

# Auswertung

June 25, 2024

## 1 Auswertung B3.1

```
[1]: using CSV
      using DataFrames
      using Plots
      using LaTeXStrings
      using LsqFit
      using Measurements
      using Statistics
```

### 1.1 Rechnungen zur Vorbereitung

```
[2]: T_12 = 30.08 * 365.25 * 24 * 60 * 60 # s
```

```
[2]: 9.492526079999999e8
```

```
[3]: log(2) / T_12
```

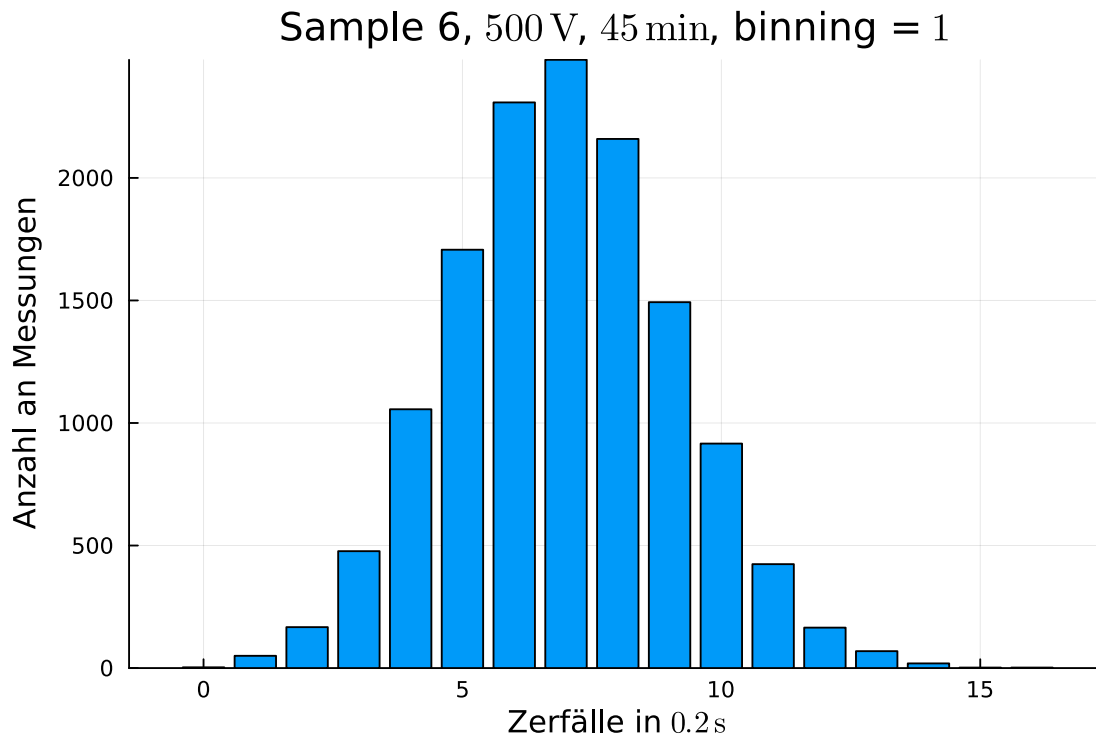
```
[3]: 7.302030826339804e-10
```

### 1.2 0: Rohe Messdaten

Sample 6, 500 V, 45min:

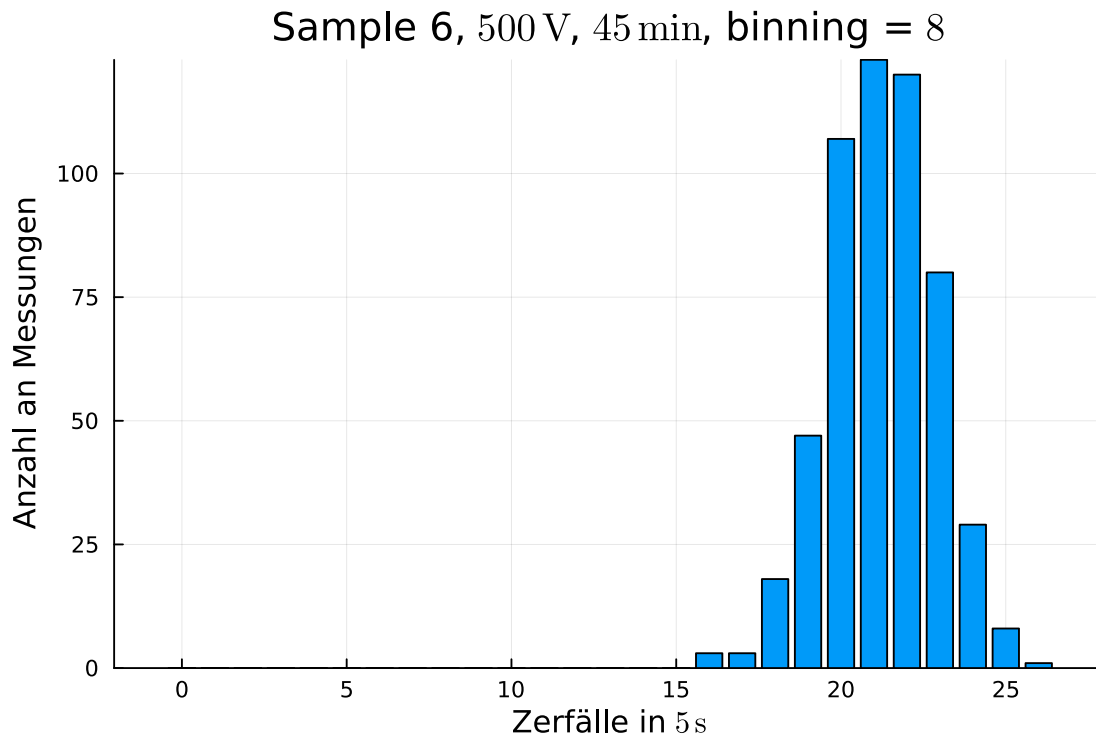
```
[4]: # ΔT = 0.2, binning = 1
      zerfälle_6_500_1 = [0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16]
      anzahl_6_500_1 = [
        ↪ [3,50,167,477,1056,1707,2308,2482,2159,1493,916,424,165,69,19,2,2]
      ]
      plot(bar(zerfälle_6_500_1,anzahl_6_500_1,label=""), title=L"Sample 6, ↪
        ↪ $500\mathrm{\,V}$, $45\mathrm{\,min}$, binning = $1$")
      xlabel!(L"Zerfälle in $0.2\mathrm{\,s}$")
      ylabel!(L"Anzahl an Messungen")
```

```
[4]:
```



```
[5]: # ΔT = 5s, binning = 8
zerfälle_6_500_2 = □
↳ [0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26]
anzahl_6_500_2 = □
↳ [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,3,18,47,107,123,120,80,29,8,1]
plot(bar(zerfälle_6_500_2,anzahl_6_500_2,label=""), title=L"Sample 6, □
↳ $500\mathrm{\,V}$, $45\mathrm{\,min}$, binning = $8$")
xlabel!(L"Zerfälle in $5\mathrm{\,s}$")
ylabel!("Anzahl an Messungen")
```

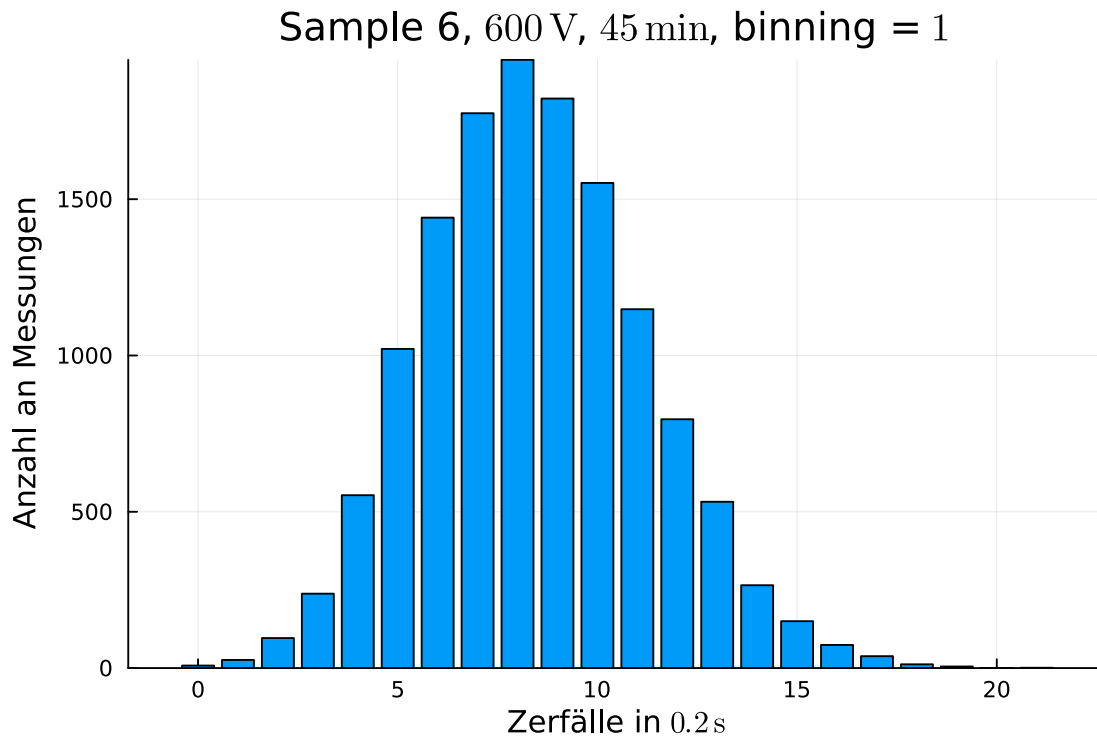
[5]:



Sample 6, 600V, 45min:

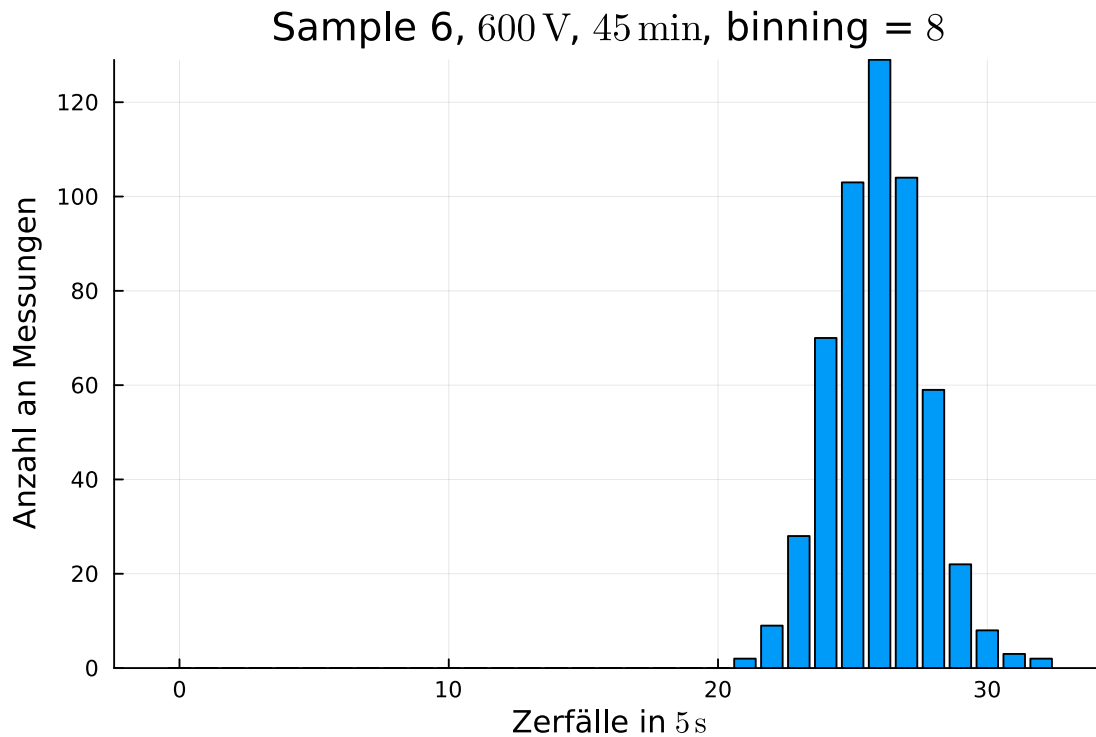
```
[6]: # ΔT = 0.2s, binning = 1
zerfälle_6_600_1 = [0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21]
anzahl_6_600_1 = [
    ↪ [8,26,96,238,553,1021,1441,1775,1946,1822,1552,1148,796,532,265,150,74,38,12,5,0,1]
]
plot(bar(zerfälle_6_600_1,anzahl_6_600_1,label=""), title=L"Sample 6,↪
    ↪ $600\mathrm{\,V}$, $45\mathrm{\,min}$, binning = $1$")
xlabel!(L"Zerfälle in $0.2\mathrm{\,s}$")
ylabel!("Anzahl an Messungen")
```

[6]:



```
[7]: # ΔT = 5s, binning = 8
zerfälle_6_600_2 = ␣
    ↪ [0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32]
anzahl_6_600_2 = ␣
    ↪ [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,9,28,70,103,129,104,59,22,8,3,2]
plot(bar(zerfälle_6_600_2,anzahl_6_600_2,label=""), title=L"Sample 6,␣
    ↪ $600\mathrm{\,V}$, $45\mathrm{\,min}$, binning = $8$")
xlabel!(L"Zerfälle in $5\mathrm{\,s}$")
ylabel!("Anzahl an Messungen")
```

[7]:



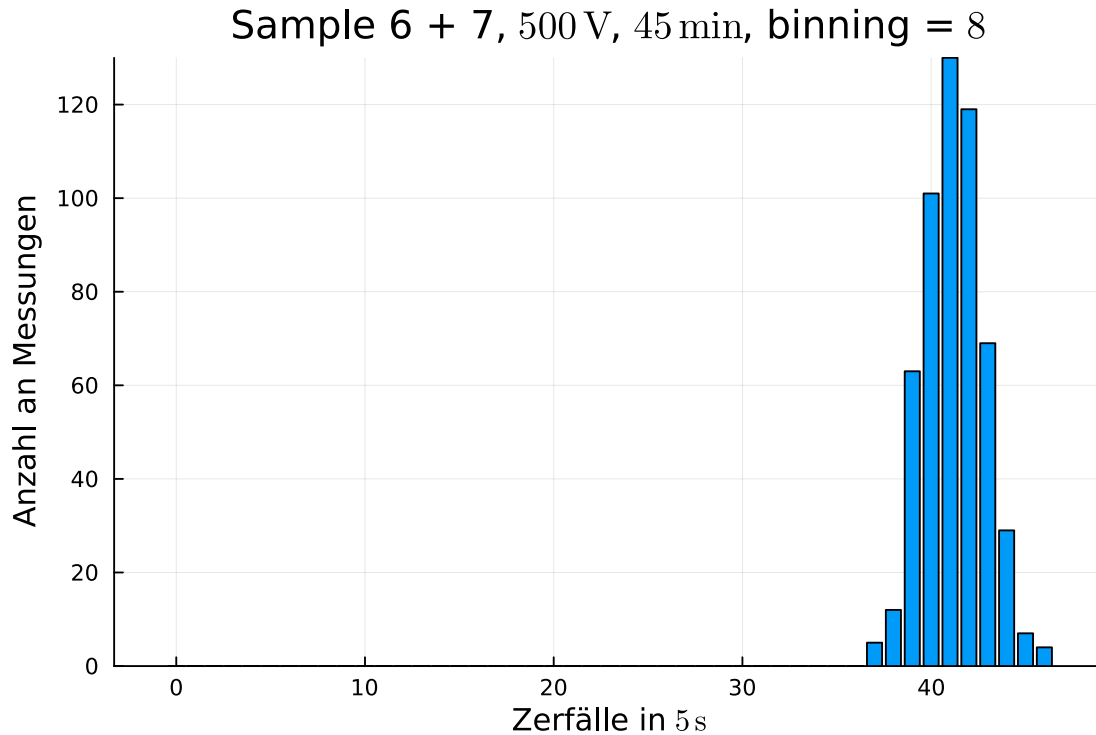
Sample 6+7, 500V, 45min:

```
[8]: # ΔT = 0.2s, binning = 1
zerfälle_6und7_500_1 = ␣
    ↪ [0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23]
anzahl_6und7_500_1 = ␣
    ↪ [0,0,0,0,1,5,26,78,227,443,875,1373,1917,2175,2140,1712,1193,768,375,139,38,11,2,1]
plot(bar(zerfälle_6und7_500_1,anzahl_6und7_500_1,label=""), title=L"Samples 6 +  

    ↪ 7, $500\mathrm{\,V}$, $45\mathrm{\,min}$, binning = $1$")
xlabel!(L"Zerfälle in $0.2\mathrm{\,s}$")
ylabel!("Anzahl an Messungen")
```

[8]:





### 1.3 1: Poisson-Verteilung

```
[10]: mittel(anzahl, zerfälle) = sum(anzahl .* zerfälle)/sum(anzahl)
      Poisson(n, N, lambda) = N * lambda^big(n) * exp(-lambda) / factorial(big(n))

      # runde auf 3 Nachkommastellen ohne "0" am Ende zu verschlucken
      rounded_string(value) = rpad(round(value, digits=3),
      ↪length(string(round(value))) + 2 , "0")
```

```
[10]: rounded_string (generic function with 1 method)
```

```
[11]: function plotte_poisson(anzahl, zerfälle; new_plot=true)
      # Passende Poissonverteilung:
      lambda = mittel(anzahl, zerfälle)
      lambda_rounded = rounded_string(lambda)
      P(n) = Poisson(n, sum(anzahl), lambda)

      if new_plot
          plot()
          bar!(zerfälle,anzahl,label=L"\mathrm{Messdaten}")
      end
```

```

bar!(0:25,P,label=LaTeXString("\$\\mathrm{Poisson}\\ mit\\ } \\lambda =\r\n
↪$\\lambda_{rounded}$"),alpha=0.5)
xlabel!(L"\mathrm{Anzahl\\ der\\ Zerf\u00e4lle\\ in\\ }0.2\mathrm{\\,s}")
ylabel!(L"\mathrm{Anzahl\\ der\\ Messungen}")
end

```

[11]: `plotte_poisson` (generic function with 1 method)

