auswertung222

February 11, 2025

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

plt.rcParams.update({'font.size': 12})

%run ../lib.ipynb
```

[2]:
```
class DatenAufgabe1:
    heizstrom = ValErr(0.94, 0.01) * 5.0
    heizspannung = ValErr(4.74, 0.01)
    durchfluss = [ 252.2, 251.2, 254.9, 253.0, 250.3 ]
    drehzahl = [ 316.6, 318.8, 317.6, 315.9, 317.9 ] # übernommen aus Aufagbe 2
    T_zu = ValErr(17.9, 0.1) + 273.15
    T_ab = ValErr(19.6, 0.1) + 273.15

class DatenAufgabe2:
    motorstrom = ValErr(2.1, 0.1)
    motorspannung = ValErr(24.1, 0.1)
    drehzahl = [ 316.6, 318.8, 317.6, 315.9, 317.9 ]
    gefrierzeit = ValErr(180.0, 2.0)

class DatenAufgabe3a:
    heizstrom = ValErr(2.51, 0.01) * 5
    heizspannung = ValErr(11.54, 0.03)
    durchfluss = [ 252.0, 249.9, 243.1, 251.9, 253.3 ]
    T_zu = ValErr(18.0, 0.1) + 273.15
    T_ab = ValErr(21.3, 0.1) + 273.15
    drehzahl = [286.6, 288.8, 289.9, 290.4, 288.9]
    flaechePV = [ 15600, 15990, 15420, 15220 ]

class DatenAufgabe3b08:
    bremskraft = ValErr(0.8, 0.02)
    drehzahl = [ 190.0, 191.1, 192.3, 193.2 ]
    flaechePV = [ 27250, 26710, 27640, 27950 ]

class DatenAufgabe3b06:
    bremskraft = ValErr(0.6, 0.02)
```

```
    drehzahl = [ 211.3, 210.2, 209.4, 210.4 ]
    flaechePV = [ 25570, 25320, 25610, 25540 ]

class DatenAufgabe3b04:
    bremskraft = ValErr(0.4, 0.02)
    drehzahl = [ 229.6, 229.2, 230.8, 229.9 ]
    flaechePV = [ 23420, 23000, 23340, 22870 ]

class DatenAufgabe3b02:
    bremskraft = ValErr(0.2, 0.02)
    drehzahl = [ 253.6, 253.8, 256.0, 255.4 ]
    flaechePV = [ 20220, 21050, 21170, 21120 ]

class Konst:
    C_W = 4180 # J / (kg K)
    rho_W = 997 # kg / m³
    lambda_W = 335 # J / g
    V_W = 1 # ml
```

0.1 Aufgabe 1

[3]:
```
v_dot = ValErr.fromMeasurements(DatenAufgabe1.durchfluss) * (1.0e-6 / 60.)
f = ValErr.fromMeasurements(DatenAufgabe1.drehzahl) * (1. / 60.)
T_diff = DatenAufgabe1.T_ab - DatenAufgabe1.T_zu

print_all(v_dot.strfmtf(4, -6, "Vdot"), f.strfmtf(4, 0, "f"), T_diff.strfmtf(4,␣
 ↪0, "ΔT"), "----------------------")

# Heizleistung
P_H = DatenAufgabe1.heizstrom * DatenAufgabe1.heizspannung

# Motorleistung
P_M = DatenAufgabe2.motorstrom * DatenAufgabe2.motorspannung

# Zugeführte mechanische Arbeit pro Umdrehung
W_M = P_M / f

# An das Kühlwasser abgegebene Wärmemenge, Kalorische Zustandsgleichung
Q_1 = (Konst.C_W * Konst.rho_W * T_diff * v_dot) / f
Q_1.setErr(Q_1.val * np.sqrt(T_diff.relerr()**2 + v_dot.relerr()**2 + f.
 ↪relerr()**2))

