R Console Input

* <- symbol assignment operator
* print is a function
* x is a numeric vector
* auto printing does occur when you just type x
* : creates a sequence
  + x <- 1:20

Data Types – R Objects and Attributes

* Five atomic classes of objects
  + Character
  + Numeric
  + Integer
  + Complex
  + Logical (t/f)
* The most basic object – Vector
  + Vector() <- empty vector
* List can include different classes
* NaN
  + Undefined value 0/0
* Attributes
  + Objects can have them
  + Types
    - Names
    - Dimensions
    - Class
    - Length
    - Attributes()

Data Types – Vectors and Lists

* C()
  + Create vector of objects (numeric, character, logical)
  + Ex
    - X <- (0.5,0.6)
    - X <- 9:29
  + X <- vector(“numeric”, length = 10)
  + Mixed objects vector
    - Creates least common denomination will do use the last
    - COERSION
  + As.\*
    - Converts classes
    - As.numeric(x)
    - As.logical(x)
* List
  + Use multiple classes

Matrices

* Vector with a dimension attribute
  + M <- matrix (nrow = 2, ncol = 3)
    - Dim(m)
      * 2 3
    - attributes(m)
* constructed column wise
* can be created directly from a vector
  + m <- 1:10
  + dim(m) <- c(2 [row],5[column])
* bind columns or rows
  + x <- 1:3
  + y < - 10:12
  + cbind(x,y) = 3 rows and 2 col
  + rbind(x,y) = 2 rows and 3 col

Factors

* represents categorical data
* used for modeling functions like lm() and glm()
* x < - factor(c(“yes”, “yes”, “no”), levels = “c(“yes”,”no”))
  + if you print x you see orders and the levels

Missing Values

* is.na()
  + used to test objects if they are NA
  + is.nan() is used for NaN
  + NaN is NA but not the other way

Data Frames

* Used to store tabular data
* Can store different classes of objects
* Row.names()
* Can be created read.table() or read.csv()
* Converted to matrix with data.matrix()
* X <- data.frame(foo = 1:4, bar = c(T,T,F,F))
  + Nrow(x)
    - 4
  + ncol(x)
    - 2

Name Attributes

* can apply names by names(x) <- x(“d”,”d1”, “s”)
* matrices can have names
  + dimnames(m)

Reading Tabular Data

* read.table, read.csv
  + tabular data = data frame
  + arguments
    - file
      * name of file
    - header
      * if it has a header
    - sep
      * how are they separated
    - colClasses
      * indicated classes
    - nrows
      * number of rows
    - skip
      * skip whatever line(s) you want
    - StringAsFactors
      * Should characters be coded as factors
  + For small to moderately size data sets use data <- read.table(“foo.txt”)
* readLines
  + lines of text file as character vector
* source
  + read code into r
* dget
  + deparse data
* load
  + binary objects

Reading Large Tables

* make a rough calculation of how much you’ll want to read in
  + if you are storing it in your computer
* Set comment.char = “” if there are no commented lines in your file
* If number of columns is small use
  + Sapply(initial, class)
* If columns # is large
  + Use read.table(“”, colClasses = classes)
* Pick number of rows if you just want 100 rows
  + Nrows
* Know your system
  + How much memory on computer
  + Operating system
* Memory requirements = r x c x 8byes/numeric = MB

Textual Data Formats – I don’t completely understand this

* Dumping
  + Can be used on multiple r object
* Dputing
  + Takes an r objects and creates r code to reconstruct it
  + Only used on one
* Not space efficient
* Textual format and contains metadata

Connect to Outside World

* File
* Gzfile
* Bzfile
* url
* useful for parts of files
  + readlines() function
* for webpafes
  + use url()
  + readlines()
    - read lines from connection

Subsetting Basics

* Subsetting
  + [
    - returns same class
      * x <- c(“a”)
      * x[1]
        + “a”
  + [[
    - used to extract list
  + $ sames as [[
  + x [ x >”a”]
  + u <- x> “a”

Subsetting – Lists

* x$bar
* [[]]
  + can be used with computed indices

Subsetting - Matrices

* x[1,2]
* x[1, ]
* x[ , ]
* drop drops dimensions
  + preserve dimension with drop = false in subset
  + can also be used to see vector

Subsetting – Partial Matching

* x [[ “a”, exact = FALSE]]

Subsetting – Removing Missing Values

* create a logical vector
  + bad <- is.na(x)
  + x[!bad]
* multiple columns with na
  + good <- complete.cases(x,y)
    - shows both element with NO NAs

Vectorized Operations

* know matrix and vector multiplication