

beim nächsten Mal bitte eine PDF-Datei abgeben
 dafür:
 1. alle Zellen des ipython notebooks ausführen
 2. speichern
 3. in Konsole: ipython nbconvert --to pdf [notebook.ipynb]
 ↑
 bzw. eigene Datei

Blatt01_Grisard_Nitschke

October 29, 2018

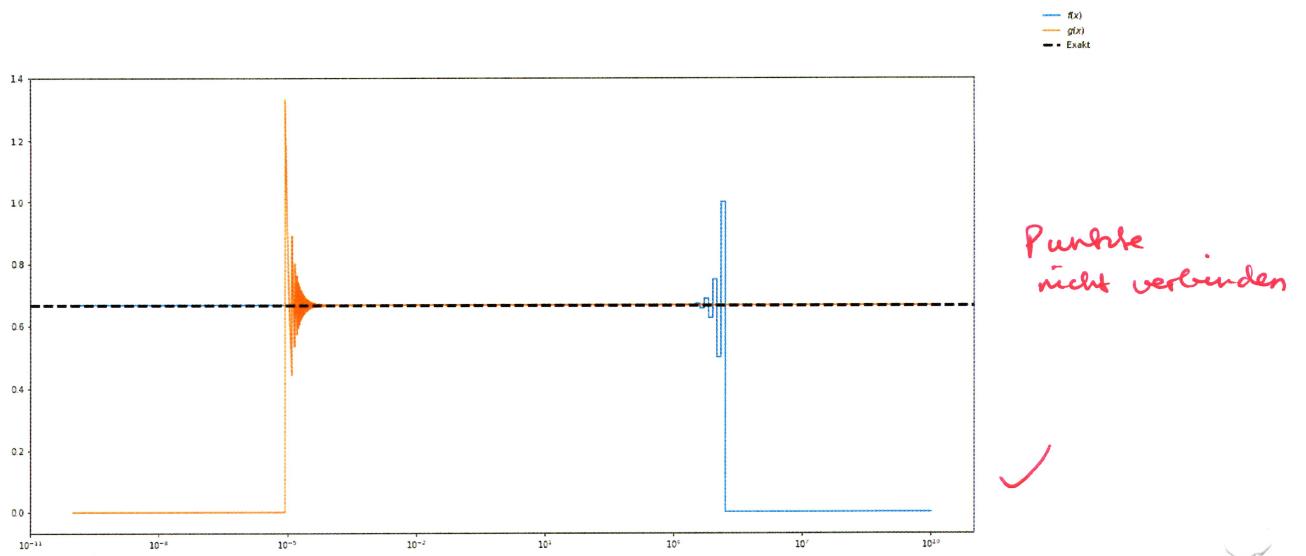
1 Blatt 1

1.1 Aufgabe 1: Numerische Stabilität

```

In [1]: import numpy as np
        import matplotlib.pyplot as plt
        from matplotlib import rcParams
        rcParams['font.size'] = 20
        ↗ das Beispiel im notebook so nicht
        probiert mal: plt.rcParams.update
        ({'font.size': 20})
In [2]: def f(x):
        return (x**3 + 1/3) - (x**3 - 1/3)
In [3]: def g(x):
        return ( (3 + x**3/3) - (3 - x**3/3) ) / x**3
In [4]: x = np.logspace(-10, 10, 100000)
In [5]: fig, ax = plt.subplots(1, 1, figsize = (20, 10))
        ax.plot(x, f(x), label = '$f(x)$')
        plt.plot(x, g(x), label = '$g(x)$')
        ax.set_xscale('log')
        ax.set_label
        ax.axhline(y = 2/3, label = 'Exakt', linestyle = '--', linewidth = 3, color = 'k')
        fig.legend()
        plt.show()
    
```

1	2	3	4	
3/3	6,5/7	6/6	3,75/4	19,25/20



Punkte
nicht verbinden

Bestimme die Schranken, für die die Funktionen den Wert 0 ergeben:

```
In [6]: f_0 = (x[f(x) == 0])[0]
g_0 = (x[g(x) == 0])[-1]
print(f'Die Funktion f liefert für x > {f_0} den Wert 0, g für x < {g_0}.')
```

