blatt_1_abgabe

October 25, 2018

1 SMD-Übungszettel Nr.1

Abgabe von: **Sebastian Pape, Andrej Kusurmann und Steven Becker**

1.1 Aufgabe 1

1.1.1 Aufgabenteil a)

```
In [1]: import numpy as np

        import matplotlib.pyplot as plt

        import scipy

        from scipy.optimize import newton, brentq, minimize
```

Definiere die in der Aufgabe gegeben Funktionen

```
In [2]: def f(x):
            return (x**3 + 1/3) - (x**3 - 1/3)

        def g(x):
            return ( (3 + x**3/3) - (3 - x**3/3)) / x**3
```

Definiere die exakten Werte f_0 und g_0 der Gleichungen:

```
In [3]: f_0 = 2/3
        g_0 = 2/3
```

Führe den relativen Fehler ein

```
In [4]: def rel_error(x_0, x_num):
            return ( np.abs(x_0 - x_num) / np.abs(x_0) )
```

Lege den Definititionsbereich fest und plotte den relativen Fehler

```
In [5]: x = np.logspace(-7,7, 1e6)
```

Ergebnisse für f(x)

```
In [6]: rel_error_array_f = rel_error(f_0, f(x))

        # x_values where the relative error is smaller than 1%
        x_where_rel_err_lower_1per_f = x[rel_error_array_f <= 0.01]

        # Getting the last element of the list because I want to upper limit
        print('f: x value where the relative error is 1%:',int(x_where_rel_err_lower_1per_f[-1]

        # x_values where the relative error is one, this equal to f(x) = 0.
        x_where_rel_err_is_one_f = x[rel_error_array_f == 1]

        # Getting the first element of the list beacause I want the lower limit
        print('f: x value where the relative error is 1, this is equal to f(x) = 0:',int(x_whe
```

```
f: x value where the relative error is 1%: 41284

f: x value where the relative error is 1, this is equal to f(x) = 0: 165140
```

Ergebnis für g(x)

```
In [7]: rel_error_array_g = rel_error(g_0, g(x))

        # x_values where the relative error is smaller than 1%
        x_where_rel_err_lower_1per_g = x[rel_error_array_g <= 0.01]

        # Getting the last element of the list because I want to upper limit
        print('g: x value where the relative error is 1%:',x_where_rel_err_lower_1per_g[0], '\n

        # x_values where the relative error is one, this equal to f(x) = 0.
        x_where_rel_err_is_one_g = x[rel_error_array_g == 1]

        # Getting the first element of the list beacause I want the lower limit
        print('g: x value where the relative error is 1, this is equal to f(x) = 0:',x_where_re
```

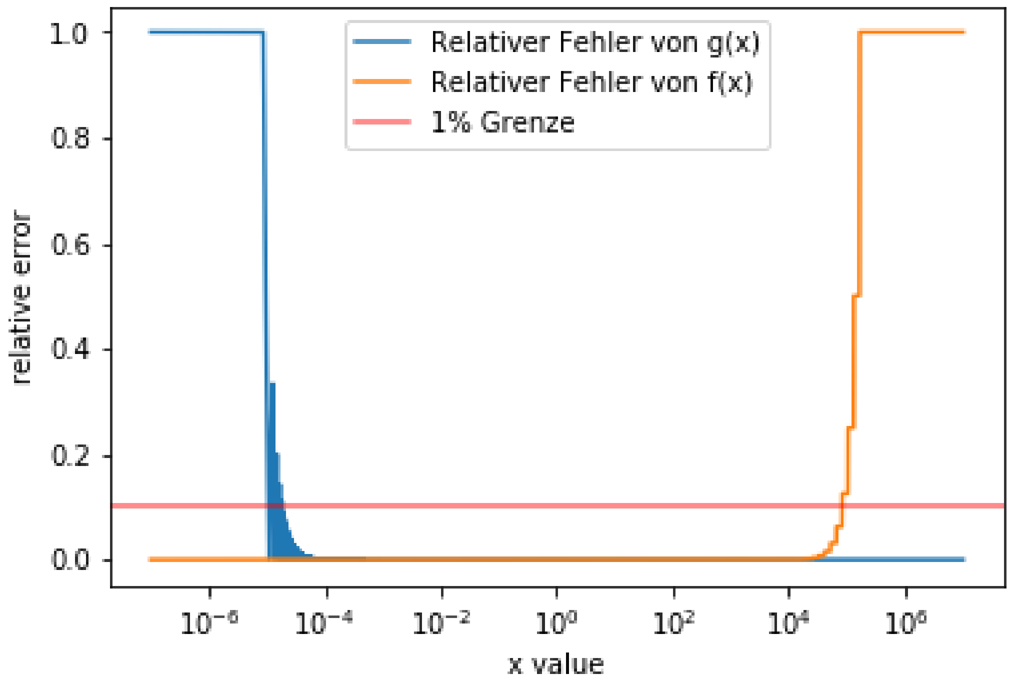
```
g: x value where the relative error is 1%: 1.0967206986851394e-05

g: x value where the relative error is 1, this is equal to f(x) = 0: 8.733452070893184e-06
```

