Chapter 1

R statistical software and review

Contents

The purpose of this chapter is to discuss R in the context of a quick review of the topics we covered last semester in ADA1[1].

[1] http://statacumen.com/teaching/ada1/

1.1 R

R is a programming language for programming, data management, and statistical analysis. So many people have written "An Introduction to R", that I refer you to the course website[2] for links to tutorials. I encourage you to learn R by (1) running the commands in the tutorials, (2) looking at the help for the commands (e.g., `?mean`), and (3) trying things on your own as you become curious. Make mistakes, figure out why some things don't work the way you expect, and keep trying. Persistence wins the day with programming (as does asking and searching for help).

R is more difficult to master (though, more rewarding) than some statistical packages (such as Minitab) for the following reasons: (1) R does not, in general, provide a point-and-click environment for statistical analysis. Rather, R uses syntax-based programs (i.e., code) to define, transform, and read data, and to define the procedures for analyzing data. (2) R does not really have a spreadsheet environment for data management. Rather, data are entered directly within an R program, read from a file, or imported from a spreadsheet. All manipulation, transformation, and selection of data is coded in the R program. Well done, this means that all the steps of the analysis are available to be repeatable and understood.

Take a minute to install the packages we'll need this semester by visiting this page and following the instructions.
`https://statacumen.com/teach/ADA1/help/Erhardt_R_Install.html`

Also, for automatic diagnostic plots, download and place `ada_functions.R`[3] in your working directory.

[2] `http://statacumen.com/teaching/ada2/`

[3] `https://statacumen.com/teach/ADA2/ada_functions.R`

1.2 ADA1 Ch 0: R warm-up

This example illustrates several strategies for data summarization and analysis. The data for this example are from 15 turkeys raised on farms in either Virginia or Wisconsin.

You should use **help** to get more information on the functions demonstrated here. Many of the lines in the program have comments, which helps anyone reading the program to more easily follow the logic. I **strongly recommend** commenting any line of code that isn't absolutely obvious what is being done and why it is being done. I also recommend placing comments before blocks of code in order to describe what the code below is *meant* to do.

```
library(tidyverse)

# load ada functions
source("ada_functions.R")

#### Example: Turkey, R warm-up
# Read the data file from the website and learn some functions

# read file and assign data to turkey variable
dat_turkey <-
  read_csv("http://statacumen.com/teach/ADA2/notes/ADA2_notes_Ch01_turkey.csv")
```

```
## Parsed with column specification:
## cols(
##   age = col_double(),
##   weight = col_double(),
##   orig = col_character()
## )
```

```
# examine the structure of the dataset, is it what you expected?
  # a data.frame containing integers, numbers, and factors
str(dat_turkey)
```

```
## Classes 'spec_tbl_df', 'tbl_df', 'tbl' and 'data.frame': 15 obs. of  3 variables:
##  $ age   : num  28 20 32 25 23 22 29 27 28 26 ...
##  $ weight: num  13.3 8.9 15.1 13.8 13.1 10.4 13.1 12.4 13.2 11.8 ...
##  $ orig  : chr  "va" "va" "va" "wi" ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   age = col_double(),
##   ..   weight = col_double(),
##   ..   orig = col_character()
##   .. )
```

```
# print dataset to screen
```

```
dat_turkey
```

```
## # A tibble: 15 x 3
##      age weight orig
##    <dbl>  <dbl> <chr>
##  1    28   13.3 va
##  2    20    8.9 va
##  3    32   15.1 va
##  4    25   13.8 wi
##  5    23   13.1 wi
##  6    22   10.4 va
##  7    29   13.1 va
##  8    27   12.4 va
##  9    28   13.2 va
## 10    26   11.8 va
## 11    21   11.5 wi
## 12    31   16.6 wi
## 13    27   14.2 wi
## 14    29   15.4 wi
## 15    30   15.9 wi
```

```
# Note: to view the age variable (column), there's a few ways to do that
dat_turkey$age               # name the variable
```

```
##  [1] 28 20 32 25 23 22 29 27 28 26 21 31 27 29 30
```

```
dat_turkey %>% pull(age)     # pull out the variable
```

```
##  [1] 28 20 32 25 23 22 29 27 28 26 21 31 27 29 30
```

```
dat_turkey[, 1]              # give the column number
```

```
## # A tibble: 15 x 1
##      age
##    <dbl>
##  1    28
##  2    20
##  3    32
##  4    25
##  5    23
##  6    22
##  7    29
##  8    27
##  9    28
## 10    26
## 11    21
## 12    31
## 13    27
## 14    29
## 15    30
```

```
dat_turkey[, "age"]          # give the column name
```

```
## # A tibble: 15 x 1
##      age
```

```
##    <dbl>
##  1    28
##  2    20
##  3    32
##  4    25
##  5    23
##  6    22
##  7    29
##  8    27
##  9    28
## 10    26
## 11    21
## 12    31
## 13    27
## 14    29
## 15    30
```

```
dat_turkey %>% select(age)  # select the column name
```

```
## # A tibble: 15 x 1
##      age
##    <dbl>
##  1    28
##  2    20
##  3    32
##  4    25
##  5    23
##  6    22
##  7    29
##  8    27
##  9    28
## 10    26
## 11    21
## 12    31
## 13    27
## 14    29
## 15    30
```

```
# and the structure is a vector
str(dat_turkey$age)
```

```
##  num [1:15] 28 20 32 25 23 22 29 27 28 26 ...
```

```
# let's create an additional variable for later
# gt25mo will be a variable indicating whether the age is greater than 25 months
dat_turkey <-
  dat_turkey %>%
  mutate(
    gt25mo = (age > 25)
  )
# now we also have a Boolean (logical) column
str(dat_turkey)
```

```
## Classes 'spec_tbl_df', 'tbl_df', 'tbl' and 'data.frame': 15 obs. of  4 variables:
##  $ age   : num  28 20 32 25 23 22 29 27 28 26 ...
##  $ weight: num  13.3 8.9 15.1 13.8 13.1 10.4 13.1 12.4 13.2 11.8 ...
##  $ orig  : chr  "va" "va" "va" "wi" ...
##  $ gt25mo: logi  TRUE FALSE TRUE FALSE FALSE FALSE ...
```

```
# there are a couple ways of subsetting the rows
dat_turkey[(dat_turkey$gt25mo == TRUE),]  # specify the rows
```

```
## # A tibble: 10 x 4
##      age weight orig  gt25mo
##    <dbl>  <dbl> <chr> <lgl>
##  1    28   13.3 va    TRUE
##  2    32   15.1 va    TRUE
##  3    29   13.1 va    TRUE
##  4    27   12.4 va    TRUE
##  5    28   13.2 va    TRUE
##  6    26   11.8 va    TRUE
##  7    31   16.6 wi    TRUE
##  8    27   14.2 wi    TRUE
##  9    29   15.4 wi    TRUE
## 10    30   15.9 wi    TRUE
```

```
subset(dat_turkey, gt25mo == FALSE)   # use subset() to select the data.frame records
```

```
## # A tibble: 5 x 4
##     age weight orig  gt25mo
##   <dbl>  <dbl> <chr> <lgl>
## 1    20    8.9 va    FALSE
## 2    25   13.8 wi    FALSE
## 3    23   13.1 wi    FALSE
## 4    22   10.4 va    FALSE
## 5    21   11.5 wi    FALSE
```

```
dat_turkey %>%
  # use filter() to select the rows
  filter(
    gt25mo == FALSE
  )
```

```
## # A tibble: 5 x 4
##     age weight orig  gt25mo
##   <dbl>  <dbl> <chr> <lgl>
## 1    20    8.9 va    FALSE
## 2    25   13.8 wi    FALSE
## 3    23   13.1 wi    FALSE
## 4    22   10.4 va    FALSE
## 5    21   11.5 wi    FALSE
```

Analyses can be then done on the entire dataset, or repeated for all subsets of a variable in the dataset.

```
# summaries of each variable in the entire dataset,
summary(dat_turkey)
```

```
##       age            weight          orig              gt25mo
##  Min.   :20.00   Min.   : 8.90   Length:15          Mode :logical
##  1st Qu.:24.00   1st Qu.:12.10   Class :character   FALSE:5
##  Median :27.00   Median :13.20   Mode  :character   TRUE :10
##  Mean   :26.53   Mean   :13.25
##  3rd Qu.:29.00   3rd Qu.:14.65
##  Max.   :32.00   Max.   :16.60
```

```
# or summarize by a variable in the dataset.
by(dat_turkey, dat_turkey$orig, summary)
```

```
## dat_turkey$orig: va
##       age            weight          orig              gt25mo
##  Min.   :20.00   Min.   : 8.90   Length:8           Mode :logical
##  1st Qu.:25.00   1st Qu.:11.45   Class :character   FALSE:2
##  Median :27.50   Median :12.75   Mode  :character   TRUE :6
##  Mean   :26.50   Mean   :12.28
##  3rd Qu.:28.25   3rd Qu.:13.22
##  Max.   :32.00   Max.   :15.10
## ------------------------------------------------------
## dat_turkey$orig: wi
##       age            weight          orig              gt25mo
##  Min.   :21.00   Min.   :11.50   Length:7           Mode :logical
##  1st Qu.:24.00   1st Qu.:13.45   Class :character   FALSE:3
##  Median :27.00   Median :14.20   Mode  :character   TRUE :4
##  Mean   :26.57   Mean   :14.36
##  3rd Qu.:29.50   3rd Qu.:15.65
##  Max.   :31.00   Max.   :16.60
```

```
# specific caculations can be done by grouping then calculating
dat_turkey %>%
  group_by(orig) %>%
  summarize(
    mean_age = mean(age)
  , max_weight = max(weight)
  , age_wt = mean_age / max_weight
  )
```

```
## # A tibble: 2 x 4
##   orig  mean_age max_weight age_wt
##   <chr>    <dbl>      <dbl>  <dbl>
## 1 va        26.5       15.1   1.75
## 2 wi        26.6       16.6   1.60
```

1.3 ADA1 Chapters 2, 4, 6: Estimation in one-sample problems

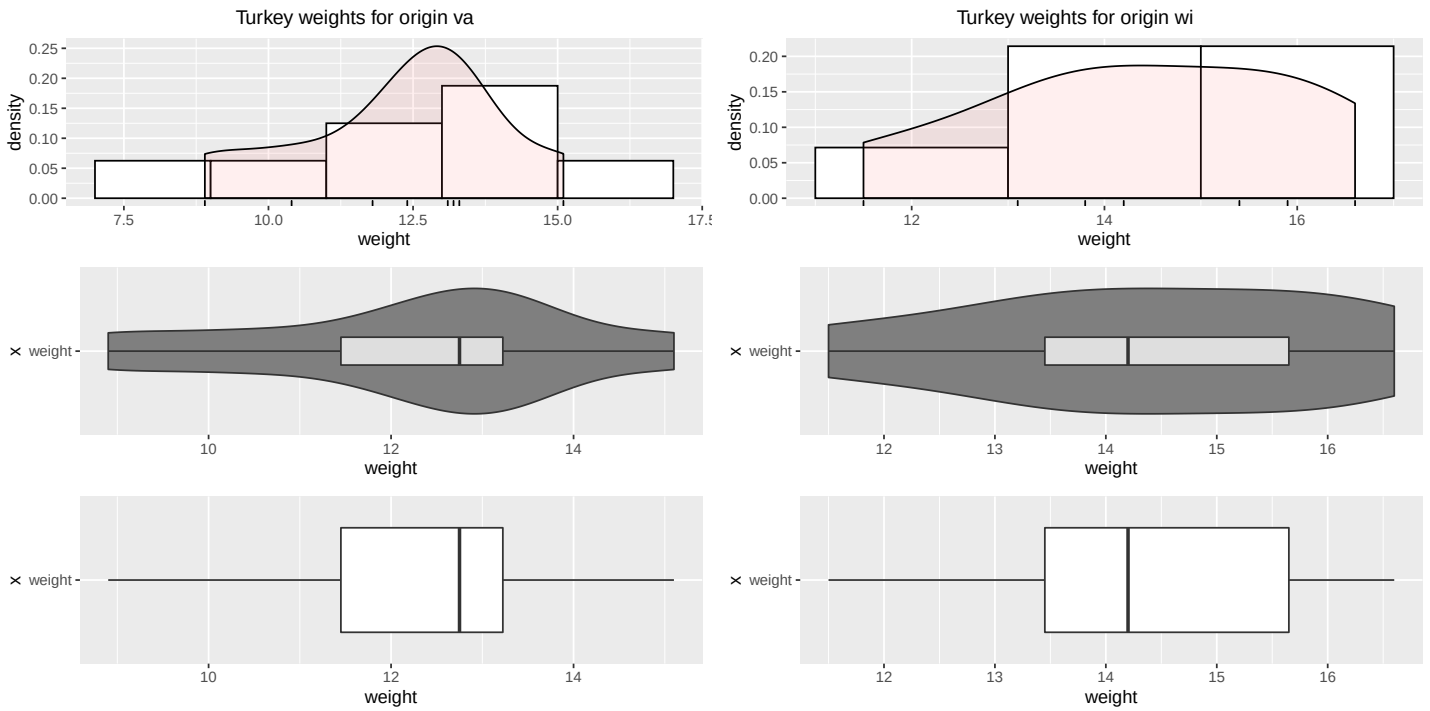
Plot the weights by origin.

```
#### Example: Turkey, Chapters 2, 4, 6
# subset the data for convenience
dat_turkeyva <-
  dat_turkey %>%
  filter(
    orig == "va"
  )
dat_turkeywi <-
  dat_turkey %>%
  filter(
    orig == "wi"
  )

# In practice, you would probably only present one of these plots.
library(ggplot2)
# Histogram overlaid with kernel density curve
p11 <- ggplot(dat_turkeyva, aes(x = weight))
  # Histogram with density instead of count on y-axis
p11 <- p11 + geom_histogram(aes(y=..density..)
                            , binwidth=2
                            , colour="black", fill="white")
  # Overlay with transparent density plot
p11 <- p11 + geom_density(alpha=0.1, fill="#FF6666")
p11 <- p11 + geom_rug()

# violin plot
p12 <- ggplot(dat_turkeyva, aes(x = "weight", y = weight))
p12 <- p12 + geom_violin(fill = "gray50")
p12 <- p12 + geom_boxplot(width = 0.2, alpha = 3/4)
p12 <- p12 + coord_flip()

# boxplot
p13 <- ggplot(dat_turkeyva, aes(x = "weight", y = weight))
p13 <- p13 + geom_boxplot()
p13 <- p13 + coord_flip()

library(gridExtra)
```

```
##
## Attaching package:  'gridExtra'
## The following object is masked from 'package:dplyr':
##
##    combine
```

