Auswertung

July 9, 2024

1 Auswertung B1.4

[1]:
```
using LaTeXStrings
using LinearAlgebra
using LsqFit
using Measurements
using Plots
using Statistics
```

1.1 Literaturwerte

[2]:
```
# aus Gerthsen Physik
h = 6.6260688e-34 # Js
Δh = 0.0000005e-34 # Js
e = 1.60217646e-19 # C
Δe = 0.00000006e-19 # C
h_durch_e = h/e # Js/C

Δh_durch_e = sqrt( (Δh/e)^2 + (h*Δe/e^2)^2 )

# aus Wikipedia (exakte Werte):
h = 6.62607015e-34 # Js
e = 1.602176634e-19 # C
h_durch_e_exakt = h/e; # Js/C
```

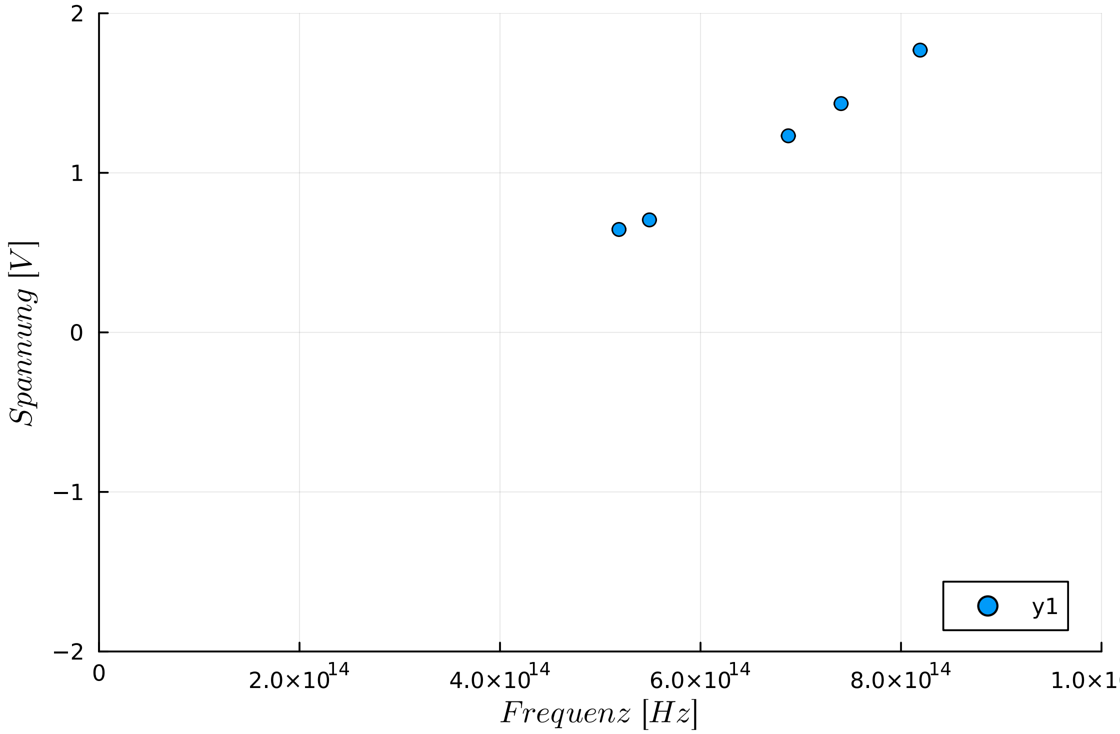
1.2 1. Bestimmung von h/e

1.2.1 Gegenspannungsmethode

[3]:
```
gegenSpannungen = [1.769, 1.434, 1.232, 0.705, 0.645] # V
ΔgegenSpannungen = 0.003 # V
wellenlaengen = [366, 405, 436, 546, 578] # nm
Δwellenlaengen = 7 # nm
c_nm = 2.99792458e17 # nm/s
frequenzen = c_nm ./ wellenlaengen # Hz
Δfrequenzen = c_nm * Δwellenlaengen ./ (wellenlaengen.^2) # Hz (nach Gaußscher
↪Fehlerfortpflanzung)
```

```julia
scatter(frequenzen, gegenSpannungen, yerr = ΔgegenSpannungen, legend=:
  ↪bottomright, xaxis=[0,10^15], yaxis=[-2,2])
xlabel!(L"Frequenz \enspace [Hz]")
ylabel!(L"Spannung \enspace [V]")
```

