Determination of Physical Properties of Commercial and Hand Welded K-Type Thermocouples, Error From Sampling Rates, and NaCl Concentration in Water

Lucas Culverhouse

Feb 1, 2023

ECH 145A Section A06

Abstract

A K-type thermocouple was constructed and tested with a commercially available thermocouple for their Seebeck coefficients, which were found to be in agreement with literature values. The time constant of the welded thermocouple was experimentally by rapid temperature change from ambient to an ice bath; a theoretical minimum time constant is also presented for comparison. The welded thermocouple was also tested at a variety of data collection and sampling rates to determine the effect on accuracy and precision of temperature measurements. Finally, the welded thermocouple was used to determine the boiling point elevation and NaCl concentration of an aqueous solution.

Introduction

The following experiments were performed to demonstrate and understand the properties and usefulness of the thermocouple, a common tool in experimental thermodynamics. Several experiments were conducted with a commercially available thermocouple and a hand welded thermocouple whose construction is described in the experimental methods section. The process of data collection using a thermocouple demonstrates the Seebeck effect, among others, which is an important theoretical result in thermodynamics.

By measuring some of these properties useful information is also gathered about error propagation and analysis, the effects of sampling rates, and comparison of literature or theoretical values to experimental values.

Additionally, concepts such as fundamental time constants, and boiling point elevation are explored; again using the constructed thermocouple.

Theory

The Seebeck Effect. The Seebeck effect is based on the principle that gradient of electric potential, or voltage, is proportional to the temperature gradient across a conductor. The relation is given by

$$-\nabla V = s\nabla T$$

Where s is the Seebeck coefficient[1]. The Seebeck coefficient is highly dependent on the composition of the material, which is why the alloys chromel and alumel are chosen for this experiment. This effect enables us to measure temperature by recording the voltage measured by a thermocouple, given we know its Seebeck coefficient.

Data Collection and Sampling Rate. The data collection and sampling rate are two related but different terms. Data collection rate refers to the amount of times per second that an analog electrical signal is converted into a digital signal to be read by a computer. The sampling rate is then the number of times data is reported by the computer to the user. The sampling rate must be equal to or lower than the data collection rate. In the case where it is lower, the data points that are collected but not reported are averaged together to give a single reported sample.

Experimental Time Constant. The time constant is a measure of how long a thermocouple will respond to a temperature change. For our experiment where the final temperature is 0°C, we can represent the temperature measured by a thermocouple during a rapid temperature change as

$$T(t) = T_0 e^{-\frac{t}{\tau}}$$

This relation is approximate and only valid when the thermocouple undergoes a large, rapid temperature change and the final temperature, often denoted as T_∞ is 0°C[2]. From this equation we can experimentally determine the time constant of our system.

Theoretical Time Constant. The theoretical time constant is much harder to determine well. It relies on several approximations and assumptions that mean the calculated value is only τ_{min} or the theoretical minimum value for the time constant. The value of τ_{min} is given by the following relation

$$\tau_{min} = \frac{a^2}{(K\pi^2)}$$

Where a is the bead diameter and K is the thermal diffusivity[2].

Boiling Point Elevation. The boiling point of water is increased by greater concentration of solutes dissolved within it. This effect is known as boiling point elevation. The boiling point elevation can be calculated as follows

$$\Delta T_b = iK_b m$$

Where ΔT_b is the boiling point elevation, i is the van't Hoff factor, K_b is the molal elevation constant, and m is the molal concentration. The van't Hoff factor is effectively a measure of how many entities result from dissolution into a solvent. For ionic compounds, like NaCl, this is effectively the number of ions that make up the compound. K_b is a proportionality constant that depends on the substance and relates molal concentration to change in boiling point[3].

Error Analysis. For this experiment several techniques of error analysis are used. The selection of the error analysis technique relies on knowing the sample size, N, of your data set.

The first technique is standard deviation and Gaussian distributions. A Gaussian or normal distribution is usually assumed for data and from this we can get the 2σ standard of error reporting. For a Gaussian distribution, two standard deviations, or 2σ, means that there is a 95.5% probability that a measurement will lie in the given interval.

Often, however, sample sizes can be much smaller than is optimal for a gaussian distribution. Here we use a modified but similar distribution called the Student's t-distribution. This is best applied when N < 30. This distribution gives rise to the confidence interval given by the relation

$$CI = \sigma\left(\frac{t}{\sqrt{N}}\right)$$

Where CI is the confidence interval, σ is the standard deviation, N is the sample size, and t is the t-value. The t-value is a special corrective factor that depends on the sample size and desired confidence level. Often a 95% confidence interval is selected for error analysis.

Experimental Methods

The experiment is broken into several parts. Unless otherwise mentioned "thermocouple" refers to the constructed thermocouple and not the commercially provided one.

Construction of the Welded Thermocouple. Two #24 American wire gauge wires of chromel and alumel were welded together with a carbon block and welder set to LOW power and 60 J. The ends of this wire were stripped to allow connection to a data acquisition unit, DAQ, for the rest of the data collection performed in this experiment.

Determination of Seebeck Coefficients. To determine the Seebeck coefficients of both the commercial and welded thermocouple, each was tested at two different known temperatures. The first is an ice bath at 0°C and the second is boiling distilled water at 100°C. The thermocouples were held in each temperature for approximately 40 seconds to collect enough data for error analysis. Another set of trials was taken for each at room temperature to be used for validation.

