The temperature dependence of resistance - Experiment 24

In[1]:=
```
<< CurveFit`
```
```
CurveFit for Mathematica v7.x thru v11.x, Version 1.96, 4/4/2018
Caltech Sophomore Physics Labs, Pasadena, CA
```

Estimating uncertainties due to noise

We estimate the uncertainties due to noise (random errors) in our measurements using the (standard deviation) in our room temperature measurement data.

Uncertainty in temperature

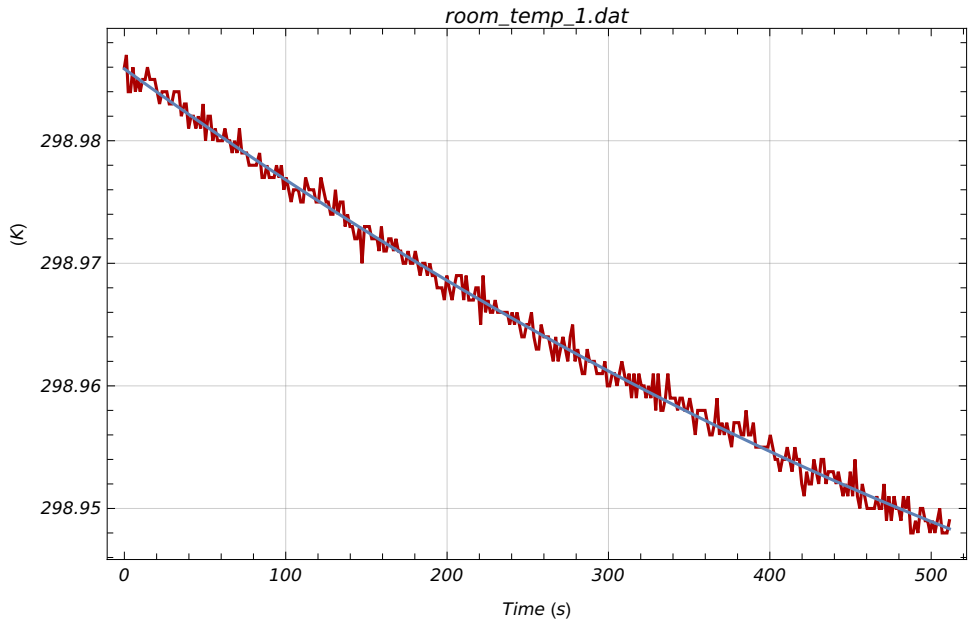
In[4]:=
```
With[ {name = SystemDialogInput["FileOpen",
     {DataFileName, {"data files" -> {"*.dat", "*.mca"}, "all files" -> {"*"}}}]},
 If[ name =!= $Canceled,
  LoadFile[name]]
]
```

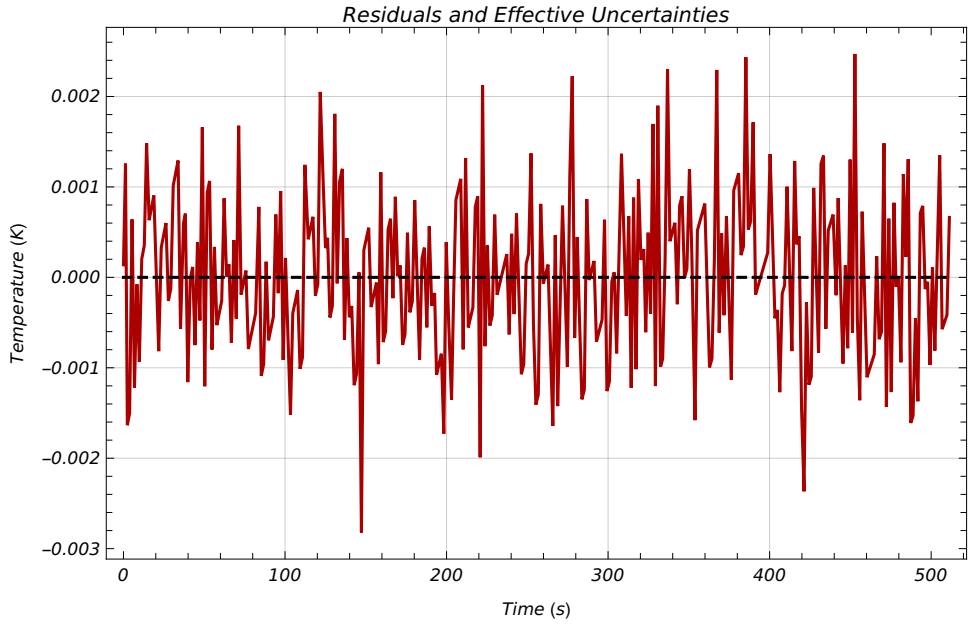
We observe that our room temperature measurement of temperature shows a decreasing trend. This is most likely due to a remaining temperature gradient between the lab temperature and the dielectric fluid temperature from the previous experiment carried out using the setup. To estimate the noise level in temperature measurements, we perform a quadratic fit on our data (it qualitatively appears to fit the decrease in temperature relatively well over our range of measurements) and use the standard deviation of the fit as an estimate of noise in our temperature measurements.

In[10]:=
```
QuadraticFit[]
```

In[12]:=
```
LinearDifferencePlot[FrameLabel → {"Time (s)", "Temperature (K)"}]
```

Out[12]=





$$y(x) = a + b\,x + c\,x^2$$

a=	b=	c=	
298.986	-0.000094645	4.15736×10^{-8}	
σ_a=	σ_b=	σ_c=	Std. deviation=
0.000141381	1.2788×10^{-6}	2.42319×10^{-9}	0.000880185

In[19]:=
```
errTemp = 0.0008801846325121835`;
```

This allows us to estimate the error in temperature measurements as $\sigma_T = 0.00088\,K$. We use a similar process to estimate the uncertainties in resistance as a function of temperature for the copper (good resistor), thermistor (pure semiconductor), and doped semiconductor data.

Uncertainty in resistance - semiconductor

In[30]:=
```
With[ {name = SystemDialogInput["FileOpen",
     {DataFileName, {"data files" -> {"*.dat", "*.mca"}, "all files" -> {"*"}}}]},
 If[ name =!= $Canceled,
  LoadFile[name]]
]
```

We use the standard deviation of the room temperature measurements of semiconductor resistance fit to a linear model (again, it appears to adequately fit the data) to estimate the error in resistance measurements for the semiconductor rod. Note that we use the data of resistance as a function of time, not of temperature, to avoid correlation issues between the error in temperature determined above and the error in resistance as a function of temperature.

In[26]:=
```
LinearFit[]
```

n = 341

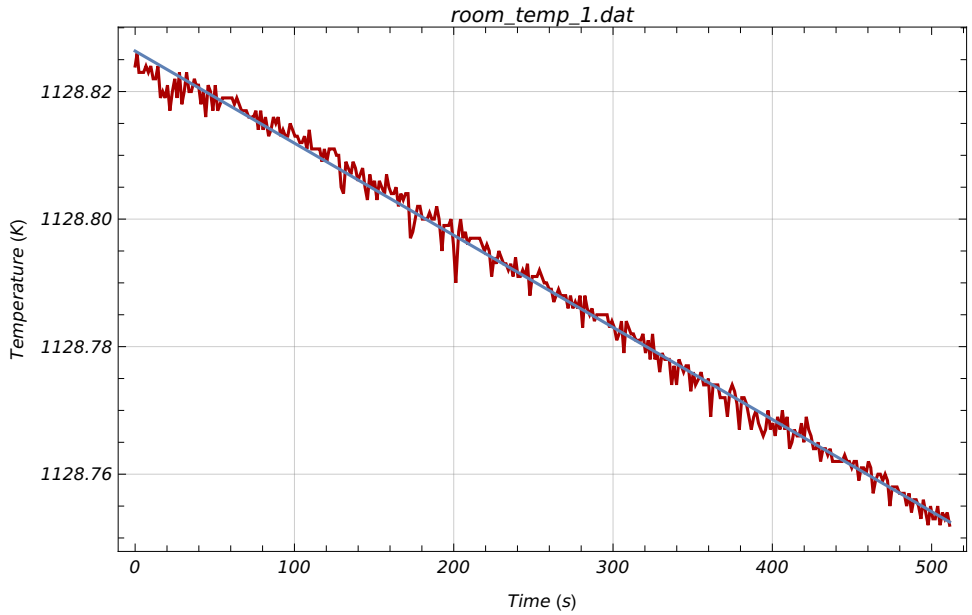
y(x) = a + b x

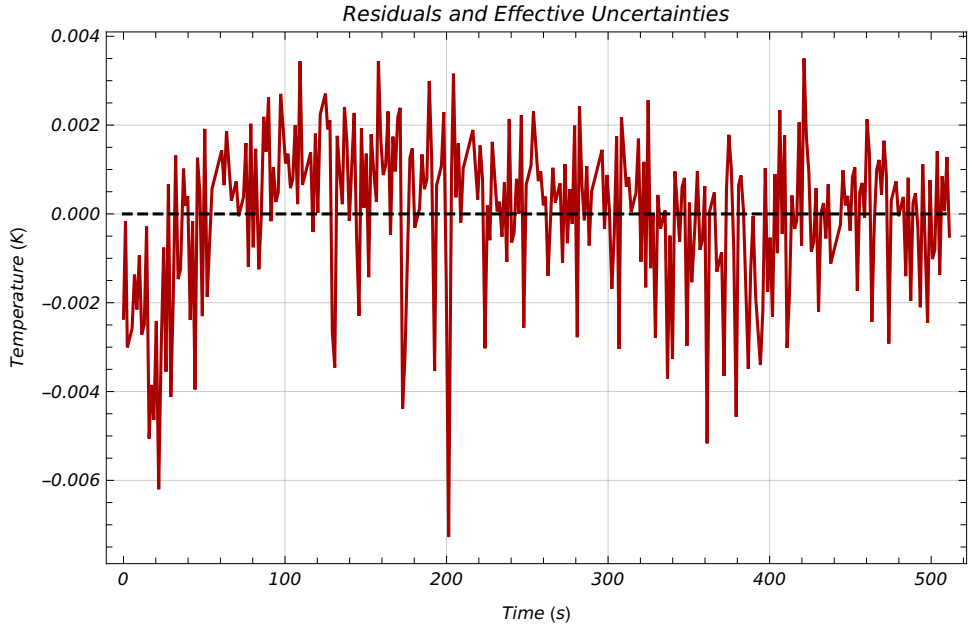
Fit of (x,y) (unweighted)

a=	b=	
1128.83	−0.000144437	
σ_a=	σ_b=	Std. deviation=
0.000184814	6.25598×10^{-7}	0.00171335

In[27]:=
```
LinearDifferencePlot[FrameLabel → {"Time (s)", "Temperature (K)"}]
```

Out[27]=





$y(x) = a + b\,x$

$a =$	$b =$	
1128.83	-0.000144437	
$\sigma_a =$	$\sigma_b =$	Std. deviation=
0.000184814	6.25598×10^{-7}	0.00171335

In[31]:=
```
errSemi = 0.0017133548093589553`;
```

This allows us to estimate the error in (doped) semiconductor resistance measurements as $\sigma_{R_{\text{semiconductor}}} = 0.0017\,\Omega$.

Uncertainty in resistance - copper resistor

In[32]:=
```
With[ {name = SystemDialogInput["FileOpen",
     {DataFileName, {"data files" -> {"*.dat", "*.mca"}, "all files" -> {"*"}}}]},
 If[ name =!= $Canceled,
  LoadFile[name]]
]
```

We use the standard deviation of the room temperature measurements of copper resistance fit to a linear model (again, it appears to adequately fit the data) to estimate the error in resistance measurements for copper. Note that we use the data of resistance as a function of time, not of temperature, to avoid correlation issues between the error in temperature determined above and the error in resistance as a function of temperature.

In[33]:=
```
LinearFit[]
```

n = 341

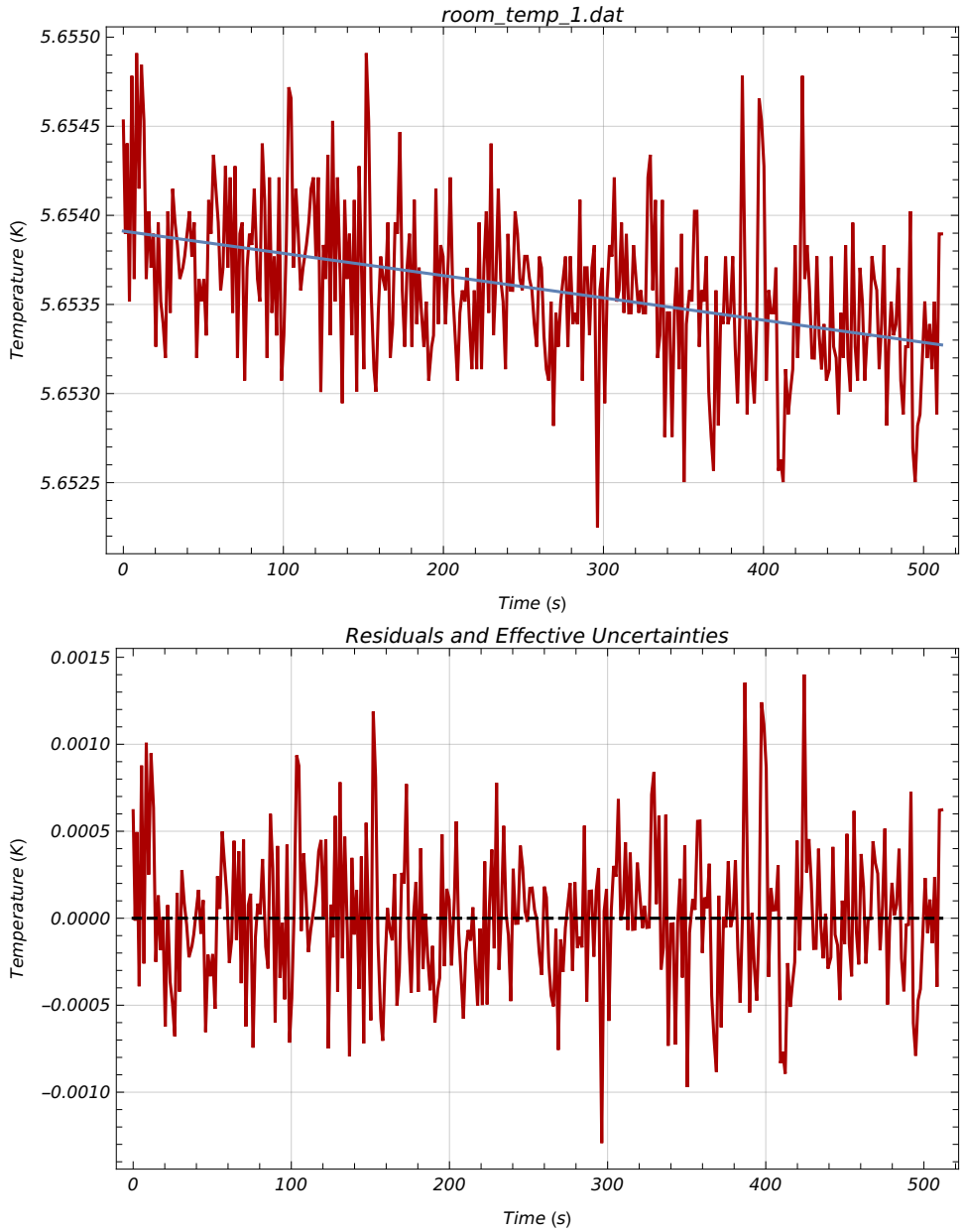
y(x) = a + b x

Fit of (x,y) (unweighted)

a=	b=	
5.65391	-1.24902×10^{-6}	
σ_a=	σ_b=	Std. deviation=
0.0000448065	1.5167×10^{-7}	0.000415383