Die Funktion f liefert für $x > 165177.99095397102$ den Wert 0, g für $x < 8.730706936207748e-6$

Bestimme alle x-Werte aus dem gewählten Bereich, für die die relativen Abweichungen vom exakten Wert $\frac{2}{3}$ kleiner als 1% sind:

```
f_1percent = (x[abs((f(x) - 2/3)) / 2 * 3 <= 0.01])
print(f'Die relative Abweichung von f ist kleiner als 1% für x < {f_1percent[-1]}.')

g_1percent = (x[abs((g(x) - 2/3)) / 2 * 3 <= 0.01])
print(f'Die relative Abweichung von g ist kleiner als 1% für x > {g_1percent[0]}.'
```

Die relative Abweichung von f ist kleiner als 1% für $x < 41280.61487266133$.

Die relative Abweichung von g ist kleiner als 1% für $x > 1.0971105782359554e-05$.

$3.984 \cdot 10^{-5}$ (v) 3/3 p.

1.2 Aufgabe 2: $e^- e^+ \rightarrow \gamma\gamma$

In [8]: alpha = 1 / 137

```
def diff_cross_section(theta, E = 50): #E in GeV
    s = 4 * E**2
    gamma = E / 511e-6 #big number
    beta = np.sqrt(1 - 1 / gamma**2) #near 1 for big gamma
    return alpha **2 / s * (2 + np.sin(theta)**2) / (1 - beta**2 * np.cos(theta)**2)
```

1P.

Betrachte den Bereich um $\theta = \pi$ genauer, da hier der Nenner wegen $\beta \approx 1$ sehr klein wird. Zudem werden zwei etwa gleich grosse Zahlen voneinander abgezogen, was zu Auslöschungseffekten führen kann.

In [9]: `theta = np.linspace(np.pi - 1e-7, np.pi + 1e-7, 10000)`

Alternative Form:

+ es wird durch eine sehr kleine Zahl dividiert

In [10]: `def diff_cross_section_alternative(theta, E = 50): #E in GeV`

hier hätte ich mir gewünscht, dass ihr die Schritte der Umformung genauer zeigt

-0,5 P

1,5 P.

