

Beginning with R

Patricia Hoffman, PhD

Tutorial

<http://patriciahoffmanphd.com/startr.php>

R References

<http://patriciahoffmanphd.com/statisticallanguageer.php>

Agenda Part I

- Introduction
- Built in Help Functions
- Data Structures - Objects
- Object Attributes
- Object Permanence
- Arithmetic Operators

Example: MyFirstRLesson.r

Agenda Part II

- Logical Operators
- Comparisons
- Control Structures
- User Defined Functions

Example: ControlComparisons.r

Example: UserDefnFunction.r



Agenda Part III

Example: Plot Demo

Example: RegressionExamples.r

Introduction to R

- ◆ R is an interpreted language
 - similar to LISP, JavaScript, or MATLAB
- ◆ Arithmetic Operators in R
 - similar to those in C, C++, Java
- ◆ Matrix & Vector Operators
 - similar to MATLAB but different syntax
- ◆ Classes and Objects
 - NOT as formal as Java or C++

R help functions

- ◆ `help.start()` General help
- ◆ `help.search("foo")` *or* `??foo`
 - Search the help system for the string `foo`
- ◆ `example("foo")` Examples of function `foo`
- ◆ `demo()` for list of all demos
 - `demo(graphics)` starts the graphics demo
- ◆ `RSiteSearch("foo")`
 - Search for the string `foo` in online help manuals and archived mailing lists

More help functions

- ◆ `apropos("foo", mode="function")`
 - List available functions with foo in their name
- ◆ `data()`
 - List datasets for currently loaded packages
- ◆ `vignette()`
 - List vignettes for currently installed packages
- ◆ `vignette("foo")`
 - Display specific vignettes for topic foo
- ◆ `options()`

Named Data Structures - Objects

- ◆ vector: numeric (integer, double), complex numbers, logical, character, character strings
- ◆ matrices / arrays - multi-dimensional vector
- ◆ factors (for categorical data)
- ◆ lists (vector of vectors)
- ◆ vector elements need not be of the same type
- ◆ data frames
- ◆ functions

Attributes of Objects

- ◆ `class(x)` returns the mode of a vector `x`
 - i.e. "numeric", "logical", "character", or "list"
- ◆ `class(x)` returns
 - "matrix", "array", "factor", or "data.frame"
- ◆ Attribute queries include
 - `typeof(x)`, `length(x)`, `dim(x)`
 - `is.numeric(x)`, `is.factor(x)`
 - `attributes(x)`, `attr(object, name)`

Casting / Coercion

◆ Setting an Attribute

- Turn integer vector into a matrix
- # start with vector x with attributes given as:
- `length(x) = 10; class(x) = "integer"`
- `attr(x, "dim") <- c(2, 5) # turns vector into matrix`
- # x can be treated as a 2 x 5 matrix
- Now `dim(x) = 2 5` and `class(x) = "matrix"`

◆ Casting x

- `as.character(x), as.integer(x), as.matrix(x)`

Object Permanence

- ◆ Created objects remain in your workspace
- ◆ To view objects in current workspace
 - `objects()`
- ◆ To remove a particular object
 - `rm(x)`
- ◆ To remove all the objects in your workspace
 - `rm(list=ls())`
- ◆ `getwd()` - Find your current workspace
- ◆ `setwd("path")` - Sets the workspace to "path"

Operators

- ◆ assignment `<-` (similar to `=` in other languages)
- ◆ arithmetic operators (generic functions)
 - `x + y` addition
 - `x - y` subtraction
 - `x * y` multiplication
 - `x / y` division
 - `x ^ y` exponentiation (`1^y` and `y^0` are always 1)
 - `x %% y` modulo
 - `x %/% y` integer division
 - `isTRUE(all.equal(x, (x %% y) + y * (x %/% y)))`

Matrix Manipulations

- ◆ $A + B$ - matrix addition
- ◆ $A - B$ - matrix subtraction
- ◆ $A \%*\% B$ - matrix multiplication
- ◆ $A*B$ - multiplication of corresponding elements
- ◆ $A \%/\% B$ and A/B is division of corresponding elts
- ◆ $\%^{\%}$ is not defined
- ◆ $\text{solve}(A,b)$ is the solution x of equation $Ax = b$
- ◆ $t(A)$ - transpose of A ; $\text{diag}()$ –returns a diag matrix

