FYS1120 Elektromagnetisme

Labøving

Grunnleggende elektromagnetisk måleteknikk.

Kristian Tuv, Hilde Solesvik Skeie og Øyvind Sigmundson Schøyen.

November 10, 2014

Innhald

Indre resistans i eit voltmeter

Oppgåve 1.1

Me lader fort opp kondensatoren ved hjelp av batteriet. Me koplar deretter ut batteriet og koplar kondensatoren til voltmeteret. Programmet `Oppgave1.1.py` plottar verdiane våre og skriv ut til fil. Me får då figure

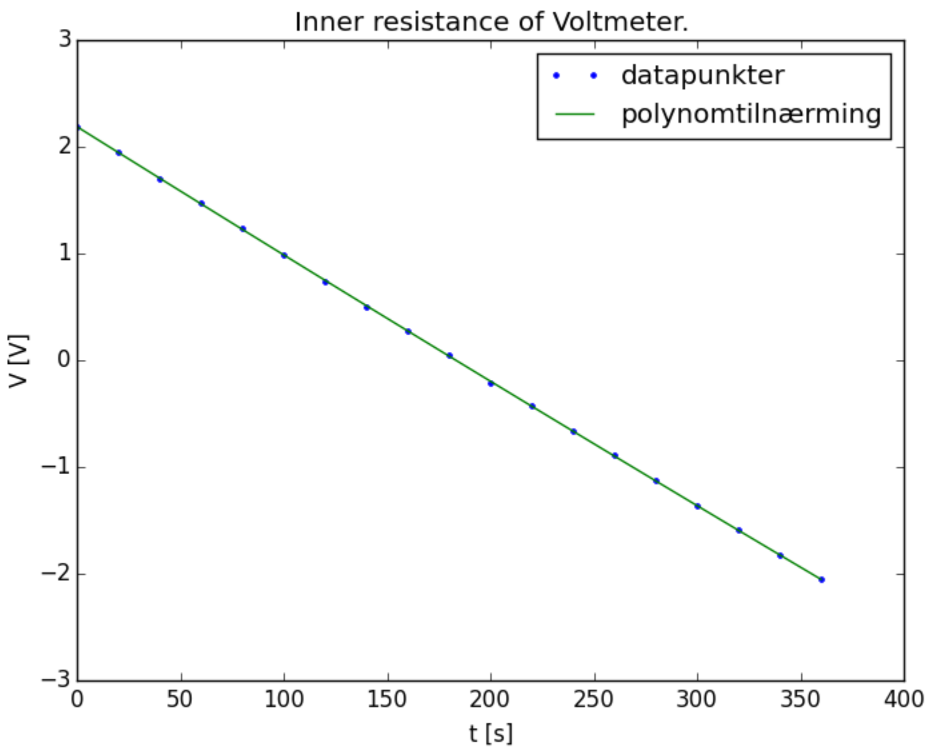

Figure 1: Me ser at målepunkta synker lineært for $\ln U$. Polynomtilnærminga stemmer bra med måleverdiane.

Programmet finner τ ved

$$\tau = -\frac{\Delta t}{\Delta(\ln U)}.$$

Då finner me den indre resistansen ved

$$\tau = RC.$$

Programmet gjer oss utskrifta

Tau is equal to: 84.8454 s
Inner resistance: 1.02223e+07 ohm

PRELAB-Oppgåve 1

Me nytter uttrykket for ladninga til ein kondensator som vert utlada. Denne finn me i boka og er gjeve ved

$$q = Q_0 e^{-t/RC}.$$

No setter me inn uttrykket for ladninga gjeve ved spenning og kapasitans.