Sampling and Data Collection Rates. To determine the effect of data collection and sampling rates on accuracy and precision several measurements were taken with a thermocouple in an ice water bath. The thermocouple was placed in the water and time allowed for the temperature to stabilize. This was repeated at 3 different data collection and sampling rates.

Temperature Step Change and Time Constant. To experimentally determine the time constant of the thermocouple several measurements were taken of the temperature readings over time after being rapidly plunged into an ice bath. An ice bath was prepared and then the thermocouple was rapidly moved from the surrounding ambient temperature to the bath and held there until the temperature stabilized.

Additionally measurements of the thermocouple bead size were taken to later determine the theoretical time constant.

Concentration of NaCl. A solution of unknown sodium chloride was provided and its boiling point elevation measured. This was done by bringing the solution to a boil while maintaining the water level. Then the thermocouple was placed into the solution to measure the temperature of the boiling solution to later determine the boiling point elevation.

Results

All the following data is reported with a 95% confidence limit and N=3 unless otherwise specified.

Seebeck Coefficients. The Seebeck coefficients of both the welded and commercial thermocouples were obtained experimentally by fitting a curve to the voltages recorded in water at known temperatures. Data was taken at ambient temperature and plotted but not used for fitting.

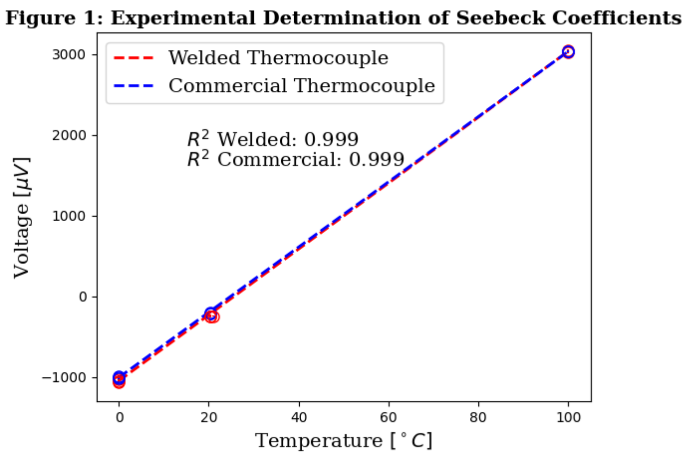


Figure 1. Sampling rates for voltage averages were 60 Hz. Ice water was used for 0°C and boiling distilled water used for 100°C. Error bars were plotted but are not visible as they are on the order of ±1 μV. Commercial Seebeck coefficient calculated is 40.37 ± 0.28 μV / °C and welded Seebeck coefficient is 40.81 ± 0.34 μV / °C . Curve is only fitted to values at 0 °C and 100 °C, values at room temperature are provided only for comparison.

The results obtained for the Seebeck coefficient (s) are presented in table 1 along with a value cited from literature.

Table 1. Experimental Value for Seebeck Coefficients with 2σ errors.

Thermocouple	s [μV / °C]
Commercial	40.37 ± 0.28
Welded	40.81 ± 0.34
Literature[4]	41

Sampling and Data Collection Rates. The following table presents the readings from the previously used welded thermocouple at various sampling and data collection rates from the DAQ.

Table 2. Thermocouple Reading Accuracy and Precision at Various Sampling and Data Collection Rates.

No. of Trials	Collection Rate [Hz]	Sample Rate [Hz]	Avg. Temp [°C]	2σ [°C]	Expected Temp [°C]
3	60	10	1.05	1.6	0
3	2000	50	-0.07	0.2	0
3	10	1	-0.05	0.1	0

Each trial was taken in an ice bath; the trials do not each have the same number of data points to create the temperature average.

Temperature Step Change and Time Constant. The following is a selected trial of a sudden temperature step change from the thermocouple being moved from ambient temperature to an ice bath suddenly. The final calculations for the time constant take into account more than just the one example trial presented here.

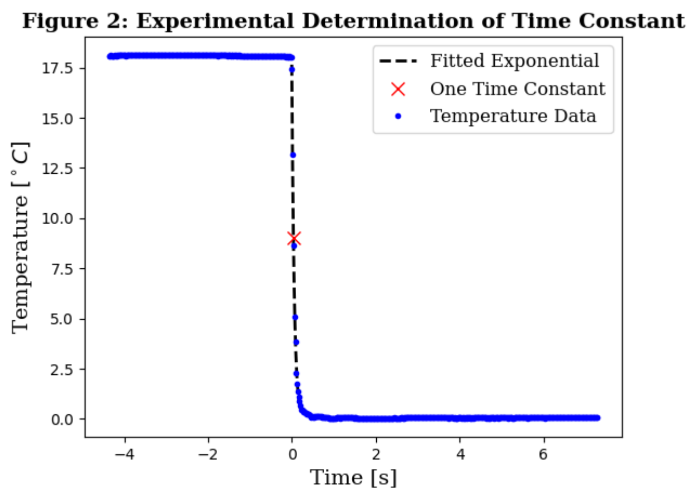


Figure 2. Temperature data collected at sampling rate of 50 Hz. The exponential curve is only fitted for the starting point and 10 points after. The overall temperature drop for this trial is approximately 18 °C. Time is normalized on the figure to have the drop start time at 0 seconds.

From the fitting method demonstrated above the experimental time constant was found to be 0.0494 ± 0.0085 s when averaged across all 3 trials. This is compared to the calculated theoretical value of 0.0324 ± 0.0001 s.

Concentration of NaCl. The concentration of sodium chloride salt in distilled water was determined using the measured average boiling point elevation of the water. For the tested sample D concentration was determined to be 6.26 ± 0.11 kg / mol.

Discussion

Analysis of Thermocouples. The observed Seebeck coefficients fall within error of each other and the common literature value of 41 μV / °C. It can be seen from the data that on average the data for the welded thermocouple is more noisy than the commercial one. Noise in this case is represented by rapid changes in temperature voltage, which is caused by a rapid change in temperature. It makes sense from the analysis that the welded thermocouple would be more noisy as it has a much smaller radius and thermal mass than the commercial thermocouple, causing it to respond to temperature changes more rapidly.

The Seebeck coefficients for each thermocouple being in close agreement with each other mean each should be about as accurate as the other. Due to the higher time constant reducing noise however, the commercial thermocouple should be more precise than the welded variant.

Effects of Data Collection and Sample Rates. From table 2 we can see an interesting effect. It would make sense that a higher sampling rate would be more precise, as there are more data points to limit outliers. However, it seems like in the data presented this is not the case. This is almost certainly due to systematic errors, which should not be affected by sampling rate. For the sampling rate of 10 Hz the first trial was anomalous, having a higher temperature than the others, thus impacting both the accuracy and precision of the data.

A higher sampling rate should reduce random errors, in fact, collecting more data is essentially the only way to remove such errors. As can be seen from table 2 though, strong systematic errors can drastically impact the accuracy and precision of data even at high sample rates.

Additionally, if measuring temperature in a more noisy environment, such as flowing boiling water, a high sample rate may be undesirable due to the high levels of noise present in the data.

Time Constant. The two time constants determined were experimental and theoretical. The theoretical value is much lower due to how it was calculated. The assumptions made mean that what was actually determined is the theoretical minimum time constant, so it makes sense the value we see is less than the one determined experimentally.

For the experimentally determined value the sampling rate should have a large impact on how accurate our measurement is. As the drop in temperature is so rapid a high sampling rate is necessary to actually capture the moment the drop begins. Another source of air also lies in determining the exact moment of the start of the drop. It is assumed that the thermocouple is at a constant temperature and instantaneously moves from one to the other. However, neither of these things are true. It can be seen that there are minor fluctuations in the ambient temperature recording prior to the drop, meaning the thermocouple is not truly at a constant temperature. Additionally, the air around the ice bath should be colder than the rest of the ambient air. This means that the drop in temperature is not actually instantaneous. These sources of error are minor in this particular experiment, but could be much more impactful with a different experimental design.

Concentration of NaCl. The salt concentration was determined to be 6.26 ± 0.11 kg / mol. Much of the error comes from the temperature measurement itself. When measuring boiling water there is a lot of random error due to the bubbles of water vapor that are floating to the surface as the water boils. While the liquid water will have a constant temperature, these bubbles that are in the gas phase can actually have temperatures much higher than the boiling point of the solution. As these bubbles pass by the thermocouple they can increase the temperature it is recording momentarily, these small and rapid temperature changes should be more pronounced when using the welded thermocouple as previously discussed.

Conclusion

From this experiment we can gather a few key results about our thermocouple and data collection in general.

For our thermocouples Seebeck coefficients it can be seen that they are in close agreement with each other and the literature with relatively low amounts of error. It was also determined that the theoretical time constant determined was much lower than the experimental result, supporting the theory.

Sampling and data collection rates were also investigated. It was determined that while sampling and data collection rates do have an impact on the precision and accuracy of the data, systematic errors still must be considered and can often dominate even in instances where the sampling rate is high.

Finally, the concept of boiling point elevation was demonstrated, and it was shown the relative ease that an unknown concentration of solute can be determined using this principle.

Notations

s: The Seebeck coefficient, often measured in V/°C

τ: The fundamental time constant of exponential decay, measured in seconds, related to half-life

T_0: The initial temperature of a system

T_∞: The final, freestream, or equilibrium temperature of a system

K: The thermal diffusivity of a material, often measured in m^2/s

ΔT_b : The boiling point elevation

i: The van't Hoff factor, a measure the number of entities when a compound dissolves into solution

K_b: The molal elevation constant, measured in molal concentration per change in boiling point

m: The molal concentration in kg solute / L of solvent

N: The sample size

σ: The standard deviation

CI: The confidence interval

t: The t-value, determined by the desired confidence level and sample size

DAQ: Data acquisition unit, used to record data to computer

References

[1] Onsager, L. (1931). *Reciprocal Relations in Irreversible Processes. I. Physical Review, 37(4), 405–426.* doi:10.1103/physrev.37.405

[2] Wan, J. *Lab 1: Thermocouples. University of California, Davis, ECH145A.* thermocouples.pdf

[3] Meranda, D., & Furter, W. F. (1977). *Elevation of the boiling point of water by salts at saturation: data and correlation*. *Journal of Chemical & Engineering Data, 22(3), 315–317.* doi:10.1021/je60074a023

[4] Manual on the Use of Thermocouples in Temperature Measurement (4th Ed.). ASTM. 1993. pp. 48–51. ISBN 978-0-8031-1466-1.

