

Versuch 251: Statistik des radioaktiven Zerfalls

Matthias Kuntz

17.06.2024

Inhaltsverzeichnis

1 Einleitung	3
1.1 Physikalische Grundlagen	3
1.1.1 Das Geiger-Müller-Zählrohr	3
1.2 Statistik	5
1.2.1 Poisson-Verteilung	5
1.2.2 Gauß-Verteilung	5
1.3 Versuchsaufbau	6
2 Versuchsprotokoll mit Messdaten	7
3 Auswertung	11
3.1 Auswertung des Plateau-Bereichs	12
3.2 Auswertung der Messung bei großer Ereigniszahl	15
3.3 Auswertung der Messung bei kleiner Ereigniszahl	16
4 Zusammenfassung der Endergebnisse	18
5 Diskussion	19

1 Einleitung

In diesem Versuch soll der statistische Prozess des radioaktiven Zerfalls eines ^{60}Co -Präparats untersucht werden, um die Unterschiede statistischer Modelle zu erarbeiten. Dazu werden zunächst die Charakteristik des verwendeten Zählrohrs und anschließend Zählraten bei verschiedenen Konfigurationen untersucht. Insbesondere interessieren wir uns dabei für die Unterschiede zwischen einer Poisson- und Gauß-Verteilung.

1.1 Physikalische Grundlagen

1.1.1 Das Geiger-Müller-Zählrohr

Das Geiger-Müller-Zählrohr, schematisch dargestellt in Abbildung 1, besteht aus einem Metallzylinder und darin axial verlaufendem Anodendraht, zwischen denen eine Spannung angelegt wird. Die Spannung sorgt dafür, dass die Ionisierung des Füllgases im Zählrohr, die passiert, wenn ein radioaktives Teilchen durch dieses durchfliegt, und die somit entstehende Gasentladung einen messbaren Strom im Zählrohr erzeugt, der über einen Widerstand einen Spannungsimpuls erzeugt, welcher anschließend von einer Zählerschaltung registriert werden kann. So kann man effektiv zählen, wie viele radioaktive Teilchen in das Zählrohr eindringen.

Durch verschiedene Effekte wie Rekombination oder die zusätzliche Ionisierung

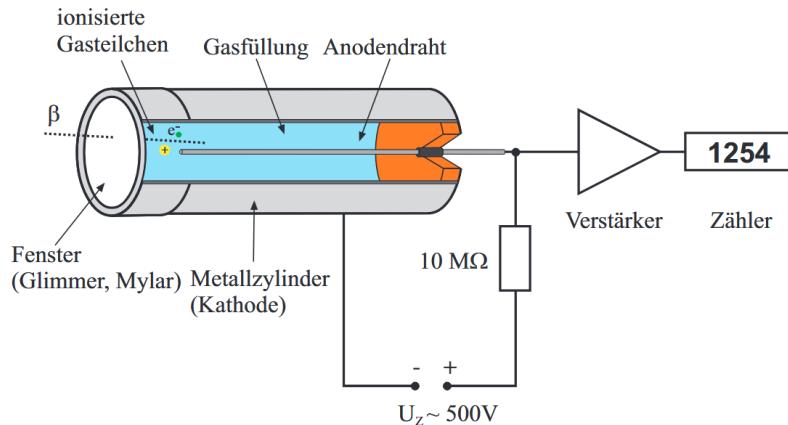


Abbildung 1: Geiger-Müller-Zählrohr - Aufbau [Quelle: PAP2.2 Skript, S.51, Stand: 30.07.2024]

sekundärer Gasmoleküle durch bereits gestoßene Elektronen bei hohen Energien der radioaktiven Strahlung ergibt sich eine Zählrohrspannungs-abhängige Charakteristik der Zählrate, zu sehen in Abbildung 2. Wir interessieren uns hier insbesondere für den Plateau-Bereich, in dem jedes einfallende Teilchen unabhängig von der eigenen Energie ein gleich großes Entladungssignal erzeugt. Das passiert, wenn die Spannung so gewählt wird, dass jedes einfallende Teilchen genug Energie hat, um eine Elektronenlawine entlang des Anodendrahtes auszulösen. In der Praxis ist aber auch dieses Plateau nicht exakt gerade, sondern besitzt aufgrund der Inhomogenität des elektrischen Feldes einen leichten Anstieg. Nach jeder Entladung benötigt das Zählrohr erstmal eine Regenerationszeit, in der es unempfindlich für neu eintretende Strahlung ist. Diese Zeit beträgt typischerweise 10^{-4} und ist somit bereits ab einer Rate von ca. 200 Impulsen pro Sekunde spürbar. Ebenso muss immer eine Hintergrundstrahlung berücksichtigt werden.

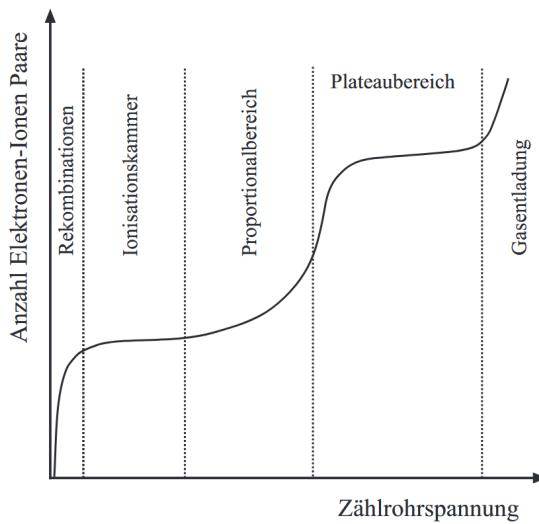


Abbildung 2: Schematische Zählrohrcharakteristik [Quelle: PAP2.2 Skript, S.52, Stand: 30.07.2024]

1.2 Statistik

Generell ist der mittlere statistische Fehler einer Zählung von n Teilchen durch \sqrt{n} gegeben. Bei unseren radioaktiven Zerfällen handelt es sich um unabhängige Ereignisse für jeden Zerfall, weshalb die Zerfallswahrscheinlichkeit p nach einer gegebenen Beobachtungszeit t gegeben ist durch:

$$p(t) = 1 - e^{-\lambda t}. \quad (1)$$

Lambda bezeichnet hierbei die Zerfallskonstante und ist eine charakteristische Größe für alle Isotope.

In den folgenden Abschnitten bezeichnet k die Anzahl der eintretenden Ereignisse A bei n unabhängigen Versuchen und der Wahrscheinlichkeit p , dass Ereignis A passiert.