# Am Zylinderkopf entzogene Wärmemenge
Q_2 = P_H / f

# Abweichung von der idealen "Wärmebilanz"
```

```
DQ = Q_2 + W_M - Q_1
DQ.setErr(np.sqrt(np.sum([Q_2.err**2, W_M.err**2, Q_1.err**2])))

# Wirkungsgrad
eta = (Q_2 / W_M)

print_all(P_H.strfmtf(4, 0, "P_H"), P_M.strfmtf(4, 0, "P_M"), W_M.strfmtf(4, 0,␣
 ↪"W_M"), Q_1.strfmtf(4, 0, "Q_1"), Q_2.strfmtf(4, 0, "Q_2"), DQ.strfmtf(4, 0,␣
 ↪"ΔQ"), eta.strfmtf(4, 0, " "))
```

```
Vdot = 4.2053e-6 ± 0.0132e-6
f = 5.2893 ± 0.0084
ΔT = 1.7000 ± 0.1414
----------------------
P_H = 22.2780 ± 0.2416
P_M = 50.6100 ± 2.4191
W_M = 9.5683 ± 0.4576
Q_1 = 5.6327 ± 0.4690
Q_2 = 4.2119 ± 0.0462
ΔQ = 8.1474 ± 0.6569
  = 0.4402 ± 0.0216
```

0.2 Aufgabe 2

Kälteleistung

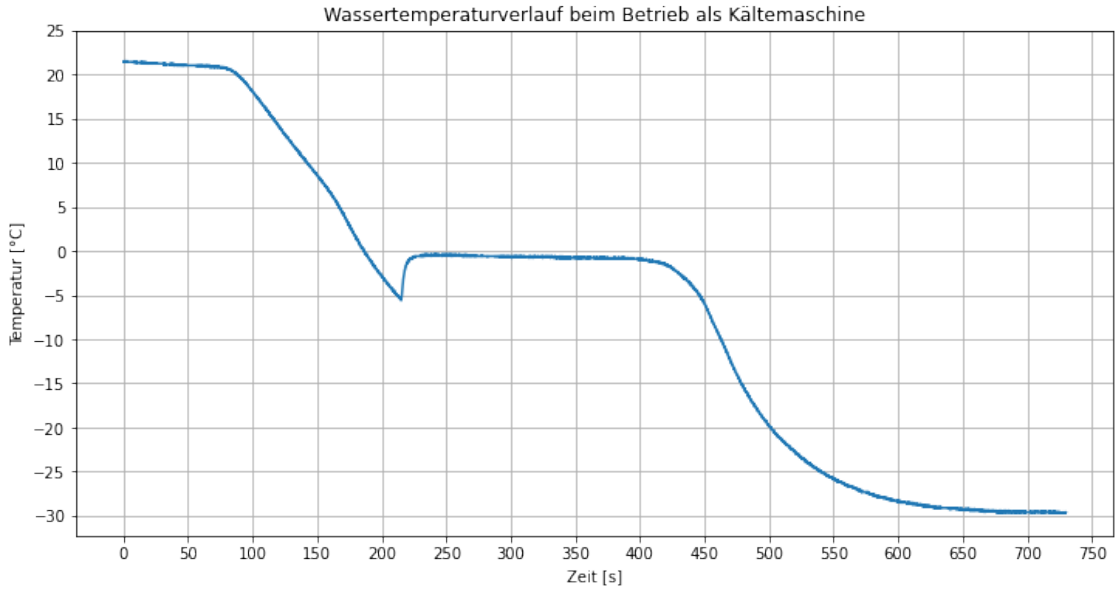
[8]:
```
masse_W = Konst.rho_W * Konst.V_W * 1e-6

print('Kälteleistung aus Gefrierzeit:', (masse_W * Konst.lambda_W * 1e3 /␣
 ↪DatenAufgabe2.gefrierzeit).strfmtf(5, 0))
```

