Blatt 1

Aufgabe 1a), b) und c)

Es werden die zwei Funktionen a)

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

und b)

$$g(x) = \frac{(3 + \frac{x^3}{3}) - (3 - \frac{x^3}{3})}{x^3}$$

empirisch untersucht. Dabei soll festgestellt werden, für welche Bereiche von x das numerische Ergebnis vom algebraischen um nicht mehr als 1% abweicht und in welchen Bereichen das Ergebnis gleich null ist. Anschließend gibt es eine graphische Auswertung.

In [1]:

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

In [2]:

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

In [3]:

```
def g(x): #Erzeugen der zweiten Funktion
    return ((3+(x**3)/3) - (3 - (x**3)/3))/x**3
```

In [4]:

```
def interval(a): #Algorithmus zur Bestimmung und Ausgabe der Intervalle für Zahl
en aus dem Bereich der natürlichen Zahlen
    if len(a) != 0:
        new = np.arange(0, len(a), 1)
        index1 = (a[new-1]+1 != a[new]) #Durch diese Bedingung werden alle Start
werte eines Intervalls bestimmt
        index2= np.append(index1[1:], index1[0])#Durch die Indexverschiebung um
 1 nach links, wird das Intervall der Endwerte bestimmt
        a1= a[index1]
        a2= a[index2]
        for i in range(len(a1)):
            if (a1[i]==a2[i]): #Wenn der Start und Endwert derselbe ist, reicht
 es, wenn man ihn einmal printed, die Liste ist auch so schon zu lang
                print("(", a1[i], ")","; ", sep="", end="")
            else:
                print("(", a1[i],",", a2[i], ")","; ", sep="", end="")
        print("")
    else: print("Leeres Array.")
```

In [5]:

```
def makeplot(func, fig, c): #Funktion, Nummer der figure und Potenz des Definitionsbereichs kann eingegeben werden
    x= np.linspace(1, 10**c, 10**c) #Definitionsbereich wird definiert
    y= func(x) #y-Werte werden definiert
    y0= 2/3* np.ones(len(x)) #"exakter Wert" 2/3
    x1= x[(y==2/3)] #numerische Werte mit "keiner" Abweichung
    x2= x[(y>((2/3)+0.01)) | (y<(2/3 - 0.01))] #numerische Abweichung unter einem Prozent
    x3= x[(y<((2/3)+0.01)) & (y>(2/3 - 0.01)) & (y!=2/3)] #numerische Abweichung über einem Prozent
    plt.figure(fig, figsize=(8,6))
    plt.xscale("log") #Logarithmierte Skala
    plt.plot(x, y0, "b", label="Tatsächlicher Wert")
    plt.plot(x1, func(x1), "k.", label="Numerische Abweichung von 0")
    plt.plot(x2, func(x2), "g.", label="Abweichung größer als 1%", linestyle="None")
    plt.plot(x3, func(x3), "r.", label="Abweichung weniger als 1%", linestyle="None")
    plt.legend()
    #Ausgabe der Werte mittels der Interval-Funktion
    print("Tatsächlicher Wert für Formel", fig,": ", end="")
    interval(x1)
    print("\n\n")
    print("Abweichung größer als 1% für Formel", fig,": ", end="")
    interval(x2)
    print("\n\n")
    print("Abweichung weniger als 1% für Formel", fig, ": ", end="")
    interval(x3)
    print("\n\n")
```

In [6]:

```
makeplot(f, 1, 6) #Ausführung für Funktion f
makeplot(g, 2, 3) #Ausführung für Funktion g, Potenz kleiner gewählt, da die Int
ervalle sonst die ganze Datei überschwemmen.
```