1.2.1 Poisson-Verteilung

Die Poisson-Verteilung geht aus der Binomialverteilung hervor in den Limits der Zerfallswahrscheinlichkeit $p \rightarrow 0$ und der Anzahl radioaktiver Atome $n \rightarrow \infty$ mit endlich bleibendem Mittelwert $\mu = \langle k \rangle = np$, durch welchen sie komplett parametrisiert wird:

$$P(k; \mu) = \frac{1}{k!} \mu^k e^{-\mu}. \quad (2)$$

Hierbei bezeichnet μ somit zugleich den Mittelwert und die Varianz, die Standardabweichung beträgt also $\sqrt{\mu}$. Für kleine μ ist die Verteilung stark asymmetrisch um den Erwartungswert und nähert sich für große μ der symmetrisch verteilten Gauß-Verteilung an.

1.2.2 Gauß-Verteilung

Die Gauß-Verteilung wird von zwei separaten Parametern beschrieben, dem Mittelwert μ und der Standardabweichung σ , und nimmt die folgende allgemeine Form an:

$$G(k; \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(\mu-k)^2}{2\sigma^2}} \quad (3)$$

Im Spezialfall einer Zählstatistik ergibt sich analog zur Poisson-Verteilung, dass der Mittelwert der Varianz entspricht. Allgemein lassen sich aber mit der Standardabweichung σ sogenannte Konfidenzintervalle definieren, die aussagen, wie wahrscheinlich es ist, dass ein Messwert stärker vom Mittelwert abweicht als $\mu \pm x \cdot \sigma$ mit $x \in \mathbb{N}$. Ebenso lässt sich aus der Standardabweichung die Halbwertsbreite $FWHM$ bestimmen:

$$FWHM \approx 2,4\sigma \quad (4)$$

1.3 Versuchsaufbau

Der Versuchsaufbau, zu sehen in Abbildung 3, besteht aus einem Geiger-Müller-Zählrohr, das auf einer Halterungsschiene angebracht ist, auf welcher ebenso das radioaktive Präparat in verstellbarem Abstand positioniert werden kann. Das Signal des Zählrohrs wird an das Betriebsgerät und über einen externen Zähler an einen Computer gesendet. Am Betriebsgerät können die Zählrohrspannung sowie die Messdauer eingestellt werden.

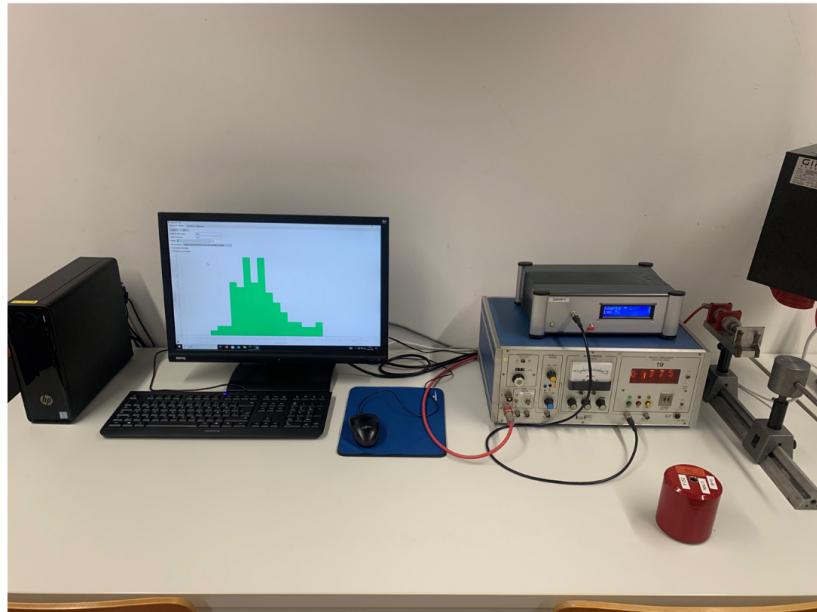


Abbildung 3: Versuchsaufbau [Quelle: PAP2.2 Skript, S.55, Stand: 30.07.2024]

2 Versuchsprotokoll mit Messdaten

Messprotokoll Versuch 251: Statistik der radioaktiven Zählrate

Tutor: Leonard Bender

17.06.2024, 14⁰⁰ – 17⁰⁰ Uhr

Nicole Scherst
Matthias Kuntz

Materialien

- Geiger - Müller Zählrohr mit Betriebegerät
- Externer Impulszähler
- PC
- Präparatschalterung mit Bleibeschirmung
- Radioaktives Präparat (⁶⁰Co)

Versuchsaufbau

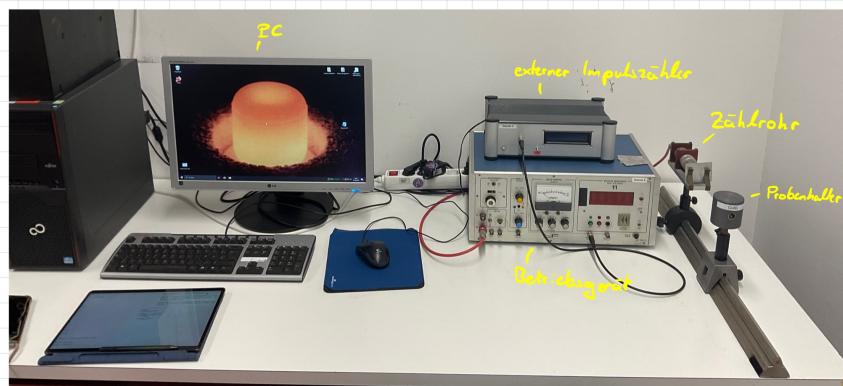


Foto: Versuchsaufbau

1) Messung der Zählrohrcharakteristik

Wir montieren unser Präparat und beginnen, indem wir die Eingangsspannung des Zählrohrs V_E bestimmen. Der Fehler wird aus der Einstellgenauigkeit abgeschätzt.

$$V_E = (460 \pm 10) V$$

Wir beginnen bei $U = V_E + 50V$ und messen bis zu $U = V_E + 150V$ in 25V-Schritten die Zählrate bei einer Messzeit von 30 Sekunden.

$U - V_E$ [V]	Zählrate
0	1655
25	1782
50	1754
75	1734
100	1735
125	1762
150	1861

Tabelle 1: Messung Zählrohrplateau

Wir stellen für die folgenden Versuchsschritte die Spannung

$$U_0 = V_E + 75 V = (535 \pm 10) V$$

ein.

