INSY 6500
Information Systems for Operations

Introduction to NumPy

Fall 2019

Jeffrey S. Smith

Industrial and Systems Engineering Department
Auburn University
jsmith@auburn.edu

What is NumPy?

NumPy (short for Numerical Python) provides an efficient interface to store and operate on dense data buffers. In some ways, NumPy arrays are like Python's built-in list type, but NumPy arrays provide much more efficient storage and data operations as the arrays grow larger in size. NumPy arrays form the core of nearly the entire ecosystem of data science tools in Python, so time spent learning to use NumPy effectively will be valuable no matter what aspect of data science interests you.

VanderPlas, Notebook 02.00

Our NumPy-related Resources

- Course Jupyter Notebook(s?)
  - 3.4.0 – Introduction to NumPy.ipynb
- VanderPlas Jupyter Notebooks
  - GitHub address: https://github.com/jakevdp/PythonDataScienceHandbook
  - Notebooks 02.00 – 02.09
- NumPy web page
  - Quickstart Tutorial: https://docs.scipy.org/doc/numpy-1.13.0/user/quickstart.html
  - Documentation: https://docs.scipy.org/doc/numpy-1.13.0/index.html

Understanding Python Data Types: Python vs. C

```c
/* C code */
int result = 0;
for(int i=0; i<100; i++){
    result += i;
}
```

```python
# Python code
result = 0
for i in range(100):
    result += i
```

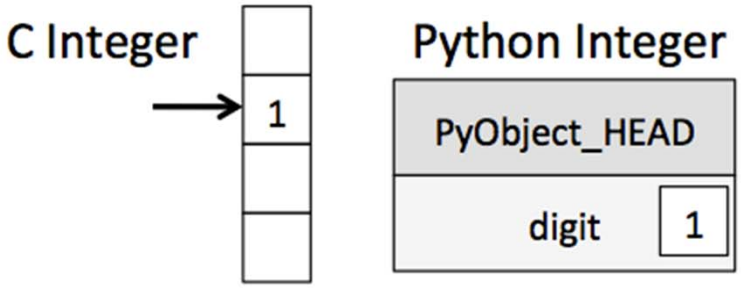
```c
/* C code */
int x = 4;
x = "four";  // FAILS
```

```python
# Python code
x = 4
x = "four"
```

https://jakevdp.github.io/PythonDataScienceHandbook/02.01-understanding-data-types.html

Python Integer Object

```
struct _longobject {
    long ob_refcnt;
    PyTypeObject *ob_type;
    size_t ob_size;
    long ob_digit[1];
};
```

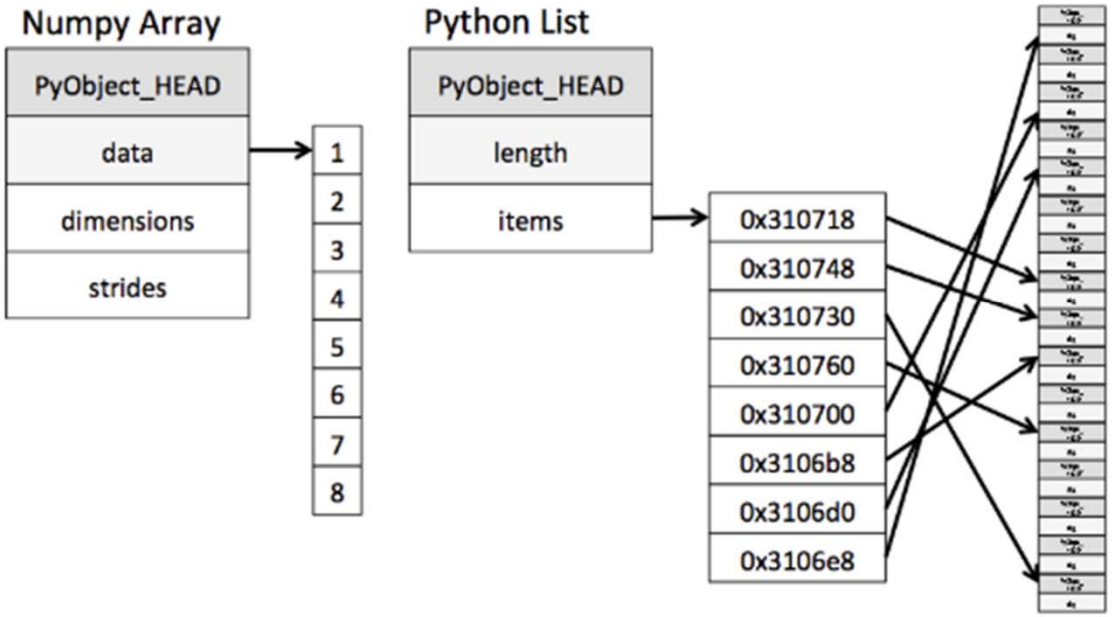


A single integer in Python 3.4 actually contains four pieces:

- `ob_refcnt`, a reference count that helps Python silently handle memory allocation and deallocation
- `ob_type`, which encodes the type of the variable
- `ob_size`, which specifies the size of the following data members
- `ob_digit`, which contains the actual integer value that we expect the Python variable to represent.

https://jakevdp.github.io/PythonDataScienceHandbook/02.01-understanding-data-types.html

Python Lists vs. Fixed (NumPy-type) Arrays



https://jakevdp.github.io/PythonDataScienceHandbook/02.01-understanding-data-types.html

NumPy Arrays

- NumPy's array class is called ndarray. It is also known by the alias array. Note that numpy.array is not the same as the Standard Python Library class array.array, which only handles one-dimensional arrays and offers less functionality. Important attributes of NumPy arrays include:
  - ndarray.ndim – the number of dimensions
  - ndarray.size – the total number of elements
  - ndarray.shape – the shape of the array (as a tuple)
  - ndarray.dtype – the data type of the elements
  - ndarray.itemsize – size (in bytes) of each element
  - ndarray.data – the data buffer.

https://docs.scipy.org/doc/numpy-dev/user/quickstart.html

Basics of NumPy Arrays

- Array Indexing
- Array Slicing
  - No-copy Views vs. Copies
- Reshaping / np.newaxis
- Array Concatenation and Splitting
- Universal Functions and the Slowness of Loops

Axes and Aggregation

- NumPy uses the term *axis* to refer to the array dimensions.
- Consider a 2-d array viewed as a matrix:
  - axis 0 represents the *rows* of the matrix
  - axis 1 represents the *columns* of the matrix
- Using axes for aggregate functions
  - Assume that we have a 2-d array, A with shape (`n, m`) – in matrix terms, this would be `n` rows by `m` columns.
  - `A.sum()` – returns the sum of all elements in A
  - `A.sum(axis = 0)` – returns an array of size `m` with the "column" sums
  - `A.sum(axis = 1)` – returns an array of size `n` with the "row" sums
  - Many aggregation functions available
- Generalizes to more that 2 dimensions (but it's more difficult to visualize).

Broadcasting

- Allows binary operations to be performed on arrays of different sizes.
- Broadcasting in NumPy follows a strict set of rules to determine the interaction between the two arrays:
  1. If the two arrays differ in their number of dimensions, the shape of the one with fewer dimensions is padded with ones on its leading (left) side.
  2. If the shape of the two arrays does not match in any dimension, the array with shape equal to 1 in that dimension is stretched to match the other shape.
  3. If in any dimension the sizes disagree and neither is equal to 1, an error is raised.

VanderPlas, Notebook 02.05

Comparisons, Masks, and Boolean Logic

- Comparison Operators as ufuncs (universal functions)
- Boolean arrays
- Boolean arrays as masks