**Sample 6, 500V, 45min,  $\Delta T = 0.2s$ , binning = 1**

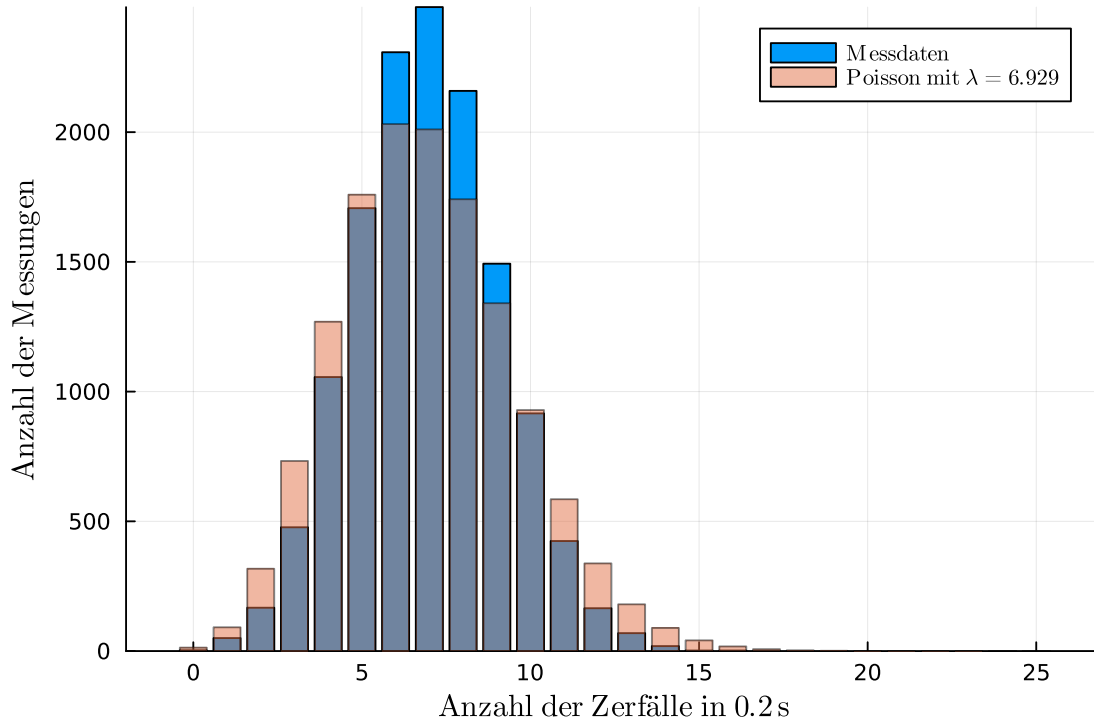
```

[12]: # Messdaten mit  $\Delta T = 0.2$ , binning = 1
zerf\u00e4lle_6_500_1 = [0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16]
anzahl_6_500_1 =\r\n
↪[3,50,167,477,1056,1707,2308,2482,2159,1493,916,424,165,69,19,2,2]

plotte_poisson(anzahl_6_500_1, zerf\u00e4lle_6_500_1)

```

[12]:



`savefig("../media/B3.1/poisson1.pdf");`

**Sample 6, 600V, 45min,  $\Delta T = 0.2s$ , binning = 1**

```

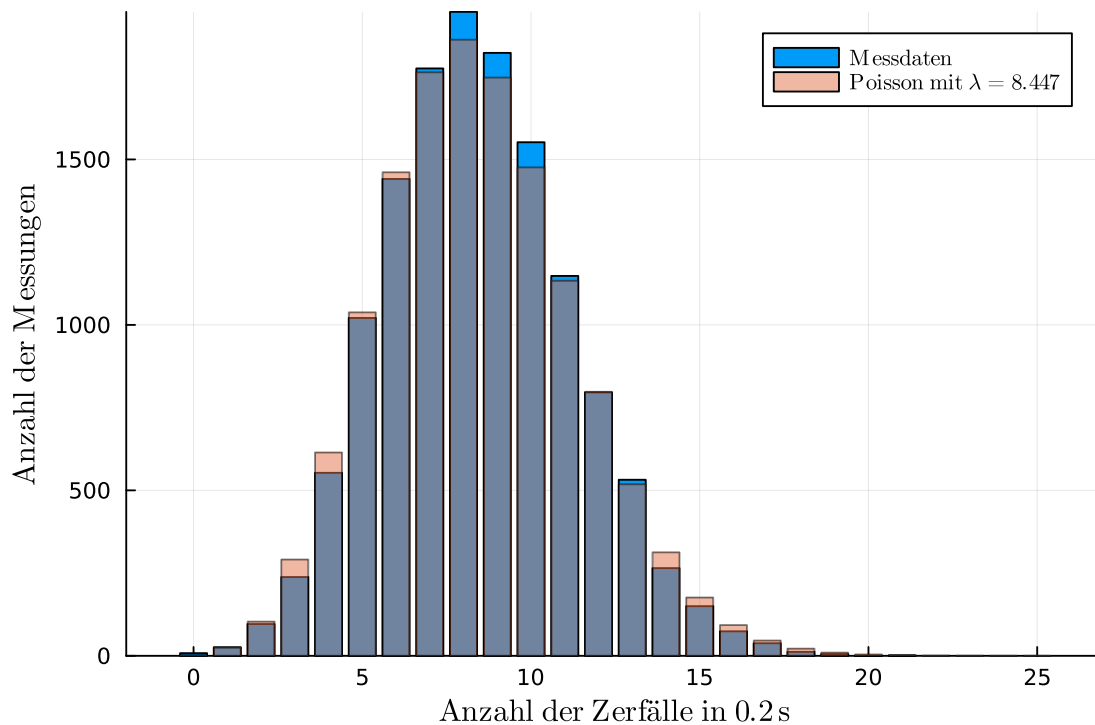
[13]: # Messdaten mit  $\Delta T = 0.2s$ , binning = 1
zerf\u00e4lle_6_600_1 = [0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21]
anzahl_6_600_1 =\r\n
↪[8,26,96,238,553,1021,1441,1775,1946,1822,1552,1148,796,532,265,150,74,38,12,5,0,1]

```



```
plotte_poisson(anzahl_6_600_1, zerfälle_6_600_1)
```

[13]:



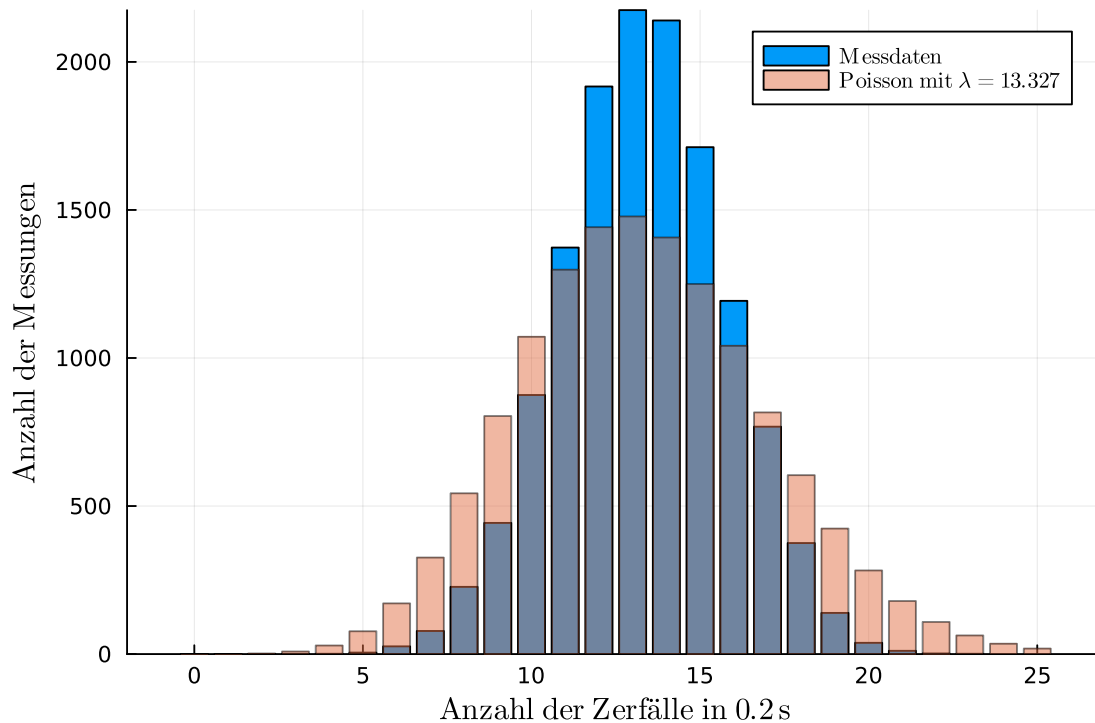
```
savefig("../media/B3.1/poisson2.pdf");
```

**Sample 6 + 7, 500V, 45min,  $\Delta T = 0.2s$ , binning = 1**

```
[14]: # Messdaten mit  $\Delta T = 0.2s$ , binning = 1
zerfälle_6und7_500_1 = ␣
    ↪ [0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23]
anzahl_6und7_500_1 = ␣
    ↪ [0,0,0,0,1,5,26,78,227,443,875,1373,1917,2175,2140,1712,1193,768,375,139,38,11,2,1]

plotte_poisson(anzahl_6und7_500_1, zerfälle_6und7_500_1)
```

[14]:

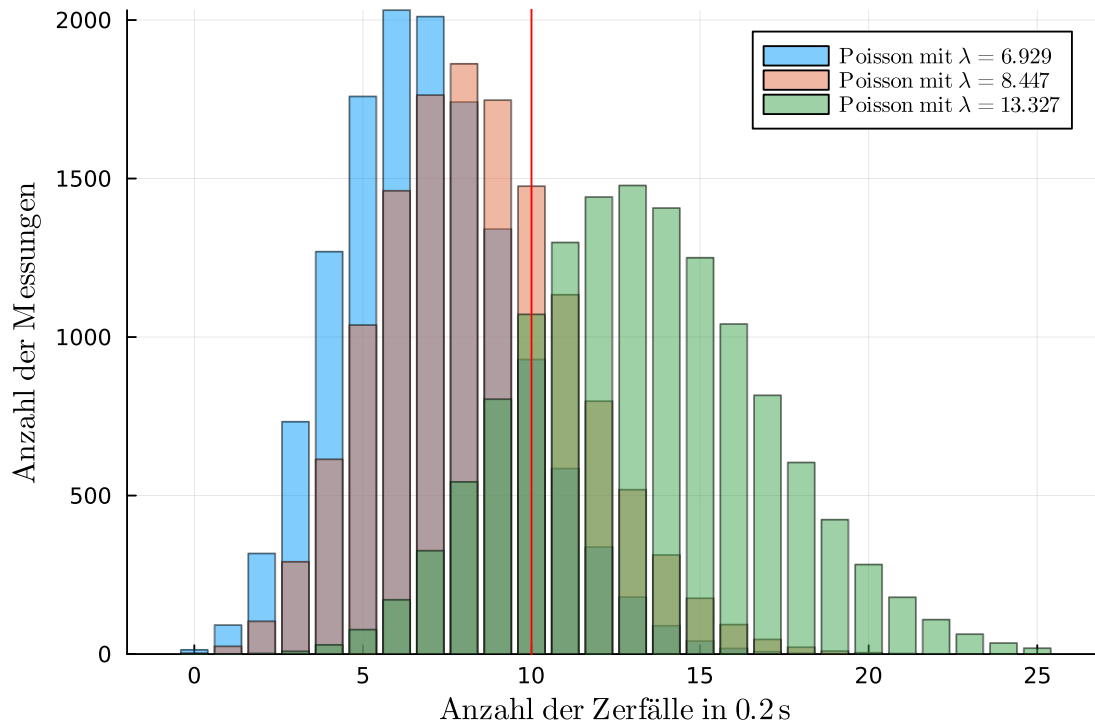


```
savefig("../media/B3.1/poisson3.pdf");
```

Alle Poisson-Verteilungen zusammen:

```
[15]: plot()
      plote_poisson(anzahl_6_500_1, zerfälle_6_500_1, new_plot=false)
      plote_poisson(anzahl_6_600_1, zerfälle_6_600_1, new_plot=false)
      plote_poisson(anzahl_6und7_500_1, zerfälle_6und7_500_1, new_plot=false)
      vline!([10], color=:red, label="")
```

[15]:



savefig("../media/B3.1/allePoisson.pdf");

## 1.4 2: Gaußverteilung

```
[16]: n = 8 # Binning (für alle Messungen gleich)
      Δt = 5 # (für alle Messungen gleich)

      G(x, m, F, n=8) = 1/sqrt(2 * pi * m) * F * sqrt(n) * exp(- (x - m)^2 / (2 * m / n))
```

[16]: G (generic function with 2 methods)

Sample 6, 500V, 45min,  $\Delta T = 5s$ , binning = 8:

```
[17]: # Messdaten mit ΔT = 5s, binning = 8
      anzahl_6_500_2 = [2,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,3,18,47,107,123,120,80,29,8,1,0,0,0]

      # Schneide interessanten Bereich hereaus
      zerfälle_6_500_2 = 16:28
      anzahl_6_500_2 = anzahl_6_500_2[zerfälle_6_500_2]

      # Passende Gaußverteilung:
      z_strich = 11044/300 # Aus Kurzmessung
```

```

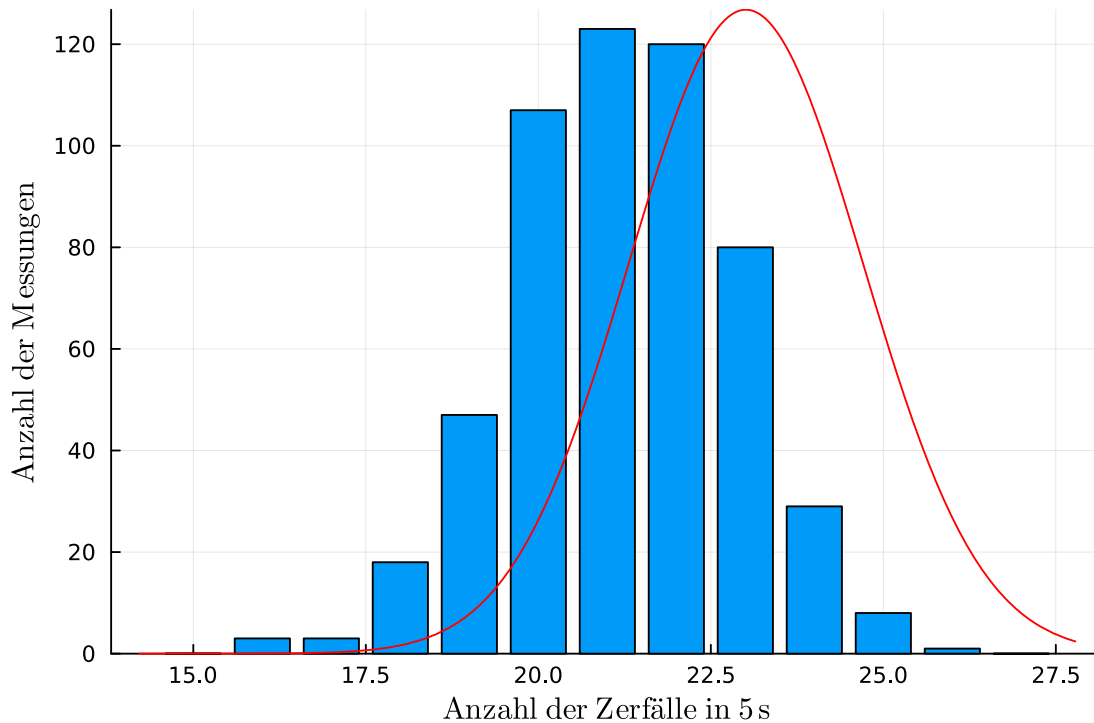
m = z_strich * Δt / n
F = sum(anzahl_6_500_2)
G(x) = G(x, m, F)

# Plot
gauß1 = plot(bar(zerfälle_6_500_2.
  ↪-1,anzahl_6_500_2,label=L"\mathrm{Messdaten}"), legend=:none)
plot!(G,color=:red,label=L"\mathrm{Gauß}")

xlabel!(L"\mathrm{Anzahl\ der\ Zerfälle\ in\ }5\mathrm{\,s}")
ylabel!(L"\mathrm{Anzahl\ der\ Messungen}")

```

[17]:



[18]: [m,F,z\_strich]

[18]: 3-element Vector{Float64}:

23.008333333333333

539.0

36.813333333333333

savefig(gauß1, "../../../../media/B3.1/gauss1.pdf");

Sample 6, 600V, 45min,  $\Delta T = 5s$ , binning = 8:

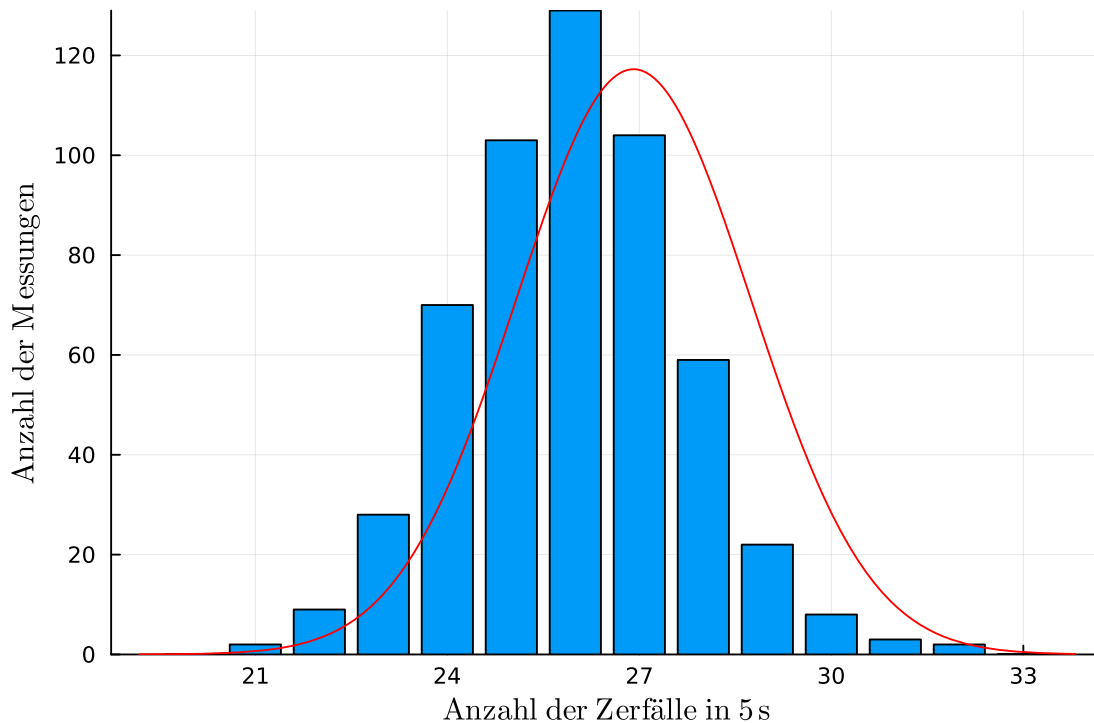
```
[19]: # Messdaten mit  $\Delta T = 5s$ , binning = 8
anzahl_6_600_2 =  $\leftarrow$ 
 $\hookrightarrow$  [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,9,28,70,103,129,104,59,22,8,3,2,0,0]

# Schneide interessanten Bereich hereaus
zerfalle_6_600_2 = 21:34
anzahl_6_600_2 = anzahl_6_600_2[zerfalle_6_600_2]

# Passende Gauverteilung:
z_strich = 12917/300 # Aus Kurzmessung
m = z_strich *  $\Delta t$  / n
F = sum(anzahl_6_600_2)
G(x) = G(x, m, F)

# Plot
gau2 = plot(bar(zerfalle_6_600_2.
 $\hookrightarrow$ -1,anzahl_6_600_2,label=L"\mathrm{Messdaten}"), legend=:none)
plot!(G,color=:red,label=L"\mathrm{Gau}")
xlabel!(L"\mathrm{Anzahl\ der\ Zerfalle\ in\ }5\mathrm{s}")
ylabel!(L"\mathrm{Anzahl\ der\ Messungen}")
```