$$C = \frac{Q}{V_{ab}} \qquad \Rightarrow \qquad Q = CV_{ab}.$$

Då finner me for $V_0 = 2V_1$ at motstanden er gjeve ved

$$\begin{aligned}
&CV_1 = CV_0 e^{-t/RC} && \Rightarrow \frac{1}{2}V_0 = V_0 e^{-t/RC} \\
\Rightarrow \quad & \frac{1}{2} = e^{-t/RC} && \Rightarrow \quad \ln\frac{1}{2} = -\frac{t}{RC} \\
\Rightarrow \quad & \ln 1 - \ln 2 = -\frac{t}{RC} && \Rightarrow \quad \ln 2 = \frac{t}{RC} \\
\Rightarrow \quad & RC = \frac{t}{\ln 2} && \Rightarrow \quad R = \frac{t}{C \ln 2} \\
\Rightarrow \quad & R = \frac{20\text{ s}}{(1 \times 10^{-6}\text{ F}) \ln 2} \approx 29\text{ M}\Omega.
\end{aligned}$$

Oppgåve 1.2

Når me lader opp kondensatoren så skjer dette motstandsfritt (med unntak av leidningar). Ved utladning tek det lengre tid då kondensatoren må gjennom voltmeteret som har ein enorm indre resistans.

Indre resistans i eit amperemeter

Oppgåve 2.1

Me koplar resistansen og amperemeteret i serie og koplar voltmeteret i parallell med amperemeteret. Måleverdiane er gjeve ved

R [Ω]	I [A]	V [V]
500	0.01728	0.20061
700	0.01245	0.14456
1000	0.00877	0.10184
1200	0.00733	0.08507
1500	0.00588	0.068224

PRELAB-Oppgåve 2

Me nytter det same programmet me nytta i utrekningane våre av Hall-effekten.

```
# coding: utf-8
from numpy import polyfit, polyval, zeros, array
from matplotlib.pylab import plot, show, title, xlabel, ylabel,\
        legend, hold, savefig

"""
Klasse som les punkter frå ei fil, interpolerer med eit polynom og
plottar resultatet mot einannan.
"""
class EvaluatePoints:

    """
    Konstruktør som tek imot ei fil med x- og y-verdiar.
    """
    def __init__(self, datapoints):
        self.datapoints = datapoints
        # Finn antal punkter i fila.
        with open(self.datapoints, 'r') as f:
            self.n = sum(1 for line in f)

    """
    Metode som les verdiane frå fila og lagrar dei i klassa.
    """
    def storeValues(self):
        self.points = zeros((2, self.n))
        with open(self.datapoints, 'r') as f:
            counter = 0
```

```
        for values in f:
            self.points[0, counter] = float(values.split()[0])
            self.points[1, counter] = float(values.split()[1])
            counter += 1

"""
Metode som nyttar numpy-biblioteket til å interpolere punkta
med eit polynom.
"""
def interpolate(self, deg, evalPoint):
    # Lager eit polynom av grad 'deg' ut frå punkta.
    self.poly = polyfit(self.points[0, :], self.points[1, :], deg)
    self.evaluated = zeros(self.n)
    for i in range(self.n):
        self.evaluated[i] = polyval(self.poly, self.points[0, i])

"""
Metode som plottar datapunkta og polynomet mot einannan.
"""
def plotPoints(self, TITLE, saveName, xlab, ylab):
    plot(self.points[0, :], self.points[1, :], '.')
    hold('on')
    plot(self.points[0, :], self.evaluated, '-')
    hold('off')
    title(TITLE)
    xlabel(xlab)
    ylabel(ylab)
    legend(('datapunkter', 'polynomtilnærming',), loc=1)
    savefig(saveName)
    show()
```

