Notes and Exercises Chapter 22 – Inference for Comparing Many Means
Math 217 **Name** _____
Saidi **Date** _____

22 Inference for comparing many means

Sometimes we want to compare means across many groups. We might initially think to do pairwise comparisons. For example, if there were three groups, we might be tempted to compare the first mean with the second, then with the third, and then finally compare the second and third means for a total of three comparisons. However, this strategy can be treacherous. If we have many groups and do many comparisons, it is likely that we will eventually find a difference just by chance, even if there is no difference in the populations. Instead, we should apply a holistic test to check whether there is evidence that at least one pair groups are in fact different, and this is where **ANOVA** saves the day.

In this section, we will learn a new method called **analysis of variance (ANOVA)** and a new test statistic called an F-statistic (which we will introduce in our discussion of mathematical models). ANOVA uses a single hypothesis test to check whether the means across many groups are equal:

- Ho: The mean outcome is the same across all groups.

$$\mu_1 = \mu_2 = \mu_3 = \dots = \mu_i$$

- Ha: At least one mean is different.

Generally we must check three conditions on the data before performing ANOVA:

- the observations are independent within and between groups,
- the responses within each group are nearly normal, and
- the variability across the groups is about equal.

homogeneous variability across groups

When these three conditions are met, we may perform an ANOVA to determine whether the data provide convincing evidence against the null hypothesis that all the are equal.

Example 1 **Sweet Corn**

Researchers want to study the question of whether organic methods can be successfully used to control harmful insects. They compared the weights of ears of corn under 5 conditions.

Treatment 1: Nematodes
Treatment 2: Parasitic wasps
Treatment 3: Nematodes and parasitic wasps
Treatment 4: Bacteria
Treatment 5: Control

} ***I = 5***

Result: Ears of corn were randomly sampled from each plot and weighed.

Look below at the summary and distribution information. ***Notice there is variation both between the groups and within each individual treatment group.***

Table 11.1.1 Weights (ounces) of ears of sweet corn

	Treatment				
	1	2	3	4	5
	16.5	11.0	8.5	16.0	13.0
	15.0	15.0	13.0	14.5	10.5
	11.5	9.0	12.0	15.0	11.0
	12.0	9.0	10.0	9.0	10.0
	12.5	11.5	12.5	10.5	14.0
	9.0	11.0	8.5	14.0	12.0
	16.0	9.0	9.5	12.5	11.0
	6.5	10.0	7.0	9.0	9.5
	8.0	9.0	10.5	9.0	18.5
	14.5	8.0	10.5	9.0	17.0
	7.0	8.0	13.0	6.5	10.0
	10.5	5.0	9.0	8.5	11.0
Mean	11.58	9.63	10.33	11.13	12.29
SD	3.47	2.42	1.96	3.12	2.87
n	12	12	12	12	12

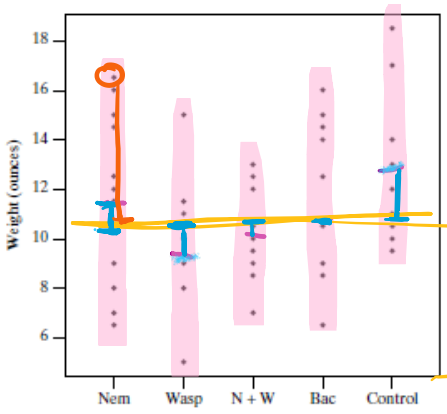


Figure 11.1.1 Weights of ears of corn receiving five different treatments

Between $(\bar{y}_i - \bar{\bar{y}})$

within $(y_{ij} - \bar{y}_i)$

6 (hypothetical)

$\bar{\bar{y}}$ = grand mean = mean of all means

y_{ij} = one observation

We extend the idea of a **randomization test** from Chapter 20 to compare all ***I*** groups.

Question:

If each of the results in the table were placed in one of the 5 groups at random, how likely would it be that one of the five would be as large as, say, 12.29 (the mean weight of the control group)? Look at the largest difference between (control – treatment 2 = 2.66)

In a simulation of 10,000 trials to answer this question, 1626 had a difference as large or more than 2.66. Just as we did this type of simulation in chapter 7 with the randomization test, we can see that this proportion of differences of 1626/10000 gives a p-value of 0.163. This means the observed differences could have been due to chance.

22.2.1 Observed data

The method of analysis of variance in this context focuses on answering one question: is the variability in the sample means so large that it seems unlikely to be from chance alone? This question is different from earlier testing procedures since we will *simultaneously* consider many groups, and evaluate whether their sample means differ more than we would expect from natural variation. We call this variability the **mean square between groups (MSG)**, and it has an associated degrees of freedom, when there are k groups. The MSG can be thought of as a scaled variance formula for means. If the null hypothesis is true, any variation in the sample means is due to chance and shouldn't be too large. Details of MSG calculations are provided in the footnote.

The mean square between the groups is, on its own, quite useless in a hypothesis test. We need a benchmark value for how much variability should be expected among the sample means if the null hypothesis is true. To this end, we compute a pooled variance estimate, often abbreviated as the **mean square error (**MSE), which has an associated degrees of freedom value . It is helpful to think of MSE as a measure of the variability within the groups.

When the null hypothesis is true, any differences among the sample means are only due to chance, and the MSG and MSE should be about equal. As a test statistic for ANOVA, we examine the F-statistic, which is a ratio:

The **MSG represents a measure of the between-group variability,** and **MSE measures the variability within** each of the groups.

Conditions:

- independent observations, both within and across groups

- large samples and no extreme outliers

Example 2:

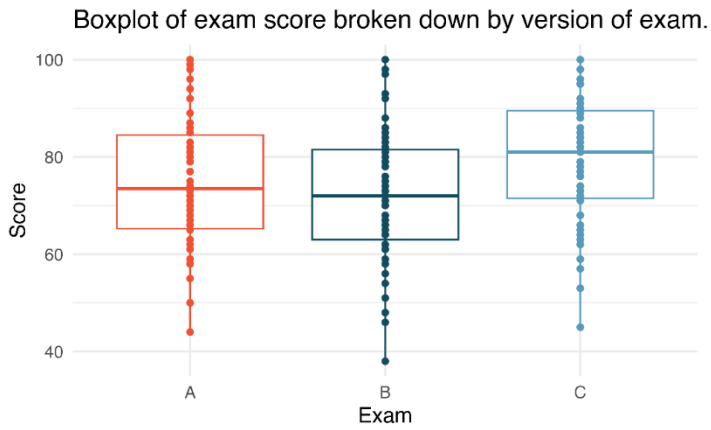
Suppose that a teacher had had such an extremely large class that three different exams were given: A, B, and C. The table and figure below provide a summary of the data including exam C. Again, we would like to investigate whether the difficulty of the exams is the same across the three exams, so the test is

- Ho: The inherent average difficulty is the same across the three exams.
- Ha: At least one of the exams is inherently more (or less) difficult than the others.

The classdata data can be found in the **openintro** R package.

Table: Summary statistics of scores for each exam version.

Exam	n	Mean	SD	Min	Max
A	58	75.1	13.9	44	100
B	55	72.0	13.8	38	100
C	51	78.9	13.1	45	100



The figure below shows the process of randomizing the three different exams to the observed exam scores. If the null hypothesis is true, then the score on each exam should represent the true student ability on that material. It shouldn't matter whether they were given exam A or exam B or exam C. By reallocating which student got which exam, we are able to understand how the difference in average exam scores changes due only to natural variability. There is only one iteration of the randomization process in Figure 22.4, leading to three different randomized sample means (computed assuming the null hypothesis is true).

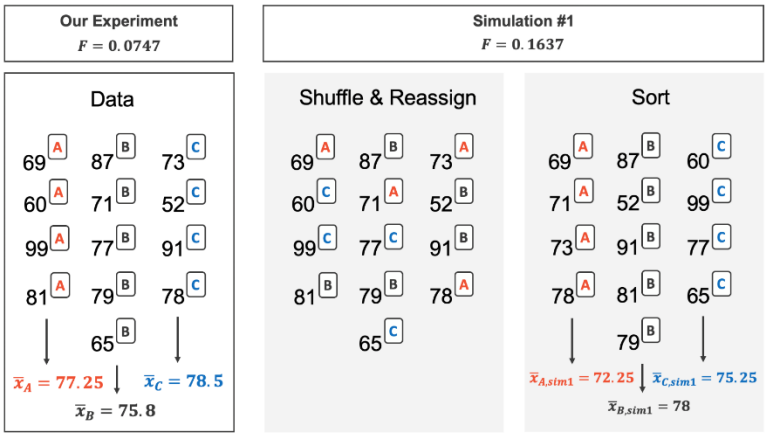


Figure: The version of the test (A or B or C) is randomly allocated to the test scores, under the null assumption that the tests are equally difficult.

In the two-sample case, the null hypothesis was investigated using the difference in the sample means. However, as noted above, with three groups (three different exams), the comparison of the three sample means gets slightly more complicated. We have already derived the F-statistic which is exactly the way to compare the averages across three or more groups! Recall, the F statistic is a ratio of how the groups differ (MSG) as compared to how the observations within a group vary (MSE).

The figure below shows the values of the simulated F statistics over 1,000 random simulations. We see that, just by chance, the F statistic can be as large as 7.

22.2.3 Observed statistic vs. null statistic

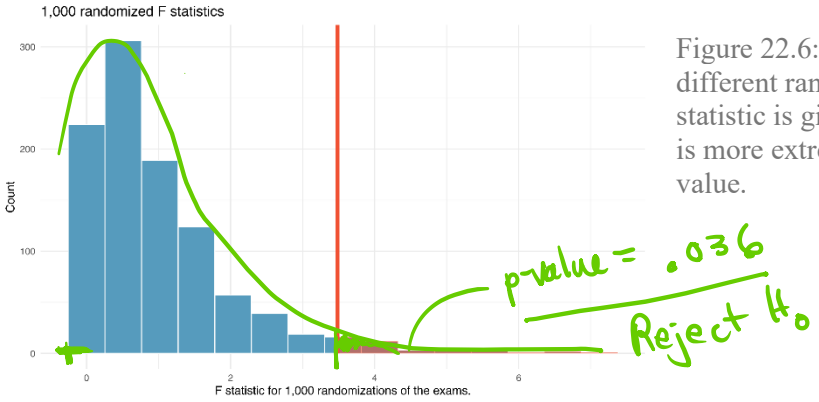


Figure 22.6: Histogram of F statistics calculated from 1000 different randomizations of the exam type. The observed F statistic is given as a red vertical line 3.48. The area to the right is more extreme than the observed value and represents the p-value.

1) $\mu_A = \mu_B$ $\alpha = .05$

2) $\mu_B = \mu_C$ $\alpha = .05$

3) $\mu_A = \mu_C$ $\alpha = .05$

Using statistical software, we can calculate that 3.6% of the randomized F test statistics were at or above the observed test statistic of F=3.48. That is, the p-value of the test is 0.036. Assuming that we had set the level of significance to be α=0.05, the p-value is smaller than the level of significance which would lead us to reject the null hypothesis. We claim that the difficulty level (i.e., the true average score, μ) is different for at least one of the exams.

While it is tempting to say that exam C is harder than the other two (given the inability to differentiate between exam A and exam B in Section 20.1), we must be very careful about conclusions made using different techniques on the same data.

When the null hypothesis is true, random variability that exists in nature produces data with p-values less than 0.05. How often does that happen? 5% of the time. That is to say, if you use 20 different models applied to the same data where there is no signal (i.e., the null hypothesis is true), you are reasonably likely to get a p-value less than 0.05 in one of the tests you run.

The details surrounding the ideas of this problem, called a **multiple comparisons test** or **multiple comparisons problem**, are outside the scope of this textbook, but should be something that you keep in the back of your head. To best mitigate any extra type I errors, we suggest that you set up your hypotheses and testing protocol before running any analyses.

Comparison of 2 Groups : Type 1 error .05 | 5% reject a true null

$5C_2 = 10$ comparisons (pairwise)

ANOVA prevents multiple comparison test problem.

Notation

"y-i-j"

y_{ij} = observation j in group i

Thus, the first observation in the first group is y_{11}, the second observation in the first group is y_{12}, the third observation in the second group is y_{23}, and so on.

We will also use the following notation:

I = number of groups
n_i = number of observations in group i
$\bar{y}_i$ = mean for group i
s_i = standard deviation for group i

The total number of observations is

$$n_\cdot = \sum_{i=1}^{I} n_i$$

"n-dot"

Finally, the **grand mean**—the mean of all the observations—is

$$\bar{\bar{y}} \quad \bar{\bar{y}} = \frac{\sum_{i=1}^{I}\sum_{j=1}^{n_i} y_{ij}}{n_\cdot}$$

Equivalently we can express $\bar{\bar{y}}$ as a weighted average of the group means

$$\bar{\bar{y}} \quad \bar{\bar{y}} = \frac{\sum_{i=1}^{I} n_i \bar{y}_i}{\sum_{i}^{I} n_i} = \frac{\sum_{i=1}^{I} n_i \bar{y}_i}{n_\cdot}$$

$$S_{pooled} = \sqrt{MS(within)} = \sqrt{\frac{SSwithin}{df\ within}}$$, where **df within** = $n_\cdot - I$

$$MSbetween = \frac{SSbetween}{df\ between}$$, where **df between** = $I - 1$

** Note **

Here, you will see the following notation and definitions we will NOT calculate by hand:

- Grand mean $\bar{\bar{y}}$
- SS(within) – "sum of squares **within groups"** (this is the numerator of the pooled variance)
- MS(within) – "mean square **within groups"** Also known as **MSE,** or Mean Square Error (this also means the pooled variance) The **MSE** can ***quantify the training error*** MSW = MSE
- SS(between) – "sum of squares **between groups"**
- MS(between) – "mean square **between groups**"
- SS(total) – total sum of squares

The Fundamental Relationship of ANOVA

$(y_{ij} - \bar{\bar{y}}) = (y_{ij} - \bar{y}_i) + (\bar{y}_i - \bar{\bar{y}})$

grand

"within" difference

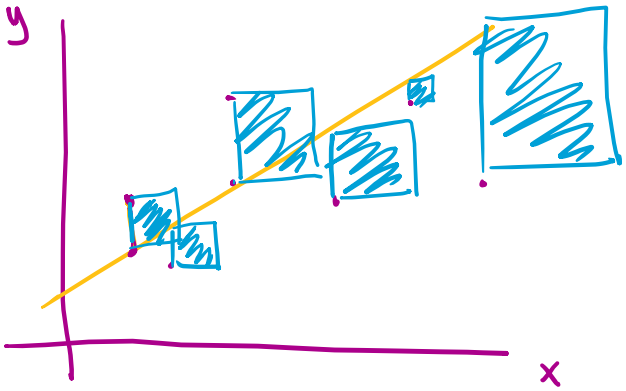
"between" difference

This expresses the deviation of an observation from the grand mean as the sum of two parts: the "within group" deviation $(y_{ij} - \bar{y}_i)$ and the "between group" deviation $(\bar{y}_i - \bar{\bar{y}})$.

This means that: **SS(total) = SS(within) + SS(between)**

ANOVA Table

ANOVA Quantities with Formulas			
Source	**df**	**SS(Sum of squares)**	**MS(Mean Square)**
Between groups	$I - 1$ = 5-1	$\sum_{n=1}^{I} n_i(\bar{y}_i - \bar{\bar{y}})^2$	SS/df
Within groups	$n_\bullet - I$ = 60-5	$\sum_{i=1}^{I} (y_{ij} - \bar{y})^2$	SS/df
Total	$n_\bullet - 1$ = 60-1	$\sum_{i=1}^{I}\sum_{j=1}^{n_i} (y_{ij} - \bar{\bar{y}})^2$	



"predicted error"
Residual = vertical distance
= observed value − predicted

Sum of Squares Residuals

22.3.3 Reading an ANOVA table from software

The calculations required to perform an ANOVA by hand are tedious and prone to human error. For these reasons, it is common to use statistical software to calculate the F-statistic and p-value.

An ANOVA can be summarized in a table very similar to that of a regression summary, which we saw in Chapters 7 and 8. The table below shows an ANOVA summary to test whether the mean of on-base percentage varies by player positions in the MLB. Many of these values should look familiar; in particular, the F-statistic and p-value can be retrieved from the last two columns.

term	df	sumsq	meansq	statistic	p.value
position	2	0.0161	0.0080	5.08	0.0066
Residuals	426	0.6740	0.0016		

5.08

One statistic not presented on the ANOVA table that might be of interest is the percentage of the variability is in the position explained by the factor levels of the categorical variable. We can find this as the ratio of the sum of squares for position divided by the total sum of squares.

```
0.02332995
```

Percentage of explained variability = 2.33 %

In this case, 2.33% of the variability in position is explained by factor levels of the categorical variable. This is the same as the value we would obtain if instead performed a linear regression.

Chapter 22 interactive notes

R Saidi

Load the libraries and data

Load this data from the class

```
library(tidyverse)
```

```
── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
✔ dplyr     1.1.4     ✔ readr     2.1.5
✔ forcats   1.0.0     ✔ stringr   1.5.1
✔ ggplot2   3.5.1     ✔ tibble    3.2.1
✔ lubridate 1.9.3     ✔ tidyr     1.3.1
✔ purrr     1.0.2
── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
✖ dplyr::filter() masks stats::filter()
✖ dplyr::lag()    masks stats::lag()
ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to
become errors
```

```
library(tidymodels)
```

```
── Attaching packages ─────────────────────────────────────── tidymodels 1.2.0 ──
✔ broom        1.0.6      ✔ rsample      1.2.1
```

```
✔ dials        1.2.1      ✔ tune         1.2.1
✔ infer        1.0.7      ✔ workflows    1.1.4
✔ modeldata    1.3.0      ✔ workflowsets 1.1.0
✔ parsnip      1.2.1      ✔ yardstick    1.3.1
✔ recipes      1.0.10
── Conflicts ──────────────────────────────────────── tidymodels_conflicts() ──
✖ scales::discard() masks purrr::discard()
✖ dplyr::filter()   masks stats::filter()
✖ recipes::fixed()  masks stringr::fixed()
✖ dplyr::lag()      masks stats::lag()
✖ yardstick::spec() masks readr::spec()
✖ recipes::step()   masks stats::step()
• Use suppressPackageStartupMessages() to eliminate package startup messages
```

```
library(openintro)
```

```
Loading required package: airports
Loading required package: cherryblossom
Loading required package: usdata

Attaching package: 'openintro'

The following object is masked from 'package:modeldata':

    ames
```

```
data("nycflights")
head(nycflights)
```

```
# A tibble: 6 × 16
   year month   day dep_time dep_delay arr_time arr_delay carrier tailnum flight
  <int> <int> <int>    <int>     <dbl>    <int>     <dbl> <chr>   <chr>    <int>
1  2013     6    30      940        15     1216        -4 VX      N626VA     407
2  2013     5     7     1657        -3     2104        10 DL      N3760C     329
3  2013    12     8      859        -1     1238        11 DL      N712TW     422
4  2013     5    14     1841        -4     2122       -34 DL      N914DL    2391
5  2013     7    21     1102        -3     1230        -8 9E      N823AY    3652
6  2013     1     1     1817        -3     2008         3 AA      N3AXAA     353
# ⓘ 6 more variables: origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
#   hour <dbl>, minute <dbl>
```

The nycflights dataset has 32735 observations with 16 variables, including flight origin (origin) and arrival delay (arr_delay)

What are the origin airports?

```
unique(nycflights$origin)
```

```
[1] "JFK" "LGA" "EWR"
```

“JFK” (Kennedy Airport) “LGA” (Laguardia Airport) “EWR” (Newark Airport)

Before performing ANOVA, check conditions

Constant variance?

```
summarytable <- nycflights |>
  # Group by origin
  group_by(origin) |>
  # Calculate the std dev of arrival delay as std_delay
  #summarise(sd_delay = sd(arr_delay))
  summarise(mean_delay = mean(arr_delay), sd_delay = sd(arr_delay), median_delay = median
(arr_delay), count = n())
summarytable
```

```
# A tibble: 3 × 5
  origin mean_delay sd_delay median_delay count
  <chr>       <dbl>    <dbl>        <dbl> <int>
1 EWR          9.33     46.1           -3 11771
2 JFK          5.98     44.5           -5 10897
3 LGA          5.71     43.1           -5 10067
```

The standard deviations do not appear too different.

Generate a sample from the "population" of flights in 2013

use this for all other calculations

```
set.seed(9357)
samp <- sample_n(nycflights, 200, replace = TRUE)
head(samp)
```

```
# A tibble: 6 × 16
   year month   day dep_time dep_delay arr_time arr_delay carrier tailnum flight
  <int> <int> <int>    <int>     <dbl>    <int>     <dbl> <chr>   <chr>    <int>
1  2013     2    17     2001         1     2259        -8 UA      N507UA     961
2  2013     9     8     1758        -2     2107       -11 DL      N343NB    2139
3  2013     5    16     1440        10     1633         6 EV      N709EV    4975
4  2013     7    27      611        -4      844       -21 WN      N465WN      64
5  2013     3    11      754        -1     1011       -15 UA      N564UA     495
6  2013    12     8     1152        -8     1526        16 AA      N338AA       3
# ℹ 6 more variables: origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
#   hour <dbl>, minute <dbl>
```

Is there a difference in mean arrival delay among the 3 NYC airports?

Ho: There is no difference in mean arrival delay among the three NYC airports.

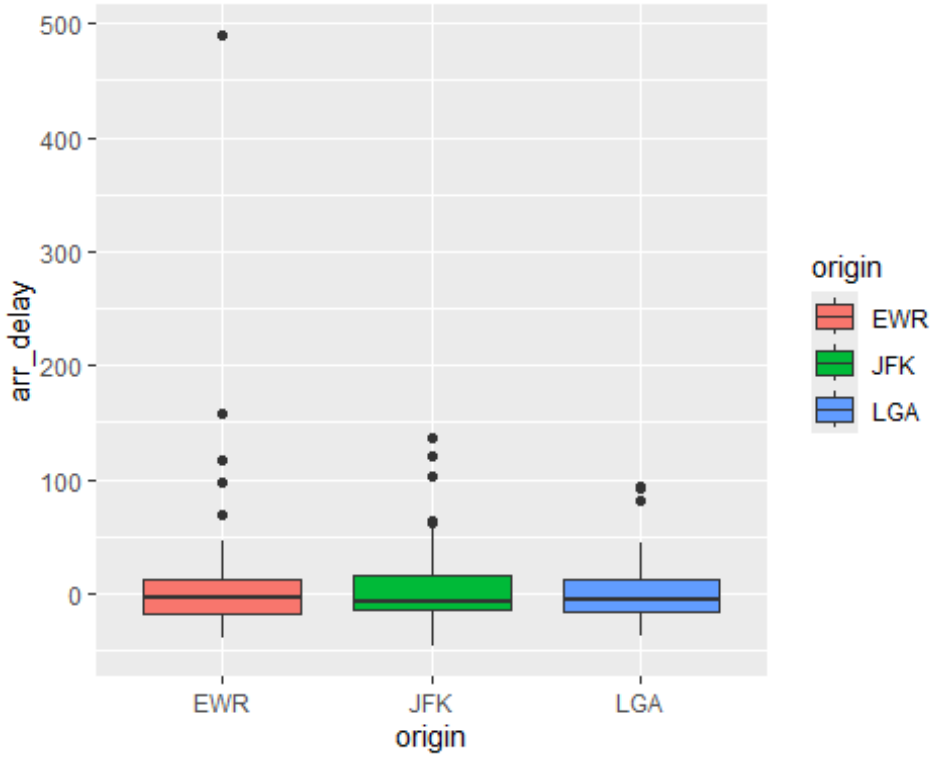
Ha: At least one airport's mean arrival delay is different.

```
summarytable <- samp |>
  # Group by origin
  group_by(origin) |>
  # Calculate the std dev of arrival delay as std_delay
  #summarise(sd_delay = sd(arr_delay))
  summarise(mean_delay = mean(arr_delay), sd_delay = sd(arr_delay), median_delay = median
(arr_delay), count = n())
summarytable
```

```
# A tibble: 3 × 5
  origin mean_delay sd_delay median_delay count
  <chr>       <dbl>    <dbl>        <dbl> <int>
1 EWR          9.05     63.0         -4      81
2 JFK          4.94     33.8         -7      67
3 LGA          1.88     29.0         -5.5    52
```

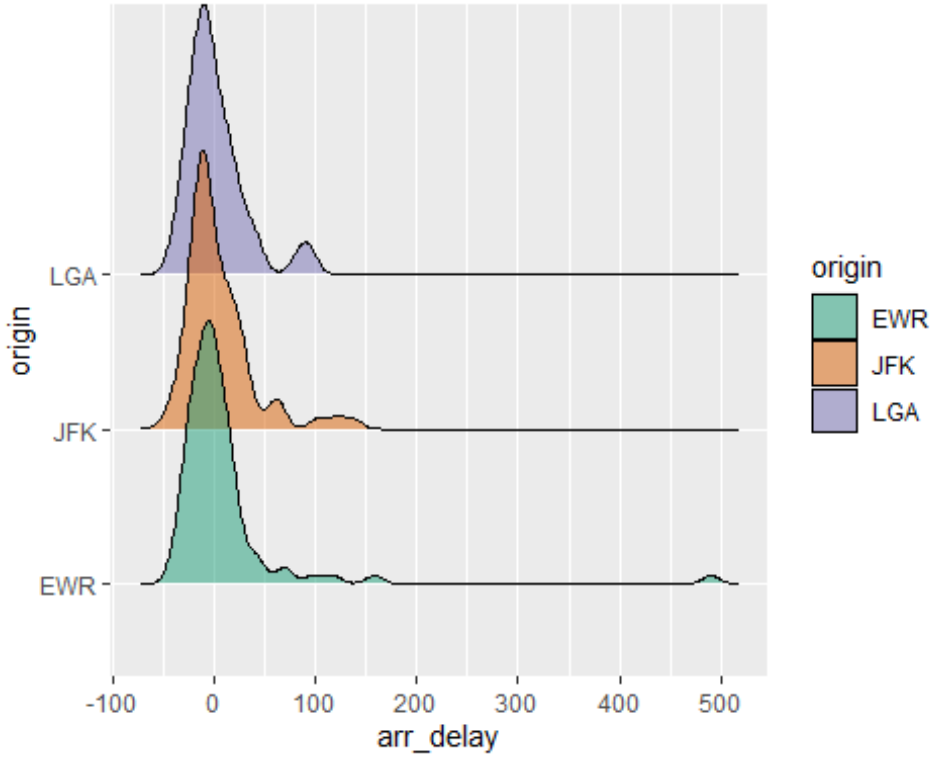
Create boxplots density ridges plot to compare the 3 group distributions

```
ggplot(samp, aes(origin, arr_delay, fill = origin)) +
         geom_boxplot()
```




```
# install the package ggridges
library(ggridges)
samp |>
  # Map wordsum to the x-axis and class to the y-axis
  ggplot(aes(x = arr_delay, y = origin)) +
  # Add density ridges to the plot!
  geom_density_ridges(aes(fill = origin), alpha = 0.5) +
  scale_fill_brewer(palette = "Dark2")

Picking joint bandwidth of 8.6
```



Again, the distributions appear very similar.

```
# Run an analysis of variance on score vs. rank
aov_delay_origin <- aov(arr_delay ~ origin, data = samp) #y~x
#summary(aov_delay_origin)
# Tidy the model
tidy(aov_delay_origin)
```

```
# A tibble: 2 × 6
  term         df   sumsq meansq statistic p.value
  <chr>     <dbl>   <dbl>  <dbl>     <dbl>   <dbl>
1 origin        2   1702.   851.     0.385   0.681
2 Residuals   197 435435.  2210.    NA      NA
```

Interpret the results

BUT from a random sample, the p-value is very large, meaning we fail to reject the null. This suggests that simply having a large sample size will coerce an effect, when there really is no effect.

There is no compelling evidence that there is a difference in mean arrival delay among the different origins.

Simulating samples under the null hypothesis

First calculate the observed statistic

In the case of an ANOVA, the statistic we are interested in is the F-statistic. While we can plot this statistic on an F-distribution, it really is just another statistic that we can calculate (like the mean or median). We like this statistic because it allows for us to summarize how different multiple means are from each other, relative to how variable the observations are within each group.

We can calculate the F-statistic using the tools from the infer package we are familiar with. The only part that is new is the stat that we calculate. Here, we use the “F” statistic.

```
obs_stat <- samp |>
  specify(arr_delay ~ origin) |>  #y~x
  calculate(stat = "F")
obs_stat
```

```
Response: arr_delay (numeric)
Explanatory: origin (factor)
# A tibble: 1 × 1
   stat
  <dbl>
1 0.385
```

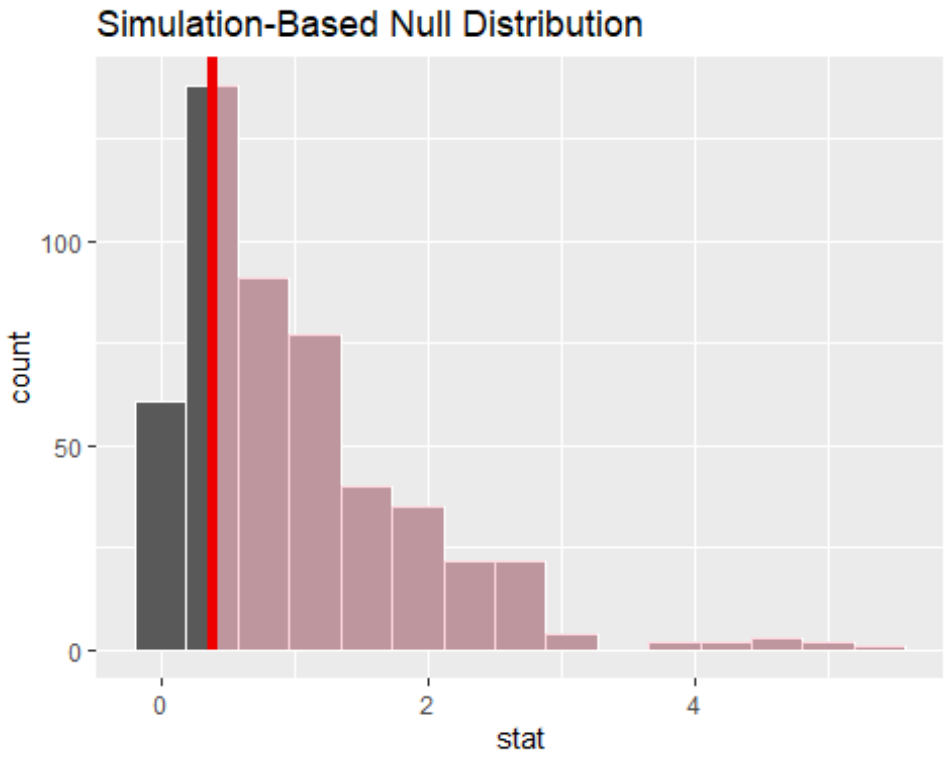
The next step is to simulate what we would expect for arrival delays to look like, if the null hypothesis was true. This is similar to the method for a difference in means, except now we have three groups: jfk, lga, and ewr. The underlying process, however, looks the same:

Step 1: Write the values of arr_delay on 32735 index cards (one card per person). Step 2: Shuffle the cards and randomly split them into three new piles, of the same size as the original groups. Step 3: Calculate and record the test statistic: F-statistic Step 4: Repeat steps (1) and (2) 1000 to generate the sampling distribution of the difference in means under the null hypothesis. Step 5: Calculate p-value as the percentage of simulations where the test statistic is at least as extreme as the observed F-statistic

```
null_distr <- samp |>
 specify(arr_delay ~ origin) |>
  hypothesize(null = "independence") |>
  generate(reps = 500, type = "permute") |>
  calculate(stat = "F")
null_distr
```

```
Response: arr_delay (numeric)
Explanatory: origin (factor)
Null Hypothesis: independence
# A tibble: 500 × 2
   replicate   stat
       <int>  <dbl>
 1         1 1.02
 2         2 0.486
 3         3 0.838
 4         4 0.0450
 5         5 0.649
 6         6 0.551
 7         7 2.25
 8         8 0.264
 9         9 1.26
10        10 0.437
# ℹ 490 more rows
```

```
null_distr |>
  visualise() +
  shade_p_value(obs_stat = obs_stat, direction = "greater")
```



We can see that the observed F-statistic is very close to zero. therefore we fail to reject the null. There is no compelling evidence that the mean arrival delay is different at at least one airport.

Post hoc testing

If you reject the null, now you can determine by pair-wise testing which group's mean is different using a "family" error rate, and then distribute that level to each of the tests we are performing. The "family" error rate specifies an overall Type I error rate we are willing to have for all of tests you wish to perform. We will use alpha = 0.05

There are many pairwise tests. We will use the TukeyHSD test on our original model, aov_delay_origin.

```
TukeyHSD(aov_delay_origin)
```

```
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = arr_delay ~ origin, data = samp)

$origin
             diff       lwr      upr     p adj
JFK-EWR -4.109084 -22.44402 14.22585 0.8570826
LGA-EWR -7.164767 -26.89399 12.56446 0.6676137
LGA-JFK -3.055683 -23.57500 17.46363 0.9341246
```

How to read this output

The p-adjusted values for comparing mean delays for JFK and EWR are not significant. The p-adjusted values for comparing mean delays for LGA and EWR are not significant. The p-adjusted values for comparing mean delays for LGA and JFK are not significant.

Non-parametric test Kruskal Wallis Test

```
# and on the sample
ktest_samp <- kruskal.test(arr_delay ~ origin, data = samp) #syntax is y~x
ktest_samp


    Kruskal-Wallis rank sum test

data:  arr_delay by origin
Kruskal-Wallis chi-squared = 0.20372, df = 2, p-value = 0.9032
```

The conclusion is: With a large p-value, we fail to reject the null. There is no compelling evidence that there is a difference in distributions for arrival delays based on the airport of origin in NY.

Post hoc test for Kruskal Wallis test

If your p-value is small for the ktest (at least one group's distribution is shifted), the post-hoc test is the Dunn Test.

```
library(dunn.test)
dunn.test(samp$arr_delay, samp$origin) # syntax is dunn.test(df$y, df$x)

  Kruskal-Wallis rank sum test

data: x and group
Kruskal-Wallis chi-squared = 0.2037, df = 2, p-value = 0.9

                           Comparison of x by group
                                (No adjustment)
Col Mean-|
Row Mean |        EWR        JFK
---------+----------------------
     JFK |  -0.007597
         |     0.4970
         |
     LGA |   0.406216   0.397363
         |     0.3423     0.3455

alpha = 0.05
Reject Ho if p <= alpha/2
```

Here is an example where we reject the null

Use the fastfood dataset from openintro

```
# from openintro
data(fastfood)
head(fastfood)

# A tibble: 6 × 17
  restaurant item       calories cal_fat total_fat sat_fat trans_fat cholesterol
  <chr>      <chr>         <dbl>   <dbl>     <dbl>   <dbl>     <dbl>       <dbl>
1 Mcdonalds  Artisan G…      380      60         7       2       0            95
2 Mcdonalds  Single Ba…      840     410        45      17       1.5         130
```

```
3 Mcdonalds  Double Ba…     1130     600        67      27       3           220
4 Mcdonalds  Grilled B…      750     280        31      10       0.5         155
5 Mcdonalds  Crispy Ba…      920     410        45      12       0.5         120
6 Mcdonalds  Big Mac          540     250        28      10       1            80
# ⓘ 9 more variables: sodium <dbl>, total_carb <dbl>, fiber <dbl>, sugar <dbl>,
#   protein <dbl>, vit_a <dbl>, vit_c <dbl>, calcium <dbl>, salad <chr>
```

Is there a difference in mean number of calories overall among the fast food restaurants?

Ho: There is no difference in overall mean number of calories among the fast food restaurants.

Ha: At least one restaurant's overall mean number of calories is different among the fast food restaurants.

```
unique(fastfood$restaurant)
```

```
[1] "Mcdonalds"   "Chick Fil-A" "Sonic"       "Arbys"       "Burger King"
[6] "Dairy Queen" "Subway"      "Taco Bell"
```

There are 8 fast food restaurants.

Check the basic conditions. First, create a table of counts.

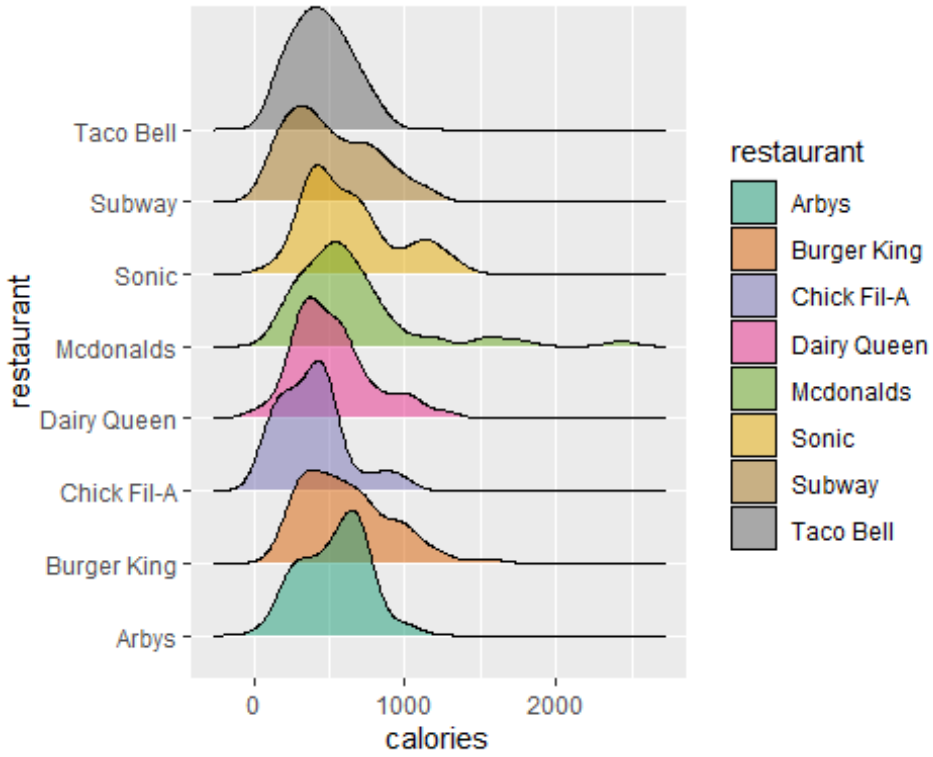
```
fast <- fastfood|>
group_by(restaurant) |>
summarise(mean_calorie = mean(calories), sd_calories = sd(calories), count = n())
fast
```

```
# A tibble: 8 × 4
  restaurant  mean_calorie sd_calories count
  <chr>              <dbl>       <dbl> <int>
1 Arbys               533.        210.    55
2 Burger King         609.        290.    70
3 Chick Fil-A         384.        220.    27
4 Dairy Queen         520.        259.    42
5 Mcdonalds           640.        411.    57
6 Sonic               632.        301.    53
7 Subway              503.        282.    96
8 Taco Bell           444.        184.   115
```

Check the distributions of calories over the restaurants with ridge plots

```
ggplot(fastfood, aes(x=calories, y=restaurant))+
  geom_density_ridges(aes(fill = restaurant), alpha = 0.5) +
  scale_fill_brewer(palette = "Dark2")
```

```
Picking joint bandwidth of 93.8
```



We can see that the all the distributions of calories are right skewed, but they do seem a bit different.

Since the basic conditions may have been met, we can use ANOVA test.

Alternatively, we could use Kruskal Wallis test (below)

```
aov_calories <- aov(calories ~ restaurant, data = fastfood)
tidy(aov_calories)
```

```
# A tibble: 2 × 6
  term          df     sumsq  meansq statistic      p.value
  <chr>      <dbl>     <dbl>   <dbl>     <dbl>        <dbl>
1 restaurant     7  3177729. 453961.      6.08  0.000000775
2 Residuals    507 37824143.  74604.     NA    NA
```

The p value is very small. Reject the null. At least one restaurant's mean overall calories is different.

