02-jupyter

January 20, 2017

1 Jupyter Notebooks

- notebook built out of cells, which are either python input/output, or 'markdown' text
- easy to navigate around cells
- for development, run your own server
- in a command interpreter(like bash) type:
- cd dir-with-notebooks
- jupyter notebook
- a browser window will open automatically
- look at help/keyboard shortcuts
- Jupyter notebooks were previously known as "IPython notebooks"

1.0.1 Notebooks can be converted to pdf, html

- jupyter nbconvert –to pdf homework-1-sols.ipynb
- jupyter nbconvert –to html homework-1-sols.ipynb

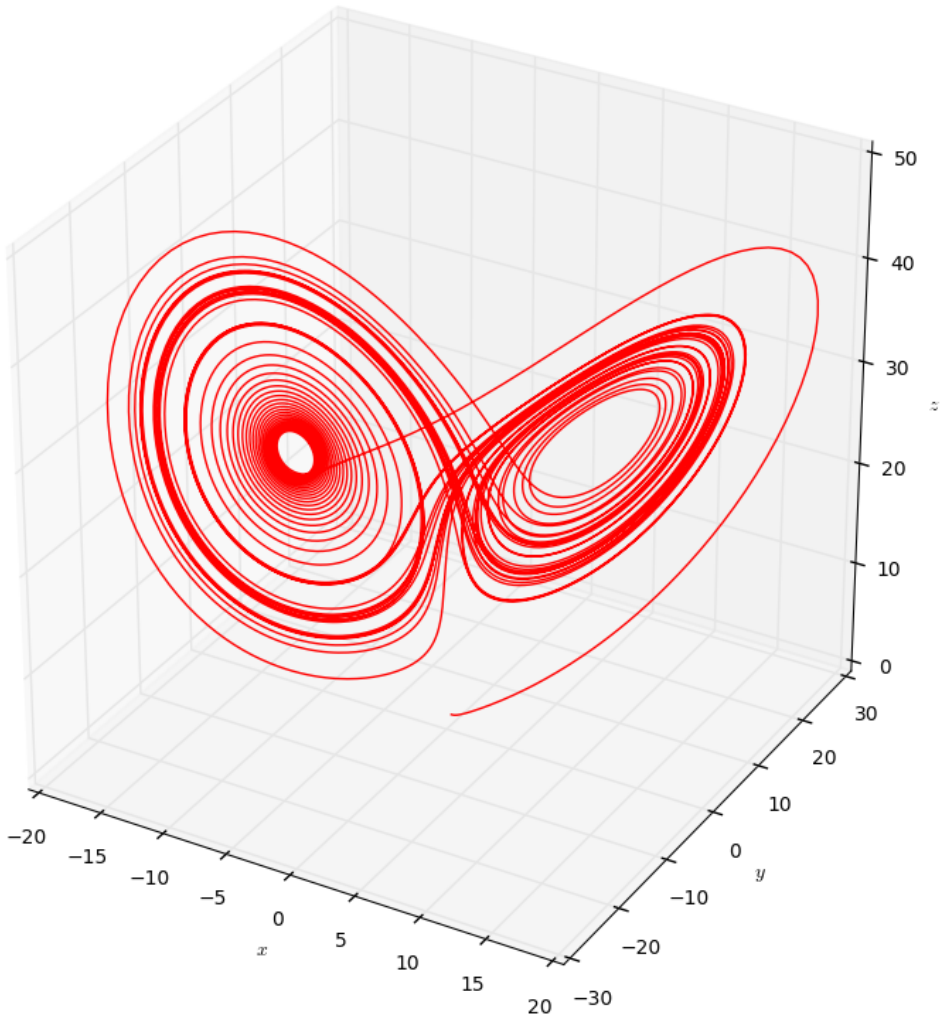
2 Mathematical typesetting with latex

$$\kappa(1+n_{\rm th})\left(a\rho a^\dagger - \frac{1}{2}a^\dagger a\rho - \frac{1}{2}\rho a^\dagger a\right)$$

