Purrr, Expand.grid and Furrr

A brief overview

Michael Truong

Contents

Purrr is part of tidyverse, but I will load tidyverse instead because I also need ggplot & dplyr.

1 Generate Fake Data

```
# We store the data in the list "obs"
obs <- list()

# This is the generated data for the hit rate
obs$nback.hit <- data.frame(
  Participant = c(1:50),
  Condition = rep(c("control", "treatment"), each = 25),
  NBack = as.factor(rep(c(1, 2), each = 100)),
```

```
  Stimuli = rep(c("Pictures", "Strings"), each = 50),
  Score = c(runif(
    n = 200, min = .5, max = .9
  ))
)

# This is the generated data for the false alarm rate
obs$nback.false_alarm <- data.frame(
  Participant = c(1:50),
  Condition = rep(c("control", "treatment"), each = 25),
  NBack = as.factor(rep(c(1, 2), each = 100)),
  Stimuli = rep(c("Pictures", "Strings"), each = 50),
  Score = c(runif(
    n = 200, min = .3, max = .7
  ))
)
```

2 Brief look at the data structure

Hit Rates

Participant	Condition	NBack	Stimuli	Score
1	control	1	Pictures	0.894
1	control	1	Strings	0.715
1	control	2	Pictures	0.663
1	control	2	Strings	0.734
26	treatment	1	Pictures	0.804
26	treatment	1	Strings	0.779
26	treatment	2	Pictures	0.801
26	treatment	2	Strings	0.720

False Alarm Rates

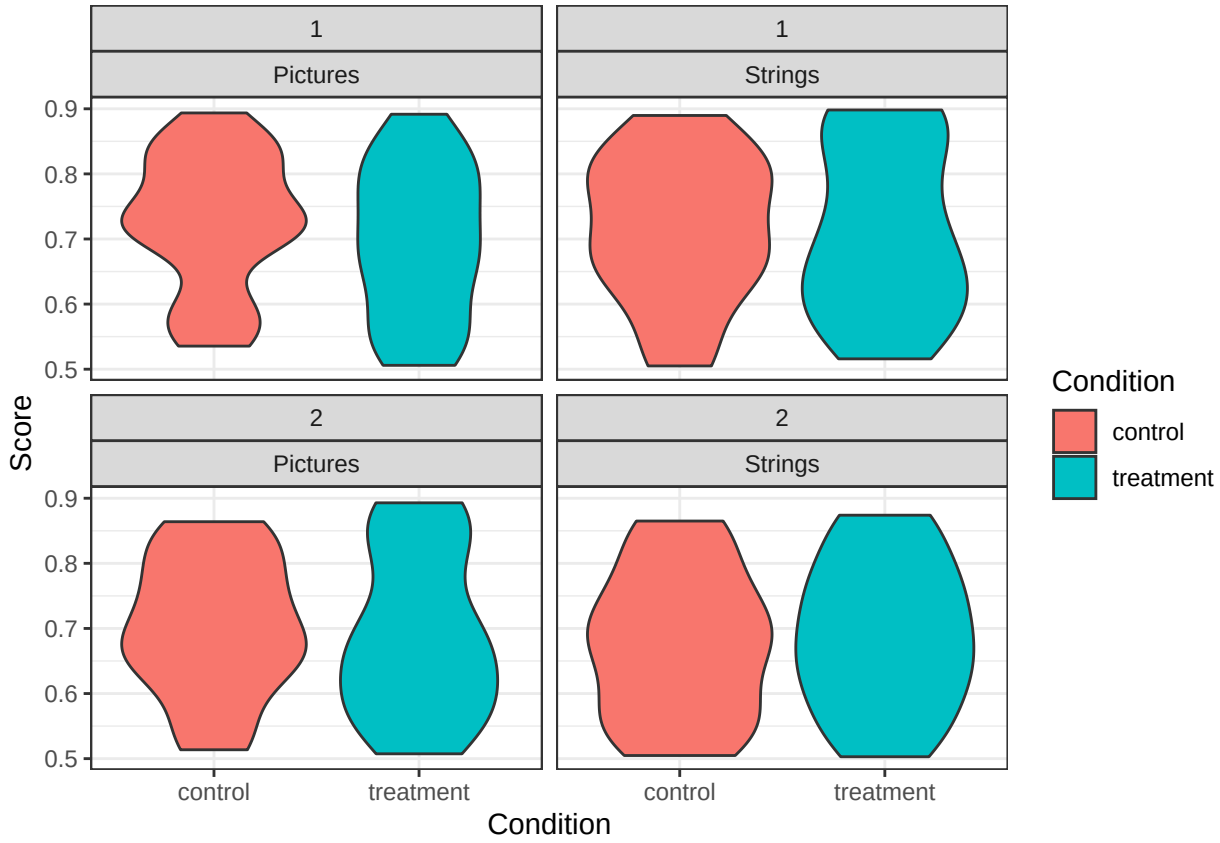
Participant	Condition	NBack	Stimuli	Score
1	control	1	Pictures	0.526
1	control	1	Strings	0.463
1	control	2	Pictures	0.612
1	control	2	Strings	0.486
26	treatment	1	Pictures	0.496
26	treatment	1	Strings	0.699
26	treatment	2	Pictures	0.492
26	treatment	2	Strings	0.681

3 Purrr example

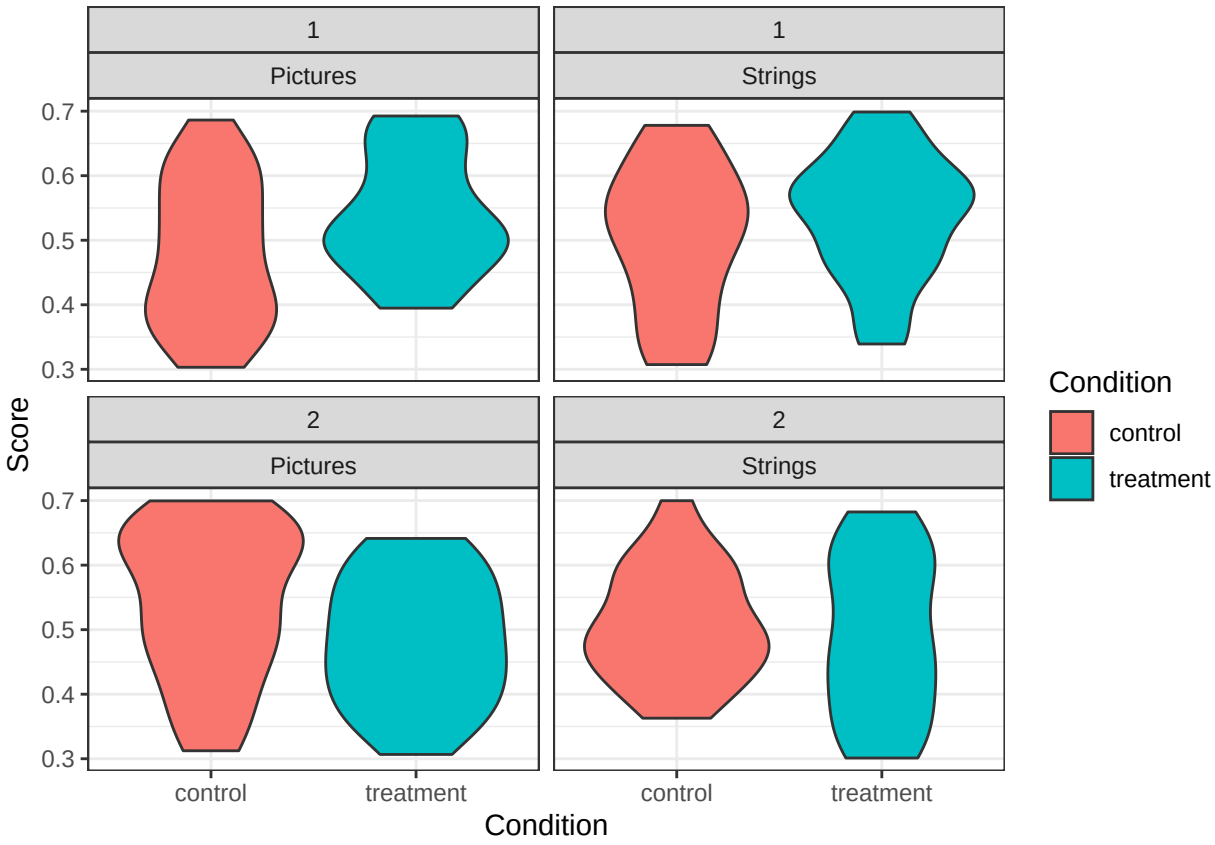
3.1 Function

```
get.graph.per.df <- function(df) {
  graph <- df %>%
    ggplot(aes(x = Condition,
               y = Score,
               fill = Condition)) +
    geom_violin() +
    theme_bw() +
    facet_wrap(vars(NBack, Stimuli))
  return(graph)
}
```

```
get.graph.per.df(df = obs$nback.hit)
```




```
get.graph.per.df(df = obs$nback.false_alarm)
```



