Blatt1

May 11, 2020

1 Aufgabe 1 Numerische Stabilität

(a)()=(3+1/3)(31/3) und (b)()=((3+3/3)(33/3))/3 Bestimmen Sie empirisch, für welche Bereiche von (grob) das numerische Ergebnis vom algebraischen um nicht mehr als 1% abweicht, gleich Null ist.

a) algebraisches Ergebnis: $f(x) = \frac{2}{3} \forall x$

```
In [1]: import matplotlib.pyplot as plt
        import numpy as np
```

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


        Abweichung =(2/3-f(100000))/(2/3) #für verschiedene Werte für x und damit f(x) muss di
        #ich habe einfach mal verschiedene x-Werte eingesetzt und geschaut, wann die Abweichun
        #Es gibt bestimmt ein Befehl, der einem den genauen x-Wert herausgibt, bei dem die Abw
        #den habe ich nicht gefunden :(


        print(Abweichung)
```

```
-0.12500000000000006
```

Für $x > 10^5$ beträgt die Abweichung mehr als 1% und für $x \to \infty$ wird das numerische Ergebnis 0

b) algebraisches Ergebnis: $f(x) = \frac{2}{3} \forall x$

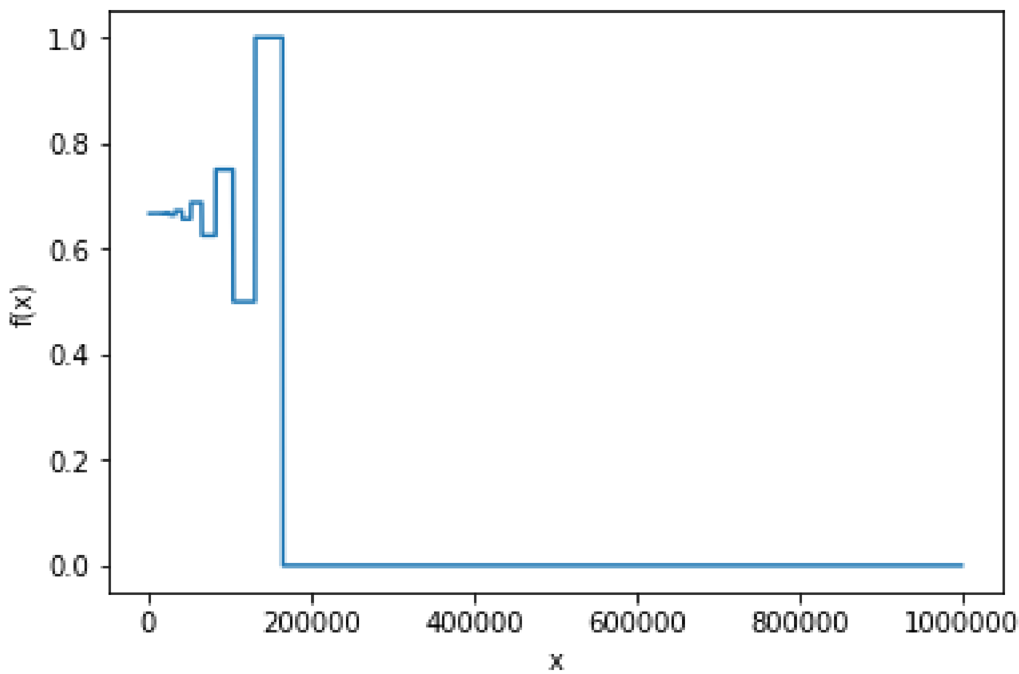
```
In [3]: def g(x):
            return ((3+x**3/3)-(3-x**3/3))/x**3

        Abweichung =(2/3-g(0.00001))/(2/3)

