VISUALIZATION WITH PYTHON
Matplotlib

2D plotting library for python
Can be used in scripts and in interactive shell
Publication quality in various hardcopy formats
“Easy things easy, hard things possible”
Some 3D functionality
Matplotlib interfaces

- Simple command style functions similar to Matlab

```python
import pylab as pl
...
pl.plot(x, y)
```

- Powerful object oriented API for full control of plotting
Basic concepts

- Figure: the main container of a plot
- Axes: the “plotting” area, a figure can contain multiple Axes
- Graphical objects: lines, rectangles, text
- Command style functions are used for creating and manipulating figures, axes, lines, ...
- The command style interface is stateful:
  - track is kept about current figure and plotting area
Simple plot

plot: create a simple plot. Figure and axes are created if needed
Interactive vs. batch mode

In many installations batch mode is default
- Figures do not show up without `show()` function
- Batch mode is useful e.g. for writing out files during simulation and for heavy rendering

Mode can be controlled as:
- `ion()` : turn on interactive mode
- `ioff()` : turn on interactive mode
Multiple subplots

A tale of 2 subplots

- subplot: create multiple axes in the figure and switch between subplots
Histograms

```
hist : create histogram

Latex can be used with matplotlib
```
Bar and pie charts

**Bar:** bar charts

**Pie:** pie charts

Scores by group and gender

- **Men:** red bars
- **Women:** yellow bars

Raining Hogs and Dogs

- **Hogs:** 45.0%
- **Frogs:** 15.0%
- **Logs:** 10.0%
- **Dogs:** 30.0%
Summary of basic functions

- Simple plot: `plot`
- Interactive vs. batch mode: `ion / ioff`
- Hardcopies: `savefig`
- Multiple plots: `subplot`
- Histograms: `hist`
- Bar charts: `bar`
- Pie charts: `pie`
- Switch plotting on top of existing figure: `hold`
- Contour plots: `contour, contourf`
Summary

Matplotlib provides a simple command style interface for creating publication quality figures.

Interactive plotting and different output formats (.png, .pdf, .eps)

Simple plots, multiplot figures, decorations

Possible to use Latex in text
Mayavi

- General purpose, cross-platform tool for 3-D scientific data visualization
- Visualization of scalar, vector and tensor data in 2 and 3 dimensions
- Easy scriptability using Python
- Convenient functionality for rapid scientific plotting via mlab
Simple example

Surface described by three 2D arrays

```python
>>> from mayavi import mlab
>>> from numpy import pi, sin, cos, mgrid
>>> dphi, dtheta = pi/250.0, pi/250.0
>>> [phi,theta] = mgrid[0:pi+dphi*1.5:dphi,0:2*pi+dtheta*1.5:dtheta]
>>> m0 = 4; m1 = 3; m2 = 2; m3 = 3; m4 = 6; m5 = 2; m6 = 6; m7 = 4;
>>> r = sin(m0*phi)**m1 + cos(m2*phi)**m3 + sin(m4*theta)**m5 +
   cos(m6*theta)**m7
>>> x = r*sin(phi)*cos(theta)
>>> y = r*cos(phi)
>>> z = r*sin(phi)*sin(theta)
>>> mlab.mesh(x,y,z)
```
Simple example 2

Iso-surfaces for a 3D volume

```python
>>> from mayavi import mlab
>>> import numpy as np
>>> x, y, z = np.ogrid[-5:5:64j, -5:5:64j, -5:5:64j]
>>> scalars = x * x * 0.5 + y * y + z * z * 2.0
>>> mlab.contour3d(scalars, contours=4, transparent=True)
```
Additional basic functions

- **imshow**: view a 2D array as an image
- **surf**: view a 2D array as a carpet plot
- **quiver3d**: plot arrows to represent vectors at data points
- **savefig**: write out a hardcopy
Summary

- Mayavi is easy-to-use tool for 3D visualization
- Surfaces, iso-surfaces, vector fields
- Hardcopies in various formats
- Vast set of more advanced features