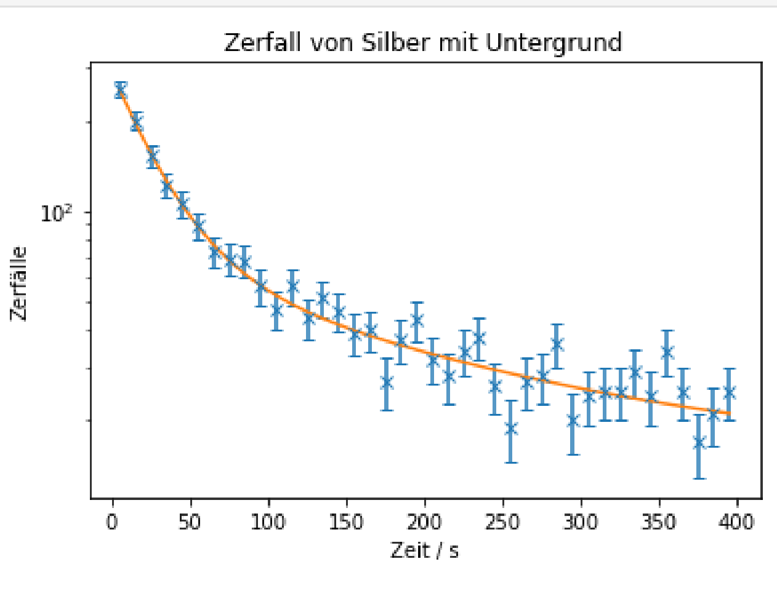
Da sowohl ^{108}Ag als auch ^{110}Ag zerfallen, fitten wir die Superposition der Zerfallsfunktionen gewähltund addieren den Untergrund A_{UG}:

$$A(t) = A_1 e^{-\lambda_1 t} + A_2 e^{-\lambda_2 t} + A_{UG}$$

In [3]:

```
def fit_func(x, A1, l1, A2, l2):
    return A1 * np.exp(-x *l1)+A2*np.exp(-x * l2) + y0
y0 = mittelw_unterg
popt, pcov = curve_fit(fit_func, t, N, p0 = [500,0.02,50,0.001], sigma = Fehler_N)

plt.errorbar(t, N, Fehler_N, linestyle='None', fmt = 'x', capsize = 3)
plt.xlabel('Zeit / s')
plt.ylabel('Zerfälle')
plt.title('Zerfall von Silber mit Untergrund')
plt.yscale('log')
plt.plot(t, fit_func(t, *popt))
```



In [4]:

```
print("A1 = ({0} ± {1}) Ereignisse / s".format(int(popt[0]), int(np.sqrt(pcov[0][0]))))
print("l1 = ({0:.3f} ± {1:.3f}) / s".format(popt[1], np.sqrt(pcov[1][1])))
print("A2 = ({0} ± {1}) Ereignisse / s".format(int(popt[2]), int(np.sqrt(pcov[2][2]))))
print("l2 = ({0:.4f} ± {1:.4f}) / s".format(popt[3], np.sqrt(pcov[3][3])))
```

```
A1 = (217 ± 16) Ereignisse / s
l1 = (0.035 ± 0.005) / s
A2 = (57 ± 12) Ereignisse / s
l2 = (0.0055 ± 0.0009) / s
```

In [5]:

```
chi2_ = np.sum((fit_func(t, *popt) - N)**2/ Fehler_N**2)
dof = len(N) - 4 #dof:degrees of freedom, Freiheitsgrad
chi2_red = chi2_ / dof

print("chi2 = {0:.1f}".format(chi2_))
print("chi2_red = {0:.1f}".format(chi2_red))
```

```
chi2 = 28.9
chi2_red = 0.8
```

In [6]:

```
prob = round(1 - chi2.cdf(chi2_, dof), 2) * 100
print("Fitwahrscheinlichkeit: ", prob, "%")
```

```
Fitwahrscheinlichkeit:  79.0 %
```

