

ML-Driven Self Adaptive Differential amplifier

Team: [g_esw79](#) Team Id : 39

Abstract:

The project focuses on the development of a self-adaptive differential amplifier circuit that can mitigate the effects of environmental variations, such as changes in voltage and temperature, on circuit performance. The core of the circuit is a BJT-based differential amplifier, designed to amplify the voltage difference between two input signals while rejecting common-mode noise, making it highly suitable for applications requiring signal clarity.

Introduction:

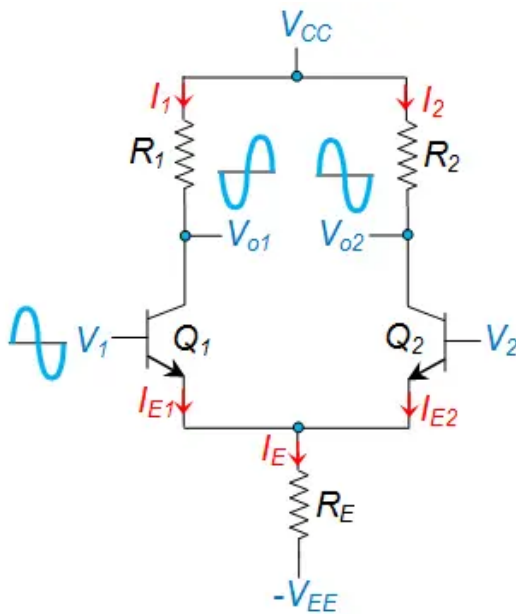


Figure 1 A BJT Differential Amplifier

A BJT Differential Amplifier is a widely used electronic circuit that amplifies the difference between two input signals. This type of amplifier, often referred to simply as a Differential Amplifier (DA), is essential in applications requiring precise signal amplification and noise rejection.

The circuit consists of two bipolar junction transistors (BJTs) connected in a differential configuration. When input signals are applied to the bases of these transistors, the amplifier operates in the active region, allowing current to flow through the circuit. The difference in input voltages generates a corresponding difference in currents,

which is then converted into an amplified output voltage.

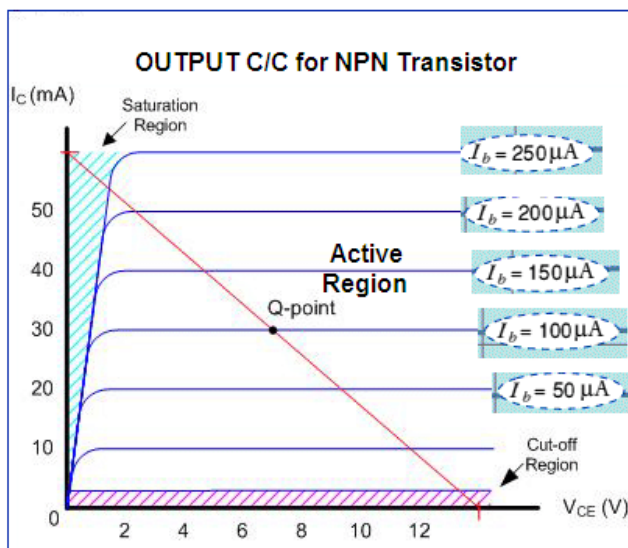
Applications of BJT Differential Amplifier

A key advantage of the differential amplifier is its ability to eliminate common-mode noise. Noise signals, which tend to affect both inputs equally, are effectively canceled out by the differential mechanism, leaving only the desired signal to be amplified. This feature makes the BJT Differential Amplifier particularly suitable for applications requiring clean and accurate signal processing.

Additionally, the high gain provided by the amplifier enables the detection and amplification of weak signals, further expanding its utility in fields like instrumentation, communication systems, and analog circuit design.

They are commonly found in audio systems, instrumentation, and communication devices, where they amplify weak signals while minimizing interference. Additionally, they are used in biomedical devices like ECG machines and in noise-canceling headphones for clear, precise signal processing.

Characteristic graph



The **Q-point** (Quiescent Point) is the point on the curve that represents the steady-state operating condition of the transistor, where both V_{CE} and I_C are stable.

Use of this property in our circuit:

Linear Operation: Ensures the amplifier remains in the active region for faithful amplification.