2) Untersuchung des Plateauanstiegs

Mit dem Präparat so nah wie möglich am Zählrohr messen wir bei unterschiedlichen Messdauern die Zählrate bei U_0 & $U_0 + 100V$.

U	Messdauer [min]	Zählrate
U_0	1	14328
U_0	3	42853
$U_0 + 100V$	1	14910
$U_0 + 100V$	3	44817

Tabelle 2: Plateauanstieg

3) Verifizierung der statistischen Natur des radioaktiven Zerfalls

Bei der Spannung U_0 messen wir digital 2000 mal die Zerfälle innerhalb einer festen Torszeit von 500ms.

Das Programm speichert dabei automatisch ein Histogramm ab und wir notieren zusätzlich die automatisch berechneten Werte:

Größe	Wert
#Messungen	2000
μ	72,3
σ_{exp}	8,36
σ_{theo}	8,5

Tabelle 3: Langzeitmessung, $T=500\text{ms}$

4) Vergleich der Poisson- & Gaußverteilung bei kleinen Zählraten

Wir wiederholen den letzten Versuchsteil mit einer Torszeit von 100ms und ca.

5000 Messungen.

Größe	Wert
#Messungen	5000
μ	4,61
σ_{exp}	2,13
σ_{theo}	2,15

Tabelle 4: Langzeitmessung, $T=100\text{ms}$

Berl

3 Auswertung

In dieser Evaluation werden alle Fehler, sofern keine spezifische Angabe gemacht wird, mithilfe der Gauss'schen Fehlerfortpflanzung berechnet. Dies bedeutet, dass ein Wert F , der mit der Formel $f(a_1, \dots, a_n)$ berechnet wird, den Fehler ΔF annimmt:

$$\Delta F = \sqrt{\sum_n \left(\frac{\partial f}{\partial a_n} \cdot \Delta a_n \right)^2}. \quad (5)$$

Des Weiteren erfolgen Signifikanztests von zwei Werten a und a' über die folgende Formel:

$$\sigma = \frac{|a - a'|}{\sqrt{(\Delta a)^2 + (\Delta a')^2}}. \quad (6)$$

Die Auswertung sowie Berechnung erfolgen über das dem Dokument angehängte Python-Programm. Hierbei erfolgen Fits von Funktionen mithilfe der 'curve_fit'-Funktion des 'SciPy'-Packages und Plots werden mit 'matplotlib' erstellt.

Die Güte eines Fits wird mit der χ^2 -Summe bewertet:

$$\chi^2 = \sum_i^N \left(\frac{\text{Funktionswert}_i - \text{Messwert}_i}{\text{Fehler}_i} \right)^2 \quad (7)$$

Auch verwendet wird $\chi^2_{red} = \chi^2/f$, wobei der Freiheitsgrad f die Anzahl der Messwerte minus die Anzahl der Fitparameter ist. Der auf die Freiheitsgrade normierte Wert soll bei einem guten Fit ungefähr 1 sein.

3.1 Auswertung des Plateau-Bereichs

Wir beginnen, indem wir unsere Messungen aus Tabelle 1 des Messprotokolls in ein Diagramm auftragen, wobei sich die Fehler der Zählraten N gemäß \sqrt{N} ergibt. An den Verlauf fitten wir eine lineare Funktion, wobei wir nur Punkte berücksichtigen, die auch im Plateau-Bereich liegen. Es ergibt sich das in Abbildung 4 dargestellte Plot. Wie gut zu erkennen ist, ist ein deutlicher Anstieg der Zählraten über dem Plateau zu erkennen und die Gerade hat eine sichtbare Steigung. Durchaus interessant ist jedoch, dass unsere Messwerte kurzzeitig abzufallen scheinen, was aber im Endeffekt auf statistische Schwankungen zurückzuführen ist.

Anschließend bestimmen wir aus den in Tabelle 2 aufgezeichneten Messwerten quantitativ die prozentualen Plateau-Anstiege A gemäß

$$A = \frac{n(U_0 + 100V) - n(U_0)}{n(U_0)}$$

$$\Rightarrow \Delta A = A \sqrt{\left(\frac{\sqrt{\Delta n(U_0 + 100V)^2 + \Delta n(U_0)^2}}{n(U_0 + 100V) - n(U_0)} \right)^2 + \left(\frac{\Delta n(U_0)}{n(U_0)} \right)^2} \quad (8)$$

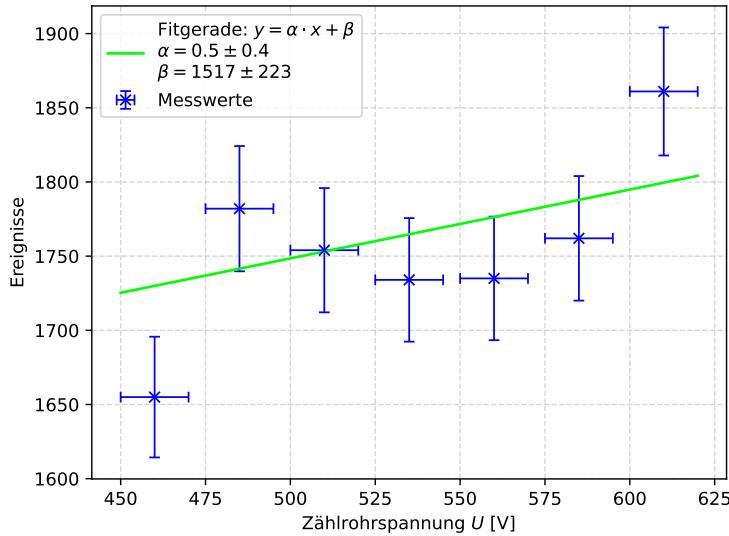


Abbildung 4: A1 - Plateaumessung mit Linearem Fit

Die Ergebnisse für die verschiedenen Messdauern werden zusammen mit Signifikanztests in Tabelle 5 eingetragen. Die Signifikanztests erfolgen zu dem theoretischen Wert einer Steigung von 0%.

