Throw Matlab out of Windows...

- Matlab
- Python
- NumPy
- Scipy
- Matplotlib
- IPython

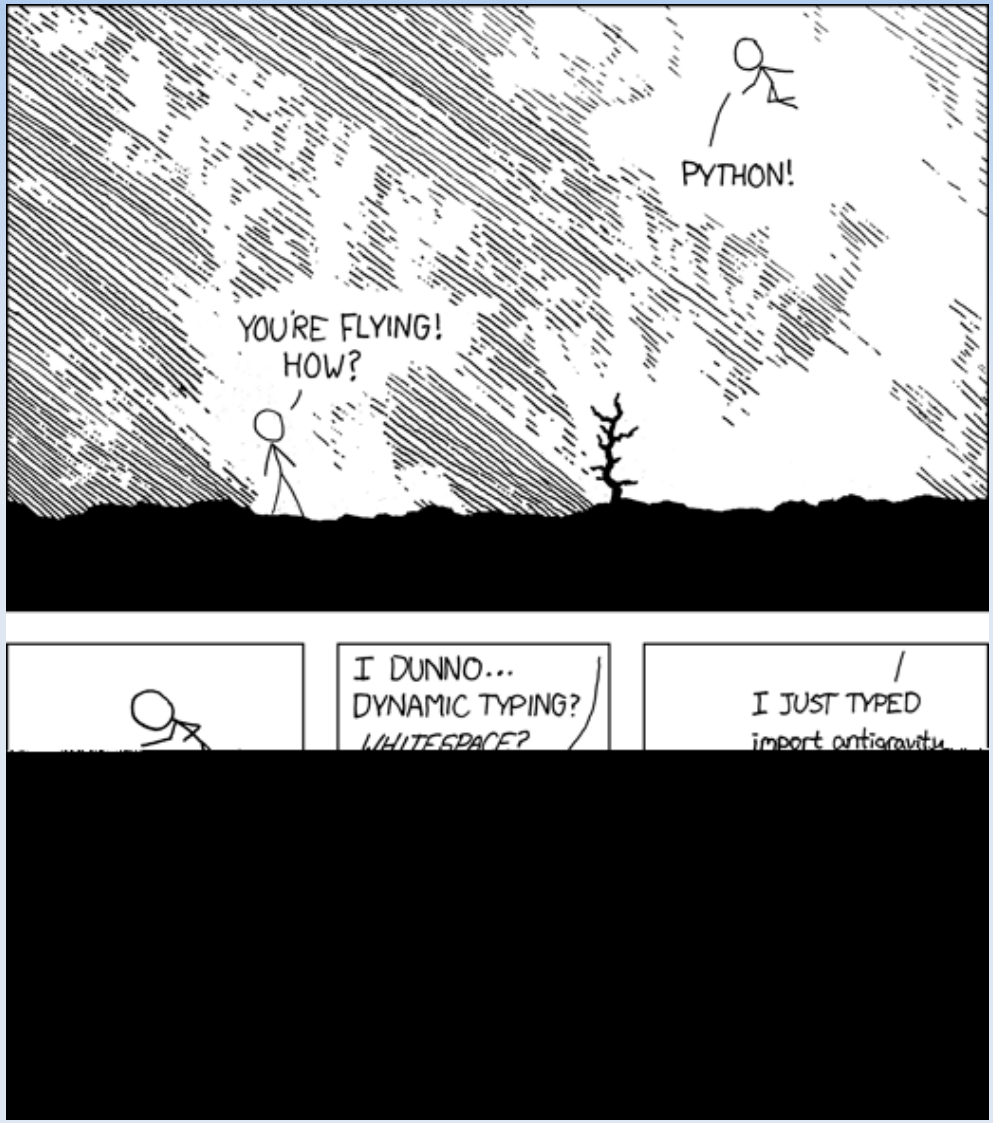
Matlab is bad!!!

- You have to pay for it
- Is not a general purpose language
- Not really Object-Oriented
-
- But bal bla bla...

Python

- 1989 Guido van Rossum

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one— and preferably only one —obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea — let's do more of those!