In[34]:=
```
LinearDifferencePlot[FrameLabel → {"Time (s)", "Temperature (K)"}]
```

Out[34]=



$y(x) = a + b\,x$

a=	b=		
5.65391	-1.24902×10^{-6}		
σ_a=	σ_b=	Std. deviation=	
0.0000448065	1.5167×10^{-7}	0.000415383	

In[35]:=
```
errCopper = 0.00041538344072802603`;
```

This allows us to estimate the error in copper resistance measurements as $\sigma_{R_{\text{copper}}} = 0.00042\,\Omega$.

Uncertainty in resistance - thermistor

In[36]:=
```
With[ {name = SystemDialogInput["FileOpen",
     {DataFileName, {"data files" -> {"*.dat", "*.mca"}, "all files" -> {"*"}}}]},
 If[ name =!= $Canceled,
  LoadFile[name]]
]
```

We use the standard deviation of the room temperature measurements of copper resistance fit to a quadratic model (again, it appears to adequately fit the data) to estimate the error in resistance measurements for copper. Note that we use the data of resistance as a function of time, not of temperature, to avoid correlation issues between the error in temperature determined above and the error in resistance as a function of temperature.

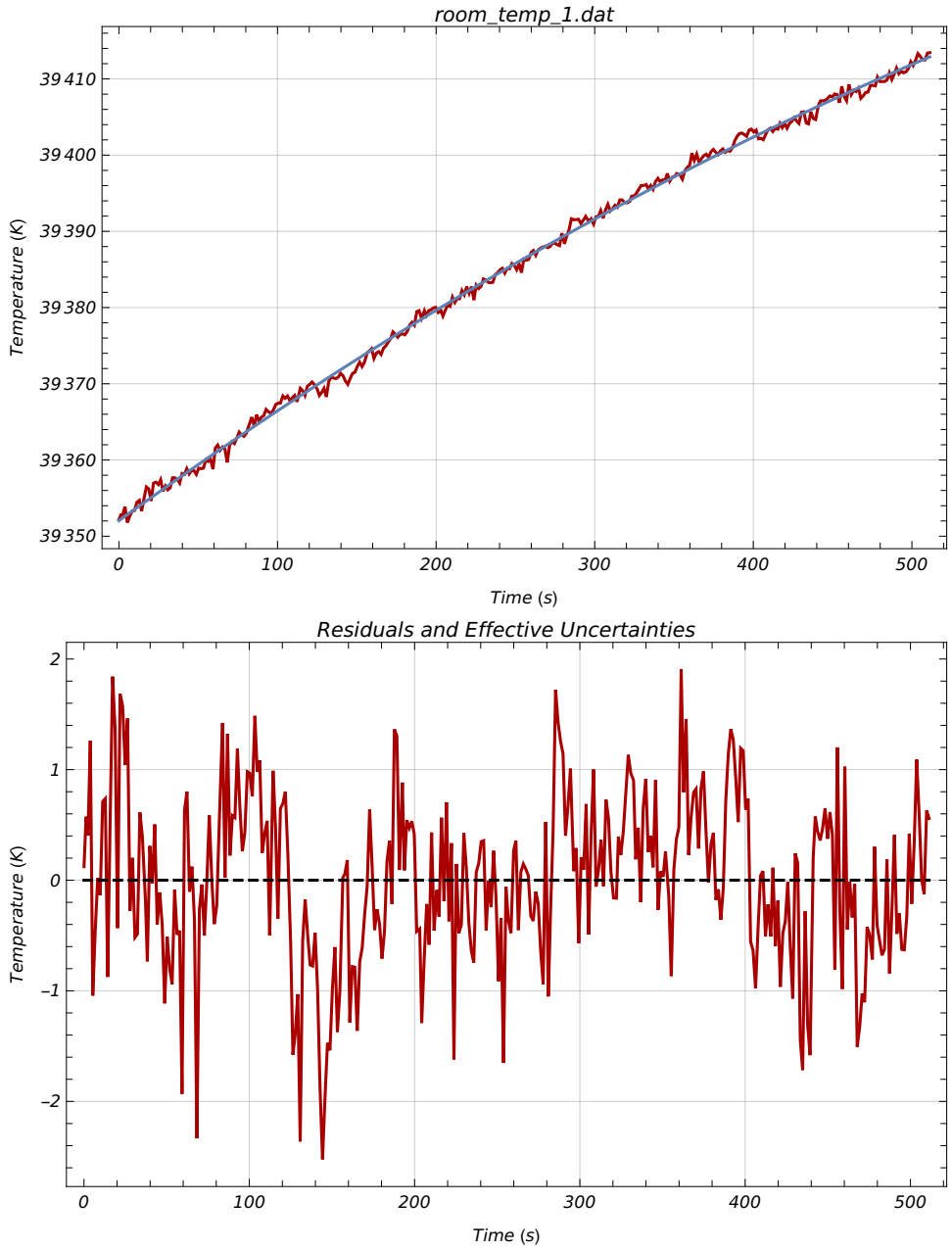
In[39]:=
```
QuadraticFit[]
```

```
n = 341

y(x) = a + b x + c x^2

Fit of (x,y)  (unweighted)
```

a=	b=	c=	
39 352.	0.150216	−0.0000610126	
σ_a=	σ_b=	σ_c=	Std. deviation=
0.122844	0.00111117	2.10555×10^{-6}	0.764792

In[40]:=
```
LinearDifferencePlot[FrameLabel → {"Time (s)", "Temperature (K)"}]
```

Out[40]=



$y(x) = a + b\,x + c\,x^2$

a= 39352.	b= 0.150216	c= −0.0000610126
σ_a= 0.122844	σ_b= 0.00111117	σ_c= 2.10555×10^{-6}

Std. deviation= 0.764792

In[41]:=
```
errThermistor = 0.7647923702705844`;
```

This allows us to estimate the error in copper resistance measurements as $\sigma_{R_{\text{Thermistor}}} = 0.76\,\Omega$.

Resistance of copper with respect to temperature

In[231]:=
```
With[ {name = SystemDialogInput["FileOpen",
     {DataFileName, {"data files" -> {"*.dat", "*.mca"}, "all files" -> {"*"}}}]},
 If[ name =!= $Canceled,
  LoadFile[name]]
]
```

In[234]:=
```
AssignYsigmas[ errCopper](* insert value between brackets *)
```

In[235]:=
```
SwitchXXandYY[]
```

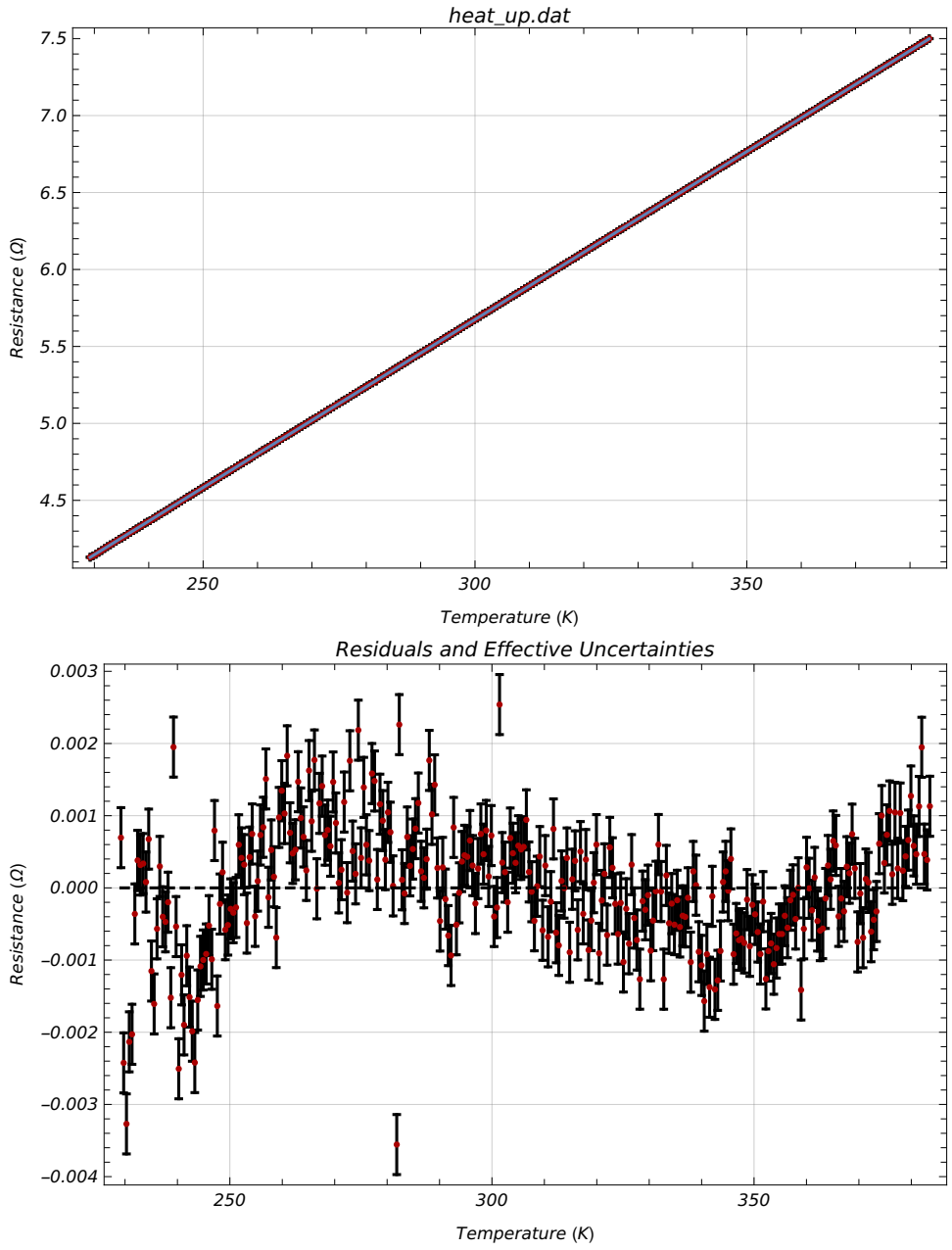
In[236]:=
```
AssignYsigmas[ errTemp](* insert value between brackets *)
```

In[237]:=
```
SwitchXXandYY[]
```

In[50]:=
```
LinearFit[]
```

In[51]:=
```
LinearDifferencePlot[FrameLabel → {"Temperature (K)", "Resistance (Ω)"}]
```