3.2 One Step Map

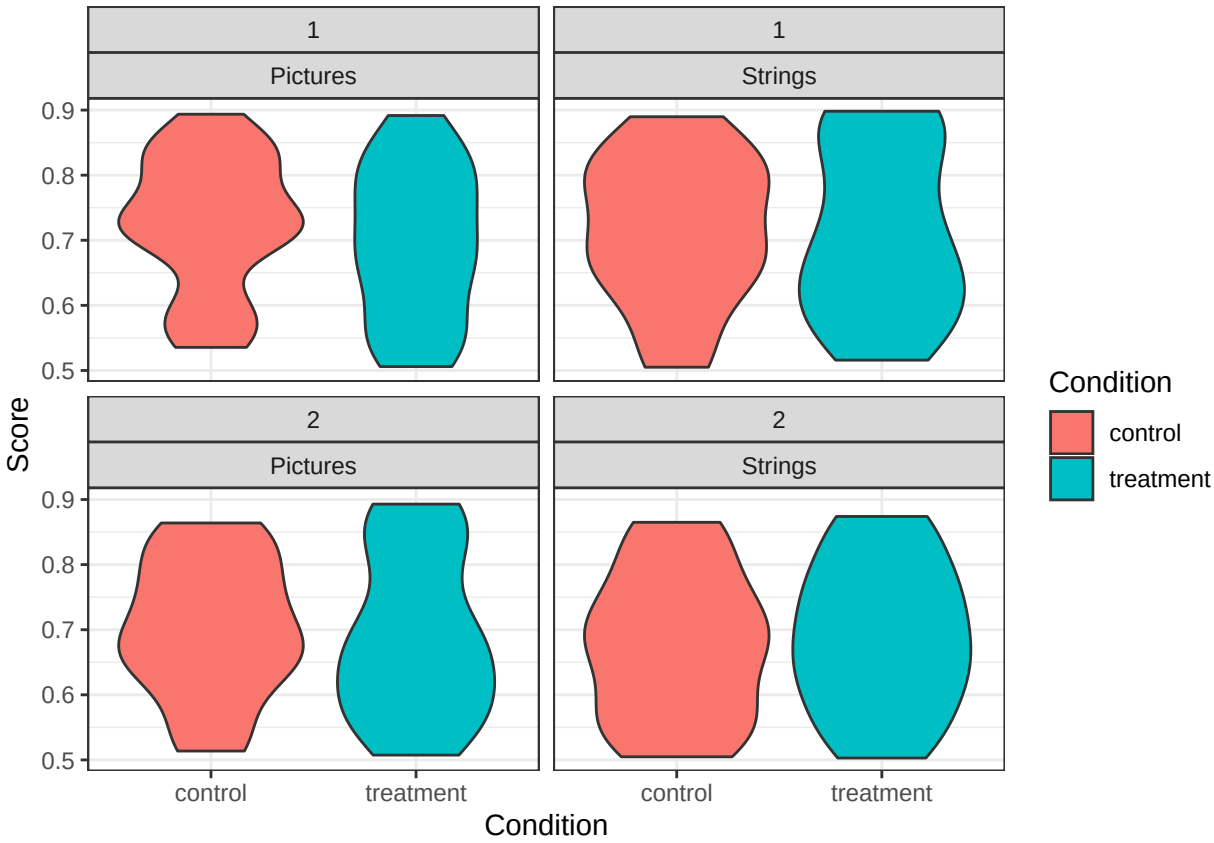
Use map() to apply the function to each element (each data frame) in obs.

```
str(obs)
```

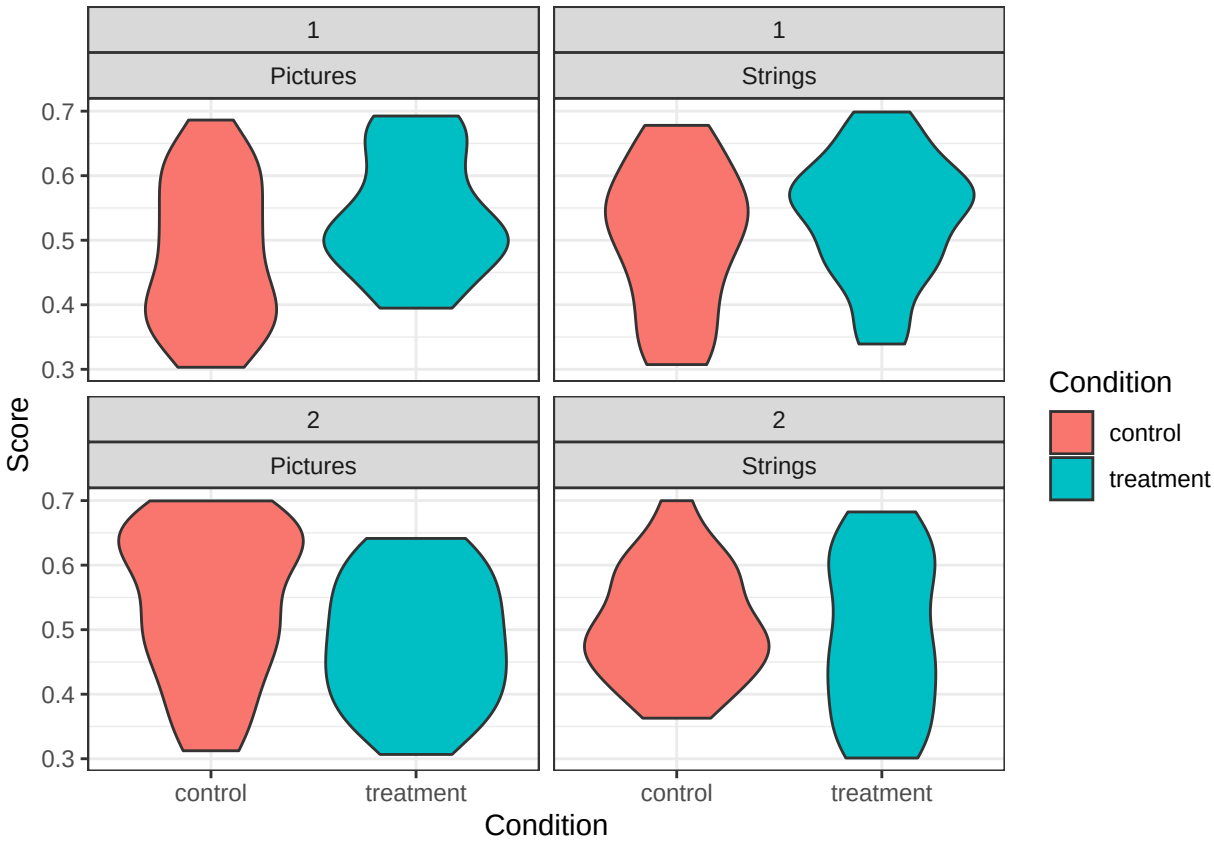
```
## List of 2
##  $ nback.hit        :'data.frame':   200 obs. of  5 variables:
##   ..$ Participant: int [1:200] 1 2 3 4 5 6 7 8 9 10 ...
##   ..$ Condition  : chr [1:200] "control" "control" "control" "control" ...
##   ..$ NBack      : Factor w/ 2 levels "1","2": 1 1 1 1 1 1 1 1 1 1 ...
##   ..$ Stimuli    : chr [1:200] "Pictures" "Pictures" "Pictures" "Pictures" ...
##   ..$ Score      : num [1:200] 0.894 0.567 0.836 0.811 0.789 ...
##  $ nback.false_alarm:'data.frame':   200 obs. of  5 variables:
##   ..$ Participant: int [1:200] 1 2 3 4 5 6 7 8 9 10 ...
##   ..$ Condition  : chr [1:200] "control" "control" "control" "control" ...
##   ..$ NBack      : Factor w/ 2 levels "1","2": 1 1 1 1 1 1 1 1 1 1 ...
##   ..$ Stimuli    : chr [1:200] "Pictures" "Pictures" "Pictures" "Pictures" ...
##   ..$ Score      : num [1:200] 0.526 0.396 0.391 0.504 0.381 ...
```

```
purrr::map(.x = obs,.f = get.graph.per.df)
```

```
## $nback.hit
```




```
##
## $nback.false_alarm
```



