Hardware Project: PT-100 Temperature Sensor Analysis

Divyansh (EE25BTECH11037)   Shriyansh (EE25BTECH11052)

November 1, 2025

Abstract

This report details a project aimed at measuring and analyzing the voltage output of a PT-100 platinum resistance temperature detector (RTD) at varying temperatures. The project involves setting up a circuit with an Arduino, collecting data, and fitting it to a theoretical model using the least squares method. The experiment successfully demonstrates the functionality of the PT-100 sensor for precise temperature measurement and verifies its linear relationship between temperature and resistance (and consequently voltage).

1 Aim

To measure and analyze the voltage output of a PT-100 sensor at varying temperatures.

2 Components Required

1. Breadboard
2. Arduino UNO
3. PT-100 Sensor
4. Resistor (300 Ω)
5. Jumper Cables
6. Electric Kettle
7. Thermometer

3 Introduction to PT-100 Sensor

A PT-100 is a platinum resistance temperature detector (RTD). The resistance of the platinum element changes with temperature. The "PT" stands for platinum, and "100" indicates that it has a resistance of 100 Ω at 0°C.

4 Procedure

4.1 Connections

The circuit components were arranged as shown in the diagram below. The PT-100 sensor and a 300 Ω resistor were set up in a voltage divider configuration, with the Arduino's analog input pin (A0) used to measure the voltage across the PT-100.

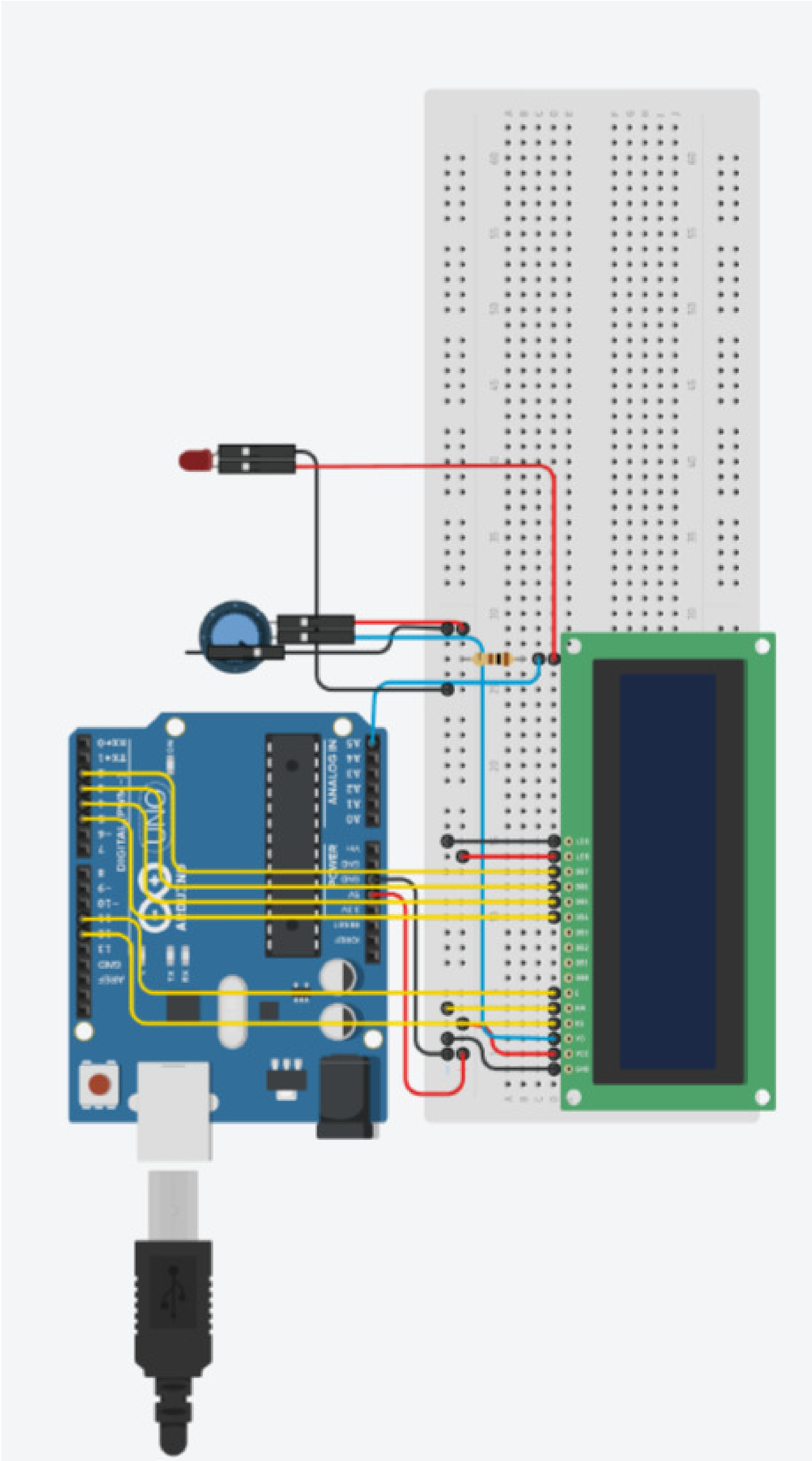

Figure 1: Circuit Diagram and Breadboard Setup.

An electric kettle was used to heat water, thereby increasing the temperature of the PT-100 sensor immersed in it. The corresponding voltage values were measured using the Arduino setup.

4.2 Code

The following C++ code was uploaded to the Arduino UNO to read the sensor values.

Listing 1: Arduino Code for Reading Sensor Voltage

```
#include "Arduino.h"

void setup() {
        Serial.begin(9600);
}

void loop() {
        int sensorValue = analogRead(A0);
        double voltage = (5.0 * sensorValue) / 1023.0;
        Serial.println(voltage);
        delay(1000);
}
```

Code Explanation

- **setup():** The `Serial.begin(9600)` function initializes serial communication at a baud rate of 9600 bits per second (bps). This allows the Arduino to send data to the serial monitor.
- **loop():** This function runs continuously.
  - `analogRead(A0)` reads the voltage from the analog pin A0, where the sensor is connected. The analog-to-digital converter (ADC) returns a value between 0 and 1023.
  - This raw sensor value is converted to an approximate voltage using the formula:

    $$\text{Voltage} = \frac{5.0 \times \text{sensorValue}}{1023.0}$$

    This maps the ADC's 0-1023 range to the Arduino's 0-5V range.
  - `Serial.println(voltage)` sends the calculated voltage to the serial monitor.
  - `delay(1000)` pauses the loop for 1000 milliseconds (1 second) before the next reading.

5 Theory

The relationship between temperature and the resistance of a platinum sensor can be described by the Callendar-Van Dusen equation. For this project, we model the output voltage $V(T)$ as a function of temperature T (in °C) using a simplified polynomial form:

$$V(T) = V_0\left(1 + AT + BT^2\right)$$

This can be rewritten in a linear matrix form suitable for the least squares method.

$$C = n^T x$$

where:

$$C = V(T), \quad x = \begin{pmatrix} 1 \\ T \\ T^2 \end{pmatrix}, \quad n = \begin{pmatrix} V(0) \\ A \cdot V(0) \\ B \cdot V(0) \end{pmatrix}$$

For multiple data points $(T_i, V(T_i))$, the system of equations is given by:

$$Xn = c$$

where:

$$X = \begin{pmatrix} 1 & T_1 & T_1^2 \\ 1 & T_2 & T_2^2 \\ \vdots & \vdots & \vdots \\ 1 & T_n & T_n^2 \end{pmatrix}, \quad c = \begin{pmatrix} V(T_1) \\ V(T_2) \\ \vdots \\ V(T_n) \end{pmatrix}$$

5.1 The Theory of Least Squares

The matrix equation $Xn = c$ is an overdetermined system because the number of data points is typically greater than the number of unknown coefficients. An exact solution does not exist. The goal of the least squares method is to find the vector n that minimizes the sum of the squared errors between the model's prediction and the actual measured voltages.

The error vector e is defined as:

$$e = c - Xn$$

We want to minimize the squared Euclidean norm of this error, $\|e\|^2$. This leads to the normal equations:

$$X^T X n = X^T c$$

Assuming that $X^T X$ is invertible, we can solve for n:

$$n = (X^T X)^{-1} X^T c$$

This is the fundamental equation of the least squares method, which provides the best-fit coefficients.

From our experimental data and applying the least squares method, we get the coefficient vector n:

$$n = \begin{pmatrix} 1.2482 \\ 4.1397 \times 10^{-3} \\ -7.0833 \times 10^{-6} \end{pmatrix}$$

6 Results

6.1 Training Data

Table 1: Training Data: Measured Voltage at Different Temperatures.

Temp (°C)	**Voltage (V)**	**Temp (°C)**	**Voltage (V)**	**Temp (°C)**	**Voltage (V)**
27.2	1.359	51.0	1.442	79.0	1.530
39.0	1.398	54.3	1.461	80.9	1.535
40.0	1.403	58.0	1.466	83.0	1.554
36.5	1.388	68.8	1.496	89.6	1.564
33.0	1.378	69.3	1.500	94.9	1.579
32.0	1.373	71.1	1.505	96.9	1.584
39.5	1.395	72.5	1.510	93.8	1.574
43.8	1.413	76.0	1.520	93.0	1.569
44.0	1.417	77.0	1.525	92.0	1.564

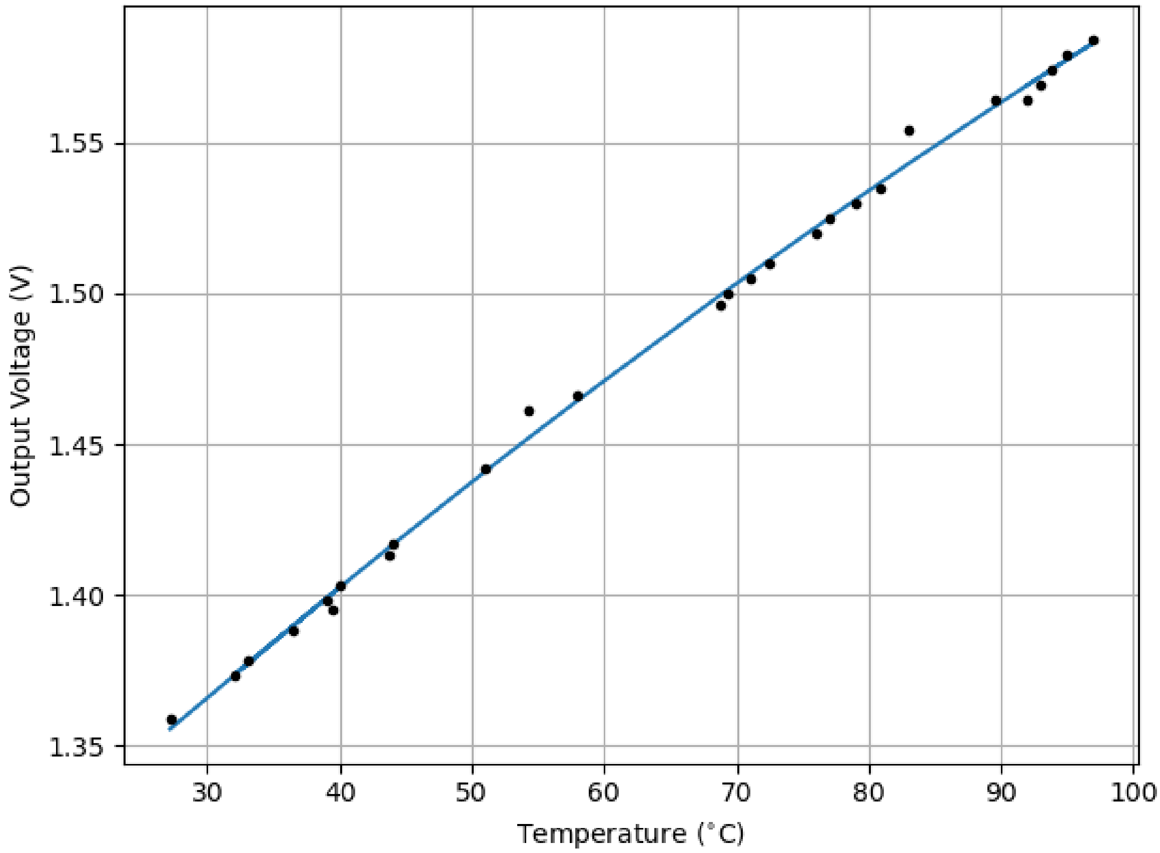

Figure 2: Output Voltage vs. Temperature for Training Data.

Table 2: Validation Data.