Out[51]=



$y(x) = a + b\ x$

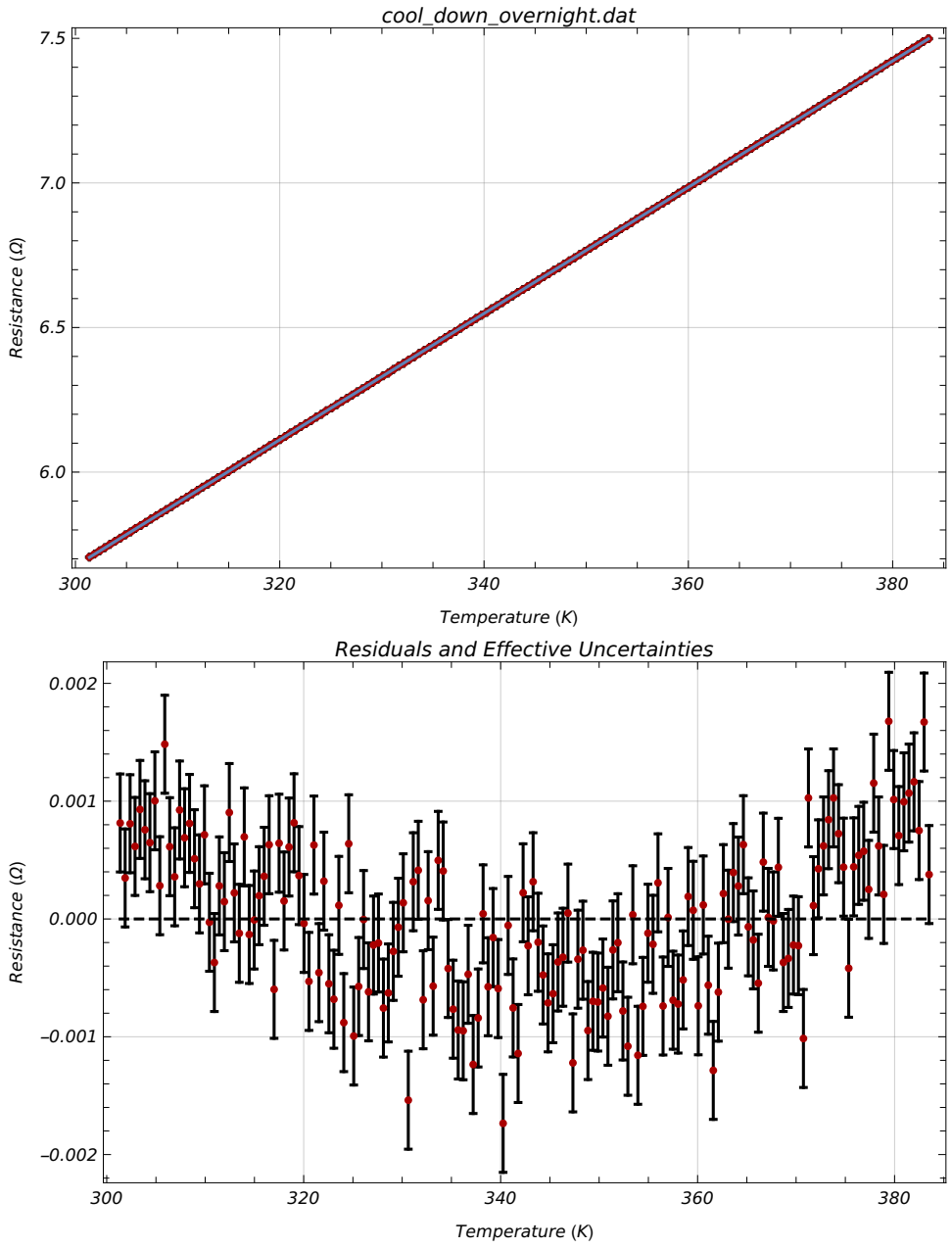
$a =$	$b =$	
-0.874088	0.0218298	
$\sigma_a =$	$\sigma_b =$	$\chi^2/(n-2) =$
0.000166589	5.37538×10^{-7}	4.50818

We observe a relatively good fit of the data to a linear model $\left(\tilde{\chi}^2 = 4.51\right)$, as expected from the theory presented in the lab manual. We perform the same fit over the overnight cool-down dataset (we will later use this to estimate the systematic uncertainties due to temperature gradients).

```
In[248]:= With[ {name = SystemDialogInput["FileOpen",
            {DataFileName, {"data files" -> {"*.dat", "*.mca"}, "all files" -> {"*"}}}]},
          If[ name =!= $Canceled,
           LoadFile[name]]
         ]

In[249]:= AssignYsigmas[ errCopper](* insert value between brackets *)

In[250]:= SwitchXXandYY[]

In[251]:= AssignYsigmas[ errTemp](* insert value between brackets *)

In[252]:= SwitchXXandYY[]

In[253]:= LinearFit[]
```

In[254]:=
```
LinearDifferencePlot[FrameLabel → {"Temperature (K)", "Resistance (Ω)"}]
```

Out[254]=



$y(x) = a + b\,x$

$a =$	$b =$	
-0.874498	0.0218324	
$\sigma_a =$	$\sigma_b =$	$\chi^2/(n-2) =$
0.000468538	1.36536×10^{-6}	2.54075

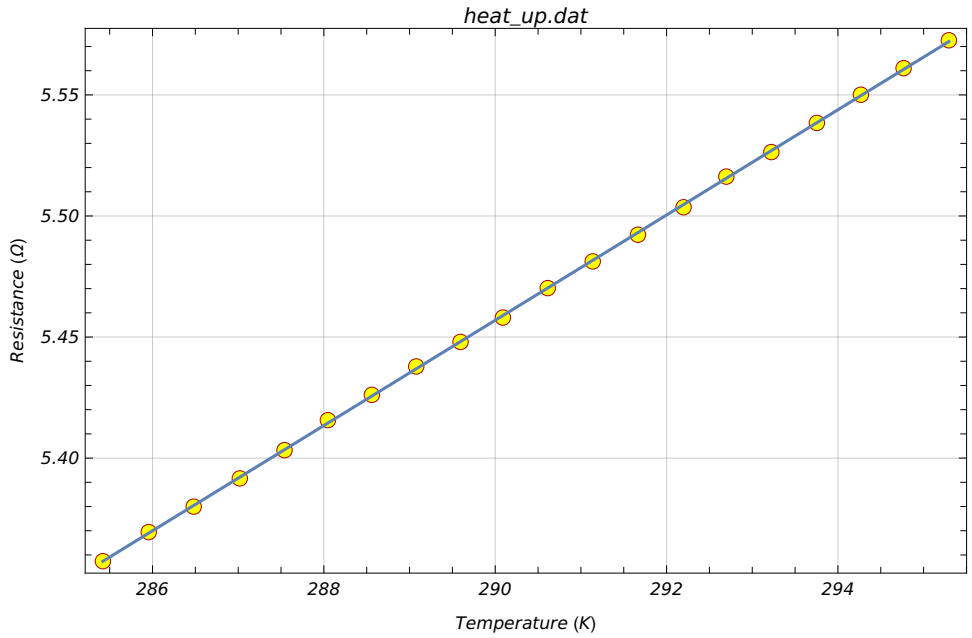
We now restrict our fit around 0°C and 20°C and use the fits to calculate the temperature coefficient of resistance, α.

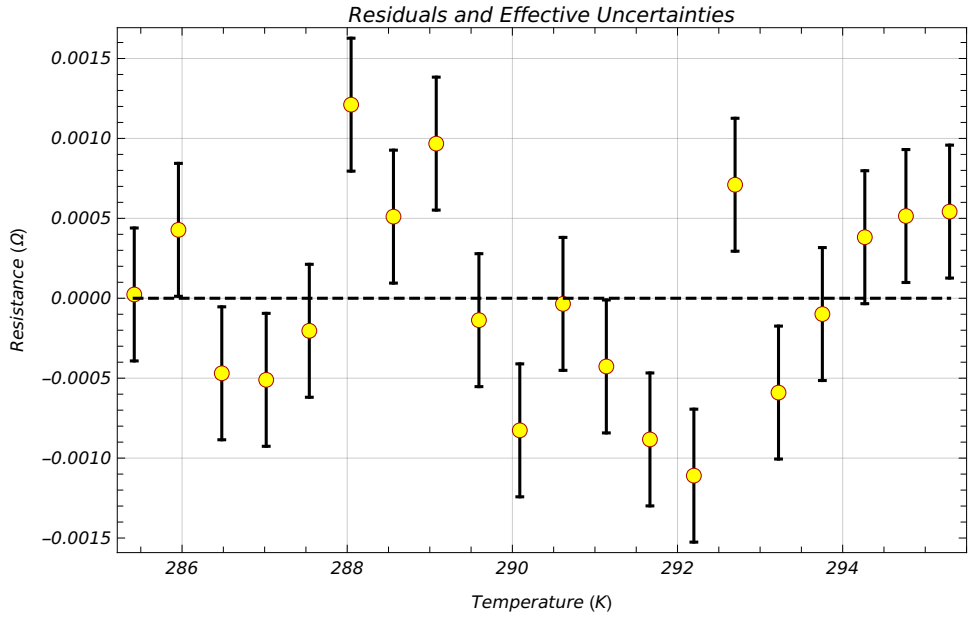
Estimating $\alpha_{20\,^\circ C}$

In[53]:=
```
With[{x = SetXRange[ LinearDataPlot[], Log -> False,
     Label -> "Set the X values for the range you wish to keep." ]},
 Print[x];

 XRangeKeep[Sequence @@ x]
]
```

In[54]:=
```
LinearFit[]
```

In[55]:=
```
LinearDifferencePlot[FrameLabel → {"Temperature (K)", "Resistance (Ω)"}]
```





$y(x) = a + b\,x$

$a =$	$b =$	
−0.847052	0.0217379	
$\sigma_a =$	$\sigma_b =$	$\chi^2/(n-2) =$
0.00903143	0.0000311015	2.51647

We observe an even better fit of the data over this range than over the whole dataset.

In[543]:=
```
ab20 = {-0.8470524979828181`, 0.021737859891715228`};
```

In[56]:=
```
tempCoeff[t0_] := b/(a + b (t0)) {1, Sqrt[(sigb/b)^2 + ((siga^2 + (t0 sigb)^2))/((a + b (t0))^2)]}
```

In[58]:=
```
tempCoeff[293.15]
```

Out[58]=
```
{0.00393417, 0.0000107321}
```

In[541]:=
```
alpha20 = %58;
```

From this, we can estimate $\alpha_{20\,°C} = 0.003934 \pm 0.000011\,°C^{-1}$, which is in fact within error of the published value of $0.00393\ °C^{-1}$ (National Bureau of Standards Circular No. 31 (1914 Edition)).

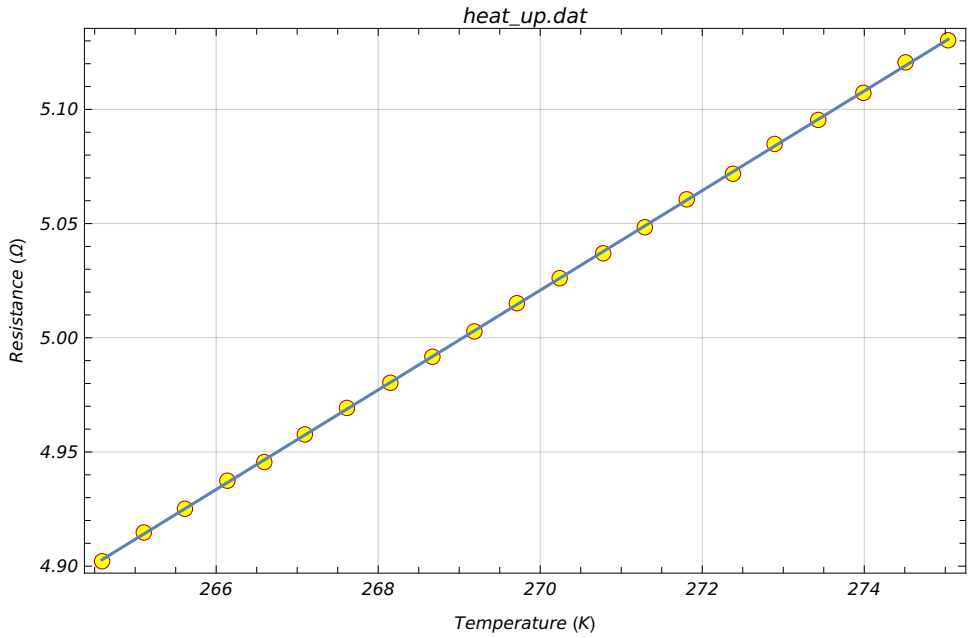
Estimating $\alpha_{0\,°C}$

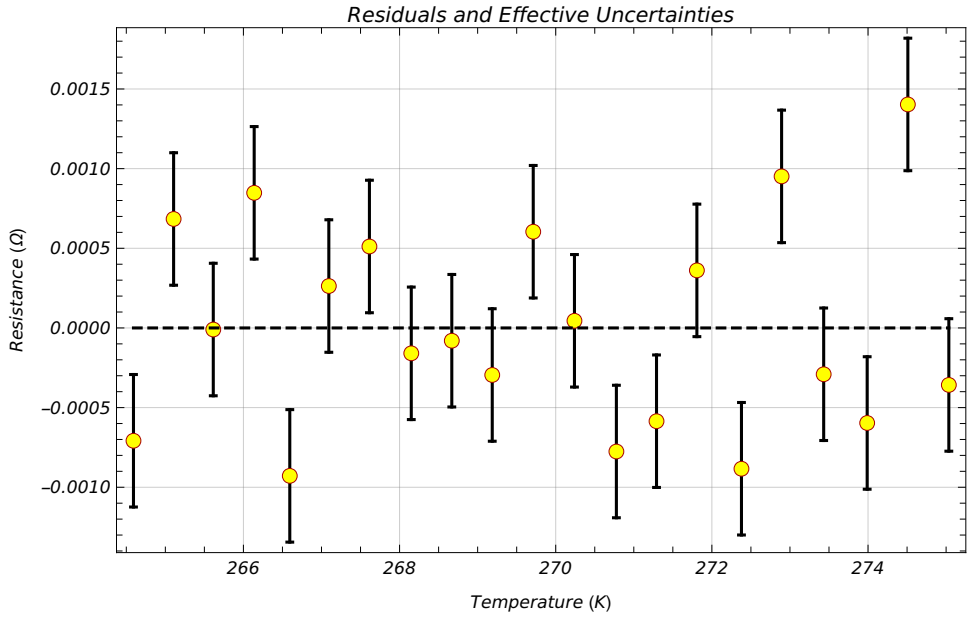
In[232]:=
```
With[{x = SetXRange[ LinearDataPlot[], Log -> False,
     Label -> "Set the X values for the range you wish to keep."]},
 Print[x];

 XRangeKeep[Sequence @@ x]
]
```

In[238]:=
```
LinearFit[]
```

In[239]:=
```
LinearDifferencePlot[FrameLabel → {"Temperature (K)", "Resistance (Ω)"}]
```

Out[239]=





y(x) = a + b x

a =	b =	
−0.868654	0.0218128	
σ_a =	σ_b =	$\chi^2/(n-2)$ =
0.00772821	0.0000286472	2.63762

In[544]:=
```
ab0 = {-0.8686541861092515`, 0.021812845360786835`};
```

Again, we observe an even better fit of the data over this range than over the whole dataset.

In[240]:=
```
tempCoeff[273.15]
```

Out[240]=
```
{0.00428583, 0.0000108376}
```

In[542]:=
```
alpha0 = %240;
```

From this, we can estimate $\alpha_{20\,^\circ C} = 0.004286 \pm 0.000011\,^\circ C^{-1}$, which is in fact within error of the published value of $0.00428\ ^\circ C^{-1}$ (National Bureau of Standards Circular No. 31 (1914 Edition)).