4 Expand.grid

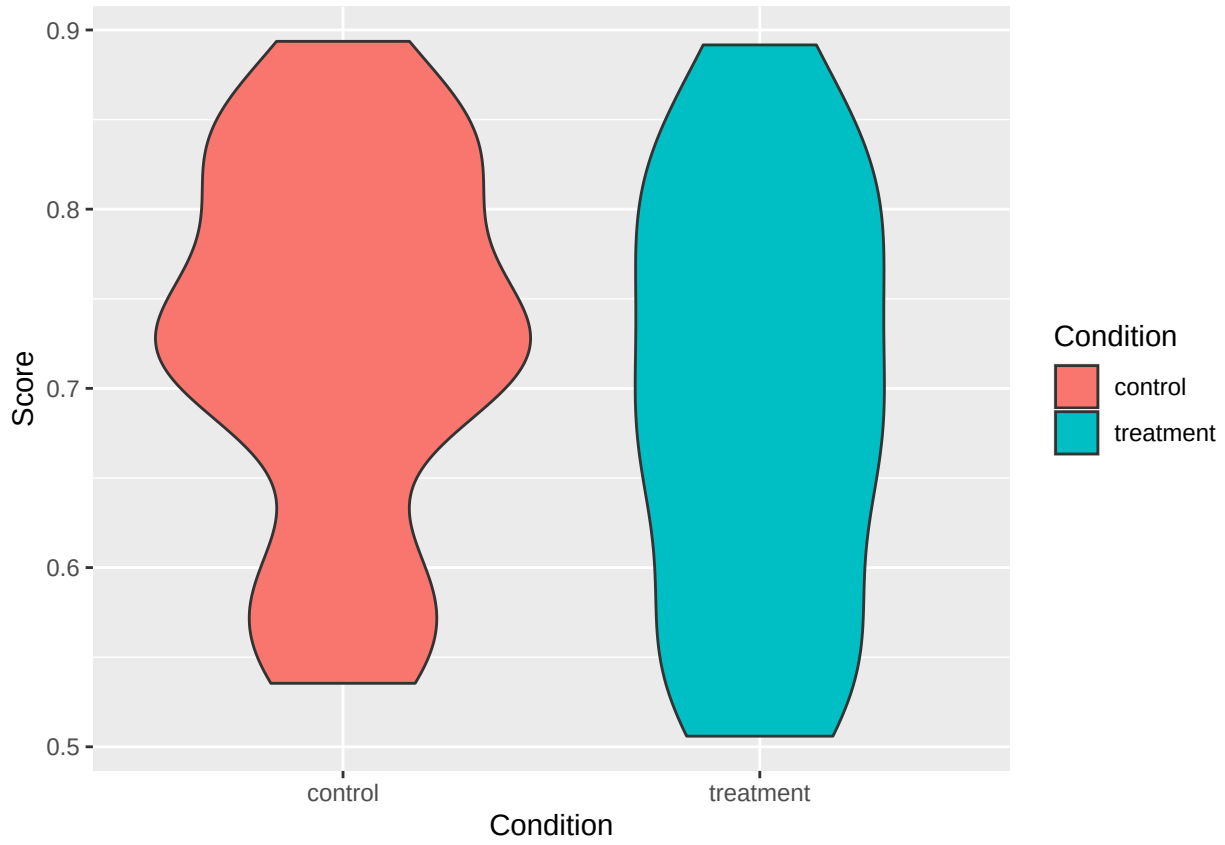
4.1 Simple Function Approach

Here is the function that we will use

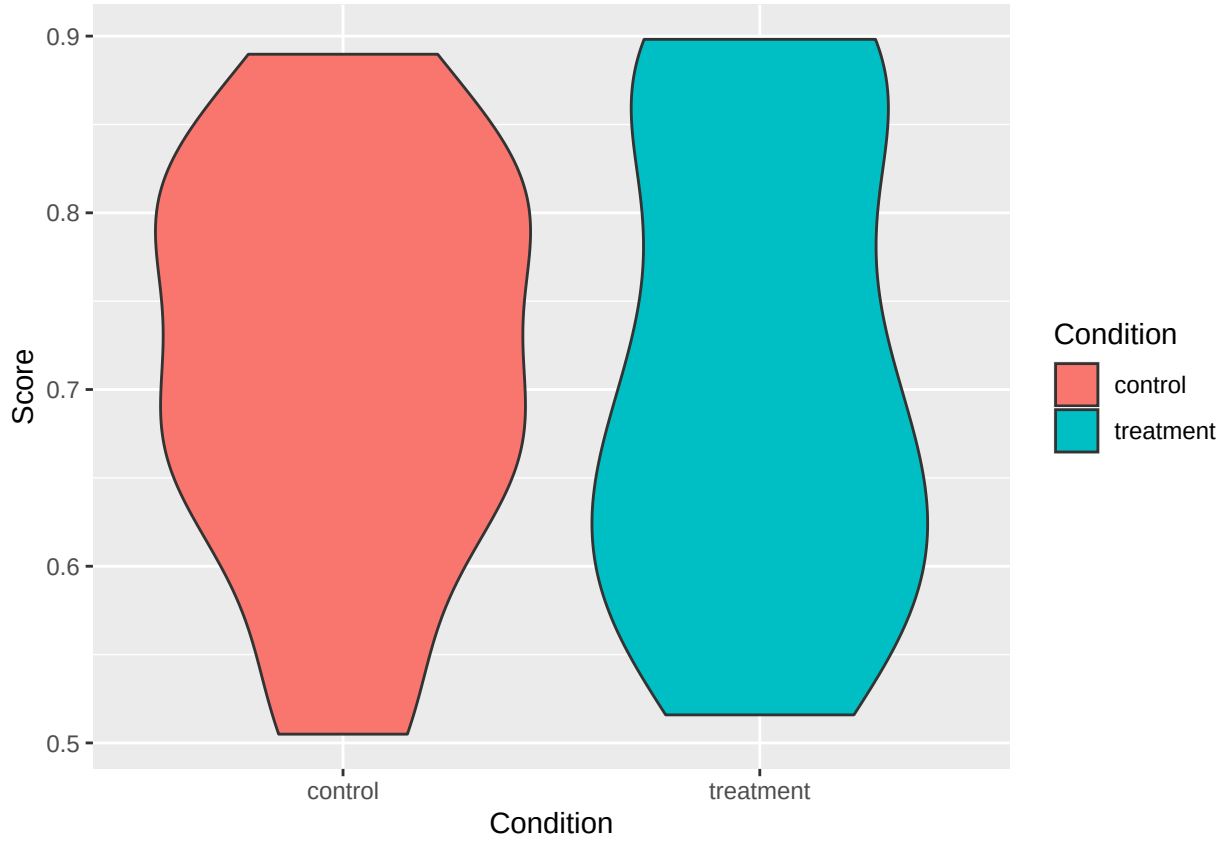
```
get.graph <- function(df, NBackNumber, StimuliType) {

  graph <- df %>%
    dplyr::filter(NBack == NBackNumber,
           Stimuli == StimuliType) %>%
    ggplot(aes(x = Condition, y = Score, fill = Condition)) +
    geom_violin()

  return(graph)
}
```

Here I apply the function to each individual combination

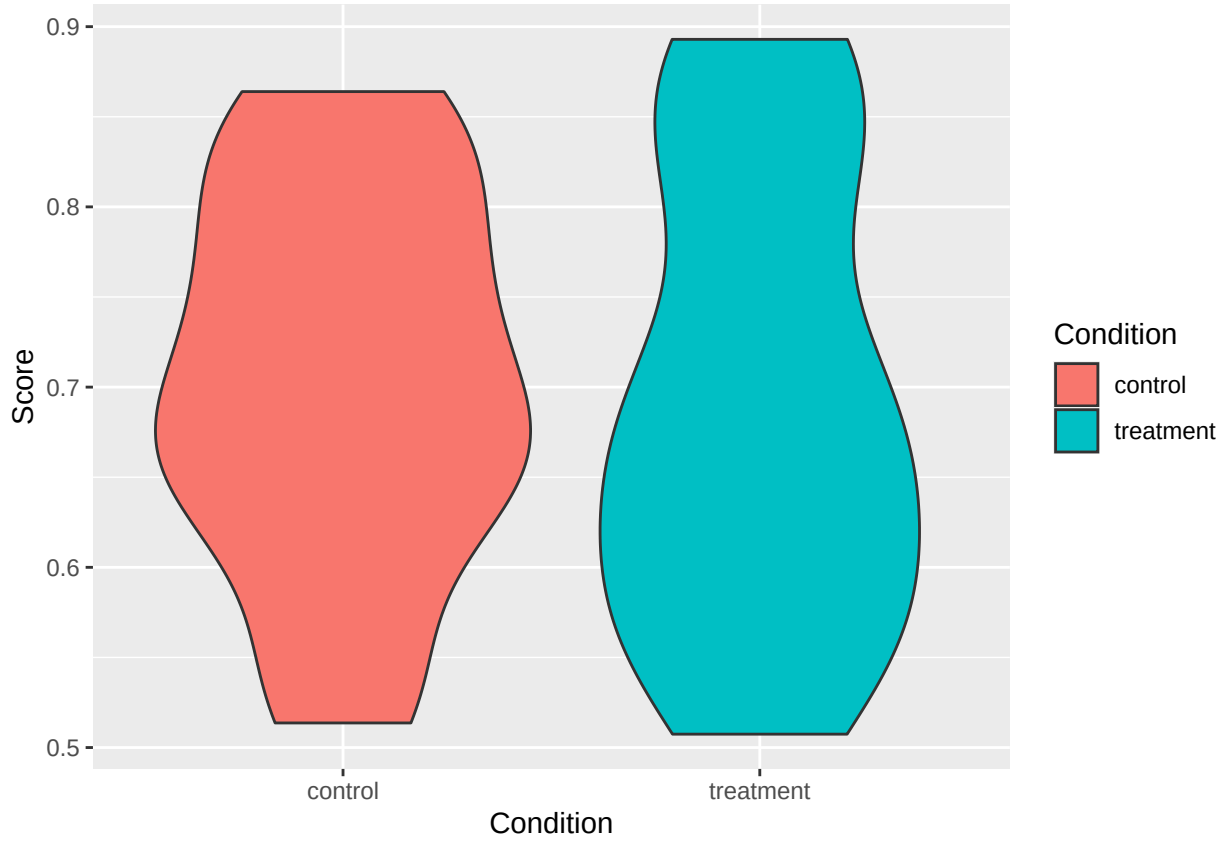
```
get.graph(
  df = obs$nback.hit,
  NBackNumber = 1,
  StimuliType = "Pictures"
)
```