[3]:



[4]:
```julia
frequenzen ./ 1e14
```

[4]:
```
5-element Vector{Float64}:
 8.191050765027322
 7.4022829135802475
 6.87597380733945
 5.490704358974359
 5.186720726643599
```

[5]:
```julia
Δfrequenzen ./ 1e14
```

[5]:
```
5-element Vector{Float64}:
 0.1566594408611783
 0.12794069233348573
 0.1103940748884774
 0.0703936456278764
 0.0628149568970678
```

Gerade fitten Steigung abschätzen

[6]:
```
Δx = frequenzen[1]-frequenzen[length(frequenzen)]
```

[6]:
```
3.004330038383724e14
```

[7]:
```
Δy = gegenSpannungen[1]-gegenSpannungen[length(gegenSpannungen)]
```

[7]:
```
1.1239999999999999
```

[8]:
```
Δy / Δx
```

[8]:
```
3.741266723827359e-15
```

Fit

[9]:
```
modelFunction(x, a) = a[1]*x .+ a[2] # a = fit-parameter
a0 = [10e-15, -1.5]
fit = curve_fit(modelFunction, frequenzen, gegenSpannungen, a0);
```

[10]:
```
a_gegen_fit = fit.param[1]
```

[10]:
```
3.770981129124753e-15
```

[11]:
```
b_gegen_fit = fit.param[2]
```

[11]:
```
-1.3429140603905638
```

[12]:
```
Δa_gegen_fit = sqrt(estimate_covar(fit)[1,1])
```

[12]:
```
1.1575749616935338e-16
```

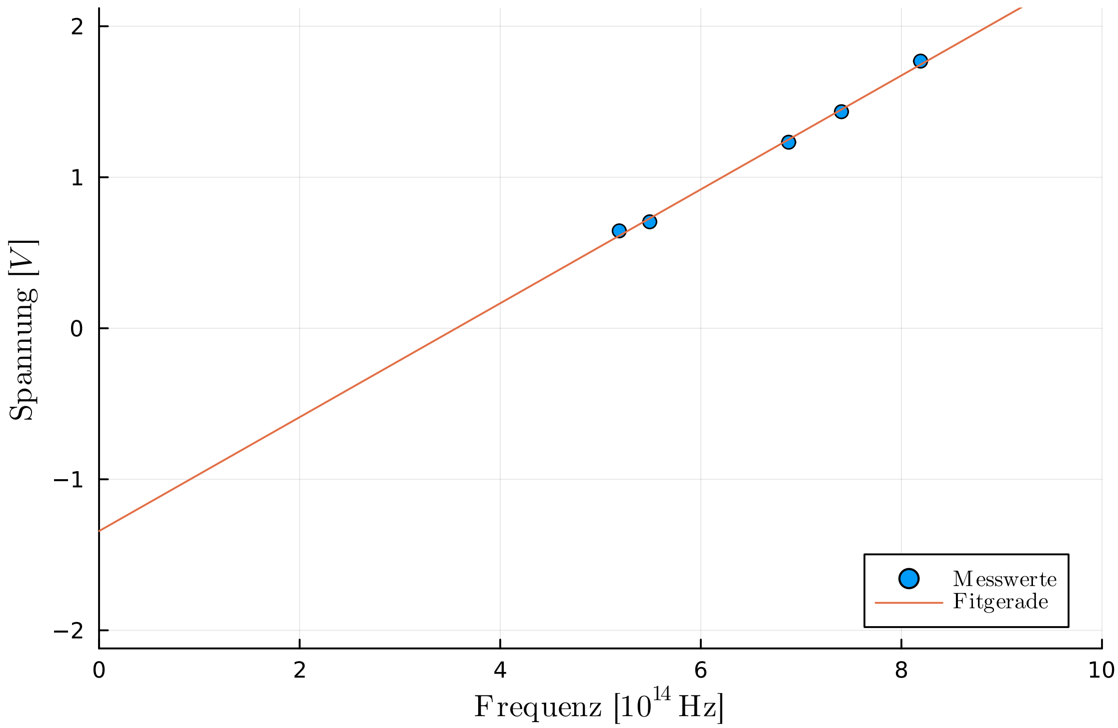
[13]:
```
Δb_gegen_fit = sqrt(estimate_covar(fit)[2,2])
```

