R Users Group 1

Welcome!

Welcome to R Users Group. We will be exploring the R programming and stastical language with a focus on applications in investments and finance.

You are joining a large community of enthusiastic R users. The Institute of Electronics Engineers 2016 puts R in the top 5 programming languages http://spectrum.ieee.org/computing/software/the-2016-top-programming-languages. The other languages in the top 5 are two different versions of C, Java and Python. The R user base is growing rapidly. R did not make the top 10 in this ranking only two years ago.

R is a leading language for "Big Data" because of its extensive and flexible features for handling data in arrays, superior statistical capabilities and powerful graphics supprt. R is a leading language for theoretical and empirical finance with an extensive library of packages implementing all major methods and topics in finance.

Our Goals

We will simultaneously focus on the entire process of analytical projects throughout these sessions. A typical analysis follows steps something like these:

- Form a hypothesis
  - Devise a game plan to gather data to test the hypothesis
  - Determine the relevant statstical or analytical methods to test the hypothesis
- Capture the data from some source
  - Screen the data for spurious or missing values and format the data in a way that is usable by the methods selected
  - This process is sometimes called "mungeing"
- Perform the statistical analysis or simulations
  - Research R methods and packages available to help with your analysis and coding
- Review the results to determine the outcome of the test
- Select the best graphical method to communicate the outcome
  - Format the outcome in tables and graphs
- Write a report incorporating the tables and graphs

Each session will incorporate methods for both analysis and presentation of results in high quality reports.

This is an R Markdown Document

Our first report preparation tool that we will work with is R Markdown. In a markdown document, you have a combination of text, instructions for formatting and computer code to perform calculations and format graphs. Within this text you will formatting instructions – for example, when a line begins with single "#" that is a signal that the line is a large type heading. You can learn more about different formatting commands in the documents referenced below.

To begin learning about R Markdown documents, check out these websites and documents: http://rmarkdown.rstudio.com/lesson-1.html https://www.rstudio.com/wp-content/uploads/2015/03/rmarkdown-reference.pdf

In particular, we will be working with R Notebooks. R Notebooks are explained here: http://rmarkdown.rstudio.com/r_notebooks.html

Markdown is a flexible tool for creating documents, but it does not have comprehensive tools for everything you might want to do. Later on, we will introduce Latex, a prfessional quality document formatting tool that is used to prepare many scientific books and journals.

Starting out with R

We can perform calculations in R by inserting code chunks. Suppose we want to know the result of the calculation of two plus two. We can insert a code chunk like this.

```
2+2
```

```
## [1] 4
```

R can calculate the normal arithmetic using '+', '-', '/', '*', and'^' which stand for addition, subtraction, division, multiplication and exponentiation respectively.

R also works with characters. You can say "Hello world!" like this.

```
'Hello world!'
```

```
## [1] "Hello world!"
```

Often, you will want to assign intermediate results so you can recall them and use them later. Let's store the result of the calculation 5+5 in a variable called 'x'.

```
x = 5+5
```

Now, we can work with that result by referring to by its name 'x'. Let's say we want to show what the value of x is.

```
x
```

```
## [1] 10
```

R has many built in functions and later on we will learn how to create our own functions. An example of a built in function is one for square root. Let's calculate the square root of x.

```
sqrt(x)
```

```
## [1] 3.162278
```

Now, do some calculations on your own by inserting code chunks.

Calculate 3 plus the square root of 2 (in a code chunk after this line) and assign it to a variable 'y'.

Getting help using R

R is a big language that is constantly growing as people add new packages extending its capability. You will never learn all of it and you will often find yourself trying to remember how to do something in R.

Not a problem! Among the millions of user of R, there are thousands of helpful folks who write blogs and post solutions on web sites like "Stack Exchange". So, if you run in to a problem usually your best bet is just to google a question and more often than not you will find an answer.

So, prepare a code chunk to display the value for 'y' that you calculated above.

Now, let's say you want to display this number rounded to two digits after the decimal place. Try googline something like "rounding values in R" and see what you get. I just tried in there are a number of results. Read those results and then insert a code chunk that rounds the value of y to two digits.

Working with dates and displaying calculations embedded in text

Because we are focused on finance, dates are important to us. Financial values are almost always stated as of some particular time. The closing value of a stock is only meaningful if you say on what day it closed at that value.

Now create a variable for the first day of the Trump presidency.

As mentioned before, much of the power of R is the ability to extend its functionality with packages. We often use a package called 'lubridate' to help with date calculations. Here is how you load the package. The second line of this chunk loads this package. (If you get an error message, you may need to install the package. Click on the Packages tab in the lower right panel, then click install and type 'lubridate' in the box.)

```
inaug_day=as.Date("2017-1-20")
library('lubridate')
```

```
## Warning: package 'lubridate' was built under R version 3.2.5
```

```
##
## Attaching package: 'lubridate'
```

```
## The following object is masked from 'package:base':
##
##     date
```

Now we will illustrate embedding code in text. Donald Trump was inaugurated on 2017-01-20. Donald Trump has been president for 13 days. The hundredth day of the Trump presidency will occur on 2017-04-29.

Note: we used a function called 'today' in this calculation. You now have a created a dynamic document. Every time you open this document, it updates itself based on the current day.

Next we want to calculate how many days are left in President Trump's first term. We use a function called 'years' to calculate the fourth anniversary of the inauguration and how many days remain from today until then.

There are 1448 days remaining in the first term of the Trump presidency.

Summary

We have become familiar with the R Studio environment for creating documents using Markdown and R programming language. We illustrated some simple calculations with numbers and dates. We introduced the use of packages. We introduced the concept of dynamic documents.

Homework

Study the markdown documentation focusing on the 'markdown reference'. Create a R Notebook file and use it to write about anything. Format some headers in various sizes. Learn how to embed a live link to a URL. Insert some code chunks calculating this and that. Pick a celebrity and find out their birthday. Do some calculations of time using that date. How many days until their next birthday and stuff like that.

Additional Reading and Exercises

R notebooks are an example of something called literate programming. The widipedia article for literate programming is https://en.wikipedia.org/wiki/Literate_programming. Donald Knuth, a giant in the field of computer science, advocated this style of programming. He originated Tex as a language for literate programming. Tex morphed in to LaTex which became the standard typesetting technical documents in academia and science. Literate programming is associated with the notions of dynamic documents (mentioned above) and reproducible research where a report, computer code and data are published as an integrated whole. Advocates of reproducible research believe science advances more reliably and quickly with this type of transparency.

Google lubridate and read documentation and blog posts for this package. It will give you a sense of the package and the types of problems it can help you solve when working with date and time data. Experiment with time zones.