Python and Scientific Computing Notes

John D. Hunter
Fernando P

CHAPTER 1

Why python?

hello why python

std::cout

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 memory to handle them. The integer type, if it overflows, willThe

2.3. Accessing the standard library

Arithmetic is fine, but before long you may find yourself tiring of it and wanting to compute logarithms and exponentials,)Tj 11.9704 0 Td (sines)Tj 23.8626 0 Td (and)Tj 19.3097 0 Td (cosines)Tj /R53 9.96264 Tf -120.

```
        Return the sine of x (measured in radians).
```

and for the whole math library

```
>>> help(math)
Help on module math:

NAME
    math

FILE
    /usr/local/lib/python2.3/lib-dynload/math.so

DESCRIPTION
    This module is always available.  It provides access to the
    mathematical functions defined by the C standard.

FUNCTIONS
    acos(...)
        acos(x)

        Return the arc cosine (measured in radians) of x.

    asin(...)
        asin(x)

        Return the arc sine (measured in radians) of x.
```

And much more which is snipped. Likewise, we can get information on the complex object in the same way

```
>>> x = complex(0,1)
>>> dir(x)
['__abs__', '__add__', '__class__', '__coerce__', '__de-
lattr__', '__div__', '__div-
mod__', '__doc__', '__eq__', '__float__', '__floor-
div__', '__ge__', '__getattribute__', '__get-
newargs__', '__gt__', '__hash__', '__init__', '__int__', '__le__', '__long__', '__lt__',
vmod__', '__reduce__', '__reduce_ex__', '__repr__', '__rfloor-
div__', '__rmod__', '__rmul__', '__rpow__', '__rsub__', '__rtrue-
div__', '__setattr__', '__str__', '__sub__', '__truediv__', 'conju-
gate', 'imag', 'real']
```

Notice that called `dir` **or** `help` **on the** 38-22388RR4499(00264jT7424465009091d901)-11.8264103-05T06(ddRe2)T9

```
>>>
```

```
    print fname
```

Now as promised, this will print out the 4 file names above, but it has three flaws: it doesn't scale to 10 or more files, it is inefficient, and it is not cross platform. It doesn't scale because it hard-codes the '0' after myexp, it is inefficient because to add several strings requires the creation of temporary strings, and it is not cross-platform because it hard-codes the directory separator '/'.

```
# On the path to elightenment
for i in (1,2,3,4):
    T
```

```
>>> x = []                      # create the empty list
>>> x.append(1)                 # add the integer one to it
>>> x.extend(['hi', 'mom'])  # append two strings to it
>>> x
[1, 'hi', 'mom']
>>> x.reverse()                 # reverse the list, in place
>>> x
['mom', 'hi', 1]
>>> len(x)
3
```

We mentioned list comprehensions in the last section

The first line use used to create an instance of the class Normalize**, and the special method** __init__ **is implicitly called. The second line implicitly calls the special** __call__**method**

```
>>> norm = Normalize(65356) # good for 16 bit images
>>> norm(255)               # call this function
0.0039017075708427688
# We can reset the maxval attribute, and the call method
# is automagically updated
>>> norm.maxval = 255       # reset the maxval
>>> norm(255)               # and call it again
1.0
# We can pass the norm instance to the psd function we de-
fined above, which
# is expecting a function
>>> pdf(X, normalize=norm)
```

Exercise **2.12**. **Pretend that** complex **were not built-in to the python core, and write your own 87.2905 0 that**

2.13. Files and file like objects

Working with files is one of the most common and important things we do in scientific computing because that is usually where the data lives. In Section2.4, we went through the mechanics of automatically building file names like

```
data/myexp01.dat
data/myexp02.dat
data/myexp03.dat
data/myexp04.dat
```

but we didn't actually do anything with these files. Here we'll show how to read 23.977.5n709(9)Tj Tdn(the)Tj 27.268

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

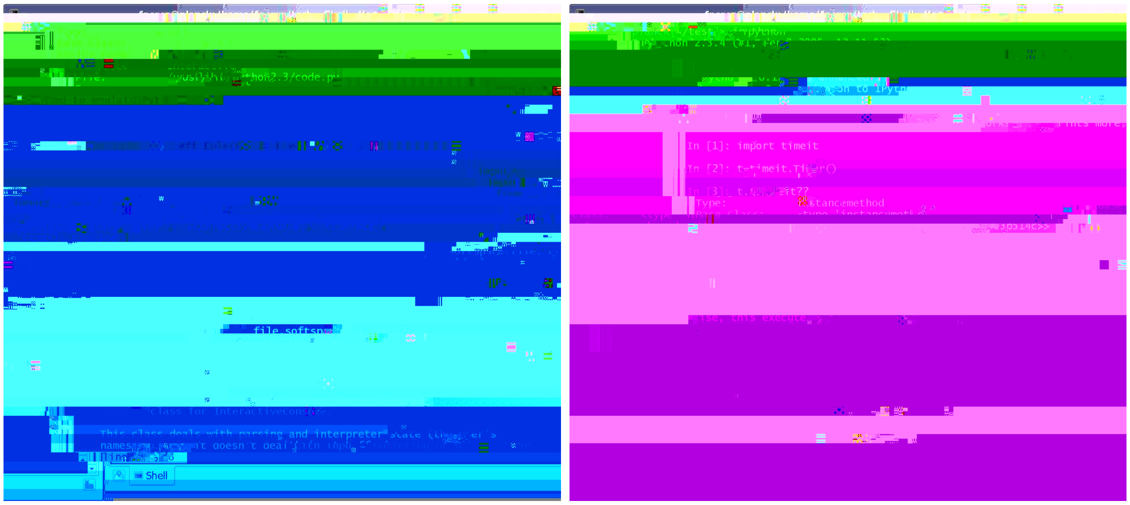
Notice how this works like a pipeline: `fh.readline` **returns a line of text as a string; we call the string method** `strip` **which returns returns** [illegible]

