IPython: Python at your fingertips

Fernando Pérez
Fernando.Perez@berkeley.edu
http://fperez.org

Brian E. Granger (Cal Poly San Luis Obispo), Min Ragan-Kelley (UC Berkeley), Thomas Kluyver (U Sheffield), Evan Patterson (Enthought).

PyCon 2012
March 9, 2012
Why IPython?

I is for interactive...

In scientific computing, we typically don’t know what we’re doing.

*Exploratory* computing is *not just for scientists*
Why IPython?

I is for interactive...

In scientific computing, we typically don’t know what we’re doing. *Exploratory* computing is *not just for scientists*
Why IPython?

I is for interactive...

In scientific computing, we typically don’t know what we’re doing.

Exploratory computing is not just for scientists.
Why IPython?

I is for interactive...

In scientific computing, we typically don’t know what we’re doing.

*Exploratory* computing is *not just for scientists*
Python: an excellent *base* for an interactive environment
I said a base...
Mmh, introspection?

```
dreamweaver[~] > python
Python 2.6.6 (r266:84292, Sep 15 2010, 16:22:56)
[GCC 4.4.5] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> ls
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'ls' is not defined
>>> os?
  File "<stdin>", line 1
    os?
    ^
SyntaxError: invalid syntax
>>> 
```
Basic comforts?

```
dreamweaver[~]$ python
Python 2.6.6 (r266:84292, Sep 15 2010, 16:22:56)
[GCC 4.4.5] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> ls
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'ls' is not defined
>>> os?
  File "<stdin>", line 1
  os?
^  
SyntaxError: invalid syntax
>>> execfile('~/scratch/err.py')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
IOError: [Errno 2] No such file or directory: '~/scratch/err.py'
>>> 
```
Useful error info

Python 2.6.6 (r266:84292, Sep 15 2010, 16:22:56)
[GCC 4.4.5] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> ls
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'ls' is not defined
>>> os?
  File "<stdin>", line 1
    os?
^  
SyntaxError: invalid syntax
>>> execfile('/~scratch/err.py')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
IOError: [Errno 2] No such file or directory: '/~scratch/err.py'
>>> execfile('/home/fperez/scratch/err.py')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
    File "~/home/fperez/scratch/err.py", line 9, in <module>
      foo33
NameError: name 'foo33' is not defined
>>>  
>>>  


We can do better...
dreamweaver[~]$ ipython
Python 2.6.6 (r266:84292, Sep 15 2010, 16:22:56)
Type "copyright", "credits" or "license" for more information.

IPython 0.11.dev -- An enhanced Interactive Python.
?   -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help     -> Python's own help system.
object?  -> Details about 'object', use 'object??' for extra details.

In [1]: ls ~/scratch/er*py
/home/fperez/scratch/err25.py  /home/fperez/scratch/error.py*
/home/fperez/scratch/err_comps.py /home/fperez/scratch/err.py

In [2]:
Some object details?

Type: module
Base Class: <type 'module'>
String Form: <module 'os' from '/usr/lib/python2.6/os.pyc'>
Namespace: Interactive
File: /usr/lib/python2.6/os.py
Docstring:

OS routines for Mac, NT, or Posix depending on what system we're on.

This exports:
- all functions from posix, nt, os2, or ce, e.g. unlink, stat, etc.
- os.path is one of the modules posixpath, or ntpath
- os.name is 'posix', 'nt', 'os2', 'ce' or 'riscos'
- os.getcwd is a string representing the current directory ('. ' or '::')
- os.pardir is a string representing the parent directory ('..' or '::')
- os.sep is the (or a most common) pathname separator ('/' or ':' or '\\')
- os.extsep is the extension separator ('.' or '/')
- os.altsep is the alternate pathname separator (None or '/')
- os.pathsep is the component separator used in $PATH etc.
- os.linesep is the line separator in text files ('' or '
' or '
')
- os.defpath is the default search path for executables
- os.devnull is the file path of the null device ('/dev/null', etc.)

Programs that import and use 'os' stand a better chance of being

lines 1-23
Utilities needed to emulate Python's interactive interpreter.