Messdauer [min]	Anstieg [%]	Signifikanztest [σ]
1	$4,1 \pm 1,2$	3,40
3	$4,6 \pm 0,7$	6,61

Tabelle 5: Prozentualer Anstieg mit Signifikanztests

Es ist zu erkennen, dass bei beiden Messzeiten die bestimmten Steigungen signifikant von 0 abweichen was darauf schließen lässt, dass bei einem realistischen Zählrohr wie erwartet eine nicht vernachlässigbare Steigung des Plateaus erhalten bleibt aufgrund von Inhomogenitäten des elektrischen Feldes und Nachentladungen.

Wir möchten nun bestimmen, wie lange man messen müsste, um den Plateau-Anstieg auf 1% genau zu kennen. Dazu betrachten wir den relativen Fehler des absoluten Anstiegs $d = n(U_0 + 100V) - n(U_0)$, wobei die Anzahl proportional zur Messzeit t und somit der Fehler proportional zu \sqrt{t} ist:

$$\frac{\Delta d}{d} = \frac{\sqrt{\Delta n(U_0 + 100V)^2 + \Delta n(U_0)^2}}{n(U_0 + 100V) - n(U_0)} \propto \frac{\sqrt{t}}{t} = \frac{1}{\sqrt{t}}. \quad (9)$$

Da nun $\Delta d/d = 0,01$ gelten soll erhalten wir den folgenden Zusammenhang mit der Messzeit t einer Messung und der Gesamtmeessdauer t_{ges} :

$$\begin{aligned} \frac{\sqrt{t_{ges}}}{\sqrt{t}} &= \frac{\Delta d/d}{0,01} \\ \Rightarrow t_{ges} &= \left(\frac{\Delta d/d}{0,01} \right)^2 t \end{aligned} \quad (10)$$

Somit erhalten wir die in Tabelle 6 dargestellten nötigen Gesamtmeessdauern bei den gegebenen Meessdauern einer Einzelmessung, um den Anstieg auf 1% Genauigkeit zu bestimmen.

Messdauer [min]	rel Fehler $\Delta d/d$ [%]	t_{ges} [s]	t_{ges} [h]
1	29,38	51790,84	14,39
3	15,12	41159,34	11,43

Tabelle 6: Benötigte Messdauern

Zuletzt betrachten wir noch die Konfidenzintervalle unserer prozentualen Steigungen, dargestellt in Tabelle 7, mit den Werten für die 1-Minute Messung in der ersten und denen der 3-Minuten Messung in der zweiten Zeile. Dazu addieren beziehungsweise subtrahieren wir den Fehler beziehungsweise den doppelten Fehler von unserem Ergebnis.

-2σ [%]	$-\sigma$ [%]	A [%]	$+\sigma$ [%]	$+2\sigma$ [%]
1,67	2,87	$4,1 \pm 1,2$	5,26	6,45
3,19	3,88	$4,6 \pm 0,7$	5,26	5,95

Tabelle 7: Prozentualer Anstieg - Konfidenzintervalle

Diese Werte sagen aus, dass bei einer Messzeit von 1 Minute etwa 68% der gemessenen Steigungen im Bereich $[2,87, 5,26]$ (1σ) und ca. 95% im Intervall $[1,67, 6,45]$ (2σ) liegen. Analog liegen bei einer Messzeit von 3 Minuten 68% im Intervall $[3,88, 5,26]$ und 95% im Intervall $[3,19, 5,95]$.

3.2 Auswertung der Messung bei großer Ereigniszahl

Wir beginnen, indem wir die abgespeicherten Daten aus Teil 3 in Python importieren und die Häufigkeit als Funktion der Anzahl der Zerfälle pro Zeiteinheit in ein Diagramm eintragen. Dazu fitten wir an die Messwerte sowohl eine Poisson- als auch eine Gauß-Verteilung. Der resultierende Plot ist in Abbildung 5 dargestellt. Die Parameter der Verteilungen sowie die χ^2 -Summen der Fits und Fit-Wahrscheinlichkeiten sind in Tabelle 8 dargestellt. Der Parameter α steht hierbei jeweils für die Amplitude der Verteilungen.

	Poisson	Gauß
$\alpha [10^3]$	$1,99 \pm 0,05$	$1,99 \pm 0,05$
μ	$72,18 \pm 0,22$	$72,00 \pm 0,22$
σ	-	$8,40 \pm 0,21$
χ^2	23,55	24,56
χ^2_{red}	0,71	0,77
Wsk.	89,0	82,0

Tabelle 8: A3 - Fitparameter und Fitbewertung

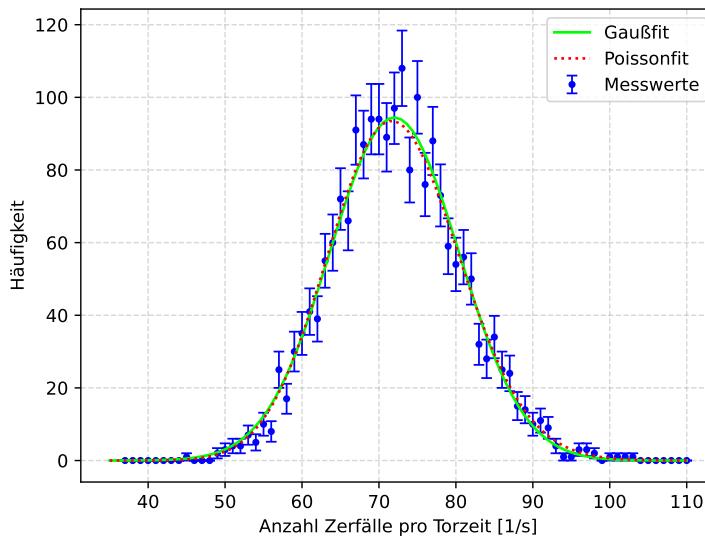


Abbildung 5: A3 - Messung bei großer Ereigniszahl

Sowohl am Plot als auch an den Fitparametern sowie der quantitativen Fitbewertung ist gut zu erkennen, dass beide Verteilungen innerhalb der signifikanten Stellen fast dieselben Ergebnisse liefern und ähnlich gut zu den Messwerten passen, was anhand der reduzierten χ^2 -Summe und der Fit-Wahrscheinlichkeit zu erkennen ist. Somit bestätigt sich, dass bei Zählstatistiken und großer Ereigniszahl die Poisson-Verteilung zur Gauß-Verteilung übergeht.

3.3 Auswertung der Messung bei kleiner Ereigniszahl

Wir wiederholen die soeben gemachten Schritte für die Messdaten aus Versuchsteil 4. Wir erhalten das in Abbildung dargestellte Diagramm und die in Tabelle notierten Parameter und Fitbewertungen.