[13]:
```
0.0778620452885972
```

Plot

[14]:
```
fitGeradeGegen(x) = a_gegen_fit * 1e14 * x + b_gegen_fit
gegenPlot = scatter(frequenzen ./ 1e14, gegenSpannungen, yerr=ΔgegenSpannungen,␣
 ↪legend=:bottomright,
    xlims=[0,10], ylims=[-2,2], xwiden=false, ywiden=true,␣
 ↪label=L"\mathrm{Messwerte}")
xlabel!(L"\mathrm{Frequenz} \enspace [10^{14} \mathrm{\, Hz}]")
ylabel!(L"\mathrm{Spannung} \enspace [V]")
plot!(fitGeradeGegen, label=L"\mathrm{Fitgerade}")
```

[14]:

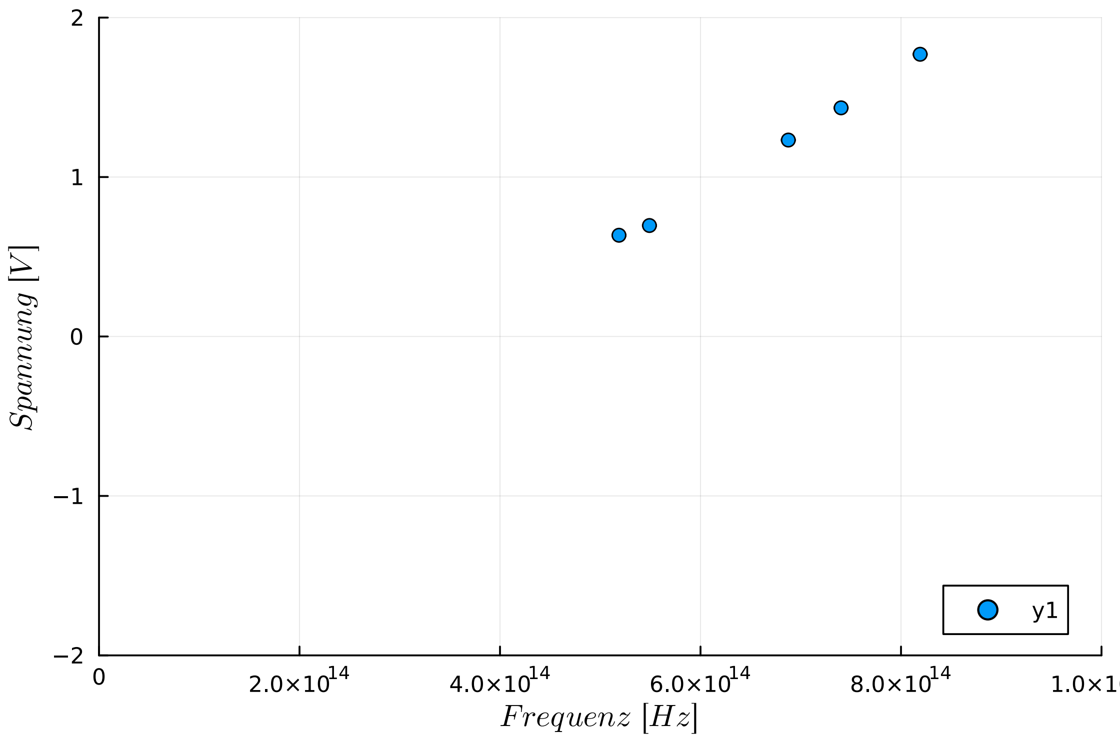


savefig(gegenPlot, "../../media/B1.4/gegenPlot.pdf");

1.2.2 Direkte Methode

[15]:
```
direkteSpannungen = [1.770, 1.434, 1.232, 0.696, 0.635] # V
ΔdirekteSpannungen = ΔgegenSpannungen
scatter(frequenzen, direkteSpannungen, yerr = ΔdirekteSpannungen, legend=:
  ↪bottomright, xaxis=[0,10^15], yaxis=[-2,2])
xlabel!(L"Frequenz \enspace [Hz]")
ylabel!(L"Spannung \enspace [V]")
```

[15]:



Fit

```
[16]: modelFunction(x, a) = a[1]*x .+ a[2] # a = fit-parameter
      a0 = [10e-15, -1.5]
      fit = curve_fit(modelFunction, frequenzen, direkteSpannungen, a0);
```

```
[17]: a_direkt_fit = fit.param[1]
```

```
[17]: 3.8115005813539555e-15
```

```
[18]: b_direkt_fit = fit.param[2]
```

```
[18]: -1.3733758093303134
```

```
[19]: Δa_direkt_fit = sqrt(estimate_covar(fit)[1,1])
```

```
[19]: 1.1177527711785443e-16
```

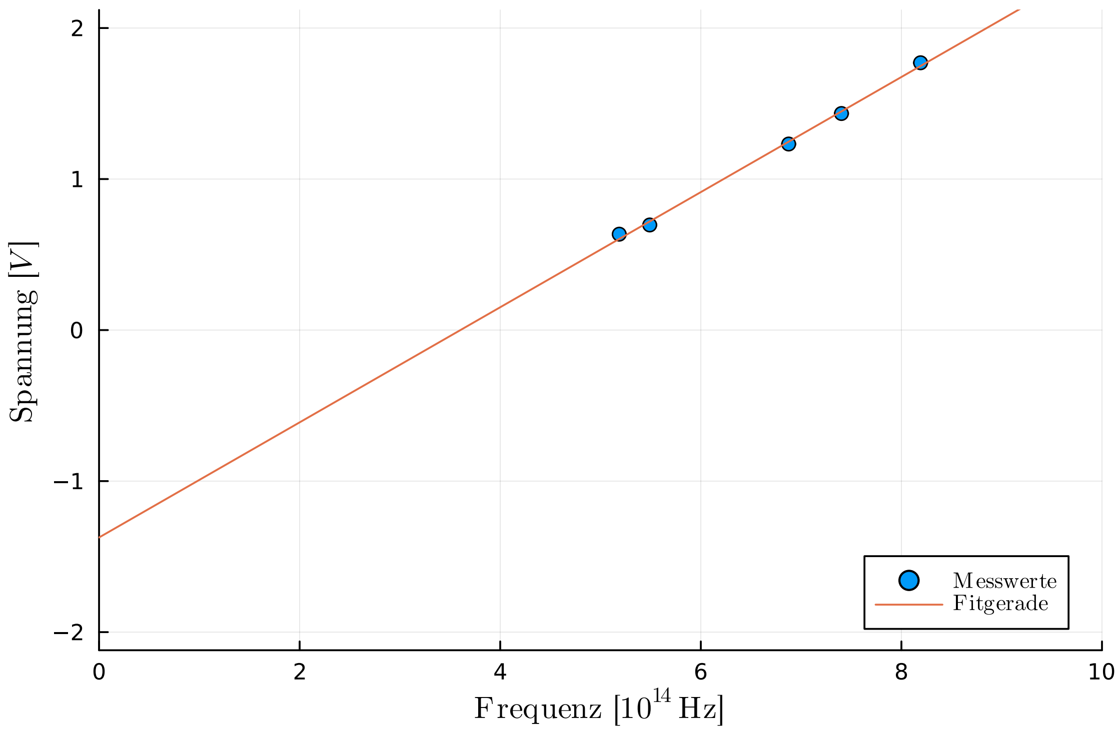
```
[20]: Δb_direkt_fit = sqrt(estimate_covar(fit)[2,2])
```

```
[20]: 0.07518348251365051
```

Plot

[21]:
```
fitGeradeDirekt(x) = a_direkt_fit * 1e14 * x + b_direkt_fit
direktPlot = scatter(frequenzen ./ 1e14, direkteSpannungen,␣
 ↪yerr=ΔdirekteSpannungen, legend=:bottomright,
    xlims=[0,10], ylims=[-2,2], xwiden=false, ywiden=true,␣
 ↪label=L"\mathrm{Messwerte}")
xlabel!(L"\mathrm{Frequenz} \enspace [10^{14} \mathrm{\, Hz}]")
ylabel!(L"\mathrm{Spannung} \enspace [V]")
plot!(fitGeradeDirekt, label=L"\mathrm{Fitgerade}")
```

[21]:



savefig(direktPlot, "../../media/B1.4/direktPlot.pdf");