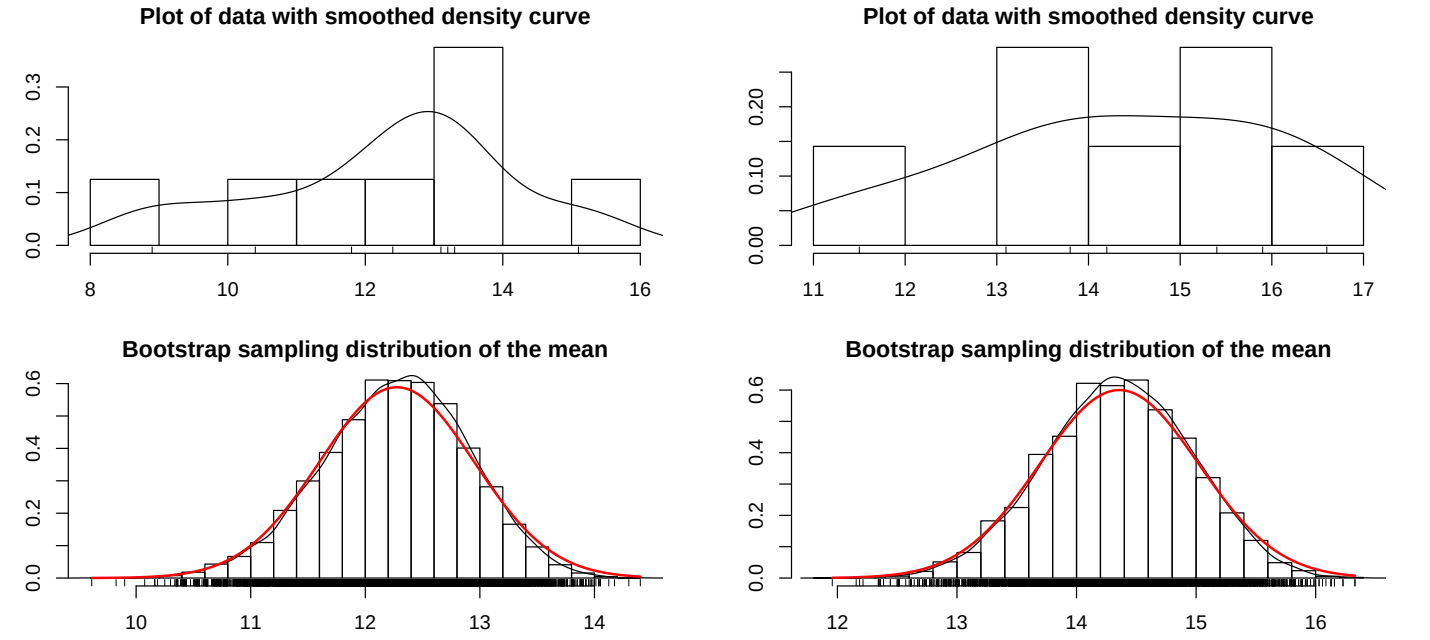
```
#grid.arrange(p11, p12, p13, ncol=1, main="Turkey weights for origin va")
## add grobs = list(), and main= becomes top=
grid.arrange(grobs = list(p11, p12, p13), ncol=1, top="Turkey weights for origin va")


# Histogram overlaid with kernel density curve
p21 <- ggplot(dat_turkeywi, aes(x = weight))
  # Histogram with density instead of count on y-axis
p21 <- p21 + geom_histogram(aes(y=..density..)
                            , binwidth=2
                            , colour="black", fill="white")
  # Overlay with transparent density plot
p21 <- p21 + geom_density(alpha=0.1, fill="#FF6666")
p21 <- p21 + geom_rug()

# violin plot
p22 <- ggplot(dat_turkeywi, aes(x = "weight", y = weight))
p22 <- p22 + geom_violin(fill = "gray50")
p22 <- p22 + geom_boxplot(width = 0.2, alpha = 3/4)
p22 <- p22 + coord_flip()

# boxplot
p23 <- ggplot(dat_turkeywi, aes(x = "weight", y = weight))
p23 <- p23 + geom_boxplot()
```

```
p23 <- p23 + coord_flip()

library(gridExtra)
grid.arrange(grobs = list(p21, p22, p23), ncol=1, top="Turkey weights for origin wi")
```

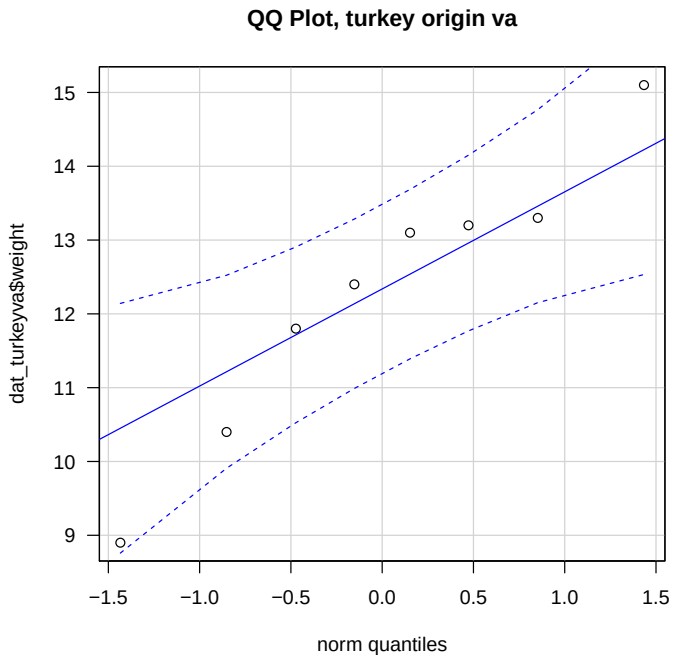


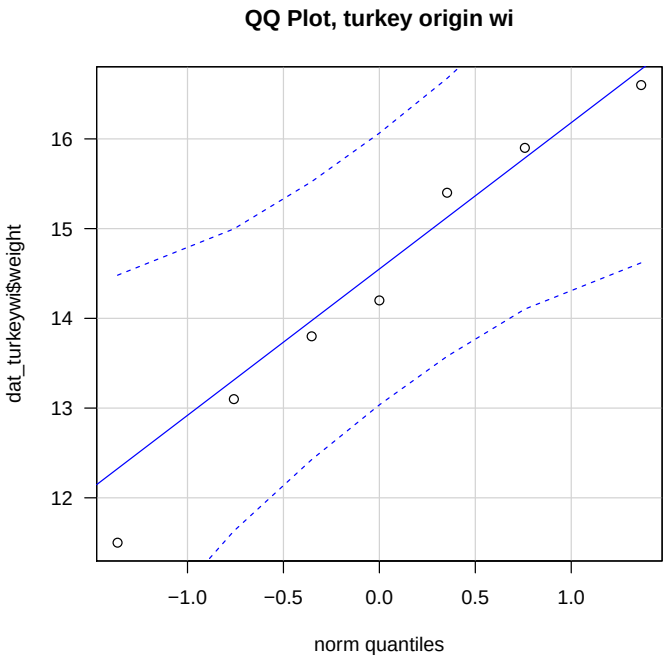
Check normality of each sample graphically with with bootstrap sampling distribution and normal quantile plot and formally with normality tests.

```
# Bootstrap sampling distribution
bs_one_samp_dist(dat_turkeyva$weight)
bs_one_samp_dist(dat_turkeywi$weight)
```



```
# normal quantile-quantile (QQ) plot of each orig sample
library(car)
# qq plot
#   las    =  1 : turns labels on y-axis to read horizontally
#   id.n   =  n : labels n most extreme observations, and outputs to console
#   id.cex =  1 : is the size of those labels
#   lwd    =  1 : line width
qqPlot(dat_turkeyva$weight, las = 1, id = list(n = 0, cex = 1), lwd = 1
     , main="QQ Plot, turkey origin va")
qqPlot(dat_turkeywi$weight, las = 1, id = list(n = 0, cex = 1), lwd = 1
     , main="QQ Plot, turkey origin wi")
```






```
# Normality tests

# VA
shapiro.test(dat_turkeyva$weight)
##
##  Shapiro-Wilk normality test
##
## data:  dat_turkeyva$weight
## W = 0.95414, p-value = 0.7528
library(nortest)
ad.test(dat_turkeyva$weight)
##
##  Anderson-Darling normality test
##
## data:  dat_turkeyva$weight
## A = 0.283, p-value = 0.5339
# lillie.test(dat_turkeyva£weight)
```

```
cvm.test(dat_turkeyva$weight)
```

```
##
##  Cramer-von Mises normality test
##
## data:  dat_turkeyva$weight
## W = 0.050135, p-value = 0.4642
```

```
# WI
shapiro.test(dat_turkeywi$weight)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  dat_turkeywi$weight
## W = 0.97326, p-value = 0.9209
```

```
library(nortest)
ad.test(dat_turkeywi$weight)
```

```
## Error in ad.test(dat_turkeywi$weight):  sample size must be greater than 7
# lillie.test(dat_turkeywi£weight)
cvm.test(dat_turkeywi$weight)
## Error in cvm.test(dat_turkeywi$weight):  sample size must be greater than 7
## Note: The errors above are expected due to small sample sizes.
```