	Poisson	Gauß
$\alpha [10^3]$	$4,99 \pm 0,07$	$4,94 \pm 0,07$
μ	$4,62 \pm 0,03$	$4,54 \pm 0,03$
σ	-	$2,108 \pm 0,025$
χ^2	7,17	84,42
χ^2_{red}	0,80	9,38
Wsk.	62,0	0,0

Tabelle 9: A4 - Fitparameter und Fitbewertung

Diesmal ist deutlich zu erkennen, dass die Poisson-Verteilung deutlich besser zu den Messwerten passt als die Gauß-Verteilung. Dies ist sowohl am reduzierten χ^2 -Wert als auch an der Fit-Wahrscheinlichkeit des Gauß-Fits zu erkennen. Ebenso liegt der Mittelwert der Poisson-Verteilung näher an dem in Tabelle 4 notierten Mittelwert. Somit lässt sich bestätigen, dass für geringe Ereigniszahlen die asymmetrische Poisson-Verteilung akkuratere Ergebnisse liefert als die Gauß-Verteilung.

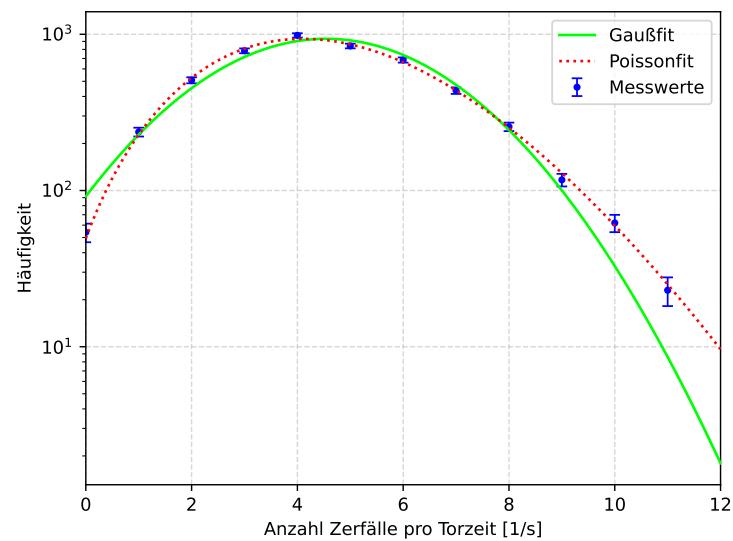


Abbildung 6: A4 - Messung bei kleiner Ereigniszahl

4 Zusammenfassung der Endergebnisse

In diesem Versuch wurde der statistische Prozess des radioaktiven Zerfalls eines ^{60}Co -Präparats untersucht. Dazu analysierten wir im ersten Versuchsteil die Charakteristik des verwendeten Geiger-Müller-Zählrohrs und bestimmten die prozentualen Steigungen des Plateau-Bereichs bei den Messzeiten 1 & 3 Minuten:

$$\begin{aligned} A_1 &= (4,1 \pm 1,2)\% \\ A_3 &= (4,6 \pm 0,7)\% \end{aligned} \tag{11}$$

Diese Steigungen weichen beide signifikant von 0 ab, was bestätigt, dass bei realistischen Zählrohren ein nicht zu vernachlässigender Anstieg im Plateau-Bereich bleibt.

Daraufhin bestimmten wir die benötigen Gesamtmesszeiten, um den Plateau-Anstieg auf 1%-ige Genauigkeit zu bestimmen. Dabei erhielten wir für Einzelmessungen von 1 Minute eine Gesamtzeit von 14,39 Stunden und bei Einzelmessungen von 3 Minuten eine Zeit von 11,43 Stunden. Zuletzt wurden noch die Konfidenzintervalle der Steigungen mit den zugehörigen Wahrscheinlichkeiten einer Messung identifiziert.

Anschließend verglichen wir die Poisson- und Gauß-Verteilung bei der Messung mit hoher Ereigniszahl. Hier konnte wie erwartet beobachtet werden, dass die beiden Verteilung fast analoge Ergebnisse liefern und die Poisson-Verteilung in die Gauß-Verteilung übergeht.

Zum Vergleich analysierten wir im letzten Schritt die Messung bei niedriger Ereigniszahl und konnten beobachten, wie die asymmetrische Poisson-Verteilung hier deutlich besser zu den Messwerten passte als die von Gauß.

5 Diskussion

Insgesamt konnten in diesem Versuch die gewünschten Ergebnisse und Beobachtungen erzielt werden, was an den positiven Ergebnissen in der Zusammenfassung zu sehen ist. Bei allen Versuchsteilen wurden die gewünschten Resultate erreicht und es konnten zur Theorie passende Schlüsse gezogen werden. Insbesondere der Vergleich der Gauß- und Poisson-Verteilungen in den letzten beiden Versuchsteilen lieferte aussagekräftige Ergebnisse über die Genauigkeit und Anwendbarkeit beider Verteilungen unter verschiedenen Randbedingungen.

Ebenso konnten bei der Analyse des Plateau-Bereichs interessante Ergebnisse erzielt werden. Insbesondere der Teil über die benötigte Gesamtmessdauer zeigt, dass akkurate Ergebnisse einen enorm viel größeren Zeitaufwand vorausgesetzt und somit den Rahmen des Physikalischen Anfängerpraktikums mehr als gesprengt hätten.

Somit lässt sich abschließend sagen, dass in diesem Versuch durchweg positive und dem Rahmen des Praktikums angemessene Ergebnisse erzielt werden konnten, die als lehrreicher und informativer Einblick in statistische Prozesse dienen.