[5] Sundqvist, B. (1992). *Thermal diffusivity and thermal conductivity of Chromel, Alumel, and Constantan in the range 100–450 K. Journal of Applied Physics, 72(2), 539–545*. doi:10.1063/1.351885

[6] Hoyt, C. S., & Fink, C. K. (1937). *The Constants of Ebullioscopy. The Journal of Physical Chemistry, 41(3), 453–456.* doi:10.1021/j150381a010

Supplementary Materials

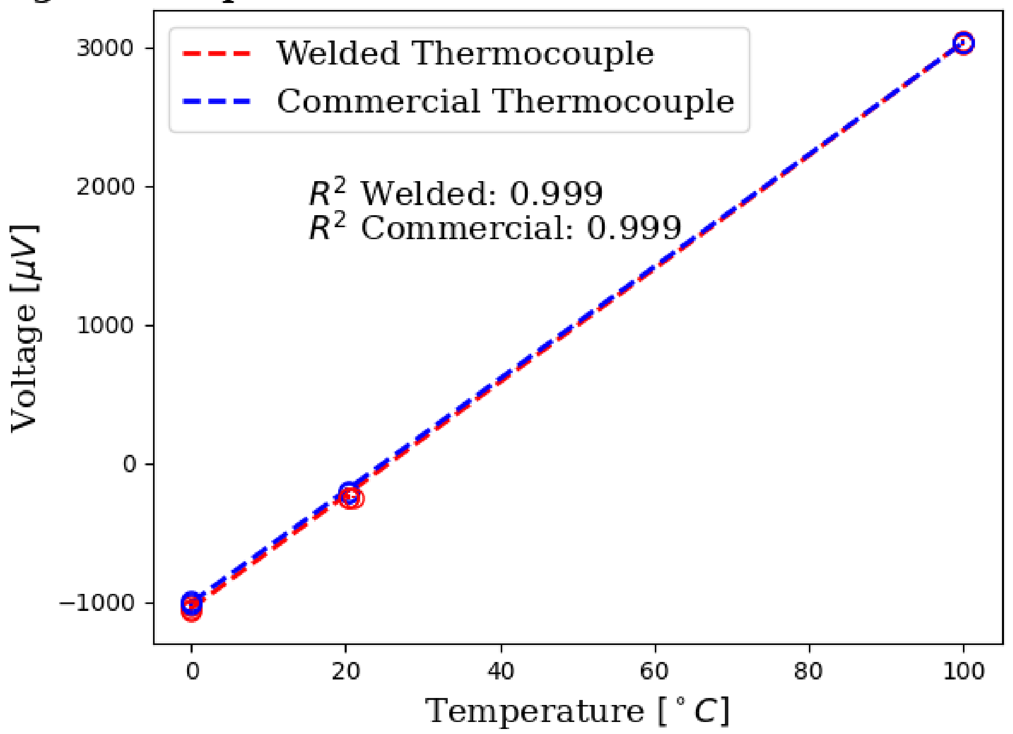
Below is a copy of the notebook file that was used for all the calculations and data analysis in converted pdf form. A copy of the original .ipynb file is also included along with this paper.