Because we do not have any serious departures from normality (the data are consistent with being normal, as well as the sampling distribution of the mean) the t-test is appropriate. We will also look at a couple nonparametric methods.

```
# Is the average turkey weight 12 lbs?

# t-tests of the mean

# VA
t.summary <- t.test(dat_turkeyva$weight, mu = 12)
t.summary
```

```
##
##  One Sample t-test
##
## data:  dat_turkeyva$weight
## t = 0.40582, df = 7, p-value = 0.697
## alternative hypothesis: true mean is not equal to 12
## 95 percent confidence interval:
##  10.67264 13.87736
## sample estimates:
## mean of x
##    12.275
```

```
# WI
t.summary <- t.test(dat_turkeywi$weight, mu = 12)
t.summary
```

```
##
##  One Sample t-test
##
## data:  dat_turkeywi$weight
## t = 3.5442, df = 6, p-value = 0.01216
## alternative hypothesis: true mean is not equal to 12
## 95 percent confidence interval:
##  12.72978 15.98450
## sample estimates:
## mean of x
##  14.35714
```

```
# Sign test for the median

# VA
library(BSDA)
```

```
## Loading required package:  lattice
##
## Attaching package:  'BSDA'
## The following objects are masked from 'package:carData':
##
##    Vocab, Wool
## The following object is masked from 'package:datasets':
##
##    Orange
```

```
SIGN.test(dat_turkeyva$weight, md=12)
```

```
##
##  One-sample Sign-Test
##
## data:  dat_turkeyva$weight
## s = 5, p-value = 0.7266
## alternative hypothesis: true median is not equal to 12
## 95 percent confidence interval:
##   9.9125 13.8850
## sample estimates:
## median of x
##       12.75
##
## Achieved and Interpolated Confidence Intervals:
##
##                   Conf.Level  L.E.pt U.E.pt
## Lower Achieved CI     0.9297 10.4000 13.300
## Interpolated CI       0.9500  9.9125 13.885
## Upper Achieved CI     0.9922  8.9000 15.100
```

```
# WI
SIGN.test(dat_turkeywi$weight, md=12)
```

```
##
##  One-sample Sign-Test
```

```
##
## data:  dat_turkeywi$weight
## s = 6, p-value = 0.125
## alternative hypothesis: true median is not equal to 12
## 95 percent confidence interval:
##  12.00286 16.38000
## sample estimates:
## median of x
##        14.2
##
## Achieved and Interpolated Confidence Intervals:
##
##                   Conf.Level  L.E.pt U.E.pt
## Lower Achieved CI     0.8750 13.1000  15.90
## Interpolated CI       0.9500 12.0029  16.38
## Upper Achieved CI     0.9844 11.5000  16.60
```

```
# Wilcoxon sign-rank test for the median (or mean, since symmetric assumption)

# VA
# with continuity correction in the normal approximation for the p-value
wilcox.test(dat_turkeyva$weight, mu=12, conf.int=TRUE)
```

```
## Warning in wilcox.test.default(dat_turkeyva$weight, mu = 12, conf.int = TRUE): cannot compu
exact p-value with ties
## Warning in wilcox.test.default(dat_turkeyva$weight, mu = 12, conf.int = TRUE): cannot compu
exact confidence interval with ties
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  dat_turkeyva$weight
## V = 21.5, p-value = 0.674
## alternative hypothesis: true location is not equal to 12
## 95 percent confidence interval:
##  10.40005 14.09997
## sample estimates:
## (pseudo)median
##       12.47331
```

```
# without continuity correction
wilcox.test(dat_turkeyva$weight, mu=12, conf.int=TRUE, correct=FALSE)
```

```
## Warning in wilcox.test.default(dat_turkeyva$weight, mu = 12, conf.int = TRUE, :  cannot
compute exact p-value with ties
## Warning in wilcox.test.default(dat_turkeyva$weight, mu = 12, conf.int = TRUE, :  cannot
compute exact confidence interval with ties
##
##  Wilcoxon signed rank test
##
## data:  dat_turkeyva$weight
## V = 21.5, p-value = 0.6236
```

```
## alternative hypothesis: true location is not equal to 12
## 95 percent confidence interval:
##  10.64999 13.75002
## sample estimates:
## (pseudo)median
##       12.47331
```

```
# WI
# with continuity correction in the normal approximation for the p-value
wilcox.test(dat_turkeywi$weight, mu=12, conf.int=TRUE)
```

```
##
##  Wilcoxon signed rank test
##
## data:  dat_turkeywi$weight
## V = 27, p-value = 0.03125
## alternative hypothesis: true location is not equal to 12
## 95 percent confidence interval:
##  12.65 16.00
## sample estimates:
## (pseudo)median
##         14.375
```

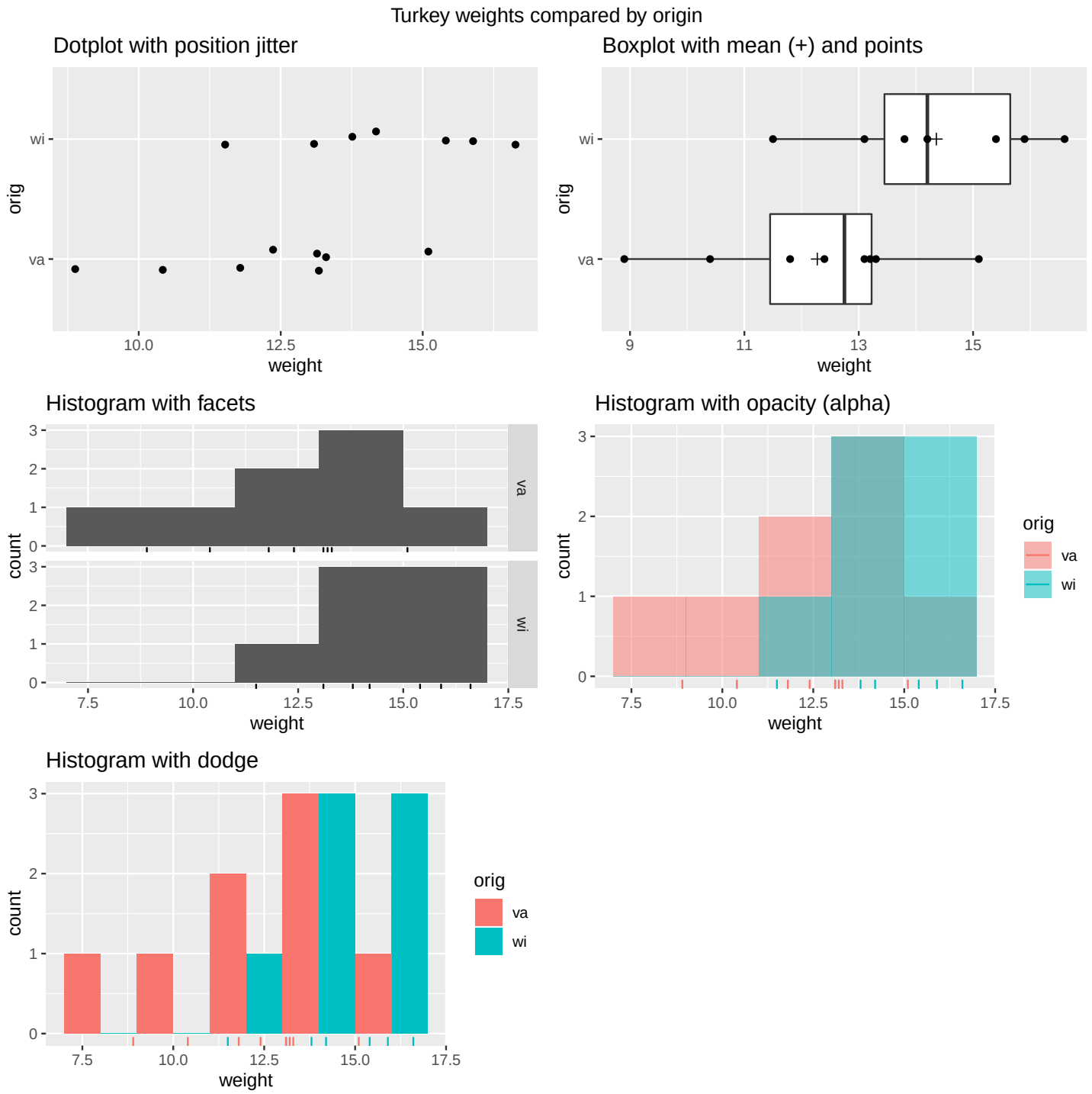
```
# without continuity correction
wilcox.test(dat_turkeywi$weight, mu=12, conf.int=TRUE, correct=FALSE)
```

```
##
##  Wilcoxon signed rank test
##
## data:  dat_turkeywi$weight
## V = 27, p-value = 0.03125
## alternative hypothesis: true location is not equal to 12
## 95 percent confidence interval:
##  12.65 16.00
## sample estimates:
## (pseudo)median
##         14.375
```

1.4 ADA1 Chapters 3, 4, 6: Two-sample inferences

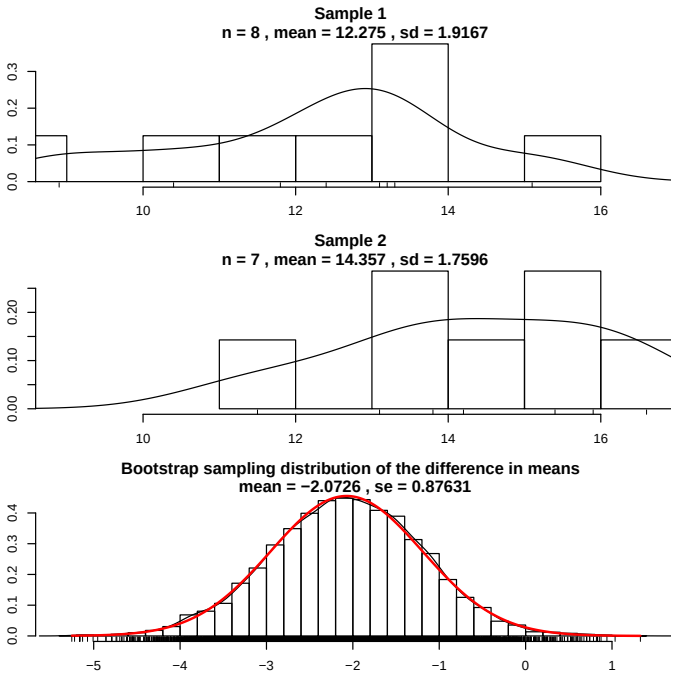
Presume it is of interest to compare the center of the weight distributions between the origins. There are many ways to plot the data for visual comparisons.

```
#### Example: Turkey, Chapters 3, 4, 6
# stripchart (dotplot) using ggplot
library(ggplot2)
p1 <- ggplot(dat_turkey, aes(x = weight, y = orig))
p1 <- p1 + geom_point(position = position_jitter(h=0.1))
p1 <- p1 + labs(title = "Dotplot with position jitter")

# boxplot
p2 <- ggplot(dat_turkey, aes(x = orig, y = weight))
p2 <- p2 + geom_boxplot()
# add a "+" at the mean
p2 <- p2 + stat_summary(fun.y = mean, geom = "point", shape = 3, size = 2)
p2 <- p2 + geom_point()
p2 <- p2 + coord_flip()
p2 <- p2 + labs(title = "Boxplot with mean (+) and points")

# histogram using ggplot
p3 <- ggplot(dat_turkey, aes(x = weight))
p3 <- p3 + geom_histogram(binwidth = 2)
p3 <- p3 + geom_rug()
p3 <- p3 + facet_grid(orig ~ .)
p3 <- p3 + labs(title = "Histogram with facets")

p4 <- ggplot(dat_turkey, aes(x = weight, fill=orig))
p4 <- p4 + geom_histogram(binwidth = 2, alpha = 0.5, position="identity")
p4 <- p4 + geom_rug(aes(colour = orig))
p4 <- p4 + labs(title = "Histogram with opacity (alpha)")

p5 <- ggplot(dat_turkey, aes(x = weight, fill=orig))
p5 <- p5 + geom_histogram(binwidth = 2, alpha = 1, position="dodge")
p5 <- p5 + geom_rug(aes(colour = orig))
p5 <- p5 + labs(title = "Histogram with dodge")

library(gridExtra)
grid.arrange(grobs = list(p1, p2, p3, p4, p5), ncol=2, nrow=3
           , top="Turkey weights compared by origin")
```



Using the two-sample t-test, first check the normality assumptions of the sampling distribution of the mean difference between the populations.

```
bs_two_samp_diff_dist(dat_turkeyva$weight, dat_turkeywi$weight)
```



Two-sample t-test is appropriate since the bootstrap sampling distribution of the difference in means is approximately normal. This is the most powerful test and detects a difference at a 0.05 significance level.

```
# Two-sample t-test
## Equal variances
#   var.equal = FALSE is the default
# two-sample t-test specifying two separate vectors
t.summary.eqvar <- t.test(dat_turkeyva$weight, dat_turkeywi$weight, var.equal = TRUE)
t.summary.eqvar
```

```
##
##  Two Sample t-test
##
## data:  dat_turkeyva$weight and dat_turkeywi$weight
## t = -2.1796, df = 13, p-value = 0.04827
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -4.14595971 -0.01832601
## sample estimates:
## mean of x mean of y
##  12.27500  14.35714
```

```
# two-sample t-test with unequal variances (Welch = Satterthwaite)
#   specified using data.frame and a formula, weight by origin
t.summary.uneqvar <- t.test(weight ~ orig, data = dat_turkey, var.equal = FALSE)
t.summary.uneqvar
```

```
##
##  Welch Two Sample t-test
##
## data:  weight by orig
```

```
## t = -2.1929, df = 12.956, p-value = 0.04717
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -4.13407696 -0.03020875
## sample estimates:
## mean in group va mean in group wi
##         12.27500         14.35714
```