Resistance of thermistor (pure semiconductor) with respect to temperature

In[163]:=
```
With[ {name = SystemDialogInput["FileOpen",
    {DataFileName, {"data files" -> {"*.dat", "*.mca"}, "all files" -> {"*"}}}]},
 If[ name =!= $Canceled,
  LoadFile[name]]
]
```

We observe an offset in the first few points in the heat-up data, most likely caused by a hardware issue. We therefore disregard those points in our analysis.

In[164]:=
```
With[{x = SetXRange[ LinearDataPlot[], Log -> False,
    Label -> "Set the X values for the range you wish to keep." ]},
 Print[x];

 XRangeKeep[Sequence @@ x]
]
```

In[165]:=
```
AssignYsigmas[ errThermistor](* insert value between brackets *)
```

In[166]:=
```
SwitchXXandYY[]
```

In[167]:=
```
AssignYsigmas[ errTemp](* insert value between brackets *)
```

In[168]:=
```
SwitchXXandYY[]
```

From the theory presented in the lab manual and our work from the pre-lab, we expect an exponential variation of resistance with inverse of temperature.

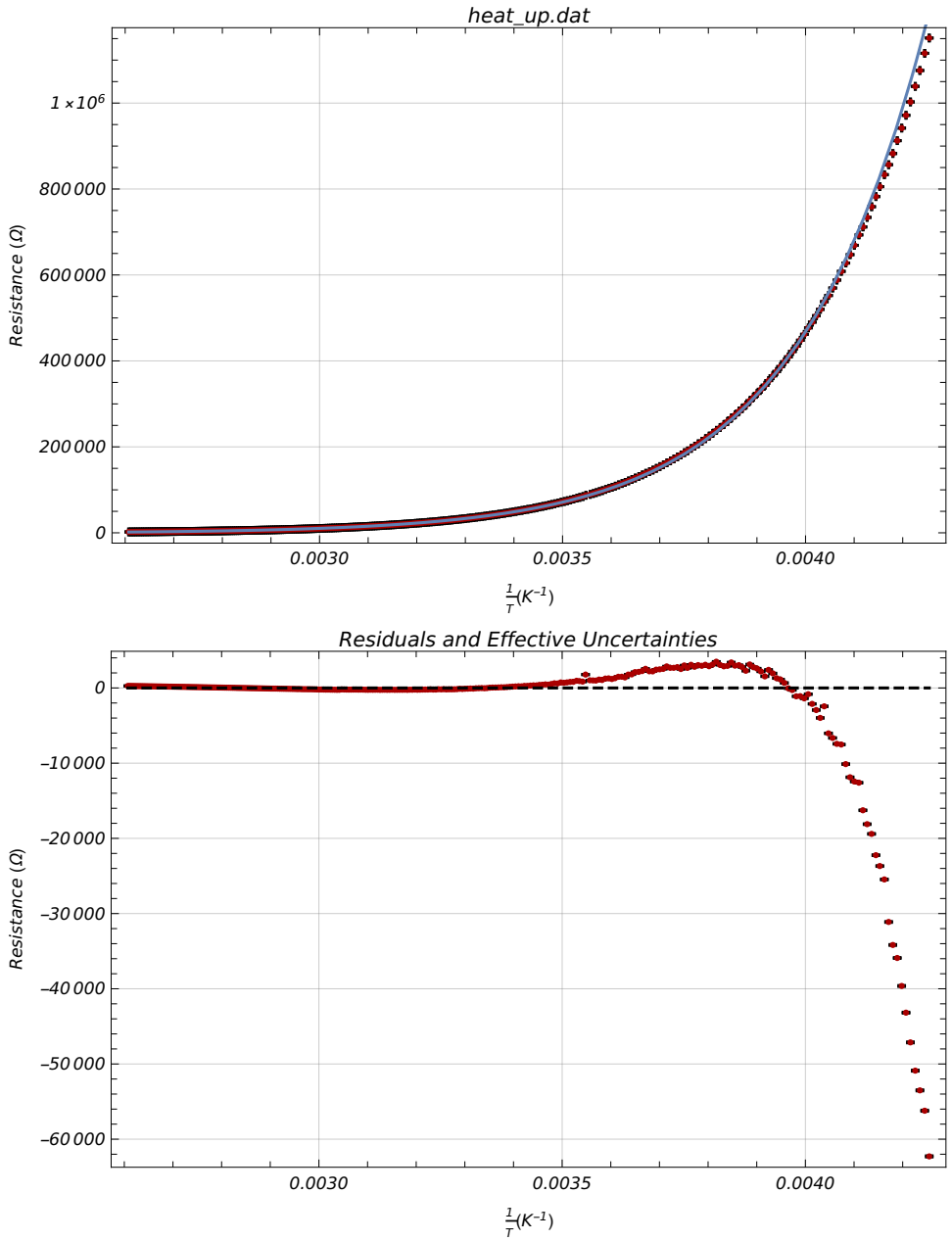
In[133]:=
```
kb = 1.3806 * 10^-23;
```

In[169]:=
```
xnew[ x_, y_ ] := 1.0 / (x)
ynew[ x_, y_ ] := y
DataTransform[]
(* Use Undo[] if you don't like the results. *)
```

In[172]:=
```
SemilogCFit[]
```

In[173]:=
```
LinearDifferencePlot[FrameLabel → {"1/T (K^-1)", "Resistance (Ω)"}]
```

Out[173]=



$y(x) = a\ \mathrm{Exp}[b\ x] + c$

a=	b=	c=	
0.146532	3744.27	−752.948	
σ_a=	σ_b=	σ_c=	$\chi^2/(n-3)$=
0.0000107457	0.0114185	0.150532	68 004.1

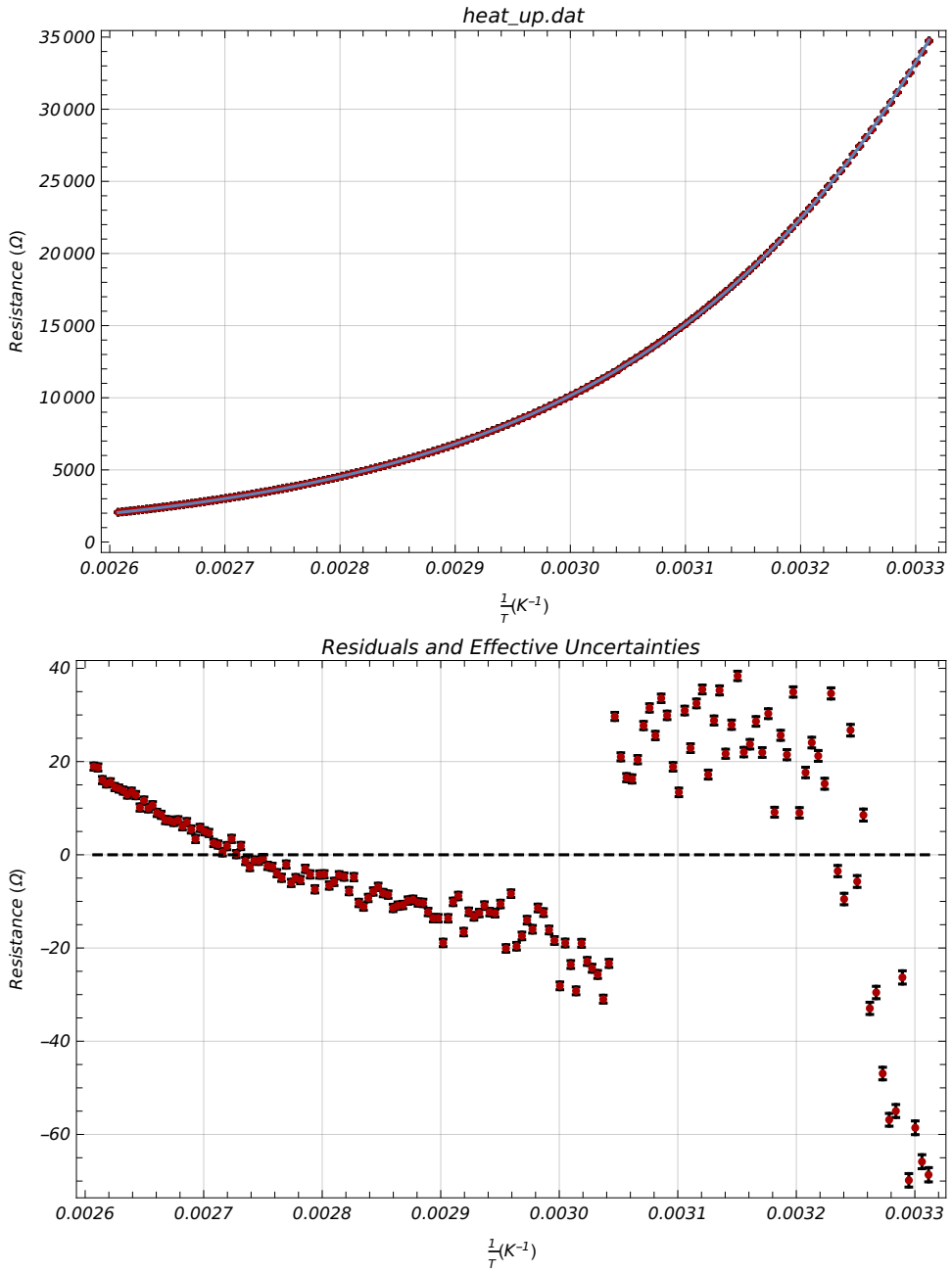
We observe a very bad fit. This might indicate that the semiconductor is not as pure as we would expect. We remove the lower temperature values of our dataset and attempt the fit again.

In[174]:=
```
With[{x = SetXRange[ LinearDataPlot[], Log -> False,
    Label -> "Set the X values for the range you wish to keep." ]},
 Print[x];

 XRangeKeep[Sequence @@ x]
]
```

In[190]:=
```
SemilogCFit[]
```

In[191]:=
```
LinearDifferencePlot[FrameLabel → {"1/T (K^-1)", "Resistance (Ω)"}]
```

Out[191]=



$y(x) = a\ \mathrm{Exp}[b\ x] + c$

a =	b =	c =	
0.080729	3919.25	−169.691	
σ_a =	σ_b =	σ_c =	$\chi^2/(n-3)$ =
0.0000167373	0.518848	0.30351	421.105

This is a much better fit, as expected, but it is still relatively bad $\left(\tilde{\chi}^2 = 421\right)$ and the residuals still have a distinctive feature indicative either of an error in our model or of a measurement error. Since we need to reduce the range of temperatures to the high-temperature range to obtain a decent fit anyway, we may as well use the overnight cool-down dataset to minimize systematic errors due to temperature

gradients.

In[192]:=
```
With[ {name = SystemDialogInput["FileOpen",
     {DataFileName, {"data files" -> {"*.dat", "*.mca"}, "all files" -> {"*"}}}]},
 If[ name =!= $Canceled,
  LoadFile[name]]
]
```

As before, we observe the weird offset in the dataset at low temperatures, indicative of a hardware issue. We remove the offset portion of the dataset.

In[226]:=
```
With[{x = SetXRange[ LinearDataPlot[], Log -> False,
     Label -> "Set the X values for the range you wish to keep." ]},
 Print[x];

 XRangeKeep[Sequence @@ x]
]
```

In[210]:=
```
AssignYsigmas[ errThermistor](* insert value between brackets *)
```

In[211]:=
```
SwitchXXandYY[]
```

In[212]:=
```
AssignYsigmas[ errTemp](* insert value between brackets *)
```

In[213]:=
```
SwitchXXandYY[]
```

From the theory presented in the lab manual and our work from the pre-lab, we expect an exponential variation of resistance with inverse of temperature.

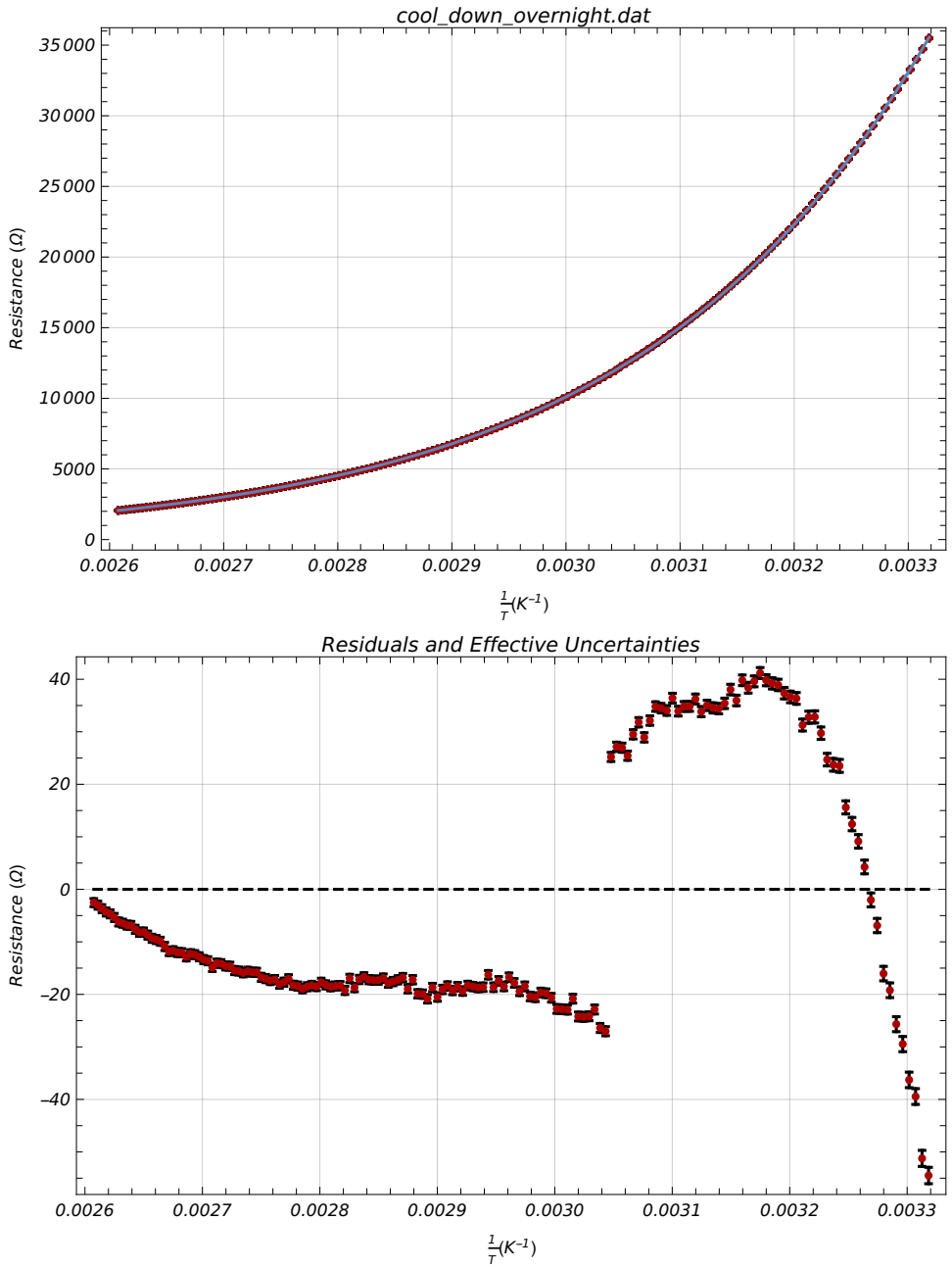
In[•]:=
```
kb = 1.3806 * 10^-23;
```

In[214]:=
```
xnew[ x_, y_ ] := 1.0 / (x)
ynew[ x_, y_ ] := y
DataTransform[]
(* Use Undo[] if you don't like the results. *)
```

