Vakuum

May 27, 2024

1 Fakultät für Physik

1.1 Physikalisches Praktikum P2 für Studierende der Physik

Versuch P2-41, 42, 22 (Stand: April 2024)

Raum F1-19

2 Vakuum

Tin Vrkic E-Mail: uyvpq@student.kit.edu

Mika Nock E-Mail: uttzi@student.kit.edu

Gruppennummer: Mo32

Betreuer: Marcel Gaisdörfer

Versuch durchgeführt am: 13.05.2024

Beanstandungen zu Protokoll Version _____:

Testiert am: _____ Testat: _____

[1]:
```
# importieren aller nötiger Module
from uncertainties import ufloat
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import kafe2
import pathlib
```

[2]:
```
# erstellen einer Funktion für kafe2 Fits
def fit_funktion(xy_data, model_function, xy_error, xy_label, title,␣
 ↪constraint=[]):
    xy_data = kafe2.XYContainer(xy_data[0], xy_data[1])
    xy_data.label = title
    fit = kafe2.XYFit(xy_data = xy_data, model_function = model_function)
    fit.add_error(axis = 'x', err_val = xy_error[0])
    fit.add_error(axis = 'y', err_val = xy_error[1])
    for i in range(len(constraint)):
        fit.add_parameter_constraint(name = constraint[i][0], value =␣
 ↪constraint[i][1], uncertainty = constraint[i][2])
    fit.do_fit()
    plot = kafe2.Plot(fit)
    plot.x_label, plot.y_label = xy_label[0], xy_label[1]

    return fit.parameter_values, fit.parameter_errors, plot
```

3 Durchführung

Die Anleitung zu diesem Versuch finden Sie hier.

3.1 Aufgabe 1: Versuchsaufbau

Hinweise zu Aufgabe 1 finden in der Datei Hinweise-Versuchsdurchfuehrung.md.

- Machen Sie sich mit dem Versuchsaufbau vertraut.
- Bearbeiten Sie hierzu die folgenden Aufgaben.

3.1.1 Aufgabe 1.1: Orientierung und Beschreibung des Versuchsaufbaus

- Verschaffen Sie sich einen Überblick über die verwendeten Apparaturen und beschreiben Sie sie in eigenen Worten.

- Verfolgen Sie hierzu die Leitungen und identifizieren Sie die verwendeten Elemente in der zugehörigen Skizze.

Der vorgefundene Aufbau besteht aus einer Glaskuppel, zwei Pumpen, mehreren Ventilen und verschiedenen Manometern.

In der Glaskuppel (RZ) ist das Verdampferschiffchen zu finden, wobei die Kugelelektroden allerdings fehlen. Von dieser Glaskuppel geht ein Rohr ab, welches einersites das Ionisationsvakuumeter (IM) beinhaltet und durch zwei Ventile von weiteren Teilen abgetrennt ist. V3 führt hierbei zu einem Wärmeleitvakuumeter (T3), zu einem Referenzvolumen (RV) und einem Auslassventil (B2). Hinter V2 befinden sich die Turbomolekularpumpe (TMP), ein Wärmeleitvakuumeter (T2), ein austauschbares Leitungselement (L), ein Ventil (V1), noch ein Wärmeleitvakuumeter (V2), ein weiteres Auslassventil und die Drehschieberpumpe (DSP).

Alle Geräte waren mit Atmosphärendruck belüftet und der Verdampfer war mit Indium ausgestatten.

3.1.2 Aufgabe 1.2: Gasentladung (Demonstrationsversuch)

- Schalten Sie das Hochspannungsgerät zur Erzeugung der Gasentladungen ein.
- Evakuieren Sie RZ und die Gasentladungsröhre gemeinsam mit Hilfe der DSP.
- Senken Sie den Druck kontinuierlich, bis die Gasentladung erlischt.
- Skizzieren und beschreiben Sie die Gasentladung in Abhängigkeit vom Gasdruck.

In diesem Versuch treten an einer Anode, die sich in einer Glasröhre befindet, Elektronen aus. Diese werden in Richtung der Kathode beschleunigt und können je nach Druck in der Röhre verschiedene Erscheinungen verursachen.

Zu Beginn befindet sich soviel Luft in der Röhre, dass die Elektronen mit diesen Stoßen und die Energie, die sie durch die Beschleunigung erhalten wieder abgeben. Wird nun die Röhre evakuiert können die Elektronen immer mehr Energie aufbauen, bevor sie wieder mit einem Atom stoßen. Ab einem bestimmten Druck können sie schnell genug werden um die Luftmoleküle zu ionisieren, wobei beim rekombinieren dieser Photonen frei werden. Es ergibt sich somit ein Leuchten, das sich mit sinkendem Druck von der Kathode in Richtung Anode ausbreitet. Bei einer weiteren Entleerung des Rohres können die Elektronen nun auf sehr kurzer Strecke die nötige Energie aufbauen um

Photonen freizusetzen. Sie werden somit aus der Anode gelöst, werden stark beschleunigt, treffen auf ein Atom, welches sie zum Leuchten anregen und werden so wieder abgebremst. Dieser Zustand ist in der unteren Grafik zu sehen. Hierbei sind die dunklen Zonen die Strecken, in denen die Elektronen beschleunigt werden und die hellen Zonen die, in denen sie mit Atomen stoßen und so Licht freisetzen.

In der Theorie sollte außerdem das Leuchten bei sehr niedrigen Drücken erlischen, da dann nicht-mehr genug Atome vorhanden sind, um eine wahrnehmbare Menge an Photonen zu erzeugen. Dieser Punkt konnte jedoch im vorliegenden Aufbau nicht erreicht werden.

3.2 Aufgabe 2: Saugvermögen und Leitwert

Hinweise zu Aufgabe 1 finden in der Datei Hinweise-Versuchsdurchfuehrung.md.

- Untersuchen Sie das effektive Saugvermögen von DSP und TMP, sowie den Strömungsleitwert eines dünnen Rohrs.
- Bearbeiten Sie hierzu die folgenden Aufgaben.

3.2.1 Aufgabe 2.1: Saugvermögen der DSP

Evakuieren Sie die Apparatur mit Hilfe der DSP und stellen Sie die folgenden funktionalen Zusammenhänge des Druck jeweils **bei T1** geeignet graphisch dar: - Den Druck als Funktion der Zeit $p(t)$. - Das Saugvermögen als Funktion des Drucks $S(p)$.

---

[3]:
```
t_5min  = np.linspace(0, 60, 61) * 5 # seconds
t_5min_std = .1 # seconds
druck_std = .1 # 10%

vol_schlauch = 530 * np.pi * (23 * 10**(-3) / 2)**2 * 10**(-3) # m^3
vol_rohr = 530 * np.pi * (2 * 10**(-3) / 2)**2 * 10**(-3) # m^3
volumen_rz_44 = 9.2 * 10**(-3) # m^3
ges_vol_schlauch = vol_schlauch + volumen_rz_44 + .5 * 10**(-3) # m^3
ges_vol_rohr = vol_rohr + volumen_rz_44 + .5 * 10**(-3) # m^3
```

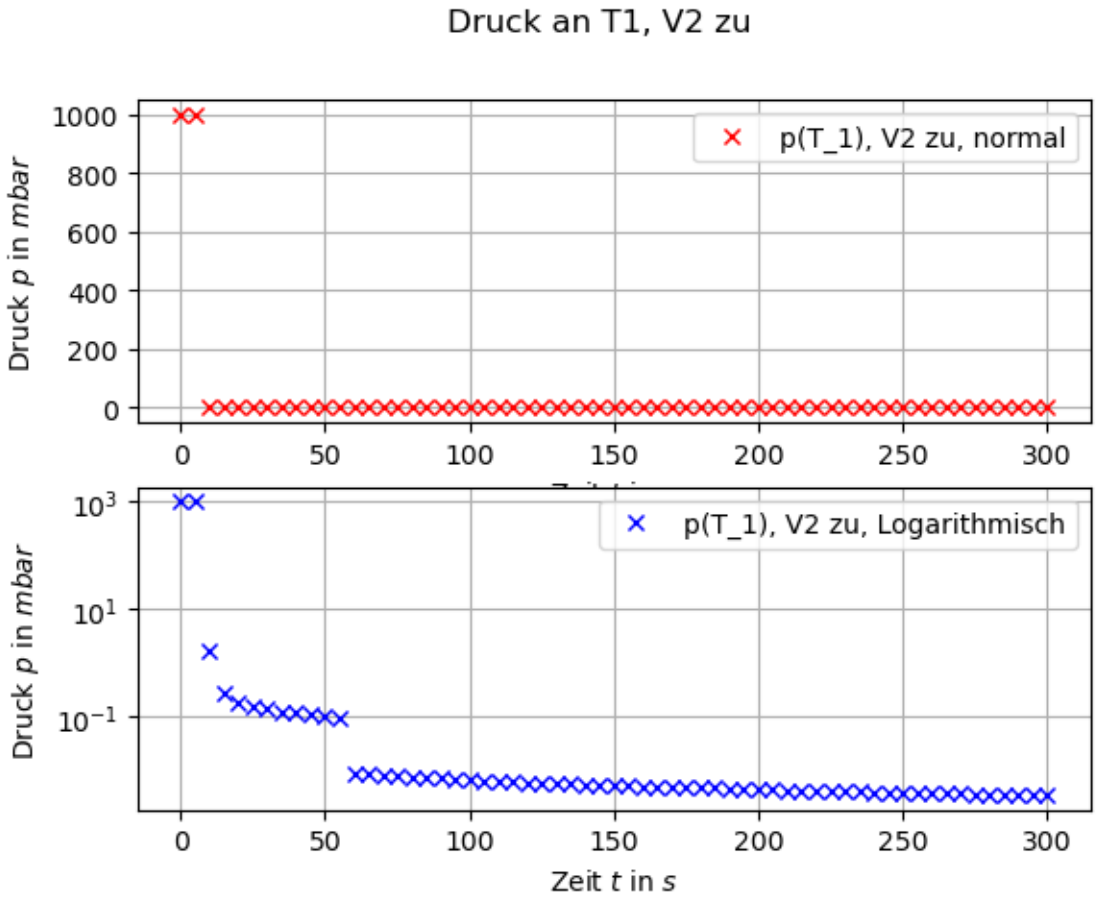
Bei dieser ersten Messungen wurde zunächst ausversehen das Ventil V2 geschlossen gehalten. Zum Vergleich, ob es einen nennenswerten Unterschied macht, wurde die Messreihe zusätzlich zur richtigen trotzdem durchgeführt.

Die Unsicherheit auf die Zeitablesung wird auf etwa 0.1 s geschätzt, die auf die Druckablesung beträgt für das Messgerät 10 %.