(Wilcoxon-)Mann-Whitney two-sample test is appropriate because the shapes of the two distributions are similar, though their locations are different. This is a less powerful test, but doesn't require normality, and fails to detect a difference at a 0.05 significance level.

```
# with continuity correction in the normal approximation for the p-value
wilcox.test(dat_turkeyva$weight, dat_turkeywi$weight, conf.int=TRUE)
```

```
## Warning in wilcox.test.default(dat_turkeyva$weight, dat_turkeywi$weight, :  cannot compute
exact p-value with ties
## Warning in wilcox.test.default(dat_turkeyva$weight, dat_turkeywi$weight, :  cannot compute
exact confidence intervals with ties
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  dat_turkeyva$weight and dat_turkeywi$weight
## W = 11.5, p-value = 0.06384
## alternative hypothesis: true location shift is not equal to 0
## 95 percent confidence interval:
##  -4.19994493  0.09993686
## sample estimates:
## difference in location
##              -2.152771
```

```
# without continuity correction
wilcox.test(dat_turkeyva$weight, dat_turkeywi$weight, conf.int=TRUE, correct=FALSE)
```

```
## Warning in wilcox.test.default(dat_turkeyva$weight, dat_turkeywi$weight, :  cannot compute
exact p-value with ties
## Warning in wilcox.test.default(dat_turkeyva$weight, dat_turkeywi$weight, :  cannot compute
exact confidence intervals with ties
##
##  Wilcoxon rank sum test
##
## data:  dat_turkeyva$weight and dat_turkeywi$weight
## W = 11.5, p-value = 0.05598
## alternative hypothesis: true location shift is not equal to 0
## 95 percent confidence interval:
##  -4.100049e+00  1.445586e-05
## sample estimates:
## difference in location
##              -2.152771
```

1.5 ADA1 Chapters 5, 4, 6: One-way ANOVA

The Waste Run-up data[4] refer to five suppliers of the Levi-Strauss clothing manufacturing plant in Albuquerque. The firm's quality control department collects weekly data on percent-age waste (run-up) relative to what can be achieved by computer layouts of patterns on cloth. It is possible to have negative values, which indicate that the plant employees beat the computer in controlling waste. Under question are differences among the five supplier plants (PT1, . . . , PT5).

```
#### Example: Waste Run-up, Chapters 5, 4, 6
# convert to a data.frame by reading the text table
dat_waste <- read.table(text = "
 PT1   PT2   PT3   PT4   PT5
 1.2  16.4  12.1  11.5  24.0
10.1  -6.0   9.7  10.2  -3.7
-2.0 -11.6   7.4   3.8   8.2
 1.5  -1.3  -2.1   8.3   9.2
-3.0   4.0  10.1   6.6  -9.3
-0.7  17.0   4.7  10.2   8.0
 3.2   3.8   4.6   8.8  15.8
 2.7   4.3   3.9   2.7  22.3
-3.2  10.4   3.6   5.1   3.1
-1.7   4.2   9.6  11.2  16.8
 2.4   8.5   9.8   5.9  11.3
 0.3   6.3   6.5  13.0  12.3
 3.5   9.0   5.7   6.8  16.9
-0.8   7.1   5.1  14.5    NA
19.4   4.3   3.4   5.2    NA
 2.8  19.7  -0.8   7.3    NA
13.0   3.0  -3.9   7.1    NA
42.7   7.6   0.9   3.4    NA
 1.4  70.2   1.5   0.7    NA
 3.0   8.5    NA    NA    NA
 2.4   6.0    NA    NA    NA
 1.3   2.9    NA    NA    NA
", header=TRUE) %>% as_tibble()
dat_waste %>% print(n = Inf)
```

```
## # A tibble: 22 x 5
##      PT1   PT2   PT3   PT4   PT5
```

[4]From `http://lib.stat.cmu.edu/DASL/Stories/wasterunup.html`, the Data and Story Library (DASL, pronounced "dazzle") is an online library of datafiles and stories that illustrate the use of basic statistics methods. "Waste Run-up" dataset from L. Koopmans, Introduction to Contemporary Statistical Methods, Duxbury Press, 1987, p. 86.

```
##    <dbl> <dbl> <dbl> <dbl> <dbl>
##  1   1.2  16.4  12.1  11.5  24
##  2  10.1  -6     9.7  10.2  -3.7
##  3  -2   -11.6   7.4   3.8   8.2
##  4   1.5  -1.3  -2.1   8.3   9.2
##  5  -3     4    10.1   6.6  -9.3
##  6  -0.7  17     4.7  10.2   8
##  7   3.2   3.8   4.6   8.8  15.8
##  8   2.7   4.3   3.9   2.7  22.3
##  9  -3.2  10.4   3.6   5.1   3.1
## 10  -1.7   4.2   9.6  11.2  16.8
## 11   2.4   8.5   9.8   5.9  11.3
## 12   0.3   6.3   6.5  13    12.3
## 13   3.5   9     5.7   6.8  16.9
## 14  -0.8   7.1   5.1  14.5  NA
## 15  19.4   4.3   3.4   5.2  NA
## 16   2.8  19.7  -0.8   7.3  NA
## 17  13     3    -3.9   7.1  NA
## 18  42.7   7.6   0.9   3.4  NA
## 19   1.4  70.2   1.5   0.7  NA
## 20   3     8.5  NA    NA    NA
## 21   2.4   6    NA    NA    NA
## 22   1.3   2.9  NA    NA    NA
```

```
dat_waste_long <-
  dat_waste %>%
  pivot_longer(
    cols      = starts_with("PT")
  , names_to  = "plant"
  , values_to = "runup"
  , values_drop_na = TRUE    # drop the NA values in long format
  ) %>%
  mutate(
    plant = factor(plant)
  )

str(dat_waste_long)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame': 95 obs. of  2 variables:
##  $ plant: Factor w/ 5 levels "PT1","PT2","PT3",..: 1 2 3 4 5 1 2 3 4 5 ...
##  $ runup: num  1.2 16.4 12.1 11.5 24 10.1 -6 9.7 10.2 -3.7 ...
```

```
head(dat_waste_long)
```

```
## # A tibble: 6 x 2
##   plant runup
##   <fct> <dbl>
## 1 PT1     1.2
## 2 PT2    16.4
## 3 PT3    12.1
## 4 PT4    11.5
```

```
## 5 PT5    24
## 6 PT1    10.1
```

```
tail(dat_waste_long)
```

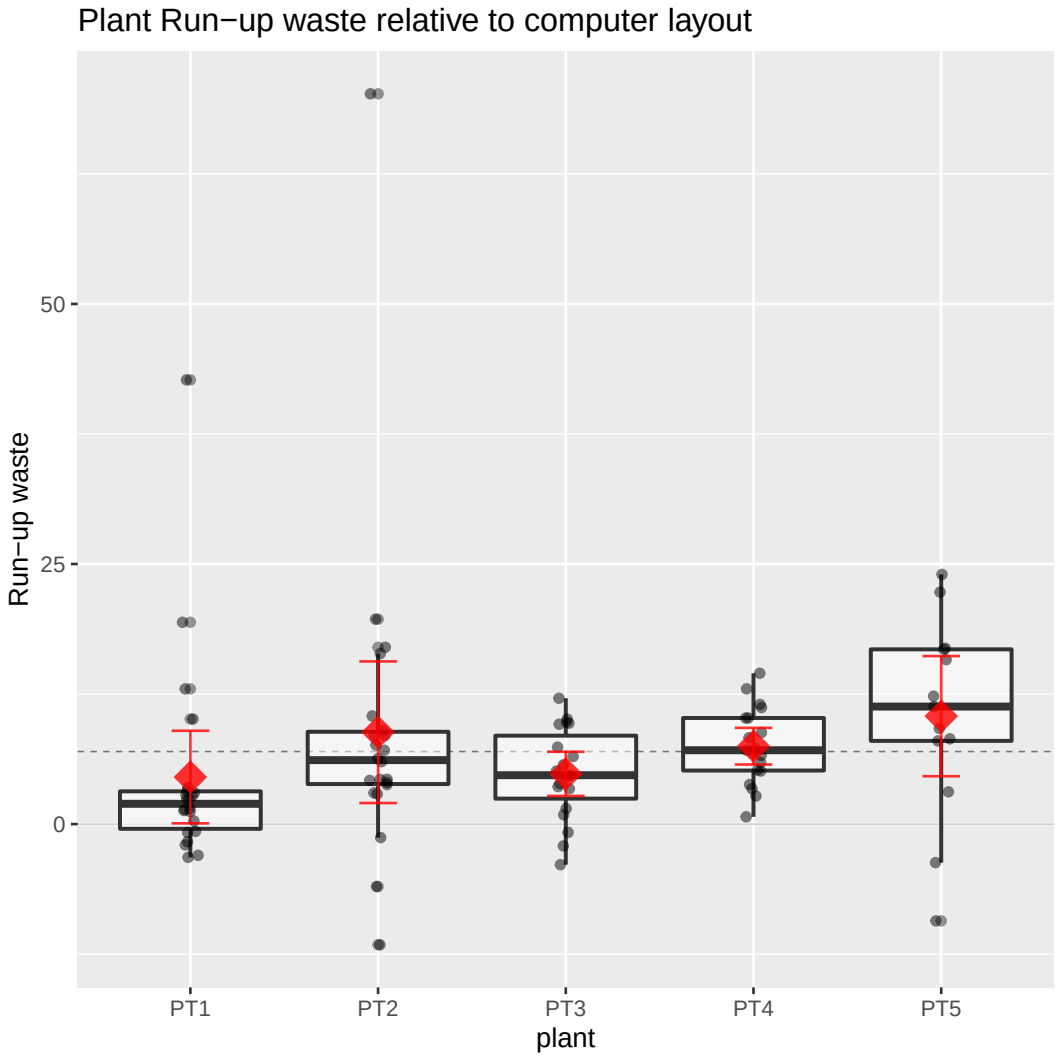
```
## # A tibble: 6 x 2
##   plant runup
##   <fct> <dbl>
## 1 PT1     3
## 2 PT2     8.5
## 3 PT1     2.4
## 4 PT2     6
## 5 PT1     1.3
## 6 PT2     2.9
```

```
# Calculate summaries by plant
# The pattern "split, apply, and combine" is performed by
#   group_by() and summarize(), with an optional ungroup()

sum_waste <-
  dat_waste_long %>%
  group_by(plant) %>%
  summarize(
    # mean, sd, n, and se for the plants
    m    = mean(runup)
  , s    = sd(runup)
  , n    = length(runup)
    # standard error
  , se   = s / sqrt(n)
    # margin-of-error
  , moe  = qt(1 - 0.05 / 2, df = n - 1) * se
    # individual confidence limits
  , ci_l = m - moe
  , ci_u = m + moe
)

sum_waste
```

```
## # A tibble: 5 x 8
##   plant     m     s     n    se   moe   ci_l  ci_u
##   <fct> <dbl> <dbl> <int> <dbl> <dbl>  <dbl> <dbl>
## 1 PT1    4.52 10.0     22 2.14   4.45 0.0748  8.97
## 2 PT2    8.83 15.4     22 3.27   6.81 2.02   15.6
## 3 PT3    4.83  4.40    19 1.01   2.12 2.71    6.95
## 4 PT4    7.49  3.66    19 0.839  1.76 5.73    9.25
## 5 PT5   10.4   9.56    13 2.65   5.77 4.60   16.2
```

```
# Plot the data using ggplot
library(ggplot2)
p <- ggplot(dat_waste_long, aes(x = plant, y = runup))
# plot a reference line for the global mean (assuming no groups)
```

```
p <- p + geom_hline(aes(yintercept = 0),
                    colour = "black", linetype = "solid", size = 0.2, alpha = 0.3)
p <- p + geom_hline(aes(yintercept = mean(runup)),
                    colour = "black", linetype = "dashed", size = 0.3, alpha = 0.5)
# boxplot, size=.75 to stand out behind CI
p <- p + geom_boxplot(size = 0.75, alpha = 0.5)
# points for observed data
p <- p + geom_point(position = position_jitter(w = 0.05, h = 0), alpha = 0.5)
# diamond at mean for each group
p <- p + stat_summary(fun.y = mean, geom = "point", shape = 18, size = 6,
                      colour = "red", alpha = 0.8)
# confidence limits based on normal distribution
p <- p + stat_summary(fun.data = "mean_cl_normal", geom = "errorbar",
                      width = .2, colour = "red", alpha = 0.8)
p <- p + labs(title = "Plant Run-up waste relative to computer layout")
p <- p + ylab("Run-up waste")
print(p)
```