In[229]:=
```
SemilogCFit[]
```

In[230]:=
```
LinearDifferencePlot[FrameLabel → {"1/T (K^-1)", "Resistance (Ω)"}]
```

Out[230]=



$y(x) = a\ \mathrm{Exp}[b\ x] + c$

$a =$	$b =$	$c =$	
0.0785559	3925.67	−129.672	
$\sigma_a =$	$\sigma_b =$	$\sigma_c =$	$\chi^2/(n-3) =$
0.00726604	16.1235	86.4626	630.075

We obtain a similarly bad fit, indicating that systematic errors due to temperature gradients are not the issue. By inspection of the structure of the residuals above, the error cannot correspond to a linear correction term, and therefore it would be difficult to correct for this issue by modifying the fitting function without knowing more about the nature of the discrepancy.

The structure of the residuals shown above, and the persistence of that structure over both datasets, however, indicates that the error may be due to the fact (c.f. Experiment 13) that E_g is non-constant with respect to temperature, contrary to what we assume in our model.

We simply use our best fit on the overnight cool-down data (to minimize temperature gradient systematic errors) to obtain an estimate of the gap energy in the thermistor's semiconductor material from the fit parameters using $R \propto e^{E_g/(2\,k_b\,T)}$.

In[242]:=
```
eg = 2 kb / (1.602 * 10^-19) {3925.6724376520274`, 16.123484554446605`} (* in eV *)
```

Out[242]=
```
{0.676627, 0.00277904}
```

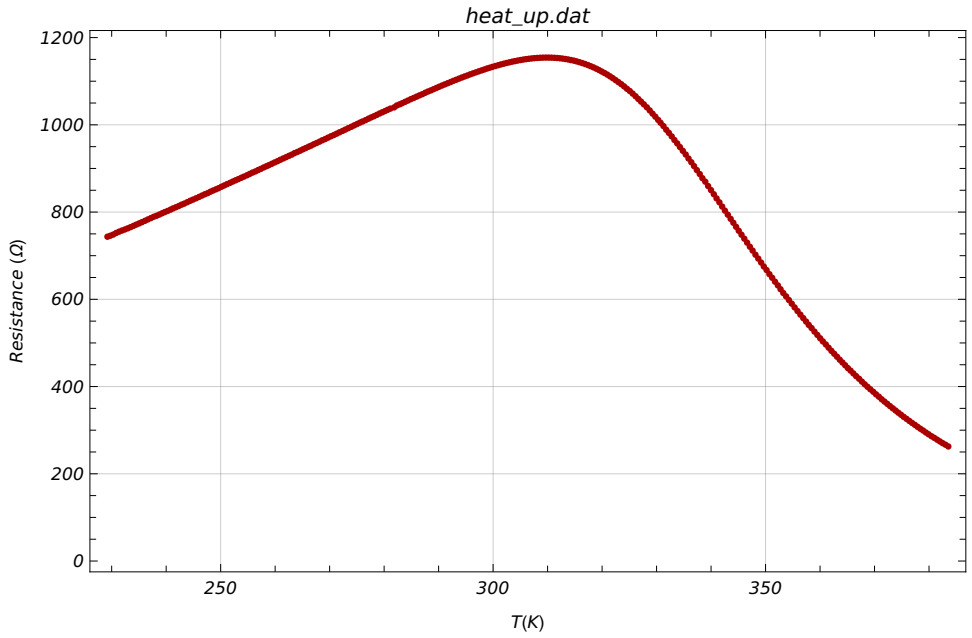
We can therefore estimate $E_g = 0.6766 \pm 0.0028$ eV, which is within a couple standard errors of the expected value of 0.67 eV (note that we have yet to include systematic uncertainties).

Resistance of semiconductor rod (doped semiconductor) with respect to temperature

In[357]:=
```
With[ {name = SystemDialogInput["FileOpen",
     {DataFileName, {"data files" -> {"*.dat", "*.mca"}, "all files" -> {"*"}}}]},
 If[ name =!= $Canceled,
  LoadFile[name]]
]
```

In[358]:=
```
LinearDataPlot[FrameLabel → {"T(K)", "Resistance (Ω)"}]
```

Out[358]=



In[359]:=
```
AssignYsigmas[ errSemi](* insert value between brackets *)
```

In[360]:=
```
SwitchXXandYY[]
```

In[361]:=
```
AssignYsigmas[ errTemp](* insert value between brackets *)
```

In[362]:=
```
SwitchXXandYY[]
```

Low-temperature behavior

We start by analyzing the asymptotic behavior at low temperatures, which should follow $R \propto \frac{T^{\frac{3}{2}}}{N_d}$. For all following analysis, we algebraically modify our dataset so as to linearize the model we need to fit our data to. This approach is used in order to obtain a weighted fit (including a goodness-of-fit characteristic), because the fitting routines provided seem to be unable to fit to restricted datasets, and the full datasets fit very poorly to our models.

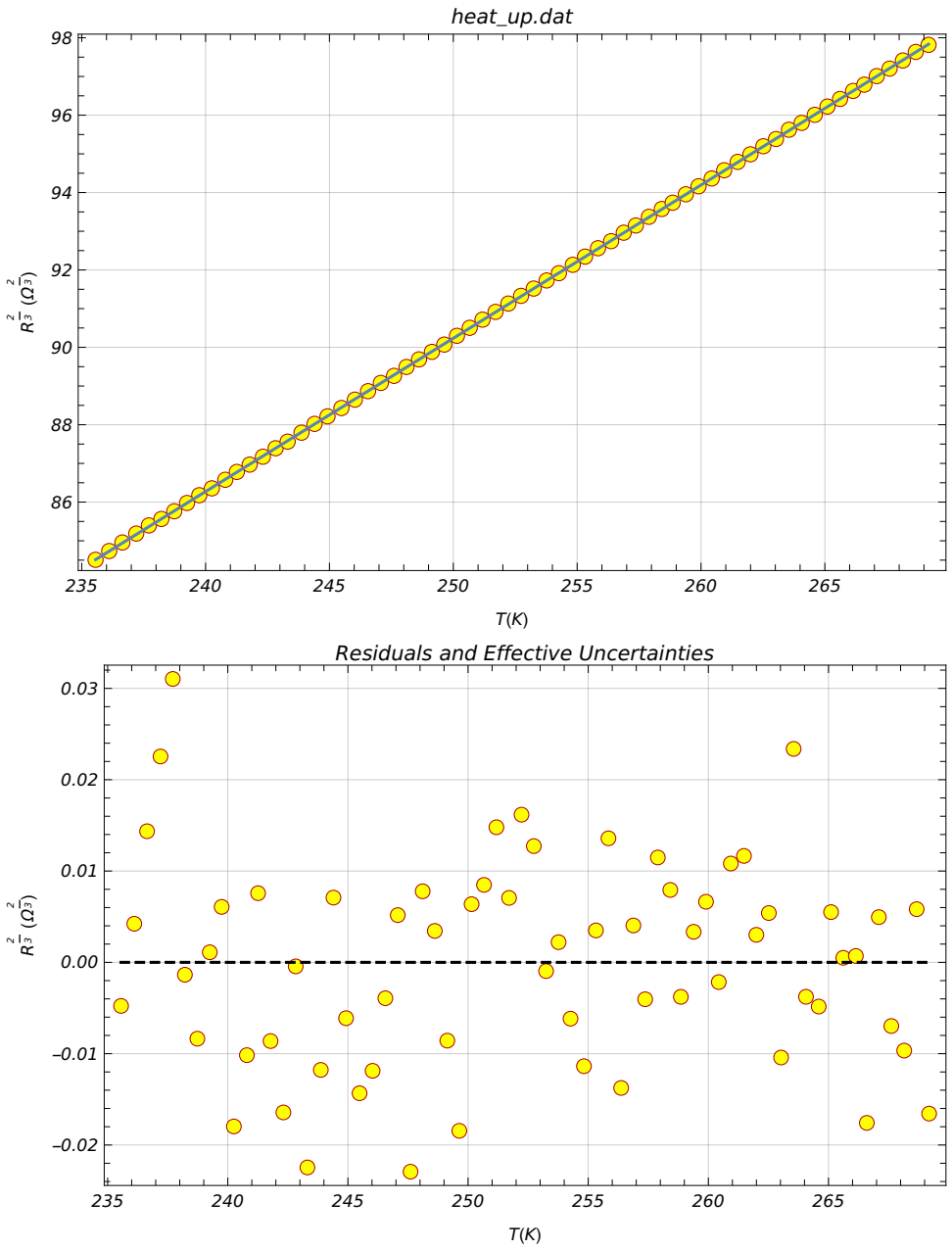
In[370]:=
```
With[{x = SetXRange[ LinearDataPlot[], Log -> False,
    Label -> "Set the X values for the range you wish to keep."]},
 Print[x];

 XRangeKeep[Sequence @@ x]
]
```

In[363]:=
```
xnew[ x_, y_ ] := x
ynew[ x_, y_ ] := y^(2/3)

DataTransform[]

(* Use Undo[] if you don't like the results. *)
```

In[371]:=
```
LinearFit[]
```

In[373]:=
```
LinearDifferencePlot[FrameLabel → {"T(K)", "R^(2/3) (Ω^(2/3))"}]
```

Out[373]=



$y(x) = a + b\,x$

a =	b =	
−8.79105	0.396099	
σ_a =	σ_b =	$\chi^2/(n-2)$ =
0.0021289	8.4237×10^{-6}	947.463

We obtain a bad fit even over a very restricted dataset, but the residuals have no apparent structure. This indicates the presence of a temperature-dependent source of error either in the temperature measurements or in the resistance measurements that the room-temperature measurements we have used to estimate noise have not allowed us to detect.

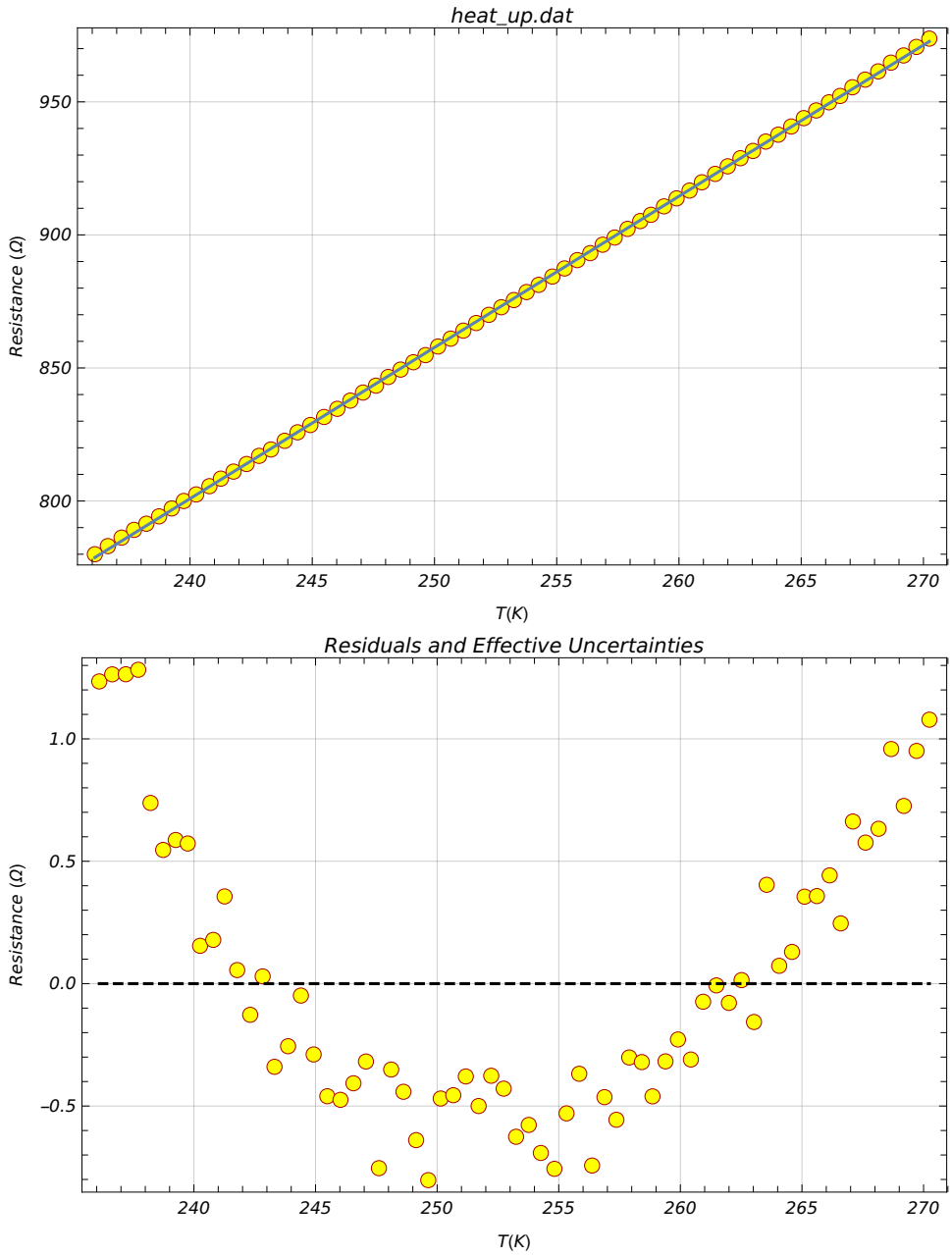
We now check for over-fitting by fitting the same dataset to a linear model.