[4]:
```
druck_2_1_zu = np.array([1000, 1000, 1.71, 0.266, 0.189, 0.157, 0.139, 0.126, 0.
↪117, 0.108, 0.101, 0.0951,
                         0.00902, 0.00863, 0.00828, 0.00796, 0.00770, 0.00744,␣
↪0.00716, 0.00692, 0.00669, 0.00646, 0.00629, 0.00614,
                         0.00597, 0.00584, 0.00571, 0.00558, 0.00546, 0.00538,␣
↪0.00526, 0.00516, 0.00511, 0.00502, 0.00495, 0.00488,
                         0.00482, 0.00473, 0.00465, 0.00457, 0.00450, 0.00445,␣
↪0.00434, 0.00428, 0.00422, 0.00416, 0.00410, 0.00404,
                         0.00398, 0.00395, 0.00391, 0.00387, 0.00382, 0.00376,␣
↪0.00373, 0.00369, 0.00367, 0.00362, 0.00357, 0.00355,
                         0.00352])
druck_2_1_auf = np.array([1000, 1000, 1000, 1000, 1000, 1000, 460, 193, 113, 69.
↪9, 45.8, 31.3,
                          22.5, 15.9, 11.4, 8.33, 5.98, 4.44, 3.31, 2.48, 1.89,␣
↪1.48, 1.18, 0.934,
                          0.763, 0.624, 0.526, 0.450, 0.385, 0.338, 0.301, 0.
↪269, 0.242, 0.222, 0.207, 0.194,
```

```
                        0.181, 0.171, 0.162, 0.154, 0.148, 0.142, 0.137, 0.
↪133, 0.129, 0.126, 0.123, 0.121,
                        0.118, 0.116, 0.113, 0.111, 0.109, 0.107, 0.106, 0.
↪104, 0.103, 0.101, 0.100, 0.00991,
                        0.00974])
```

[5]:
```
fig0, ax0 = plt.subplots(2)
ax0[0].plot(t_5min, druck_2_1_zu, "rx", label="p(T_1), V2 zu, normal")
ax0[0].legend(), ax0[0].grid()
ax0[1].plot(t_5min, druck_2_1_zu, "bx", label="p(T_1), V2 zu, Logarithmisch")
ax0[1].set_yscale("log")
ax0[1].legend(), ax0[1].grid()
fig0.suptitle("Druck an T1, V2 zu")

for ax in ax0:
    ax.set(xlabel="Zeit $t$ in $s$", ylabel="Druck $p$ in $mbar$")
```

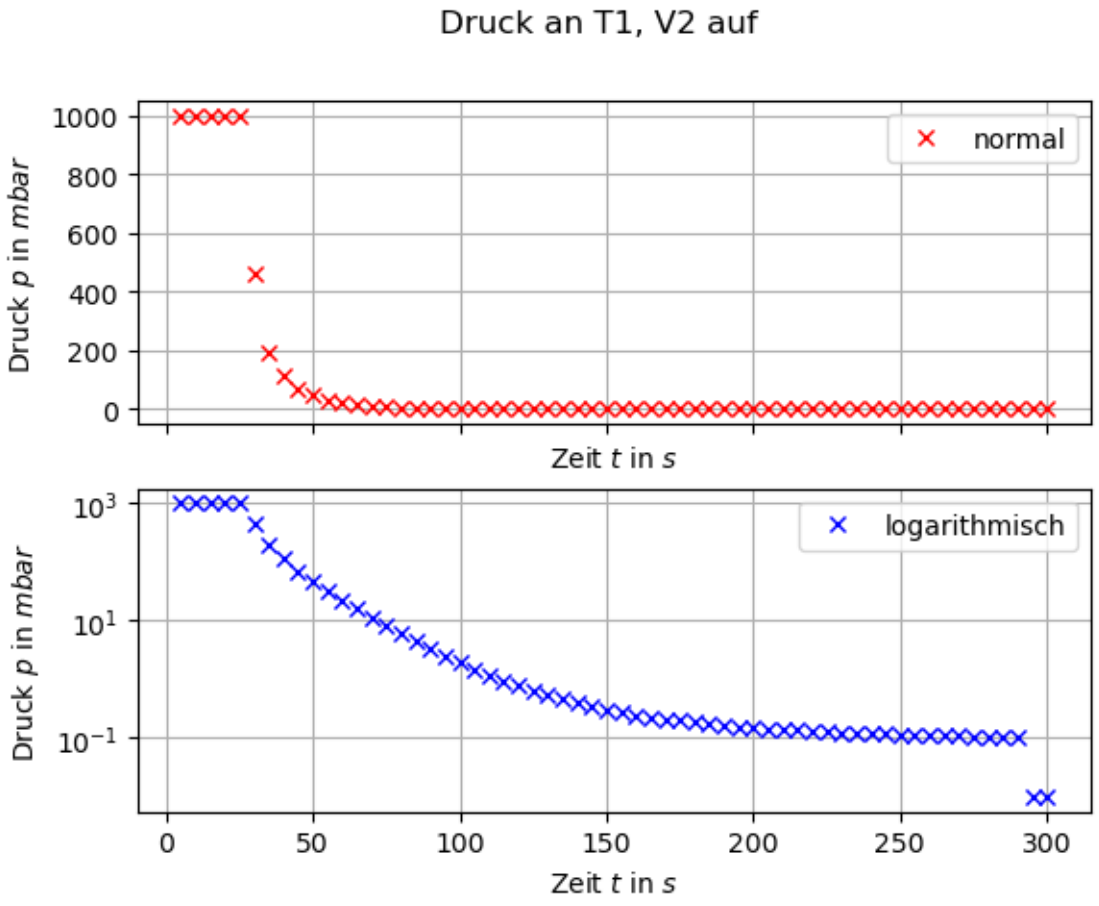


In der logarithmischen Darstellung sieht man gut, welche komischen Dinge die Werte machen, deshalb beschränken wir uns im folgenden, wie von der Aufgabe gefordert, auf die Messreihe mit

offenem Ventil V2.

Der erste Wert bei $t = 0\,s$ wird jedoch weggelassen, da in der Formel für das Saugvermögen durch den Zeitpunkt t_0 geteilt wird. Um die Division durch 0 zu verhindern, werden dann alle Werte ab einschließlich Sekunde 5 genutzt. Der Vergleichbarkeit wegen wird für folgenden Plot der Wert bei $0\,s$ ebenfalls weggelassen.

[6]:
```
fig, ax = plt.subplots(2, sharex=True)
ax[0].plot(t_5min[1:], druck_2_1_auf[1:], "rx", label="normal")
ax[0].legend(), ax[0].grid()

ax[1].plot(t_5min[1:], druck_2_1_auf[1:], "bx", label="logarithmisch")
ax[1].set_yscale("log")
ax[1].legend(), ax[1].grid()

fig.suptitle("Druck an T1, V2 auf")

for ax in ax:
    ax.set(xlabel="Zeit $t$ in $s$", ylabel="Druck $p$ in $mbar$")
```



In der logarithmischen Darstellung sollte der mittlere Teil der Kurve eigentlich einen einigermaßen geraden Verlauf haben. Warum das so ist, ist schwer zu sagen, auf jeden Fall liegt es an den Messwerten, die nicht ganz so ausgefallen sind wie erwartet.

Um nun das Saugvermögen der DSP zu bestimmen, gehen wir von der, in der Vorbereitung hergeleiteten, Formel aus:

$$ln(\frac{p}{p_0}) = -\frac{S}{V}(t - t_0)$$

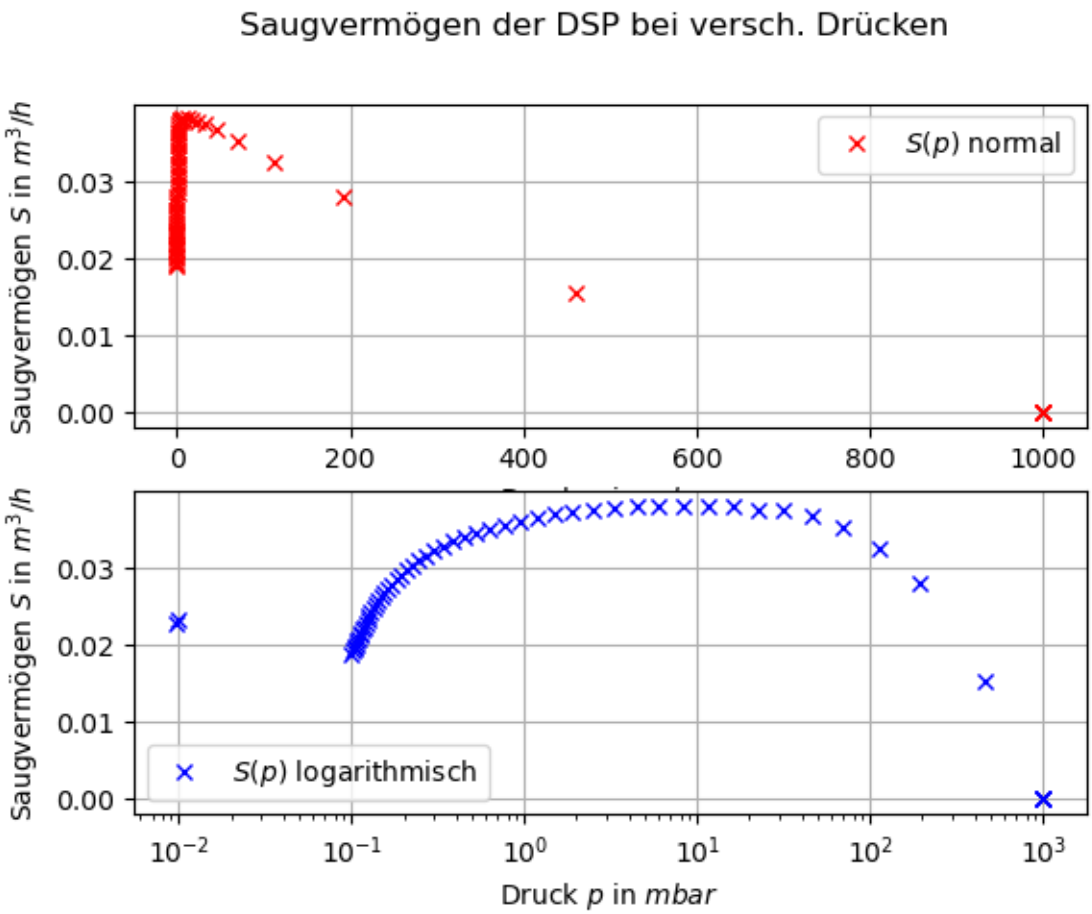
$$\Leftrightarrow S(p) = -\frac{V}{(t - t_0)} ln(\frac{p}{p_0})$$

mit dem Anfangsdruck $p_0 = 1000\,mbar$ und dem Volumen des Rezipienten $V_{RZ} = 9.2\,l = 9.2\,dm^3$.

Das Saugvermögen bei einem Druck von $10\,mbar$ beträgt $S(10\,mbar) = 0.038\,\frac{m^3}{h}$

[7]:
```
saugvermögen = - ( ges_vol_schlauch / t_5min[1:] ) * np.log(druck_2_1_auf[1:] /␣
 ↪druck_2_1_auf[1]) * 60

fig1, ax1 = plt.subplots(2, sharey=True)
ax1[0].plot(druck_2_1_auf[1:], saugvermögen, "rx", label="$S(p)$ normal")
ax1[0].legend(), ax1[0].grid()

ax1[1].plot(druck_2_1_auf[1:], saugvermögen, "bx", label="$S(p)$ logarithmisch")
ax1[1].set_xscale("log")
ax1[1].legend(), ax1[1].grid()

fig1.suptitle("Saugvermögen der DSP bei versch. Drücken")

for ax in ax1:
    ax.set(xlabel="Druck $p$ in $mbar$", ylabel="Saugvermögen $S$ in $m^3/h$")

saugvermögen_10mbar = -60*(ges_vol_schlauch/t_5min[14])*np.
 ↪log(druck_2_1_auf[14]/druck_2_1_auf[1])
print(f"Saugvermögen bei 10mbar: S(10mbar)={saugvermögen_10mbar} m³/h")
```

```
Saugvermögen bei 10mbar: S(10mbar)=0.038043764597377426 m³/h
```



In der normalen Darstellung erkennt man wenig von den drei erwarteten Bereichen. Schaut man sich den Graphen jedoch logarithmisch in p an, kann man grob die drei Bereiche identifizieren:

- Bereich I: Saugvermögen nimmt von $10^3\,mbar$ bis etwa $5 \cdot 10^1\,mbar$ zu, bis zu einem Saugvermögen von knapp unter $0.04\,\frac{m^3}{h}$
- Bereich II: Saugvermögen bleibt zwischen $5 \cdot 10^1\,mbar$ bis etwa $10^0\,mbar$ relativ konstant
- Bereich III: Saugvermögen sinkt wieder, da das Grob- in ein Feinvakuum übergeht, wobei der Graph bei einem Druck von $10^{-1}\,mbar$ einen Knick hat. Das liegt daran, dass hier die Messwerte selbst einen größeren Sprung aufweisen, den man im zweiten Plot oben gegen Ende der Messreihe in der logarithmischen Darstellung sieht.

Laut Datenblatt wird für die DSP ein Saugvermögen von etwa $2.5\,\frac{m^3}{h}$ erwartet. Warum unsere Messung einen Wert liefert, der 2 Größenordnungen unter dieser Erwartung liegt, ist nach einigen Stunden Brüten immer noch unklar. Die Messung scheint nicht so extrem schlecht ausgefallen zu sein, um dieses Ergebnis zu rechtfertigen, wir sehen sonst aber keinen anderen Faktor, der das Ergebnis so beeinflussen könnte.

Schließlich noch der exponentielle Fit an $p(t)$:

In der Vorbereitung wird für die Modellfunktion auf Gleichung (5) in der Datei Hinweise-Vakuum.md verwiesen. In dieser Datei existiert aber keine Gleichung (5), die letzte Gleichung (Gleichung (4)) wurde schon für das Saugvermögen oben genutzt. Für den Fit macht Gleichung (1) aus der Datei Hinweise-Vakuum-a.md mehr Sinn, wobei diese nur eine andere Form der selben Gleichung ist.

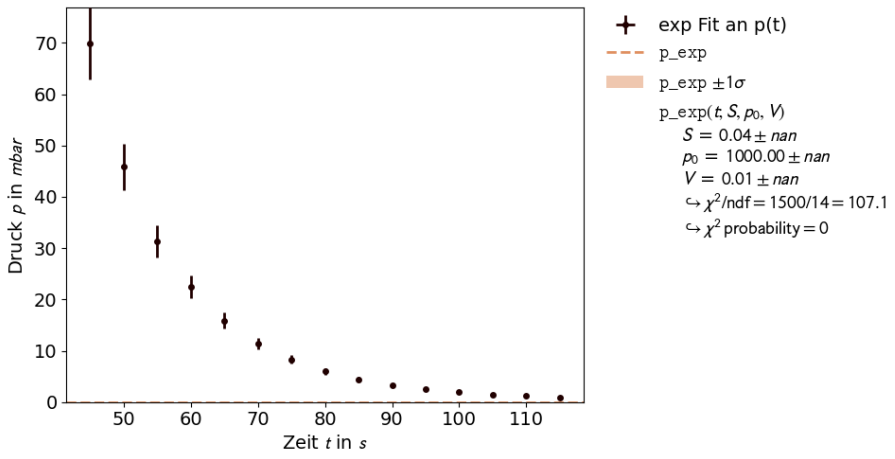
$$p(t) = p_0 \cdot exp(-\frac{S}{V}(t - t_0))$$

Die Werte vom Anfangsdruck p_0 und dem Gesamtvolumen der Apparatur V werden constraint mit den bekannten, festen Werten, das Saugvermögen S wird mit dem Anfangswert $S \approx 0.038\ \frac{m^3}{h}$ übergeben, wird aber für die Anpassung flexibel gelassen.

[8]:
```
def p_exp(t, S=saugvermögen_10mbar, p_0=1000, V=ges_vol_schlauch):
    return p_0 * np.exp(- S * t / V)
```

[9]:
```
fit_2_1_data = np.array([ t_5min[9:24] , druck_2_1_auf[9:24] ])
# .1s Unsicherheit auf Zeitablesung
# 10% Unsicherheit auf Druckmessung
fit_2_1_error = np.array([ np.ones(15)*t_5min_std , druck_2_1_auf[9:24] *␣
 ↪druck_std ])
fit_2_1_label = [ "Zeit $t$ in $s$" , "Druck $p$ in $mbar$" ]
fit_2_1_title = "exp Fit an p(t)"
fit_2_1_const = [ ["p_0", 1000, 100] , ["V", ges_vol_schlauch, .1] ]

fit_res_2_1 = fit_funktion(fit_2_1_data, p_exp, fit_2_1_error, fit_2_1_label,␣
 ↪fit_2_1_title, fit_2_1_const)

fit_res_2_1[2].plot()
fit_res_2_1[2].show()
```

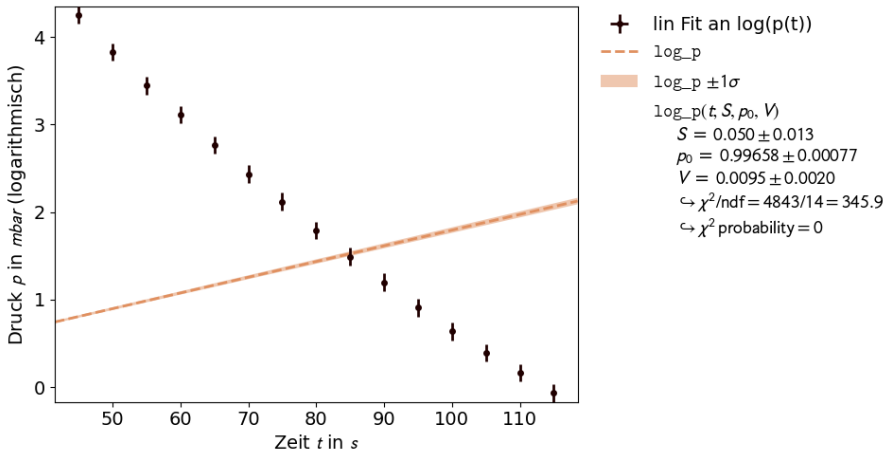
```
Warning: the cost function has been evaluated as infinite. The fit might not
converge correctly.
```



Auch hier verstehe ich nicht, warum der Fit kein Stück funktioniert. Im folgenden wird der selbe Fit nochmal gemacht, nur dass $p(t)$ zu $log(p)$ wird, sodass eine Lineare Modellfunktion angesetzt werden kann. Dadurch wird die Formel zu $\ln(p) = -\ln(p_0)\frac{S}{V}(t - t_0)$.
Trotz der dadurch vereinfachten Modellfunktion fällt der Fit ganz komisch aus. Es wird sicherlich wieder an den Messwerten liegen, aber warum genau diese so schlecht ausgefallen sind, kann ich nicht verstehen. Aus diesem Grund macht es auch wenig Sinn, auf den χ^2-Wert einzugehen, da dieser bei 0 liegt.
Zusätzlich kann man noch erwähnen, dass beim zweiten Fit die Fit Constraints komplett ignoriert werden.

[10]:
```
def log_p(t, S=saugvermögen_10mbar, p_0=1000, V=ges_vol_schlauch):
    return (- np.log(p_0) * (S/V) * t)

fit_2_1_data_log = np.array([ t_5min[9:24] , np.log(druck_2_1_auf[9:24]) ])
fit_2_1_error_log = np.array([ np.ones(15)*t_5min_std , np.absolute(␣
 ↪druck_2_1_auf[9:24] * druck_std / druck_2_1_auf[9:24] ) ] )
fit_2_1_label_log = [ "Zeit $t$ in $s$" , "Druck $p$ in $mbar$ (logarithmisch)"␣
 ↪]
fit_2_1_title_log = "lin Fit an log(p(t))"

fit_res_2_1_log = fit_funktion(fit_2_1_data_log, log_p, fit_2_1_error_log,␣
 ↪fit_2_1_label_log, fit_2_1_title_log, fit_2_1_const)
fit_res_2_1_log[2].plot()
fit_res_2_1_log[2].show()
```

```
C:\Users\probz\AppData\Local\Temp\ipykernel_16584\2323453377.py:2:
RuntimeWarning: invalid value encountered in log
  return (- np.log(p_0) * (S/V) * t)

Warning: the cost function has been evaluated as infinite. The fit might not
converge correctly.
```



Schließlich noch ein Wort zu der Warnung beim zweiten Fit: die einzigen Variablen, die eine solche Warnung auslösen könnten, sind das Volumen, das im Nenner steht, oder der Anfangsdruck, der im Logarithmus steht. Beide sollten durch die Constraints feste Werte haben, wodurch der Nenner nie null werden sollte und das Argument des Logarithmus auch nicht, aber da kafe2 die Werte selbstständig variiert (p_0 sollte z.B. fest bei 1000 $mbar$ bleiben, der zweite Fit liefert aber etwa 1 $mbar$), können hier natürlich verbotene Werte auftauchen.