```
Kälteleistung aus Gefrierzeit: 1.85553 ± 0.02062
```

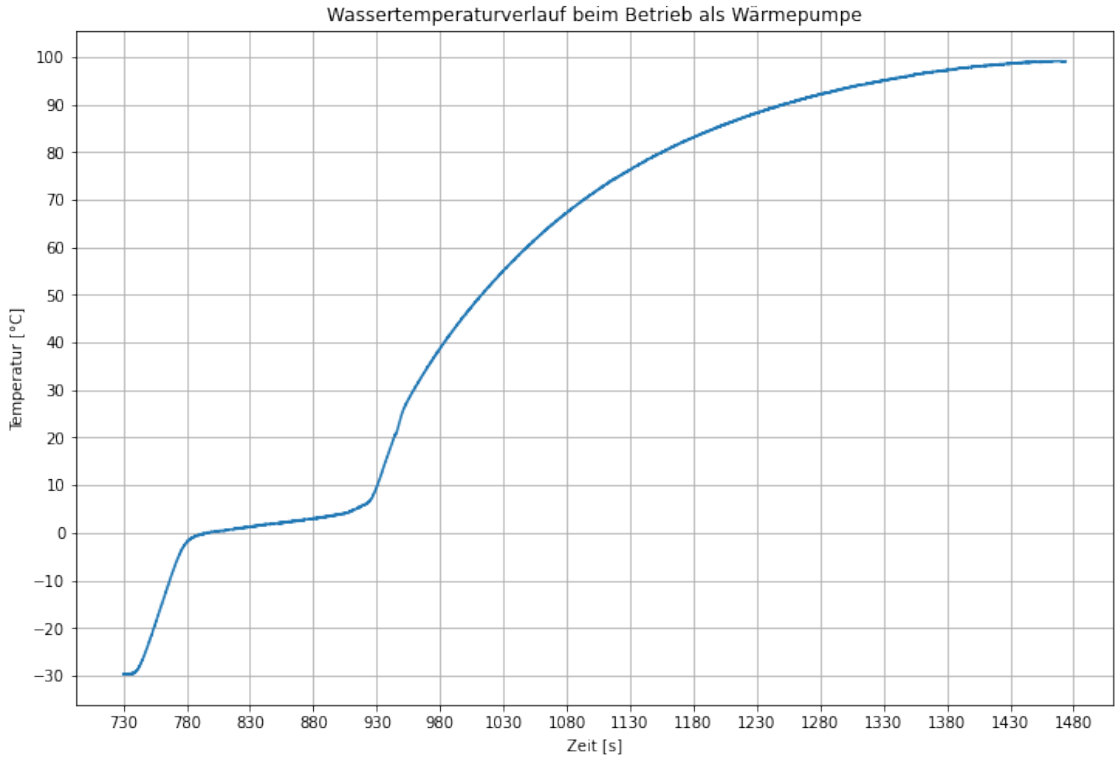
Temperaturverlauf

[9]:
```
temperaturverlauf = np.loadtxt('aufgabe2_wassertemp_daten.txt', skiprows=1,␣
 ↪usecols=(0,1), unpack=True)
```

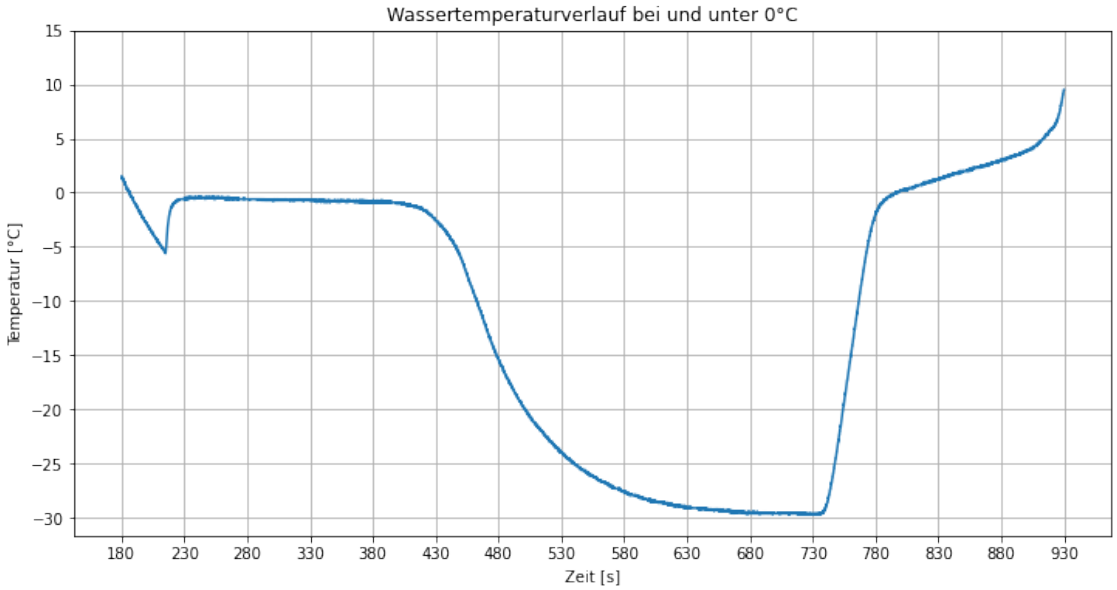
[10]:
```
plt.figure(figsize=(12,6))
plt.plot(temperaturverlauf[0][0:7300], temperaturverlauf[1][0:7300])
plt.ylabel(r'Temperatur [°C]')
plt.xlabel('Zeit [s]')
plt.yticks(np.arange(-30, 30, 5))
plt.xticks(np.arange(0, 760, 50))
plt.title(r'Wassertemperaturverlauf beim Betrieb als Kältemaschine')
plt.grid()
plt.savefig("tempverlauf_kalt.png", format="png")
```



[11]:
```
plt.figure(figsize=(12,8))
plt.plot(temperaturverlauf[0][7300:], temperaturverlauf[1][7300:])
plt.ylabel(r'Temperatur [°C]')
plt.xlabel('Zeit [s]')
plt.yticks(np.arange(-30, 110, 10))
plt.xticks(np.arange(730, 1500, 50))
plt.title(r'Wassertemperaturverlauf beim Betrieb als Wärmepumpe')
plt.grid()
plt.savefig("tempverlauf_warm.png", format="png")
```



[12]:
```
plt.figure(figsize=(12,6))
plt.plot(temperaturverlauf[0][1800:9300], temperaturverlauf[1][1800:9300])
plt.ylabel(r'Temperatur [°C]')
plt.xlabel('Zeit [s]')
plt.yticks(np.arange(-30, 20, 5))
plt.xticks(np.arange(180, 950, 50))
plt.title(r'Wassertemperaturverlauf bei und unter 0°C')
plt.grid()
plt.savefig("tempverlauf_unter0.png", format="png")
```



0.3 Aufgabe 3

Leerlaufmessung

[19]:
```
v_dot = ValErr.fromMeasurements(DatenAufgabe3a.durchfluss) * (1.0e-6 / 60.)
f = ValErr.fromMeasurements(DatenAufgabe3a.drehzahl)  * (1. / 60.)
T_diff = DatenAufgabe3a.T_ab - DatenAufgabe3a.T_zu