Perform the TukeyHSD post-hoc test for pairwise comparisons

```
TukeyHSD(aov_calories)
```

```
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = calories ~ restaurant, data = fastfood)

$restaurant
                              diff        lwr        upr     p adj
Burger King-Arbys         75.844156  -73.95075 225.639063 0.7847525
Chick Fil-A-Arbys       -148.282828 -343.63404  47.068382 0.2899281
Dairy Queen-Arbys        -12.489177 -182.84316 157.864802 0.9999986
Mcdonalds-Arbys          107.623604  -49.50767 264.754883 0.4262776
```

```
Sonic-Arbys                98.970840  -61.04575 258.987428 0.5636313
Subway-Arbys              -29.706439 -170.29304 110.880158 0.9982372
Taco Bell-Arbys           -89.075099 -225.36592  47.215720 0.4901923
Chick Fil-A-Burger King -224.126984 -412.46039 -35.793574 0.0077024
Dairy Queen-Burger King  -88.333333 -250.59194  73.925277 0.7150354
Mcdonalds-Burger King     31.779449 -116.53649 180.095385 0.9980728
Sonic-Burger King         23.126685 -128.24269 174.496056 0.9997899
Subway-Burger King      -105.550595 -236.21034  25.109152 0.2158498
Taco Bell-Burger King   -164.919255 -290.94531 -38.893203 0.0019839
Dairy Queen-Chick Fil-A  135.793651  -69.27082 340.858123 0.4723572
Mcdonalds-Chick Fil-A    255.906433   61.68697 450.125891 0.0017798
Sonic-Chick Fil-A        247.253669   50.69256 443.814773 0.0036044
Subway-Chick Fil-A       118.576389  -62.51903 299.671807 0.4876909
Taco Bell-Chick Fil-A     59.207729 -118.57345 236.988906 0.9723288
Mcdonalds-Dairy Queen    120.112782  -48.94218 289.167747 0.3764793
Sonic-Dairy Queen        111.460018  -60.28005 283.200085 0.4996730
Subway-Dairy Queen       -17.217262 -171.01562 136.581092 0.9999744
Taco Bell-Dairy Queen    -76.585921 -226.46764  73.295798 0.7766704
Sonic-Mcdonalds           -8.652764 -167.28571 149.980181 0.9999998
Subway-Mcdonalds        -137.330044 -276.33973   1.679647 0.0555520
Taco Bell-Mcdonalds     -196.698703 -331.36232 -62.035084 0.0002868
Subway-Sonic            -128.677280 -270.94029  13.585729 0.1095728
Taco Bell-Sonic         -188.045939 -326.06536 -50.026521 0.0010247
Taco Bell-Subway         -59.368659 -174.29752  55.560200 0.7667861
```

*Look for each of the pairwise comparisons that have small p-adj values. Those are the pairs with meaningfully different overall calories.

```
# Chick Fil-A-Burger King -224.126984 -412.46039 -35.793574 0.0077024
# Taco Bell-Burger King   -164.919255 -290.94531 -38.893203 0.0019839
# Mcdonalds-Chick Fil-A    255.906433   61.68697 450.125891 0.0017798
# Sonic-Chick Fil-A        247.253669   50.69256 443.814773 0.0036044
# Subway-Mcdonalds        -137.330044 -276.33973   1.679647 0.0555520
# Taco Bell-Mcdonalds     -196.698703 -331.36232 -62.035084 0.0002868
# Taco Bell-Sonic         -188.045939 -326.06536 -50.026521 0.0010247
```

Perform the same test using the non-parametric approach

```
ktest <- kruskal.test(calories ~ restaurant, data =fastfood)
ktest
```

```
    Kruskal-Wallis rank sum test

data:  calories by restaurant
Kruskal-Wallis chi-squared = 33.829, df = 7, p-value = 1.854e-05
```

Note the small pvalue. Reject the null. At least one restaurant's distribution of calories is different.

```
dunn.test(fastfood$calories, fastfood$restaurant)
```

```
  Kruskal-Wallis rank sum test

data: x and group
Kruskal-Wallis chi-squared = 33.8294, df = 7, p-value = 0
```

```
                        Comparison of x by group
                             (No adjustment)
Col Mean-|
Row Mean |      Arbys   Burger K   Chick Fi   Dairy Qu   Mcdonald      Sonic
---------+------------------------------------------------------------------
Burger K |  -0.843075
         |     0.1996
         |
Chick Fi |   2.873237   3.650859
         |    0.0020*    0.0001*
         |
Dairy Qu |   0.693987   1.506927  -2.160622
         |     0.2438     0.0659    0.0154*
         |
Mcdonald |  -0.707719   0.101698  -3.462553  -1.357122
         |     0.2396     0.4595    0.0003*     0.0874
         |
   Sonic |  -1.170625  -0.403192  -3.808534  -1.779101  -0.479815
         |     0.1209     0.3434    0.0001*     0.0376     0.3157
         |
  Subway |   1.319416   2.386202  -2.075139   0.437382   2.134362   2.620580
         |     0.0935    0.0085*    0.0190*     0.3309    0.0164*    0.0044*
         |
Taco Bel |   2.391306   3.588159  -1.323971   1.385690   3.245999   3.718553
         |    0.0084*    0.0002*     0.0928     0.0829    0.0006*    0.0001*
Col Mean-|
Row Mean |     Subway
---------+-----------
Taco Bel |   1.221807
         |     0.1109

alpha = 0.05
Reject Ho if p <= alpha/2
```

Homework Chapter 22

1. Review section 22.4 (the chapter review)
2. Suggested problems from textbook section 22.5 exercises: 4-7, 9, 14
3. Suggested tutorials:

    8 - Comparing many means