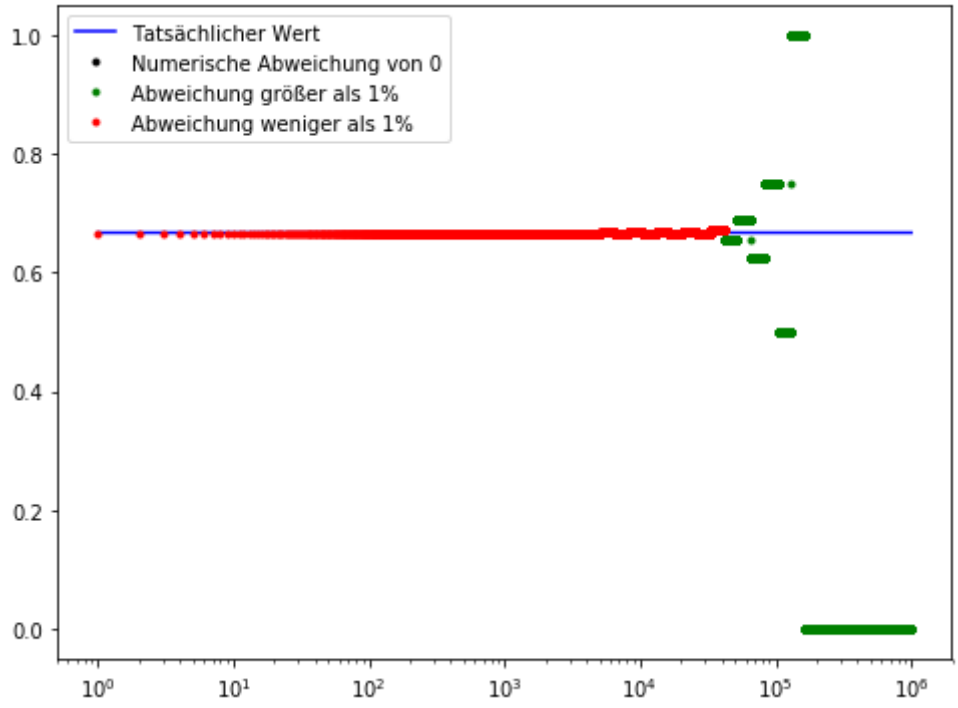
```
Tatsächlicher Wert für Formel 1 : Leeres Array.


Abweichung größer als 1% für Formel 1 : (41286.0,1000000.0);


Abweichung weniger als 1% für Formel 1 : (1.0,41285.0);


Tatsächlicher Wert für Formel 2 : (3.0,10.0); (12.0); (14.0,16.0);
(18.0); (20.0,21.0); (23.0,25.0); (27.0,28.0); (30.0); (32.0,33.0);
(35.0,37.0); (39.0,40.0); (42.0,43.0); (45.0,48.0); (50.0,51.0); (5
3.0,54.0); (56.0,57.0); (60.0,61.0); (63.0,64.0); (66.0,67.0); (69.
0,70.0); (72.0,75.0); (77.0,78.0); (80.0,81.0); (83.0,84.0); (86.0,8
7.0); (89.0,90.0); (92.0,94.0); (96.0,97.0); (99.0,100.0); (102.0,10
3.0); (105.0,106.0); (108.0,109.0); (111.0,112.0); (114.0,115.0); (1
17.0); (119.0,120.0); (122.0,123.0); (125.0,126.0); (128.0,129.0);
(131.0,132.0); (134.0,135.0); (137.0,138.0); (140.0,141.0); (143.0,1
44.0); (146.0,148.0); (150.0,151.0); (153.0,154.0); (156.0,157.0);
(159.0,160.0); (162.0,163.0); (165.0,166.0); (168.0,169.0); (171.0,1
72.0); (174.0,175.0); (177.0,178.0); (180.0,181.0); (183.0,186.0);
(188.0,189.0); (191.0,192.0); (194.0,195.0); (197.0,198.0); (200.0,2
01.0); (203.0,204.0); (206.0,207.0); (209.0,210.0); (212.0,213.0);
(215.0,216.0); (218.0,219.0); (221.0,222.0); (224.0,225.0); (227.0,2
28.0); (230.0,231.0); (234.0,235.0); (237.0,238.0); (240.0,241.0);
(243.0,244.0); (246.0,247.0); (249.0,250.0); (252.0,253.0); (255.0,2
56.0); (258.0,259.0); (261.0,262.0); (264.0,265.0); (267.0,268.0);
(270.0,271.0); (273.0,274.0); (276.0,277.0); (279.0,280.0); (282.0,2
83.0); (285.0,286.0); (288.0,289.0); (291.0,292.0); (294.0); (296.0,
297.0); (299.0,300.0); (302.0,303.0); (305.0,306.0); (308.0,309.0);
(311.0,312.0); (314.0,315.0); (317.0,318.0); (320.0,321.0); (323.0,3
24.0); (326.0,327.0); (329.0,330.0); (332.0,333.0); (335.0,336.0);
(338.0,339.0); (341.0,342.0); (344.0,345.0); (347.0,348.0); (350.0,3
51.0); (353.0,354.0); (356.0,357.0); (359.0,360.0); (362.0,363.0);
(365.0,366.0); (368.0,370.0); (372.0,373.0); (375.0,376.0); (378.0,3
79.0); (381.0,382.0); (384.0,385.0); (387.0,388.0); (390.0,391.0);
(393.0,394.0); (396.0,397.0); (399.0,400.0); (402.0,403.0); (405.0,4
06.0); (408.0,409.0); (411.0,412.0); (414.0,415.0); (417.0,418.0);
(420.0,421.0); (423.0,424.0); (426.0,427.0); (429.0,430.0); (432.0,4
33.0); (435.0,436.0); (438.0,439.0); (441.0,442.0); (444.0,445.0);
(447.0,448.0); (450.0,451.0); (453.0,454.0); (456.0,457.0); (459.0,4
60.0); (462.0,463.0); (465.0); (467.0,468.0); (470.0,471.0); (473.0,
474.0); (476.0,477.0); (479.0,480.0); (482.0,483.0); (485.0,486.0);
(488.0,489.0); (491.0,492.0); (494.0,495.0); (497.0,498.0); (500.0,5
01.0); (503.0,504.0); (506.0,507.0); (509.0,510.0); (512.0,513.0);
(515.0,516.0); (518.0,519.0); (521.0,522.0); (524.0,525.0); (527.0,5
28.0); (530.0,531.0); (533.0,534.0); (536.0,537.0); (539.0,540.0);
(542.0,543.0); (545.0,546.0); (548.0,549.0); (551.0,552.0); (554.0,5
55.0); (557.0,558.0); (560.0,561.0); (563.0,564.0); (566.0,567.0);
(569.0,570.0); (572.0,573.0); (575.0,576.0); (578.0,579.0); (581.0,5
82.0); (584.0,585.0); (588.0,589.0); (591.0,592.0); (594.0,595.0);
(597.0,598.0); (600.0,601.0); (603.0,604.0); (606.0,607.0); (609.0,6
10.0); (612.0,613.0); (615.0,616.0); (618.0,619.0); (621.0,622.0);
(624.0,625.0); (627.0,628.0); (630.0,631.0); (633.0,634.0); (636.0,6
37.0); (639.0,640.0); (642.0,643.0); (645.0,646.0); (648.0,649.0);
(651.0,652.0); (654.0,655.0); (657.0,658.0); (660.0,661.0); (663.0,6
64.0); (666.0,667.0); (669.0,670.0); (672.0,673.0); (675.0,676.0);
(678.0,679.0); (681.0,682.0); (684.0,685.0); (687.0,688.0); (690.0,6
```