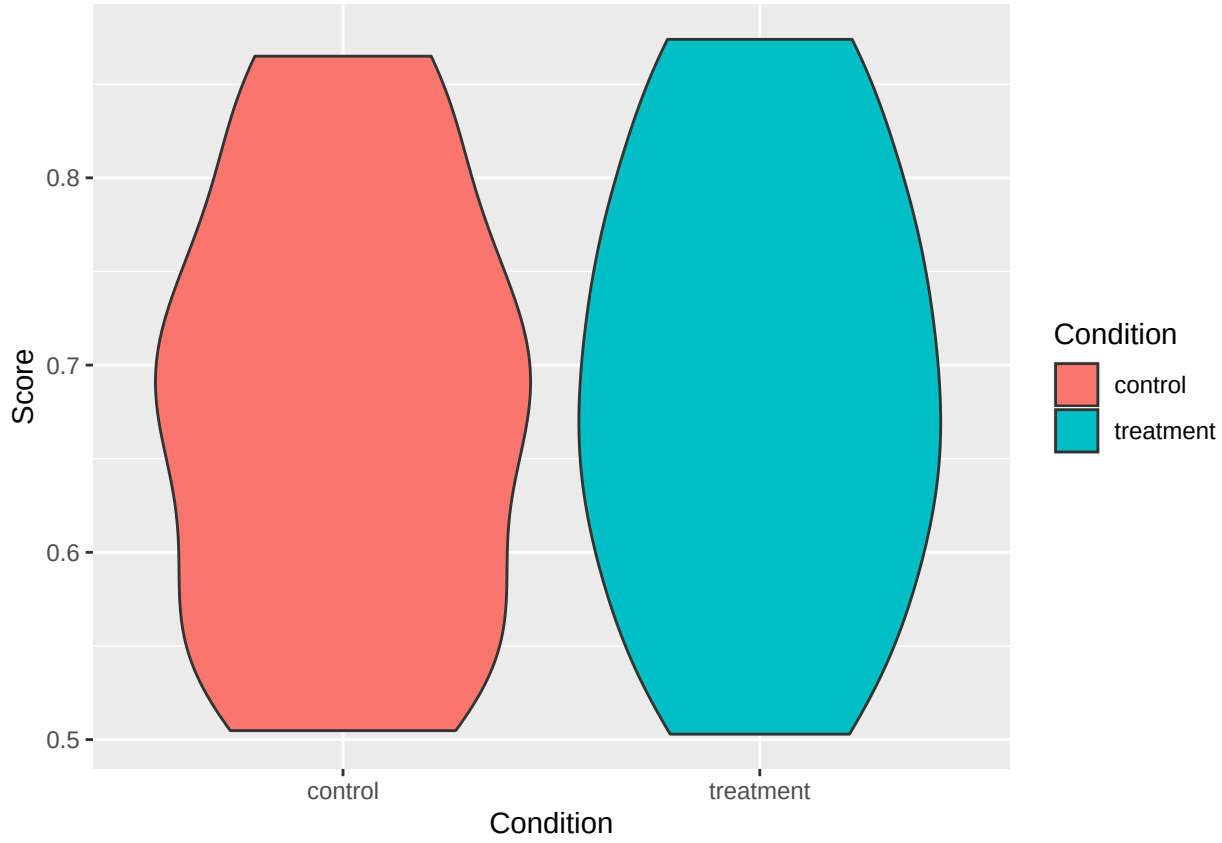

```
get.graph(
  df = obs$nback.hit,
  NBackNumber = 1,
  StimuliType = "Strings"
)
```



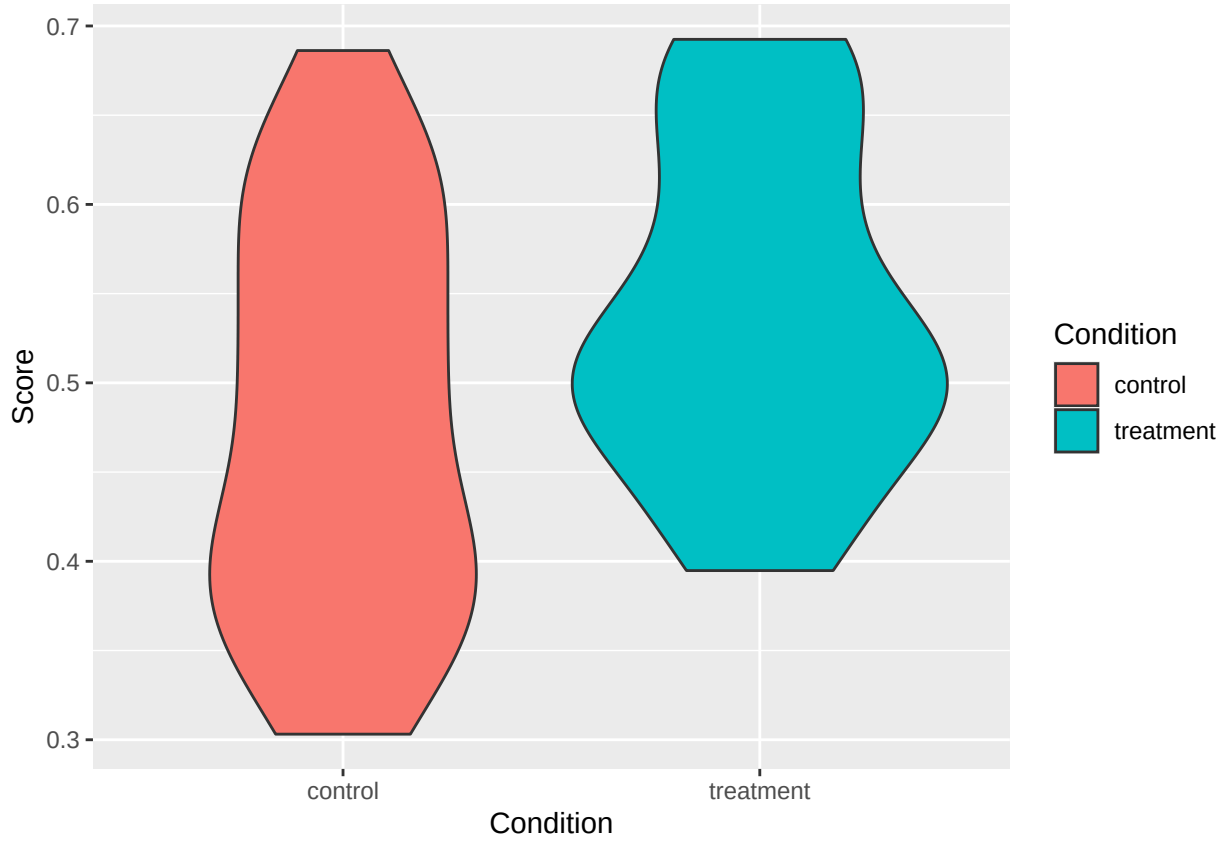

```
get.graph(
  df = obs$nback.hit,
  NBackNumber = 2,
  StimuliType = "Pictures"
)
```



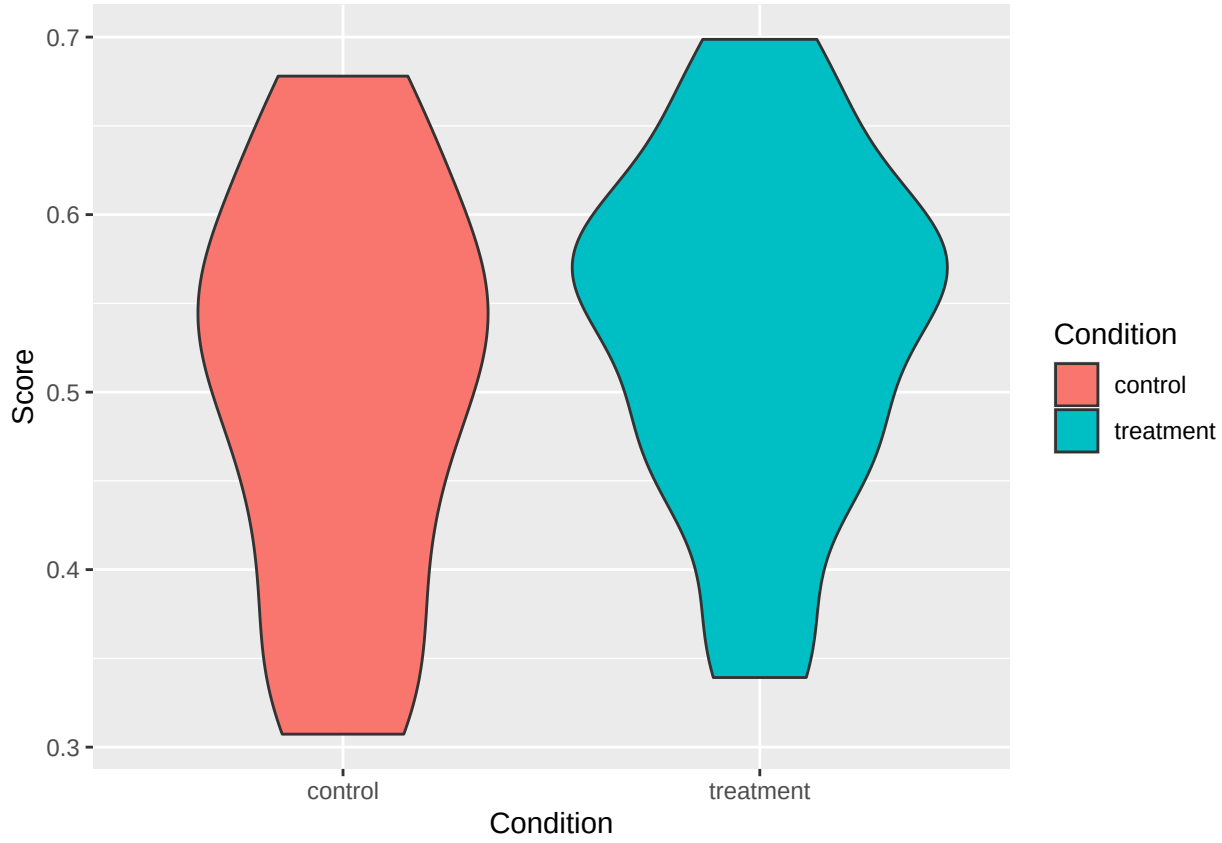

```
get.graph(
  df = obs$nback.hit,
  NBackNumber = 2,
  StimuliType = "Strings"
)
```