Python

- Multiplatform Object-Oriented Interpreted Language for genearal porpose programming
- Easy to learn, read, use
- Lots of libraries for doing everything
- Extensible (add new modules)
- Easy communication with other languages like C/C++/Fortran/whatever
- Interactive-shell programming

NumPy

- A powerful N-dimensional array object
- Integration of *Numeric* and *Numarray*
- Advanced array slicing methods (to select array elements)
- Convenient array reshaping methods
- Basic linear algebra functions
- Basic Fourier transforms
- Sophisticated random number capabilities

NumPy for Matlab Users

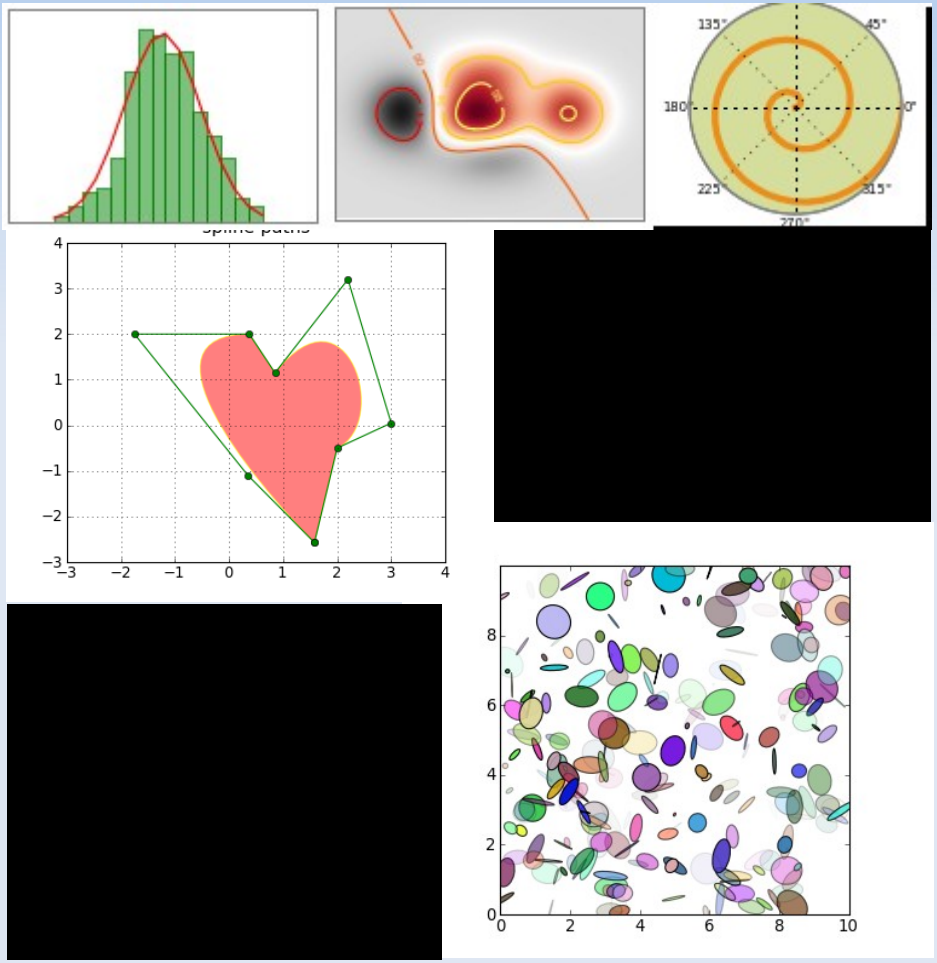
- Many similarities, but Not a Matlab clone
  *from numpy import **
- Multidimensional array class (also Matrix class)
  a=array([[1,2,3],[4,5,6]]);m=mat([[1,2,3],[4,5,6]])
- Indexing from 0
  r=a[0,0] # r=1
- Pass-by-reference
  def hola(p): # p can be a very big array
  p[1000,1000,1000]=1
- Broadcasting and Universal Functions
  r=a+a[0,:] #r=[[2,4,6][5,7,9]]
- Slicing as a view
  r=a[:2,:2] # r points to the same data as a

Scipy

- statistics
- optimization
- numerical integration
- interploation
- linear algebra
- signal processing
- image processing
- genetic algorithms
- Fourier transforms
- clustering
- maxentropy
- sparse matrix
- Weave (c/c++ integration)
- ODE solvers
- Input/Output
- Spatial structures (KDTree)
- Special functions

Matplotlib

- Library for making 2D plots of arrays in Python
- Highly Integrated with NumPy
- Publication quality
- Postscript output for TeX documents
- Embeddable in a graphical user interface for application development
- Making plots should be easy



Matplotlib

- Pylab Interface:

  similar to Matlab way of plotting (for prototiping)
- Matplotlib frontend or API:

  Object-Oriented interface (for appliactions)
- Backends:
  Implementation of the interface in a display device like PS, PDF, SVG, Agg, GTK, Wx...