# Inspired by similar code by Jeff Epler and Fredrik Lundh.

```python
import sys
import traceback
from codeop import CommandCompiler, compile_command

__all__ = ["InteractiveInterpreter", "InteractiveConsole", "interact", "compile_command"]

def softspace(file, newvalue):
    oldvalue = 0
    try:
        oldvalue = file.softspace
    except AttributeError:
        pass
    try:
        file.softspace = newvalue
lines 1-28
```
When things go wrong

```
In [13]: run ~/scratch/error
reps: 5

ValueError
Traceback (most recent call last)
/home/fperez/scratch/error.py in <module>()
    70 if __name__ == '__main__':
    71     #explode()

---
    72     main()
    73     g2='another global'

/home/fperez/scratch/error.py in main()
    60     array_num = zeros(size,'d')
    61     for i in xrange(reps):

---
    62         RampNum(array_num, size, 0.0, 1.0)
    63         RNtime = time.clock()-t0
    64         print 'RampNum time:', RNtime

/home/fperez/scratch/error.py in RampNum(result, size, start, end)
    43     tmp = zeros(size+1)
    44     step = (end-start)/(size-1-tmp)

---
    45     result[:] = arange(size)*step + start
    46
    47     def main():

ValueError: shape mismatch: objects cannot be broadcast to a single shape
```
Interactive architecture

IPython Kernel

ZeroMQ

Terminal console

Qt Console

Web Notebook

Client: monitor email, publish, ..
Terminal console with visualization

```python
In [1]: import math, numpy
In [2]: from scipy.integrate import quad
In [3]: from scipy.special import j0
In [4]: def j0i(x):
    ...:     """Integral form of J_0(x)"""
    ...:     def integrand(phi):
    ...:         return math.cos(x*math.sin(phi))
    ...:     return (1.0/math.pi)*quad(integrand,0,math.pi)[0]
    ...
In [5]: x = numpy.linspace(0,20,200) # sample grid: 200 points between 0 and 20
In [6]: y = j0i(x) # sample J0 at all values of x
In [7]: x1 = x[:10] # subsample the original grid every 10th point
In [8]: y1 = map(j0i,x1) # evaluate the integral form at all points in x1
In [9]: # Make a plot with these values (the ; suppresses output)
In [10]: plt.plot(x,y,label="J_0(x)");
In [11]: plt.plot(x1,y1,'ro',label="J_0(integ)");
In [12]: axhline(0,color='green',label='nolegend_');
In [13]: title(r'\textbf{Verify } J_0(x) = \frac{1}{\pi} \int_0^\pi \cos(x \sin \phi) \, d\phi$
In [14]: xlabel('x$');
In [15]: legend();
In [16]: matshow(numpy.random.randn((32,32)))
Out[16]: <matplotlib.figure.Figure instance at 0x4630042c>
```
Qt console: inline plots, html, multiline editing, ...
Evan Patterson (Enthought)

In [1]: import scipy.linalg as la
       ...: mineigs = []
       ...: n = 256
       ...: for i in range(10):
       ...:     a = rand(n, n)
       ...:     mineigs.append(la.eigvals(a).min().real)
       ...: mean(mineigs)

Out[1]: -4.589467643237938

In [2]: %run mapping_seismic_stations.py

Seismic stations in the Himalaya

In [3]: |
This example computes PI to certain precision using 4 processors and a monte carlo simulation.

```python
import random
from mpi4py import MPI
comm = MPI.COMM_WORLD
import numpy as np

def computePi(nsamples):
    rank, size = comm.Get_rank(), comm.Get_size()
    oldpi, pi, mypi = 0.0, 0.0, 0.0

done = False
while not done:
    inside = 0
    for i in xrange(nsamples):
        x = random.random()
        y = random.random()
        if (x**2) + (y**2) < 1:
            inside += 1

    oldpi = pi
    mypi = (inside * 1.0) / nsamples
    comm.Allreduce(mypi, pi, op=MPI.SUM)
```

```python
In [1]: from IPython.parallel import Client
In [2]: rc = Client()
In [3]: rc.ids
[0, 1, 2, 3]
In [4]: dview = rc[:].load_all_engines
In [5]: serial_result = map(lambda x:x**10, range(32))
In [6]: parallel_result = dview.map_sync(lambda x:
             x**10, range(32))
In [7]: serial_result == parallel_result
True
In [8]: parallel_result
[0,
  1,
  1024,
  59049,
  1048576,
  9765625,
  60466176,
  282475249,
  1073741824,
  3486784401L,
  10000000000L,
  25937424601L,
  61917364224L,
  137858491840L,
  289254654976L,
  576659399888L]`
Simple spectral analysis

An illustration of the Discrete Fourier Transform

\[ X_k = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi}{N} kn} \quad k = 0, \ldots, N - 1 \]

using windowing, to reveal the frequency content of a sound signal.

We begin by loading a data file using SciPy’s audio file support:

```
In [1]: from scipy.io import wavfile
    : rate, x = wavfile.read('test_mono.wav')
```

And we can easily view its spectral structure using matplotlib’s built-in `specgram` routine:

```
In [2]: fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(12, 4))
    : ax1.plot(x); ax1.set_title('Raw audio signal')
    : ax2.specgram(x); ax2.set_title('Spectrogram');
```
Interactive and high-level parallel APIs
Min Ragan-Kelley, Brian Granger

[Diagram showing the interaction between clients Alice and Bob with IPython Controller and engines]

- Instructions
- Objects
October/November 2001: “just a little afternoon hack“

- $PYTHONSTARTUP: ipython-0.0.1.py (259 lines)
- IPP (Interactive Python Prompt) by Janko Hauser (Oceanography)
- LazyPython by Nathan Gray (CalTech)

2002: Drop John Hunter’s Gnuplot patches: matplotlib

2004: Brian Granger, Min Ragan-Kelley: Parallel on Twisted...

2005-2009: Mayavi, Wx support, refactoring; slow period.

2010: discover ØMQ, Enthought support.
- Move to Git/Github.
- Build Qt console (Evan Patterson).
- Rewrite parallel support with ZeroMQ.
- Python 3 port (Thomas Kluyver).

How did we get here?
A brief history of IPython

- **October/November 2001:** “just a little afternoon hack“
  - $PYTHONSTARTUP: ipython-0.0.1.py (259 lines)
  - IPP (Interactive Python Prompt) by Janko Hauser (Oceanography)
  - LazyPython by Nathan Gray (CalTech)

- **2002:** Drop John Hunter’s Gnuplot patches: matplotlib

- **2004:** Brian Granger, Min Ragan-Kelley: Parallel on Twisted...

- **2005-2009:** Mayavi, Wx support, refactoring; slow period.

- **2010:** discover ØMQ, Enthought support.
  - Move to Git/Github.
  - Build Qt console (Evan Patterson).
  - Rewrite parallel support with ZeroMQ.
  - Python 3 port (Thomas Kluyver).

- **2011:** Web Notebook.
How did we get here?
A brief history of IPython

- **October/November 2001**: “just a little afternoon hack“
  - `$PYTHONSTARTUP`: ipython-0.0.1.py (259 lines)
  - IPP (Interactive Python Prompt) by Janko Hauser (Oceanography)
  - LazyPython by Nathan Gray (CalTech)

- **2002**: Drop John Hunter’s Gnuplot patches: matplotlib

- **2004**: Brian Granger, Min Ragan-Kelley: Parallel on Twisted...

- **2005-2009**: Mayavi, Wx support, refactoring; slow period.

- **2010**: discover ØMQ, Enthought support.
  - Move to Git/Github.
  - Build Qt console (Evan Patterson).
  - Rewrite parallel support with ZeroMQ.
  - Python 3 port (Thomas Kluyver).

- **2011**: Web Notebook.
How did we get here?
A brief history of IPython

- **October/November 2001:** “just a little afternoon hack“
  - $PYTHONSTARTUP: ipython-0.0.1.py (259 lines)
  - IPP (Interactive Python Prompt) by Janko Hauser (Oceanography)
  - LazyPython by Nathan Gray (CalTech)

- **2002:** Drop John Hunter’s Gnuplot patches: matplotlib

- **2004:** Brian Granger, Min Ragan-Kelley: Parallel on Twisted...

- **2005-2009:** Mayavi, Wx support, refactoring; slow period.

- **2010:** discover ØMQ, Enthought support.
  - Move to Git/Github.
  - Build Qt console (Evan Patterson).
  - Rewrite parallel support with ZeroMQ.
  - Python 3 port (Thomas Kluyver).

- **2011:** Web Notebook.
How did we get here?
A brief history of IPython

- October/November 2001: “just a little afternoon hack“
  - $PYTHONSTARTUP: ipython-0.0.1.py (259 lines)
  - IPP (Interactive Python Prompt) by Janko Hauser (Oceanography)
  - LazyPython by Nathan Gray (CalTech)

- 2002: Drop John Hunter’s Gnuplot patches: matplotlib

