Data_preparation

Maohua Nie

2023-11-22

```
# File names
file_names <- paste0("csv_data/jatos_results_", 1:19, ".csv")

# Read files and merge
raw_data <- do.call(rbind, lapply(file_names, read.csv, stringsAsFactors = FALSE))
```

Read data

```
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ─
─
## ✔ dplyr     1.1.0     ✔ readr     2.1.4
## ✔ forcats   1.0.0     ✔ stringr   1.5.0
## ✔ ggplot2   3.4.2     ✔ tibble    3.1.8
## ✔ lubridate 1.9.2     ✔ tidyr     1.3.0
## ✔ purrr     1.0.1
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ─
─
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the □]8;;http://conflicted.r-lib.org/□conflicted package□]8;;□ to force a
ll conflicts to become errors
```

```
library(jsonlite)
```

```
##
## Attaching package: 'jsonlite'
##
## The following object is masked from 'package:purrr':
##
##     flatten
```

```
library(dplyr)
library(lubridate)
library(data.table)
```

```
##
## Attaching package: 'data.table'
##
## The following objects are masked from 'package:lubridate':
##
##     hour, isoweek, mday, minute, month, quarter, second, wday, week,
##     yday, year
##
## The following objects are masked from 'package:dplyr':
##
##     between, first, last
##
## The following object is masked from 'package:purrr':
##
##     transpose
```

```
library(magrittr)
```

```
##
## Attaching package: 'magrittr'
##
## The following object is masked from 'package:purrr':
##
##     set_names
##
## The following object is masked from 'package:tidyr':
##
##     extract
```

```
library(jsonlite)

#Load the probabilities and outcomes datasets
control_int_data <- read.csv("stimuli_control_int.csv", header = TRUE)
lr_int_data <- read.csv("stimuli_lr_int.csv", header = TRUE)
noskew_int_data <- read.csv("stimuli_noskew_int.csv", header = TRUE)
rl_int_data <- read.csv("stimuli_rl_int.csv", header = TRUE)
```

Add date and time

```
#Show full numbers
options(scipen = 999)



#Convert time
#raw_data$start_time = raw_data$start_time / 1000

#Change tge date / time format
raw_data <- raw_data %>%
  mutate(start_time = as.POSIXct(raw_data$start_time, origin="1970-01-01", tz="GMT")) %>%
  rename("date" = "start_time")

#Convert time_elapsed into XMinutes XSecondes format
#raw_data <- raw_data %>% mutate(time_elapsed = seconds_to_period(time_elapsed/1000))
```

```
library(dplyr)



# Calculate question count for each subject
question_counts <- raw_data %>%
  group_by(subject) %>%
  summarize(question_count = sum(trial_type_label == "question"))
question_counts
```

```
## # A tibble: 19 × 2
##    subject  question_count
##    <chr>             <int>
##  1 1n6a9tzd              3
##  2 1ul65b0b              3
##  3 4ld6kjtr              9
##  4 4qg4v4l8              6
##  5 4xf5xlk2              4
##  6 5kq4u8za              5
##  7 5wqbolns              3
##  8 66ud1zz5              3
##  9 6em6s1gf              4
## 10 8llaj19l              6
## 11 auuwrelr              4
## 12 bbk5j4xw              3
## 13 g0e422t7              3
## 14 j2k7n49z              6
## 15 lrtvvg5x              3
## 16 m73bj2hn              7
## 17 nq128d3l              5
## 18 stj6ooxx              3
## 19 vp5s5sbf              3
```

```
# Join the question counts back with the original data
raw_data_with_counts <- raw_data %>%
  left_join(question_counts, by = "subject")

# Filter for subjects with question_count less than 6
raw_data1 <- raw_data_with_counts %>%
  filter(question_count <= 6)
```

