We will examine the properties of a spectrum! The code within this tutorial contains mistakes that you will correct.

For this tutorial, you will be answering some questions below. Please record your answers in a file called `arrayindexing_spectroscopy.txt` in your tutorials directory. The questions are numbered.

Please use `gvim` to answer these questions. If you're confused, stuck, or just generally having trouble with it, ask Adam. If you're still confused, stuck, and getting frustrated, then you can switch to `gedit` if you must.

(reminder: use `i` to get into "insert" mode, which is just like any regular old text editor, then `esc` to get into command mode)

## Read the data

There is a spectrum file sitting in the shared data directory:
`/home/shared/astr2600/data/spectrum.txt`
The wavelength is in units of Angstroms (Å). The flux units are not specified.

Open this in a text editor (`gedit`, `vim`, or whatever).
Examine it. **1a.** How many lines are in the file? ________________ **1b.** How many columns? ________________

Keep that path handy as you open IDL.

In IDL, find out how many lines there are in the file using `file_lines`:

```
nlines = file_lines('/home/shared/astr2600/data/spectrum.txt')
```

Check to make sure that agrees with the number you found above.

Now, make two variables to store the wavelength and flux arrays. Which column is the flux and which is the wavelength?
**2a.** Flux Column #: ________________ **2b.** Wavelength Column #: ________________

```
wavelength = fltarr(nlines)
flux = fltarr(nlines)
; replace ncols with the number of columns
xy = fltarr(ncols, nlines)
```

Now, read the data in from the file.

```
openr,lun,'/home/shared/astr2600/data/spectrum.txt',/get_lun
readf,lun,xy
wavelength = xy[0,*]
flux = xy[1,*]
free_lun,lun
```

Remember what all that does? Just to see some mistakes you could have made, let's try plotting some things.

```
plot,xy
```

What does this look like? Why? (just think about it. Maybe print out a few elements of `xy` to check)

Try plotting a different way:
`plot,wavelength,flux`

Hopefully that makes more sense.

## Examine the data

Zoom in on the line centered at 1330 angstroms.

First, set the width of the zoom and the center:

```
zoomwidth=10
center=1330
```

Then, set your upper and lower bounds:

```
lowerbound = center - zoomwidth/2
upperbound = center + zoomwidth/2
```

Now, use the `where` function to pick out the region you want to plot:

```
zoomregion = where( (wavelength gt upperbound) and (wavelength lt lowerbound), npts )
```

**3a.** What is `npts`? ________________
**3b.** Why? What did we do wrong? Fix it!

Now plot up your results:

```
plot,wavelength[zoomregion],flux
```

**4.** Does this look like the region centered around 1330 that we had looked at before?
Why not? What did we do wrong this time? Again, fix it!

## Measurements

First, what is $\Delta\lambda$? $\Delta\lambda$ means the separation between wavelengths, i.e. $\Delta x$. I expect you to be able to get this, but ask a neighbor if you're not sure how (don't just get the answer, understand the method)
**5a.** $\Delta\lambda =$________________

What is the continuum? (approximately, get it from the plot)
**5b.** *continuum* $\approx$________________

Now, determine how much light is absorbed. There are multiple ways you can do this. One way is to determine the "absorption spectrum" as we did in the lecture example. Another would be to determine the "area under the continuum" and the "area under the spectrum" and take the difference. Pick one of these methods - or another if you can think of any - and determine the absorbed flux.

**5c.** Absorbed flux: ________________

## Equivalent Width

What is the equivalent width of the line? Recall that equivalent width means "the width of the box that has the same area as the absorption line." It is a box going from the bottom to the top of the plot.
**6.** $EQW =$________________Å

Overplot a box centered on the line center with the correct height and width. Use `color='00AAFF'x` (orange).

To plot a box, you need to draw 4 lines. You can determine what the 4 corners are:
**7a.** $x1 =$________________ $y1 =$________________
**7b.** $x2 =$________________ $y2 =$________________
**7c.** $x3 =$________________ $y3 =$________________
**7d.** $x4 =$________________ $y4 =$________________

Then, you need an array with *5* elements! (each pair of corners gets you one line, but plot doesn't automatically draw from the last back to the first)

```
oplot,[x1,x2,x3,x4,x1],[y1,y2,y3,y4,y1],color='00AAFF'x
```

You've now measured and visualized Equivalent Width.

Finally, save the figure:

```
img=tvrd()
write_jpeg,'eqw_idl.jpg',img
```

# Python

Do it again in python (but this time, you don't need to record the answers). This is part of your homework as Exercise A.

Start python with `ipython --pylab`.

```
# Count the number of lines in the file:
# the "with" statement automatically closes the file when it's done
# If you have a : at the end of the line, as in this example, ipython
# will give you a "continuation" line that looks like ...:
with open('/home/shared/astr2600/data/spectrum.txt','r') as f:
    print len(f.readlines())

# Load the file using numpy "magic":
data = np.loadtxt('/home/shared/astr2600/data/spectrum.txt')
# The data have 2 columns and 601 rows
print data.shape
# We can "unpack" the data into two variables, but we need to
# change the shape of data from [601,2] to [2,601] so that the
# first number in the shape matches the number of variables
wavelength,flux = data.transpose()

# What happens if you try to plot(data) the way we tried plot,xy in IDL?
plot(data)

# That's a little better, but still not what we're looking for
# clear the figure, then plot the "right thing"
clf()
plot(wavelength,flux)

# The zooming part is pretty much the same as in IDL
zoomwidth = 10
center = 1330
lowerbound = center - zoomwidth/2
upperbound = center + zoomwidth/2

# However, this part is different - there's no need for "where":
zoomregion = (wavelength < upperbound) & (wavelength > lowerbound)

# Unlike in IDL, zoomregion has the same shape as wavelength
# We can use this to "highlight" the selected region
# This command fills in the region between two "lines" with a solid
# color.  "alpha" is the "transparency" - we're highlighting with
# a semi-transparent yellow color
fill_between(wavelength,zoomregion*0,zoomregion,alpha=0.5,color='yellow')

# Now let's plot the zoomed-in region in a new window:
figure()
plot(wavelength[zoomregion],flux[zoomregion])

# Measurements:
# Delta-lambda is pretty easy:
print np.diff(wavelength)
dl = wavelength[1]-wavelength[0]
```

```
# Can approximate the continuum (though this is a little sketchy, but it works
# for our data):
continuum = np.median(flux[zoomregion])

# integrate to get the absorbed flux
absorbed = np.sum( (continuum - flux[zoomregion]) * dl )

# Calculate the EQW
EQW = absorbed / continuum
# then plot it.  This will be a bit easier than the IDL way, since we're just
# going to fill in the area
fill_between([center-EQW/2,center+EQW/2], [0,0], [continuum,continuum],
    color='orange', alpha=0.5)

# To save these figures, you can click the "floppy disk" icon, or use savefig:
figure(2)
savefig("zoomed_eqw.png")
figure(1)
savefig("full_spectrum.png")
```

As usual, turn this in by moving your journal and `ipython_log` file to the appropriate location and pushing it on git. But this time, the IDL journal should go in your `tutorials` directory, while the `ipython_log` should be moved to `ipython_log_exerciseA.py` and placed in your `assignment3` directory.

You should also move the `.jpg` and `.png` files to the appropriate directories and `git add` them.