Blue = do not alter. Red = data specific (names, factor levels, etc.). Green = a choice you must make. Choose one, delete the others.

**Descriptive Statistics**

mean(data$variable) median(data$variable) sd(data$variable) min(data$variable) max(data$variable)

summary(data$variable) table(data$factorvariable) tapply(data$quantitativevariable, data$factorvariable, mean/sd/min/max/summary)

**Data Wrangling – Load tidyverse() package**

Convert to factor variable: data$variable <- factor(data$variable)

Recode factor levels: data$variable <- fct\_recode(data$variable, "level0name" = 0, "level1name" = 1)

Order factor levels: data$variable <- ordered(data$variable, levels=c('level1name', 'level2name', 'level3name', 'etc'))

Find variable position: which(colnames(data)=="VariableName")

Rename variable: names(data)[3] <- "NewVariable3Name"

Combine data by columns: NewData <- cbind(data1, data2[column #'s], vector1) #All must be same length and observations sorted identically

Combine data by rows: NewData <- rbind(data1, data2, data3) #All must have the same variables, spelled identically.

Inequalities, "and", "or": == "equals", != "not equal to", >= "greater or equal to", <= "lesser or equal to", > "greater than", < "less than", & "and", | "or"

Subset using Brackets [ ]

Selecting variables: newdata <- data[variables]

e.g., newdata <- data[c("age", "gender", "diabetic")] Includes only age, gender and diabetic variables

e.g., newdata <- data[c(1:3, 7)] Includes only the first 3 and 7th variables.

Filtering cases: newdata <-data[cases, ]

e.g., newdata <- data[data$gender=="female" & data$age >25, ] Females AND older than 25

e.g., newdata <- data[data$age <= 55 | data$bmi < 25, ] Less than or equal to age 55 OR BMI less than 25

Selecting and filtering simultaneously: newdata <- data[cases, variables]

e.g., newdata <- data[data$age >= 25 & data$diabetic=="no", 1:3] At least 25 years old, non-diabetic, and only the first 3 variables

Subset with tidyverse() commands:

Selecting variables:

newdata <- select(data, variable1, variable2, variable3, variable7) OR newdata <- select(data, c(1:3, 7)) First 3 variables and the 7th

Filtering cases:

newdata <- filter(data, variable1=="level name") Note: add filter criteria as arguments separated by commas, not with c()

Wide 🡪 Long (reshaping repeated measures data):

data.l <-  pivot\_longer(data, cols = c(#:#), names\_to = "ExplanatoryVariableName", values\_to = "ResponseVariableName")

Create new variables using ifelse() and logical statements

data$variable <- ifelse(logical\_statement, "value\_if\_true","value\_if\_false") e.g., data$overweight <- ifelse(data$BMI >= 25, "overweight", "normal weight")

**Graphing** - **Load ggplot2() or tidyverse() package.**

**Step 1.** Create an object called p (or any simple name) using the ggplot() command with “data” and an aes (aesthetics) argument that identifies the plot variables:

p <- ggplot(data, aes(x=variable1, y=variable2)

**Step 2. (Single assignment approach**) This tells R what data and what variables to graph, but not *how* to graph them. Layers of detail can be added by placing a + sign at the end of the initial command and before advancing to the next line. R automatically indents the next line and expects more commands to follow. Continue adding one layer per line, ending each line with "+", until the last layer is added.

p <- ggplot(data, aes(x=variable1, y=variable2) +

geom\_bar() +

ggtitle("Graph Title Here")

**Step 2. (Reassignment approach**) In subsequent lines, the object is reassigned to the original object, plus additional layers following the + sign.

p <- ggplot(data, aes(x=variable1, y=variable2)

p <- p + geom\_bar()

p <- p + ggtitle("Graph Title Here")

**Step 3.** Display the completed graph by executing the graph's name:

**Optional Layers for ggplot Graphs**

p + ggtitle("Graph Title Here") Inserting “\n” continues the title on subsequent line when title is too long.

p + labs(x="X-axis Label Here", y= " Y-axis Label Here")

p + facet\_wrap(~CategoricalVariable) Plots the data in separate panels, one for each level of a categorical variable

**Bar graphs for count data (C distributions and C 🡪 C Relationships)**

1 categorical variable: p <- ggplot(data, aes(x=categoricalvariable, fill=categoricalvariable )) + geom\_bar()

