ACS1_Lab_01

August 2, 2017

1 ACS Lab 01 - Scientific Visualization

The first lab of the semester will be structured differently than the rest. The focus of this lab is to familiarize yourself with a wide variety of scientific visualizations. All of the labs this semester will be language agnostic, meaning you can use your language of choice, but they often require a higher level language such as python, matlab or R.

All of my solutions will be written in python. I highly recommend installing python via the Anaconda distribution (Installation Instructions). If you install this way you will have all of the packages I used to complete this assignment.

For this lab you will be asked to complete any 10 of the following 11 visualization tasks.

Remember to label your axes when appropriate and when plotting multiple results on a single plot remember to include a legend.

```
In [1]: import matplotlib as mpl
        import matplotlib.pyplot as plt
        import numpy as np
        import pandas as pd
        import scipy
        import sys

        print('Python version: {}'.format(sys.version[:5]))
        print('Matplotlib version: {}'.format(mpl.__version__))
        print('Numpy version: {}'.format(np.__version__))
        print('Scipy version: {}'.format(scipy.__version__))
        print('Pandas version: {}'.format(pd.__version__))
```

```
Python version: 3.6.0
Matplotlib version: 2.0.0
Numpy version: 1.13.1
Scipy version: 0.19.1
Pandas version: 0.20.3
```

1.1 LogLog Convergence

Often when running numerical simulations researchers are interested in how the relative error changes as the resolution of our model is adjusted. A rate of convergence can be calculated numerically but it is also often visualized on a loglog plot. The file `convergence.csv` contains the

errors associated with a Forward Euler method, Trapezoidal method, and the spacial resolution used. Plot dx vs error for both runs on a loglog scale. Include a dashed line with the exact slope of dx^1 and dx^2 to help guide the eye.

Hints: Use `plt.loglog()` to create a loglog plot. The equation for a straight line in log space is $y = bx^m$ where m is the slope and b is the y intercept.

Solution

In [2]:

1.2 Iris by Species

Many experiments require the same measurements to be taken on different labeled subjects. A famous example of labeled data is known as the Iris flower data set. In 1936 Ronald Fisher studied the length and the width of the sepals and petals of three species of iris (Iris setosa, Iris virginica and Iris versicolor). The file `iris.csv` contains this dataset. Plot the petal length vs petal width and color the points based on their species label.

Hints: Look into the pandas function `groupby()`

Solution

In [2]:

1.3 Anscombe Subplot

A common task in scientific visualization is creating figures with multiple subplots. A classic dataset to promote the important of visualization is known as the Anscombe's quartet. All four of the data sets have the same mean, standard deviation, and linear regression fit. If you don't believe me check it out yourself! The file `anscombe.csv` contains the dataset. Make a single figure with 4 subplots of each xy pair.

Hints: Look into the `subplots()` function

Solution

In [2]:

1.4 Gaussian Overlap

In statistics we are often interested in the overlap of two statistical distributions. Plot two normal distributions $\mathcal{N}_1(\mu = -1, \sigma = 1)$ and $\mathcal{N}_2(\mu = 1, \sigma = 1)$. Shade in the region where they overlap. Note their intersection is at $x = 0$.

Hints: You can use `from matplotlib.mlab import normpdf` for an easy way to calculate the normal distribution. Look up `plt.fill_between` for the shading.

Solution

In [2]:

1.5 Double Y-Axis

Sometimes it is useful to see how two different quantities vary with respect to a common variable. Spurious Correlations is a famous website for emphasizing that correlation is not the same as causation. The file `div_mar.csv` contains the number of divorces in Maine per 1000 people and

the per capita consumption of margarine measured in pounds for each year from 2000 to 2009. Plot both of these variables with respect to the year in a single figure with a different y axis for the right and left side of the figure. If you would like, also numerically calculate the correlation.

Hints: Look up the `ax.twinx()` function. Use `np.corrcoef()` for the correlation calculation

Solution

```
In [2]:
```

1.6 Brain Slice

Data can come in many dimensions. One technique for looking into internal structures of 3D data is to plot a 2D slice along a fixed axis. The file `brain.csv` contains the data of an average brain taken from MRI's of 152 healthy individuals from the Montreal Neurological Institute. The data has been saved as a 1D array and must be reshaped into a (91, 109, 91) array before plotting. Make three images of the x = 47, y = 40, and z = 50 planes.

Hints: Use `plt.imshow()` for visualization and `cmap='gray'` to make the image gray scale. Use `np.reshape()` to make the data the correct shape.

Solution

```
In [2]:
```

1.7 ODE Streamline

When presented with a set of ordinary differential equations one can explore the phase portrait to find equilibrium points. Use the following ODE to create a streamline plot and approximately determine the equilibrium points.

$$\frac{dx}{dt} = xy + 3y$$

$$\frac{dy}{dt} = xy - 3x$$

Hints: Look up `ax.streamplot()`.

Solution

```
In [2]:
```

1.8 Volcano Contour

One way to visualize a 3D surface is with the help of contour maps. The file `volcano.csv` contains topographic information for Auckland's Maunga Whau Volcano on a 10m by 10m grid. Create a contour map, including labeled contour lines.

Hints: Use `plt.contour()` for the plotting and `plt.clabel()` for the labels

Solution

```
In [2]:
```

1.9 3D Gamma Function

In complex analysis people study how a funciton changes on the complex plane. A pole of the function $f(z)$ at point a is defined by the function approaching infinity as z approaches a. Make a 3D surface plot of the absolute value of the gamma function on the complex plane. Notice the poles along the real axis.

Hints: Use `from scipy.special import gamma` for the gamma function. Create a real mesh using `np.meshgrid()` then call the gamma function using `(X + Y*1j)` to make it complex. Adjust the resolution of the meshgrid if needed.

Solution

```
In [2]:
```

1.10 Great Circles

When creating a world map you have to project a 3D object into a 2D plane. Different projections produce distortions on the resulting map. The shortest path between two points in Euclidean space is a straight line, but when dealing with curved surfaces, such as the earth, the shortest path becomes more complicated. On a sphere the shortest path between two points is called the Great Circle. Create a world map using the Robinson projection and draw the great circle connecting Tallahassee, Florida (30.445062,-84.299628) to Berlin, Germany (52.518059, 13.405331), then from Berlin to Johannesburg, South Africa (-26.201209, 28.046225) and finally from Johannesburg to Sydney, Australia (-33.856292, 151.215395).

Hints: Look up how to install `Basemap` for python. There is a function called `drawgreatcircle()` which is useful.

Solution

```
In [2]:
```

1.11 Cycloid Animation

The curve traced by a point on the rim of a circular wheel as the wheel rolls along a straight line without slippage is known as a cycloid. For a circle of radius r the parametric equations for the cycloid are

$$x = r(t - \sin(t))$$

$$y = r(1 - \cos(t))$$

Make an animation of the line traced out by a cycloid of radius $r = 1$ that completes two rotations. Try to include the cycloid point, the line traced out by the point and the circle that creates the cycloid.

Hints: For given t, the circle's centre lies at $x = rt$, $y = r$. Use `from matplotlib import animation` for the animation.

Solution

```
In [2]:
```