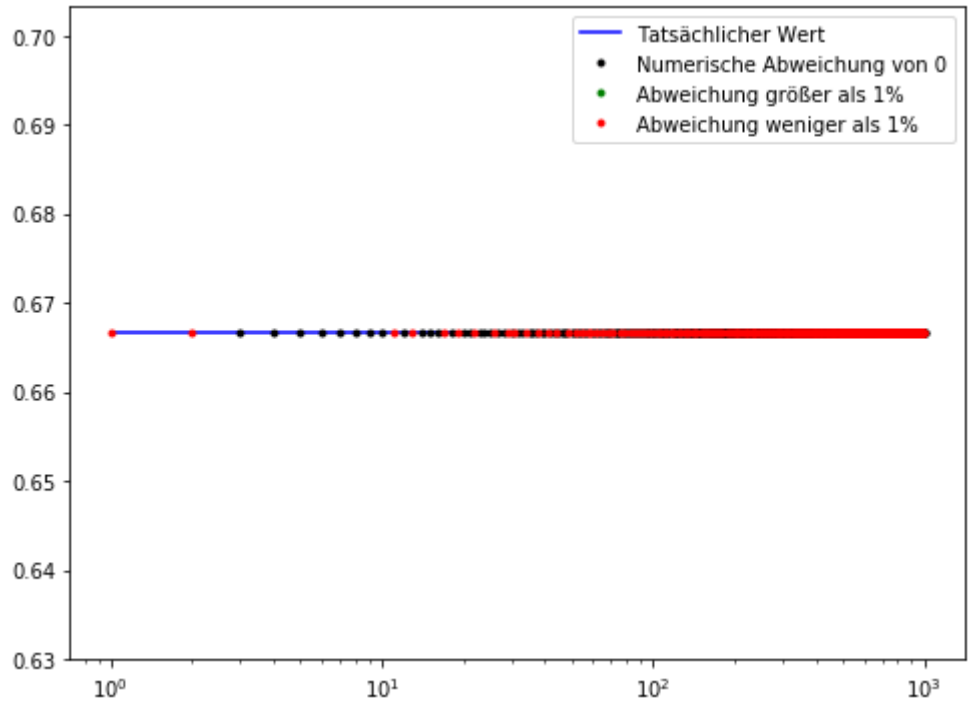
91.0); (693.0,694.0); (696.0,697.0); (699.0,700.0); (702.0,703.0);
(705.0,706.0); (708.0,709.0); (711.0,712.0); (714.0,715.0); (717.0,7
18.0); (720.0,721.0); (723.0,724.0); (726.0,727.0); (729.0,730.0);
(732.0,733.0); (735.0,736.0); (738.0); (740.0,741.0); (743.0,744.0);
(746.0,747.0); (749.0,750.0); (752.0,753.0); (755.0,756.0); (758.0,7
59.0); (761.0,762.0); (764.0,765.0); (767.0,768.0); (770.0,771.0);
(773.0,774.0); (776.0,777.0); (779.0,780.0); (782.0,783.0); (785.0,7
86.0); (788.0,789.0); (791.0,792.0); (794.0,795.0); (797.0,798.0);
(800.0,801.0); (803.0,804.0); (806.0,807.0); (809.0,810.0); (812.0,8
13.0); (815.0,816.0); (818.0,819.0); (821.0,822.0); (824.0,825.0);
(827.0,828.0); (830.0,831.0); (833.0,834.0); (836.0,837.0); (839.0,8
40.0); (842.0,843.0); (845.0,846.0); (848.0,849.0); (851.0,852.0);
(854.0,855.0); (857.0,858.0); (860.0,861.0); (863.0,864.0); (866.0,8
67.0); (869.0,870.0); (872.0,873.0); (875.0,876.0); (878.0,879.0);
(881.0,882.0); (884.0,885.0); (887.0,888.0); (890.0,891.0); (893.0,8
94.0); (896.0,897.0); (899.0,900.0); (902.0,903.0); (905.0,906.0);
(908.0,909.0); (911.0,912.0); (914.0,915.0); (917.0,918.0); (920.0,9
21.0); (923.0,924.0); (926.0,927.0); (929.0,931.0); (933.0,934.0);
(936.0,937.0); (939.0,940.0); (942.0,943.0); (945.0,946.0); (948.0,9
49.0); (951.0,952.0); (954.0,955.0); (957.0,958.0); (960.0,961.0);
(963.0,964.0); (966.0,967.0); (969.0,970.0); (972.0,973.0); (975.0,9
76.0); (978.0,979.0); (981.0,982.0); (984.0,985.0); (987.0,988.0);
(990.0,991.0); (993.0,994.0); (996.0,997.0); (999.0,1000.0);

Abweichung größer als 1% für Formel 2 : Leeres Array.

Abweichung weniger als 1% für Formel 2 : (1.0,2.0); (11.0); (13.0);
(17.0); (19.0); (22.0); (26.0); (29.0); (31.0); (34.0); (38.0); (41.
0); (44.0); (49.0); (52.0); (55.0); (58.0,59.0); (62.0); (65.0); (6
8.0); (71.0); (76.0); (79.0); (82.0); (85.0); (88.0); (91.0); (95.
0); (98.0); (101.0); (104.0); (107.0); (110.0); (113.0); (116.0); (1
18.0); (121.0); (124.0); (127.0); (130.0); (133.0); (136.0); (139.
0); (142.0); (145.0); (149.0); (152.0); (155.0); (158.0); (161.0);
(164.0); (167.0); (170.0); (173.0); (176.0); (179.0); (182.0); (187.
0); (190.0); (193.0); (196.0); (199.0); (202.0); (205.0); (208.0);
(211.0); (214.0); (217.0); (220.0); (223.0); (226.0); (229.0); (232.
0,233.0); (236.0); (239.0); (242.0); (245.0); (248.0); (251.0); (25
4.0); (257.0); (260.0); (263.0); (266.0); (269.0); (272.0); (275.0);
(278.0); (281.0); (284.0); (287.0); (290.0); (293.0); (295.0); (298.
0); (301.0); (304.0); (307.0); (310.0); (313.0); (316.0); (319.0);
(322.0); (325.0); (328.0); (331.0); (334.0); (337.0); (340.0); (343.
0); (346.0); (349.0); (352.0); (355.0); (358.0); (361.0); (364.0);
(367.0); (371.0); (374.0); (377.0); (380.0); (383.0); (386.0); (389.
0); (392.0); (395.0); (398.0); (401.0); (404.0); (407.0); (410.0);
(413.0); (416.0); (419.0); (422.0); (425.0); (428.0); (431.0); (434.
0); (437.0); (440.0); (443.0); (446.0); (449.0); (452.0); (455.0);
(458.0); (461.0); (464.0); (466.0); (469.0); (472.0); (475.0); (478.
0); (481.0); (484.0); (487.0); (490.0); (493.0); (496.0); (499.0);
(502.0); (505.0); (508.0); (511.0); (514.0); (517.0); (520.0); (523.
0); (526.0); (529.0); (532.0); (535.0); (538.0); (541.0); (544.0);
(547.0); (550.0); (553.0); (556.0); (559.0); (562.0); (565.0); (568.
0); (571.0); (574.0); (577.0); (580.0); (583.0); (586.0,587.0); (59
0.0); (593.0); (596.0); (599.0); (602.0); (605.0); (608.0); (611.0);
(614.0); (617.0); (620.0); (623.0); (626.0); (629.0); (632.0); (635.
0); (638.0); (641.0); (644.0); (647.0); (650.0); (653.0); (656.0);
(659.0); (662.0); (665.0); (668.0); (671.0); (674.0); (677.0); (680.
0); (683.0); (686.0); (689.0); (692.0); (695.0); (698.0); (701.0);

```
(704.0); (707.0); (710.0); (713.0); (716.0); (719.0); (722.0); (725.
0); (728.0); (731.0); (734.0); (737.0); (739.0); (742.0); (745.0);
(748.0); (751.0); (754.0); (757.0); (760.0); (763.0); (766.0); (769.
0); (772.0); (775.0); (778.0); (781.0); (784.0); (787.0); (790.0);
(793.0); (796.0); (799.0); (802.0); (805.0); (808.0); (811.0); (814.
0); (817.0); (820.0); (823.0); (826.0); (829.0); (832.0); (835.0);
(838.0); (841.0); (844.0); (847.0); (850.0); (853.0); (856.0); (859.
0); (862.0); (865.0); (868.0); (871.0); (874.0); (877.0); (880.0);
(883.0); (886.0); (889.0); (892.0); (895.0); (898.0); (901.0); (904.
0); (907.0); (910.0); (913.0); (916.0); (919.0); (922.0); (925.0);
(928.0); (932.0); (935.0); (938.0); (941.0); (944.0); (947.0); (950.
0); (953.0); (956.0); (959.0); (962.0); (965.0); (968.0); (971.0);
(974.0); (977.0); (980.0); (983.0); (986.0); (989.0); (992.0); (995.
0); (998.0);
```





Zu 1)

Somit ist zu erkennen, dass die erste Funktion für keinen Wert dem exakten Wert $\frac{2}{3}$ entspricht, für das Intervall zwischen 1 und 41286 bei unter einem Prozent Abweichung liegt und ab diesem Wert über einem Prozent bis zum Ende des Definitionsbereichs.

Die zweite Funktion ist deutlich stabiler. Bei ihr hat kein Wert eine Abweichung über einem Prozent. Die Werte unter einem Prozent springen für verschiedene Intervalle zwischen exakt $\frac{2}{3}$ und unter einem Prozent hin und her, wie man an der langen Liste der Intervalle erkennen kann.

Aufgabe 2a)

Es ist ein Term des differektiellen Wirkungsquerschnitts für die Reaktion $\mathrm{e^-e^+} \rightarrow \gamma\gamma$ gegeben und es soll bestimmt werden, ob dieser numerisch stabil ist. Außerdem soll der Bereich von θ bestimmt werden, in dem die Gleichung für $\mathrm{E_e} = 50\,\mathrm{GeV}$ numerisch instabil ist.

Der Ausdruck ist numerisch nicht stabil, da es, wenn θ Werte nahe π annimmt, im Nenner zu einer Subtraktion zweier fast gleich großer Zahlen kommt, was immer mit großen Rundungsfehlern behaftet ist.

b)

Das Stabilitätsproblem soll durch eine geeignete analytische Umformung gelöst werden.

Mithilfe der angegebenen Umformungen kann man den Term umschreiben zu:

$$\frac{d\sigma}{d\Omega} = \frac{\alpha^2}{s}\left(\frac{2+\sin^2(\theta)}{\sin^2(\theta)+\frac{1}{\gamma}\cos^2(\theta)}\right)$$

c)

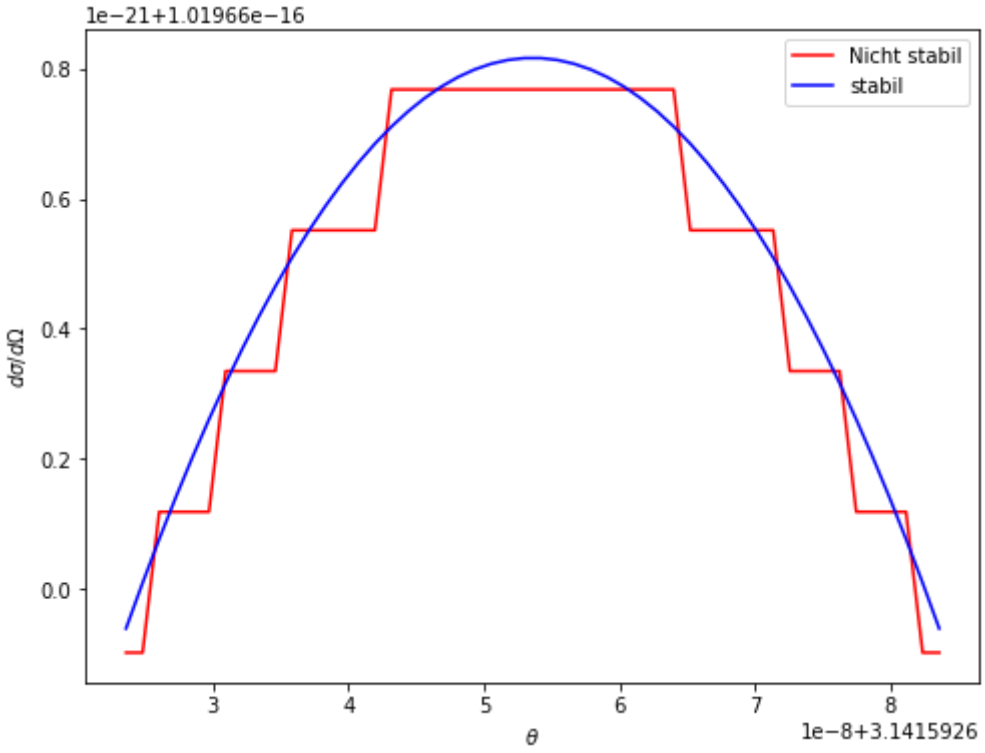
Es soll gezeigt werden, dass die Stabilitätsprobleme behoben wurden, indem die Gleichungen in den kritischen Intervallen graphisch dargestellt werden.