`s = 4 * E**2`

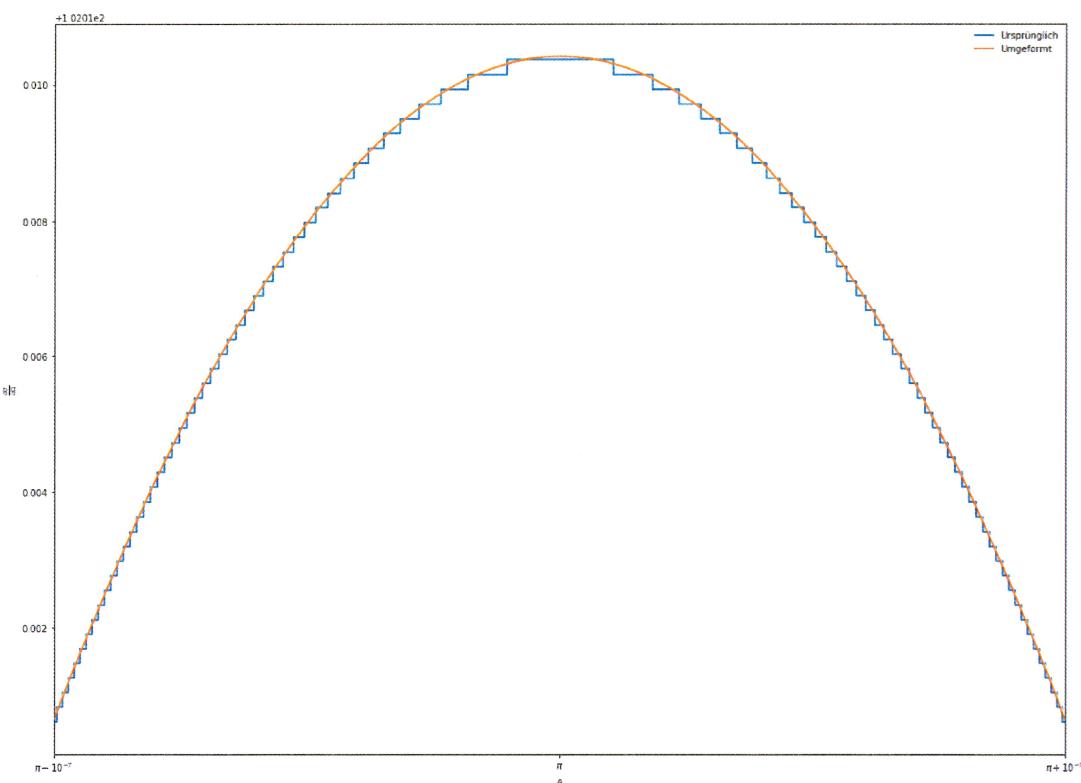
`gamma = E / 511e-6 #big number`

`beta = np.sqrt(1 - 1 / gamma**2) #near 1 for big gamma`

`return alpha **2 / s * (2 + np.sin(theta)**2) / (1 / gamma**2 + beta**2 * np.`

In [11]: `plt.figure(figsize=(20, 15))`

`plt.plot(theta, diff_cross_section(theta), label = 'Ursprünglich', linewidth = 2)`
`plt.plot(theta, diff_cross_section_alternative(theta), label = 'Umgeformt', linewidth = 2)`
`plt.xlim(theta[0], theta[-1])`
`plt.xlabel(r'ϑ')`
`plt.ylabel(r'$\frac{d\sigma}{d\Omega}$')`
`plt.xticks([theta[0], np.pi, theta[-1]], [r'$\pi - 10^{-7}$', 'π', r'$\pi + 10^{-7}$'])`
`plt.legend()`
`plt.show()`



Man sieht, dass die ursprüngliche Funktion instabil um π ist. Durch die Umformung wurde dieses Problem gelöst. ✓

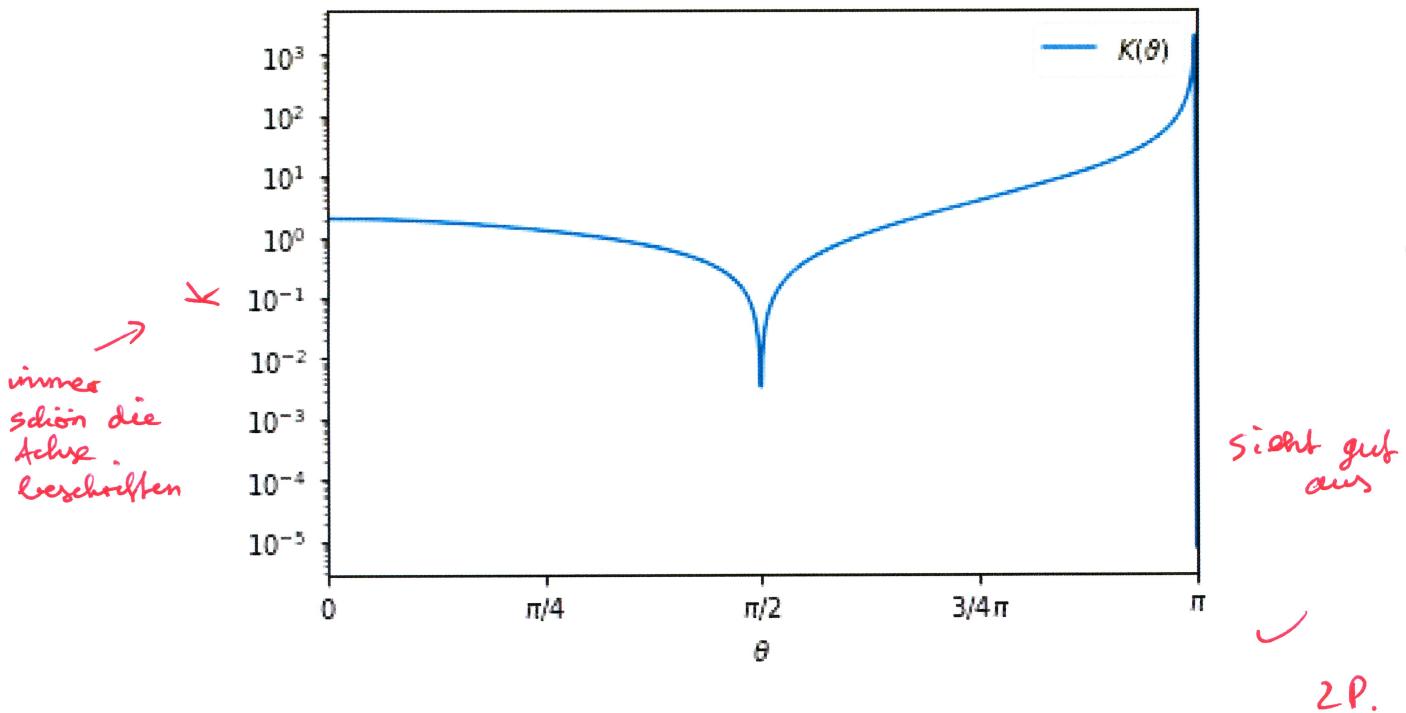
1P.

