Topic 8: Useful Functions (and packages)

1. Useful Base-R Functions
2. More Data Management
3. ggplot2 plotting package
4. More advanced stuff (FYI)
5. List of useful packages
6. Some useful resources
7. Extensions to R

1. Useful Base-R Functions, calculations

- `sum(), mean(), median(), sd()`
- `var(x) = (sd(x))^2`
- `sqrt(x) = x^.5`
- `exp()` – raises e to a value, inverse: `log()` – is natural log
- `summary()` – multi-use function, depends on argument
  - `summary(x)` where x is a vector of numeric gives 5-number summary
  - `min(x), max(x), median()`, 1st quartile, 3rd quartile... And `mean()`
  - `quantile(x, .25)` = 1st quantile, `median(x) = quantile(x, .5)`

Useful Base-R Functions, distributions

- Normal distribution functions
  - dnorm(x) – calculates normally **distributed** value for given x
  - pnorm(x) – calculates normally **probability** for all values less than a given x
  - qnorm(x) – calculates the appropriate **quantile** for a given probability
    - inverse of pnorm(): qnorm(pnorm(x)) = x
  - rnorm(x) – generates **random** normally distributed value(s)
- Arguments vary based on type of distribution function
  - For norm, specify a mean and standard deviation
  - Default is mean = 0, sd = 1, i.e. standard normal
  - Different distributions have different parameters
  - For r-distribution_name, must specify number of randomly generated values
- Some other distributions with similar 4 functions (with appropriate first letter)
  - beta – beta distribution
  - binom – binomial
  - exp - exponential
  - t – t distribution
  - unif – uniform distribution

Other Useful Base-R Functions

- `seq()` – creates a sequence or vector of values
- `rep()` – repeats values

- `t.test()`
  - Common function for simple inference about means
  - Many different argument formats
  - Basic syntax is similar to boxplot (but with only 2 levels to factor)
  - See more about boxplot examples in this topic (i.e. using '~' or not)
  - Much more about `t.test()` will be used in coursework

2. More with data management

- `with()` – another way to deal with data frame and columns

```
with(mtcars, mpg[cyl == 8 & disp > 350])
```

- `aggregate()` - summarize data frame

```
with(mtcars, aggregate(mpg, by = list(cyl), FUN = "mean"))
```

- Combining data into a single data frame (no examples with these functions)
  - `rbind()` – binds rows... Just stacks more rows (of same type of data and columns)
  - `cbind()` – binds columns to one data frame, but will need same number of rows for each column
  - `merg()` – combines columns of two data frames, but each data frame must have common identifier to link the two.

More with data management

- `subset()`
  - Selects part of a data frame

```
cyl4mpg <- subset(mtcars, cyl == "4", select = c(mpg))
```

- "stacking" two columns of a dataset
  - reshape2 package, `melt()` function

- `sample()` selects a random sample from data set

- See R Example for more detail and code

More with data management, summary statistics

- create a SumStats object (name is arbitrary)
- Use `ddply()` function
  - Like `aggregate()` but more options

```
mtcars$cyl <- as.factor(mtcars$cyl)
SumStats <- ddply(mtcars, c("cyl"),summarise,
                          n = length(mpg),
                          mean = mean(mpg),
                          sd = sd(mpg),
                          SE = sd/sqrt(n))
SumStats
```

3. Plotting with ggplot2 package

- common plotting tool in R
- R Basics Bootcamp does not cover plotting with ggplot2
  - Plotting options are diverse and unique to fields
  - Coursework will provide opportunity for plotting experience
- Comprehensive plotting package
  - Faceting
  - Multiple dimensions
  - Spatial plotting
- Takes some time to get used to syntax, but eventually intuitive
- Please see the ggplot2 cheat sheet for examples

4. More advanced stuff (FYI)

- Creating functions
  - Example of mean and standard error of a vector d

```
mean.fun <- function (d)
{ m <- mean(d)
n <- length(d)
se <- sd(d)/sqrt(n)
c(m, se)
}
x <- c(2,3,5,9,2,4,6,4,9,5)
mean.fun(x)
```

More advanced stuff (FYI)

- Loops/simulations are great in R
- Not necessarily need ‘for loops’ or ‘while loops’
- Can use ddply or random number generators (e.g. rnorm) to create many values
  - Then perform operations on entire sequence of values
  - Out put values to new data frame etc...
  - See ddply package for potential options
- Conditional statements are typically straightforward in R also
  - If, ifelse or subset selections etc...
  - See resources and helps for syntax

More advanced stuff (FYI)

- Missing Values!

```
a <- c(1, 3, NA, 7, 9)
sum(a)
help(sum)
sum(a, na.rm = TRUE)
```

  `x[!is.na(x)]` to remove NA's

- Saving Data frames
  - `write.csv(data,"data.csv", row.names=FALSE)`
  - Row names may cause confusion when saving
  - `write.table(data, "data.csv", sep="\t", row.names=FALSE, col.names=FALSE)`

    "\t" separates with tabs.

5. Useful packages

- ggplot2 (of course)
- plyr – data management
- reshape2 – for melt function
- car – companion to applied regression
- emmeans – comparing means (ANOVA)
- MASS – modern applied statistics with S (various modeling/plotting)
- tidyverse – a major tool for managing data, a “game changer”
- Many others: stay tuned for suggestions from instructors, colleagues, and referenced work of others in your field

6. Useful Resources, other than google

- RStudio (of course)
  - Cheat sheets
  - Community stuff
- Data camp
  - https://www.statmethods.net/index.html
- Stack over flow
- Github
- Coding and Cookies
  - CSU seminar
  - https://lib.colostate.edu/services/data-management/coding-cookies/

7. Other Stuff R Can Do

- Create presentations
- Manage Data
- Embed Python Code
- Run API’s
- RShiny allow for HTML interaction
- Interactive plots
- Spatial/Mapping functionality