In [7]:

```
import scipy.constants as const
```

In [8]:

```
#Definition der Konstanten in der Funktion
Ee = 50*1e9
me = 511*1e3
s = (2*Ee)**2
gamma = Ee/me
beta = np.sqrt(1-gamma**(-2))
alpha= const.alpha

def wirkung(theta): #instabile Funktion
    return alpha**2 /s * ((2 + np.sin(theta)**2)/(1-beta**2 *np.cos(theta)**2))

def verbessert(theta): #umgeformte, stabile Funktion
    return alpha**2 /s * ((2 + np.sin(theta)**2)/(np.sin(theta)**2+1/(gamma**2)*
np.cos(theta)**2))

x=np.linspace(np.pi-3e-8, np.pi+3e-8) #Definitionsbereich in kleinem Bereich um
 pi gewählt

plt.figure(3, figsize=(8,6))
plt.plot(x, wirkung(x), 'r-', label="Nicht stabil")
plt.plot(x, verbessert(x), 'b-', label="stabil")
plt.xlabel(r'$\theta$')
plt.ylabel(r'$d\sigma/d\Omega$')
plt.legend()
None
```



d)

Die Konditionszahl soll berechnet werden und es soll erklärt werden, wie diese von θ abhängt.

Die Konditionszahl ergibt sich zu

$$K = \left| \frac{f'(\theta)}{f(\theta)} \right| \theta = \left| \frac{2\sin(\theta)\cos(\theta)\left(3m_e^2 - 2E_e\right)}{\left(\sin(\theta)^2 + 2\right)\left(m_e^2\cos(\theta)^2 + E_e^2\sin(\theta)^2\right)}\,\theta \right|.$$

Für Werte um $\theta = \frac{n}{2}\pi, \quad \forall n \in \mathbb{N}$ ist die Konditionierung gut, da dann entweder der $\cos$- oder der $\sin$-Teil null werden.

e)

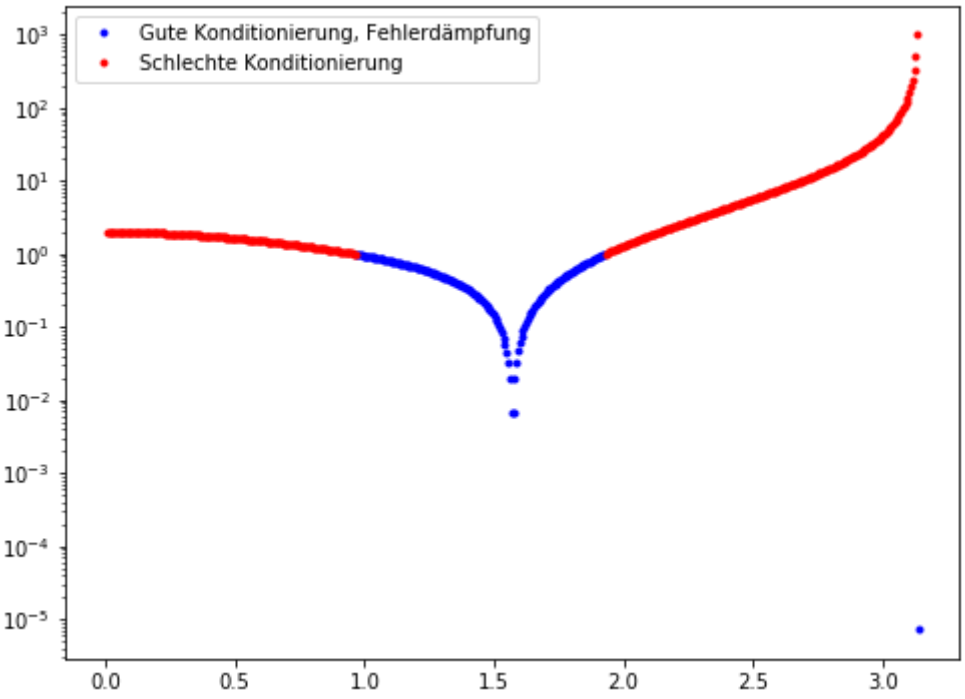
Der Verlauf der Konditionszahl soll als Funktion von θ im Intervall $(0 \leq \theta \leq \pi)$ graphisch dargestellt werden. Außerdem soll erklärt werden, in welchem Bereich das Problem gut und in welchem schlecht konditioniert ist.

