Bayesian Computing for Data Science (DATS 6311)

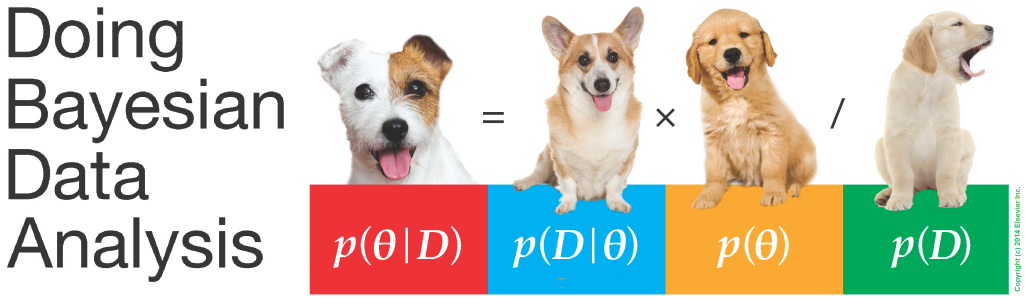
The R Programming Language

Yuxiao Huang

Data Science, Columbian College of Arts & Sciences
George Washington University
yuxiaohuang@gwu.edu

Summer 2020

Reference



Picture courtesy of the book website

- This set of slices is an excerpt of the book by Professor John K. Kruschke, with some trivial changes by the creator of the slides
- Please find the reference to and website of the book below:
  - Kruschke, J. K. (2014). *Doing Bayesian Data Analysis: A Tutorial with R, JAGS, and Stan. 2nd Edition.* Academic Press / Elsevier
  - `https://sites.google.com/site/doingbayesiandataanalysis/`

Overview

1. Get the software
2. Basic commands and operators in R
3. Variable types
4. Loading and saving data
5. Some utility functions
6. Programming in R
7. Graphical plots: opening and saving

Get R and RStudio

- Go to: http://cran.r-project.org/ (Google "R language" if broken link)
- Download R for your operating system
- Go to: http://www.rstudio.com/ (Google "RStudio" if broken link)
- Download RStudio for your operating system

Example: plot $y = x^2$

- R:
  - see SimpleGraph.R (available on the book website)
  - explain:
    - Seq
    - from, to, by
    - type = "l"
- RStudio:
  - Run: executes the line of code where the cursor currently resides, or the lines of code currently selected[1]
  - Source: executes the entire document[1]

[1] https://support.rstudio.com/hc/en-us/articles/200484448-Editing-and-Executing-Code

Learn from examples

- Reading from (good) examples can get you start quickly
- One such example is ExampleOfR.R (available on the book website)
- Read the code and comment to see how things are done in R
- Tweak the code and see if the result is the same as what you expected

Other resources

- *An Introduction to R*, available at this address:
  `http://cran.r-project.org/doc/manuals/r-release/R-intro.pdf`
- *R Reference Card*, available at this address:
  `https://cran.r-project.org/doc/contrib/Short-refcard.pdf`

Getting help in R

- help.start(): returns a list of online documentation, including *An Introduction to R*
- help(name) or ?name: returns a single help page explaining the topic with name *name*
- ??name: returns a list of help pages containing the word *name*
- Google it!

Arithmetic operators

- Arithmetic operators:
  - $+$
  - $-$
  - $*$
  - $/$
  - ^: power
- Precedence
  - from left to right
  - ^ higher than $*$, $/$ higher than $+$, $-$
  - what is $1 + 2 * 3\text{^}2$?

Arithmetic operators

- Arithmetic operators:
  - $+$
  - $-$
  - $*$
  - $/$
  - ^: power
- Precedence
  - from left to right
  - ^ higher than $*$, $/$ higher than $+$, $-$
  - what is $1 + 2 * 3\text{^}2$?
  - use parentheses to explicitly express the order that you intended
  - $1 + 2 * (3\text{^}2)$

Logic operators

- Logic values:
  - TRUE
  - FALSE
- Logic operators:
  - !: negation
  - &&: logic and
  - ||: logic or
- Precedence
  - from left to right
  - ! higher than && higher than ||
  - again, use parentheses to explicitly express the order that you intended

Relational operators and tests of equality

- Relational operators:
  - $<$
  - $>$
  - $!=$
  - $==$
  - all.equal()
- Tests of equality:
  - $==$
    - what is $0.5 - 0.3 == 0.3 - 0.1$?

