PSYC 7710 Lab

Lab 2 Activity

Andrew Graves, Department of Psychology, University of Virginia

Directions:

A. Load the speed_acc dataset from the rtdists package in R. Call ?speed_acc in the R console to get more information on the data set.
B. Answer the questions that follow as they pertain to the speed_acc dataset, and save the code you used in an R script.
C. You have until the end of lab to complete.
D. Set the seed at 42 before **EACH** random draw.
E. Ask me programming questions as needed!

Questions:

1. What is your hypothesis about the **rt** variable under the speed vs. accuracy **condition**?

```
# RTs will be faster under the speed condition relative to the accuracy condition.
```

2. Create a data frame named *my_data* that only includes the **id**, **condition**, and **rt** columns, with only the rows from the first four participants and trials not labelled as outliers **OR** errors (see ?speed_acc for outlier and error labels). Print the dimensions of *my_data*.

```
library(tidyverse)
library(rtdists)
data(speed_acc)

id_vec <- c("1", "2", "3", "4") # Create a vector of IDs to filter

my_data <- speed_acc %>%
  filter(id %in% id_vec, !censor, response != "error") %>% # Remove id rows outside of
  # id_vec, remove censor rows, and remove error rows
  select(id, condition, rt) # Remove all columns except for id, condition, and rt
dim(my_data) # dim provides dimensions for the data
```

```
## [1] 6551    3
```

3. Compute the 95% confidence interval around the mean **rt** using equations from class for **id** 1, 2, and 3 separately.

```
 # Remove id 4 from data set
id_vec_small <- id_vec[-4]

# Get a list of id data that just includes rt as a vector
id_rt_data_list <- map(.x = id_vec_small, ~
  filter(my_data, id == .x) %>%
  select(rt) %>%
  as_vector() %>%
  unname())

# Equation-based confidence interval function
get_conf_int <- function(x){
```

```
  se <- sqrt(sum((x - mean(x))^2) / (length(x) - 1)) / sqrt(length(x))

  lower <- mean(x) + qnorm(.025) * se
  upper <- mean(x) + qnorm(.975) * se
  ci_95 <- data.frame(lower, mean_rt = mean(x), upper)

}

# Map (apply) the confidence interval function to my list of data
ans_3 <- map_dfr(id_rt_data_list, get_conf_int) %>%
  bind_cols("id" = id_vec_small) %>%
  as_tibble()
ans_3
```

```
## # A tibble: 3 x 4
##   lower mean_rt upper id
##   <dbl>   <dbl> <dbl> <chr>
## 1 0.557   0.565 0.573 1
## 2 0.680   0.698 0.716 2
## 3 0.645   0.656 0.667 3
```

4. Compute the 95% confidence interval around the mean **rt** using a custom bootstrap resampling function for **id** 1, 2, and 3 separately. Use 10,000 iterations.

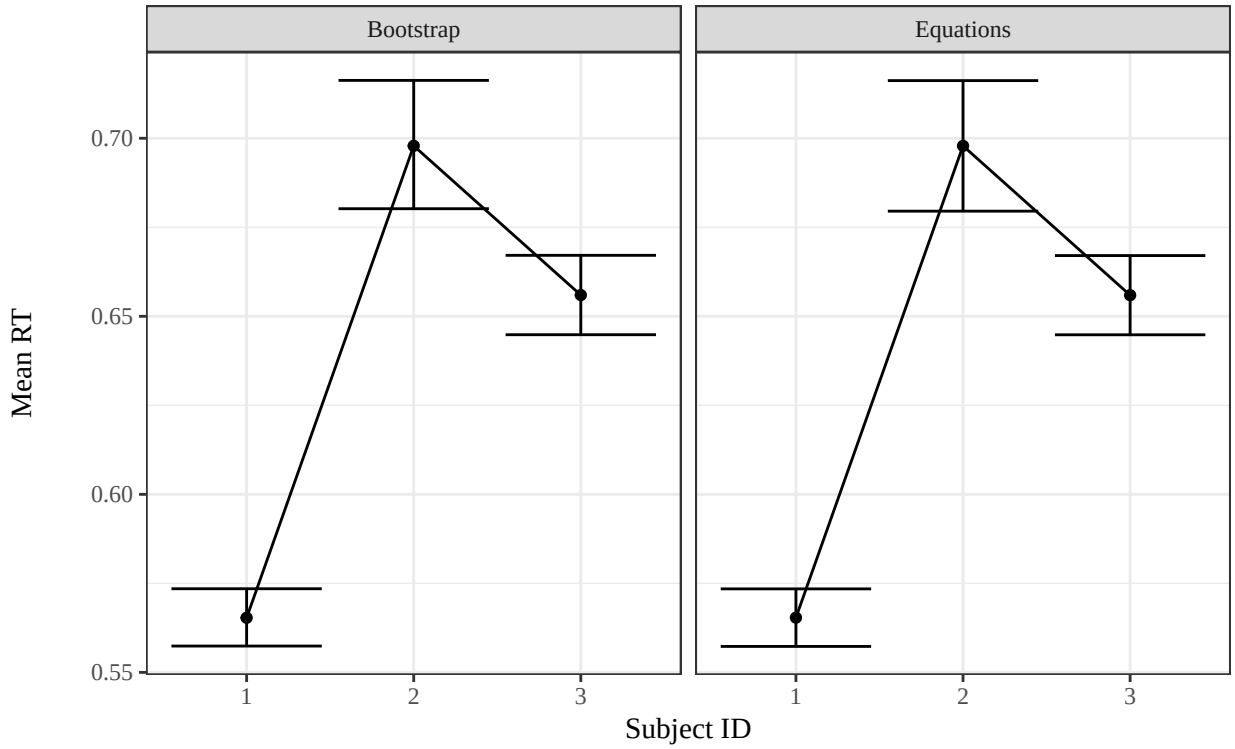
```
# Bootstrap function that requires a vector and function to generate a
# bootstrap distribution
bootstrap <- function(x, func, iter = 10000){
  set.seed(42)
  boot <- rep(NA, iter)

  for (i in 1:iter){
  boot[i] <- sample(x, replace = TRUE) %>%
    func()
  }

  data <- data.frame(samp = boot,
                     mean = mean(boot),
                     lower = unname(quantile(boot, .025)),
                     upper = unname(quantile(boot, .975)))
}

# Map the bootstrap function and compute the mean with my list of data
boot <- map(id_rt_data_list, bootstrap, func = mean)

# Flatten the results into a dataframe
ans_4 <- do.call(rbind, boot) %>%
  select(lower, mean, upper) %>%
  rename(mean_rt = mean) %>%
  unique() %>%
  as_tibble() %>%
  bind_cols("id" = id_vec_small)
ans_4
```

```
## # A tibble: 3 x 4
##   lower mean_rt upper id
```

```
##   <dbl>   <dbl> <dbl> <chr>
## 1 0.557   0.565 0.573 1
## 2 0.680   0.698 0.716 2
## 3 0.645   0.656 0.667 3
```

5. Generate a line plot with the parameter estimates and confidence intervals as error bars from question 3. Do the same for results from question 4. Align these two plots on the same row.

```r
# Modify ggplot themes
theme_set(theme_bw())
theme_update(text = element_text(family = "serif"),
axis.title.y = element_text(margin = margin(r = 20)))

# Creates a new column called method for plotting in the facet labels
plot_data <- ans_3 %>%
  bind_rows(ans_4, .id = "method") %>%
  mutate(method = recode(method, `1` = "Equations", `2` = "Bootstrap"))

plot_data %>%
  ggplot(aes(x = id, y = mean_rt)) +
  geom_point() +
  geom_line(aes(group = 1)) +
  geom_errorbar(aes(ymin = lower, ymax = upper)) +
  facet_grid(~method) +
  ylab("Mean RT") +
  xlab("Subject ID")
```

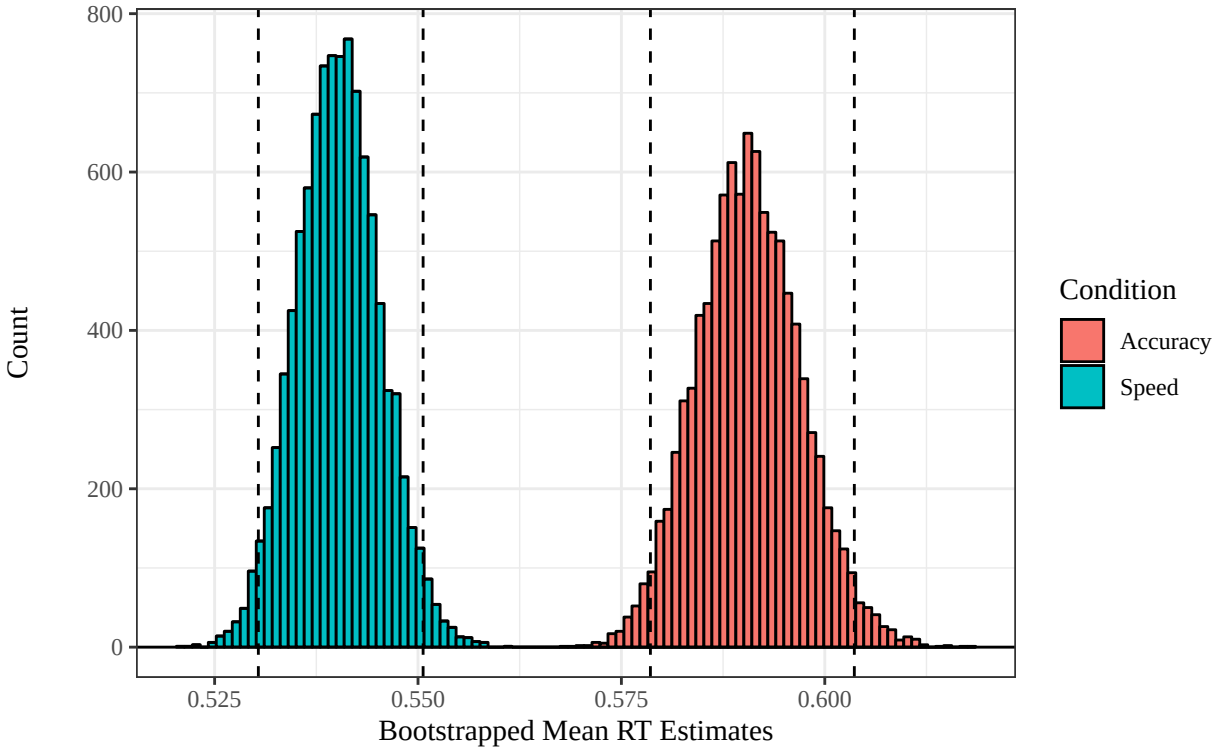


6. Do the equation based confidence intervals and bootstrapped confidence intervals look substantially different or do they overlap considerably?

```
# They overlap considerably.
```