The outliers here suggest the ANOVA is not an appropriate model. The normality tests below suggest the distributions for the first two plants are not

normal.

```
# the base R version provides helpful headers for each analysis
by(dat_waste_long$runup, dat_waste_long$plant, ad.test)
```

```
## dat_waste_long$plant: PT1
##
##  Anderson-Darling normality test
##
## data:  dd[x, ]
## A = 2.8685, p-value = 1.761e-07
##
## ------------------------------------------------------
## dat_waste_long$plant: PT2
##
##  Anderson-Darling normality test
##
## data:  dd[x, ]
## A = 2.5207, p-value = 1.334e-06
##
## ------------------------------------------------------
## dat_waste_long$plant: PT3
##
##  Anderson-Darling normality test
##
## data:  dd[x, ]
## A = 0.23385, p-value = 0.7624
##
## ------------------------------------------------------
## dat_waste_long$plant: PT4
##
##  Anderson-Darling normality test
##
## data:  dd[x, ]
## A = 0.12363, p-value = 0.9834
##
## ------------------------------------------------------
## dat_waste_long$plant: PT5
##
##  Anderson-Darling normality test
##
## data:  dd[x, ]
## A = 0.27445, p-value = 0.6004
```

```
# the tidyverse version doesn't provide the headers
dat_waste_long %>%
  group_by(plant) %>%
  # . or .x to refer to the subset of rows for the given group
  group_map(
    ~ ad.test(.x$runup)
  )
```

```
## [[1]]
##
##  Anderson-Darling normality test
##
## data:  .x$runup
## A = 2.8685, p-value = 1.761e-07
##
##
## [[2]]
##
##  Anderson-Darling normality test
##
## data:  .x$runup
## A = 2.5207, p-value = 1.334e-06
##
##
## [[3]]
##
##  Anderson-Darling normality test
##
## data:  .x$runup
## A = 0.23385, p-value = 0.7624
##
##
## [[4]]
##
##  Anderson-Darling normality test
##
## data:  .x$runup
## A = 0.12363, p-value = 0.9834
##
##
## [[5]]
##
##  Anderson-Darling normality test
##
## data:  .x$runup
## A = 0.27445, p-value = 0.6004
```

For review purposes, I'll fit the ANOVA, but we would count on the following nonparametric method for inference.

```
fit_w <-
  aov(
    runup ~ plant
  , contrasts = list(plant = "contr.sum")
  , data = dat_waste_long
  )
summary(fit_w)
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
## plant        4    451  112.73    1.16  0.334
## Residuals   90   8749   97.21
```

```
fit_w
```

```
## Call:
##    aov(formula = runup ~ plant, data = dat_waste_long, contrasts = list(plant = "contr.sum"
##
## Terms:
##                     plant Residuals
## Sum of Squares    450.921  8749.088
## Deg. of Freedom         4        90
##
## Residual standard error: 9.859619
## Estimated effects may be unbalanced
```

```
# all pairwise comparisons among plants
# Fisher's LSD (FSD) uses "none"
# Contrasts to perform pairwise comparisons
cont_w <-
  emmeans::emmeans(
    fit_w
  , specs = "plant"
  )

# Means and CIs
cont_w                                  # adjust = "tukey" is default
```

```
##  plant emmean   SE df lower.CL upper.CL
##  PT1     4.52 2.10 90    0.347     8.70
##  PT2     8.83 2.10 90    4.656    13.01
##  PT3     4.83 2.26 90    0.338     9.33
##  PT4     7.49 2.26 90    2.996    11.98
##  PT5    10.38 2.73 90    4.944    15.81
##
## Confidence level used: 0.95
```

```
confint(cont_w, adjust = "bonferroni")
```

```
##  plant emmean   SE df lower.CL upper.CL
##  PT1     4.52 2.10 90    -1.01     10.1
##  PT2     8.83 2.10 90     3.30     14.4
##  PT3     4.83 2.26 90    -1.12     10.8
##  PT4     7.49 2.26 90     1.54     13.4
##  PT5    10.38 2.73 90     3.18     17.6
##
## Confidence level used: 0.95
## Conf-level adjustment: bonferroni method for 5 estimates
```

```
# Pairwise comparisons
cont_w %>% pairs(adjust = "bonf")       # adjust = "tukey" is default
```

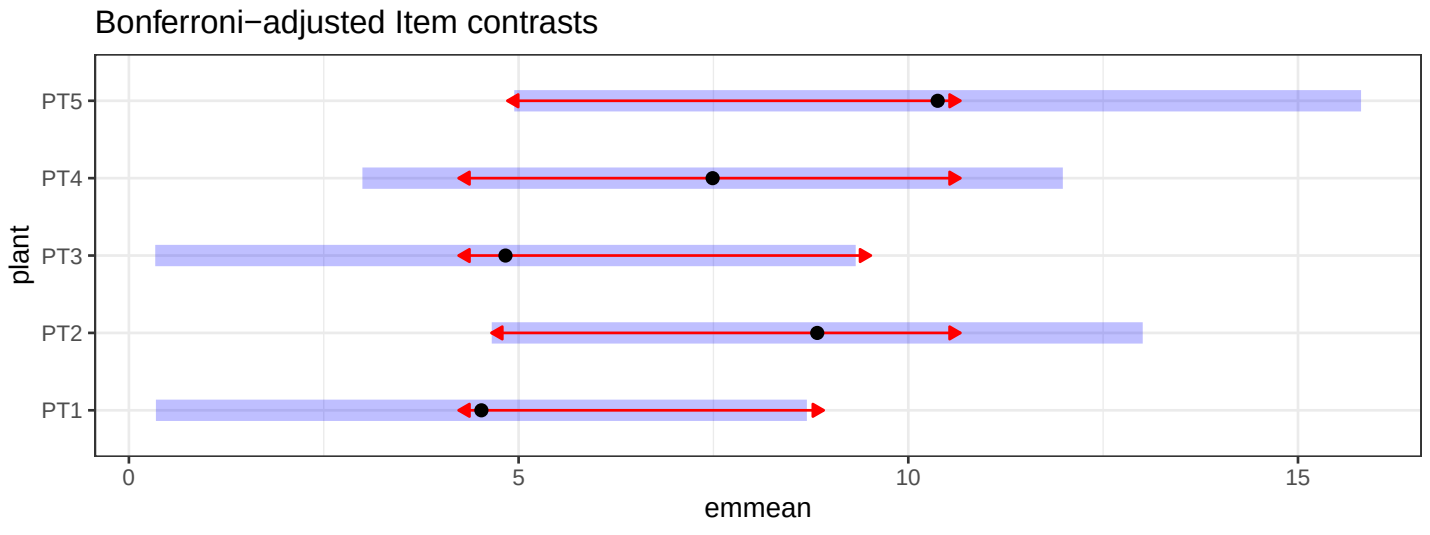
```
##  contrast  estimate   SE df t.ratio p.value
```

```
##  PT1 - PT2   -4.309 2.97 90 -1.450  1.0000
##  PT1 - PT3   -0.309 3.09 90 -0.100  1.0000
##  PT1 - PT4   -2.967 3.09 90 -0.961  1.0000
##  PT1 - PT5   -5.854 3.45 90 -1.697  0.9310
##  PT2 - PT3    4.000 3.09 90  1.295  1.0000
##  PT2 - PT4    1.342 3.09 90  0.435  1.0000
##  PT2 - PT5   -1.545 3.45 90 -0.448  1.0000
##  PT3 - PT4   -2.658 3.20 90 -0.831  1.0000
##  PT3 - PT5   -5.545 3.55 90 -1.563  1.0000
##  PT4 - PT5   -2.887 3.55 90 -0.814  1.0000
##
## P value adjustment: bonferroni method for 10 tests
```

EMM plot interpretation

This **EMM plot (Estimated Marginal Means, aka Least-Squares Means)** is only available when conditioning on one variable. The **blue bars** are confidence intervals for the EMMs; don't ever use confidence intervals for EMMs to perform comparisons – they can be very misleading. The **red arrows** are for the comparisons among means; the degree to which the "comparison arrows" overlap reflects as much as possible the significance of the comparison of the two estimates. If an arrow from one mean overlaps an arrow from another group, the difference is not significant, based on the adjust setting (which defaults to "tukey").

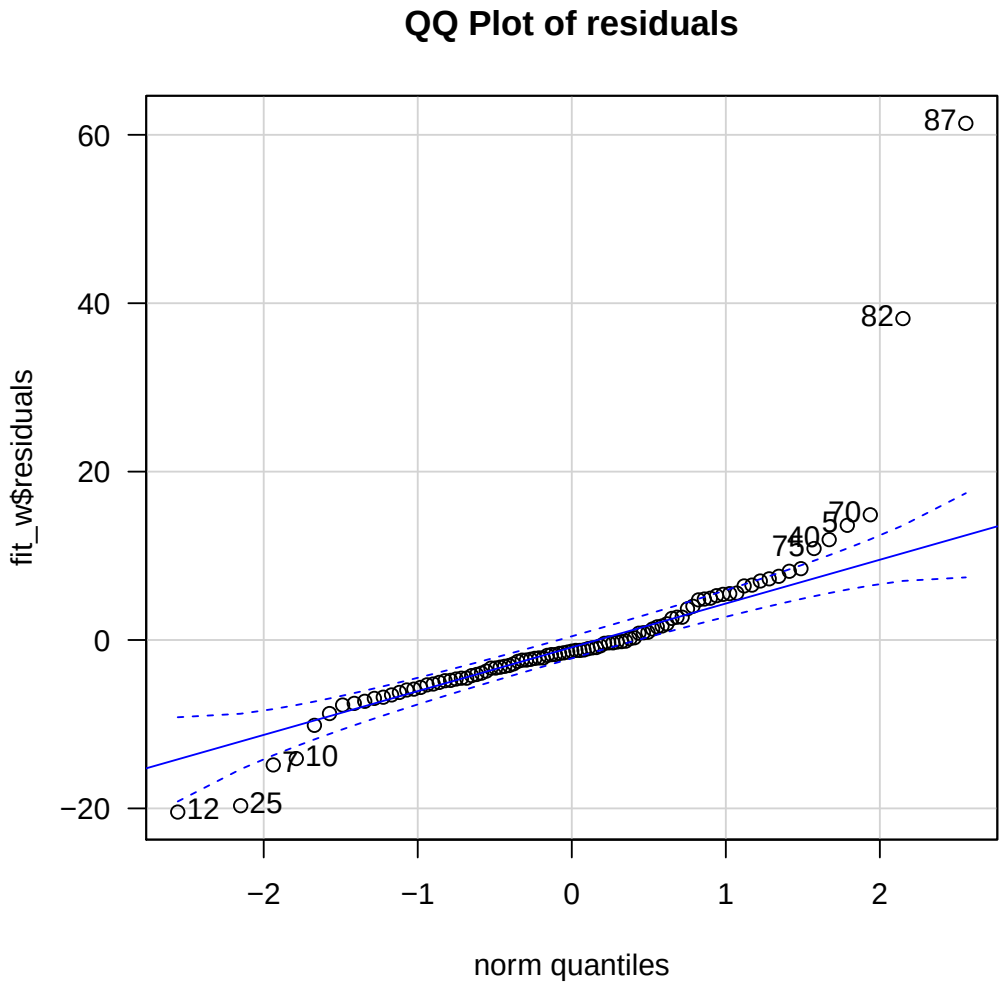
```
# Plot means and contrasts
p <- plot(cont_w, comparisons = TRUE, adjust = "bonf")  # adjust = "tukey" is default
p <- p + labs(title = "Bonferroni-adjusted Item contrasts")
p <- p + theme_bw()
print(p)
```



The residuals show many outliers and a leptokurtic deviation from normality.

```
# QQ plot
par(mfrow=c(1,1))
library(car)
qqPlot(fit_w$residuals, las = 1, id = list(n = 10, cex = 1), lwd = 1
       , main="QQ Plot of residuals")
```

```
##  [1] 87 82 12 25 70  7 10  5 40 75
```