Die Konditionszahl $K = K(\vartheta)$ ist gegeben durch:

$$K(\vartheta) = \left| \vartheta \frac{\frac{d}{d\vartheta} \left(\frac{d\sigma}{d\Omega} \right)}{\left(\frac{d\sigma}{d\Omega} \right)} \right| = \left| \frac{\vartheta}{2 + \sin^2 \vartheta} \left\{ \frac{\sin \vartheta \cos \vartheta (2 - 6\beta^2)}{1 - \beta^2 \cos^2 \vartheta} \right\} \right|$$

```
In [12]: def K(theta, E = 50):
    gamma = E / 511e-6
    beta = np.sqrt(1 - 1 / gamma**2)
    return abs(theta / (2 + np.sin(theta)**2)
                * np.sin(theta) * np.cos(theta)
                * (2 - 6 * beta**2)
                / (1 - beta**2 * np.cos(theta)**2))
```

```
In [13]: theta = np.linspace(0, np.pi, 1000)
plt.plot(theta, K(theta), label = r'$K(\vartheta)$')
plt.yscale('log')
plt.xlabel(r'$\theta$')
plt.xlim(0, np.pi)
plt.xticks([0, np.pi/4, np.pi/2, 3*np.pi/4, np.pi], ['$0$', '$\pi/4$', '$\pi / 2$'])
plt.legend()
plt.show()
```



Man sieht, dass die Konditionszahl $K(\vartheta)$ für $\vartheta \rightarrow \pi$ gegen große Werte strebt. Die Funktion $\frac{d\sigma}{d\Omega}$ ist für $\vartheta = \pi$ also schlecht konditioniert. Dies deckt sich mit den Beobachtungen aus Aufgabenteil c). Das Problem ist sehr gut für $\theta = \pi/2$ konditioniert.

1 P.

2 Aufgabe 3: Maxwell'sche Geschwindigkeitsverteilung

Die Normierungskonstante ergibt:

$$N = \left(\frac{m}{2\pi k_B T} \right)^{\frac{3}{2}} \checkmark$$

1 P.
Ihr könnt ruhig den Rechenweg dazuschreiben, sonst kann ich keine Teilpunkte geben, wenn das Ergebnis falsch ist

a) Der Wahrscheinlichste Wert v_m ergibt sich aus der Extremalbedingung

$$f'(v) = 0 \quad \checkmark$$

1 P.

zu

$$v_m = \sqrt{\frac{2k_B T}{m}}. \quad \checkmark \quad 1 P.$$

b) Der Mittelwert $\langle v \rangle$ errechnet sich zu

$$\langle v \rangle = \int_0^\infty v f(v) dv = \frac{2v_m}{\sqrt{\pi}}. \quad \checkmark$$

1 P.

c) Für den Median $v_{0.5}$ gilt

$$\frac{1}{2} = \int_0^{v_{0.5}} f(v) dv. \quad \checkmark$$

Dies lässt sich mit der Substitution $u = \frac{v}{v_m}$ umformen:

$$\frac{1}{2} = \frac{4}{\sqrt{\pi}} \int_0^{\frac{v_{0.5}}{v_m}} u^2 \exp(-u^2) du$$

In dieser Darstellung wird ein numerischer Wert für den Quotienten $\frac{v_{0.5}}{v_m}$ bestimmt, sodass der Median für ein beliebiges v_m berechnet werden kann.

