MAT 128 Lab 19: Introduction to R

Work through the first two Chapters of DataCamp's free Introduction to R course:

Chapter 1: Intro to Basics
Chapter 2: Vectors

Access R on the computer by signing into Lehman One. Your user name and password should be the same as for your Lehman email.

You can also access Lehman One by going to http://lehman.edu, clicking on the Login link at the top right, and selecting 'Lehman One Access'.

On the first page of Lehman One, under the heading 'Lehman Apps', you should see "R Studio". Click it.

You can access R Studio this way from any computer.

The following lab is adapted from Katherine St. John's Lab 11, CMP 108/MAT 135/SOC 251, Spring 2017

Plotting Vector Data

We will try using some of the ideas from the DataCamp tutorials to analyze population and weather data.

In R Studio, at the prompt, create a vector of population for the Bronx:

```
bronxPop <-
c(200507,430980,732016,1265258,1394711,1451277,1424815,1471701,1
168972,1203789,1332650,1385108)
```

This is the population from 1900, 1910, 1920, ... 2010 that we used in Lab 1.

What does it look like? Let's try plotting it:

```
plot(bronxPop)
```

It's okay, but we're missing the years. Let's create another vector with the years. This code will create a vector that starts at 1900, ends at 2010, and steps up by 10 in between:

```
years <- c(seq(1900,2010,10))
```

Check to make sure we have the right numbers by typing its name at the prompt:

```
years
```

Now, we can use `years` as the x-axis and `bronxPop` as the y-axis:

```
plot(years,bronxPop)
```

To make it prettier, let's add in some color:

```
plot(years,bronxPop,col="red")
```

Let's also try making a barplot:

```
barplot(bronxPop)
```

The labels are missing. `barplot()` works a bit differently than `plot` for specifying the labels for the x-axis. We can either give the labels as a parameter:

```
barplot(bronxPop, names.arg=c(seq(1900,2010,10)))
```

Or, we can set the names for `bronxPop` (as in the DataCamp tutorial) and then use `barplot`:

```
names(bronxPop) = c(seq(1900,2010,10))
barplot(bronxPop)
```

will yield the same plot.

Some More Useful Vector Commands

In addition to the vector commands in the tutorial, there are several other ones that we will use often to analyze the data.

The first set takes as input a vector and returns a number:

- `sum(myVec)` returns a number: the sum of the values in myVec
- `max(myVec)` returns a number: the max of the values in myVec
- `min(myVec)` returns a number: the max of the values in myVec
- `mean(myVec)` returns a number: the average of the values in myVec

The next set takes a vector as input and also returns a vector:

- `cumax(myVec)` returns a vector: the values are a running maximum of those seen so far in `myVec`.

- `cumin(myVec)` returns a vector: the values are a running maximum of those seen so far in `myVec`.
- `diff(myVec)` returns a vector: the values are the difference between entries in `myVec`.

Here's an example, of the high temperatures in New York City in March 2017:

```
temps =
c(66,63,38,29,35,44,50,59,60,47,28,30,35,32,27,39,47,38,47,51,59
,49,43,55,56,42,50,46,58,51,43)
```

Try the following:

1. First, use `plot()` to visualize the data as we did above.
2. Using the commands above, what is the maximum high temperature for the month?
3. What is the minimum high temperature for the month?
4. What is the average (mean) high temperature for the month?
5. Create a vector of the daily differences between temperatures (i.e. how much the temperature changed day-to-day). Plot the vector.