3.2.2 Aufgabe 2.2: Leitwert eines dünnen Rohrs

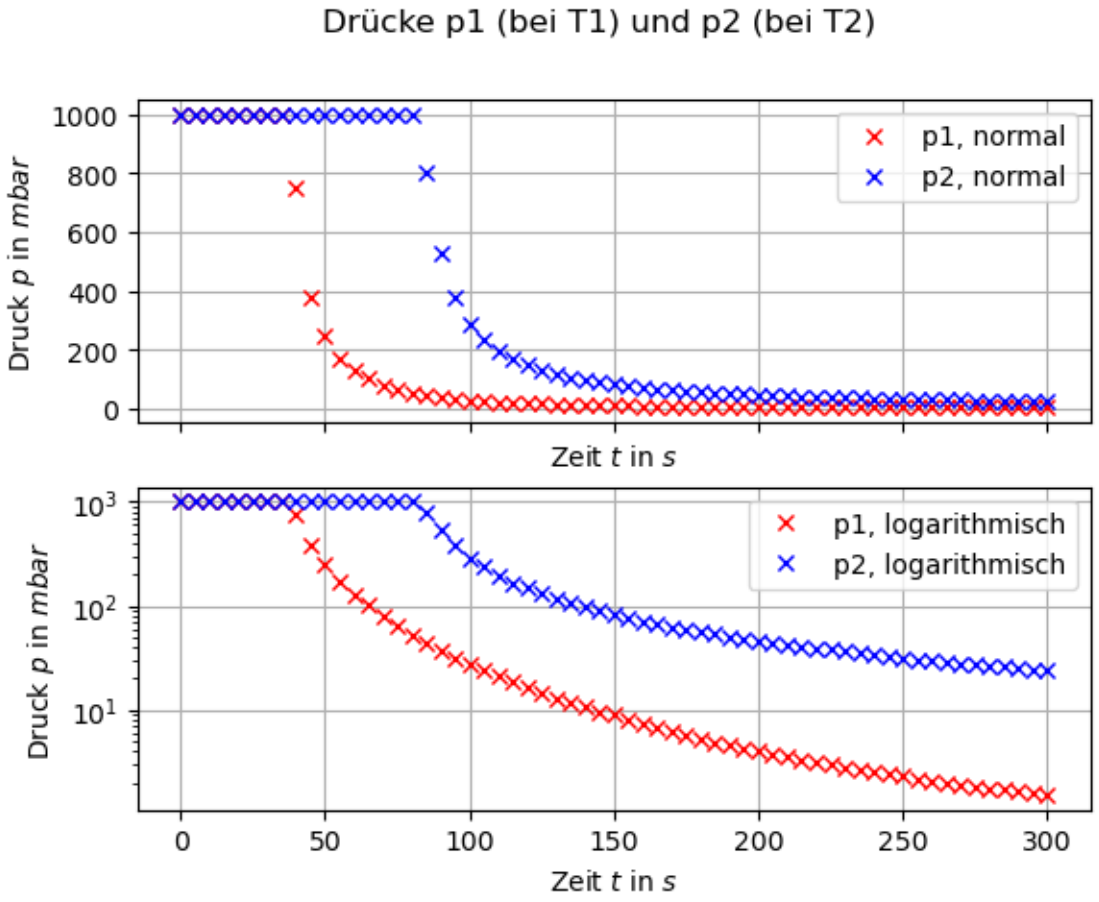
- Bei der Verbindungsleitung L hier handelt es sich im Originalaufbau um einen Metallwellschlauch. Tauschen Sie diesen gegen das bereitliegende etwa gleichlange dünne Metallrohr aus.
- Bestimmen Sie die Drucke p_1 (bei T1) vor und p_2 (bei T2) hinter dem Rohr als Funktion der Zeit.
- Bestimmen Sie aus den gewonnenen Daten den Leitwert des Rohrs als Funktion des Drucks.

---

[11]:
```
druck_p1_2_2 = np.array([1000, 1000, 1000, 1000, 1000, 1000, 1000, 1000, 750,␣
↪380, 245, 171,
                         128, 102, 78.0, 64.4, 52.3, 43.9, 37.4, 31.4, 27.5, 24.
↪1, 20.9, 18.4,
                         16.3, 14.6, 12.9, 11.7, 10.6, 9.61, 8.91, 8.00, 7.25,␣
↪6.71, 6.26, 5.74,
                         5.29, 4.90, 4.61, 4.29, 4.01, 3.76, 3.54, 3.33, 3.14,␣
↪2.97, 2.80, 2.67,
                         2.53, 2.41, 2.30, 2.18, 2.09, 2.00, 1.91, 1.84, 1.76,␣
↪1.70, 1.64, 1.57,
                         1.52])
druck_p2_2_2 = np.array([1000, 1000, 1000, 1000, 1000, 1000, 1000, 1000, 1000,␣
↪1000, 1000, 1000,
                         1000, 1000, 1000, 1000, 1000, 800, 530, 375, 288, 236,␣
↪196, 166,
                         148, 131, 115, 106, 97.1, 89.3, 81.8, 75.4, 70.2, 66.
↪4, 62.6, 59.3,
                         56.1, 53.0, 50.1, 47.3, 45.3, 43.6, 42.1, 40.5, 39.2,␣
↪37.9, 36.4, 35.1,
                         33.9, 32.8, 31.6, 30.5, 29.3, 28.5, 27.8, 27.1, 26.5,␣
↪26.0, 25.3, 24.6,
                         24.1])
```

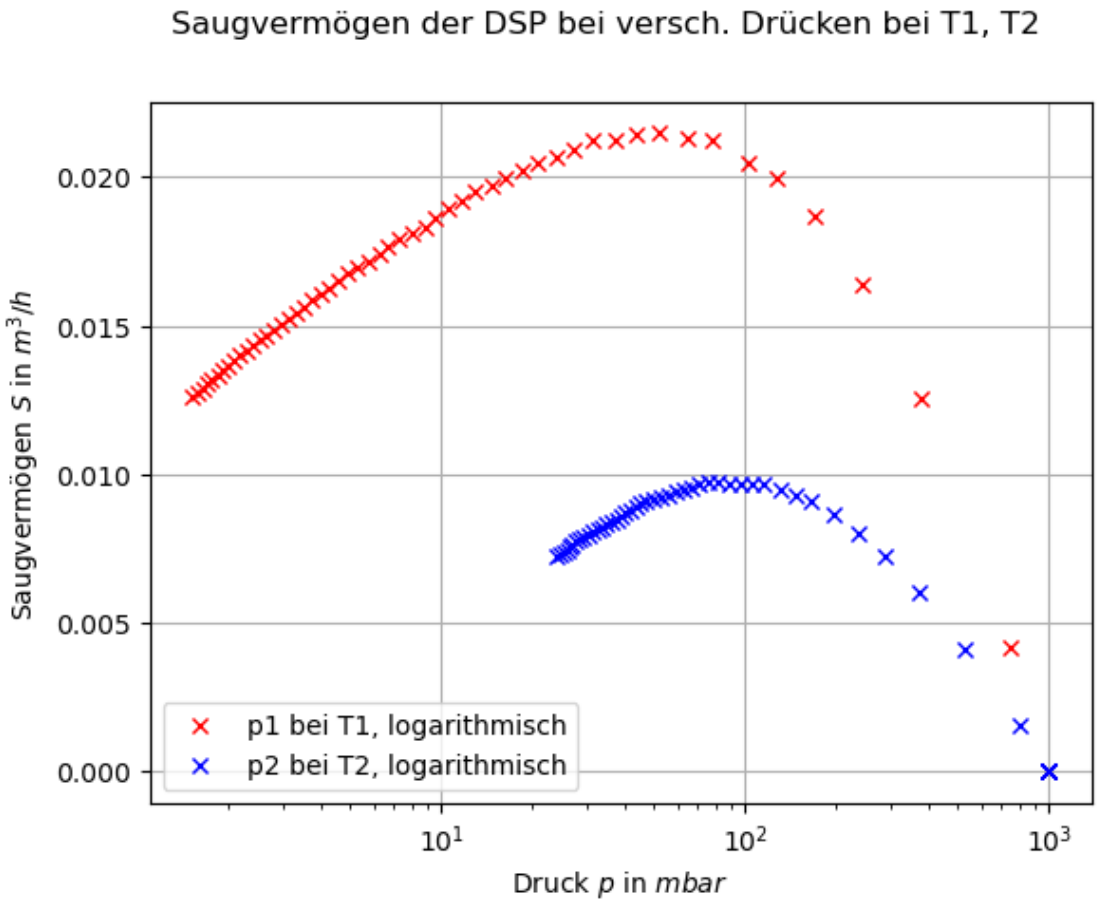
[12]:
```
fig2, ax2 = plt.subplots(2, sharex=True)
ax2[0].plot(t_5min, druck_p1_2_2, "rx", label="p1, normal")
ax2[0].plot(t_5min, druck_p2_2_2, "bx", label="p2, normal")
ax2[0].legend(), ax2[0].grid()
```

```
ax2[1].plot(t_5min, druck_p1_2_2, "rx", label="p1, logarithmisch")
ax2[1].plot(t_5min, druck_p2_2_2, "bx", label="p2, logarithmisch")
ax2[1].set_yscale("log")
ax2[1].legend(), ax2[1].grid()

fig2.suptitle("Drücke p1 (bei T1) und p2 (bei T2)")

for ax in ax2:
    ax.set(xlabel="Zeit $t$ in $s$", ylabel="Druck $p$ in $mbar$")
```



[13]:
```
saugvermögen2_p1 = - ( ges_vol_rohr / t_5min[1:] ) * np.log(druck_p1_2_2[1:] /␣
 ↪druck_p1_2_2[1]) * 60
saugvermögen2_p2 = - ( ges_vol_rohr / t_5min[1:] ) * np.log(druck_p2_2_2[1:] /␣
 ↪druck_p2_2_2[1]) * 60

fig3, ax3 = plt.subplots()
ax3.plot(druck_p1_2_2[1:], saugvermögen2_p1, "rx", label="p1 bei T1,␣
 ↪logarithmisch")
```

```
ax3.plot(druck_p2_2_2[1:], saugvermögen2_p2, "bx", label="p2 bei T2,␣
 ↪logarithmisch")
ax3.set_xscale("log")
ax3.legend(), ax3.grid()
fig3.suptitle("Saugvermögen der DSP bei versch. Drücken bei T1, T2")
ax3.set(xlabel="Druck $p$ in $mbar$", ylabel="Saugvermögen $S$ in $m^3/h$")
plt.show()


saugvermögen2_p1_50mbar = -60*(ges_vol_rohr/t_5min[16])*np.log(druck_p1_2_2[16]/
 ↪druck_p1_2_2[1])
print(f"Saugvermögen bei T1 bei etwa 52.3mbar: S(52.
 ↪3mbar)={saugvermögen2_p1_50mbar} m³/h")

saugvermögen2_p2_100mbar = -60*(ges_vol_rohr/t_5min[28])*np.
 ↪log(druck_p2_2_2[28]/druck_p2_2_2[1])
print(f"Saugvermögen bei T2 bei etwa 97.1mbar: S(97.
 ↪1mbar)={saugvermögen2_p2_100mbar} m³/h")
```



```
Saugvermögen bei T1 bei etwa 52.3mbar: S(52.3mbar)=0.02147045591285119 m³/h
Saugvermögen bei T2 bei etwa 97.1mbar: S(97.1mbar)=0.0096961790450355 m³/h
```

Obiger Plot zeigt wieder den Druck logarithmisch gegen das Saugvermögen aufgetragen an, wie in 2.1, nur diesmal mit den beiden Drücken p_1 und p_2 bei T_1 und T_2. Es wird wieder der konstante zweite Bereich gesucht, der für
- p_1 zwischen 20.9 $mbar$ und 78 $mbar$
- p_2 zwischen 70.2 $mbar$ und 115 $mbar$

liegt (die Werte sind schon aus dem Array der Messung oben identifiziert worden).

