C91AR: Advanced Statistics using R

Wrangling Real Data

Dr Peter E McKenna

2025-02-11

Contents

1 Packages for todays session

```
# Load packages
pacman::p_load(tidyverse,
               broom,
               GGally,
               nortest,
               psych)
```

2 What are we going to cover today

- Wrangling, summarising, and visualising numeric data
- Assessing the distribution of numeric data (though we will cover this more thoroughly in week 7)
- Wrangling, summarising, and visualising categorical data

3 Summarising data in R

- Now that we've looked at how to keep code tidy and how to tidy our data, build on what we've started to learn from the data wrangling operations
- I've shown you how to filter and select your data, and by adding more commands to your chain of pipes (`|>`) you can also start to generate summaries of your data

4 Using pipes to generate summaries

- Think about your data and what summaries would be useful for your research
- Let's read in our tidy data to get started

```
# Read in the tidy data
df <-
  read_csv("data_tidy/c91ar_maze_data.csv",
           col_types = "cccdccd") |>
  drop_na()# omit NAs
```

5 Examine the data

```
# Examine the data
glimpse(df)
```

6 Dataset Metadata

- `id` = anon participant code
- `condition` = ToM manipulation with three levels (Baseline, No-ToM, and ToM)
- `trial` = experiment trial
- `rt` = response time
- `follow robot` = whether participants followed the robot's suggestion
- `accuracy` = whether or not their maze route selection was correct
- `conf` = participants self reported confidence for each route decision.

7 Reseach questions

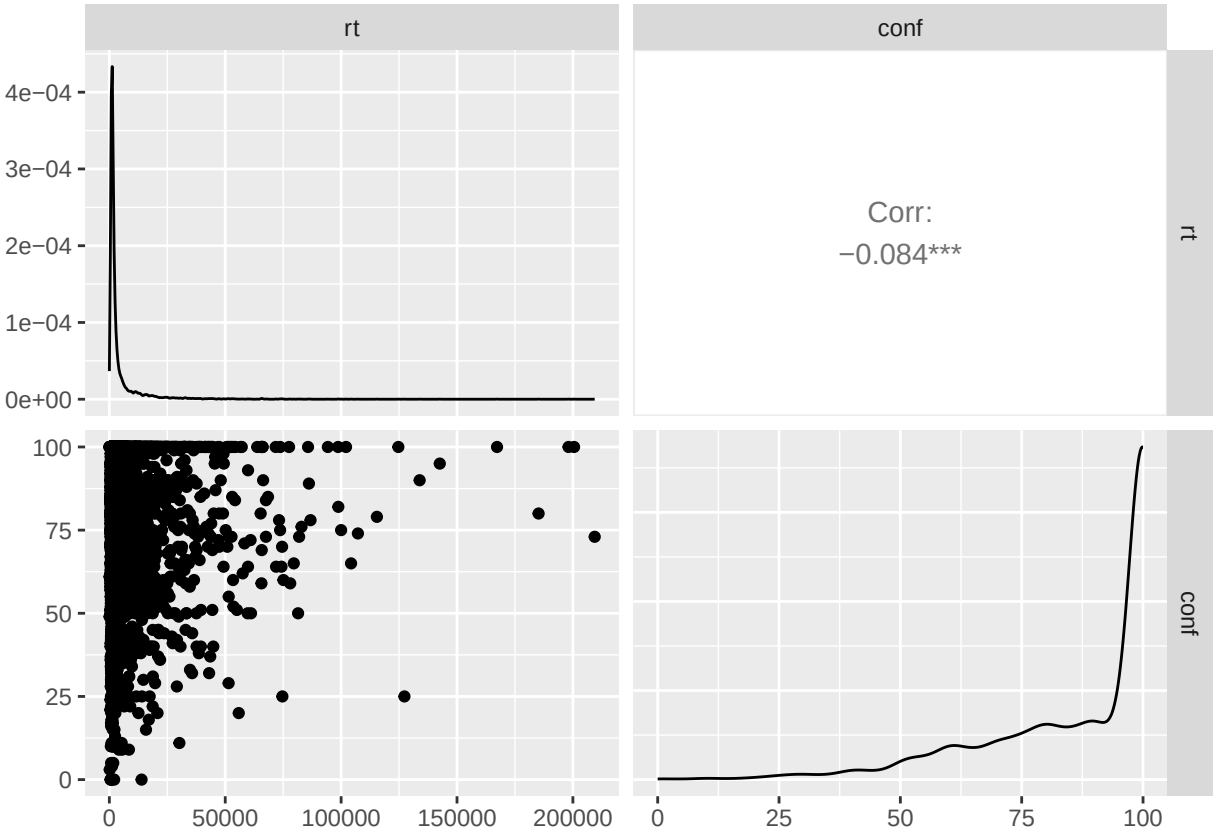
- Did robot ToM affect participants trust?

- Did robot ToM affect participants task performance?
  - Decision time
  - Confidence

8 Descriptive Statistics

- We use a combination of *Tidyverse* verbs to select, group and summarise data
  - `group_by`
  - `select`
  - `arrange`
  - `filter`
  - `summarise`

```
df |>
  select(rt,
         conf) |>
  ggpairs() # from GGally
```



- You can see that `rt` is positively skewed and `conf` is negatively skewed
- Think, **where is the tail?**

```
# Test the normality of rt
# shapiro.test(df$rt)



# Test the normlisaty of conf
# shapiro.test(df$conf)



# For large datasets, use Anderson-Darling instead
  # Null hypothesis is that the data follows a particular distribution
#library(nortest)

# response time
ad.test(df$rt)

# confidence
ad.test(df$conf)
```

- So, both `rt` and `conf` are not normally distributed

9 Working with reaction/decision time data

- Sometimes, you have to make a judgement call about what constitutes a theoretically valid response in your experiment.
- The minimum RT here is below zero, which is not possible
- One way to make an educated guess is to examine the histogram and hone in on the region of interest
- But, before I go any further…

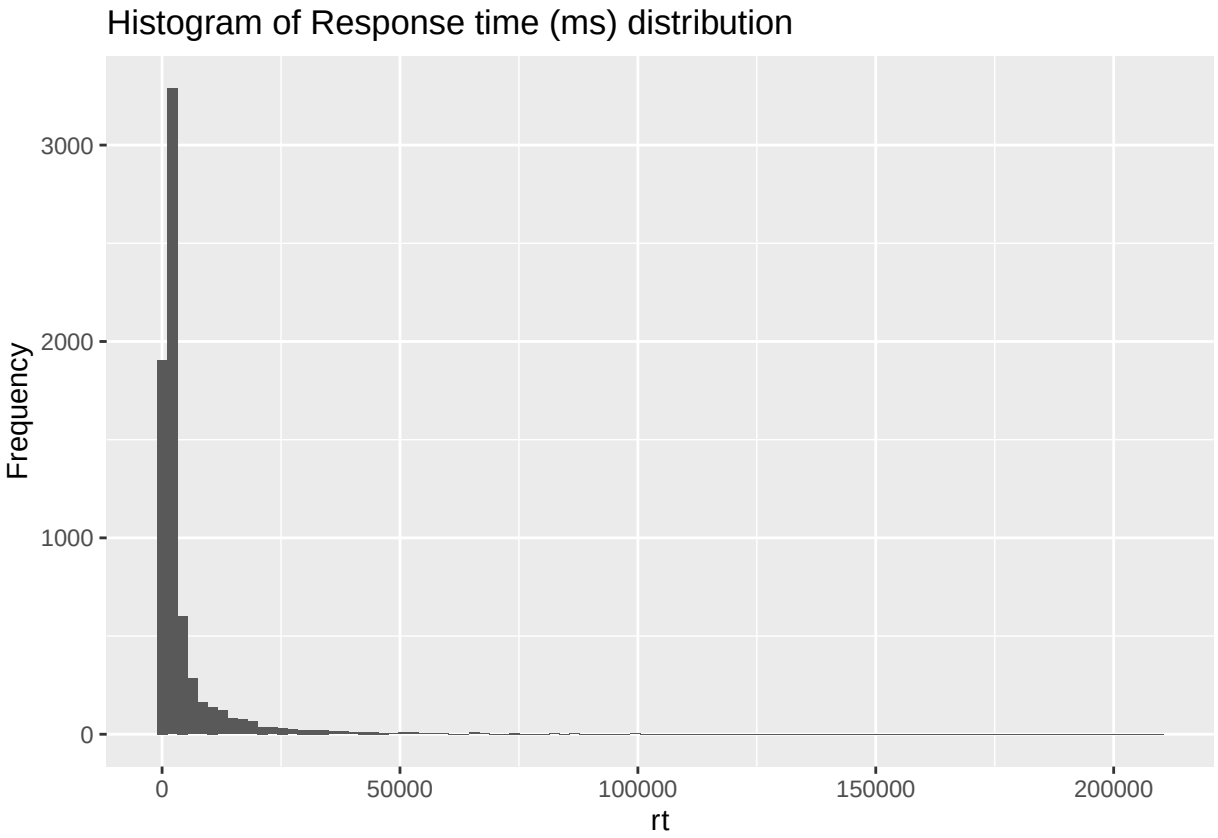
10 Data Visualisation with `ggplot2`

- This is the package used to produce plots in R
- It comes with a whole host of its own functions and is very flexible in terms of the graphical aesthetics.
- Annoyingly, it does not use `|>` (e.g., pipes) but instead uses the `+` symbol.

- So, when you combine *Tidyverse* and *ggplot2* you often see a combination beginning with |> and ending with + to chain commands together.

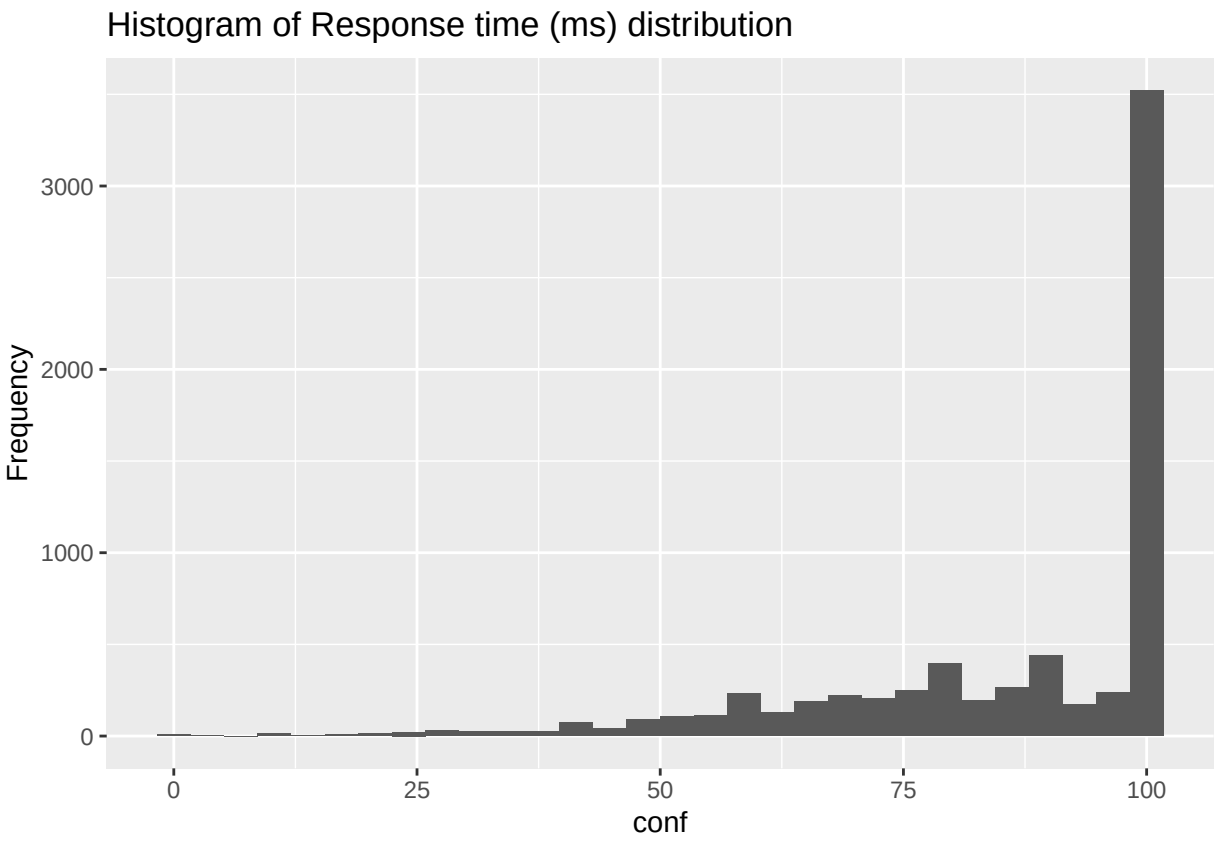
11 Visualising distributions using ggplot2

```
# Visualise the full rt distribution
df |>
  select(rt) |>
  ggplot(mapping = aes(rt)) +
  geom_histogram(bins = 100) +
  labs(y = "Frequency",
       title = "Histogram of Response time (ms) distribution")
```



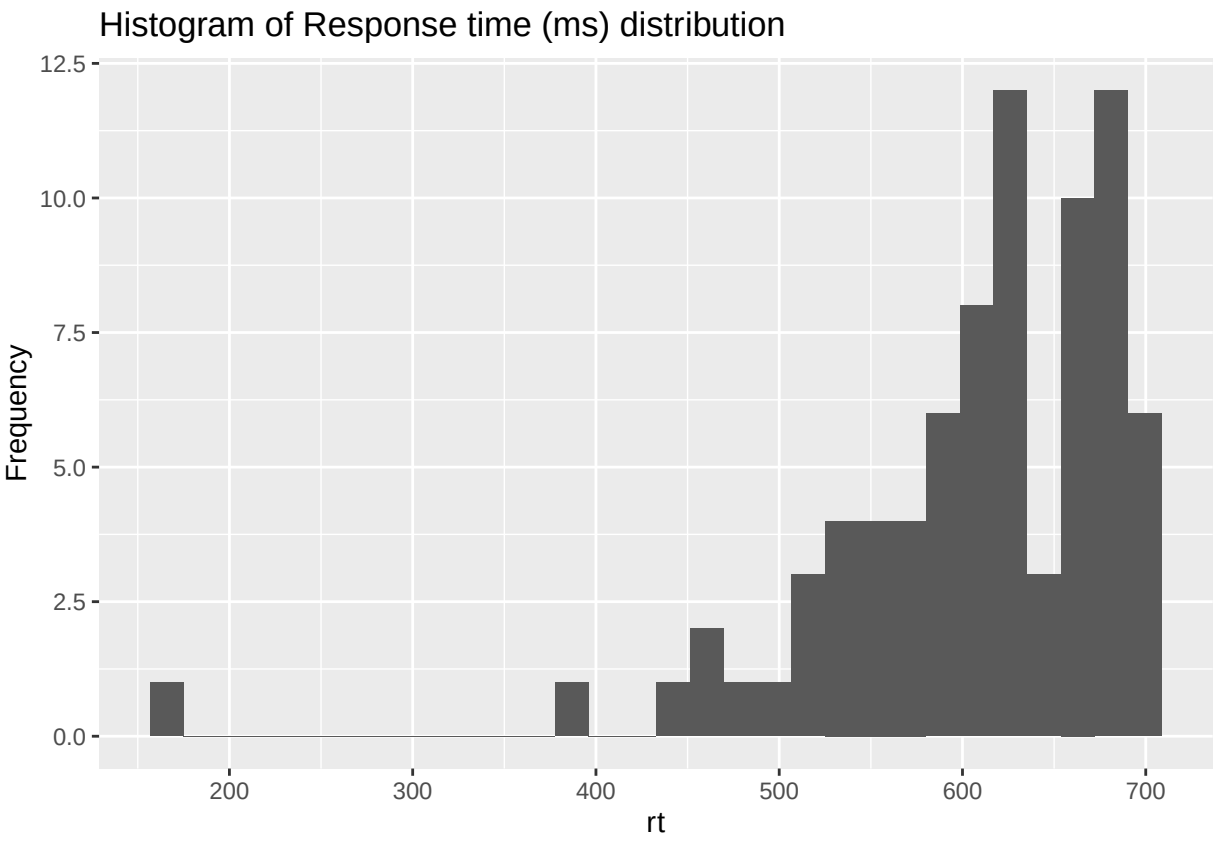
```
# Visualise the full conf distribution
df |>
  select(conf) |>
  ggplot(mapping = aes(conf)) +
  geom_histogram(bins = 30) +
```

```
  labs(y = "Frequency",
       title = "Histogram of Response time (ms) distribution")
```

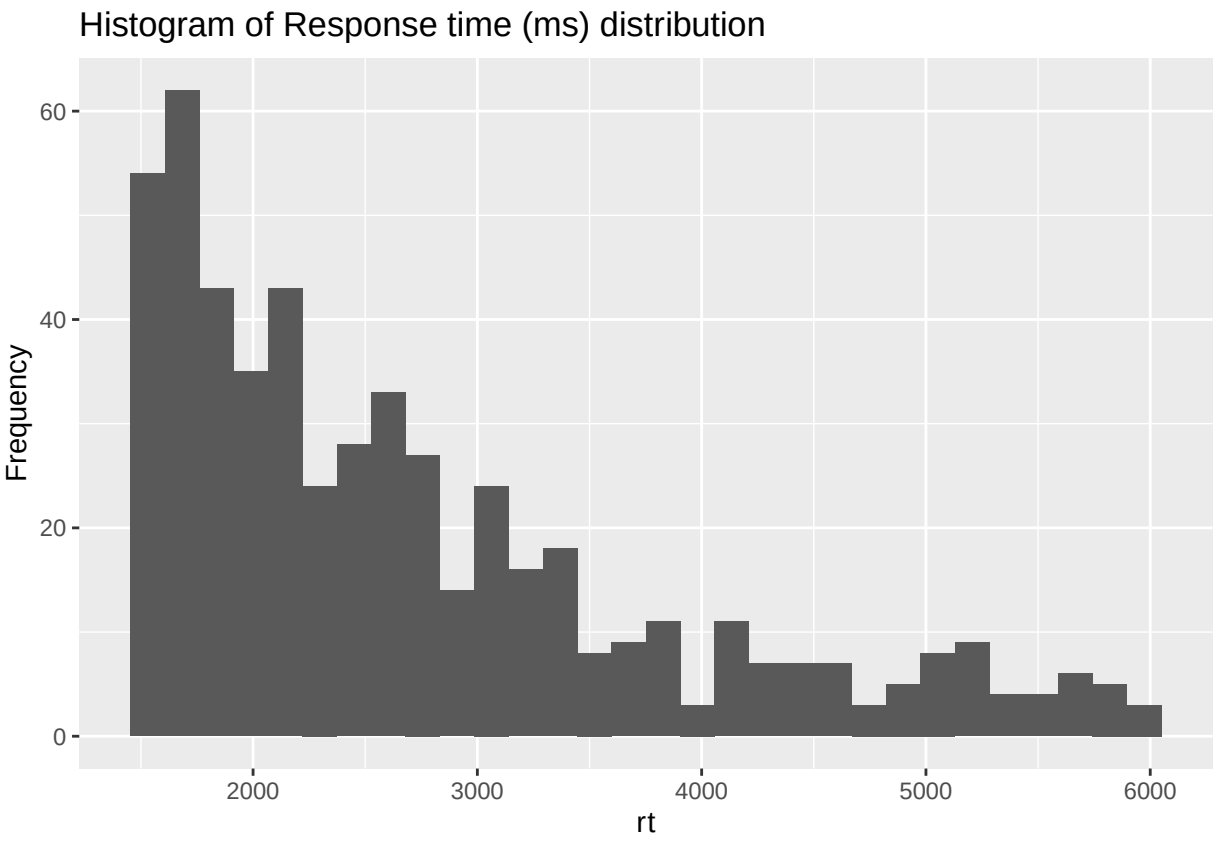


12 Taking a closer look at the distribution extremities

```
# Visualise the observations near the y-axis origin
df |>
  filter(rt %in% c(0:700)) |> # I'm not interested in values below zero
  ggplot(mapping = aes(rt)) +
  geom_histogram(bins = 30) +
  labs(y = "Frequency",
       title = "Histogram of Response time (ms) distribution")
```



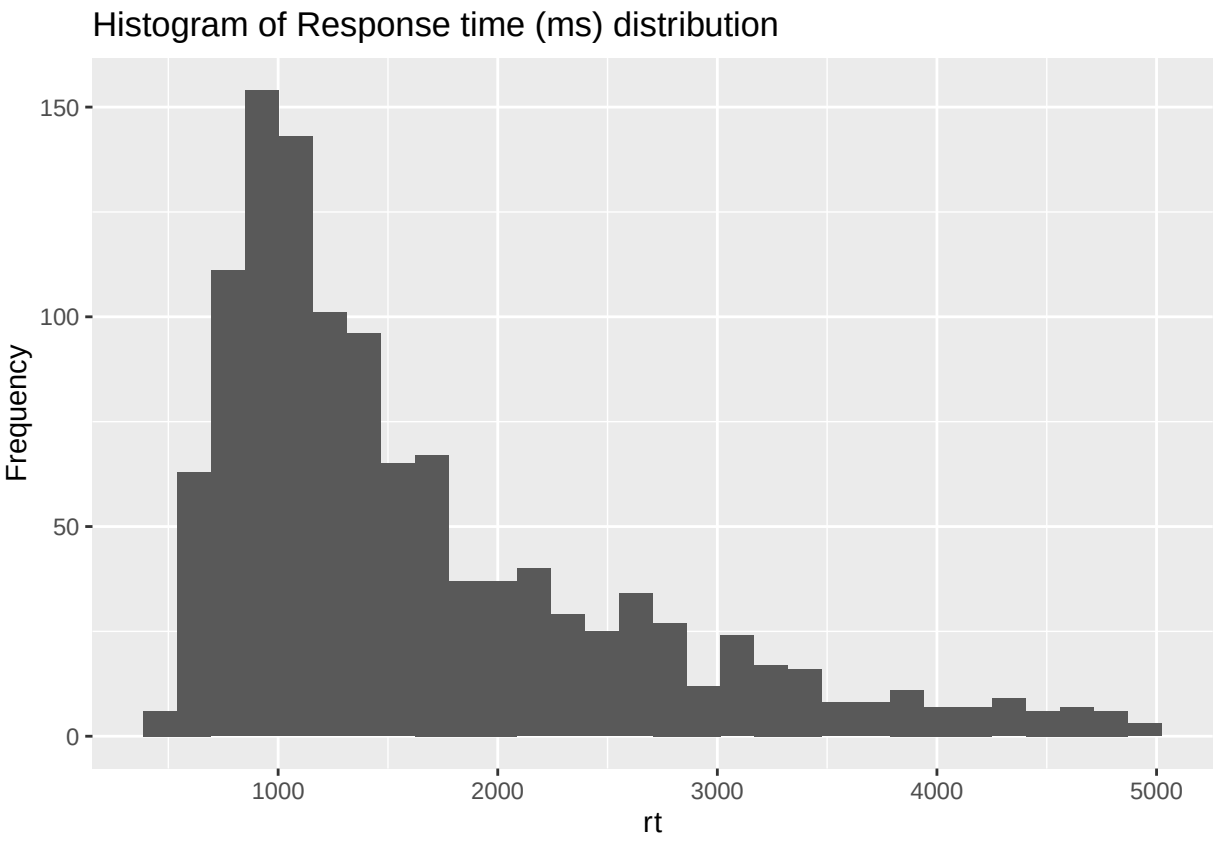
```
# Visualise the observations at the tail of the distribution
df |>
  filter(rt %in% c(1500:6000)) |>
  ggplot(mapping = aes(rt)) +
  geom_histogram(bins = 30) +
  labs(y = "Frequency",
       title = "Histogram of Response time (ms) distribution")
```



13 Based on the following plot, where would say is safe to start our distribution?

14 Visualise new distribution

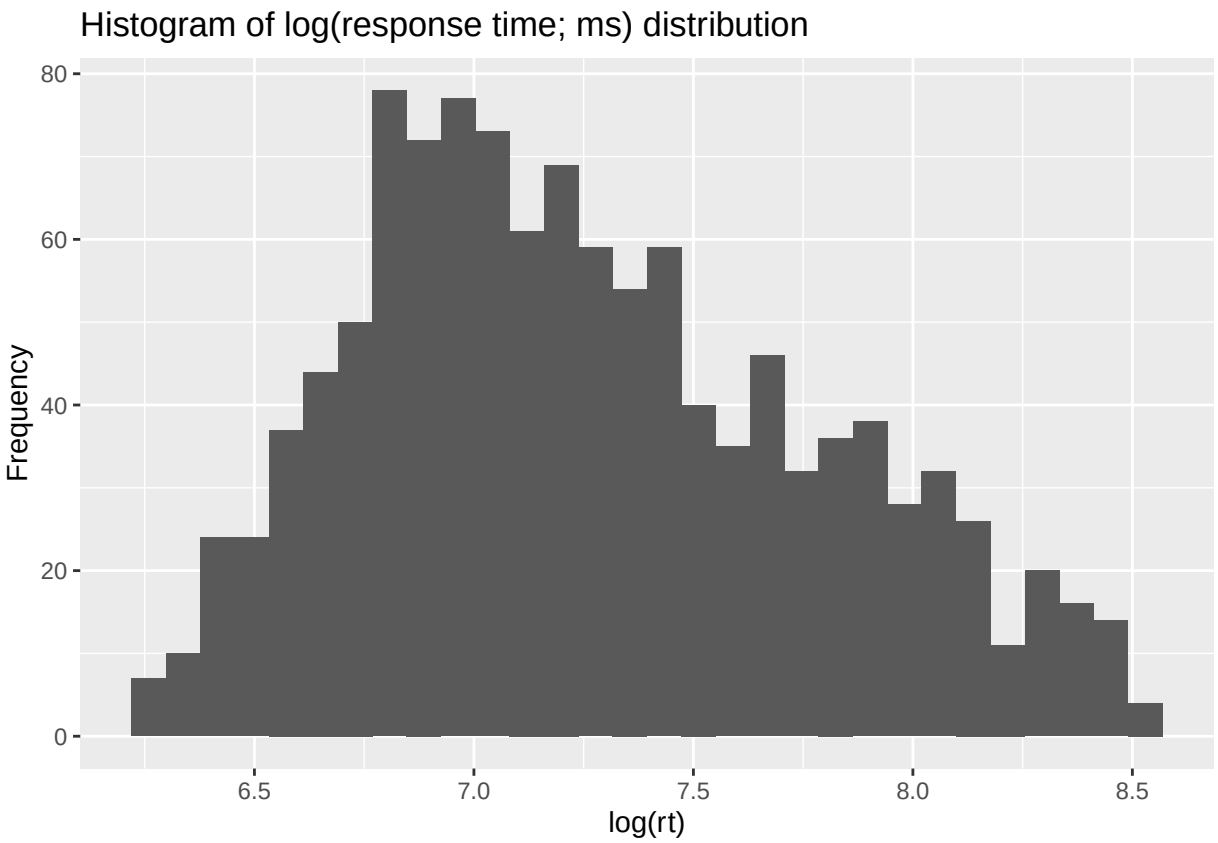
```
# Visualise the observations included in our new limits
df |>
  filter(rt %in% c(500:5000)) |>
  ggplot(mapping = aes(rt)) +
  geom_histogram(bins = 30) +
  labs(y = "Frequency",
       title = "Histogram of Response time (ms) distribution")
```



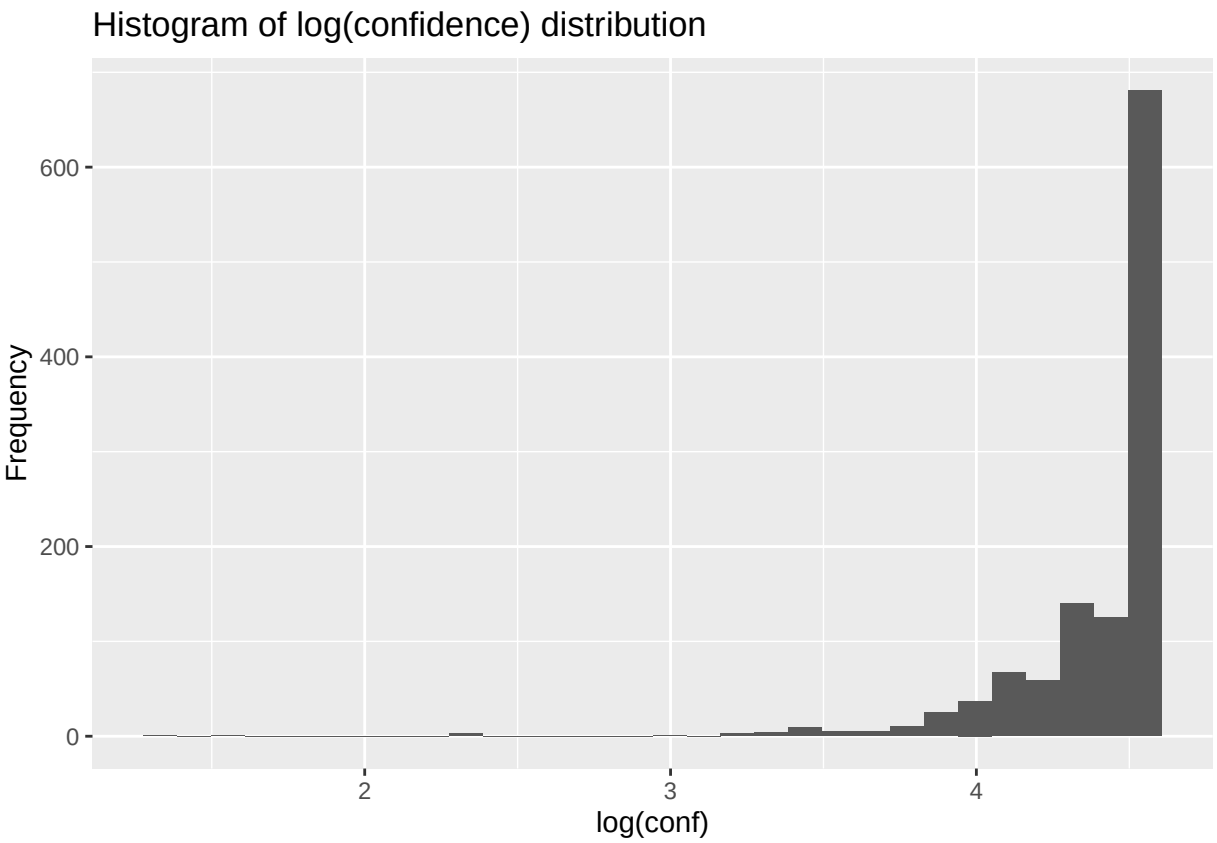
15 Visualise transformed data

- This measure is one approach to normalise positively and negatively skewed distributions

```
# Visualise the log transformed rt
df |>
  dplyr::filter(rt %in% c(500:5000)) |>
  ggplot(mapping = aes(log(rt))) +
  geom_histogram(bins = 30) +
  labs(y = "Frequency",
       title = "Histogram of log(response time; ms) distribution")
```

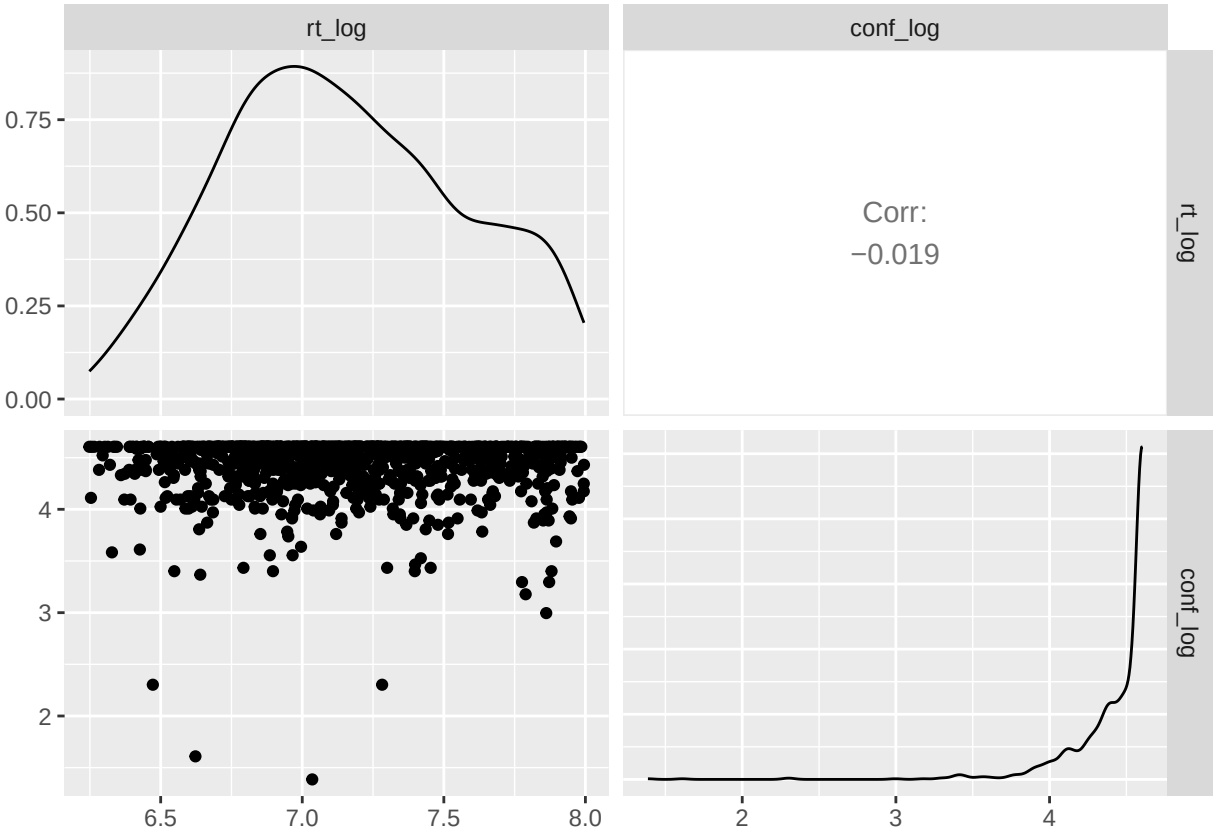


```
# Visualise the log transformed conf
df |>
  filter(rt %in% c(500:5000)) |>
  ggplot(mapping = aes(log(conf))) +
  geom_histogram(bins = 30) +
  labs(y = "Frequency",
       title = "Histogram of log(confidence) distribution")
```



16 Checking correlations based on new parameters

```
df |>
  filter(rt %in% c(500:3000)) |> # filter the data
  mutate(rt_log = log(rt),
         conf_log = log(conf)) |> # create a log of rt vector
  select(rt_log,
         conf_log) |> # only include numeric vectors
  ggpairs()
```



- According to our new limits and log transformation, there is no correlation between response time and confidence.

17 ToM condition descrtiptives

```
# create a new object based on our chosen rt parameters

df1 <-
  df |>
  filter(rt %in% c(500:5000)) |>
  mutate(rt_log = log(rt),
         conf_log = log(conf)) |>
  mutate_if(is.character, as.factor)
```

18 Creating new tibbles/objects

- Be sure to create a comment explicitly stating what df1 represents
- You could do this in a separate .txt file or in your script

- The last line of the chain tells R to treat the character cols as factor, because converting factors back to characters is the default behaviour when creating new data objects
- df1 = data with rt 500:5000

```
# Summarise the data
df1 |>
  select(condition,
         rt_log,
         conf_log) %>%
  group_by(condition) %>%
  summarise(avg_rt = mean(rt_log), # remember, we are operating in the log space now
            sd_rt = sd(rt_log),
            med_rt = median(rt_log),
            avg_conf = mean(conf_log),
            sd_conf = sd(conf_log),
            med_conf = median(conf_log))
```

```
## # A tibble: 3 x 7
##   condition avg_rt sd_rt med_rt avg_conf sd_conf med_conf
##   <fct>      <dbl> <dbl>  <dbl>    <dbl>   <dbl>    <dbl>
## 1 baseline    7.18 0.523   7.06     4.38   0.346     4.49
## 2 No-ToM      7.32 0.512   7.27     4.45   0.289     4.61
## 3 ToM         7.34 0.499   7.27     4.46   0.224     4.60
```

19 Describing numeric vectors using psych package

```
# Overall summary of rt
df1 |>
  describe(omit = TRUE) # omit non-numeric vectors
```

20 Descritives of your numeric variables per each level of the predictors

```
# Condition level summary of rt
describe(rt_log ~ condition, data=df1) # remember, tilde means modelled by
```

21 Exercise: use the `describe` function from the *psych* package to summarise both overall and group trends from the confidence rating participants gave after each decision.

22 Summarising the categorical variables

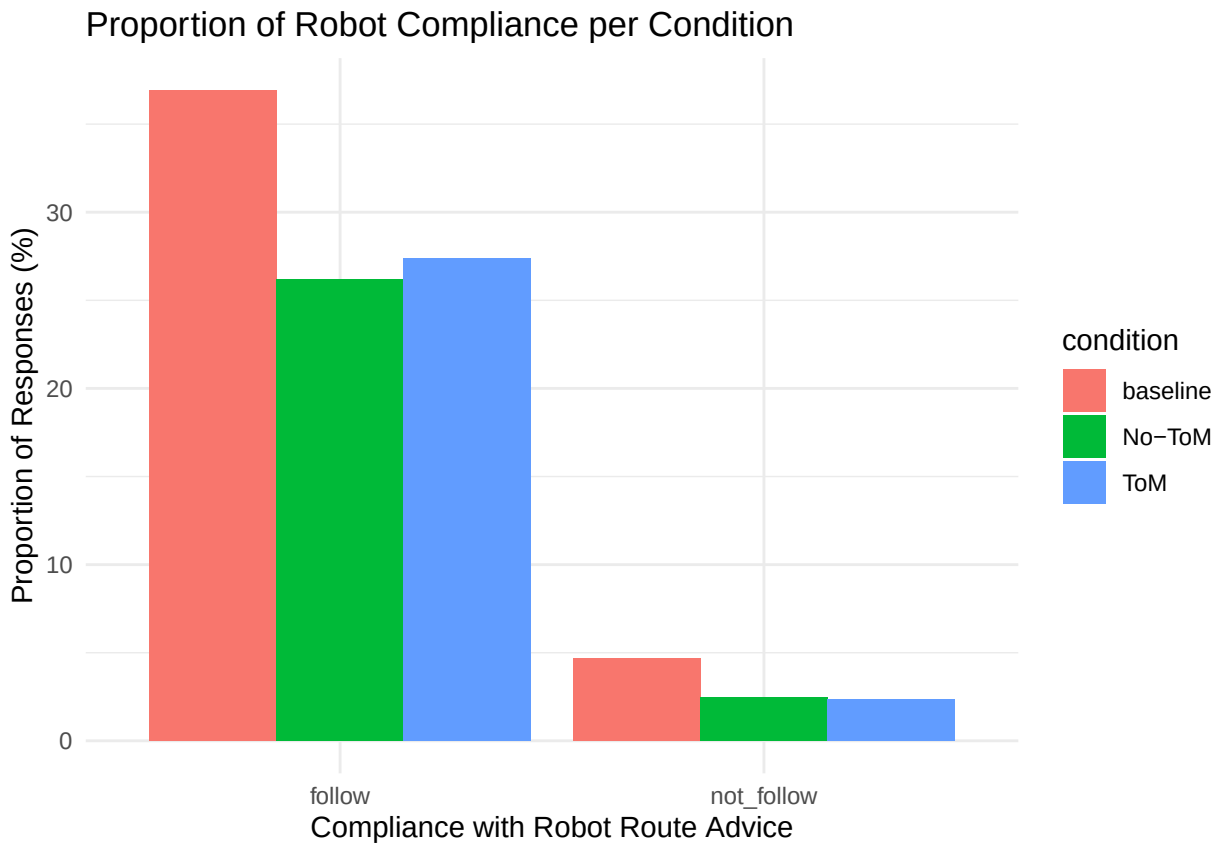
- In the long data format (which is the go-to for R) each row represents a unique observation in the data.
- So, when you summarise categorical data, you need to input the number of rows into the calculation to compute overall proportions.

```
# How often did participants in each group follow the robots advice?
total_n <-
  nrow(df1) # nrow counts the number of rows

df1 |>
  group_by(condition, follow_robot) |>
  summarise(n = n(),
            .groups = 'drop') |> # ensures that the grouping is dropped after summarising
  mutate(freq = n / total_n) |>
  mutate(freq = round(freq * 100, digits = 2))
```

23 Visualising categorical data trends

```
p_follow <-
  df1 |>
  group_by(condition, follow_robot) |>
  summarise(n = n(),
            .groups = 'drop') |> # ensures that the grouping is dropped after summarising
  mutate(freq = n / total_n) |>
  mutate(freq = round(freq * 100, digits = 2))
```

```
p_follow |>
  ggplot(aes(x = follow_robot,
             y = freq,
             fill = condition,
             group = condition)) +
  geom_col(stat = "identity",
           position = "dodge") +
  labs(title = "Proportion of Robot Compliance per Condition",
       x = "Compliance with Robot Route Advice",
       y = "Proportion of Responses (%)") +
  theme_minimal()
```

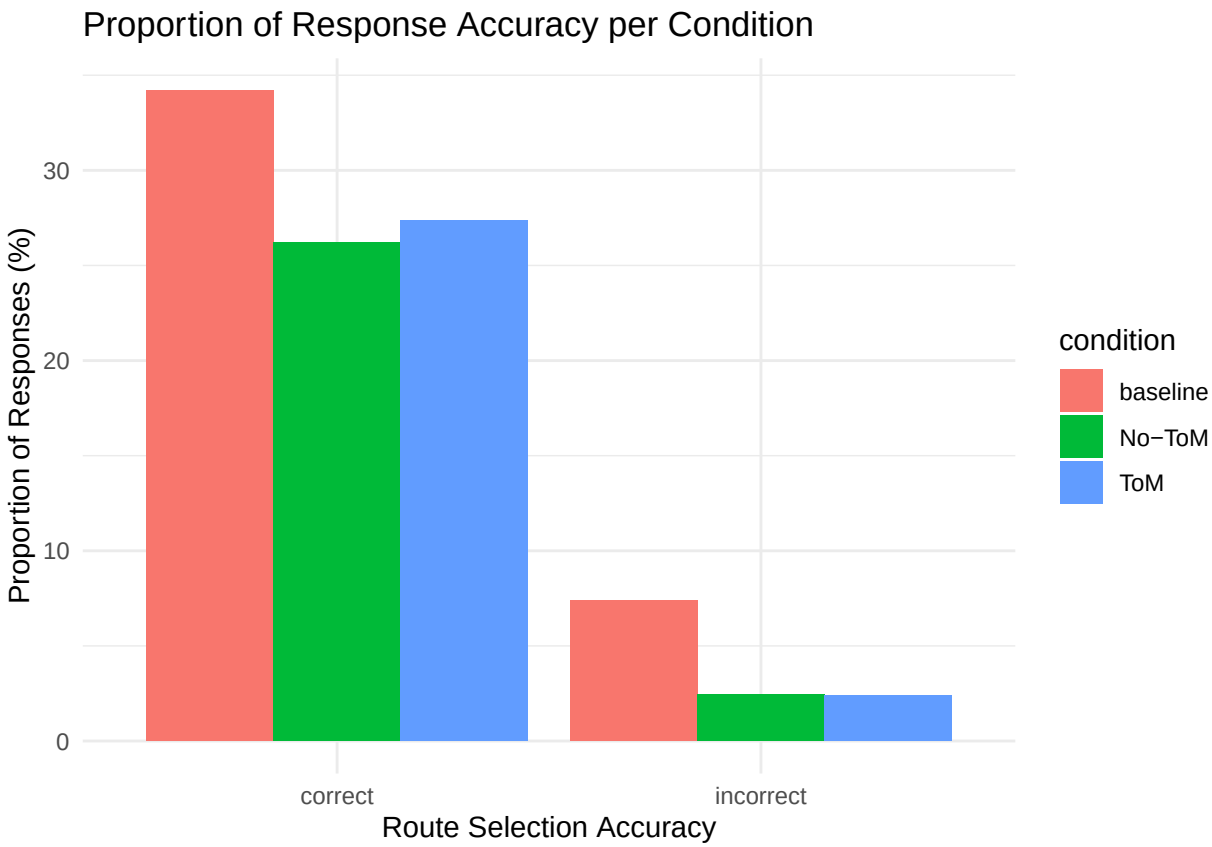


```
# How often were participants correct in their route selection?

df1 |>
  group_by(condition,
           accuracy) |>
  summarise(n = n(),
            .groups = 'drop') |>
```

```
  mutate(freq = n / total_n) |>
  mutate(freq = round(freq * 100, digits = 2))
```

```
## # A tibble: 6 x 4
##   condition accuracy      n  freq
##   <fct>     <fct>     <int> <dbl>
## 1 baseline  correct     402 34.2
## 2 baseline  incorrect    87  7.4
## 3 No-ToM    correct     308 26.2
## 4 No-ToM    incorrect    29  2.47
## 5 ToM       correct     322 27.4
## 6 ToM       incorrect    28  2.38
```

```
p_acc <-
  df1 |>
  group_by(condition,
           accuracy) |>
  summarise(n = n(),
            .groups = 'drop') |>
  mutate(freq = n / total_n) |>
  mutate(freq = round(freq * 100, digits = 2))

p_acc |>
  ggplot(aes(x = accuracy,
                y = freq,
                fill = condition,
                group = condition)) +
  geom_col(stat = "identity",
           position = "dodge") +
  labs(title = "Proportion of Response Accuracy per Condition",
       x = "Route Selection Accuracy",
       y = "Proportion of Responses (%)") +
  theme_minimal()
```



24 Wrap up

- Today we covered
  - How to wrangle and summarise numeric and categorical data
  - Had brief explosure to checking the normality of numeric distributions
  - You got your first taste of `ggplot2` plotting
- Next week there are no official lectures or tutorials. Please get in touch to discuss anything. I'm happy to help de-bug your code.

25 Cleanup

```
# Clear data
rm(list = ls())  # Removes all objects from environment

# Clear packages
p_unload(all)  # Remove all contributed packages

# Clear plots
```

```r
graphics.off()  # Clears plots, closes all graphics devices


# Clear console
cat("\014")  # Mimics ctrl+L
```

26 References