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

In [2]:
```
# Data paths
vch = [
    "./Data/Volt/Boiling Water/Trial 1 Commercial.csv",
    "./Data/Volt/Boiling Water/Trial 2 Commercial.csv",
    "./Data/Volt/Boiling Water/Trial 3 Commercial.csv",
]
vcc = [
    "./Data/Volt/Ice Water/Trial 1 Commercial.csv",
    "./Data/Volt/Ice Water/Trial 2 Commercial.csv",
    "./Data/Volt/Ice Water/Trial 3 Commercial.csv",
]
vca = [
    "./Data/Volt/Ambient/Trial 1 Commercial.csv",
    "./Data/Volt/Ambient/Trial 2 Commercial.csv",
    "./Data/Volt/Ambient/Trial 3 Commercial.csv",
]
vbh = [
    "./Data/Volt/Boiling Water/Trial 1 Built.csv",
    "./Data/Volt/Boiling Water/Trial 2 Built.csv",
    "./Data/Volt/Boiling Water/Trial 3 Built.csv",
]
vbc = [
    "./Data/Volt/Ice Water/Trial 1 Built.csv",
    "./Data/Volt/Ice Water/Trial 2 Built.csv",
    "./Data/Volt/Ice Water/Trial 3 Built.csv",
]
vba = [
    "./Data/Volt/Ambient/Trial 1 Built.csv",
    "./Data/Volt/Ambient/Trial 2 Built.csv",
    "./Data/Volt/Ambient/Trial 3 Built.csv",
]
ta = [
    "./Data/Temp/Ambient/Trial 1.csv",
    "./Data/Temp/Ambient/Trial 2.csv",
    "./Data/Temp/Ambient/Trial 3.csv",
]


# Ambient temp pairs
vca1 = np.loadtxt(vca[0], delimiter=",", skiprows=7, usecols=[2])
vca2 = np.loadtxt(vca[1], delimiter=",", skiprows=7, usecols=[2])
vca3 = np.loadtxt(vca[2], delimiter=",", skiprows=7, usecols=[2])

vba1 = np.loadtxt(vba[0], delimiter=",", skiprows=7, usecols=[2])
vba2 = np.loadtxt(vba[1], delimiter=",", skiprows=7, usecols=[2])
vba3 = np.loadtxt(vba[2], delimiter=",", skiprows=7, usecols=[2])

ta1 = np.loadtxt(ta[0], delimiter=",", skiprows=7, usecols=[2, 3])
ta2 = np.loadtxt(ta[1], delimiter=",", skiprows=7, usecols=[2, 3])
ta3 = np.loadtxt(ta[2], delimiter=",", skiprows=7, usecols=[2, 3])
```

```python
# Hand built thermocouple Seebeck coefficient calculations
vbh1 = np.loadtxt(vbh[0], delimiter=",", skiprows=7, usecols=[2])
vbh2 = np.loadtxt(vbh[1], delimiter=",", skiprows=7, usecols=[2])
vbh3 = np.loadtxt(vbh[2], delimiter=",", skiprows=7, usecols=[2])

vbc1 = np.loadtxt(vbc[0], delimiter=",", skiprows=7, usecols=[2])
vbc2 = np.loadtxt(vbc[1], delimiter=",", skiprows=7, usecols=[2])
vbc3 = np.loadtxt(vbc[2], delimiter=",", skiprows=7, usecols=[2])

seebeck_coeff_built = np.polyfit(
    [0, 0, 0, 100, 100, 100],
    [vbc1.mean(), vbc2.mean(), vbc3.mean(), vbh1.mean(), vbh2.mean(), vbh3.m
    deg=1,
)


def built_fit(T):
    return (T * seebeck_coeff_built[0]) + seebeck_coeff_built[1]


# Commercial thermocouple Seebeck coefficient calculations
vch1 = np.loadtxt(vch[0], delimiter=",", skiprows=7, usecols=[2])
vch2 = np.loadtxt(vch[1], delimiter=",", skiprows=7, usecols=[2])
vch3 = np.loadtxt(vch[2], delimiter=",", skiprows=7, usecols=[2])

vcc1 = np.loadtxt(vcc[0], delimiter=",", skiprows=7, usecols=[2])
vcc2 = np.loadtxt(vcc[1], delimiter=",", skiprows=7, usecols=[2])
vcc3 = np.loadtxt(vcc[2], delimiter=",", skiprows=7, usecols=[2])

seebeck_coeff_comm = np.polyfit(
    [0, 0, 0, 100, 100, 100],
    [vcc1.mean(), vcc2.mean(), vcc3.mean(), vch1.mean(), vch2.mean(), vch3.m
    deg=1,
)


def comm_fit(T):
    return (T * seebeck_coeff_comm[0]) + seebeck_coeff_comm[1]


# Create curves for plotting fit line
temp = np.linspace(0, 100, 1000)
seebeck_curve_comm = comm_fit(temp)