In [9]:

```
def K(theta): #Konditionszahl als Funktion von theta
    return np.abs((2*np.sin(theta)* np.cos(theta)*(3*me**2 -2*Ee**2)) / ((np.sin
(theta)**2 +2)*(me**2 *np.cos(theta)**2+ Ee**2 * np.sin(theta)**2))*theta)
```

In [10]:

```
theta= np.linspace(-0, np.pi, 500) #Definitionsbereich recht klein gewählt, dami
t es nicht so viele Werte in der Ausgabe sind
y= K(theta)
plt.figure(1, figsize=(8,6))
plt.plot(theta[y<1], y[y<1], "b.", label="Gute Konditionierung, Fehlerdämpfung")
plt.plot(theta[y>1], y[y>1], "r.", label="Schlechte Konditionierung")
plt.legend(loc="best")
plt.yscale("log") #logarithmierte y-Skala zur besseren Darstellung der Werte
```



In [11]:

```
print("Gute Konditionierung für die Werte:", end="")# Ausgabe der gut konditioni
erten x-Werte
print(theta[y<1])
print("\n\n")
print("Schlechte Konditionierung für die Werte:", end="") #Ausgabe der schlecht
 konditionierten x-Werte
print(theta[y>1])
print("\n\n")
```

```
Gute Konditionierung für die Werte:[0.         0.96954964 0.97584541
0.98214119 0.98843697 0.99473274
 1.00102852 1.0073243  1.01362007 1.01991585 1.02621163 1.03250741
 1.03880318 1.04509896 1.05139474 1.05769051 1.06398629 1.07028207
 1.07657784 1.08287362 1.0891694  1.09546517 1.10176095 1.10805673
 1.1143525  1.12064828 1.12694406 1.13323983 1.13953561 1.14583139
 1.15212717 1.15842294 1.16471872 1.1710145  1.17731027 1.18360605
 1.18990183 1.1961976  1.20249338 1.20878916 1.21508493 1.22138071
 1.22767649 1.23397226 1.24026804 1.24656382 1.2528596  1.25915537
 1.26545115 1.27174693 1.2780427  1.28433848 1.29063426 1.29693003
 1.30322581 1.30952159 1.31581736 1.32211314 1.32840892 1.33470469
 1.34100047 1.34729625 1.35359203 1.3598878  1.36618358 1.37247936
 1.37877513 1.38507091 1.39136669 1.39766246 1.40395824 1.41025402
 1.41654979 1.42284557 1.42914135 1.43543712 1.4417329  1.44802868
 1.45432445 1.46062023 1.46691601 1.47321179 1.47950756 1.48580334
 1.49209912 1.49839489 1.50469067 1.51098645 1.51728222 1.523578
 1.52987378 1.53616955 1.54246533 1.54876111 1.55505688 1.56135266
 1.56764844 1.57394422 1.58023999 1.58653577 1.59283155 1.59912732
 1.6054231  1.61171888 1.61801465 1.62431043 1.63060621 1.63690198
 1.64319776 1.64949354 1.65578931 1.66208509 1.66838087 1.67467664
 1.68097242 1.6872682  1.69356398 1.69985975 1.70615553 1.71245131
 1.71874708 1.72504286 1.73133864 1.73763441 1.74393019 1.75022597
 1.75652174 1.76281752 1.7691133  1.77540907 1.78170485 1.78800063
 1.79429641 1.80059218 1.80688796 1.81318374 1.81947951 1.82577529
 1.83207107 1.83836684 1.84466262 1.8509584  1.85725417 1.86354995
 1.86984573 1.8761415  1.88243728 1.88873306 1.89502884 1.90132461
 1.90762039 1.91391617 1.92021194 1.92650772 3.14159265]



Schlechte Konditionierung für die Werte:[0.00629578 0.01259155 0.018
88733 0.02518311 0.03147888 0.03777466
 0.04407044 0.05036621 0.05666199 0.06295777 0.06925355 0.07554932
 0.0818451  0.08814088 0.09443665 0.10073243 0.10702821 0.11332398
 0.11961976 0.12591554 0.13221131 0.13850709 0.14480287 0.15109864
 0.15739442 0.1636902  0.16998598 0.17628175 0.18257753 0.18887331
 0.19516908 0.20146486 0.20776064 0.21405641 0.22035219 0.22664797
 0.23294374 0.23923952 0.2455353  0.25183107 0.25812685 0.26442263
 0.27071841 0.27701418 0.28330996 0.28960574 0.29590151 0.30219729
 0.30849307 0.31478884 0.32108462 0.3273804  0.33367617 0.33997195
 0.34626773 0.3525635  0.35885928 0.36515506 0.37145083 0.37774661