        print(Abweichung)
```

```
-0.33226762955018757
```

Für $x < 10^{-5}$ beträgt die Abweichung mehr als 1% und für $x \to 0$ wird das numerische Ergebnis 0

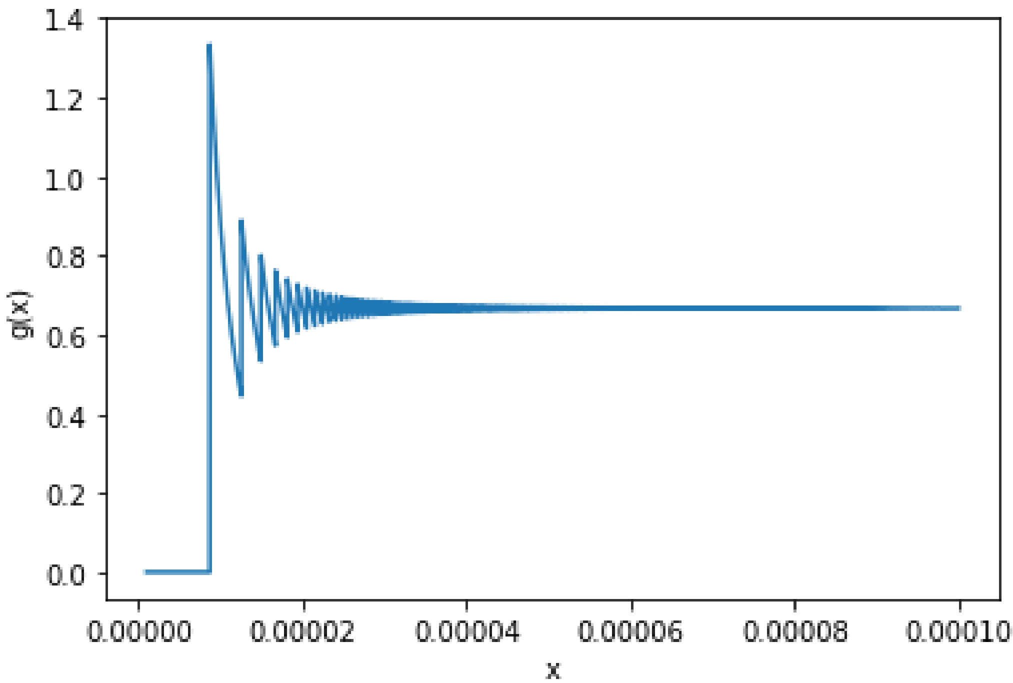
c) Stellen Sie das Ergebnis in geeigneter Form graphisch dar (d.h. z.B. logarithmische-Skala)

```
In [4]: x = np.geomspace(1000, 1000000, 10000)  #logarithmische Darstellung durch logspace hat
        plt.plot(x, f(x))
        plt.ylabel(r'f(x)')
        plt.xlabel(r"x")
        plt.show()
```



Für $x \to \infty$ wird f(x) null. Allerdings gibt es auch schon bei kleineren x-Werten groSSe Abweichungen zum algebraischen Wert. (Warum?)

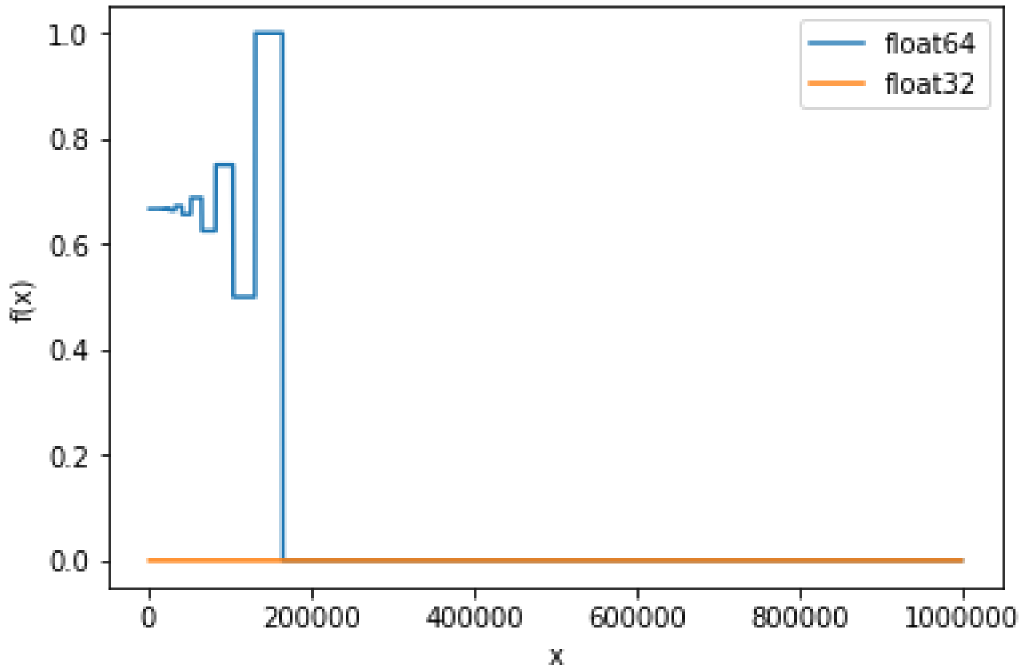
```
In [5]: x = np.geomspace(10**-6, 10**-4, 10000)
        plt.plot(x, g(x))
        plt.ylabel(r'g(x)')
        plt.xlabel(r"x")
        plt.show()
```



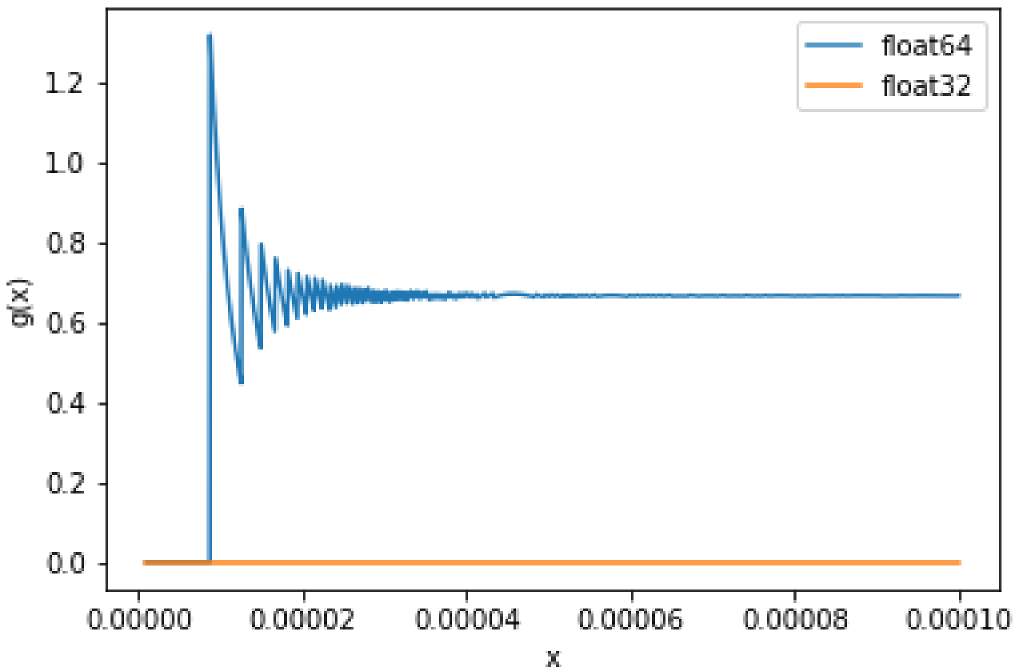
Hier ist anschaulich dargestellt, dass g(x) für $x \to 0$ null wird. Wie bei f(x) sind hier auch bei gröSSer werdenden x-Werten Abweichungen zu sehen.

d) Wie ändert sich die Darstellung, wenn Sie die Datenpunkt mit dem Datentyp float32 bzw. float64 erstellen?

In [6]:
```
x_32 = np.geomspace(1000, 1000000, 10000, dtype='float32')
x_64 = np.geomspace(1000, 1000000, 10000, dtype='float64')

plt.plot(x_64, f(x_64), label = 'float64')
plt.plot(x_32, f(x_32), label = 'float32')

plt.ylabel(r'f(x)')
plt.xlabel(r"x")
plt.legend(loc="best")
plt.show()
```




```
In [7]: x_32 = np.geomspace(10**-6, 10**-4, 1000, dtype='float32')
        x_64 = np.geomspace(10**-6, 10**-4, 1000, dtype='float64')

        plt.plot(x_64, g(x_64), label = 'float64')
        plt.plot(x_32, g(x_32), label = 'float32')

        plt.ylabel(r'g(x)')
        plt.xlabel(r"x")
        plt.legend(loc="best")
        plt.show()
```



Zu d) es ist zu sehen, dass die Abweichung von f(x) bzw. g(x) zum algebraischen Wert beim Datentyp float32 gröSSer ist.

2 Aufgabe 2 Numerische Stabilität und Kondition

Der Ausdruck f(E,) stellt einen Summanden des differentiellen Wirkungsquerschnitts für die Reak- tion e e + dar und ist gegeben durch f(E,) = 2 + sin 2 . 1 2 cos 2 mit = 1 2 , = E e m e (m e = 511 keV).

(a) Ist diese Gleichung für f(E,) numerisch stabil? In welchem Bereich von ist die Gleichung für E e = 50 GeV numerisch instabil?

Was sorgt für Instabilität? 1. Subtraktion gleich groSSer Zahlen 2. Division durch kleine Zahlen 3. Multiplikation mit groSSen Zahlen

2 + sinš: hier kann keine Unstabilität auftreten.

š: Um zu berechnen muss zunächst berechnet werden. Hier kann Punkt 2 verletzt werden, wenn E >> m ist. Dann ist groSS und somit besteht bei der Berechnung von kein Problem. Ist jedoch ungefähr 1 (also E ungefähr m), werden bei der Berechnung von gleich groSSe Zahlen subrahiert, was Regel 1 verletzt.

šcosš: und cos sind beide durch 1 beschränkt und somit kann Regel 3 nicht verletzt werden. 1-šcosš: Regel 1 wird verletzt, wenn sowohl als auch der Betrag von cos ca. 1 sind. Dies ist der Fall, wenn E >> m ist oder wenn 0 oder π ist. In diesem Fall würde auch der Zähler klein werden, was Regel 1 verletzen würde.

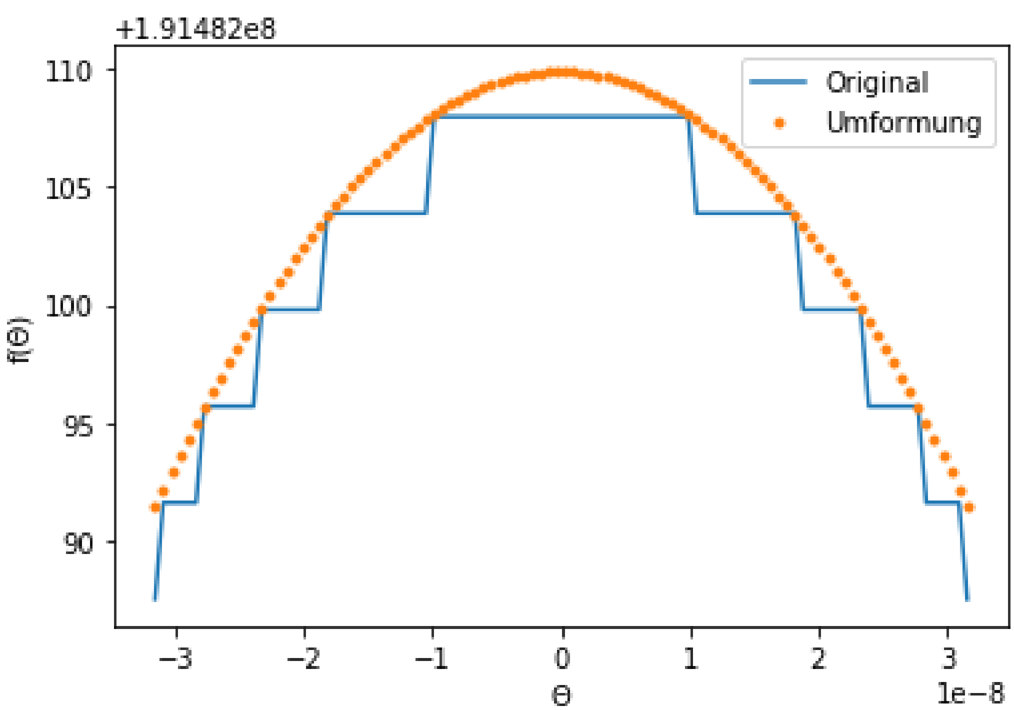
Ist E = 50GeV wird ca. 1. Somit ist die Gleichung instabil, wenn der zuletzt beschriebene Fall eintritt und im Bereich von 0 oder π ist.

(b) Beheben Sie die Stabilitätsprobleme durch eine geeignete analytische Umformung. (Hinweis: Nutzen Sie 1   2 = 1 / 2 und 1 = sin 2 + cos 2)

$\frac{2+sin\theta}{1-\beta cos\theta} = \frac{2+sin\theta}{sin\theta+cos\theta-\beta cos\theta} = \frac{2+sin\theta}{sin\theta+1/\gamma cos\theta}$

(c) Zeigen Sie, dass Sie die Stabilitätsprobleme behoben haben, indem Sie beide Gleichungen im kritischen Intervall darstellen.

```
In [8]: import matplotlib.pyplot as plt
        import numpy as np

        #da nur der betrag von sin bzw cos entscheident ist wird nur der bereich um 0 dargeste
         = np.linspace(-10**-7.5,10**-7.5,100)
         = 5e9/511e3
         = np.sqrt(1-**(-2))


        plt.plot(, (2+np.sin()**2)/(1- **2*np.cos()**2), label="Original")
        plt.plot(, (2+np.sin()**2)/(1/(**2)*np.cos()**2+np.sin()**2),".", label="Umformung",)
        plt.ylabel("f()")
        plt.xlabel("")
        plt.legend()

        None
```



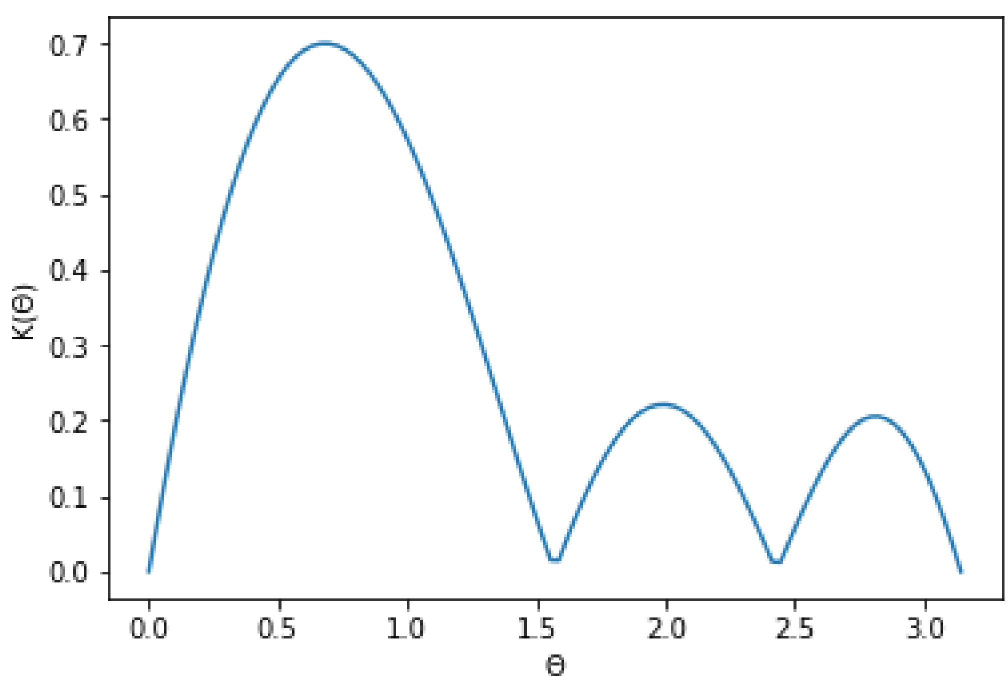
Da durch die Umformung im Nenner keine Differenz gleich groSSer Zahlen ist, sondern eine Summe von jeweils sehr kleinen Zahlen, ist der Graph "glatter", da nicht auf Grund der Rundung Nachkommastellen in verschiedenen GröSSenordungen abgeschnitten werden. Dadurch kommt es bei der nicht umgeformten Formel zu Ungenauigkeiten.

(d) Berechnen Sie die Konditionszahl. Wie hängt diese von ab?

Aus der Vorlesung: $K = |x\frac{f'(x)}{f(x)}|$
$f'(\theta) = \frac{2cos(\theta)sin(\theta)}{1-\beta cos(\theta)} - \frac{2\beta cos(\theta)sin(\theta)}{(1-\beta cos(\theta))}$
$K = |\frac{\theta}{2+sin\theta} \cdot (2cos\theta sin\theta - \frac{2\beta cos(\theta)sin(\theta)}{1-\beta cos(\theta)}|$

(e) Stellen Sie den Verlauf der Konditionszahl als Funktion von (0   ) grafisch dar. In welchem Bereich ist das Problem gut bzw. schlecht konditioniert?

```
In [9]: import matplotlib.pyplot as plt
        import numpy as np

         = np.linspace(0, np.pi ,100)
         = 5e9/511e3
         = np.sqrt(1-**(-2))

        plt.plot(, np.abs(/( 2 + np.sin()**2 )* ( 2* np.cos() * np.sin()) - (2 * **2 * np.cos(
        plt.ylabel("K()")
        plt.xlabel("")

        None
```



Da die Konditionszahl überall kleiner 1 ist, wird die Fehlerfortpflanzung im gesamten Intervall gedämpft. Damit ist das Problem gut konditioniert.

(f) Was ist der Unterschied zwischen Stabilität und Kondition?

Die Stabilität beschreibt, wie gut eine Lösung bei nicht exakter Rechnung ist. Durch umformen der Funktion kann die Stabilität verbessert werden.

Die Kondition beschreibt, wie sich Fehler der Anfangswerte fortpflanzen. Sie kann für ein bestehendes Problem (Funktion) nicht verbessert werden.