1.2 Aufgabenteil b)

Plotte die relativen Fehler und führe zusätzlich noch die Grenzen mit ein.

```
In [8]: plt.plot(x,rel_error_array_g, label = 'Relativer Fehler von g(x)')
        plt.plot(x,rel_error_array_f, label = 'Relativer Fehler von f(x)')
```

```
plt.axhline(0.1, color = 'r', alpha = 0.6, label = '1% Grenze')

plt.xscale('log')
plt.xlabel('x value')
plt.ylabel('relative error')

plt.legend()

plt.savefig('./results/A1_rel_error.pdf')
plt.show()
```



Was in dem Plot auffällt ist, dass $f(x)$ ungenau für groSSe x ist. Analog ist $g(x)$ ungenau für kleine x.

Erklärung für f(x): Bei groSSen Werten für x vernachlässigt der Computer die $\frac{1}{3}$ und das Resultat ist $f(x) = 0$.

Erklärung für g(x): Bei sehr kleinen Werten für x kommt es mit Zähler von g(x) zur Auslöschung, da x^3 noch kleiner wird. Weiterhin sind die Fluktuation (blau gefüllter Bereich), damit zu erklären, dass in $g(x)$ durch x geteilt wird und somit sich der Term an einer Polstelle aufhält.

1.3 Aufgabe 2

1.3.1 Aufgabenteil a)

Verwende die Werte aus der Aufgabenstellung

In [9]:
```
E = 50 #in units of GeV
me = 511e-6 # in units of GeV
```

```
In [10]: def beta(energy, mass_of_electron):
             gamma = energy / mass_of_electron

             return np.sqrt( 1 - 1/gamma**2)
```

```
In [11]: def nenner(beta, theta):
             return 1 - beta**2 * np.cos(theta)**2
```

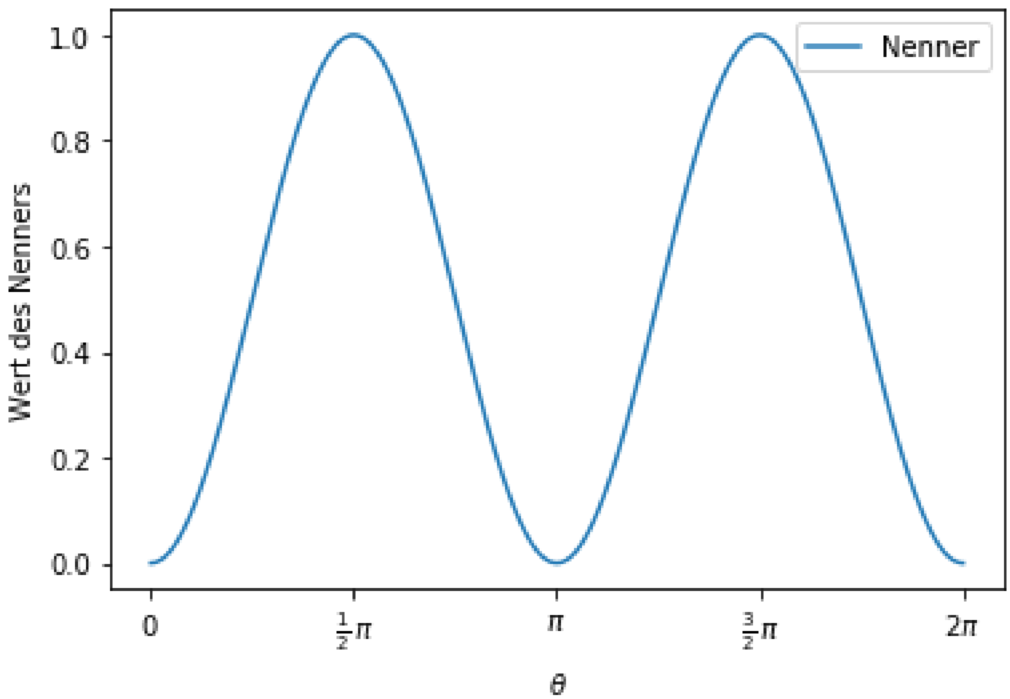
Numerische Instabilität ensteht unter anderem, wenn durch eine kleine Zeit geteilt wird. Aus diesem Grund schaue ich mir den Nenner des differntiellen Wirkungsquerschnitt σ an.