Um den Fehler des Untergrundes in den Fehler der Zerfallskonstante einzubeziehen, wird der Fit für den Mittelwert $\pm$ dem 1σ-Fehler wiederholt, die Differenz zu λ gemittelt und diesen Fehler quadratisch zu dem ursprünglichen Fehler addiert:

In [7]:

```
y0 = mittelw_unterg - fehler_unterg
popt1, pcov1 = curve_fit(fit_func, t, N, p0 = [500,0.02,50,0.001], sigma = Fehler_N)

y0 = mittelw_unterg + fehler_unterg
popt2, pcov2 = curve_fit(fit_func, t, N, p0 = [500,0.02,50,0.001], sigma = Fehler_N)

err_1 = np.mean(np.array([np.abs(popt1 - popt)[1] , np.abs(popt2 - popt)[1]]))
err_2 = np.mean(np.array([np.abs(popt1 - popt)[3] , np.abs(popt2 - popt)[3]]))

err_1 = np.sqrt(err_1**2 + pcov[1][1])
err_2 = np.sqrt(err_2**2 + pcov[3][3])

print('Zerfallskonstante 1: ({0:.3f} ± {1:.3f})/s'.format(popt[1], err_1))
print('Zerfallskonstante 2: ({0:.4f} ± {1:.4f})/s'.format(popt[3], err_2))
```

```
Zerfallskonstante 1: (0.035 ± 0.005)/s
Zerfallskonstante 2: (0.0055 ± 0.0011)/s
```

Nach Gleichung (4) finden wir den Fehler der Halbwertszeit zu

$$\Delta t_H = \frac{\ln(2)}{\lambda^2}\Delta\lambda$$

In [8]:

```
t_hw_1 = np.log(2) / popt[1]
t_hw_2 = np.log(2) / popt[3]

t_hw_1_err = t_hw_1 * err_1 / popt[1]
t_hw_2_err = t_hw_2 * err_2 / popt[3]

print('Halbwertszeit 1: ({0:.1f} ± {1:.1f})s'.format(t_hw_1, t_hw_1_err))
print('Halbwertszeit 2: ({0} ± {1})s'.format(int(t_hw_2), int(t_hw_2_err)))
```

```
Halbwertszeit 1: (19.8 ± 2.8)s
Halbwertszeit 2: (125 ± 24)s
```

Da die Halbwertszeit von ^{110}Ag kürzer als die von ^{108}Ag ist, gehört die erste Halbwertszeit zu ^{110}Ag, die zweite zu ^{108}Ag.

In [9]:

```
t_lit_1 = 24.6
t_lit_2 = 2.41 * 60

sigma_1 = np.abs(t_lit_1 - t_hw_1) / t_hw_1_err
sigma_2 = np.abs(t_lit_2 - t_hw_2) / t_hw_2_err

print('Abweichung vom Literaturwert 1: {0:.2f}'.format(sigma_1))
print('Abweichung vom Literaturwert 1: {0:.2f}'.format(sigma_2))
```

```
Abweichung vom Literaturwert 1: 1.74
Abweichung vom Literaturwert 1: 0.81
```

Bestimmung der Halbwertszeit von Indium

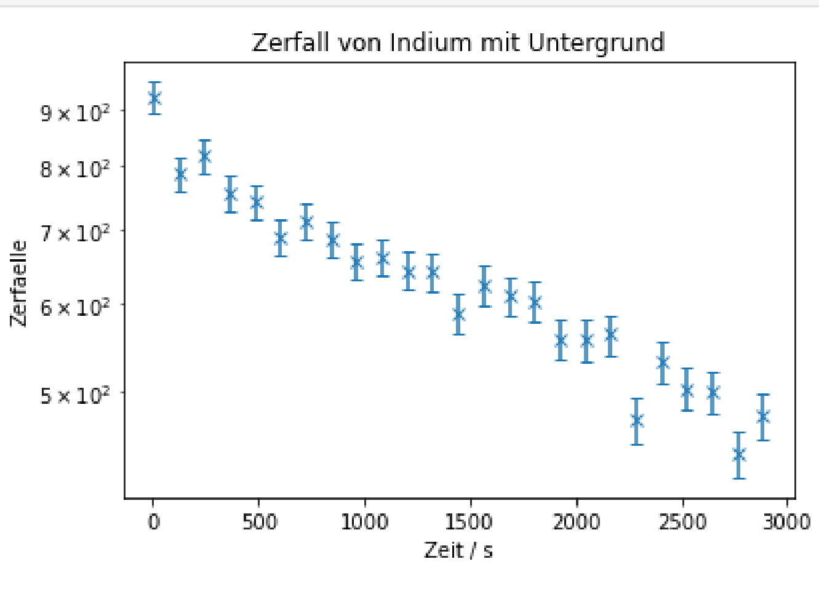
Da die Messzeit bei Indium bei 120s lag, muss der Untergrund $\cdot 12$ genommen werden, damit die Untergrundereignisse / 120s bekannt sind:

In [10]:

```
N = np.loadtxt('indium.dat', usecols = [1])
Fehler_N = np.sqrt(N)
t = np.arange(5,3005,120)

mittelw_unterg = np.mean(unterg) * 12 # Ereignisse / 120s
fehler_unterg = np.std(unterg) / np.sqrt(len(unterg)) * 12

plt.errorbar(t, N, Fehler_N, linestyle = 'None', fmt = 'x', capsize = 3)
plt.xlabel('Zeit / s')
plt.ylabel('Zerfaelle')
plt.title('Zerfall von Indium mit Untergrund')
plt.yscale('log')
```



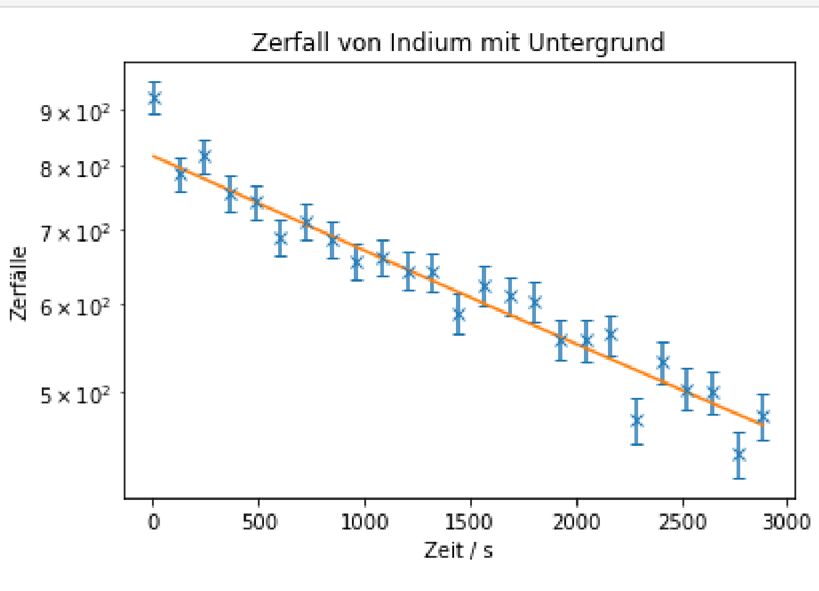
Wir vernachlässigen den ersten Messwert um das metastabile $^{116m}\mathrm{In}$ nicht mit einzubeziehen und fitten eine Funktion wie in (3) an.

In [11]:

```
def fit_func(x, A1, l1):
    return A1 * np.exp(-x * l1) + y0

y0 = mittelw_unterg
popt, pcov = curve_fit(fit_func, t[1:], N[1:], p0 = [500,0.0], sigma = Fehler_N[1:])

plt.errorbar(t, N, Fehler_N, linestyle='None', fmt = 'x', capsize = 3)
plt.xlabel('Zeit / s')
plt.ylabel('Zerfälle')
plt.title('Zerfall von Indium mit Untergrund')
plt.yscale('log')
plt.plot(t, fit_func(t, *popt))
plt.savefig('indium.jpeg')
```



In [12]:

```
print("A1 = ({0} ± {1}) Ereignisse / s".format(int(popt[0]), int(np.sqrt(pcov[0][0]))))
print("l1 = ({0:.6f} ± {1:.6f}) / s".format(popt[1], np.sqrt(pcov[1][1])))
```

```
A1 = (772 ± 12) Ereignisse / s
l1 = (0.000209 ± 0.000010) / s
```

In [13]:

```
chi2_ = np.sum((fit_func(t[1:], *popt) - N[1:])**2/ Fehler_N[1:]**2)
dof = len(N[1:]) - 2 #dof:degrees of freedom, Freiheitsgrad
chi2_red = chi2_ / dof

print("chi2 = {0:.1f}".format(chi2_))
print("chi2_red = {0:.1f}".format(chi2_red))
```

```
chi2 = 20.9
chi2_red = 0.9
```

In [14]:

```
prob = round(1 - chi2.cdf(chi2_, dof), 2) * 100
print("Fitwahrscheinlichkeit = ", prob, "%")
```

```
Fitwahrscheinlichkeit =  53.0 %
```

In [15]:

```
y0 = mittelw_unterg - fehler_unterg
popt1, pcov1 = curve_fit(fit_func, t, N, p0 = [500,0.02], sigma = Fehler_N)

y0 = mittelw_unterg + fehler_unterg
popt2, pcov2 = curve_fit(fit_func, t, N, p0 = [500,0.02], sigma = Fehler_N)

err_1 = np.mean(np.array([np.abs(popt1 - popt)[1] , np.abs(popt2 - popt)[1]]))

err_1 = np.sqrt(err_1**2 + pcov[1][1])

print('Zerfallskonstante: ({0:.6f} ± {1:.6f})/s'.format(popt[1], err_1))
```

```
Zerfallskonstante: (0.000209 ± 0.000016)/s
```

In [16]:

```
t_hw_1 = np.log(2) / popt[1]

t_hw_1_err = t_hw_1 * err_1 / popt[1]

print('Halbwertszeit: ({0} ± {1})s'.format(int(t_hw_1 / 10) * 10, int((t_hw_1_err+5) / 10) * 10))
```

```
Halbwertszeit: (3310 ± 250)s
```

In [17]:

```
t_lit_1 = 54 * 60

sigma_1 = np.abs(t_lit_1 - t_hw_1) / t_hw_1_err

print('Abweichung vom Literaturwert: {0:.2f}'.format(sigma_1))
```

```
Abweichung vom Literaturwert: 0.28
```

Diskussion

In diesem Versuch haben wir das Zerfallsgesetz verifizieren können. Dazu wurden die Halbwertszeiten von $^{116}\mathrm{In}$, $^{110}\mathrm{Ag}$ und $^{108}\mathrm{Ag}$ bestimmt, die sich nicht signifikant von den Literaturwerten unterscheiden ($^{116}\mathrm{In} : 0,28\sigma$, $^{110}\mathrm{Ag} : 1,74\sigma$, $^{108}\mathrm{Ag} : 0,81\sigma$). Insbesondere die Halbwertszeit von Indium wurde also sehr präzise bestimmt. Aufgrund der ohnehin statistischen Natur des Zerfalles scheinen alle erhaltenen Werte sehr plausibel, die Fits liefern hohe Fitwahrscheinlichkeiten (>50%, höhere Fitwahrscheinlichkeiten haben ohnehin nur geringere Aussagekraft, da der chi2-red wert dann zu klein wird und die Fitwahrscheinlichkeit bei Overfitting nicht wieder abnimmt.)

Zusammenfassend konnten wir die vorgelegte Theorie und Literaturwerte gut bestätigen.