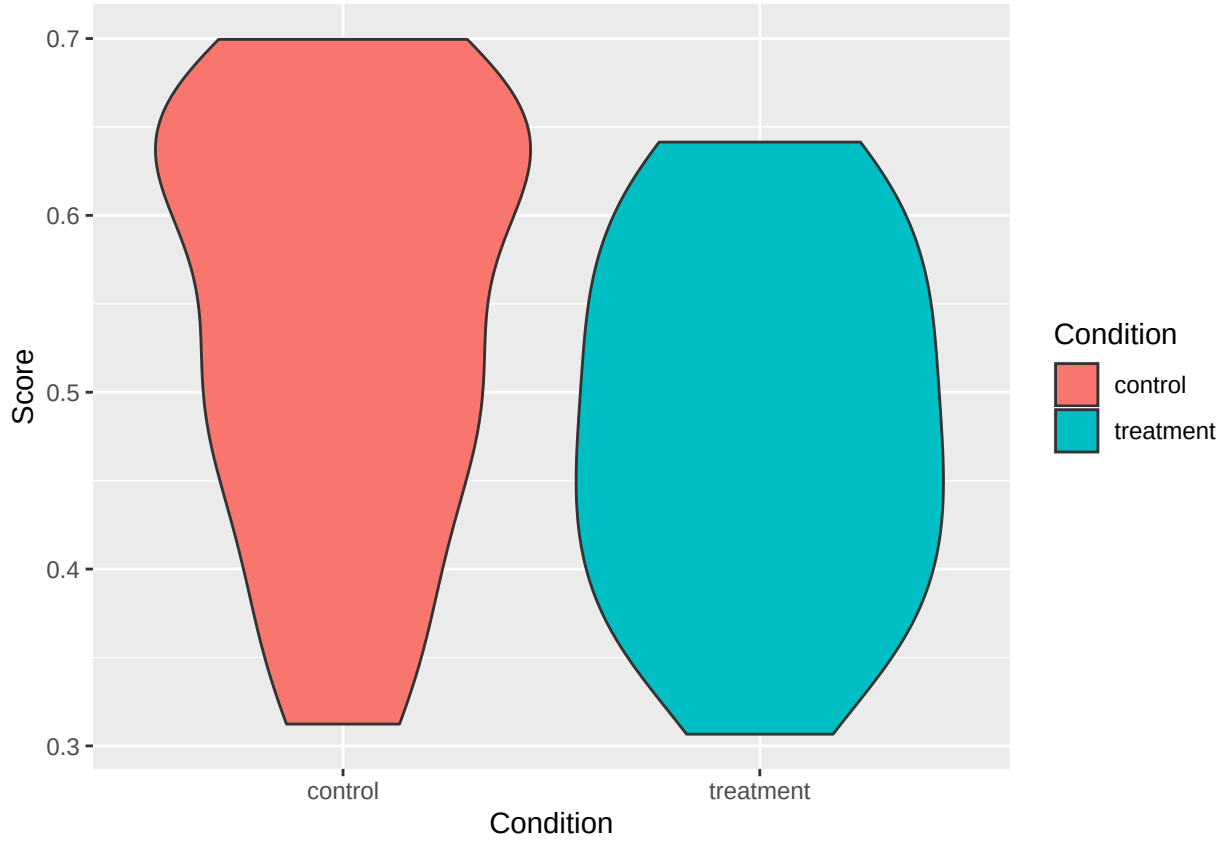

```
get.graph(
  df = obs$nback.false_alarm,
  NBackNumber = 1,
  StimuliType = "Pictures"
)
```



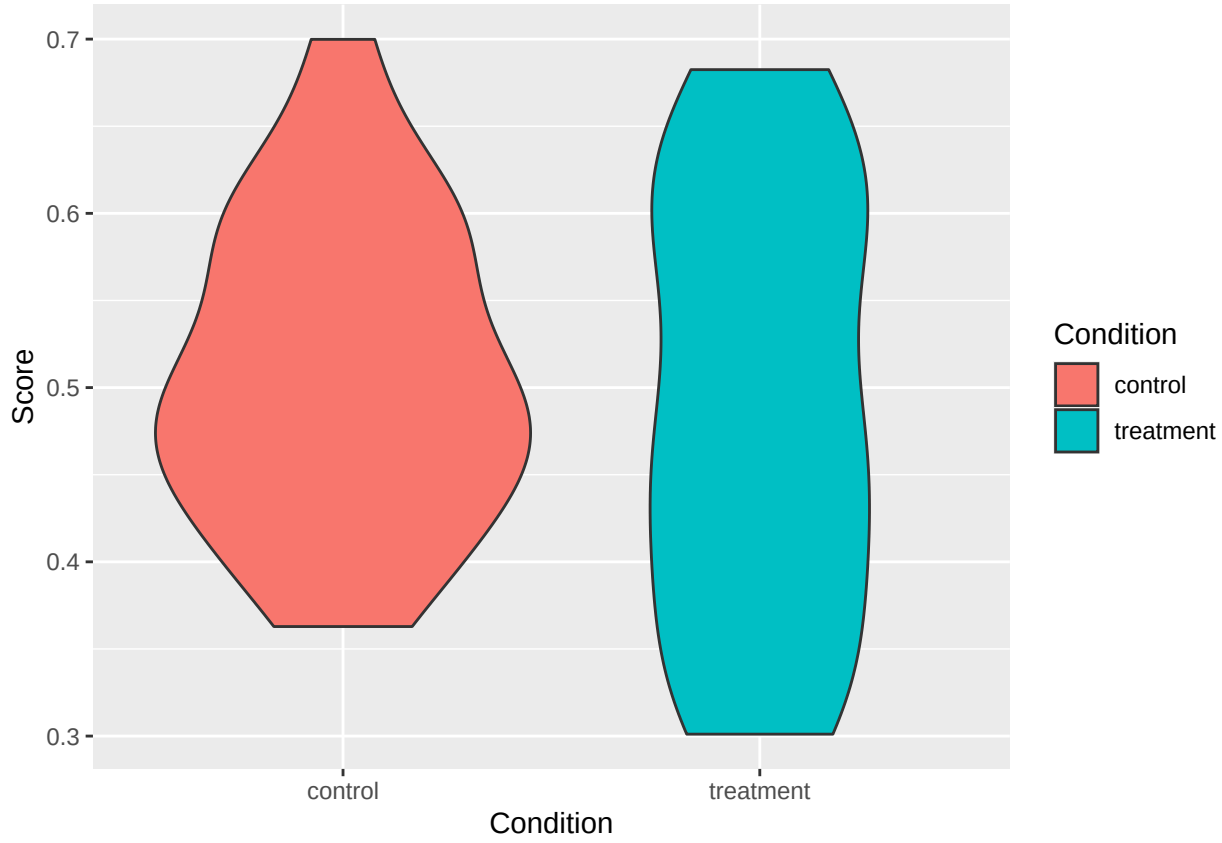

```
get.graph(
  df = obs$nback.false_alarm,
  NBackNumber = 1,
  StimuliType = "Strings"
)
```




```
get.graph(
  df = obs$nback.false_alarm,
  NBackNumber = 2,
  StimuliType = "Pictures"
)
```




```
get.graph(
  df = obs$nback.false_alarm,
  NBackNumber = 2,
  StimuliType = "Strings"
)
```



Very long and repetitive, error prone.

4.2 Expand.grid and purrr approach

4.2.1 Making a combination of arguments

```
arguments <- expand.grid(
  df = list(obs$nback.hit, obs$nback.false_alarm),
  NBackNumber = c(1, 2),
  StimuliType = c("Pictures", "Strings")
)

df.name <- (as.vector(rep(c("Hit", "False Alarm"), times = 4)))
```

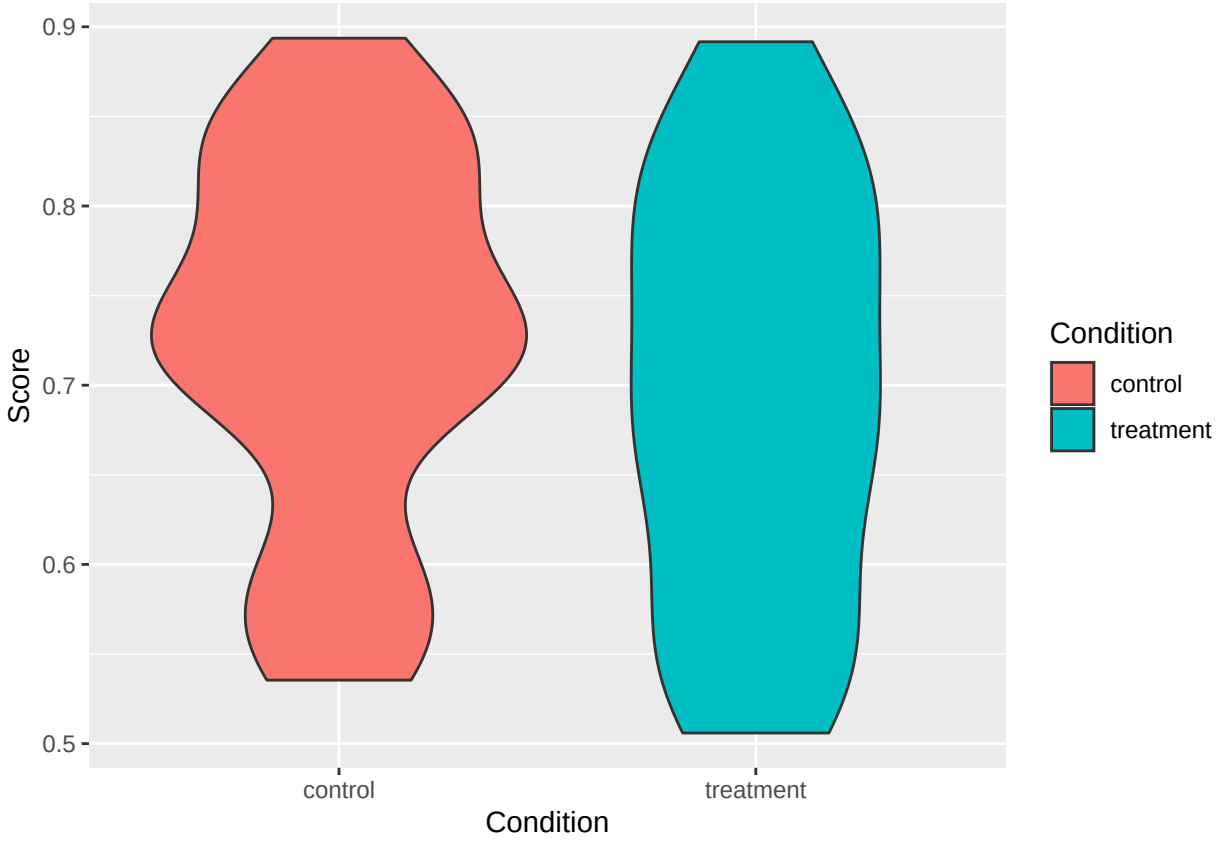
```
cbind(df.name,arguments[,c(2,3)]) %>%
  kable()
```

df.name	NBackNumber	StimuliType
Hit	1	Pictures
False Alarm	1	Pictures
Hit	2	Pictures
False Alarm	2	Pictures
Hit	1	Strings
False Alarm	1	Strings
Hit	2	Strings
False Alarm	2	Strings

