Homework 2

Author: Matthew J. Crossley

Last update: 09 April, 2024

Learning objectives

- This homework shares all learning objective with hw1. In addition, this homework will assess your ability to use inferential statistics (e.g., *t*-tests) to make simple decisions on the basis of data.

Instructions

- Write a `.Rmd` script that exactly replicates each `ggplot` figure and `data.table` output shown below.
- When you are finished, knit your `.Rmd` to a `.pdf` file being careful to set `eval=T` and `echo=T` for each code chunk so that both your code syntax and plots / output are visible. **Please submit the resulting `.pdf` file through ilearn.**
- This homework is structured as walk-thru of an analysis of a simple experiment. Throughout, I write comments and ask prompting questions. Usually, these sorts of things are bulleted. In any case, they are there simply to guide you along my thought process, and in so doing, are there to give you hints that direct you towards methods that will help you reproduce my output and plots.
- Again, you are marked on your ability to reproduce my plots and output, and also perhaps on your ability to explain and defend your code should it be flagged as seeming too heavily influenced by ChatGPT or by group collaboration. It is fine, by the way, to have even all your code written by ChatGPT, but only if it produces a deep understanding of your own code. You may lose marks if you write code that you do not understand.

Marking

For each question:

- 10% goes to whether your code runs without error.
- 10% goes to whether your R code chunks produce only the output shown below. Your code should not produce warnings, error messages, or any other output in the final pdf that is not shown below. The best way to prevent warnings from being displayed is by thinking deeply about any warnings that may occur, and then modifying your code to prevent them from happening. Another way – less scientifically awesome but still perfectly okay for marks on this assigment – is to simply set `warning=F` in your code chunks.
- The remaining 80% goes to whether you replicate the data.table output and plots plots below. Please note that the arrangement of the plots into columns or rows **will** be an aspect of your mark.
- To get full marks you must produce all outputs and figures with exactly the same values, in exactly the format, and with exactly the same aesthetics as shown. In essence, you must create a clone of what I have done. Arrangement matters. Colour matters. Labels matter. Etc.
- Creating a clone of my histograms may be challenging since small differences in binning lead to qualitatively identical but nevertheless inexact plots. In cases like this, the guiding principle is that it needs to be clear from your plot that you are plotting the same data and revealing the same major

trends (which I comment on in the bullets). A good start towards this is to make sure the range on your x and y axes look the same as mine without you having to manually adjust them. However, you do not need to spend hours fiddling with bin sizes to get them truly identical to mine. It should, however, be close enough that at a glance it is not obvious that you have used different bins than me.

- Please note that the previous bullet on histograms does not apply to box plots. If you are using the same data as me, then it should be trivial to obtain identical box plots as shown below.
- You will get zero marks for any component that does not include a printout of the code used to generate the output and plots in your pdf. This means that it is very important to use `echo=T` in your `Rmd` code chunks.
- When knitting to `pdf` you may find that long lines of code are not wrapped in the output. It is important that you alter the formatting of your code to fit on the page as displayed in the final `pdf` you submit. You may lose marks if you do not do this (e.g., we cannot give partial credit if we cannot see your code).

0. Setup

- Step 0 is essential but is worth **0% of your HW2 total mark**. It is here to help you get started.
- Load the `data.table` and `ggplot2` libraries and remove any lingering variables from your `R` session memory.

```
library(data.table)
library(ggplot2)
library(ggpubr)
rm(list=ls())
```

1. File I/O (from online experiment)

- **This step contains three `R` and `data.table` outputs and zero `ggplot` figures.**
- Consider the experiment here:
- `https://gitlab.pavlovia.org/demos/eriksen_flanker`
- `https://run.pavlovia.org/demos/eriksen_flanker/`
- You should participate in the experiment to get a feel for it.
- Data for this experiment is here:
- `https://gitlab.pavlovia.org/demos/eriksen_flanker/-/tree/master/data`
- Download the entire data directory, wrangle the data into a single `data.table`, and add an column named `subject` that indicates from what subject each observation was taken.
- The key columns in each data file are:
- `subject`: participant indicator
- `stimulus`: the presented stimulus
- `condition`: whether the stimulus is congruent or incongruent
- `key_resp.keys`: the response made by the participant
- `response`: the correct response
- `key_resp.rt`: the response time of the key press
- You may also find it helpful to add a column encoding accuracy (i.e., whether or not `key_resp.keys` is equal to `response` on each trial for each subject).
- We will also clean up the `condition` column so that it contains only the following values: `c("congruent", "incongruent")`
- To ensure that you are able to replicate my results **exactly** we will filter out any data that was collected after 2024-01-01. You saw how to do this in HW1.
- Your `data.table` should look very similar to this:

```
## [1] "Number of unique subject (i.e., files):"
```