P_el = DatenAufgabe3a.heizstrom * DatenAufgabe3a.heizspannung
Q_el = P_el / f

P_ab = Konst.C_W * Konst.rho_W * T_diff * v_dot
Q_ab = P_ab / f

#1 hPa cm^3 = 10^-4 J

W_pV = Q_pV = ValErr.fromMeasurements(DatenAufgabe3a.flaechePV) * 10**-4
P_pV = W_pV * f

Q_V = Q_el - Q_ab - W_pV
Q_V.setErr(np.sqrt(np.sum([Q_el.err**2, Q_ab.err**2, W_pV.err**2])))

eta_th = W_pV / Q_el

print_all(f.strfmtf(4, 0, "f"), P_el.strfmtf(4, 0, "P_el"), Q_el.strfmtf(4, 0,␣
 ↪"Q_el"), P_ab.strfmtf(4, 0, "P_ab"), Q_ab.strfmtf(4, 0, "Q_ab"),
```

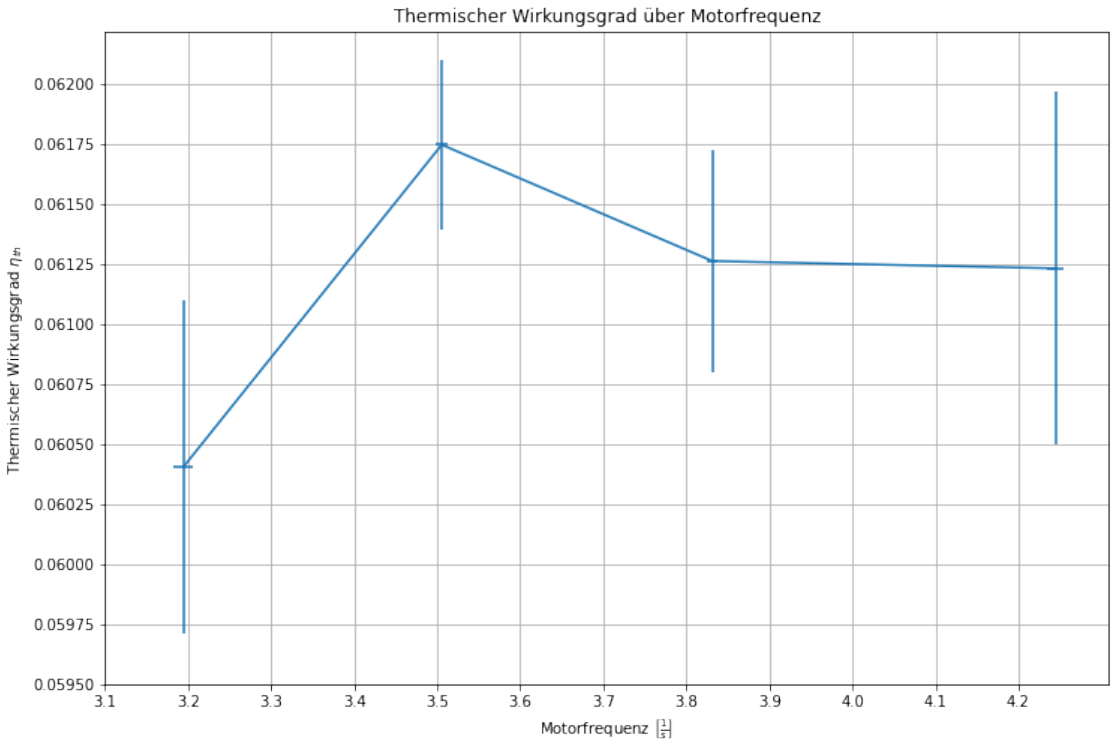
```
          P_pV.strfmtf(4, 0, "P_pV"), Q_pV.strfmtf(4, 0, "Q_pV"), Q_V.
 ↪strfmtf(4, 0, "Q_V"), eta_th.strfmtf(5, 0, "eta_th"),␣
 ↪"--------------------------------")


def eta_err_calc(var_w, var_f, var_p):
    eta_value = (var_w.val * var_f.val) / var_p.val
    eta_error = math.sqrt(np.sum([
        ((var_f.val / var_p.val) * var_w.err )**2 +
        ((var_w.val / var_p.val) * var_f.err )**2 +
        ((var_w.val * var_f.val / var_p.val**2) * var_p.err )**2]))

