Advanced Statistical Analysis with R & Python MSBA 320

Dr. Siamak Zadeh
Associate Professor & Data Scientist
Ageno School of Business
Golden Gate University

Preface

- R, as with any open source platform, is a rapidly evolving and advancing programming language. It's ultimately the student's responsibility to monitor online resources related to this open source platform for the latest releases, libraries, and development updates.

- The links provided in these slides and all other materials / tutorials in the course are dated and may not work at the time of viewing these slides. It's the student's responsibility to search online for the latest and most updated links and documentations for installing and using R or any related libraries or interactive development environments.

- The examples provided in these slides, the textbooks, exercises, and all other materials and resources throughout the course are dated. It's the student's responsibility to identify equivalent versions of these examples and functions in R.

Introduction to R: R Basics
Lecture 1

Outline

- Why R, and R Paradigm
- References, Tutorials and links
- R Overview
- R Interface
- R Workspace
- Help
- R Packages
- Input/Output
- Reusing Results

Why R?

It's free!

It runs on a variety of platforms including Windows, Unix and MacOS.

It provides an unparalleled platform for programming new statistical methods in an easy and straightforward manner.

It contains advanced statistical routines not yet available in other packages.

It has state-of-the-art graphics capabilities.

R has a Steep Learning Curve

(steeper for those that knew SAS or other software before)

First, while there are many introductory tutorials (covering data types, basic commands, the interface), none alone are comprehensive. In part, this is because much of the advanced functionality of **R** comes from hundreds of user contributed packages. Hunting for what you want can be time consuming, and it can be hard to get a clear overview of what procedures are available.

R has a Learning Curve

(steeper for those that knew SAS or other software before)

The **second** reason is more transient. As users of statistical packages, we tend to run one controlled procedure for each type of analysis. Think of PROC GLM in SAS. We can carefully set up the run with all the parameters and options that we need. When we run the procedure, the resulting output may be a hundred pages long. We then sift through this output pulling out what we need and discarding the rest.

R paradigm is different

Rather than setting up a complete analysis at once, the process is highly interactive. You run a command (say fit a model), take the results and process it through another command (say a set of diagnostic plots), take those results and process it through another command (say cross-validation), etc. The cycle may include transforming the data, and looping back through the whole process again. You stop when you feel that you have fully analyzed the data.

How to download?

- Google it using R or CRAN

(Comprehensive R Archive Network)

- http://www.r-project.org
- https://cran.r-project.org/

Tutorials

Each of the following tutorials are in PDF format.

- P. Kuhnert & B. Venables, An Introduction to R: Software for Statistical Modeling & Computing
- J.H. Maindonald, Using R for Data Analysis and Graphics
- B. Muenchen, R for SAS and SPSS Users
- W.J. Owen, The R Guide
- D. Rossiter, Introduction to the R Project for Statistical Computing for Use at the ITC
- W.N. Venebles & D. M. Smith, An Introduction to R

Web links

- Paul Geissler's excellent R tutorial
- Dave Robert's Excellent Labs on Ecological Analysis
- Excellent Tutorials by David Rossitier
- Excellent tutorial an nearly every aspect of R (c/o Rob Kabacoff)
  MOST of these notes follow this web page format
- Introduction to R by Vincent Zoonekynd
- R Cookbook
- Data Manipulation Reference

Web links

- R time series tutorial
- R Concepts and Data Types presentation by Deepayan Sarkar
- Interpreting Output From lm()
- The R Wiki
- An Introduction to R
- Import / Export Manual
- R Reference Cards

Web links

- KickStart
- Hints on plotting data in R
- Regression and ANOVA
- Appendices to Fox Book on Regression
- JGR a Java-based GUI for R [Mac|Windows|Linux]
- A Handbook of Statistical Analyses Using R(Brian S. Everitt and Torsten Hothorn)

R Interface

- We will use RStudio as the console for R.
- RStudio is an integrated development environment (IDE) for R.
- It includes a console, syntax-highlighting editor that supports direct code execution, as well as tools for plotting, history, debugging and workspace management.

Click here to see more RStudio features.

R Overview

R is a comprehensive statistical and graphical programming language and is a dialect of the S language:

1988 - S2: RA Becker, JM Chambers, A Wilks

1992 - S3: JM Chambers, TJ Hastie

1998 - S4: JM Chambers

R: initially written by Ross Ihaka and Robert Gentleman at Dep. of Statistics of U of Auckland, New Zealand during 1990s.

Since 1997: international “R-core” team of 15 people with access to common CVS archive.

R Overview

You can enter commands one at a time at the command prompt (>) or run a set of commands from a source file.

There is a wide variety of data types, including vectors (numerical, character, logical), matrices, dataframes, and lists.

To quit R, use

>q()

R Overview

Most functionality is provided through built-in and user-created functions and all data objects are kept in memory during an interactive session.

Basic functions are available by default. Other functions are contained in packages that can be attached to a current session as needed

R Overview

- A key skill to using **R** effectively is learning how to use the built-in help system. Other sections describe the working environment, inputting programs and outputting results, installing new functionality through packages and etc.

- A fundamental design feature of **R** is that the output from most functions can be used as input to other functions. This is described in reusing results.

R Key objects

- An important strength of R is that it is very rich in the types of objects
- that it supports.
- That strength is rather a disadvantage when you are first learning R.
- But to start, you only need to get your head around a few types of objects.