1.2.3 Vergleich

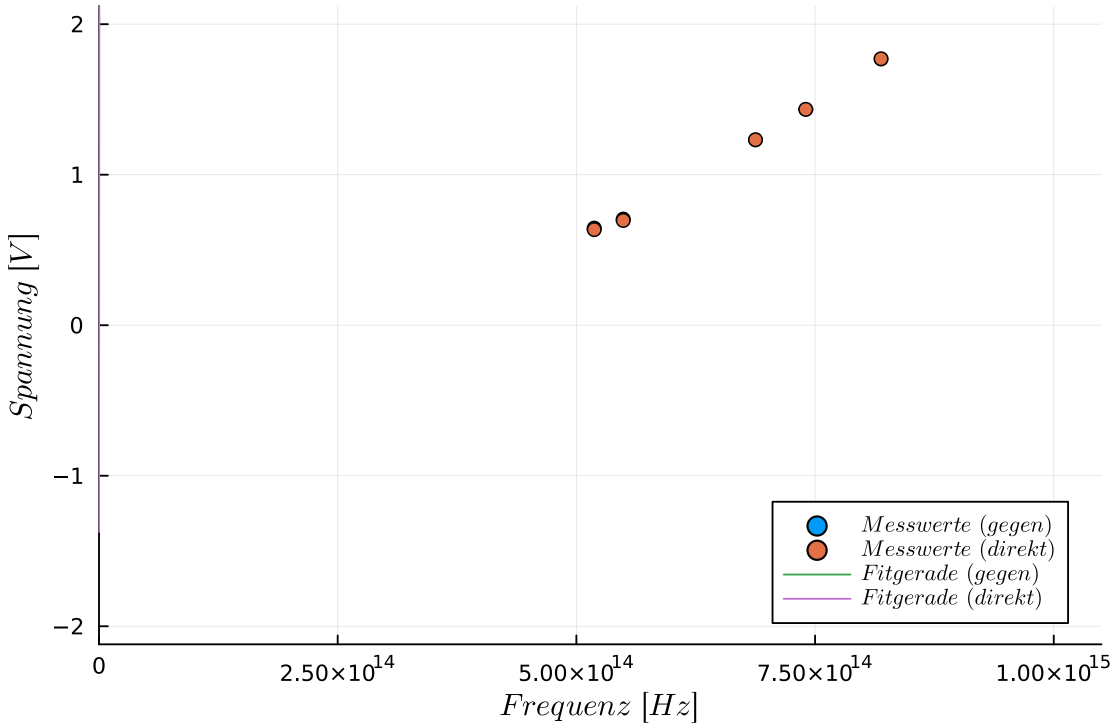
[22]:
```
[[a_gegen_fit, a_direkt_fit], [b_gegen_fit, b_direkt_fit]]
```

[22]:
```
2-element Vector{Vector{Float64}}:
 [3.770981129124753e-15, 3.811500581353955e-15]
 [-1.3429140603905638, -1.3733758093303134]
```

[23]:
```
scatter(frequenzen, gegenSpannungen, yerr=ΔgegenSpannungen, legend=:bottomright,
    xlims=[floor(0),1.05*10^15], ylims=[-2,2], xwiden=false, ywiden=true,␣
 ↪label=L"Messwerte \enspace (gegen)")
scatter!(frequenzen, direkteSpannungen, yerr=ΔdirekteSpannungen,␣
 ↪label=L"Messwerte \enspace (direkt)")
```