Kruskal-Wallis ANOVA is a non-parametric method for testing the hypothesis of equal population medians against the alternative that not all population medians are equal. It's still not perfect here because the distributional shapes are not all the same, but it is a better alternative than the ANOVA.

```
# KW ANOVA
fit_wk <- kruskal.test(runup ~ plant, data = dat_waste_long)
fit_wk
```

```
##
```

```
##  Kruskal-Wallis rank sum test
##
## data:  runup by plant
## Kruskal-Wallis chi-squared = 15.319, df = 4, p-value =
## 0.004084
```

```
# Bonferroni 95% pairwise comparisions with continuity correction
#   in the normal approximation for the p-value

# names for plants being compared
plant_levels <- levels(dat_waste_long$plant)

# loop over all pairs of plants
for (i1_pt in 1:(length(plant_levels) - 1)) {
  for (i2_pt in (i1_pt+1):(length(plant_levels))) {
    wt <-
      wilcox.test(
        dat_waste_long %>% filter(plant == plant_levels[i1_pt]) %>% pull(runup)
      , dat_waste_long %>% filter(plant == plant_levels[i2_pt]) %>% pull(runup)
      , conf.int   = TRUE
      , conf.level = 1 - 0.05 / choose(length(plant_levels), 2)
      )
    # print a header and the wilcox result
    cat(plant_levels[i1_pt], plant_levels[i2_pt], "-------------------------")
    print(wt)
  }
}
```

```
## Warning in wilcox.test.default(dat_waste_long %>% filter(plant == plant_levels[i1_pt])
%>% :  cannot compute exact p-value with ties
## Warning in wilcox.test.default(dat_waste_long %>% filter(plant == plant_levels[i1_pt])
%>% :  cannot compute exact confidence intervals with ties
## PT1 PT2 -------------------------
##  Wilcoxon rank sum test with continuity correction
##
## data:  dat_waste_long %>% filter(plant == plant_levels[i1_pt]) %>% pull(runup) and dat_wast
## W = 131.5, p-value = 0.009813
## alternative hypothesis: true location shift is not equal to 0
## 99.5 percent confidence interval:
##  -8.299958  1.599947
## sample estimates:
## difference in location
##              -4.399951
## Warning in wilcox.test.default(dat_waste_long %>% filter(plant == plant_levels[i1_pt])
%>% :  cannot compute exact p-value with ties
## Warning in wilcox.test.default(dat_waste_long %>% filter(plant == plant_levels[i1_pt])
%>% :  cannot compute exact confidence intervals with ties
## PT1 PT3 -------------------------
##  Wilcoxon rank sum test with continuity correction
##
```

```
## data:  dat_waste_long %>% filter(plant == plant_levels[i1_pt]) %>% pull(runup) and dat_wast
## W = 141.5, p-value = 0.07978
## alternative hypothesis: true location shift is not equal to 0
## 99.5 percent confidence interval:
##  -6.900028  2.700029
## sample estimates:
## difference in location
##              -2.500047
## Warning in wilcox.test.default(dat_waste_long %>% filter(plant == plant_levels[i1_pt])
%>% :  cannot compute exact p-value with ties
## Warning in wilcox.test.default(dat_waste_long %>% filter(plant == plant_levels[i1_pt])
%>% :  cannot compute exact confidence intervals with ties
## PT1 PT4 -------------------------
##  Wilcoxon rank sum test with continuity correction
##
## data:  dat_waste_long %>% filter(plant == plant_levels[i1_pt]) %>% pull(runup) and dat_wast
## W = 85, p-value = 0.001241
## alternative hypothesis: true location shift is not equal to 0
## 99.5 percent confidence interval:
##  -8.900056 -1.099910
## sample estimates:
## difference in location
##              -5.300051
## Warning in wilcox.test.default(dat_waste_long %>% filter(plant == plant_levels[i1_pt])
%>% :  cannot compute exact p-value with ties
## Warning in wilcox.test.default(dat_waste_long %>% filter(plant == plant_levels[i1_pt])
%>% :  cannot compute exact confidence intervals with ties
## PT1 PT5 -------------------------
##  Wilcoxon rank sum test with continuity correction
##
## data:  dat_waste_long %>% filter(plant == plant_levels[i1_pt]) %>% pull(runup) and dat_wast
## W = 76, p-value = 0.02318
## alternative hypothesis: true location shift is not equal to 0
## 99.5 percent confidence interval:
##  -15.50007   3.60001
## sample estimates:
## difference in location
##              -8.703538
## Warning in wilcox.test.default(dat_waste_long %>% filter(plant == plant_levels[i1_pt])
%>% :  cannot compute exact p-value with ties
## Warning in wilcox.test.default(dat_waste_long %>% filter(plant == plant_levels[i1_pt])
%>% :  cannot compute exact confidence intervals with ties
## PT2 PT3 -------------------------
##  Wilcoxon rank sum test with continuity correction
##
## data:  dat_waste_long %>% filter(plant == plant_levels[i1_pt]) %>% pull(runup) and dat_wast
## W = 238, p-value = 0.4562
## alternative hypothesis: true location shift is not equal to 0
```

```
## 99.5 percent confidence interval:
##  -3.50007  7.20000
## sample estimates:
## difference in location
##               1.352784
## Warning in wilcox.test.default(dat_waste_long %>% filter(plant == plant_levels[i1_pt])
%>% :  cannot compute exact p-value with ties
## Warning in wilcox.test.default(dat_waste_long %>% filter(plant == plant_levels[i1_pt])
%>% :  cannot compute exact confidence intervals with ties
## PT2 PT4 ------------------------
##  Wilcoxon rank sum test with continuity correction
##
## data:  dat_waste_long %>% filter(plant == plant_levels[i1_pt]) %>% pull(runup) and dat_wast
## W = 186, p-value = 0.5563
## alternative hypothesis: true location shift is not equal to 0
## 99.5 percent confidence interval:
##  -5.900014  3.800023
## sample estimates:
## difference in location
##              -1.099914
## Warning in wilcox.test.default(dat_waste_long %>% filter(plant == plant_levels[i1_pt])
%>% :  cannot compute exact p-value with ties
## Warning in wilcox.test.default(dat_waste_long %>% filter(plant == plant_levels[i1_pt])
%>% :  cannot compute exact confidence intervals with ties
## PT2 PT5 ------------------------
##  Wilcoxon rank sum test with continuity correction
##
## data:  dat_waste_long %>% filter(plant == plant_levels[i1_pt]) %>% pull(runup) and dat_wast
## W = 99, p-value = 0.1375
## alternative hypothesis: true location shift is not equal to 0
## 99.5 percent confidence interval:
##  -13.100001   7.400037
## sample estimates:
## difference in location
##              -4.630905
## Warning in wilcox.test.default(dat_waste_long %>% filter(plant == plant_levels[i1_pt])
%>% :  cannot compute exact p-value with ties
## Warning in wilcox.test.default(dat_waste_long %>% filter(plant == plant_levels[i1_pt])
%>% :  cannot compute exact confidence intervals with ties
## PT3 PT4 ------------------------
##  Wilcoxon rank sum test with continuity correction
##
## data:  dat_waste_long %>% filter(plant == plant_levels[i1_pt]) %>% pull(runup) and dat_wast
## W = 117, p-value = 0.06583
## alternative hypothesis: true location shift is not equal to 0
## 99.5 percent confidence interval:
##  -6.800001  1.699976
## sample estimates:
```

```
## difference in location
##              -2.400038
##
## PT3 PT5 -------------------------
##  Wilcoxon rank sum test
##
## data:  dat_waste_long %>% filter(plant == plant_levels[i1_pt]) %>% pull(runup) and dat_wast
## W = 67, p-value = 0.03018
## alternative hypothesis: true location shift is not equal to 0
## 99.5 percent confidence interval:
##  -13.4   1.7
## sample estimates:
## difference in location
##                   -6.6
## Warning in wilcox.test.default(dat_waste_long %>% filter(plant == plant_levels[i1_pt])
%>% :  cannot compute exact p-value with ties
## Warning in wilcox.test.default(dat_waste_long %>% filter(plant == plant_levels[i1_pt])
%>% :  cannot compute exact confidence intervals with ties
## PT4 PT5 -------------------------
##  Wilcoxon rank sum test with continuity correction
##
## data:  dat_waste_long %>% filter(plant == plant_levels[i1_pt]) %>% pull(runup) and dat_wast
## W = 82, p-value = 0.1157
## alternative hypothesis: true location shift is not equal to 0
## 99.5 percent confidence interval:
##  -11.099965   4.799945
## sample estimates:
## difference in location
##              -4.000035
```

1.6 ADA1 Chapter 7: Categorical data analysis

Returning to the turkey dataset, below is the cross-classification of `orig` by `gt25mo`.

```
#### Example: Turkey, Chapter 7
# create a frequency table from two columns of categorical data
xt <- xtabs( ~ orig + gt25mo, data = dat_turkey)
# display the table
xt
```

```
##     gt25mo
## orig FALSE TRUE
##   va     2    6
##   wi     3    4
```

```
# summary from xtabs() is the same as chisq.test() without continuity correction
summary(xt)
```

```
## Call: xtabs(formula = ~orig + gt25mo, data = dat_turkey)
## Number of cases in table: 15
## Number of factors: 2
## Test for independence of all factors:
##  Chisq = 0.5357, df = 1, p-value = 0.4642
##  Chi-squared approximation may be incorrect
```

```
# same as xtabs()
x_summary <- chisq.test(xt, correct=FALSE)
```

```
## Warning in chisq.test(xt, correct = FALSE): Chi-squared approximation may be incorrect
```

```
x_summary
```

```
##
##  Pearson's Chi-squared test
##
## data:  xt
## X-squared = 0.53571, df = 1, p-value = 0.4642
```

```
# the default is to perform Yates' continuity correction
chisq.test(xt)
```

```
## Warning in chisq.test(xt):  Chi-squared approximation may be incorrect
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  xt
## X-squared = 0.033482, df = 1, p-value = 0.8548
```

```
# Fisher's exact test
fisher.test(xt)
```

```
##
##  Fisher's Exact Test for Count Data
```

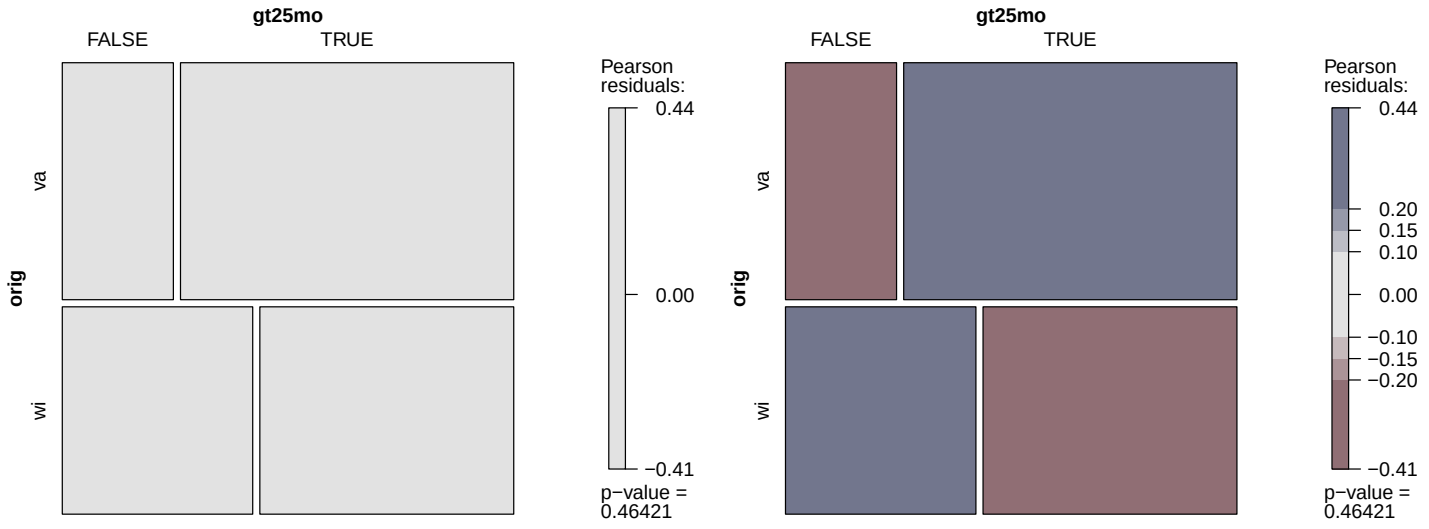
```
##
## data:  xt
## p-value = 0.6084
## alternative hypothesis: true odds ratio is not equal to 1
## 95 percent confidence interval:
##  0.02687938 6.23767632
## sample estimates:
## odds ratio
##  0.4698172
```

A mosaic plot is for categorical data. Area represents frequency. The default shading is a good start, since colors only appear when there's evidence of association related to those cell values. In our example, there's insufficient evidence for association, so the default shading is all gray.

```
library(vcd)    # for mosaic()
```

```
## Loading required package:  grid
##
## Attaching package:  'vcd'
## The following object is masked from 'package:BSDA':
##
##     Trucks
```

```
# shading based on significance relative to appropriate chi-square distribution
mosaic(xt, shade=TRUE)
# you can define your own interpolated shading
mosaic(xt, shade=TRUE, gp_args = list(interpolate = seq(.1,.2,.05)))
```