- 2004: Brian Granger, Min Ragan-Kelley: Parallel on Twisted...

- 2005-2009: Mayavi, Wx support, refactoring; slow period.

- **2010: discover ØMQ, Enthought support.**
  - Move to Git/Github.
  - Build Qt console (Evan Patterson).
  - Rewrite parallel support with ZeroMQ.
  - Python 3 port (Thomas Kluyver).

How did we get here?
A brief history of IPython

- October/November 2001: “just a little afternoon hack“
  - $PYTHONSTARTUP: ipython-0.0.1.py (259 lines)
  - IPP (Interactive Python Prompt) by Janko Hauser (Oceanography)
  - LazyPython by Nathan Gray (CalTech)

- 2002: Drop John Hunter’s Gnuplot patches: matplotlib

- 2004: Brian Granger, Min Ragan-Kelley: Parallel on Twisted...

- 2005-2009: Mayavi, Wx support, refactoring; slow period.

- 2010: discover ØMQ, Enthought support.
  - Move to Git/Github.
  - Build Qt console (Evan Patterson).
  - Rewrite parallel support with ZeroMQ.
  - Python 3 port (Thomas Kluyver).

- **2011: Web Notebook.**
(Incomplete) Cast of Characters

- **Brian Granger** - Physics, Cal State San Luis Obispo
- **Min Ragan-Kelley** - UC Berkeley
- **Thomas Kluyver** - U. Sheffield
- **Jörgen Stenarson** - SP Technical Research Institute of Sweden
- **Paul Ivanov** - UC Berkeley
- **Robert Kern** - Enthought
- **Evan Patterson** - Caltech/Enthought
- Stefan van der Walt - UC Berkeley
- Prabhu Ramachandran - Aerospace Engineering, IIT Bombay
- Satra Ghosh - MIT Neuroscience
- Gaël Varoquaux - Neurospin (Orsay, France)
- Ville Vainio - CS, Tampere University of Technology, Finland
- Ondrej Certik - Physics, U Nevada Reno
- Darren Dale - Cornell
- Justin Riley - MIT
- Mark Voorhies - UC San Francisco
- Nicholas Rougier - INRIA Nancy Grand Est
- Thomas Spura - Fedora project
- Julian Taylor - Debian/Ubuntu
- Many more! (~140 commit authors)
Some quick stats. http://www/ohloh.net/p/ipython
Other projects using IPython

Scientific
- **EPD**: Enthought Python Distribution.
- **Sage**: open source mathematics.
- **PyRAF**: Space Telescope Science Institute
- **CASA**: Nat. Radio Astronomy Observatory
- **Ganga**: CERN
- **PyMAD**: neutron spectrom., Laue Langevin
- **Sardana**: European Synchrotron Radiation
- **ASCEND**: eng. modeling (Carnegie Mellon).
- **JModelica**: dynamical systems.
- **DASH**: Denver Aerosol Sources and Health.
- **Trilinos**: Sandia National Lab.
- **DoD**: baseline configuration.
- **Mayavi**: 3d visualization, Enthought.
- **NiPyype**: computational pipelines, MIT.
- **PyIMSL** Studio, by Visual Numerics.
- ...

Web/Other
- **Visual Studio 2010**: MS.
- **Django**.
- **Turbo Gears**.
- **Pylons** web framework
- **Zope** and **Plone** CMS.
- **Axon Shell**, BBC **Kamaelia**.
- **Schevo** database.
- **Pitz**: distributed task/bug tracking.
- **iVR** (interactive Virtual Reality).
- **Movable Python** (portable Python environment).
- ...

...
- **Enthought**, Austin, TX: **Lots!**
- **Tech-X Corporation**, Boulder, CO: Parallel/notebook (previous versions)
- **Microsoft**: WinHPC support, Visual Studio integration
- **NIH**: via NiPy grant
- **NSF**: via Sage compmath grant
- **DoD/HPTi**: funding through Sept. 2012 (thanks to Jose Unpingco!).
IPython in brief

1. A better Python shell
2. Embeddable Kernel and powerful interactive clients
   1. Terminal
   2. Qt console
   3. Web notebook
3. Flexible parallel computing

http://ipython.org

http://github.com/ipython
Demo time!
Booth 201 - on your left by the entrance