    return ValErr(eta_value, eta_error)

def eta_brems(daten):
    W_D_br = 0.25 * daten.bremskraft * 2 * np.pi
    W_pV_br = ValErr.fromMeasurements(daten.flaechePV) * 10**-4
    f_br = ValErr.fromMeasurements(daten.drehzahl)  * (1. / 60.)
    eta_th_br = eta_err_calc(W_pV_br, f_br, P_el)
    eta_eff_br = eta_err_calc(W_D_br, f_br, P_el)

    print_all(daten.bremskraft.strfmtf(2, 0, "F"), f_br.strfmtf(4, 0, "f"),␣
 ↪W_D_br.strfmtf(4, 0, "W_D"), W_pV_br.strfmtf(4, 0, "W_pV"), eta_th_br.
 ↪strfmtf(5, 0, "eta_th"), eta_eff_br.strfmtf(5, 0, "eta_eff"),␣
 ↪"--------------------------------")

    return (eta_th_br, eta_eff_br, f_br)

etas = [eta_brems(DatenAufgabe3b08()), eta_brems(DatenAufgabe3b06()),␣
 ↪eta_brems(DatenAufgabe3b04()), eta_brems(DatenAufgabe3b02())]
```

```
f = 4.8153 ± 0.0109
P_el = 144.8270 ± 0.6890
Q_el = 30.0762 ± 0.1584
P_ab = 57.3117 ± 2.4912
Q_ab = 11.9019 ± 0.5180
P_pV = 7.4915 ± 0.0806
Q_pV = 1.5557 ± 0.0164
Q_V = 16.6185 ± 0.5420
eta_th = 0.05173 ± 0.00061
--------------------------------
F = 0.80 ± 0.02
f = 3.1942 ± 0.0116
W_D = 1.2566 ± 0.0314
W_pV = 2.7388 ± 0.0267
eta_th = 0.06040 ± 0.00069
eta_eff = 0.02772 ± 0.00071
```

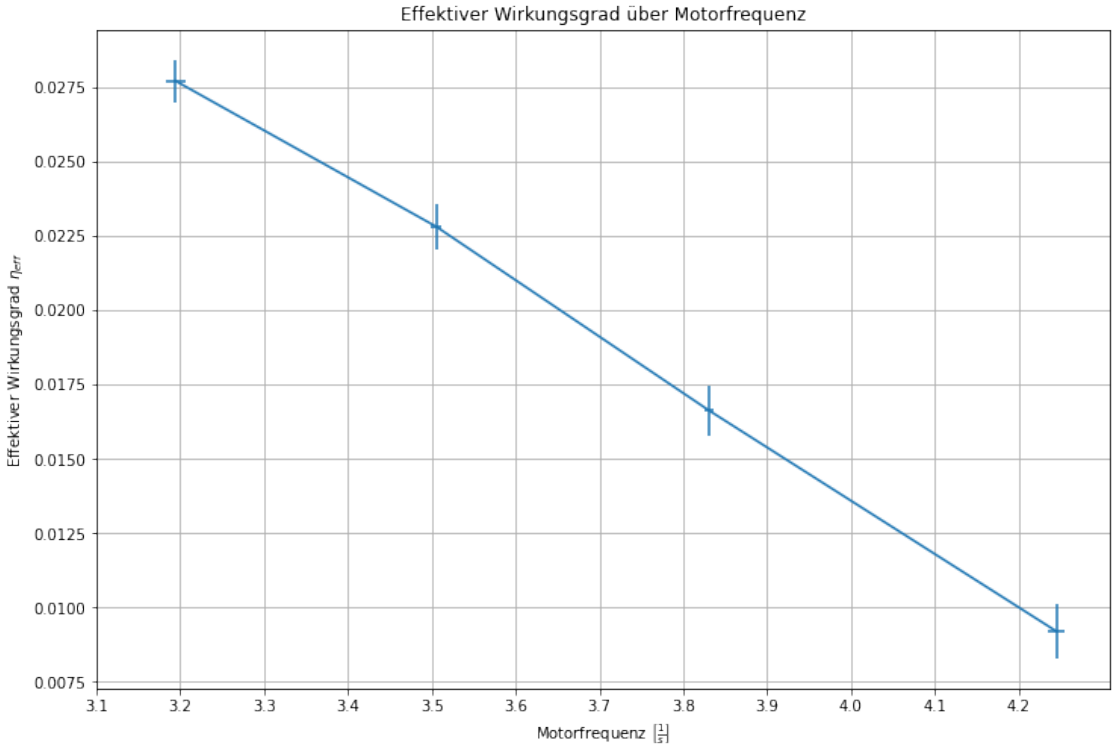
```
--------------------------------
F = 0.60 ± 0.02
f = 3.5054 ± 0.0065
W_D = 0.9425 ± 0.0314
W_pV = 2.5510 ± 0.0065
eta_th = 0.06174 ± 0.00035
eta_eff = 0.02281 ± 0.00077
--------------------------------
F = 0.40 ± 0.02
f = 3.8312 ± 0.0057
W_D = 0.6283 ± 0.0314
W_pV = 2.3157 ± 0.0132
eta_th = 0.06126 ± 0.00046
eta_eff = 0.01662 ± 0.00084