7. Overlay two histograms from **id** 1's data. One histogram should display a bootstrap distribution of mean **rt** under the speed condition, and the other histogram should display a bootstrap distribution of mean **rt** under the accuracy condition. Plot vertical lines at the 2.5% and 97.5% quantiles for both histograms.

```
# Plotting function to avoid redundancy
speed_acc_id_hist <- function(data, id_input){

acc_data <- my_data %>%
  filter(id == id_input, condition == "accuracy")
speed_data <- my_data %>%
  filter(id == id_input, condition == "speed")

acc_boot <- bootstrap(acc_data$rt, func = mean)
speed_boot <- bootstrap(speed_data$rt, func = mean)

# Creates a new column called Condition for plotting in the legends
plot_id <- acc_boot %>%
  bind_rows(speed_boot, .id = "Condition") %>%
  mutate(Condition = recode(Condition, `1` = "Accuracy",`2` = "Speed"))

plot_id %>%
  ggplot(aes(x = samp, fill = Condition)) +
  geom_histogram(bins = 100, color = "black") +
  geom_vline(aes(xintercept = quantile(acc_boot$samp, .025)),
             lty = 2) +
  geom_vline(aes(xintercept = quantile(acc_boot$samp, .975)),
             lty = 2) +
  geom_vline(aes(xintercept = quantile(speed_boot$samp, .025)),
             lty = 2) +
  geom_vline(aes(xintercept = quantile(speed_boot$samp, .975)),
             lty = 2) +
  geom_hline(aes(yintercept = 0), color = "black") +
  xlab("Bootstrapped Mean RT Estimates") +
  ylab("Count")

}

# Apply plotting function to ID 1 data.
speed_acc_id_hist(my_data, id_input = "1")
```

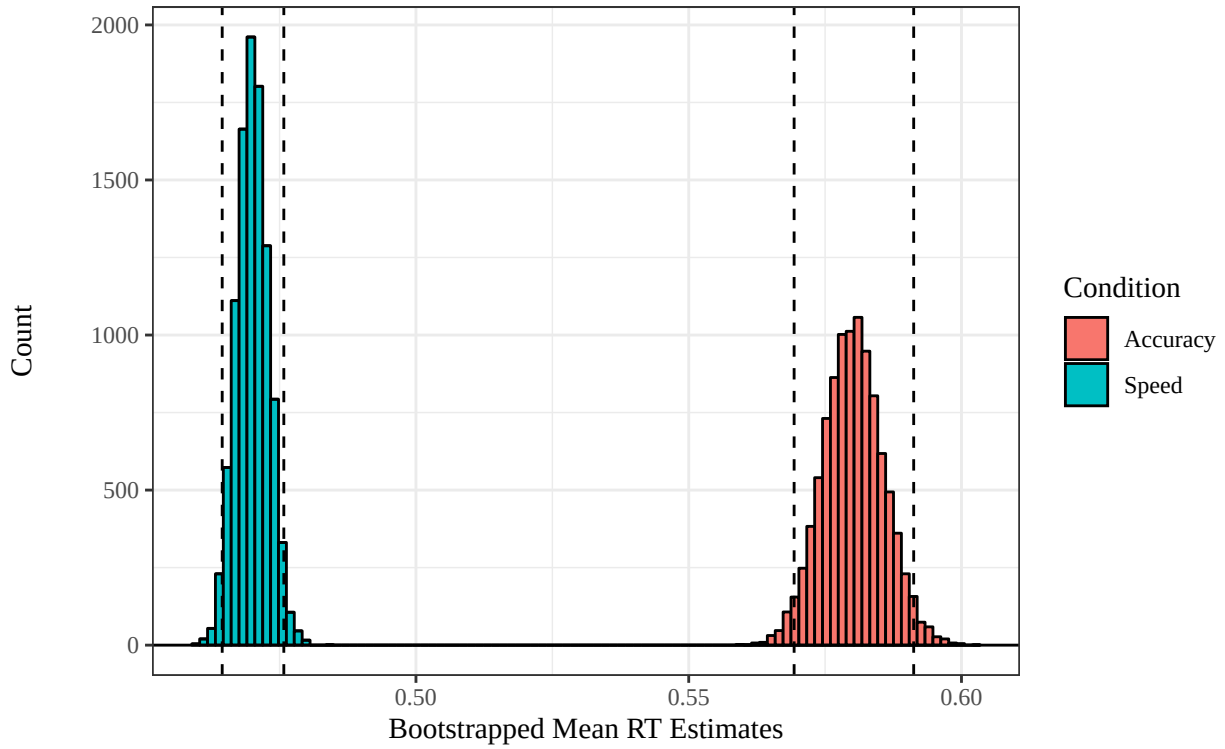


8. Based on this exploratory plot from question 7, do you think there is evidence that participant 1 responded faster when instructed to respond quickly? Explain your reasoning.

```
# Yes, the speed distribution of mean parameter estimates on RT is much
# lower than the accuracy distribution. There is no overlap between the two
# distributions, suggesting a different data generation process for each
# condition.
```

