Methods 01: Portfolio Exam

Marie Vestergaard Thomsen - 202208819

202208819@post.au.dk

Group Assignments are made by:

- Matilda Sif Rhys-Kristensen *(MRK)* – 202204611@post.au.dk
- Liv Drasbek *(LD)* – 202208117@post.au.dk
- Thomas Byrum *(TVB)* – 202205950@post.au.dk
- Hugo von Essing-Müller *(HVEM)* – 202206530@post.au.dk
- Marie Vestergaard Thomsen *(MVT)* – 202208819@post.au.dk

Table of Contents:

Assignment 01 – Individual ………………………………………………….. Page 2-13
Assignment 02 – Group ……….……………………………………………….. Page 14-35
Assignment 03 – Group ……..………………………………………………….. Page 36-51
Assignment 04 – Individual ………...………………………………………….. Page 52-63

Portfolio Assignment 1

Marie Thomsen (202208819@post.au.dk)

2022-12-13

Assignment 01 (Individual)

Introduction The goal of this exam is to write a short data mining report on the CogSci Intro Week Personality Test Data in which you answer the following questions **in prose, code and graphs**.

First of all, let's start by looking at the setup chunk. If you need to load packages or set your working directory, do so here:

Now you have to import the personality data. Once you have done so, use the `head()` function to print the first 10 lines of the data set.

Question 1

Question 1.1 Who can hold their breath the longest on average — those with right or left ocular dominance? Notice that the column is called `ocular_dom`, and that right ocular dominance if "Option 1", while left ocular dominance is "Option 2". Therefore, you want to only filter out the data in this column which corresponds to either "Option 1" or "Option 2".

Plot the data using `ggplot2` to find out. The plots should include error bars depicting the standard error of the mean: you can add these using the `geom_errorbar()` function and specifying `stat = "summary"`, `fun.data = "mean_se"`.

```r
#Bar graph showing the standard error of the mean
ggplot(filter(df, ocular_dom != 'Option 3'),
  aes(x = ocular_dom,
      y = breathhold,
      colour = 'black')) + #Creating the basic dimensions of the plot
  geom_bar(stat='summary',
           fun.y = mean,
           width = 0.5,
           colour = 'black',
           fill = 'lightblue') + #Making it a bar-graph
  geom_errorbar(stat = "summary",
                fun.data = "mean_se",
                width = 0.2,
                colour ='red')+#Adding error bar with standard error of the mean
  labs(x = 'Ocular dominance',
       y = 'Breathhold') + #Adding labels for the x and y axis
  ggtitle('Ocular Dominance Versus Breathhold') + #Adding a title
  theme_minimal()
```

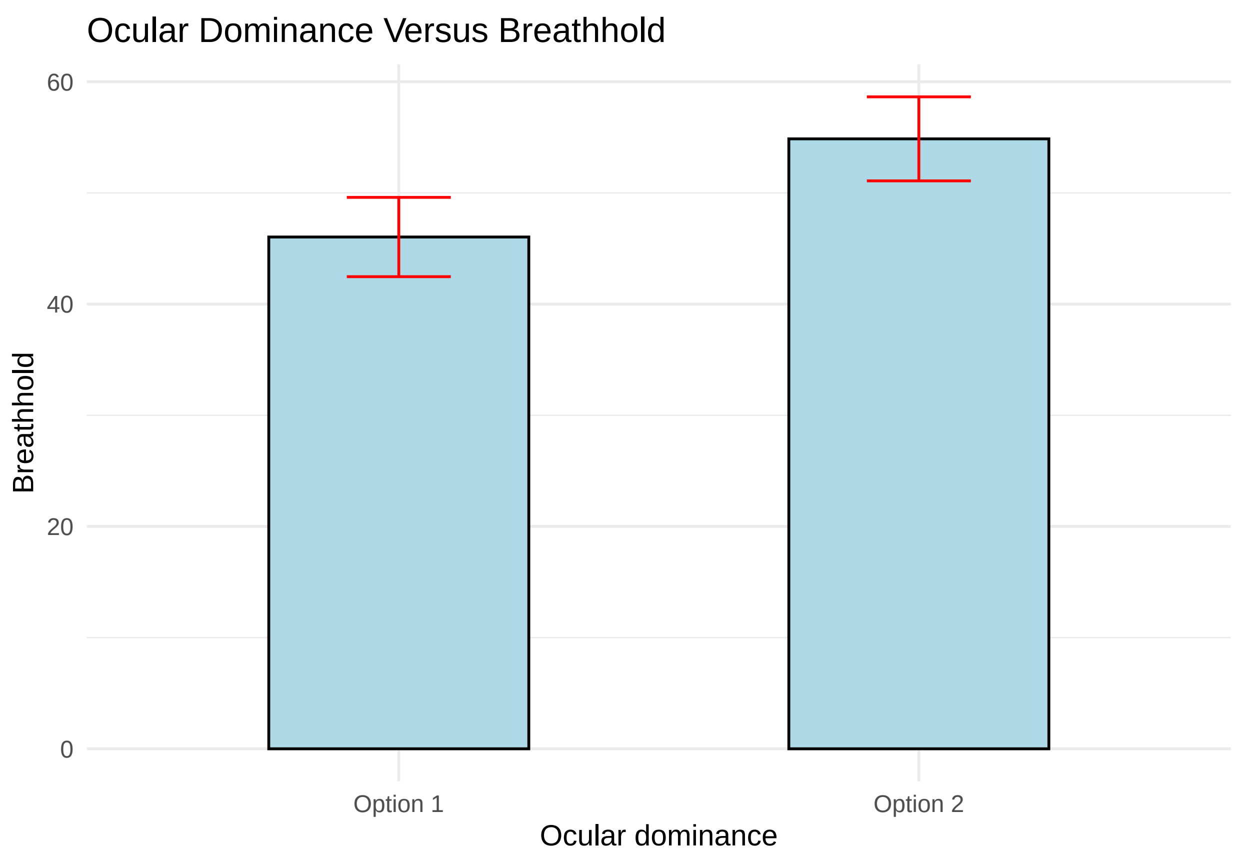


Figure 1: Bar graph with standard error of the mean

Question 1.2 Then use the `mean()` and `sd()` functions *within a tidyverse pipe* to make a summary data set, in which you show mean and standard deviation of the two eye dominance groups.

Bonus question: If you feel brave, you can instead try making a boxplot (`geom_boxplot()`) or a violin plot (`geom_violin()`) which are better at representing the actual distribution of the data (compared to a bar plot, which only depicts mean and standard deviation).

```
knitr::kable(df_summary,
             caption = "Statistical Summary for Breath Hold for each Ocular Dominance",
             digits = c(4,4,4,4,30))
```

Table 1: Statistical Summary for Breath Hold for each Ocular Dominance

ocular_dom	mean_breathhold	sd_breathhold
Option 1	46.0312	14.2647
Option 2	54.8615	24.1960

Figure 2: Summary output of mean and standard deviation of breath hold as grouped by ocular dominance

```
#Creating a boxplot
ggplot(df_filter,
       aes(x = ocular_dom,
           y = breathhold,
           colour = 'black')) + #Creating the basic dimensions of the plot
  geom_boxplot(width = 0.5,
               colour = 'black',
               fill = 'lightblue') + #Creating boxplot
  stat_summary(fun = mean,
               geom = 'point',
               colour = 'red',
               shape = 20) + #Showing a dot identifying the mean
  labs(x = 'Ocular dominance',
       y = 'Breathhold') +
  ggtitle('Boxplot of Ocular Dominance Versus Breathhold')
```

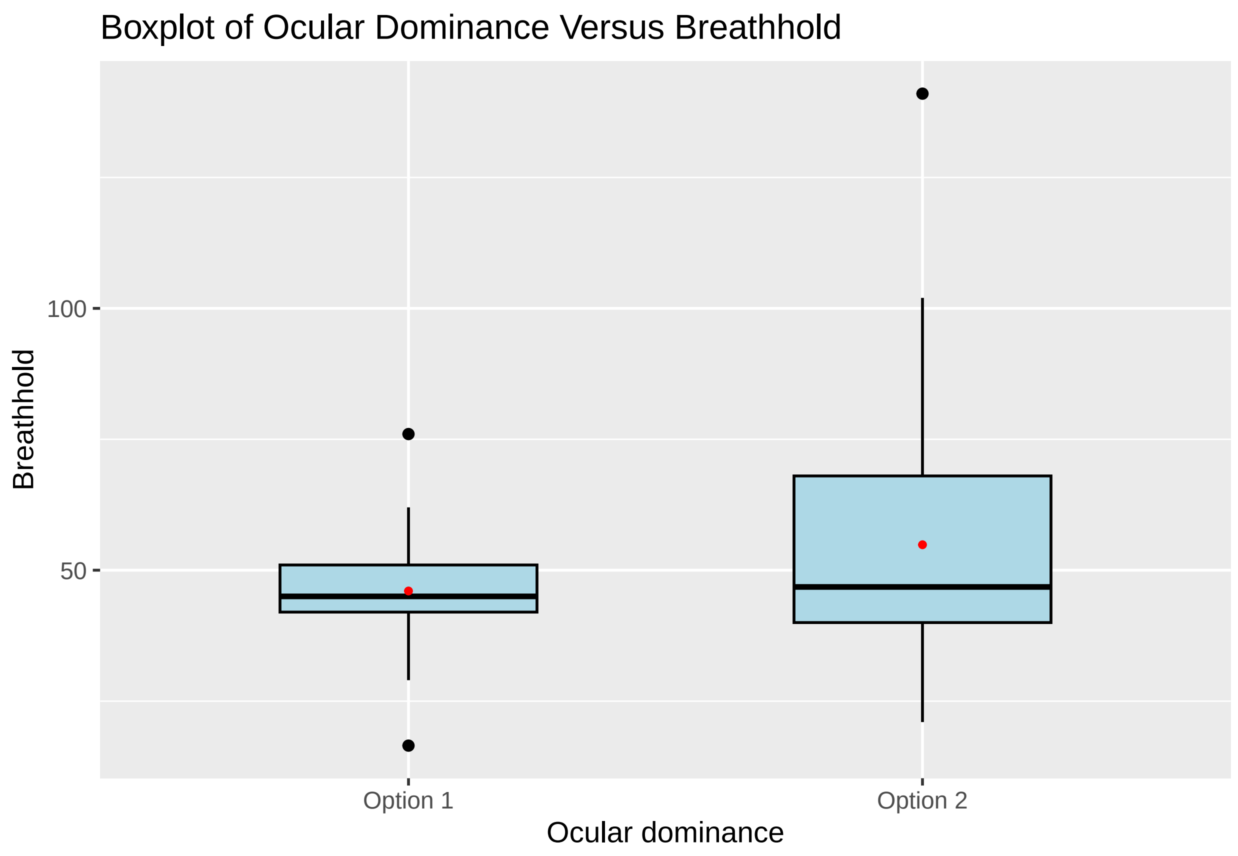


Figure 3: Boxplot displaying breathhold spread for left and right ocular dominance

The results suggest that people with right ocular dominance (option 1) can hold their breath for approximately 46 seconds on average, whereas people with left ocular dominance (option 2) can hold their breath for approximately 55 seconds on average. This indicates that people with left ocular dominance can hold their breath for longer on average. The results also suggest that there is a larger spread in the data for left ocular dominance (option 2) as indicated by the standard deviation (SD = 24.2).

Question 2 Who likes silence vs. noise best – by gender? Also in this case you should plot the data using `ggplot2` (including error bars depicting the standard error of the mean), then use the `mean()` and `sd()`

functions to find mean and standard deviation of the two genders (still making a summary data set with tidyverse and pipes).

Bonus question: If you feel brave, you can instead try making a boxplot (`geom_boxplot()`) or a violin plot (`geom_violin()`) which are better at representing the actual distribution of the data (compared to a bar plot, which only depicts mean and standard deviation).

```r
#Creating a bar plot showing the mean and the standard error of the mean
ggplot(df,
       aes(x = gender,
           y = sound_level_pref,
           fill = gender)) +
  geom_bar(stat='summary',
           fun.y = mean,
           width = 0.5,
           colour = 'black') +
  stat_summary(fun.data = "mean_se",
               geom = "errorbar",
               color = 'red',
               width = 0.1) + #Adding error bar with standard error of the mean
  labs(x = 'Gender',
       y = 'Sound Level Preference') +
  ggtitle('Silence Versus Noise by Gender') +
  theme_minimal()
```

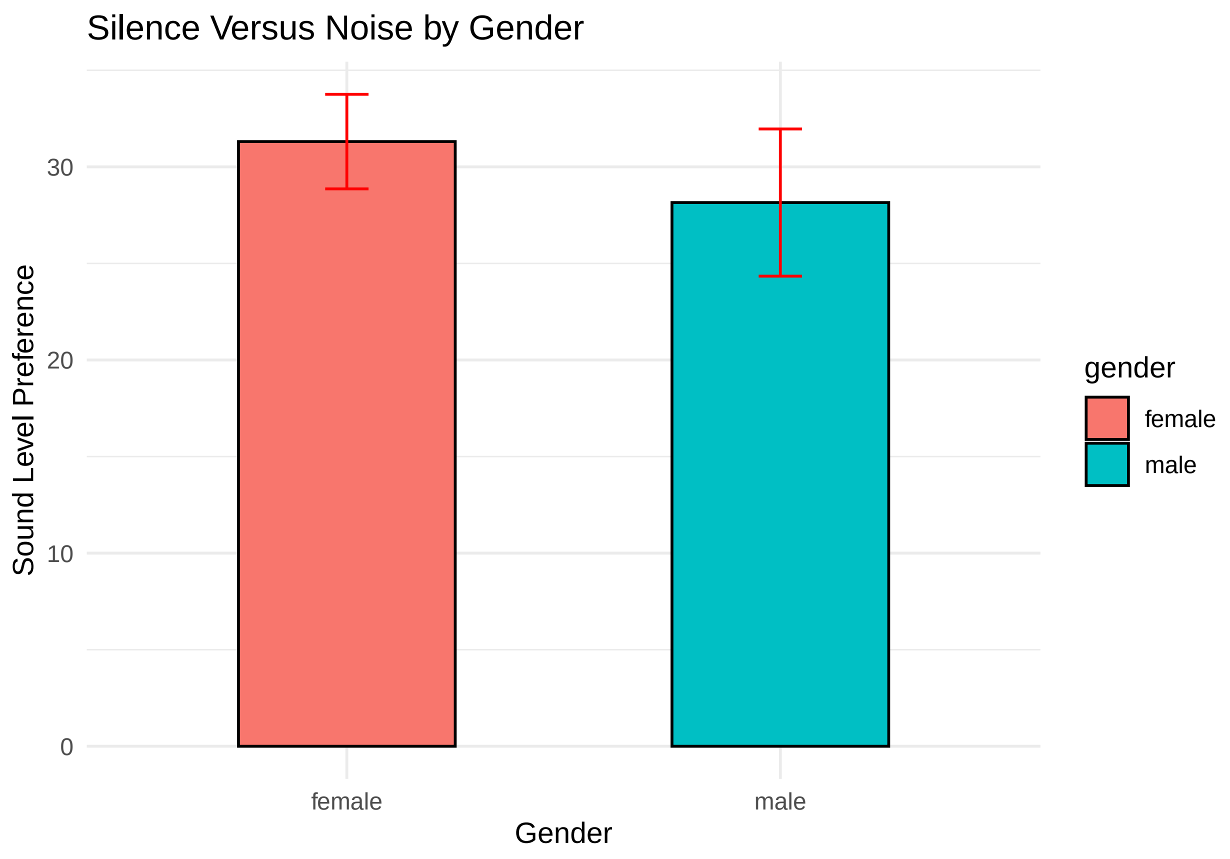


Figure 4: Sound level preference by gender

```r
knitr::kable(df_summary02,
             caption = "Statistical Summary of Sound Level Preference for each Gender",
             digits = c(4,4,4,4,30))
```

Table 2: Statistical Summary of Sound Level Preference for each Gender

gender	mean_sound_level_pref	sd_sound_level_pref
female	31.3077	15.2818
male	28.1500	17.0426

Figure 5: Summary output of sound level perference for each gender

```r
#Creating a violin plot
ggplot(df, aes(x = gender, y = sound_level_pref, fill = gender)) +
  geom_violin() +
  stat_summary(fun = mean, geom = "point", shape = 23, colour = "Black") +
  labs(x = 'Gender', y = 'Sound Level Preference') +
  ggtitle('Silence Versus Noise by Gender') +
  theme_minimal()
```

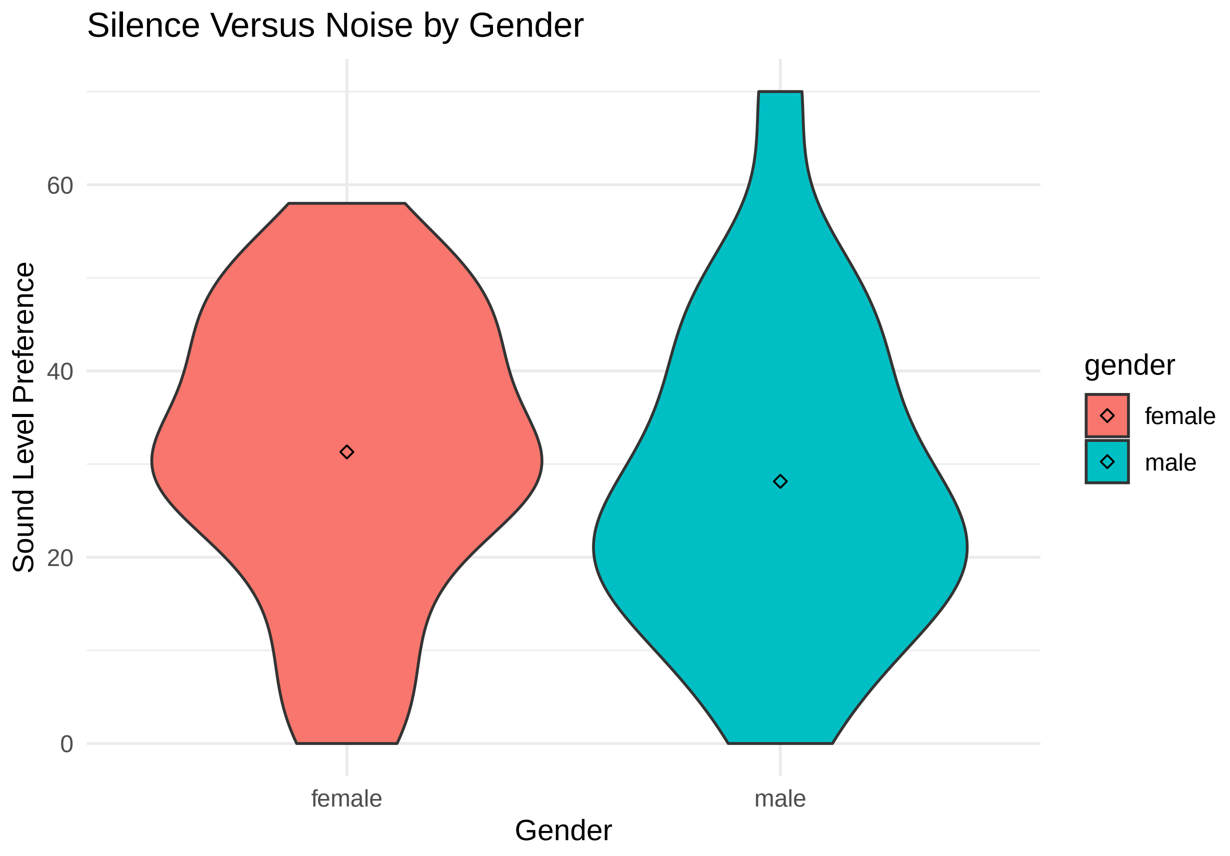


Figure 6: Violin plot showing distribution of sound level preference for each gender

Both genders seem to prefer roughly the same sound level, however it seems that women prefer slightly higher sounds level, as indicated by the mean of approximately 31. The standard error of the mean as depicted by the red bars in the bar graph is greater for males than females. This indicates that the female sample is more representative of the population then the male sample.

Question 3 We predict that there is a positive relation between shoe size and how long time CogSci students can hold their breath. Try plotting the two sets of data against each other using a *scatter plot* (hint: both variables are continuous variables). You can make a scatter plot in `ggplot2` using the `geom_point()` function and plotting one variable on each axis. Use grouping in your plot to distinguish the relationship between shoe size and holding breath for males and females, since we expect males and females to have different show sizes. You can for instance use the `color` parameter within the `aes()` function to color by gender.

```r
#Scatter plot showing relationship between shoe size and breath hold as grouped by gender
ggplot(df, aes(x = shoesize, y = breathhold, fill = gender, colour = gender)) +
  geom_point(size = 2) + #Creating the scatterplot
  geom_smooth(method = lm, se = TRUE) + #Adding a regression line
  labs(x = 'Shoesize', y = 'Breath hold') +
  ggtitle('Shoesize Versus Breath hold as Grouped by Gender') +
  theme_minimal()
```

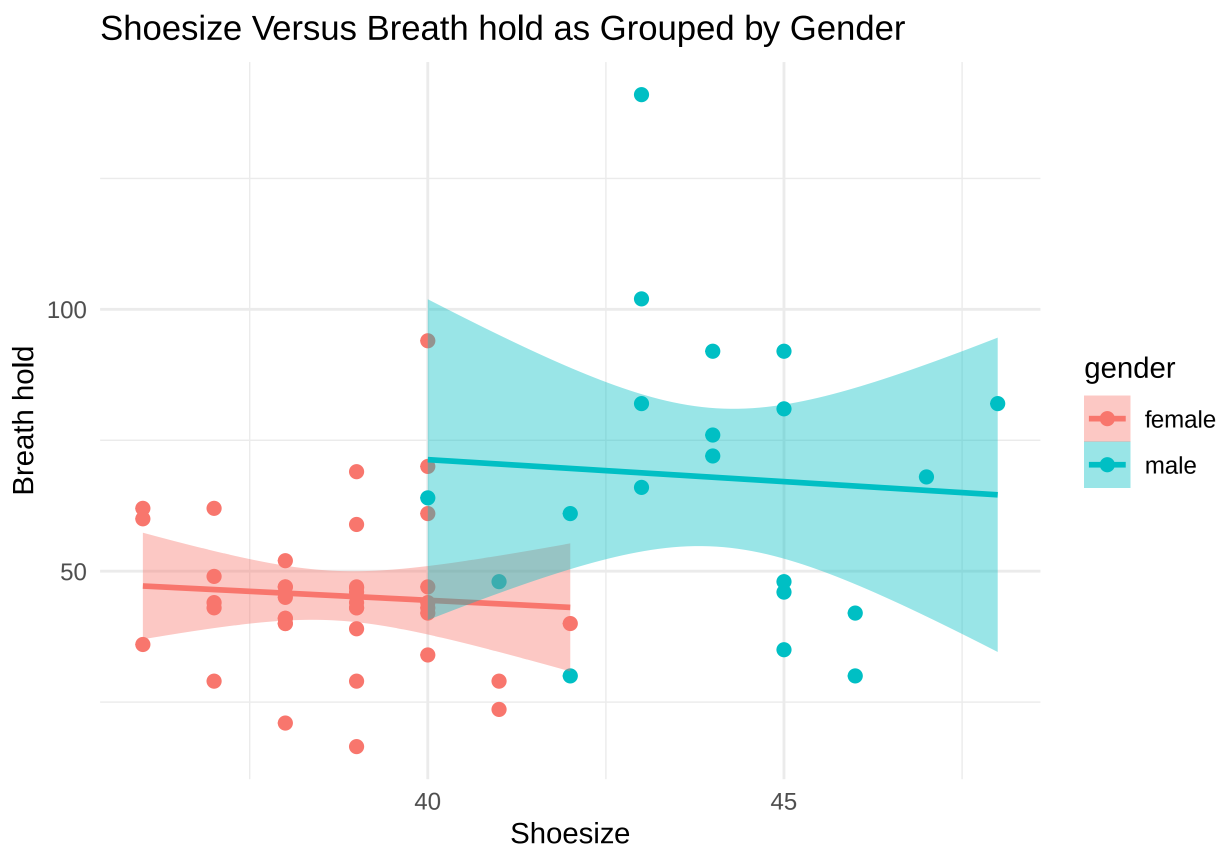


Figure 7: Scatterplot for each gender displaying shoesize and breathhold.

The breath hold seems to slightly decrease as shoe size increases for each gender, which rejects the prediction. This suggests that there is a negative relation between breath hold and shoe size. The data also suggests that people in the male category generally can hold their breath longer than females. —

Question 4 Is the breathhold variable normally distributed? Provide both visual (histogram and QQ-plot) and numeric (Shapiro-Wilk test and skewness/kurtosis values) support for your answer.

```
#Creating a histogram displaying the probability density function
ggplot(df, aes(x = breathhold)) +
  geom_histogram(aes(y = ..density..),
               binwidth = 7.5,
               color = "black",
               fill = "lightblue") +
  stat_function(fun = dnorm,
              args = list(mean = mean(df$breathhold, na.rm = TRUE),
              sd = sd(df$breathhold, na.rm = TRUE)),
              colour= "red", size = 1) + #plotting the normal curve
  theme_minimal()+
  labs(x = "Breath Hold", y = "Density")
```

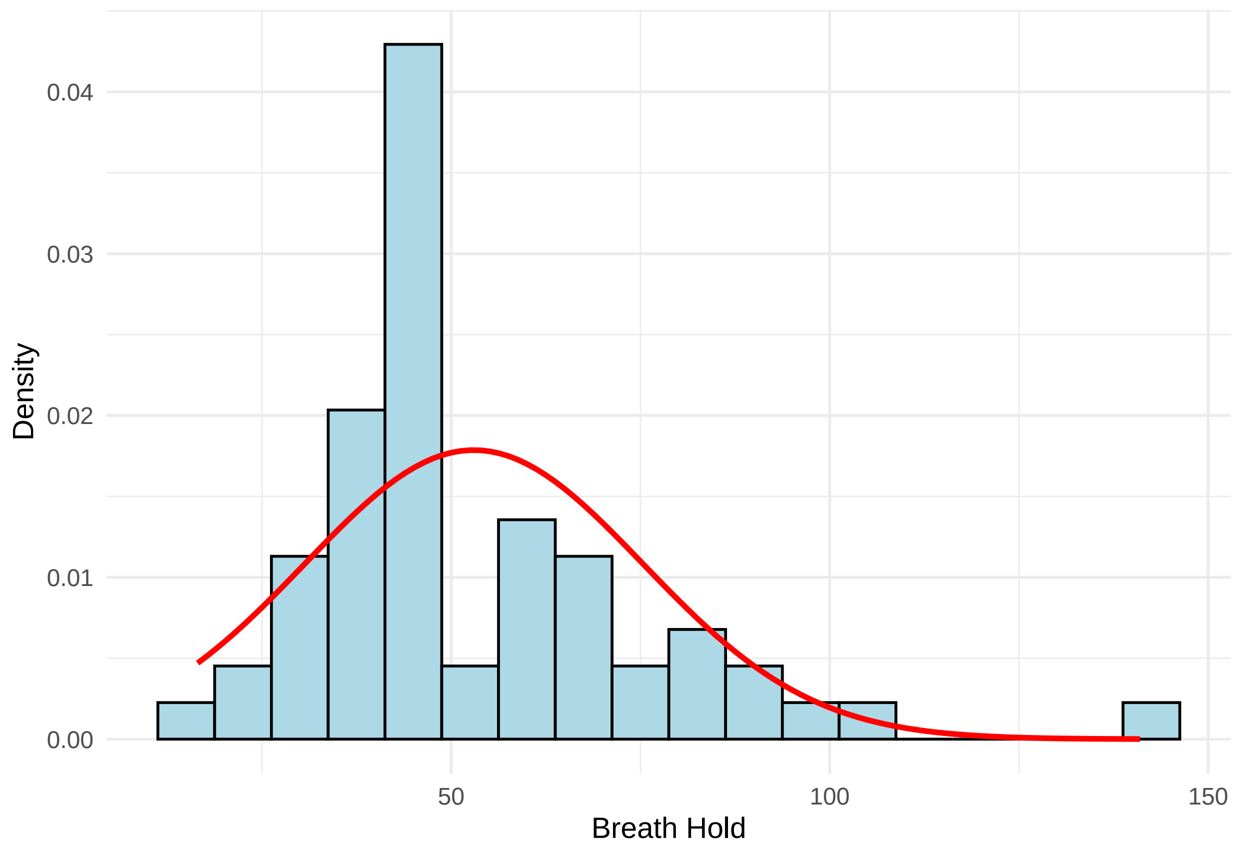
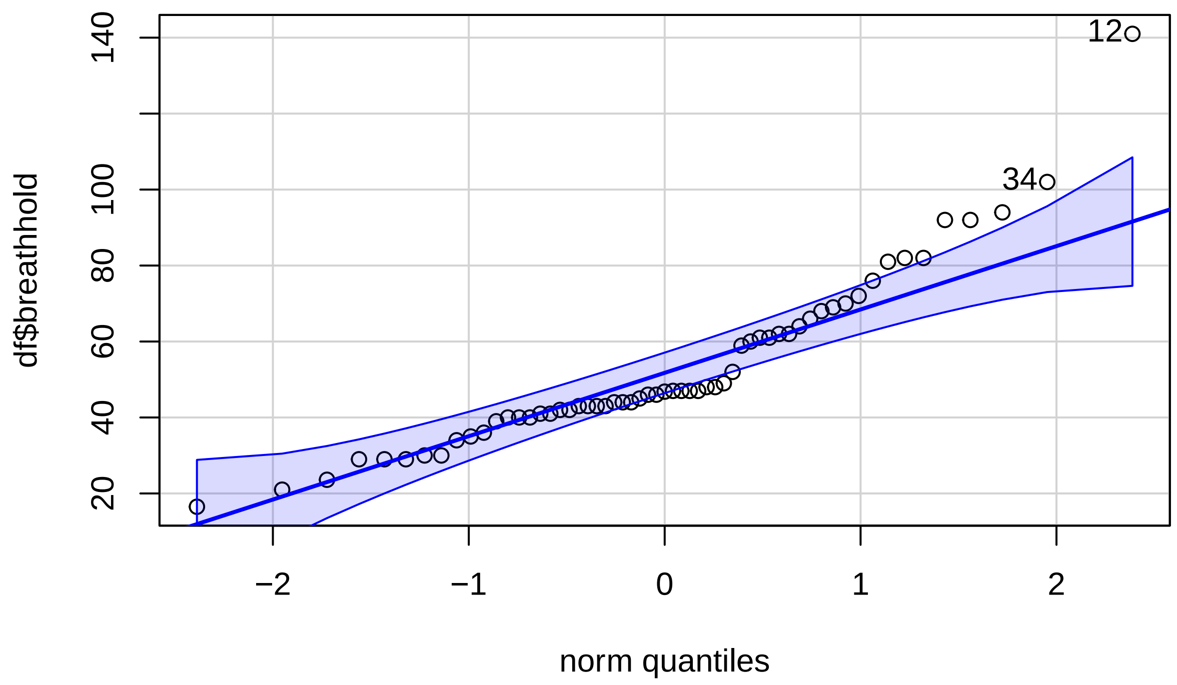


Figure 8: Probability Density Function of breathhold

```
#qq-plot of data
qqPlot(df$breathhold)
```



```
## [1] 12 34
```

Figure 9: QQ-plot of the variable breathhold

```
#Numeric values to test for normality
round(pastecs::stat.desc(df$breathhold,
                         basic = FALSE,
                         norm = TRUE), digits = 2)
```

```
##       median         mean      SE.mean CI.mean.0.95          var      std.dev
##        46.80        53.00         2.91         5.82       498.73        22.33
##     coef.var     skewness     skew.2SE     kurtosis     kurt.2SE   normtest.W
##         0.42         1.34         2.15         2.56         2.09         0.90
##   normtest.p
##         0.00
```

```
shapiro.test(df$breathhold)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  df$breathhold
## W = 0.89612, p-value = 0.0001084
```

The data is not normally distributed, which can be seen by the qq-plot as it is not fully linear but rather increases more towards the end, which indicates that it is skewed to the right i.e positively skewed. It is also displayed by the numerical values given by the Shapiro-Wilk test. As p<0.05 it indicates that the distribution is not normal. It can also be seen by the skew.2SE as it is a very large value, which also suggests that the distribution is skewed. —

Question 5 Are the two balloon reaction time variables (`balloon` and `balloon_balance`) normally distributed? Provide visual (histogram and QQ-plot) and numeric (Shapiro-Wilk test and skewness/kurtosis values) support for your answer. If they are not, then discuss your results below.

```
#Balloon histogram
ggplot(df, aes(x = balloon)) +
  geom_histogram(aes(y = ..density..),
                binwidth = 1,
                color = "black",
                fill = "lightblue") +
  stat_function(fun = dnorm,
               args = list(mean = mean(df$balloon, na.rm = TRUE),
               sd = sd(df$balloon, na.rm = TRUE)),
               colour= "red", size = 1) +
  theme_minimal()+
  xlim(range(df$balloon))+
  labs(x = "Balloon", y = "Density") +
  ggtitle("Balloon")
```

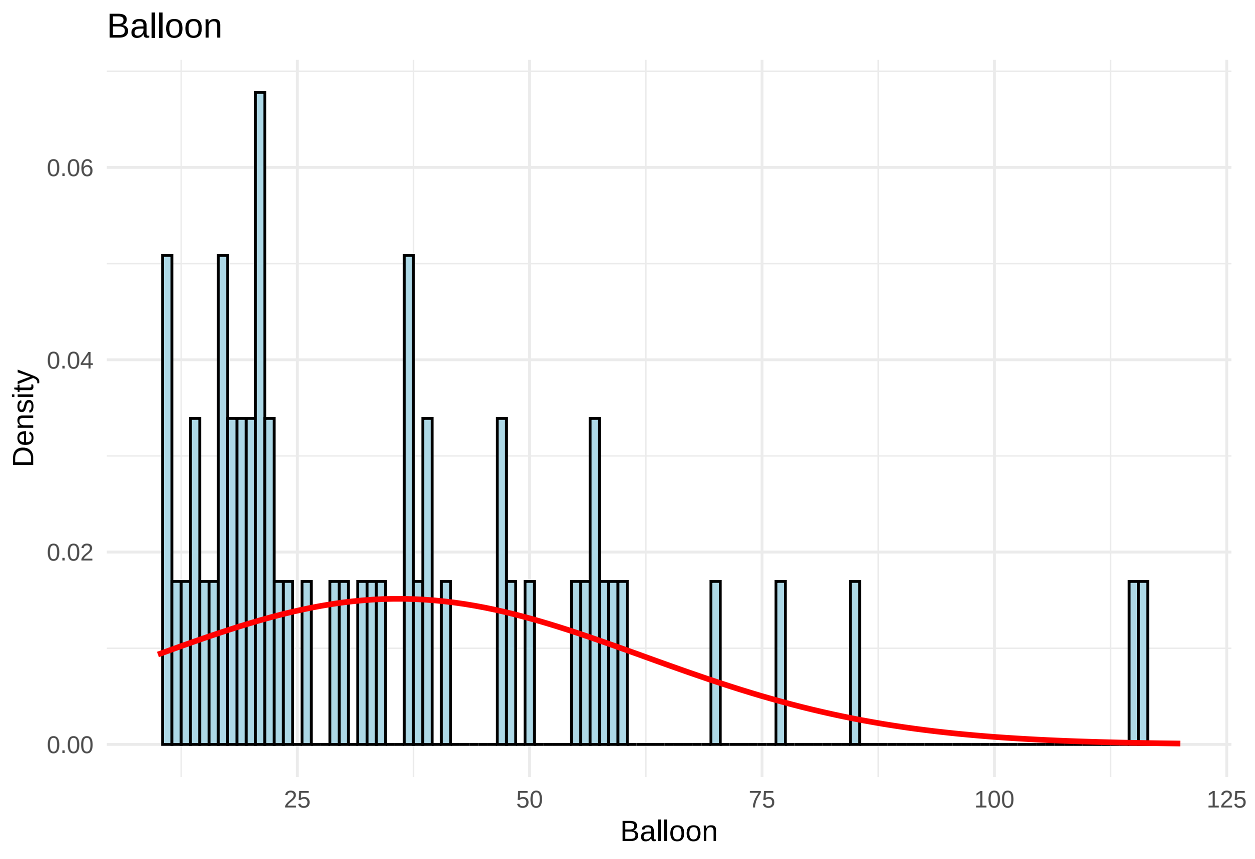


Figure 10: Probability Density Function of the variable 'balloon'

```
#qq-plot of balloon
qqnorm(df$balloon)
qqline(df$balloon)
```

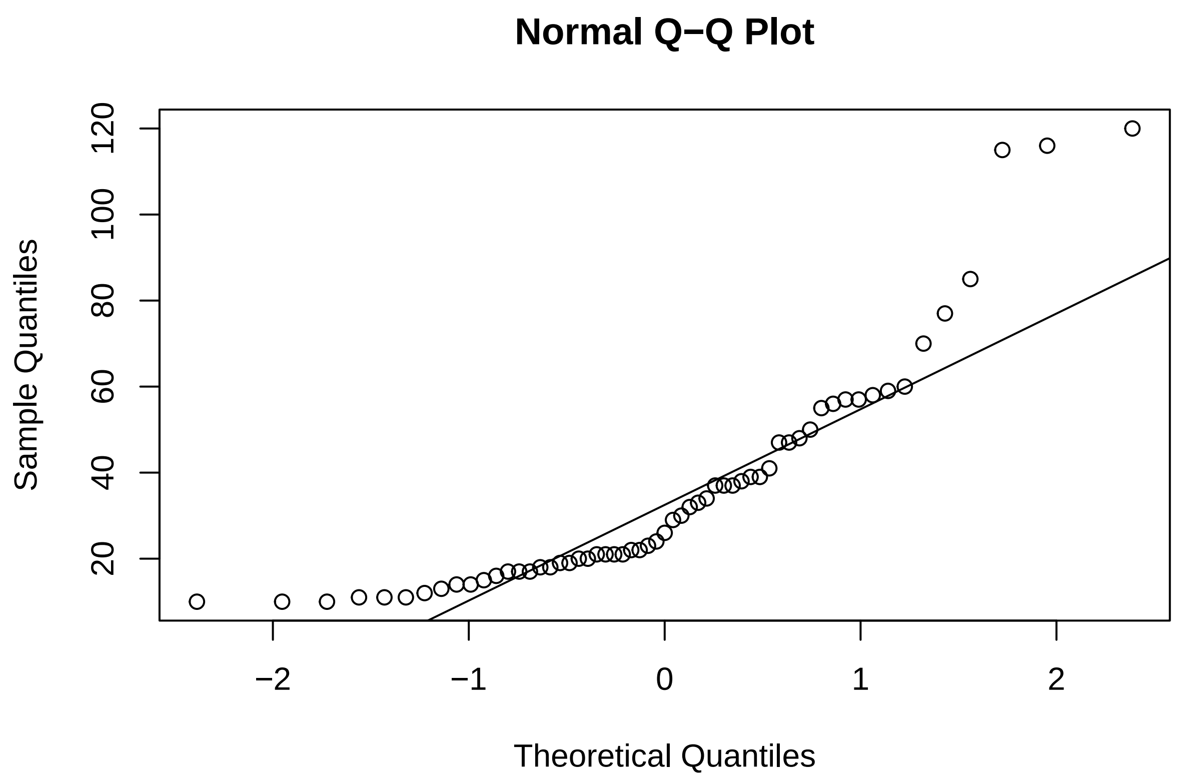


Figure 11: QQ-plot the 'balloon'

```
#Numeric values for balloon
round(pastecs::stat.desc(df$balloon, basic = FALSE, norm = TRUE), digits = 2)
```

```
##       median         mean      SE.mean CI.mean.0.95          var      std.dev
##        26.00        35.86         3.43         6.87       694.05        26.34
##     coef.var     skewness     skew.2SE     kurtosis     kurt.2SE   normtest.W
##         0.73         1.56         2.50         2.19         1.78         0.82
##   normtest.p
##         0.00
```

```
shapiro.test(df$balloon)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  df$balloon
## W = 0.82011, p-value = 5.277e-07
```

```
#Balloon Balance histogram
ggplot(df, aes(x = balloon_balance)) +
  geom_histogram(aes(y = ..density..),
                 binwidth = 1,
                 color = "black",
                 fill = "lightblue") +
  stat_function(fun = dnorm,
                args = list(mean = mean(df$balloon_balance, na.rm = TRUE),
                sd = sd(df$balloon_balance, na.rm = TRUE)),
                colour= "red", size = 1) +
  theme_minimal()+
  xlim(range(df$balloon_balance))+
  labs(x = "Balloon Balance", y = "Density") +
  ggtitle("Balloon Balance")
```

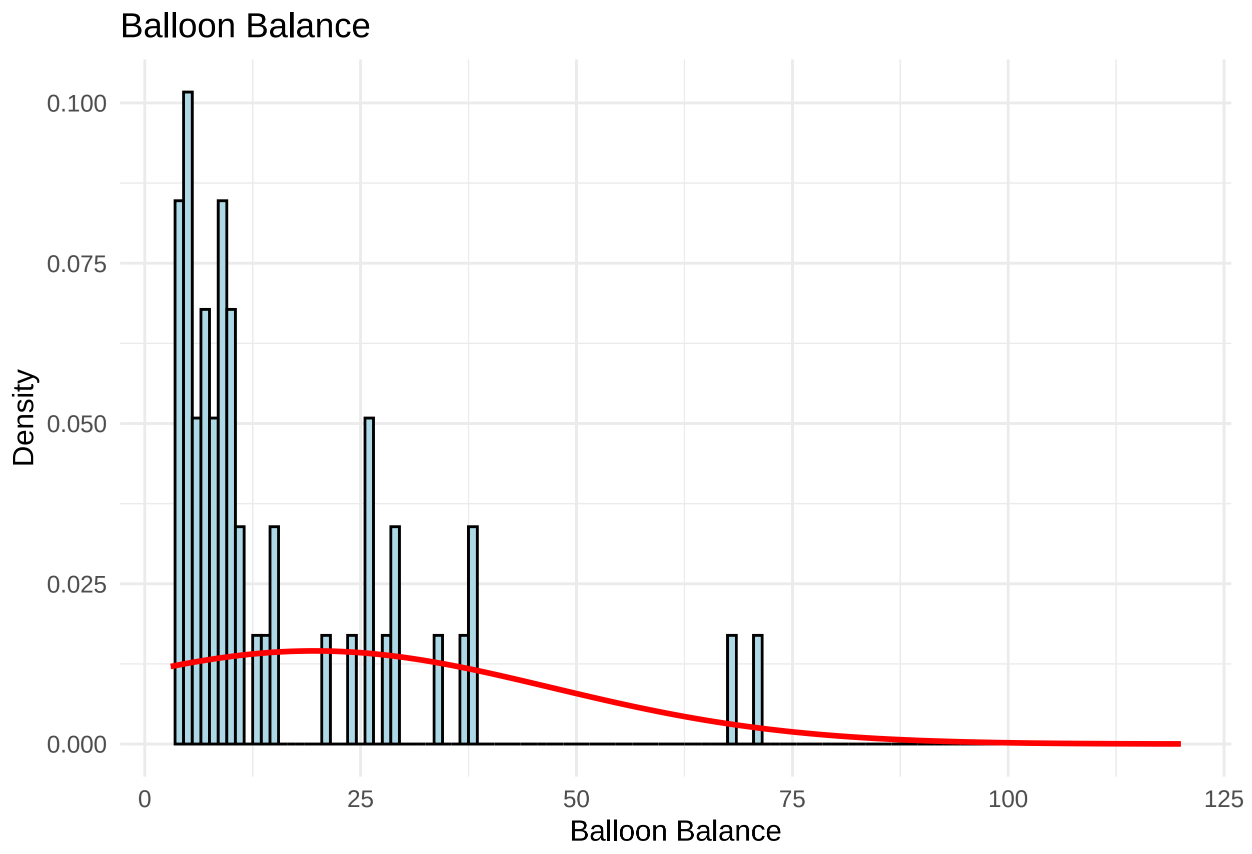


Figure 12: Probability Density Function of 'balloon balance'

```
#qq-plot of balloon balance
qqnorm(df$balloon_balance)
qqline(df$balloon_balance)
```

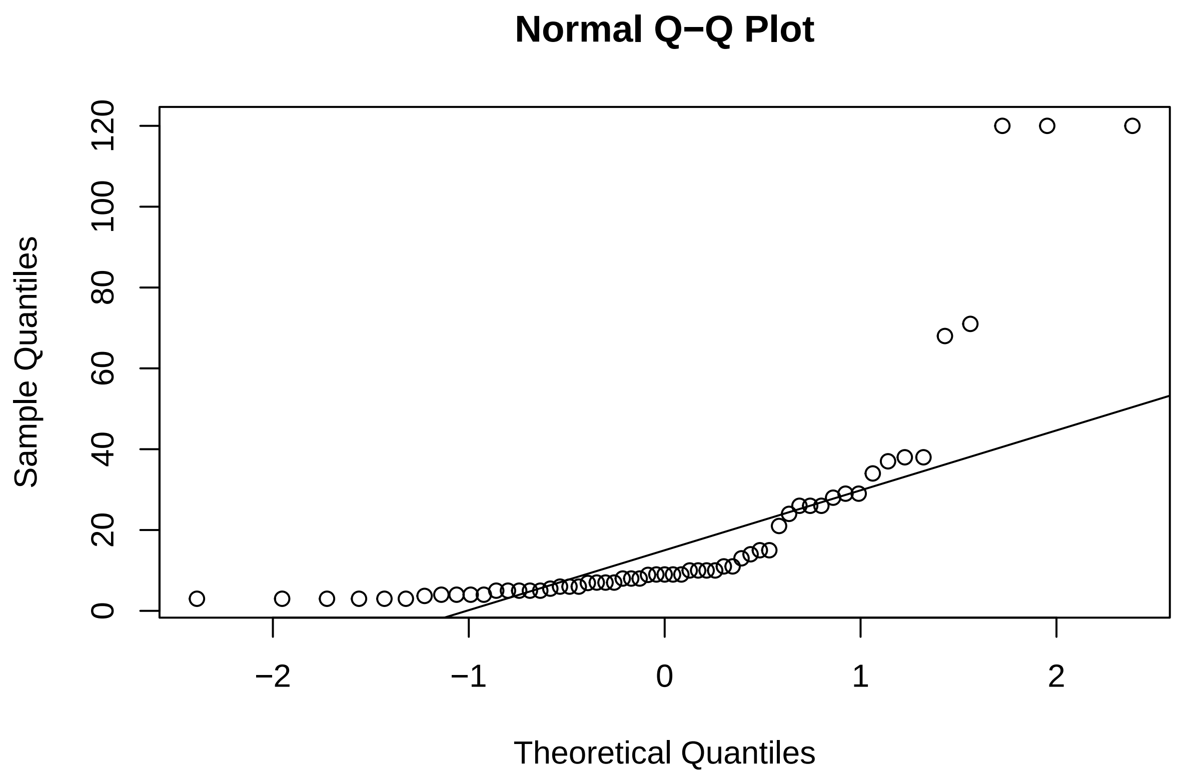


Figure 13: QQ-plot of 'balloon balance'

```
#Numeric values for balloon balance
round(pastecs::stat.desc(df$balloon_balance, basic = FALSE, norm = TRUE), digits = 2)
```

```
##       median         mean      SE.mean CI.mean.0.95          var      std.dev
##         9.00        19.61         3.58         7.16       754.17        27.46
##     coef.var     skewness     skew.2SE     kurtosis     kurt.2SE   normtest.W
##         1.40         2.63         4.22         6.50         5.30         0.59
##   normtest.p
##         0.00
```

```
shapiro.test(df$balloon_balance)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  df$balloon_balance
## W = 0.59238, p-value = 1.542e-11
```

Both balloon and balloon balance is skewed to the right (positively skewed) which can be seen by the qq-plots as they are increasing more towards the end of the function, compared to the line representing the normal distribution The p-value for both balloon and balloon balance is $p<0.05$, indicating that it is not normally distributed. The skew.2SE and kurt.2SE are greater than 1, which also indicates that the data is not normally distributed

Portfolio Assignment 2

Marie Thomsen (202208819@post.au.dk)

2022-12-13

Assignment 2: Reading Experiment (Group)

Assignment by Marie Vestergaard Thomsen (MVT), Matilda Rhys-Kristensen (MRK), Thomas Villads Byrum (TVB), Hugo von Essen-Müller (HVEM) & Liv Drasbek(LD)

Independent variables: (1) Coherent (not surprising word) (2) Incoherent (surprising word)

Dependent variable: Reaction time

```
knitr::opts_chunk$set(echo = TRUE, include = TRUE, message = FALSE, warning = FALSE)
knitr::opts_knit$set(root.dir =
↪  '/work/CogSci_Methods01/portolio_assignment_02-marievthomsen')
knitr::opts_chunk$set(fig.align = "center")
```

```
library(tidyverse)
library(gridExtra)
library(pastecs)
library(stringi)
library(WRS2)
library(ggplot2)
library(lawstat)
library(knitr)
library(ggpubr)
```

Python code (reading_experiment.py)(all)

```
# Import modules
from psychopy import visual, core, event, gui, data
import glob
import random
import pandas as pd
from numpy.random import shuffle

#show dialogue box
dialogue = gui.Dlg(title = "The Reading Experiment")
dialogue.addField("Participant ID:")
dialogue.addField("Age:")
dialogue.addField("Declared gender:", choices = ["Female", "Male", "Other"])
dialogue.show()
if dialogue.OK:
    # write dialogue inputs to variables
    ID = dialogue.data[0]
    Age = dialogue.data[1]
    Gender = dialogue.data[2]
elif dialogue.Cancel:
    core.quit()

# Define window
win = visual.Window(color = "pink", fullscr = True)

#define stop watch
stopwatch = core.Clock()

# Define logfile
columns = ["ID", "Age", "Gender", "Word", "Reaction Time", "Condition", "Word length"]
logfile = pd.DataFrame(columns = columns)

# get date/time
date = data.getDateStr()

#Instructions
Instructions = '''
Hi! Welcome to the experiment!

In a moment, you will be presented with a text one word at a time.
You will do a self-paced reading by pressing the spacebar to skip to the next word.

Press spacebar when you are ready to begin the experiment
'''
goodbye = '''
You have completed the experiment, thank you for your participation!
'''


# Define functions
list1 = "It was 2:00 in the morning. Finding that she could not go back to sleep, she
↪  slowly got out of bed and walked downstairs to the kitchen. Putting on an apron, she
↪  created the filling by mixing ground pork, chives, minced garlic and soy sauce
↪  together. Then, taking the freshly-kneaded dough, she rolled it into several thin
↪  sheets. When the ingredients were prepared, she placed a thin sheet of dough on the
↪  palm of her hand and gently scooped up some of the filling. She placed the filling at
↪  the center of the dough sheet and pinched the ends of the dough sheet together.
↪  Afterwards, she stir-fried the dumplings in oil until they got crispy. Placing the
↪  Chinese dumplings on a plate, she took out a pair of chopsticks and began savoring
↪  the taste. When she had a hard time sleeping at night, eating her dumplings would
↪  always help.".split()
list2 = "It was 2:00 in the morning. Finding that she could not go back to sleep, she
↪  slowly got out of bed and walked downstairs to the kitchen. Putting on an apron, she
↪  created the filling by mixing ground pork, chives, minced garlic and soy sauce
↪  together. Then, taking the freshly-kneaded dough, she rolled it into several thin
↪  sheets. When the ingredients were prepared, she placed a thin sheet of dough on the
↪  palm of her hand and gently scooped up some of the filling. She placed the filling at
↪  the center of the dough sheet and pinched the ends of the dough sheet together.
↪  Afterwards, she stir-fried the dumplings in oil until they got crispy. Placing the
↪  Chinese dumplings on a plate, she took out a pair of socks and began savoring the
↪  taste. When she had a hard time sleeping at night, eating her dumplings would always
↪  help.".split()
```

```
#Function for showing text and waiting for key press
def msg(txt):
    msg = visual.TextStim(win, text = txt)
    msg.draw()
    win.flip()
    key = event.waitKeys(keyList = ["space"])

#Stimulus
word_stim = visual.TextStim(win, text = 'XXXX')

#Begin experiment
#Instructions
msg(Instructions)

#Prepare text
sentences = [list1, list2]
sentence_shuffle = shuffle(sentences)

if sentences[0] == list1:
    condition = 'Control'
else:
    condition = 'Experimental'

for word in sentences[0]:
    word_stim.text = word
    wordlength = len(word)
    word_stim.draw()
    win.flip()
    stopwatch.reset()
    # wait for a response
    key = event.waitKeys(keyList = ["space"])
    reaction_time = stopwatch.getTime()

    logfile = logfile.append({
    "ID": ID,
    "Age": Age,
    "Gender": Gender,
    "Word": word,
    "Reaction Time": reaction_time,
    "Word length":wordlength,
    "Condition": condition}, ignore_index = True)


# save the logfile
logfile_name = "logfiles/logfile_{}_{}.csv".format(ID, date)
logfile.to_csv(logfile_name)

# show goodbye message
msg(goodbye)

# Flip the window
win.flip()
```

```
# Wait for 2s
core.wait(2)
```

R code analysis

Preparing data (all)

```
#Loading the data
files <- list.files(path = 'logfile_new.zip', pattern = ".csv",full.names = T)
```

Anonymising the data - only run once (all)

```
# data_out <- list()
#   num_files <- length(files)
#   rand_ids <- sample(seq(1,num_files,1))
#   cnt_f <- 0
#   for (f in files){
#     cnt_f <- cnt_f + 1
#     data_out[[f]] <- read_csv(file = f, col_names = TRUE)
#     data_out[[f]]$ID <- paste(c("snew", rand_ids[cnt_f]), collapse = "")
#     out_name <-
↪   paste(c('/work/CogSci_Methods01/portolio_assignment_02-marievthomsen/logfile_new.zip',
↪   "/logfile_", unique(data_out[[f]]$ID[1]), ".csv"), collapse = "")
#     write_csv(data_out[[f]], out_name, na = "NA")
#     file.remove(f)
#   }
```

Creating data frame (all)

```
files <- list.files(path =  getwd(), pattern = "*logfile_snew*", full.names = T)

data <- map_dfr(files, read_csv)
```

Cleaning data (TVB)

```
#Changing names of columns
names(data)[6] <- 'reaction_time'
names(data)[8]<- 'word_length'
names(data)[1]<- 'column_1'
names(data)[7]<- 'condition'

#Removing punctuation from words in the data frame
```

```
data$Word <- sub(",","",data$Word)
data$Word <- sub("\\.","",data$Word)
data$word_length <- nchar(data$Word)

#Ensuring ID and Condition are factors
data$condition <-as.factor(data$condition)
data$ID <- as.factor(data$ID)
```

Aggregating by test subject to detect irregularities (HVEM)

```
ggplot(data, aes(x = ID, y = reaction_time, fill = ID)) +
  geom_boxplot(outlier.shape = NA) +
  coord_cartesian(ylim = c(0.1, 1.5)) +
  stat_summary(fun = mean, geom = "point", shape = 23)
```

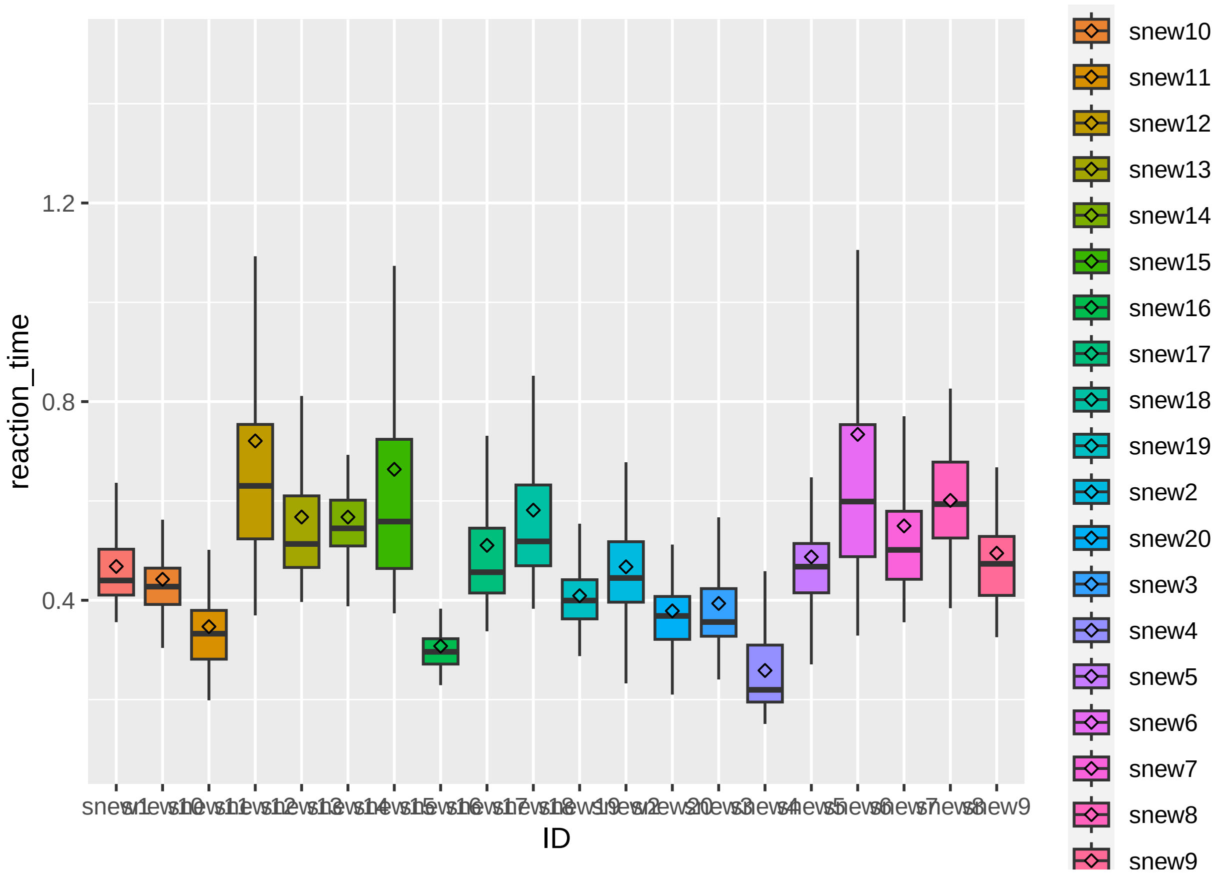


Figure 1: Showing aggregation by test subject

Correlation section

Preparing Data (TVB)

```r
#New data frame where the rows with chopsticks and socks are eliminated
data_filter <- filter(data, Word != "socks")
data_filter <- filter(data_filter, Word != "chopsticks")

#Adding transformations to the data frame
data_filter <- data_filter %>%
  mutate(log_reaction_time = log(reaction_time),
         sqrt_reaction_time = sqrt(reaction_time),
         reaction_time_1 = (1/reaction_time))

#Creating a column with ordinal word number
ordinal_number <- data_filter$column_1 + 1
data_filter <- mutate(data_filter, ordinal_number)
```

Testing for normality in reaction time (LD)

```r
#Probability density histogram of reaction time
  ggplot(data_filter, aes(x = reaction_time)) +
  geom_histogram(aes(y = after_stat(density)),
               binwidth = 0.25,
               fill = 'lightblue',
               colour = 'black') +
  labs(x = 'Reaction Time', y = 'Probability Density') +
  ggtitle("Probability Density of Reaction Time") +
  stat_function(fun = dnorm,
              args = list(mean = mean(data_filter$reaction_time,
                                      na.rm = TRUE),
              sd = sd(data_filter$reaction_time, na.rm = TRUE)),
              colour= "red", linewidth = 1) +
  theme_minimal() + xlim(range(data_filter$reaction_time))
```

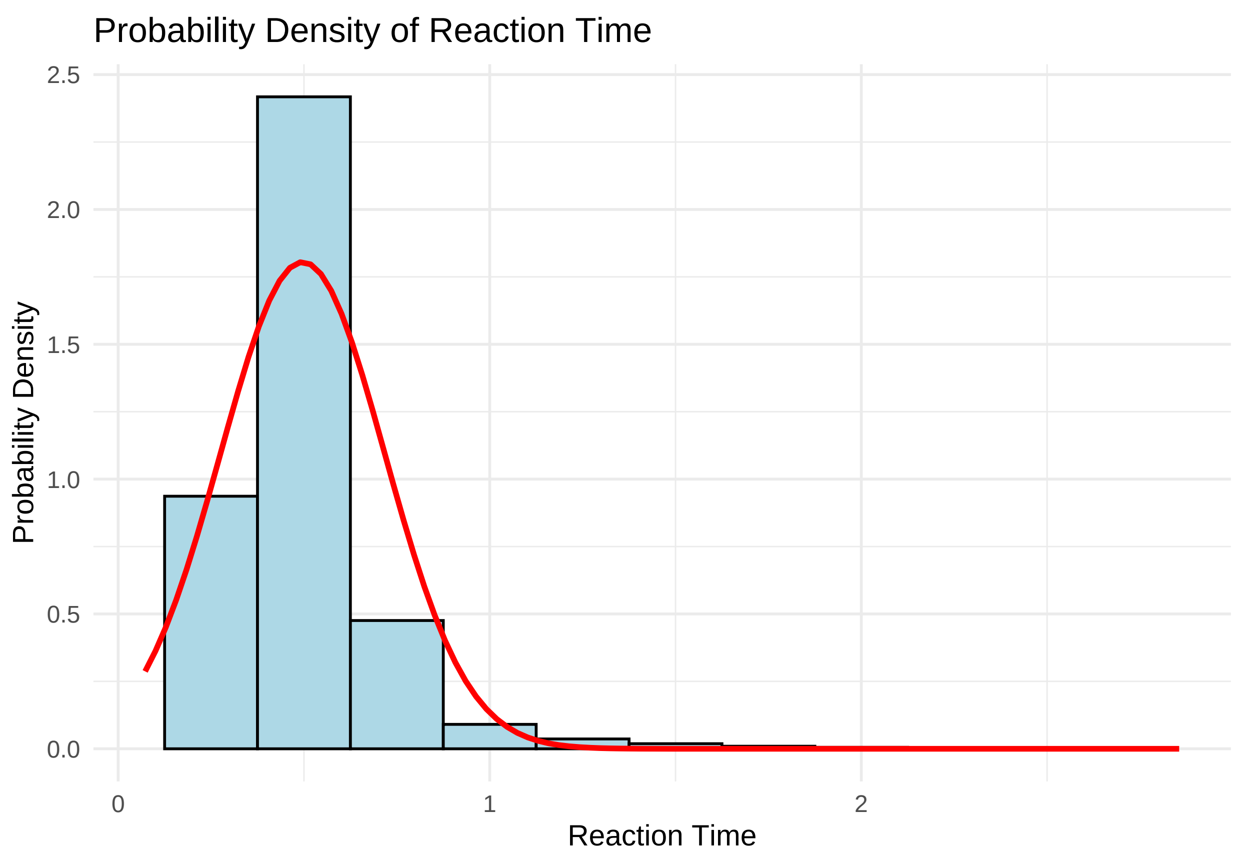


Figure 2: Probability density histogram of reaction time

```r
#QQ-plot
qqnorm(data_filter$reaction_time)
qqline(data_filter$reaction_time)
```

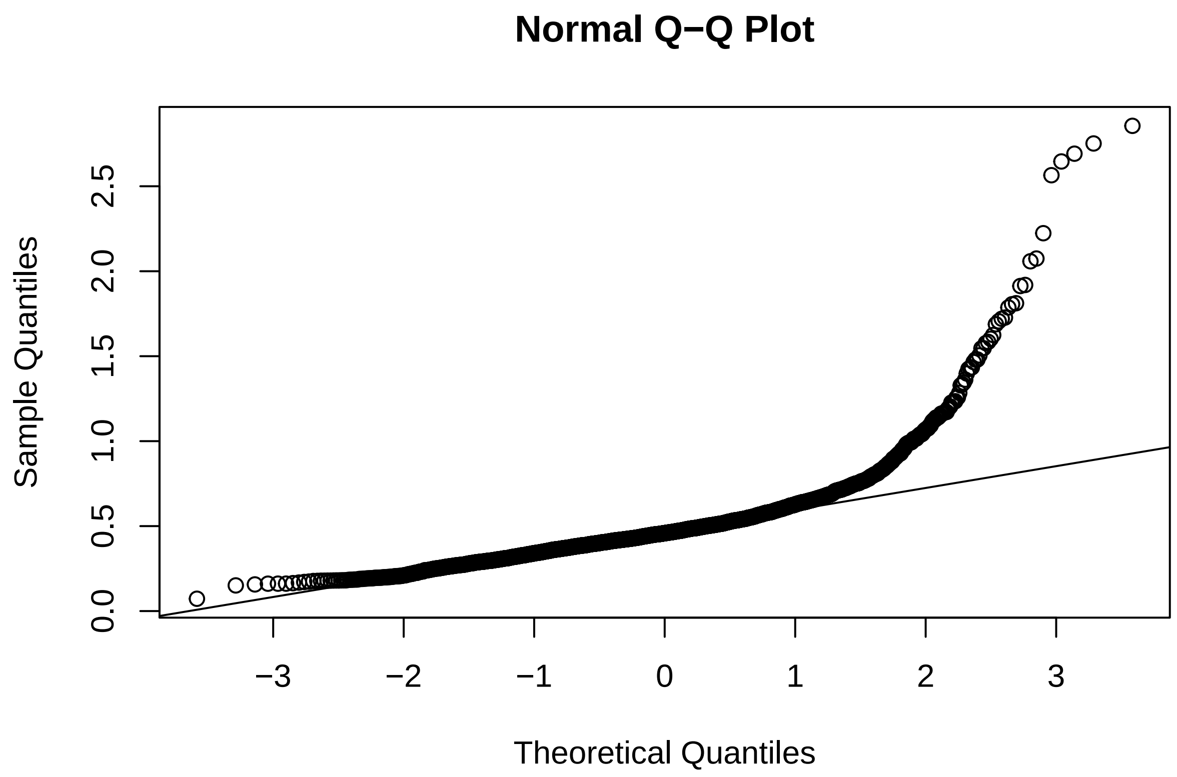


Figure 3: QQ-plot of reaction time

```
#Shapiro-Wilks test
shapiro.test(data_filter$reaction_time)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  data_filter$reaction_time
## W = 0.7388, p-value < 2.2e-16
```

Both the probability density function as well as the qq-plot shows a positive skew and the data is therefore not normally distributed. We therefore, have to see if we can transform the data to make it normal.

Transforming the data (MRK)

```
data_transform <- select(data_filter, reaction_time) %>%
  mutate(log_reaction_time = log(reaction_time),
         sqrt_reaction_time = sqrt(reaction_time),
         reaction_time_1 = (1/reaction_time))
```

```
#Numeric values to determine normality
round(stat.desc(data_transform, norm = TRUE),
      digits = 2)[c("skew.2SE", "kurt.2SE", "normtest.p"), ]
```

```
##            reaction_time log_reaction_time sqrt_reaction_time reaction_time_1
## skew.2SE           40.67              4.88              19.46           22.24
## kurt.2SE          137.30             13.30              42.95           75.54
## normtest.p          0.00              0.00               0.00            0.00
```

Transformation using logarithm (MRK)

```
grid.arrange(ggplot(data_filter, aes(x = log_reaction_time)) +
  geom_histogram(aes(y = after_stat(density)),
  binwidth = 0.25,
  color= 'black', fill = 'lightblue') +
  ggtitle('Reaction Time Log Transformed') +
  labs(x = 'Logged Reaction Time', y = 'Probability Density') +
  stat_function(fun = dnorm,
                args = list( mean = mean(data_filter$log_reaction_time,
                                         na.rm = TRUE),
                sd = sd(data_filter$log_reaction_time, na.rm = TRUE)),
                colour = "red", linewidth = 1),
ggplot(data_filter, aes(sample = log_reaction_time)) +
    stat_qq() +
    stat_qq_line() +
    labs(x = "Theoretical quantiles", y = "Sample quantiles"), nrow = 2)
```

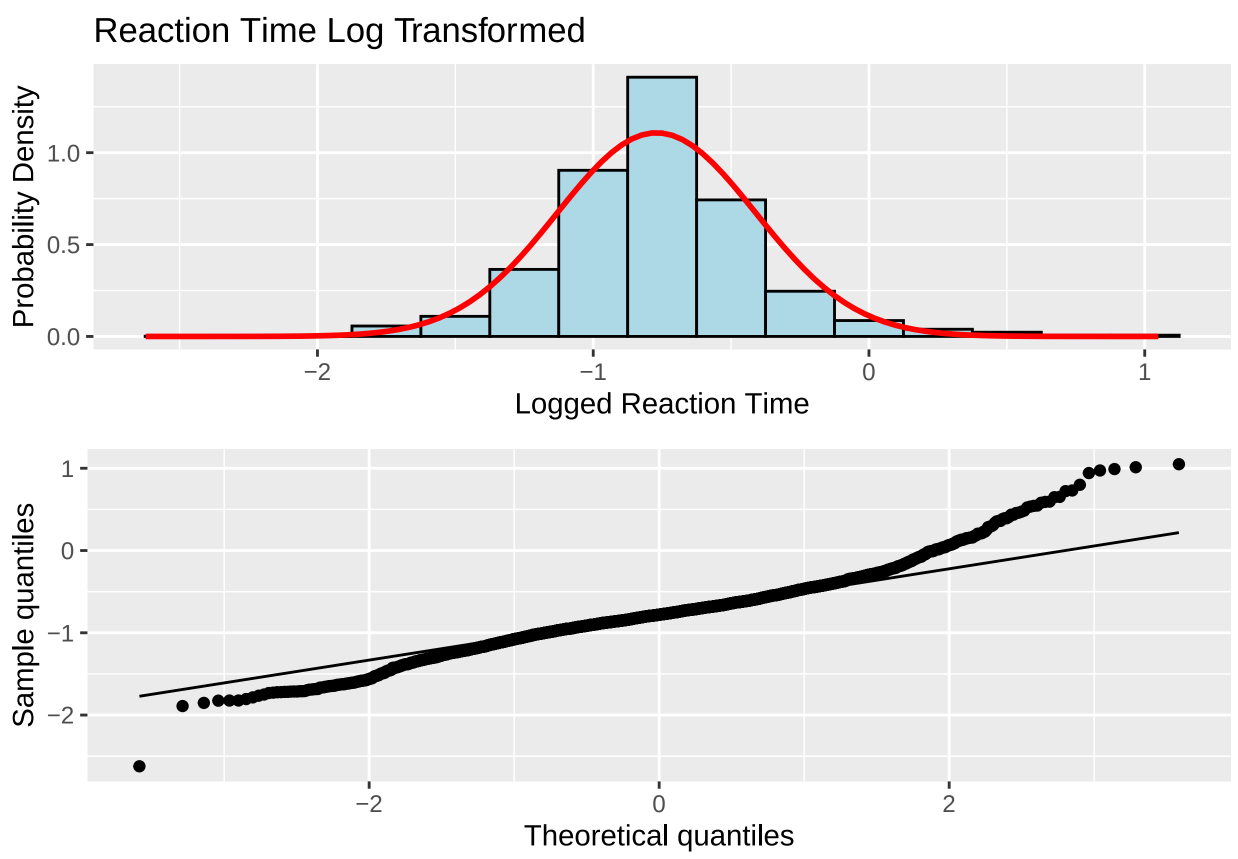


Figure 4: Logarithmic transformation probability density function and QQ-plot

Transformation using square root (MRK)

```
grid.arrange(ggplot(data_filter, aes(x = sqrt_reaction_time)) +
  geom_histogram(aes(y = after_stat(density)),
  binwidth = 0.25,
  color= 'black', fill = 'lightblue') +
  ggtitle('Reaction Time Sqrt-transformed') +
  labs(x = 'Squareroot Transformed Reaction Time', y = 'Probability Density') +
  stat_function(fun = dnorm,
                args = list( mean = mean(data_filter$reaction_time_1,
                                         na.rm = TRUE),
                sd = sd(data_filter$reaction_time_1, na.rm = TRUE)),
                colour = "red", linewidth = 1),
ggplot(data_filter, aes(sample = reaction_time_1)) +
    stat_qq() +
    stat_qq_line() +
    labs(x = "Theoretical quantiles", y = "Sample quantiles"), nrow = 2)
```

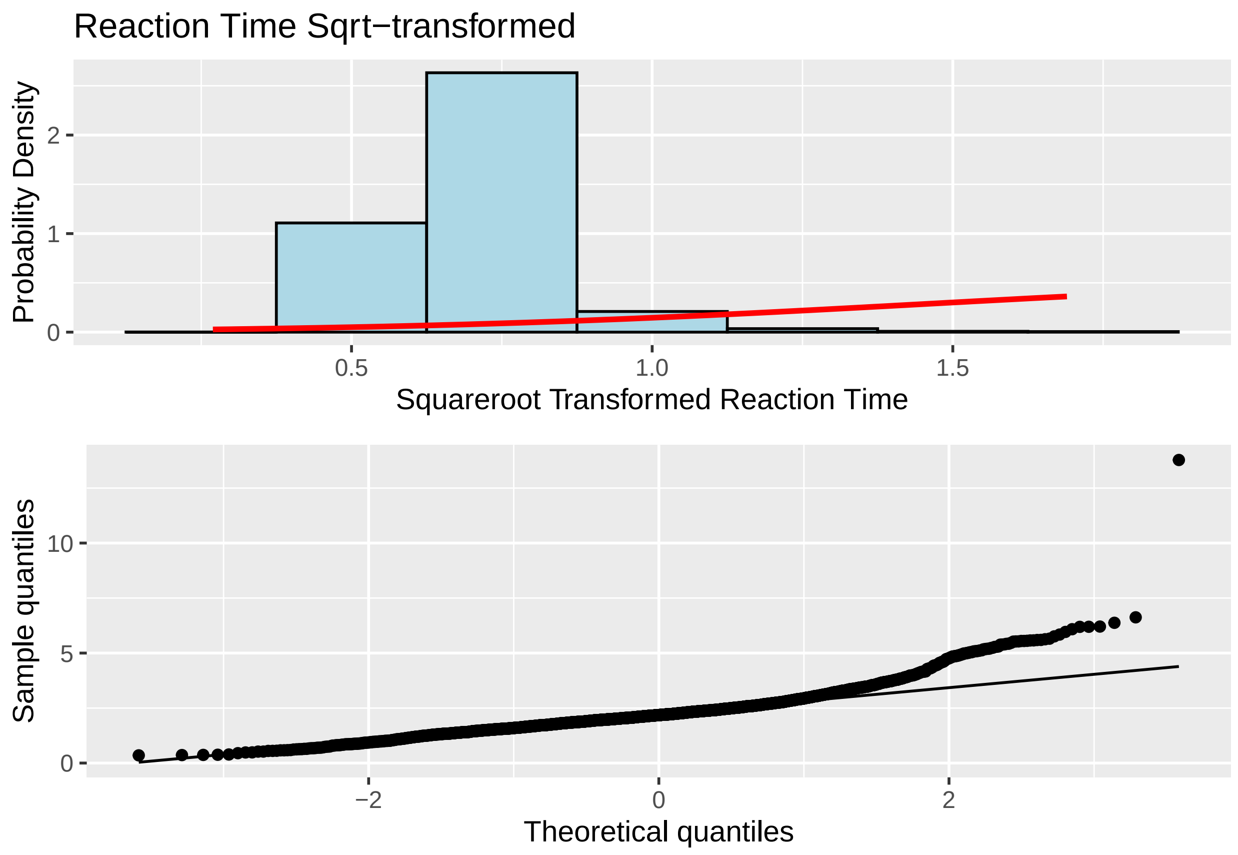


Figure 5: Square root transformation probability density function and qq-plot

Transformation using reciprocity (MRK)

```
grid.arrange(ggplot(data_filter, aes(x = reaction_time_1)) +
  geom_histogram(aes(y = after_stat(density)),
  binwidth = 0.25,
  color= 'black', fill = 'lightblue') +
  ggtitle('Reaction Time Reciprocally Transformed') +
  labs(x = 'Reciprocal Reaction Time', y = 'Probability Density') +
  stat_function(fun = dnorm,
                args = list( mean = mean(data_filter$sqrt_reaction_time,
                                         na.rm = TRUE),
                sd = sd(data_filter$sqrt_reaction_time, na.rm = TRUE)),
                colour = "red", linewidth = 1),
ggplot(data_filter, aes(sample = sqrt_reaction_time)) +
    stat_qq() +
    stat_qq_line() +
    labs(x = "Theoretical quantiles", y = "Sample quantiles"), nrow = 2)
```

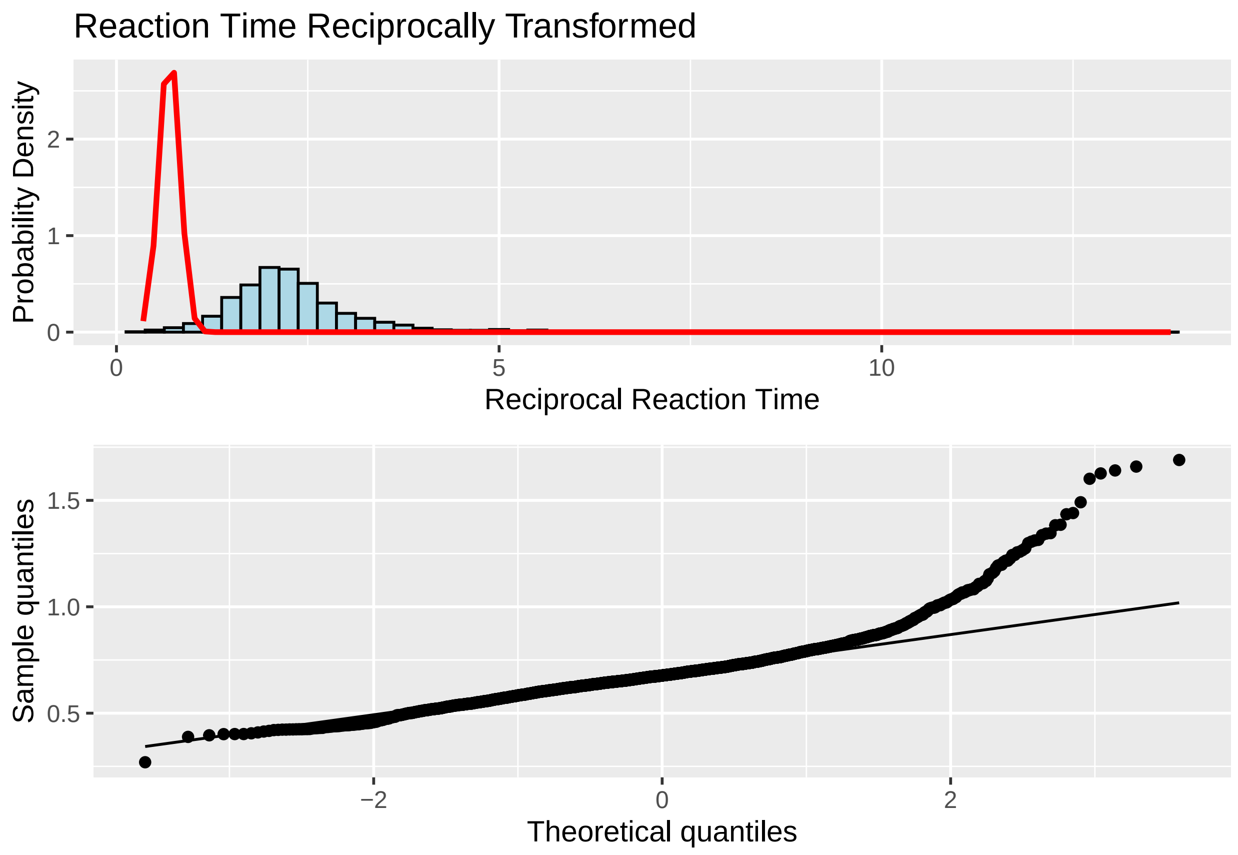


Figure 6: Probability density function and qq-plot of reciprocal transformation

None of the transformations make the distribution normal. The log transformation looks very normal on the plot, however when looking at the qq-plot and the kurtosis and skewness values we can see that it is still not normally distributed, as they are very large numbers compared to a normal distribution where the values lie closer to 1. We will therefore, have to use non-parametric tests.

Word length vs Reaction Time (MVT)

```r
#Creating a scatter plot with word length and the reaction time
ggplot(data_filter, aes(x = word_length, y = reaction_time)) +
  geom_point() +
  geom_smooth(method = lm, se=TRUE, colour = 'red') +
  labs(x = 'Word Length', y = 'Reaction Time') +
  ggtitle('Word Length vs Reaction Time') +
  theme_minimal()
```

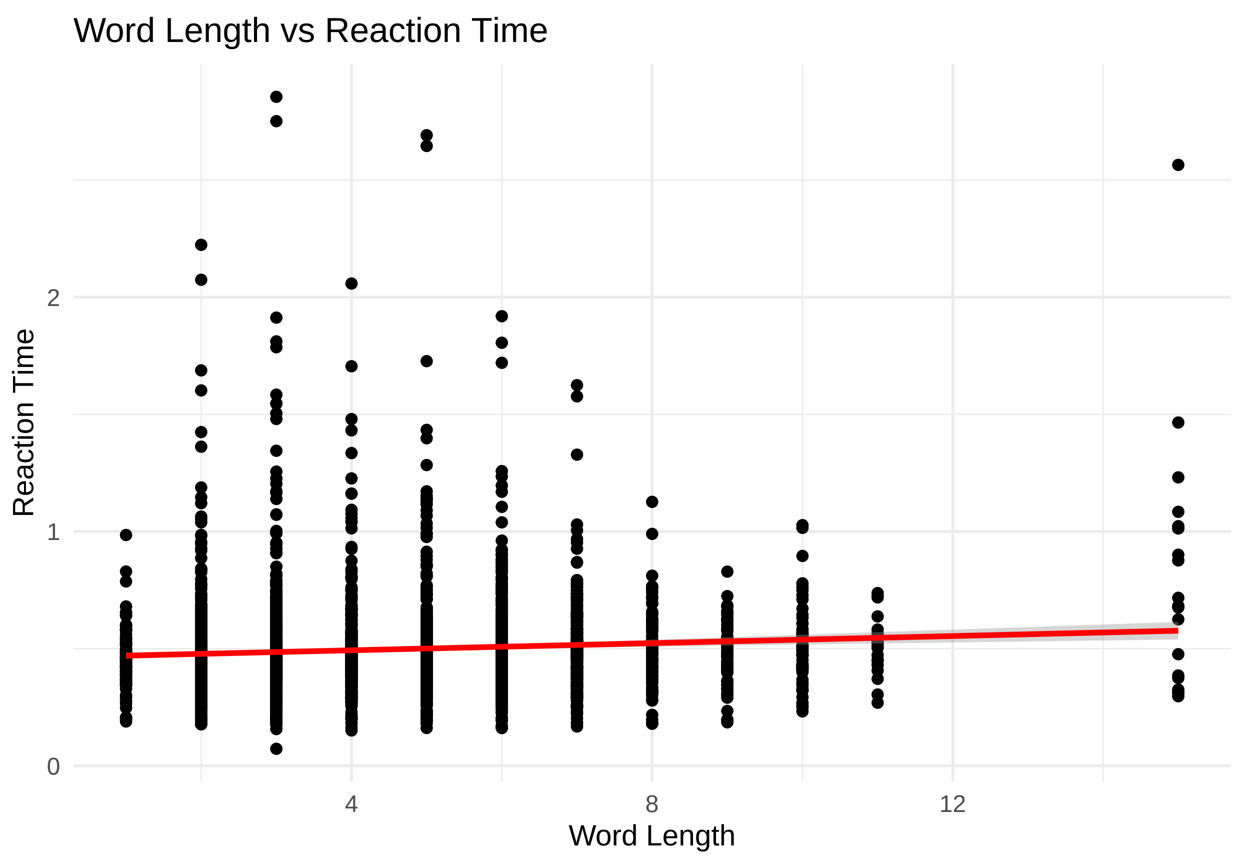


Figure 7: Scatterplot of word length and reaction time

The data does meet the assumption of linearity nor homoscedasticity as displayed visually on the scatter plot. Also, for this reason we will use a non-parametric correlation test.

As the word items are not independent, we should have aggregated in order to use a correlation test which assumes independence. In order to do this, we should have averaged the reaction time for word length, across all words and across all subjects, i.e. both within and between subjects. This would give us an average reaction time for each word length which we could use for the correlation test. This would also result in the degrees of freedom being equal to the number of words in the experiment.

Testing correlational assumption (MVT)

```
cor.test(data_filter$word_length, data_filter$reaction_time, method = 'kendall')
```

```
##
##  Kendall's rank correlation tau
##
## data:  data_filter$word_length and data_filter$reaction_time
## z = 4.0349, p-value = 5.462e-05
## alternative hypothesis: true tau is not equal to 0
## sample estimates:
##        tau
## 0.05298853
```

In this case we are using Kendall's correlation test as our data does not meet the assumptions of a parametric test, as demonstrated earlier. Kendall's correlation test is preferred over Spearman's as it is a relatively small sample with a frequent occurrence of shared ranks.

Kendall's tau(df) = .053, $p < .001$ The tau value indicates there is no correlation and the data is very spread out. The p value shows significant results, which means the tau value is not very likely to be due to chance, however we cannot make any conclusions as our effect size is very small, with a very small sample size.

Word frequency vs Reaction Time (TVB)

Preparing data (TVB)

```
#Loading in our data frame with word frequency values
wf <- read_csv("Word_frequency.csv")

#Removing punctuation from the data frame
wf$Word <- sub(",","",wf$Word)
wf$Word <- sub("\\.","",wf$Word)

#Changing the name of the second column in the word frequency data frame
names(wf)[2] <- 'frequency'

#Adding word frequencies for each word
wf <- wf %>%
  select(Word,frequency) %>%
           distinct()
data_filter <- data_filter %>%
  inner_join(wf)

#Creating data frame with mean reaction time of each word
mean_rt <- data_filter %>%
          group_by(Word) %>%
          summarise(mean_rt = mean(reaction_time), frequency) %>%
          distinct()
```

Word frequencies are derived from the MRC psycholinguistic database, and words that are not covered have been marked as NA (not applicable).

Plotting mean reaction time in scatterplot (TVB)

```
#Creating scatter plot with assigned word frequency and reaction time
ggplot(mean_rt, aes(x = frequency, y = mean_rt)) +
  geom_point() +
  geom_smooth(method = lm, colour = 'red') +
  labs(x = 'Word Frequency', y = 'Reaction Time') +
  ggtitle('Word Frequency vs Reaction Time') +
  theme(axis.text.x=element_blank())
```

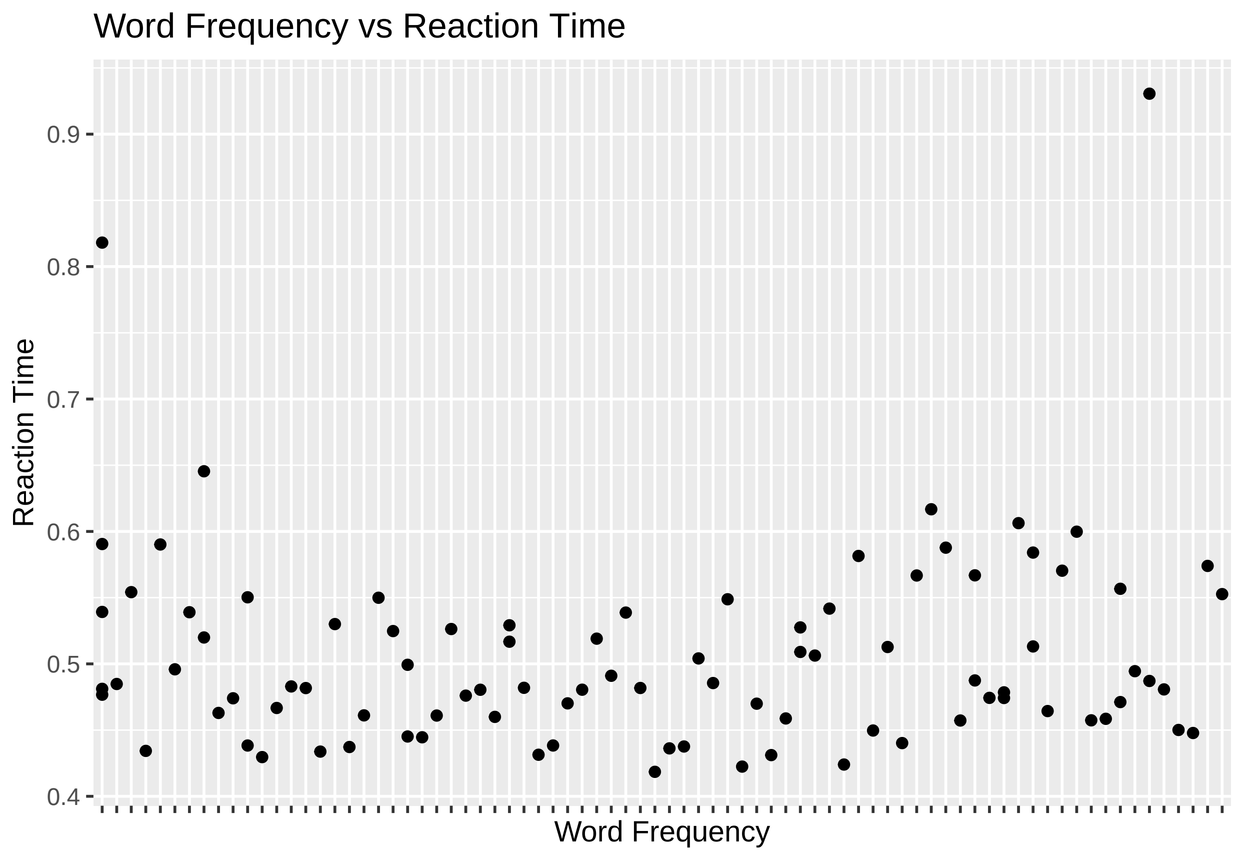


Figure 8: Scatterplot of word frequency and reaction time

As displayed by the scatterplot and inspected visually, the data does not meet the assumption of linearity. However, it could be argued that the assumption of homoscedasticity is met. We therefore use a non-parametric correlation test, as the data does not meet the assumptions or linearity and normality (as tested earlier).

In this case, we should have also averaged across all word frequencies and across all subjects. In the mean_rt data frame, we averaged across all subjects, however the next step would be to average across all word frequencies and use the new data frame for the correlation test. The degrees of freedom will then equal the different amount of word frequencies.

Testing correlational assumption (TVB)

```
cor.test(data_filter$reaction_time, as.numeric(data_filter$frequency), method =
↪  'spearman')
```

```
##
##  Spearman's rank correlation rho
##
## data:  data_filter$reaction_time and as.numeric(data_filter$frequency)
## S = 3940851927, p-value = 4.398e-05
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
```

```
##         rho
## -0.07712803
```

In this case we are using Spearman's correlation test as the data is non-parametric. r(df) = -0.077, p < .001 The rho value indicates that the data is very spread out, with a very slight negative correlation. The p-value suggests that the test is significant, however due to effect size we cannot make any conclusions.

Ordinal word number vs Reaction Time (HVEM)

```
#Creating scatter plot with ordinal word number and logged reaction time
ggplot(data_filter, aes(x = ordinal_number, y = reaction_time)) +
  geom_point() +
  geom_smooth(method = lm, se=TRUE, colour = 'red') +
  labs(x = 'Ordinal Word Number', y = 'Reaction Time') +
  ggtitle('Ordinal Word Number vs Reaction Time') +
  theme_minimal()
```

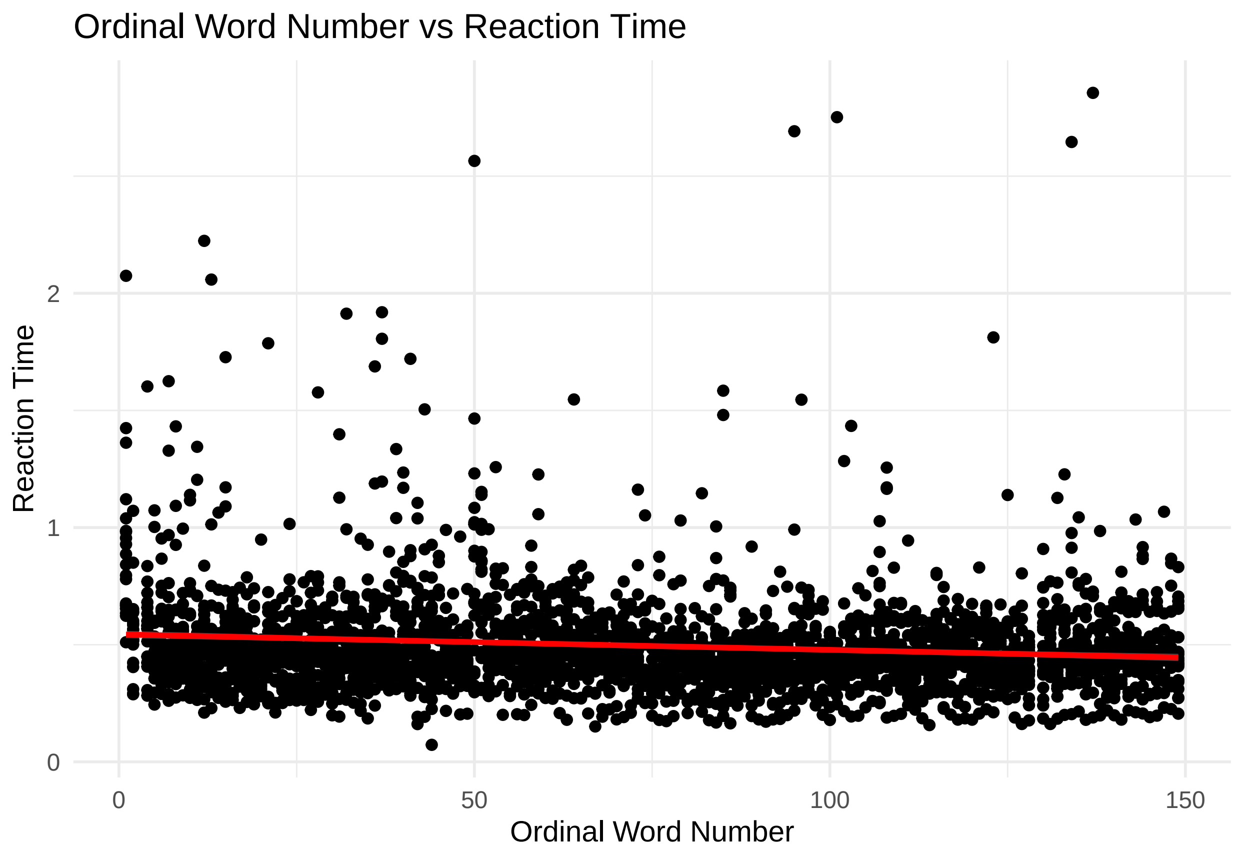


Figure 9: Scatter plot of ordinal word number and reaction time

As displayed by the scatter plot and by inspecting visually, it may be argued that the assumption of linearity and homoscedasticity is met as the data is fairly linear, and not 'funnel' shape is displayed. However, we still need to use a non-parametric test as the data did not meet the assumption of normality.

For the ordinal word number section, we do not need to average across ordinal word number as no words will have the same ordinal number, however we should aggregate across subjects, i.e. making a data frame with an average reaction time for each ordinal word number. This should then be used for the Spearman correlation test. The degrees of freedom would then be equal to the number of words in the text.

Testing correlational assumption (HVEM)

```
cor.test(data_filter$ordinal_number, data_filter$reaction_time,
         method = 'spearman')
```

```
##
##  Spearman's rank correlation rho
##
## data:  data_filter$ordinal_number and data_filter$reaction_time
## S = 4.772e+09, p-value = 5.409e-12
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##        rho
## -0.1266933
```

In this case we are using Spearman's correlation test as the data is non-parametric. r (df) = -0.127, p < .001 The rho value indicates there is no correlation (you may argue a very slight negative correlation) and the data is very spread out. The p value indicates that our correlation coefficient is statistically significant. This slight correlation suggests that people respond slightly faster to words towards the end of the text, however, we cannot make any conclusions due to our small effect size.

Hypothesis Testing

Null-hypothesis H0: there is no difference in the mean response time between the experimental and control condition **Alternative hypothesis HA:** there is a difference in the mean response time between the experimental and control condition

Making dataframes with the control, surprising & following word (in both conditions)(LD + MRK)

```
# Data frame with the control word
data_ctr_target <- filter(data, Word == "chopsticks")

# Data frame with the surprising word
data_exp_target <- filter(data, Word == "socks")

# Data frame with the following word in the control condition
data_ctr_follow <- filter(data, column_1 == 129, condition == "Control")

# Data frame with the following word in the experimental condition
data_exp_follow <- filter(data, column_1 == 129, condition == "Experimental")
```

Testing for normality (LD + MRK)

```
round(stat.desc(data_ctr_target$reaction_time, norm = TRUE),
      digits = 2)[c("skew.2SE", "kurt.2SE", "normtest.p")]
```

```
##   skew.2SE   kurt.2SE normtest.p
##       0.78      -0.04       0.07
```

```
round(stat.desc(data_ctr_follow$reaction_time, norm = TRUE),
      digits = 2)[c("skew.2SE", "kurt.2SE", "normtest.p")]
```

```
##   skew.2SE   kurt.2SE normtest.p
##       0.45      -0.45       0.23
```

```
round(stat.desc(data_exp_target$reaction_time, norm = TRUE),
      digits = 2)[c("skew.2SE", "kurt.2SE", "normtest.p")]
```

```
##   skew.2SE   kurt.2SE normtest.p
##       0.18      -0.58       0.39
```

```
round(stat.desc(data_exp_follow$reaction_time, norm = TRUE),
      digits = 2)[c("skew.2SE", "kurt.2SE", "normtest.p")]
```

```
##   skew.2SE   kurt.2SE normtest.p
##      -0.35      -0.41       0.60
```

```
grid.arrange(ggplot(data_ctr_follow, aes(sample = reaction_time)) +
    stat_qq() +
    stat_qq_line() +
    labs(x = "Theoretical quantiles", y = "Sample quantiles")+
      ggtitle("QQ-plot: Following word in control condition"),

ggplot(data_ctr_target, aes(sample = reaction_time)) +
    stat_qq() +
    stat_qq_line() +
    labs(x = "Theoretical quantiles", y = "Sample quantiles")+
  ggtitle("QQ-plot: Target word in control condition"),

ggplot(data_exp_target, aes(sample = reaction_time)) +
    stat_qq() +
    stat_qq_line() +
    labs(x = "Theoretical quantiles", y = "Sample quantiles") +
  ggtitle("QQ-plot: Target word in experimental condition"),

ggplot(data_exp_follow, aes(sample = reaction_time)) +
    stat_qq() +
    stat_qq_line() +
    labs(x = "Theoretical quantiles", y = "Sample quantiles")+
  ggtitle("QQ-plot: Following word in experimental condition"), nrow = 2)
```

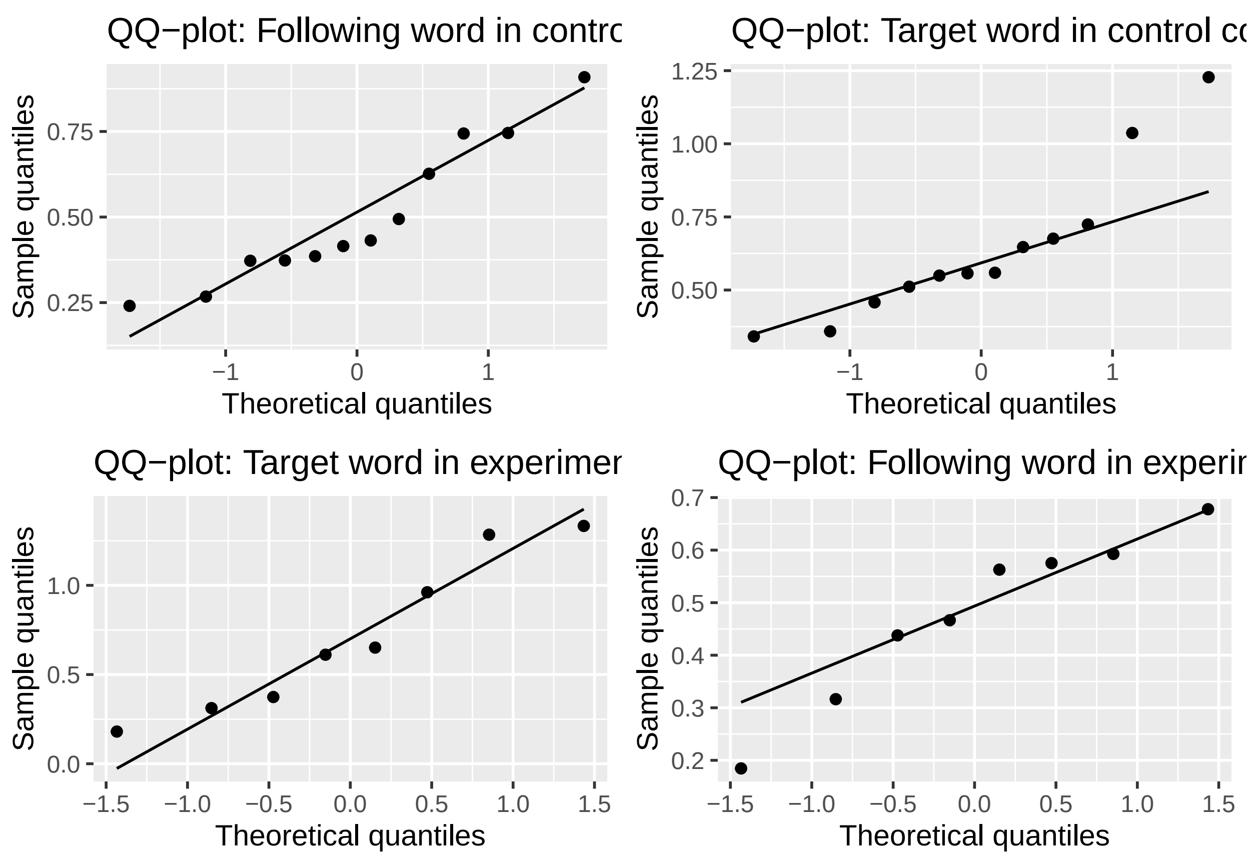


Figure 10: QQ-plots of each condition with either the target word or the following word

As p > 0.05 for each word, the sample is normally distributed and we can therefore use a normal t-test. This can also be interpreted from the skew.2SE- and kurt.2SE- values as they are close to zero, i.e., smaller than 1 and larger than -1 which indicates a normal distribution.

Testing for Homogenity of Variance (MVT)

```
df_target <- filter(data, Word == "chopsticks"| Word == "socks")
df_target$condition <- as.factor(df_target$condition)
df_following <- filter(data, column_1 == 129)
df_following$condition <- as.factor(df_following$condition)
```

```
levene.test(df_target$reaction_time, df_target$condition, location = c("mean"),
            trim.alpha = 0.25, bootstrap = TRUE, kruskal.test = FALSE)
```

```
##
##  bootstrap Classical Levene's test based on the absolute deviations from
##  the mean ( none not applied because the location is not set to median )
##
## data:  df_target$reaction_time
## Test Statistic = 3.9945, p-value = 0.114
```

```
levene.test(df_following$reaction_time, df_following$condition,
            location = c("mean"),
            trim.alpha = 0.25, bootstrap = TRUE, kruskal.test = FALSE)
```

```
##
##  bootstrap Classical Levene's test based on the absolute deviations from
##  the mean ( none not applied because the location is not set to median )
##
## data:  df_following$reaction_time
## Test Statistic = 0.92246, p-value = 0.423
```

As the p>.05 in the Levene's test for both the target word and the following word, the data does not meet the homogeneity of variance assumption, and we will therefore have to use a non-parametric t-test.

Boxplot visualising the control, surprising & following word (LD + MRK)

```
# Making a plot for the 4 words
df_plot <- filter(data, Word == "chopsticks"| Word == "socks" | Word == "and")
```

```
df_plot %>%
  ggplot(aes(x=condition, y = reaction_time, fill=condition)) +
  geom_boxplot()+
  facet_wrap(~Word)+
  scale_fill_brewer()
```

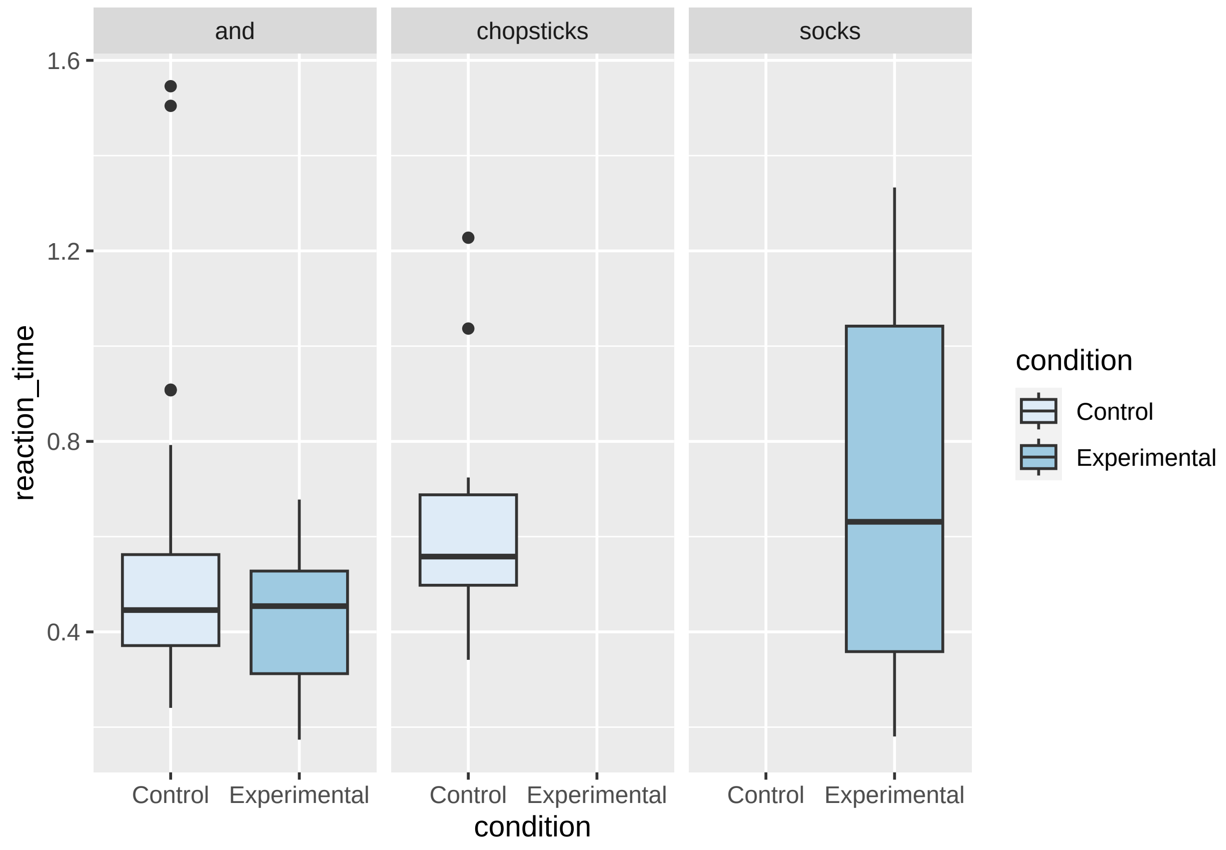


Figure 11: Boxplot visualising the control word, surprising word, and the following word

The plot indicates that all means are relatively similar, which means we might predict a non-significant t-test as the effect size is very small and barely any.

T-test (MVT)

```
# T-test for the surprising and control word
t.test(data_ctr_target$reaction_time, data_exp_target$reaction_time,
       var.equal = FALSE) #This t-test
```

```
##
##  Welch Two Sample t-test
##
## data:  data_ctr_target$reaction_time and data_exp_target$reaction_time
## t = -0.44194, df = 10.337, p-value = 0.6676
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.4586237  0.3062460
## sample estimates:
## mean of x mean of y
## 0.6371957 0.7133846
```

We found that the unexpected word did not significantly increase reading time of the word, using a Welch non-parametric t-test t (10.34) = -0.44, p > .05, r = 0.08, (M exp = 0.64, M unexp = 0.71)

```
# T-test for the following word in both conditions
t.test(data_ctr_follow$reaction_time, data_exp_follow$reaction_time,
       var.equal = FALSE) #This t-test
```

```
##
##  Welch Two Sample t-test
##
## data:  data_ctr_follow$reaction_time and data_exp_follow$reaction_time
## t = 0.28387, df = 17.474, p-value = 0.7798
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.1516480  0.1989118
## sample estimates:
## mean of x mean of y
## 0.5003803 0.4767485
```

We found that the word following the unexpected word did not significantly increase reading time of the word following the expected word, using a Welch t-test. t (17.47) = 0.28, p > .05, r = 0.02, (M ctr = 0.50, M exp = 0.48)

Conclusion

As p > 0.05 for each t-test it means we cannot reject the null hypothesis and the t-test are not statistically significant.

Portfolio Assignment 3

Marie Thomsen (202208819@post.au.dk)

2022-12-13

Age-Determination: Young Adults Are Better At Determining Age Of Their Contemporaries (Group)

Report written by Marie Vestergaard Thomsen (MVT), Matilda Rhys-Kristensen (MRK), Thomas Villads Byrum (TVB), Hugo von Essen-Müller (HVEM) & Liv Drasbek(LD)

```
data <- read_csv("age_answers.csv")

as.factor(data$condition)
```

(TVB)

```
### Cleaning the Data (TVB)
data <- data %>%
  mutate(condition = replace(condition, condition == 1, "Child")) %>%
  mutate(condition = replace(condition, condition == 2, "Young Adult")) %>%
  mutate(condition = replace(condition, condition == 3, "Adult"))
```

(TVB)

```
### Data Exploration (TVB)
grid.arrange( ggplot(data, aes(x=condition,
                 y=correct_1,
                 fill = condition)) +
  geom_boxplot() +
  guides(fill=FALSE) +
  labs(title = "Boxplot of Correct responses for each condition",
       y = "No. correct answers (+/-1)",
       x= "Condition") +
  theme_minimal()+
  scale_x_discrete(limits = c("Child", "Young Adult", "Adult")),
data %>%
  ggplot(aes(condition, correct_1, color = condition))+
  geom_point(size=2, shape=18)+
  geom_smooth(method=lm, se=TRUE)+
  scale_fill_brewer(palette ="BuPu")+
  labs( x = "Condition",
        y = "No. correct answers (+/-1)")+
  scale_x_discrete(limits = c("Child", "Young Adult", "Adult")),
ncol=2
)
```

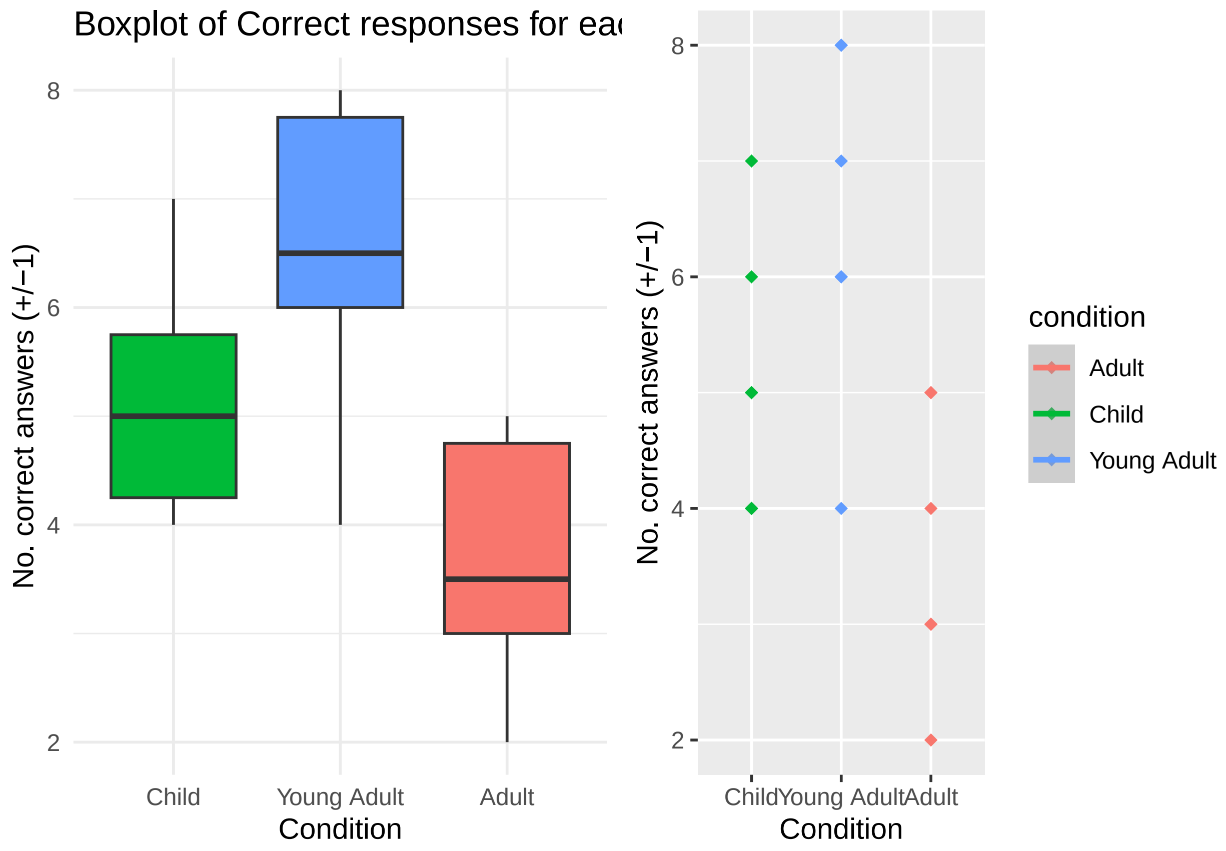


Figure 1: Boxplot of correct responses to each condition

(HVEM)

```
#running anova and storing the output
anova_model <- aov(correct_1 ~ condition, data)

#looking at the output
summary(anova_model)
```

```
##              Df Sum Sq Mean Sq F value  Pr(>F)
## condition     2  24.11  12.056   7.045 0.00696 **
## Residuals    15  25.67   1.711
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

(LD)

```
## Testing assumptions of model (LD)
# check homogeneity
bartlett.test(correct_1 ~ condition, data)


# check linearity.
# 1. make data frame with predicted versus observed values.
df <- data.frame(predicted = predict(anova_model), observed = data$correct_1)

# 2. now manually plot predicted versus observed.
ggplot(df, aes(x=predicted, y= observed)) +
  geom_point() +
  geom_abline(intercept=0, slope=1) +
  labs(x='Predicted Values for mpg', y='Actual Values for mpg', title='Predicted vs.
  ↪  Actual Values')

plot(anova_model)
m4 <- plot(anova_model,4) # check for influential cases.
```

(LD)

```
## New dataframe without outliers
datanew <- data %>%
  filter(id != "6", id !=  "9", id != "18")
```

(MRK)

```
## New ANOVA model
anova_model2 <- aov(correct_1 ~ condition, datanew)
summary(anova_model2)
```

```
##             Df Sum Sq Mean Sq F value   Pr(>F)
## condition    2  24.13   12.07   13.41 0.000873 ***
## Residuals   12  10.80    0.90
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
#Box plot to visualise ANOVA results
ggplot(datanew, aes(x = condition, y = correct_1, colour = condition)) +
  geom_boxplot(width = 0.5) +
  ggtitle("No. answer guessed correctly (+/-1)")+
  stat_summary(fun = mean, geom = "point", shape = 23, colour = "Black")
```

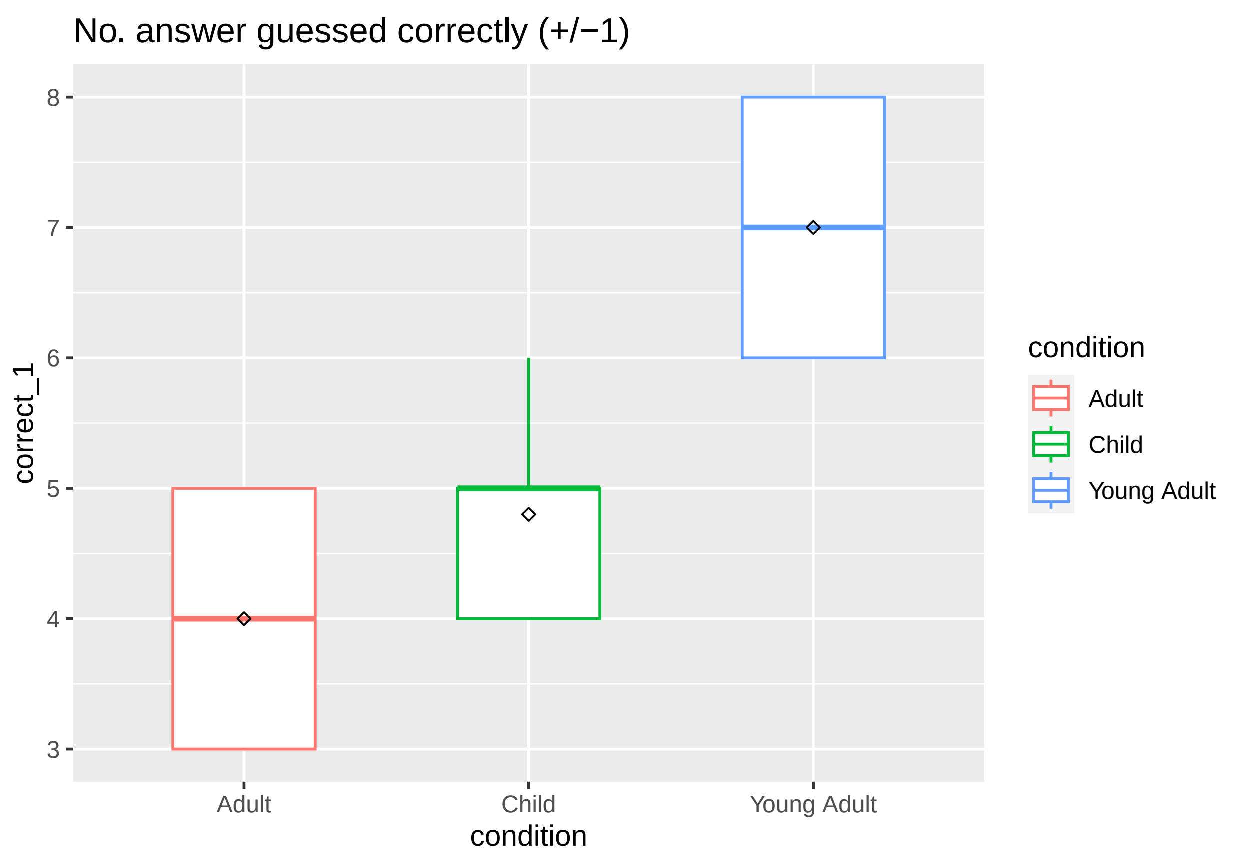


Figure 2: Box plot to visualise ANOVA results

(MRK)

```
## Testing assumptions on new ANOVA model
# check homogeneity
bartlett.test(correct_1 ~ condition, datanew)


# check linearity.
# 1. make data frame with predicted versus observed values.
df2 <- data.frame(predicted = predict(anova_model2), observed = datanew$correct_1)

# 2. now manually plot predicted versus observed.
ggplot(df2, aes(x=predicted, y= observed)) +
  geom_point() +
  geom_abline(intercept=0, slope=1) +
  labs(x='Predicted Values for mpg', y='Actual Values for mpg', title='Predicted vs.
  ↪  Actual Values')

plot(anova_model2)
```

(MVT)

```
#### New dataframes for assumption testing
data_child <- datanew %>%
  filter(condition == "Child")

data_ya <- datanew %>%
  filter(condition == "Young Adult")

data_adult <- datanew %>%
  filter(condition == "Adult")
```

```
car::leveneTest(datanew$correct_1, datanew$condition, center = mean)
```

```
## Levene's Test for Homogeneity of Variance (center = mean)
##       Df F value Pr(>F)
## group  2  0.2177 0.8075
##       12
```

(MVT, TVB, HVEM)

```
#Histogram and QQ-plot for Children
grid.arrange(data_child %>%
  ggplot(aes(x = correct_1)) +
  geom_histogram(aes(y = ..density..), binwidth = 1, color = "black", fill = "white") +
  stat_function(fun = dnorm, args = list(mean = mean(data_child$correct_1), sd =
  ↪  sd(data_child$correct_1)), color = "indianred1", size = 1)+
  labs(title = "Child Histogram", x="Correct_1") +
  theme_bw(),
data_child %>%
  ggplot(aes(sample = correct_1)) + stat_qq() +
  stat_qq_line(colour = "indianred1") +
  labs(x = "Theoretical Quantiles", y = "Sample Quantiles") + ggtitle("Child (QQ)") +
  theme_bw(), ncol = 2)
```

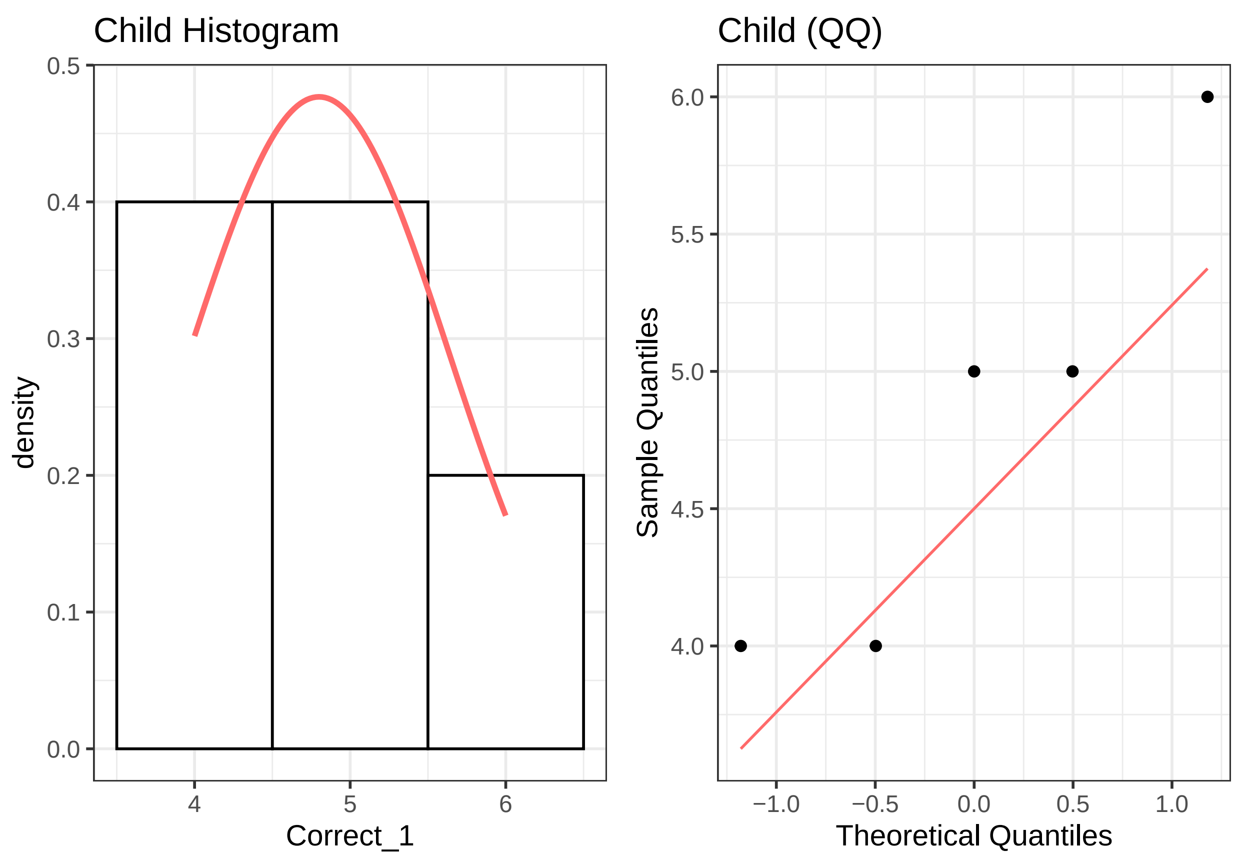


Figure 3: Histogram and QQ-plot for Children condition

```
#Histogram and QQ-plot for Young Adults
grid.arrange(data_ya %>%
  ggplot(aes(x = correct_1)) +
  geom_histogram(aes(y = ..density..), binwidth = 1, color = "black", fill = "white") +
  stat_function(fun = dnorm, args = list(mean = mean(data_ya$correct_1), sd =
  ↪  sd(data_ya$correct_1)), color = "darkturquoise", size = 1) +
  labs(title = "Young Adult Histogram", x="Correct_1") +
  theme_bw(),
  data_ya %>%
  ggplot(aes(sample = correct_1)) + stat_qq() +
  stat_qq_line(colour = "darkturquoise") +
  labs(x = "Theoretical Quantiles", y = "Sample Quantiles") + ggtitle("Young Adult (QQ)")
  ↪  +
  theme_bw(), ncol = 2)
```

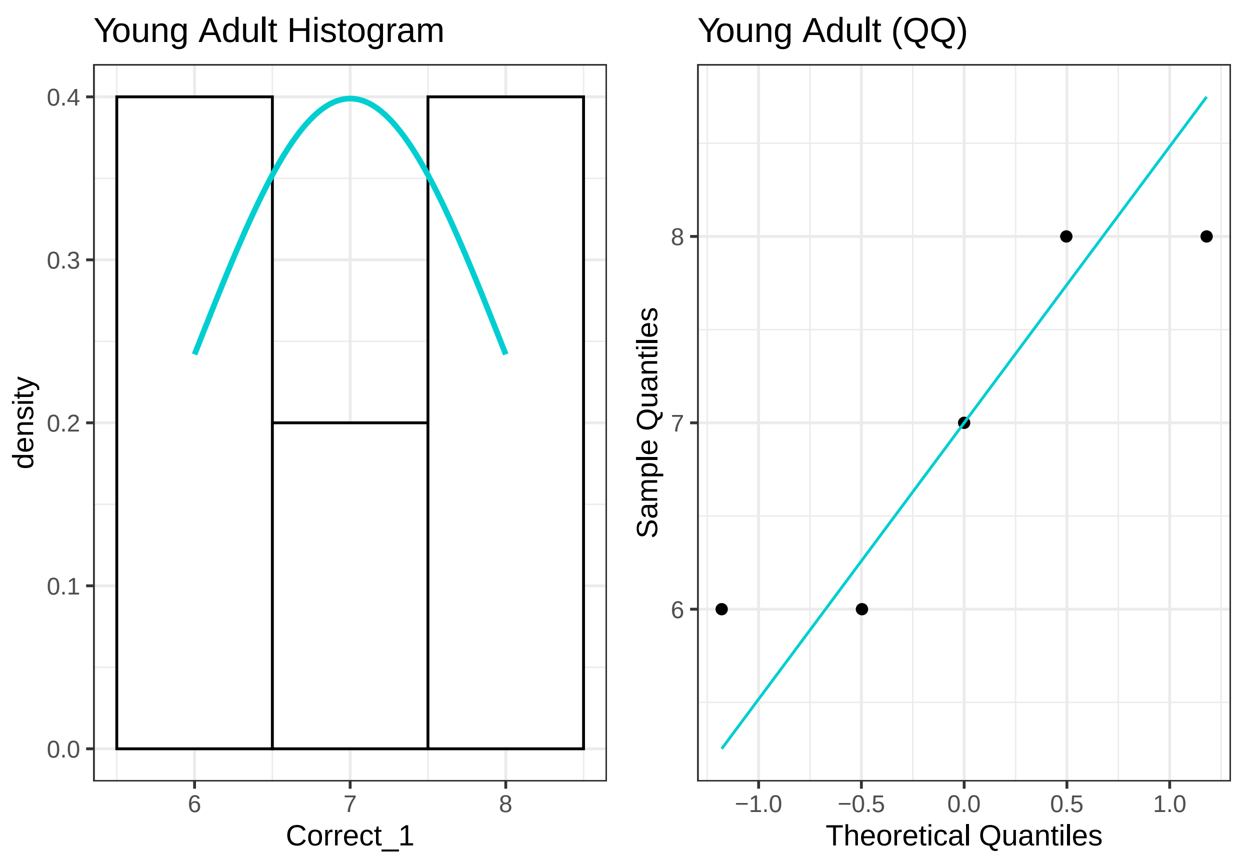


Figure 4: Histogram and QQ-plot for Young Adults condition

```
#Histogram and QQ-plot for adults
grid.arrange(data_adult %>%
  ggplot(aes(x = correct_1)) +
  geom_histogram(aes(y = ..density..), binwidth = 1, color = "black", fill = "white") +
  stat_function(fun = dnorm, args = list(mean = mean(data_adult$correct_1), sd =
  ↪  sd(data_adult$correct_1)), color = "darkorchid1", size = 1) +
  labs(title = "Adult Histogram", x ="Correct_1") +
  theme_bw(),
qq_adult <- data_adult %>%
  ggplot(aes(sample = correct_1)) + stat_qq() +
  stat_qq_line(colour = "darkorchid1") +
  labs(x = "Theoretical Quantiles", y = "Sample Quantiles") + ggtitle("Adult (QQ)") +
  theme_bw(), ncol = 2)
```

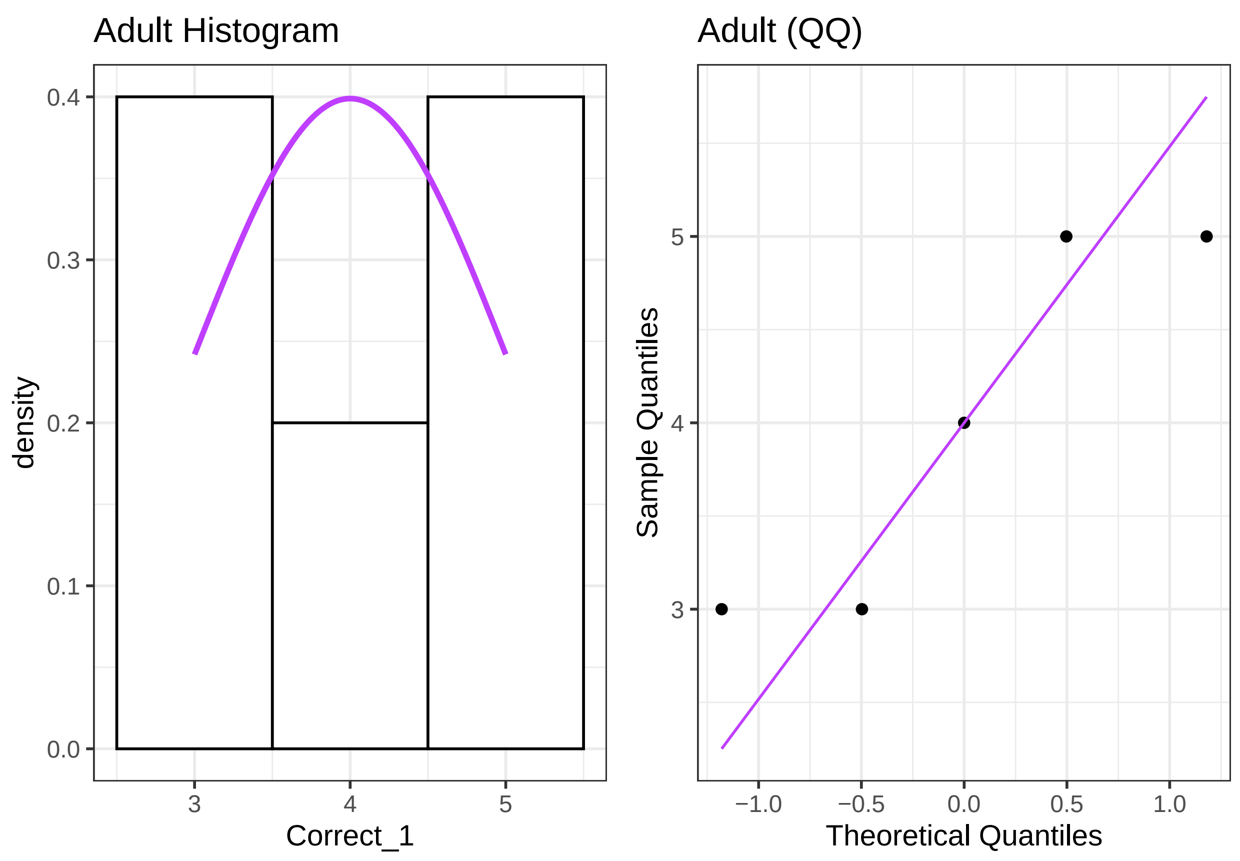


Figure 5: Histogram and QQ-plot for Adults

```
#Checking for normality within conditions numerically
round(pastecs::stat.desc(cbind("Child" = data_child$correct_1, "Young Adult" =
↪  data_ya$correct_1, "Adult" =  data_adult$correct_1), basic = FALSE, norm = TRUE),
↪  digits = 3)[c("skew.2SE", "kurt.2SE", "normtest.p"), ]
```

```
##              Child Young Adult  Adult
## skew.2SE     0.135       0.000  0.000
## kurt.2SE    -0.454      -0.550 -0.550
## normtest.p   0.314       0.119  0.119
```

```
shapiro.test(data_child$correct_1)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  data_child$correct_1
## W = 0.88104, p-value = 0.314
```

```
shapiro.test(data_ya$correct_1)
```

```
##
```

```
##  Shapiro-Wilk normality test
##
## data:  data_ya$correct_1
## W = 0.82083, p-value = 0.1185
```

```
shapiro.test(data_adult$correct_1)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  data_adult$correct_1
## W = 0.82083, p-value = 0.1185
```

```
# Shapiro wilkes result: *p* > .05 for all conditions meaning that the data is normally
↪  distributed.

leveneTest(correct_1 ~ condition, datanew )
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##       Df F value Pr(>F)
## group  2  0.2857 0.7564
##       12
```

```
# the levene test shows that *p*>.05 (*p* = 0.76) so there is similar variance across
↪  conditions
```

(LD)

```
## Posthoc tests
# Our model:
summary(anova_model2)
```

```
##             Df Sum Sq Mean Sq F value   Pr(>F)
## condition    2  24.13   12.07   13.41 0.000873 ***
## Residuals   12  10.80    0.90
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Benferroni correction:

t.test <- pairwise.t.test(datanew$correct_1, datanew$condition, p.adjust.method =
↪  "bonferroni")
t.test_pvals <- t.test$p.value

compared_conditions <- c("Adult - Child", "Adult - Young Adult", "Child - Young Adult")
anova_pvalues <- c(t.test_pvals[1], t.test_pvals[2], t.test_pvals[4])

t.test_results <- data.frame(compared_conditions, anova_pvalues)
```

```
t.test_results$anova_pvalues <- round(anova_pvalues, 7)

t.test_results <- t.test_results %>%
  rename("Compared Conditions" = compared_conditions,
         "P-value" = anova_pvalues)

t.test_results
```

```
##   Compared Conditions   P-value
## 1       Adult - Child 0.6215430
## 2 Adult - Young Adult 0.0009279
## 3 Child - Young Adult 0.0096798
```

(LD)

```
## Visualising results of different conditions
mean(data_child$correct_1)
mean(data_ya$correct_1)
mean(data_adult$correct_1)

sd(data_child$correct_1)
sd(data_ya$correct_1)
sd(data_adult$correct_1)
```

(MRK, MVT)

```
grid.arrange(
  # Child Bar Chart
  data_child %>%
  ggplot(aes(x= correct_1))+
  geom_bar(binwidth = 1, fill = "lightblue3")+
  labs( x = "Total no. correct answers")+
  scale_y_discrete(name ="Number of participants",
                    limits=c("1","2"))+
  scale_x_discrete(name ="Total no. correct answers",
                    limits=c("1","2","3","4","5","6","7","8","9","10","11","12"))+
  ggtitle("Child Condition"),

# Young Adult Bar Chart
data_ya %>%
  ggplot(aes(x= correct_1))+
  geom_bar(binwidth = 1, fill = "lightblue3")+
  labs( x = "Total no. correct answers")+
  scale_y_discrete(name ="Number of participants",
                    limits=c("1","2"))+
  scale_x_discrete(name ="Total no. correct answers",
                    limits=c("1","2","3","4","5","6","7","8","9","10","11","12"))+
  ggtitle("Young Adult Condition"),
# Adult Bar Chart
data_adult %>%
  ggplot(aes(x= correct_1))+
  geom_bar(binwidth = 1, fill = "lightblue3")+
  labs( x = "Total no. correct answers")+
  scale_y_discrete(name ="Number of participants",
                   limits=c("1","2"))+
  scale_x_discrete(name ="Total no. correct answers",
                   limits=c("1","2","3","4","5","6","7","8","9","10","11","12"))+
  ggtitle("Adult Condition"),
ncol =3)
```

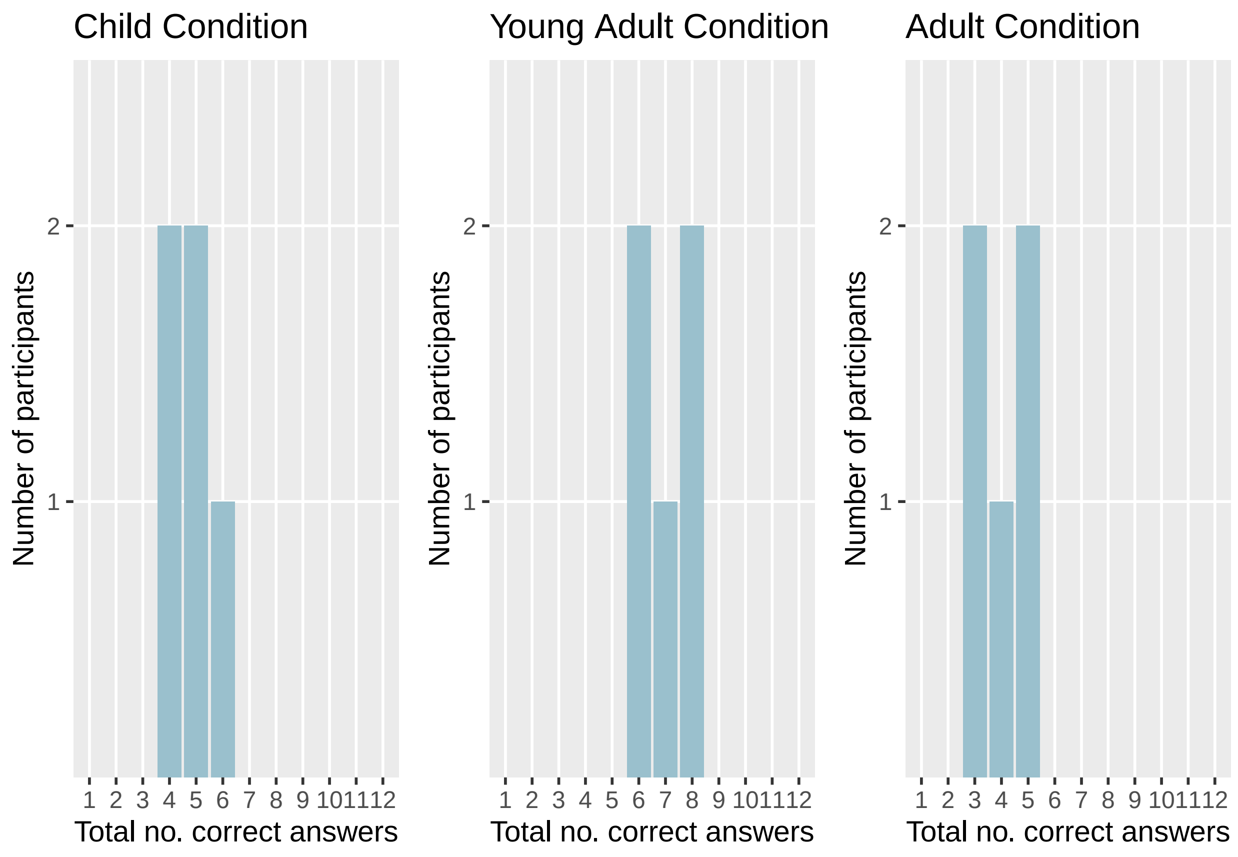


Figure 6: Total number of correct answers in each condition

```
# All conditions Bar Chart
data%>%
  ggplot(aes(x= correct_1, fill = condition))+
  geom_bar(binwidth = 1, position = "dodge")+
  labs(x = "Total no. correct answers")+
  scale_y_discrete(name ="Number of participants",
                   limits=c("1","2"))+
  scale_x_discrete(name ="Total no. correct answers",
                   limits=c("1","2","3","4","5","6","7","8","9","10","11","12"))+
  ggtitle("Bar plot showing how many participants have gotten ")+
  scale_fill_brewer(palette ="BuPu")+
  theme_minimal()
```

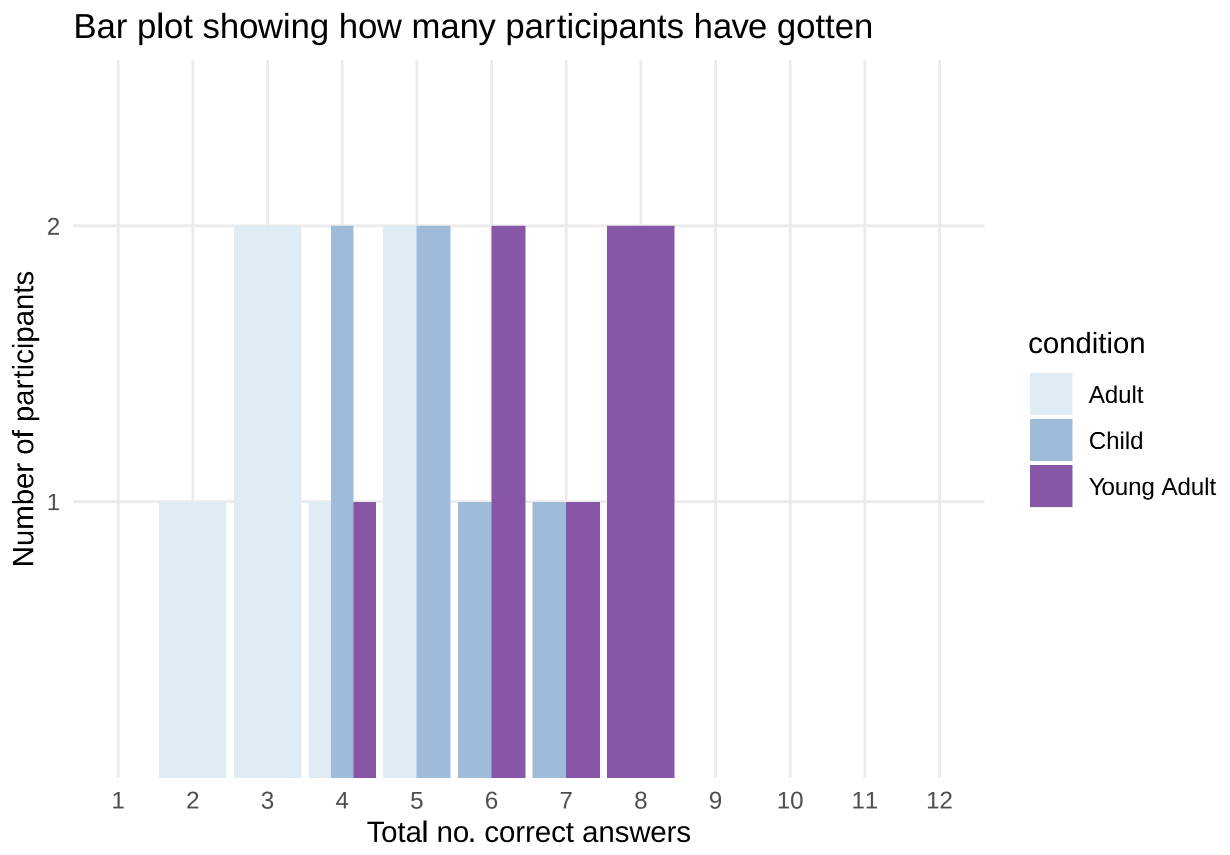


Figure 7: All conditions shown in one bar graph

Introduction (HVEM + LD + TVB)

Many have tried to be in an awkward situation where we are not sure about another person's age. As well as being identified as being younger or older than we are. However, perhaps we are better at determining the age of our own contemporaries and as a result can avoid these problems? In 2012, M. Voelkle et al. explored whether age had an impact on age determination. They found that young adults and older adults were more accurate in determining ages like their own, than with other age groups (Voelkle et al., 2012). We found this topic interesting, as we have recently been placed in new social dynamics with new people at university, where age has become less relevant than before. This study investigates whether one's own age plays a role when determining the age of others. Our participants were of the ages 20-24, as this is the age of our peers and therefore our most immediate participant source. In addition, this group performed well in the previously mentioned study, from which we drew inspiration. In the experiment, participants were presented with one of three conditions, these conditions included pictures of faces. Participants were asked to estimate the age of the face in the picture. We state the following hypothesis: If people are better at guessing the age of their contemporaries, then young adults should be better at determining the age of young adults, than that of children and adults.
Null hypothesis: All means for each condition are equal. Alternative hypothesis: At least one of the means between the conditions are different.

Methods

Materials/Stimuli (LD + HVEM)

Our material consisted of 36 pictures of faces from people aged 5-40 of equal amount of male and female gender as well as multiple ethnicities. The faces were taken from UTKFace: Large Scale Face Dataset (UTKFace, u.å.). These pictures were divided into three groups, being our conditions for the experiment: condition 1: Children (ages 5-16), condition 2: Young adults (ages 17-28) and condition 3: Adults (ages 29-40). The ages within each category were equally distributed around their median.

Equipment (TVB)

We used a survey format for the study with Google Forms. Thereafter participants were asked to use their own laptops or a borrowed one from a researcher with a link to the survey. Our statistical analysis was done through Rstudio.

Procedure (TVB, HVEM)

First, a total of 18 participants were recruited for the study. The participants are of similar age to those in the images in the Young Adult condition (M-age=21.8. range=20-24. SD=1.17). They were randomly assigned to one of the surveys containing one of the three conditions. Then informed about the general task and the age-range fitting this exact condition. Then they were presented with one picture at a time and asked to write their guessed age under each of the visual stimuli. This procedure was repeated 18 times with different participants. The data was collected in an excel-sheet. We plotted the data in the different conditions based on the number of accurate guesses (with an accurate guess being the correct age, +/-1 year). We decided to determine accuracy based on the guessed age +/- 1, as we predicted this would give more accurate results given the nature of age.

Analysis (HVEM + TVB)

Due to the use of multiple conditions and between subject design we created an anova model. As a result, we had to check if the criteria could be fulfilled, for us to use anova. With our original data we checked assumptions and outliers with Cook's Distance for influential cases and found that data points 6, 9 and 18 were outliers. We subsequently made a new data frame without these outliers and ran the assumptions tests again.

Results (MVT + MRK)

Our graphs show that there were more participants who got a higher total number of correct answers in the Young Adult condition than in the other conditions (Young Adult: M=7, SD=1 - Child M=4.8, SD=0.84 - Adult: M=4, SD=1). The Adult condition was the condition where the participants generally got a lower total correct number of answers. We removed outliers, data points 6, 9 and 18 from our data and made a new anova model with our new data frame. Our new anova model: aov(correct_1 ~ condition, datanew)

We tested for normality before running a bonferroni correction t-test and found that using a shapiro wilkes test p>.05 for all conditions, meaning that the data is normally distributed. This can also be seen through visual inspection of QQ plots and historgrams. We also conducted a levene's test to check for equal variance across conditions. Our levene test shows that p>.05 so there is similar variance across conditions.

Our ANOVA model shows that there is a significant effect of age group images on the number of answers participants guessed correctly: F(2,12)=13.41, p<.001. As a result, we needed to compute a post hoc test to find out which of the groups significantly differ from each other.

As our data was normally distributed the individual pairwise effects were tested with post-hoc Bonferroni-corrected t-tests. The t-test indicates that there is a significant effect of the different age group images on accuracy in guessing age $F(2,12) = 13.41$, $p<.001$. The test revealed that the accuracy was significantly lower in the Adult and Child condition than in the Young Adult condition ($p<.001$).

The test revealed that there was not a significant difference in the Child-Adult ($p>.05$) conditions, and we cannot assume that the variation is anything other than chance. However, there was a significant difference when comparing Child-Young Adult conditions ($p<.01$) and an especially significant result comparing Adult-Young Adult conditions ($p<.001$). As a result of these significant results satisfying our alpha level, we can assume that the variations are not due to random chance.

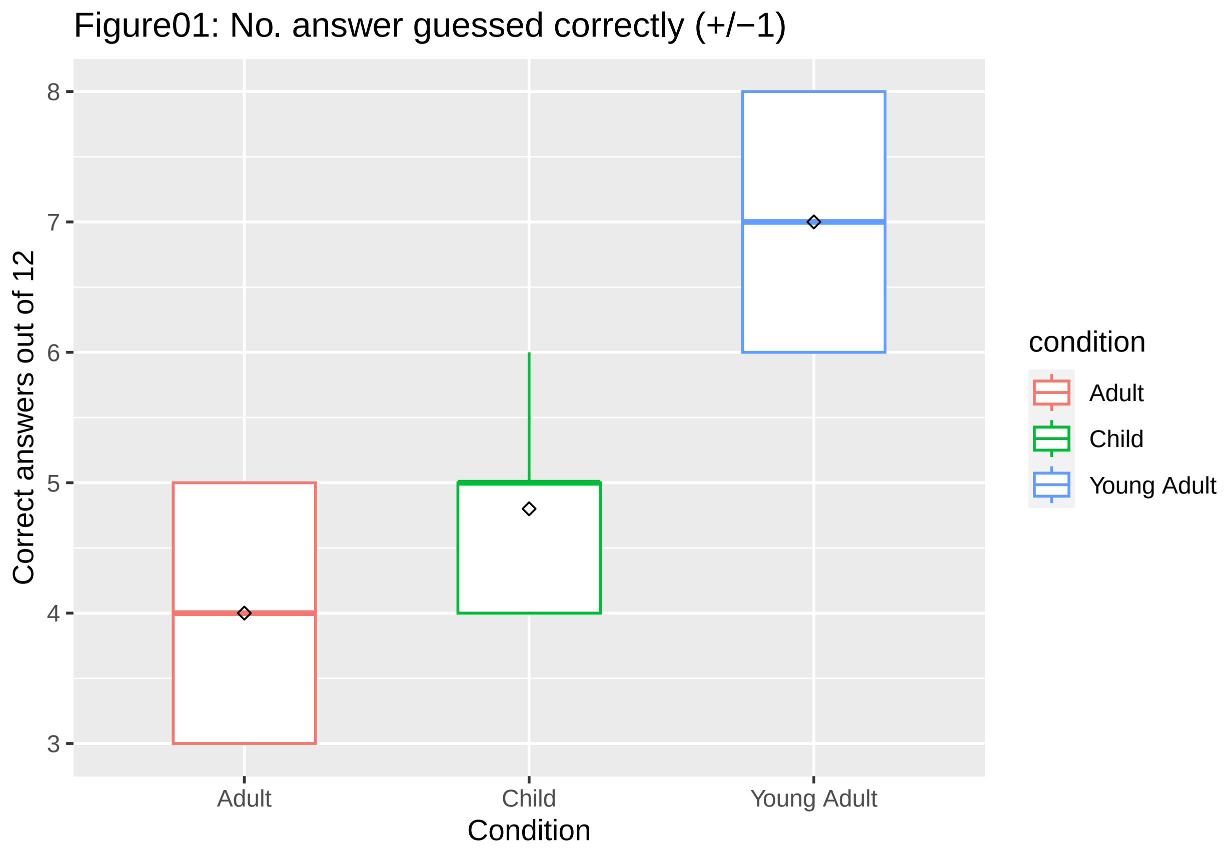


Figure 8: Boxplot visualising ANOVA results

```
##             Df Sum Sq Mean Sq F value   Pr(>F)
## condition    2  24.13   12.07   13.41 0.000873 ***
## Residuals   12  10.80    0.90
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##   Compared Conditions   P-value
## 1       Adult - Child 0.6215430
## 2 Adult - Young Adult 0.0009279
## 3 Child - Young Adult 0.0096798
```

Discussion (LD)

Our results suggest that young adults are in fact better at identifying the age of their contemporaries compared to the ages of younger and older people. Previous research found a similar correlation (Voelkle et al., 2012). Our results reflect the own-age bias, which is that people are often better at predicting the age of other people's faces when they are close to their own age (Moyse & Brédart, 2012). Different factors may affect our perception of age. For example, context (Pilz & Lou, 2022) or physical social exposure of certain age groups (He et al., 2011). We are often in groups with people of the same age group e.g., school & work. This could be one of the possible explanations of our results and own-age bias. For further research you could investigate the different factors influencing age perception further; perhaps the rise in social media and non-physical images could affect newer generations' abilities to perceive age. A limitation is our sample size, which is less than desirable for establishing statistically significant evidence. Furthermore, all our participants were WEIRD; western, educated, industrialized, rich and democratic (Henrich et al., 2010) - this is not representative of our world population. However, our results are still interesting, as they indicate a positive correlation between a person's age and age determination.

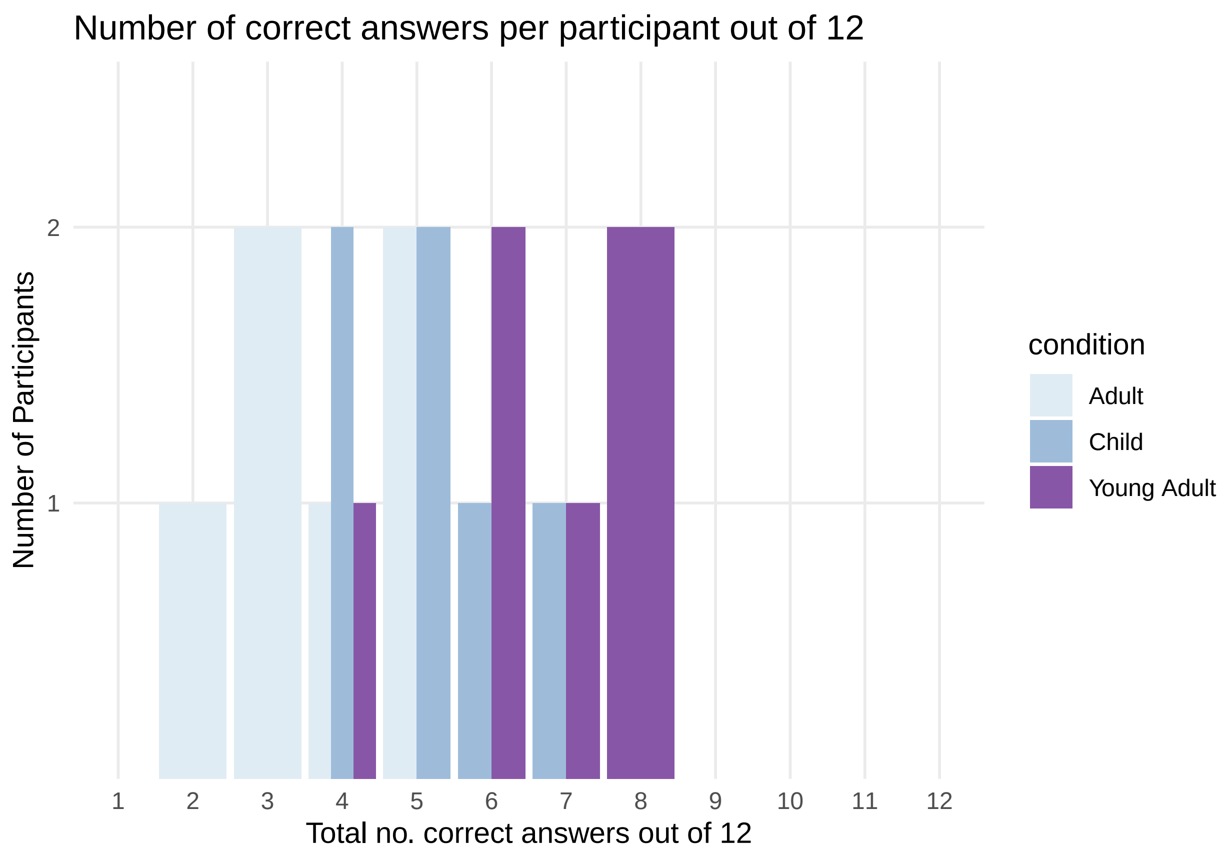


Figure 9: Total number of correct answers for each condition

References

He, Y., Ebner, N. C., & Johnson, M. K. (2011). What Predicts the Own-Age Bias in Face Recognition Memory? Social Cognition, 29(1), 97–109. https://doi.org/10.1521/soco.2011.29.1.97

Henrich, J., Heine, S. J., & Norenzayan, A. (2010). Most people are not WEIRD. Nature, 466(7302), Art. 7302. https://doi.org/10.1038/466029a

Moyse, E., & Brédart, S. (2012). An own-age bias in age estimation of faces. European Review of Applied Psychology, 62(1), 3–7. https://doi.org/10.1016/j.erap.2011.12.002

Pilz, K. S., & Lou, H. (2022). Contextual and own-age effects in age perception. Experimental Brain Research, 240(9), 2471–2480. https://doi.org/10.1007/s00221-022-06411-w

UTKFace. (u.å.). UTKFace. Hentet 7. december 2022, fra https://susanqq.github.io/UTKFace/

Voelkle, M. C., Ebner, N. C., Lindenberger, U., & Riediger, M. (2012). Let me guess how old you are: Effects of age, gender, and facial expression on perceptions of age. Psychology and Aging, 27, 265–277. https://doi.org/10.1037/a0025065

Portfolio Assignment 4

Marie Thomsen (202208819@post.au.dk)

2022-12-13

Assignment 4: Mixed-effects models and logistic regression (Individual)

```r
knitr::opts_chunk$set(echo = TRUE, include = TRUE, message = FALSE, warning = FALSE)
knitr::opts_knit$set(root.dir =
↪  '/work/CogSci_Methods01/portfolio_assignment_04-marievthomsen')
```

```r
#Loding the data
df_cake <- read.csv('../data/cake.csv')
df_titanic <- read.csv('../data/titanic.csv')
```

Analysis 1: Linear Mixed Model

This analysis was created using R (R Core Team, 2019) and lmerTest (Kuznetsova, Brockhoff and Christensen, 2017) to produce a linear mixed effects analysis showing the relationship between the angle of breakage and the temperature the cake was baked at. The temperature at which the cake was baked at, was used as fixed effects. In the model, a random slope and intercept was used. The random intercept was replicate, and the random slope was recipe. This model was compared against the null model and displayed a better fit with a lower AIC value. The syntax of the model was as follows:

angle ~ temp + (1 + recipe | replicate)

Both fixed and random effects accounted for approximately 70% of variance. When looking at the plots of the residuals, it suggests that the assumptions of homoscedasticity and linearity are met. The angle of breakage significantly inflects the temperature at which the cake was baked at, (U+02B2) = 0.158, SE = 0.016, t(~240) = 9.8, p < .001.

```r
#Trying out various models in order to test for the best one with the lowest AIC value
m01 <- lmerTest::lmer(angle ~ temp + (1|replicate), data = df_cake, REML = F)
m02 <- lmerTest::lmer(angle ~ recipe + (1|replicate), data = df_cake, REML = F)
m03 <- lmerTest::lmer(angle ~ recipe + (1+recipe|replicate), data = df_cake, REML = F)
m04 <- lmerTest::lmer(angle ~ replicate + (1|replicate) + (1|recipe), data = df_cake,
↪  REML = F)
m05 <- lmerTest::lmer(angle ~ temp + (1 +recipe|replicate), data = df_cake, REML = F)
↪  #Model 5 showed the lowest AIC, and therefore is the best model
m06 <- lmerTest::lmer(angle ~ temp + replicate + (1|replicate), data = df_cake, REML = F)
m07 <- lmerTest::lmer(angle ~ recipe * temp + (1|recipe:replicate), data = df_cake, REML
↪  = F)
null <- lmerTest::lmer(angle ~ (1|recipe), data=df_cake, REML = F)
```

```
summary(m01)
summary(m02)
summary(m03)
summary(m04)
summary(m05)
summary(m06)
summary(m07)
```

```
anova <- anova(m01, m02, m03, m04, m05, m06, m07, null)
```

```
MuMIn::r.squaredGLMM(m05)
```

```
##            R2m       R2c
## [1,] 0.1062652 0.7023687
```

```
summary(m05)
```

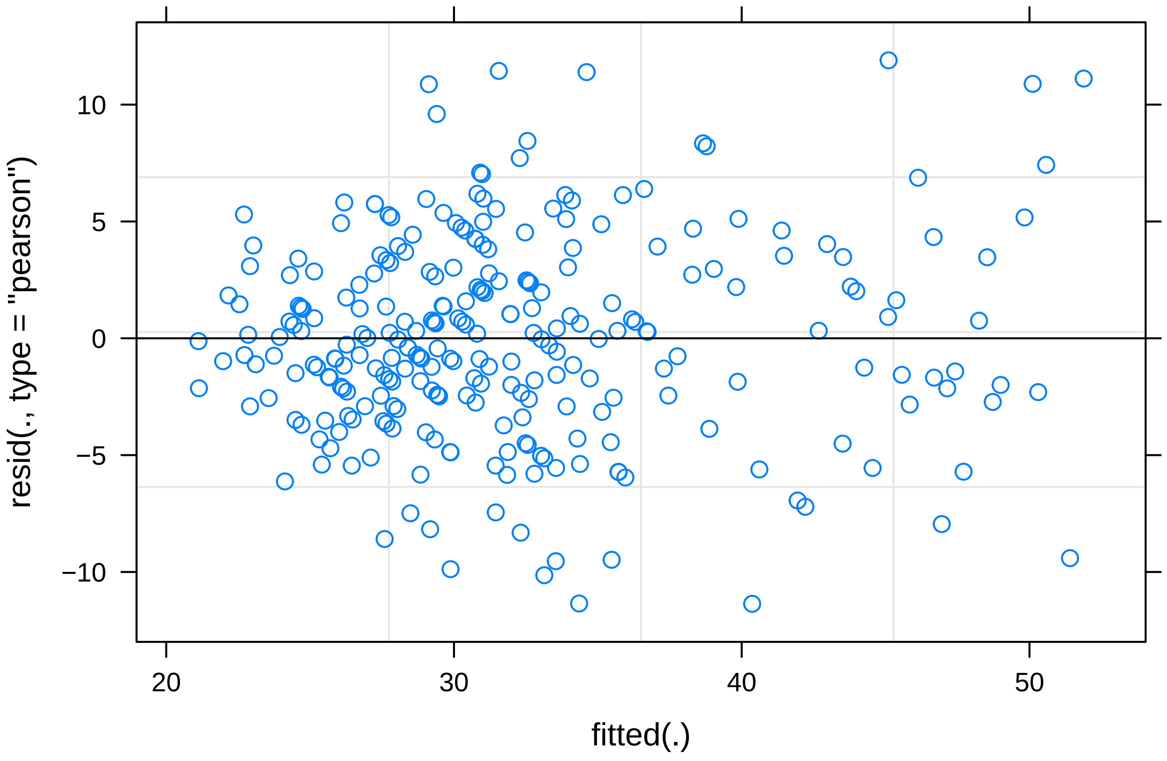
```
## Linear mixed model fit by maximum likelihood . t-tests use Satterthwaite's
##   method [lmerModLmerTest]
## Formula: angle ~ temp + (1 + recipe | replicate)
##    Data: df_cake
##
##      AIC      BIC   logLik deviance df.resid
##   1666.2   1698.6   -824.1   1648.2      261
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -2.51095 -0.56465 -0.01979  0.62483  2.62895
##
## Random effects:
##  Groups    Name        Variance Std.Dev. Corr
##  replicate (Intercept) 24.981   4.998
##            recipeB      8.513   2.918    0.42
##            recipeC     15.347   3.918    0.31 0.99
##  Residual              20.477   4.525
## Number of obs: 270, groups:  replicate, 15
##
## Fixed effects:
##             Estimate Std. Error        df t value Pr(>|t|)
## (Intercept)  1.77214    3.50194 219.36537   0.506    0.613
## temp         0.15803    0.01613 239.97848   9.800   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##      (Intr)
## temp -0.921
```

```
#Creating a summary table
knitr::kable(summary(m05)$coefficients, caption = "Model 5: angle ~ temp + (1
↪  +recipe|replicate)", digits = c(4,4,4,4,30))
```

Table 1: Model 5: angle ~ temp + (1 +recipe|replicate)

	Estimate	Std. Error	df	t value	Pr(>|t|)
(Intercept)	1.7721	3.5019	219.3654	0.5060	6.133343e-01
temp	0.1580	0.0161	239.9785	9.8001	2.736626e-19

```
plot(m05)
```



```
knitr::kable(anova, digits = 3) # the digits argument controls rounding
```

	npar	AIC	BIC	logLik	deviance	Chisq	Df	Pr(>Chisq)
null	3	1908.283	1919.078	-951.142	1902.283	NA	NA	NA
m01	4	1676.226	1690.620	-834.113	1668.226	234.057	1	0
m02	5	1746.440	1764.432	-868.220	1736.440	0.000	1	1
m04	5	1738.860	1756.853	-864.430	1728.860	7.580	0	NA
m06	5	1666.971	1684.963	-828.486	1656.971	71.889	0	NA

	npar	AIC	BIC	logLik	deviance	Chisq	Df	Pr(>Chisq)
m07	8	1712.073	1740.861	-848.037	1696.073	0.000	3	1
m05	9	1666.211	1698.597	-824.106	1648.211	47.862	1	0
m03	10	1746.808	1782.792	-863.404	1726.808	0.000	1	1

Analysis 2: Logistic Regression & Interactions

```
df_titanic$Survived <- as.factor(df_titanic$Survived)
df_titanic$Pclass <- as.factor(df_titanic$Pclass)
df_titanic$Sex <- as.factor(df_titanic$Sex)
```

```
# Creating logistic regression - mixed model
m01_titanic <- glm(Survived ~ Sex + Age + Pclass, data = df_titanic, family = binomial)
m02_titanic <- glm(Survived ~ Sex, data = df_titanic, family = binomial)
m03_titanic <- glm(Survived ~ Age, data = df_titanic, family = binomial)
m04_titanic <- glm(Survived ~ Pclass, data = df_titanic, family = binomial)

summary(m01_titanic)
```

```
##
## Call:
## glm(formula = Survived ~ Sex + Age + Pclass, family = binomial,
##     data = df_titanic)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6811  -0.6653  -0.4137   0.6367   2.4505
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  3.63492    0.37045   9.812  < 2e-16 ***
## Sexmale     -2.58872    0.18701 -13.843  < 2e-16 ***
## Age         -0.03427    0.00716  -4.787 1.69e-06 ***
## Pclass2     -1.19911    0.26158  -4.584 4.56e-06 ***
## Pclass3     -2.45544    0.25322  -9.697  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1182.77  on 886  degrees of freedom
## Residual deviance:  801.59  on 882  degrees of freedom
## AIC: 811.59
##
## Number of Fisher Scoring iterations: 5
```

```
summary(m02_titanic)
```

```
##
## Call:
## glm(formula = Survived ~ Sex, family = binomial, data = df_titanic)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6462  -0.6496  -0.6496   0.7725   1.8218
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   1.0566     0.1290   8.191 2.58e-16 ***
## Sexmale      -2.5051     0.1672 -14.980  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1182.77  on 886  degrees of freedom
## Residual deviance:  916.12  on 885  degrees of freedom
## AIC: 920.12
##
## Number of Fisher Scoring iterations: 4
```

```
summary(m03_titanic)
```

```
##
## Call:
## glm(formula = Survived ~ Age, family = binomial, data = df_titanic)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.0864  -1.0017  -0.9439   1.3562   1.5806
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.209189   0.159494  -1.312   0.1897
## Age         -0.008774   0.004947  -1.774   0.0761 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1182.8  on 886  degrees of freedom
## Residual deviance: 1179.6  on 885  degrees of freedom
## AIC: 1183.6
##
## Number of Fisher Scoring iterations: 4
```

```
summary(m04_titanic)
```

```
##
## Call:
```

```
## glm(formula = Survived ~ Pclass, family = binomial, data = df_titanic)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.4094  -0.7486  -0.7486   0.9619   1.6788
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   0.5306     0.1409   3.766 0.000166 ***
## Pclass2      -0.6394     0.2041  -3.133 0.001731 **
## Pclass3      -1.6596     0.1760  -9.430  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1182.8  on 886  degrees of freedom
## Residual deviance: 1080.9  on 884  degrees of freedom
## AIC: 1086.9
##
## Number of Fisher Scoring iterations: 4
```

```
anova(m01_titanic, m02_titanic, m03_titanic, m04_titanic)
```

```
## Analysis of Deviance Table
##
## Model 1: Survived ~ Sex + Age + Pclass
## Model 2: Survived ~ Sex
## Model 3: Survived ~ Age
## Model 4: Survived ~ Pclass
##   Resid. Df Resid. Dev Df Deviance
## 1       882     801.59
## 2       885     916.12 -3 -114.528
## 3       885    1179.59  0 -263.470
## 4       884    1080.88  1   98.716
```

```
vif(m01_titanic)
```

```
##            GVIF Df GVIF^(1/(2*Df))
## Sex    1.085634  1        1.041938
## Age    1.351874  1        1.162701
## Pclass 1.448873  2        1.097129
```

```
MuMIn::r.squaredGLMM(m01_titanic)
```

```
##                   R2m       R2c
## theoretical 0.4668127 0.4668127
## delta       0.4055986 0.4055986
```

The analysis was created using R (R Core Team, 2019) to perform a logistic regression analysis, to determine the survival rates for each passenger class, sex and for the median age of each gender. The first model i built included all the predictors and used the following syntax: Survived ~ Sex + Age + Pclass In the model, fixed effects accounted for roughly 46.7% of variance. To test the multi-collinearity assumption, I used VIF (Variance Inflation Factors). The output for each predictor was approximately 1, which means the models meet the assumptions.

The survival rate for men were found to be significantly lower than women, B = -2.58872, SE = 0.18701, z = -13.843, $p < .0001$. The table displays the summary output of the model.

```
knitr::kable(summary(m01_titanic)$coefficients, caption = "Model 01: Survived ~ Sex + Age
↪  + Pclass", digits = c(4,4,4,4,30))
```

Table 3: Model 01: Survived ~ Sex + Age + Pclass

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	3.6349	0.3705	9.8120	0
Sexmale	-2.5887	0.1870	-13.8430	0
Age	-0.0343	0.0072	-4.7869	0
Pclass2	-1.1991	0.2616	-4.5841	0
Pclass3	-2.4554	0.2532	-9.6967	0

```
# Calculating survival rate for each class in percentages
summary(m04_titanic)
```

```
##
## Call:
## glm(formula = Survived ~ Pclass, family = binomial, data = df_titanic)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.4094  -0.7486  -0.7486   0.9619   1.6788
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   0.5306     0.1409   3.766 0.000166 ***
## Pclass2      -0.6394     0.2041  -3.133 0.001731 **
## Pclass3      -1.6596     0.1760  -9.430  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1182.8  on 886  degrees of freedom
## Residual deviance: 1080.9  on 884  degrees of freedom
## AIC: 1086.9
##
## Number of Fisher Scoring iterations: 4
```

```
Class1 <- boot::inv.logit(0.5306)
Class2 <- boot::inv.logit(0.5306 - 0.6394)
Class3 <- boot::inv.logit(0.5306 - 1.6596)
```

```r
Class <- c("Class 1", "Class 2", "Class 3")
percentage_class <-  round(c(Class1, Class2, Class3), 5) * 100

class_survival <- data.frame(Class, percentage_class)
```

The model for each class was built using the following syntax: Survived ~ Pclass The survival rate for the third passenger group found to significantly decrease compared to the first and second passenger group, B = -1.6596, SE = 0.1760, z = -9.430, $p < .0001$. This is also displayed in the summary table with class 3 having the lowest percentage of survival being ~24%.

```r
# Survival Rate for Each Class in Percentages
knitr::kable(class_survival, caption = "Survival Rate for Each Class in Percentages")
```

Table 4: Survival Rate for Each Class in Percentages

Class	percentage_class
Class 1	62.962
Class 2	47.283
Class 3	24.435

```r
knitr::kable(summary(m04_titanic)$coefficients, caption = "Model: Survived ~ Pclass",
↪  digits = c(4,4,4,4,30))
```

Table 5: Model: Survived ~ Pclass

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	0.5306	0.1409	3.7660	0.0002
Pclass2	-0.6394	0.2041	-3.1329	0.0017
Pclass3	-1.6596	0.1760	-9.4298	0.0000

```r
# Calculating survival rate for each sex in percentages
summary(m02_titanic)
```

```
##
## Call:
## glm(formula = Survived ~ Sex, family = binomial, data = df_titanic)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6462  -0.6496  -0.6496   0.7725   1.8218
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   1.0566     0.1290   8.191 2.58e-16 ***
## Sexmale      -2.5051     0.1672 -14.980  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1182.77  on 886  degrees of freedom
## Residual deviance:  916.12  on 885  degrees of freedom
## AIC: 920.12
##
## Number of Fisher Scoring iterations: 4
```

```
Female <- boot::inv.logit(1.0566)
Male <- boot::inv.logit(0.5306 - 2.5051)

Sex <- c("Female", "Male")
percentage_sex <-  round(c(Female, Male), 5) * 100

sex_survival <- data.frame(Sex, percentage_sex)
```

The model for each sex was built using the following syntax: Survived ~ Sex The survival rate for males found to be significantly lower compared to the survival rate of females, B = -2.5051, SE = 0.1672, z = -14.980, *p* < .0001. This is also displayed in the summary table with the survival rate for females being approximately ~74% and ~12% for men.

```
# Survival Rate for Each Sex in Percentages
knitr::kable(sex_survival, caption = "Survival Rate for Each Sex in Percentages")
```

Table 6: Survival Rate for Each Sex in Percentages

Sex	percentage_sex
Female	74.204
Male	12.191

```
knitr::kable(summary(m02_titanic)$coefficients, caption = "Model: Survived ~ Sex", digits
↪  = c(4,4,4,4,30))
```

Table 7: Model: Survived ~ Sex

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	1.0566	0.1290	8.1915	0
Sexmale	-2.5051	0.1672	-14.9798	0

```
# Calculating survival rate for median age for each sex, displayed in percentages
df_titanic %>%
  group_by(Sex) %>%
  summarise(Median_age = median(Age))
```

```
## # A tibble: 2 x 2
##   Sex    Median_age
##   <fct>       <dbl>
## 1 female         27
## 2 male           28
```

```
m05_titanic <- glm(Survived ~ Sex * Age, data = df_titanic, family = binomial)
summary(m05_titanic)
```

```
##
## Call:
## glm(formula = Survived ~ Sex * Age, family = binomial, data = df_titanic)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.0256  -0.6891  -0.5904   0.7653   2.2467
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.41192    0.28211   1.460 0.144254
## Sexmale     -1.28930    0.37615  -3.428 0.000609 ***
## Age          0.02423    0.00980   2.472 0.013437 *
## Sexmale:Age -0.04376    0.01265  -3.459 0.000541 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1182.77  on 886  degrees of freedom
## Residual deviance:  903.49  on 883  degrees of freedom
## AIC: 911.49
##
## Number of Fisher Scoring iterations: 4
```

```
Female_age <- boot::inv.logit(0.4119 + (27*0.0242))
Male_age <- boot::inv.logit(0.41192 - 1.28930 + (0.02423-0.04376)*28)

Age <- c("Female","Male")
percentage_age <-  round(c(Female_age, Male_age), 5) * 100

age_survival <- data.frame(Age, percentage_age)
```

The model for each age was built using the following syntax: Survived ~ Sex*Age The survival rate for males with a median age of 28 was found to be significantly lower compared to the survival rate of females with the median age of 27.

```
# Survival Rate for the Median Age for Each Sex in Percentages
knitr::kable(age_survival, caption = "Survival Rate for the Median Age for Each Sex in
↪  Percentages")
```

Table 8: Survival Rate for the Median Age for Each Sex in Percentages

Age	percentage_age
Female	74.37

Age	percentage_age
Male	19.40

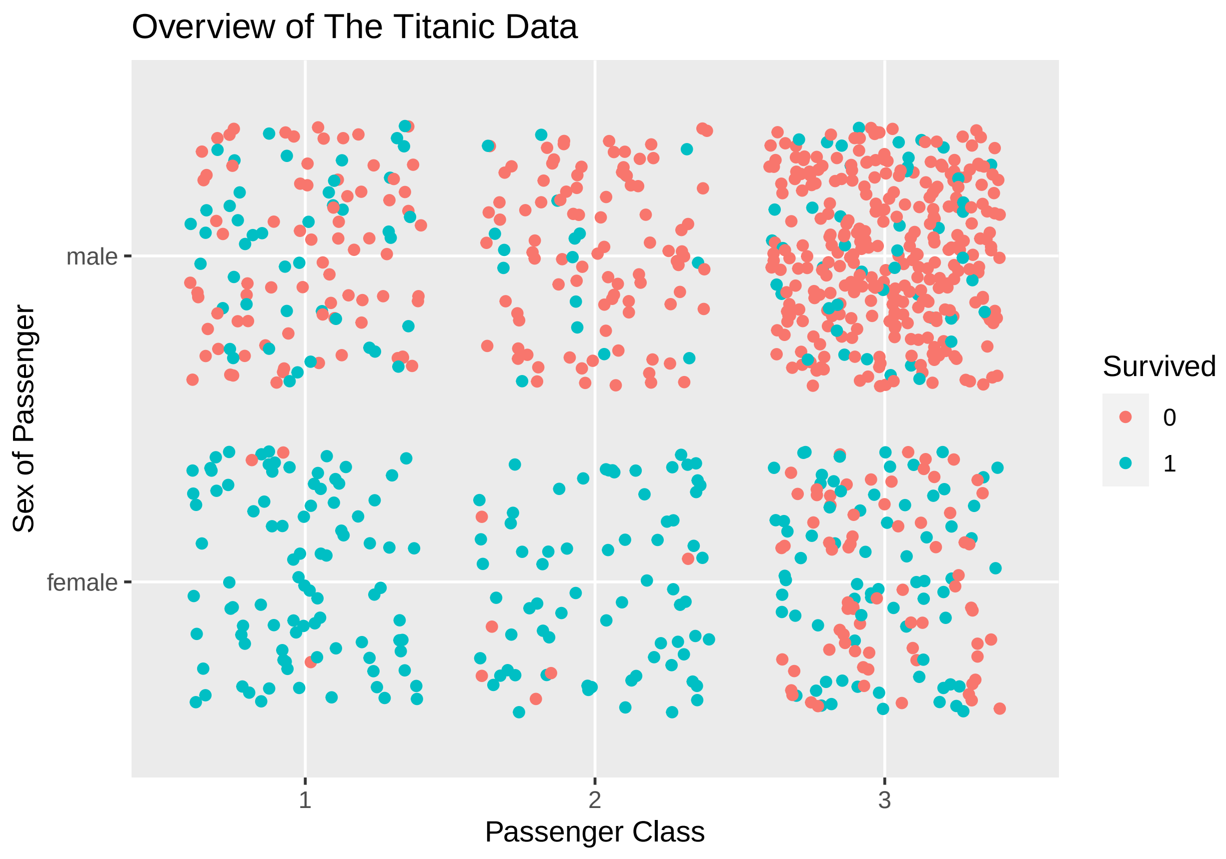
```
knitr::kable(summary(m04_titanic)$coefficients, caption = "Model: Survived ~ Sex*Age",
↪  digits = c(4,4,4,4,30))
```

Table 9: Model: Survived ~ Sex*Age

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	0.5306	0.1409	3.7660	0.0002
Pclass2	-0.6394	0.2041	-3.1329	0.0017
Pclass3	-1.6596	0.1760	-9.4298	0.0000

The plot displays the distribution of people who survived and did not survive, separated by gender and class.

```
# Plot to display the data
ggplot(df_titanic, aes(Pclass, Sex, Age, color = Survived)) +
  geom_jitter(width = .4, height = .4) +
  ggtitle("Overview of The Titanic Data") +
  labs(x = 'Passenger Class', y = 'Sex of Passenger')
```



```
MuMIn::r.squaredGLMM(m05_titanic)
```

```
##                  R2m       R2c
## theoretical 0.3250300 0.3250300
## delta       0.2728924 0.2728924
```