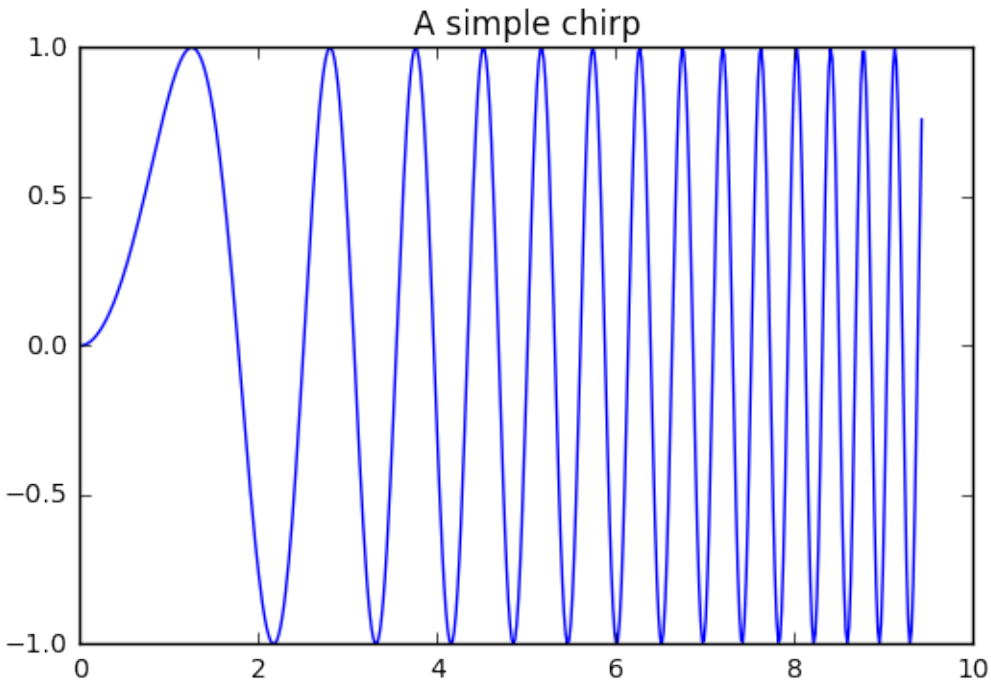
3 Inline Graphics

In [5]:
```
%matplotlib inline

import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import odeint
from mpl_toolkits.mplot3d import Axes3D

def make_lorenz(sigma, r, b):
    def func(statevec, t):
        x, y, z = statevec
        return [ sigma * (y - x), r * x - y - x * z, x * y - b*z]
    return func
```

```
lorenz_eq = make_lorenz(10., 28., 8./3.)
tmax = 50
tdelta = 0.005
tvalues = np.arange(0, tmax, tdelta)
ic = np.array([0.0, 1.0, 0.0])
sol = odeint(lorenz_eq, ic, tvalues)
x, y, z = np.array(list(zip(*sol)))
fig = plt.figure(figsize=(10,10))
ax = fig.add_subplot(111, projection='3d')
ax.plot(x, y, z, lw=1, color='red')
ax.set_xlabel('$x$')
ax.set_ylabel('$y$')
ax.set_zlabel('$z$')
plt.show()
```



