15,25 P.
~~14.75 P~~ / 20 P.

Code läuft durch, braucht aber sehr lange und ist hart kompliziert geschrieben
Versucht den Code (sowie die Kommentare) knapp & effektiv zu halten.
⇒ Notebook von Max N. aus Vorlesung

SMD-Abgabe

1. Übungsblatt

Lars Kolk
lars.kolk@tu-dortmund.de

Julia Sobolewski
julia.sobolewski@tu-dortmund.de

Jannine Salewski
jannine.salewski@tu-dortmund.de

Abgabe: 25.10.2018

TU Dortmund – Fakultät Physik

1 Aufgabe 1

1.1 Teilaufgaben a) und b)

Für die Funktionen

$$f(x) = \left(x^3 + \frac{1}{3}\right) - \left(x^3 - \frac{1}{3}\right) \tag{1}$$

$$g(x) = \frac{\left(3 + \frac{x^3}{3}\right) - \left(3 - \frac{x^3}{3}\right)}{x^3} \tag{2}$$

sollte empirisch ein Bereich bestimmt werden, in denen sie um nicht mehr als 1% abweichen. Dazu wurde die Funktion "check" in Aufgabe01.py geschrieben. Diese nimmt, unter Anderem, einen Anfangs- und Grenzwert und überprüft, ab welchem x die Funktionen um 1% abweichen. Dabei ergaben sich folgende Ausgaben:

Ausgabe des Programms: für (1)

Fuer ein x von 0 bis 180000 mit 0 Dezimalstellen ergeben sich folgende Abweichungen:
x=41286 f(x)=0.65625, Fehler : 1.5625%

Zusaetzlich ergibt sich als Nullstelle:
x=165141f(x)=0.0, Fehler : 100.0%

1.P.

Ausgabe des Programms: für (2)

Fuer ein x von 1 bis 0 mit 8 Dezimalstellen ergeben sich folgende Abweichungen:
x=4.013000000002709e-05 f(x)=0.6734243528892833, Fehler : 1.0136529333924837 %

Zusaetzlich ergibt sich als Nullstelle:
x=8.730000000012339e-06f(x)=0.0, Fehler : 100.0 %

1P.

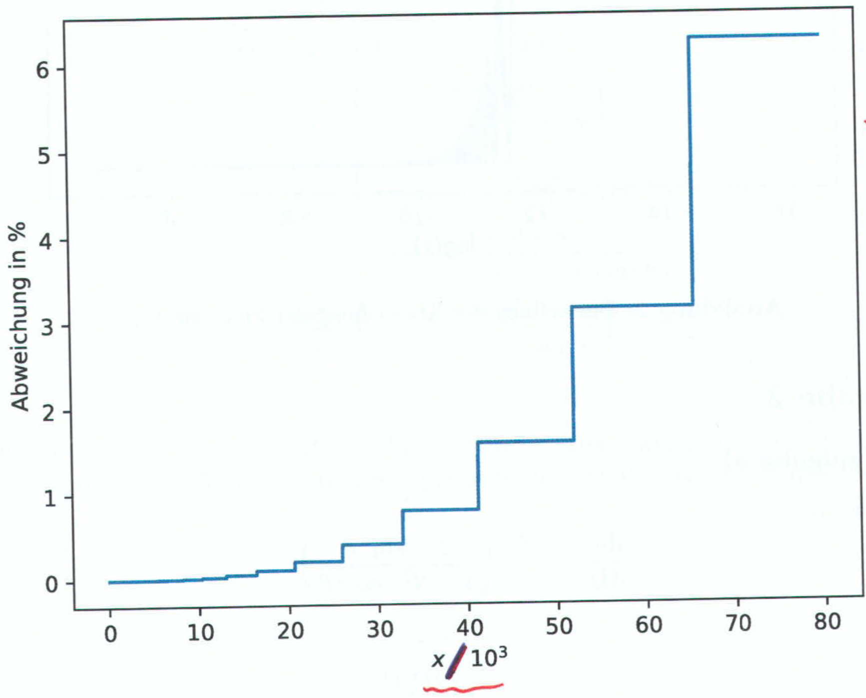
Die zu untersuchenden Bereiche wurden bewusst so gewählt.

- der Term $x^3 \pm \frac{1}{3}$ lässt Abweichungen bei großen x vermuten, da dies bei großen x Rundungsfehler erzeugt. →große ganzzahlige x überprüfen.
- der Term $\frac{1}{x^3}$ lässt Abweichungen nahe 0 vermuten, da so durch kleine Zahlen dividiert wird und dies numerisch nicht stabil ist →kleine x mit möglichst vielen Dezimalstellen überprüfen.