seebeck_curve_built = built_fit(temp)


# Find Rsquared values
def r_squared(exper, fit):
    return np.corrcoef(exper, fit)[0, 1] ** 2


exper_built = [
    vbc1.mean(),
    vbc2.mean(),
    vbc3.mean(),
    vbh1.mean(),
    vbh2.mean(),
    vbh3.mean(),
]
fit_built = built_fit(np.array([0, 0, 0, 100, 100, 100]))
rsq_built = r_squared(exper_built, fit_built)

exper_comm = [
    vcc1.mean(),
    vcc2.mean(),
    vcc3.mean(),
    vch1.mean(),
    vch2.mean(),
    vch3.mean(),
]
fit_comm = comm_fit(np.array([0, 0, 0, 100, 100, 100]))
rsq_comm = r_squared(exper_comm, fit_comm)


# Plotting
font = dict(family="serif", size=14)
title = dict(family="serif", size=14, weight="bold")
comm_markers = dict(
    color="blue", linestyle="none", marker="o", markersize=8, fillstyle="non
)
built_markers = dict(
    color="red", linestyle="none", marker="o", markersize=8, fillstyle="none
)
comm_line = dict(color="blue", linestyle="--", linewidth=2)
built_line = dict(color="red", linestyle="--", linewidth=2)

plt.errorbar(
    [100, 100, 100],
    [vbh1.mean(), vbh2.mean(), vbh3.mean()],
    yerr=[2 * vbh1.std(), 2 * vbh2.std(), 2 * vbh3.std()],
    **built_markers,
)
plt.errorbar(
    [0, 0, 0],
    [vbc1.mean(), vbc2.mean(), vbc3.mean()],
    yerr=[2 * vbc1.std(), 2 * vbc2.std(), 2 * vbc3.std()],
    **built_markers,
)
plt.plot(temp, seebeck_curve_built, label="Welded Thermocouple", **built_lin

plt.errorbar(
    [100, 100, 100],
    [vch1.mean(), vch2.mean(), vch3.mean()],
    yerr=[2 * vch1.std(), 2 * vch2.std(), 2 * vch3.std()],
    **comm_markers,
)
plt.errorbar(
    [0, 0, 0],
    [vcc1.mean(), vcc2.mean(), vcc3.mean()],
    yerr=[2 * vcc1.std(), 2 * vcc2.std(), 2 * vcc3.std()],
    **comm_markers,
)
plt.plot(temp, seebeck_curve_comm, label="Commercial Thermocouple", **comm_l

plt.errorbar(
    [ta1[:, 0].mean(), ta2[:, 0].mean(), ta3[:, 0].mean()],
    [vca1.mean(), vca2.mean(), vca3.mean()],
    yerr=[2 * vca1.std(), 2 * vca2.std(), 2 * vca3.std()],
    **comm_markers,
)
plt.errorbar(
    [ta1[:, 1].mean(), ta2[:, 1].mean(), ta3[:, 1].mean()],
    [vba1.mean(), vba2.mean(), vba3.mean()],
    yerr=[2 * vba1.std(), 2 * vba2.std(), 2 * vba3.std()],
    **built_markers,
)


plt.annotate(f"$R^2$ Commercial: {str(rsq_comm)[:5]}", (15, 1600), **font)
plt.annotate(f"$R^2$ Welded: {str(rsq_built)[:5]}", (15, 1850), **font)
plt.xlabel("Temperature $[^\circ C]$", **font)
plt.ylabel("Voltage $[\mu V]$", **font)
plt.title("Figure 1: Experimental Determination of Seebeck Coefficients", **
plt.legend(prop=font)
plt.show()

print(f"Commercial Seebeck Coefficient: {seebeck_coeff_comm[0]}")
print(f"Welded Seebeck Coefficient: {seebeck_coeff_built[0]}")

err_comm = abs(
    seebeck_coeff_comm[0]
    - (
        seebeck_coeff_comm[0]
        + (
            max([2 * vcc1.std(), 2 * vcc2.std(), 2 * vcc3.std()])
            + max([2 * vch1.std(), 2 * vch2.std(), 2 * vch3.std()])
        )
        / 100
    )
)

err_built = abs(
    seebeck_coeff_built[0]
    - (
        seebeck_coeff_built[0]
        + (
            max([2 * vbc1.std(), 2 * vbc2.std(), 2 * vbc3.std()])
            + max([2 * vbh1.std(), 2 * vbh2.std(), 2 * vbh3.std()])
        )
        / 100
    )
)

print(f"Commercial Seebeck Error: {err_comm}")
print(f"Welded Seebeck Error: {err_built}")
```

