Experiment Activity 3: Photoelectric Effect: Measuring Planck's Constant

Data Collection

We collected data in an Excel sheet, see github

Problem 1.

Dark Current Characteristic Curve Data

Current $I_D \pm \delta I_D$	Voltage $V_D \pm \delta V_D$

Photocurrent Characteristic Curve Data

Filter 1		Filter 2	
$\lambda =$		$\lambda =$	
Current $I_1 \pm \delta I_1$	Voltage $V_1 \pm \delta V_1$	Current $I_2 \pm \delta I_2$	Voltage $V_2 \pm \delta V_2$

Photocurrent Characteristic Curve Data

Filter 3		Filter 4	
$\lambda =$		$\lambda =$	
Current $I_3 \pm \delta I_3$	Voltage $V_3 \pm \delta V_3$	Current $I_4 \pm \delta I_4$	Voltage $V_4 \pm \delta V_4$

Photocurrent Characteristic Curve Data

Filter 5	
$\lambda =$	
Current $I_5 \pm \delta I_5$	Voltage $V_5 \pm \delta V_5$

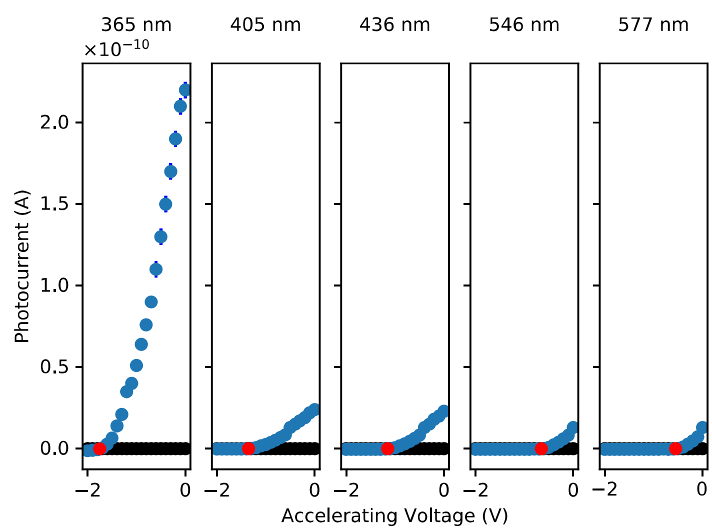
Problem 2. Plotting the Data & Finding the Stopping Voltages

Take the data you collected in the dark current and photocurrent tables in Problem 1 and (if you haven't done so already) enter all this data into several `*.csv` files (one for the dark current data and one for the photocurrent data for each interference filter or LED, depending on the model of the Planck's constant apparatus you used).

(a.) Write a Python program to plot the photocurrent as a function of voltage for each interference filter/LED. These are the photocurrent characteristic curves corresponding to each interference filter/LED wavelength. On top of each photocurrent plot, overlay a plot of the dark current characteristic curve. You will use this information in Part (b.) to determine the stopping potential for each wavelength. You will need to add to and/or otherwise modify this code to help answer Problem 8.

Make sure your plots are appropriately labeled and include a legend. (Follow the *Guidelines for Plotting Data* given in the appendix ***Data Analysis & Visualization***.)

Your plots should look very similar to the the subplots in the figure below:



Photocurrent Characteristic Curves for Various Wavelengths

Note that your plots will only be formatted together in one figure like this if you spend the time playing with Python and researching on the internet to learn how to combine them. For this lab activity, one plot figure for each set of interference filter/LED data (overlaid with the dark current data) in separate figures, with appropriate labels (title and axes), is all that is required for credit. However, if you do take the time to learn how to present your photocurrent plots in the above way or similarly – as long as the collective figure looks nice – you will receive extra credit.

(b.) The stopping potential for each wavelength will be the point in each filter/LED plot where the photocurrent characteristic curve intersects the dark current characteristic curve. We need to write some code to find the intersection point between our photocurrent and dark current curves.

Add a new function called `DataIntersectionPoint` to your `DataAnalysisTools.py` file – which you should have made in *Activity 5* (*Data Analysis & Visualization in Python 3*). The goal of this function will be to find the point where two data sets intersect.

In the following box, you will find a program written in Python-like ***pseudocode*** which reads in from files two data curves y_1 and y_2 which share the same set of x values, plots the two curves in the same plot, and finds and prints their intersection point with uncertainty. **Refactor this pseudocode into a working function.** This function will be your `DataIntersectionPoint` function and it should take in two data sets and their corresponding set of uncertainties as input and return the intersection point and its corresponding uncertainty.

```
# Assume x data from data1.txt is the same as the x data from data2.txt
x_data = np.loadtxt('data1.txt', usecols=0)
x_unc = np.loadtxt('data1.txt', usecols=1)

# y1_data is the list of y values for the first curve.
y1_data = np.loadtxt('data1.txt', usecols=3)
y1_unc = np.loadtxt('data1.txt', usecols=4)

# y2_data is the list of y values for the second curve.
y2_data = np.loadtxt('data2.txt', usecols=3)
y2_unc = np.loadtxt('data2.txt', usecols=4)

# Plot the data: one curve on top of the other.
plt.errorbar(x_data, y1_data, yerr=y1_unc, xerr=x_unc, fmt='o',color='k')
plt.errorbar(x_data, y2_data, yerr=y2_unc, xerr=x_unc, fmt='o',color='b')

# Make a list of all the indices of points where one curve crosses another.
idx = np.argwhere(np.diff(np.sign(y1_data - y2_data)) != 0).flatten()

# Find all the intercepts.
for i in range(len(idx)):
    # The real x-intercepts are probably between data points, so we
    # estimate the value of the intercept by interpolation:
    xInt[i] = (x_data[idx[i]]+x_data[idx[i]+1])/2

    # Because the x-intercept values are in between data points
    # (x_data[idx[i]] and x_data[idx[i]+1]), the uncertainty associated
    # with each estimated x-intercept, xInt_unc[i], will be the whole
    # range from x_data[idx[i]]-x_unc[idx[i]] to
```

```
    # x_data[idx[i]+1]+x_unc[idx[i]+1]:
    xInt_unc[i] = abs(x_data[idx[i]+1]-x_data[idx[i]]
        + x_unc[idx[i]+1]+x_unc[idx[i]])

    # y-intercepts and their uncertainties are computed similarly to
    # the x-intercepts and their uncertainties:
    yInt[i] = (y1_data[idx[j]+1]+y1_data[idx[j]]
        + y2_data[idx[j]+1]+y2_data[idx[j]])/4
    yInt_unc[i] = abs(y1_data[idx[i]+1]-y1_data[idx[i]]
          + y2_data[idx[i]+1]-y2_data[idx[i]]
            + y1_unc[idx[i]+1]+y1_unc[idx[i]]
                +y2_unc[idx[i]+1]+y2_unc[idx[i]])/2


# Plot the intercept points on top of the curves:
plt.errorbar(xInt, yInt, yerr=yInt_unc, xerr=xInt_unc, fmt='o',color='r')

# Print the values of the x- and y-intercepts (with uncertainties) as
# ordered pairs.
# Note: the string \u00B1 is the unicode representation of the
# "plus-or-minus" symbol we use for uncertainties.
# If you don't want to use unicode, just use " +/- " instead.
print("("+str(xInt)+" \u00B1 "+str(xInt_unc)+", \n"
+str(yInt)+" \u00B1 "+str(yInt_unc)+")")

# Show full plot
plt.show()
```

We used our own algorithm, because our x-data was non-uniform. See DataAnalysis.py for details.

(c.) Use the Python function you wrote in Part (b.) to find the stopping potential for each wavelength. Use the following table to report the values you found (or make sure you submit code that prints these values.).

Assume $\delta\lambda_B = 5\%$

	Wavelength: $\lambda \pm \delta\lambda$	Stopping Potential: $V_s \pm \delta V_s$
Digital	611nm (6100 ± 300) Å	0.100 ± 0.0005 V
	585nm (5900 ± 300) Å	0.165 ± 0.0008 V
	525nm (5300 ± 300) Å	0.490 ± 0.003 V
	505nm (5000 ± 300) Å	0.511 ± 0.003 V
	472nm (4720 ± 200) Å	0.671 ± 0.003 V
Analog	5770Å (5800 ± 300) Å	−1.73 ± 0.07 V
	5461Å (5500 ± 300) Å	−1.39 ± 0.06 V
	4358Å (4400 ± 200) Å	−1.14 ± 0.06 V
	4047Å (4000 ± 200) Å	−0.61 ± 0.05 V
	3650Å (3600 ± 200) Å	−0.50 ± 0.04 V

Problem 3. Determining Planck's Constant via Linear Regression

(a.) Using the values of the stopping potential you found for each wavelength in the previous problem, write a Python program to plot the stopping potential as a function of frequency. Make sure your plot includes a title, axes labels, and error bars. Save and modify this code to help you answer Problem 9.

(b.) Use the plot from part (a.) to determine Planck's constant. Think about the following question: How can we use this plot to determine Planck's constant if we know that the maximum kinetic energy of a free electron is $KE_{\max} = hf - \phi$ and the current will drop to its minimum when the electric potential energy hill is high enough that no electrons will have sufficient kinetic energy to overcome it? Recall that electric potential energy U for an electron in a potential is given by $U = eV$. Remember that your determination of Planck's constant should include an uncertainty: $h \pm \delta h$.

<u>Hint:</u> This should look familiar from the *Data Analysis & Visualization* activity.

For these problems, we use the PL(N)(d) data set, as it has significantly lower uncertainty.

(a) Refer to the semi-report for graph, and the github for code.

slope of graph (a), m

(b) $eV_s = KE_{max} \Rightarrow V_s = \frac{h}{e} f - \frac{\phi}{e}$

This implies that

$$h = m \cdot e, \quad \delta h = \sqrt{\left(\frac{\partial h}{\partial m}\delta m\right)^2 + \left(\frac{\partial h}{\partial e}\delta e\right)^2}$$

$$= \sqrt{e^2 \delta m^2 + m^2 \delta e^2}$$

where e = elementary charge, and m is the slope of the line of best fit.

Running this in Python yields $\boxed{h = (6.62 \pm 0.02) \cdot 10^{-34}\ J \cdot s}$

Problem 4. Determining Measurement Accuracy

According to the National Institute of Standards and Technology (NIST), the value of Planck's constant is exactly

$$h = 6.626\ 070\ 15 \times 10^{-34}\ J \cdot s.$$

$\delta h_{nist} = 0$

(See: `http://physics.nist.gov/cgi-bin/cuu/Value?h`)

(a.) Does the value of h which you found in the previous problem agree with the accepted value reported by NIST? Show your work!

$|h_{exp} - h_{nist}| = 6.07015 \times 10^{-37} = \Delta h$

$|\delta h_{exp} + \delta h_{nist}| = |0.02 \cdot 10^{-34} + 0| = 2 \cdot 10^{-36} = \delta$

$\Delta h < \delta$, so the discrepancy is insignificant.

h and h_{nist} agree

(b.) What is the percent difference between your value for Planck's constant and the accepted value?

$$\left| \frac{h_{exp} - h_{nist}}{h_{nist}} \right| \cdot 100 = \boxed{0.103\ \%}$$

Problem 5. Determining the Work Function & Electron Velocity

(a.) Use your plot of stopping potential versus frequency of incident light (as well as the mathematical analysis that you used to determine Planck's constant) to determine the work function of the device's cathode (with uncertainty).

$$V_S = \frac{h}{e} f - \frac{\phi}{e}$$ ← b from our data fit

$$\frac{\phi}{e} = b \Rightarrow \phi_{best} = be = -3.09 \cdot 10^{-19}\ V$$

$$\delta\phi = \sqrt{e^2 \delta b^2 + b^2 \delta e^2} = 1 \cdot 10^{-21}\ V$$

$$\phi = \phi_{best} \pm \delta\phi = (-3.09 \pm 0.01) \cdot 10^{-21}\ V$$

(b.) Compute the maximum velocity with which the electrons escape from the cathode for each wavelength of light you used.

$$KE_{max} = hf - \phi = \frac{1}{2} m_e v^2 \Rightarrow v = \sqrt{\frac{2hf - 2\phi}{m_e}}$$

$$\delta v = \sqrt{\left(\frac{\partial v}{\partial h}\delta h\right)^2 + \left(\frac{\partial v}{\partial \phi}\delta\phi\right)^2 + \left(\frac{\partial v}{\partial f}\delta f\right)^2}$$

$$= \sqrt{\left(\frac{f}{m_e}\left(\frac{2h-2\phi}{m_e}\right)^{-1/2}\delta h\right)^2 + \left(\frac{h}{m_e}\left(\frac{2h-2\phi}{m_e}\right)^{-1/2}\delta f\right)^2 + \left(-\frac{1}{m_e}\left(\frac{2h-2\phi}{m_e}\right)^{-1/2}\right)^2}$$

Results:

611 nm: $(1.180 \pm 0.002) \cdot 10^6$ m/s

588: $(1.192 \pm 0.002) \cdot 10^6$ m/s

525: $(1.228 \pm 0.002) \cdot 10^6$ m/s

505: $(1.242 \pm 0.002) \cdot 10^6$ m/s

472: $(1.266 \pm 0.002) \cdot 10^6$ m/s

Design Considerations

Problem 6. Improving the Experiment

Consider the sources of uncertainty required for your computation of Planck's constant in this experiment. Suppose you are designing a new Planck's Constant Apparatus based on the particular model you used in this experiment (or perhaps you're simply upgrading the model you used in the experiment). Based on your uncertainty analysis, what changes could you make to the apparatus to achieve a more accurate estimate for Planck's constant (as compared to the NIST value)?

Analog Apparatus could be improved a lot:

- Fix the distance between lamp and lens
- Clean the lenses
- Reduce warm-up time
- Make Voltage source easier to control
- Make current measurement easier by automatically scaling