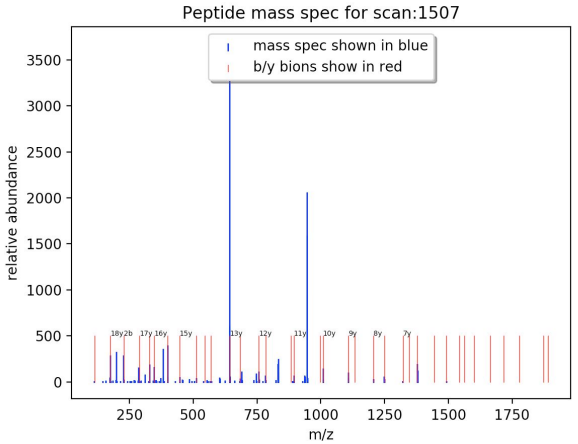
MS/MS Viewer

Rebecca Fuchs

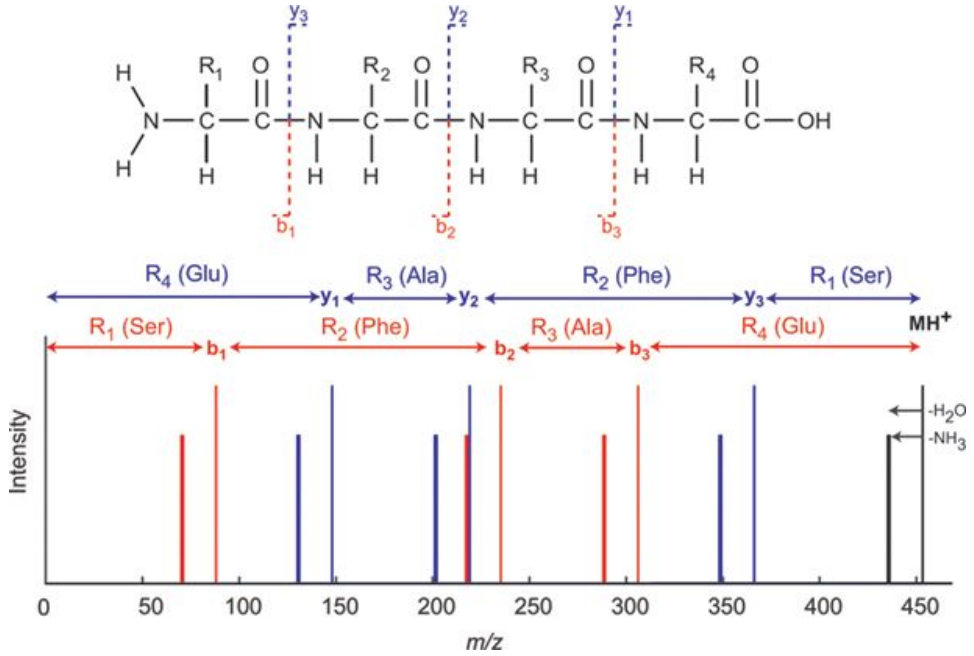
Project Requirements and Final Output

- Program will display peptide fragmentation spectra from mzXML file
- Using the known peptide sequence, b/y calculated ions will be displayed onto the figure
- Matching b/y ion peaks to fragmentation spectra peaks with labels
- Aid user to determine if the peptide is a good match

Rebeccas-MacBook-Pro:mass-spec-project rebeccafuchs$ python xmlproject_peptidesweights.py 17mix_test2.mzxml 1507 LITAIGDVVNHDPVVGDR
The peptide sequence you input ['L', 'I', 'T', 'A', 'I', 'G', 'D', 'V', 'V', 'N', 'H', 'D', 'P', 'V', 'V', 'G', 'D', 'R']
Here are the b & y ion matches, they are labeled in the figure too
{'2b': 227.32}
{'7y': 1321.8199999999997, '8y': 1206.7899999999997, '9y': 1107.7199999999998, '10y': 1008.65, '11y': 894.6099999999999, '12y': 757.47, '13y': 642.44, '15y': 446.25, '16y': 347.17999999999995, '17y': 290.13, '18y': 175.1}



“Tandem MS (or MS/MS, MSn) is a technique to break down selected ions (precursor ions) into fragments (product ions). The fragments then reveal aspects of the chemical structure of the precursor ion.” https://nationalmaglab.org/user-facilities/icr/techniques/tandem-ms



Closer look at my code

- Import sys to get file name, scan number, and peptide sequence from the command line
- Hard coded dictionary of amino acid weights (unlikely they will change…)
- Create b/y ion dictionaries with loop taking segments of peptide, looping through the shorter sequence to pull values of amino acid weights out of the dictionary to sum

```
for i in range(1,plength+1):
    bweight=0
    yweight=0
    bion=peptide_try[0:i]
    yion=peptide_try[i-1:plength+1]
    #print (i)
    for aab in bion:
        bweight=aaweights[aab]+bweight
    bion_dict['%db'%i]=bweight+1

    for aay in yion:
        yweight=aaweights[aay]+yweight
    yion_dict['%dy'%i]=yweight+19
```

- Import xml etree
- Get the root of tree, then loop for child in root. Use an if statement to get num for scan number equaling the input number. Oupt the binary peak data
- Use base64 to decode binary data, separate peaks mzs and ints from output array

```
scannum=input2 #scan number
stringnum=str(scannum)

for child in root:
    if 'num' in child.attrib:
        if child.attrib['num'] ==str(scannum):
            peaks_binary=child.attrib
            #print (child.attrib)
            for peaks in child:
                #print (peaks.text)
                peaks_text_binary=peaks.text
```

```
from base64 import b64decode
from array import array

   # peaks elt is the XML element corresponding to the peaks list
try:
    peaks = array('f',b64decode(peaks_text_binary))

    if sys.byteorder != 'big':
        peaks.byteswap()
except NameError:
    print("Scan number does not exist:"+stringnum)
    sys.exit(1)
```

- Compare mzs to b/y ion dictionaries separately using for loop of mz, if value is within range of 0.2 it's a match, printed on command line for user
- Plot everything peaks using lollipop plot from pylab, use text to plot matching labels, and label axis

```
for key, value in yion_dict.items():
    #print(value)
    for mz in mzs_list:
        #print(mz)
        if (mz-value) > -0.2 and (mz-value)<0.2:

            #print (mz,value)
            yion_dict_matches[key]=value
print ("Here are the b & y ion matches, they are labeled in the figure too")
print (bion_dict_matches)
print (yion_dict_matches)
```

```
#peaks from scan
(markers, stemlines, baseline) = plt.stem(mzs, ints, markerfmt=" ")
plt.setp(stemlines, linestyle="-", color="blue", linewidth=1)
plt.setp(baseline, visible=False)

#peak from bion
(markers, stemlines, baseline) = plt.stem(bion_list, bionints, markerfmt=" ")
plt.setp(stemlines, linestyle="-", color="red", linewidth=0.5)
plt.setp(baseline, visible=False)

#label for matches
for key, value in bion_dict_matches.items():
    plt.text(value, 501, key,fontsize=5)

#peak from yion
(markers, stemlines, baseline) = plt.stem(yion_list,yionints, markerfmt=" ")
plt.setp(stemlines, linestyle="-", color="red", linewidth=0.5)
plt.setp(baseline, visible=False)

#label for matches
for key, value in yion_dict_matches.items():
    plt.text(value, 501, key,fontsize=5)

#put the peptide on the figure
#plt.text(right,bottom, peptide_try, fontsize=5, ha="right", color='.5', horizontalali
#doesnt look right for all figures :(

#labels
plt.xlabel("m/z")
plt.ylabel("relative abundance")
plt.title('Peptide mass spec for scan:'+stringnum )
plt.legend (( 'mass spec shown in blue', 'b/y bions show in red'), loc='upper center',
plt.show()
```

General notes to help user

- All matches are printed out in command line and labeled on figure
- Loads of comments and simple variable names to help user understand code
- Exceptions in place incase user forgets to input command line requirement

```
#command line code
import sys
try:
    input1=sys.argv[1]
    input2=sys.argv[2]
    input3=sys.argv[3]
except IndexError:
    print ("You forgot to provide mzxml.file OR scan.number OR peptide.seq")
    sys.exit(1)
```

Packages

- Pylab
  - Easy to plot multiple datasets, easy to use labels, and titles, but I found it challenging to add general text to the figure in a specified corner, you needed coordinates but coordinates changed for each model
- xml.etree.ElementTree
  - Straightforward to get root and loop through to find scan number
- Base64
  - With help from Dr.Edwards using this package was doable, but on my own I found it very confusing, but I hadn't worked with binary data

Scientific trade offs

- If “matches” of b/y ions are with ms/ms noise?
  - Should I have used a threshold of intensity
  - What threshold would work well for all files
- How close should the spectra m/z to the b/yions?
  - My code makes matches that are within 0.2
  - Perhaps this could be improved when knowing the precision of the machine

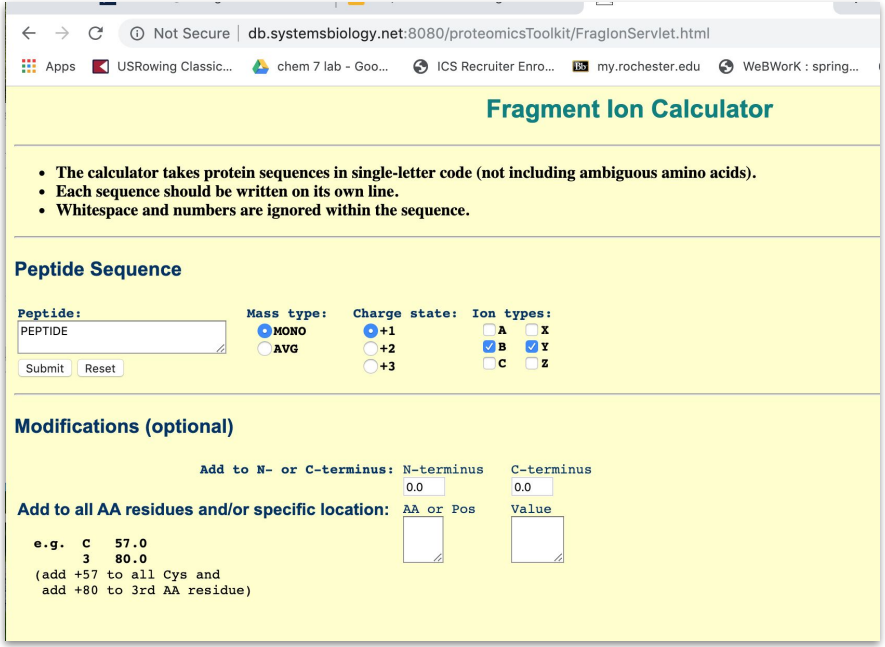
Tricky moments

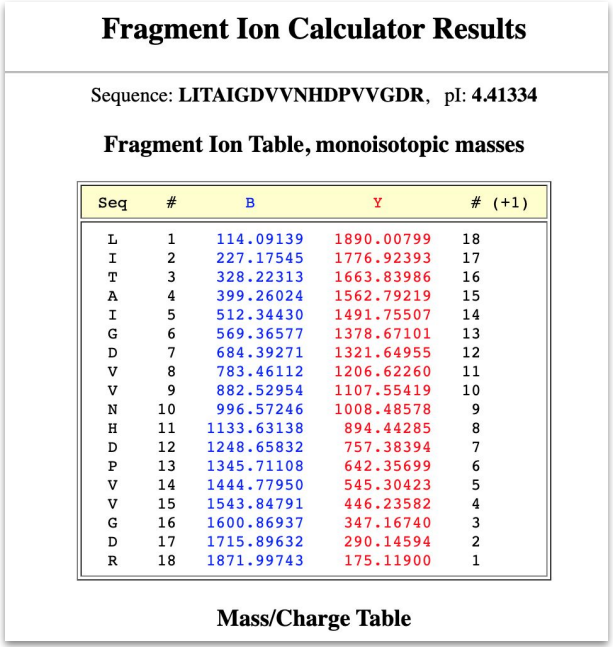
- Amino acid weights dictionary, I typed it in and switched two… made for a very frustrating day… Perhaps I should have looked for a way to read in an amino acid weight table
- How much to add to b/y ions…
  - Mass of b-ions = $\sum$ (residue masses) + 1 (H^+)
  - Mass of y-ions = $\sum$ (residue masses) + 19 (H_2O+H^+)
- Deciding if I wanted to do parts of the code in separate files or keep it all together
  - Keeping all the code on the same file kept it more simple and easier to follor

Easier tasks vs. challenges

- What aspects of your program was straightforward ?
  - Pulling out one line from XML was much easier than I expected, used an if statement while looping through all scans
  - Graphing both spectra and coding the loop to find matching spectra
  - Looping through the peptides to get each b/y ions weight
- What aspect of your program was difficult to make work?
  - I found it difficult to add extra text to my figure because I couldn’t hard code in coordinates since the figure changed size every run
  - Deciding on the intensity of b/y ions

Testing my program





Seq	#	B	Y	# (+1)
L	1	114.09139	1890.00799	18
I	2	227.17545	1776.92393	17
T	3	328.22313	1663.83986	16
A	4	399.26024	1562.79219	15
I	5	512.34430	1491.75507	14
G	6	569.36577	1378.67101	13
D	7	684.39271	1321.64955	12
V	8	783.46112	1206.62260	11
V	9	882.52954	1107.55419	10
N	10	996.57246	1008.48578	9
H	11	1133.63138	894.44285	8
D	12	1248.65832	757.38394	7
P	13	1345.71108	642.35699	6
V	14	1444.77950	545.30423	5
V	15	1543.84791	446.23582	4
G	16	1600.86937	347.16740	3
D	17	1715.89632	290.14594	2
R	18	1871.99743	175.11900	1

http://db.systemsbiology.net:8080/proteomicsToolkit/FragIonServlet.html

Takeaways

- What I learned about bioinformatics
  - You need to understand the biology/chemistry of the topic to really get your program right (example how much to add to b/y ions)
  - Starting early makes the process less stressful
- What I learned about programming
  - Keeping everything organized/ symmetrical helps you understand the code and keeps things more simple (do the same process for b and y ions)
  - Test user input errors to see if you print a helpful output error

Examples to try

1301 TYDSYLGDDYVR

1298 TYDSYLGDDYVR

845 AAQKPDVLTTGGGNPVGDK

1505 LITAIGDVVNHDPVVGDR

1506 LITAIGDVVNHDPVVGDR

1290 HSTVFDNLPNPEDR

1504 LITAIGDVVNHDPVVGDR

1508 LITAIGDVVNHDPVVGDR