Merge the csv data frames

```
#Prepare the files for merging:
## add a variable called "test_part" to fit the final dataframe
## modify the number of trial to fit the format of the final dataframe
## keep only the variables, which are needed
lr_int_data <- lr_int_data %>%
  mutate(test_part = "lr",
         trial_numb = row_number()) %>%
  select(trial_numb, test_part, P_A1, O_A1, P_A2, O_A2, P_B1, O_B1, P_B2, O_B2, x
1, y1, h1, i1, j1, k1, x2, y2, h2, i2, j2, k2, eva, evb, evd, sda, sdb, sdd)

noskew_int_data <- noskew_int_data%>%
  mutate(test_part = "ns",
         trial_numb = row_number()) %>%
  select(trial_numb, test_part, P_A1, O_A1, P_A2, O_A2, P_B1, O_B1, P_B2, O_B2, x
1, y1, h1, i1, j1, k1, x2, y2, h2, i2, j2, k2,eva, evb, evd, sda, sdb, sdd)

rl_int_data <- rl_int_data %>%
  mutate(test_part = "rl",
         trial_numb = row_number()) %>%
  select(trial_numb, test_part, P_A1, O_A1, P_A2, O_A2, P_B1, O_B1, P_B2, O_B2, x
1, y1, h1, i1, j1, k1, x2, y2, h2, i2, j2, k2, eva, evb, evd, sda, sdb, sdd)

control_int_data <- control_int_data %>%
  mutate(test_part = "control",
         trial_numb = row_number()) %>%
  select(trial_numb, test_part, P_A1, O_A1, P_A2, O_A2, P_B1, O_B1, P_B2, O_B2, x
1, y1, h1, i1, j1, k1, x2, y2, h2, i2, j2, k2, eva, evb, evd, sda, sdb, sdd)

#Combine the dataframes
pb_data_combined = bind_rows(lr_int_data, noskew_int_data, rl_int_data, control_in
t_data)

names(pb_data_combined)[names(pb_data_combined) == "test_part"] <- "skew"
```

Merge datasets from participants (raw_data) and probabilities (pb_data_combined)

```
#Prepare raw_data for merging
raw_data1 = raw_data1 %>%
  mutate(trial_numb = gsub(".*?(\\d+).*", "\\1", optionA_Stimulus))



#Match the type of object, in order to merge them
pb_data_combined$trial_numb = as.character(pb_data_combined$trial_numb)

#Merge filtered by trial number and type of test
raw_data_merged <- raw_data1 %>%
  full_join(pb_data_combined, by=c("skew","trial_numb"))
```

Changing the order of the variables

```
#Change the order of columns the dataframe
# I added a few variables, needed for some tests later
cleaned_data <- raw_data_merged %>%
  select(date, subject, time_elapsed, trial_type_label, test_part, skew, risk_index, response, optionA_Stimulus, optionB_Stimulus, rt,  P_A1, O_A1, P_A2, O_A2, P_B1, O_B1, P_B2, O_B2, eva, evb, evd, sda, sdb, sdd, BNT1_answer, BNT2_answer, BNT3_answer, BNT4_answer, accuracy_BNT,accuracy_HMT, total_bonus, Bonus_pay)
```

```
cleaned_data$rt = as.numeric(cleaned_data$rt)
#quantile(cleaned_data$rt, 0.05, na.rm = TRUE)
#quantile(cleaned_data$rt, 0.95, na.rm = TRUE)
library(dplyr)

cleaned_data1 <- cleaned_data %>%
  filter(trial_type_label == 'test',
         rt >= quantile(rt, 0.05, na.rm = TRUE),
         rt <= quantile(rt, 0.95, na.rm = TRUE),
         rt >= 1000)
```

Control Trials Check

```r
library(dplyr)

cleaned_data_test <- cleaned_data %>%
  select(subject, test_part, risk_index, response, evd, skew) %>%
  filter(skew == "control") %>%
  mutate(true_response = response)

cleaned_data_test <- cleaned_data_test %>%
  mutate(true_response = case_when(
    risk_index == 1  ~ response,
    risk_index == -1 ~ if_else(response == "f", "j", "f"),
    TRUE             ~ true_response  # This line keeps the original value in other cases
  ))


cleaned_data_test <- cleaned_data_test %>%
  mutate(accuracy = case_when(
    evd > 0 & true_response == 'f' ~ 1,
    evd < 0 & true_response == 'j' ~ 1,
    TRUE ~ 0
  ))

cleaned_data_test <- cleaned_data_test %>% filter(!is.na(risk_index))

cleaned_data_test <- cleaned_data_test %>%
  filter(test_part %in% c('ss', 'cc') )


mean_accuracy_per_participant_conditions <- cleaned_data_test %>%
  group_by(subject, test_part) %>%
  summarize(mean_accuracy = mean(accuracy)) %>%
  ungroup()
```

```
## `summarise()` has grouped output by 'subject'. You can override using the
## `.groups` argument.
```

```r
final_accuracy_data <- mean_accuracy_per_participant_conditions %>%
  pivot_wider(names_from = test_part, values_from = mean_accuracy,
              names_prefix = "accuracy_")


final_accuracy_data <- final_accuracy_data %>%
  mutate(accuracy_diff = accuracy_ss - accuracy_cc)

final_accuracy_data
```

```
## # A tibble: 17 × 4
##    subject  accuracy_cc accuracy_ss accuracy_diff
##    <chr>          <dbl>       <dbl>         <dbl>
##  1 1n6a9tzd       1           1             0
##  2 1ul65b0b       0.875       0.75         -0.125
##  3 4qg4v4l8       0.875       1             0.125
##  4 4xf5xlk2       1           1             0
##  5 5kq4u8za       1           1             0
##  6 5wqbolns       1           1             0
##  7 66ud1zz5       1           1             0
##  8 6em6s1gf       0.875       1             0.125
##  9 8llaj19l       0.875       1             0.125
## 10 auuwrelr       1           1             0
## 11 bbk5j4xw       0.75        1             0.25
## 12 g0e422t7       1           0.875        -0.125
## 13 j2k7n49z       0.875       0.625        -0.25
## 14 lrtvvg5x       1           1             0
## 15 nq128d3l       0.75        0.875         0.125
## 16 stj6ooxx       0.75        1             0.25
## 17 vp5s5sbf       1           0.875        -0.125
```