Biasing: Properly sets the Q-point in the active region for efficient operation. Base voltage $> 0.7V$ for active region .

Current Gain: The base current (I_b) controls the collector current (I_C), which determines the output voltage.

Common-Mode Rejection: Helps to reject unwanted signals and amplify the difference between the inputs

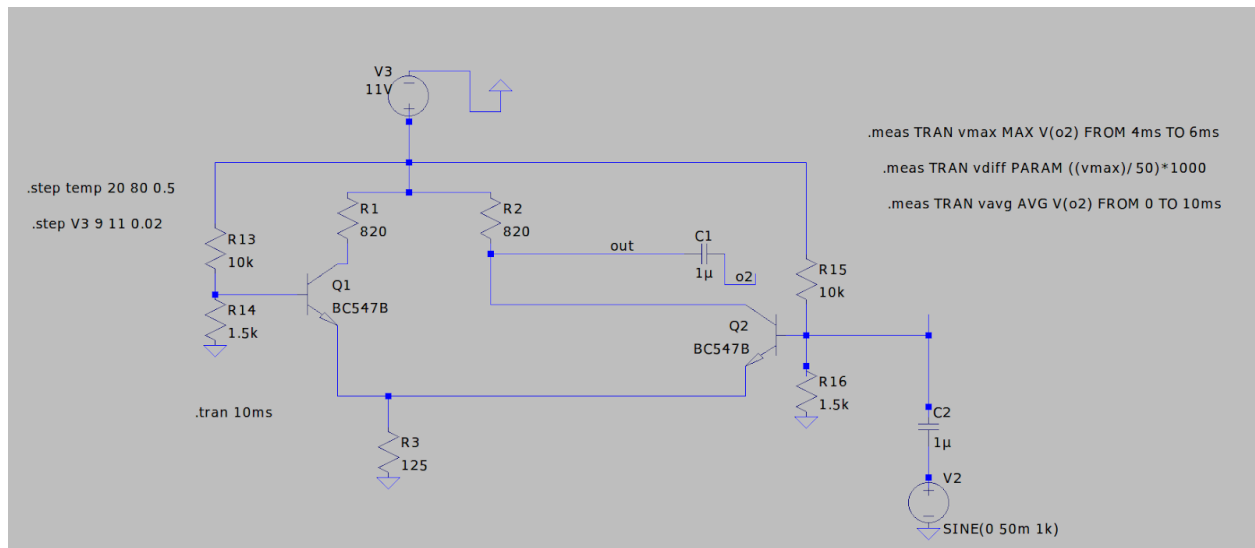
Existing Model:

Traditional differential amplifier circuits are designed to amplify the voltage difference between two input signals while minimizing common noise. However, these circuits are sensitive to changes in environmental conditions such as voltage fluctuations and temperature variations. These factors can significantly impact the performance and reliability of the differential amplifier, leading to inaccuracies and potential failure in critical applications.

Proposed Solution:

We implemented a machine learning-driven self-adaptive differential amplifier using a linear regression model on an Arduino board. The model predicts the optimal resistor values required to maintain the desired gain under varying temperature and voltage conditions. Based on these predictions, the resistor values are manually adjusted to ensure stable circuit performance.

Circuit Overview:



This is a BJT-based **Differential Amplifier** using two NPN transistors (**Q1** and **Q2**), designed to amplify the voltage difference between two input signals while rejecting any common-mode signals (e.g., noise). The circuit also includes various resistors and capacitors to establish biasing, load, and coupling functionalities. It is a single ended differential amplifier.

Component Details and Their Roles

1. **Q1 and Q2 (BC547B Transistors):**

These form the core of the differential amplifier. Input signals are applied to the bases of these transistors, which control the current flow through their collector-emitter paths.

2. **R3 (Emitter Resistor):**

The common emitter resistor helps establish a stable operating point and provides negative feedback, improving the circuit's linearity and stability.

3. **R13, R14, R15, R16 (Biasing Resistors):**

These resistors set up the base-emitter voltage and ensure proper biasing of the transistors to operate in the active region.

4. **R1 and R2 (Collector Resistors):**

These resistors are connected to the collectors of Q1 and Q2. The voltage drop across them depends on the collector current, which varies with the input signal, generating the amplified output signal.

5. **V2 (Input Signal Source): SINE wave , Amplitude 50mV ,Frequency 1KHz**

This represents one of the differential input signals (a sinusoidal input), while the other input is often grounded or given another signal (common-mode rejection can be observed here).

6. **C1 and C2 (Coupling Capacitors):**

These block DC components while allowing AC signals to pass. They couple the output to the load or the next stage.

7. **V3 (DC Power Supply): varies between 9V to 11V**

This provides the required voltage for the circuit's operation.

Working of the Circuit

1. **Differential Input:**

- Input signals are applied to the bases of Q1 and Q2.
- If both inputs are identical (common-mode), the currents through Q1 and Q2 are equal, and the difference is zero, resulting in no output signal (ideal noise rejection).
- If there's a difference between the input signals, this voltage difference is amplified.

2. Amplification:

- The input voltage difference causes a corresponding change in the base-emitter voltages of Q1 and Q2, altering their collector currents.
- These variations create a voltage drop across R1 and R2, which forms the output signal.

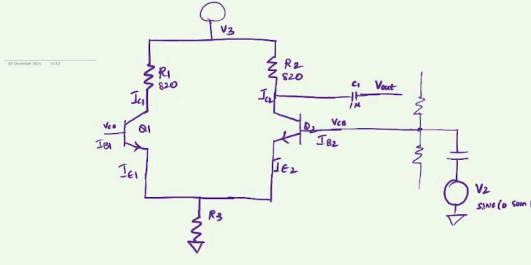
3. Output :

- The amplified difference signal appears at the collectors of Q1 and Q2, with the output taken through the coupling capacitors (C1 and C2) for further processing.

Hardware Components:

- 1. A pair of KTN2222A Differential Amplifiers:** These are NPN bipolar junction transistors used to construct the differential amplifier. They amplify the voltage difference between the two input signals.
- 2. Arduino Uno:** A microcontroller board that will be used to implement the machine learning model. It processes the input signals, environmental data, and predicts adaptive changes to the circuit.
- 3. Temperature Sensor (DHT 11):** Measures the ambient temperature. The data is used by the Arduino to adjust the differential amplifier's performance to compensate for temperature variations.
- 4. Voltage Sensor:** Monitors the voltage levels within the circuit. The readings help the Arduino make necessary adjustments to maintain consistent circuit performance. We Used it to Measure biasing DC voltage .
- 5. Lab Equipment:**
 - Waveform Generator: Produces various input signals to test the differential amplifier's response.
 - DC Source: Provides the necessary power supply for the circuit.
 - Oscilloscope: Visualizes the input and output waveforms, helping in analyzing the circuit's performance.
- 6. Resistors , Jumper Wires and Breadboard**

Gain Calculation :



DC Analysis :-

$$I_C = I_S \exp\left(\frac{V_{BE}}{V_T}\right)$$

$I_C \rightarrow$ collector current
 $I_S \rightarrow$ reverse saturation current
 $V_{BE} \rightarrow$ Base emitter voltage
 $V_T \rightarrow$ Thermal voltage
 $\alpha = \frac{I_C}{I_E}$; $\beta = \frac{I_C}{I_B}$; $\alpha = \frac{\beta}{\beta+1}$
 $\alpha \rightarrow$ current gain (common base gain)
 $\beta \rightarrow$ current gain (common emitter gain)
 Voltage divider (voltage to base)
 $V_B = V_3 \cdot \frac{R_{16}}{R_{15} + R_{16}}$
 $I_{E1} = I_{E2} = I_{Q1}$ (say)
 $I_{B1} = I_{B2} = \frac{I_{Q1}}{1+\beta}$; $I_{C1} = I_{C2} = \alpha I_{Q1}$
 to work in active mode let $R_1 = R_2 = R_C$; $R_E = R_3$
 $V_{CB} = V_C - V_B = (V_3 \cdot \alpha I_E R_C) - \left(\frac{R_{16}}{R_{15} + R_{16}}\right) V_3$
 $V_{CE} = V_C - V_E = V_C - (V_B - V_{BE}) = V_{CB} + V_{BE}$

AC Analysis :-

no V_1 from left side. $\therefore V_1 = 0$

at $r_{\pi} = \frac{V_T}{I_B}$ $r_{ie} \rightarrow$ internal emitter resistance (npn)

Ohm's law $r_x \rightarrow$ base spread resistance

Since $I_1 = I_{E2} = \frac{V_1 \cdot V_2}{2(R_3 + r_{ie})}$

$R_B = \frac{V_B}{I_B}$ $r_{ie} = \frac{R_B + r_{\pi} + r_x}{1+\beta}$

$I_{E2} \approx \frac{-(-V_2)}{2(R_E + r_{ie})} = \frac{V_2}{2(R_E + r_{ie})}$

$V_{out} = -\alpha I_{E1} R_C = \frac{-\alpha R_C}{2(R_{ie} + R_3)} V_2$

gain = $\frac{V_{out}}{V_2} = \frac{-\alpha R_C}{2(R_{ie} + R_3)} \cdot \frac{V_2}{V_2}$

$= \frac{-\alpha R_C}{2(R_{ie} + R_3)}$

$\approx \frac{-\alpha R_C}{2(R_E + \frac{V_B/I_B + V_T/I_B + r_x}{1+\beta})}$

$g_{m2} = 2 R_3 + r_{ie} + \frac{V_B}{I_B} - \frac{V_T}{I_B}$

- The gain is directly proportional to R_C (R_2) and Inversely proportional to R_E (R_3)

Software Requirements:

- LTspice:** For circuit simulation and generating data points for machine learning analysis.
- Arduino IDE:** To implement the machine learning-generated formula and control the circuit on the Arduino board.
- Python:** For training the **linear regression model**.

Interfacing Arduino with the hardware circuit

To achieve seamless integration between the Arduino microcontroller and the hardware circuit, we followed these steps:

1. Sensor Connections:

- The **DHT11 sensor** was connected to measure the ambient temperature.
- A **voltage sensor** was interfaced with the Arduino to measure the DC input voltage of the circuit.

2. Arduino Code Implementation:

- The Arduino code was designed to read temperature and voltage values from the connected sensors.
- The machine learning model's predicted formula for calculating the appropriate resistor values to achieve the desired gain was implemented in the code.

3. ML Integration:

- By embedding the ML formula directly into the Arduino program, the microcontroller was able to dynamically compute the expected gain based on the sensor inputs.

This approach allowed us to implement the machine learning model on the Arduino and adapt the circuit's behavior to varying temperature and voltage conditions effectively.

Measuring DC Voltage Using Voltage Divider

Voltage Divider Setup:

- **Vin**: Actual input DC voltage.
- **Vout**: Scaled-down voltage read by the sensor.
- **R1, R2**: Resistor values in the divider.

Why Use a Voltage Divider?

- ADCs (e.g., Arduino) have a limited voltage range (e.g., 0–1V for the sensor).
- Voltage dividers scale down higher voltages to protect the ADC and ensure accurate readings.

Steps to Measure DC Voltage:

1. **Read Vout:** Use `analogRead(dcSensorPin)` to get an ADC value (0–1023).

- Formula:

$$V_{out} = dcReading \times 0.029$$

- Where **0.029** is the scaling factor (1V/1024).

2. **Calculate Vin:**

$$V_{in} = V_{out} \times (R1 + R2) / R2$$

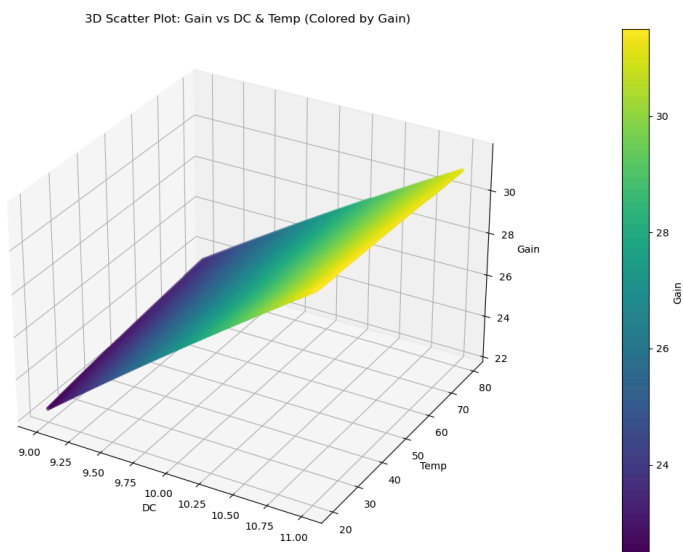
- **R1 = 9400Ω** and **R2 = 390Ω**.