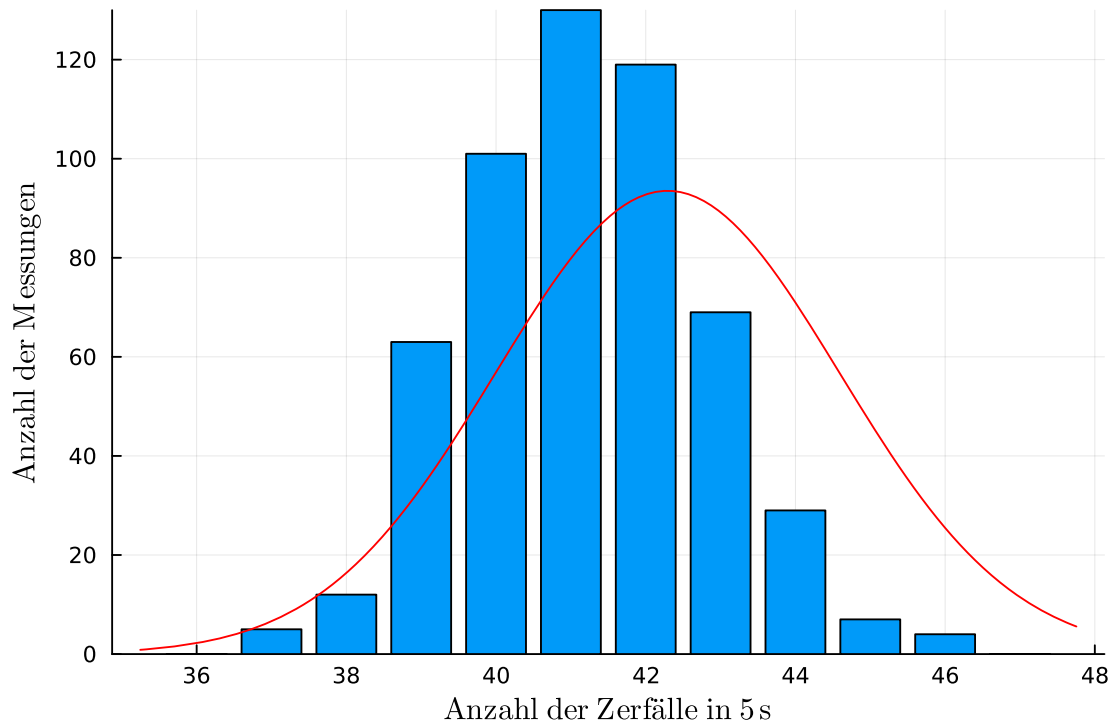
[19]:



[20]: [m,F,z\_strich]

Sample 6 + 7, 500V, 45min,  $\Delta T = 5$ s, binning = 8:

[21] :



```
[22]: [m,F,z_strich]
```

```
[22]: 3-element Vector{Float64}:
 42.29166666666667
 539.0
 67.66666666666667
```

```
savefig(gauß3, "../media/B3.1/gauss3.pdf");
```

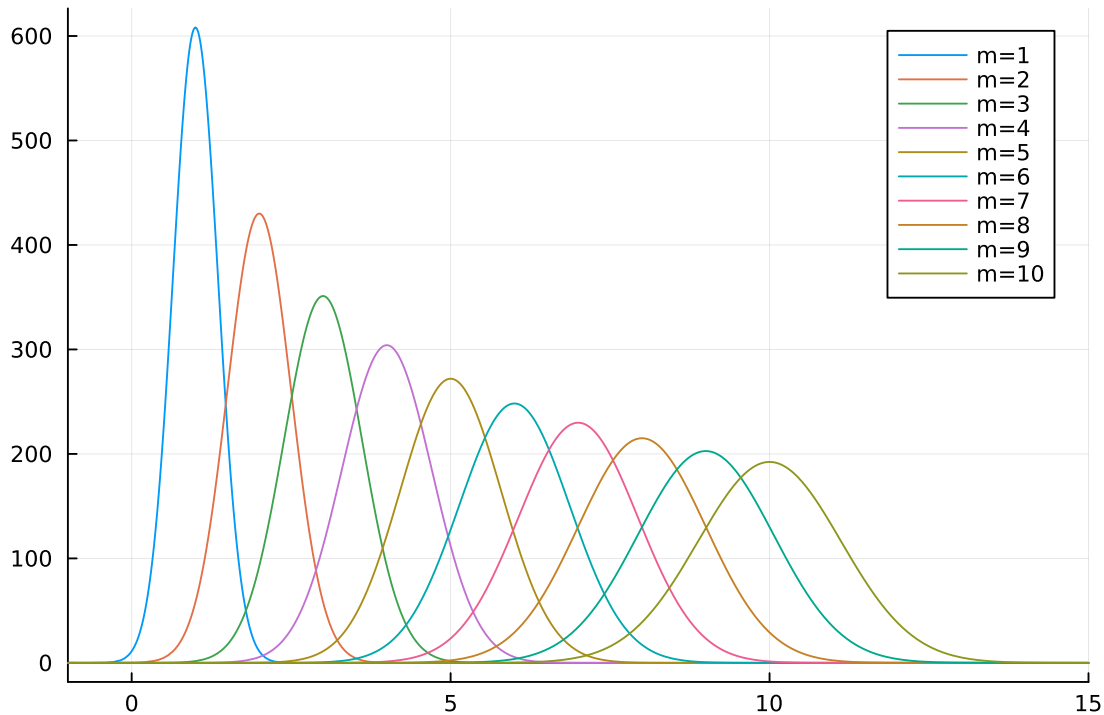
```
[23]: G(x) = G(x, m, F)
m = 1
plot(G, xaxis=[-1,15], label="m=$m")
m=2
plot!(G, label="m=$m")
m=3
plot!(G, label="m=$m")
m=4
plot!(G, label="m=$m")
m=5
plot!(G, label="m=$m")
m=6
plot!(G, label="m=$m")
m=7
plot!(G, label="m=$m")
```

```

m=8
plot!(G, label="m=$m")
m=9
plot!(G, label="m=$m")
m=10
plot!(G, label="m=$m")

```

[23]:



### 1.5 3: Intervall-Verteilung

#### Plotten der Messwerte

```

[24]: interval0 = CSV.read("sample_6/500V_45min/interval_0.001_1.csv", DataFrame)
interval1 = CSV.read("sample_6/500V_45min/interval_0.001_2.csv", DataFrame)
interval2 = CSV.read("sample_6/500V_45min/interval_0.001_3.csv", DataFrame);

```

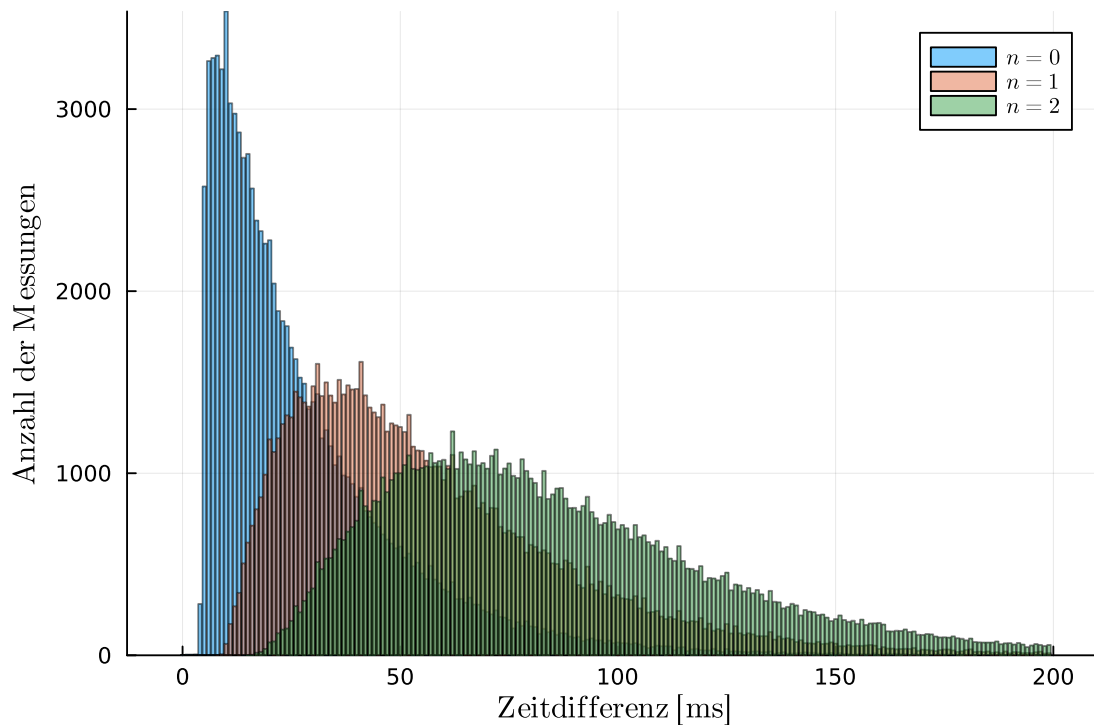
```

[25]: # Sample 6, 500V, 45min
interval = bar(interval0[1:200, :intervall], interval0[1:200, :count], alpha=0.
    ↪5, label=L"n = 0")
bar!(interval1[1:200, :intervall], interval1[1:200, :count], alpha=0.5,
    ↪label=L"n = 1")
bar!(interval2[1:200, :intervall], interval2[1:200, :count], alpha=0.5,
    ↪label=L"n = 2")
xlabel!(L"\mathrm{Zeitdifferenz\ } [\mathrm{ms}]")
ylabel!(L"\mathrm{Anzahl\ der\ Messungen}")

```



[25]:



```
savefig(interval, "../../../media/B3.1/interval.pdf");
```

### Fitten für n=0

- Alle Messwerte im Bereich  $t \in [0, \text{totzeit}]$  abschneiden und den Rest fitten

```
[26]: totzeitIndex = 11
modelFunction(t, a_get) = 1000*2700 * a_get[1]^2 * exp.(- a_get[1] * t) # a = _
    ↪ fit-parameter
a0 = [0.043]
fit = curve_fit(modelFunction, interval0[(totzeitIndex+1):100, :intervall], _
    ↪ interval0[(totzeitIndex+1):100, :count], a0)
a_fit = fit.param[1]
```

[26]: 0.043206773430525904

```
[27]: Δa_fit = sqrt(estimate_covar(fit)[1])
```

[27]: 0.00012693544086906926

```
[28]: a_strich = 11044/300 * 10^(-3) # ms-1 Gemessene Zählrate aus Kurzzeitmessung _
    ↪ (in ms-1), weil a_fit auch in ms-1 ist)
    = 1/a_strich - 1/a_fit # ms
```

[28]: 4.019551724727979

```
[29]: Δ = Δa_fit/a_fit^2
```

[29]: 0.06799535172900686

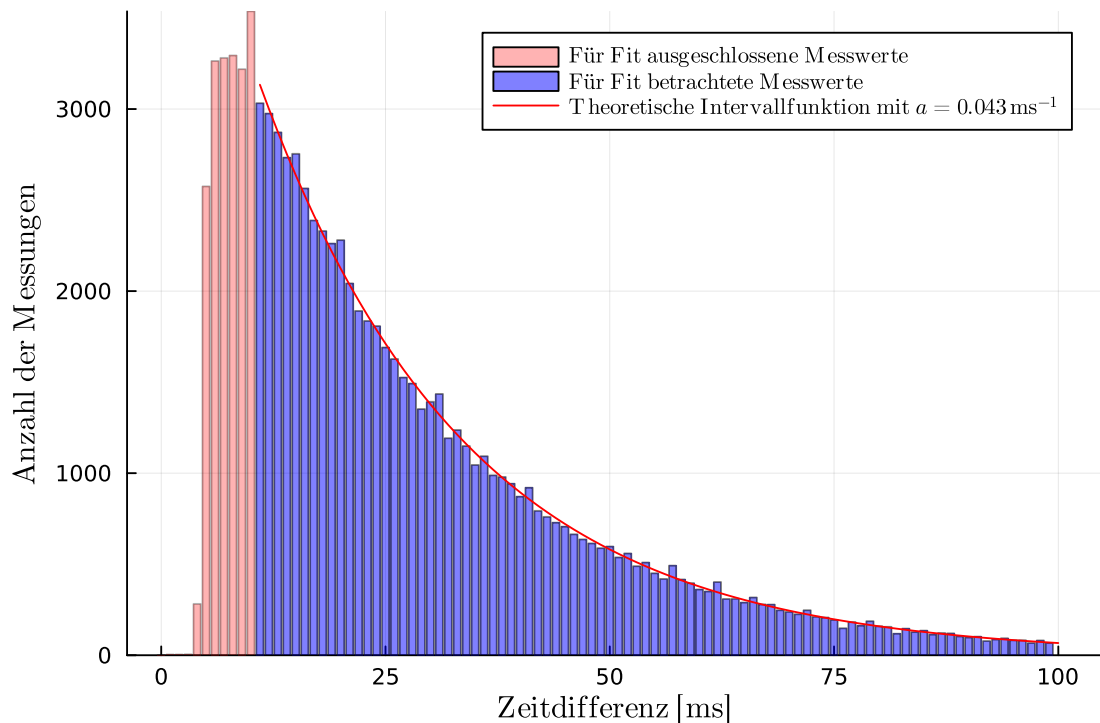
```
[30]: I(t) = modelFunction(t, [a_fit])

# Plot
intervalFit = bar(interval0[1:totzeitIndex, :intervall], interval0[1:
    ↳totzeitIndex, :count],
    color=:red, alpha=0.3, label=L"\mathrm{Für\ Fit\ ausgeschlossene\ }
    ↳Messwerte}", title="", legend=:topright)
bar!(interval0[(totzeitIndex+1):100, :intervall],
    interval0[(totzeitIndex+1):100, :count], color=:blue, alpha=0.5,
    ↳label=L"\mathrm{Für\ Fit\ betrachtete\ Messwerte}", title="")

rounded_a = rounded_string(a_fit)
plot!(totzeitIndex:100,I,label=LaTeXString("\$\mathrm{Theoretische\ }
    ↳Intervallfunktion\ mit\ } a = \$rounded_a \mathrm{\,ms^{-1}}\$"), color=:
    ↳red)

xlabel!(L"\mathrm{Zeitdifferenz\ } [\mathrm{ms}]")
ylabel!(L"\mathrm{Anzahl\ der\ Messungen}")
```

[30]:



```
savefig(intervalFit, "../../media/B3.1/intervalFit.pdf");
```

## 1.6 4: Totzeit

### 1.6.1 Aus Intervallverteilung

$$a = a' / (1 - a' * \tau)$$

$$\Leftrightarrow \tau = 1/a' - 1/a$$

Benötigt  $a$  aus der Intervallverteilung. Falls  $a$  nicht schon definiert ist, wird es hier definiert:

```
[31]: if ! isdefined(Main, :a)
      a = 0.043
      end
```

```
[31]: 0.043
```

```
[32]: a_strich = 11044/300 * 10^(-3) # ms Gemessene Zählrate aus Kurzzeitmessung
      tau = 1/a_strich - 1/a # ms
```

```
[32]: 3.908257035283807
```

### Aus kurzen Messungen

```
[33]: short_measures = CSV.read("short_measurements.csv", DataFrame)
      short_measures.rates = short_measures.count ./ short_measures.seconds
      short_measures.rate_errors = sqrt.(short_measures.count) ./ short_measures.seconds

      short_measures.rate_w_error = measurement.(short_measures.rates, short_measures.
      ↪rate_errors)
      short_measures
```

```
[33]:
```

	sample_name	seconds	voltage	count	rates	rate_errors	rate_w_error
	String15	Int64	Int64	Int64	Float64	Float64	Measurem...
1	sample_6_7	300	500	20300	67.6667	0.474927	67.67 ± 0.47
2	sample_6_7	300	550	25509	85.03	0.532385	85.03 ± 0.53
3	sample_6_7	300	600	28522	95.0733	0.562949	95.07 ± 0.56
4	sample_6	300	500	11044	36.8133	0.350301	36.81 ± 0.35
5	sample_6	300	550	12260	40.8667	0.369083	40.87 ± 0.37
6	sample_6	300	600	12917	43.0567	0.378843	43.06 ± 0.38
7	sample_7	300	500	15833	52.7767	0.419431	52.78 ± 0.42
8	sample_7	300	550	17914	59.7133	0.446144	59.71 ± 0.45
9	sample_7	300	600	19421	64.7367	0.464531	64.74 ± 0.46
10	no_sample	300	500	2396	7.98667	0.163163	7.99 ± 0.16
11	no_sample	300	550	2513	8.37667	0.167099	8.38 ± 0.17
12	no_sample	300	600	2550	8.5	0.168325	8.5 ± 0.17

```
[34]: function totzeit_in_milliseconds(data, voltage)
    filtered_voltage = short_measures[short_measures.voltage .== voltage, :]

    n6 = filtered_voltage[filtered_voltage.sample_name .== "sample_6", :
↪rate_w_error][1]
    n7 = filtered_voltage[filtered_voltage.sample_name .== "sample_7", :
↪rate_w_error][1]
    n67 = filtered_voltage[filtered_voltage.sample_name .== "sample_6_7", :
↪rate_w_error][1]
    n0 = filtered_voltage[filtered_voltage.sample_name .== "no_sample", :
↪rate_w_error][1]

    A = n0*n67*n7-n6*n67*n7+n0*n67*n6-n0*n6*n7
    B = -2*n67*n0+2*n6*n7
    C = n67-n6+n0-n7

    t1 = (-B+sqrt(B^2 - 4*A*C))/(2*A) * 1000
    t2 = (-B-sqrt(B^2 - 4*A*C))/(2*A) * 1000

    return t1, t2
end
```

[34]: totzeit\_in\_milliseconds (generic function with 1 method)

Ermittle realistische Totzeiten

```
[35]: tau_1 = totzeit_in_milliseconds(short_measures, 500)[1]
tau_2 = totzeit_in_milliseconds(short_measures, 550)[1]
tau_3 = totzeit_in_milliseconds(short_measures, 600)[1];
```

Stelle alle Totzeiten als L<sup>A</sup>T<sub>E</sub>X-Tabelle dar

```
[36]: digits = 2

for voltage in (500, 550, 600)
    print("\$$voltage\$")
    for t in totzeit_in_milliseconds(short_measures, voltage)
        value = round(Measurements.value(t), digits=digits)
        uncert = round(Measurements.uncertainty(t), digits=digits)
        print(" & \$$value \pm $uncert\$")
        perc = round(Integer, uncert / value * 100)
        print(" \$(\pm $perc \%,\%)\$")
    end
    println(" \\\\")
end
```

```
$500$ & $6.41 \pm 0.33$ $(\pm 5 \%,\%)$ & $22.04 \pm 0.15$ $(\pm 1 \%,\%)$ \\\
$550$ & $2.32 \pm 0.25$ $(\pm 11 \%,\%)$ & $19.79 \pm 0.12$ $(\pm 1 \%,\%)$ \\\
$600$ & $1.13 \pm 0.22$ $(\pm 19 \%,\%)$ & $18.51 \pm 0.11$ $(\pm 1 \%,\%)$ \\\
```

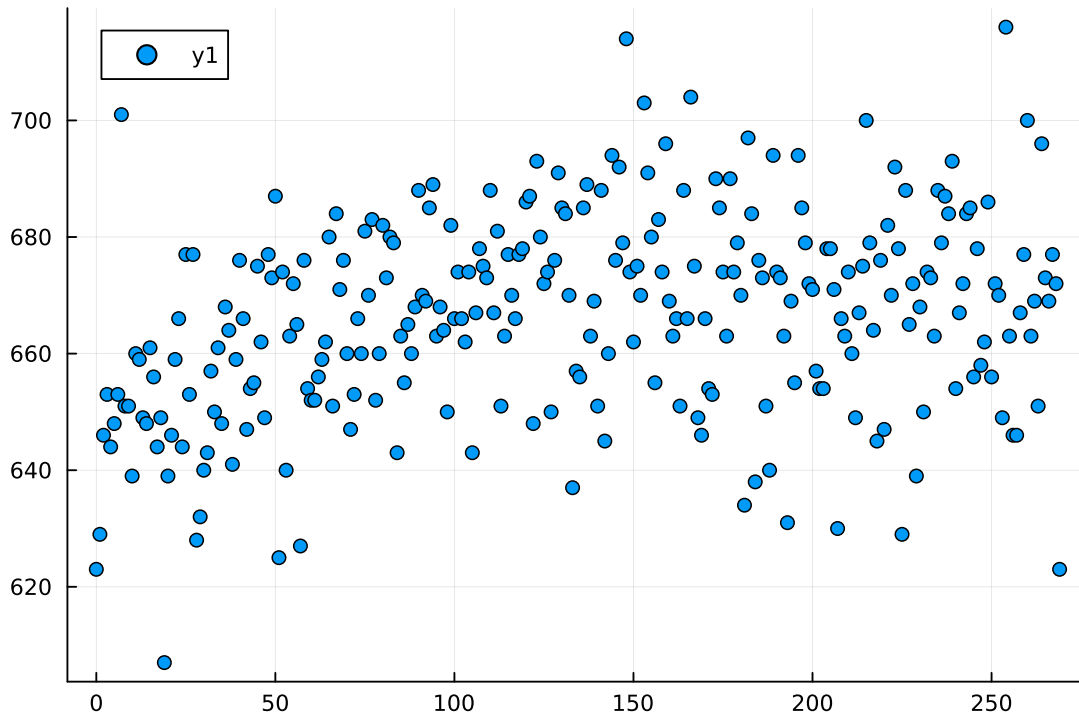
## 1.7 5: Aufgaben zum $\chi^2$ -Test

Benötigt Totzeit.

Daten einlesen

```
[37]: data = CSV.File("sample_6_7/500V_45min/divide_10.csv")  
      scatter(0:length(data), data.Zerfälle_in_10_sec)
```

[37]:



```
[38]: pos_min = 0  
      pos_max = 50  
      x_data = pos_min:pos_max  
      decays = data.Zerfälle_in_10_sec[x_data.+1]  
  
      avg_decays = mean(decays)
```

[38]: 653.8039215686274

```
[39]: std(decays)
```

[39]: 16.509415020336895

```
[40]: avg_decays
```

[40]: 653.8039215686274

```
[41]: avg_decays_corrected = avg_decays / (1 - a_strich*tau)
```

```
[41]: 763.6789739437968
```

```
[42]: avg_decays_corrected/avg_decays
```

```
[42]: 1.1680550525172038
```

```
[43]: decays_corrected = decays ./ (1 .- a_strich*tau);
```

### 1.7.1 1. Hypothesen

Zeigen Sie zeichnerisch, was die Hypothesen a, b und c besagen.

- a: Die Präparatstärke ist konstant im betrachteten Zeitraum und gleicht dem Mittelwert der 51 Messwerte.
- b: Die Präparatstärke ist konstant im betrachteten Zeitraum und gleicht dem Mittelwert der 51 Messwerte minus 10%.
- c: Die Präparatstärke nimmt im betrachteten Zeitraum linear mit der Zeit ab (als erste Näherung eines exponentiellen Abfalls). Die Anfangszählrate ist der Mittelwert, und der Abfall von einer Messung zur anderen sei 1.

Hypothese  $H_1$ : “Die Präparatstärke ist konstant im betrachteten Zeitraum und gleicht dem Mittelwert der 51 Messwerte.”

Diese Hypothese besagt, jeder erwartete Zählmenge  $n_1(i)$  sei gleich dem Mittelwert der 51 Messungen  $n_i$ .

$$\begin{aligned}\bar{x} &= \bar{n} \\ n_1(i) &= \frac{1}{51} \sum_{i=1}^{51} n_i \\ \chi_1^2 &= \sum_i \frac{(n_i - \bar{n})^2}{\bar{n}}\end{aligned}$$

Hypothese  $H_2$ : “Die Präparatstärke ist konstant im betrachteten Zeitraum und gleicht dem Mittelwert der 51 Messwerte minus 10%.”

Dies bedeutet, dass die erwarteten Zählungen  $n_2(i)$  um 10% kleiner als die Zählungen nach  $H_1$  sein müssen.

$$\begin{aligned}n_2(i) &= \frac{9}{10} \cdot n_1(i) \\ \chi_2^2 &= \sum_i \frac{(n_i - 0.9 \bar{n})^2}{0.9 \bar{n}}\end{aligned}$$

Hypothese  $H_3$ : “Die Präparatstärke nimmt im betrachteten Zeitraum linear mit der Zeit ab (als erste Näherung eines exponentiellen Abfalls). Die Anfangszählrate ist der Mittelwert, und der Abfall von einer Messung zur anderen sei 1.”

Die erwarteten Ereignisse starten demnach bei  $n_1(0)$  und fallen dann linear mit einer Steigung von 1 ab.

$$n_3(i) = n_1(0) - i$$

$$\chi_3^2 = \sum_i \frac{(n_i - (n - i))^2}{(n - i)}$$

```
[44]: function plot_hypotheses(x_data, decays, title; legend_pos)
    scatter(x_data, decays, label=L"\mathrm{Messwerte}", legend=legend_pos,
    yrange=[580, 830])

    title!(title)
    xlabel!(L"\mathrm{Zeitintervalle je}\ 10\mathrm{\,s}")
    ylabel!(L"\mathrm{Ereignisse}\ \left[(10\,\mathrm{s})^{-1}\right]")

    h1_results = fill(mean(decays), length(x_data))

    plot!(
        x_data,
        h1_results,
        label=L"H_1",
        color=:red4,
        lw=2
    )

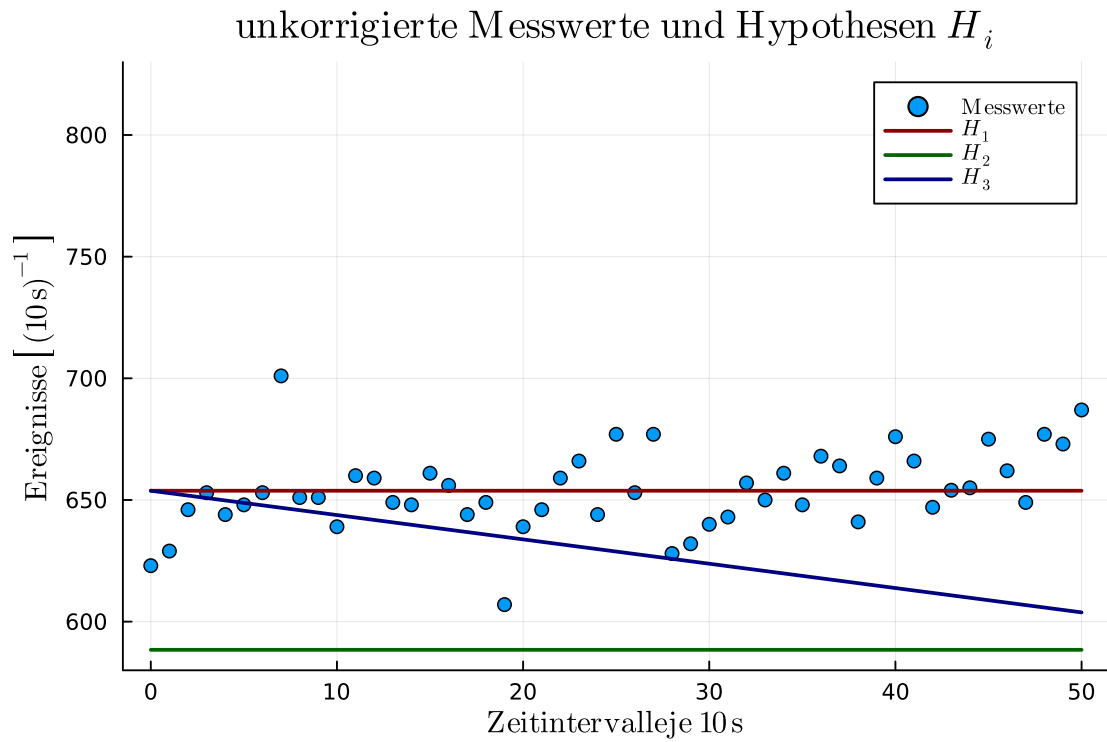
    plot!(
        x_data,
        0.9 .* h1_results,
        label=L"H_2",
        color=:darkgreen,
        lw=2
    )

    plot!(
        x_data,
        h1_results .- x_data,
        label=L"H_3",
        color=:navy,
        lw=2
    )
end
```