Oppgåve 2.2

Då får me ut målepunkta med polynomtilnærminga

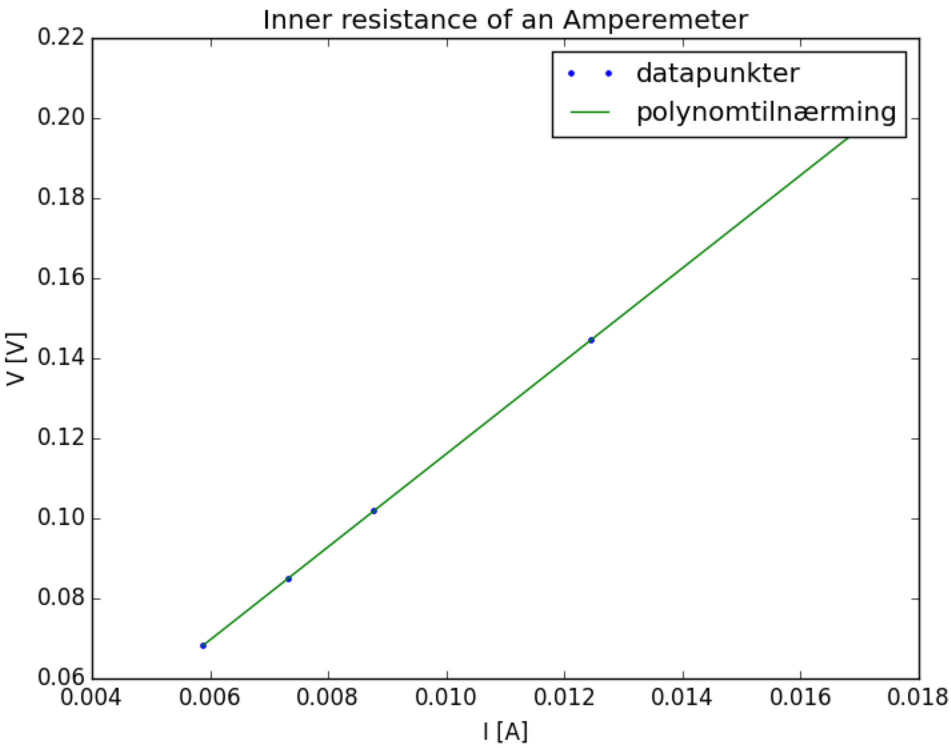

Figure 2: Me får ein ny lineær graf kor tilnærminga fylgjer tett.

Programmet `Oppgave2.1.py` finner den indre resistansen ved

$$R = \frac{\Delta V}{\Delta I}.$$

Me får utskrifta

Inner resistance of amperemeter: 11.6128 ohm

Indre resistans i eit termoelement (Peltier-element)

Oppgåve 3.1

Me kopla Peltier-elementet til voltmeteret og la ei hand over elementet. Me såg då at spenninga steig drastisk som ein følgje av temperaturtilførsel frå handa. Når me snudde side på elementet vert spenninga negativ når me auka temperaturen. Ved kopling til ei straumkjelde merka me at elementet vart varmare på den eine sida og kaldare på den andre.

Oppgåve 3.2

No nyttar me ein kopp med varmt vatn til å varme opp Peltier-elementet og koplar det til voltmeteret i serie med resistansen. Måleverdiane ser me i tabellen under.

R [Ω]	V [V]
1	0.119
1.5	0.1376
2.5	0.162
4	0.1814
10	0.2072

Oppgåve 3.3

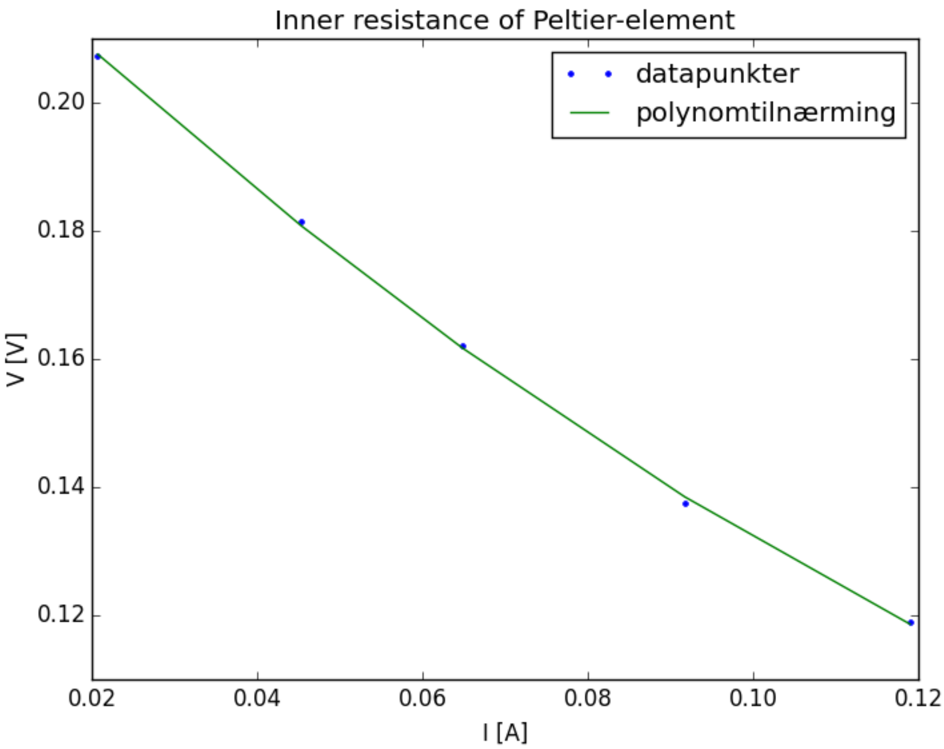

Figure 3: Me har ei ny lineær nedgang i potensialet som ein funksjon av straum.

Me forventa ei lineær nedgang. Programmet `Oppgave3.2.py` gjer oss utskrifta

EMS of Peltier-element: 0.222099 V
Inner resistance of Peltier-element: 0.897436 ohm

Me fann R_i ved formelen

$$R_i = -\frac{\Delta V}{\Delta I},$$

kor minusteiknet kjem frå omforming av formelen

$$U_{ab} = \varepsilon - R_i I. \tag{1}$$

Me fann ε ved likning (1) for mange forskjellige verdiar og såg at han var konstant.

Firepunktsmåling av resistans

Oppgåve 4.1

Me kopla amperemeteret og motstanden i serie. Fyrst brukte me ei firepunktsmåling kor voltmeteret gjekk gjennom motstanden og deretter over. Me bytta mellom ein aluminiumsmotstand og ein koppermotstand. Måleverdiar samt utrekna verdiar for motstand ved R_4 for firepunktsmåling og R_2 ved topunktsmåling. Fyrste tabell gjer oss for aluminium og den andre for koppar.

R_4 [Ω]	I_4 [A]	V_4 [V]	R_2 [Ω]	I_2 [A]	V_2 [V]
5.73123e-05	1.012	5.8e-05	0.00390258	1.006	0.003926
5.62867e-05	1.2614	7.1e-05	0.00385839	1.257	0.00485
5.62541e-05	1.511	8.5e-05	0.00383543	1.507	0.00578
5.61862e-05	1.762	9.9e-05	0.00382336	1.755	0.00671
5.60701e-05	1.9975	0.000112	0.00381909	2.001	0.007642

R_4 [Ω]	I_4 [A]	V_4 [V]	R_2 [Ω]	I_2 [A]	V_2 [V]
3.27381e-05	1.008	3.3e-05	0.0020854	1.007	0.0021
3.18725e-05	1.255	4e-05	0.00199521	1.253	0.0025
3.17355e-05	1.5125	4.8e-05	0.00196559	1.511	0.00297
3.125e-05	1.76	5.5e-05	0.0019364	1.761	0.00341
3.14685e-05	2.002	6.3e-05	0.00191617	2.004	0.00384

Me ser at resistansane held seg tilnærma konstant uavhengig av straum. Differansen går som ein faktor 1.0×10^{-2}.

Magnetfeltet til jordkloten

PRELAB-Oppgåve 3

Me nyttar definisjonen av Faradays lov. Då kan me skrive

$$\varepsilon = -N\frac{d\Phi_B}{dt} = -\frac{d}{dt}\left(NBA\cos(\phi(t))\right),$$

kor $\phi(t) = \omega t$. Me deriverer med hensyn på tid og setter inn $\varepsilon = \varepsilon_0$. Det gjer oss

$$\varepsilon_0 = NBA\omega\sin(\omega t). \tag{2}$$

Me ser at forholdet mellom ω og $t_2 - t_1$ gjer oss vinkelen $\phi(t_2 - t_1)$. Då vil

$$\omega \propto \frac{1}{t_2 - t_1}.$$

Frå likning (2) ser me at ε_0 vil ha sin største verdi når $\omega t = \frac{\pi}{2}$.