Lege den Definitionsbereich für θ fest.

```
In [12]: theta = np.linspace(0, 2*np.pi, 2e6)
```

```
/home/beckstev/.local/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.py:1: Deprecatio
  """Entry point for launching an IPython kernel.
```

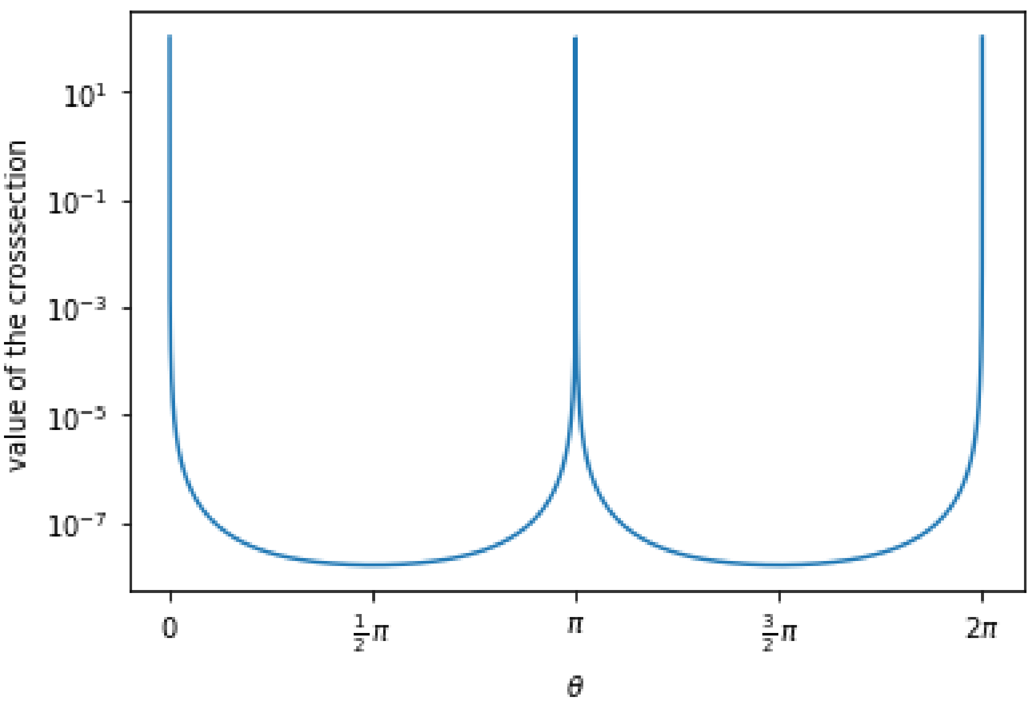
```
In [13]: plt.plot(theta, nenner(beta(E, me), theta),label = 'Nenner')

         plt.xticks([0, np.pi / 2, np.pi, 3 * np.pi / 2,  2 * np.pi],
                    [r"$0$", r"$\frac{1}{2}\,\pi$", r"$\pi$", r"$\frac{3}{2}\,\pi$", r"$2\pi$"]
         plt.ylabel('Wert des Nenners')
         plt.xlabel(r'$\theta$')
         plt.legend()

         plt.savefig('./results/A2_nenner.pdf')
```



In der obigen Grafik ist deutlich zu erkennen dass der Nenner in Umgebung um $\theta = n\pi$, $n \in \mathbb{N}_0$ gegen Null strebt. In der Nähe der Polstellen ist der differentielle Wirkungsquerschnitt σ numerisch instabil. Zusätzlich findet im Nenner eine Subtraktion gleich groSSer Zahlen statt.

```
In [14]: def differential_crosssection(energy, beta, theta):
             alpha = 1/137
             s = 4*energy**2

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

