



Technical University of Lodz
Institute of Electronics

Algorithms and Data Structures

IPython

Łódź 2012





IPython – ipython.org

IP[y]: IPython Interactive Computing

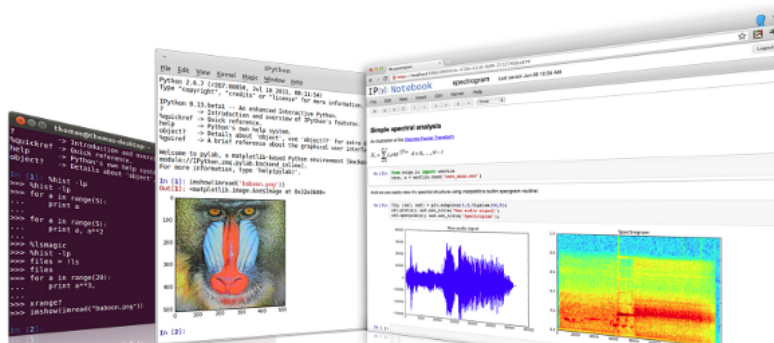
[Home](#) · [Download](#) · [Documentation](#) · [Videos](#) · [Wiki](#) · [News](#) · [Citation](#)

IPython provides a rich toolkit to help you make the most out of using Python, with:

- Powerful Python shells (terminal and Qt-based).
- A web-based notebook with the same core features but support for code, text, mathematical expressions, inline plots and other rich media.
- Support for interactive data visualization and use of GUI toolkits.
- Flexible, embeddable interpreters to load into your own projects.
- Easy to use, high performance tools for parallel computing.

To learn more about IPython, you can watch our [videos and screencasts](#), download our [talks and presentations](#), or read our [extensive documentation](#). IPython is open source (BSD license), and is used by a range of [other projects](#); add your project to that list if it uses IPython as a library, and please don't forget to [cite the project](#).

IPython supports Python 2.6 to 2.7 and 3.1 or newer. Our older 0.10 series supports Python 2.5, and can be used with Python 2.4.



Google™ Custom Search x

MEMORIAL

John Hunter

1968–2012

[J. Hunter Memorial Fund](#)

VERSIONS

Stable

0.13 – June 2012

[Download](#)

Development

pre-0.14

[Github](#)



LINKS

[Issue tracker](#)

Mailing lists:

[User](#) · [Dev](#)

[IRC channel](#)

(Freenode, #ipython)

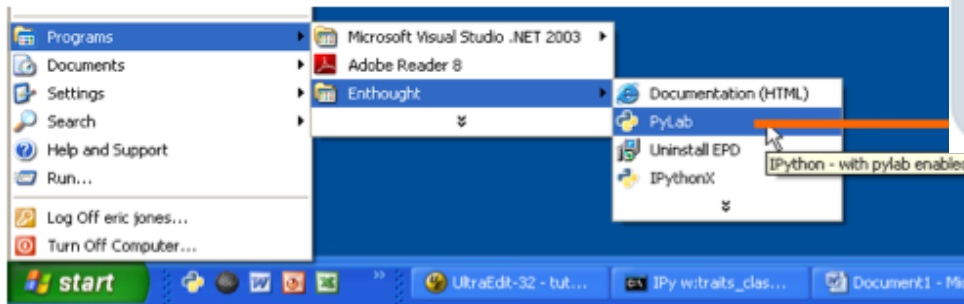
[Try IPython online](#)



IPython

ENTHOUGHT

Starting PyLab



IPython has a 'pylab' mode where it imports all of NumPy, Matplotlib, and SciPy into the namespace for you as a convenience. It also enables threading for showing plots.

or...



or... Mac OS

```
demo mk$  
demo mk$  
demo mk$  
demo mk$  
demo mk$  
demo mk$  
demo mk$  
demo mk$  
demo mk$  
demo mk$  
demo mk$  
demo mk$ ipython --pylab
```