Oppgåve 5.1

Me brukte eit kompass til å avgjere vinkelen mellom bakken og B-feltet til jorda. Deretter satte me spolen normalt på retninga til B-feltet. Me dreier spolen med ein konstant vinkelhastighet 180 grader. Me bruker målepunkta under.

t [s]	ε_0 [V]
4.37	0.0006
2.09	0.0011
1.36	0.0015
1.01	0.0028
7.41	0.0004
9.65	0.0002
0.66	0.003

Programmet `Oppgave5.py` gjer oss gjennomsnittsverdien for B ved å rekne ut

$$B = \frac{\varepsilon t}{4NA}.$$

Denne formelen får me ved å sjå på grafen i oppgåveteksten og gjer ei tilnærming ved å behandle kurva som to trekantar.

Mean value of B: 5.41006e-05 T

Dette er ei god tilnærming då magnetfeltet til jordkloten ligg mellom 25 μT - 65 μT.

Me kan ikkje gje ei tilnærming av B ved å ta gjennomsnittet av ε_0 og $t_2 - t_1$ separat då dei er avhengige av einannan.

Programma

Oppgave1.1.py

```
import os, sys
sys.path.append(os.path.abspath('../../Hall-effekt/src/'))
from PRELABOppgave1 import EvaluatePoints
from numpy import exp, log, array
from matplotlib.pylab import plot, show, xlabel, ylabel, title, savefig

U_list = array([8.9110, 7.0, 5.5, 4.35, 3.44, 2.70, 2.1, 1.65, 1.32, 1.05,\
                0.81, 0.65, 0.515, 0.410, 0.324, 0.256, 0.204, 0.161, 0.128])
t_list = array([0, 20, 40, 60, 80, 100, 120, 140, 160, 180, 200, 220,\
                240, 260, 280, 300, 320, 340, 360])

with open('datapointsIRV.txt', 'w') as f:
    for i in range(U_list.size):
        f.write("%g \t %g\n" % (t_list[i], log(U_list[i])))

tau = -1.0/((log(U_list[-1]) - log(U_list[0]))/float(t_list[-1] - t_list[0]))

with open("InnerRV.txt", 'w') as f:
    f.write("Tau is equal to: %g s\\\\ \n" % tau)
    f.write("Inner resistance: %g ohm" % (tau/float(8.3e-6)))

EP = EvaluatePoints('datapointsIRV.txt')
EP.storeValues()
EP.interpolate(2, 0)
EP.plotPoints("Inner resistance of Voltmeter.", "InnerRV.png", "t [s]", "V [V]")
```

Oppgave2.1.py

```
import sys, os
sys.path.append(os.path.abspath('../../Hall-effekt/src/'))
from PRELABOppgave1 import EvaluatePoints
from numpy import array
from matplotlib.pylab import plot, show, xlabel, ylabel, title, savefig

R = array([500, 700, 1000, 1200, 1500]) # Ohm
I = array([17.28, 12.45, 8.77, 7.33, 5.88])*10**(-3) # A
V = array([200.61, 144.56, 101.84, 85.07, 68.224])*10**(-3) # V

with open('Oppgave2.1table.txt', 'w') as f:
    for i in range(R.size):
        f.write("%g & %g & %g \\\\ \n" % (R[i], I[i], V[i]))
```

```
with open('Oppgave2.1datapoints.txt', 'w') as f:
    for i in range(R.size):
        f.write("%g \t %g \n" % (I[i], V[i]))

EP = EvaluatePoints('Oppgave2.1datapoints.txt')
EP.storeValues()
EP.interpolate(2, 0)
EP.plotPoints("Inner resistance of an Amperemeter", "InnerRA.png", "I [A]", "V [V]")

dI = I[-1] - I[0]
dV = V[-1] - V[0]
R = dV/float(dI)

with open('InnerRA.txt', 'w') as f:
    f.write("Inner resistance of amperemeter: %g ohm" % R)
```