In[377]:=
```
With[{x = SetXRange[ LinearDataPlot[], Log -> False,
    Label -> "Set the X values for the range you wish to keep."]},
 Print[x];

 XRangeKeep[Sequence @@ x]
]
```

In[378]:=
```
LinearFit[]
```

In[380]:=
```
LinearDifferencePlot[FrameLabel → {"T(K)", "Resistance (Ω)"}]
```

Out[380]=



$y(x) = a + b\ x$

a=	b=	
−562.695	5.68157	
σ_a=	σ_b=	$\chi^2/(n-2)$=
0.257234	0.0010147	11 825.4

As expected, we obtain a much worse fit to the linear model and a definite structure to our residuals, indicating that we have not been over-fitting a linear dataset (as it may appear to be at first glance) to a power-law model.

We now test for the presence of linear correction terms that may allow us to obtain a better fit by fitting

a Log-Log of our data to a linear model (Log-Log will linearize any linear, polynomial, or exponential terms to our model).

In[326]:=
```
With[{x = SetXRange[ LinearDataPlot[], Log -> False,
    Label -> "Set the X values for the range you wish to keep."]},
 Print[x];

 XRangeKeep[Sequence @@ x]
]
```

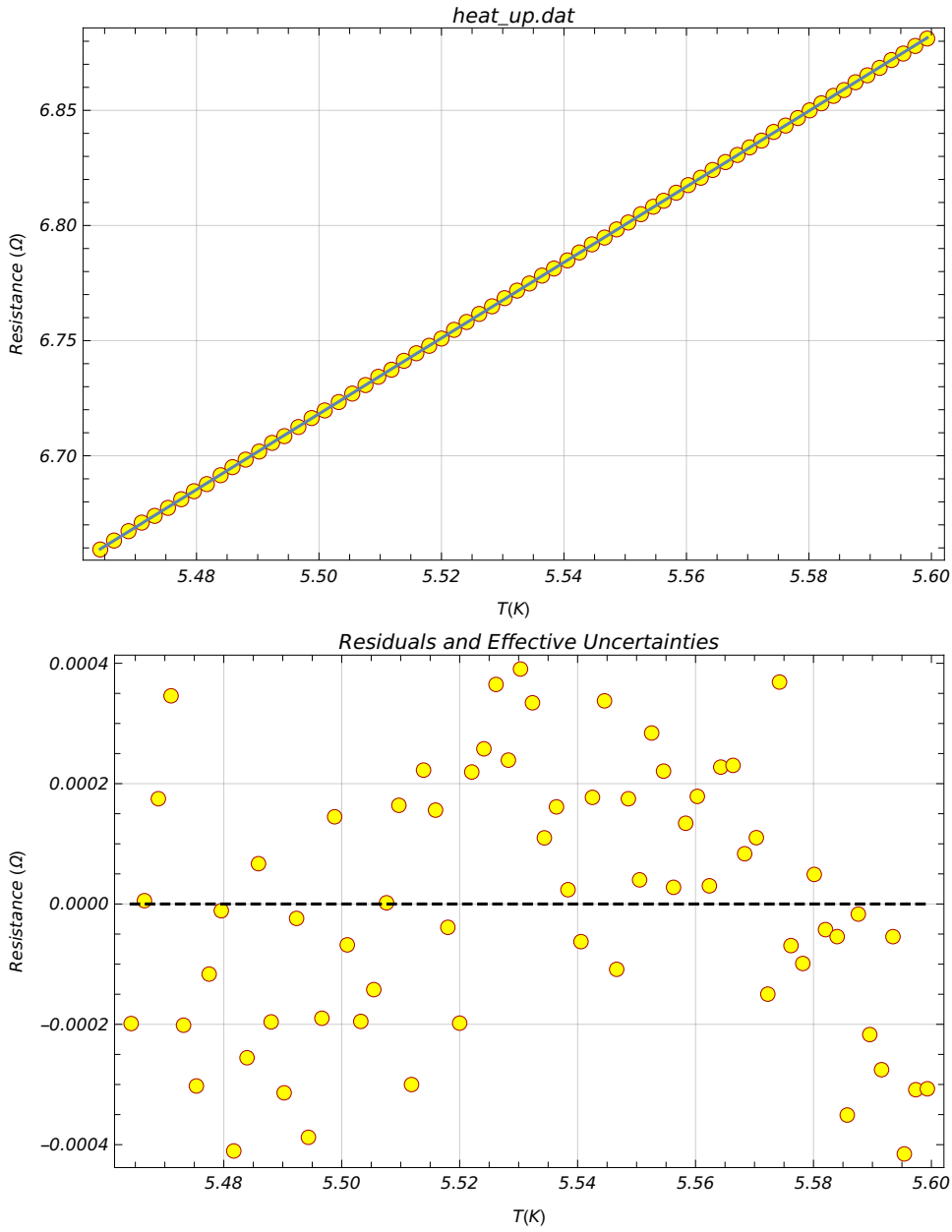
In[381]:=
```
xnew[ x_, y_ ] := Log[x]
ynew[ x_, y_ ] := Log[y]

DataTransform[]

(* Use Undo[] if you don't like the results. *)
```

```
All 67 points transformed.
```

In[384]:=
```
LinearFit[]
```

In[385]:=
```
LinearDifferencePlot[FrameLabel → {"T(K)", "Resistance (Ω)"}]
```

Out[385]=



$y(x) = a + b\,x$

a =	b =	
−2.32423	1.64408	
σ_a =	σ_b =	$\chi^2/(n-2)$ =
0.000103808	0.0000187483	1312.54

We do not obtain a significantly better fit that using the power-law model (in fact it is slightly worse), so we conclude that the bad-fit issue cannot be resolved by adding degrees of freedom (correction parameters) to our model. Therefore there must indeed be a source of error in the resistance and /or temperature measurements that the room-temperature data did not allow us to detect. This source of noise must then either be temperature-dependent or transient.

High-temperature behavior

We now analyze the asymptotic behavior at high temperatures. At high temperatures, the charge carriers are principally intrinsic in nature, and resistance follows an exponential relationship to the inverse of temperature as it did for the pure semiconductor. We algebraically modify the dataset to linearize such a model and restrict our fits to linear regions of the modified dataset.

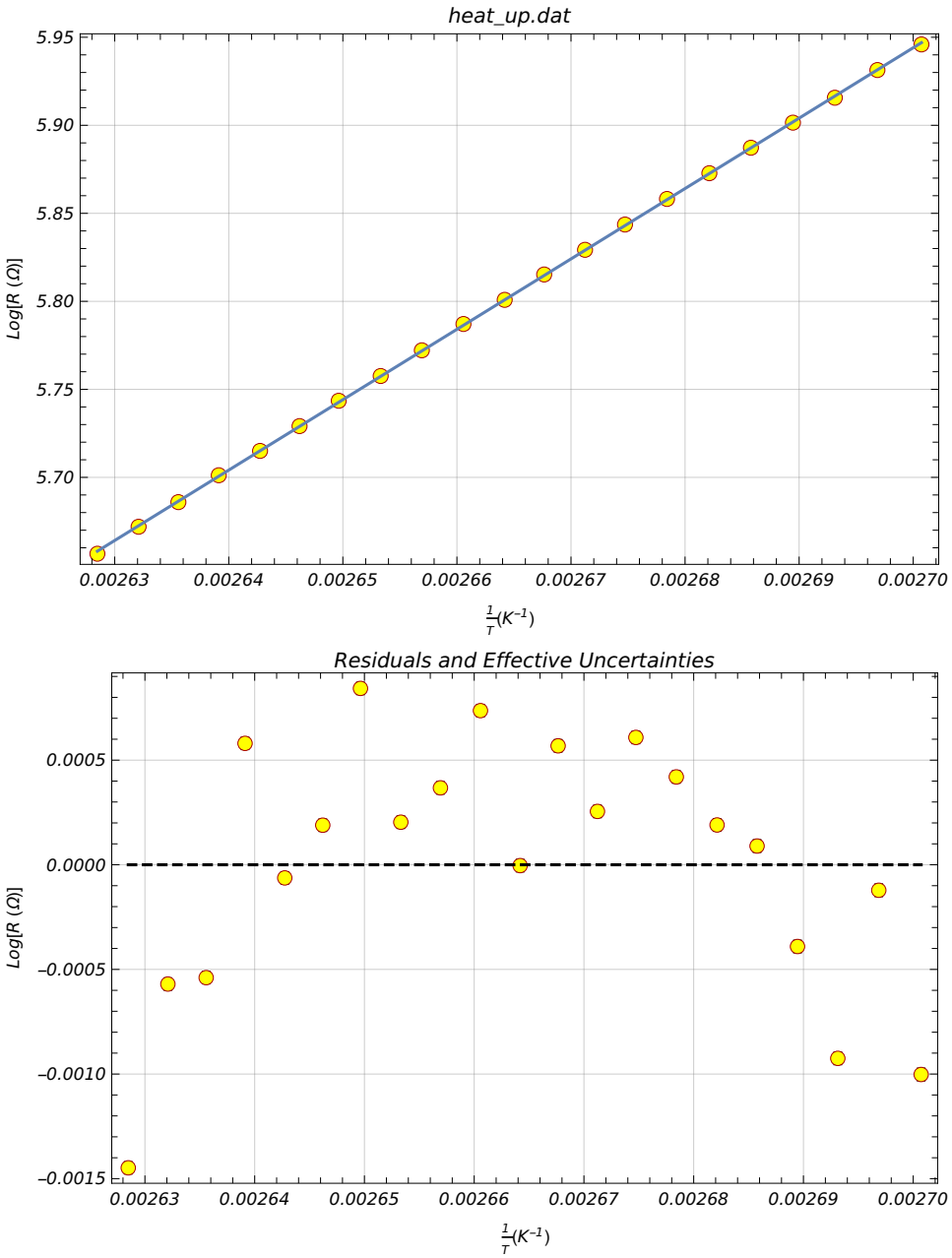
In[344]:=
```
With[{x = SetXRange[ LinearDataPlot[], Log -> False,
    Label -> "Set the X values for the range you wish to keep."]},
 Print[x];

 XRangeKeep[Sequence @@ x]
]
```

In[338]:=
```
xnew[ x_, y_ ] := 1.0 / x
ynew[ x_, y_ ] := Log[y]

DataTransform[]

(* Use Undo[] if you don't like the results. *)
```

In[345]:=
```
LinearFit[]
```

In[346]:=
```
LinearDifferencePlot[FrameLabel → {"1/T (K^-1)", "Log[R (Ω)]"}]
```

Out[346]=




```
y(x) = a + b x
a=                b=
-4.85165          3998.4
σa=               σb=             χ^2/(n-2) =
0.000681282       0.255756        611.727
```

Again we obtain a bad fit, with some structure remaining in the residuals.

We attempt the same approach for the overnight cool-down dataset in an attempt to minimize systematic errors due to temperature gradients (this dataset contains no asymptotic low-temperature

behaviour, so we could not do that for the high-temperature behavior).

In[386]:=
```
With[ {name = SystemDialogInput["FileOpen",
     {DataFileName, {"data files" -> {"*.dat", "*.mca"}, "all files" -> {"*"}}}]},
 If[ name =!= $Canceled,
  LoadFile[name]]
]
```