```
In [15]: plt.plot(theta, differential_crosssection(E, beta(E, me), theta))

         plt.xticks([0, np.pi / 2, np.pi, 3 * np.pi / 2,  2 * np.pi],
                    [r"$0$", r"$\frac{1}{2}\,\pi$", r"$\pi$", r"$\frac{3}{2}\,\pi$", r"$2\pi$"]
         plt.xlabel(r'$\theta$')
         plt.ylabel('value of the crosssection')
         plt.yscale('log')

         plt.savefig('./results/A2_plot_crosssection.pdf')
```



In der Grafik oben erkennt man die Auswirkung der Polstellen sehr gut.

1.3.2 Aufgabenteil b)

Ziel ist es den Nenner so umzuformulieren, das dieser numerisch stabiler ist. Verwende dazu

$$
\begin{aligned}
\text{Nenner} &= 1 - \beta^2 \cos^2(\theta) \\
&\text{mit} \\
\cos^2(\theta) &= 1 - \sin^2(\theta) \\
\beta^2 &= 1 - \frac{1}{\gamma^2} \\
&\text{folgt} \\
\text{Nenner} &= 1 - \left(1 - \frac{1}{\gamma^2}\right)\left(1 - \sin^2(\theta)\right) \\
&= \sin^2(\theta) + \frac{1}{\gamma^2} - \frac{1}{\gamma^2}\sin^2(\theta) \\
&= \sin^2(\theta) + \frac{1}{\gamma^2}\cos^2(\theta)
\end{aligned}
$$

Durch den Faktor $\frac{1}{\gamma^2}$ wird aber immer noch bei $\theta = 0, \pi, \ldots$ durch eine sehr kleine Zahl geteilt. Aus diesem Grund erweitere ich σ mit γ

$$
\begin{aligned}
\sigma &\propto \frac{2 + \sin^2(\theta)}{\sin^2(\theta) + \frac{1}{\gamma^2}\cos^2(\theta)} \\
\Leftrightarrow \quad &= \frac{(2 + \sin^2(\theta))\gamma^2}{\gamma^2 \sin^2(\theta) + \cos^2(\theta)}
\end{aligned}
$$

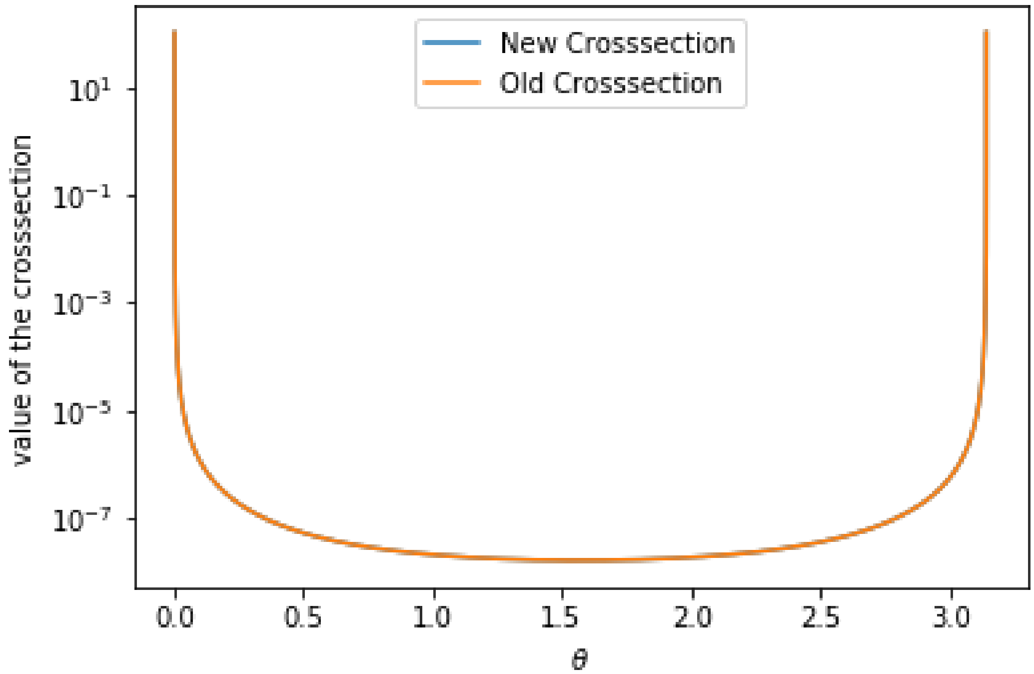
Im Nenner findet nun somit keine Subtraktion gleich groSSer Zahlen statt.

Definiere den *neuen* Wirkungsquerschnitt in `python`.