Als Anfangswert für das Saugvermögen für den Fit werden die Werte
- $S_1 \approx 0.021\,\frac{m^3}{h}$ bei $p_1 = 52.3\,mbar$
- $S_2 \approx 0.0097\,\frac{m^3}{h}$ bei $p_2 = 97.1\,mbar$

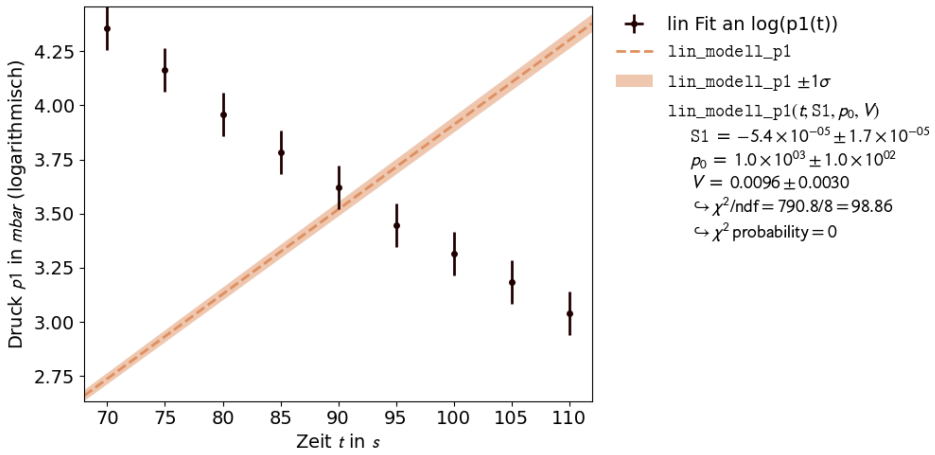
genutzt.

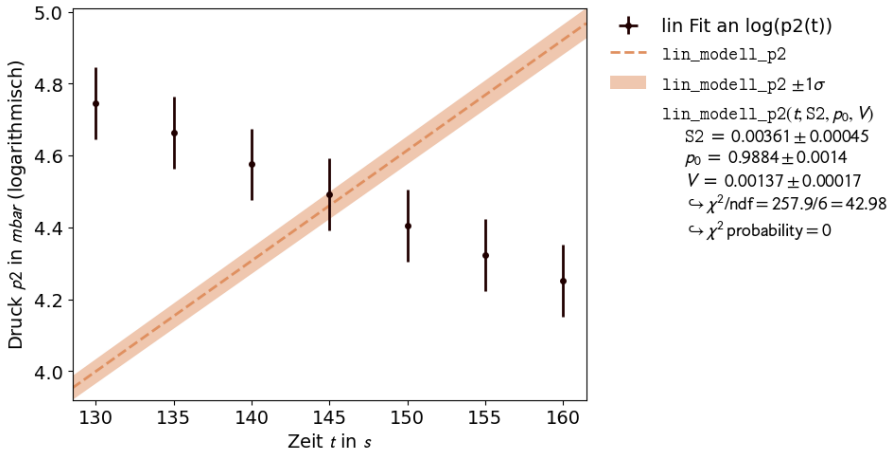
Für die tatsächliche Bestimmung von S_1 und S_2 gehen wir wie in 2.1 vor und nutzen die Modellfunktion in logarithmischer Darstellung:

[14]:
```
def lin_modell_p1(t, S1=saugvermögen2_p1_50mbar, p_0=1000, V=ges_vol_rohr):
    return (- np.log(p_0) * (S1/V) * t)

def lin_modell_p2(t, S2=saugvermögen2_p2_100mbar, p_0=1000, V=ges_vol_rohr):
    return (- np.log(p_0) * (S2/V) * t)

fit_2_2_data1 = np.array([ t_5min[14:23] , np.log(druck_p1_2_2[14:23]) ])
fit_2_2_data2 = np.array([ t_5min[26:33] , np.log(druck_p2_2_2[26:33]) ])

fit_2_2_error1 = np.array([ np.ones(9)*t_5min_std , np.absolute(␣
 ↪druck_p1_2_2[14:23] * druck_std / druck_p1_2_2[14:23] ) ] )
fit_2_2_error2 = np.array([ np.ones(7)*t_5min_std , np.absolute(␣
 ↪druck_p2_2_2[26:33] * druck_std / druck_p2_2_2[26:33] ) ] )

fit_2_2_label1 = [ "Zeit $t$ in $s$" , "Druck $p1$ in $mbar$ (logarithmisch)" ]
fit_2_2_label2 = [ "Zeit $t$ in $s$" , "Druck $p2$ in $mbar$ (logarithmisch)" ]

fit_2_2_title1 = "lin Fit an log(p1(t))"
fit_2_2_title2 = "lin Fit an log(p2(t))"

fit_2_2_const = [ ["p_0", 1000, 100] , ["V", ges_vol_rohr, .1] ]

fit_res1_2_2 = fit_funktion(fit_2_2_data1, lin_modell_p1, fit_2_2_error1,␣
 ↪fit_2_2_label1, fit_2_2_title1, fit_2_2_const)
fit_res2_2_2 = fit_funktion(fit_2_2_data2, lin_modell_p2, fit_2_2_error2,␣
 ↪fit_2_2_label2, fit_2_2_title2, fit_2_2_const)

fit_res1_2_2[2].plot()
fit_res2_2_2[2].plot()
```

```
fit_res1_2_2[2].show()
fit_res2_2_2[2].show()
```

```
Warning: the cost function has been evaluated as infinite. The fit might not
converge correctly.

C:\Users\probz\AppData\Local\Temp\ipykernel_16584\1585440495.py:2:
RuntimeWarning: invalid value encountered in log
  return (- np.log(p_0) * (S1/V) * t)
C:\Users\probz\AppData\Local\Temp\ipykernel_16584\1585440495.py:5:
RuntimeWarning: invalid value encountered in log
  return (- np.log(p_0) * (S2/V) * t)

Warning: the cost function has been evaluated as infinite. The fit might not
converge correctly.
```





Eigentlich kann es nicht sein, dass bei zwei verschiedenen Datensätzen auf die selbe Weise der

falsche Fit herauskommt. Entweder ist bei der Messung ein systematischer Fehler gemacht worden, oder ich bin schon zu lange mit dieser Aufgabe beschäftigt, dass ich einen groben Fehler in der Auswertung gemacht habe und diesen nicht mehr sehen kann.

Beim ersten Fit hat kafe2 zumindest die Constraints nicht ignoriert, es kommt für das Saugvermögen aber ein Wert 3 Größenordnungen niedriger heraus als was eingegeben wurde, und dieser Wert liegt, wie in 2.1 auch schon, selbst 2 Größenordnungen unter dem erwarteten Wert.

Beim zweiten hat kafe2 die Constraints wieder völlig ignoriert, dafür liegt der Wert für das Saugvermögen nur 1 Größenordnung unter dem, was eingegeben wurde.

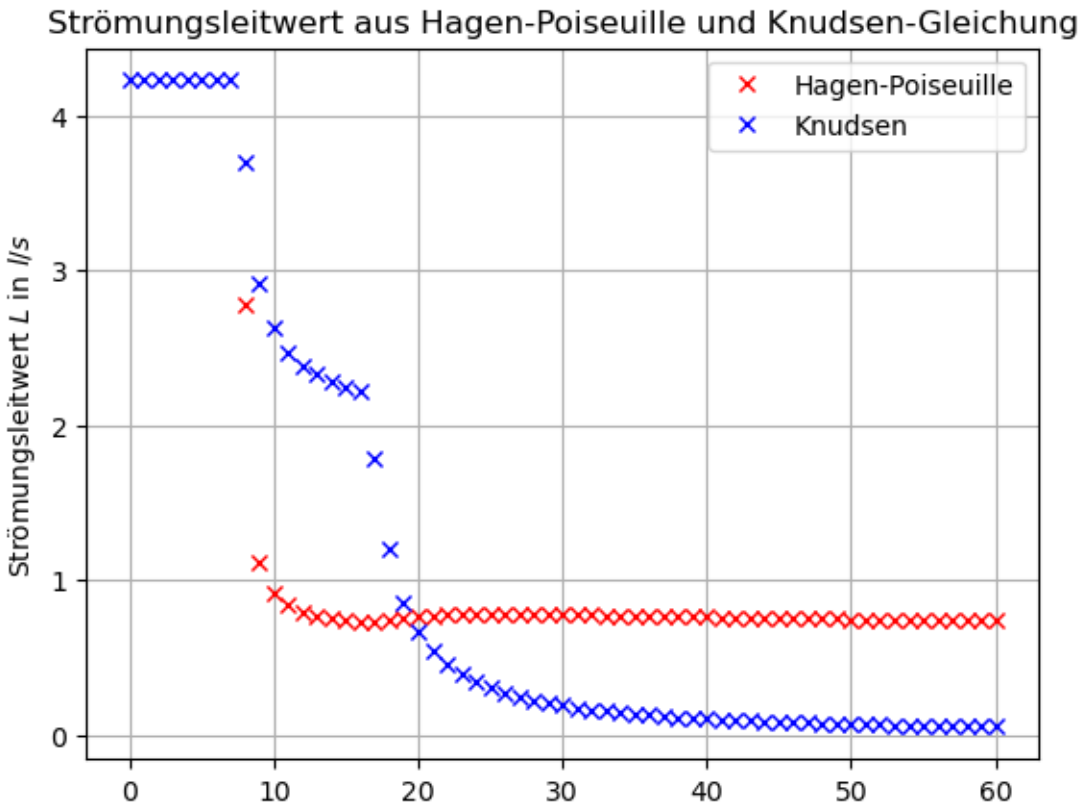
Im Prinzip kann man die Erklärung aus der 2.1, warum das Ergebnis so ausfällt, wie es das tut, wieder genau so anführen.