From the `chisq.test()` above, we make a table to summarize important

values from that analysis and compare the observed and expected frequencies in plots.

```
# use output in x_summary and create table
x_table <-
  data.frame(
    obs    = x_summary$observed
  , exp    = x_summary$expected
  , res    = x_summary$residuals
  , chisq  = x_summary$residuals^2
  , stdres = x_summary$stdres
  )
```

```
## Warning in data.frame(obs = x_summary$observed, exp = x_summary$expected, :  row names
were found from a short variable and have been discarded
```

```
# There are duplicate row and col identifiers in this x_table
#   because we're creating vectors from a two-way table
#   and columns identifying row and col names are automatically added.
# Can you figure out the naming scheme?
x_table
```

```
##   obs.orig obs.gt25mo obs.Freq exp.FALSE exp.TRUE res.orig res.gt25mo
## 1       va      FALSE        2  2.666667 5.333333       va      FALSE
## 2       wi      FALSE        3  2.333333 4.666667       wi      FALSE
## 3       va       TRUE        6  2.666667 5.333333       va       TRUE
## 4       wi       TRUE        4  2.333333 4.666667       wi       TRUE
##     res.Freq chisq.orig chisq.gt25mo chisq.Freq stdres.orig
## 1 -0.4082483         va        FALSE 0.16666667          va
## 2  0.4364358         wi        FALSE 0.19047619          wi
## 3  0.2886751         va         TRUE 0.08333333          va
## 4 -0.3086067         wi         TRUE 0.09523810          wi
##   stdres.gt25mo stdres.Freq
## 1         FALSE  -0.7319251
## 2         FALSE   0.7319251
## 3          TRUE   0.7319251
## 4          TRUE  -0.7319251
```

```
# create a single column with a joint cell name
x_table$cellname <- paste(as.character(x_table$obs.orig)
                        , as.character(x_table$obs.gt25mo)
                        , sep="_")
# expected frequencies in a single column
x_table$exp    <- c(x_table$exp.FALSE[1:2], x_table$exp.TRUE[3:4])
# create a simpler name for the obs, res, chisq, stdres
x_table$obs    <- x_table$obs.Freq
x_table$res    <- x_table$res.Freq
x_table$chisq  <- x_table$chisq.Freq
x_table$stdres <- x_table$stdres.Freq

# include only the "cleaned" columns
x_table <-
```

```
  x_table %>%
  select(cellname, obs, exp, res, chisq, stdres)
x_table
```

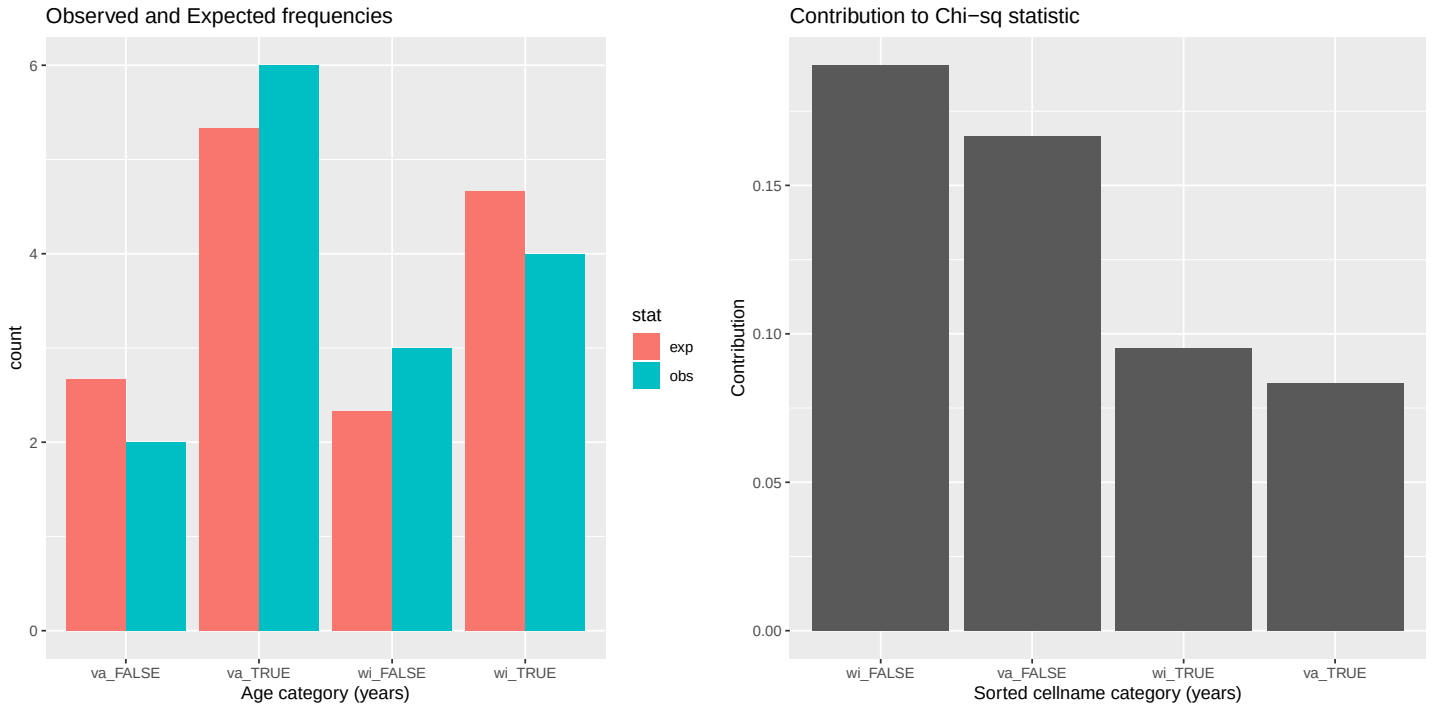
```
##   cellname obs      exp        res      chisq     stdres
## 1 va_FALSE   2 2.666667 -0.4082483 0.16666667 -0.7319251
## 2 wi_FALSE   3 2.333333  0.4364358 0.19047619  0.7319251
## 3  va_TRUE   6 5.333333  0.2886751 0.08333333  0.7319251
## 4  wi_TRUE   4 4.666667 -0.3086067 0.09523810 -0.7319251
```

```
x_table_obsexp <-
  x_table %>%
  select(
    cellname, obs, exp
  ) %>%
  pivot_longer(
    cols      = c("obs", "exp")
  , names_to  = "stat"
  , values_to = "value"
  ) %>%
  mutate(
    plant = factor(stat)
  )
```

Plot observed vs expected frequencies, and the contribution to chi-square statistic sorted decending.

```
# Observed vs Expected counts
library(ggplot2)
p <- ggplot(x_table_obsexp, aes(x = cellname, fill = stat, weight=value))
p <- p + geom_bar(position="dodge")
p <- p + labs(title = "Observed and Expected frequencies")
p <- p + xlab("Age category (years)")
print(p)


# Contribution to chi-sq
# pull out only the cellname and chisq columns
x_table_chisq <-
  x_table %>%
  select(
    cellname, chisq
  ) %>%
  mutate(
    # reorder the cellname categories to be descending relative to the chisq statistic
    cellname = reorder(cellname, -chisq)
  )

p <- ggplot(x_table_chisq, aes(x = cellname, weight = chisq))
p <- p + geom_bar()
p <- p + labs(title = "Contribution to Chi-sq statistic")
```

```
p <- p + xlab("Sorted cellname category (years)")
p <- p + ylab("Contribution")
print(p)
```



1.7 ADA1 Chapter 8: Correlation and regression

Rocket Propellant Data A rocket motor is manufactured by bonding an igniter propellant and a sustainer propellant together inside a metal housing. The shear strength of the bond between the two types of propellant is an important quality characteristic. It is suspected that shear strength is related to the age in weeks of the batch of sustainer propellant. Twenty observations on these two characteristics are given below. The first column is shear strength in psi, the second is age of propellant in weeks.

```
#### Example: Rocket, Chapter 8
# this file uses spaces as delimiters, so use read_table()
dat_rocket <-
  read_table("http://statacumen.com/teach/ADA2/notes/ADA2_notes_Ch01_rocket.dat")
```

```
## Parsed with column specification:
## cols(
##   shearpsi = col_double(),
##   agewks = col_double()
## )
```

```
dat_rocket <-
  dat_rocket %>%
  mutate(
    id = 1:n() # add an id variable to identify observations
  )
str(dat_rocket)
```

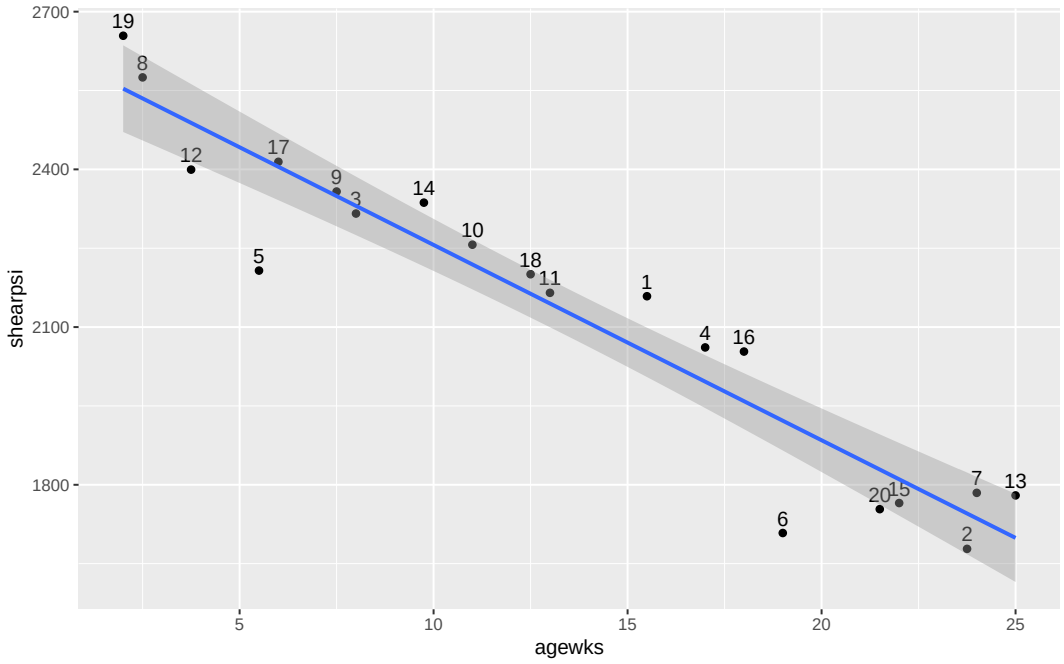
```
## Classes 'spec_tbl_df', 'tbl_df', 'tbl' and 'data.frame': 20 obs. of  3 variables:
##  $ shearpsi: num  2159 1678 2316 2061 2208 ...
##  $ agewks  : num  15.5 23.8 8 17 5.5 ...
##  $ id      : int  1 2 3 4 5 6 7 8 9 10 ...
```

```
head(dat_rocket)
```

```
## # A tibble: 6 x 3
##   shearpsi agewks    id
##      <dbl>  <dbl> <int>
## 1    2159.   15.5     1
## 2    1678.   23.8     2
## 3    2316     8       3
## 4    2061.   17       4
## 5    2208.    5.5     5
## 6    1708.   19       6
```

```
# ggplot: Plot the data with linear regression fit and confidence bands
library(ggplot2)
```

```
p <- ggplot(dat_rocket, aes(x = agewks, y = shearpsi, label = id))
p <- p + geom_point()
# plot labels next to points
p <- p + geom_text(hjust = 0.5, vjust = -0.5)
# plot regression line and confidence band
p <- p + geom_smooth(method = lm)
print(p)
```



The data are reasonably linear, so fit the regression.

