# Lesson 22 Companion

**Research question:**

I'm going to use the "ambigious prose" example for this document. The research question in this example asks: is there an association between when a participant is shown a picture (cue condition) and their long-run recall score. To measure this researchers split their participants into three groups: one saw a picture before hearing the passage, one saw the picture after being read the passage, and the last didn't get to see the picture at all. A few hours later the participants were tested on their recall and that score was recorded in the data set. The hypothesis for our statistical test are as follows:

$H_0$ : There is no association between cue condition and recall score.
$H_a$ : There is an association between cue condition and recall score.

The parameters of interest we will use is long-run mean recall score ($\mu_{condition}$), the statistics will be the sample mean recall score ($\bar{x}_{condition}$), and the hypothesis can be rewritten as:

$H_0 : \mu_{before} = \mu_{after} = \mu_{none}$
$H_a$: At least one $\mu$ is different.

Here again, the chapter discusses two test statistics to utilize for analysis. We will concentrate on the $F$-statistic for this example. In the code below I'm using a concept that I've haven't used before in this class: defining my own function. You, obviously, could do all these calculations by hand (or use $R$'s built in function) but YOLO!

```
library(tidyverse)

recall = read_table2("http://www.isi-stats.com/isi/data/chap9/Recall.txt")

head(recall)
```

```
## # A tibble: 6 x 2
##   Condition Recall
##   <chr>      <dbl>
## 1 After          7
## 2 After          5
## 3 After          5
## 4 After          5
## 5 After          2
## 6 After          8
```

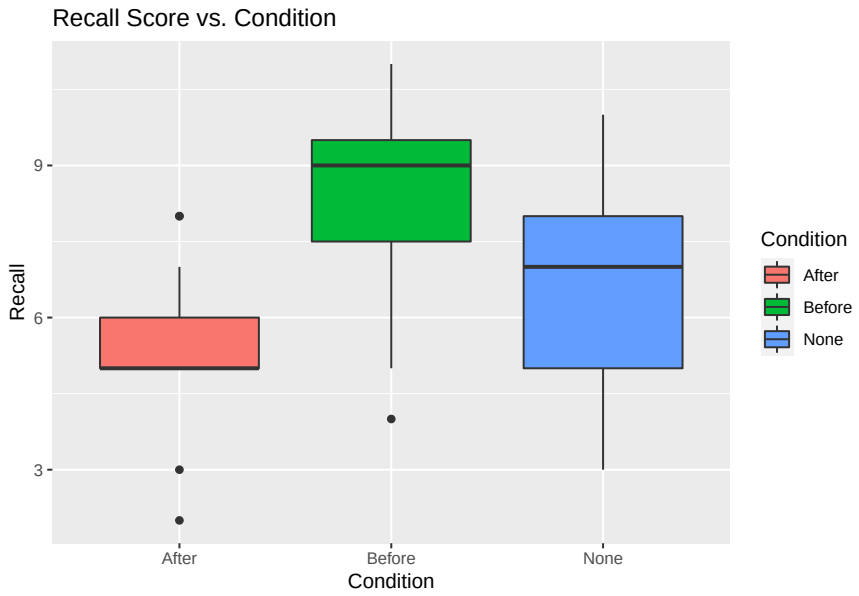
```
#Define "between group variability" function
b_g_v <- function(g_s, o_s, k){
  num = 0
  for (i in 1:k){
    num = num + (g_s[[4]][[i]] * (g_s[[2]][[i]] - o_s[[1]][[1]])^2)
  }
  denom = k - 1
  return(num / denom)
}

#Define "within group variability" function
```

```r
w_g_v <- function(g_s, o_s, k, s_s){
  num = 0
  for (i in 1:k){
    num = num + ((g_s[[4]][[i]] - 1) * (g_s[[3]][[i]]^2))
  }
  denom = s_s - k
  return(num / denom)
}

#Calculate the stats needed to calculate the sample F-stat
group_stats = recall %>%
  group_by(Condition) %>%
  summarise(mean = mean(Recall),
            sd = sd(Recall),
            n = n())

overall_stats = recall %>%
    summarise(mean = mean(Recall),
            var = var(Recall))

sample_size = nrow(recall)

# "k" is the number of groups being compared
k = nrow(group_stats)

#Call our functions with the statistics we calculated as arguments
btwn_grp_var = b_g_v(group_stats, overall_stats, k)
with_grp_var = w_g_v(group_stats, overall_stats, k, sample_size)

sample_f_stat = btwn_grp_var / with_grp_var
```