Schließlich soll noch der Strömungsleitwert des Rohrs berechnet werden. Da die Fits keinerlei brauchbares Saugvermögen liefert, wird, damit zumindest der Rest der Aufgabe ein gutes Ergebnis liefert, der Wert für das Saugvermögen aus dem Datenblatt genutzt: $S_{exp} = 2.5\,\frac{m^3}{h}$.

[15]:
```
saugvermögen_exp = 25/36
delta_p = druck_p2_2_2 - druck_p1_2_2
p_bar = ( druck_p2_2_2 + druck_p1_2_2 ) / 2

plt.plot( druck_p2_2_2 * saugvermögen_exp / delta_p, "rx",␣
 ↪label="Hagen-Poiseuille")
plt.plot(140 * .2**4 * p_bar / 53.0, "bx", label="Knudsen")
plt.ylabel("Strömungsleitwert $L$ in $l/s$")
plt.legend(), plt.grid()
plt.title("Strömungsleitwert aus Hagen-Poiseuille und Knudsen-Gleichung")
```

```
C:\Users\probz\AppData\Local\Temp\ipykernel_16584\183294435.py:5:
RuntimeWarning: divide by zero encountered in divide
  plt.plot( druck_p2_2_2 * saugvermögen_exp / delta_p, "rx", label="Hagen-
Poiseuille")
```

[15]:
```
Text(0.5, 1.0, 'Strömungsleitwert aus Hagen-Poiseuille und Knudsen-Gleichung')
```



Da in der Vorbereitung nicht steht, wie man den Strömungsleitwert berechnen soll, und beide Gleichungen von den Drücken vor und hinter dem Rohr abhängen, die sich natürlich mit der Zeit ändern, kann ich hier keinen konkreten Wert liefern.
Zusätzlich hängt die Hagen-Poiseuille-Gleichung noch von einem dritten Druck p ab, der nicht erklärt ist, daher wurde für obigen Plot einfach geraten, dass dieser Druck p_{aus} entspricht. Da p_{aus} aber schon als solches bezeichnet ist, macht es eigentlich keinen Sinn, dass er noch einmal unter anderem Namen auftaucht.

3.2.3 Aufgabe 2.3: Saugvermögen der TMP

- Nehmen Sie die TMP in Betrieb und bestimmen Sie analog zu **Aufgabe 2.1** das Saugvermögen der TMP als Funktion des Drucks bei IM.
- Stellen Sie S(p) für die DSP und die TMP in einem Diagramm graphisch dar und diskutieren Sie Ihr Ergebnis.
- Schätzen Sie die Kraft ab, mit der die Glasglocke auf die Gummidichtung gedrückt wird.

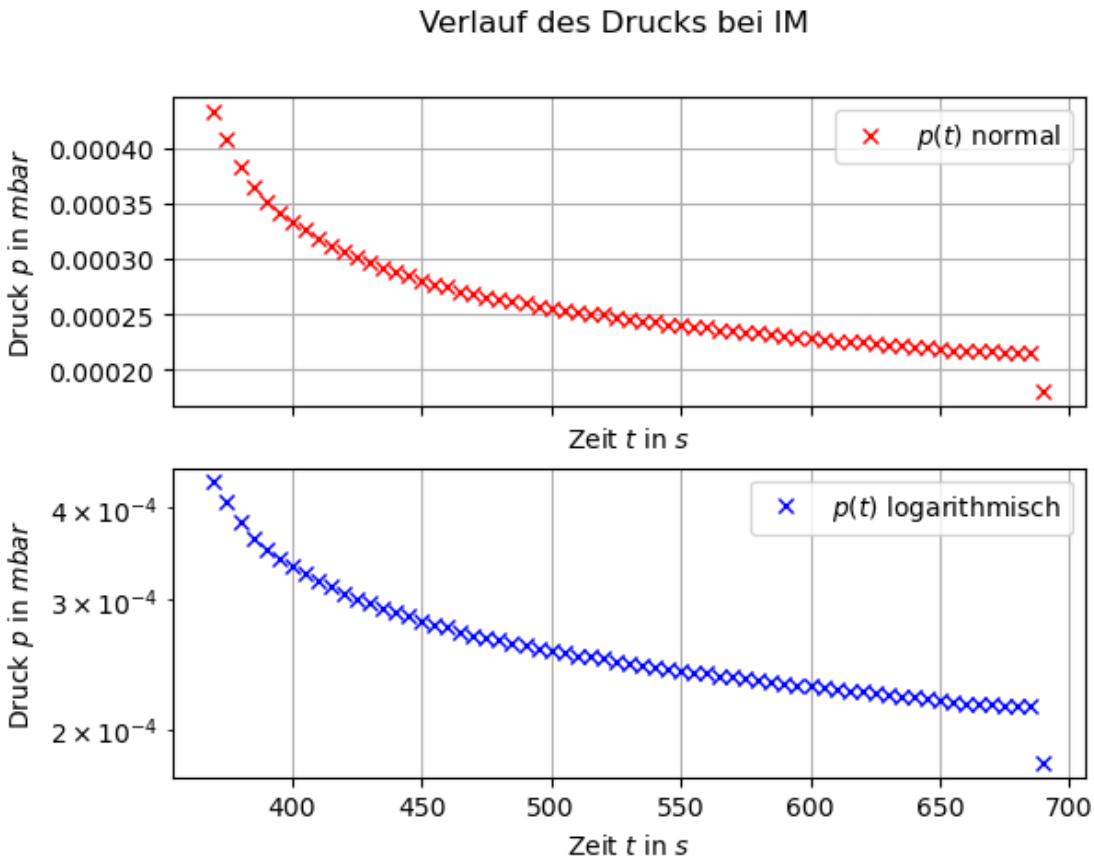
---

[16]:
```
# Anfang 3:20, Ende: 10:15, IM Start: 8:05, IM Norm: 9:25 (T3 bei 0.00338)
druck_t3 = np.array([0.204, 0.196, 0.189, 0.180, 0.174, 0.168, 0.163, 0.158, 0.
↪153, 0.150, 0.147, 0.144,
```

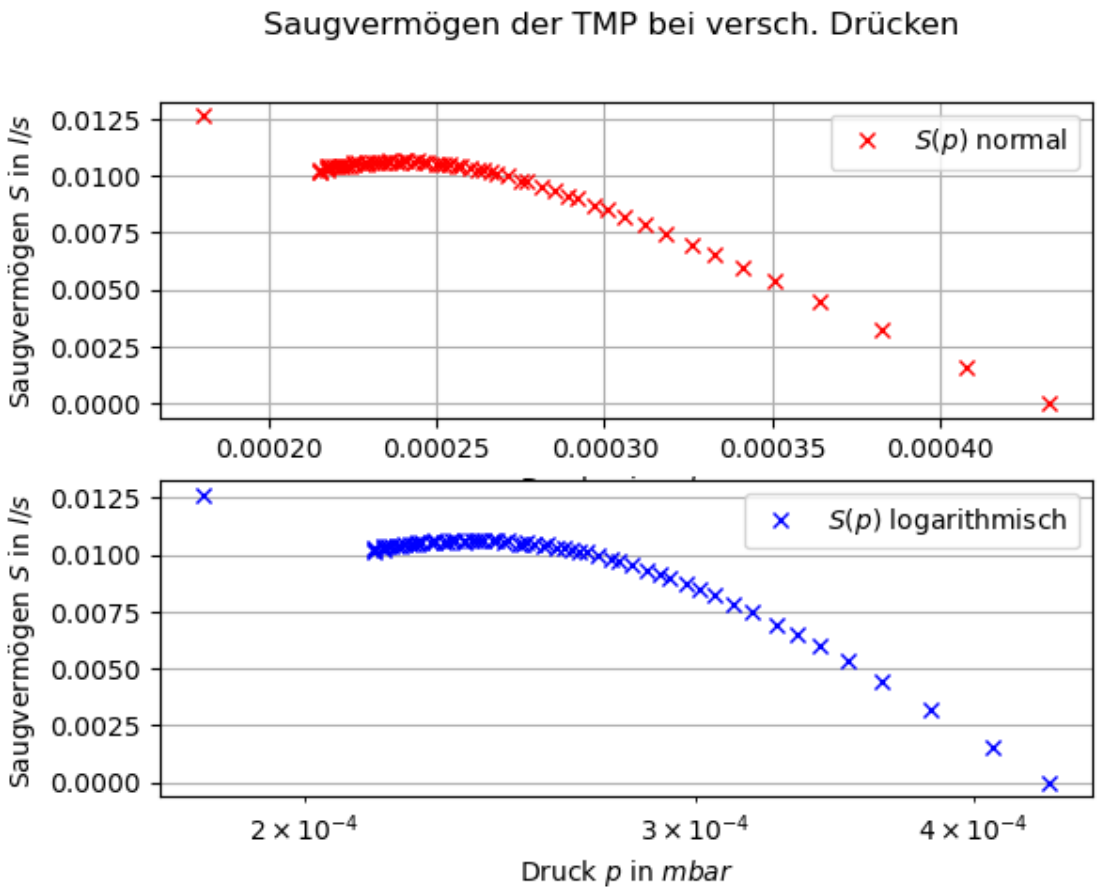
```
                    0.141, 0.138, 0.136, 0.134, 0.132, 0.130, 0.128, 0.127, 0.
 ↪125, 0.124, 0.122, 0.121,
                    0.120, 0.119, 0.118, 0.117, 0.116, 0.114, 0.113, 0.112, 0.
 ↪111, 0.110, 0.109, 0.109,
                    0.108, 0.107, 0.106, 0.106, 0.104, 0.104, 0.103, 0.103, 0.
 ↪102, 0.101, 0.101, 0.100,
                    0.0996, 0.0987, 0.0982, 0.0977, 0.0972, 0.0966, 0.0962, 0.
 ↪0954, 0.0950, 0.0944, 0.0936, 0.0873,
                    0.0791, 0.0675, 0.0543, 0.0410, 0.0297, 0.0206, 0.0144, 0.
 ↪0104, 0.00773, 0.00576, 0.00478, 0.00404,
                    0.00359, 0.00338, 0.00319, 0.00306, 0.00296, 0.00287, 0.
 ↪00281, 0.00271, 0.00266, 0.00264, 0.00259, 0.00254])
# IM Messung: 9:25
druck_im = np.array([4.33, 4.08, 3.83, 3.64, 3.51, 3.41, 3.33, 3.26, 3.18, 3.
 ↪12, 3.06,
                    3.01, 2.97, 2.92, 2.89, 2.85, 2.81, 2.77, 2.75, 2.71, 2.
 ↪68, 2.66, 2.64,
                    2.62, 2.60, 2.57, 2.56, 2.54, 2.52, 2.51, 2.50, 2.47, 2.
 ↪46, 2.44, 2.43,
                    2.41, 2.40, 2.39, 2.38, 2.36, 2.36, 2.34, 2.33, 2.32, 2.
 ↪31, 2.29, 2.29,
                    2.28, 2.26, 2.25, 2.25, 2.24, 2.23, 2.22, 2.21, 2.20, 2.
 ↪19, 2.18, 2.17,
                    2.17, 2.17, 2.15, 2.15, 2.15, 1.80]) * 10**(-4)

t_14min = np.linspace(0, 138, 139) * 5
```

Verlauf des Drucks bei IM, nach der Vorlaufzeit von, in unserem Fall, etwa 1.5 min der TMP. Man sieht keinen großen Unterschied zwischen der normalen und der logarithmischen Darstellung.

