numpy review

- Numpy provides a high performance multi-dimensional array and tools to create/manipulate these arrays.
- All elements of a numpy array must be of the same type.
- number of dimensions is rank (array.ndim) → different from matrix rank of course.
- size of each dimension is shape (array.shape)
- there are built in initializers np.zeros() and np.ones(). np.full() is a array where all elements are the same. np.eye() is the identical matrix array of whatever rank as input. np.empty_like(x) creates an empty array with the same shape as array **a.**
- You can also mix integer indexing with slice indexing. However, doing so will yield an array of lower rank than the original array
- A slice of an array is a view into the same data, so modifying it will modify the original array.
- Note that `::` is the same as `:` and means select all indices along this axis.

```
Ellipsis expand to the number of : objects needed to make a selection tuple
of the same length as x.ndim.
```

- Each `newaxis` object in the selection tuple serves to expand the dimensions of the resulting selection by one unit-length dimension. The added dimension is the position of the `newaxis` object in the selection tuple.
- The term broadcasting describes how numpy treats arrays with different shapes during arithmetic operations. Subject to certain constraints, the smaller array is “broadcast” across the larger array so that they have compatible shapes. Broadcasting provides a means of vectorizing array operations so that looping occurs in C instead of Python.
- broadcasting two arrays together follows these rules:
    - If the arrays do not have the same rank, prepend the shape of the lower rank array with 1s until both shapes have the same length.
    - The two arrays are said to be *compatible* in a dimension if they have the same size in the dimension, or if one of the arrays has size 1 in that dimension.
    - The arrays can be broadcast together if they are compatible in all dimensions.
    - After broadcasting, each array behaves as if it had shape equal to the elementwise maximum of shapes of the two input arrays.
    - In any dimension where one array had size 1 and the other array had size greater than 1, the first array behaves as if it were copied along that dimension
- Functions that support broadcasting are known as *universal functions.* Broadcasting makes code not only concise but also faster. Double win!
- np.reshape(array, new_shape, order) re-shapes the array without changing the data. order can be C-like, Fortran-like or automatic (A).
- Advanced indexing is triggered when the selection object, *obj* in array[obj], is a non-tuple sequence object, an `ndarray` (of data type integer or bool), or a tuple with at least one sequence object or ndarray (of data type integer or bool). There are two types of advanced indexing: integer and boolean.
- Advanced indexing always returns a *copy* of the data (contrast with basic slicing that returns a *view*).
- The definition of advanced indexing means that `x[(1,2,3),]` is fundamentally different than `x[(1,2,3)]`. The latter is equivalent to `x[1,2,3]` which will trigger basic selection while the former will

trigger advanced indexing.

- Integer array indexing allows selection of arbitrary items in the array based on their *N*-dimensional index. Each integer array represents a number of indexes into that dimension.
- When the index consists of as many integer arrays as the array being indexed has dimensions, the indexing is straight forward.
- Boolean array indexing:

```
a = np.array([[1,2], [3, 4], [5, 6]])
bool_idx = (a > 2) # Find the elements of a that are bigger than 2;
# this returns a numpy array of Booleans of the same
# shape as a, where each slot of bool_idx tells
# whether that element of a is > 2.
print bool_idx # Prints "[[False False]
# [ True True]
# [ True True]]"

# We use boolean array indexing to construct a rank 1 array
# consisting of the elements of a corresponding to the True values
# of bool_idx
print a[bool_idx] # Prints "[3 4 5 6]"
```

- Numpy tries to guess a datatype when you create an array, but functions that construct arrays usually also include an optional argument to explicitly specify the datatype.
- dot() function: For 2-D arrays it is equivalent to matrix multiplication, and for 1-D arrays to inner product of vectors (without complex conjugation). For N dimensions it is a sum product over the last axis of *a* and the second-to-last of *b.*
- SciPy builds on this, and provides a large number of functions that operate on numpy arrays and are useful for different types of scientific and engineering applications.
- SciPy provides some basic functions to work with images. For example, it has functions to read images from disk into numpy arrays, to write numpy arrays to disk as images, and to resize images.
- Matplotlib is a plotting library.