**Data exploration:**

I think the logical choice to explore this data before analysis is a set of boxplots.

```r
recall %>%
  ggplot(aes(x = Condition, y = Recall, fill = Condition)) +
  geom_boxplot() +
  labs(title = "Recall Score vs. Condition")
```



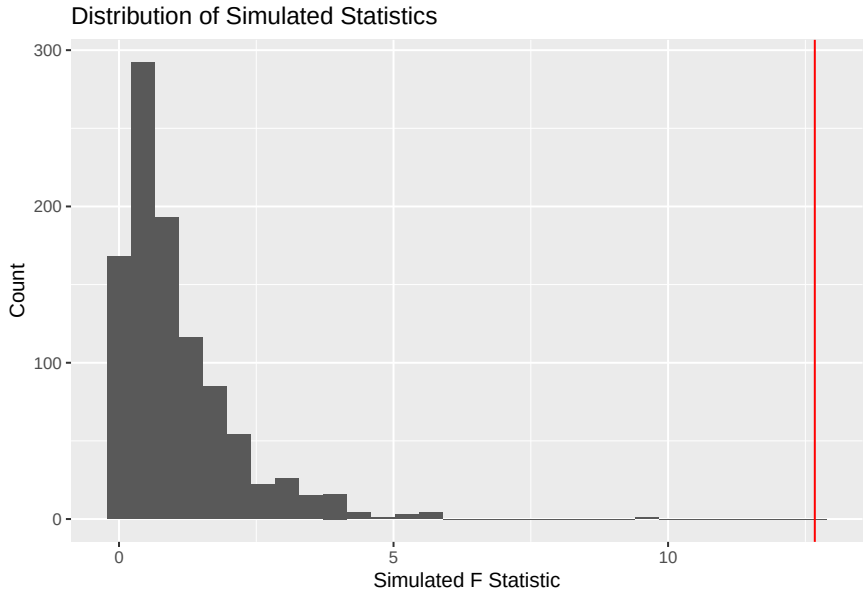

It appears from this boxplot that the *Before* condition has the highest recall scores of all the conditions with *None* followed by *After*. This is a bit different than I would expected as I would have imagined that some picture was better than no picture at all. The *None* condition has the highest variance in scores with a range from 8 to 5 points.

**Simulation-based approach:**

To build the null distribution here we are doing to reassign conditions to recall scores. This is a lot less involved than the simulations we've been doing in recent lessons.

```
replications_dataframe = NULL

num_reps = 1000

for (i in 1:num_reps){

  trial_recall = recall %>%
    mutate(New_Condition = sample(Condition, size = n(), replace = FALSE))

  group_stats = trial_recall %>%
    group_by(New_Condition) %>%
    summarise(mean = mean(Recall),
              sd = sd(Recall),
              n = n())

  overall_stats = trial_recall %>%
    summarise(mean = mean(Recall),
              var = var(Recall))

  sample_size = nrow(trial_recall)

  k = nrow(group_stats)

  btwn_grp_var = b_g_v(group_stats, overall_stats, k)
  with_grp_var = w_g_v(group_stats, overall_stats, k, sample_size)
```

```
  trial_f_stat = btwn_grp_var / with_grp_var

  replications_dataframe = rbind(replications_dataframe, as.data.frame(trial_f_stat))

}

replications_dataframe %>%
  ggplot(aes(x = trial_f_stat)) +
  geom_histogram() +
  labs(x = "Simulated F Statistic", y = "Count",
       title = "Distribution of Simulated Statistics") +
  geom_vline(xintercept = sample_f_stat, color = "red")
```




```
replications_dataframe %>%
  summarise(pvalue = sum(trial_f_stat >= sample_f_stat) / n())
```

```
##   pvalue
## 1      0
```