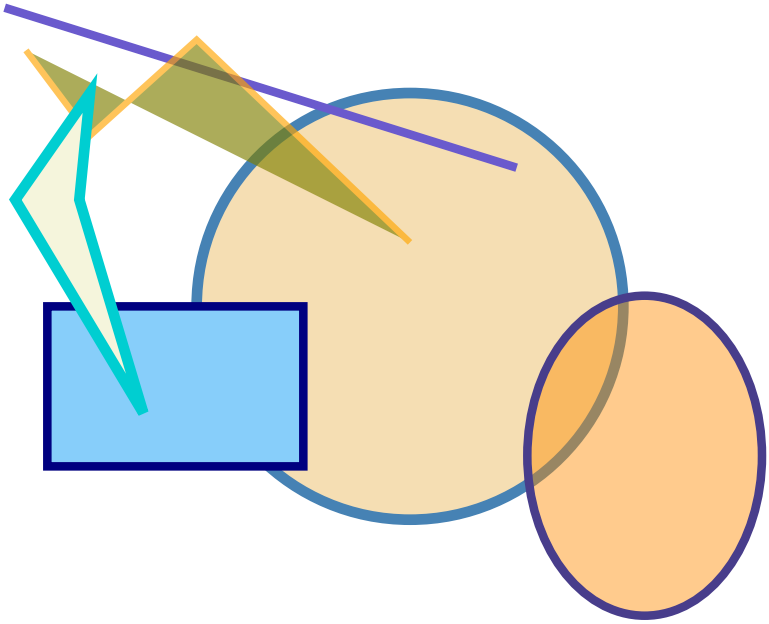
```
In [6]: x = np.linspace(0, 3*np.pi, 500)
        plt.plot(x, np.sin(x**2))
        plt.title('A simple chirp');
```



3.0.1 SVG - scalable vector graphics

```
In [7]: %%SVG
        <svg width="400" height="300">
            <circle  cx="200" cy="150" r="100"
                style="fill:Wheat; stroke:SteelBlue; stroke-width:5;"/>
            <line x1="10" y1="10" x2="250" y2="85"
                style="stroke:SlateBlue; stroke-width:4"/>
            <polyline points="20,30 50,70 100,25 200,120"
                style="stroke:orange; stroke-width:3;
                       fill:olive; opacity:0.65;"/>
            <rect x="30" y="150" width="120" height="75"
                style="stroke:Navy; stroke-width:4; fill:LightSkyBlue;"/>
            <ellipse cx="310" cy="220" rx="55" ry="75"
                style="stroke:DarkSlateBlue; stroke-width:4;
                       fill:DarkOrange; fill-opacity:0.45;"/>
            <polygon points="50,50 15,100 75,200 45,100"
```

```
        style="stroke:DarkTurquoise; stroke-width:5; fill:Beige;"/>
</svg>
```

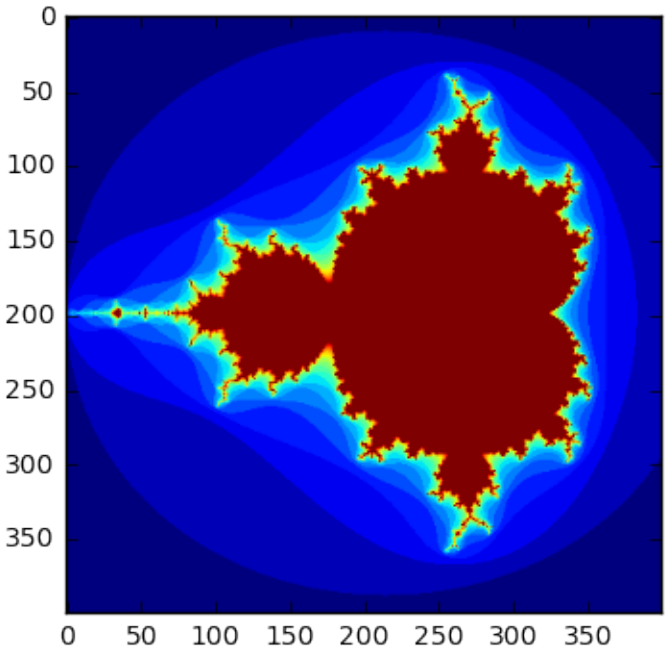


4 Display Images

In [8]:
```
from IPython.display import Image

Image('http://www.imagesource.com/Doc/IS0/Media/TR5/7/7/f/4/IS09A9H4K.jpg')
```

Out[8]:

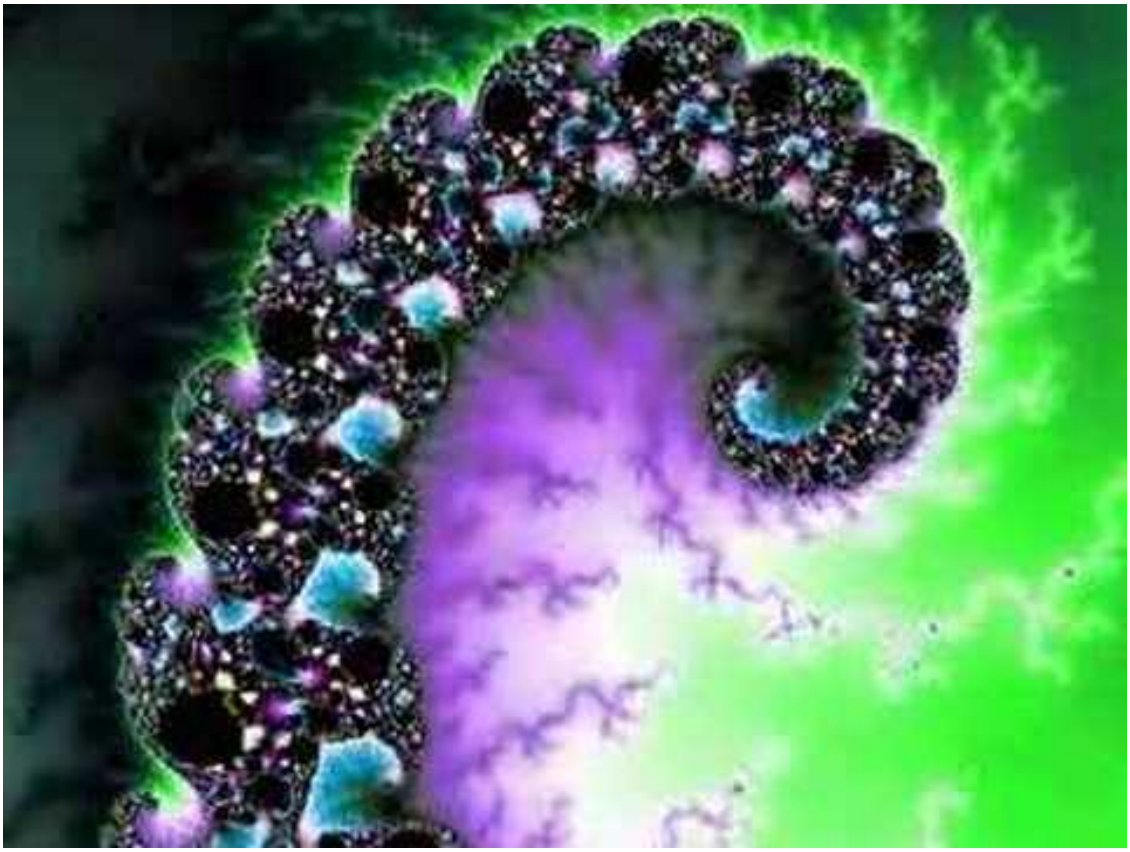
In [3]:
```
from numpy import *
import pylab

def mandelbrot( h,w, maxit=20 ):
        '''Returns an image of the Mandelbrot fractal of size (h,w).
        '''
        y,x = ogrid[ -1.4:1.4:h*1j, -2:0.8:w*1j ]
        c = x+y*1j
        z = c
        divtime = maxit + zeros(z.shape, dtype=int)

        for i in range(maxit):
                z  = z**2 + c
                diverge = z*conj(z) > 2**2            # who is diverging
                div_now = diverge & (divtime==maxit)  # who is diverging no
                divtime[div_now] = i                  # note when
                z[diverge] = 2                        # avoid diverging too

        return divtime

pylab.imshow(mandelbrot(400,400))
pylab.show()
```



5 Youtube

```
In [4]: from IPython.display import YouTubeVideo
        YouTubeVideo('G_GBwuYuOOs')
```

Out[4]:



6 Public server for Notebooks

- single atom laser
- gallery of interesting notebooks

7 Comparison with Mathematica

- Jupyter notebooks modeled after Mathematica's
- Mathematica still quite a bit more sophisticated

In [ ]: