Quiz Section Week 10
May 30, 2017

Numpy

Bash/command line refresher

Workshop prep

Python has a lot of useful modules

Big ones:

- numpy: "numerical python" - numerical analysis
- scipy: "scientific python" - stats
- pandas: "python data analysis" - organizing data
- scikit-learn: machine learning
- matplotlib: plotting

Numpy provides an a powerful data structure: arrays

```
import numpy as np
x = np.array([[1,3,4],[1,5,0]])
print x
print x.shape
print x.size
print x.ndim
print x[1,2]
y = np.zeros([4,3])
print y
z = np.ones([3,6])
rand = np.random.random([3,5])
y.fill(2)
print y
```

Must be all a single data type!! What happens if not?
What if lists of lists are not all the same length?

https://docs.scipy.org/doc/numpy/reference/arrays.ndarray.html

Many methods and functions to manipulate and do calculations on arrays

```
x.tolist()
x.tofile('out.txt',sep = ' ')
new_x = np.loadtxt('out.txt')
x.transpose()
x.flatten()
x.reshape([6,1])
np.count_nonzero(x)
x.put([1,2],200)
x.sort(axis = 0)
x.sort()
x.sum()
x.sum(0) #What do these do?
x.sum(1)
x.min(0)
x.max(1)
```

```
x.clip(0,10)
x > x
x > y

x*x
np.dot(x, x.transpose())
np.dot(x,x)
np.dot(x,y)
```

Slicing and accessing array subsets

```
x[1,2]
x[0]
x[:][1]
x[x > 2]
x[x[0] > 2]
x[0][x[0] > 2]
x[0][-2]
x[0][-2:]
x[1:2,1]
x[1][[2,1]]
```

Numpy can be faster than standard lists

```
xlist =[1,3,4,5,0]
x = np.array(xlist)

start = time.time()
print x.sum()
print (time.time()-start)

start2 = time.time()
sumlist = 0
for element in xlist:
    sumlist=sumlist + element

print sumlist
print (time.time()-start2)
```

Exercises

- Create a random vector of size 10 and replace the max value with 0

```
randmat = np.random.random(10)
randmat[randmat.argmax()] = 0
print(randmat)
```

- Find and print the element of each row of x that is closest to 2 (hint: look up the np.argmin() function)

```
x=np.array([[3,2],[4,7],[8,12],[2,100],[0,2],[1,
3]])
mins = np.argmin(abs(x-2), axis=1)
print x[np.arange(0,len(mins)),mins]
```

What if we have data in multiple types?

- e.g. a bed file or sam file!

```
r001    163     ref     7       30      8M4I4M1D3M      =       37      39      TTAGATAAAGAGGATACTG     *       XX:B:S,12561,2,20,112
r002    0       ref     9       30      1S2I6M1P1I1P1I4M2I      *       0       0       AAAAGATAAGGGATAAA       *
r003    0       ref     9       30      5H6M    *       0       0       AGCTAA  *
r004    0       ref     16      30      6M14N1I5M       *       0       0       ATAGCTCTCAGC    *
r003    16      ref     29      30      6H5M    *       0       0       TAGGC   *
r001    83      ref     37      30      9M      =       7       -39     CAGCGCCAT       *
```

Pandas! http://pandas.pydata.org

Command line refresher

Variables:

```
x="myfile.txt"
echo "some words!" > myfile.txt
cat myfile.txt
ls -lh myfile.txt
mkdir new_directory
mv myfile.txt new_directory
echo $x
echo $PATH
```

Studying for the final

Primarily 2nd half of course

Practice! Re-do problems from old homeworks, quiz sections

Practice explaining concepts in your own words

Questions?

Course Takeaways

- Bioinformatics is not magic: there are always assumptions, uncertainties, ambiguities
- Have an understanding of what is happening inside black boxes – it might not be as complicated as it seems initially
- Start small and be clear in your own analysis and programming

Before tomorrow: download and install Docker!

- A Docker container is basically a mini computer with its own operating system and programs that will run on any computer
- Tomorrow you'll download a Docker container with some sequence analysis programs and data
- Make sure you have ~5-8 GB of storage free on your computer