[17]:
```
fig4, ax4 = plt.subplots(2, sharex=True)
ax4[0].plot(t_14min[74:], druck_im, "rx", label="$p(t)$ normal")
ax4[0].grid(), ax4[0].legend()

ax4[1].plot(t_14min[74:], druck_im, "bx", label="$p(t)$ logarithmisch")
ax4[1].set_yscale("log")
ax4[1].grid(), ax4[1].legend()

fig4.suptitle("Verlauf des Drucks bei IM")

for ax in ax4:
    ax.set(xlabel="Zeit $t$ in $s$", ylabel="Druck $p$ in $mbar$")
```



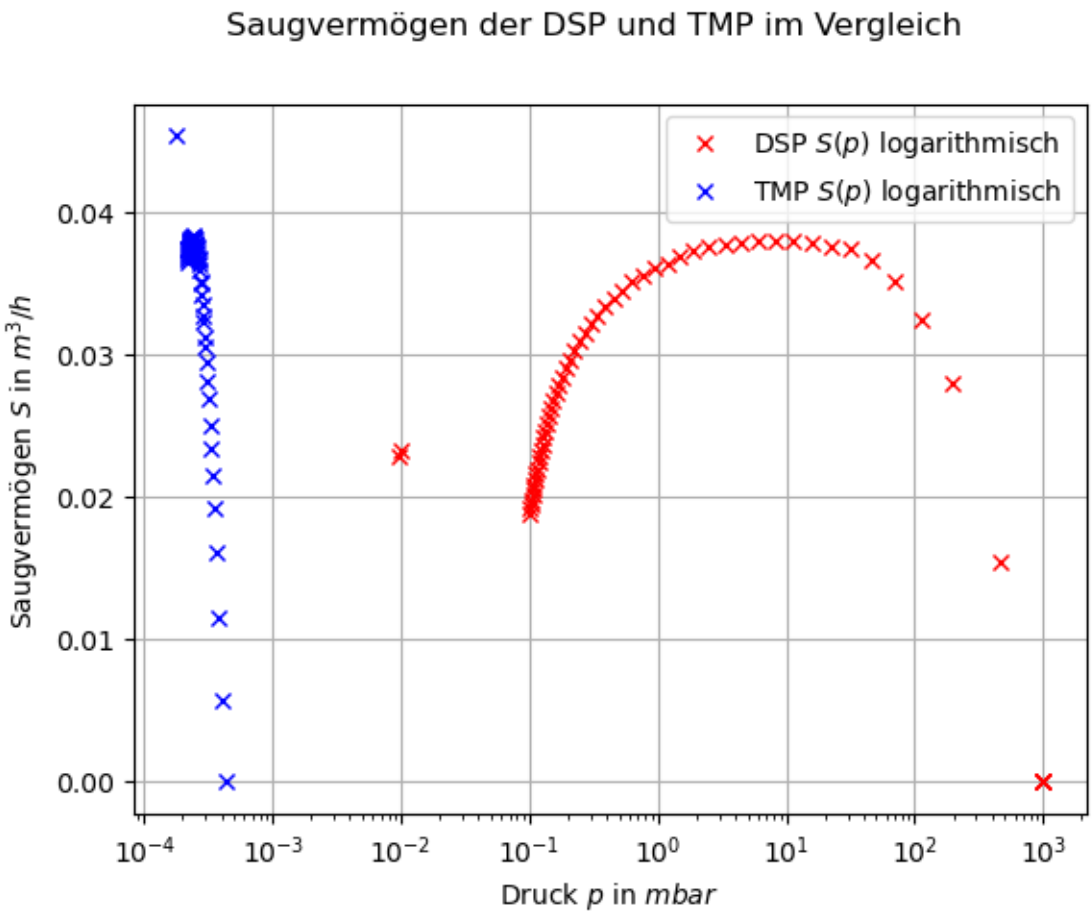
[18]:
```
saugvermögen3 = - ( ges_vol_schlauch * 10**3 / t_14min[74:] ) * np.log(druck_im␣
↪/ druck_im[0])

fig5, ax5 = plt.subplots(2, sharey=True)
ax5[0].plot(druck_im, saugvermögen3, "rx", label="$S(p)$ normal")
ax5[0].grid(), ax5[0].legend()

ax5[1].plot(druck_im, saugvermögen3, "bx", label="$S(p)$ logarithmisch")
ax5[1].set_xscale("log"), ax5[1].grid(), ax5[1].legend()

fig5.suptitle("Saugvermögen der TMP bei versch. Drücken")

for ax in ax5:
    ax.set(xlabel="Druck $p$ in $mbar$", ylabel="Saugvermögen $S$ in $l/s$")
```



Zwischen $p \approx 2.15 \cdot 10^{-4}\, mbar$ und $p \approx 2.5 \cdot 10^{-4}\, mbar$ bleibt das Saugvermögen der TMP relativ konstant, das sieht man schon in der normalen Darstellung. Inzwischen beschleicht mich aber das Gefühl, dass unsere Appatur defekt oder zumindest irgendwo undicht war, denn das Saugvermögen liegt in diesem konstanten Bereich in der Größenordnung 10^{-2}, obwohl in der Vorbereitung $30\, l/s$ gegeben waren, also wieder 3 Größenordnungen Unterschied.

[19]:
```
fig6, ax6 = plt.subplots()

ax6.plot(druck_2_1_auf[1:], saugvermögen, "rx", label="DSP $S(p)$␣
 ↪logarithmisch")
ax6.plot(druck_im, saugvermögen3*3.6, "bx", label="TMP $S(p)$ logarithmisch")
ax6.set_xscale("log")
ax6.legend(), ax6.grid()
ax6.set(xlabel="Druck $p$ in $mbar$", ylabel="Saugvermögen $S$ in $m^3/h$")
fig6.suptitle("Saugvermögen der DSP und TMP im Vergleich")
```

[19]:
```
Text(0.5, 0.98, 'Saugvermögen der DSP und TMP im Vergleich')
```



Wie man erwarten konnte, ist dieser Plot nicht sehr aussagekräftig. Es gibt keinerlei Überschneidung der Saugvermögen der Pumpen, da ihre optimalen Betriebsdrücke Größenordnungen auseinanderliegen. Was man aber sehen kann, ist, dass die Saugvermögen bei diesen jeweiligen optimalen Betriebsdrücken in etwa gleich hoch sind, nämlich bei ungefähr $S = 0.04\,m^3/h$. Das Saugvermögen der TMP sollte aber eigentlich viel höher sein.

Es ist schade, dass diese Aufgabe so gut wie gar nicht geklappt hat und ich zudem nicht verstehe, wieso.

3.3 Aufgabe 3: Experimente im Vakuum

Hinweise zu Aufgabe 1 finden in der Datei Hinweise-Versuchsdurchfuehrung.md.

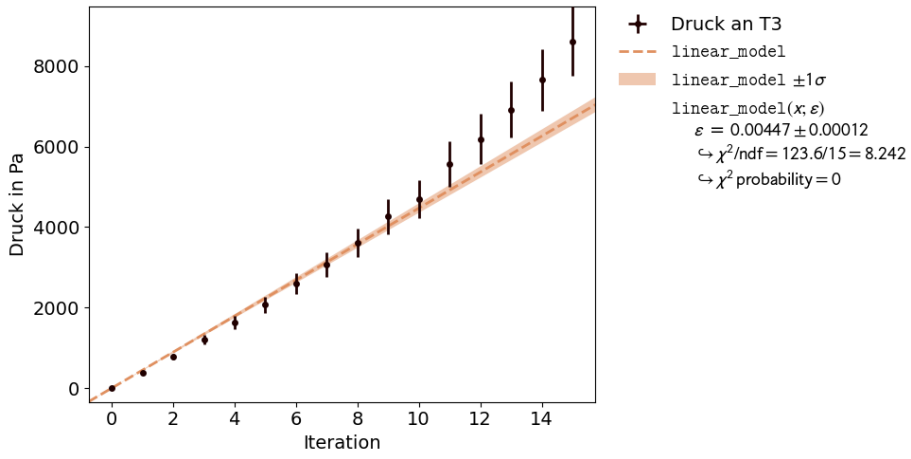
- Für diese Aufgabe führen Sie einige einfache Experimente im Vakuum durch.
- Bearbeiten Sie hierzu die folgenden Aufgaben.

3.3.1 Aufgabe 3.1: Statische Kalibration von T3

Führen Sie mit Hilfe des Referenzvolumes RV eine statische Kalibration des Vakuummeters T3 durch.

[20]:
```
def linear_model(x,epsilon=0.0034):
    return 100000*epsilon*x
```

[21]:
```
druck_t3 = np.array([3.19*10**(-2), 3.73, 7.78, 12.1, 16.3, 20.7, 26.0, 30.6,␣
 ↪36.1, 42.6, 47.0, 55.7, 61.9, 69.2, 76.6, 86.2])*100
iter = 16

xy_data = kafe2.XYContainer(np.arange(iter),druck_t3)
xy_data.label = 'Druck an T3'

fit = kafe2.XYFit(xy_data, model_function=linear_model)
fit.add_error(axis='y', err_val=0.1, relative=True)
fit.data_container.axis_labels = ('Iteration', 'Druck in Pa')
fit.do_fit()

plot=kafe2.Plot(fit)
plot.plot()
plt.show()

print(f'Das rechnerisch bestimmte Epsilon beträgt {0.034 / 9.2}')
```




```
Das rechnerisch bestimmte Epsilon beträgt 0.003695652173913044
```

Im oben stehenden Fit wurde der gemessene Druck in RV gegenüber der Anzahl an erfolgten Iterationen aufgetragen und dann durch einen Fit die Steigung bestimmt. Diese Steigung ist das Verdichtungsverhältnis und beträgt für die genommenen Messdaten $\epsilon = 0.00447 \pm 0.00012$. Das, mithilfe der Formel $\epsilon = \frac{V_{RV}}{V_{RZ}}$ berechnete, beträgt für den vorliegenden Aufbau $\epsilon = 0.0037$. Ein Grund für die Abweichung könnte sein, dass die Apparatur nicht vollständig Luftdicht ist, wodruch der Druck schneller steigt als erwartet.