Set up final dataframe

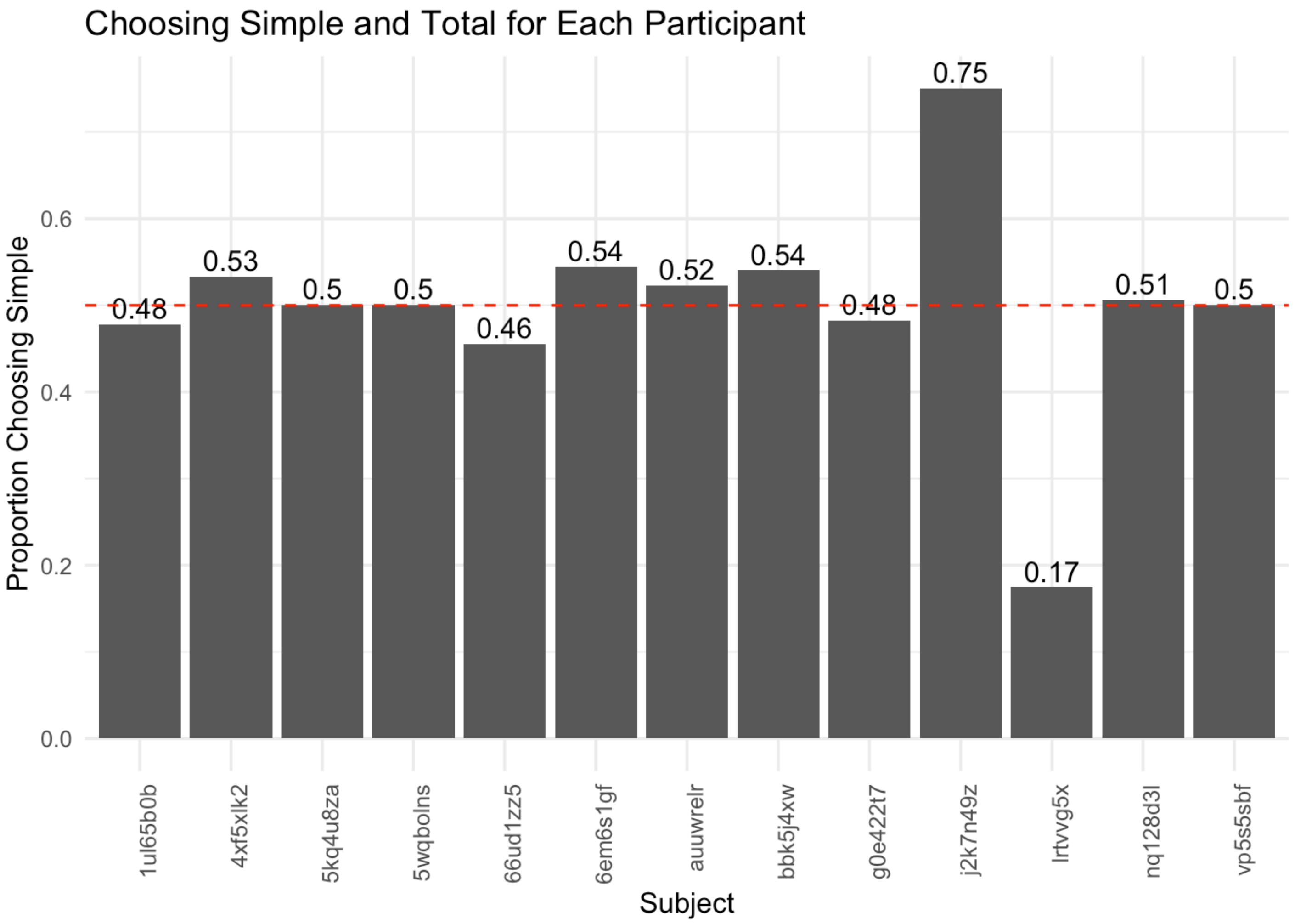
```
# new table, only have the id, test part, P_A1, O_A1, P_A2, O_A2, P_B1, O_B1, P_B2, O_B2, eva, evb, evd, sda, sdb, sdd, cho, rt

#This dataframe has been cleaned and removed the ones, who did not pass the previous tests
final_data <- cleaned_data1 %>%
  select(subject, test_part,skew, P_A1, O_A1, P_A2, O_A2, P_B1, O_B1, P_B2, O_B2, eva, evb, evd, sda, sdb, sdd, risk_index, response, rt)

final_data_all <- cleaned_data %>%
  select(subject, test_part,skew, P_A1, O_A1, P_A2, O_A2, P_B1, O_B1, P_B2, O_B2, eva, evb, evd, sda, sdb, sdd, risk_index, response, rt)


final_data <- final_data %>%
  filter(test_part %in% c('cs', 'sc','ss', 'cc'))

final_data_all <- final_data_all %>%
  filter(test_part %in% c('cs', 'sc','ss', 'cc'))

final_data1 = final_data %>%
  mutate(cho = 0,
         cho = ifelse(response == "f", 1*risk_index, cho),
         cho = ifelse(response == "j", -1*risk_index, cho))

final_data_all1 = final_data_all %>%
  mutate(cho = 0,
         cho = ifelse(response == "f", 1*risk_index, cho),
         cho = ifelse(response == "j", -1*risk_index, cho))
```

```
write.csv(final_data1, "final_data.csv", row.names=FALSE)
```

```r
df_filtered <- final_data1[final_data1$test_part %in% c('cs', 'sc'), ]

df_filtered$cho <- ifelse(df_filtered$test_part == "sc", -df_filtered$cho, df_filtered$cho)

df_filtered <- df_filtered %>%
  filter(skew %in% c('rl', 'lr','ns'))

result_table <- df_filtered %>%
  group_by(subject) %>%
  summarise(
    Simple = sum(cho == -1),
    Complex = sum(cho == 1),
    Total = Simple + Complex
  ) %>%
  filter(Total >= 85) %>%
  mutate(
    choosing_simple = Simple / Total
  )


# View the result
print(result_table)
```

```
## # A tibble: 13 × 5
##    subject  Simple Complex Total choosing_simple
##    <chr>     <int>   <int> <int>           <dbl>
##  1 1ul65b0b     43      47    90           0.478
##  2 4xf5xlk2     48      42    90           0.533
##  3 5kq4u8za     44      44    88           0.5  
##  4 5wqbolns     45      45    90           0.5  
##  5 66ud1zz5     41      49    90           0.456
##  6 6em6s1gf     49      41    90           0.544
##  7 auuwrelr     46      42    88           0.523
##  8 bbk5j4xw     47      40    87           0.540
##  9 g0e422t7     41      44    85           0.482
## 10 j2k7n49z     66      22    88           0.75 
## 11 lrtvvg5x     15      71    86           0.174
## 12 nq128d3l     43      42    85           0.506
## 13 vp5s5sbf     45      45    90           0.5  
```

```
# Load ggplot2 package
library(ggplot2)

# Create the plot
ggplot(result_table, aes(x = subject, y = choosing_simple)) +
  geom_bar(stat = "identity") +  # Bar plot for choosing_simple
   geom_text(aes(label = round(choosing_simple,2)), vjust = -0.3)+
  geom_hline(yintercept = 0.5, color = "red", linetype = "dashed") +  # Red line a
t 50%
  theme_minimal() +
  labs(title = "Choosing Simple and Total for Each Participant",
       x = "Subject",
       y = "Proportion Choosing Simple") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))  # Rotate x-axis labels
```



```r
# Function to perform paired t-test for each subject
perform_ttest <- function(data) {
  cc_data <- filter(data, test_part == "cc")$rt
  ss_data <- filter(data, test_part == "ss")$rt

  t.test(cc_data, ss_data, paired = TRUE)
}


results <- final_data_all1 %>%
  group_by(subject) %>%
  do(ttest_result = perform_ttest(.))