 0.38404239 0.39033817 0.39663394 0.40292972 0.4092255  0.41552127
 0.42181705 0.42811283 0.4344086  0.44070438 0.44700016 0.45329593
 0.45959171 0.46588749 0.47218326 0.47847904 0.48477482 0.4910706
 0.49736637 0.50366215 0.50995793 0.5162537  0.52254948 0.52884526
 0.53514103 0.54143681 0.54773259 0.55402836 0.56032414 0.56661992
 0.57291569 0.57921147 0.58550725 0.59180302 0.5980988  0.60439458
 0.61069036 0.61698613 0.62328191 0.62957769 0.63587346 0.64216924
 0.64846502 0.65476079 0.66105657 0.66735235 0.67364812 0.6799439
 0.68623968 0.69253545 0.69883123 0.70512701 0.71142279 0.71771856
 0.72401434 0.73031012 0.73660589 0.74290167 0.74919745 0.75549322
 0.761789   0.76808478 0.77438055 0.78067633 0.78697211 0.79326788
 0.79956366 0.80585944 0.81215522 0.81845099 0.82474677 0.83104255
 0.83733832 0.8436341  0.84992988 0.85622565 0.86252143 0.86881721
 0.87511298 0.88140876 0.88770454 0.89400031 0.90029609 0.90659187
 0.91288764 0.91918342 0.9254792  0.93177498 0.93807075 0.94436653
 0.95066231 0.95695808 0.96325386 1.9328035  1.93909927 1.94539505
 1.95169083 1.9579866  1.96428238 1.97057816 1.97687393 1.98316971
 1.98946549 1.99576126 2.00205704 2.00835282 2.0146486  2.02094437
 2.02724015 2.03353593 2.0398317  2.04612748 2.05242326 2.05871903
 2.06501481 2.07131059 2.07760636 2.08390214 2.09019792 2.09649369
```

```
2.10278947 2.10908525 2.11538103 2.1216768  2.12797258 2.13426836
2.14056413 2.14685991 2.15315569 2.15945146 2.16574724 2.17204302
2.17833879 2.18463457 2.19093035 2.19722612 2.2035219  2.20981768
2.21611346 2.22240923 2.22870501 2.23500079 2.24129656 2.24759234
2.25388812 2.26018389 2.26647967 2.27277545 2.27907122 2.285367
2.29166278 2.29795855 2.30425433 2.31055011 2.31684588 2.32314166
2.32943744 2.33573322 2.34202899 2.34832477 2.35462055 2.36091632
2.3672121  2.37350788 2.37980365 2.38609943 2.39239521 2.39869098
2.40498676 2.41128254 2.41757831 2.42387409 2.43016987 2.43646565
2.44276142 2.4490572  2.45535298 2.46164875 2.46794453 2.47424031
2.48053608 2.48683186 2.49312764 2.49942341 2.50571919 2.51201497
2.51831074 2.52460652 2.5309023  2.53719807 2.54349385 2.54978963
2.55608541 2.56238118 2.56867696 2.57497274 2.58126851 2.58756429
2.59386007 2.60015584 2.60645162 2.6127474  2.61904317 2.62533895
2.63163473 2.6379305  2.64422628 2.65052206 2.65681784 2.66311361
2.66940939 2.67570517 2.68200094 2.68829672 2.6945925  2.70088827
2.70718405 2.71347983 2.7197756  2.72607138 2.73236716 2.73866293
2.74495871 2.75125449 2.75755027 2.76384604 2.77014182 2.7764376
2.78273337 2.78902915 2.79532493 2.8016207  2.80791648 2.81421226
2.82050803 2.82680381 2.83309959 2.83939536 2.84569114 2.85198692
2.85828269 2.86457847 2.87087425 2.87717003 2.8834658  2.88976158
2.89605736 2.90235313 2.90864891 2.91494469 2.92124046 2.92753624
2.93383202 2.94012779 2.94642357 2.95271935 2.95901512 2.9653109
2.97160668 2.97790246 2.98419823 2.99049401 2.99678979 3.00308556
3.00938134 3.01567712 3.02197289 3.02826867 3.03456445 3.04086022
3.047156   3.05345178 3.05974755 3.06604333 3.07233911 3.07863488
3.08493066 3.09122644 3.09752222 3.10381799 3.11011377 3.11640955
3.12270532 3.1290011  3.13529688]
```

Die Werte 0 und π sowie das Intervall zwischen ca. 0.97 und ca. 1.93 sind gut konditioniert, da K dort unter dem Wert 1 liegt. In den Intervallen dazwischen liegen die Werte teilweise sogar sehr weit über 1 und sind somit schlecht konditioniert.