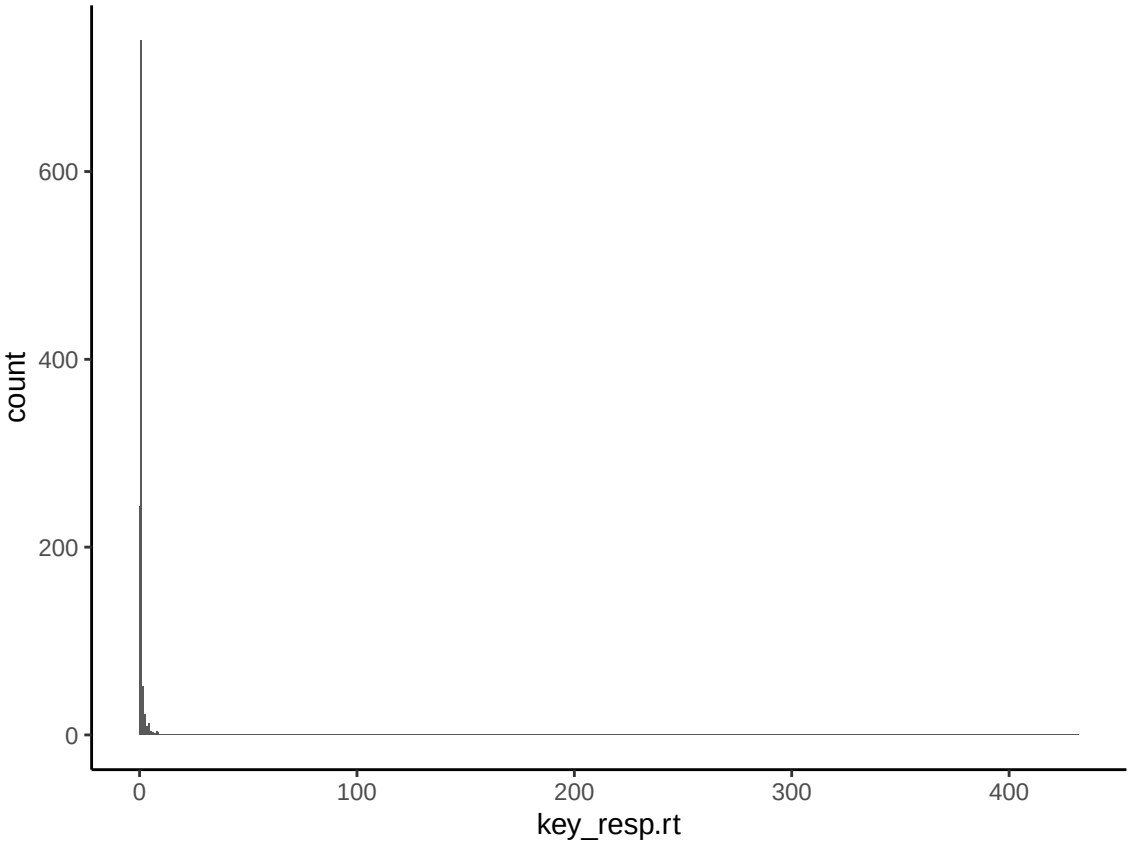
```
## [1] 121
```

```
## Classes 'data.table' and 'data.frame':   1103 obs. of  33 variables:
##  $ instructions.started: num  NA NA NA NA NA NA NA NA NA NA ...
##  $ instructions.stopped: num  NA NA NA NA NA NA NA NA NA NA ...
##  $ startKey.keys       : chr  "" "" "" "" ...
##  $ startKey.rt         : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ startKey.duration   : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ participant         : chr  "0" "0" "0" "0" ...
##  $ session             : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ date                : chr  "2023-11-01_23h55.48.521" "2023-11-01_23h55.48.521" "2023-11-01_23h55.4
##  $ expName             : chr  "eriksen_flanker" "eriksen_flanker" "eriksen_flanker" "eriksen_flanker
##  $ psychopyVersion     : chr  "2023.2.2" "2023.2.2" "2023.2.2" "2023.2.2" ...
##  $ OS                  : chr  "Win32" "Win32" "Win32" "Win32" ...
##  $ frameRate           : num  60.1 60.1 60.1 60.1 60.1 ...
##  $ subject             : int  2 2 2 2 2 2 2 2 2 2 ...
##  $ trial.started       : num  2.72 6.58 9.69 12.11 14.76 ...
##  $ trial.stopped       : num  5.59 8.69 11.11 13.76 16.28 ...
##  $ key_resp.keys       : chr  "right" "right" "right" "left" ...
```

