Python and Scientific Computing

Notes of John D. Hunter and Fernando Perez

Contents

Chapter 1. Why python? 5

9.1. weave 61
9.2. swig 61
9.3. f2py 61
9.4. Others 61

Bibliography 63

CHAPTER 1

Why python?

CHAPTER 2

$2^{200}$is a huge number!

```
>>> 2**200
1606938044258990275541962092341162602522202993782792835301376L
```

but python will blithely compute it and much larger numbers for you as long as you have CPU and

```
>>> last = 'Hunter'
>>> last[0]
'H'
>>> last[1]
'u'
>>> last[-1]
'r'
```

To access substrings, or generically in terms of the sequence protscol, slices, you use a colon to indicate a range

```
# string slicing
>>> last[0:2]
'Hu'
>>> last[2:4]
'nt'
```

As this example shows, python uses "one-past-the-end" indexing when defining a range; eg, in the range `indmin:indmax`, the element of `imax` is not included. You can use negative indices when slicing tso; eg, ts get everything befsre the last character

```
>>> last[0:-1]
'Hunte'
```

```
# s is a single string and we split it into a list of strings
```

needed to compute the product of two numbers between 1 and 100 in an inner loop – you could use a dictionary to cache the cube of all odd of numbers < 100; if you were inteterested in all numbers, you might simply use a list to store the cached cubes – I am cacheing only the odd numbers to show you how a dictionary can be used to represent a sparse data structure

```
>>> cubes = dict([ ( i, i**3 ) for i in range(1,100,2)])
>>> cubes[5]
125
```

The last example is syntactically a bit challenging, but bears careful study. We are initializing a dictionary with a list comprehension. The list comprehension is made up of length 2 tuples (`i, i**3`). When a dictionary is initialized with a sequence of length 2 tuples, it assumes the first element of the tuple `i` is the *key* and the second element i**3is the *value*

```
['First', 'Last', 'Age', 'Weight', 'Height', 'Birthday']
```

Notice how this works like a pipeline: `fh.readline` returns a line of text as a string; we call the string method `strip` which returns a string with all white space (spaces, tabs, newlines) removed from the left and right; we then call the `split` method on this stripped string to split it into a list of strings.Notilretuiirt spht a46(sp)1(ab)11(,yo)27orehwhiuthe.

3.2. E□ective interactive work

3.3. Customizing IPython

3.3.1. Basics. Flexible configuration system. It uses a configuration file which allows per-

CHAPTER 4

Numeric tut

Figure 5.3.2. The default marker plot, before marker customization

```
fontproperties: a matplotlib.font_manager.FontProperties instance
horizontalalignment or ha: [ 'center' | 'right' | 'left' ]
label: any string
lod: [True | False]
multialinment: [ 'left' | 'right' | 'center' ]
name or fontname: string eg, ['Sans' | 'Courier' | 'Helvetica' ...]
position: (x,y)
rotation: [ angle in degrees 'vertical' | '
```

```
from matplotlib.numerix.mlab import mean
x = n.arange(100)
y = n.take(x, range(10,20))
print mean(y)
```

For the remainder of this manual, the term numerix is used to mean either the Numeric or numarray package. To select numarray or Numeric from the prompt, run your matplotlib script with

```
> python myscript.py --numarray  # use numarray
> python myscript.py --Numeric   # use Numeric
```

Typically, however, users will choose one or the other and make this setting in their rc file using either numerix : Numeric or numerix : numarray.

Figure 5.7.1. A simple image plot of a 2D matrix, using nearest neighbor interpolation and the `jet` colormap.

```
In [15]:In [15]: arange(100.0); x.shape = (10,10)In [16]: im = imshow(x, interpolation='neares
```

which creates the image 1(magd)1((e)1(s)-358(B58(Ba63T7(i)1(c)gu(r)1(e)1(a63T)1(7.)-71)1-775(5)Y3(TPou357(cr

5.8. Customizing text and mathematical expressions

5.9. Event handling: Tracking the mouse and keyboard

CHAPTER 6

A tour of scipy

Purpose
Module overview
Some examples

CHAPTER 7

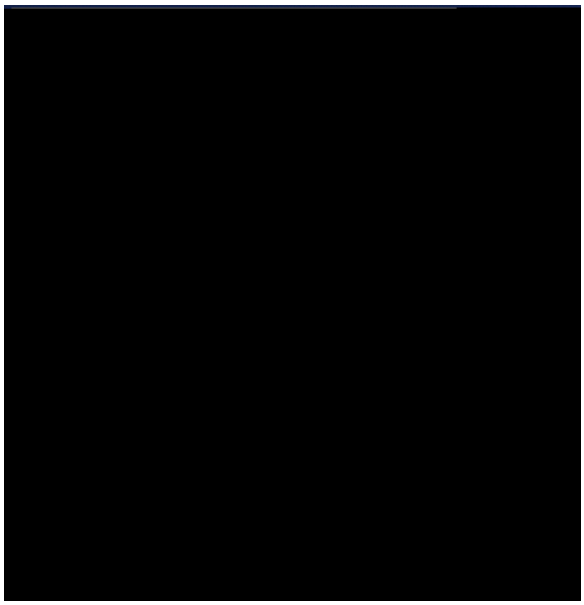
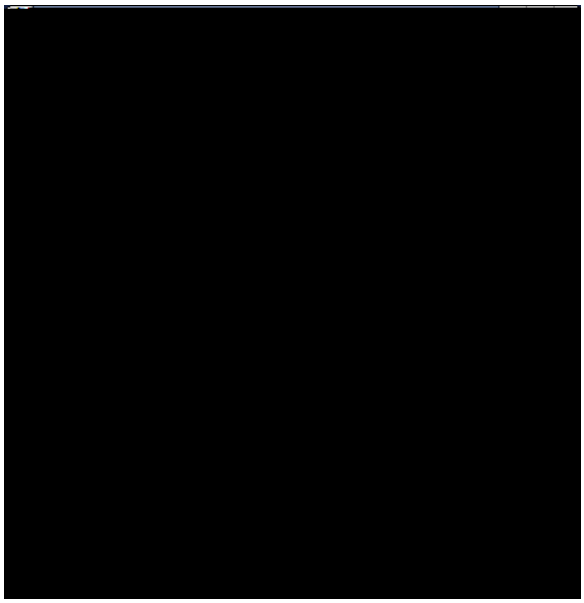


Figure 7.1.1. A Cube, brought to you by VTK

```
# Ready, set,
```



CHAPTER 8

3D visualization with MayaVi

8.1. Generalities

8.2. Scripted examples

Bibliography

[1] D. Beasley.