NENS 230
Assignment #6: Best practices

Writing clean, maintainable code

Due: Tues 4 Nov 2014

Goals

- Practice refactoring a piece of unwieldy code.
- Become comfortable digesting and explaining code that someone else has written.
- Practice debugging software.
- Review concepts from the first half of the course (plotting)

Overview

Each of the following problems is independent. Feel free to work on them in any order. The first requires you to understand a badly written piece of code, and refactor it using the best practices we discussed in lecture. The second problem asks you to find and fix a few bugs in a piece of code. Finally, the third problem will get you to review concepts we have covered in the first part of the course. Happy coding!

Problem 1: Benford's law

For this problem, we will play with data consisting of population estimates of 239 countries. The data is located in `populationData.mat`, and was obtained from the CIA World Factbook (https://www.cia.gov/library/publications/the-world-factbook/rankorder/2119rank.html). The `.mat` file contains three vectors: a list of ranks in `rank`, the corresponding name of the country in `country`, and the corresponding population in `population`. Note that the names are stored as a cell array, another type of data structure.

Someone (who didn't have the good fortune of taking NENS 230) has run some preliminary analysis on this data to explore an interesting phenomenon known as Benford's law. Benford's law states that in numbers obtained from many real-life sources of data, the frequency of the leading digit (1-9) follows a specific non-uniform distribution. Intuitively, I think most people would expect that if we got a bunch of numbers from a data set, we would expect roughly the same number of numbers to start with an 8, or a 2, or a 1, for example. However, this is not the case. The code given to you `benfordslaw_obfuscated.m` explores this by computing the frequency of leading digits in the population of different countries and comparing this to the distribution predicted by Benford's law. If you haven't heard of Benford's law before, feel free to read up on it on Wikipedia (https://en.wikipedia.org/wiki/Benford%27s_law), because it is pretty interesting.

The code given to you (which should run fine, it works but is badly written) makes a plot the distribution of digits using the `bar` command, and compares this empirical distribution to the prediction given by Benford's law by plotting the prediction in red. Benford's law predicts that the distribution of leading digits is given by:

$$p(d) = \log_{10}\left(1 + \frac{1}{d}\right)$$

Where d is a digit, 1-9.

Your task is simple. Go through the code in `benfordslaw_obfuscated.m` and understand what it does (you can run the script as well, it should run, but it is hard to decipher because it is poorly written). Once you get a sense of what is going on, refactor the code by changing variable names, adding comments, etc. (use the checklist we discussed in lecture as a guide on what to do, it is included in the lecture slides). Save your new script as `benfordslaw_refactored.m`.

Problem 2: Bug finding

In this problem, you will work with simulated (fake) data describing the number of people waiting in line at Peet's cafe in the Clark Center and the Med Cafe in LKSC, sampled every hour for an entire year. The code given to you, `coffee_bugs.m`, loads this data from `coffeeData.mat` and analyzes it.

The analysis is split into three sections, each which can be run independently. Each of these sections has a number of bugs, which cause errors when you run the script. It is your job to find the bugs, and fix them. This will require understanding what the code is doing first, and figuring out where the original author went wrong when writing it. Be careful, some of the bugs are subtle and a bit tricky to find. Find and fix all the bugs, and save a copy of your clean version as `coffee_clean.m`

(A) This part computes the average number of visitors at each hour during the day, averaged over the entire year. It then plots the mean and standard error of this quantity (for both Peet's and LKSC) as a function of time (hour in the day).

(B) This piece first computes the average over the entire day, for each day of the year. It then uses a loop to compute the mean number of visitors each month (roughly by using 31 day bins). It plots this yearly trend for both Peet's and LKSC against the month (with labeled axis ticks).

(C) Finally, this part computes the best hour to get coffee during the day at each establishment, averaged over the entire year. It tries to print this information in a human readable format (e.g., 2:00pm instead of hour 14)

Problem 3: Scatter plots of iris data

In this problem, you will be analyzing a somewhat famous data set, the Fisher iris data set (it is a collection of measurements from the iris of flower species, first introduced by the statistician and biologist Sir Ronald Fisher).

The data consists of 150 different iris measurements from three species of flower. The data contains a variable called `species`, which contains the name of the species for each of the 150 measurements, and a variable called `meas` which contains the following measurements: Sepal Length, Sepal Width, Petal Length, and Petal Width (in each of the four columns, respectively).

You can find out more about this data set on Wikipedia (http://en.wikipedia.org/wiki/Iris_flower_data_set) (yes there is an entire Wikipedia article devoted to it!), and this Matlab demo may also be useful (http://www.mathworks.com/products/demos/statistics/classdemo.html#1).

You can load the data (it is built-in to Matlab) with the following command: `load fisheriris`. Your task is to reproduce the following figure (taken from the Wikipedia page). You don't have to reproduce it exactly (things like tick marks, text labels, etc. don't matter), but the 4x4 subplots with the appropriate scatter plots in the appropriate locations should be present. This is purposefully open ended, we aren't giving you any instructions on where to start. Focus on making your script clean, readable, and easy to follow. Save your script that generates the figure as `irisPlot.m` *Hint: you may find that the `gscatter` command is more useful than than `scatter` on this one*

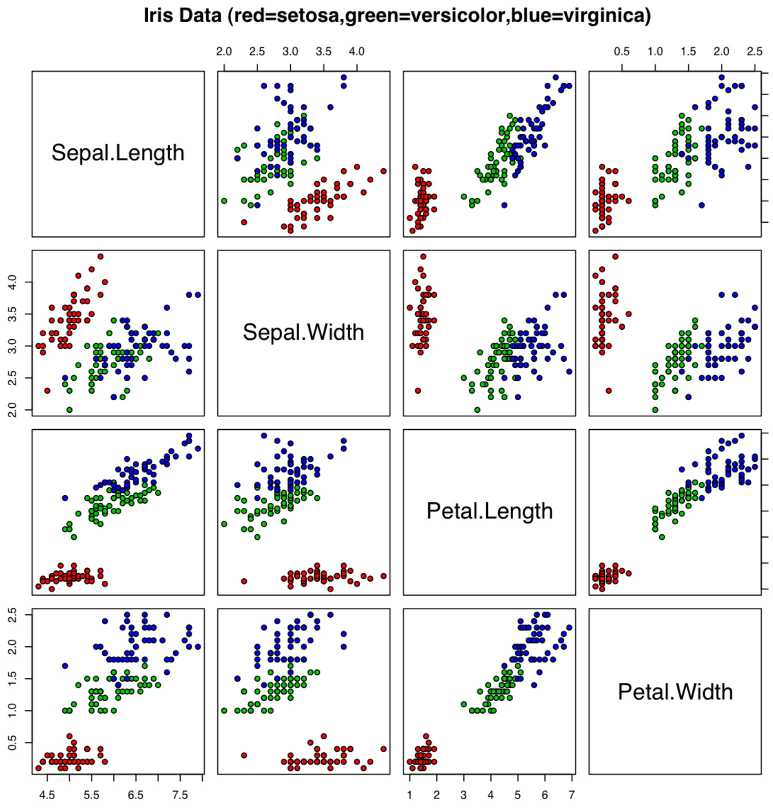

Figure 1: Visualization of the Fisher Iris data set

Submission

When you are finished, publish your work using the `publish` command (once for each script):
`publish('benfordslaw_refactored.m','pdf')`, `publish('coffee_clean.m','pdf')` and `publish('irisPlot.m','pdf')`
Email your scripts and the generated PDF documents to `nens230@gmail.com` to submit your assignment - you should receive an automatic confirmation email noting that we received your submission.