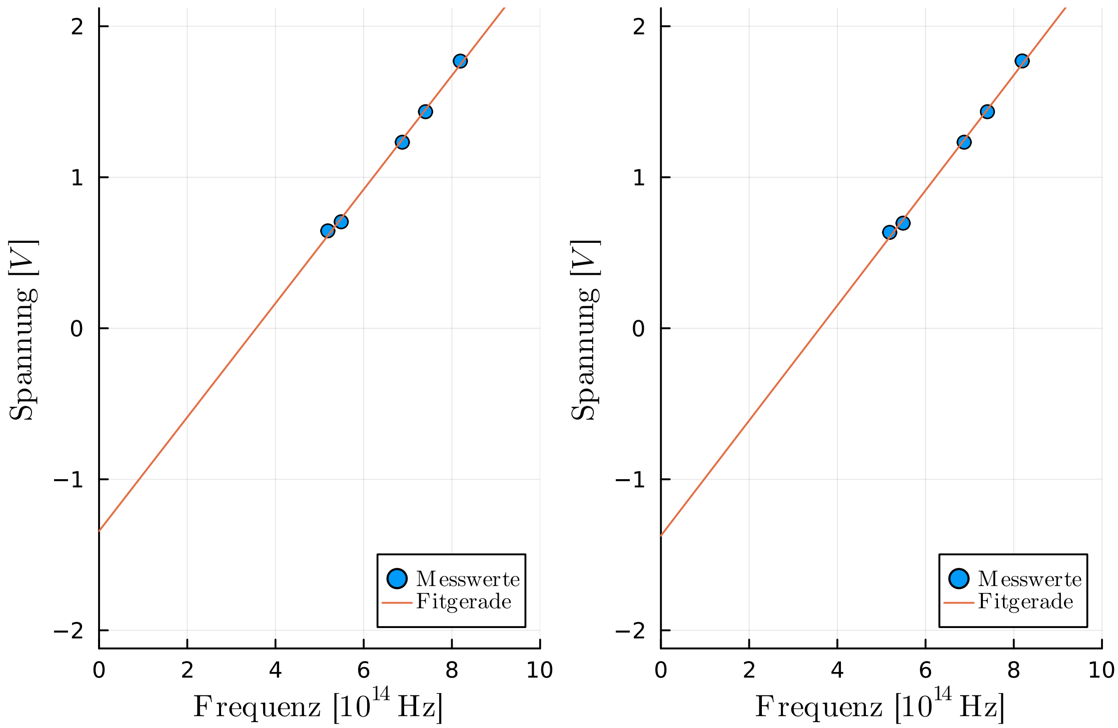
```
xlabel!(L"Frequenz \enspace [Hz]")
ylabel!(L"Spannung \enspace [V]")
plot!(fitGeradeGegen, label=L"Fitgerade \enspace (gegen)")
plot!(fitGeradeDirekt, label=L"Fitgerade \enspace (direkt)")
```

[23]:



[24]:
```
plot(gegenPlot, direktPlot, layout=(1,2))
```

[24]:



1.3 2. Berechnung von W_A

[25]:
```
W_A_gegen = abs(b_gegen_fit * e) # J
```

[25]: 2.151585529027826e-19

[26]:
```
W_A_gegen_ev = W_A_gegen / 1.602176634e-19 # eV
```

[26]: 1.3429140603905638

[27]:
```
ΔW_A_gegen = Δb_gegen_fit * e # J
```

[27]: 1.2474874963684022e-20

[28]:
```
ΔW_A_gegen_ev = ΔW_A_gegen / 1.602176634e-19 # eV
```

[28]: 0.0778620452885972

[29]:
```
W_A_direkt = abs(b_direkt_fit * e) # J
```

[29]: 2.2003906314098673e-19

[30]:
```
W_A_direkt_ev = W_A_direkt / 1.602176634e-19 # eV
```

[30]: 1.3733758093303134

[31]:
```
ΔW_A_direkt = Δb_direkt_fit * e # J
```

[31]: 1.2045721894611842e-20

[32]:
```
ΔW_A_direkt_ev = ΔW_A_direkt / 1.602176634e-19 # eV
```

[32]: 0.07518348251365051

1.4 3. Photostrom

[33]:
```
# gemessene Spannungen
U_blau = [1.486, 1.067, 0.807, 0.655, 0.440, 0.338]
U_grün = [0.536, 0.357, 0.265, 0.190, 0.126, 0.090];
```

[34]:
```
# Relative Intensität
T_blau = [68., 48., 33., 28., 20., 14.]
T_grün = [67., 46., 31., 23., 16., 11,]
ΔT = 1;
```

[35]:
```
# berechne Photostrom
R=10000
I_blau = U_blau ./ R
I_grün = U_grün ./ R;
```

[36]:
```
ΔI = 0.005 / R
```

[36]: 5.0e-7

[37]:
```
I_Blau = measurement.(I_blau, ΔI)
I_Grün = measurement.(I_grün, ΔI)
T_Blau = measurement.(T_blau, ΔT)
T_Grün = measurement.(T_grün, ΔT);
```

[38]:
```
# Funktion zur Berechnung der linearen Regression
function fit_func(x, y)
    X = [ones(length(x)) x]
    coef = X \ y
    return coef[1], coef[2]
end
```

[38]: fit_func (generic function with 1 method)

[39]:
```
rounded_string(value) = rpad(round(value, digits=3),␣
↪length(string(round(value))) +2 , "0")
```