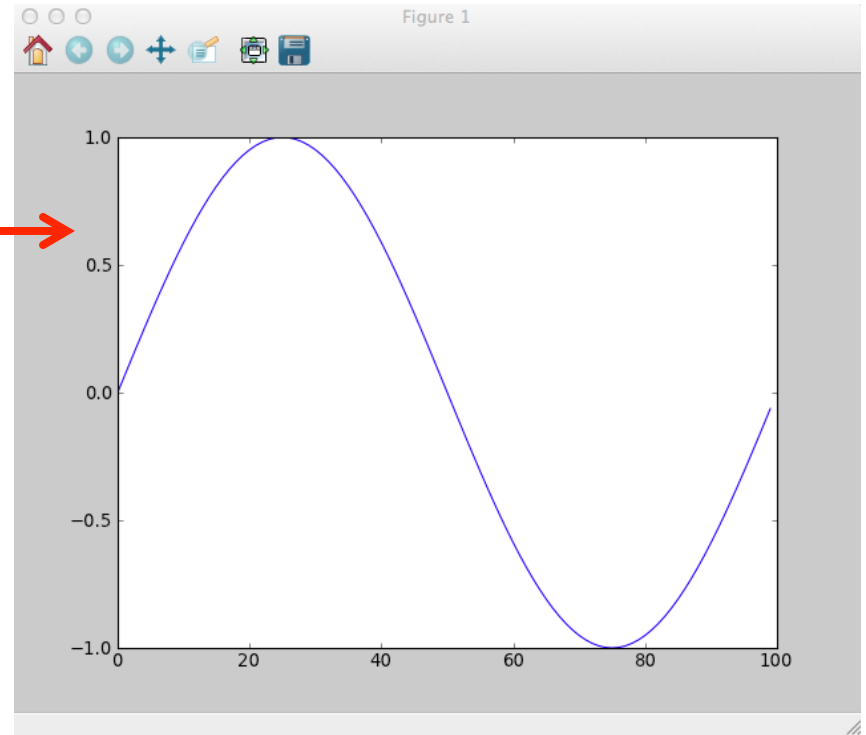
```
PyLab  
Enthought Python Distribution (4.0.30001) -- http://code.enthought.com  
Python 2.5.2 (EPD 2.5.2001) (release25-maint:60919M, Feb 21 2008, 10:31:43) [MSC  
v.1310 32 bit (Intel)]  
Type "copyright", "credits" or "license" for more information.  
  
IPython 0.9.beta -- An enhanced Interactive Python.  
? -> Introduction and overview of IPython's features.  
%quickref -> Quick reference.  
help -> Python's own help system.  
object? -> Details about 'object'. ?? prints more.  
  
Welcome to pylab, a matplotlib-based Python environment.  
For more information, type 'help(pylab)'.  
  
In [1]: a = arange(100.)  
In [2]: plot(a, sin(a*pi/50.))  
Out[2]: [matplotlib.lines.Line2D object at 0x02ADE650]  
In [3]:
```



PyLab – Interactive Python Environment

```
demo — Python — 74x15
Welcome to pylab, a matplotlib-based Python environment [backend: WXAgg].
For more information, type 'help(pylab)'.

In [1]: a = arange(100.)
In [2]: plot(a*sin(pi/50))
Out[2]: [<matplotlib.lines.Line2D at 0x69b85d0>]
In [3]: plot(a, a*sin(pi/50))
Out[3]: [<matplotlib.lines.Line2D at 0x5c7e2d0>]
In [4]: plot(a, sin(a*pi/50))
Out[4]: [<matplotlib.lines.Line2D at 0x6a59850>]
In [5]: □
```



`x = arange(100.)`

`plot(x, sin(x*pi/50))`

`#Now change display option`

`hold(False)`

`plot(x, cos(x*pi/50))`

`# and`

`hold(True)`

`plot(x, sin(x*pi/50))`

`plot(x, cos(x*pi/50))`



IPython Introducion

STANDARD PYTHON

```
In [1]: a=1
```

```
In [2]: a
```

```
Out [2]: 1
```

AVAILABLE VARIABLES

```
In [3]: b = [1,2,3]
```

```
# Comments # (to the end of the line)
```

```
# List available variables
```

```
In [4]: %whos
```

```
Variable Type Data/Info
```

```
-----
```

```
a      int    1
```

```
b      list   n=3
```