```
In [16]: def new_differential_crosssection(energy, electron_mass, theta):
             alpha = 1 / 137
             s = 4*energy**2
             gamma = energy / electron_mass

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

```
In [17]: theta = np.linspace(0, np.pi, 1e6)

         plt.plot(theta,new_differential_crosssection(E, me, theta), label = 'New Crosssection
         plt.plot(theta, differential_crosssection(E, beta(E, me), theta), label ='Old Crosssec

         plt.xlabel(r'$\theta$')
         plt.ylabel('value of the crosssection')
         plt.legend()
         plt.yscale('log')
         #plt.savefig('./results/A2_differenz.pdf')
```

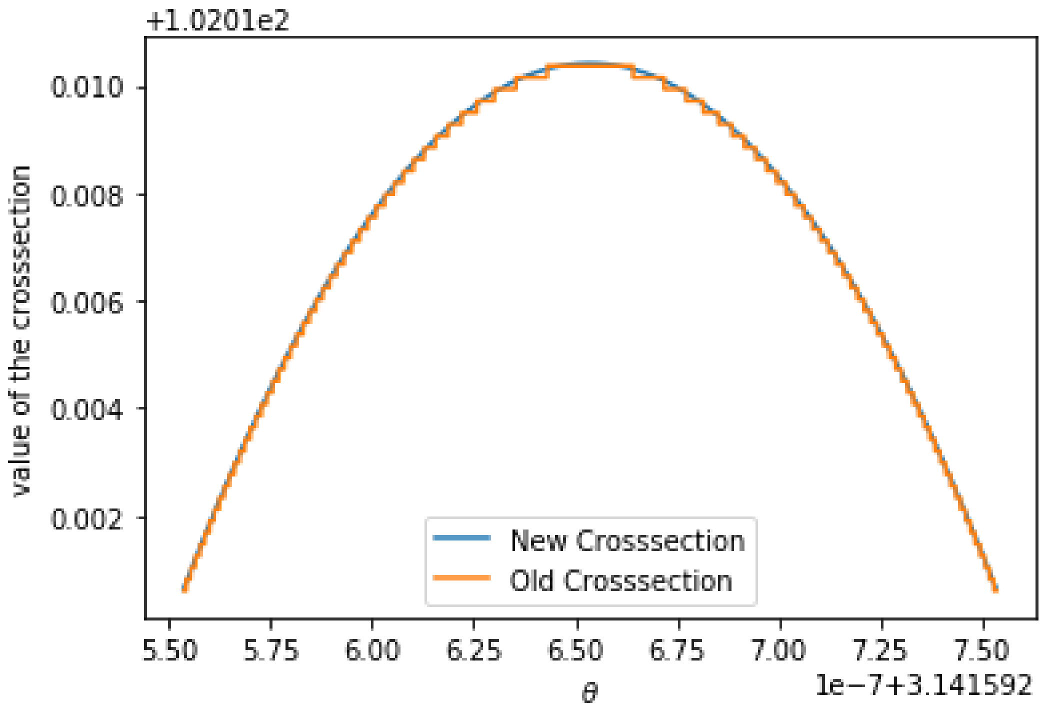
```
/home/beckstev/.local/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.py:1: Deprecatio
  """Entry point for launching an IPython kernel.
```



Auf den ersten Blick zeigen sich keine numerische Effekte.
Erst bei einer sehr genauen Betrachtung der Maxima sind numerische Effekte erkennbar.

```
In [18]: theta = np.linspace(np.pi-1e-7, np.pi+1e-7, 1e6)

         plt.plot(theta,new_differential_crosssection(E, me, theta), label = 'New Crosssection
         plt.plot(theta, differential_crosssection(E, beta(E, me), theta), label ='Old Crossse

         plt.xlabel(r'$\theta$')
         plt.ylabel('value of the crosssection')
         plt.legend()

         plt.savefig('./results/A2_differenz.pdf')
```

```
/home/beckstev/.local/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.py:1: Deprecatio
  """Entry point for launching an IPython kernel.
```



Ergebnis Deutlich zu erkennen sind numerische Effekte, Stufen, bei dem alten Wirkungsquerschnitt. Erst die Stabilisierung von γ führt zu einer glatten Kurve.

1.3.3 Aufgabenteil c)

Die Konditionszahl einer Funktion $f(x)$ ist definiert als:

$$K(x) = \left| x \frac{f'(x)}{f(x)} \right|$$

Aus der Quotientenregel folgt für σ':

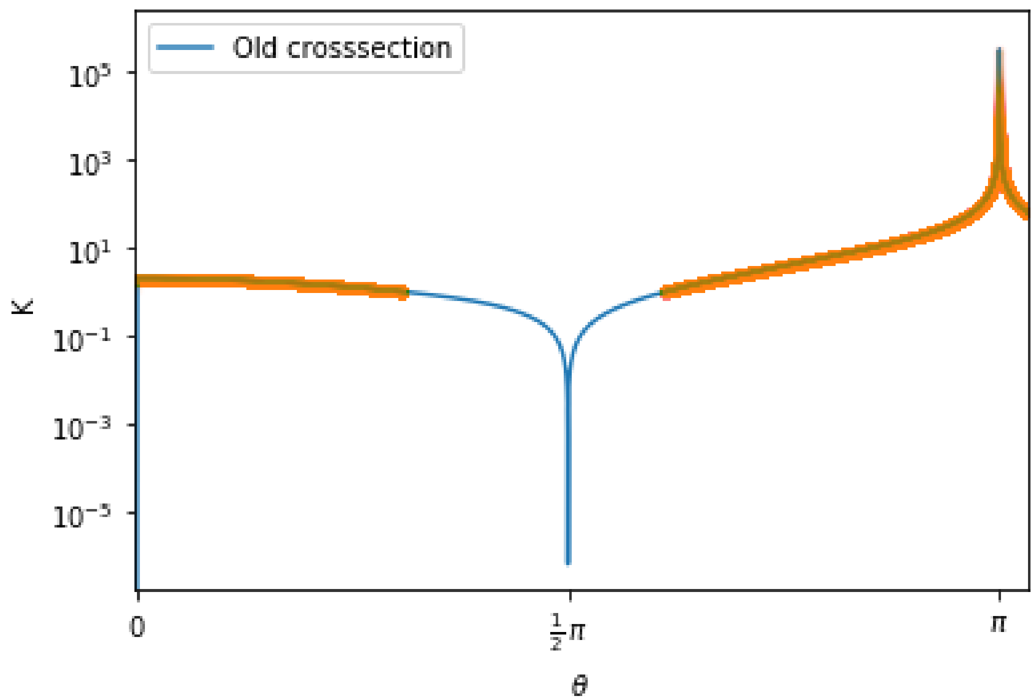
$$\sigma' = \frac{\alpha^2}{s} \frac{2\sin(\theta)\cos(\theta)(-3\beta^2+1)}{(1-\beta^2\cos^2(\theta))^2}$$

```
In [19]: def derivation_differential_crosssection(energy, beta, theta):
             alpha = 1/137
             s = 4 * energy**2

             return alpha**2 / s *  ( (2*np.sin(theta)*np.cos(theta) * (-3*beta**2+1) )/( (1-be
```

```
In [20]: theta = np.linspace(0,2*np.pi, 5e6)
```

```
/home/beckstev/.local/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.py:1: Deprecatio
  """Entry point for launching an IPython kernel.
```

```
In [21]: K = np.abs(theta * derivation_differential_crosssection(E, beta(E,me), theta)/differe


         plt.plot(theta,K, label = 'Old crosssection' )
         plt.plot(theta[K>=1][::2], K[K>=1][::2], '.', alpha=0.01)

         plt.xticks([0, np.pi / 2, np.pi, 3 * np.pi / 2,  2 * np.pi],
                    [r"$0$", r"$\frac{1}{2}\,\pi$", r"$\pi$", r"$\frac{3}{2}\,\pi$", r"$2\pi$"]
         plt.xlim(-0.01,np.pi+0.1)
         plt.xlabel(r'$\theta$')
         plt.ylabel('K')
         plt.yscale('log')
         plt.legend()

         plt.savefig('./results/A2_konditionierung.pdf')
```



Der obige Plot zeigt in orange die Bereiche die eine schlechte Konditionierung respektive $K > 1$ besitzen. Der blaue Bereich ist gut Konditioniert $K \leq 1$.

1.4 Aufgabe 3

Bestimmte zunächst die Nomierung N.

$$\begin{aligned}
1 &= \int_0^\infty N\exp\left(-\frac{mv^2}{2\mathrm{k_B}T}\right)4\pi v^2\mathrm{d}v \\
&= 4N\pi\int_0^\infty \left(\exp\left(-\Gamma v^2\right)v\right)v\mathrm{d}v
\end{aligned}$$

Partielle Integration

$$\begin{aligned}
&= 4N\pi\left(\left[-\frac{1}{2\Gamma}\exp\left(-\Gamma v^2\right)v\right]_0^\infty + \frac{1}{2\Gamma}\int_0^\infty \exp\left(-\Gamma v^2\right)\mathrm{d}v\right) \\
&= N\sqrt{\frac{\pi^3}{\Gamma^3}} \\
\Leftrightarrow \quad N &= \left(\frac{m}{2\mathrm{k_B}T\pi}\right)^{\frac{3}{2}}
\end{aligned}$$

1.4.1 Aufgabenteil a)

Die wahrscheinlichste Geschwindigkeit v_m. Der Wert v_m kann berechnet werden, indem der Hochpunkt der Verteilung $f(v)$ gesucht wird:

$$f'(v_m) \stackrel{!}{=} 0$$

Dazu die Ableitung der Verteilung:

$$f'(v) = 8\pi N\exp\left(-\Gamma v^2\right)v\left(1 - v^2\Gamma\right)$$

Daraus folgt mit der obigen Extremalbedingung für v_m:

$$v_m = \pm\sqrt{\frac{1}{\Gamma}} = \pm\sqrt{\frac{2\mathrm{k_B}T}{m}}$$

Der Definitionsbereich der Geschwindigkeit ist $v \in \mathbb{R}^+$, deshalb wird nur die positive Lösung betrachtet.