In[387]:=
```
AssignYsigmas[ errSemi](* insert value between brackets *)
```

In[388]:=
```
SwitchXXandYY[]
```

In[389]:=
```
AssignYsigmas[ errTemp](* insert value between brackets *)
```

In[390]:=
```
SwitchXXandYY[]
```

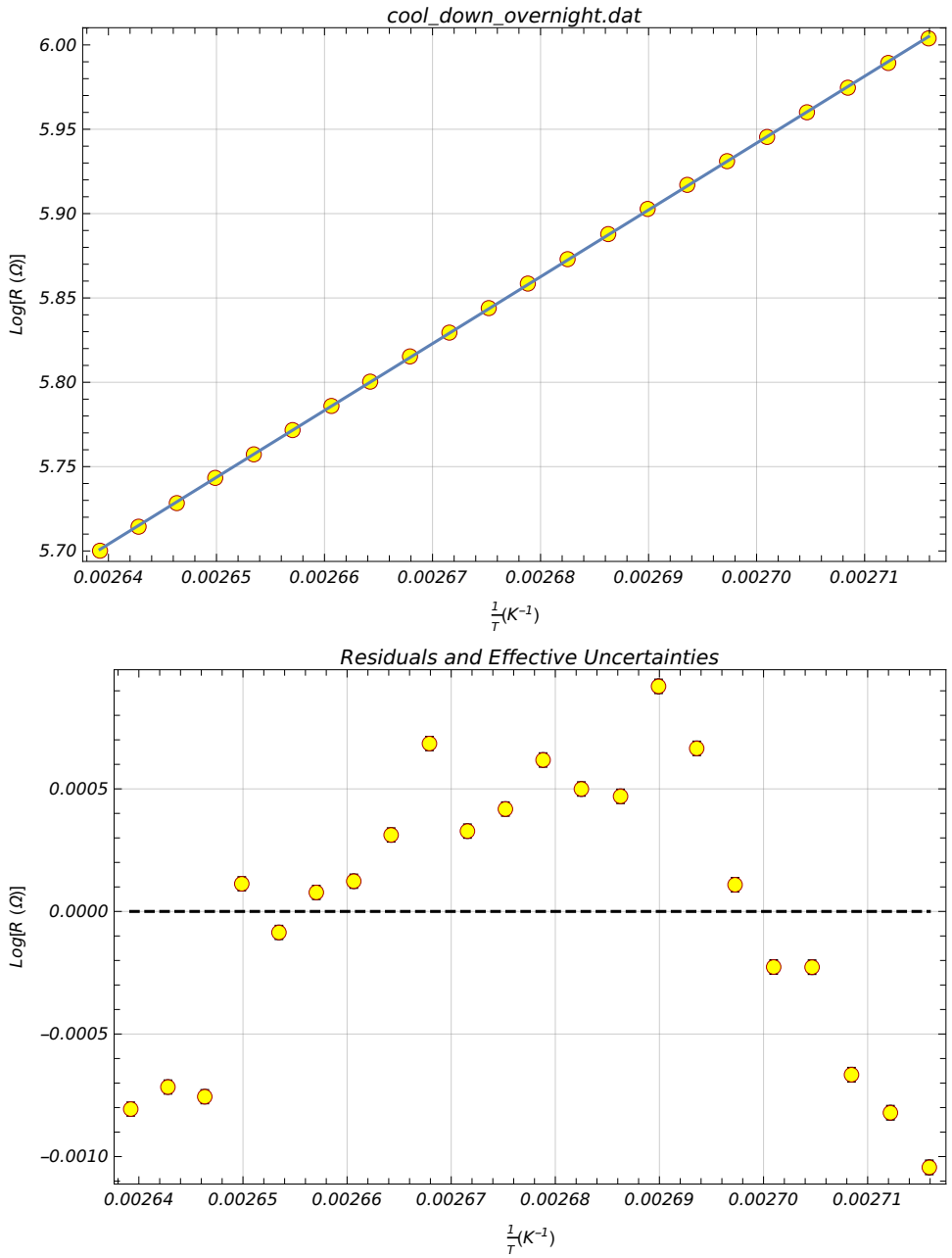
In[397]:=
```
With[{x = SetXRange[ LinearDataPlot[], Log -> False,
     Label -> "Set the X values for the range you wish to keep." ]},
 Print[x];

 XRangeKeep[Sequence @@ x]
]
```

In[391]:=
```
xnew[ x_, y_ ] := 1.0 / x
ynew[ x_, y_ ] := Log[y]

DataTransform[]

(* Use Undo[] if you don't like the results. *)
```

In[398]:=
```
LinearFit[]
```

In[399]:=
```
LinearDifferencePlot[FrameLabel → {"1/T (K^-1)", "Log[R (Ω)]"}]
```

Out[399]=



$y(x) = a + b\,x$

$a =$	$b =$	
−4.75817	3962.96	
$\sigma_a =$	$\sigma_b =$	$\chi^2/(n-2) =$
0.000627943	0.234595	541.948

We only obtain a slightly better fit, again indicating that temperature gradients are not the issue. Again, there is some structure to the residuals, indicative of a temperature-dependent source of error in either temperature or resistance measurements, or as a correction term in our model. Given the temperature range and behavior we are investigating, this could be as suggested in the analysis of the pure semicon-

ductor, due to the variation of E_g with temperature.

Nevertheless, we use our best fit parameters to estimate the energy gap of the doped semiconductor.

In[476]:=
```
egDoped = 2 kb / (1.602 * 10^-19) {3962.9602891968248`, 0.23459536880888965`} (* in eV *)
```

Out[476]=
```
{0.683054, 0.0000404348}
```

Therefore, we estimate $E_g = 0.68305 \pm 0.00004$ eV for the energy gap of the doped semiconductor. While this is not quite within error of the expected energy gap for Germanium, and keeping in mind that we have yet to consider systematic uncertainties in our parameters, this is close enough to confidently conclude that the doped semiconductor is made of Germanium.

Estimating the dopant fraction

We know that the curves for asymptotic behavior of resistance at low temperatures and high temperatures (respectively going to $T^{\frac{3}{2}}$ and to $e^{E_g/(2\,k_b\,T)}$) intersect when $N_d = 2\,n_i$. We therefore use the intersection of lines on a Log-Log of our dataset to solve for the temperature at which this is the case. We then used the variation of n_i with temperature given in the lab manual (and used several times above) along with the value of $\frac{n_i}{N}$ at room temperature (300 K) to find $\frac{n_i}{N}$ at that temperature, and hence solve for $\frac{N_d}{N.} = 2\,\frac{n_i}{N}$

In[406]:=
```
With[ {name = SystemDialogInput["FileOpen",
    {DataFileName, {"data files" -> {"*.dat", "*.mca"}, "all files" -> {"*"}}}]},
 If[ name =!= $Canceled,
  LoadFile[name]]
]
```

In[415]:=
```
AssignYsigmas[ errSemi](* insert value between brackets *)
```

In[416]:=
```
SwitchXXandYY[]
```

In[417]:=
```
AssignYsigmas[ errTemp](* insert value between brackets *)
```

In[418]:=
```
SwitchXXandYY[]
```

In[419]:=
```
xnew[ x_, y_ ] := Log[x]
ynew[ x_, y_ ] := Log[y]

DataTransform[]

(* Use Undo[] if you don't like the results. *)
```

In[426]:=
```
With[{x = SetXRange[ LinearDataPlot[], Log -> False,
    Label -> "Set the X values for the range you wish to keep." ]},
 Print[x];

 XRangeKeep[Sequence @@ x]
]
```

In[427]:=
```
LinearFit[]
```

```
n = 37

y(x) = a + b x

Fit of (x,y)  (unweighted)

a=            b=
-2.35241      1.64919
σa=           σb=           Std. deviation=
0.00726253    0.0013153     0.000177206

Fit of (x,y±σy)

a=              b=
-2.3515         1.64903
σa=             σb=             χ^2/(n-2)=
0.0000820634    0.0000148574    7844.19

Fit of (x±σx,y±σy)

a=              b=
-2.35201        1.64912
σa=             σb=             χ^2/(n-2)=
0.000251959     0.0000456213    831.509
```

In[464]:=
```
a1 = {-2.352008390458569`, 0.00025195861243940506`};
b1 = {1.6491193023567654`, 0.00004562134771860273 4`};
```

In[445]:=
```
Undo[]
```

In[448]:=
```
LinearDifferencePlot[FrameLabel → {"Log[T]", "Log[R]"},
 PlotRange → {Automatic, {5.6, 7.5}}]
```

In[454]:=
```
With[{x = SetXRange[ LinearDataPlot[], Log -> False,
    Label -> "Set the X values for the range you wish to keep." ]},
 Print[x];

 XRangeKeep[Sequence @@ x]
]
```

In[455]:=
```
LinearFit[]
```

```
n = 16

y(x) = a + b x

Fit of (x,y) (unweighted)

a=          b=
67.8551     -10.4678
σa=         σb=          Std. deviation=
0.132623    0.0224426    0.000582378

Fit of (x,y±σy)

a=             b=
67.8167        -10.4613
σa=            σb=             χ²/(n-2)=
0.000973175    0.000164709     19052.7

Fit of (x±σx,y±σy)

a=             b=
67.8611        -10.4689
σa=            σb=             χ²/(n-2)=
0.00579526     0.00098069      526.407
```

In[466]:=
```
a2 = {67.861086558961`, 0.005795255742176451`};
b2 = {-10.468855441210536`, 0.0009806895850823203`};
```

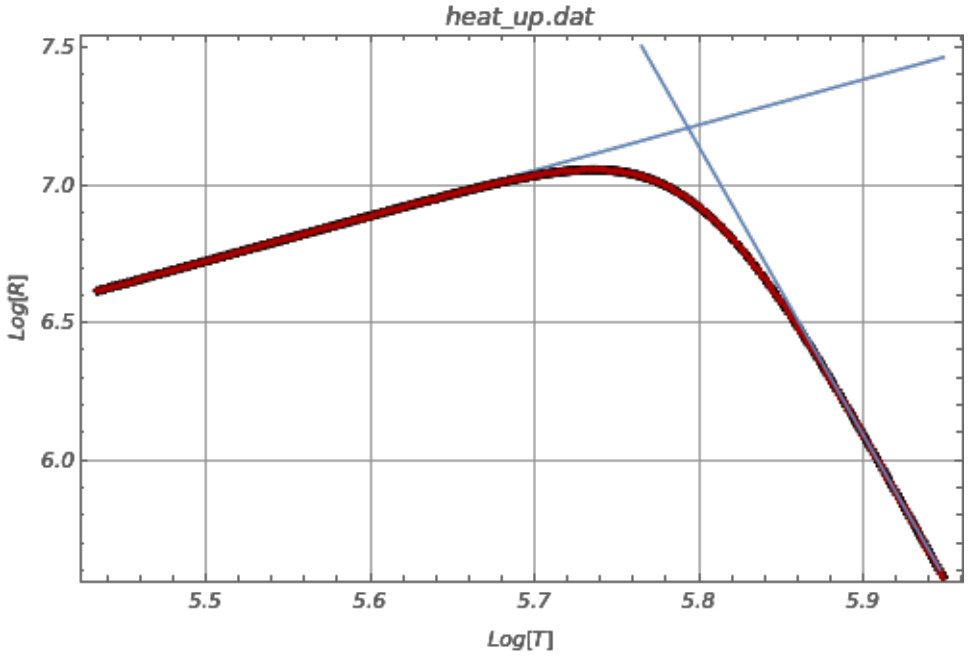
In[458]:=
```
Undo[]
```

In[459]:=
```
LinearDifferencePlot[FrameLabel → {"Log[T]", "Log[R]"},
 PlotRange → {Automatic, {5.6, 7.5}}]
```

Here is a plot of the intersection of our fits to the asymptotic models. We define the appropriate functions to calculate the intersection temperature T (K) from the fit parameters above, and use this to calculate $\frac{N_d}{N}$.

In[462]:=
```
ImageCompose[plt1, plt2]
```

Out[462]=



In[463]:=
```
intersection[a1_, b1_, a2_, b2_] :=
  (b2[[1]] - b1[[1]])/(a1[[1]] - a2[[1]]) {1, Sqrt[b1[[2]]^2 + b2[[2]]^2/(b2[[1]] - b1[[1]])^2 + a1[[2]]^2 + a2[[2]]^2/(a1[[1]] - a2[[1]])^2]};
```

In[473]:=
```
int = intersection[b1, a1, b2, a2];
```

In[475]:=
```
temp = ℯ^int
```

Out[475]= {328.366, 1.00067}

In[477]:=
```
<< SimpleErrorPropagation`
```

In[478]:=
```
? PropagateUncertainties
```

PropagateUncertainties[func, vars] propagates uncertainties in variables listed
in vars = {variable1, variable2,...} and retuns the propagated uncertainty in func.
func must necessarily be differentiable with respect to all variables in vars,
and the uncertainties in variables in vars must be independent.

It will return an algebraic expression in terms of the variables
in vars, with uncertainty in some variable denoted by sigmaVariable.
It can be used either by providing an algebraic expression, then substituting values for the
variables and uncertainties in a subsequent operation, or simply by assigning values
and uncertainties to each variable using the naming conventions mentioned. In the
latter case, it will be necessary to localize the variables in vars using Module or Block.

Note: Loading SimpleErrorPropagation` clears any variables named 'variance', 'varianceList'
or 'PropagateUncertainties' due to function and subfunction names in the Package.

In[482]:=
```
prop[t_] := (5.4 * 10^-10) t^(3/2) ℯ^(-0.67*1.602*10^-19/(2 kb t))/(300^(3/2) ℯ^(-0.67*1.602*10^-19/(2 kb 300)))
```

In[484]:=
```
PropagateUncertainties[prop[t], {t}] // Simplify
```

Out[484]= $6.60898 \times 10^{-8} \sqrt{\frac{e^{-7774.45/t}\ \text{sigmaT}^2\ (2591.48 + 1.\ t)^2}{t}}$

In[485]:=
```
prop2[t_, sigmaT_] := {prop[t], 6.60898433030618`*^-8 Sqrt[(1/t) ℯ^(-7774.445893089961`/t) sigmaT^2 (2591.4819643633205` + 1.` t)^2]};
```

In[487]:=
```
niFrac = prop2[temp[[1]], temp[[2]]];
```

In[488]:=
```
npFrac = 2 niFrac
```

Out[488]= $\{3.78785 \times 10^{-9},\ 1.53964 \times 10^{-10}\}$

Therefore we can estimate $\frac{N_d}{N} = (3.79 \pm 0.15) \times 10^{-9}$ as the doping concentration for our sample.

Systematic uncertainties

Due to temperature gradients

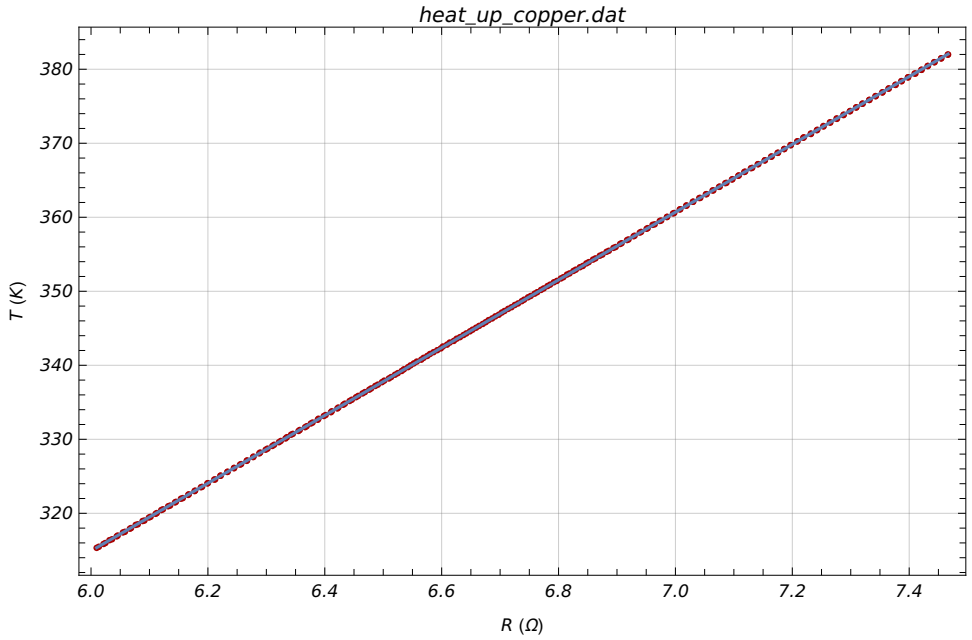
We observe that the only dataset with a good enough fit to provide us with a reliable estimate of the offset in temperature (systematic error) caused by temperature gradients in the dielectric fluid is the copper resistance data. However, the linear offset parameters are within error of each other, making it difficult to infer a temperature offset from a comparison of the heat-up and overnight cool-down data (the latter should be relatively free of such temperature gradients).