Außerdem lassen sich die Daten sehr gut durch eine Gerade nähern, man erkennt jedoch gerade

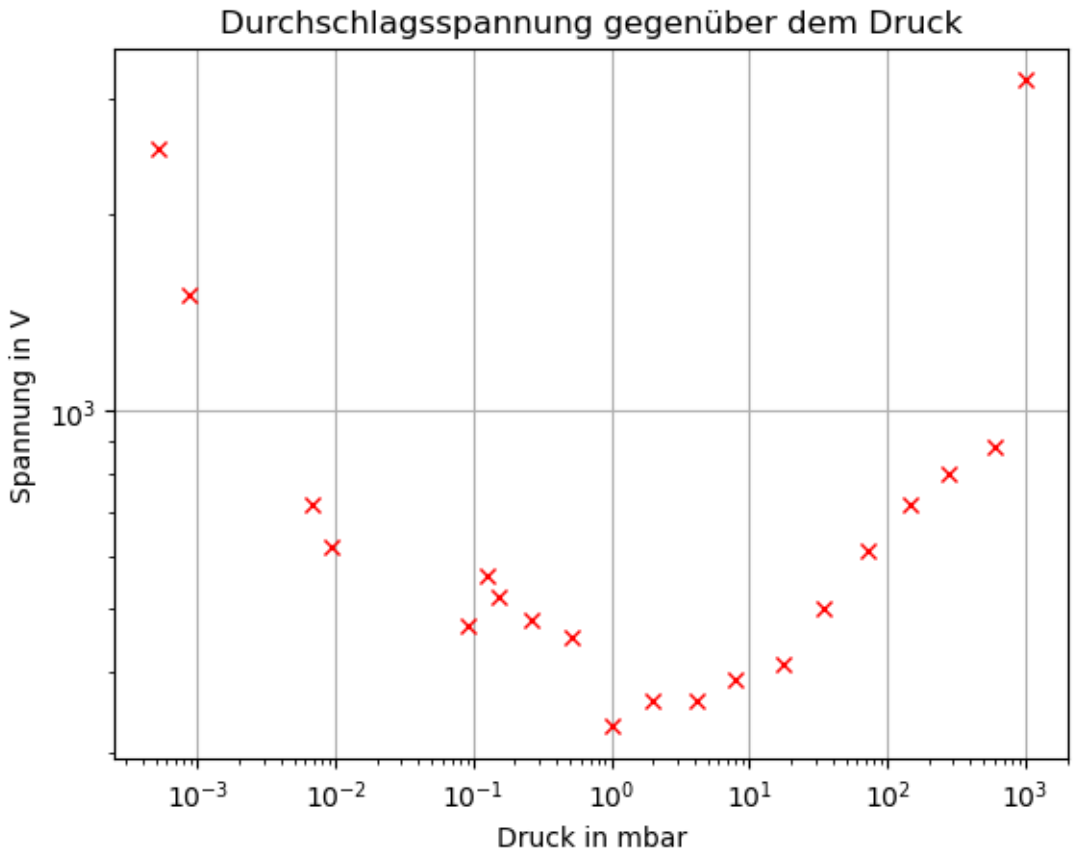
bei hohen Drücken, dass sich ein nichtlinearer Fehler dazuaddiert. Dieser Fehler ist aber laut der Vorbereitung erwartet.

3.3.2 Aufgabe 3.2: Elektrische Durchschlagfestigkeit

- Bestimmen Sie die elektrische Durchschlagfestigkeit der KE als Funktion des Umgebungsdrucks in Luft.
- Diese Aufgabe kann nur an den Apparaturen 41 und 42 durchgeführt werden.

---

[22]:
```
# Druck an T3
p = [1000, 600, 276, 145, 71.4, 35.0, 17.4, 8.01, 4.15, 2.01, 1.00, 0.51, 0.
 ↪266, 0.15, 0.09] # mbar
U = [319.5, 88, 80, 72, 61, 50, 41, 39, 36, 36, 33, 45, 48, 52, 47] # es fehlt␣
 ↪Faktor 10

p_rev = [5.2*10**(-4), 8.8*10**(-4), 68*10**(-4), 92*10**(-4), 1260*10**(-4)]␣
 ↪#mbar
U_rev = [251, 150, 72, 62, 56] # es fehlt Faktor 10

data_p = np.array(p + p_rev[::-1])
data_U = np.array(U + U_rev[::-1]) * 10

U_std = 30

fig = plt.figure()
ax = fig.add_subplot()

ax.plot(data_p,data_U,'rx')
ax.set_xscale('log')
ax.set_yscale('log')

ax.grid()
ax.set_title('Durchschlagsspannung gegenüber dem Druck')
ax.set_xlabel('Druck in mbar')
ax.set_ylabel('Spannung in V')
plt.show()
```



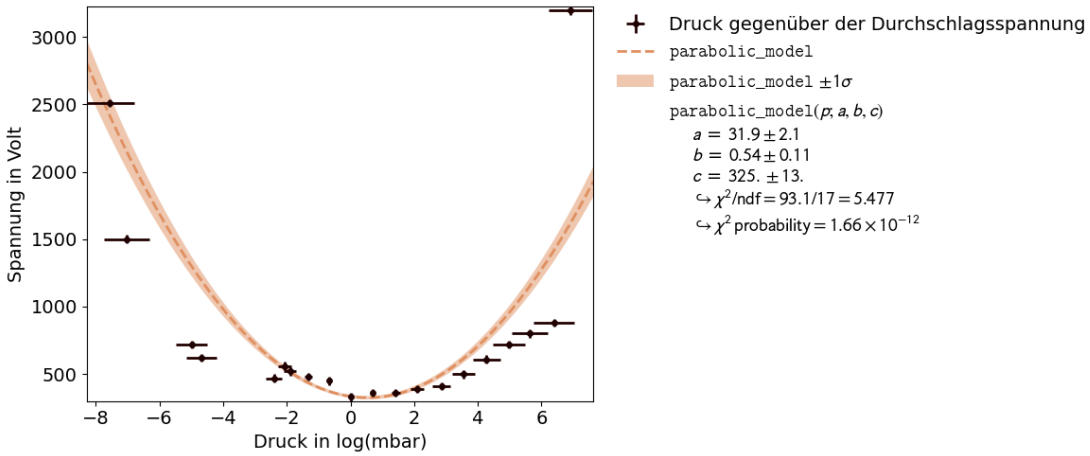
Im oben gezeigten Diagramm ist die Durchschlagsfestigkeit gegenüber dem Druck im inneren der Kuppel aufgetragen. Da dieser Versuch in der vorhandenen Apparatur fehlte, wird hier mit den Daten einer anderen Gruppe gearbeitet. Die Durchschlagsfestigkeit bezieht sich hierbei auf die Spannung, die benötigt wird, um zwischen zwei Kugeln, die sich unter der Vakuumglocke befinden, einen Übersprung der Elektronen zu erzeugen.

Zu Beginn ist die mittlere freie Weglänge gering, weshalb sich die Elektronen aus der Anode nicht sehr weit bewegen können, bevor sie stoßen. Die Entladungen finden also erst bei sehr hohen Spannungen statt. Sinkt nun der Druck, steigt die freie Weglänge und die Elektronen können sich weiter bewegen, wodurch auch bei geringeren Spannungen ein Durschlag stattfindet. Unter einer bestimmten Schwelle muss jedoch wieder mehr Spannung aufgebracht werden, da nicht genug zu ionisierende Moleküle vorhanden sind.

Außerdem war zu beobachten, dass sich für einige Messungen kein Überschlag mehr erkennen lässt, sondern lediglich die Spannung nicht weiter ansteigt. Ebenso ist ab einem bestimmten Punkt der Druck nichtmehr genau einstellbar, da hier durch bedienen der Ventile und der Pumpen nur bedingt Dinge verändert werden können.

[23]:
```
def parabolic_model(p,a,b,c):
    return a*(p-b)**2 + c
```

```
xy_data = kafe2.XYContainer(np.log(data_p),data_U)
xy_data.label = ('Druck gegenüber der Durchschlagsspannung')

fit = kafe2.XYFit(xy_data=xy_data,model_function=parabolic_model)
fit.add_error(axis='x',err_val=0.1,relative=True)
fit.add_error(axis='y',err_val=30)
fit.do_fit()

plot = kafe2.Plot(fit)
plot.y_label = "Spannung in Volt"
plot.x_label = "Druck in log(mbar)"
plot.plot()
plt.show()
```



[24]:
```
b = ufloat(fit.parameter_values[1],fit.parameter_errors[1])
p = 100*np.e**b

M_m = 28
f = 0.74
roh_fl = 807
N_A = 6.022 * 10**23
k_B = 1.38 * 10**-20
T = 293.15
V = M_m * f / (N_A * roh_fl)
r = (3*V/(4*np.pi))**(1/3)
sigma = np.pi * r**2
lamda = k_B*T/(sigma*p)

lamda
```

[24]: 0.0016029148861757385+/-0.0001761316824216527 7

In der Darstellung der Datenpunkte ist eine Parabelform zu erkennen. Aufgrund dessen wird ein Fit durchgeführt um den Druck herauszufinden, bei dem die Spannung, für die ein Durchschlag auftritt, minimal ist. Bei diesem Druck sollte die mittlere freie Weglänge dem Abstand der Kugelelektroden entsprechen. Es ergibt sich somit durch verschiedenen Formeln aus der Vorbereitung $\lambda = 1.602 \pm 0.176mm$

3.3.3 Aufgabe 3.3: Aufdampfen von Indium

Dampfen Sie mit Hilfe einer Schablone bei zwei verschiedenen Drucken Indium auf eine Plexiglasplatte auf.

In diesem Versuch zeigte sich das Ablenkvermögen der Luftmoleküle und man erkennt klar die unterschiedlichen Drücke. Bei höherem Druck sind mehr Luftmoleküle vorhanden, mit denen die aufsteigenden Indiumatome stoßen und so gestreut werden. Diese Streuung resultiert in einem größeren Punkt auf der Plexiglasplatte. Der untere Punkt wurde somit bei hohem und der untere bei niedrigem Druck aufgedampft.