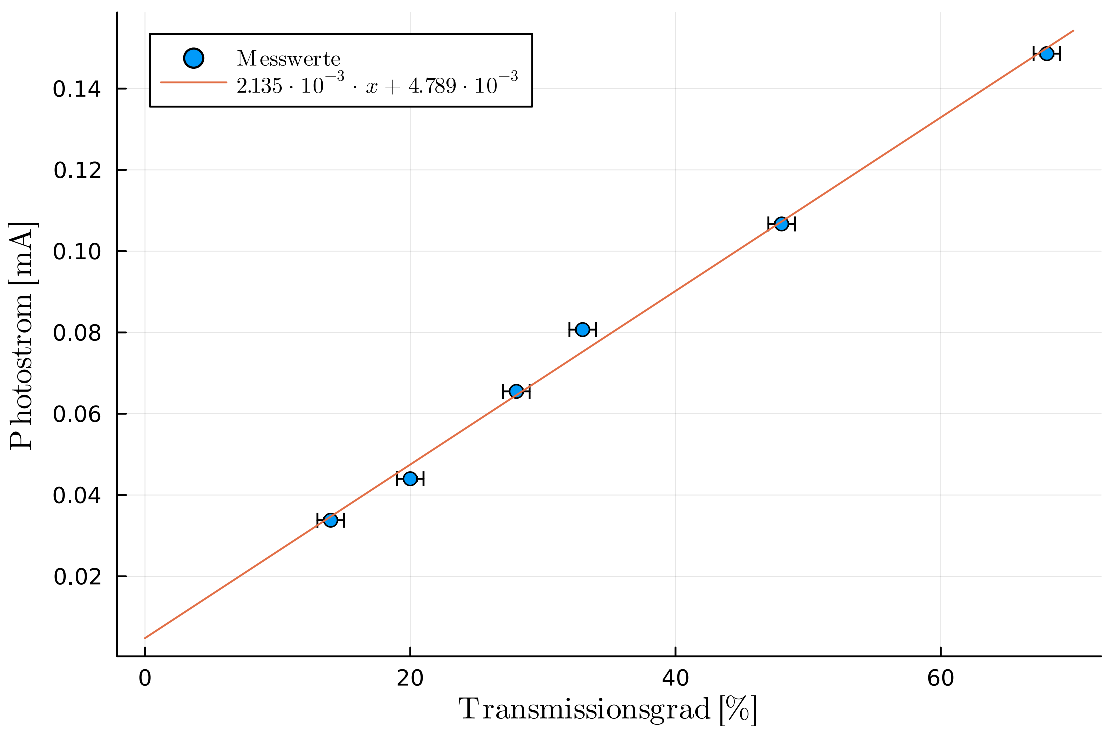
[39]: rounded_string (generic function with 1 method)

[40]:

```
# Berechnung der linearen Regression
a, b = fit_func(T_blau, I_blau .* 1e3)

@show a
@show b

# Erstellen der Fitlinie
f(x) = a .+ b .* x
a_rounded = rounded_string(a*1e3)
b_rounded = rounded_string(b*1e3)

blau = scatter(T_blau, I_blau.*1e3, xerr=1, yerr=ΔI .* 10^3,
    label=L"\mathrm{Messwerte}", xlabel=L"\mathrm{Transmissionsgrad}\ [\%]",␣
 ↪ylabel=L"\mathrm{Photostrom}\ [\mathrm{mA}]")
plot!(0:70, f, label="\$$b_rounded\\cdot10^{-3}\\cdot x +␣
 ↪$a_rounded\\cdot10^{-3}\$", linewidth = 1)
```

```
a = 0.0047886687463114435
b = 0.002135393305792092
```

[40]:



savefig(blau, "../../media/B1.4/Photostrom_blau.pdf");