Daher weist (1) für den Bereich von $0 < |x| < 41000$ eine Abweichung von $< 1\%$ auf. (2) ist dagegen für den Bereich von $5 \cdot 10^{-5} < |x| < \infty$ genau.
Wobei strenggenommen ∞ nicht die obere Grenze ist, da es irgendwann zu "NaN"-Fehlern kommt.

1.2 Teilaufgabe c)

Die Plots sind in Abbildung 1 und 2 dargestellt. Diese bestätigen die erwartete Tendenz, die in 1 erläutert wurde, wobei die Abweichung für (2) in 2 unter 10% schlecht zu erkennen sind.



In dem plot sieht man f(x)=0 nicht

Log-Scala und np.logspace() machen es übersichtli

Abbildung 1: Halblogarithmische Darstellung der Abweichung der Funktion (1)

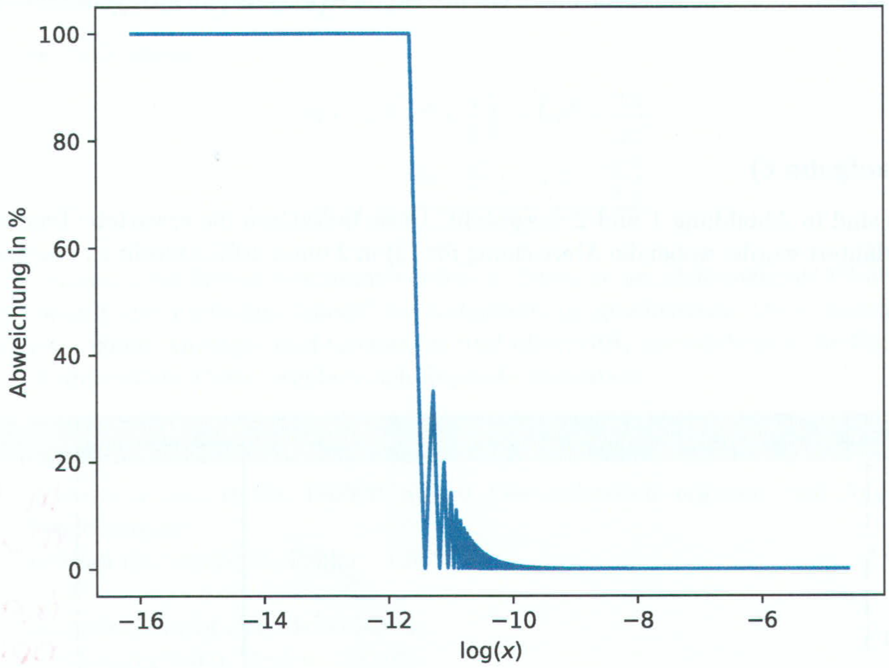


Abbildung 2: Darstellung der Abweichung der Funktion (2)

0.5 P.

2.5 P.

2 Aufgabe 2

2.1 Teilaufgabe a)

Die Gleichung

$$\frac{\mathrm{d}\sigma}{\mathrm{d}\Omega} = \frac{\alpha^2}{s} \cdot \left(\frac{2 + \sin^2\theta}{1 - \beta^2 \cdot \cos^2\theta} \right) \tag{3}$$

mit

$$s = (2E_e)^2$$
$$\beta = \sqrt{1 - \gamma^{-2}}$$
$$\gamma = \frac{E_e}{m_e}$$
$$m_e = 511\,\mathrm{keV}$$

ist numerisch nicht stabil. Dies lässt sich mit der Folge der vielen Operationen begründen. Außerdem lässt sich in der ausgeschriebenden Form

$$\frac{d\sigma}{d\Omega} = \frac{\alpha^2}{4E_e^2} \cdot \left(\frac{2+\sin^2\theta}{1-\left(1-\left(\frac{m_e}{E_2}\right)^2\right)\cdot\cos^2\theta} \right) \tag{4}$$

sehen, dass die Gleichung für den Fall $\theta = n \cdot \pi$, $n \in \mathbb{Z}$ besonders instabil ist , da $\cos^2 \pm\pi = 1$ gilt und somit im Nenner zwei gleichgroße Zahlen voneinander subtrahiert werden. Zusätzlich findet im Fall $E_e = 50\,\text{GeV}$ eine Division durch eine kleine Zahl statt $\left(\frac{m_e}{E_2}^2 \ll 1\right)$, womit die Gleichung numerisch instabil ist. ✓ 1P.