In [14]: `import scipy.integrate as integrate`

In [15]: `def integrand(u):`

`return 4 / np.sqrt(np.pi) * u**2 * np.exp(-u**2)`

`v = np.linspace(0, 2, 100000) #discrete sample of possible values for v_{0.5}`
`I = np.array([integrate.quad(integrand, 0, i)[0] for i in v]) #calculate the integ`

`v_05 = v[np.argmin(abs(I - 1/2))]` #search the best matching velocity

`print(f'Der Beste Wert für den Quotienten aus dem obigen Integral lautet {v_05}.')`
`print(f'Der Wert des Integrals beträgt hierfür {integrate.quad(integrand, 0, v_05)}`

Der Beste Wert für den Quotienten aus dem obigen Integral lautet 1.0876508765087651.
Der Wert des Integrals beträgt hierfür 0.4999990551211203.

d) Für $f(v_{FWHM})$ gilt:

$$f(v_{FWHM}) = \frac{f(v_m)}{2} \quad \checkmark$$

$$0 = 2u^2e^{-u^2} - \frac{1}{e}$$

Die Nullstellen können nun numerisch bestimmt werden

In [16]: `from scipy.optimize import brentq` \checkmark sehr schön

```
def f(x):
    return 2 * x **2 *np.exp(-x**2) - 1/np.e

Root_1 = brentq(f, 0.1, 0.5)
Root_2 = brentq(f, 1.5, 2)
FWHM = Root_2-Root_1

print(FWHM)
```

1.1549423602510807

Die Halbwertsbreite beträgt ca. 1.1549 v_m \checkmark

e)

Die Standardabweichung errechnet sich wie folgt:

$$\sigma_v^2 = \int_0^\infty (v - \bar{v})^2 f(v) dv = \int_0^\infty (v^2 - 2v\bar{v} - \bar{v}^2) f(v) dv = \langle v^2 \rangle - 2\langle v \rangle^2 + \langle v \rangle^2 = \langle v^2 \rangle - \langle v \rangle^2 \Leftrightarrow \sigma_v = \sqrt{\langle v^2 \rangle - \langle v \rangle^2}$$

$$\langle v^2 \rangle = \int_0^\infty v^2 f(v) dv = \frac{3}{2}v_m^2$$

$$\sigma_v = \sqrt{\frac{3}{2}v_m^2 - \frac{4v_m^2}{\pi}} = v_m \sqrt{\frac{3}{2} - \frac{4}{\pi}} \quad \checkmark$$

0,5P
+0,5P

2.1 Aufgabe 4

Die Ergebnisse ergeben sich alle durch abzählen der Möglichkeiten

a) $P(w_r + w_b = 9) = \frac{1}{9}$. \checkmark 0,5P

b) $P(w_r + w_b \geq 9) = \frac{10}{36}$. \checkmark 0,5P

c) $P(w_1 = 4, w_2 = 5) = \frac{1}{18}$. \checkmark 0,5 P

d) $P(w_r = 4, w_b = 5) = \frac{1}{36}$. \checkmark 0,5P

e) $P(w_r + w_b = 9 | w_r = 4) = \frac{1}{6}$. \checkmark 0,5 P

f) $P(w_r + w_b \geq 9 | w_r = 4) = \frac{1}{3}$. \checkmark 0,5P

g) $P(w_r = 4, w_b = 5 | w_r = 4) = \frac{1}{6}$. \checkmark 0,5P

- 0,25P

mehrere Bedingungen mit 'and' verbinden

3,75P