Matplotlib

Nick Thompson

April 20, 2015

Table of contents

Installation

Non-interactive Plots and Gotchas

Ubuntu

```
sudo apt-get install tk-dev
sudo apt-get install python3-tk # For tkinter backe
sudo apt-get install libblas-dev
sudo apt-get install liblapack-dev # For SciPy
pip3 install -r requirements.txt
```

Mac

```
pip3 install -r requirements.txt
```

Minimal Working Example

```
>>> import matplotlib
>>> from pylab import *
>>> plot([1, 2, 3, 2, 1])
[<matplotlib.lines.Line2D object at 0x7f4bf8ad14a8>
>>> show()
```

Seriously Annoying; Probably indispensable

The call to `show()` is very often a no-op.
This occurs whenever `Matplotlib` doesn't have access to system graphics libraries (happens in virtualenvs all the time . . .)
`show()` is a blocking call; `Matplotlib` scripts can be run in batch-mode by setting the *backend* to agg

Matplotlib Backend

```
>>> import matplotlib
>>> matplotlib.get_backend()
'TkAgg'
```

The backend is useful for embedding `Matplotlib` in other applications; unsurprisingly `TkAgg` is for embedding in `Tkinter` GUIs.

Setting Matplotlib Backend

Direct editing of `matplotlibrc`:

```
>>> import matplotlib
>>> matplotlib.matplotlib_fname()
'/home/nthompson/matplotlib_talk/lib/python3.4/site

$$ head -40 /home/nthompson/matplotlib_talk/lib/pytl
#### CONFIGURATION BEGINS HERE

# The default backend; one of GTK GTKAgg GTKCairo G
# CocoaAgg MacOSX Qt4Agg Qt5Agg TkAgg WX WXAgg Agg
# Template.
# You can also deploy your own backend outside of m
# referring to the module name (which must be in th
# 'module://my_backend'.
backend          : tkagg
```

Setting Matplotlib Backend

Choose backend at runtime:

```
>>> import matplotlib
>>> matplotlib.use('agg')
>>> from pylab import *
>>> plot([1, 2, 3, 2, 1])
>>> show()
```

What is the Matplotlib backend?

The backend chooses the rendering engine (vector or raster).
The most common is the anti-grain geometry library.
More backends are described in the Matplotlib FAQs.

Super simple example

```
./super_simple.py
```

Keyboard commands on Default plots

- Click "Pan and zoom" (or p); hold x and y with right or left mouse-buttons
- Click "Zoom to rectangle" (or o); then push left and right arrow keys for back/forwards.
- Click "Configure subplots" to control spacing.
- Ctrl-f for toggling fullscreen
- Mouse over axes + g: Toggle grid

For awesome examples, check out the Matplotlib gallery.

Interactive Matplotlib

Matplotlib becomes interactive via a call to `ion()`.
This makes calls to `show()` non-blocking.

```
$$ python
>>> import matplotlib.pyplot as plt
>>> plt.ion()
>>> plt.plot([1,2,3,2,1])
>>> plt.title(``Hello'')
>>> plt.title(``This is a tent'')
>>> plt.xlabel(``X axis'')
>>> plt.ylabel(``Y axis'')
>>> plt.ioff()
>>> plt.title(``Goodbye'')
```

On Mac, replace `python` by `ipython` or set your backend to `TkAgg` to avoid a known bug.