Example: MyFirstRLesson.r

- ◆ Access Help
 - ◆ <http://lib.stat.cmu.edu/R/CRAN/doc/manuals/fullrefman.pdf>
 - ◆ <http://127.0.0.1:27514/library/base/html/Syntax.html>
- ◆ Reading and Writing Files (next slide)
- ◆ Matrix Manipulations
- ◆ Practice
 - Objects
 - Missing Values
 - Simple Functions

matrixletters.csv

first clm	second clm	third clm
11	12	13
21	22	23
31	32	33
41	forty two	43
51	52	53
61	62	63
71	72	-
81	NA	83



More Examples

- ◆ Sampling
- ◆ Working with Graphs

Agenda Part II

- Logical Operators
- Comparisons
- Control Structures
- User Defined Functions

Example: ControlComparisons.r

Agenda Part II

- Logical Operators
- Comparisons
- Control Structures
- User Defined Functions

Example: ControlComparisons.r

Example: UserDefnFunction.r

Comparisons

- ◆ $x < y$
- ◆ $x > y$
- ◆ $x \leq y$
- ◆ $x \geq y$
- ◆ $x \neq y$
- ◆ $x == y$ `identical()`, `all.equal()`
- ◆ `isTRUE(all.equal())` recommended
inside an if clause



Comparison Details

- ◆ NA and NaN are non-comparable even to themselves;
- ◆ comparisons involving them always result in NA



Logical Operators

- ◆ `! x` NOT
- ◆ `x & y` AND (for vectors)
- ◆ `x && y` AND (first element of vector)
- ◆ `x | y` OR
- ◆ `x || y` OR
- ◆ `xor(x, y)` exclusive OR (exactly one true)

Logical Operator Details

- ◆ `&` and `|` element wise comparison on vectors
- ◆ `&&` and `||` works only on the first element
appropriate for control-flow (i.e. if clause)
- ◆ `(NA & TRUE)` evaluates to `NA`
- ◆ `(NA & FALSE)` evaluates to `FALSE`
- ◆ logical vectors are coerced into integer or numeric vectors `FALSE = 0`, `TRUE = 1`

all

- ◆ `all(x, na.rm = TRUE)`
 - returns a logical vector of length one
 - TRUE if all of the values in `x` are TRUE (including if there are no values)
 - FALSE if at least one of the values in `x` is FALSE
 - If `na.rm = TRUE` is specified then all NA's are removed before evaluation.

Terminology - Definitions

- ◆ var - A syntactical name for a variable.
- ◆ expr, cons.expr, alt.expr
 - An expression in a formal sense (simple or compound)
 - compound expression of the form { expr1; expr2 }
- ◆ cond - length-one logical vector (not NA)
 - Accepts conditions of length greater than one with a warning, (only first element used)
 - Other types are coerced to logical if possible
- ◆ seq - expression evaluating to a vector



Control Structures

- ◆ `if(cond) expr`
- ◆ `if(cond) cons.expr else alt.expr`
- ◆ `for(var in seq) expr`
- ◆ `break` - Used to break out of a for loop
 - Control is transferred to the first statement outside of the inner-most loop
- ◆ `next` - Halts processing of current iteration
 - advances the inner-most loop index

Example: ComparisonControl.r

- ◆ Logical Expressions
- ◆ Comparison Examples
- ◆ Control Statements
 - Loops - for
 - Conditionals - if
 - Simple Functions
- ◆ which statement

User Defined Function

```
# Function Definition for myfunction
myfunction <- function(arg1, arg2, ... ){
  statements
  return(object)
}

# Function Call
y <- myfunction(parm1,parm2, ...)
```

Example: UserDefnFunction.r

- ◆ Compare user defined function
 - `userScale()` with `scale()`
- ◆ $(X^T X)^{-1} X^T y$
 - Compare user defined function `bslash`
 - With `lm`

Agenda Part III

Example: RegressionExamples.r

Example: Plot Demo

<http://cran.r-project.org/>

<http://patriciahoffmanphd.com/statisticalanguager.php>