OREGON STATE UNIVERSITY – PHYSICS DEPARTMENT

Thesis

Characterizing a Coupled Photodiode-LED System for Neuro-Sensing Applications

Gregg Stevens
Advisor: Dr. Ethan Minot
5/6/2016

I investigate the feasibility of a micron-sized biosensor designed to measure neural activity. The circuitry of the biosensor is composed of graphene, some photodiodes, and a light-emitting diode. Many of these biosensors would be injected into a subject's brain, and when powered by near-infrared light, they would glow in response to a neuron's action potential. A model for the circuitry of these biosensors is developed and tested to predict the behavior of the biosensor. The ratio of optical output power to optical input power is greatest when the resistance of the graphene is minimized. However, the signal-to-noise ratio is better when the graphene has a higher resistance. The trade-off between these two goals is optimized by a specific resistance value that provides the greatest sensitivity of the instrument. In a test run, this model achieved a 13% energy conversion rate. This efficiency could be increased by using higher-quality components. These micro-biosensors could solve several of the current challenges facing neuro-sensing technology, including measuring the behavior of a single neuron.

Table of Contents

Table of Figures

1. Introduction

Since our understanding of nanotechnology has grown, scientists have considered using microscopic sensors to interface with neurons of the brain to effectively map out its pathways. If this imaging method is achieved, we would be able to understand and treat neural problems.

Researchers have injected micro-sensors into a host's brain. In doing so, they have found size of the sensor to be an obstacle, because tissue reactions occur when a device is larger than the diameter of a cell (according to Seo et al.). These reactions "are ultimately detrimental to performance and occur on the timescale of a couple months" [1]. We would like to make a smaller biosensor to overcome this limitation.

I investigated the feasibility of a biosensor constructed as in Figure 1. This biosensor is designed to be created at a micro-scale and injected en masse into a host's brain. The biosensors will be optically powered by an infrared photodiode. As a laser shines through the subject's head, the photodiode will power the circuit and send out a signal through the LED. The output signal strength will vary based on the resistance of a sheet of graphene. In operation, there will always be a faint glow from the LED. It is expected that as the neuron fires, the resistance of the graphene will drop, which will result in more current flowing through the circuit, and thus a surge of light output. Using an infrared detector, this signal can be measured to find the location of neural activity. The size of this device would allow us to receive signals from individual neurons.

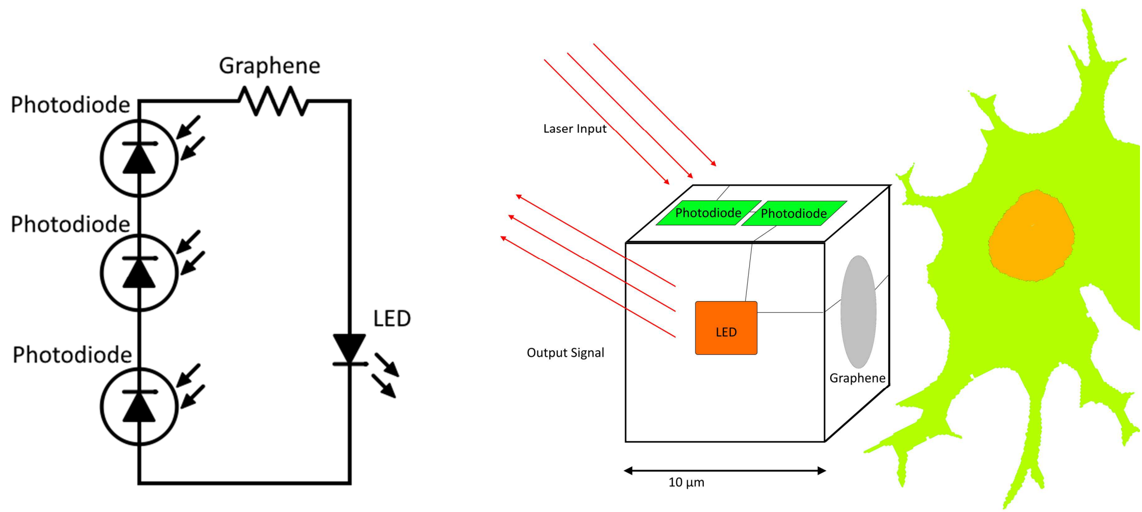

Figure 1: Model of a micro-biosensor. On right is a biosensor in operation: it has bound itself to a neuron and is sending a signal. On left is the circuitry making up the biosensor.

1.1 Importance

In a review published 2014, Fattahi, et. al. recognize the importance of projects like this because "obtaining quantitative information of neuronal activity is one core task in neuroscience." Their review considers the materials possible for creating a neuronal sensor, finding graphene to be a very promising candidate as it is not very toxic to a cell in high quantities. Additionally, they investigate methods to secure graphene onto a neuron [2].

1.2 Exposure Limitations

One requirement of the proposed biosensor design is a light power source. Unfortunately, there are negative health consequences from massive amounts of EM radiation—mostly due to heating—so there is a limit to the amount of light that can be used. The American National Standards Institute (ANSI) has set up a maximum permissible exposure (MPE) limit. In the near-infrared range, this limit is $MPE = 0.2 * 10^{2(\lambda-0.700)} \frac{W}{cm^2}$ for wavelengths of 0.7 - 1.05 µm [3]. This means that the maximum intensity is $0.2 - 1.0 \frac{W}{cm^2}$.

One point to address is that both the incident and emitted radiation would be in the near-infrared zone. When considering the photons involved, conservation of energy would dictate that the photon coming in must have more energy than the photon being emitted, because there is energy loss in each component of the biosensor. By this train of thought, it would be difficult to keep both input and output radiation in the near-infrared regions. However, by using multiple photodiodes in series, the device can absorb multiple incident photons for each emitted photon. For example, if there are eight photodiodes absorbing 700 nm light, the energy intake is $8 * 1.8eV = 14.4\ eV$. The circuit could be just 12% efficient but would still have sufficient energy for an infrared photon. Therefore, conservation of energy does not dictate a specific output wavelength.

1.3 Prior Research

1.3.1 Acoustically-powered devices

Seo et. al. at the University of California Berkeley consider a light-powered neural-imaging device to be impractical due to the MPE of light. Consequently, they formulated plans for a micro biosensor using a piezoelectric base that can generate a current when excited by a sound wave. They anticipate that brain activity can be measured after many of their devices are injected into the brain [1]. Their calculations on light-powered devices are not complete, as they did not consider the near-infrared range of light, where the head is less absorptive.

1.3.2 Light absorption

Haeussinger et. al. simulated how far near-infrared light can penetrate through the human head. As shown in Figure 2, they found that 5% of incident radiation can penetrate through a distance of 23 mm [4].

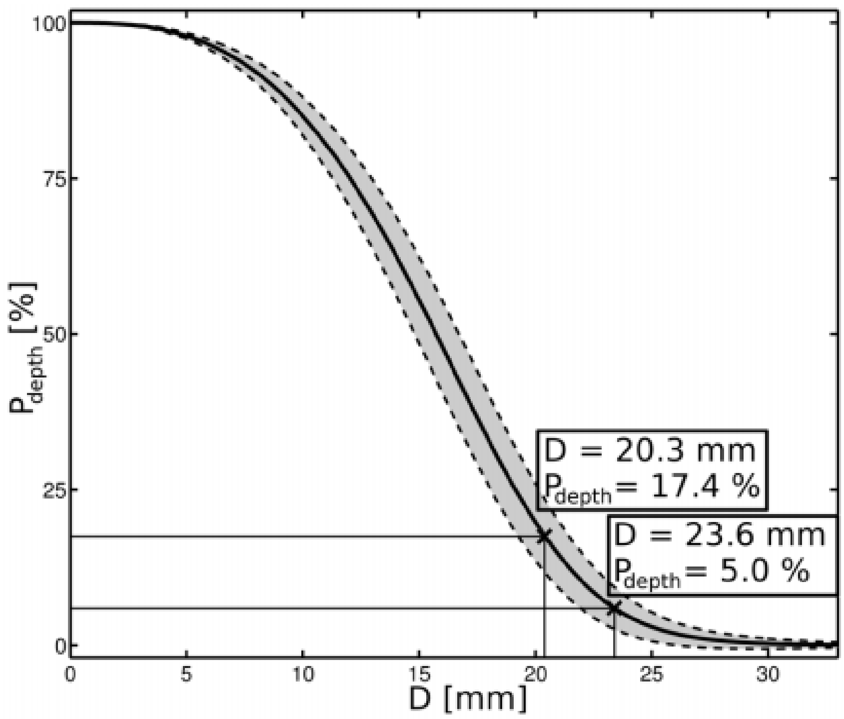

Figure 2: Penetration depth of near-infrared light through a human head. Five-percent of the initial radiation can penetrate to a depth of 23mm. Figure reproduced from [4].

1.3.3 Optoelectronic micro-sensor is feasible

Contrary to the acoustic-only thoughts of Seo et. al, researchers Kim et. al. have injected a µ-ILED device (approximately 20 x 400 x 200 µm) into rat brains. When the rats are hit by a radio wave (~ 0.5 mW/cm^2), the devices respond by stimulating neurons. By targeting specific neurons, they have been able to affect and control the behavior of rats. Even at this small scale, these devices contained a temperature sensor and four LEDs, and allowed for real-time programming [5]. This suggests that devices this small are feasible.

1.3.4 Graphene responds to neuron's action potential

Du et. al. have investigated the physical properties of graphene and developed an array of graphene microelectrodes to sense neurons. By connecting one of these arrays to an amplifier, they were able to detect action potentials with a signal-noise-ratio of ~10. The signal was recorded as the voltage difference between the graphene array and the surrounding environment [6]. This may be the first step in altering graphene's resistance based on its environment. This inspires the use of graphene as the transducer in our circuit.

1.4 Scope of project

The goal of this project is 1) to find if a light-powered sensor is feasible, and 2) to develop code to easily model the new biosensor design.

2. Theory

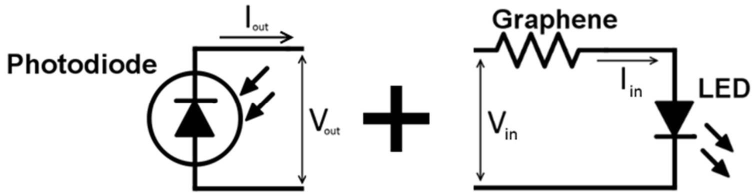

Figure 3: The biosensor circuit can be split up into two components. This makes it easier to understand how the circuit is operating and lends itself to easier computation. Note that $V_{out} = V_{in}$ and $I_{out} = I_{in}$.

There is no closed-form expression for the relationship between current and voltages in Figure 1. To solve the I-V curve of the circuit, it must be deconstructed into two parts: one circuit has just the photodiode; the other contains the resistor and LED (see Figure 3). Since the two are connected in series, the current and voltage at the nodes must be equivalent.

2.1 LED-Graphene Sub-circuit

Attention will first be given to the LED sub-circuit (Figure 4).

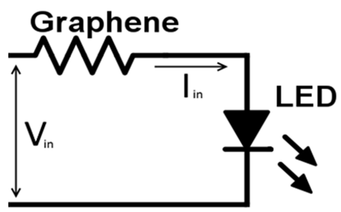

Figure 4: LED Sub-circuit

The I-V curve of a light-emitting diode is exponential:

$$I = I_0\left(e^{\frac{qV}{nkT}} - 1\right) \qquad \text{Equation 1}$$

where I_0 is the dark saturation current, *n* is an ideality factor on the order of 1, *k* is the Boltzmann constant, *T* is the device temperature, q is the fundamental charge, and V is the voltage applied across the LED. The graph starts with very little current until the turn-on voltage (which is dependent on the band gap of the semiconductor used). After the turn-on voltage, the current spikes sharply.

Now, the LED has some internal resistance, so the current cannot continue increasing exponentially. Since it is in series with the graphene, the two resistances add. The resistance in the circuit changes to

$$\Delta V_{\text{in}} = \Delta V_{\text{Graphene}} + \Delta V_{\text{LED}} = \frac{nkT}{q}\ln\left(1 + \frac{I_{\text{in}}}{I_0}\right) + I_{\text{in}}R_{\text{eff}} \qquad \text{Equation 2}$$

where $R_{\text{eff}} = R_{\text{LED}} + R_{\text{Graphene}}$. Equation 2 gives a perfect I-V curve for the LED sub-circuit. However, it is much easier to approximate the graph with two lines: one region has no current (approximately $0 - 0.6\ V$), and the following region is linearly increasing. The curve is approximated using Ohm's law and a turn-on voltage, V_0:

$$V_{\text{in}} = IR_{\text{eff}} + V_0 \qquad \text{Equation 3}$$

2.2 Photodiode Sub-circuit

Now, looking at the other half of the circuit, a photodiode is best modeled as a current source paired with a diode and a set of resistors (see Figure 5). R_S represents a series resistor, and R_{Sh} represents a shunt resistor:

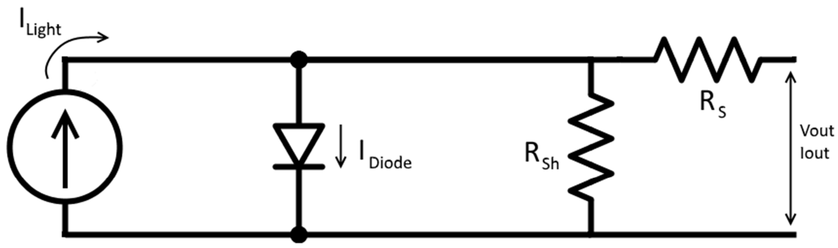

Figure 5: Model of a photodiode. The diode gives the overall behavior of an ideal photodiode. The series and shunt resistors give the non-idealities of an actual photodiode. This circuit is equivalent to the left half of Figure 3.

This model explains the behavior of a photodiode. At low voltages, the diode and shunt resistor have high resistance, so most of the current must go through the output. However, when the voltage is high, the diode functions as a short-circuit, letting most of the current pass through. The output current in this figure is

$$I_{\text{out}} = I_{\text{Light}} - I_{\text{Diode}_0} * e^{\frac{q(V_{\text{out}}+I_{\text{out}}R_S)}{nkT}} - \frac{V_{\text{out}} + I_{\text{out}}R_S}{R_{Sh}} \qquad \text{Equation 4}$$

where I_{Diode_0} is the dark saturation current of the diode in Figure 5, *q* is the fundamental charge, *n* is an ideality factor, *k* is Boltzmann's constant, and *T* is the temperature.

The quality of material will change the series and shunt resistances. Ideally, the series resistance should be low (or 0), and the shunt resistance should be high (infinite). Figure 6 and Figure 7 below show how the series and shunt resistances affect the I-V curve.

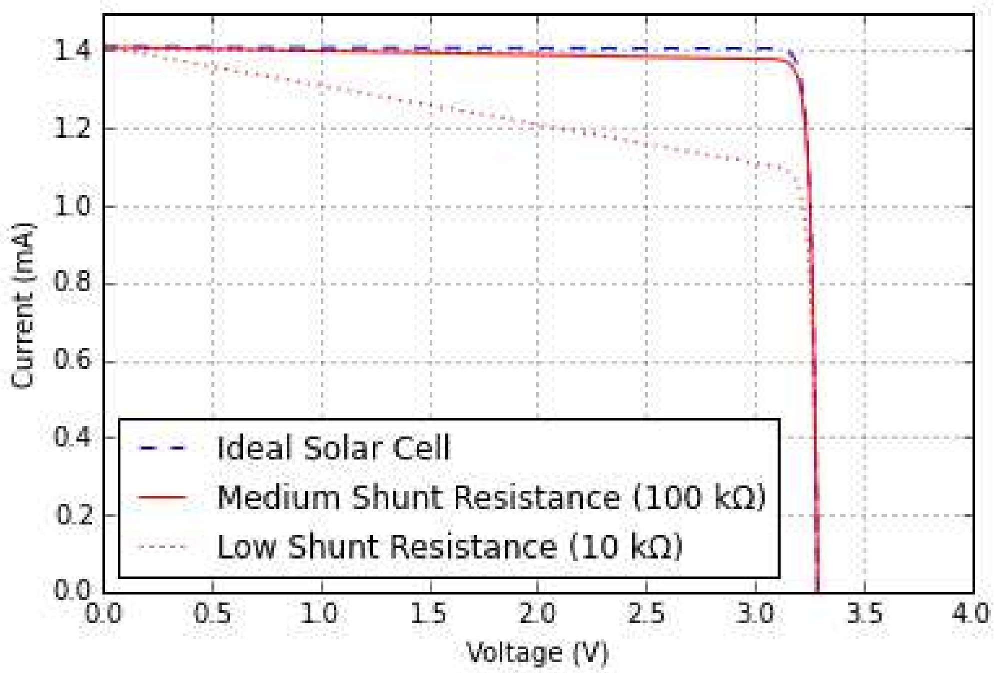

Figure 6: Typical I-V curve with low shunt resistance. The parameters in this figure are as follows: *n* = 1, $R_S = 0$, $T = 310\ K$, $V_{OC} = 3.3\ V$, and $I_{SC} = 1.41\ mA$.

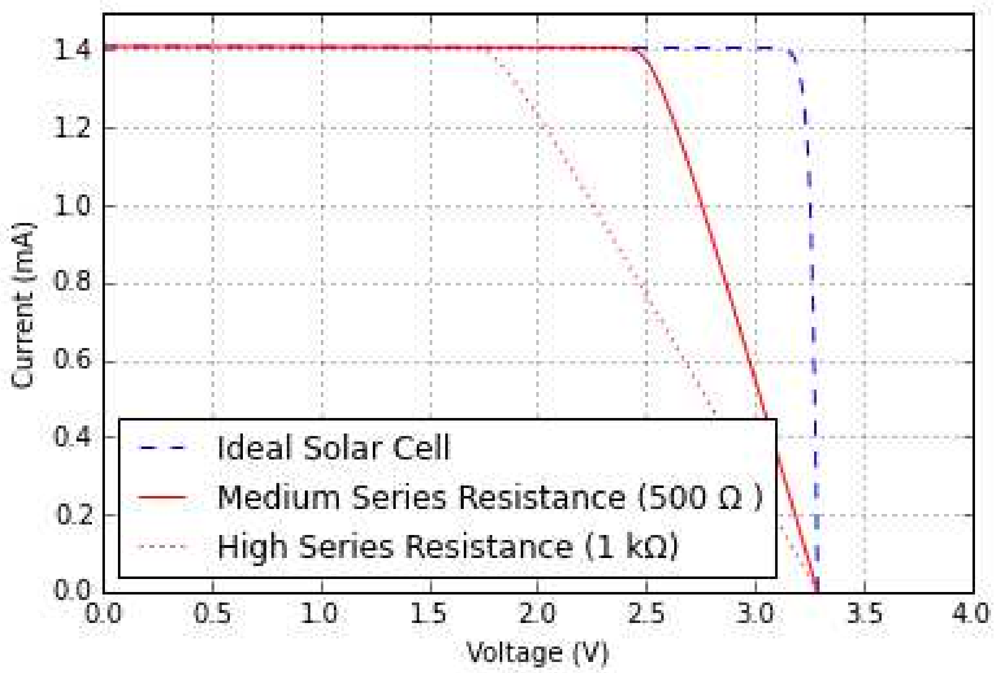

Figure 7: Typical I-V Curve with various series resistance values. The parameters in this figure are as follows: *n* = 1, $R_{Sh} \sim \infty$, $T = 310\ K$, $V_{OC} = 3.3\ V$, and $I_{SC} = 1.41\ mA$.

2.3 The Combined Circuit

After examining both sub-circuits, we have come to two equations relating each other:

LED system	$V_{in} = I_{in}R_{eff} + V_0$	Equation 5
Photodiode	$I_{out} = I_{Light} - I_{0_{Diode}} * e^{\frac{q(V_{out}+I_{out}R_S)}{nkT}} - \frac{V_{out} + I_{out}R_S}{R_{Sh}}$	Equation 6

When the elements are combined, the current and voltage must match: $I_{out} = I_{in}$; $V_{out} = V_{in}$.

3. Experimental Methods and Results

I constructed a model circuit and measured the parameters in the model above. I used a 5 in. x 6 in. solar cell for the photodiode. I used a red 630 nm LED for the LED and various resistors ranging from 1-10 kΩ in place for the graphene. First I found the values of the series and shunt resistances (the non-idealities defined in Figure 5).

3.1 Photodiode Series Resistance Measurements

I measured the series resistance according to the method devised by Cabestany and Castañer [7]. When the circuit in Figure 8 is tested under dark conditions, the series resistance is given by

$$R_{series} = \frac{\frac{KT}{q}\ln\frac{I_2}{I_1} + I_2R_{ext_2} - I_1R_{ext_1}}{I_1 - I_2} \quad \text{Equation 7}$$

where the external source is a constant voltage source and I_1 and I_2 are the currents measured with external resistors R_{ext_1} and R_{ext_2} respectively.

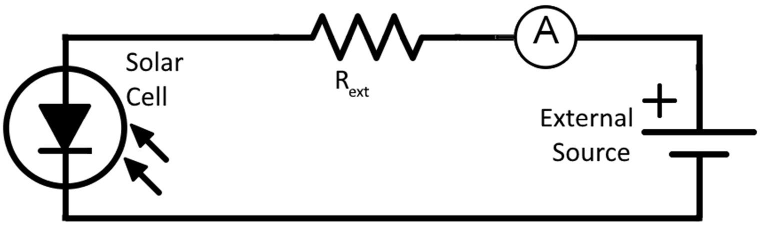

Figure 8: Circuit to measure series resistance. The solar cell is connected in the dark to an external voltage supply and resistor. The current is measured at various resistances and used to calculate the series resistance.

I constructed the Figure 8 circuit under a box (for darkness). At room temperature, I applied a 3.4 V source voltage across the circuit. I measured the current with the ammeter, using three values for R_{ext}: 1 kΩ, 500 Ω, and 0 Ω.

3.2 Series Resistance Results

I made 3 resistance measurements according to the Cabestany and Castañer [7] method (as shown in Table 1):

Series resistance data	Trial 1	Trial 2	Trial 3
External Resistor (Ω)	978	0	500
Current (mA)	0.86	2.27	1.14

Table 1: Series resistance data measured from Figure 8

These three trials give different values for series resistance (R_S) depending on which pair is used: 520 Ω, 580 Ω, and 943 Ω. More data points could be obtained for higher precision, but the average series resistance is 680 Ω.

3.3 Photodiode Shunt Resistance Measurements

I measured the shunt resistance by the method proposed by Chan and Phang [8]. Under very low light conditions, the shunt resistance of a photodiode is equivalent to the Thevenin resistance:

$$R_{Sh} = \frac{I_{sc}}{V_{oc}} \quad \text{Equation 8}$$

Thus, the shunt resistance is calculated from two measurements of the solar panel, as shown in Figure 9.

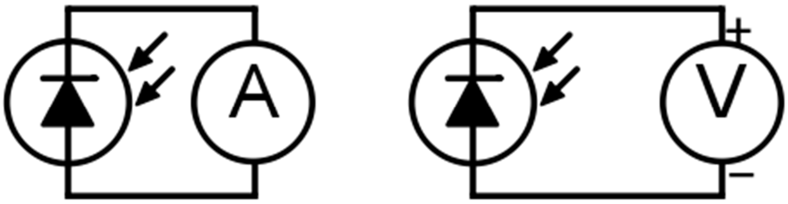

Figure 9: Circuits to measure shunt resistance. With the short-circuit current and open-circuit voltage are measured by connecting a voltmeter or ammeter directly to the solar panel. The shunt resistance is found as the average Thevenin resistance at various light levels.

I put the photodiode under a box and let its wires stick out so that I could alter the circuit without changing the amount of light incident on it. I let in a small amount of ambient light and measured the short-circuit current and open-circuit voltage, as shown in Figure 9. Then I raised the box to let more light in and repeated the experiment.

3.4 Shunt Resistance Results

I used a linear regression on the shunt data, using the open circuit voltage as the y-values and the short circuit current as the x-values. The slope of this line (R_{Sh}) is approximately 62.5 kΩ (Figure 10).

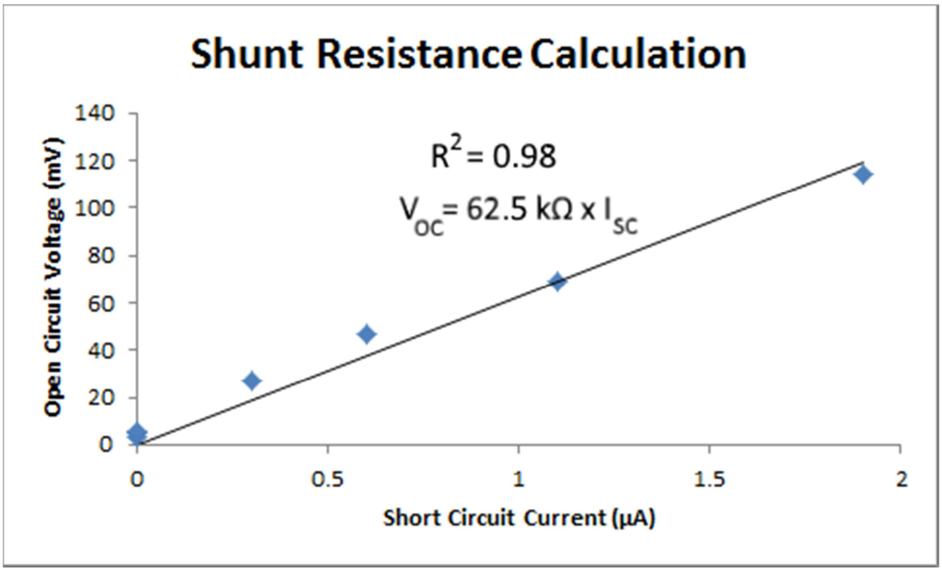

Figure 10: Linear fit to calculate the solar cell's shunt resistance. The slope of this line (62.5 kΩ) can be interpreted as the shunt resistance of the solar cell.

3.5 Photodiode I-V curve measurements

I measured the I-V curve of the photodiode by constructing the circuit in Figure 11. I set the photodiode in an isolated room lit only by ambient ceiling lights to eliminate fluctuations due to weather. I started the variable resistor at 0 Ω and gradually increased its resistance. At each state, I recorded the current and voltage of the circuit.

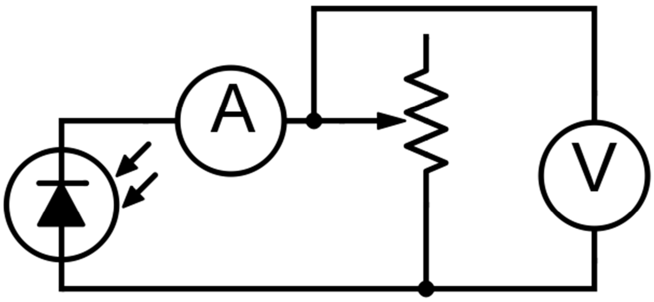

Figure 11: Circuit to measure the I-V curve of a solar cell. The solar cell is connected to a variable resistor. The current and voltage is measured across that circuit using an ammeter and voltmeter. The resistor is varied to produce the I-V curve of the solar cell.

3.6 Photodiode I-V Curve

Figure 12 shows the I-V curve I measured from the experimental photodiode.

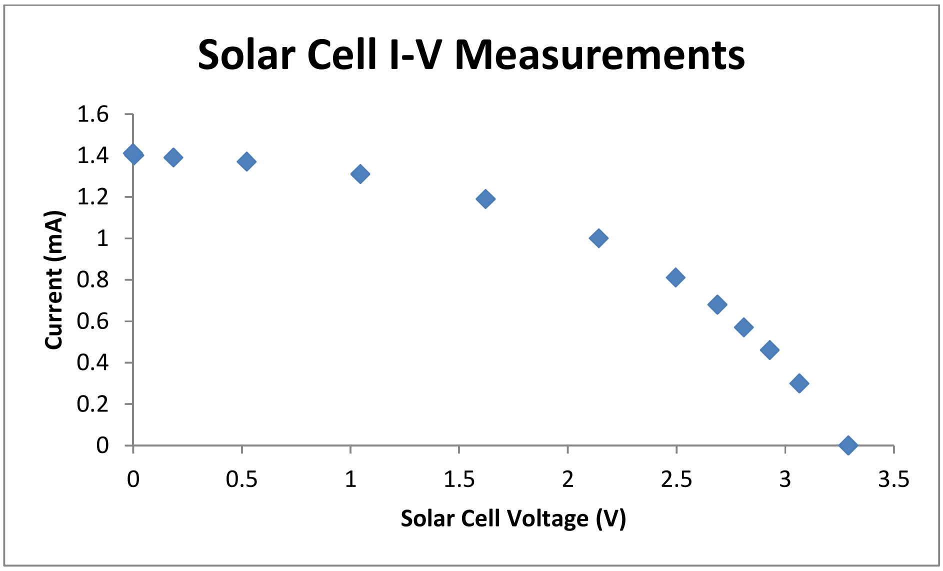

Figure 12: Current from photodiode. Under the given light conditions, this solar cell can produce a maximum of 1.41 µA or 3.29 V.

3.7 Linear Approximation of LED

To test the validity of a linear model of the LED circuit, I measured the I-V curves of the LED/resistor under various resistances. I used a red LED and connected the circuit shown in Figure 13 under room temperature. For each resistor, I recorded the voltage drop across the voltmeter as I increased the source voltage. Using Ohm's law, I solved for the current through the circuit.

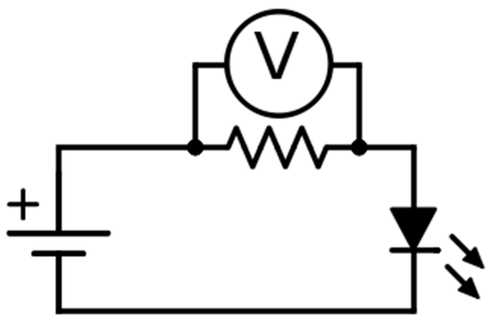

Figure 13: Circuit to test linear model of LED. A voltage supply is connected to an LED and a resistor on the order of 1 kΩ. The I-V curve is measured by increasing the input voltage from the voltage supply.

3.8 Linear Approximation Results

I found a linear fit to the I-V curves recorded by the method shown in Figure 13. As seen in Figure 14, the linear approximation fits the data very accurately ($R^2 \geq 0.99$). The turn-on voltage is the same no matter what resistance is used.

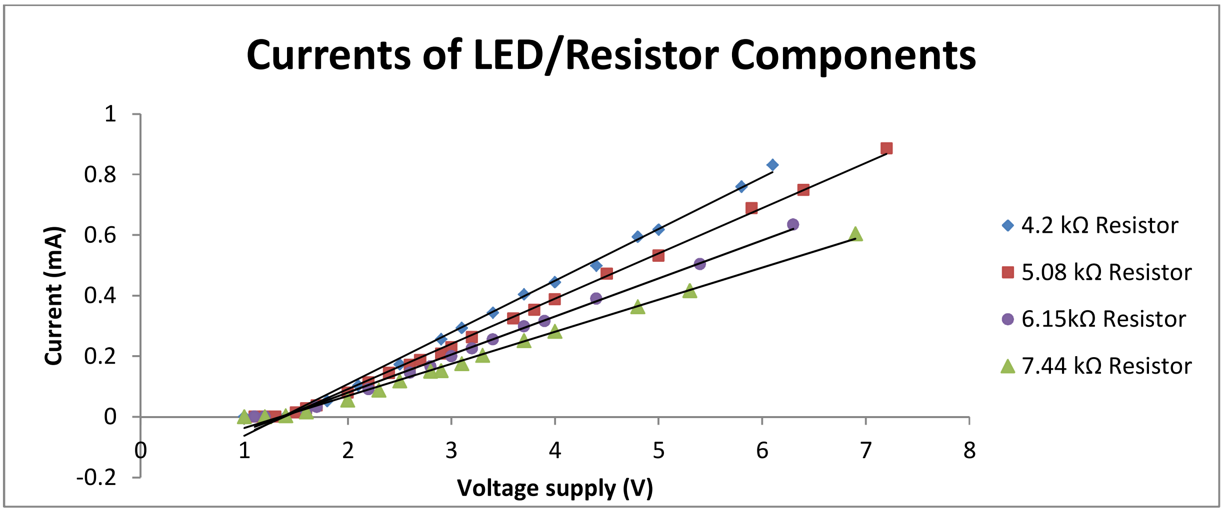

Figure 14: Linear approximations of LED/Resistor components. Little current is produced until the turn-on voltage, 1.55 V. At that point, the current increases linearly with applied voltage. The slope represents both the LED's internal resistance and the added resistor.

3.9 Complete Circuit Current

I connected the circuit in Figure 15 to find the current in the combined circuit. I found the voltage across the resistor at various resistances between 0 kΩ and 10 kΩ. Using Ohm's law, I interpreted the current through the circuit.

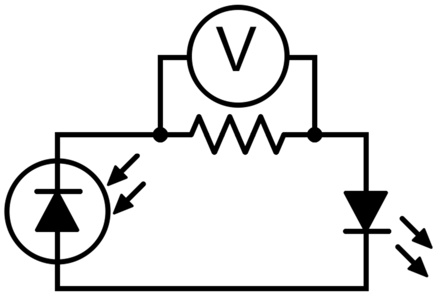

Figure 15: Circuit to measure total current. This circuit is the model for the biosensor (Figure 1). A voltmeter is used across the resistor to measure the current through the circuit. The current is measured with different resistors.

3.10 Complete Circuit Results

I found a few data points for the current through the whole circuit (Table 2). These are compared with the computational results below in "Section 4.6 System at various resistances."

Resistance (Ω)	Voltage drop (V)	Current (mA)
5100	1.208	0.2369
4300	1.109	0.2579
2000	0.85	0.425
8200	1.259	0.1535

Table 2: Test Currents of Complete Circuit

4. Computational methods

I developed a python model to solve the biosensor model in general. Using Python's Anaconda library (i.e. matplotlib, numpy, and scipy), I simulated the current in the circuit for different values of the resistor, R_{Graphene}.

4.1 Photodiode Model

The photodiode model from the section "2.2 Photodiode Sub-circuit" requires 4 input parameters: 1) the series resistance, 2) the shunt resistance, 3) the open-circuit voltage, and 4) the short-circuit current. From these constants, the program displays the solar panel's I-V curve.

However, there is one more degree of freedom—the recombination factor. This ideality factor, *n*, can be estimated from the curve if given set of data points. I plotted the experimental data with the 4 input parameters. Then I tried different values for *n* until the graph closely matched the experiment.

4.2 I-V curve of Photodiode

Figure 16 shows the I-V curve of the solar panel under ambient light. The red squares represent the measured data, and the dotted red line shows the model. The best ideality factor, *n*, was found by trial-and-error to be 15. This is significantly higher than expected, but the curve fits the data well.

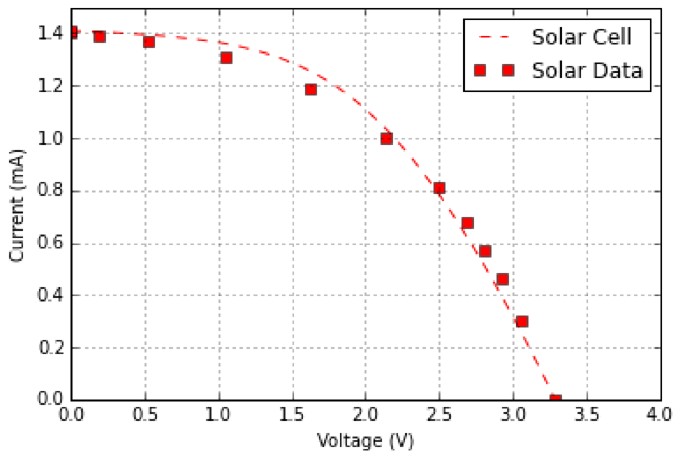

Figure 16: Photodiode data compared with model. The parameters for this graph are measured as $R_S = 579\ \Omega$, $R_{Sh} = 62.5\ k\Omega$, and $n = 15$.

In this model, we fit the series and shunt resistance with independent parameters. This results in the model deviating slightly from the measured data in the 0.5-2.0 V range. An alternative method is to fit these imperfections from the photodiode's I-V curve, as seen in Figure 17. Note that the fitted line matches the data more closely.

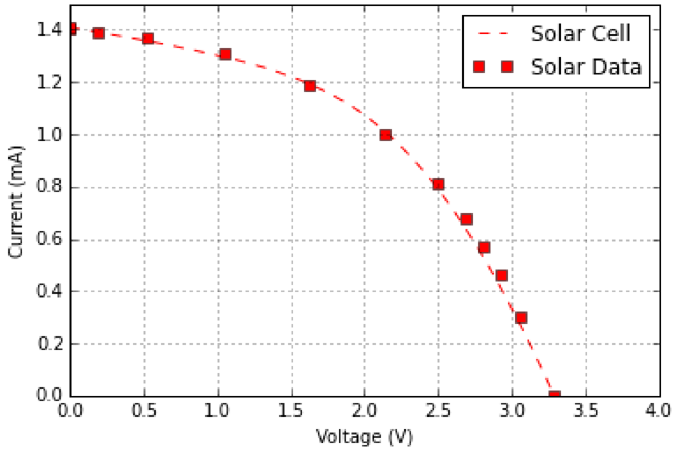

Figure 17: Photodiode data compared with fitted model. The parameters for this curve are fitted as $R_S = 500\ \Omega$, $R_{Sh} = 10\ k\Omega$, and $n = 14$.

4.3 LED Model

The diode model requires fewer parameters. Since the graphene's resistance is a known quantity, the only unknowns are the turn-on voltage and the internal resistance of the LED. I interpreted this from the data. Then I modeled the I-V curve with various levels of added resistance.

4.4 I-V curve of LED

Using the data from Figure 14, I found that the turn-on voltage of the LED system is $V_0 = 1.55\ V$. The LED has an internal resistance of 1.3 kΩ. Figure 18 shows that these numbers match the data well.

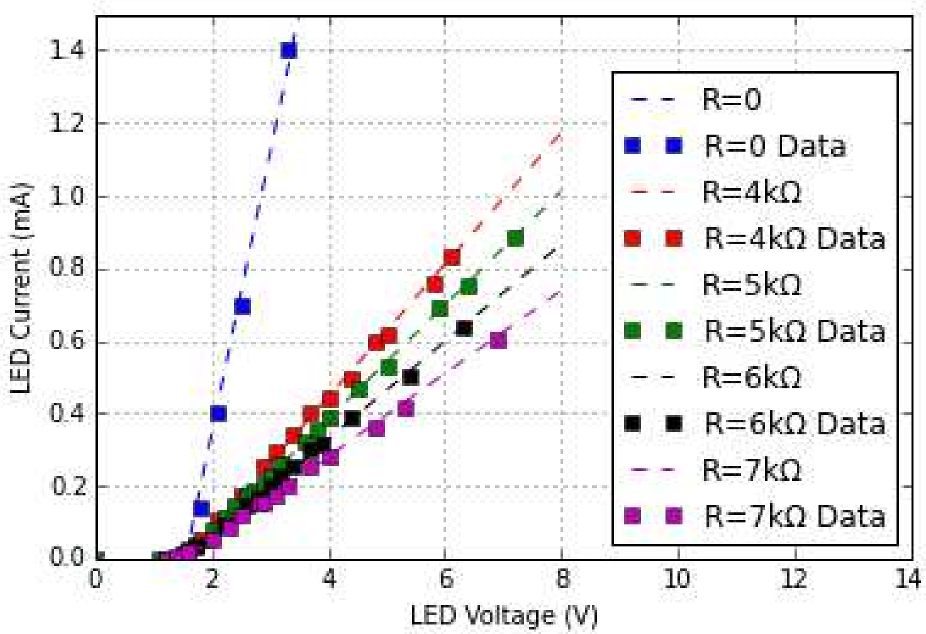

Figure 18: LED data vs. model. A linear model fits the LED sub-circuit well, regardless of the resistance used.

4.5 Solving the circuit

With the two models working as described above, we find the intersection point of the I-V graphs to find the solution of the complete circuit (demonstrated in Figure 19).

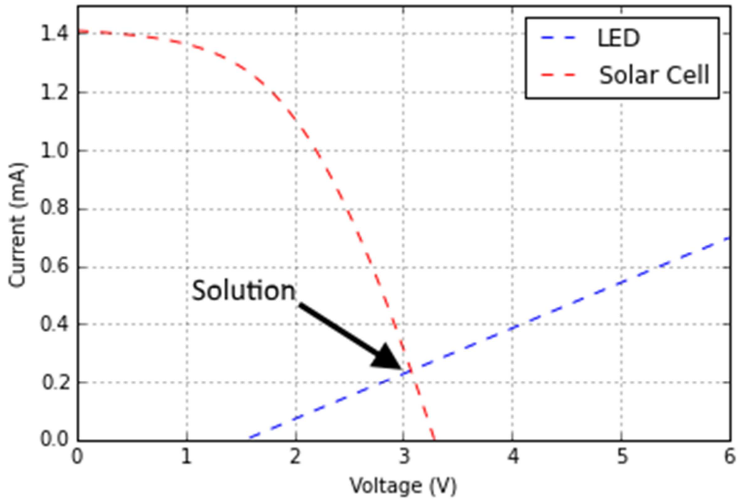

Figure 19: Solving the complete circuit. The intersection of the two I-V curves represents how the circuit will operate given these parameters. In this case, the solar cell would output 3.1 V, and the current would be 0.24 mA.

I extended the model to find the current as the graphene's resistance changes. For each resistance, I found the intersection point (as shown in Figure 20).

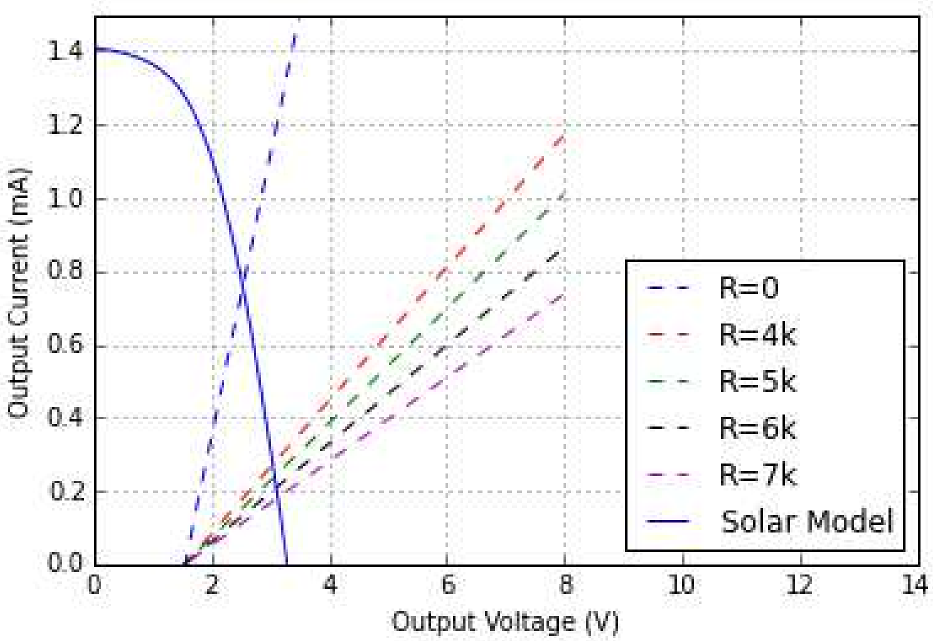

Figure 20: Intersections at various resistances. By finding the solution at multiple resistance values, the current can be plotted as a function of resistance.

4.6 System at various resistances

I found the intersections of the two I-V curves at different resistances, and plotted the current against the graphene's resistance. Figure 21 compares the model with the experimental data. The model fits the data, with a residual sum of squared errors of $5.1 * 10^{-4}\ mA^2$. In other words, the model is off by an average of $0.01\ mA$, which shows that the theory and computational methods are sound.

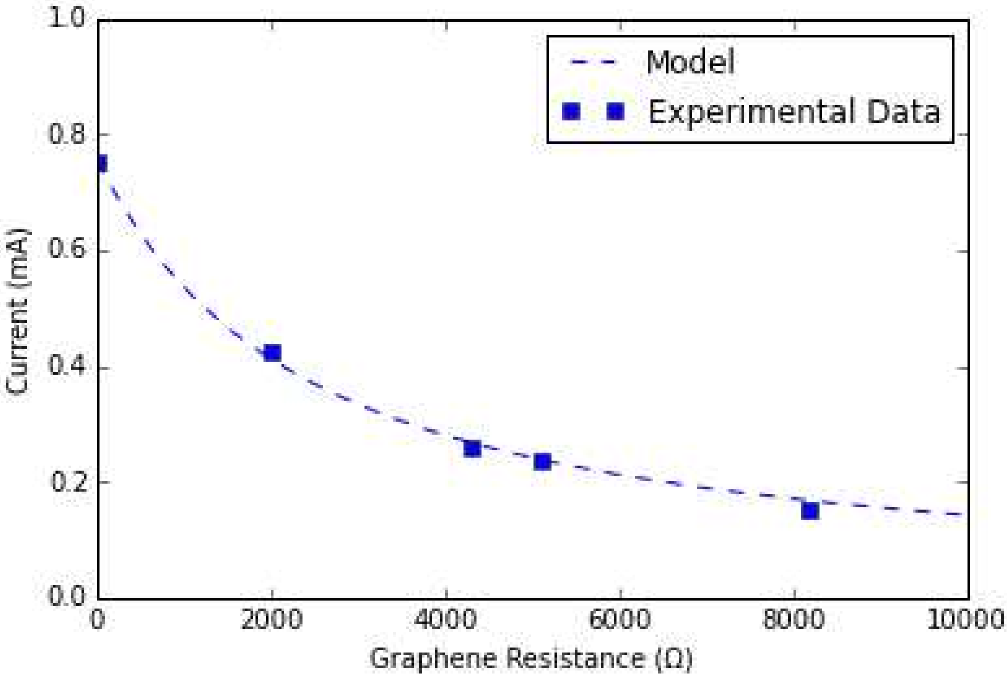

Figure 21: System current vs. graphene resistance. This graph shows how the circuit will behave at various resistance values. As resistance increases, the current (and thus intensity) decreases. This can be used to optimize efficiency of the system.

For completeness, I also plotted the output voltage against the graphene's resistance, but these results are not as practical (the output light intensity is related to current, not voltage). This graph is included in the appendix.

5. Discussion/Analysis

5.1 Interpreting Current as Intensity

In an LED, the intensity of the light is proportional to the current. In an efficient LED, each electron-hole recombination will emit a photon. Thus the number of photons is equal to the number of electrons passing through the LED—or the current. Figure 21 then becomes very important, as the axes can be interpreted as the intensity against the resistance. At low R_{Graphene}, the current is high, and consequently, the intensity of light is also high.

5.2 Using the Normalized Change in Resistance – δ_R

In a practical sense, we expect that the graphene's resistivity changes depending on its environment. This means that the graphene's total resistance changes as a percentage of its original resistance. For example, if the graphene started at 4 kΩ before the neuron fired, it will decrease by say 10%, so the new resistance will be 0.9 * 4 kΩ. Thus, when optimizing the system, we must consider the normalized change in resistance (hereafter denoted as δ_R), not the total change in resistance (dR). The two are related by Equation 9:

$$\delta_R \equiv \frac{\Delta R}{R} \qquad \text{Equation 9}$$

5.3 Maximizing the Change in Light

Using the model from Figure 21, I took the derivative with respect to the normalized change in resistance, δ_R. This can be calculated as

$$\frac{dI}{\delta_R} = \frac{dI}{dR/_R} = R\frac{dI}{dR} \qquad \text{Equation 10}$$

Figure 22 shows that the greatest change in light for the prototype can be achieved in the 1-3 kΩ range.

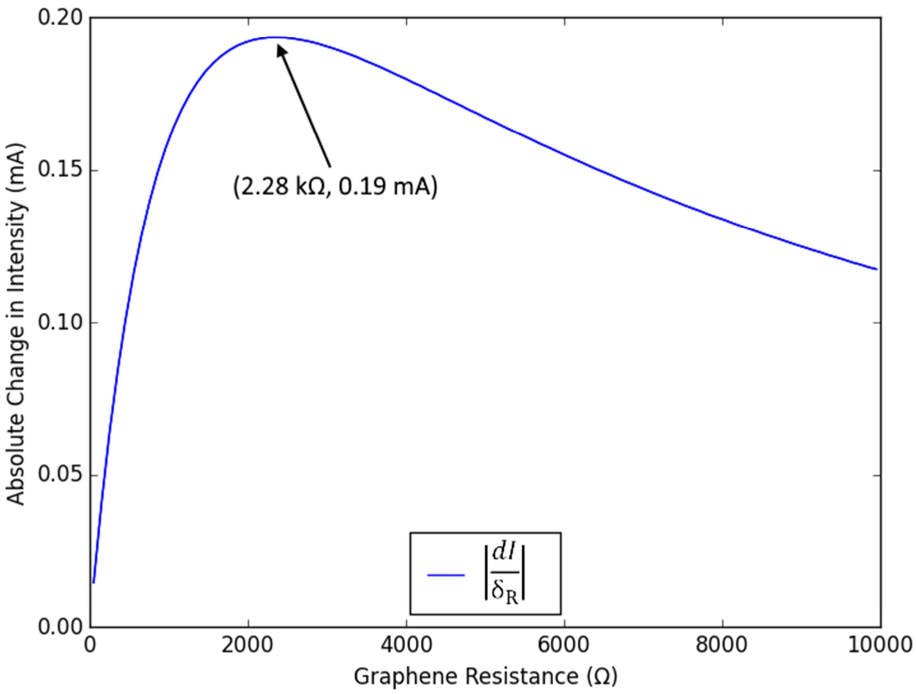

Figure 22: Maximizing the total change in light. At low resistances, a fractional change in resistance does not have much effect on the current because the resistance is not significant compared with the other circuit components. At higher resistances, a fractional change in resistance does not change the intensity as much because there is less current flowing. In our test model, the optimal value is the range surrounding 2.28 kΩ.

It is important to understand where the extremum in Figure 22 comes from. While the slope in Figure 21, $\frac{dI}{\delta_R}$, is negative but becoming less steep, Figure 22 takes into account the normalized change in resistance, not the total change in resistance. It is multiplied by a linear factor of R, which is why $\frac{dI}{\delta_R}$ has an extremum at 2.28 kΩ. The graph shows a maximum because the graph is an absolute value.

5.4 Maximizing Normalized Change in Light

An argument can also be made that the device should be operated where there is a maximum normalized change in the intensity (hereafter denoted as δ_I). A bigger normalized change will make the output signal easier to detect. Similar to the resistance, we can relate the normalized change of the intensity as

$$\delta_I \equiv \frac{\Delta I}{I} \qquad \text{Equation 11}$$

We take the derivative of the normalized change in light,

$$\frac{\delta_I}{\delta_R} = \frac{dI/_I}{dR/_R} = \frac{R}{I} * \frac{dI}{dR} \qquad \text{Equation 12}$$

Figure 23 shows that for our system, the maximum normalized change in light is achieved at the highest resistance.

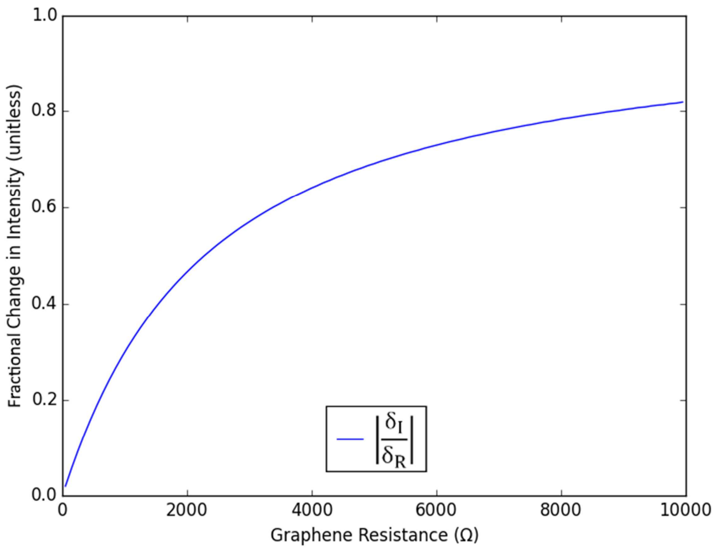

Figure 23: Optimizing the normalized change in light. At higher resistance values, a fractional change in resistance will result in a larger fractional change in current. This is caused by two factors: 1) the resistor has more significant resistance compared to the solar cell's and LED's series resistance, and 2) the slope of the solar cell I-V curve becomes increasingly vertical, so a change in the resistance (or slope of the LED curve) will have a bigger effect.

One consequence of the data in Figure 23 is that in the limit of very high resistance, we can at most achieve a 1:1 ratio between δ_{I} and δ_{R}.

5.5 Total Light vs. Normalized Light

So which do we choose? Should we optimize $\frac{\delta_{\mathrm{I}}}{\delta_{\mathrm{R}}}$ or $\frac{dI}{\delta_{\mathrm{R}}}$? If we operate at low resistance, then we optimize the total change in light. However, there the signal will not be very distinct because the normalized change will be small. On the other hand, if we operate at high resistance, then we get a distinct signal, but the intensity will be faint.

We return to the initial constraints: 1) the total power is limited, and 2) the human body absorbs much of the light that hits it. In order to make the best use of the power available, we must conclude that the total change in light $\left(\frac{dI}{\delta_{\mathrm{R}}}\right)$ must be optimized.

5.6 Efficiency

Figure 24 shows the relationship between input power and output power. As the resistance increases, the efficiency decreases. With no added resistance, the experimental solar cell converts 41% of the optical power into electrical energy. When coupled with the LED, the device can convert up to 25% of the optical input power into optical output. In the operational range ($2.28\ k\Omega$), the device converts 24.4% of its optical power into electrical energy. The circuit as a whole converts 12.9% of its input power into optical output. Thus, disregarding absorption in its environment, this device is 12.9% efficient.

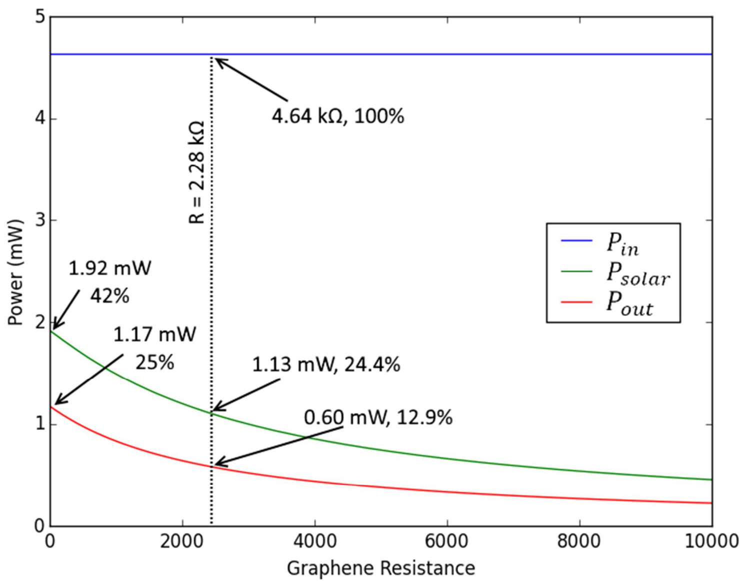

Figure 24: Power vs. resistance. At minimal resistance, the prototype circuit converts 25% of its optical input energy into an optical signal. At the operational resistance, it converts 12.9% of its energy into an output optical signal.

The power values in Figure 24 were calculated as follows:

$$P_{in} = V_{OC} * I_{SC} \quad \text{Equation 13}$$

$$P_{solar} = V_{out} * I_{out} \quad \text{Equation 14}$$

$$P_{out} = V_{LED} * I_{out} \quad \text{Equation 15}$$

5.7 Increasing Efficiency

The greatest portion of power was lost in the photodiode, mostly due to the quality of the photodiode we used. There are 3 methods in which the efficiency of the prototype can be increased:

- Increase the quality of the photodiode (fill factor) by
    - Lowering the recombination rate, "n"
    - Increasing the shunt resistance
    - Decreasing the series resistance
- Decrease the internal resistance of the LED
- Operate the device at a lower resistance

6. Conclusion

I have analyzed the optoelectronic characteristics of a new biosensor circuit. I have found that the fractional change in light intensity, δ_I, can approach a 1:1 ratio with the fractional change in resistance, δ_R. Consequently, the biosensor should be feasible without the use of wires.

I have developed code that will model the biosensor circuit. Given 5 measured parameters of a photodiode's condition (R_{Shunt}, R_{Series}, V_{OC}, I_{SC}, and n) the code can generate the photodiode's correct I-V curve. Likewise, given 2 measured parameters of an LED (the turn-on voltage, and internal resistance) the code can generate the correct I-V curve of an LED. Using these curves, the code will optimize the efficiency and normalized change in light $\left(\frac{dI}{\delta_R}\right)$ so that it is easy to choose the correct resistance for the system.

I built a macro-scale prototype of this circuit (13cm x 17cm) to check the code. The prototype demonstrated that the computer code modeled the circuit very well—the two data sets differed by 10 µA on average which is less than 1%. In the system tested, I found that the most light is emitted when there is no resistance, but the greatest normalized change in light occurs at 2.28 kΩ. If this physical prototype were used for neuro-sensing applications, it would use a resistor valued at 2.28 kΩ and reemit 13% of the incident light as an output signal.

This code works, and it gives a positive outlook towards making wireless biosensors, as a 1:1 signal ratio between δ_I and δ_R is very promising. This biosensor has not been created at a micro-scale yet, but if researchers attempt that, they can use this model to simulate and optimize the device's behavior before actually building it.

6.1 Future projects

6.1.1 Maximize graphene's percentage change in resistance

One topic of future research is to find what properties cause the largest change in the fractional change of graphene resistance (δ_R). This project relies on a drop in resistance, so it is necessary to investigate what will optimize that drop. The important part is to optimize the percentage change, as it is easy to change the total resistance by adjusting the size of the graphene.

6.1.2 Increase range by subcutaneous/subdural emitters and receivers

Using an emitter device inside the skull (for example, between the skull bones and the dura mater) would increase the power received by the devices. Likewise, a receiver in the same place would increase the output signal. This is because the power would have less absorbing material to travel through. A topic of future research could be to develop a safe method to install these devices while still keeping spatial resolution.

7. Acknowledgements

Ethan Minot, advisor

Bryan Kenote, code consulting

8. Appendix

8.1 System voltage at various resistances

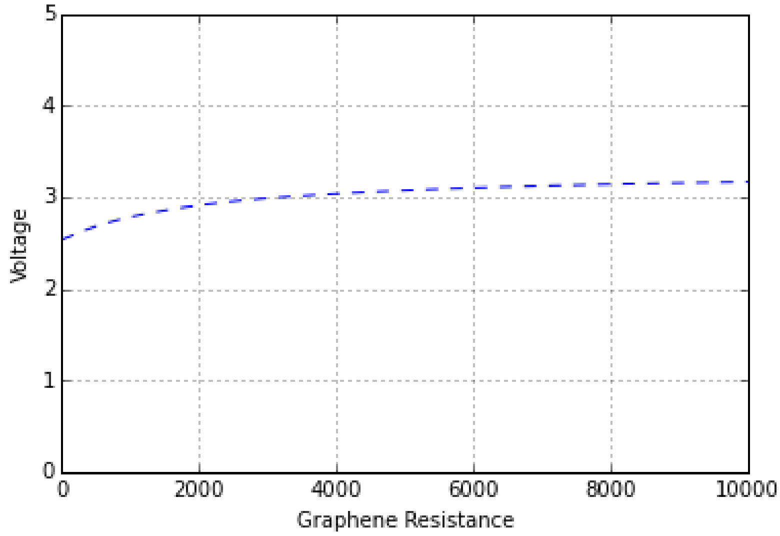

Figure 25: Photodiode's voltage output at various LED/Resistor loads. This is the complementary graph to Figure 21.

8.2 Python code

```python
#!/usr/bin/python
# -*- coding: utf-8 -*-
"""
transmitter_IV.py
Created on Sun Jan 17 15:16:26 2016

@author: Gregg Stevens

This is the main method.  It is dependent on the LED_and_Graphene.py file and
the Solar_cell.py file.
Once you create an LED object, you can call it for the I-V curve associated with it.
"""



import numpy as np #this does lots of mathematical functions.
import matplotlib.pyplot as plt
from Solar_cell import SolarCell
from LED_and_Graphene import LED

#INITIAL RANGE CONSTANTS
minvoltage = .5             #I don't think this one is important.
maxvoltage = 5              #The maximum voltage that the program should ever look f
resolution_steps = 5000     #CHANGE THIS to get a clearer picture.
errortolerance = .000001    #This is used to test that everything works.

#These are the LED experiments with a 5.08 k-Ohm resistor.
ExperimentalIListLED = np.array([0.00000000, 0.00000000, 0.00000020, 0.00000041, 0.0
ExperimentalVListLED = np.array([0.0, 1.1, 1.2, 1.3, 1.5, 1.6, 1.7, 2, 2.2, 2.4, 2.6
led = LED(Rgraphene=5080,Vmax=8,number_of_steps = resolution_steps)

#These are the Solar Cell experimental data
ExperimentalVList = np.array([3.29, 3.065, 2.929, 2.811, 2.688, 2.497, 2.142, 1.622,
ExperimentalIList = np.array([0, 0.0003, 0.00046, 0.00057, 0.00068, 0.00081, 0.001,
sc = SolarCell(n=15, number_of_steps = resolution_steps)

#These are the resistances used when looking at the I-R graph (maximizing intensity,
Rmin = 0000
Rmax = 10000
Rsteps = 100
Rlist = np.linspace(Rmin,Rmax,Rsteps)
R_LED_Internal = 1300

#These are the solutions of the circuit at each resistance value. They are initializ
outputI = np.zeros(len(Rlist))          #This has same length as Rlist
outputV = np.zeros(len(Rlist))          #These are initialized in the main() functi

#These are the arrays that maximize the efficiency. They are initialized in plot_opt
dI_dr = np.zeros(len(Rlist)-1)          #This is one shorter than Rlist because Res
di_dr = np.zeros(len(Rlist)-1)
Res_ave = np.zeros(len(Rlist)-1)        #The average resistance between Rlist[i] ar

def main():
    plot_IR() #This also initializes the values.
    PlotGraph(data=1) #change data to 1 if you don't want the experimental data to s
        #This is the solar cell curve and the LED curve.
    plotExperimentals(data=1,plotSC=1)
           #plot_RV()           #this one is not useful
    compare_Results(data=1)     #compares I-R with experimental data
    calculate_efficiency()
```

```
    #plot_optimization()    #This works, but needs resolution_steps to be really hig
        #plot_resistances()     #This works, but we don't need it. It is a conceptu
                #It shows the effect of series and shunt resistances.
    return

def plot_resistances():                               #External Method
    """This method plots the graph of the Series and Shunt Resistances. Compares
        them with the ideal solar cell."""
    print("plot_resistances")
    a = SolarCell(n=1,number_of_steps = resolution_steps,Rs=1,Rsh=1000000)
    b = SolarCell(n=1,number_of_steps = resolution_steps,Rs=500,Rsh=1000000)
    c = SolarCell(n=1,number_of_steps = resolution_steps,Rs=1,Rsh=100000)
    d = SolarCell(n=1,number_of_steps = resolution_steps,Rs=1000,Rsh=1000000)
    e = SolarCell(n=1,number_of_steps = resolution_steps,Rs=1,Rsh=10000)

    plt.plot(a.Vlist, a.Ilist*1000, 'b--', label= "Ideal Solar Cell")
    plt.plot(b.Vlist, b.Ilist*1000, 'r-', label="Medium Series Resistance (500 Ω )")
    plt.plot(d.Vlist, d.Ilist*1000, 'r:', label="High Series Resistance (1 kΩ)")
    plt.legend( loc='lower left')
    plt.xlabel("Voltage (V)")
    plt.ylabel("Current (mA)")
    plt.grid()
    plt.ylim(0,1.5)
    plt.xlim(0,4)
    plt.show()

    plt.plot(a.Vlist, a.Ilist*1000, 'b--', label= "Ideal Solar Cell")
    plt.plot(c.Vlist, c.Ilist*1000, 'r-', label="Medium Shunt Resistance (100 kΩ)")
    plt.plot(e.Vlist, e.Ilist*1000, 'r:', label="Low Shunt Resistance (10 kΩ)")
    plt.legend( loc='lower left')
    plt.xlabel("Voltage (V)")
    plt.ylabel("Current (mA)")
    plt.grid()
    plt.ylim(0,1.5)
    plt.xlim(0,4)
    plt.show()
    return

def plot_optimization():                     #External Method. Initializes you use di
    """This plots the graph that shows the best point to operate our device.
    You must run optimize_dI_dr() first, or it won't work."""
    optimize_dI_dr()
    print("plot_optimization")
    plt.plot(Res_ave,-dI_dr*1000,'b-', label = "dI/dr")
    plt.legend( loc='lower center')
    plt.xlabel("Graphene Resistance (Ω)")
    plt.ylabel("Absolute Change in Intensity (mA)")
    plt.ylim(0,0.2)
    plt.figure(dpi=100)
    plt.show()

    plt.plot(Res_ave,-di_dr,'b-', label = "di/dr")
    plt.legend( loc='lower center')
    plt.xlabel("Graphene Resistance (Ω)")
    plt.ylabel("Percentage Change in Intensity (unitless)")
    plt.ylim(0,1)
    plt.figure(dpi=100)
    plt.show()

def optimize_dI_dr():                          #Internal Method
```

```
    print("optimize_dI_dr")
    for a in range(0,len(Rlist)-1):
        dI = outputI[a+1] - outputI[a]
        dR = Rlist[a+1] - Rlist[a]
        R = (Rlist[a+1] + Rlist[a]) / 2
        I = (outputI[a+1] + outputI[a]) / 2

        dI_dr[a] = R*dI/dR
        di_dr[a] = R/I*dI/dR
        Res_ave[a] = R
    return

def calculate_efficiency():                         #External Method
    """Return a graph of the input and output power.

    """
    print("calculate_efficiency")
    #Define output power as IV - I2R over the two resistors
    power_IN = np.ones(len(Rlist)) * sc.IoutSC * sc.VoutOC
    power_SC = outputI*outputV
    power_LED = outputI*outputV-outputI*outputI*(R_LED_Internal+Rlist)
    plt.plot(Rlist,power_IN*1000, label = "Input Power")
    plt.plot(Rlist,power_SC*1000, label = "Solar Cell Power")
    plt.plot(Rlist,power_LED*1000, label = "Output Power")
    plt.legend( loc='center right')
    plt.xlabel("Graphene Resistance")
    plt.ylabel("Power (mW)")
    #plt.grid()
    plt.show()
    return

def compare_Results(data=0):                        #External
    print("compare Results")
    plt.plot(Rlist,outputI*1000,'b--', label = "Model")
    resultR = np.array([5100,4300,2000,8200,0])
    resultI = np.array([.0002369,.0002579,.000425,.0001535,.00075])
    if(data==1):
        plt.plot(resultR,resultI*1000,'bs', label = "Experimental Data")
    plt.legend( loc='upper right')
    plt.xlabel("Graphene Resistance (Ω)")
    plt.ylabel("Current (mA)")
    plt.ylim(0,1)
    plt.show()

def plot_IR():
    Ilist = findIsfromRs()
    plt.plot(Rlist,Ilist*1000, 'b--', label = "Total Current")
    plt.legend( loc='upper right')
    plt.xlabel("Graphene Resistance (Ω)")
    plt.ylabel("Current (mA)")
    plt.grid()
    plt.ylim(0,1)
    plt.show()
    return

def plot_RV():
    Vlist = outputV
    plt.plot(Rlist,Vlist, 'b--')
    plt.legend( loc='upper right')
    plt.xlabel("Graphene Resistance")
```

```
    plt.ylabel("Voltage")
    plt.grid()
    plt.ylim(0,5)
    plt.show()
    return
def plot_IV_FindIntersect():        #External
    print("This is the plot_IV_FindIntersect method")
    if(checkError(led.check) and checkError(sc.check)):
        PlotGraph()
        intersect = findIntersectIndex(led,sc)
        intersectionI = sc.Ilist[intersect]
        intersectionV = sc.Vlist[intersect]
        print("the intersection point is at (V,I) = (%a, %a)" % (intersectionV,inter
    return

###This will plot I vs. V
def PlotGraph(a=led,b=sc, data=0):                          #External Method
    print("This is the PlotGraph method")
    plt.plot(a.Vlist, a.Ilist*1000, 'b--', label= "LED")
    plt.plot(b.Vlist, b.Ilist*1000, 'r--', label="Solar Cell")
    if(data==1):
        plt.plot(ExperimentalVList,ExperimentalIList*1000,'rs', label = "Solar Data'
        plt.plot(ExperimentalVListLED,ExperimentalIListLED*1000, 'bs', label = "LED
    plt.legend( loc='upper right')
    plt.xlabel("Voltage (V)")
    plt.ylabel("Current (mA)")
    plt.grid()
    plt.ylim(0,1.5)
    plt.xlim(0,4)
    plt.show()
    return

def findIntersectIndex(object1, object2):               #Internal method
    """Returns the index for the intersection point of the two IV curves.
    :param object1: must be an object with a CURRENT LIST named Ilist.
    :param object2: must be an object with a CURRENT LIST named Ilist.
    The two current lists must be the same size. It makes sense to just use
    the parameter resolution_steps as the length of Ilist."""
    intersect = np.argmin(np.abs(object1.Ilist - object2.Ilist))
    return intersect

def checkError(mylist):                             #Internal method
    """Are there problems with the I-V curve?"""
    outputboolean = 0
    if all(abs(j) < errortolerance for j in mylist):
        outputboolean = 1
        print("The error is tolerable--that means the I-V list is good.")
        print("That means you can ignore the error encountered on line ~86.")
    else:
        print("Something went wrong! The error is not tolerable.")
        for j in mylist:
            if(abs(j) >= errortolerance):
                print("The index is wrong at %a" %j)
        print(mylist)

    return outputboolean

def plotExperimentals(data=0, plotSC=1):                        #External methoc
    #These are the LED experiments WITHOUT a resistor.
    ExperimentalIListLED0 = np.array([ 0.0000002 ,  0.0000119 ,  0.0001413 ,  0.000(
```

```
    ExperimentalVListLED0 = np.array([1.3, 1.5, 1.8, 2.1, 2.5, 3.3, 3.8, 4.2, 5.1, 6
    led0 = LED(Rgraphene=0,Vmax=4,number_of_steps = resolution_steps)

    #These are the LED experiments with a 4.2 k-Ohm resistor.
    ExperimentalIListLED4 = np.array([0.00000166 ,  0.00005202 ,  0.00010356 ,  0.00
    ExperimentalVListLED4 = np.array([1.4, 1.8, 2.1, 2.5, 2.9, 3.1, 3.4, 3.7, 4, 4.4
    led4 = LED(Rgraphene=4200,Vmax=8,number_of_steps = resolution_steps)

    #These are the LED experiments with a 5.08 k-Ohm resistor.
    ExperimentalIListLED5 = np.array([0.00000000, 0.00000000, 0.00000020, 0.00000041
    ExperimentalVListLED5 = np.array([0.0, 1.1, 1.2, 1.3, 1.5, 1.6, 1.7, 2, 2.2, 2.4
    led5 = LED(Rgraphene=5080,Vmax=8,number_of_steps = resolution_steps)

    #These are the LED experiments with a 6.15 k-Ohm resistor.
    ExperimentalIListLED6 = np.array([ 0.000000146 ,  0.000016585 ,  0.000032000 ,
    ExperimentalVListLED6 = np.array([1.2, 1.6, 1.7, 2.2, 2.6, 2.8, 3, 3.2, 3.4, 3.7
    led6 = LED(Rgraphene=6150,Vmax=8,number_of_steps = resolution_steps)

    #These are the LED experiments with a 7.44 k-Ohm resistor.
    ExperimentalIListLED7 = np.array([ 0.000000027 ,  0.000003495 ,  0.000015726 ,
    ExperimentalVListLED7 = np.array([1.2, 1.4, 1.6, 2, 2.3, 2.5, 2.8, 2.9, 3.1, 3.3
    led7 = LED(Rgraphene=7440,Vmax=8,number_of_steps = resolution_steps)

    plt.plot(led0.Vlist, 1000*led0.Ilist, 'b--', label= "R=0")
    if(data==1): plt.plot(ExperimentalVListLED0,1000*ExperimentalIListLED0,'bs', lab

    plt.plot(led4.Vlist, 1000*led4.Ilist, 'r--', label="R=4kΩ")
    if(data==1): plt.plot(ExperimentalVListLED4,1000*ExperimentalIListLED4,'rs', lab

    plt.plot(led5.Vlist, 1000*led5.Ilist, 'g--', label="R=5kΩ")
    if(data==1): plt.plot(ExperimentalVListLED5,1000*ExperimentalIListLED5,'gs', lab

    plt.plot(led6.Vlist, 1000*led6.Ilist, 'k--', label="R=6kΩ")
    if(data==1): plt.plot(ExperimentalVListLED6,1000*ExperimentalIListLED6,'ks', lab

    plt.plot(led7.Vlist, 1000*led7.Ilist, 'm--', label="R=7kΩ")
    if(data==1): plt.plot(ExperimentalVListLED7,1000*ExperimentalIListLED7,'ms', lab

    if(plotSC==1):
        plt.plot(sc.Vlist,sc.Ilist*1000, label = "Solar Model")

    plt.legend( loc='lower right')
    plt.xlabel("LED Voltage (V)")
    plt.ylabel("LED Current (mA)")
    plt.grid()
    plt.ylim(0,1.5)
    plt.xlim(0,14)
    plt.show()
    return

def findIsfromRs(Rrange = Rlist, solarcell = sc):         #INTERNAL METHOD
    """This takes in a list of resistances and gives back the currents that
    satisfy the circuit.
    :param Rrange: the list of currents
    :param solarcell: the solar cell in the other part of the circuit"""
    if(checkError(solarcell.check)):

        #create led_list
        led_list = []
        for R in Rlist:
```

```python
            led_list.append(LED(Rgraphene = R,Rled = R_LED_Internal, Vmax = maxvolta

        #find intersection of graphs
        for a in range(0,len(Rlist)):
            if(checkError(led_list[a].check)):  #make sure that the LED list is gooc
                index = findIntersectIndex(led_list[a],solarcell)
                outputI[a] = led_list[a].Ilist[index]
                outputV[a] = led_list[a].Vlist[index]

    return outputI

if __name__ == "__main__":
    main()
```

```python
#!/usr/bin/python
# -*- coding: utf-8 -*-
"""
LED_and_Graphene.py
Created on Thu Jul  9 22:55:38 2015

@author: Gregg Stevens

This contains three important lists:
Vlist  (Volts)
Ilist  (milli-amps)
check  (should be close to 0's)

Once you create an LED object, you can call it for the I-V curve associated with it.

"""

import numpy as np #this does lots of mathematicl functions.
import scipy.optimize as sp #this solves the transcedental equations.
import matplotlib.pyplot as plt

class LED:
    #Absolute constants in equation
    kT = 1#1.38064852e-4 * 310.15 # eV at room temp
    q = 1#1.60217662 # C -- elementary charge
    def __init__ (self, Rgraphene=5080, Rled = 1300, n=2, Vmin = 0, Vmax = 5, number
        #Cell-specific constants
        self.n = n #Unitless recombination factor
        self.I0 = 10**-18 # Amps
        self.Reff = Rgraphene + Rled #k-Ohms
        self.number_of_steps = number_of_steps
        self.VBreak = VBreak #x-intercept--voltage at which current starts (linear m

        self.Vlist = np.linspace(Vmin, Vmax, number_of_steps)
        self.Ilist = np.zeros(number_of_steps)            ## MILLI-AMPS
        self.check = np.ones(number_of_steps)

        self.initialize_current_list(self.Vlist,self.Ilist,self.check)

    def initialize_current_list(self, voltages, currents, checklist ):
        """This function initializes a given list of currents.
        :param led_object: This is the object for this current list.
        :param voltages: This is the master list of voltages (x-axis).
        :param currents: This will be the output (currents).
        :param checklist: This makes sure that the output is close enough."""
        for a in range(0,len(voltages)):
            currents[a] = self.find_I_from_V(voltages[a])
            checklist[a] = self.checkIV(currents[a],voltages[a])
        return
    def find_I_from_V(self, V,limitlow=-1,limithigh=1):
        """Give it your voltage, and it spits out current.
        :param V: This is the voltage you pass into it.
        :param limitlow: You must give a range to the current. This is the minimum.
        :param limithigh: This is the maximum current."""
        #This is the function for the current of the Solar Cell
        def funct(I): return self.checkIV(I,V)
        # Use the numerical solver to find the roots
        I_solution = sp.brentq(funct, limitlow,limithigh, maxiter=500000) #This solv
        return I_solution
```

```
def plot(self):
    plt.plot(self.Vlist,self.Ilist)
    return

def checkIV(self,I,V):
    """This is the function. It will return 0 if the I-V combination works out.'
    #return V-I*self.Reff  #This is my test function...a cubic function.
    #old version of function
    #return -V + self.n*self.kT/self.q*(np.log(1+I/self.I0)) + I*self.Reff
    return -V + I * self.Reff + self.VBreak
```

```python
#!/usr/bin/python
# -*- coding: utf-8 -*-
"""
Solar_cell.py
Created on Thu Jul  9 22:55:38 2015

@author: Gregg Stevens
"""

import numpy as np #this does lots of mathematictl functions.
import scipy.optimize as sp #this solves the transcedental equations.
class SolarCell:
    #ABSOLUTE CONSTANTS IN EQUATION
    kT = 1.38064852e-4 * 310.15 # eV at room temp
    q = 1.60217662 # C -- elementary charge
    def __init__ (self, n=2, Rs=579, Rsh=62500, VoutOC=3.29, IoutSC=0.00141, Vmin =
        kT = SolarCell.kT
        q = SolarCell.q
        #Cell-Specific Constants
        self.n = n # ideality factor
        self.Rs = Rs # Ohms
        self.Rsh = Rsh # Ohms
        self.VoutOC= VoutOC #Volts
        self.IoutSC= IoutSC #Amps
        self.Idiode0 = (IoutSC-(VoutOC-IoutSC*Rs)/Rsh)/(np.exp(q*VoutOC/n/kT)-np.exp
        self.Ilight = self.Idiode0 * np.exp(q*VoutOC/n/kT) + VoutOC / Rsh # Amps

        self.Vlist = np.linspace(Vmin, Vmax, number_of_steps)
        self.Ilist = np.zeros(number_of_steps)              ## MILLI-AMPS
        self.check = np.ones(number_of_steps)

        self.initialize_current_list(self.Vlist,self.Ilist,self.check)

    def initialize_current_list(self, voltages, currents, checklist ):
        """This function initializes a given list of currents.
        :param led_object: This is the object for this current list.
        :param voltages: This is the master list of voltages (x-axis).
        :param currents: This will be the output (currents).
        :param checklist: This makes sure that the output is close enough."""
        for a in range(0,len(voltages)):
            currents[a] = self.find_I_from_V(voltages[a])
            checklist[a] = self.checkIV(currents[a],voltages[a])
        return

    def find_I_from_V(self, V,limitlow=-500000,limithigh=1000000):
        """Give it your voltage, and it spits out current.
        :param V: This is the voltage you pass into it.
        :param limitlow: You must give a range to the current. This is the minimum.
        :param limithigh: This is the maximum current."""
        #This is the function for the current of the Solar Cell
        def funct(Iout): return self.checkIV(Iout,V)
        # Use the numerical solver to find the roots
        Iout_solution = sp.brentq(funct, limitlow,limithigh, maxiter=50000) #This sc
        return Iout_solution

    def checkIV(self, I,V):
        """This is the function. It will return 0 if the I-V combination works out.
        :param I: The Current MUST BE IN AMPS."""
        #return I - V**3 +.0000001  #This is my test function...a cubic function.
        return self.Ilight - I - self.Idiode0*(np.exp(self.q/self.n/self.kT*(V+I*sel
```

9. References

[1] "Neural Dust: An Ultrasonic, Low Power Solution for Chronic Brain-Machine Interfaces," D. Seo, J. M. Carmena, J. M. Rabaey, E. Alon, and M. M. Maharbiz, arXiv **1307**.2196 (2013).

[2] "A Review of Organic and Inorganic Biomaterials for Neural Interfaces," P. Fattahi, G. Yang, G. Kim, and M. R. Abidian, Adv. Mater. **26**, 12 (2014).

[3] "Laser Safety Handbook" *Northwestern University: Office for Research Safety*. (2011). http://www.research.northwestern.edu/ors/forms/laser-safety-handbook.pdf

[4] "Simulation of Near-Infrared Light Absorption Considering Individual Head and Prefrontal Cortex Anatomy: Implications for Optical Neuroimaging," F. B. Haeussinger, S. Heinzel, T. Hahn, M. Schecklmann, A. Ehlis, and A. J. Fallgatter, PLoS ONE **6**, 10 (2011).

[5] "Injectable, Cellular-Scale Optoelectronics with Applications for Wireless Optogenetics," T. Kim, et. al, Science **340**, 6129 (2013).

[6] "Graphene Microelectrode Arrays for Neural Activity Detection," X. Du, L. Wu, J. Cheng, S. Huang, Q. Cai, Q. Jin, and J. Zhao, J. Biol. Phys. **41**, 4 (2014)

[7] "A Simple Photodiode Series Resistance Measurement Method," J. Cabestany and X. Castañer, Revue de Physique Appliquee **18**, 9 (1983).

[8] "A Method for the Direct Measurement of Photodiode Shunt Resistance," D. Chan and J Phang, IEEE Trans Electron Devices **31**, 3 (1984).