Numerical Python

numpy.array, plotting

CS101 Lecture #15

2016-11-14

Warm-up Questions

Multidimensional indexing

a = [[1, 2], [3, 4], [5, 6, 7]]

- What is the type of a?
- What is the value of a[2][2]?
- What is the value of a[1][2]?

Multidimensional indexing

Create the following two-dimensional list:

```
x = [ [1,0,0,0],
      [0,1,0,0],
      [0,0,1,0],
      [0,0,0,1] ]
```

Array

The problem

```
mydata = [ 4.5, 6.0, 1.2, 5.4 ]
from math import sin
sin(mydata)
```

- Why doesn't this work?
  - `list` can contain any type!
- Also operators don't do what we "want":

```
mydata * 2.0  # doesn't double values!
```

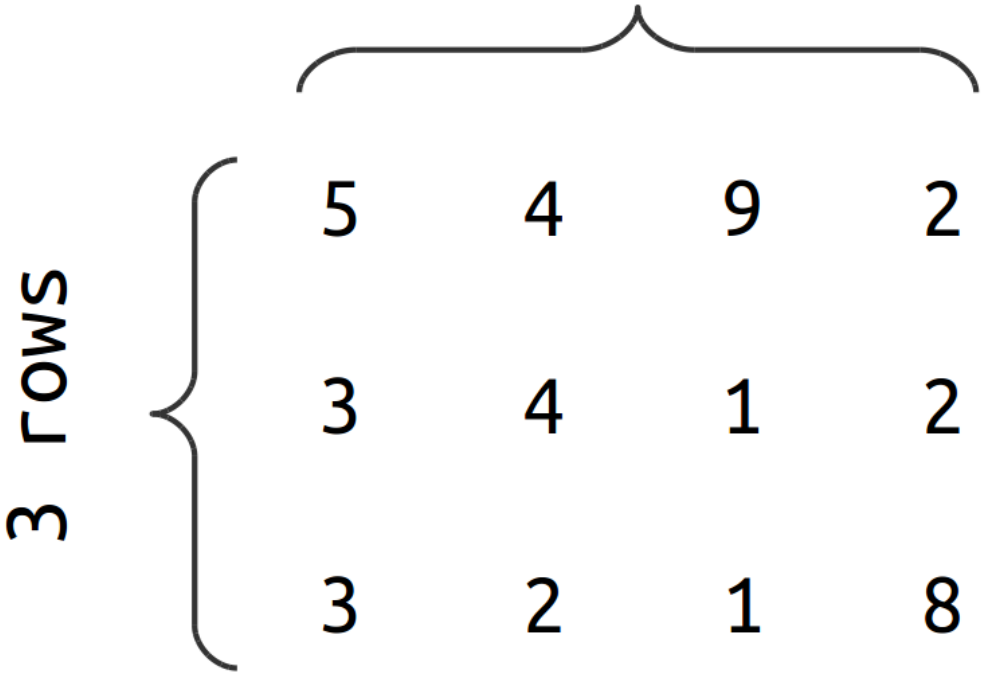
numpy.array

```
import numpy
import numpy as np  # better way
```

- numpy provides arrays and mathematical functions.

```
data = np.array( [ 4.5, 6.0, 1.2, 5.4 ] )
data * 2.0
np.exp(data)
```

Indexing



- `numpy.array` indexes an element by `array[row][col]`.
- `numpy.array` indexes a row by `array[row]`.
- `numpy.array` indexes a sub-array by `array[r1:r2, c1:c2]`.

Indexing arrays

4 columns

3 rows

5	4	9	2
3	4	1	2
3	2	1	8

- What is `a[1:3][1:2]`?.

Question

$$x = \begin{pmatrix} 1 & 1 \\ 2 & 2 \\ 3 & 3 \end{pmatrix}$$

What will produce this array?

A `np.array([[1,2,3],[1,2,3]])`

B `np.array([2,3])`

C `np.array([3,2])`

D `np.array([[1,1],[2,2],[3,3]])`

Question

$$x = \begin{pmatrix} 1 & 1 \\ 2 & 2 \\ 3 & 3 \end{pmatrix}$$

What will produce this array?

A `np.array([[1,2,3],[1,2,3]])`

B `np.array([2,3])`

C `np.array([3,2])`

D `np.array([[1,1],[2,2],[3,3]])`

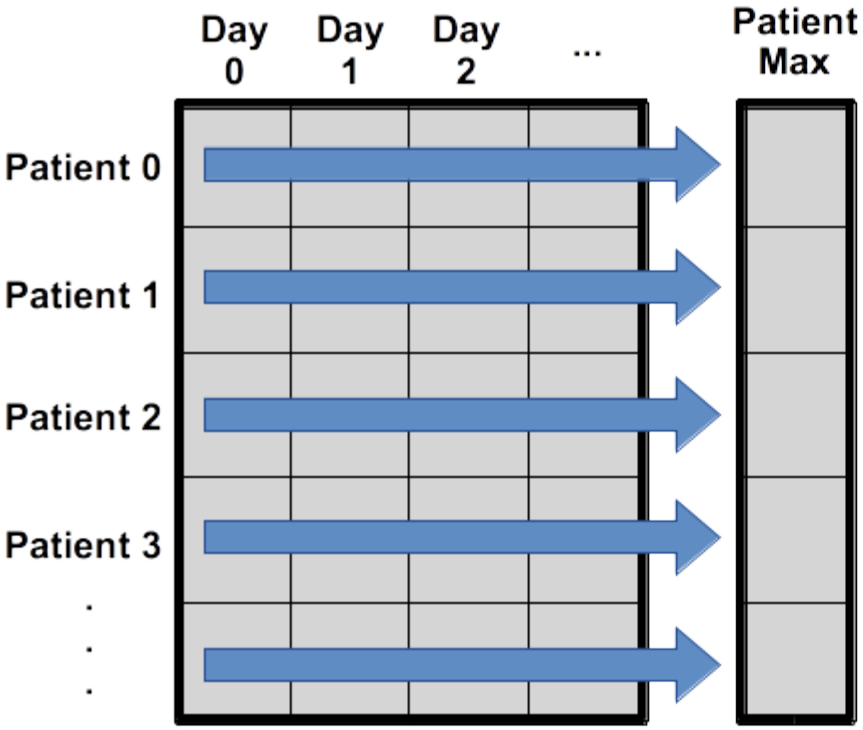
$\star$

numpy.array

Consider a data set containing patient inflammation records for 60 patients over a period of 40 days, contained in `inflammation.csv`.

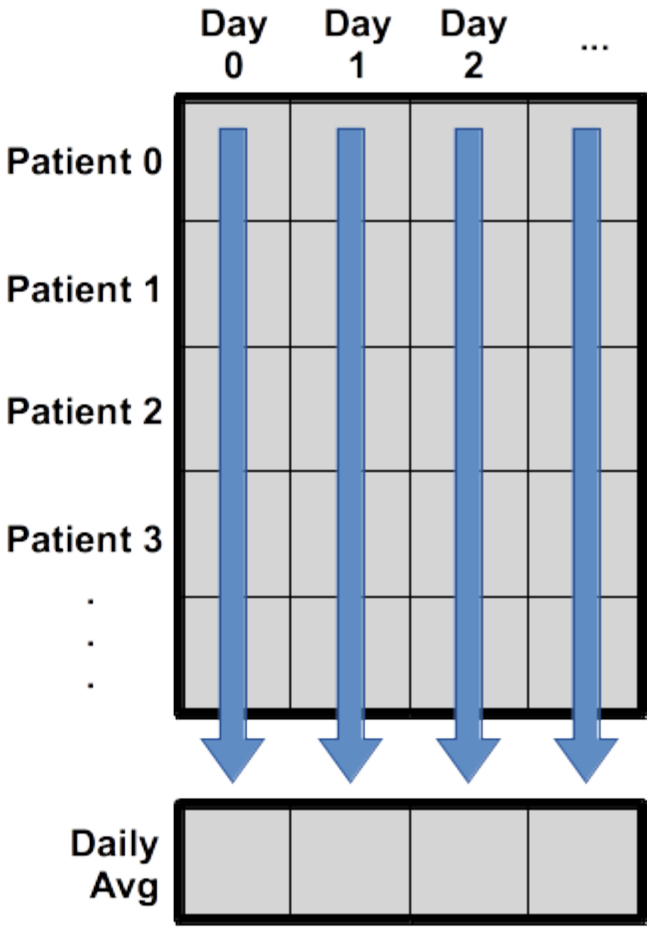
```
data = np.loadtxt( './data/inflammation.csv',
                   delimiter=',' )
```

numpy.array



Max for each patient

`data.max(axis=1)`



Average for each day

`data.mean(axis=0)`

Other arrays

```
x = np.zeros([2,3] ) # zeroes
y = np.ones([4,1])   # ones, 4 rows, 1 column
y = np.ones([1,4])   # ones, 1 row, 4 columns
y = np.ones([4])     # ones, a row vector
```

- Produce arrays of zeros or ones with specified dimensions.

Other arrays

```
z = np.eye( 5 )          # 5x5 identity matrix
```

- Produces identity matrix of specified square dimension.

Other arrays

```
w = np.linspace( 0,10,101 )
v = np.linspace( start, finish, n)
```

- Produce arrays from `start` to `finish` of `n` points (*not* spacing!).
- Excellent for grids and coordinates.
- May also see `arange`: [start, stop), but I recommend avoiding its use:

```
u = np.arange( 0,10,0.1 )  # tricky!
u == array( [ 0, 0.1, 0.2, ..., 9.9 ] )
```

Plotting (`matplotlib`)

matplotlib

```
import matplotlib.pyplot as plt
```

- A plotting environment similar to MATLAB.
- Can plot `lists` or `arrays`.

```
xs = list( range(4) )
ys = [ 4.5, 6.0, 1.2, 5.4 ]
plt.plot( xs, ys )
plt.show()
```

matplotlib

- *Always* include labels:
  - `plt.xlabel( 'domain' )`
  - `plt.ylabel( 'range' )`
  - `plt.title( 'topical data' )`

```
plt.plot( xs, ys )
plt.xlabel( 'x' )
plt.ylabel( 'y' )
plt.title( 'some values' )
plt.show()
```

matplotlib

- Basic cycle:
  - Add data to plot.
  - Add labels to plot.
  - Show plot.

matplotlib

- Two kinds of plots today:
  - `plt.plot( x, y ) # for ptwise data`
  - `plt.imshow( A ) # for array data`
- `plot`: third argument is *format string* (optional; color string + line style string); default as 'b-' for a solid blue line.

```
plt.plot( xs, ys, 'r.' )
plt.show()
```

- `plot`: can also take *keyword arguments*.

```
plt.plot( xs, ys, 'r.', linewidth=1.0 )
plt.show()
```

matplotlib

More options of plot:
http://stackoverflow.com/questions/8376926/plotting-many-graphs-with-matplotlib

https://matplotlib.org/api/pyplot_api.html

Plot $\sin(x)$ for $x \in [0, 2\pi]$ using numpy.

```
import matplotlib.pyplot as plt
%matplotlib inline
import numpy as np
x = np.linspace( 0,2*np.pi,101 )
y = np.sin( x )

plt.plot( x,y,'k-' )
plt.xlim( 0,2*pi )
plt.ylim( -1,1 )
plt.show()
```