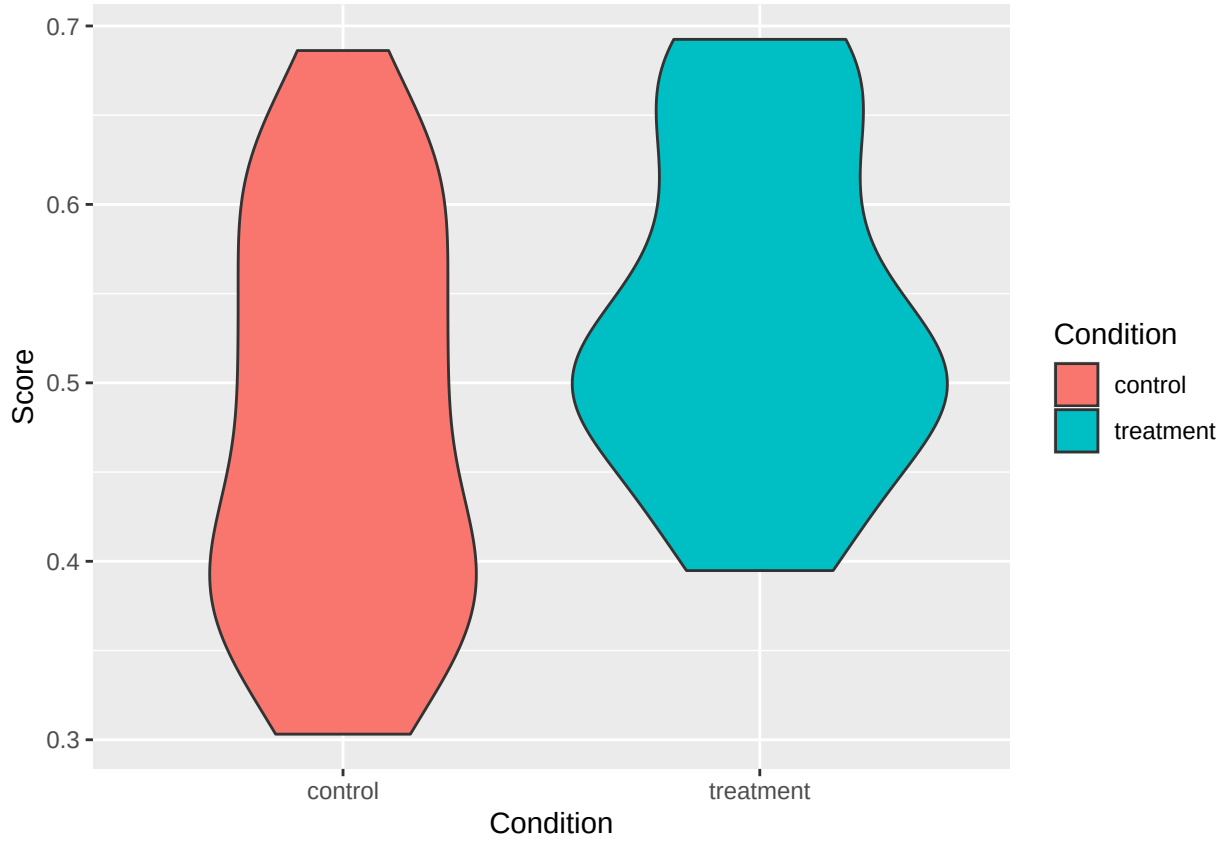
4.2.2 Using purrr to visualize the data

```
purrr::pmap(.l = arguments, .f = get.graph)
```

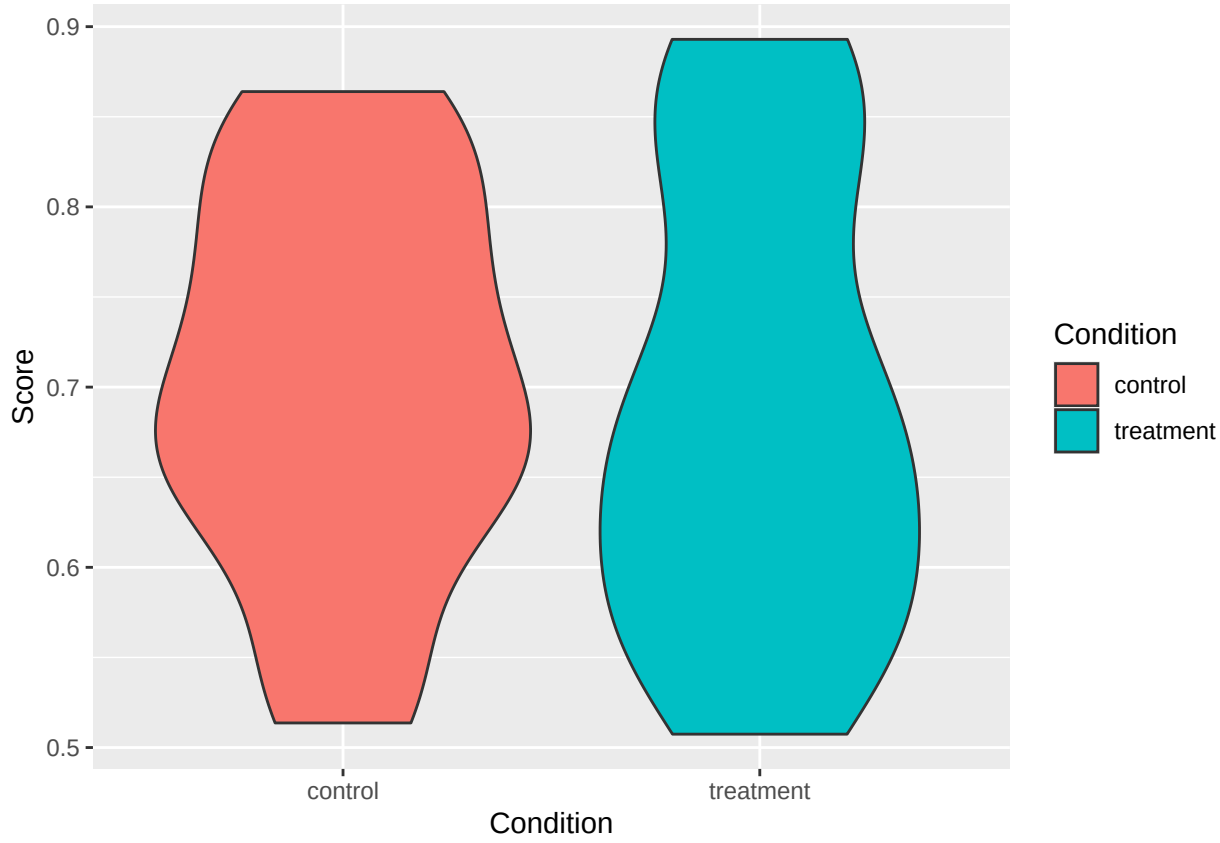
```
## [[1]]
```



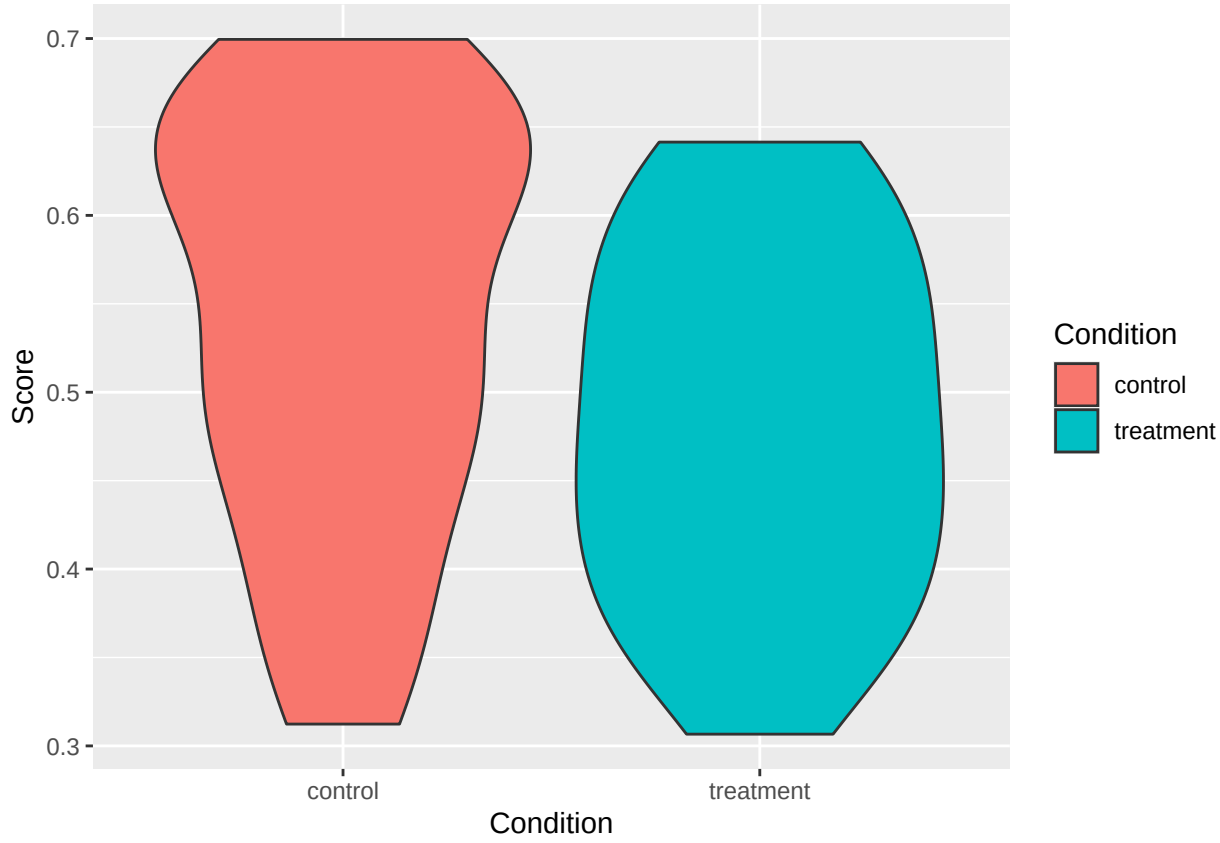

```
##
## [[2]]
```


0.7
0.6
0.5
0.4
0.3
Score
control
treatment
Condition
Condition
control
treatment

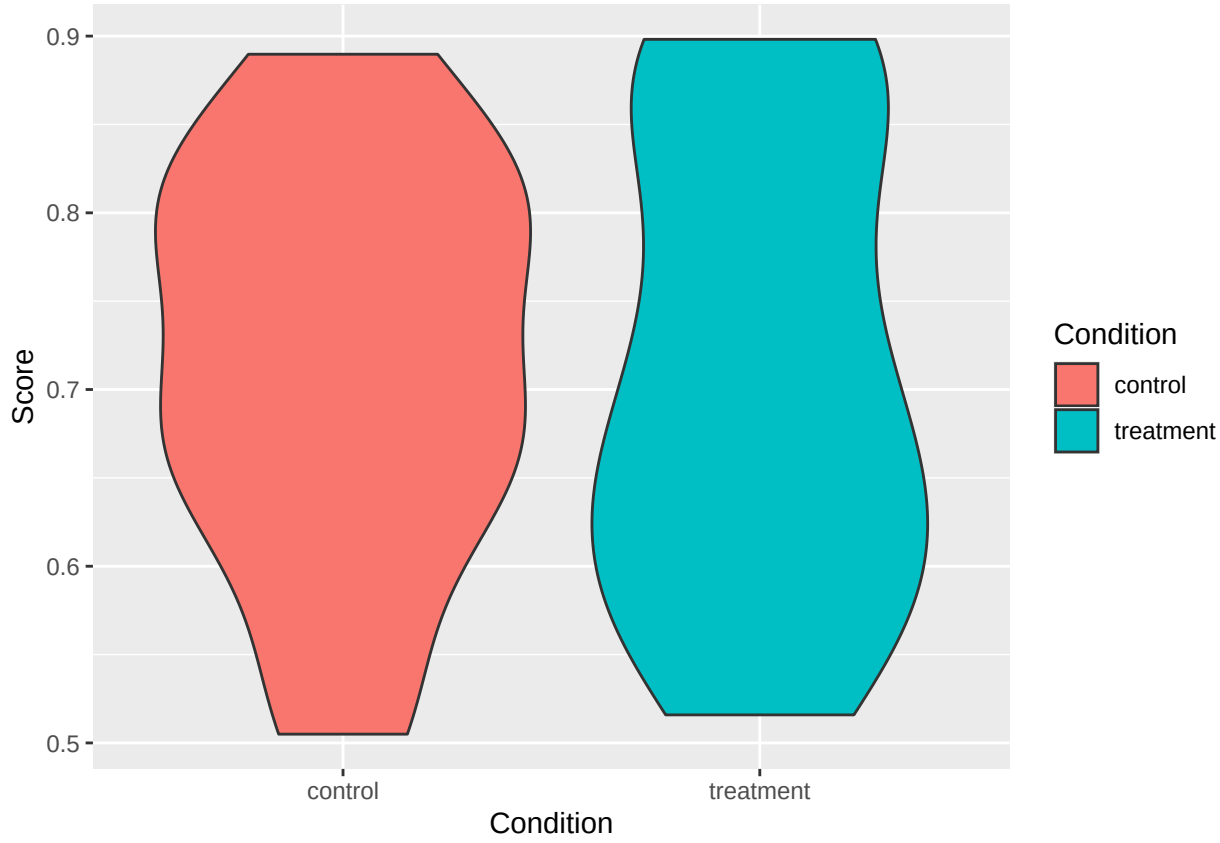
```
##
## [[3]]
```


0.9
0.8
0.7
0.6
0.5
Score
control
treatment
Condition
Condition
control
treatment

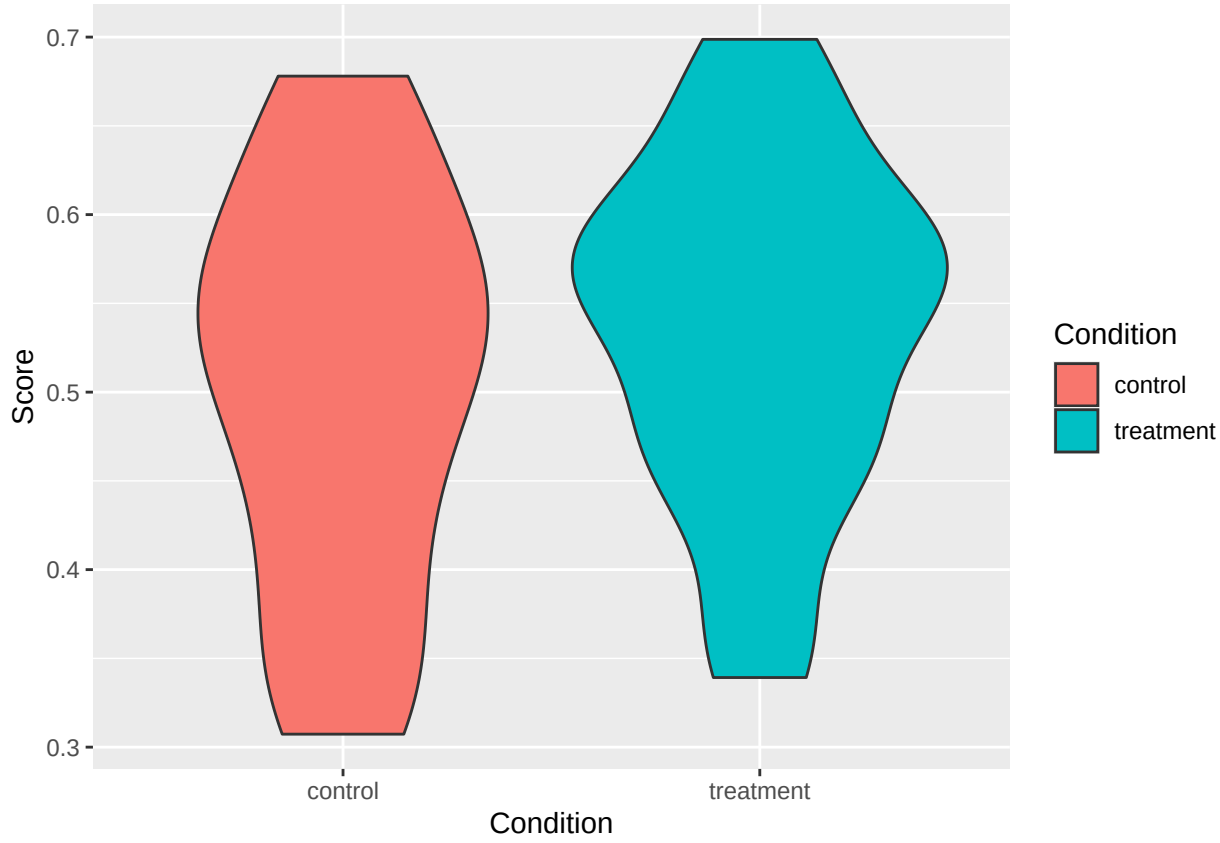
```
##
## [[4]]
```


0.7
0.6
0.5
0.4
0.3
Score
control
treatment
Condition
Condition
control
treatment

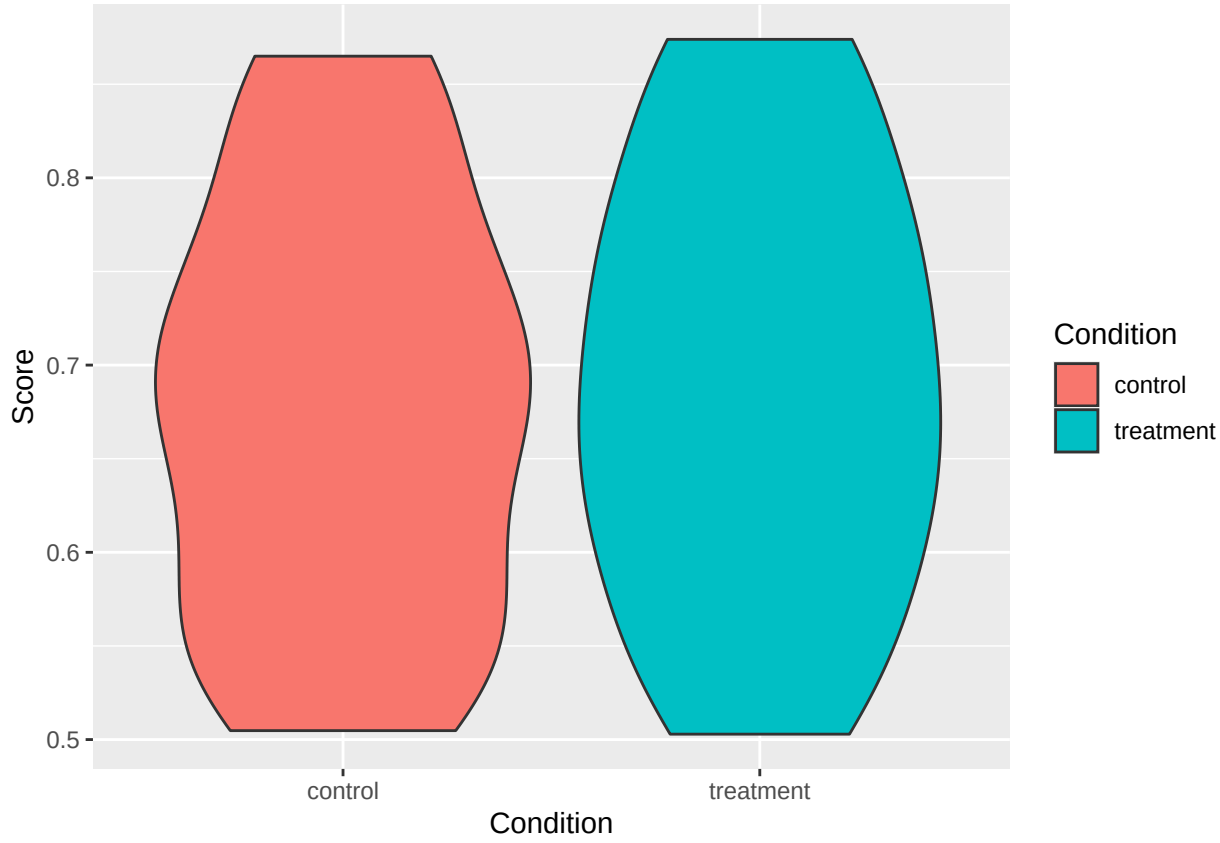
```
##
## [[5]]
```


0.9
0.8
0.7
0.6
0.5
Score
control
treatment
Condition
Condition
control
treatment

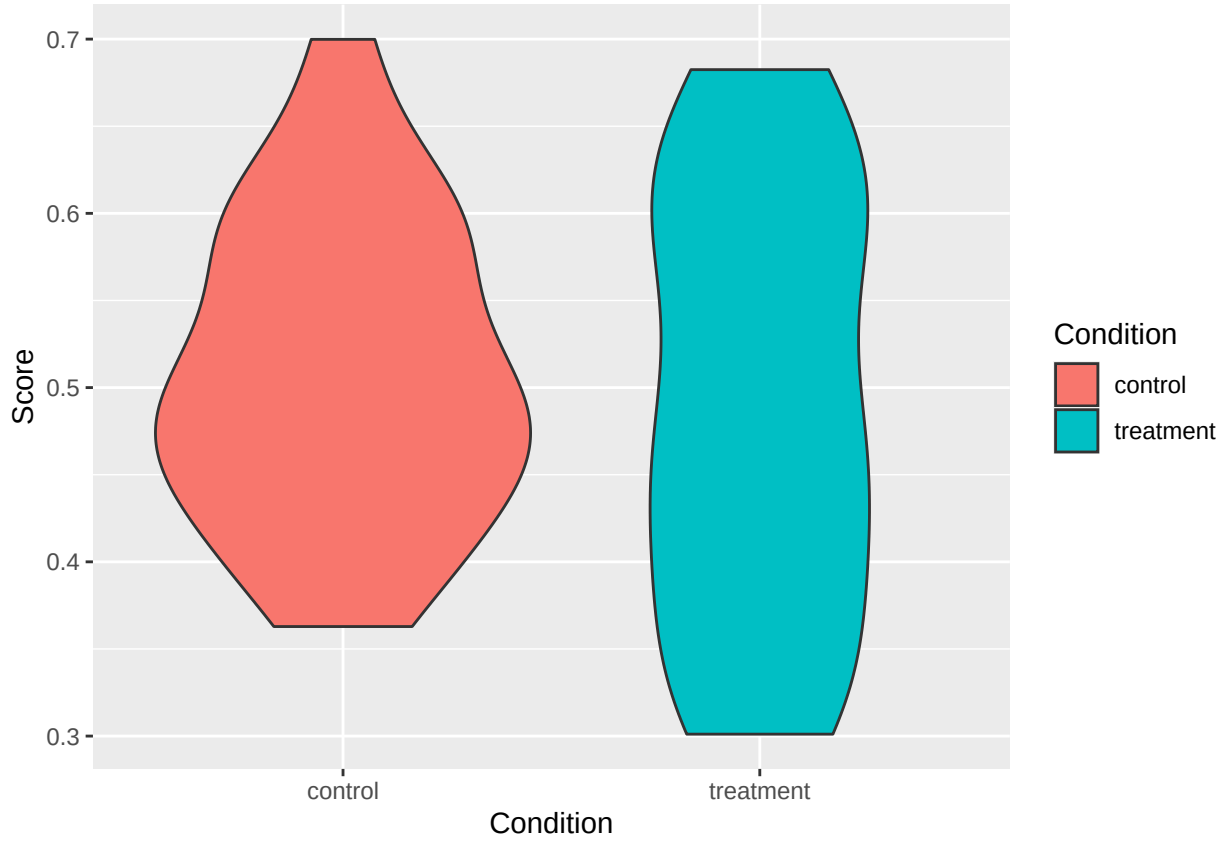
```
##
## [[6]]
```


0.7
0.6
0.5
0.4
0.3
Score
control
treatment
Condition
Condition
control
treatment

```
##
## [[7]]
```


0.8
0.7
0.6
0.5
Score
control
treatment
Condition
Condition
control
treatment

```
##
## [[8]]
```



5 Furrr example

5.1 Initialize future

```
future::plan(multisession, workers = 4)
```

Here I let it do 4 at a time, so *normally* it will be faster, but see Amdahl's law. Furthermore, perhaps grouped dataframes could be done differently based on this

5.2 Parallelized Visualization

```
# Single
tictoc::tic()
single <- purrr::pmap(.l = arguments, .f = get.graph)
tictoc::toc()
```

```
## 0.02 sec elapsed
```

```
# Parallel
tictoc::tic()
parallelized <- furrr::future_pmap(.l = arguments, .f = get.graph)
tictoc::toc()
```

```
## 1.81 sec elapsed
```

6 Session Information

```
sessionInfo()
```

```
## R version 4.0.4 (2021-02-15)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 19043)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_Canada.1252  LC_CTYPE=English_Canada.1252
## [3] LC_MONETARY=English_Canada.1252 LC_NUMERIC=C
## [5] LC_TIME=English_Canada.1252
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] tictoc_1.0.1    furrr_0.2.2     future_1.21.0   forcats_0.5.1
##  [5] stringr_1.4.0   dplyr_1.0.6     purrr_0.3.4     readr_1.4.0
##  [9] tidyr_1.1.3     tibble_3.1.2    ggplot2_3.3.3   tidyverse_1.3.1
## [13] knitr_1.33
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_1.0.6         lubridate_1.7.10   listenv_0.8.0      assertthat_0.2.1
##  [5] digest_0.6.27      utf8_1.2.1         parallelly_1.25.0  R6_2.5.0
##  [9] cellranger_1.1.0   backports_1.2.1    reprex_2.0.0       evaluate_0.14
## [13] httr_1.4.2         highr_0.9          pillar_1.6.1       rlang_0.4.11
## [17] readxl_1.3.1       rstudioapi_0.13    rmarkdown_2.8      labeling_0.4.2
## [21] munsell_0.5.0      broom_0.7.6        compiler_4.0.4     modelr_0.1.8
## [25] xfun_0.23          pkgconfig_2.0.3    globals_0.14.0     htmltools_0.5.1.1
## [29] tidyselect_1.1.1   codetools_0.2-18   fansi_0.5.0        crayon_1.4.1
## [33] dbplyr_2.1.1       withr_2.4.2        grid_4.0.4         jsonlite_1.7.2
## [37] gtable_0.3.0       lifecycle_1.0.0    DBI_1.1.1          magrittr_2.0.1
## [41] scales_1.1.1       cli_2.5.0          stringi_1.6.2      farver_2.1.0
## [45] fs_1.5.0           xml2_1.3.2         ellipsis_0.3.2     generics_0.1.0
## [49] vctrs_0.3.8        tools_4.0.4        glue_1.4.2         hms_1.1.0
## [53] parallel_4.0.4     yaml_2.2.1         colorspace_2.0-1   rvest_1.0.0
## [57] haven_2.4.1
```

```
setwd("~/RStudio/PSYC6135/MiniPresentation")
dir.create("SessionInfo", showWarnings = FALSE)
setwd(paste(getwd(), "SessionInfo", sep = "/"))
writeLines(paste(capture.output(sessionInfo()),
           paste("\n", Sys.time()), sep = ""),
           paste(Sys.Date(), "sessionInfo.txt", sep = ""))
write_bib(file = "UsedPackages.bib")
```