- Basic Objects

  There are three important basic objects:

  - "atomic vector"
  - "list"
  - NULL

R Key objects

Atomic Vectors

R has six basic ('**atomic**') **vector*** types:

- integer
- real
- complex
- logical
- string (or character)
- raw

- The thing to remember about atomic vectors is that all of the elements in them are only of one type. There can not be an atomic vector that has both numbers and character strings, for instance. **Vectors** with length zero **are** possible (and useful).

* **Vector** is a basic data structure in **R**. It contains element of the same type.

R Key objects

list

- Lists can have different types of items in different components.
- A component of a list is allowed to be another list as well as an atomic vector (and other things).

NULL

- The final object in the list above is NULL.
- This is an object that has zero length.
- Virtually all of the other objects that you deal with will have length greater than zero.

R Key objects

Derived Objects

- There are three important types of what might be called derived — or non-basic — objects:
  - matrix
  - data frame
  - factor

R Key objects

Matrix and data frame:

- Matrices and data frames are both rectangular data objects.
- The difference between them is that everything in a matrix has to be of the same atomic type, but data frames can have different types in different columns.
- Each column of a data frame has to be of a single type.
- A matrix can look exactly like a data frame, but they are implemented entirely differently.
- Sometimes it doesn’t matter whether you have a matrix or a data frame. Other times it is very important to know which you have.

R Overview

- Results of calculations can be stored in objects using the assignment operators:
  - An arrow (<-) formed by a smaller than character and a hyphen without a space!
  - The equal character (=).

R Overview

- These objects can then be used in other calculations. To print the object just enter the name of the object. There are some restrictions when giving an object a name:
  - Object names cannot contain `strange' symbols like !, +, -, #.
  - A dot (.) and an underscore ( ) are allowed, also a name starting with a dot.
  - Object names can contain a number but cannot start with a number.
  - R is case sensitive, X and x are two different objects, as well as temp and temP.

An example

```
> # An example
> x <- c(1:10)
> x[(x>8) | (x<5)]
> # yields 1 2 3 4 9 10
> # How it works
> x <- c(1:10)
> X
>1 2 3 4 5 6 7 8 9 10
> x > 8
> F F F F F F F F T T
> x < 5
> T T T T F F F F F F
> x > 8 | x < 5
> T T T T F F F F T T
> x[c(T,T,T,T,F,F,F,F,T,T)]
> 1 2 3 4 9 10
```

R Warning !

- R is a case sensitive language.
- FOO, Foo, and foo are three different objects.

R Workspace

- Objects that you create during an R session are hold in memory.
- The collection of objects that you currently have is called the workspace.
- This workspace is not saved on disk unless you tell R to do so.
- This means that your objects are lost when you close R and not save the objects, or worse when R or your system crashes on you during a session.

R Help

Once **R** is installed, there is a comprehensive built-in help system. At the program's command prompt you can use any of the following:

help.start() # general help
help(*foo*) # help about function *foo*
?*foo* # same thing
apropos("*foo*") # list all function containing string foo
example(*foo*) # show an example of function *foo*

R Datasets

R comes with a number of sample datasets that you can experiment with. Type

> data( )

to see the available datasets. The results will depend on which packages you have loaded. Type

help(*datasetname***)**

for details on a sample dataset.

R Packages

- One of the strengths of R is that the system can easily be extended. The system allows you to write new functions and package those functions in a so called `R package' (or `R library'). The R package may also contain other R objects, for example data sets or documentation. There is a lively R user community and many R packages have been written and made available on CRAN for other users. Just a few examples, there are packages for portfolio optimization, drawing maps, exporting objects to html, time series analysis, spatial statistics and the list goes on and on.

R Packages

- When you download R, already a number (around 30) of packages are downloaded as well. To use a function in an R package, that package has to be attached to the system. When you start R not all of the downloaded packages are attached, only seven packages are attached to the system by default. You can use the function search to see a list of packages that are currently attached to the system, this list is also called the search path.

```
> search()
[1] ".GlobalEnv" "package:stats" "package:graphics"
[4] "package:grDevices" "package:datasets" "package:utils"
[7] "package:methods" "Autoloads" "package:base"
```

R Packages

To attach another package to the system you can use the menu or the library function. Via the menu:

Select the `Packages' menu and select `Load package...', a list of available packages on your system will be displayed. Select one and click `OK', the package is now attached to your current R session. Via the library function:

```
> library(MASS)
> shoes
$A
[1] 13.2 8.2 10.9 14.3 10.7 6.6 9.5 10.8 8.8 13.3
$B
[1] 14.0 8.8 11.2 14.2 11.8 6.4 9.8 11.3 9.3 13.6
```

R Packages

- The function library can also be used to list all the available libraries on your system with a short description. Run the function without any arguments

```
> library()
Packages in library 'C:/PROGRA~1/R/R-25~1.0/library':
base                    The R Base Package
Boot            Bootstrap R (S-Plus) Functions  (Canty)
class                   Functions for Classification
cluster                 Cluster Analysis Extended Rousseeuw  et al.
codetools               Code Analysis Tools for R
datasets                The R Datasets Package
DBI                     R Database Interface
foreign                 Read Data Stored by Minitab, S, SAS,
      SPSS, Stata, Systat, dBase, ...
graphics                The R Graphics Package
```