Temp (°C)	**Voltage (V)**
27.2	1.359
36.5	1.388
39.5	1.395
51.0	1.442
68.8	1.496
72.5	1.510
79.0	1.530
89.6	1.564
93.8	1.574

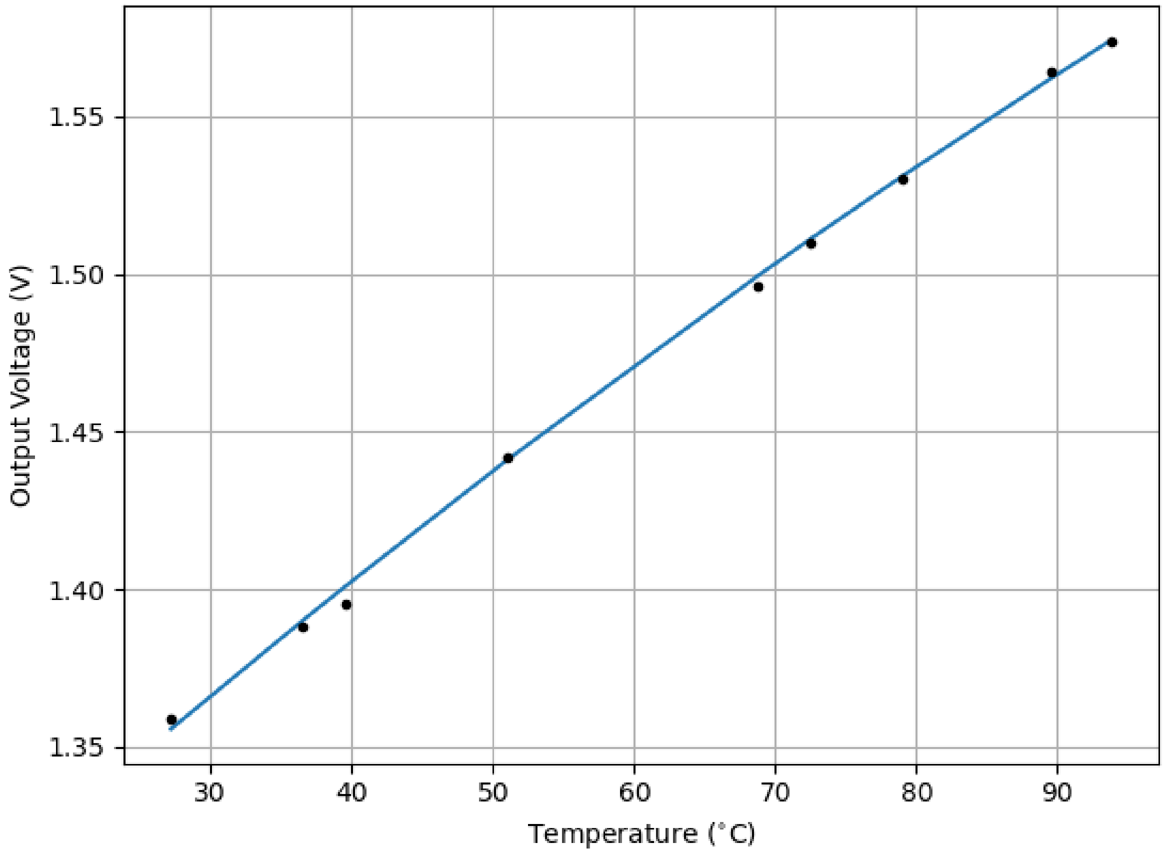

Figure 3: Validation Data

7 Model Concavity and Optimization Choice

An important observation is that the coefficient of the T^2 term $(B \cdot V(0))$ is negative (-7.0833×10^{-6}). This means the governing function $V(T)$ is strictly concave. This has a significant implication from a machine learning perspective.

Many optimization problems in machine learning are solved using gradient descent, an iterative algorithm that finds the minimum of a function. Gradient descent works by taking steps in the direction of the negative gradient of the function. This process is guaranteed to find the global minimum only if the function is convex.

Since our function is concave, it has a maximum, not a minimum. Applying gradient descent here would fail to find the correct parameters because the algorithm is designed to minimize a cost function, not maximize it. Therefore, the analytical least squares method, which directly solves for the optimal parameters via the normal equations, is the appropriate and more robust choice for this linear regression problem.

8 Conclusion

In conclusion, this report successfully demonstrates the use of the PT-100 sensor with an Arduino for temperature measurement. The project involved setting up a circuit, coding, and data collection to observe the sensor's response across varying temperatures. The experiment verified the PT-100's effectiveness in providing reliable temperature readings, making it a valuable component for precise temperature monitoring in various applications.