2 categorical variables: p <- ggplot(data, aes(x=categoricalvariable1, fill=categoricalvariable2)) + geom\_bar(position=position\_dodge())

Using frequency values: p <- ggplot(data, aes(explanatoryvariable, freq\_values))+ geom\_bar(stat='identity', position=position\_dodge())

**Bar graphs for Quantitative Data (means with error bars of C 🡪 Q relationships and Long Formatted Data)**

1 categorical variable

p <- ggplot(data, aes(x=explanatoryvariable1, y=responsevar)) + stat\_summary(geom ='bar', fun.y = mean, fill='rcolor')

2 categorical variables

p <- ggplot(data, aes(x=expvar1, y=resvar, fill = expvar2))

p <- p + stat\_summary(geom ='bar', fun.y = mean, position=position\_dodge(.95))

Adding error bars: 1) Standard deviation, 2) confidence intervals, 3) Standard error

1. p <- p + stat\_summary(geom = "errorbar", fun.data = mean\_sdl, fun.args=list(mult=1), position=position\_dodge(.95), width=0.2)
2. p <- p + stat\_summary(geom = "errorbar", fun.data = mean\_cl\_normal, fun.args=list(conf.int=.95), position=position\_dodge(.95), width=0.2)
3. p <- p + stat\_summary(geom = "errorbar", fun.data = mean\_se, fun.args=list(mult=1), position=position\_dodge(.95), width=0.2)

**Histograms** p <- ggplot(data, aes(x=quantitativevariable)) + geom\_histogram() Optional to geom\_histogram: color='black', fill = 'r color', binwidth = #

**Boxplots**

All boxes within a boxplot filled with a single R Color

p <- ggplot(data, aes(x=explanatoryvariable, y=responsevariable)) + geom\_boxplot(fill = 'r color')

Each box within a boxplot filled with a unique color

p <- ggplot(data, aes(x=explanatoryvariable1, y=responsevariable, fill=explanatoryvariable2)) + geom\_boxplot()

**Scatterplots & Simple Linear Regression**

p <- ggplot(data, aes(x=explanatorylvariable, y=responsevariable)) + geom\_point() #Creates a scatterplot object called SPlot

p <- p + stat\_smooth(method="lm", se = FALSE) #Add a single regression line

**Multiple linear relationships split by a categorical variable**

p <- p + geom\_point(aes(color= categoricalvariable)) #Creates a scatterplot with color coded levels of a categorical variable (E.g. gender).

p <- p + stat\_smooth(aes(color= categoricalvariable), method="lm", se = FALSE) #Color coded regression lines

**Statistical Tests**

**χ2 test of Independence or Goodness of Fit**

tbl <- table(data$handedness)

chisq.test(tbl)

**t-Tests**

1-Sample t-test: t.test(data$variable, mu=#)

2-Sample Independent t-test: t.test(responsevariable ~ explanatoryvariable, data)

2-Sample Paired t-test

 Long Data Format: t.test(responsevariable ~ explanatoryvariable, data, paired= TRUE)

 Wide Data Format: t.test(data$responsevariable2, data$responsevariable1, paired= TRUE)

Optional Arguments:

 alternative='two.sided/less/greater' Choose one form by deleting the other two. The default is two-sided

 conf.level=.95 The default is 0.95

**Correlation**

Between two variables: cor(data$explanatoryvar, data$responsevar)

Correlation matrix: rcorr.adjust(data[c(#:#)]) Requires the RcmdrMisc package

**Linear Regression & Multiple Regression**

summary(lm(responsevariable ~ explanatoryvariable, data))

summary(lm(responsevariable ~ explanatoryvariable1 + explanatory variable2 + explanatoryvariable3, data))

**ANOVA & Posthoc**

1-Way ANOVA:

Step 1) model <- (aov(responsevariable ~ explanatoryvariable, data))

Step 2) summary(model)

Repeated Measures ANOVA (long format data)

library(ez)

ezANOVA(data.l, dv=(responsevariable), wid =(subjects), within =(explanatoryvariable), detailed=TRUE)

Tukey’s test: TukeyHSD(model)

Bonferroni Correction for multiple tests:

unpaired t-tests: pairwise.t.test(data$responsevar, data$explanatoryvar, p.adjust.method = "bonferroni")

paired t-tests: pairwise.t.test(data.l$responsevariable, data.l$explanatoryvariable, paired=TRUE, p.adjust.method='bonferroni')