9. Repeat question 7 for **id** 4. Which participant was faster under the speed **condition**, **id** 1 or **id** 4? The accuracy **condition**?

```
# Apply plotting function to ID 4 data.
speed_acc_id_hist(my_data, id_input = "4")
```




```
# ID 4 was faster than ID 1 in both the speed and accuracy conditions!
```

10. **BONUS**: Compute a mean-difference 95% confidence interval for **id** 4 using bootstrapping. You will need to bootstrap the mean difference between the **rt** under the speed and accuracy **condition**. Use 10,000 iterations. Compare your results to the 95% confidence intervals output from the t.test function in base R. Which interval is more narrow, the bootstrapped interval or the t-test interval? Do the bonus question again, but with **accuracy** instead of **rt**. To get **accuracy**, you will have to create a new column (see ?speed_acc for information on existing columns).

```
# Create the acc column by returning matches between stimulus category and response.
id_4_w_acc <- speed_acc %>%
  mutate(acc = speed_acc$stim_cat == as.character(speed_acc$response)) %>%
  filter(id == 4, !censor, response != "error") %>%
  select(condition, rt, acc)

acc_data <- id_4_w_acc %>%
  filter(condition == "accuracy")

speed_data <- id_4_w_acc %>%
  filter(condition == "speed")

# Bootstrap function requires two data frames and a specified dependent variable
bootstrap_mean_diff <- function(acc, speed, iter = 10000, dv){

  set.seed(42)
  mean_diff <- rep(NA, iter)

  dv <- enquo(dv)

  speed <- speed %>%
```

```
    select(!!dv) %>%
    as_vector() %>%
    unname()

  acc <- acc %>%
    select(!!dv) %>%
    as_vector() %>%
    unname()

  for (i in 1:iter){

  resample_acc <- sample(acc, replace = TRUE) %>%
    mean()

  resample_speed <- sample(speed, replace = TRUE) %>%
    mean()

  # Compute the differences of means between the bootstrapped conditions
  mean_diff[i] <- resample_acc - resample_speed

  }

  data <- c(lower = unname(quantile(mean_diff, .025)),
            upper = unname(quantile(mean_diff, .975)))

}

rt_mean_diff <- bootstrap_mean_diff(acc = acc_data, speed = speed_data, dv = rt)
rt_test_conf <- t.test(acc_data$rt, speed_data$rt)$conf[1:2]
rt_mean_diff
```

```
##      lower      upper
## 0.09776996 0.12271689
```

```
rt_test_conf
```

```
## [1] 0.09765385 0.12248592
```

```
# This if statement tests if the difference between the two bounds of the confidence
# intervals is greater (less narrow) in the bootstrap estimate relative to the
# t-test estimate.
if (diff(rt_mean_diff) > diff(rt_test_conf)){
  print("T-test confidence intervals for RT are more narrow")
} else {
  print("Bootstrap confidence intervals for RT are more narrow")
}
```

```
## [1] "T-test confidence intervals for RT are more narrow"
```

```
acc_mean_diff <- bootstrap_mean_diff(acc = acc_data, speed = speed_data, dv = acc)
acc_test_conf <- t.test(acc_data$acc, speed_data$acc)$conf[1:2]
acc_mean_diff
```

```
##      lower      upper
## 0.03536996 0.07705835
```

```
acc_test_conf
```

```
## [1] 0.03501785 0.07741480
```

```
if (diff(acc_mean_diff) > diff(acc_test_conf)){
  print("T-test confidence intervals for accuracy are more narrow")
} else {
  print("Bootstrap confidence intervals for accuracy are more narrow")
}
```

```
## [1] "Bootstrap confidence intervals for accuracy are more narrow"
```