2.2 Teilaufgabe b

$$\frac{\mathrm{d}\sigma}{\mathrm{d}\Omega} = \frac{\alpha^2}{s} \cdot \left(\frac{2+\sin^2\theta}{1-\beta^2\cdot\cos^2\theta} \right) \tag{5}$$

$$= \frac{\alpha^2}{s} \cdot \left(\frac{2+\sin^2\theta}{1-\beta^2+\beta^2\cdot\sin^2\theta} \right) \tag{6}$$

$$= \frac{\alpha^2}{s} \cdot \left(\frac{2+\sin^2\theta}{\gamma^{-2}+(1-\gamma^{-2})\cdot\sin^2\theta} \right) \tag{7}$$

$$= \frac{\alpha^2}{s} \cdot \left(\frac{2+\sin^2\theta}{\gamma^{-2}\cdot(1-\sin^2\theta)+\sin^2\theta} \right) \tag{8}$$

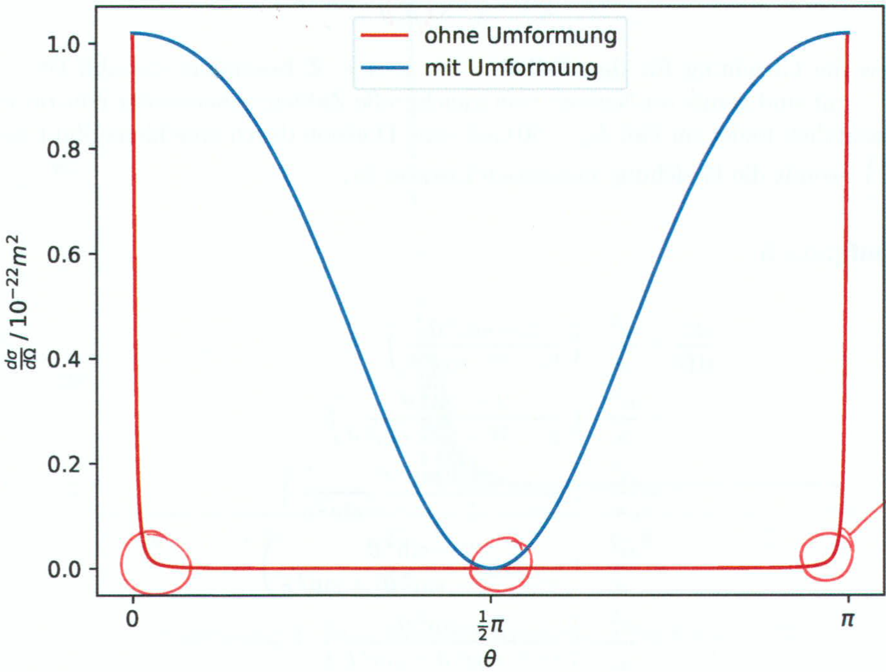
$$= \frac{\alpha^2}{s} \cdot \left(\frac{2+\sin^2\theta}{\gamma^{-2}\cdot\cos^2\theta+\sin^2\theta} \right) \tag{9}$$

$$\tag{10}$$

Das Ergebnis ist numerisch stabiler, da die Substraktion zwei gleich großer Zahlen im Nenner aufgrund der Addition von zwei quadratischen Funktionen nicht mehr auftreten kann. ✓ 2P.

2.3 Teilaufgabe c)

Instabilität wird nicht deutlich.



kritische Bereiche

f siehe code

OP.

Abbildung 3: Darstellung des differentiellen Wirkungsquerschnitts, mit und ohne die Umformung aus 2.2

Wie in 3 zu sehen ist, hebt sich die Kurve, die mit der Gleichung aus (9) berechnet wurde, deutlich von der anderen ab. Statt einem Plateau ist nun eine (grob) parabelförmige Funktion zu sehen, woraus sich schließen lässt, dass dass die neue Funktion (9) numerisch stabiler ist.