results$Summary <- lapply(results$ttest_result, function(x) {
  if (is.na(x$p.value)) {
    return(x$message)
  } else {
    return(paste("t =", round(x$statistic, 2),
                 ", df =", x$parameter,
                 ", p-value =", round(x$p.value, 4)))
  }
})

print(results$Summary)
```

```
## [[1]]
## [1] "t = 0.77 , df = 52 , p-value = 0.4452"
##
## [[2]]
## [1] "t = 2.63 , df = 52 , p-value = 0.0113"
##
## [[3]]
## [1] "t = 0.61 , df = 52 , p-value = 0.5416"
##
## [[4]]
## [1] "t = 7.45 , df = 52 , p-value = 0"
##
## [[5]]
## [1] "t = -0.49 , df = 52 , p-value = 0.6277"
##
## [[6]]
## [1] "t = 3.46 , df = 52 , p-value = 0.0011"
##
## [[7]]
## [1] "t = 1.99 , df = 52 , p-value = 0.0519"
##
## [[8]]
## [1] "t = 3.34 , df = 52 , p-value = 0.0016"
```

```
##
## [[9]]
## [1] "t = 0.71 , df = 52 , p-value = 0.4781"
##
## [[10]]
## [1] "t = 2.85 , df = 52 , p-value = 0.0062"
##
## [[11]]
## [1] "t = 3.43 , df = 52 , p-value = 0.0012"
##
## [[12]]
## [1] "t = -1.05 , df = 52 , p-value = 0.2992"
##
## [[13]]
## [1] "t = 2.09 , df = 52 , p-value = 0.0418"
##
## [[14]]
## [1] "t = 3.36 , df = 52 , p-value = 0.0015"
##
## [[15]]
## [1] "t = 0.43 , df = 52 , p-value = 0.6701"
##
## [[16]]
## [1] "t = 2.72 , df = 52 , p-value = 0.0089"
##
## [[17]]
## [1] "t = 1.17 , df = 52 , p-value = 0.2492"
```

```
df <- final_data1[final_data1$test_part %in% c('ss'), ]
df <- df %>%
  filter(skew %in% c('rl', 'lr','ns'))

library(lme4)
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'Matrix'
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

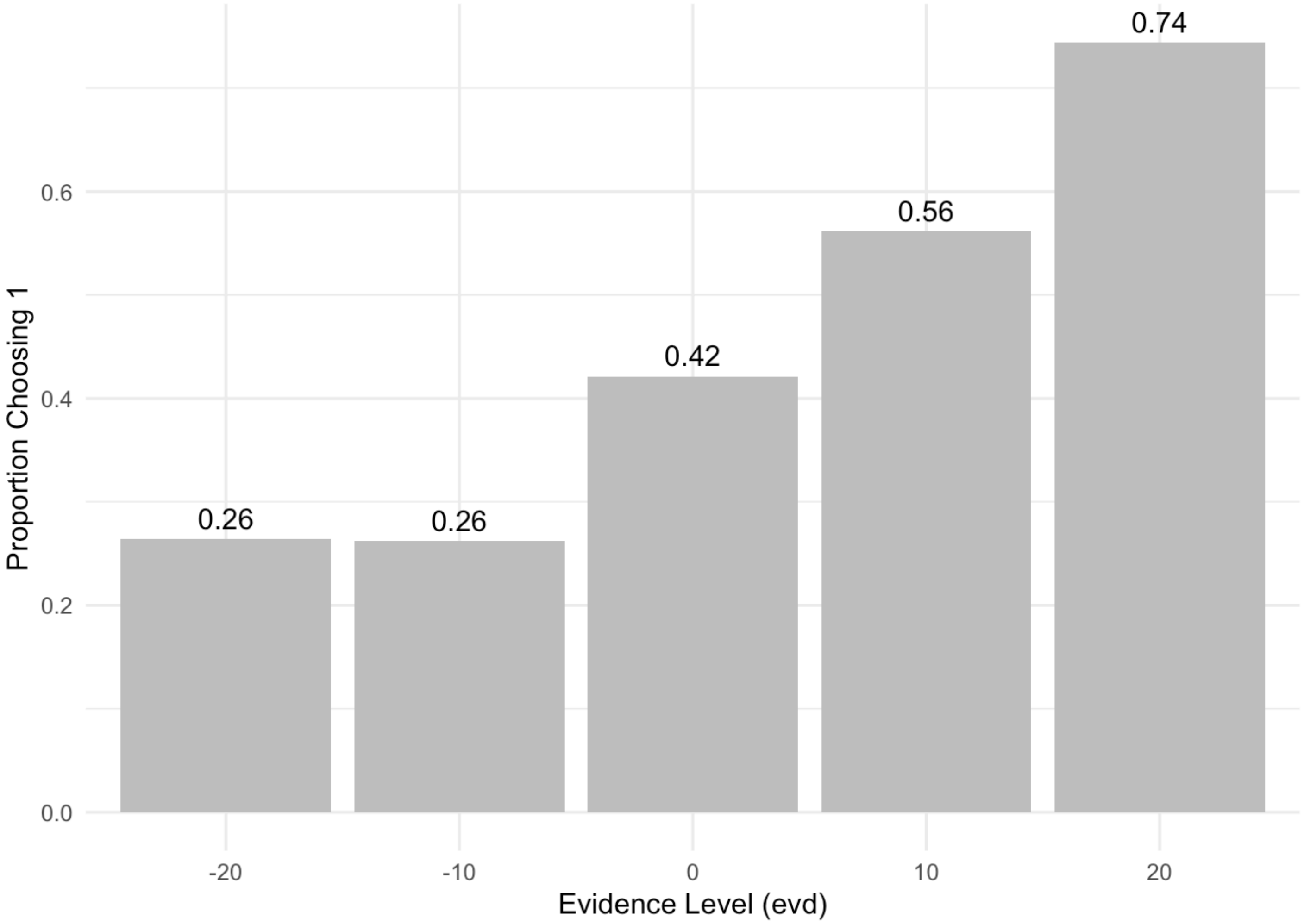
```
df$cho <- ifelse(df$cho == -1, 0, 1)
model <- glmer(cho ~ evd + (evd | subject), data = df, family = binomial)
```

```
## boundary (singular) fit: see help('isSingular')
```

```
summary(model)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: cho ~ evd + (evd | subject)
##    Data: df
## 
##      AIC      BIC   logLik deviance df.resid 
##    865.8    888.5   -427.9    855.8      690 
## 
## Scaled residuals: 
##     Min      1Q  Median      3Q     Max 
## -2.3572 -0.8258 -0.4127  0.8544  2.6780 
## 
## Random effects:
##  Groups  Name        Variance Std.Dev. Corr 
##  subject (Intercept) 0.026106 0.16157       
##          evd         0.001225 0.03499  -1.00
## Number of obs: 695, groups:  subject, 17
## 
## Fixed effects:
##             Estimate Std. Error z value     Pr(>|z|)    
## (Intercept) -0.26814    0.09357  -2.866      0.00416 ** 
## evd          0.05787    0.01078   5.366 0.0000000806 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Correlation of Fixed Effects:
##     (Intr)
## evd -0.396
## optimizer (Nelder_Mead) convergence code: 0 (OK)
## boundary (singular) fit: see help('isSingular')
```

```
df$evd_cat <- cut(df$evd,
                  breaks = c(-Inf, -15, -5, 5, 15, Inf),
                  labels = c("-20", "-10", "0", "10", "20"),
                  right = FALSE)

proportions <- df %>%
  group_by(evd_cat) %>%
  summarize(proportion = mean(cho == 1))

ggplot(proportions, aes(x = evd_cat, y = proportion)) +
  geom_bar(stat = "identity", fill = "grey") +
  geom_text(aes(label = round(proportion, 2)),
            vjust = -0.5, # Adjust text position
            color = "black") +
  labs(x = "Evidence Level (evd)", y = "Proportion Choosing 1") +
  theme_minimal()
```



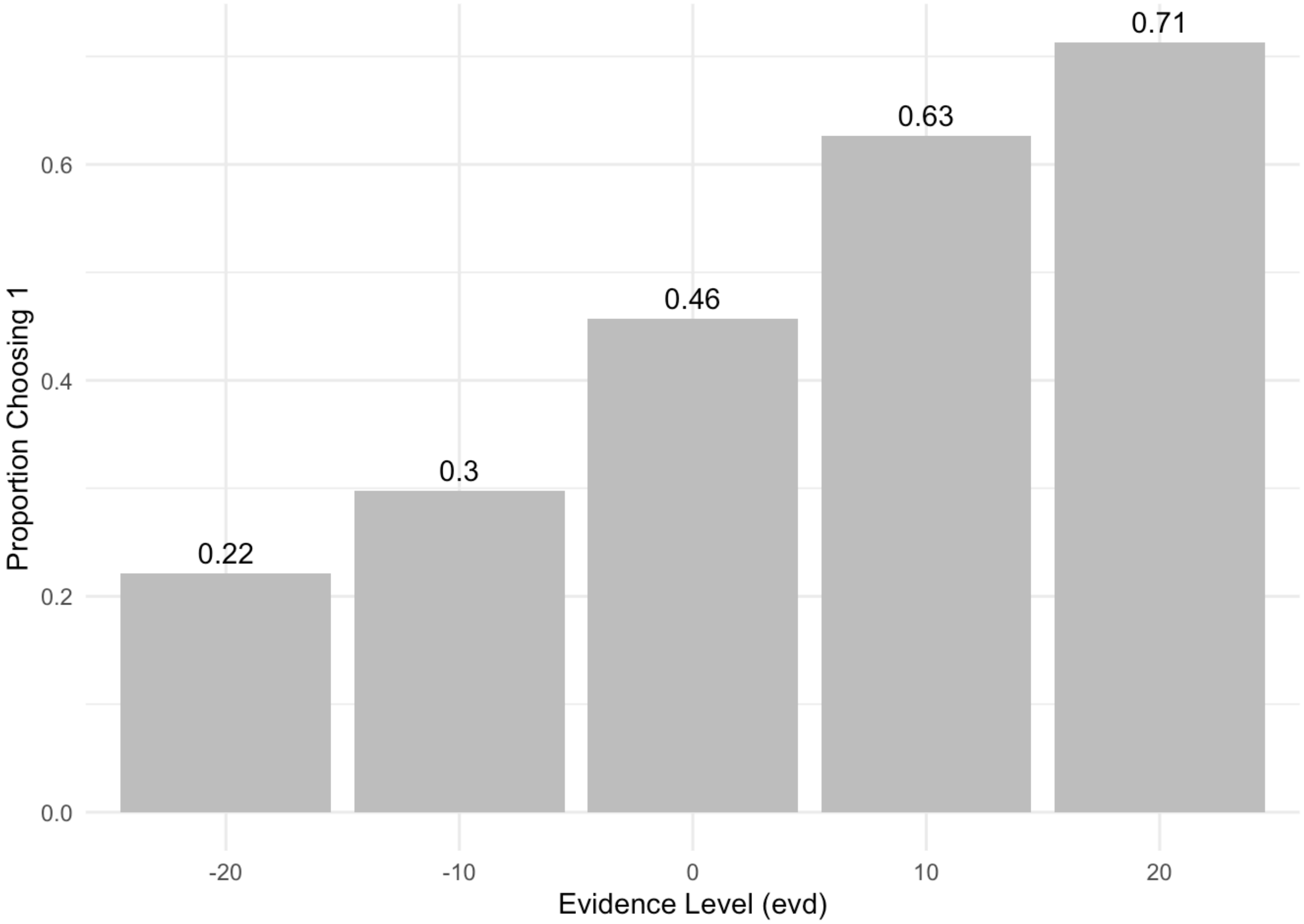
```
df <- final_data1[final_data1$test_part %in% c('cc'), ]
df <- df %>%
  filter(skew %in% c('rl', 'lr','ns'))

library(lme4)
df$cho <- ifelse(df$cho == -1, 0, 1)
model <- glmer(cho ~ evd + (evd | subject), data = df, family = binomial)
```

```
## boundary (singular) fit: see help('isSingular')
```

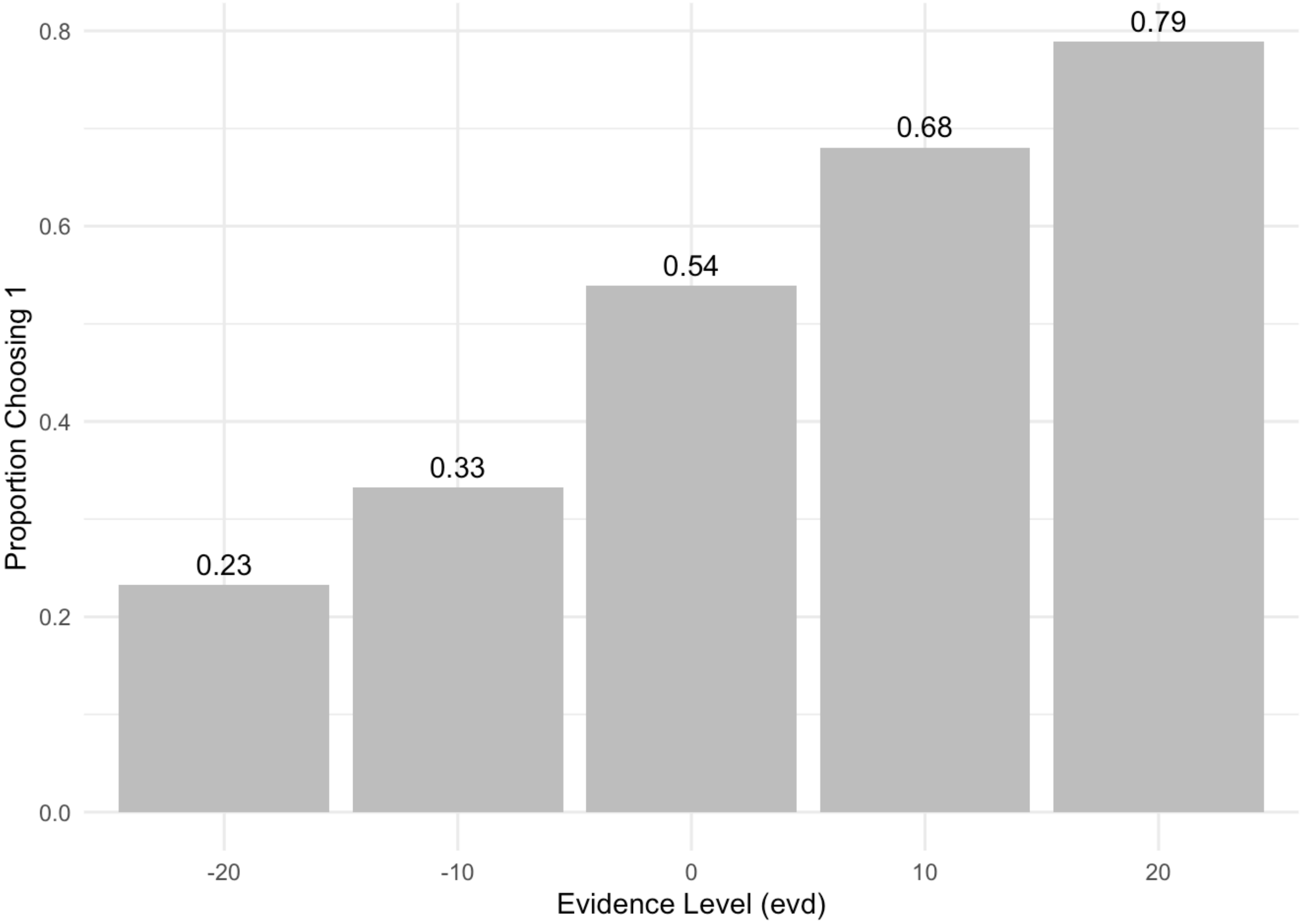
```
summary(model)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: cho ~ evd + (evd | subject)
##    Data: df
## 
##      AIC      BIC   logLik deviance df.resid 
##    849.1    871.8   -419.6    839.1      686 
## 
## Scaled residuals: 
##     Min      1Q  Median      3Q     Max 
## -2.8828 -0.7771 -0.3597  0.8425  2.3245 
## 
## Random effects:
##  Groups  Name        Variance Std.Dev. Corr
##  subject (Intercept) 0.00000  0.00000      
##          evd         0.00167  0.04087   NaN
## Number of obs: 691, groups:  subject, 17
## 
## Fixed effects:
##             Estimate Std. Error z value    Pr(>|z|)    
## (Intercept) -0.17629    0.08407  -2.097       0.036 *  
## evd          0.06426    0.01215   5.288 0.000000123 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Correlation of Fixed Effects:
##     (Intr)
## evd -0.020
## optimizer (Nelder_Mead) convergence code: 0 (OK)
## boundary (singular) fit: see help('isSingular')
```

```
df$evd_cat <- cut(df$evd,
                  breaks = c(-Inf, -15, -5, 5, 15, Inf),
                  labels = c("-20", "-10", "0", "10", "20"),
                  right = FALSE)

proportions <- df %>%
  group_by(evd_cat) %>%
  summarize(proportion = mean(cho == 1))

ggplot(proportions, aes(x = evd_cat, y = proportion)) +
  geom_bar(stat = "identity", fill = "grey") +
  geom_text(aes(label = round(proportion, 2)),
            vjust = -0.5, # Adjust text position
            color = "black") +
  labs(x = "Evidence Level (evd)", y = "Proportion Choosing 1") +
  theme_minimal()
```



```
# Select only the rows where test_part is 'cs' or 'sc'
df_filtered <- final_data1[final_data1$test_part %in% c('cs', 'sc'), ]


df_filtered$cho <- ifelse(df_filtered$test_part == "sc", -df_filtered$cho, df_filt
ered$cho)

df_filtered <-df_filtered %>%
  filter(skew %in% c('rl', 'lr','ns'))


df_filtered <- df_filtered %>%
  mutate(
    complex_index = ifelse(test_part == 'cs', 1, -1),
    evd = evd * complex_index,
    sdd = sdd * complex_index,
    chose_complex = ifelse((complex_index == 1 & cho == 1) | (complex_index == -1
& cho == -1), 1, -1)
  )




df_filtered$evd_cat <- cut(df_filtered$evd,
                    breaks = c(-Inf, -15, -5, 5, 15, Inf),
                    labels = c("-20", "-10", "0", "10", "20"),
                    right = FALSE)
proportions <- df_filtered %>%
  group_by(evd_cat) %>%
  summarize(proportion = mean(cho == 1))

ggplot(proportions, aes(x = evd_cat, y = proportion)) +
  geom_bar(stat = "identity", fill = "grey") +
  geom_text(aes(label = round(proportion, 2)),
            vjust = -0.5, # Adjust text position
            color = "black") +
  labs(x = "Evidence Level (evd)", y = "Proportion Choosing 1") +
  theme_minimal()
```


0.23
0.33
0.54
0.68
0.79
Proportion Choosing 1
0.0
0.2
0.4
0.6
0.8
-20
-10
0
10
20
Evidence Level (evd)