--------------------------------
F = 0.20 ± 0.02
f = 4.2450 ± 0.0099
W_D = 0.3142 ± 0.0314
W_pV = 2.0890 ± 0.0225
eta_th = 0.06123 ± 0.00073
eta_eff = 0.00921 ± 0.00092
--------------------------------
```

[20]:
```
eta_th_val = [x[0].val for x in etas]
eta_th_err = [x[0].err for x in etas]
eta_eff_val = [x[1].val for x in etas]
eta_eff_err = [x[1].err for x in etas]
f_br_val = [x[2].val for x in etas]
f_br_err = [x[2].err for x in etas]
```

[21]:
```
plt.figure(figsize=(12,8))
plt.errorbar(f_br_val, eta_th_val, xerr=f_br_err, yerr=eta_th_err)
plt.xticks(np.arange(3.1, 4.3, 0.1))
plt.yticks(np.arange(0.0595, 0.062, 0.00025))
plt.ylabel(r'Thermischer Wirkungsgrad $\eta_{th}$')
plt.xlabel(r'Motorfrequenz $\left[\frac{1}{s}\right]$')
plt.title(r'Thermischer Wirkungsgrad über Motorfrequenz')
#plt.gca().invert_xaxis()
plt.grid()
plt.savefig("eta_th_freq.png", format="png")
```



[22]:
```
plt.figure(figsize=(12,8))
plt.errorbar(f_br_val, eta_eff_val, xerr=f_br_err, yerr=eta_eff_err)
plt.xticks(np.arange(3.1, 4.3, 0.1))
plt.yticks(np.arange(0.0075, 0.0280, 0.0025))
plt.ylabel(r'Effektiver Wirkungsgrad $\eta_{eff}$')
plt.xlabel(r'Motorfrequenz $\left[\frac{1}{s}\right]$')
plt.title(r'Effektiver Wirkungsgrad über Motorfrequenz')
#plt.gca().invert_xaxis()
plt.grid()
plt.savefig("eta_eff_freq.png", format="png")
```


Effektiver Wirkungsgrad über Motorfrequenz
Effektiver Wirkungsgrad η_{eff}
0.0275
0.0250
0.0225
0.0200
0.0175
0.0150
0.0125
0.0100
0.0075
3.1
3.2
3.3
3.4
3.5
3.6
3.7
3.8
3.9
4.0
4.1
4.2
Motorfrequenz $[\frac{1}{s}]$

[ ]: