MoS2 Background

1 Introduction to LEED Background Subtraction using Python

1.0.1 Before getting started, use the PLEASE software package to extract the I(V) curve you are interested in along with one or more background I(V) curves. PLEASE can automate the background selection and extraction as well as write the curves to files.

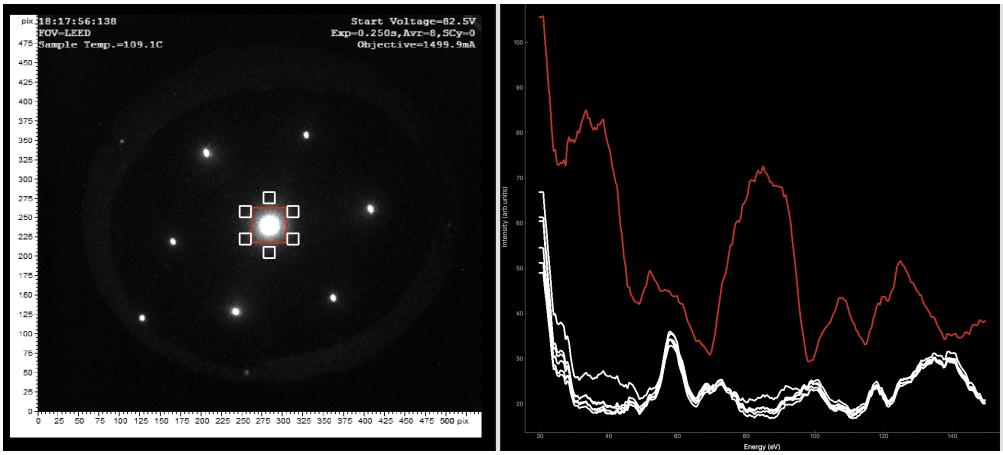
1.0.2 This guide assumes that in the current directory of this notebook there are at minimum two text files, the first being the beam intensity output from PLEASE, the second and beyond are the background intensity output from PLEASE.

Let's begin by importing the plotting libraries and setting the color scheme. The fourth line here can use any available color scheme. A list of available color schemes can be found by executing `print(plt.style.available)`

```
In [1]: import matplotlib.pyplot as plt
        import matplotlib.cm as cm
        %matplotlib inline  # Show plots inline with the notebook
        plt.style.use("seaborn-pastel")
```

Next we import the libraries needed to grab the files from the current working directory and make a list of the paths to the required textfiles.

```
In [2]: import glob
        import os

        files = glob.glob(os.path.join(os.getcwd(), "*.txt"))

In [3]: files

Out[3]: ['/Users/Maxwell/Dropbox/MoS2/MoS2-LEED-Background/MoS2-00-LEEDbeam_0.txt',
         '/Users/Maxwell/Dropbox/MoS2/MoS2-LEED-Background/MoS2-00-LEEDbeam_0bkgd_0.txt',
         '/Users/Maxwell/Dropbox/MoS2/MoS2-LEED-Background/MoS2-00-LEEDbeam_0bkgd_1.txt',
         '/Users/Maxwell/Dropbox/MoS2/MoS2-LEED-Background/MoS2-00-LEEDbeam_0bkgd_2.txt',
         '/Users/Maxwell/Dropbox/MoS2/MoS2-LEED-Background/MoS2-00-LEEDbeam_0bkgd_3.txt',
         '/Users/Maxwell/Dropbox/MoS2/MoS2-LEED-Background/MoS2-00-LEEDbeam_0bkgd_4.txt',
         '/Users/Maxwell/Dropbox/MoS2/MoS2-LEED-Background/MoS2-00-LEEDbeam_0bkgd_5.txt']
```



LEED-I(V) and Background from PLEASE

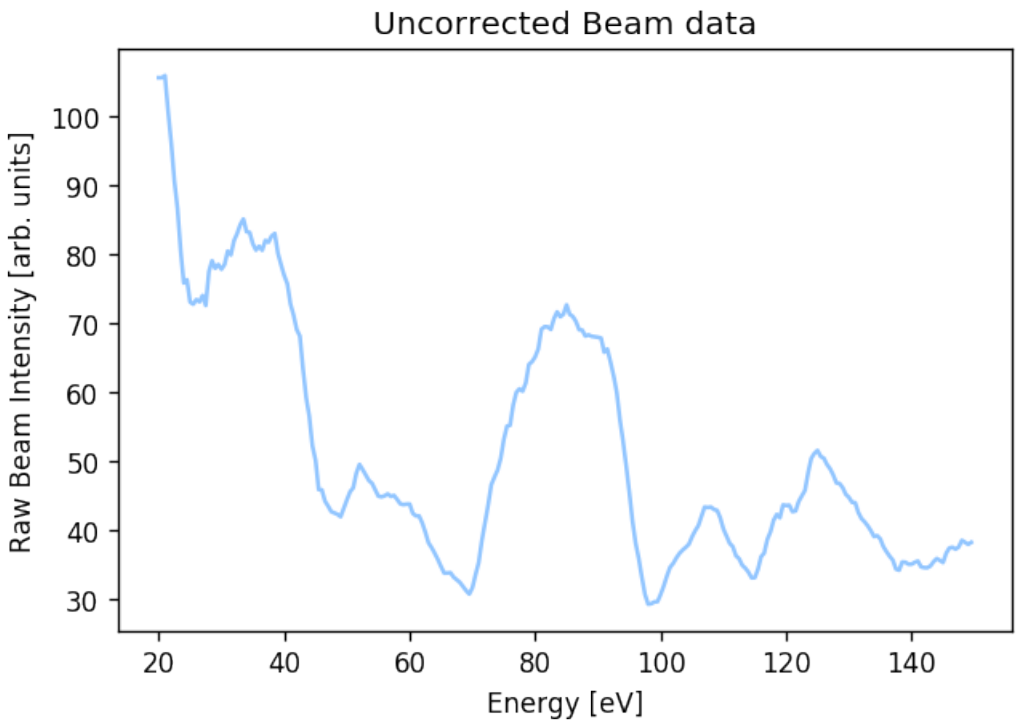
We can see that the contents of the files list are paths to various text files. There is one text file corresponding to the Beam data and then six text files corresponding to local background data. To show how we got these files, the first figure shows a screenshot of the LEED-I(V) data set and the automated background extraction accomplished by the PLEASE software package.

Now we need to grab some information from the first file, which contains the beam data for the (00) beam, the red curve in the first figure. First we create two lists for Energy Values and Beam Intensity Values. Then we parse the first text file for the required data. The text files from PLEASE are output as tab-delimited files with Energy in the first column and Intensity in the second column. So we can simply read the lines of the text file, discard the first header line, and split the lines bases on the TAB character to grab the relevant information.

```
In [4]: # Get Energy values and (00) Beam Intensity Values
        Energy = []
        Beam_Intensity = []
        with open(files[0], 'r') as f:
            lines = f.readlines()[1:]  # discard header
        for line in lines:
            Energy.append(float(line.split("\t")[0]))
            Beam_Intensity.append(float(line.split("\t")[1]))
```

To double check that we got the correct data, let's plot the raw I(V) curve using the two lists we just created

```
In [5]: plt.figure(dpi=120)
        plt.plot(Energy, Beam_Intensity)
        plt.xlabel("Energy [eV]")
        plt.ylabel("Raw Beam Intensity [arb. units]")
        plt.title("Uncorrected Beam data")

Out[5]: <matplotlib.text.Text at 0x11c59b860>
```



Next we need to get the intensity of the background beams as output by PLEASE, which are the white curves in the first figure. We are going to sum the intensity for each energy step then average by the number of beams at the end. We start by creating a list of zeros with the same length as Beam_Intensity. Then we add the intensity parsed from the text files at each step through energy space.

```
In [6]: Background_Intensity = [0]*len(Beam_Intensity)
```

```
In [7]: for fl in files[1:]:  # loop over all background files
            with open(fl, 'r') as f:
                lines = f.readlines()[1:]  # discard header
            for idx, line in enumerate(lines):
                # Sum intensity for all background beams
                Background_Intensity[idx] += float(line.split("\t")[1])
```

What have now is a list of the total background intensity from all beams output by PLEASE. Next, we want to average the list by the number of beams to get the average background intensity.

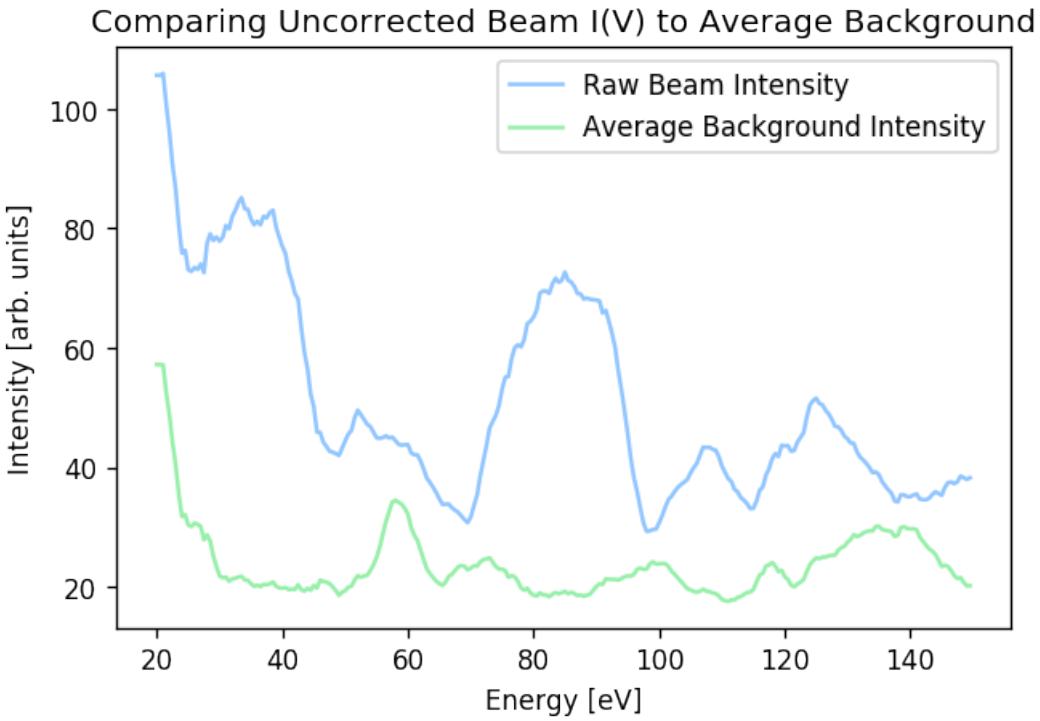
```
In [8]: # Average the total intensity by the number of background beams (6)
        Background_Intensity = [i/6 for i in Background_Intensity]
```

Again, let's double check that this is correct by plotting the raw Beam Intensity and the Average Background Intensity.

```
In [9]: fig, ax = plt.subplots(nrows=1, ncols=1, dpi=120)
        ax.plot(Energy, Beam_Intensity, label="Raw Beam Intensity")
        ax.plot(Energy, Background_Intensity, label="Average Background Intensity")
```

```
ax.legend()
plt.xlabel("Energy [eV]")
plt.ylabel("Intensity [arb. units]")
plt.title("Comparing Uncorrected Beam I(V) to Average Background")
```

Out[9]: <matplotlib.text.Text at 0x11c8d9358>

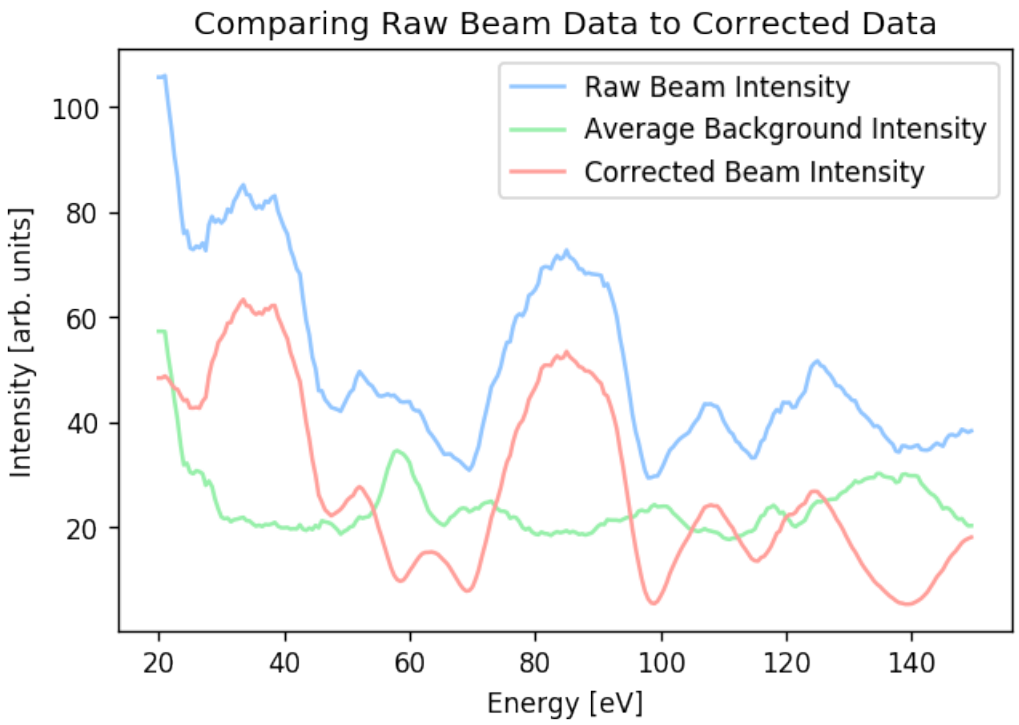


Finally, let's perform a simple background subtraction by taking the Raw Beam Intensity and subtracting the Average Background Intensity for each step through energy space.

In [10]:
```
# Subtract Background from Beam
Corrected_Beam_Intensity = [Beam_Intensity[idx] - Background_Intensity[idx]
                            for idx, _ in enumerate(Beam_Intensity)]
```

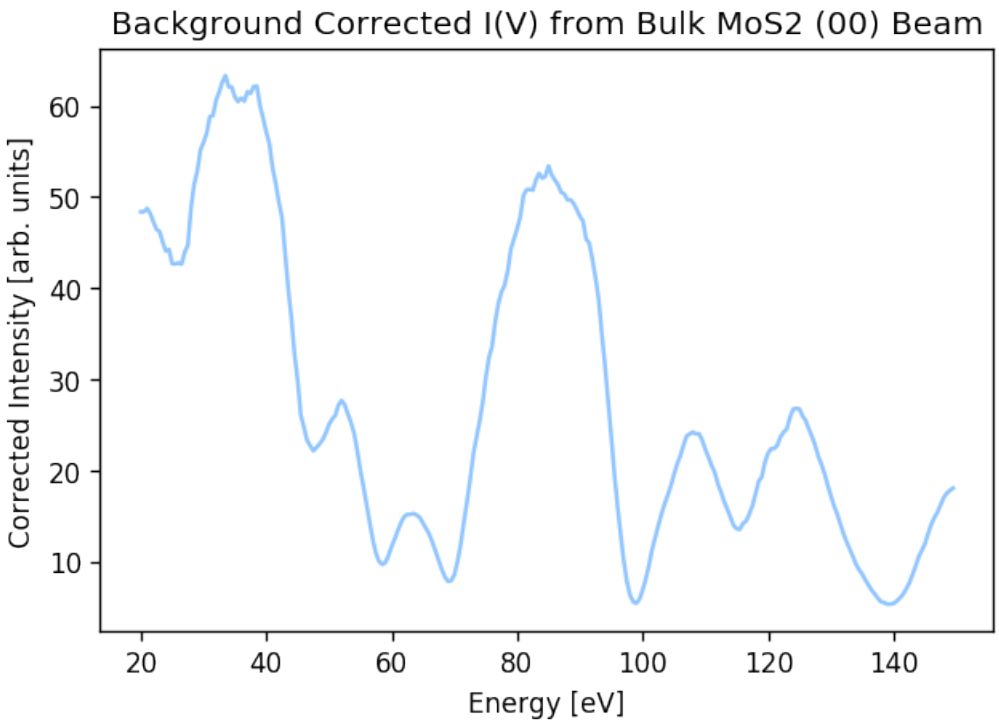
In [11]:
```
fig, ax = plt.subplots(nrows=1, ncols=1, dpi=120)
ax.plot(Energy, Beam_Intensity, label="Raw Beam Intensity")
ax.plot(Energy, Background_Intensity, label="Average Background Intensity")
ax.plot(Energy, Corrected_Beam_Intensity, label="Corrected Beam Intensity")
ax.legend()
plt.xlabel("Energy [eV]")
plt.ylabel("Intensity [arb. units]")
plt.title("Comparing Raw Beam Data to Corrected Data")
```

Out[11]: <matplotlib.text.Text at 0x11ca99b38>



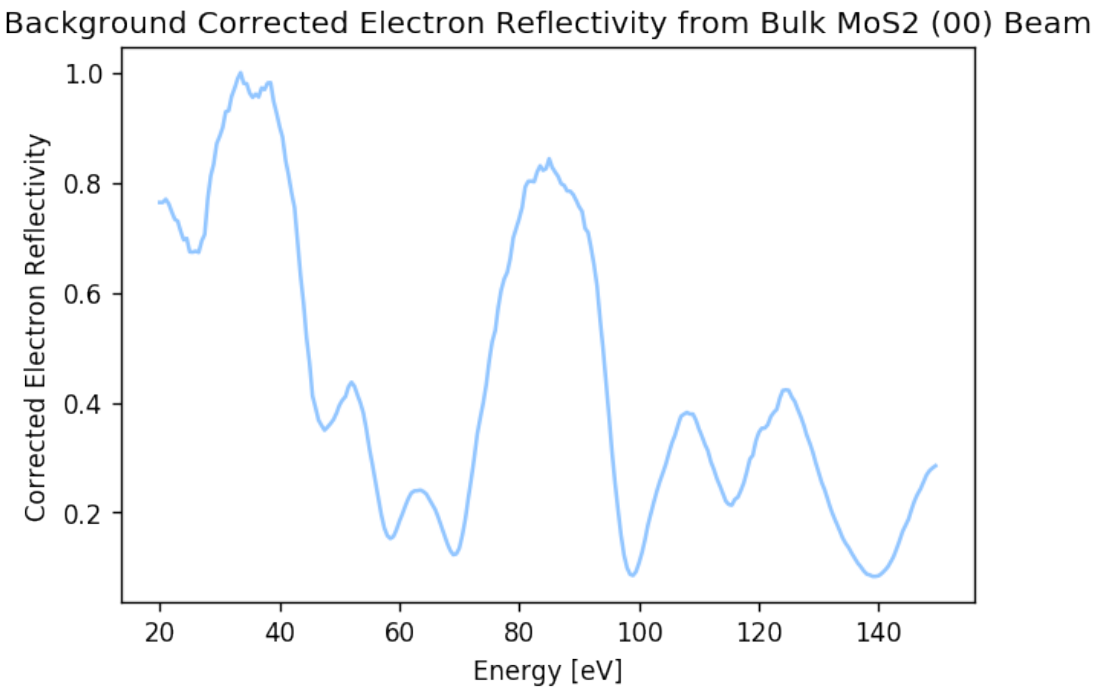
1.1 Here's the final result

```
In [12]: plt.figure(dpi=120)
         plt.plot(Energy, Corrected_Beam_Intensity)
         plt.xlabel("Energy [eV]")
         plt.ylabel("Corrected Intensity [arb. units]")
         plt.title("Background Corrected I(V) from Bulk MoS2 (00) Beam")

Out[12]: <matplotlib.text.Text at 0x11cb221d0>
```



1.2 Bonus: The electron beam we are analyzing is the (00) beam, which is the spectrally reflected beam. Thus we can rescale the Y-axis and plot Electron Reflectivity vs Energy

```
In [13]: max_int = max(Corrected_Beam_Intensity)
         plt.figure(dpi=120)
         plt.plot(Energy, [k/max_int for k in Corrected_Beam_Intensity])  #
         plt.xlabel("Energy [eV]")
         plt.ylabel("Corrected Electron Reflectivity")
         plt.title("Background Corrected Electron Reflectivity from Bulk MoS2 (00) Beam")

Out[13]: <matplotlib.text.Text at 0x11ccb98d0>
```


Background Corrected Electron Reflectivity from Bulk MoS2 (00) Beam
Corrected Electron Reflectivity
1.0
0.8
0.6
0.4
0.2
20
40
60
80
100
120
140
Energy [eV]