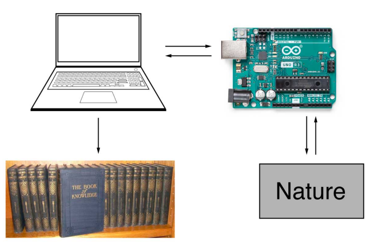
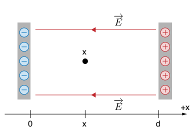
Devices

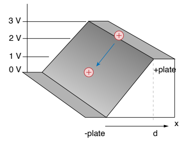


The general schematic for this class.

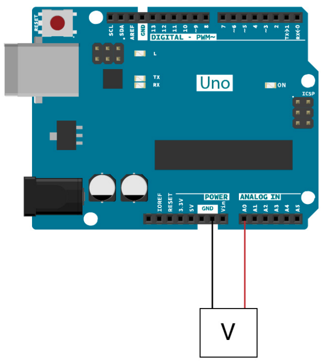
The graphic shows the basic schematic of this course. Nature provides some sort of phenomenon. We can interface with it (to be observe and influence). Then we can analyze the data. Then we can interpret it and repeat the cycle.

1. Voltage



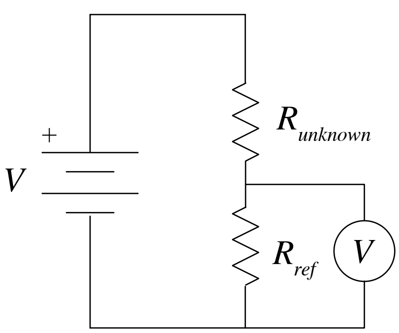


Voltage is what we have called the electric potential difference. Just like mass will move from high to low potential in a gravitational field, a charge will move from a high to low potential in an electric field. And just like gravity, the difference in potential between to locations is determined by the density of stuff. For gravity, it's mass. For voltage, it's charge. If the charge is distributed uniformly everywhere, then there is no potential difference. If charges are separated, then




```
/*
  AnalogReadSerial

  Reads an analog input on pin 0, prints the result to the Serial Monitor.
  Graphical representation is available using Serial Plotter (Tools > Serial Plotter menu).
  Attach the center pin of a potentiometer to pin A0, and the outside pins to +5V and ground.

  This example code is in the public domain.

  https://www.arduino.cc/en/Tutorial/BuiltInExamples/AnalogReadSerial
*/

// the setup routine runs once when you press reset:
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
}

// the loop routine runs over and over again forever:
void loop() {
  // read the input on analog pin 0:
  int sensorValue = analogRead(A0);
  // print out the value you read:
  Serial.println(sensorValue);
  delay(1);        // delay in between reads for stability
}
```

Primarily, what the arduino will measure is voltages.

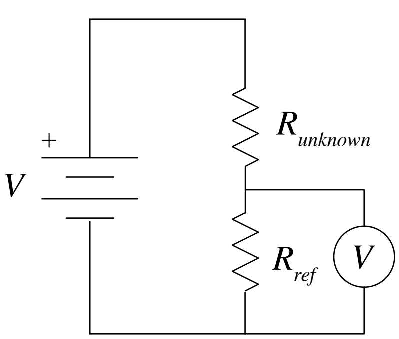


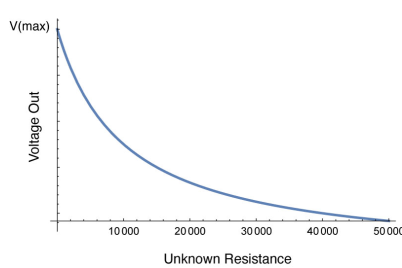
The voltage divider

The **Voltage Divider** circuit is the easiest way to measure an unknown resistance. It has some limitations, but will essentially work for many applications

$$V_{\text{out}} = \frac{R_{ref}}{R_{unknown} + R_{ref}} \cdot V_{\text{in}}$$

Voltage divider limitations





The voltage divider

2. $R_?$

Let's consider the ways we can affect $R_?$.

The first equation you saw that even mentioned R was probably Ohm's law.

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

While Ohm's Law is indeed useful for many things, it doesn't tell us anything about the material or thing that R applies to. It simple tells us how R relates to voltage and current (that are applied externally).

2.1 Deeper look at R.

Consider the resistivity instead:

$$\rho = \frac{E}{J}$$

A little substitution leads to

$$R = \rho\frac{L}{A}$$

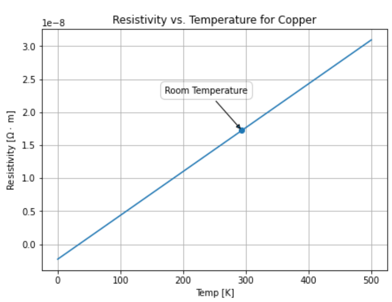
A little more leads to a relation between resistivity and some of the properties of a particular material.

$$\rho = \frac{m}{e^2 n\tau}$$

(m is the mass of an electron, e its charge, n is the electron density (i.e. carriers per volume), and τ is the mean free time (i.e. time between collisions))

Resistivity is the resistance of a certain material, not a particular object. i.e. the difference between the metal copper and a piece of copper rod that's 18 cm long and 2.5 cm in diameter, for example.)

2.2 R as a function of Temperature



Resistivity vs. Temperature (linear)

One of the first discoveries is that the resistivity will change due to temperature.

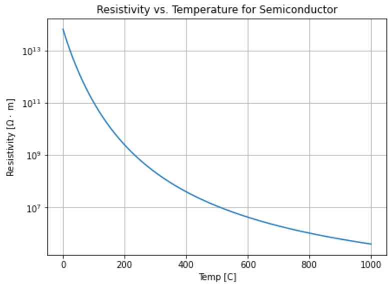
$$\rho - \rho_0 = \rho_0 \alpha (T - T_0)$$

This relation is a linear approximation and is not sufficient for many situations. Though, for most metals within certain T ranges, it suffices.

Python code snippit to make the above plot:

```
import numpy as np
import matplotlib.pyplot as plt

temp = np.linspace(0, 500, 500)
T_0 = 293
alpha = .00386
rho_0 = 1.72E-8
rho = rho_0*(1+alpha*(temp-T_0))

fig, ax = plt.subplots(figsize=(7, 5))

ax.plot(temp, rho)
ax.set_xlabel('Temp [K]')
ax.set_ylabel('Resistivity [$\\Omega \cdot$ m]')
ax.set_title('Resistivity vs. Temperature for Copper')
ax.plot(temp[293], rho[293],'o', color = 'tab:blue')
ax.grid()
plt.annotate("Room Temperature", # this is the text
                xy=(temp[293], rho[293]), # these are the coordinates to position the label
                textcoords="offset points", # how to position the text
                xytext=(-40,40), # distance from text to points (x,y)
                ha='center',
                va='bottom',
                bbox=dict(boxstyle='round,pad=0.5', fc='white', alpha=0.2),
                arrowprops=dict(arrowstyle = '-|>', connectionstyle='arc3, rad=0'))
plt.show()
```

2.3 Other materials/situations



Resistivity vs. Temperature for an intrinsic semiconductor)

Very Low temperatures: superconductivity

Semiconductors, different relationship:

$$\rho = \rho_0 e^{-aT}$$

(Note the log scale)

2.4 The Thermistor

These devices are designed to have resistance values that are very dependent on temperature. They are semi-conductors and can be custom made for specific temperature ranges.

The Steinhart–Hart equation approximates their temperature/resistance dependance.

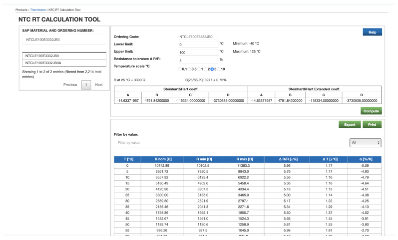
$$\frac{1}{T} = A + B \ln R + C(\ln R)^3$$

where, A, B, and C are constants specific to the actual device. (Note: there are several other forms of this equation. Use the product data sheet when you do your lab.)

A Thermistor

Two kinds: NTC (Negative Temperature Coefficient) and PTC (Positive Temperature Coefficient)

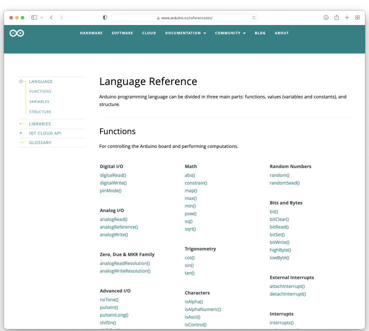
Found in kitchens, living rooms, hospitals, labs, etc...



Calibration tool (online at https://www.vishay.com/en/thermistors/ntc-rt-calculator/

3. Measuring T with arduino

It's not much different than what we did before, except now we have a time variable to record also.



The Arduino Programming Referencehttps://www.arduino.cc/reference/en/

The language reference is something to get used to working with.

For example, we need to find some way to record the time:

`millis()`

This function just returns the number of milliseconds since the sketch started running. Learn how to use it here:

https://www.arduino.cc/reference/en/language/functions/time/millis/

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

void loop() {

  int sensorValue = analogRead(A0);

  // Add time to the serial out put.

  Serial.print(millis());
  Serial.print(',');
  Serial.println(sensorValue);

  delay(10);
}
```

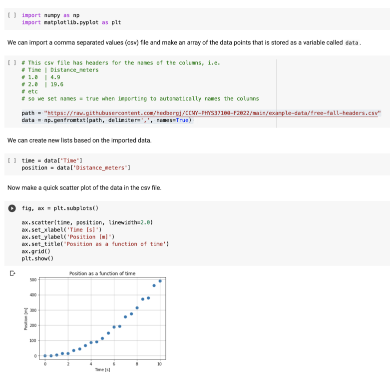
Here is our familiar `analog read` sketch. We just added some more stuff to send to the console.

`Serial.print()` vs. `Serial.println()`

3.5 Data Format

A very common data format is the `csv` (comma separated values)

```
DataColumn1, DataColumn2
0.0,432
0.1,433
0.2,436
...
```



```
[ ] import numpy as np
    import matplotlib.pyplot as plt
```

We can import a comma separated values (csv) file and make an array of the data points that is stored as a variable called `data`.

```
[ ] # This csv file has headers for the names of the columns, i.e.
    # Time | Distance_meters
    # 1.0  | 4.9
    # 2.0  | 19.6
    # etc
    # so we set names = true when importing to automatically names the columns

    path = "https://raw.githubusercontent.com/hedbergj/CCNY-PHYS37100-F2022/main/example-data/free-fall-headers.csv"
    data = np.genfromtxt(path, delimiter=',', names=True)
```

We can create new lists based on the imported data.

```
[ ] time = data['Time']
    position = data['Distance_meters']
```

Now make a quick scatter plot of the data in the csv file.

```
fig, ax = plt.subplots()

ax.scatter(time, position, linewidth=2.0)
ax.set_xlabel('Time [s]')
ax.set_ylabel('Position [m]')
ax.set_title('Position as a function of time')
ax.grid()
plt.show()
```


Example on how to read CSV from a file.
Tutorial is here:
https://hedberg.ccnysites.cuny.edu/PHYS371/tutorials/import-data/

4.