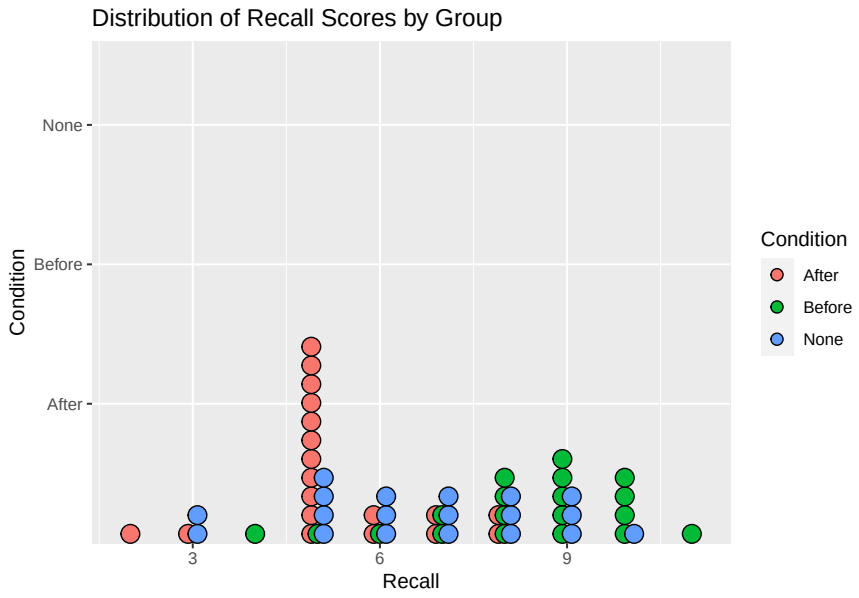
A p-value of 0 indicates we have very strong evidence against the null which suggests that the long-run average recall score different for at least one condition.

**Theory-based approach:**

The validity conditions for the theory-based approach as as follows: * Either the sample size is at least 20 for all groups, without strong skewness or outliers **OR** the distribution of the quantitative variable is approximately symmetric in all the samples. **AND** * The standard deviations of the samples are approximately equal to each other (the largest shouldn't be more than twice the smallest).

To check these conditions, we need to build a dotplot for each group.

```
recall %>%
  ggplot(aes(x = Recall, y = Condition, fill = Condition)) +
  geom_dotplot(position = "dodge") +
  labs(title = "Distribution of Recall Scores by Group")
```