Figure 1: Experimental Determination of Seebeck Coefficients



```
Commercial Seebeck Coefficient: 40.372166657774905
Welded Seebeck Coefficient: 40.81320341151732
Commercial Seebeck Error: 0.2819934494566567
Welded Seebeck Error: 0.34090555729406447
```

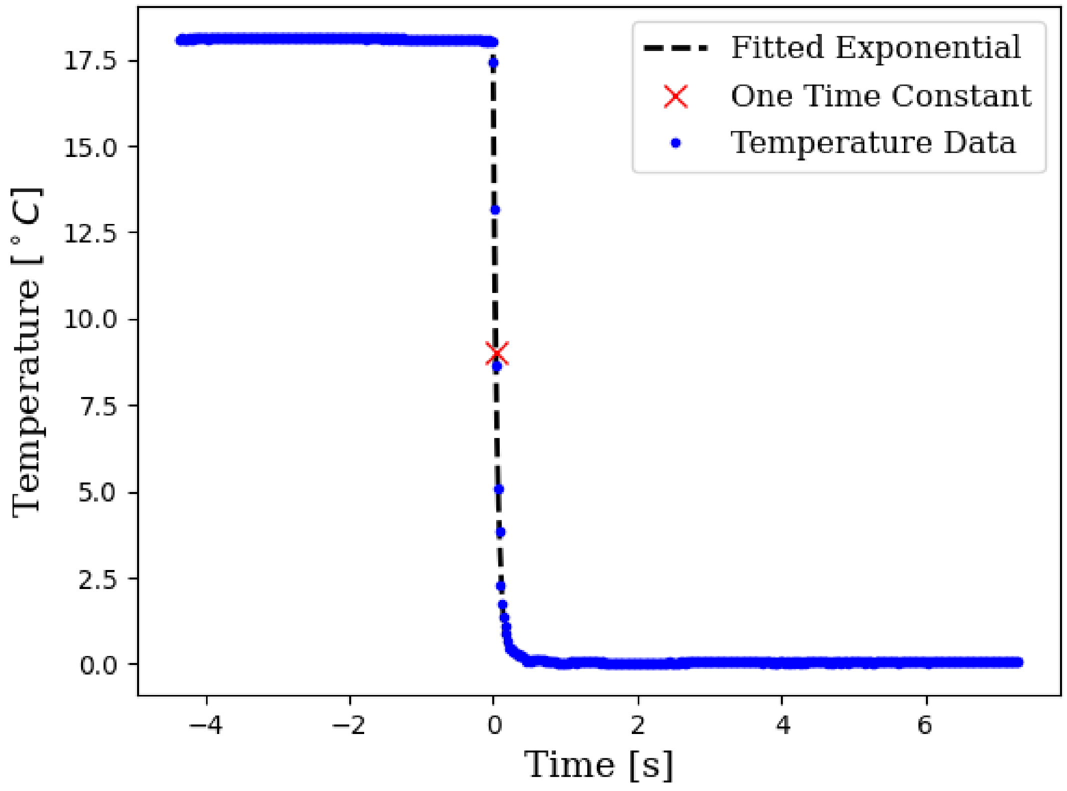
In [3]:
```
# Data paths
t1r1 = [
    "./Data/EXP1.2 - TASK1/rate_1_trial_1.csv",
    "./Data/EXP1.2 - TASK1/rate_1_trial_2.csv",
    "./Data/EXP1.2 - TASK1/rate_1_trial_3.csv",
]
t1r2 = [
    "./Data/EXP1.2 - TASK1/rate_2_trial_1.csv",
    "./Data/EXP1.2 - TASK1/rate_2_trial_2.csv",
    "./Data/EXP1.2 - TASK1/rate_2_trial_3.csv",
]
t1r3 = [
    "./Data/EXP1.2 - TASK1/rate_3_trial_1.csv",
    "./Data/EXP1.2 - TASK1/rate_3_trial_2.csv",
    "./Data/EXP1.2 - TASK1/rate_3_trial_3.csv",
]

# Load data
drop1 = np.loadtxt(t1r2[0], delimiter=",", skiprows=7, usecols=[1, 2])
drop2 = np.loadtxt(t1r2[1], delimiter=",", skiprows=7, usecols=[1, 2])
drop3 = np.loadtxt(t1r2[2], delimiter=",", skiprows=7, usecols=[1, 2])

# Theoretical calculations
t_value = 3.182
bead_Derr = (np.array([1.3, 1.44, 1.44]) * (1/1000)).std() * (t_value / np.s
print(f"bead error {bead_Derr}")
bead_D = 1.3933 * (1 / 1000)
```

```
print(f"bead D {2 * bead_D}")

# Values for K cited from TABLE 1 of:
# Sundqvist, B. (1992).
# Thermal diffusivity and thermal conductivity of Chromel, Alumel, and Const
# Journal of Applied Physics, 72(2), 539–545.
# doi:10.1063/1.351885
K_chromel = 4.91 * 10**-6
K_alumel = 7.25 * 10**-6
K_avg = (K_chromel + K_alumel) / 2
tau_theory = (bead_D**2) / (K_avg * np.pi**2)

# Getting initial temp
T0_1 = drop1[:100, 1].mean()
T0_2 = drop2[:100, 1].mean()
T0_3 = drop3[:100, 1].mean()
print(f"T0_1 {T0_1}")


def T(t, T0, tau):
    return T0 * np.exp(-t / tau)


def fit_exp(t, y):
    y = np.log(y)
    K, A_log = np.polyfit(t, y, 1)
    A = np.exp(A_log)
    return A, K


def find_nearest(data, value):
    idx = (np.abs(data - value)).argmin()
    return idx


# Start and end of each temp drop trial
start_idx1 = np.diff(drop1[:, 1]).argmin() - 1
end_idx1 = start_idx1 + 10

start_idx2 = np.diff(drop2[:, 1]).argmin() - 1
end_idx2 = start_idx2 + 10

start_idx3 = np.diff(drop3[:, 1]).argmin() - 1
end_idx3 = start_idx3 + 10

t_exp1 = drop1[start_idx1:end_idx1, 0] - drop1[start_idx1, 0]
T_exp1 = drop1[start_idx1:end_idx1, 1]

t_exp2 = drop2[start_idx2:end_idx2, 0] - drop2[start_idx2, 0]
T_exp2 = drop2[start_idx2:end_idx2, 1]

t_exp3 = drop3[start_idx3:end_idx3, 0] - drop3[start_idx3, 0]
T_exp3 = drop3[start_idx3:end_idx3, 1]

# Exponential data fit for each trial
time = np.linspace(-0.005, 0.2, 1000)
```

```
A1, K1 = fit_exp(t_exp1, T_exp1)
A2, K2 = fit_exp(t_exp2, T_exp2)
A3, K3 = fit_exp(t_exp3, T_exp3)
drop1_fit = A1 * np.exp(K1 * time)
drop2_fit = A2 * np.exp(K2 * time)
drop3_fit = A3 * np.exp(K3 * time)

# Plot data, fitted curve and theoretical curve
font = dict(family="serif", size=14)
title = dict(family="serif", size=14, weight="bold")
line = dict(color="black", linestyle="--", linewidth=2)


tau_fit1 = time[find_nearest(drop1_fit, T0_1 / 2)] / np.log(2)
tau_fit2 = time[find_nearest(drop2_fit, T0_2 / 2)] / np.log(2)
tau_fit3 = time[find_nearest(drop3_fit, T0_3 / 2)] / np.log(2)
t_value = 3.182
tau_avg = (tau_fit1 + tau_fit2 + tau_fit3) / 3
tau_error = np.array([tau_fit1, tau_fit2, tau_fit3]).std() * (t_value / np.s

plt.plot(time, drop1_fit, label="Fitted Exponential", **line)
plt.plot(
    tau_fit1 * np.log(2),
    drop1_fit[find_nearest(drop1_fit, T0_1 / 2)],
    "rx",
    label="One Time Constant",
    markersize=8
)
plt.plot(
    drop1[:, 0] - drop1[start_idx1, 0], drop1[:, 1], "b.", label="Temperatur
)

plt.xlabel("Time [s]", **font)
plt.ylabel("Temperature $[^\circ C]$", **font)
plt.title("Figure 2: Experimental Determination of Time Constant", **title)
plt.legend(prop={"family": "serif", "size": 12})
plt.show()


print(tau_avg)
print(tau_error)
print(tau_theory)
```

```
bead error 0.00012124429895722756
bead D 0.0027866
T0_1 18.10729
```

Figure 2: Experimental Determination of Time Constant



```
0.04943047723512955
0.008534264192430694
0.032350868888451574
```

In [4]:
```
# Load data
bp1 = np.loadtxt(
    "./Data/EXP1.2 - TASK2/hotT_trial1.csv", delimiter=",", skiprows=7, usec
)
bp2 = np.loadtxt(
    "./Data/EXP1.2 - TASK2/hotT_trial2.csv", delimiter=",", skiprows=7, usec
)
bp3 = np.loadtxt(
    "./Data/EXP1.2 - TASK2/hotT_trial3.csv", delimiter=",", skiprows=7, usec
)

bp = np.array([bp1.mean(), bp2.mean(), bp3.mean()])
bp_mean = bp.mean()
bp_err = bp.std() * (t_value / np.sqrt(3))

# Value for Kb cited from TABLE 1 of:
# Hoyt, C. S., & Fink, C. K. (1937).
# The Constants of Ebullioscopy.
# The Journal of Physical Chemistry, 41(3), 453–456.
# doi:10.1021/j150381a010
Kb = 0.51
i = 2
delta_T = bp_mean - 100
m = delta_T / (i * Kb)
print(f"NaCl Concentration [mol salt / kg solvent]: {m}")
print(f"err {bp_err}")
```

```
max_error = np.array([bp1.std(), bp2.std(), bp3.std()]).max()
print(max_error)
t_value = 3.182
ci = max_error * (t_value / np.sqrt(3))
print(ci)
```

```
NaCl Concentration [mol salt / kg solvent]: 6.256602788671128
err 0.1145210424453708
0.0818296449441906
0.15033157750025186
```

In [5]:

```
# Data paths
t1r1 = [
    "./Data/EXP1.2 - TASK1/rate_1_trial_1.csv",
    "./Data/EXP1.2 - TASK1/rate_1_trial_2.csv",
    "./Data/EXP1.2 - TASK1/rate_1_trial_3.csv",
]
t1r2 = [
    "./Data/EXP1.2 - TASK1/rate_2_trial_1.csv",
    "./Data/EXP1.2 - TASK1/rate_2_trial_2.csv",
    "./Data/EXP1.2 - TASK1/rate_2_trial_3.csv",
]
t1r3 = [
    "./Data/EXP1.2 - TASK1/rate_3_trial_1.csv",
    "./Data/EXP1.2 - TASK1/rate_3_trial_2.csv",
    "./Data/EXP1.2 - TASK1/rate_3_trial_3.csv",
]

# Load data
rate1 = [
r1t1 := np.loadtxt(t1r1[0], delimiter=",", skiprows=7, usecols=[2]),
r1t2 := np.loadtxt(t1r1[1], delimiter=",", skiprows=7, usecols=[2]),
r1t3 := np.loadtxt(t1r1[2], delimiter=",", skiprows=7, usecols=[2])
]

rate2 = [
r2t1 := np.loadtxt(t1r2[0], delimiter=",", skiprows=7, usecols=[2]),
r2t2 := np.loadtxt(t1r2[1], delimiter=",", skiprows=7, usecols=[2]),
r2t3 := np.loadtxt(t1r2[2], delimiter=",", skiprows=7, usecols=[2])
]

rate3 = [
r3t1 := np.loadtxt(t1r3[0], delimiter=",", skiprows=7, usecols=[2]),
r3t2 := np.loadtxt(t1r3[1], delimiter=",", skiprows=7, usecols=[2]),
r3t3 := np.loadtxt(t1r3[2], delimiter=",", skiprows=7, usecols=[2])
]

avg1 = np.array([r1t1[125:].mean(), r1t2[125:].mean(), r1t3[125:].mean()]).m
dev1 = 2 * np.array([r1t1[125:].mean(), r1t2[125:].mean(), r1t3[125:].mean()

avg2 = np.array([r2t1[400:].mean(), r2t2[400:].mean(), r2t3[400:].mean()]).m
dev2 = 2 * np.array([r2t1[400:].mean(), r2t2[400:].mean(), r2t3[400:].mean()

avg3 = np.array([r3t1[10:].mean(), r3t2[10:].mean(), r3t3[10:].mean()]).mean
dev3 = 2 * np.array([r3t1[10:].mean(), r3t2[10:].mean(), r3t3[10:].mean()]).
```

```
print(f"rate 1:{avg1} C +- {dev1}")
print(f"rate 2:{avg2} C +- {dev2}")
print(f"rate 3:{avg3} C +- {dev3}")
```

```
rate 1:1.0534422695659929 C +- 1.6074496386155115
rate 2:-0.07021248628150305 C +- 0.18666271555146743
rate 3:-0.0451984126984127 C +- 0.09433676935268795
```