Therefore, we simply obtain an upper bound for the magnitude of the systematic error by doubling the standard deviation of a linear fit to the combined heat-up and overnight cool-down datasets for temperature against copper resistance (as opposed to our previous fits of resistance against temperature).

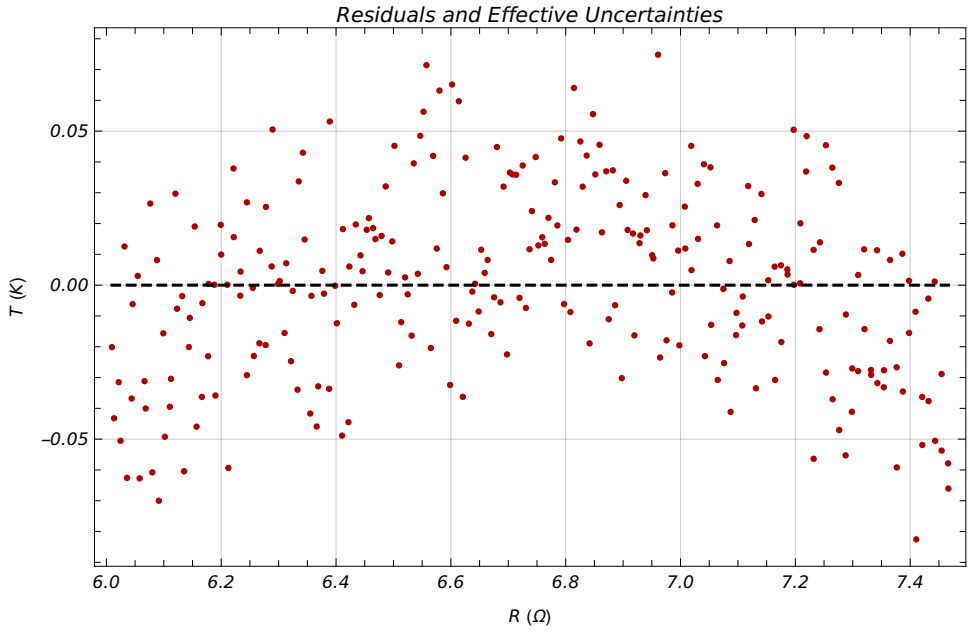
Merging datasets

Estimating error from standard deviation

In[516]:=
```
LinearDifferencePlot[FrameLabel → {"R (Ω)", "T (K)"}]
```



Out[516]=



y(x) = a + b x

a=	b=	
40.2118	45.783	
σ_a=	σ_b=	Std. deviation=
0.0300577	0.00445253	0.03055

In[•]:=
```
gradErr = 2 (0.030550010541453718`);
```

Due to temperature probe calibration

According to the probe data sheet, the maximum admissible error for the temperature measured is of the order of 0.37 K for our temperature range.

In[520]:=
```
calErr = 0.37;
```

Therefore the total systematic error in temperature is found by adding in quadrature:

In[521]:=
```
sysTempErr = Sqrt[calErr^2 + gradErr^2]
```

Out[521]=
```
0.375011
```

So the systematic error in temperature is of the order of 0.38 K.

Due to resistance measurement calibration

Copper

Using the accuracy specifications on pg 326 of the Agilent 39470A's user manual, we can calculate:

In[523]:=
```
errResCopper = ((0.0030 + 0.0006) * 10) + ((0.0035 + 0.0005) * 100) / 100 (* in Ω *)
```

Out[523]=
```
0.04
```

Pure semiconductor (thermistor)

In[528]:=
```
errResThermistor = ((0.002 + 0.0006) * 4 * 10^4 + (0.0005 + 0.0001) * 10^4) / 100 (* in Ω *)
```

Out[528]=
```
1.1
```

Doped semiconductor

In[529]:=
```
errDoped = (0.002 + 0.0006 + 0.0006 + 0.0001) * 10^3 / 100 (* in Ω *)
```

Out[529]=
```
0.033
```

Parameter estimates for samples (including systematic uncertainties)

Temperature coefficient of resistance

Recall that in terms of fit parameters and temperature T, $\alpha = \frac{b}{a+b\,(T)}$. Then a systematic error in R corresponds to introducing an error in a, so that the uncertainty in α is given by:

In[538]:=
```
PropagateUncertainties[b/(a + b T), {a, T}] // Simplify
```

Out[538]=
$$\sqrt{\frac{b^2\,\left(\text{sigmaA}^2 + b^2\,\text{sigmaT}^2\right)}{\left(a + b\,T\right)^4}}$$

In[546]:=
```
finalAlpha0 =
 {alpha0[[1]], alpha0[[2]], alpha0[[1]]^2 Sqrt[(errResCopper/ab0[[2]])^2 + sysTempErr^2]}
```

Out[546]= {0.00428583, 0.0000108376, 0.0000343807}

In[547]:=
```
finalAlpha20 =
 {alpha20[[1]], alpha20[[2]], alpha20[[1]]^2 Sqrt[(errResCopper/ab20[[2]])^2 + sysTempErr^2]}
```

Out[547]= {0.00393417, 0.0000107321, 0.000029066}

Gap energy

Observe that the logarithmic operation on the data to linearize the intrinsic charge carrier datasets (pure semiconductor or doped at high temperature) discards any effect of a systematic error in resistance. Therefore we only need to worry about the systematic error induced in the gap energies by the systematic error in temperature. Given the linear relationship between the fit parameter b and the gap energy, the systematic error in the gap energy should then simply be the gap energy scaled by the fractional systematic error in $\frac{1}{T}$. We obtain an upper bound on the error simply by taking T to be the lowest temperature in our dataset.

In[550]:=
```
finalEgPure = {eg[[1]], eg[[2]], sysTempErr/230 eg[[1]]}
```

Out[550]= {0.676627, 0.00277904, 0.00110323}

In[554]:=
```
finalEgDoped = {egDoped[[1]], egDoped[[2]], sysTempErr/370 egDoped[[1]]}
```

Out[554]= {0.683054, 0.0000404348, 0.000692305}

Doping fraction

Observe that the Log-Log operation and solving for the asymptotic intersection temperature discards any systematic error in resistance (whether a scaling or an offset), and introduces an error in the intersection temperature in accordance with the systematic error in T, as expected. Therefore we simply propagate the uncertainty in T as before, but this time consider the systematic uncertainty in T.

In[557]:=
```
finalNp = {npFrac[[1]], npFrac[[2]], 2 prop2[temp[[1]], sysTempErr][[2]]}
```

Out[557]= $\{3.78785 \times 10^{-9}, 1.53964 \times 10^{-10}, 5.76995 \times 10^{-11}\}$

Final parameter estimates

In[561]:=
```
param = {{"Sample", "Parameter", "Estimate"},
    {"Copper", "α0 °C (°C^-1)", "0.004286 ± 0.000011 ± 0.000029"},
    {"Copper", "α20 °C (°C^-1)", "0.003934 ± 0.000011 ± 0.000029"},
    {"Thermistor", "Eg (eV)", "0.6766 ± 0.0028 ± 0.0011"},
    {"Doped Semiconductor", "Eg (eV)", "0.68305 ± 0.00004 ± 0.00069"},
    {"Doped Semiconductor", "Np/N", "(3.79 ± 0.15 ± 0.06) × 10^-9"}};
```

In[562]:=
```
Grid[param, Alignment → Left, Spacings → {2, 1}, Frame → All,
 ItemStyle → "Text", Background → {{Gray, None}, {LightGray, None}}]
```

Out[562]=

Sample	Parameter	Estimate
Copper	$\alpha_{0\,°C}$ ($°C^{-1}$)	0.004286 ± 0.000011 ± 0.000029
Copper	$\alpha_{20\,°C}$ ($°C^{-1}$)	0.003934 ± 0.000011 ± 0.000029
Thermistor	E_g (eV)	0.6766 ± 0.0028 ± 0.0011
Doped Semiconductor	E_g (eV)	0.68305 ± 0.00004 ± 0.00069
Doped Semiconductor	$\frac{N_p}{N}$	(3.79 ± 0.15 ± 0.06) × 10^{-9}

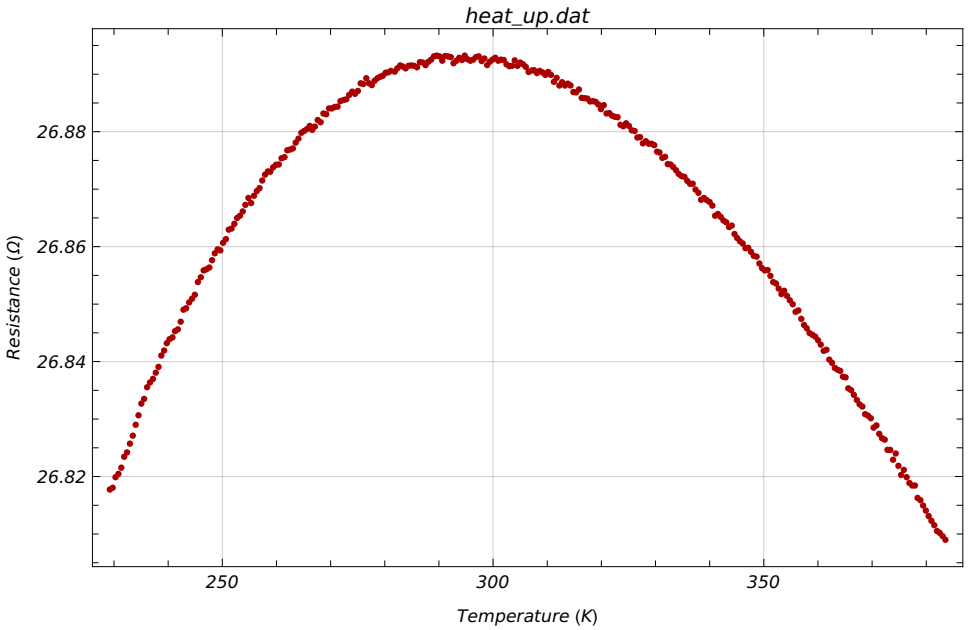
Other resistors

Manganin wire

In[563]:=
```
With[ {name = SystemDialogInput["FileOpen",
     {DataFileName, {"data files" -> {"*.dat", "*.mca"}, "all files" -> {"*"}}}]},
  If[ name =!= $Canceled,
   LoadFile[name]]
 ]
```

In[565]:=
```
LinearDataPlot[FrameLabel → {"Temperature (K)", "Resistance (Ω)"}]
```

Out[565]=

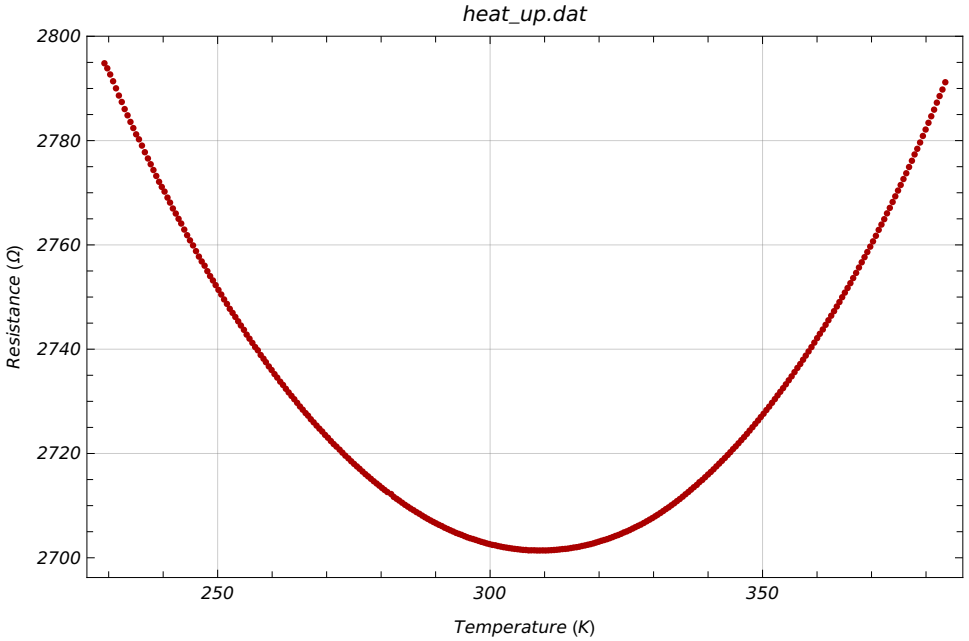


Commercial resistor

In[566]:=
```
With[ {name = SystemDialogInput["FileOpen",
     {DataFileName, {"data files" -> {"*.dat", "*.mca"}, "all files" -> {"*"}}}]},
 If[ name =!= $Canceled,
  LoadFile[name]]
]
```

In[567]:=
```
LinearDataPlot[FrameLabel → {"Temperature (K)", "Resistance (Ω)"}]
```

Out[567]=



In both cases, we observe a very small fractional change in resistance over our temperature range. Furthermore, they both have a stationary point near room temperature. This appears to be by design, seeing that this ensures that resistance is essentially constant (the derivative of resistance with respect to temperature vanishes near the stationary point) over a range of operating temperatures which we of course expect to be around room temperature. This is essential to ensure proper functioning of devices using these components as resistors (we want the resistance to be as rated at room temperature and remain unchanged over a range of reasonable working temperatures to account for temperature fluctuations and self-heating).