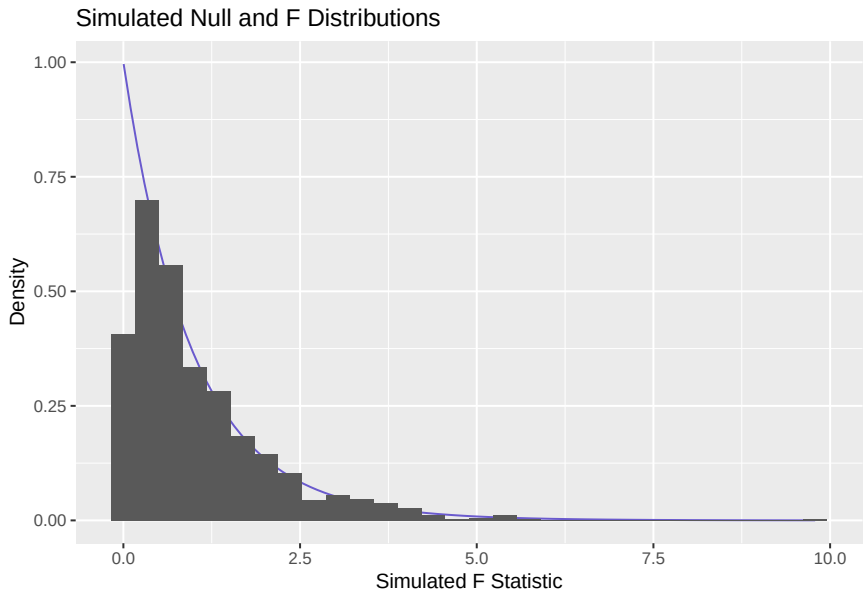

As we don't fit the requirement for having a sample size of at least 20, we must ensure that our quantitative variable (*Recall*) is approximately symmetric in all samples. It appears that we meet that condition for these three groups. We can ensure we meet the second validity condition by calculating the standard deviation for each group.

```
recall %>%
  group_by(Condition) %>%
  summarise(SD = sd(Recall))
```

```
## # A tibble: 3 x 2
##   Condition    SD
##   <chr>     <dbl>
## 1 After      1.46
## 2 Before     1.82
## 3 None       2.01
```

Now that we have met the validity conditions, let's see how well the F-distribution fits our simular null distribution. While not strictly a necessary step, I include it here for instructional purposes.

```
df_1 = k - 1
df_2 = sample_size - k

replications_dataframe %>%
  ggplot(aes(x = trial_f_stat)) +
  stat_function(fun = df,
                args = list(df1 = df_1, df2 = df_2),
                color = "slateblue") +
  geom_histogram(aes(y = ..density..)) +
  labs(x = "Simulated F Statistic", y = "Density",
       title = "Simulated Null and F Distributions")
```




Now I will demonstrate two methods of finding a p-value using the theory-based approach.

```
#Using the PDF of the F-distribution
1 - pf(sample_f_stat, df1 = df_1, df2 = df_2)
```

```
## [1] 3.073869e-05
```

Of course, *R* also has a built in function to compute this p-value but before we get to that, we need to discuss ANOVA. The ANOVA (or *Analysis of Variance* test) is really another name for the theory-based approach for these problems. Any statistical package that deserves to be called a statistical package has a built-in ANOVA function. *R* has both an **anova()** and **aov()** function so ensure that you are using the correct one: **aov()**.

```
#Using the built-in function
anova_model = aov(recall$Recall ~ recall$Condition)

summary(anova_model)
```

```
##                   Df Sum Sq Mean Sq F value   Pr(>F)
## recall$Condition  2  80.04   40.02   12.67 3.07e-05 ***
## Residuals        54 170.53    3.16
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

**Further analysis:**

Remember that the ANOVA test is a "overall test" that tells that at least one of these means is different. If we want to figure out which one, we should look at doing pairwise comparisons using a two-sample t-test.

```
before = recall %>%
  filter(Condition == "Before")

after = recall %>%
  filter(Condition == "After")

none = recall %>%
  filter(Condition == "None")
```

```r
t.test(x = before$Recall, y = after$Recall)
```

```
##
##  Welch Two Sample t-test
##
## data:  before$Recall and after$Recall
## t = 5.4048, df = 34.385, p-value = 4.954e-06
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  1.806734 3.982740
## sample estimates:
## mean of x mean of y
##  8.263158  5.368421
```

```r
t.test(x = before$Recall, y = none$Recall)
```

```
##
##  Welch Two Sample t-test
##
## data:  before$Recall and none$Recall
## t = 2.6252, df = 35.668, p-value = 0.01267
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.3706945 2.8924634
## sample estimates:
## mean of x mean of y
##  8.263158  6.631579
```

```r
t.test(x = none$Recall, y = after$Recall)
```

```
##
##  Welch Two Sample t-test
##
## data:  none$Recall and after$Recall
## t = 2.2188, df = 32.904, p-value = 0.03351
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.1047849 2.4215309
## sample estimates:
## mean of x mean of y
##  6.631579  5.368421
```

Just ensure that you remember that order matters in regards to which group is $x$ and which is $y$ when it comes to interpreting the results of these tests.

**What about these other numbers in my ANOVA output?:**

I encourage you to check out my "ANOVA Exploration" document on GitHub for a little more information about ANOVA and it's connection to tests we've covered previously.