2.4 Teilaufgabe d)

Die Konditionszahl ist definiert durch

$$k = \left| \frac{f'(x)}{f(x)} \cdot x \right| \tag{11}$$

$$f(\theta) = \frac{\alpha^2}{s} \cdot \left(\frac{2+\sin^2\theta}{1-\beta^2\cdot\cos^2\theta} \right) \tag{12}$$

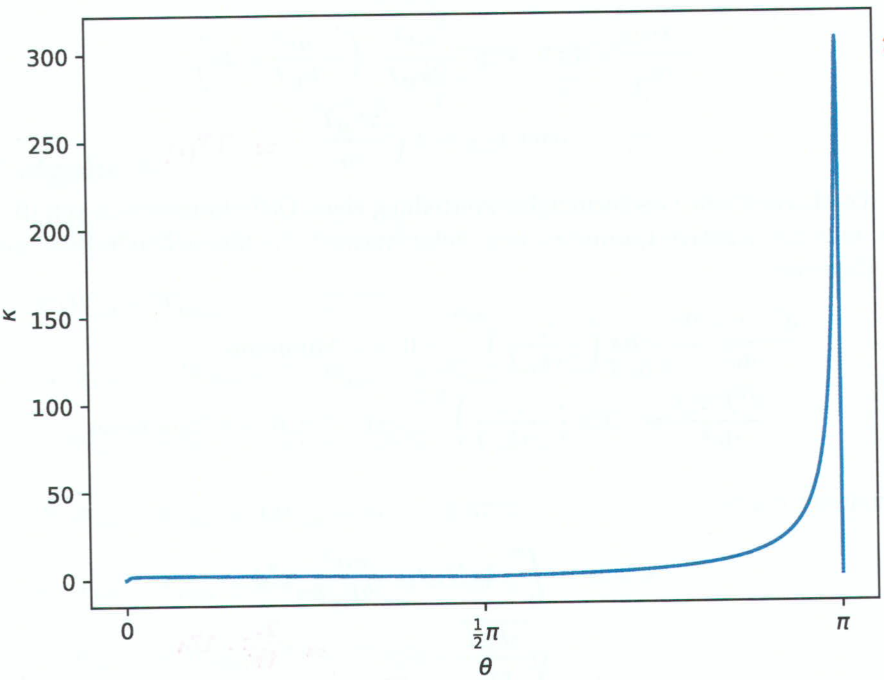
$$f'(\theta) = -\frac{a^2}{s} \cdot \frac{\cos\theta\sin\theta\cdot(3\cancel{\beta}^2-1)\cdot 2}{(-\cancel{\beta}^2\cos^2\theta+1)^2} \tag{13}$$

$$\rightarrow k = \frac{|\,(3\cancel{\beta}^2-1)\,\theta\cdot\sin(2\theta)\,|}{|\,(\sin^2\theta+2)(\beta^2\cdot\cos^2\theta-1)\,|} \tag{14}$$

✓

2P.

2.5 Teilaufgabe e)



log-scale

Der interessante Bereich liegt bei K=1 ⇒ Skalen entsprechend wählen

Abbildung 4: Darstellung der Konditionszahl k aus 2.4

0P. /5P.

Wie in Abbildung 4 zu sehen ist, ist das Problem gut für $0 < x < \pi$ konditioniert.

3 Aufgabe 3

Zunächst normieren:

$$
\begin{aligned}
1 &\overset{!}{=} \int_0^\infty f(v)\mathrm{d}v \\
1 &= \int_0^\infty 4\pi N \cdot \exp\left(-\frac{mv^2}{2k_bT}\right) \cdot v^2 \mathrm{d}v \\
&= 4\pi N \cdot \sqrt{\frac{\pi}{2}} \left(-\frac{mv^2}{m}\right)^{2/3} \\
=> N &= \left(\frac{m}{2\pi k_B T}\right)^{3/2}
\end{aligned}
$$

✓

1P.

1. Wahrscheinlichste Geschwindigkeit $\frac{\mathrm{d}f(v)}{\mathrm{d}v} \overset{!}{=} 0$

0.5P

$$
\begin{aligned}
\frac{\mathrm{d}f(v)}{\mathrm{d}v} &= 4\pi N \cdot \exp - \frac{mv^2}{2k_BT} \cdot \left(-\frac{mv^3}{k_BT} + 2v\right) \\
v_1 = 0 \;oder\; v_{2,3} &= \pm\sqrt{\frac{2k_BT}{m}} = v_m
\end{aligned}
$$

Da die Maxwell'sche Geschwindigkeitsverteilung einen Definitionsbereich von $(0, \infty]$ hat, fällt die negative Lösung v_3 weg. Jetzt nurnoch die hinreichende Bedingung für Extrema:

$$
\begin{aligned}
\frac{\mathrm{d}^2 f(v=0)}{\mathrm{d}v^2} &= 8\pi \left(\frac{m}{2\pi k_BT}\right)^{3/2} > 0 \; => \text{Minimum} \\
\frac{\mathrm{d}^2 f(v_2)}{\mathrm{d}v^2} &= -32\pi \left(\frac{m}{2\pi k_BT}\right)^{3/2} \cdot \exp -1 < 0 \; => \text{Maximum}
\end{aligned}
$$

0.5P.

