Lecture 7 - Arrays

* Prayer / AMA / Quiz

I. Array Basics

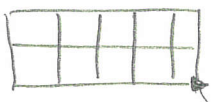
* In mathematics, we often encounter vectors and matrices.

e.g. $\underline{v} = \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix}$ ← x-component of velocity vector

$\underline{\sigma} = \begin{bmatrix} \sigma_{xx} & \sigma_{xy} & \sigma_{xz} \\ \sigma_{yx} & \sigma_{yy} & \sigma_{yz} \\ \sigma_{zx} & \sigma_{zy} & \sigma_{zz} \end{bmatrix}$ ← x-z component of stress tensor (matrix)

* In numerical computing, vectors and matrices are represented by objects called arrays.

* An array is really just a list of variables. Think of it like an ice-cube tray:



Each section holds a value.

* In Excel, an array is just a group of cells in a worksheet., e.g. A1-A5

* Python has three "array" type objects.

  - A list
  - A tuple
  - A numpy array.

* List:

A list object is made by assigning a variable to a list of things surrounded by square brackets:

```
my_list = [4, 7, 5]
```

* Tuple

A tuple object is made by assigning a variable to a list of things surrounded by parentheses:

```
my_tuple = (4, 7, 5)
```

* Numpy Array

A numpy array is made by assigning a variable to a numpy array function with a list as an argument:

```
import numpy as np

my_array = np.array([4,7,5])
```

* when use each?

  - lists are good for, well, storing lists of variables
  - tuples are good for passing arguments in and out of functions. (This is what you do when you put a comma in a return x,y statement).
    - ↳ tuples & lists do basically the same thing, except you can't change the elements of a tuple once it has been created.
  - arrays are good for doing math. we will use them the most.

* You access the elements of a list, tuple or array using square brackets:

```
print( my_list[2] )
                ↖ "index", must be an integer.
>>> 5
```

(*) There are more tricks here. see the examples.

* the array index starts counting with 0!

  So, 0 is the 1st element, 1 is the 2nd, so on.

  Computer scientists start counting with 0.

* Do not get functions and arrays confused.

  - calling a function: `my_func(5)` ← argument
  - accessing an element of an array: `my_array[2]` ← index

Activity Array Basics

II. Loops and arrays

* Using loops and arrays often go hand in hand.

For example, we can use a loop to fill an array:

```
vel = np.zeros(3)  # initialize array to [0,0,0]
for i in range(3):
    vel[i] = i
# vel is now [0,1,2]
```

* we can also make 2D arrays (matrices) and fill it using a nested loop:

```
sig = np.zeros((3,3))  # 3x3 matrix
for i in range(3):
    for j in range(3):
        sig[i,j] = i*j
# sig is a matrix
```

* Using loops and arrays, we can do vector and matrix operations from math.

Dot product: $\underline{v} \cdot \underline{w} = v_x w_x + v_y w_y + v_z w_z$

$= \sum_i v_i w_i$

matrix-vector multiplication:

$$\underline{\underline{A}} \cdot \underline{x} = \sum_j A_{ij} x_j$$

matrix-matrix multiplication:

$$\underline{\underline{A}} \cdot \underline{\underline{B}} = \sum_j A_{ij} B_{jk}$$

Activity: Arrays & Loops