[44]: plot\_hypotheses (generic function with 1 method)

```
[45]: plot_hypotheses(x_data, decays, L"\mathrm{unkorrigierte\ Messwerte\ und\ }
    Hypothesen\ } H_i", legend_pos=:topright)
```

[45]:



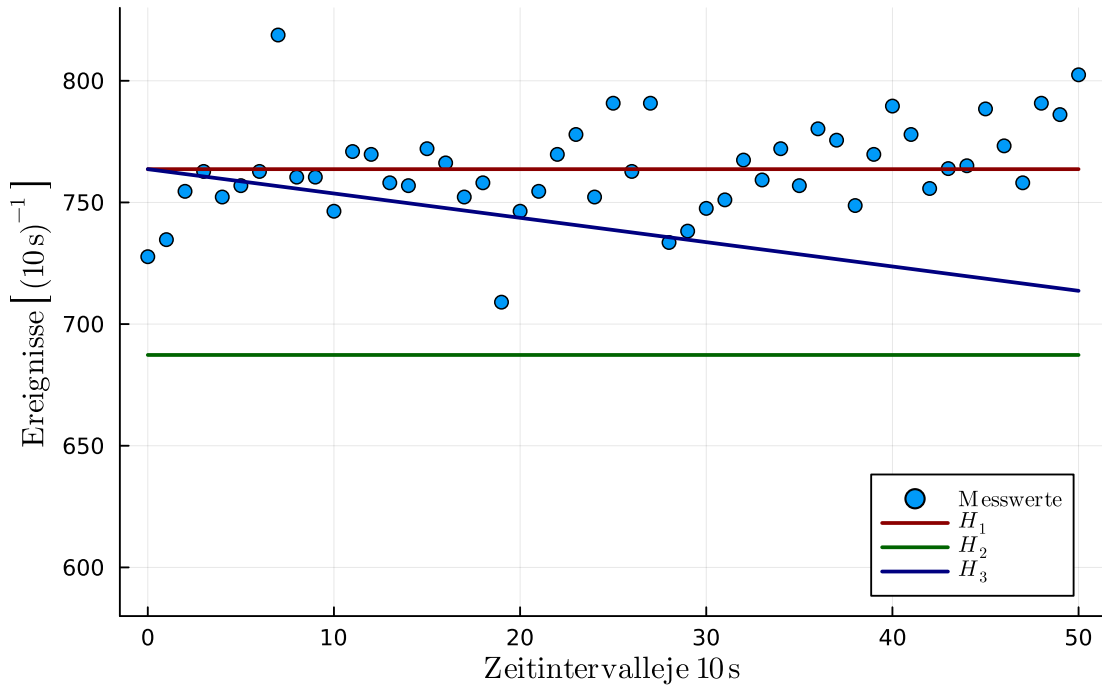
```
savefig("../media/B3.1/Hypothesen_plot.pdf");
```

```
[46]: plot_hypotheses(x_data, decays_corrected, L"\mathrm{totzeitkorrigierte\_\u2192Messwerte\ und\ Hypothesen\ } H_i"; legend_pos=:bottomright)
```

[46]:



### totzeitkorrigierte Messwerte und Hypothesen $H_i$



savefig("../media/B3.1/Hypothesen\_plot\_corr.pdf");

Man Erkennt, dass  $H_1$  die Messwerte am besten beschreibt.

#### 1.7.2 2. Hypothesentest

Nun werden die drei Hypothesen  $H_i$  mithilfe des  $\chi^2$ -Tests geprüft. Hierzu werden 51 Messwerte aus der 45 min-Messung beider Proben gewählt, deren Zählungen über 10 s gemittelt werden. Dies wird sowohl für die nicht-totzeitkorrigierten als auch für die totzeitkorrigierten Daten durchgeführt.

Durch die Bildung des Mittelwertes gibt es noch 50 statistische Freiheitsgrade. Dadurch können die erlaubten Grenzen für  $\chi^2$  für ein System mit 50 Freiheitsgraden und einer Signifikanz von 5% verwendet werden [6].

$$\chi_{\min}^2 = 32.357$$

$$\chi_{\max}^2 = 71.420$$

unkorrigiert

$$\chi_1^2 = \sum_i \frac{(x_i - \bar{x})^2}{\bar{x}}$$

$$\chi_2^2 = \sum_i \frac{(x_i - 0.9 \bar{x})^2}{0.9 \bar{x}}$$

$$\chi_3^2 = \sum_i \frac{(x_i - (\bar{n} - i))^2}{(\bar{n} - i)}$$

```
[47]: chi_squared_1 = sum((decays .- avg_decays).^2 ./ avg_decays)
```

```
[47]: 20.84422984644914
```

```
[48]: (1 - chi_squared_1 / 32.357)*100
```

```
[48]: 35.58046219844504
```

```
[49]: chi_squared_2 = sum((decays .- 0.9*avg_decays).^2 ./ (0.9*avg_decays))
```

```
[49]: 393.6491442738329
```

```
[50]: chi_squared_2 / 71.420
```

```
[50]: 5.511749429765232
```

```
[51]: chi_squared_3 = sum((decays .- (avg_decays .- x_data)).^2 ./ (avg_decays .-  
↪x_data))
```

```
[51]: 106.02106295610936
```

```
[52]: chi_squared_3 / 71.420
```

```
[52]: 1.4844730181477088
```

totzeitkorrigiert

$$\chi_{i,\text{korr}}^2 = \frac{1}{1 - \frac{m}{\Delta t} \tau} \cdot \chi_i^2$$

```
[53]: chi_squared_1_corrected = 1 / (1-a_strich*tau) * chi_squared_1
```

```
[53]: 24.34720798797481
```

```
[54]: (1 - chi_squared_1_corrected / 32.357)*100
```

```
[54]: 24.75443339007074
```

```
[55]: chi_squared_2_corrected = 1 / (1-a_strich*tau) * chi_squared_2
```

```
[55]: 459.80387188812415
```

```
[56]: chi_squared_2_corrected / 71.420
```

```
[56]: 6.438026769646095
```

```
[57]: chi_squared_3_corrected = 1 / (1-a_strich*tau) * chi_squared_3
```

```
[57]: 123.83843825912807
```

```
[58]: chi_squared_3_corrected / 71.420
```

```
[58]: 1.7339462091728937
```

### 1.7.3 3. Halbwertszeit

Welche Halbwertszeit ergibt sich aus der Hypothese c, wenn Sie einen zeitlichen Abstand der Messungen von 10 Sekunden annehmen als Näherung eines exponentiellen Zerfalls?

Nun soll nach Hypothese  $H_3$  die Halbwertszeit  $T_{1/2}$  der Proben bestimmt werden. Hierzu wird der exponentielle Zerfall durch eine lineare Kurve beschrieben. Es wird erwartet, dass diese Halbwertszeit deutlich geringer als die ca. 30yr der Probe ist.

Dabei wird davon ausgegangen, dass innerhalb der Messdauer  $\Delta t = 10\text{s}$  der Wert  $N_0$  um 1 sinkt. Weiterhin wird die Halbwertszeit nach Gleichung ?? bestimmt. Nach Hypothese  $H_3$  ist  $N(0) = \bar{n}$  der Mittelwert der Zählraten.

```
[59]: T_12 = Integer(round(10 * log(2) / log(avg_decays/(avg_decays-1)))) # seconds
println(T_12, " seconds")
```

```
T_12_str = string("\$T_{1/2}(m^\prime)=", Integer(round(T_12/3600)), "\,
↪ \"\\,\\mathrm{h}\\,\", Integer(round(T_12%60)), \"\\,\\mathrm{min}\\$\",)
println(T_12_str)
latexstring(T_12_str)
```

4528 seconds

$\$T_{1/2}(m^\prime)=1\,,\mathrm{h}\,,28\,,\mathrm{min}\$$

```
[59]:  $T_{1/2}(m') = 1\text{ h }28\text{ min}$ 
```

```
[60]: T_12_corr = Integer(round(10 * log(2) / log(avg_decays_corrected/
↪ (avg_decays_corrected-1)))) # seconds
println(T_12_corr, " seconds")
```

```
T_12_corr_str = string("\$T_{1/2}(m)=", Integer(round(T_12_corr/3600)), "\,
↪ \"\\,\\mathrm{h}\\,\", Integer(round(T_12_corr%60)), \"\\,\\mathrm{min}\\$\",)
println(T_12_corr_str)
latexstring(T_12_corr_str)
```

5290 seconds

$\$T_{1/2}(m)=1\,,\mathrm{h}\,,10\,,\mathrm{min}\$$

```
[60]:  $T_{1/2}(m) = 1\text{ h }10\text{ min}$ 
```