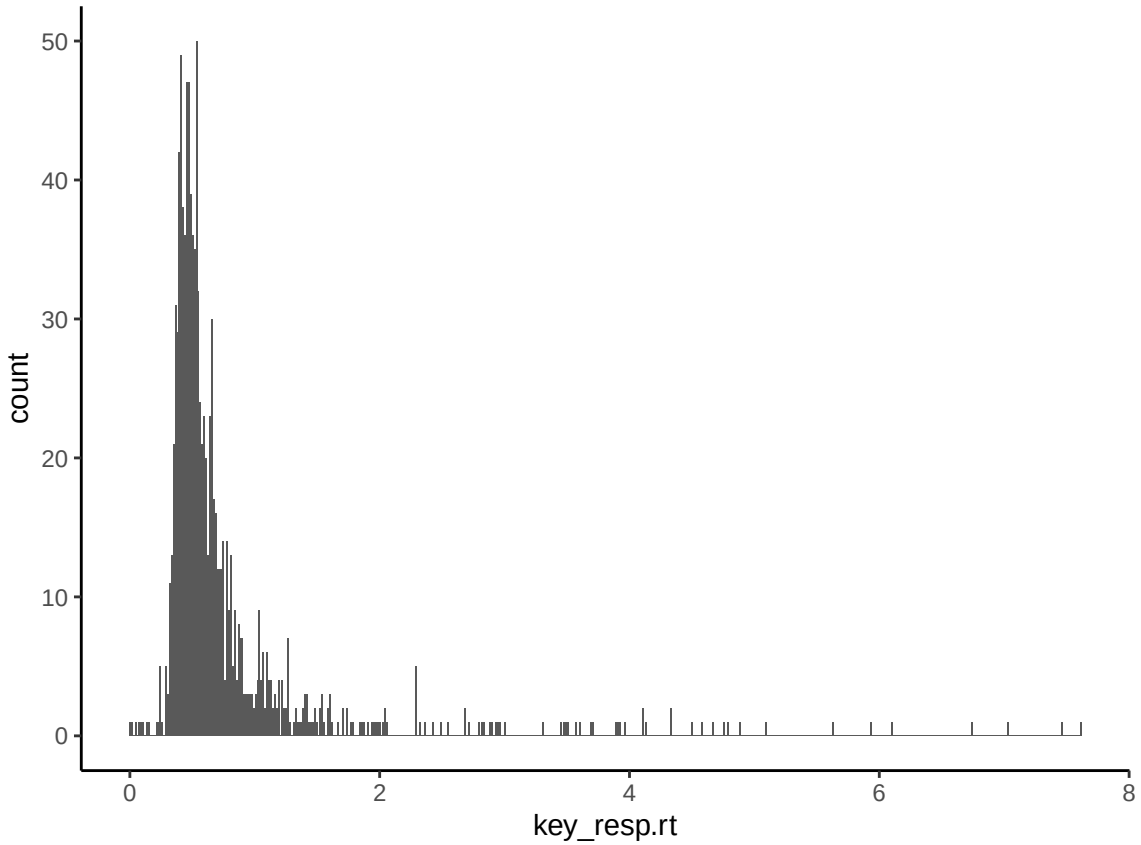
```
##  $ key_resp.corr       : int  1 1 1 1 1 1 0 1 1 1 ...
##  $ key_resp.rt         : num  2.324 1.581 0.893 1.127 0.984 ...
##  $ key_resp.duration   : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ feedback.started    : num  5.59 8.69 11.11 13.76 16.28 ...
##  $ feedback.stopped    : num  6.58 9.68 12.11 14.76 17.27 ...
##  $ trials.thisRepN     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ trials.thisTrialN   : int  0 1 2 3 4 5 6 7 8 9 ...
##  $ trials.thisN        : int  0 1 2 3 4 5 6 7 8 9 ...
##  $ trials.thisIndex    : int  8 5 1 6 2 0 9 3 7 4 ...
##  $ trials.ran          : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ stimulus            : chr  "<<><<" ">>>>>" ">>>>>" "<<<<<" ...
##  $ condition           : chr  "incongruent" "congruent" "congruent" "congruent" ...
##  $ response            : chr  "right" "right" "right" "left" ...
##  $ end.started         : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ trials.order        : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ acc                 : logi  TRUE TRUE TRUE TRUE TRUE TRUE ...
##  $ date_time           : POSIXct, format: "2023-11-01 23:55:48" "2023-11-01 23:55:48" ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

2. Initial look at data and exclusions

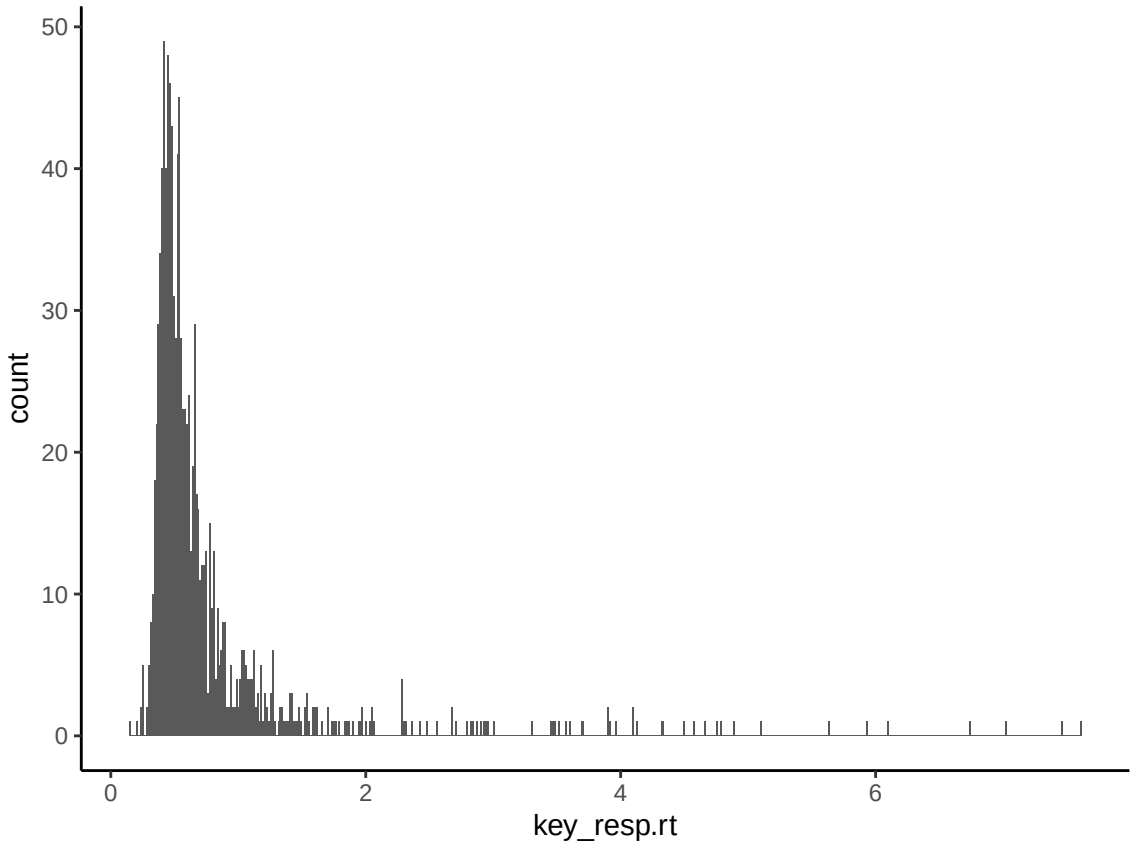
- **This step contains 4 stand alone `ggplot` figures, 2 `ggplot` figures arranged in a single row, and 2 `data.table` outputs.**
- Get a feel for the data by plotting a histogram of response times.



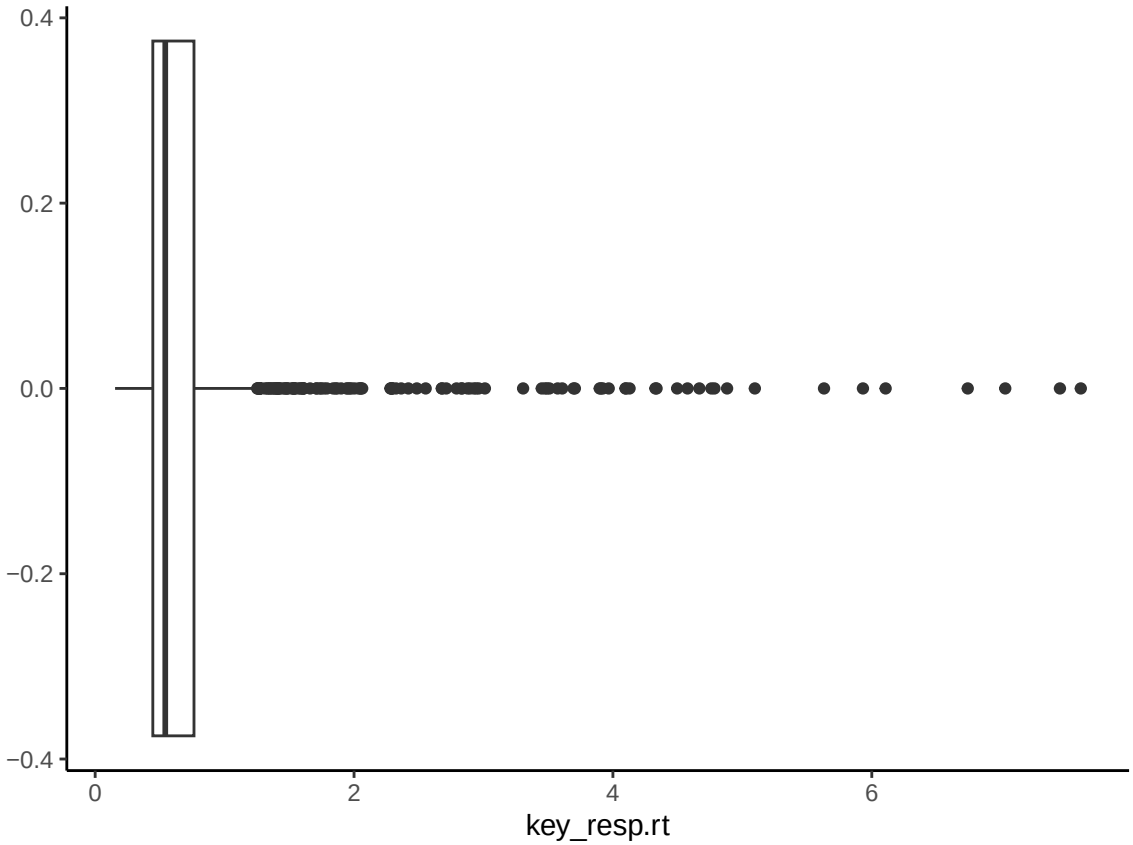
- We see that the range of the observed response times spans values about as small as 0 and larger than 400. From this plot – along with our intuition that values in the hundreds do not likely represent real signal – it is clear that we should restrict our attention to short response times. Lets exclude the longest 1% of these observations from the data.



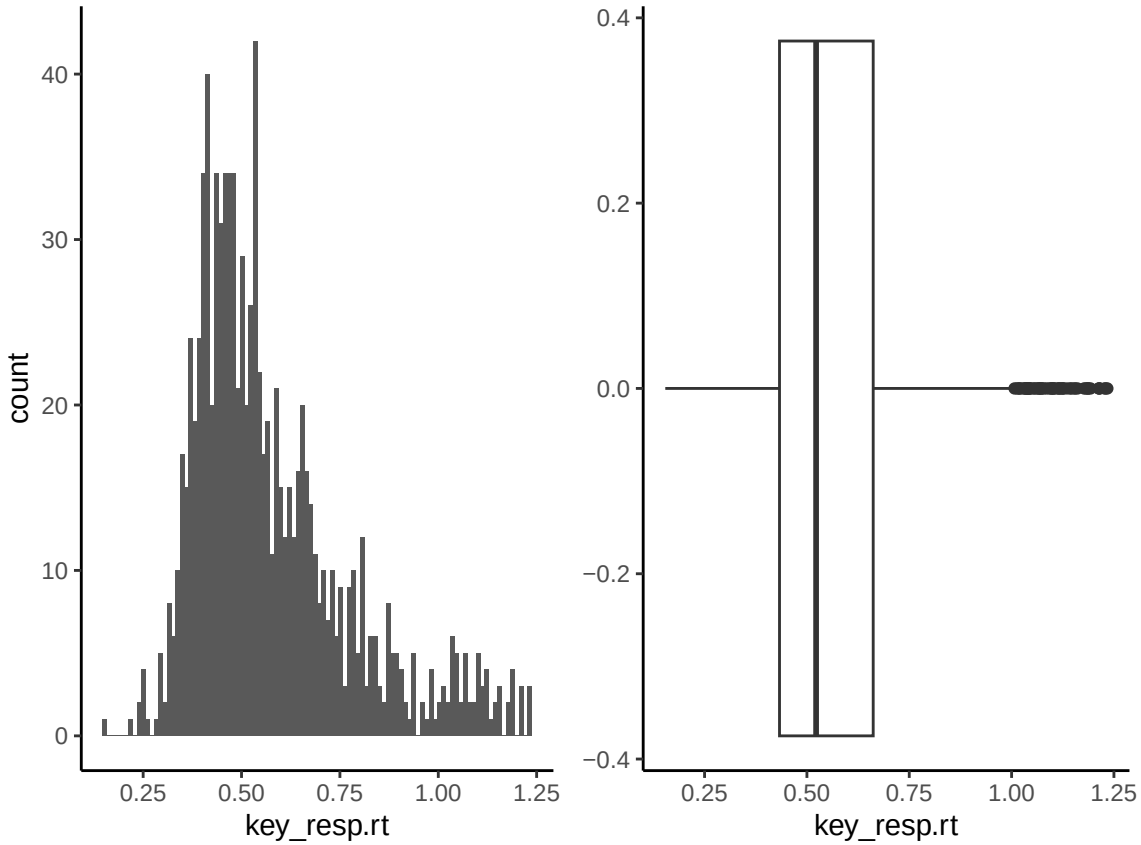
- The resulting distribution of response times looks pretty reasonable now with the exception of a few response times at implausibly small values. This is likely for the same reason discussed in hw1 (i.e., some people are just mashing the key pad with their palm without paying any attention to the task). It takes at least about 150 ms for visual information to reach motor resources, so we will call anything below that unreasonable. Exclude any trial that has a response time less than 150 ms and re-plot the histogram.



- That cleaned up the low range, but there are still some very long response times in the data that likely do not reflect a meaningful signal. E.g., even 2 seconds is quite a long time in this task and it is difficult to believe that anybody paying attention and actively engaged in the task would produce response times that large
- There is no universally agreed upon method for detecting which of these observations is real data and which is just noise introduced by people not doing the task, but one reasonable approach is to plot this histogram as a boxplot.



- In `ggplot`, the default is to extend the whiskers to 1.5 times the interquartile range (see `?IQR`), and to treat everything beyond those whiskers as an outlier (plotted as filled black circles). Lets go along with this idea by removing from our data all observations beyond the whiskers and again plotting the resulting histogram and boxplot.



- This looks much more reasonable, but we do see that the box plot still has dots beyond the whisker (i.e., outliers). So why not continue to exclude until we get a box plot with no points beyond the whiskers?
- A major concern when making exclusions is that by cherry picking your data you will be introducing a result that isn't really there and in doing so, you will be making the universe a very sad place.
- The more we exclude the more we risk making this mistake.
- We can check if we've done this here by asking if the observations we've excluded come more from congruent or incongruent trial types. If they come from both about equally, then the universe will approve of our exclusions in this case.
- First, report how many congruent / incongruent trials there are that survive our exclusion.

```
##      condition     N
##         <char> <int>
## 1:   congruent   806
## 2: incongruent   160
```

- It looks like there is a bit more than a 4 to 1 ratio of congruent to incongruent trials.
- Next, report how many congruent / incongruent trials there are that that we have excluded.

```
##      condition     N
##         <char> <int>
## 1: incongruent    57
## 2:   congruent    59
```

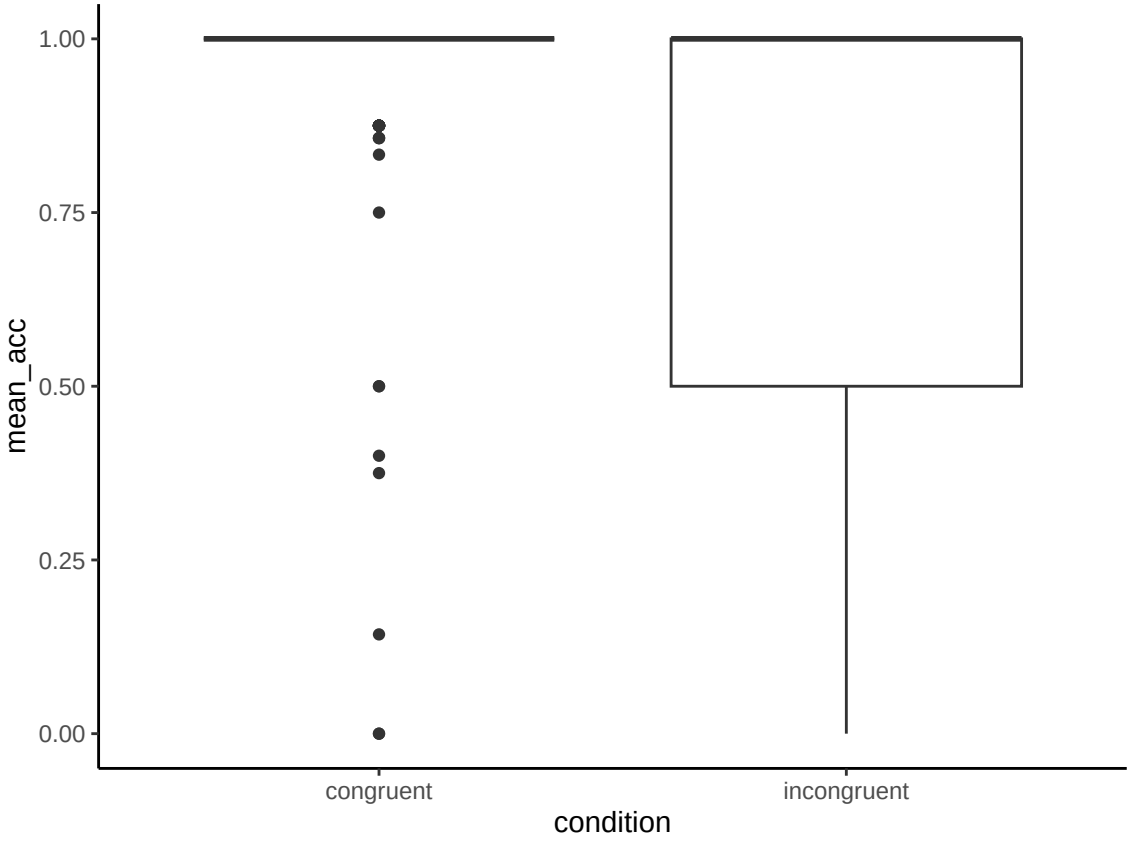
- There is an about equal number of incongruent and congruent trials excluded, but since there are more congruent trials overall, the proportion of excluded incongruent trials is higher than the proportion of

excluded congruent trials. Since trial type is central to our question, we cannot make this exclusion comfortably. So it looks like we might be stuck with a few very long resposne times.

- Moving forward, use a the data.table that you generated just before making exlcusions on the basis of the IQR and box plot.

3. Look for the Flanker effect

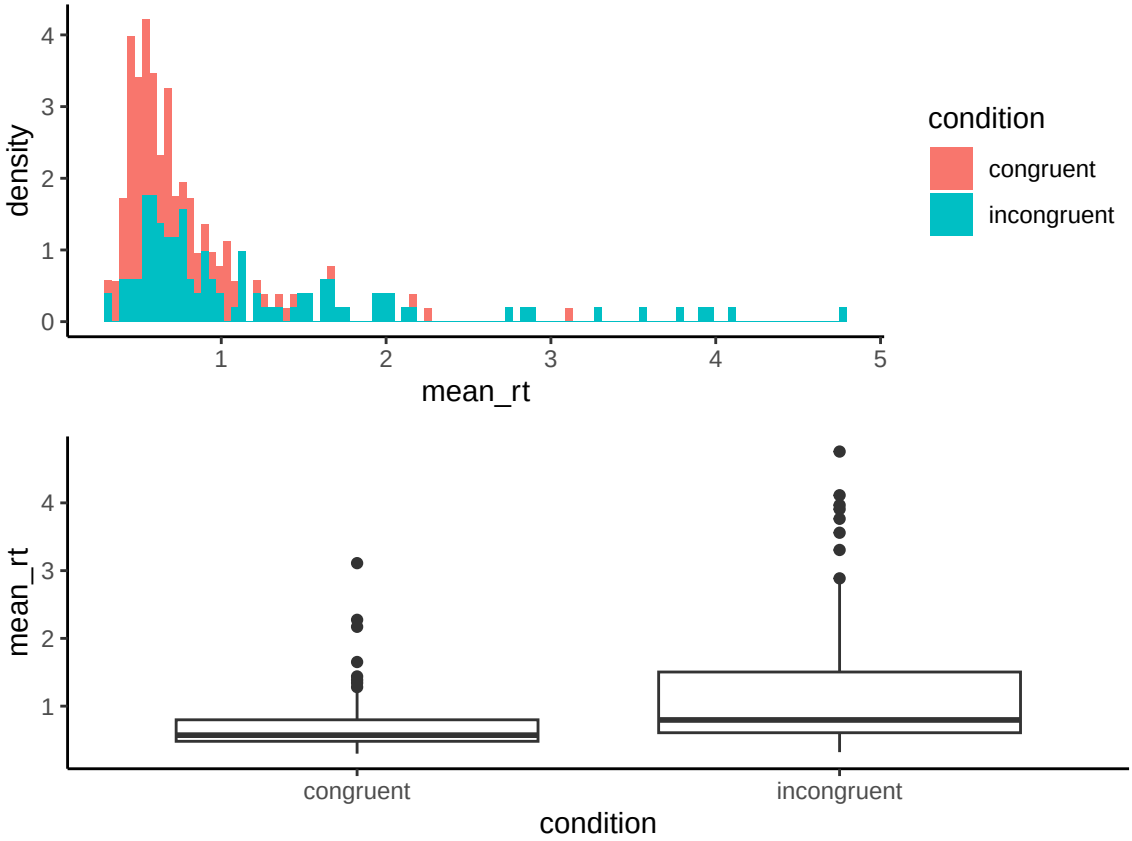
- **This step contains 1 stand alone `ggplot` figure, 2 `ggplot` figures arranged in a single column, and 2 `R` outputs reporting t-test results.**
- The classic finding in a flanker experiment is that accuracy and reaction times are both better on congruent (e.g., `stimulus == '<<<<<'`) trials than on incongruent trials (e.g., `stimulus != '<<>><<'`). Test this hypothesis using an appropriate t-test and corresponding figure.



- We can see that there just aren't very many mistakes made at all on congruent trials, which makes the `mean_acc` for these trials very non-normally distributed. Consequently, a t-test isn't the best idea for this particular inference (because it assumes normality).
- However, since we haven't covered any alternative approaches that deal with non-normality – and because t-tests are fairly robust to violations of normality – we will go ahead and use the most appropriate t-test we can muster.
- The universe isn't too sad about this because it's just a homework problem, but please be aware of this sort of thing and make better decisions as you grow into your careers.

```
##
##  Welch Two Sample t-test
##
## data:  dd[condition == "congruent", mean_acc] and dd[condition == "incongruent", mean_acc]
## t = 7.1163, df = 153.27, p-value = 1.996e-11
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
##  0.2285249       Inf
## sample estimates:
## mean of x mean of y
```

```
## 0.9349374 0.6371681
```

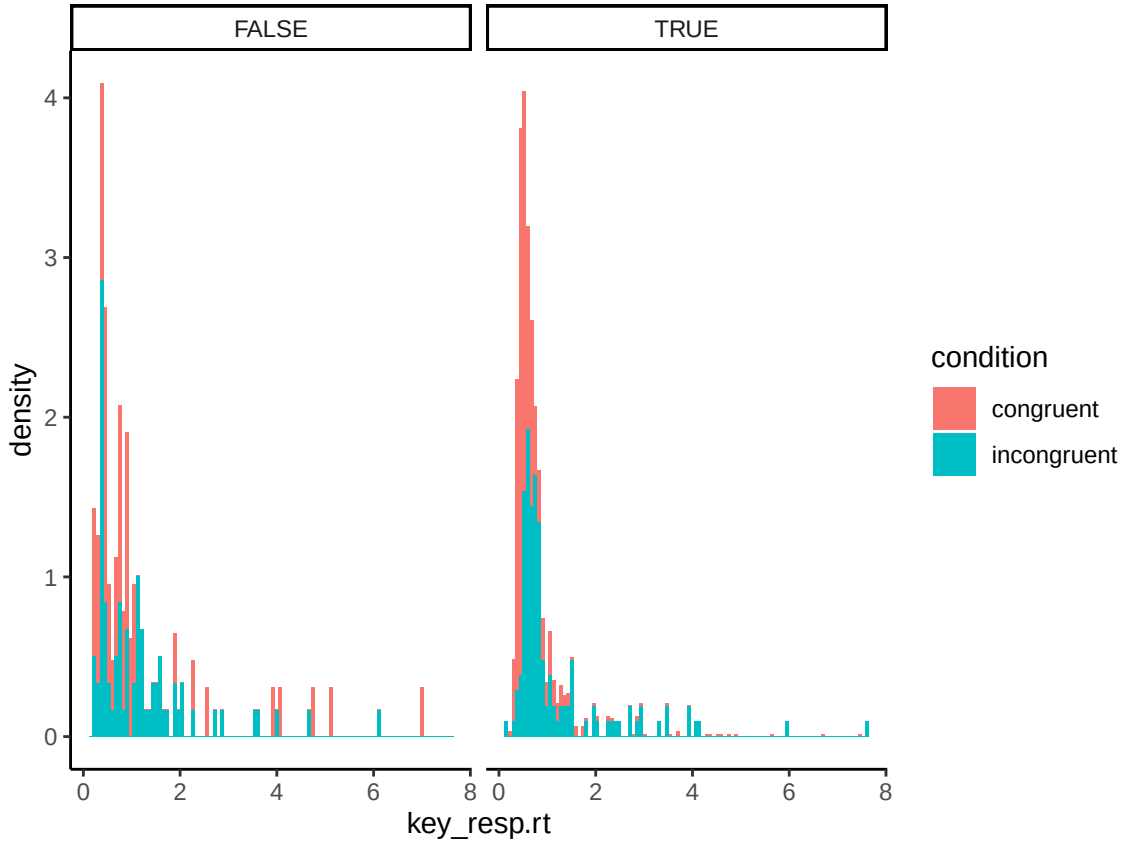


- The distribution of response times is also not very normal primarily because it is skewed. There are many approaches to dealing with non-normality and unfortunately we probably won't cover them in this unit. As such we will just carry on with the t-test with the same caveats as above.
- Here is the t-test:

```
##
##  Welch Two Sample t-test
##
## data:  dd[condition == "congruent", mean_rt] and dd[condition == "incongruent", mean_rt]
## t = -5.1719, df = 152.21, p-value = 3.598e-07
## alternative hypothesis: true difference in means is less than 0
## 95 percent confidence interval:
##        -Inf -0.3234202
## sample estimates:
## mean of x mean of y
## 0.7037691 1.1793765
```

4. Look for a speed-accuracy trade-off

- **This step contains 2 `ggplot` figure arranged into a single row and 1 `R` output reporting a t-test result.**
- As a final inspection we will look for a speed-accuracy trade-off in our data. It is common to many behaviours that fast responses are more prone to error than slow responses. Do we see that here?
- As a first try, lets look at the mean accuracy and mean response time per subject. Lets inspect this with a histogram:



- It is difficult to say, so any difference we do observe is likely to be pretty small. Lets ask using formal statistics.
- Perform an appropriate t-test assessing whether or not there is a significant difference in response time on correct vs incorrect trials. Since each subject may be more or less fast overall, it is important that we compute within-subject difference scores and perform inference on these.

```
##
##  One Sample t-test
##
## data:  D[, V1]
## t = -2.8058, df = 67, p-value = 0.006565
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -0.6522245 -0.1099875
## sample estimates:
## mean of x
## -0.381106
```