Oppgave3.2.py

```
import os, sys
sys.path.append(os.path.abspath('../../Hall-effekt/src/'))
from PRELABOppgave1 import EvaluatePoints
from numpy import array
from matplotlib.pylab import plot, show, xlabel, ylabel, legend, title

R = array([1, 1.5, 2.5, 4, 10]) # Ohm
V = array([119.0, 137.6, 162.0, 181.4, 207.2])*10**(-3) # V

with open('Oppgave3.2table.txt', 'w') as f:
    for i in range(R.size):
        f.write("%g & %g \\\\ \n" % (R[i], V[i]))

I_formula = lambda v, r: v/float(r)

I = []

for i in range(R.size):
    I.append(I_formula(V[i], R[i]))

I = array(I)

dI = I[-1] - I[0]
dV = V[-1] - V[0]
R_i = -dV/float(dI)
```

```
with open('datapoints.txt', 'w') as f:
    for i in range(R.size):
        f.write("%g \t %g\n" % (I[i], V[i]))

EP = EvaluatePoints('datapoints.txt')
EP.storeValues()
EP.interpolate(2, 0)
EP.plotPoints("Inner resistance of Peltier-element", "InnerRP.png", "I [A]", "V [V]")

ems = V[3] + R_i * I[3]
with open('InnerRP.txt', 'w') as f:
    f.write("EMS of Peltier-element: %g V\\\\" % ems)
    f.write("Inner resistance of Peltier-element: %g ohm" % R_i)
```

Oppgave4.py

```
from numpy import array

I = array([1.0, 1.25, 1.5, 1.75, 1.99]) # A
I_A4 = array([1.012, 1.2614, 1.511, 1.762, 1.9975]) # A
I_A2 = array([1.006, 1.257, 1.507, 1.755, 2.001]) # A
V_A4 = array([0.058, 0.071, 0.085, 0.099, 0.112])*10**(-3) # V
V_A2 = array([3.926, 4.85, 5.78, 6.71, 7.642])*10**(-3) # V
I_C4 = array([1.008, 1.255, 1.5125, 1.76, 2.002]) # A
V_C4 = array([0.033, 0.04, 0.048, 0.055, 0.063])*10**(-3) # V
I_C2 = array([1.007, 1.253, 1.511, 1.761, 2.004]) # A
V_C2 = array([2.1, 2.5, 2.97, 3.41, 3.84])*10**(-3) # V

R_A4 = []
R_A2 = []
R_C4 = []
R_C2 = []

for i in range(I_A4.size):
    R_A4.append(V_A4[i]/float(I_A4[i]))
    R_A2.append(V_A2[i]/float(I_A2[i]))
    R_C4.append(V_C4[i]/float(I_C4[i]))
    R_C2.append(V_C2[i]/float(I_C2[i]))


with open('Oppgave4A.txt', 'w') as f:
    for i in range(I_A4.size):
        f.write("%g & %g & %g & %g & %g & %g\\\\ \n" % (R_A4[i], I_A4[i], V_A4[i],\
```

```
                                                R_A2[i], I_A2[i], V_A2[i]))

with open('Oppgave4C.txt', 'w') as f:
    for i in range(I_A4.size):
        f.write("%g & %g & %g & %g & %g & %g\\\\ \n" % (R_C4[i], I_C4[i], V_C4[i],\
                                                R_C2[i], I_C2[i], V_C2[i]))
```

Oppgave5.py

```
from numpy import array, sin, pi, sum as npsum

NA = 11.0 # m^2
t_list = array([4.37, 2.09, 1.36, 1.01, 7.41, 9.65, 0.66]) # s
ems_0 = array([0.6, 1.1, 1.5, 2.8, 0.4, 0.2, 3.0])*10**(-3) # V

with open('Oppgave5table.txt', 'w') as f:
    for i in range(t_list.size):
        f.write("%g & %g \\\\ \n" % (t_list[i], ems_0[i]))

B = lambda ems, t: t * ems/float(4 * NA)

B_list = []

for i in range(t_list.size):
    B_list.append(B(ems_0[i], t_list[i]))

B_list = array(B_list)
B_mean = npsum(B_list)/float(B_list.size)

with open('MeanValue.txt', 'w') as f:
    f.write("Mean value of B: %g T" % B_mean)
```