- **Dynamic object in**

3.2. Effective interactive work

IPython has been designed to try to make interactive work as fluid and efficient as possible. All of its features try to maximize the output-per-keystroke, so that as you work at an interactive console, minimal typing produces results. It makes extensive use of the

```
In [4]: %cd .. # ut cd ys orkAPfjT (  aoTf  d( A(T ori  TT  hAPE fpere foo f  T
```



```
In [1]: import timeit

In [2]: t=timeit.Timer()
```

Figure 3.2.1. **IPython can show syntax-highlighted source code for objects whose source is available.**

This can be done for entire modules, as in the prvious example, for individual functions, or even methods of object instances. The

At any time, your input history remains available. The `%hist` command can show you all previous input, without line numbers if desired (option `-n`) so you can directly copy and paste code either back in IPython or in a text editor. You can also save all your history b

You can run single statements

CHAPTER 4

CHAPTER 5

Introduction

to click Back **on your web browser before visiting a new page {nothing happens. The home**

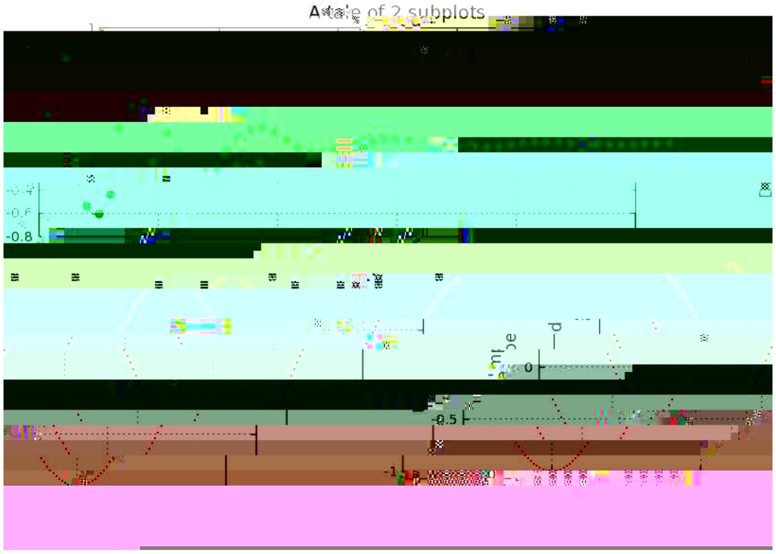


Figure 5.2.2. **It's easy to create multiple axes and subplots.**

```
t3 = arange(0.0, 2.0, 0.01)

# create and upper subplot and make it current
subplot(211)
l1, l2 = plot(t1, f(t1), 'bo', t2, f(t2), 'k--')
set(l1, markerfacecolor='g')
                           grid(True)
      'A tale of 2 subplots') title(
       'Damped oscillation')  ylabel(

# create a lower subplot and make it current
subplot(212)
plot(t3, cos(2*pi*t3), 'r.')
                           grid(True)
       'time (s)')         xlabel(
ylabel('Undamped')
savefig('../fig/mpl_subplot_demo')
show()
```

The examples in this section show how to make the simplest line plots. We'll delve into other plot types shortly, but let's take a brief tour looking at how to customize matplotlib to use it in a wide variety of settings.

5.3. A common interface to Numeric and numarray

se8tiigle **Currently the python comput 18.669738Td (comput 18.6j 10.19528121 Td (Td 8t)Tj 19.7un 0 Td (7of)39R22**

5.4. CUSTOMIZING

CHAPTER 6

A tour of scipy

CHAPTER

7.4. WORKING WITH MEDICAL WITH

W

isoActorVTK

Figure 7.4.2. **The cortical isosurface generated by a simple intensity based marching cubes application (40 lines of python listed above). More sophisticated image segmentation is available in the Insight Toolkit.**

CHAPTER 9

Interfacing with external libraries

9.1. weave

9.2. swig

9.3. f2py

9.4. Others

boost, pyrex, cxx

CHAPTER 10

lyx examples

See a [2, 9]