Visualization in Python with matplotlib
an Interactive workshop

Pete Alonzi
Research Data Services UVa Library
October 28, 2015
Outline

• System Configuration
• Our first plot
• Different interaction modes
• Histograms
• Scatter Plots
• Cosmetics
• Multiplots
• Subplots
Setup
Python Installation

- We will need to use python along with some packages.
- A convenient distribution to use is Anaconda.
- Go to: https://www.continuum.io/downloads
- Select your operating system and python version 2.7.
First Plot
A basic plot

- C:\Users\lpa2a>python
- >>> import matplotlib.pyplot as plt
- >>> x=range(10)
- >>> plt.plot(x)
- >>> plt.show()

Why blue? Why a line? Why a line with slope of 1?
Saving a plot

• There are two ways to save the plot:
  – Use the command line:
    • >>> plt.savefig('test.pdf')
    • Must be done before the show command
  – Use the gui:
    • Click the save icon
    • Type in the file name
    • select the file type
Window control

• Notice when we use `plt.show()` we lose control.

• To regain control we must close the plot window.

• Now we’ll go through other ways of interacting with matplotlib to avoid this problem.
Interaction Modes
IPython + %pylab

- Instead of loading up python at the command line with `python` use `ipython` instead.
- Ipython has a special plotting mode which you load by issuing the command `%pylab`
- `C:\Users\lpa2a>ipython`
- `In [1]: %pylab`
- **Now we can try our basic plot again.**
  - We don’t need to use the “`plt.`”
  - We don’t loose control when we plot
  - Plot appears on plot command, no more `show()`
Ipython Notebook

• Launch from start menu
• Click “New Notebook”
  – In [ ]: %pylab
  – Or
  – In [ ]: %pylab inline
Spyder

• Spyder is a python IDE (integrated development environment)
• Combines a text editor a debugger and a command line.
• Let’s take a look around.
Type of plots

http://matplotlib.org/api/pyplot_api.html
Plot Generators

• There are a few functions in matplotlib that will cause a plot to be generated.
• So far we have worked with `plot(...)`. 
• Now we’ll look at a couple more
  – `hist(...)`
  – `scatter(...)`
plot(...)

- So far we have used a simple implementation of plot. Let’s look deeper.

- `plot(range(10))`  #generates x values
- `clf()`
- `x=arange(0,2*pi,0.2)`
- `plot(x,sin(x))`
Histogram

• To plot a histogram we don’t use the function plot. We use the function hist
  \[ \text{hist} \left( \text{randn}(1000) \right) \]
• All of the tricks we just learned to manipulate the plot still work
• Here’s some examples for the binning
  \[ \text{hist} \left( \text{randn}(1000), \text{bins}=25 \right) \]
  \[ \text{hist} \left( \text{randn}(1000), \text{bins}=[-5,-4,-3,-2,-1,0,1,2,3,4,5] \right) \]
Scatter Plot

• Use the function scatter()
  \[-\text{scatter}(\text{randn}(1000), \text{randn}(1000))\]
Cosmetics
Let’s make our plot presentable

• C:\users\lpa2a> ipython --pylab

• In[1]: plot(cos(arange(0,2*pi,0.2)))
  • Grey background
  • Axis labels too small
  • Plot touches axis
  • Plot not centered on axis
  • Horizontal axis values aren’t what we want
  • No axis labels
  • Line thickness
  • Line style
Labels, Ranges, and Values

• Set axis range
  - `plt.axis([-5, 37, -1.5, 1.5])`

• Change horizontal axis values
  - `x=arange(0, 2*pi, 0.2)`
  - `y=cos(x)`
  - `plot(x, y)`
Labels and LaTeX

- Set axis labels
  - `plt.xlabel('x', fontsize=20)`
  - `plt.ylabel('cos(x)')`
  - `plt.title('Cosine')`

- You can use LaTeX as well
  - `plt.title(r'$\cos(x)$')`

- [matplotlib.org/users/pyplot_tutorial.html](http://matplotlib.org/users/pyplot_tutorial.html)
Linestyles

• You have a lot of freedom in choosing a line style.
• They can be expressed explicitly
  \[-\texttt{plot}(x, \texttt{linestyle}='--')\]
• The same goes for line color
  \[-\texttt{plot}(x, \texttt{color}='g')\]
• But you can also use shorthand
  \[-\texttt{plot}(x, 'g--')\]
Linestyles II

• matplotlib automatically interpolates between the points and puts in a line. To emphasize the points you can add markers.
  – `plot(range(10),’o’)` # markers, no line
  – `plot(range(10),’o–’)` # markers, line
  – `plot(range(10),marker=’o’)`

• Set line thickness
  – `pl = plot(arange(10))`
  – `setp(pl,linewidth=5)`
Multiplots and subplots
Multiplots

• To add multiple plots repeat the plot call
  \[-x = \text{arange}(0,2*\text{pi},0.2)\]
  \[-\text{plot1} = \text{plot}(x,\sin(x))\]
  \[-\text{plot2} = \text{plot}(x,\cos(x))\]

• Now to add a legend
  \[-\text{plot1} = \text{plot}(x,\sin(x),\text{label}=`\text{sin}`)\]
  \[-\text{plot2} = \text{plot}(x,\cos(x),\text{label}=`\text{cos}`)\]
  \[-\text{legend}(\text{loc}=`\text{best}`)\]
Subplots

• Multiple plots in the same figure
Subplots

• For better control we will explicitly catch our objects
  
  ```python
  fig = figure()
  sub1 = fig.add_subplot(2,2,1)
  sub2 = fig.add_subplot(2,2,3)
  plt.plot(arange(10))
  ```

• There is a function to do it all at once
  
  ```python
  fig,subs = plt.subplots(3,3)
  ```

  Nb: need `plt.show()` on this one