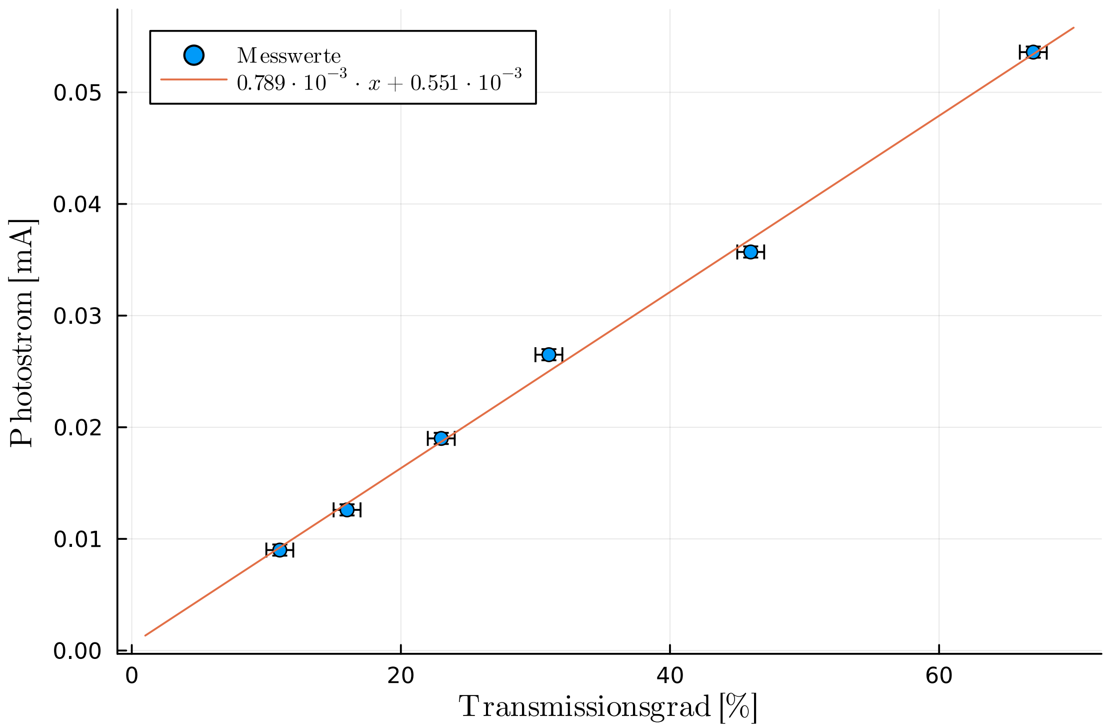
[41]:

```
# Berechnung der linearen Regression
a, b = fit_func(T_grün, I_grün .* 1e3)
```

```
@show a
@show b

# Erstellen der Fitlinie
f(x) = a .+ b .* x
a_rounded = rounded_string(a*1e3)
b_rounded = rounded_string(b*1e3)

gruen = scatter(T_grün, I_grün.*1e3, xerr=1, yerr=ΔI .* 10^3,
    label=L"\mathrm{Messwerte}", xlabel=L"\mathrm{Transmissionsgrad}\ [\%]",␣
↪ylabel=L"\mathrm{Photostrom}\ [\mathrm{mA}]")
plot!(1:70, f, label="\$$b_rounded\\cdot10^{-3}\\cdot x +␣
↪$a_rounded\\cdot10^{-3}\$", linewidth = 1)
```

```
a = 0.000551364049712036
b = 0.0007891330706274626
```

[41]:



savefig(gruen, "../../media/B1.4/Photostrom_gruen.pdf");

1.5 3. Untersuchung von LEDs mit der Photozelle

[42]:
```
ledStoppspannungen = [1.047, 0.891, 0.807] # V
ledNamen = ["blue", "verde", "true green"]
ΔledStoppspannungen = ΔgegenSpannungen

ledFrequenzen = (ledStoppspannungen .- b_direkt_fit) ./ a_direkt_fit # Hz
ledWellenlängen = c_nm ./ ledFrequenzen # nm
```

[42]:
```
3-element Vector{Float64}:
 472.0998795094924
 504.62433101618024
 524.0652198867912
```

[43]:
```
# Test ob Funktion in LaTeX auch stimmt:
ledWellenlängen2 = a_direkt_fit * c_nm ./ (ledStoppspannungen .- b_direkt_fit)
```

[43]:
```
3-element Vector{Float64}:
 472.0998795094923
 504.62433101618024
 524.0652198867912
```

[44]:
```
ΔledWellenlängen = sqrt.( (c_nm * Δa_direkt_fit ./ (ledStoppspannungen .-
 ↪b_direkt_fit)).^2 +
    (a_direkt_fit * c_nm * ΔledStoppspannungen ./ (ledStoppspannungen .-
 ↪b_direkt_fit).^2).^2 +
    (a_direkt_fit * c_nm * Δb_direkt_fit ./ (ledStoppspannungen .-
 ↪b_direkt_fit).^2).^2 ) # nm
```

[44]:
```
3-element Vector{Float64}:
 20.176024971194252
 22.36447778670769
 23.733247583742223
```