RESET & RELOAD

```
# Reset user defined variables
```

```
In [5]: %reset
```

```
Once deleted, variables cannot be recovered. Proceed  
(y/[n])? y
```

```
# Let's check our namespace
```

```
In [6]: %whos
```

```
Interactive namespace is empty.
```

```
WARNING: %reset also removes the names imported  
by PyLab.
```

```
In [7]: plot
```

```
NameError: name 'plot' is not defined
```

```
# Reload
```

```
In [8]: %pylab
```

```
Welcome to pylab, a matplotlib-based Python  
environment...
```



Directory navigation in IPython

Get path to current working directory

```
In [9]: pwd
```

```
Out[9]: u'/Users/mk'
```

Change directory (note Unix style forward slashes)

```
In [10]: cd Desktop/
```

Tab completion helps you find and type directory and file names

Check the following commands (with *pwd* command):

- *cd ..*
- *cd*

List directory contents (Unix style, not “dir”)

```
In [11]: ls
```

```
Desktop/  Library/  Public/  Documents/  Movies/  Sites/  Downloads/  
          Music/  mipav/  Dropbox/  Pictures/  mySoft/
```



Directory navigation in IPython

#Check following commands:

- `ls -l`
- `ls -a`
- `ls -l`

Go to the D: drive and create folder:

`#python_lecture_1_tests`

`cd d://`

`mkdir python_lecture_1_test`

#Go inside this directory and list its content

`cd python_lecture_1_test`

`ls`

edit and run (if exists) or create new script

`edit test_file_1.py`

fill the script with commands (the detailed
explanation will appear during next lectures)

```
import pylab as pl
import numpy as np
```

```
x = np.arange(100)
y = np.sin(x*np.pi/50.)
pl.plot(x,y)
```

```
pl.show()
```

in the IPython terminal write:

```
plot(x,cos(x*pi/50.))
```

IPython is interactive!!!



Directory navigation in IPython

Run script insight IPython(without editing it)

Interactive mode

```
%run test_file_1.py
```

To quit IPython use command

```
In [12]: quit()
```

Run script from the sysetm level (Windos, Unix, Mac OS)

Batch mode

Run system console and write

```
python test_file_1.py
```




Directory Bookmarks

Bookmark the python_lecture_test directory

```
cd python_lecture_test
```

#bookmark as some name (alias)

```
%bookmark lect1
```

bookmark 2 more folders

lists bookmarked directories

```
%bookmark -l
```

go easily to bookmarked directory

```
# go to home directory cd
```

go to lect1 directory

```
cd lect1
```



Function Info and Help

Follow the command '?' to print its documentation

```
>>> squeeze?
```

Remove single-dimensional entries from the shape of an array.

Parameters

a : array_like

Input data.

Returns

squeezed : ndarray

The input array, but with with all dimensions of length 1 removed. Whenever possible, a view on `a` is returned.



Function Info and Help

```
# Follow a command with '?' to print its source code
```

```
# compare: squeeze? And squeeze??
```

```
>>> squeeze??
```

```
def squeeze(a):
```

```
    """ Remove single-dimensional entries from the shape of an array.
```

Examples

```
>>> x = np.array([[[[0], [1], [2]]]])
```

```
>>> x.shape
```

```
(1, 3, 1)
```

```
>>> np.squeeze(x).shape
```

```
(3,) """
```

```
try:
```

```
    squeeze = a.squeeze
```

```
except AttributeError:
```

```
    return _wrapit(a, 'squeeze')
```

```
return squeeze()
```



Function Info and Help

History command

```
# list previous commands. Use 'magic' % because 'hist' is histogram function in pylab  
%hist
```

Input History

```
# list stringfrom prompt [2]
```

```
In[15] a = 5
```

```
...
```

```
>>> _i15
```

```
>>> u'a = 5'
```

```
# The up and down arrows scroll through your ipython input history
```



IPython magic functions

<http://ipython.org/ipython-doc/dev/interactive/tutorial.html#magic-functions>

```
>>> %lsmagic
```

