Introduction

UCLA MATH 156

Spring 2016

1 Overview

In this lab we will review the basic functionality of MATLAB to load, visualize, and perform basic operations on a data set. These lab assignments are organized as follows: the overview provides a simple summary of what will be done in the lab, the next section provides a list of provided resources which come with the lab, the guide is intended to give you a step-by-step procedure to arriving at lab goals, and finally a section at the end provides extra questions which are not graded but provide additional questions which build on the conclusions of the lab assignment. Each assignment is graded out of twenty points.

The "Provided Resources" section contains an itemized list of materials that have been given to you. Resources ending in `.mat` contain data for MATLAB, and can be loaded using either the MATLAB function `load()` or by dragging the file into the MATLAB window. Resources ending in `.m` contain MATLAB code, either as a function that is completed and provided or as a placeholder for code which you have to complete. Each of these files begin with a comment section which describe precisely what the code either does, or is intended to do. For example, be aware of the orientation of data for each function: you might be passing `M` when the code assumes it was passed `M'` (the transpose).

Once you have a basic understanding of the resources provided, it is on to the "Guide" section. Here the lab is outlined in steps – simply start from the beginning and work your way to the end. Some steps may simply ask you to look at something for your own edification, while others will be graded. If a step of the lab is graded then there will be a number in parenthesis indicating the point value. Generally graded steps fall into three groups: completed code for algorithms which are used in the lab (turn in the `.m` file), figures/plots generated from the data you encounter (turn in the `.fig` or `.png` file), or written responses to questions about the data. Responses should be only a couple sentences in length and will be graded on the quality of thoughtfulness. If a question is asked in the guide, it is implied that an answer will be turned in. Everything else that must be submitted will be explicitly stated as such.

The "Extra" section can be completely ignored, if you like. The questions here serve to extend the lab in interesting ways that rest beyond the original scope.

Notice: We will sometimes be working with large collections of data, which means that running code might take a very long time to finish. Try to write your code as efficiently as possible. If something is taking more than a minute to run with no end in sight, abort execution (in the command window press `CTRL+c`) and reduce the amount of data. Accurate results obtained with less than the full collection of data will be accepted.

2 Provided Resource

- `data.mat` - A single matrix X with one observation per column. The data was simply generated at random.

3 Guide

1. When using large collections of data, efficient programming is often important. Perform a brief experiment by comparing the time MATLAB takes to run through a loop versus the time the `sum` command requires. Do this by loading the matrix X from `data.mat`, then implementing the algorithm:

```
s <- 0
for i = 1 to 1,000,000
     s <- s + |X(:,i)|
end for
```

which calculates the sum of the column ℓ_2-norms of X (that is, Euclidean distance). The result should be $s \approx 1.4 \cdot 10^6$. Now, perform the same calculation using the function `sum` with no `for` statement.

2. (20) Verify that the number you get for both methods is the same, or report and discuss any differences. Time the result of the two approaches above and turn in the time required for each method. You can time a piece of code by calling `tic` before the code to be timed, followed by `toc` which evaluates to the time, in seconds, since `tic` was called.

4 Extra

None for this lab.