July 31, 2024

```
[ ]: %matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.mlab as mlab
from scipy.optimize import curve_fit
from scipy.stats import chi2
from scipy.stats import norm
import scipy.constants as scp
from scipy.integrate import quad
from tabulate import tabulate
from scipy import signal
import scipy.constants as const
from scipy.special import gamma

[ ]: def sigma(x, y, dx, dy, label):
    s = np.abs(x-y)/np.sqrt(dx**2 + dy**2)
    print('Sigmaabweichung {} ='.format(str(label)), s)
    return s
```

1 Importieren und grafische Darstellung der Messdaten

```
[ ]: Ue = 460 #V
dUe = 10 #V

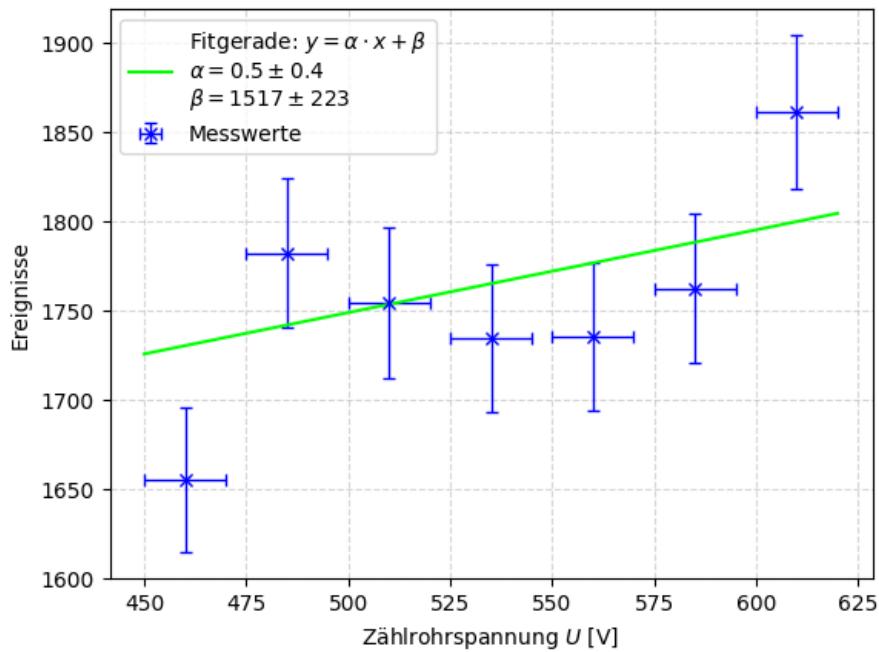
a1_U = Ue + np.array([0,25,50,75,100,125,150])
a1_dU = np.full(7, 10)
a1_N = np.array([1655,1782,1754,1734,1735,1762,1861])
a1_dN = np.sqrt(a1_N)

[ ]: def linfit(x,a,b):
    return a*x+b

[ ]: a1_pop, a1_cov = curve_fit(linfit, a1_U[1:], a1_N[1:], sigma=a1_dN[1:],  
    absolute_sigma=True)
```

```
[ ]: X = np.linspace(450,620,100)

plt.figure()
plt.grid(alpha=0.5, linestyle='--')
plt.errorbar(a1_U, a1_N, yerr=a1_dN, xerr=a1_dU, fmt='x', color='blue', u
             clabel='Messwerte', capsize=3, lw=1)
plt.xlabel(r'Zählrohrspannung $U$ [V]')
plt.ylabel(r'Ereignisse')
plt.plot(X, linfit(X, *a1_pop), color='lime',
          label="\n".join([r"Fitgerade: $y = \alpha \cdot x + \beta$",
                          r'$\alpha = {:.1f} \pm {:.1f}$'.format(a1_pop[0], np.
                          sqrt(a1_cov[0][0])),
                          r'$\beta = {:.0f} \pm {:.0f}$'.format(a1_pop[1], np.
                          sqrt(a1_cov[1][1]))]))
plt.legend()
plt.savefig('./plots/Plateau.pdf', format='PDF')
```



2 Plateaubereich des Zählrohrs

```
[ ]: U0 = 535 #V
dU0 = 10 #V

N1_0 = 14328
dN1_0 = np.sqrt(N1_0)

N3_0 = 42853
dN3_0 = np.sqrt(N3_0)

N1_1 = 14910
dN1_1 = np.sqrt(N1_1)

N3_1 = 44811
dN3_1 = np.sqrt(N3_1)

[ ]: ratio1 = N1_1 - N1_0
dr1 = np.sqrt(dN1_1**2 + dN1_0**2)

ratio3 = N3_1 - N3_0
dr3 = np.sqrt(dN3_1**2 + dN3_0**2)

perc1 = ratio1/N1_0 * 100 #Prozent
dp1 = perc1 * np.sqrt((dr1/ratio1)**2 + (dN1_0/N1_0)**2)

perc3 = ratio3/N3_0 * 100
dp3 = perc3 * np.sqrt((dr3/ratio3)**2 + (dN3_0/N3_0)**2)

print("Prozentualer Anstieg 1min = {:.2f} +/- {:.2f}".format(perc1, dp1))
print("Prozentualer Anstieg 3min = {:.2f} +/- {:.2f}".format(perc3, dp3))

Prozentualer Anstieg 1min = (4.061976549413735 +/- 1.193888486093682)
Prozentualer Anstieg 3min = (4.569108347140224 +/- 0.691275080500964)
```

2.1 Signifikanz der gemessenen Anstiege

```
[ ]: _ = sigma(perc1, 0, dp1, 0, 'Prozentualer Anstieg 1min')
_ = sigma(perc3, 0, dp3, 0, 'Prozentualer Anstieg 3min')

Sigmaabweichung Prozentualer Anstieg 1min = 3.402308169253087
Sigmaabweichung Prozentualer Anstieg 3min = 6.609681841603507
```

2.2 Benötigte Messzeit für 1%-Fehler

```
[ ]: err1 = dr1/ratio1
      err3 = dr3/ratio3

      print("rel. Fehler 1min = {}".format(err1))
      print("rel. Fehler 3min = {}".format(err3))

rel. Fehler 1min = 0.2937993605587172
rel. Fehler 3min = 0.1512160725123355

[ ]: tges1 = (err1 /0.01)**2 * (1 * 60)
      tges3 = (err3 /0.01)**2 * (3 * 60)

      print("ges. Messdauer 1min = {}s bzw. {}h".format(tges1, tges1/3600))
      print("ges. Messdauer 3min = {}s bzw. {}h".format(tges3, tges3/3600))

ges. Messdauer 1min = 51790.83855882667s bzw. 14.38634404411852h
ges. Messdauer 3min = 41159.34105490062s bzw. 11.43315029302795h
```

2.3 1- und 2- σ -Umgebung unserer Ergebnisse

```
[ ]: print(r'1- -Umgebung vom Prozentualen Anstieg bei 1min: [{}, {}]'.format(np.
      round(perc1-dp1, 2), np.round(perc1+dp1,2)))
      print(r'1- -Umgebung vom Prozentualen Anstieg bei 3min: [{}, {}]'.format(np.
      round(perc3-dp3, 2), np.round(perc3+dp3, 2)))

1- -Umgebung vom Prozentualen Anstieg bei 1min: [2.87, 5.26]
1- -Umgebung vom Prozentualen Anstieg bei 3min: [3.88, 5.26]

[ ]: print(r'2- -Umgebung vom Prozentualen Anstieg bei 1min: [{}, {}]'.format(np.
      round(perc1-2*dp1, 2), np.round(perc1+2*dp1, 2)))
      print(r'2- -Umgebung vom Prozentualen Anstieg bei 3min: [{}, {}]'.format(np.
      round(perc3-2*dp3, 2), np.round(perc3+2*dp3, 2)))

2- -Umgebung vom Prozentualen Anstieg bei 1min: [1.67, 6.45]
2- -Umgebung vom Prozentualen Anstieg bei 3min: [3.19, 5.95]
```

3 Auswertung der Daten mit hoher mittlerer Ereigniszahl

```
[ ]: a3_ZpT, a3_N = np.loadtxt('./data/data_aufgabe2.txt', unpack=True, skiprows=4, u
      delimiter=',')
      a3_dN = np.sqrt(a3_N)

[ ]: def gaussian(x, A, mu, sig):
      return A/(np.sqrt(2*np.pi)*sig) * np.exp(-(x-mu)**2/(2*sig**2))

def poisson(x, A, mu):
```

```

    return A * np.exp(-mu) * mu**x /gamma(x+1)

[ ]: a3_pop_g, a3_cov_g = curve_fit(gaussian, a3_ZpT[20:-19], a3_N[20:-19],  

    ↪sigma=a3_dN[20:-19], absolute_sigma=True, p0=[2000,75,8])

[ ]: a3_pop_p, a3_cov_p = curve_fit(poisson, a3_ZpT[20:-19], a3_N[20:-19],  

    ↪sigma=a3_dN[20:-19], absolute_sigma=True, p0=[2000,75])

[ ]: X = np.linspace(35,110,100)

plt.figure()  

plt.grid(alpha=0.5, linestyle='--')  

plt.errorbar(a3_ZpT, a3_N, yerr=a3_dN, fmt='.', color='blue',  

    ↪label='Messwerte', capsize=3, lw=1)  

plt.ylabel(r'Häufigkeit')  

plt.xlabel(r'Anzahl Zerfälle pro Torzeit [1/s]')  

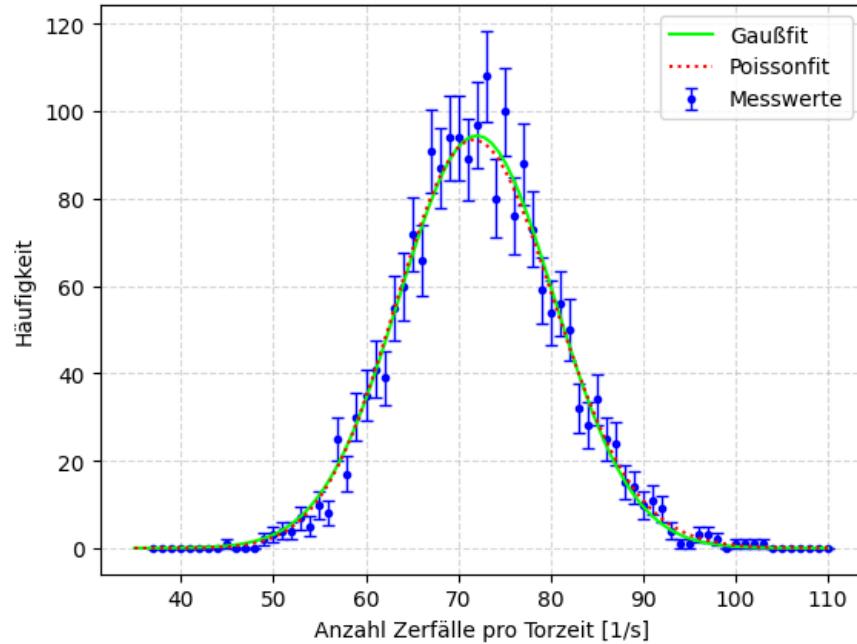
plt.plot(X, gaussian(X, *a3_pop_g), color='lime', label='Gaußfit', zorder=10)  

plt.plot(X, poisson(X, *a3_pop_p), color='red', label='Poissonfit', zorder=11,  

    ↪linestyle=':')
plt.legend()  

plt.savefig('./plots/Ngroß.pdf', format='PDF')

```



```

[ ]: print("Gaussfit:")
print("A=",a3_pop_g[0], ", Standardfehler=", np.sqrt(a3_cov_g[0][0]))
print("mu=",a3_pop_g[1], ", Standardfehler=", np.sqrt(a3_cov_g[1][1]))
print("sig=",a3_pop_g[2], ", Standardfehler=", np.sqrt(a3_cov_g[2][2]))
print("Poissonfit:")
print("A_p=",a3_pop_p[0], ", Standardfehler=", np.sqrt(a3_cov_p[0][0]))
print("mu_p=",a3_pop_p[1], ", Standardfehler=", np.sqrt(a3_cov_p[1][1]))

Gaussfit:
A= 1989.1104720815874 , Standardfehler= 46.84464991076517
mu= 72.00187445783394 , Standardfehler= 0.21915054667311396
sig= 8.402549333933674 , Standardfehler= 0.20830114508438774
Poissonfit:
A_p= 1990.3250570061998 , Standardfehler= 45.61724901438288
mu_p= 72.18173037291598 , Standardfehler= 0.21683249218455716

[ ]: #Gauss:
chi2_g=np.sum((gaussian(a3_ZpT[20:-19],*a3_pop_g)-a3_N[20:-19])**2/a3_dN[20:
    :-19]**2)
dof_g=len(a3_ZpT[20:-19])-3 #dof: degrees of freedom, Freiheitsgrad
chi2_red_g=chi2_g/dof_g
print("chi2_g=", chi2_g)
print("chi2_red_g=",chi2_red_g)

#Poisson:
chi2_p=np.sum((poisson(a3_ZpT[20:-19],*a3_pop_p)-a3_N[20:-19])**2/a3_dN[20:
    :-19]**2)
dof_p=len(a3_ZpT[20:-19])-2 #poisson hat nur 2 Parameter
chi2_red_p=chi2_p/dof_p
print("chi2_p=", chi2_p)
print("chi2_red_p=",chi2_red_p)

#Gauss:
prob_g=round(1-chi2.cdf(chi2_g,dof_g),2)*100
#Poisson:
prob_p=round(1-chi2.cdf(chi2_p,dof_p),2)*100
print("Wahrscheinlichkeit Gauss=", prob_g,"%")
print("Wahrscheinlichkeit Poisson=", prob_p,"%")

chi2_g= 24.559363909346516
chi2_red_g= 0.7674801221670786
chi2_p= 23.54616897104148
chi2_red_p= 0.7135202718497419
Wahrscheinlichkeit Gauss= 82.0 %
Wahrscheinlichkeit Poisson= 89.0 %