Available magic functions:

```
%alias %autocall %autoindent %automagic %bookmark %cd %colors %config %cpaste  
%debug %dhist %dirs %doctest_mode %ed %edit %env %fread %fwrite %gui %hist %history  
%inplace %install_default_config %install_profiles %load_ext %loadpy %logoff %logon  
%logstart %logstate %logstop %lsmagic %macro %magic %notebook %page %paste  
%pastebin %pdb %pdef %pdoc %pfile %pinfo %pinfo2 %pm %pop_err %pop_print %popd  
%pprint %precision %print_methods %print_traits %profile %prun %psearch %psource %pt  
%push_err %push_print %pushd %pwd %pycat %pylab %quickref %recall %rehashx  
%reload_ext %rep %replace_context %rerun %reset %reset_selective %run %run_examples  
%save %sc %store %sx %sym %tb %time %timeit %unalias %unload_ext %who %who_ls  
%whos %xdel %xmode
```



Read simple Tracebacks

```
>>> 100 + "Hello world!"
```

```
-----  
TypeError                                Traceback (most recent call last)  
/Users/mk/<ipython-input-7-3d139ed9b46e> in <module>()  
----> 1 100 + "Hello world!"  
TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

```
>>> undevided_variable + 1
```

```
-----  
NameError                                Traceback (most recent call last)  
/Users/mk/<ipython-input-8-8514a074ce6b> in <module>()  
----> 1 undevided_variable + 1  
NameError: name 'undevided_variable' is not defined
```



Interactive Calculator

adding two values

```
>>> 5 + 10  
15
```

setting a variable

```
>>> a = 5  
>>> a  
5
```

checking the variable's type

```
>>> type(a)  
int
```

an arbitrary long integer

```
>>> a = 12345678901234567890
```

```
>>> a  
12345678901234567890L  
>>> type(a)  
long
```

remove 'a' from the 'namespace'

```
>>> del a  
>>> a
```

```
NameError Traceback (most recent call last)  
/Users/mk/<ipython-input-27-60b725f10c9c>  
in <module>()  
----> 1 a
```

NameError: name 'a' is not defined



Interactive Calculator

real numbers

```
>>> b = 1.4 + 2.3
```

```
>>> b
```

```
3.6999999999999997
```

“prettier” version

```
>>> print b
```

```
3.7
```

```
>>> type(b)
```

```
float
```

complex numbers

```
c = 5+5.5j
```

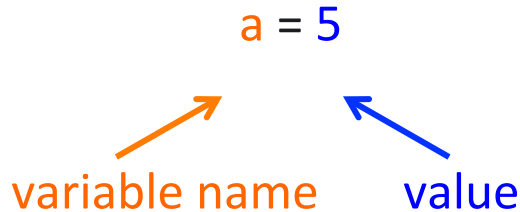
```
>>> c
```

```
(5+5.5j)
```




Variable declaration and initialization

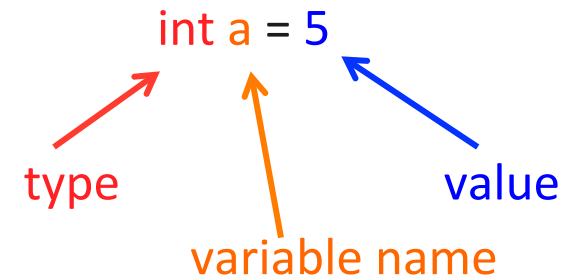
Matlab/Python



a = 'hello world'

a = 5.56

C/C++/Java



~~a = "hello world"~~
~~a = 5.5~~

float b = 5.5

std::string text = "Hello world" (C++)



Interactive Calculator

arithmetic operations

```
>>> 1+2-(3*4/6)**5+(7%2)
-27
```

simple math functions

```
>>> abs(-6e3)
6000.0
```

```
>>> max(0, min(10, -1, 4, 3))
0
```

```
>>> round(2.71235897456)
3.0
```

Python math functions:

<http://docs.python.org/library/math.html>

casting

```
>>> int(2.718281828)
2
```

```
>>> float(2)
2.0
```

```
>>> 1 + 2.
3.0
```

```
>>> int('456')
456
```

```
>>> str(65)
'65' ← string
```

in place operations +=, -=, *=, /=, etc.

```
>>> b = 2.5
```

```
>>> b+=0.5          # b = b+5
```

```
>>> b
```

3.0



Definition of function



Exercise



Homework