Machine Learning Procedure:

1. **Simulation in LTSpice:**

- Used **LTSpice** to perform a simulation by sweeping the **DC input voltage (V3)** from **9V to 11V** with a **0.02V increment**.
- For each DC voltage value, did a **temperature sweep from 20 to 80** with a **0.5 increment**.
- This resulted in 12,221 data points that captured how the output changes with respect to both the DC voltage and the temperature.

1st Graph:



This graph provides a detailed visualization of the raw data points (**Gain**, **DC**, **Temp**) in a **scatter plot format** with a color gradient based on **Gain**. (12,221 gain values generated)

2. Linear Regression:

- After gathering the simulation data, you performed **linear regression** on the **two variables** (voltage and temperature) using **12,221 data points** to create a formula (equation) that models the relationship between the output Gain and DC, Temperature in the amplifier circuit.
- The equation likely captures how the **gain** of the amplifier varies with changes in **input DC voltage** and **temperature** .
- Equations:

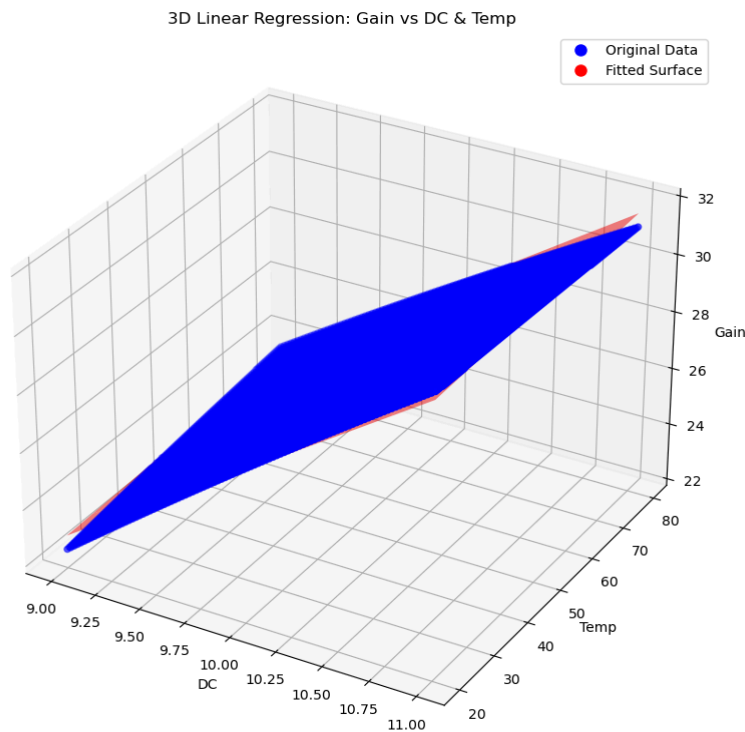
Slope for DC: 4.162296541632576

Slope for Temp: 0.00577631282353265

Intercept: -14.669442083605134

- **Linear Model: Gain = 4.1623 * DC + 0.0058 * Temp + (-14.6694) .**

2nd Graph:



Purpose:

- This graph visualizes the relationship between **Gain**, **DC**, and **Temp** using a linear regression model.
- It shows how well the linear regression plane (fitted surface) aligns with the actual data points.

Key Features:

- Blue Points: Represent the original data points derived from the **Gain**, **DC**, and **Temp** values.
- Red Surface: Represents the plane fitted by the linear regression model.
- The *red plane* provides a visual representation of the predicted values of **Gain** based on **DC** and **Temp**.

Insights:

The proximity of the blue points to the red plane shows how well the model captures the variation in the **Gain** values. This graph is useful for assessing the linear relationship between the variables.

Linear regression is a statistical method used to model the relationship between one dependent variable (in this case, Gain) and one or more independent variables (DC and Temperature in this case) using data points. The goal is to find a linear equation that best describes how the dependent variable changes with the independent variables.

Model Equation:

The regression model is:

$$\text{Gain} = \beta_0 + \beta_1 \cdot \text{DC} + \beta_2 \cdot \text{Temp}$$

- **Gain**: Target variable (dependent).
- **DC, Temp**: Predictor variables (independent).
- β_0 : Intercept.
- β_1, β_2 : Slopes of input DC and Temperature, respectively.