```

4 Auswertung der Daten mit kleiner Ereigniszahl

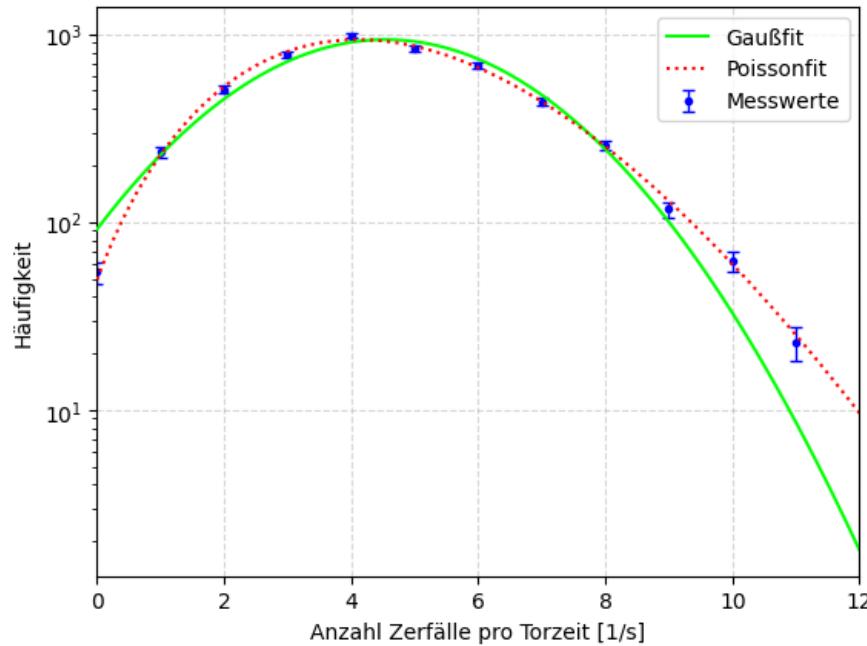
```
[ ]: a4_ZpT, a4_N = np.loadtxt('./data/data_aufgabe3.txt', unpack=True, skiprows=4, delimiter=',')
      a4_dN = np.sqrt(a4_N)

[ ]: a4_pop_g, a4_cov_g = curve_fit(gaussian, a4_ZpT[:12], a4_N[:12], sigma=a4_dN[:12], absolute_sigma=True, p0=[5000,4,2])

[ ]: a4_pop_p, a4_cov_p = curve_fit(poisson, a4_ZpT[1:12], a4_N[1:12], sigma=a4_dN[1:12], absolute_sigma=True, p0=[5000,4.5])

[ ]: X = np.linspace(0,12,100)

plt.figure()
plt.xlim((0,12))
# plt.ylim((0,1100)
plt.grid(alpha=0.5, linestyle='--')
plt.errorbar(a4_ZpT[:12], a4_N[:12], yerr=a4_dN[:12], fmt='.', color='blue', label='Messwerte', capsizes=3, lw=1)
plt.ylabel(r'Häufigkeit')
plt.xlabel(r'Anzahl Zerfälle pro Torzeit [1/s]')
plt.plot(X, gaussian(X, *a4_pop_g), color='lime', label='Gaußfit', zorder=10)
plt.plot(X, poisson(X, *a4_pop_p), color='red', label='Poissonfit', zorder=11, linestyle=':')
plt.legend()
plt.yscale('log')
plt.savefig('./plots/Nklein.pdf', format='PDF')
```



```
[ ]: print("Gaussfit:")
print("A=",a4_pop_g[0], ", Standardfehler=", np.sqrt(a4_cov_g[0][0]))
print("mu=",a4_pop_g[1], ", Standardfehler=", np.sqrt(a4_cov_g[1][1]))
print("sig=",a4_pop_g[2], ", Standardfehler=", np.sqrt(a4_cov_g[2][2]))
print("Poissonfit:")
print("A_p=",a4_pop_p[0], ", Standardfehler=", np.sqrt(a4_cov_p[0][0]))
print("mu_p=",a4_pop_p[1], ", Standardfehler=", np.sqrt(a4_cov_p[1][1]))
```

Gaussfit:
A= 4944.612364128406 , Standardfehler= 70.70423984237354
mu= 4.543426892360781 , Standardfehler= 0.03109527591610403
sig= 2.1080511178725807 , Standardfehler= 0.02454710035391609
Poissonfit:
A_p= 4990.90460929385 , Standardfehler= 71.10849982627033
mu_p= 4.615267169637306 , Standardfehler= 0.031592041453627914

```
[ ]: #Gauss:
chi2_g=np.sum((gaussian(a4_ZpT[:12],*a4_pop_g)-a4_N[:12])**2/a4_dN[:12]**2)
dof_g=len(a4_ZpT[:12])-3 #dof: degrees of freedom, Freiheitsgrad
chi2_red_g=chi2_g/dof_g
print("chi2_g=", chi2_g)
```

```

print("chi2_red_g=",chi2_red_g)

#Poisson:
chi2_p=np.sum((poisson(a4_ZpT[1:12],*a4_pop_p)-a4_N[1:12])**2/a4_dN[1:12]**2)
dof_p=len(a4_ZpT[1:12])-2 #poisson hat nur 2 Parameter
chi2_red_p=chi2_p/dof_p
print("chi2_p=", chi2_p)
print("chi2_red_p=",chi2_red_p)

#Gauss:
prob_g=round(1-chi2.cdf(chi2_g,dof_g),2)*100
#Poisson:
prob_p=round(1-chi2.cdf(chi2_p,dof_p),2)*100
print("Wahrscheinlichkeit Gauss=", prob_g, "%")
print("Wahrscheinlichkeit Poisson=", prob_p, "%")

chi2_g= 84.42090815486779
chi2_red_g= 9.380100906096422
chi2_p= 7.168593420692852
chi2_red_p= 0.7965103800769835
Wahrscheinlichkeit Gauss= 0.0 %
Wahrscheinlichkeit Poisson= 62.0 %

```