2. Mittelwert $< v >$

0.5P.

$$
\begin{aligned}
< v > &= \int_0^\infty 4\pi N \exp - \frac{mv^2}{2k_BT} \cdot v^3 \mathrm{d}v \\
&= \sqrt{\frac{8k_BT}{m\pi}} = v_m \quad = \frac{2}{\sqrt{\pi}} \cdot v_1
\end{aligned}
$$

auf dem Zettel wird die wahrscheinlichste Geschw. so genannt, also ungünstig!

0.5P.

3. Median
Hier muss folgende Integrale gelöst werden, aber leider wissen wir nicht, wie wir diese Integrale nummerisch lösen können:

$$
\begin{aligned}
0,5 &\geq \int_0^{v_\text{median}} f(v)\mathrm{d}v \\
0,5 &\geq \int_{v_\text{median}}^\infty f(v)\mathrm{d}v
\end{aligned}
$$

0.5P.

c)

$$\frac{1}{2} = \int_0^{v_{0,5}} f(v)\,dv = \int_0^{v_{0,5}} 4\pi N v^2 e^{-\frac{v^2}{v_m^2}} dv = \frac{4}{\sqrt{\pi}\, v_m} \int_0^{v_{0,5}} \frac{v^2}{v_m^2} e^{-\frac{v^2}{v_m^2}} dv \quad \Big| \; s := \frac{v}{v_m}, \; dv = v_m\, ds$$

8

$$\Rightarrow \frac{\sqrt{\pi}}{8} = \int_0^{s_{0,5}} s^2 e^{-s^2} ds = \frac{-s_{0,5}}{2} e^{-s_{0,5}^2} + \frac{\sqrt{\pi}}{4} \operatorname{erf}(s_{0,5})$$

Umstellen zu $g(s_{0,5}) := \operatorname{erf}(s_{0,5}) - \frac{2}{\sqrt{\pi}} s_{0,5}\, e^{-s_{0,5}^2} - \frac{1}{2} = 0$

Newtonverfahren mit dem Startwert $x_0 = 1$: $s_{0,5} = \frac{v_{0,5}}{v_m} \Leftrightarrow v_{0,5} = s_{0,5} \cdot v_m \approx 1{,}088\, v_m$

d) $\frac{f(v_m)}{2} = f(v_{FWHM})$; $0 = 2\left(\frac{v_{FWHM}}{v_m}\right)^2 e^{-\frac{v_{FWHM}^2}{v_m^2}} - e^{-1}$ | $u := \frac{v_{FWHM}}{v_m}$

$\Rightarrow 0 = 2u^2 e^{-u^2} - \frac{1}{e}$ → BRENTQ-Verfahren (Startintervalle: $x_0 \in [0,1]$, $x_1 \in [1,2]$)

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

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

4. volle Breite, halbe Höhe sehr richtig!!

5. Standardabweichung σ

$$\sigma^2 = <v^2> - <v>^2$$

$$<v^2> = \int_0^\infty f(v)\cdot v^2 \mathrm{d}v$$

$$= 4\pi N \exp - \frac{mv^2}{2k_BT}\cdot v^4 \mathrm{d}v$$

$$= \frac{3k_BT}{m} \checkmark$$

$$\sigma = \sqrt{\frac{k_BT}{m}\cdot\left(3-\frac{8}{\pi}\right)} \checkmark$$

$$= \sqrt{<v>\left(\frac{8\pi}{3}-1\right)} \text{ f}$$

0.75

ihr meint: $<v> \cdot \sqrt{\left(\frac{3\pi}{8}-1\right)}$

wir wollten: $\sqrt{\frac{3\pi-8}{2\pi}}\cdot v_m$

3.75P.

4 Aufgabe 4

a) $P(W_{\text{rot}} + W_{\text{blau}} = 9) = 11{,}\overline{11}\,\%$ ✓

b) $P(W_{\text{rot}} + W_{\text{blau}} \geq 9) = 27{,}78\,\%$ ✓

c) $P(W_{\text{rot}} = 4 \wedge W_{\text{blau}} = 5 \vee W_{\text{rot}} = 5, W_{\text{blau}} = 4) = 5{,}56\,\%$ ✓

d) $P(W_{\text{rot}} = 4 \wedge W_{\text{blau}} = 5) = 2{,}78\,\%$ ✓

e) $P(W_{\text{rot}} + W_{\text{blau}} = 9 | W_{\text{rot}} = 4) = 16{,}67\,\%$ ✓

f) $P(W_{\text{rot}} + W_{\text{blau}} \geq 9 | W_{\text{rot}} = 4) = 33{,}33\,\%$ ✓

g) $P(W_{\text{rot}} = 4 \wedge W_{\text{blau}} = 5 \,|\, W_{rot}=4) = 16{,}67\,\%$ ✓

~~3.5P~~ 4P.

