Advanced Arrays and Sequences

Sequences versus Elements

- Let us call anything that is ordered a "sequence." (Python tends to use this terminology.)
- Arrays are ordered (including multidimensional arrays) so they are sequences.
- We can refer to the sequence by its name. In Python/NumPy this implies an array operation or a list operation.
- We can address individual elements or slices of a sequence with subscripts.
- A[i] B[1:j+k+1] (Python–0 based; watch upper bound)

Python Sequence Objects

- Sequence objects are:

strings, Unicode strings, lists, tuples, bytearrays, buffers, and xrange objects.

xrange objects are created by the xrange function (similar to the range function we have already used). Numpy has the arange function.

Sequences support operators `in` (returns True or False) and `not in` (ditto) along with several others, see

http://docs.python.org/library/stdtypes.html

Tuples

- A tuple is a Python ordered sequence object that is similar to a list but is immutable.
- Tuples are indicated by parentheses (round brackets).
- Like all sequences, we can refer to individual elements with [n] and slices as [lb:ub] (remember the rule about ub -- the sequence is up to < ub not <=ub)
- For most Python ordered sequences, S[-1] indicates the last element, S[-2] the next to last, and so forth.

Python Dictionaries

- A dictionary is an unordered structure. Values are associated with *keys*.
- Dictionaries are indicated by curly braces {}
- Dictionaries are mutable and can be of any length (up to the limits of the system).
- Dictionaries can be nested.
- Example. Explicit key-value pairs:

```
D1={ 'bear':'panda','cat':'leopard','dog':'wolf'}
```

Creating Dictionaries

- Empty dictionary:
  - D={}
- Start with a key-value set
  - D={'Alice': '2341', 'Beth': '9102', 'Cecil': '3258'}
- To add elements
  - D[‘Dan’]=‘5837’
- Keys must be an immutable type
- Keys may not be duplicated

Dictionary Operations

- len(D)
  - Number of entries in the dictionary
- del D[k]
  - Delete item pair
- D.clear()
  - Clear dictionary
- k in D
  - check whether key k is in the dictionary
- Several others

NumPy Array Initializations

- Convert from list

```
A=array([x])
```

- Initialize to size 100 (0-99), all zeros

```
A=zeros(100)
```

- Initialize to random garbage

```
A=empty(100)
```

- Identity

```
A=eye(100) # Two-d array
```

There are others.

NumPy Built-Ins

- There is a very large number of them. These are just a few
- loadtxt, fromfile
- shape, size, reshape
- ufuncs: basic functions that operate elementwise on arrays
  - np.sin() etc.
- http://www.scipy.org/Numpy_Functions_by_Category
- http://www.scipy.org/Tentative_NumPy_Tutorial

NumPy

- Arrays are *mutable* and so defs can change their elements as a side effect.
- B=A does not copy A into a new object B.

  use the copy method

  B=A.copy