IPython

- Environment for interactive and exploratory computing for Incremental development
- Enhanced interactive Python shell
- Architecture for interactive parallel computing
- Tab completion
- Explore objects with ?
- Magic commands
- History

Speed-Up Comparison

- Matlab ~ Python+Numpy
- Matlab < Python+Numpy+blitz
- Matlab+.mex ~ Python+Numpy+ctypes(easier)
- Now Pypy and Unladen Swallow can speed-up native python with JIT interpreters

More tools...

- Graphics: PIL, py-OpenCV, VTK, Mayavi, pyOpenGL, pyGame, pyWeek
- IDE: Python(x,y), drpython, SPE, Eric, WingIDE, BoaConstructor
- Other languages: ctypes, boost++, swig, F2py, Jithon, IronPython
- GPU: pyCuda, pyOpenCL
- Learning: Orange,Elefant,aima-python
- Parallelization: PyMPI, Process, ParallelPython

Problems

- Missing a plot property editor and format for figures in matplotlib (liek .fig in matlab)
- No standard format to save file (like .mat)
- No a clear way to clean the memory in IPython
- No standard IDE (although many possible)

Small Example: Snake

```python
#implementation of a snake

class snake(object):
    "Implementation of a snake"

    def __init__(self,head=(0,0),d=(0,1),body=[],col=(0.5)):
    "Initialize the snake"
        self.head=head #head position
        self.d=d #direction
        self.body=body #list of directions
        self.col=col #color

    def dir(self,d):
    "Set snake direction"
        aux=d+self.d
    #check if is not going in the opposite direction
        if not(aux[0]==0 and aux[1]==0):
            self.d=d

    def movegrow(self):
    "Move one step toward direction"
        self.head=self.head+self.d
        self.body.insert(0,-self.d) #add part into body

    def move(self):
```

```
                self.movegrow()
                self.body.pop()

            def draw(self,M):
            "Draw the snake into matrix M"
                dimy=M.shape[0]
                dimx=M.shape[1]
            #draw head
                M[self.head[0]%dimy,self.h
            #draw body
                aux=self.head
                for l in self.body:
                    aux=aux+l
                    M[aux[0]%dimy,aux[1]%d
ead[1]%dimx]=self.col
imx]=self.col
```

```python
class fastsnake(snake):
    "As snake but without redrawing everything"

    def __init__(self,head=(0,0),d=(0,1),body=[],col=(0.5),bkcol=(0.0)):
    "Initialize also bk color and cue"
        snake.__init__(self,head=head,d=d,body=body,col=col)
        self.bkcol=bkcol
        self.cue=self.head

    def move(self):
    "Move and delete last part"
        snake.movegrow(self)
        self.cue=self.cue-self.body[-1]
        self.body.pop()

    def draw(self,M):
    "Draw only head and delete cue"
        dimy=M.shape[0]
        dimx=M.shape[1]
        M[self.head[0]%dimy,self.head[1]%dimx]=self.col
        if M[self.cue[0]%dimy,self.cue[1]%dimx]==self.col:
            M[self.cue[0]%dimy,self.cue[1]%dimx]=self.bkcol
```

Small Example: Snake

```
# show a random snake

import pylab
import numpy
import snake

d=numpy.array([[1,0],[-1,0],[0,-1],[0,1]])

pylab.figure(1)
pylab.show()
s=snake.snake(numpy.array((25,25)),numpy.array((0,1)))
c=0
while True:
    M=numpy.zeros((50,50)) #clean the display
    s.draw(M) #draw the snake
    pylab.clf() #clear figure
    pylab.axis("off") #not show axis
    # show M
    pylab.imshow(M,interpolation="nearest")
    pylab.draw()
    #pylab.show()
    if c%5==0: #every 5 movements change direction randomly
        s.dir(d[numpy.random.randint(4)])
        s.movegrow()
    else:
        s.move()
    c=c+1
```