1.4.2 Aufgabenteil b)

Der Mittelwert $\langle v\rangle$ Der Mittelwert einer Verteilung kann im allgemein über die erste Kommulante bestimmt werden:

$$\langle v\rangle = \int v\,f(v)\mathrm{d}v$$

$$\begin{aligned}
\langle v\rangle &= 4\pi N\int_0^\infty v^3\exp\left(-\Gamma v^2\right)\mathrm{d}v \\
&= 4\pi N\left(\left[\frac{-v^2}{2\Gamma}\exp(-\Gamma v^2)\right]_0^\infty + \frac{1}{\Gamma}\int_0^\infty v\exp(-\Gamma v^2)\mathrm{d}v\right) \\
&= 4\pi N\left[-\frac{1}{2\Gamma^2}\exp(-\Gamma v^2)\right]_0^\infty \\
&= \frac{2\pi N}{\Gamma^2} = \frac{2}{\sqrt{\pi}}v_m
\end{aligned}$$

1.4.3 Aufgabenteil c)

Bestimmung des Medians $v_{0,5}$. Berechnet werden kann der Median wie folgt:

$$\int_0^{v_{0,5}} f(v)\mathrm{d}v = \frac{1}{2}$$

Die Lösung der folgenden Aufgabe orrientiert sich an diesem Paper.

$$\begin{aligned}
\frac{1}{2} &= \int_0^{v_{0,5}} 4\pi N v^2 \exp\left(-v^2/v_m\right) \mathrm{d}v \\
&= \frac{4}{\sqrt{\pi} v_m} \int_0^{v_{0,5}} \frac{v^2}{v_m^2} \exp\left(-\frac{v^2}{v_m^2}\right) \mathrm{d}v \\
\text{Substitution:} \qquad s &:= \frac{v}{v_m}, \quad v_m \mathrm{d}s = \mathrm{d}v \\
\frac{1}{2} &= \frac{4}{\sqrt{\pi}} \int_0^{s_{0,5}} s^2 \exp\left(-s^2\right) \mathrm{d}s \\
\Leftrightarrow \qquad \frac{\sqrt{\pi}}{8} &= \left[\frac{-s}{2} \exp\left(-s^2\right)\right]_0^{s_{0,5}} + \frac{1}{2} \int_0^{s_{0,5}} \exp\left(-s^2\right) \mathrm{d}s \\
&= \frac{-s_{0,5}}{2} \exp\left(-s_{0,5}^2\right) + \frac{\sqrt{\pi}}{4} \mathrm{erf}\left(s_{0,5}\right) \\
\Rightarrow \qquad g(s_{0,5}) &:= \mathrm{erf}(s_{0,5}) - \frac{2}{\sqrt{\pi}} s_{0,5} \exp\left(-s_{0,5}^2\right) - \frac{1}{2} = 0
\end{aligned}$$

Bestimmte numerisch die Nullstelle der Funktion $g(s_{0,5})$.

```
In [22]: def function(s):
             return scipy.special.erf(s) - 2/np.sqrt(np.pi) * s * np.exp(-s**2) - 1/2

         nullstelle = newton(function,1)

         print('Die Nullstelle der Funktion g liegt bei:', nullstelle)
```

```
Die Nullstelle der Funktion g liegt bei: 1.0876520317581668
```

Mit Hilfe der Nullstelle kann der Median allgemein über die Relation

$$\begin{aligned}
s_{0,5} &= \frac{v_{0,5}}{v_m} \\
\Leftrightarrow \qquad v_{0,5} &= s_{0,5}\, v_m \approx 1{,}088\, v_m
\end{aligned}$$

1.4.4 Aufgabenteil d.)

Bestimmen von v_{FWHM}.

$$\begin{aligned}
\frac{f(v_m)}{2} &= f(v_{\mathrm{FWHM}}) \\
\Leftrightarrow \qquad 0 &= 2\left(\frac{v_{\mathrm{FWHM}}}{v_m}\right)^2 \exp\left(-\left(\frac{v_{\mathrm{FWHM}}}{v_m}\right)^2\right) - \frac{1}{e}
\end{aligned}$$