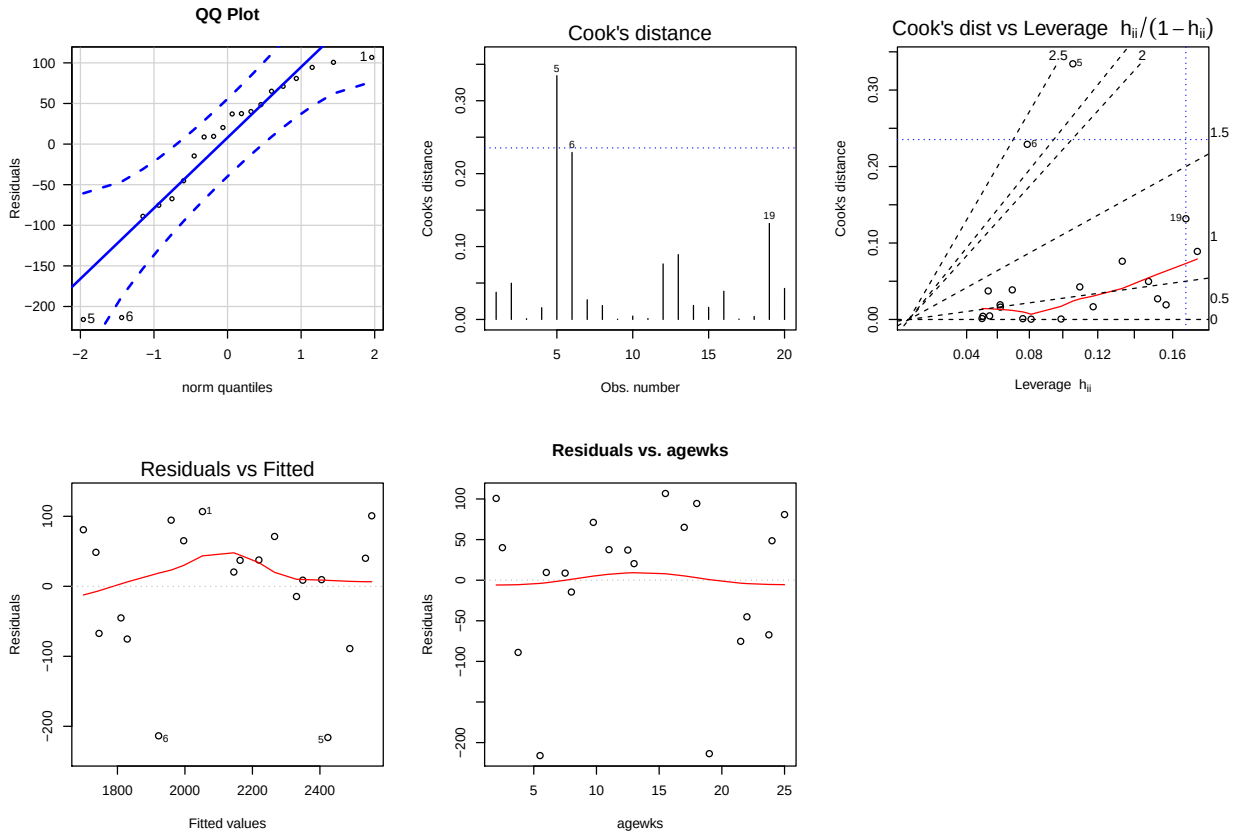
```
# fit the simple linear regression model
lm_shearpsi_agewks <- lm(shearpsi ~ agewks, data = dat_rocket)
# use summary() to get t-tests of parameters (slope, intercept)
summary(lm_shearpsi_agewks)
```

```
##
## Call:
## lm(formula = shearpsi ~ agewks, data = dat_rocket)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -215.98  -50.68   28.74   66.61  106.76
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2627.822     44.184   59.48  < 2e-16 ***
## agewks       -37.154      2.889  -12.86 1.64e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 96.11 on 18 degrees of freedom
## Multiple R-squared:  0.9018,Adjusted R-squared:  0.8964
```

```
## F-statistic: 165.4 on 1 and 18 DF,  p-value: 1.643e-10
```

Plot diagnostics.

```
# plot diagnistics
lm_diag_plots(lm_shearpsi_agewks, sw_plot_set = "simple")
```



The relationship between shear strength and age is fairly linear with predicted shear strength decreasing as the age of the propellant increases. The fitted LS line is

$$\text{Predicted shear strength} = 2627.8 - 37.2\ \text{Age}.$$

The test for $H_0 : \beta_1 = 0$ (zero slope for the population regression line) is highly significant: p-value< 0.0001. Also note that $R^2 = 0.9018$ so the linear relationship between shear strength and age explains about 90% of the variation in shear strength.

The data plot and residual information identify observations 5 and 6 as potential outliers ($r_5 = -2.38, r_6 = -2.32$). The predicted values for these observations are much greater than the observed shear strengths. These same observations appear as potential outliers in the normal scores plot and the plot

of r_i against $\hat{Y}_i$. Observations 5 and 6 also have the largest influence on the analysis; see the Cook's distance values.

A sensible next step would be to repeat the analysis holding out the most influential case, observation 5. It should be somewhat clear that the influence of case 6 would increase dramatically once case 5 is omitted from the analysis. Since both cases have essentially the same effect on the positioning of the LS line, I will assess the impact of omitting both simultaneously.

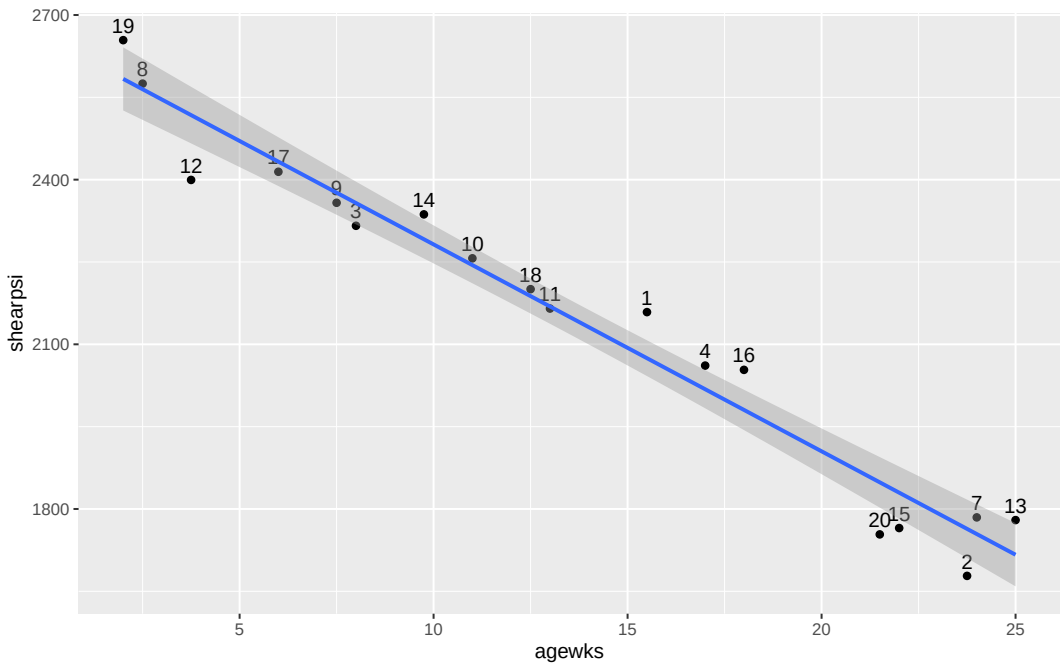
Before we hold out these cases, how do you think the LS line will change? My guess is these cases are pulling the LS line down, so the intercept of the LS line should increase once these cases are omitted. Holding out either case 5 or 6 would probably also affect the slope, but my guess is that when they are both omitted the slope will change little. (Is this my experience speaking, or have I already seen the output? Both.) What will happen to R^2 when we delete these points?

Exclude observations 5 and 6 and redo the analysis.

```
# exclude observations 5 and 6
dat_rocket_56 <-
  dat_rocket %>%
  slice(
    -c(5,6)
  )
head(dat_rocket_56)
```

```
## # A tibble: 6 x 3
##   shearpsi agewks    id
##      <dbl>  <dbl> <int>
## 1    2159.   15.5     1
## 2    1678.   23.8     2
## 3    2316     8       3
## 4    2061.   17       4
## 5    1785.   24       7
## 6    2575     2.5     8
```

```
# ggplot: Plot the data with linear regression fit and confidence bands
library(ggplot2)
p <- ggplot(dat_rocket_56, aes(x = agewks, y = shearpsi, label = id))
p <- p + geom_point()
# plot labels next to points
p <- p + geom_text(hjust = 0.5, vjust = -0.5)
# plot regression line and confidence band
p <- p + geom_smooth(method = lm)
print(p)
```



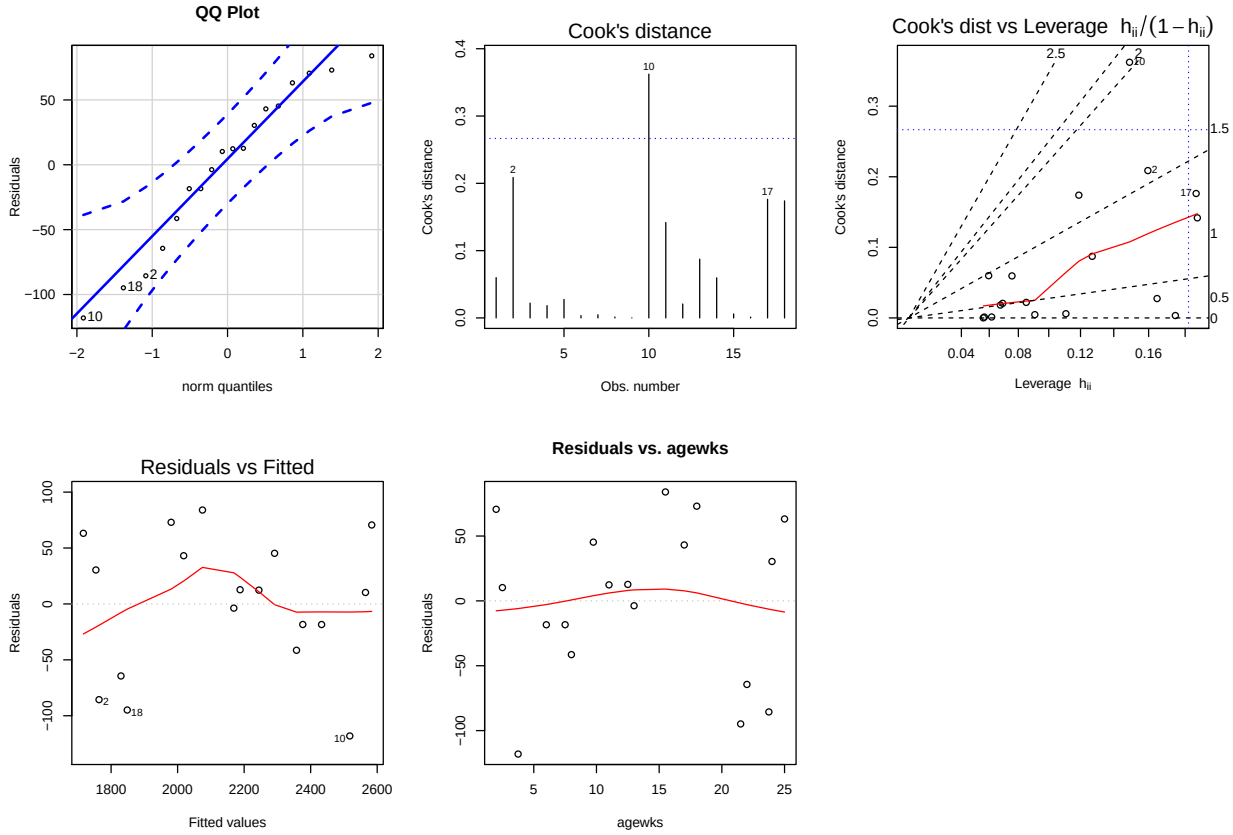
The data are reasonably linear, so fit the regression.

```
# fit the simple linear regression model
lm_shearpsi_agewks <- lm(shearpsi ~ agewks, data = dat_rocket_56)
# use summary() to get t-tests of parameters (slope, intercept)
summary(lm_shearpsi_agewks)
```

```
##
## Call:
## lm(formula = shearpsi ~ agewks, data = dat_rocket_56)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -118.07  -35.67   11.31   44.75   83.98
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2658.973     30.533   87.08  < 2e-16 ***
## agewks       -37.694      1.979  -19.05 2.02e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 62.97 on 16 degrees of freedom
## Multiple R-squared:  0.9578,Adjusted R-squared:  0.9551
## F-statistic: 362.9 on 1 and 16 DF,  p-value: 2.023e-12
```

Plot diagnostics.

```
# plot diagnistics
lm_diag_plots(lm_shearpsi_agewks, sw_plot_set = "simple")
```



Some summaries for the complete analysis, and when cases 5 and 6 are held out, are given below. The summaries lead to the following conclusions:

1. Holding out cases 5 and 6 has little effect on the estimated LS line. Predictions of shear strength are slightly larger after holding out these two cases (recall that intercept increased, but slope was roughly the same!)
2. Holding out these two cases decreases $\hat{\sigma}$ considerably, and leads to a modest increase in R^2. The complete data set will give wider CI and prediction intervals than the analysis which deletes case 5 and 6 because $\hat{\sigma}$ decreases when these points are omitted.
3. Once these cases are held out, the normal scores plot and plot of the studentized residuals against fitted values shows no significant problems. One observation has a large Cook's D but does not appear to be extremely influential.

Without any substantive reason to explain the low shear strengths for cases 5 and 6, I am hesitant to delete either case from the analysis. I feel relatively confident that including these cases will not seriously limit the use of the model.

Feature	Full data	Omit 5 and 6
b_0	2627.82	2658.97
b_1	−37.15	−37.69
R^2	0.9018	0.9578
$\hat{\sigma}$	96.10	62.96
p-val for $H_0 : \beta_1 = 0$	0.0001	0.0001

Here is a comparison of the predicted or fitted values for selected observations in the data set, based on the two fits.

Observation	Actual Shear Strength	Pred, full	Pred, omit 5,6
1	2159	2052	2075
2	1678	1745	1764
4	2061	1996	2018
8	2575	2535	2565
10	2257	2219	2244
15	1765	1810	1830
18	2201	2163	2188
20	1754	1829	1849

Review complete

Now that we're warmed up, let's dive into new material!