~~Wahrscheinlichkeiten könnt ihr gerne in Brüchen angeben. Halber Punkt Abzug, weil wir die Rechnung sehen wollten, ebenfalls in der Schreibweise P(...)~~

a) $P(W_r + W_b = 9) = \frac{4}{36} = \frac{1}{9}$

b) $P(W_r + W_b \geq 9) = \frac{10}{36} = \frac{5}{18}$

c) $P(W_{r/b} = 4 \wedge W_{b/r} = 5) = \frac{2}{36} = \frac{1}{18}$

d) $P(W_r = 4 \wedge W_b = 5) = \frac{1}{36}$

e) $P(W_r + W_b = 9 | W_r = 4) = P(W_b = 5) = \frac{1}{6}$

f) $P(W_r + W_b \geq 9 | W_r = 4) = \frac{2}{6} = \frac{1}{3}$

g) $P(W_r = 4 \wedge W_b = 5 | W_r = 4) = P(W_b = 5) = \frac{1}{6}$

Code fuer Blatt01

Kolk, Sobolewski, Salewski

26. Oktober 2018

../B/7/Blatt01_Kolk_Sobolewski_Salewski/Aufgabe01.py

```
import numpy as np
import time
import matplotlib.pyplot as plt
import datetime

#Funktion f der Aufgabe
def f(x):
    return (x**3+1/3)-(x**3-1/3)

#Funktion g der Aufgabe
def g(x):
    #Damit bei x=0 kein Fehler geworfen wird, wird bei x=0 der Wert 2/3 zurückgegeben
    try:
        return ((3+x**3/3)-(3-x**3/3))/x**3
    except ZeroDivisionError:
        return 2/3

#Funktion g der Aufgabe, jedoch ohne Exception
def g2(x):
    return ((3+x**3/3)-(3-x**3/3))/x**3

#Das Argument der zu untersuchenden Funktion für die Fälle StartwertYGrenze und umgekehrt. Wurde hier definiert um Korrekturen einfacher zu machen.
def argument(x, dezimal, start, grenze):
    if start < grenze:
        return x*10**(-dezimal)+start
    else:
        return -x*10**(-dezimal)+start


#Funktion zum empirschen Testen der Funktionen. Es wird angegeben, welche Funktionen bis wohin (grenze) getestet werden soll, wie viele verschiedene Abweichungen erfasst und auf wie viele Dezimalstellen überprüft werden solls.
#Funktioniert für positive start und grenzwerte, negative nicht getestet
def check(start, grenze, funktion, dezimal, abweichungen):
    #Misst die benötigte Zeit bzw startet die Uhr zur Messung
    start_t = time.clock()
    #Öffnen einer Textdatei zum Speichern der Ergebnisse. Das, was in den log geschrieben wird, wird vorher in "string" gespeichertund dann mit datalog.write in die Datei geschrieben. Kenne leider keinen schöneren Weg.
    datalog=open("datalog.txt", "a")
    datalog.write(str(datetime.datetime.now()))
    datalog.write(": \n \n")
    #Negative und nicht ganzzahlige Dezimalstellen werden hier abgefangen
    if dezimal <0 or dezimal%1 !=0:
        print("Die Anzahl der Dezimalstellen muss größer als 0 und eine ganze Zahl sein!")
        return 0
    #Erstellung eines Array in dem Werte gespeichert werden
    Wert=[2/3]*(abweichungen+1)
    gefunden=0
    string="Fuer ein x von " + str(start) + " bis "+ str(grenze)+ " mit " +str( dezimal)+ " Dezimalstellen ergeben sich folgende Abweichungen: \n"
    datalog.write(string)
    #Hier wird der Durclauf gestartet. Die Range wird mit den Dezimalzahlen angepasst, da z.B. ein Testdurchlauf von 0 bis 100 mit einer Dezimalstelle 1000 Durchläufe braucht und nicht 100.
    for x in range(abs((max(start, grenze)+max(-start, -grenze))*10**dezimal)):
```

```python

        #Wenn eine Abweichungen  von 1% gefunden und die gewünschte Zahl von verschieden hohen Abweichungen noch nicht gefunden wurde....
        if Fehler(funktion, argument(x, dezimal, start, grenze))>0.01 and gefunden!=abweichungen and Wert[gefunden] != funktion(argument(x, dezimal, start, grenze)):
            #....wird der Wert mit dem letzten Wert abgeglichen und ggf ausgegeben
            string="x=" + str(argument(x, dezimal, start, grenze)) + "  f(x)=" + str(funktion(argument(x, dezimal, start, grenze)) ) +", Fehler : " +str(Fehler(funktion,argument(x, dezimal, start, grenze))*100)+ "% \n"
            datalog.write(string)
            gefunden+=1
            Wert[gefunden]=funktion(argument(x, dezimal, start, grenze))

        #Wenn die gewünschte Anzahl von Abweichungen gefunden wurde, wird überprüft, ob die Nullstelle schon dabei war oder nicht. Wird leider jedes mal kontrolliertself.
        if gefunden==abweichungen and len(Wert)>np.count_nonzero(Wert) :
                #wenn nicht: kommentarlos beenden
                break

        elif funktion(argument(x, dezimal, start, grenze))==0 and gefunden == abweichungen:
            #wenn doch: Ausgabe
            datalog.write("\nZusaetzlich ergibt sich als Nullstelle: \n")
            string="x=" + str( argument(x, dezimal, start, grenze)) + "f(x)=" + str(funktion(argument(x, dezimal, start, grenze))) + ", Fehler : " + str( Fehler(funktion, argument(x, dezimal, start, grenze))*100) + "% \n"
            datalog.write(string)
            break


    string="\nUnterschiedliche Abweichungen gesucht: " + str(abweichungen) + "\nUnterschiedliche Abweichungen gefunden: " + str(gefunden) +"\n"
    datalog.write(string)
    #gibt Zeit aus
    string="Dauer= " +str( time.clock()-start_t) + " s \n \n \n"
    datalog.write(string)

#Berechnet die Abweichung
def Fehler(funktion, x):
    return abs(1-funktion(x)*1.5)


#Lässt die Funktion check für f bzw g durchlaufen. f wird von 0-180000 kontrolliert, g von 1-0.
#Bei f werden nur ganzzalige x kontrolliert, bei g wird bis zur 8. Dezimalstelle gerechnet.
check(0, 180000, f, 0, 1)
check(1, 0, g, 8, 1)
#print (Fehler(g, 8.730000000012339e-06))

#
x=np.linspace(1e-7, 0.01, 500000)
x2=np.linspace(1e-10, 80*10**3, 500000)
y=Fehler(f, x2)*100
z=Fehler(g2, x)*100

# Der Fehler g wird von e-7 bis 0.01 geplottet und die x-Achse dabei logarithmiert.
plt.plot(np.log(x), z)
plt.xlabel(r'$\log{(x)}$')
plt.ylabel(r'Abweichung in %')

plt.savefig("Fehler1b.pdf")
plt.clf()

#Der Fehler von f wird von 0 bis 180000 geplottet.
plt.plot(x2*10**(-3), y)
plt.xlabel(r'$x \: \cdot \: 10^3$')
plt.ylabel('Abweichung in %')
plt.savefig("Fehler1a.pdf")
```

benutzt: np.logspace()

benutzt plt.xscale('log')

../B/7/Blatt01_Kolk_Sobolewski_Salewski/Aufgabe02.py

```python
import numpy as np
import matplotlib.pyplot as plt
```

```python
#Definition des diff. Wirkungsquerschnitts ohne Umformung
def quer_V1(x):
    return a**2/s*((2+np.sin(x)**2)/(1-b**2*np.cos(x)**2))

#Definition des diff. Wirkungsquerschnitts mit Umformung
def quer_V2(x):
    return a**2/s*((2+np.sin(x)**2)/(gamma**(-2)*np.cos(x)**2+np.sin(x)**2))

#Konditionszahl
def konditionsquer(x):
    return abs(((3*b**2-1)*x*np.sin(2*x))/((np.sin(x)**2+2)*(b**2*np.cos(x)**2-1)))
#Zusammenhänge aus der Aufgabe
E=50*10**9
m=511*10**~~6~~3
a=1/137
gamma=E/m
s=(2*E)**2
b=np.sqrt(1-gamma**(-2))

#die beiden diff. Wirkungsquerschnitte werden geplottet.
x=np.linspace(0, np.pi, 5000)
plt.plot(x, quer_V1(x)*10**(22), 'r-', label='ohne Umformung')
plt.plot(x, quer_V2(x)*10**(22), 'b-', label='mit Umformung')
plt.xlabel(r'$\theta$')
#x-Scala wird in pi angegeben
plt.ylabel(r'$\frac{d \sigma}{d \Omega} \: / \:10^{-22} m^2$')
plt.xticks([0, np.pi / 2, np.pi],
           [r"$0$", r"$\frac{1}{2}\pi$", r"$\pi$"])
plt.legend()
plt.savefig("diffquerschnitt.pdf")

#Plot wird geleert
plt.clf()

#Konditionszahl wird geplottet
plt.plot(x, konditionsquer(x))
plt.xlabel(r'$\theta$')
plt.ylabel(r'$\kappa$')
plt.xticks([0, np.pi / 2, np.pi],
           [r"$0$", r"$\frac{1}{2}\pi$", r"$\pi$"])
plt.savefig('Konditionierungszahl.pdf')
```