Substituiere wie in Aufgabenteil c), wie folgt: $u := \frac{v_{FWHM}}{v_m}$ und erhalte damit:

$$2u^2 \exp\left(-u^2\right) - \frac{1}{e} = 0$$

Numerisch kann die Nullstelle durch das Newtonverfahren bestimmt werden. Die halbe Höhe der Verteilung wird zweimal erreicht, weshalb $v_{\text{FWHM, 1}}$ und $v_{\text{FWHM, 2}}$ gesucht werden.

```
In [23]: # u = v_FWHM / v_m
         def f_newton(u):
             return 2 * u**2 * np.exp(-u**2) - 1 / np.e

         v_FWHM_1 = brentq(f_newton, 0, 1)
         v_FWHM_2 = brentq(f_newton, 1, 2)

         print(f"v_FWHM_1 / v_m = {v_FWHM_1}")
         print(f"v_FWHM_2 / v_m = {v_FWHM_2}")
```

```
v_FWHM_1 / v_m = 0.48162324797141126
v_FWHM_2 / v_m = 1.6365656082224955
```

$$v_{\text{FWHM, 1}} \approx 0,48\, v_m$$

$$v_{\text{FWHM, 2}} \approx 1,64\, v_m$$

1.4.5 Aufgabenteil e.)

Bestimmen der Standardabweichung σ.

$$\sigma^2 = \int_0^\infty (v - \bar{v})^2 \cdot f(v) \mathrm{d}v$$

$$\Rightarrow \sigma^2 = \int_0^\infty \frac{4}{\sqrt{\pi}} (v^2 - 2\bar{v}v + \bar{v}) \cdot \frac{v^2}{v_m^3} \exp\left(-\left(\frac{v^2}{v_m}\right)^2\right) \mathrm{d}v$$

$$1.) \int_0^\infty v^2 \cdot f(v) \mathrm{d}v = \frac{3v_m^2}{2} \int_0^\infty f(v) \mathrm{d}v = \frac{3v_m^2}{2}$$

$$2.) \int_0^\infty -2\bar{v}v \cdot f(v) \mathrm{d}v = -2\bar{v}^2 = -\frac{4v_m^2}{\pi}$$

$$3.) \int_0^\infty \bar{v}^2 \cdot f(v) \mathrm{d}v = \bar{v}^2 = \frac{2v_m^2}{\pi}$$

$$\Rightarrow \sigma = \sqrt{v_m^2 \left(\frac{3}{2} - \frac{4}{\pi}\right)} = v_m \sqrt{\left(\frac{3}{2} - \frac{4}{\pi}\right)}$$

1.5 Aufgabe 4

1.5.1 Aufgabenteil a.)

$P(W_{\text{rot}} + W_{\text{blau}} = 9) = P(W_{\text{rot}} = 6 \mid W_{\text{blau}} = 3) + P(W_{\text{rot}} = 3 \mid W_{\text{blau}} = 6) + P(W_{\text{rot}} = 4 \mid W_{\text{blau}} = 5) + P(W_{\text{rot}} = 5 \mid W_{\text{blau}} = 4) = \frac{1}{9}$

1.5.2 Aufgabenteil b.)

$P(W_{\text{rot}} + W_{\text{blau}} \geq 9) = P(W_{\text{blau}} \geq 6 \mid W_{\text{rot}} = 3) + P(W_{\text{blau}} \geq 5 \mid W_{\text{rot}} = 4) + P(W_{\text{blau}} \geq 4 \mid W_{\text{rot}} = 5) + P(W_{\text{blau}} \geq 3 \mid W_{\text{rot}} = 6) = \frac{5}{18}$

1.5.3 Aufgabenteil c.)

$P(W_{\text{rot}} = 4 \mid W_{\text{blau}} = 5) + P(W_{\text{rot}} = 5 \mid W_{\text{blau}} = 4) = \frac{2}{36} = \frac{1}{18}$

1.5.4 Aufgabenteil d.)

$P(W_{\text{rot}} = 4 \wedge W_{\text{blau}} = 5) = \frac{1}{36}$

1.5.5 Aufgabenteil e.)

$P(W_{\text{blau}} = 5 \mid W_{\text{rot}} = 4) = \frac{1}{6}$

1.5.6 Aufgabenteil f.)

$P(W_{\text{rot}} + W_{\text{blau}} \geq 9 | W_{\text{rot}} = 4) = P(W_{\text{blau}} = 5) + P(W_{\text{blau}} = 6) = \frac{1}{3}$

1.5.7 Aufgabenteil g.)

$P(W_{\text{blau}} = 5 \mid W_{\text{rot}} = 4) = \frac{1}{6}$

```
In [ ]:
```