3. Implementation on Arduino:

- You then used an **Arduino** along with a **voltage and temperature sensor** to measure the input conditions (voltage and temperature).
- Using the formula derived from the regression analysis, you applied it on the Arduino to calculate the new **gain** in real-time, based on the measured voltage and temperature.

4. Calculating the New Resistance:

- Since **gain** is directly proportional to **resistance**, you calculated the **new resistance** required to maintain the same original gain as it was 27.1 gain at **27 degree C** with **820Ω** at input DC **10V**.
- By doing this, you ensured that the circuit would operate with the same gain under new temperature and voltage conditions. This is self adaptation.

How Gain is Affected by Resistance:

In a differential amplifier, the **gain** is directly proportional to the **resistance** in the collector load.

Temperature and Voltage Considerations:

- Temperature affects **transistor parameters** (such as gain).
- The **voltage sensor** helps in reading the changes in input voltage, which affects the transistor's behavior. The **Temperature sensor measures** the temperature of the room.
- The equation you obtained from **linear regression** allows you to adjust the resistance to maintain the desired gain at different operating conditions by calculating new gain & new required R_c .

Final Step:

- After calculating the new resistance needed to maintain the same gain, the **Arduino** would print out this value of resistance, allowing you to physically adjust the resistor in your circuit or use a digital potentiometer to achieve the desired gain at 27.1°C & 10V input.

You successfully applied a combination of **simulation (LTSpice)**, **linear regression**, and **Arduino-based implementation** to adapt the circuit's gain based on varying input conditions, specifically voltage and temperature. This allows you to maintain a consistent gain in the amplifier, despite changes in environmental factors.

Challenges faced :

1. One of the primary challenges was dealing with fluctuations in the input voltage, which can significantly impact the performance of the differential amplifier.
2. Voltage sensor calibration for AC voltage measurement .The voltage sensor was only capable of measuring DC input, making it unsuitable for capturing AC voltage or amplitude values required for the project.
3. Generating 12,000 samples in LTSpice & giving it to python for Linear Regression was extremely time-consuming and computationally expensive.
4. Differential amplifiers are sensitive to noise and interference, which can affect the quality of the data used to train the ML model.

Observations:

Temperature Measured : 29°C

DC Voltage <i>Hardware</i>	Gain <i>Hardware</i>	DC Voltage <i>Measured</i>	Gain <i>Calculated</i>	Predicted Resistor value
10.00V	27.7	9.58V	25.4	875.78Ω
10.60V	29.7	10.32	28.44	781.34Ω
10.90V	30.6	11.05	31.5	705.28Ω

Temperature Measured : 40°C

DC Voltage <i>Hardware</i>	Gain <i>Hardware</i>	DC Voltage <i>Measured</i>	Gain <i>Calculated</i>	Predicted Resistor value
10.05V	28.5	9.58V	28.5	779.53Ω
10.34V	29.3	10.32	28.55	778.62Ω
10.86V	31.0	11.05	31.58	702.3Ω

Note : *Voltage sensors give little errors even after calibration due to their inability to measure high voltages.*

Results :

Amplifier Performance:

- Measured gain aligns closely with the ML-predicted gain under varying temperature and voltage conditions.
- Effective noise rejection observed, with input and output signals matching expectations.

Machine Learning Model:

- Model coefficients (β_0 , β_1 , β_2) successfully calculated using linear regression.
- High accuracy with minimal error in gain prediction.

Arduino Integration:

- Successfully measured temperature (DHT11) and DC voltage using Arduino.
- ML formula implemented on Arduino for gain calculation.

Please find all the resources, such as codes used for the project and the self adaptation videos and images in the below link:

[ML-Driven Self Adaptive Differential amplifier](#)

Contribution :

1. Kashik P S [202310182] : Machine Learning part , Generation of Data Points in LTspice .
2. Sravani [2023101013] : Differential Amplifier Hardware Circuit , LTspice circuit simulation and Hardware Testing.
3. Ved prakash [2023101006] : LTspice circuit simulation , Calibration, testing of sensors and Hardware Testing.
4. Rohan Kumar [2023101003] : Machine Learning part , Arduino Integration, code and Hardware Testing.