../B/7/Blatt01_Kolk_Sobolewski_Salewski/aufgabe04.py

```python
import numpy as np

print("\n")

# a)/b)/c)/d)

rot = [1, 2, 3, 4, 5, 6]      # Werte, die der rote Würfel annehmen kann
blau = [1, 2, 3, 4, 5, 6]     # Werte, die der blaue Würfel annehmen kann

treffer_a = 0     # Variable, die die Anzahl der Treffer in a) zählen soll
treffer_b = 0     # Variable, die die Anzahl der Treffer in b) zählen soll
treffer_c = 0     # Variable, die die Anzahl der Treffer in c) zählen soll
treffer_d = 0     # Variable, die die Anzahl der Treffer in d) zählen soll

for xx in rot:
    for yy in blau:
        if yy + xx == 9:                                 # Berechnung der Treffer für a)
            treffer_a += 1
        if yy + xx >= 9:                                 # Berechnung der Treffer für b)
            treffer_b += 1
        if (xx == 4 and yy == 5) or (yy == 4 and xx == 5):  # Berechnung der Treffer für c)
            treffer_c += 1
        if xx == 4 and yy == 5:                          # Berechnung der Treffer für d)
            treffer_d += 1

daten = [treffer_a, treffer_b, treffer_c, treffer_d]
```

```python
texte = ["a) Wahrscheinlichkeit, dass die Summe der Punkte 9 ergibt:",
         "b) Wahrscheinlichkeit, dass die Summe der Punkte 9 oder mehr ergibt:",
         "c) Wahrscheinlichkeit, dass ein Wuerfel 4, der andere 5 Punkte zeigt:",
         "d) Wahrscheinlichkeit, dass der rote Wuerfel 4, der blaue 5 Punkte zeigt:"]

for data, text in zip(daten, texte):            # Berechnung und Ausgabe der Wahrscheinlichkeit für den jeweiligen Aufgabenteil
    data = data/(len(rot)*len(blau))*100
    print(text, "%0.2f %%" % data)

print("\n")

# e)/f)/g)

rot = [4]                          # Werte, die der rote Würfel annehmen kann
blau = [1, 2, 3, 4, 5, 6]          # Werte, die der blaue Würfel annehmen kann

treffer_e = 0     # Variable, die die Anzahl der Treffer in e) zählen soll
treffer_f = 0     # Variable, die die Anzahl der Treffer in f) zählen soll
treffer_g = 0     # Variable, die die Anzahl der Treffer in g) zählen soll

for xx in rot:
    for yy in blau:
        if yy + xx == 9:               # Berechnung der Treffer für e)
            treffer_e += 1
        if yy + xx >= 9:               # Berechnung der Treffer für f)
            treffer_f += 1
        if xx == 4 and yy == 5:        # Berechnung der Treffer für g)
            treffer_g += 1

daten = [treffer_e, treffer_f, treffer_g]
texte = ["e) Wahrscheinlichkeit, dass die Summe der Punkte 9 ergibt:",
         "f) Wahrscheinlichkeit, dass die Summe der Punkte 9 oder mehr ergibt:",
         "g) Wahrscheinlichkeit, dass ein Wuerfel 4, der andere 5 Punkte zeigt:"]

for data, text in zip(daten, texte):            # Berechnung und Ausgabe der Wahrscheinlichkeit für den jeweiligen Aufgabenteil
    data = data/(len(rot)*len(blau))*100
    print(text, "%0.2f %%" % data)
```