1

[1] R Documentation

Relational operators and tests of equality

- Relational operators:
  - $<$
  - $>$
  - $!=$
  - $==$
  - all.equal()
- Tests of equality:
  - $==$
    - what is $0.5 - 0.3 == 0.3 - 0.1$?
    - FALSE! (due to the limited precision of representing numbers in the computer's memory)
  - all.equal()
    - equal up to precision of computer
    - all.equal$(0.5 - 0.3, 0.3 - 0.1)$: TRUE
    - use isTRUE(all.equal()) in *if* expressions[1]

[1] R Documentation

Assignment operators

- <-
  - if there is an empty space between $<$ and -, "$<$ -"
  - then this is a logic operator followed by a negative sign
  - $x <$ - 1: TRUE when x is smaller than -1, FALSE otherwise
- $=$
  - prefered
  - differentiate $=$ with $==$

Vector

- A vector is an ordered (not sorted) list of elements of the same type
- Some useful functions
  - the combine function
  - component-by-component vector operations
  - the colon operator and sequence function
  - the replicate function
  - getting at elements of a vector
- See ExampleOfR.R for details (available on the book website)

Factor

- A factor is a vector of indices (transformed from a vector of categorical values), along with a legend that decodes each index into a level name
- Some useful functions
  - the factor function
  - reorder the levels
  - labels
- See ExampleOfR.R for details (available on the book website)

Matrix and array

- A matrix is simply a two-dimensional array of values of the same type
- An array is a generalization of a matrix to multiple dimensions
- See ExampleOfR.R for details (available on the book website)

List and data frame

- The list structure is a generic vector in which components can be of different types, and named
- A data frame is a type of list in which each component is thought of as a named column of a matrix, with different columns possibly of different types
- See ExampleOfR.R for details (available on the book website)

The read.csv function

- The read.csv function loads comma separated values (a.k.a. CSV) files into R's memory
- The data in the CSV files are stored as a data frame
- The categorial columns in the CSV files are transformed into factors in the data frame
- Some useful functions
  - re-order the levels
  - as.vector()
  - factor()
- See ExampleOfR.R for details (available on the book website)

Saving data from R

- The write.csv() saves a data frame to a CSV file
- The CSV file loses all information about levels in factors in the data frame
- The save() function saves a data frame with all the factor information to a Rdata file
- The load() function loads a Rdata file into R's working memory
- The objects() function shows the objects in R's working memory
- See ExampleOfR.R for details (available on the book website)

Some utility functions

- summary(): returns a summary appropriate for the argument
- aggregate(): summarizes data according to factor characteristics
- apply(): collapses arrays across specified dimensions, and applies a function to the data within the collapsed dimensions
- melt(): rearranges data so that there is one datum per row; needs the reshape2 package
- See ExampleOfR.R for details (available on the book website)

Variable names in R

- Variable names should be meaningful
- Variable names are case sensitive
- No space in the name of a variable
- Use *camelBack* notation for naming convention
  - the first letter of each word (except for the first one) is captialized
  - e.g., bayesianMethodsForDataScience

Running a program

- Set the working directory
- The source() function
  - accept all the commands from the argument
  - different from the load() function, which reads a Rdata format file
- The Run button: runs the line on which the cursor is presently positioned, or multiple lines if they are presently selected, and echoes the lines in the command window
- The Source button: runs the entire program without echoing the lines in the command window

Programming a function

- Arguments with explicit labels may go in any order
- Arguments without explicit labels must be in the order used in the definition of the function
- Arguments with default value do not have to be specified in the function call
- Arguments without default value must be specified
- Unlabeled argument provided in the function call is assigned to the first argument in the definition, regardless of whether the argument was defined with a default value
- See ExampleOfR.R for details (available on the book website)

Conditions and loops

- The *if-else* expression: the line containing "else" begins with a closing curly brace; a line that begins with "else" causes an error
- The *for* loop
- See ExampleOfR.R for details (available on the book website)

Measuring processing time

- The proc.time() function: returns the current computer-system time
- To measure the duration of a process, use proc.time() at the beginning and end of the process, and compute the difference
- The 'user time' is the CPU time charged for the execution of user instructions of the calling process[1]
- The 'system time' is the CPU time charged for execution by the system on behalf of the calling process[1]
- See ExampleOfR.R for details (available on the book website)

[1] R Documentation

Debugging

- Hints for avoiding producing errors
  - when creating a new program, always start with a new file name
  - use explicit parentheses to be sure that operators are applied in the intended order
  - use visual white space to make your code easily readable
  - use indenting to meaningfully group sections of the program
    - select a section of program or the entire program
    - press ctrl-i to have the program properly indented
- Hints for debugging the errors
  - read the error messages displayed by R
  - locate the first point in the code where the error occurs by printing the result of the program section by section
  - see details of debugging in RStudio at: `http://www.rstudio.com/ide/docs/debugging/overview`

The openGraph and saveGraph function

- The openGraph and saveGraph function are used for opening and saving graphs
- Need to install other packages
- See ExampleOfR.R and DBDA2E-utilities.R for details (available on the book website)