Supplemental file

Last updated 2021-08-19

Contents

1 Cleaning

The current section documents the data cleaning process.

1.1 Workspace

```
library(here) # for working with files
library(tidyverse) # for cleaning
library(janitor) # for variable names
library(stringi) # for generating random strings
library(lme4) # for multilevel modeling
library(lmerTest) # for p-values
library(sjPlot) # for figures
library(ggpubr) # for prettier plots
library(kableExtra) # for nicer tables
library(stringdist) # for scoring memory task
library(papaja) # for pretty numbers
library(psych) # for correlation tests
library(broom.mixed) # for tidying mulitilevel models
```

1.2 Change participant ID values

Before we begin, we create new versions of each data_t1 file that can be shared for purposes of reproducibility. These data_t1 files do not include variables that contain potentially identifying meta-data_t1 (e.g., IP address, latitude and longitude). Importantly, we also replace all Prolific ID values with new, random strings, to prevent the possibility that these participants are later identified. We also fix an error that can be introduced through Qualtrics, specifically that all or parts of the text string "Value will be set from panel or URL" is sometimes entered into the text box for ID. Prolific ID values are always 24 characters long and start with a number – we search for strings that meet this criteria.

(We note that the code chunks in this subsection are turned off in the RMarkdown file – `eval = F` – as readers will not be able to run these chunks.)

```
# function to load raw file, clean the names, and remove meta-data_t1
# creating a function ensures the same procedure is applied to all
# orginal datasets

load_data = function(path){

  full_path = here(path)
  data_labels = read_csv(full_path)
  data_obj = read_csv(full_path,
                  skip = 3,
                  col_names = names(data_labels))

  data_obj = clean_names(data_obj)

  data_obj = data_obj %>%
    select(-end_date,
           -ip_address,
           -progress,
```

```
            -finished,
            -recorded_date,
            -status,
            -response_id,
            -external_reference,
            -distribution_channel,
            -user_language,
            -starts_with("recipient"),
            -starts_with("location"),
            -starts_with("meta_info"),
            -prolific_pid)

  #this fixes a column naming error in dataset 2B
  if(!("proid" %in% names(data_obj))) names(data_obj) = str_replace(names(data_obj), "q763", "proid")

  data_obj = data_obj %>%
    mutate(proid = str_extract(proid, "\\d([[:alnum:]]{23})"))

  return(data_obj)
}

data_t1 <- load_data("data/Wording_July 13, 2021_20.00.text.csv")
data_2A <- load_data("data/Wording 2A_August 13, 2021_14.49.text.csv")
data_2B <- load_data("data/Wording 2B_August 4, 2021_18.49.text.csv")
data_2C <- load_data("data/Wording 2C_August 3, 2021_18.02.csv")
data_2D <- load_data("data/Wording 2D_July 29, 2021_14.55.text.csv")
```

Next, we identify all unique participant IDs. For each, we generate a new string, Then we replace the original ID values with the new strings.

```
original_id <- unique(c(data_t1$proid,
                        data_2A$proid,
                        data_2B$proid,
                        data_2C$proid,
                        data_2D$proid))

#remove missing values -- represent bots or tests
original_id = original_id[!is.na(original_id)]

#generate new ids (randoms tring of letters and numbers)
set.seed(202108)
new_id <- stri_rand_strings(n = length(original_id), length = 24)

#replace old string with new string
for(i in 1:length(original_id)){
  data_t1$proid[data_t1$proid == original_id[i]] <- new_id[i]
  data_2A$proid[data_2A$proid == original_id[i]] <- new_id[i]
  data_2B$proid[data_2B$proid == original_id[i]] <- new_id[i]
  data_2C$proid[data_2C$proid == original_id[i]] <- new_id[i]
  data_2D$proid[data_2D$proid == original_id[i]] <- new_id[i]
}
```

We end by saving each data_t1 frame as new .csv files, to be uploaded to OSF and shared for reproduction.

```
write_csv(data_t1, file = here("deidentified data/data_time1.csv"))
write_csv(data_2A, file = here("deidentified data/data_time2_A.csv"))
write_csv(data_2B, file = here("deidentified data/data_time2_B.csv"))
write_csv(data_2C, file = here("deidentified data/data_time2_C.csv"))
write_csv(data_2D, file = here("deidentified data/data_time2_D.csv"))
```

1.3 Time 1

We load the deidentified Time 1 data here.

```
data_t1 <- read_csv(here("deidentified data/data_time1.csv"))
```

We rename several columns, in order to facilitate the use of regular expressions later. Specifically, we remove the underscores (_) in the columns pertaining to broad-mindedness and self-disciplined.

```
names(data_t1) = str_replace(names(data_t1), "broad_mind", "broadmind")
names(data_t1) = str_replace(names(data_t1), "self_disciplind", "selfdisciplined")
```

We can also remove the meta-data (timing, etc) around two attention check adjectives, "human" and "asleep".

```
data_t1 = data_t1 %>%
  select(-starts_with("t_human"),
         -starts_with("t_asleep"))
```

1.3.1 Recode personality item responses to numeric

We recode the responses to personality items, which we downloaded as text strings. We chose to use text strings as opposed to numbers to avoid any possibility that the Qualtrics-set coding was incorrect. We start this process by identifying the personality items (`p_items`) using regular expressions. All personality items take a format like `outgoing_a` or `helpful_b_2`; that is, they start with the adjective, followed by a letter indicating with which condition or item format the adjective was presented, and sometimes they are followed by a 2, indicating it was the second time the participant saw the adjective. We can represent this pattern using regular expressions.

```
p_items = str_extract(names(data_t1), "^[[:alpha:]]*_[abcd](_2)?$")
p_items = p_items[!is.na(p_items)]

personality_items = select(data_t1, proid, all_of(p_items))
```

Next, we write a simple function to recode values. We find the `case_when` function to be the most clear method of communicating the recoding process when moving from string to numeric.

```
recode_p = function(x){
  y = case_when(
    x == "Very inaccurate" ~ 1,
    x == "Moderately inaccurate" ~ 2,
    x == "Slightly inaccurate" ~ 3,
    x == "Slightly accurate" ~ 4,
    x == "Moderately accurate" ~ 5,
    x == "Very accurate" ~ 6,
```

```
    TRUE ~ NA_real_)
  return(y)
}
```

Finally, we apply this function to all personality items.

```
personality_items = personality_items %>%
  # apply to all varibles except proid
  mutate(across(!c(proid), recode_p))
```

Now we merge the recoded values back into the data_t1.

```
# remove personality items from data file
data_t1 = select(data_t1, -all_of(p_items))
# merge in recoded personality items
data_t1 = full_join(data_t1, personality_items)
```

1.3.2 Drop bots and inattentive participants

1.3.2.1 Based on ID Recall that when preparing the data files for sharing, we replaced all Prolific IDs with random strings. A consequence of this cleaning is that any ID entered that did not have a string meeting the Prolific ID format requirements (24 character, starting with a number) was replaced with `NA`. To remove these bots, we can simply filter out missing ID values.

We removed 9 participants without valid Prolific IDs.

```
data_t1 = data_t1 %>%
  filter(english %in% c("Well", "Very well (fluent/native)"))
```

1.3.2.2 Based on language We removed 0 participants that do not speak english well or very well.

1.3.2.3 Based on patterns We remove any participant who provides the same response to over half of the items (17 or more items) from a given block in a row.

To proceed, first we create a dataframe containing just the responses to personality items in the first block.

```
# first, identify unique adjectives, in order
adjectives = p_items %>%
  str_remove_all("_.") %>%
  unique()

# extract block 1 questions using regular expressions
# these follow the personality item format described above, but never end with 2
block1 = data_t1 %>%
  select(proid, matches("^[[:alpha:]]+_[abcd]$"))
```

Next, we rename the variables. Instead of variable names identifying the specific adjective (e.g., outgoing_a), we need variable names which indicate the order in which the adjective was seen by the participant (e.g., trait01_a). This will help us determine patterns by item order, rather than adjective content. Participants all saw adjectives in the same order (i.e., all participants, regardless of condition, saw outgoing first).

```
#rename variables
n = 0
for(i in adjectives){ # for each adjective
  n = n+1 # identify its location in the presentation
  names(block1) = str_replace(names(block1), #in variable names
                              # replace the adjective string
                              i,
                              # with the word trait followed by its place
                              paste0("trait", str_pad(n, 2, pad = "0")))
}
```

We use `gather` and `spread` to quickly combine columns measuring the same trait. That is, instead of having columns trait01_a, trait01_b, trait01_c, and trait01_d, we now have a single column called trait01.

```
block1 = block1 %>%
  gather(item, response, -proid) %>%
  filter(!is.na(response)) %>%
  separate(item, into = c("item", "format")) %>%
  select(-format) %>%
  spread(item, response)
```

To count the number of runs, we loop through participants and, within participant, loop through columns. Within participant, we create an object called `run`. If a response to a personality item is the same as the participant's response to the previous item, we increase the value of `run` by 1. If this new value is the largest `run` value for that participant, it becomes the value of an object called `maxrun`. If the participant gives a new response, `run` is reset to 0. We record the `maxrun` value for each partipant in a variable called `block1_runs`.

```
block1_runs = numeric(length = nrow(block1))


for(i in 1:nrow(block1)){
  run = 0
  maxrun = 0
  for(j in 3:ncol(block1)){
    if(block1[i,j] == block1[i, j-1]){
      run = run+1
      if(run > maxrun) maxrun = run
      } else{ run = 0}
  }
  block1_runs[i] = maxrun
}

#add to data_t1 frame
block1$block1_runs = block1_runs
```

Here we repeat the process described above with Block 2 data.

```
# extract block 2 questions
block2 = data_t1 %>%
  select(proid, matches("^[[:alpha:]]+_[abcd]_2$"))

#rename variables
```

```
n = 0
for(i in adjectives){
  n = n+1
  names(block2) = str_replace(names(block2), i, paste0("trait", str_pad(n, 2, pad = "0")))
}

block2 = block2 %>%
  gather(item, response, -proid) %>%
  filter(!is.na(response)) %>%
  mutate(item = str_remove(item, "_2")) %>%
  separate(item, into = c("item", "format")) %>%
  select(-format) %>%
  spread(item, response)

block2_runs = numeric(length = nrow(block2))

#identify max run for each participant
for(i in 1:nrow(block2)){
  run = 0
  maxrun = 0
  for(j in 3:ncol(block2)){
    if(block2[i,j] == block2[i, j-1]){
      run = run+1
      if(run > maxrun) maxrun = run
      } else{ run = 0}
  }
  block2_runs[i] = maxrun
}

#add to data_t1 frame
block2$block2_runs = block2_runs
```

We combine the variables holding the maximum runs into a single data frame. We will remove participants if their maximum run in either block was greater than or equal to 17.

```
#combine results
runs_data = block1 %>%
  select(proid, block1_runs) %>%
  full_join(select(block2, proid, block2_runs)) %>%
  mutate(
    remove = case_when(
      block1_runs >= 17 ~ "Remove",
      block2_runs >= 17 ~ "Remove",
      TRUE ~ "Keep"
    ))
```

```
#visualize
runs_data %>%
  ggplot(aes(block1_runs, block2_runs)) +
  geom_point(aes(color = remove)) +
  scale_color_manual(values = c("black", "red")) +
  guides(color = "none") +
  labs(
```

```
    x = "block 1 runs",
    y = "block 2 runs"
  ) +
  theme_pubr()
```

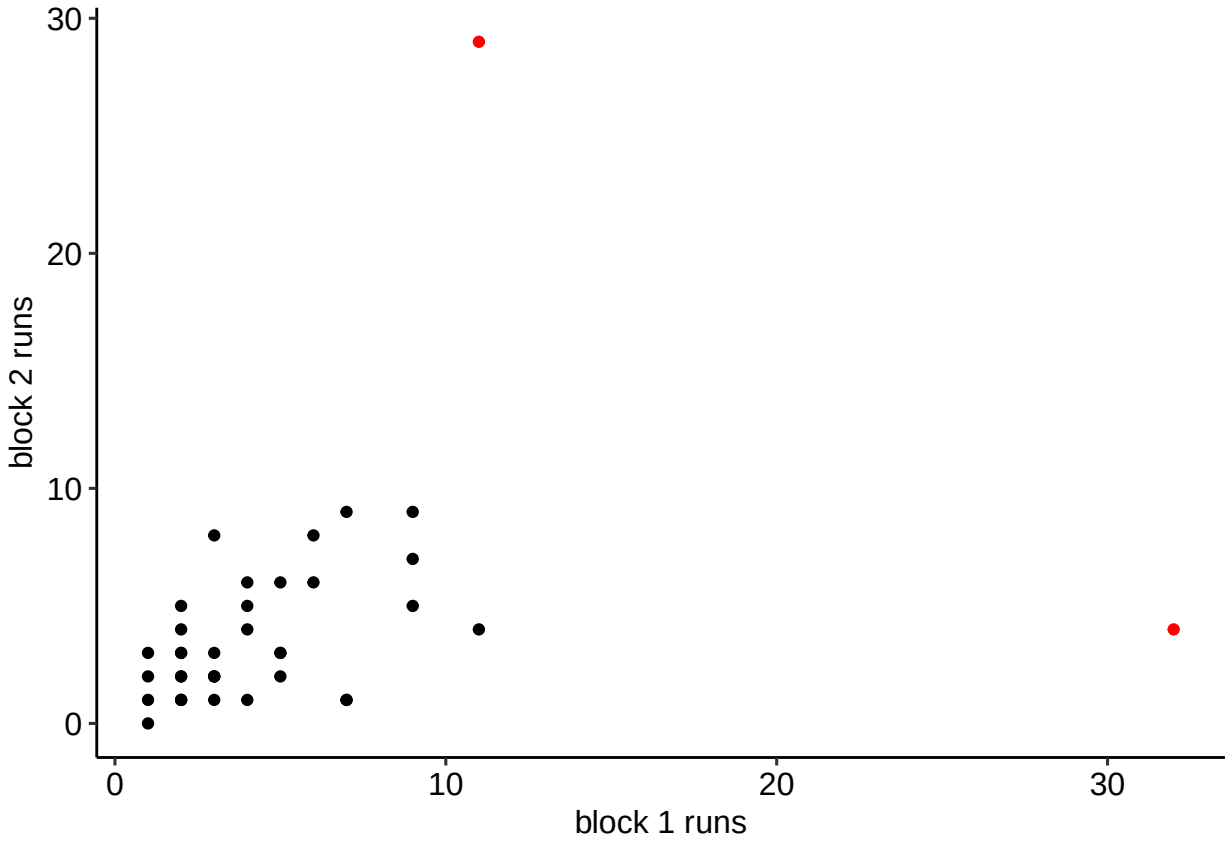


Figure 1: Maximum number of same consecutive responses in personality blocks.

There were 2 participants who provided the same answer 17 or more times in a row. These participants were removed from the analyses.

```
data_t1 = data_t1 %>%
  full_join(select(runs_data, proid, remove)) %>%
  filter(remove != "Remove") %>%
  select(-remove)

rm(runs_data)
```

1.3.2.4 Based on inattentive responding We expect to exclude any participant who has an average response of 4 ("slightly agree") or greater to the attention check items. Two items from the Inattentive and Deviant Responding Inventory for Adjectives (IDRIA) scale (Kay & Saucier, in prep) have been included here, in part to help evaluate the extent of inattentive responding but also to consider the effect of item wording on these items. The two items used here (i.e., "Asleep", "Human") were chosen to be as inconspicuous as possible, so as to not to inflate item response durations. The frequency item (i.e., "human") will be reverse-scored, so that higher scores on both the infrequency and frequency items reflect greater inattentive responding.

```r
in_average = data_t1 %>%
  # reverse score human
  mutate(across(matches("^human"),  ~(.x*-1)+7)) %>%
  # select id and attention check items
  select(proid, matches("^human"), matches("^asleep")) %>%
  gather(item, response, -proid) %>%
  filter(!is.na(response)) %>%
  group_by(proid) %>%
  summarise(avg = mean(response)) %>%
  mutate(
    remove = case_when(
      avg >= 4 ~ "Remove",
      TRUE ~ "Keep"))
```

```r
in_average %>%
  ggplot(aes(x = avg, fill = remove)) +
  geom_histogram(bins = 20, color = "white") +
  geom_vline(aes(xintercept = 4)) +
  guides(fill = "none") +
  labs(x = "Average response to inattention check items") +
  theme_pubr()
```

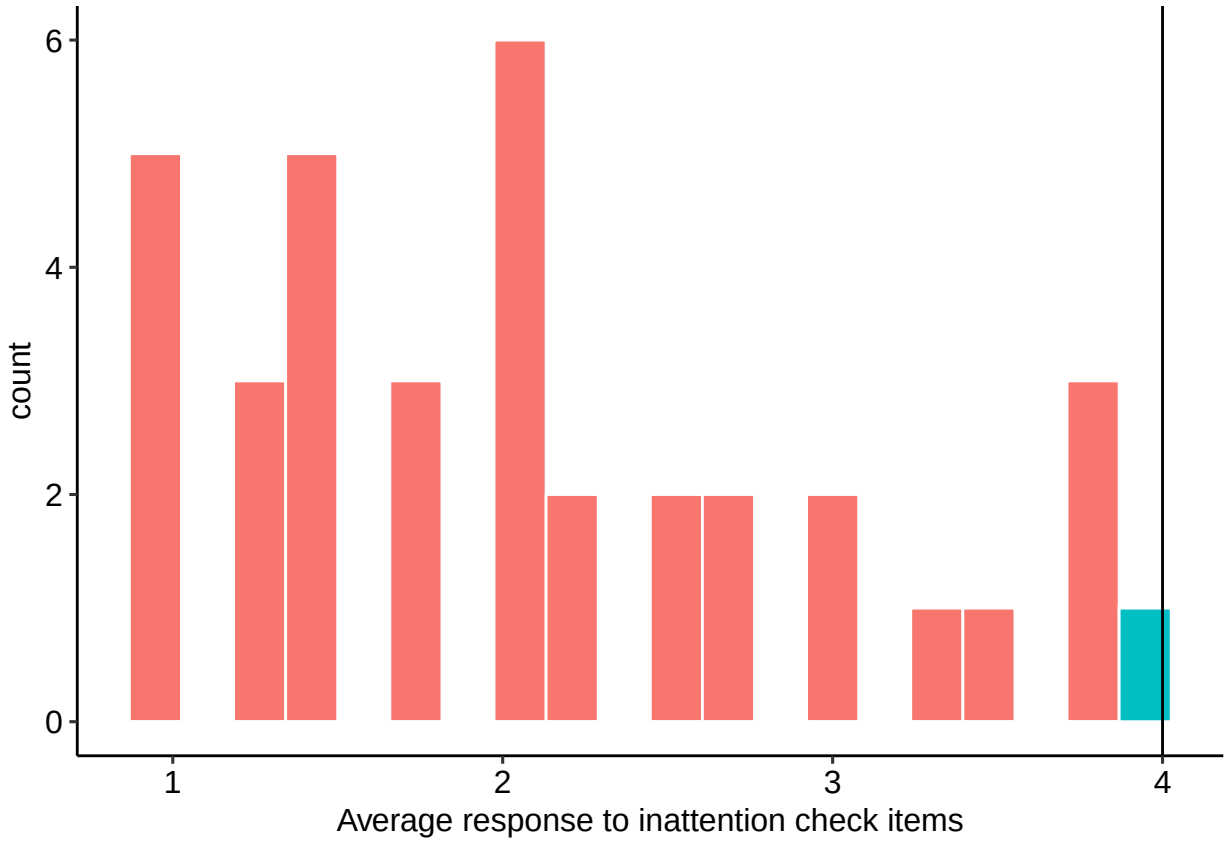


Figure 2: Average response to inattention check items

We remove 1 participants whose responses suggest inattention.

```
data_t1 = data_t1 %>%
  full_join(select(in_average, proid, remove)) %>%
  filter(remove != "Remove") %>%
  select(-remove)
```

1.3.2.5 Based on average time to respond to personality items

First, select just the timing of the personality items. We do this by searching for specific strings: "t_[someword]*[a or b or c or d]*(maybe 2_)_page_submit."

```
timing_data = data_t1 %>%
  select(proid, matches("t_[[:alpha:]]*_[abcd](_2)?_page_submit"))
```

Next we gather into long form and remove missing timing values

```
timing_data = timing_data %>%
  gather(variable, timing, -proid) %>%
  filter(!is.na(timing))
```

To check, each participant should have the same number of responses: 62.

```
timing_data %>%
  group_by(proid) %>%
  count() %>%
  ungroup() %>%
  summarise(min(n), max(n))
```

```
## # A tibble: 1 x 2
##   `min(n)` `max(n)`
##      <int>    <int>
## 1       62       62
```

Excellent! Now we calculate the average response time per item for each participant. We mark a participant for removal if their average time is less than 1 second or greater than 30. See Figure @ref(fig:timing_dist) for a distribution of average response time.

```
timing_data = timing_data %>%
  group_by(proid) %>%
  summarise(m_time = mean(timing)) %>%
  mutate(remove = case_when(
    m_time < 1 ~ "Remove",
    m_time > 30 ~ "Remove",
    TRUE ~ "Keep"
  ))
```

```
timing_data %>%
  ggplot(aes(x = m_time, fill = remove)) +
  geom_histogram(color = "white") +
  labs(x = "Average response time (seconds)", y = "Number of participants") +
  theme_pubr()
```

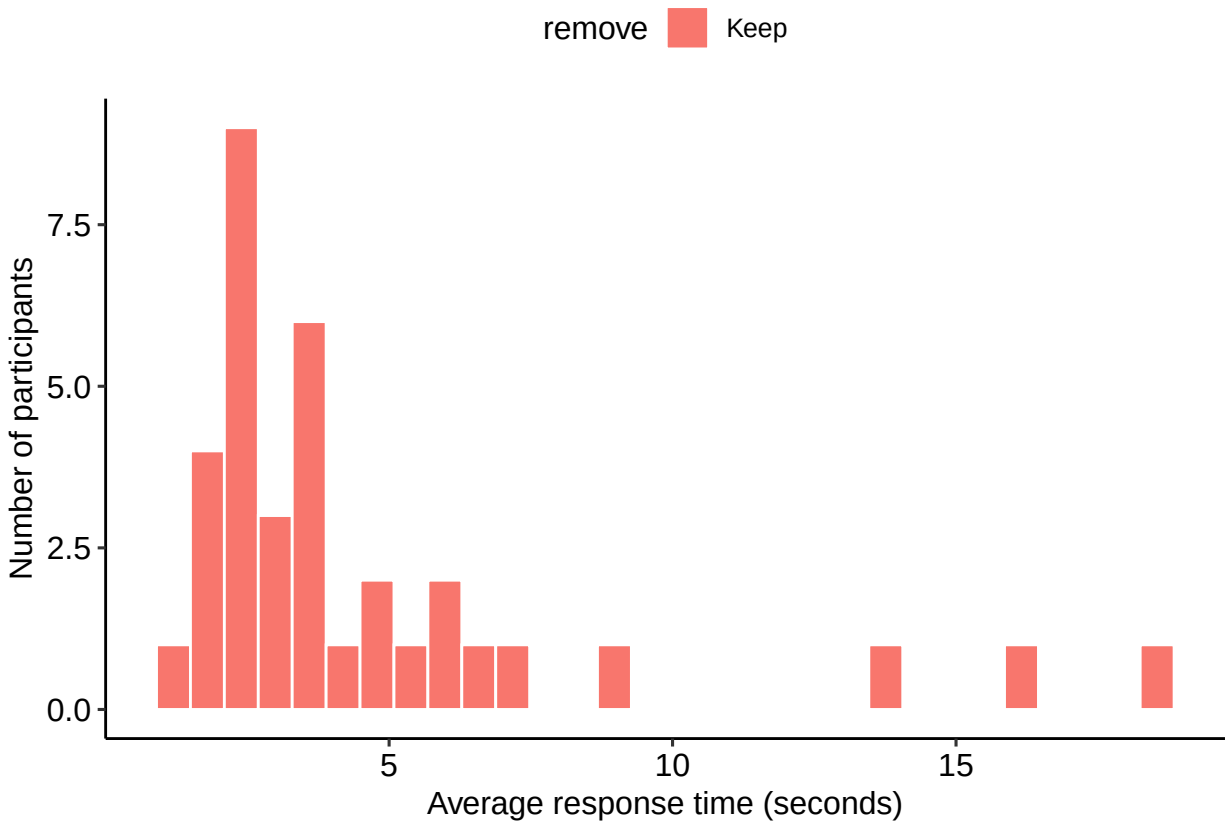


Figure 3: Distribution of average time to respond to personality items.

```
data_t1 = inner_join(data_t1, filter(timing_data, remove == "Keep")) %>%
  select(-remove)
```

Based on timing, we removed 0 participants.

We create a variable which indicates the Block 1 condition of each participant. This is used in two places: first, in recruitinig participants at Time 2 (participants are given the same format at Time 2 as they received in Block 1), and second, in selecting the corret items during the test-retest analyses.

```
data_t1 = data_t1 %>%
  mutate(condition = case_when(
    !is.na(outgoing_a) ~ "A",
    !is.na(outgoing_b) ~ "B",
    !is.na(outgoing_c) ~ "C",
    !is.na(outgoing_d) ~ "D",
  ))
```

At this point, we'll extract the Prolific ID numbers. These participants will be eligible to take the survey at Time 2.

```
data_t1 %>%
  select(proid, condition) %>%
  write_csv(file = here("data/elligible_proid"))
```

1.4 Time 2

```
data_2A <- read_csv(here("deidentified data/data_time2_A.csv"))
data_2B <- read_csv(here("deidentified data/data_time2_B.csv"))
data_2C <- read_csv(here("deidentified data/data_time2_C.csv"))
data_2D <- read_csv(here("deidentified data/data_time2_D.csv"))
```

```
data_2 = data_2A %>%
  full_join(data_2B) %>%
  full_join(data_2C) %>%
  full_join(data_2D)
```

Rename the following columns.

```
data_2 = data_2 %>%
  rename(start_date2 = start_date,
         duration_in_seconds2 = duration_in_seconds)
```

We rename several columns, in order to facilitate the use of regular expressions later. Specifically, we remove the underscores (_) in the columns pertaining to broad-mindedness and self-disciplined.

```
names(data_2) = str_replace(names(data_2), "broad_mind", "broadmind")
names(data_2) = str_replace(names(data_2), "self_disciplind", "selfdisciplined")
```

We can also remove the meta-data (timing, etc) around two attention check adjectives, "human" and "asleep".

```
data_t1 = data_t1 %>%
  select(-starts_with("t_human"),
         -starts_with("t_asleep"))
```

1.4.1 Recode personality item responses to numeric

We recode the responses to personality items, which we downloaded as text strings. Here, all items end with `_3` and sometimes with `i`.

```
p_items_2 = str_extract(names(data_2), "^[[:alpha:]]*_[abcd]_3(i)?$")
p_items_2 = p_items_2[!is.na(p_items_2)]

personality_items_2 = select(data_2, proid, all_of(p_items_2))
```

We apply the recoding function to all personality items.

```
personality_items_2 = personality_items_2 %>%
  mutate(
    across(!c(proid), recode_p))
```

Now we merge this back into the data_2.

```
data_2 = select(data_2, -all_of(p_items_2))
data_2 = full_join(data_2, personality_items_2)
```

1.4.2 Drop bots and inattentive participants

This code recreates the steps outlined in detail above for Time 1. Please refer to the descriptions above for justification and explaination of the code presented here.

1.4.2.1 Based on ID

We also check that the ID in time 2 matches an ID in time 1.

```
data_2 = data_2 %>%
  filter(proid %in% data_t1$proid)
```

We removed 35 participants without valid Prolific IDs.

1.4.2.2 Based on patterns

We remove any participant who provides the same response to over half of the items (17 or more items) from a given block in a row.

```
# first, identify unique adjectives, in order
adjectives = p_items_2 %>%
  str_remove_all("_.") %>%
  unique()

# extract block 3 questions
block3 = data_2 %>%
  select(proid, all_of(p_items_2))
```

```
#rename variables
n = 0
for(i in adjectives){
  n = n+1
  names(block3) = str_replace(names(block3), i, paste0("trait", str_pad(n, 2, pad = "0")))
}


block3 = block3 %>%
  gather(item, response, -proid) %>%
  filter(!is.na(response)) %>%
  mutate(item = str_remove(item, "_3(i)?$")) %>%
  separate(item, into = c("item", "format")) %>%
  #select(-format) %>%
  spread(item, response)

block3_runs = numeric(length = nrow(block3))

for(i in 1:nrow(block3)){
  run = 0
  maxrun = 0
  for(j in 3:ncol(block3)){
    if(block3[i,j] == block3[i, j-1]){
      run = run+1
      if(run > maxrun) maxrun = run
      } else{ run = 0}
  }
  block3_runs[i] = maxrun
}

#add to data_2 frame
block3$block3_runs = block3_runs
```

```
#combine results
runs_data_2 = block3 %>%
  select(proid, block3_runs) %>%
  mutate(
    remove = case_when(
      block3_runs >= 17 ~ "Remove",
      TRUE ~ "Keep"
    ))
```

```
#visualize
runs_data_2 %>%
  ggplot(aes(block3_runs)) +
  geom_histogram(aes(fill = remove), bins = 10,color = "white") +
  scale_color_manual() +
  guides(fill = "none") +
  labs(x = "Maximum number of repeated answers",
       y = "Participant count") +
  theme_pubr()
```

There were 0 participants who provided the same answer 17 or more times in a row. These participants were

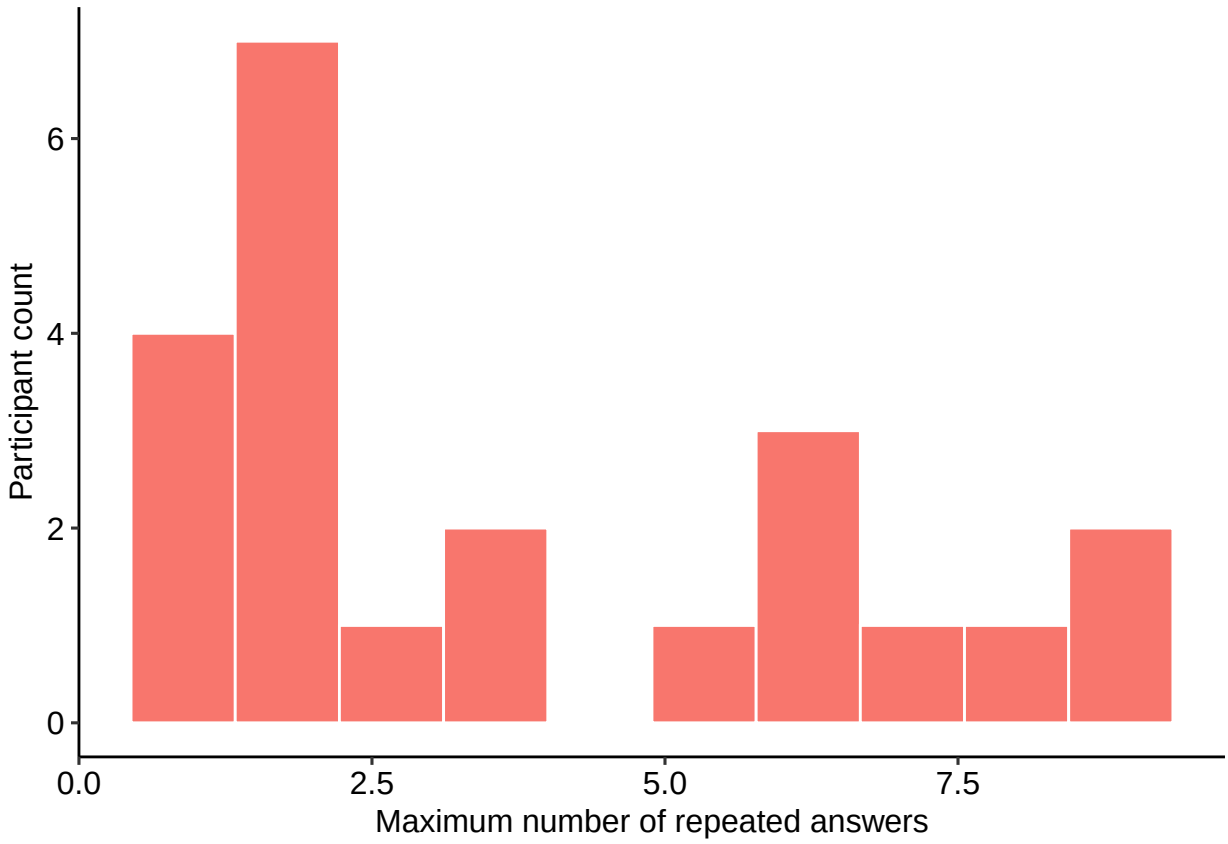


Figure 4: Maximum number of same consecutive responses in personality block 3.

removed from the analyses.

```
data_2 = data_2 %>%
  full_join(select(runs_data_2, proid, remove)) %>%
  filter(remove != "Remove") %>%
  select(-remove)

rm(runs_data_2)
```

```
in_average = data_2 %>%
  # reverse score human
  mutate(across(matches("^human"),  ~(.x*-1)+7)) %>%
  # select id and attention check items
  select(proid, matches("^human"), matches("^asleep")) %>%
  gather(item, response, -proid) %>%
  filter(!is.na(response)) %>%
  group_by(proid) %>%
  summarise(avg = mean(response)) %>%
  mutate(
    remove = case_when(
      avg >= 4 ~ "Remove",
      TRUE ~ "Keep"))
```

```
in_average %>%
  ggplot(aes(x = avg, fill = remove)) +
  geom_histogram(bins = 20, color = "white") +
  geom_vline(aes(xintercept = 4)) +
  guides(fill = "none") +
  labs(x = "Average response to inattention check items") +
  theme_pubr()
```

1.4.2.3 Based on inattentive responding

We remove 1 participants whose responses suggest inattention.

```
data_2 = data_2 %>%
  full_join(select(in_average, proid, remove)) %>%
  filter(remove != "Remove") %>%
  select(-remove)
```

```
timing_data_2 = data_2 %>%
  select(proid, matches("t_[[:alpha:]]*_[abcd]_3(i)?_page_submit"))

timing_data_2 = timing_data_2 %>%
  gather(variable, timing, -proid) %>%
  filter(!is.na(timing))
```

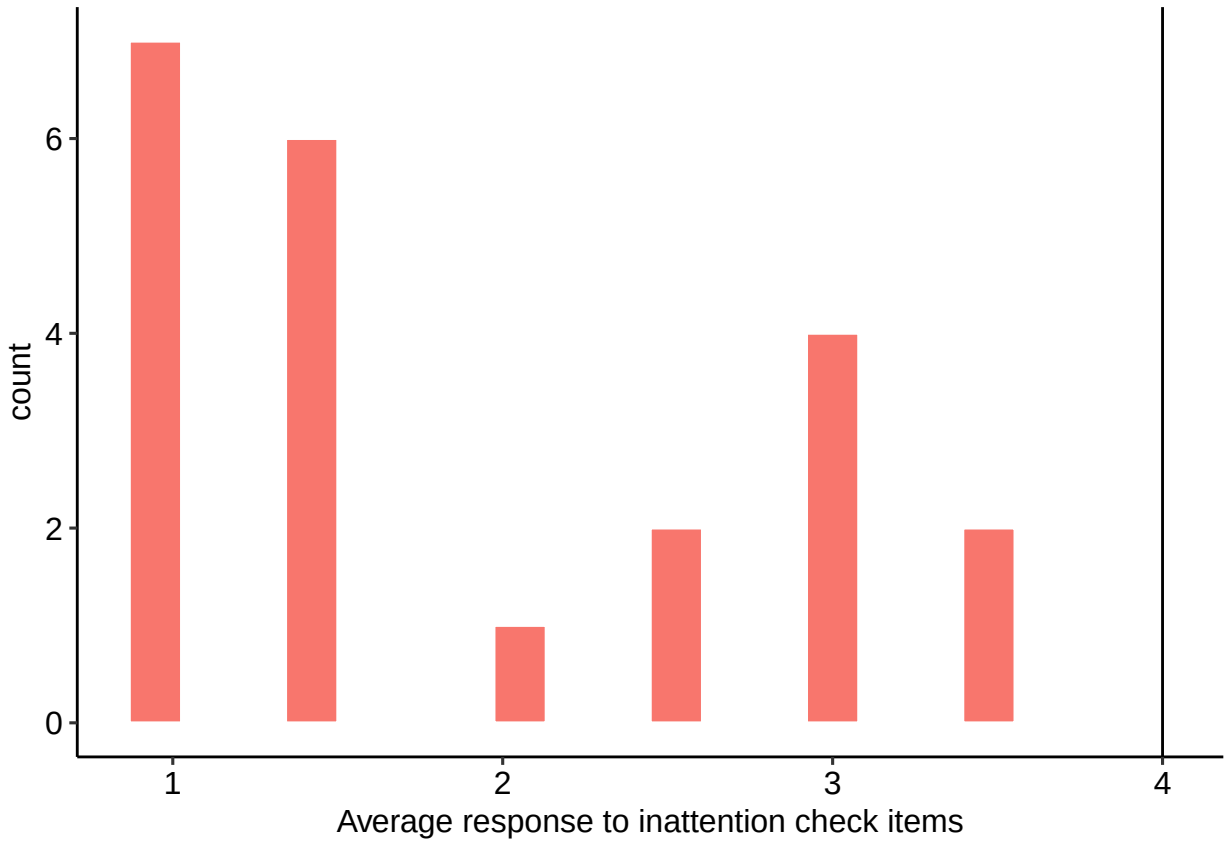


Figure 5: Average response to inattention check items

1.4.2.4 Based on average time to respond to personality items

To check, each participant should have the same number of responses: 33.

```
timing_data_2 %>%
  group_by(proid) %>%
  count() %>%
  ungroup() %>%
  summarise(min(n), max(n))
```

```
## # A tibble: 1 x 2
##   `min(n)` `max(n)`
##      <int>    <int>
## 1       33       33
```

```
timing_data_2 = timing_data_2 %>%
  group_by(proid) %>%
  summarise(m_time = mean(timing)) %>%
  mutate(remove = case_when(
    m_time < 1 ~ "Remove",
    m_time > 30 ~ "Remove",
    TRUE ~ "Keep"
  ))
```

```
timing_data_2 %>%
  ggplot(aes(x = m_time, fill = remove)) +
  geom_histogram(color = "white") +
  labs(x = "Average response time (seconds)", y = "Number of participants") +
  theme_pubr()
```

```
data_2 = inner_join(data_2, filter(timing_data_2, remove == "Keep")) %>%
  select(-remove)
```

1.4.3 Merge all datasets together

We merge the Time 1 and Time 2 datasets together here.

```
data_2 = data_2 %>%
  select(proid, start_date2, duration_in_seconds2, very_delayed_recall, contains("_3")) %>%
  mutate(time2 = "yes") #indicates participant in time 2

data = data_t1 %>% full_join(data_2)
```

1.5 All data

1.5.1 Reverse score personality items

The following items are (typically) negatively correlated with the others: reckless, moody, worrying, nervous, careless, impulsive. We reverse-score them to ease interpretation of associations and means in the later sections. In short, all traits will be scored such that larger numbers are indicative of the more socially desirable end of the spectrum.

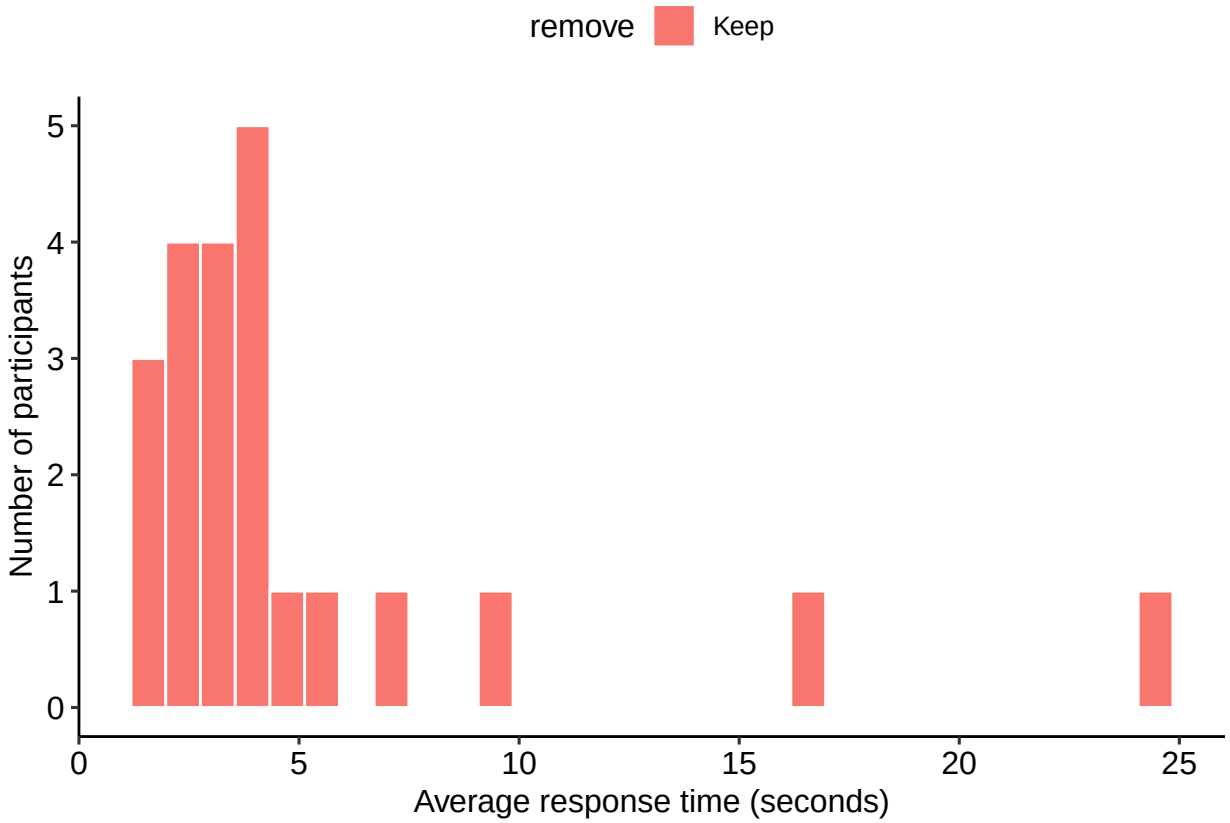


Figure 6: Distribution of average time to respond to personality items in Block 3.

```
data = data %>%
  mutate(
    across(matches("^reckless"),  ~(.x*-1)+7),
    across(matches("^moody"),     ~(.x*-1)+7),
    across(matches("^worrying"),  ~(.x*-1)+7),
    across(matches("^nervous"),   ~(.x*-1)+7),
    across(matches("^careless"),  ~(.x*-1)+7),
    across(matches("^impulsive"), ~(.x*-1)+7))
```

We also create a vector noting the items that are reverse scored. We use this later in tables, to help identify patterns when looking at analyses within-adjective. We use this object elsewhere in the analyses.

```
reverse = c("reckless", "moody", "worrying", "nervous", "careless", "impulsive")
```

1.5.2 Score memory task

Now we score the memory task. We start by creating vectors of the correct responses.

```
correct1 = c("book", "child", "gold", "hotel", "king",
             "market", "paper", "river", "skin", "tree")

correct2 = c("butter", "college", "dollar", "earth", "flag",
             "home", "machine", "ocean", "sky", "wife")

correct3 = c("blood", "corner", "engine", "girl", "house",
             "letter", "rock", "shoes", "valley", "woman")

correct4 = c("baby", "church", "doctor", "fire", "garden",
             "palace", "sea", "table", "village", "water")
```

Next we convert all responses to lowercase. Then we break the string of responses into a vector containing many strings.

```
data = data %>%
  mutate(
    across(matches("recall"),tolower), # convert to lower
    #replace carriage return with space
    across(matches("recall"), str_replace_all, pattern = "\\n", replacement = ","),
    # remove spaces
    across(matches("recall"), str_replace_all, pattern = " ", replacement = ","),
    # remove doubles
    across(matches("recall"), str_replace_all, pattern = ",,", replacement = ","),
    #remove last comma
    across(matches("recall"), str_remove, pattern = ",$"),
    # split the strings based on the spaces
    across(matches("recall"), str_split, pattern = ","))
```

1.5.2.1 Immediate recall

Now we use the `amatch` function in the `stringdist` package to look for exact (or close) matches to the target words. This function returns for each word either the position of the key in which you can find the target word or `NA` to indicate the word or a close match does not exist in the string.

```
distance = 1 #maximum distance between target word and correct response
data = data %>%
  mutate(
    memory1 = map(recall1, ~sapply(., amatch, correct1, maxDist = distance)),
    memory2 = map(recall2, ~sapply(., amatch, correct2, maxDist = distance)),
    memory3 = map(recall3, ~sapply(., amatch, correct3, maxDist = distance)),
    memory4 = map(recall4, ~sapply(., amatch, correct4, maxDist = distance))
    )
```

We count the number of correct answers. This gets complicated; in lieu of writing out a paragraph explanation, we have opted for in-text comments to orient those interested in following the code.

```
data = data %>%
  mutate(
    across(starts_with("memory"),
      #replace position with 1
      ~map(., sapply, FUN = function(x) ifelse(x >0, 1, 0))),
    across(starts_with("recall"),
           # are there non-missing values in the original response?
           ~map_dbl(.,
                    .f = function(x) sum(!is.na(x))),
           .names = "{.col}_miss"),
    across(starts_with("memory"),
      #replace position with 1
      # count the number of correct answers
      ~map_dbl(., sum, na.rm=T))) %>%
  mutate(
    memory1 = case_when(
      # if there were no responses, make the answer NA
      recall1_miss == 0 ~ NA_real_,
      # otherwise, the number of correct guesses
      TRUE ~ memory1),
    memory2 = case_when(
      recall2_miss == 0 ~ NA_real_,
      TRUE ~ memory2),
    memory3 = case_when(
      recall3_miss == 0 ~ NA_real_,
      TRUE ~ memory3),
    memory4 = case_when(
      recall4_miss == 0 ~ NA_real_,
      TRUE ~ memory4)) %>%
  # no longer need the missing count variables
  select(-ends_with("miss"))
```

Finally, we want to go from 4 columns (one for each recall test), to two: one that has the number of correct responses, and one that indicates which version they saw.

```
data = data %>%
  select(proid, starts_with("memory")) %>%
  gather(mem_condition, memory, -proid) %>%
  filter(!is.na(memory)) %>%
  mutate(mem_condition = str_remove(mem_condition, "memory")) %>%
  full_join(data)
```

To demonstrate the accuracy of the code, here we present a random subset of participants' raw responses and their assigned memory score.

```
#from memory condition 1
data %>%
  filter(mem_condition == 1) %>%
  select(recall1, memory) %>%
  sample_n(3) %>%
  mutate(recall1 = map_chr(recall1, paste, collapse = ", "))
```

```
## # A tibble: 3 x 2
##   recall1                                                          memory
##   <chr>                                                             <dbl>
## 1 tree, skin, river, paper, market, king, hotel, gold, child, book     10
## 2 book, hotel, market, chain, paper, gold                               5
## 3 book, king, market, gold                                              4
```

```
#from memory condition 2
data %>%
  filter(mem_condition == 2) %>%
  select(recall2, memory) %>%
  sample_n(3) %>%
  mutate(recall2 = map_chr(recall2, paste, collapse = ", "))
```

```
## # A tibble: 3 x 2
##   recall2                                      memory
##   <chr>                                         <dbl>
## 1 butter, college, earth, wife, ocean, machine      6
## 2 wife, warm, friendly                              1
## 3 college                                           1
```

```
#from memory condition 3
data %>%
  filter(mem_condition == 3) %>%
  select(recall3, memory) %>%
  sample_n(3) %>%
  mutate(recall3 = map_chr(recall3, paste, collapse = ", "))
```

```
## # A tibble: 3 x 2
##   recall3                                                        memory
##   <chr>                                                           <dbl>
## 1 blood, corner.girl.woman, house, shoes                              3
## 2 blood, corner, engine, girl, home, letter, rock, shoes, valley, woman     9
## 3 blood, corner, woman, rock, shoes, girl, valley                     7
```

```
#from memory condition 4
data %>%
  filter(mem_condition == 4) %>%
  select(recall4, memory) %>%
  sample_n(3) %>%
  mutate(recall4 = map_chr(recall4, paste, collapse = ", "))
```

```
## # A tibble: 3 x 2
##   recall4                                                memory
##   <chr>                                                   <dbl>
## 1 table, sea, village, church, baby, palace                   6
## 2 baby, church, fire, garden, palace, sea, table, water       8
## 3 baby, church, water, fire, garden                           5
```

Participants remember on average 5.80 words correctly ($SD = 2.73$),

```
data %>%
  ggplot(aes(x = memory)) +
  geom_histogram(bins = 11, color = "white") +
  labs(x = "Number of correct responses") +
  scale_x_continuous(breaks = 0:10) +
  theme_pubr()
```

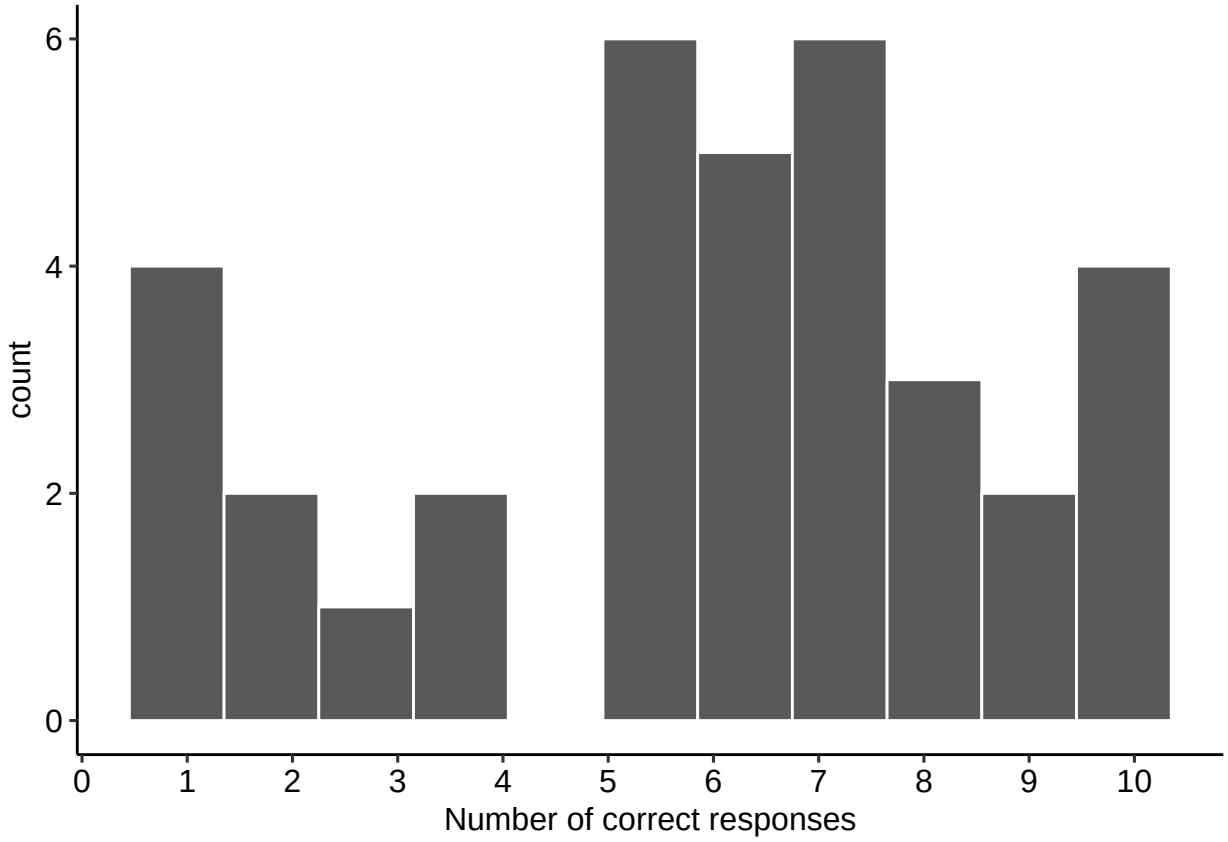


Figure 7: Correct responses on the memory task

```
data %>%
  group_by(mem_condition) %>%
  summarise(
    m = mean(memory),
    s = sd(memory),
    min = min(memory),
    max = max(memory),
```

Table 1: Memory responses by condition

Condition	Mean	SD	Min	Max	N
1	5.50	2.56	1	10	8
2	4.62	3.20	1	10	8
3	6.40	2.72	1	10	10
4	6.44	2.51	2	10	9

```
    n = n()
  ) %>%
  kable(booktabs = T,
        col.names = c("Condition", "Mean", "SD", "Min", "Max", "N"),
        digits = c(0, 2, 2, 1, 1, 1),
        caption = "Memory responses by condition") %>%
  kable_styling()
```

1.5.2.2 Delayed recall A challenge with the delayed recall task is identifying the memory condition that participants were assigned to, but this is made easier by the work done above. The following code mainly reproduces the steps used for scoring the immediate memory recall task. The main difference is that we have a single column containing all responses (`delayed_recall`), regardless of which memory condition participants were assigned to. We score this response against all four answer keys, then select the maximum (best) score.

```
mem2 = data %>%
  select(proid, mem_condition, delayed_recall) %>%
  mutate(newid = 1:nrow(.))

mem2 = mem2 %>%
  mutate(
    delayed_recall1 = map(delayed_recall, ~sapply(., amatch, correct1, maxDist = distance)),
    delayed_recall2 = map(delayed_recall, ~sapply(., amatch, correct2, maxDist = distance)),
    delayed_recall3 = map(delayed_recall, ~sapply(., amatch, correct3, maxDist = distance)),
    delayed_recall4 = map(delayed_recall, ~sapply(., amatch, correct4, maxDist = distance))
    ) %>%
  gather(variable, delayed_memory, delayed_recall1:delayed_recall4)

mem2 = mem2 %>%
  mutate(
      delayed_memory = map(delayed_memory, sapply,
                  FUN = function(x) ifelse(x >0, 1, 0)),
      # count the number of correct answers
      delayed_memory = map_dbl(delayed_memory, sum, na.rm=T))

mem2 = mem2 %>%
  group_by(proid) %>%
  filter(delayed_memory == max(delayed_memory)) %>%
  filter(row_number() == 1 ) %>%
  select(-delayed_recall, -variable, -newid)

data = inner_join(data, mem2)
```

```
data %>%
  ggplot(aes(x = delayed_memory)) +
  geom_histogram(color = "white", bins = 11) +
  scale_x_continuous("Number correct", breaks = c(0:10)) +
  labs(y = "Number of participants") +
  theme_pubr()
```

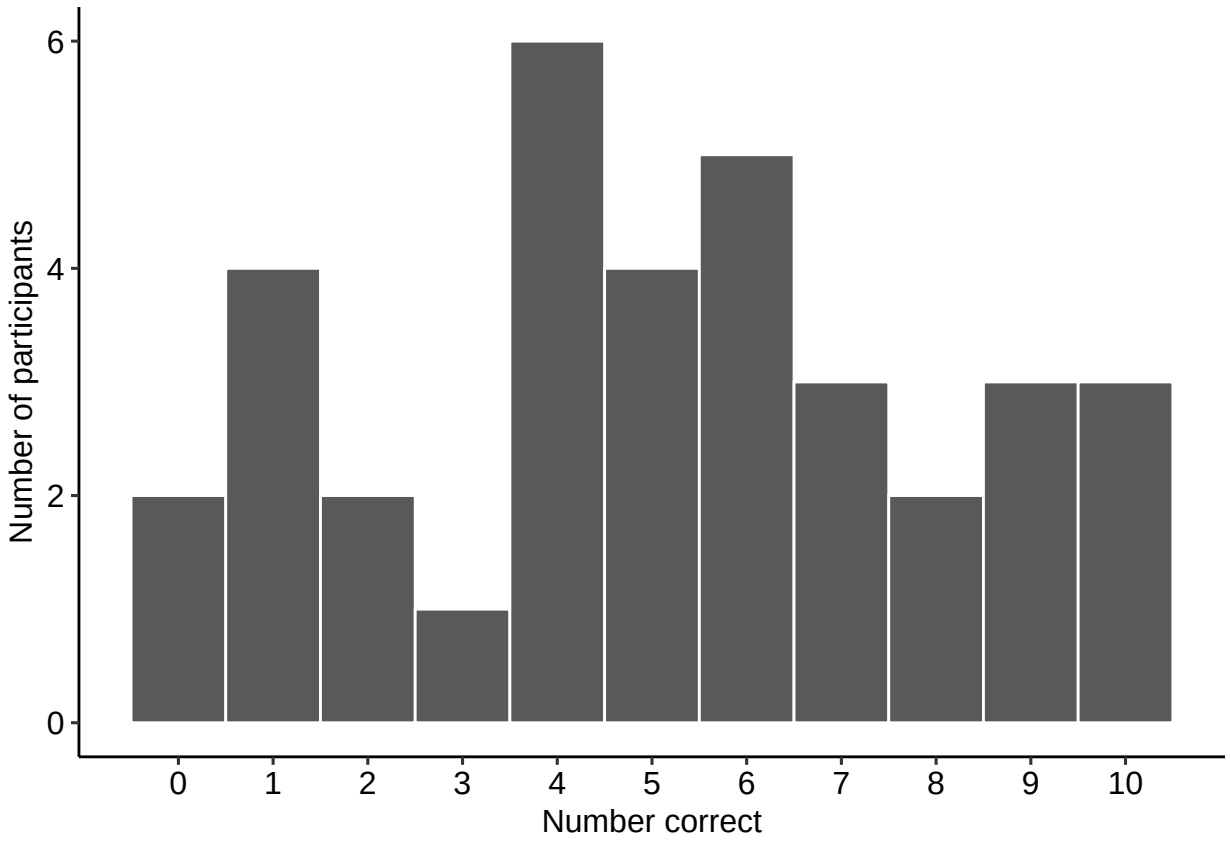


Figure 8: Distribution of delayed memory scores

```
data %>%
  ggplot(aes(x = memory, y = delayed_memory)) +
  geom_point() +
  geom_smooth(method = "lm") +
  scale_x_continuous("Immediate number correct", breaks = c(0:10)) +
  scale_y_continuous("Delayed number correct", breaks = c(0:10)) +
  labs(title = paste0("r = ", printnum(cor(data$memory, data$delayed_memory, use = "pairwise")))) +
  theme_pubr()
```

1.5.2.3 Very-delayed recall Finally, we score the memory challenge posed at Time 2. Like scoring the delayed recall task, we have a single column containing responses fromo all participants, regardless of the original memory condition.

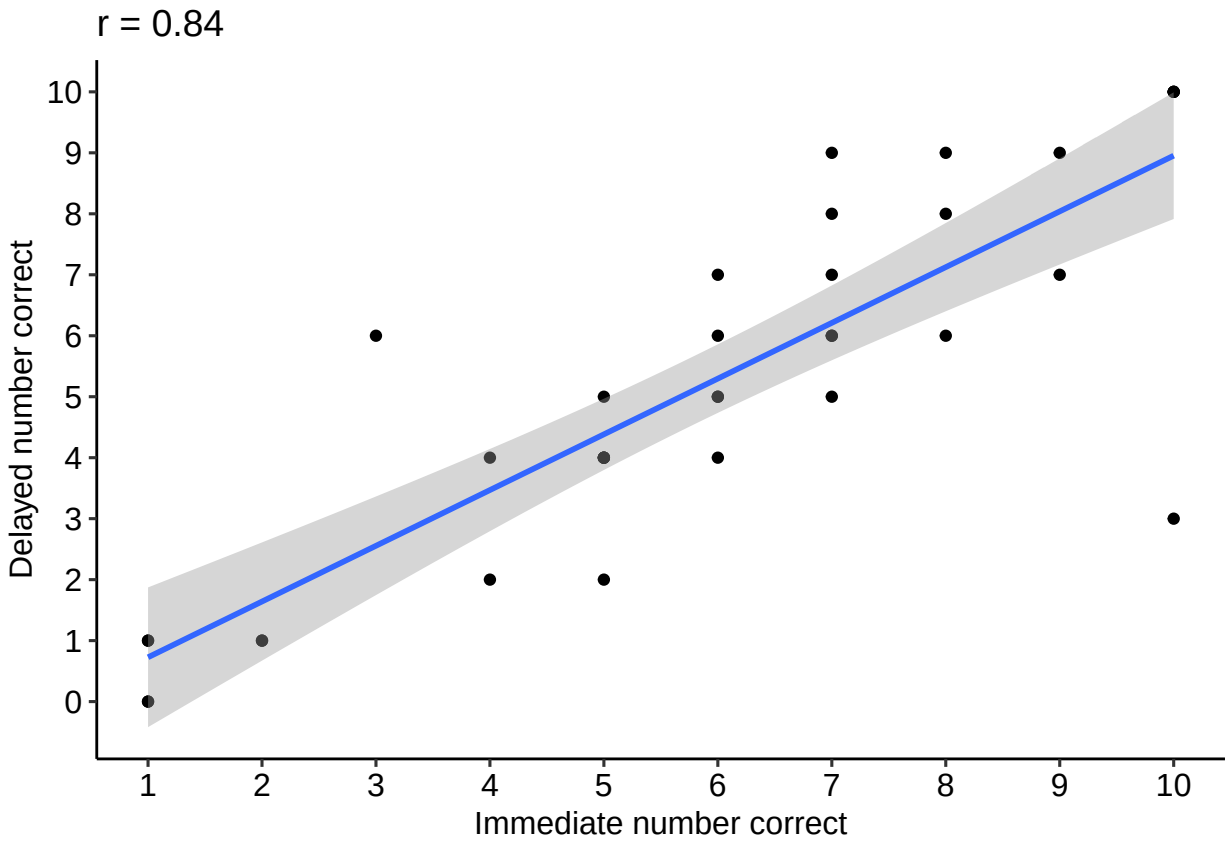


Figure 9: Relationship between immediate and delayed recall

```
mem3 = data %>%
  filter(time2 == "yes") %>%
  select(proid, mem_condition, very_delayed_recall) %>%
  mutate(newid = 1:nrow(.))

mem3 = mem3 %>%
  mutate(
    very_delayed_recall1 = map(very_delayed_recall, ~sapply(., amatch, correct1, maxDist = distance)),
    very_delayed_recall2 = map(very_delayed_recall, ~sapply(., amatch, correct2, maxDist = distance)),
    very_delayed_recall3 = map(very_delayed_recall, ~sapply(., amatch, correct3, maxDist = distance)),
    very_delayed_recall4 = map(very_delayed_recall, ~sapply(., amatch, correct4, maxDist = distance))
    ) %>%
  gather(variable, very_delayed_memory, very_delayed_recall1:very_delayed_recall4)

mem3 = mem3 %>%
  mutate(
      very_delayed_memory = map(very_delayed_memory, sapply,
                  FUN = function(x) ifelse(x >0, 1, 0)),
      # count the number of correct answers
      very_delayed_memory = map_dbl(very_delayed_memory, sum, na.rm=T))

mem3 = mem3 %>%
  group_by(proid) %>%
  filter(very_delayed_memory == max(very_delayed_memory)) %>%
  filter(row_number() == 1 ) %>%
  select(-very_delayed_recall, -variable, -newid)

data = full_join(data, mem3)
```

```
data %>%
  ggplot(aes(x = very_delayed_memory)) +
  geom_histogram(color = "white", bins = 11) +
  scale_x_continuous("Number correct", breaks = c(0:10)) +
  labs(y = "Number of participants") +
  theme_pubr()
```

```
data %>%
  ggplot(aes(x = memory, y = very_delayed_memory)) +
  geom_point() +
  geom_smooth(method = "lm") +
  scale_x_continuous("Immediate number correct", breaks = c(0:10)) +
  scale_y_continuous("Very delayed number correct", breaks = c(0:10)) +
  labs(title = paste0("r = ", printnum(cor(data$memory, data$delayed_memory, use = "pairwise")))) +
  theme_pubr()
```

```
data %>%
  select(matches("memory$")) %>%
  corr.test
```

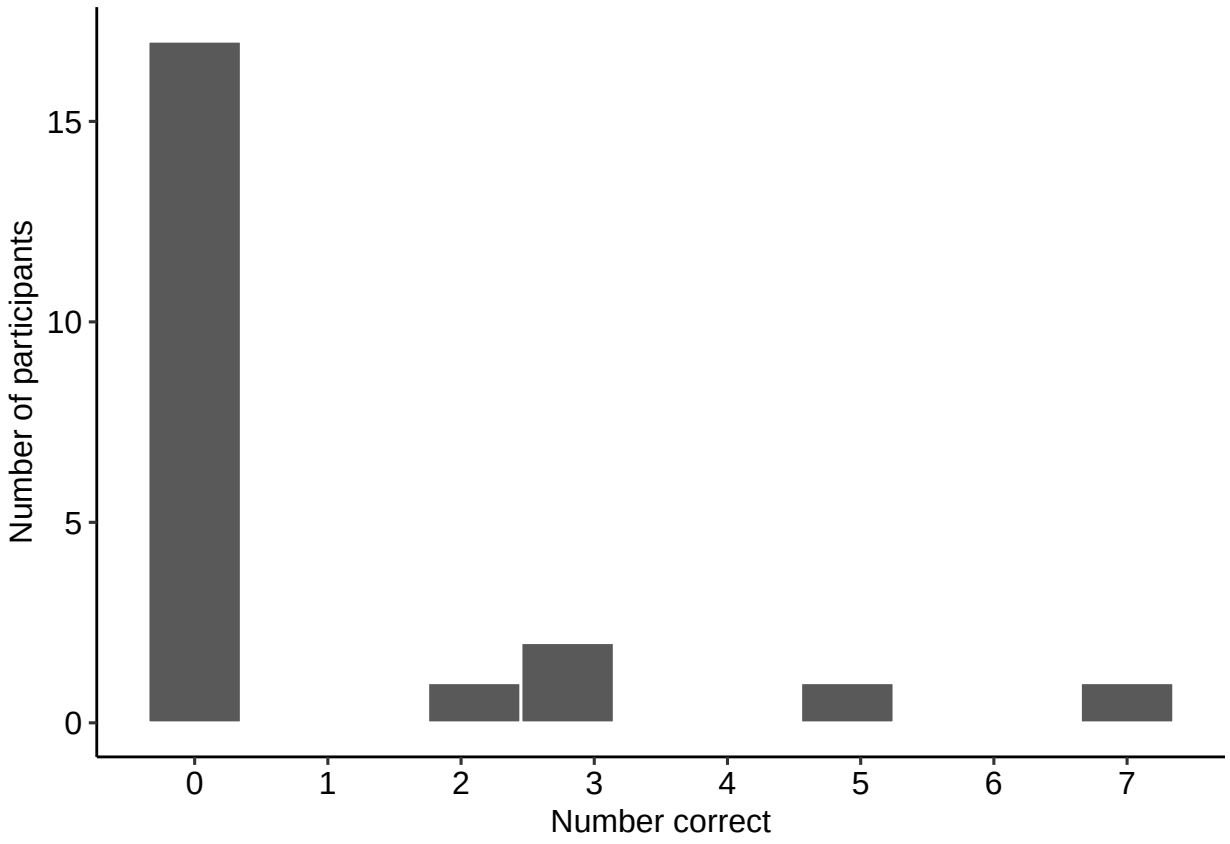


Figure 10: Distribution of delayed memory scores

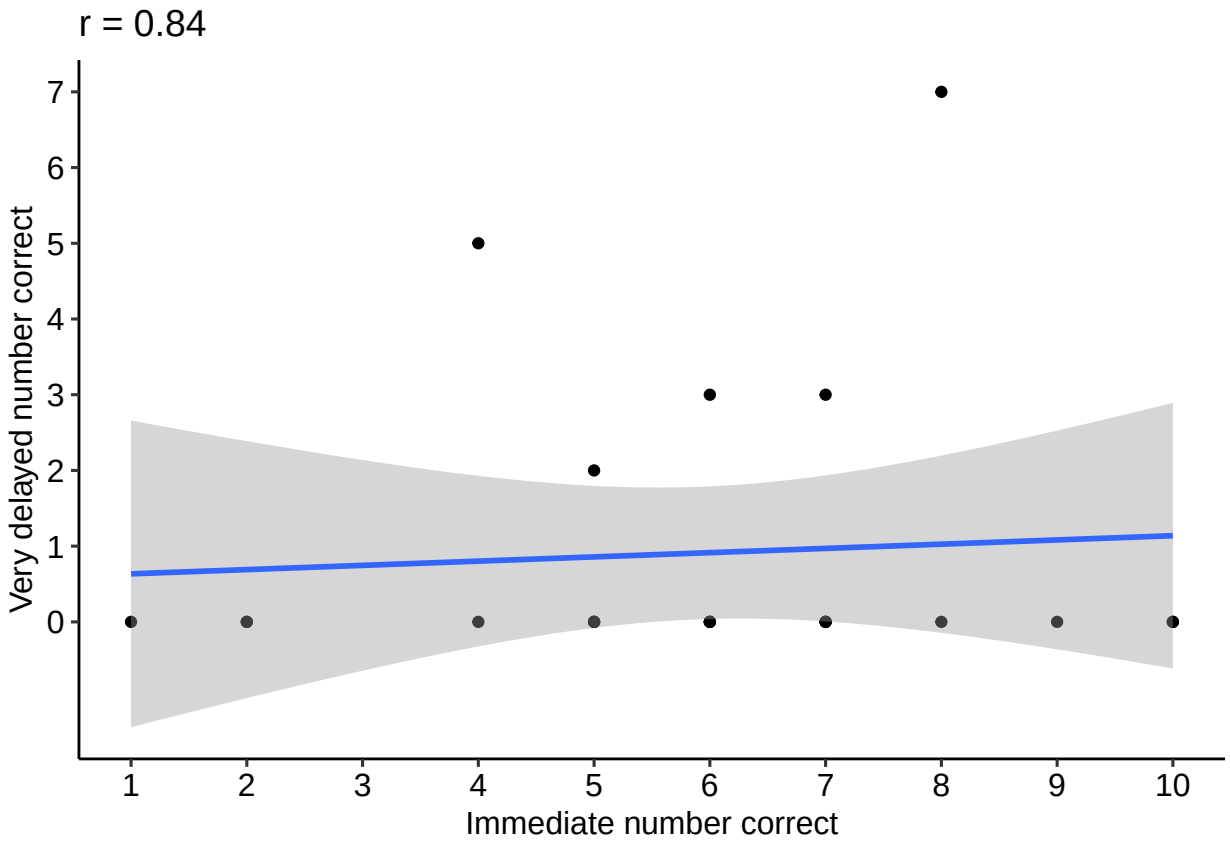


Figure 11: Relationship between immediate and delayed recall

1.5.2.4 Correlations

```
## Call:corr.test(x = .)
## Correlation matrix
##                      memory delayed_memory very_delayed_memory
## memory                 1.00           0.84                0.07
## delayed_memory         0.84           1.00                0.06
## very_delayed_memory    0.07           0.06                1.00
## Sample Size
##                      memory delayed_memory very_delayed_memory
## memory                   35             35                  22
## delayed_memory           35             35                  22
## very_delayed_memory      22             22                  22
## Probability values (Entries above the diagonal are adjusted for multiple tests.)
##                      memory delayed_memory very_delayed_memory
## memory                 0.00           0.00                   1
## delayed_memory         0.00           0.00                   1
## very_delayed_memory    0.76           0.81                   0
##
##  To see confidence intervals of the correlations, print with the short=FALSE option
```

1.5.3 Change labels of device variable

Longer labels were provided to participants ifor clarity. However, we will use shorter labels in our analyses and figures.

```
data = data %>%
  mutate(devicetype = factor(
    devicetype,
    levels = c("Desktop or laptop computer", "Mobile",
               "Tablet (for example, iPad, Galaxy Tablet, Amazon Fire, etc.)"),
    labels = c("Computer", "Mobile", "Tablet")
  ))
```

1.5.4 Reorder demographic categories

We set the order of ordinal demographic variables, which helps generate more interpretable figures and tables.

```
data = data %>%
  mutate(edu = factor(edu,
                      levels = c(
                        "Less than 12 years",
                        "High school graduate/GED",
                        "Currently in college/university",
                        "Some college/university, but did not graduate",
                        "Associate degree (2 year)",
                        "College/university degree (4 year)",
                        "Currently in graduate or professional school",
                        "Graduate or professional school degree"))) %>%
  mutate(hhinc = str_remove(hhinc, " a year"),
         hhinc = str_replace_all(hhinc, ",000", "K"),
```

```
        hhinc = str_replace_all(hhinc, " to ", "-"),
        hhinc = str_replace_all(hhinc, "less than", "<"),
        hhinc = str_replace_all(hhinc, "more than", ">"))%>%
  mutate(hhinc = factor(hhinc,
                        levels = c(
                          "< $20,000",
                          "$20K-$40K",
                          "$40K-$60K",
                          "$60K-$80K",
                          "$80K-$100K",
                          "$100K-$120K",
                          "$120K-$150K",
                          "$150K-$200K",
                          "$200K-$250K",
                          "$250K-$350K",
                          "$350K-$500K",
                          ">$500K"
                        )))
```

1.5.5 Long-form dataset

We need one dataset that contains the responses to and timing of the personality items in long form. This will be used for nearly all the statistical models, which will nest items within person. To create this, we first select the responses to the items of different formats. For this set of analyses, we use data collected in both Block 1 and Block 2 – that is, each participant saw the same format for every item during Block 1, but a random format for each item in Block 2.

These variable names have one of four formats: `[trait]_[abcd]` (for example, `talkative_a`), `[trait]_[abcd]_2` (for example, `talkative_a_2`), `[trait]_[abcd]_3` (e.g., `talkative_a_3`), or `[trait]_[abcd]_3i` (e.g., `talkative_a_3i`). We search for these items using regular expressions.

```
item_responses = str_subset(
  names(data),
  "^([[:alpha:]])+_[abcd](_2)?(_3)?(i)?$"
)
```

Similarly, we'll need to know how long it took participants to respond to these items. These variable names have one of four formats listed above followed by the string `page_submit`. We search for these items using regular expressions.

```
item_timing = str_subset(names(data), "t_([[:alpha:]])+_[abcd](_2)?(_3)?(i)?_page_submit$")
```

We extract just the participant IDs, delayed memory, and these variables.

```
items_df = data %>%
  select(proid, condition, time2,
         memory, delayed_memory, very_delayed_memory,
         devicetype,
         all_of(item_responses), all_of(item_timing))
```

Next we reshape these data into long form. This requires several steps. We'll need to identify whether each value is a response or timing; we can use the presence of the string `t_` for this. Next, we'll identify the block

based on whether the string contains `_2` or `_3`. We also identify whether it ends with `i`, indicating the item in block 3 started with "I". Then, we identify the condition based on which letter (`a`, `b`, `c`, or `d`) follows an underscore. Throughout, we'll strip the item string of extraneous information until we're left with only the adjective assessed. Finally, we'll use spread to create separate columns for the response and the timing variables.

```
items_df = items_df %>%
  gather(item, value, all_of(item_responses), all_of(item_timing)) %>%
  filter(!is.na(value)) %>%
  # identify whether timing or response
  mutate(variable = ifelse(str_detect(item, "^t_"), "timing", "response"),
         item = str_remove(item, "^t_"),
         item = str_remove(item, "_page_submit$")) %>%
  #identify block
  mutate(
    block = case_when(
      str_detect(item, "_2") ~ "2",
      str_detect(item, "_3") ~ "3",
      TRUE ~ "1"),
    item = str_remove(item, "_[23]")) %>%
  # identify presence of "I"
  mutate(i = case_when(
    str_detect(item, "i$") ~ "Present",
    TRUE ~ "Absent"),
    item = str_remove(item, "i$")) %>%
  separate(item, into = c("item", "format")) %>%
  spread(variable, value)
```

1.5.5.1 Remove 'human' and 'asleep'

We also remove responses to the adjectives "human" and "asleep", as these are not personality items per-se and included for the purpose of attention checks.

```
items_df = items_df %>%
  filter(item != "human") %>%
  filter(item != "asleep")
```

1.5.5.2 Label formatting conditions

We give labels to the formats, to clarify interpretations and aid table and figure construction.

```
items_df$format = as.factor(items_df$format)
items_df$format = relevel(items_df$format, ref = "a")
items_df$format = factor(items_df$format,
                                levels = c("a","b","c","d"),
                                labels = c("Adjective\nOnly", "Am\nAdjective", "Tend to be\nAdjective",
```

1.5.5.3 Transform seconds

The variable `seconds` appears to have a very severe right skew. We log-transform this variable for later analyses.

```
items_df = items_df %>%
  mutate(seconds_log = log(timing))
```

```r
items_df %>%
  gather(variable, value, timing, seconds_log) %>%
  mutate(variable = factor(variable,
                           levels = c("timing", "seconds_log"),
                           labels = c("Seconds (raw)", "Seconds (log)"))) %>%
  ggplot(aes(x = value)) +
  geom_histogram(bins = 100) +
  facet_wrap(~variable, scales = "free") +
  labs(x = NULL, y = "Number of participants") +
  theme_pubr()
```

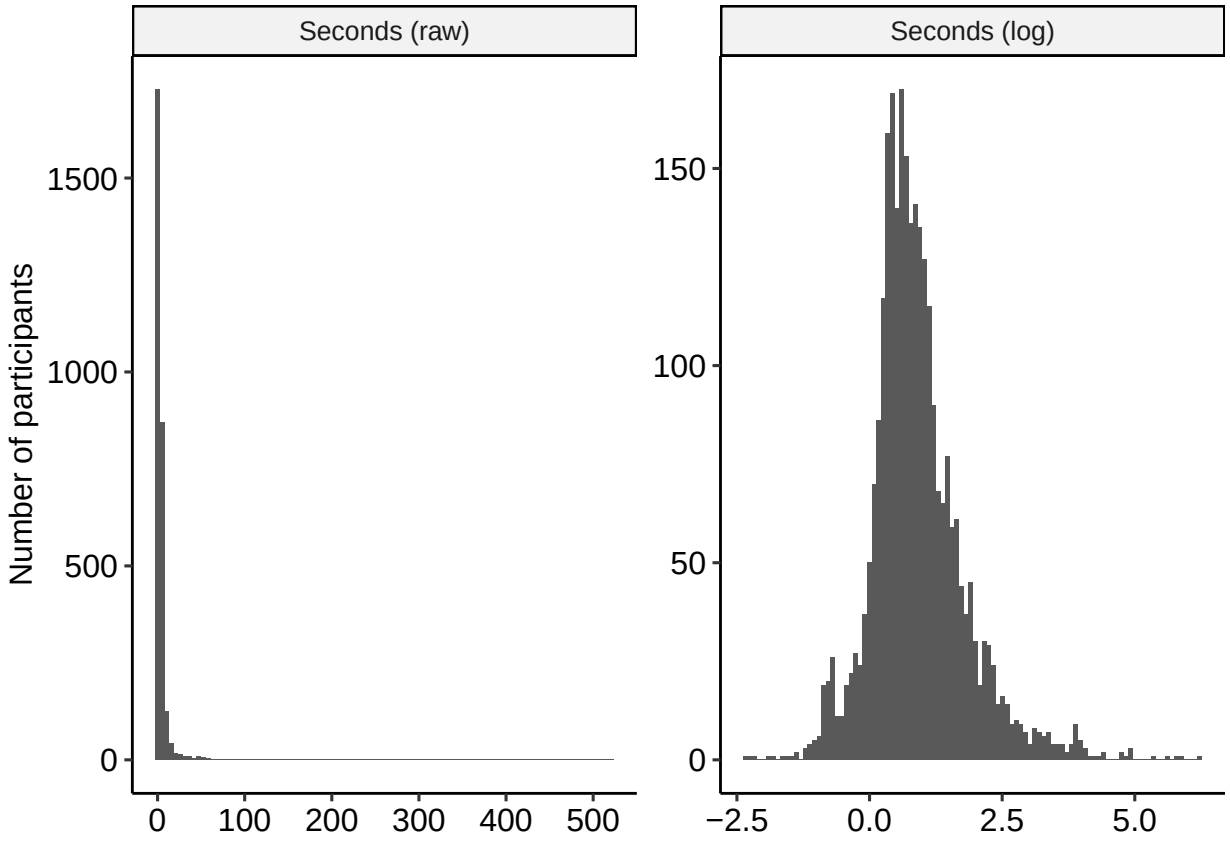


Figure 12: Distribution of seconds, raw and transformed.

2 Descriptives

Here we present the descriptive statistics reported in the manuscript.

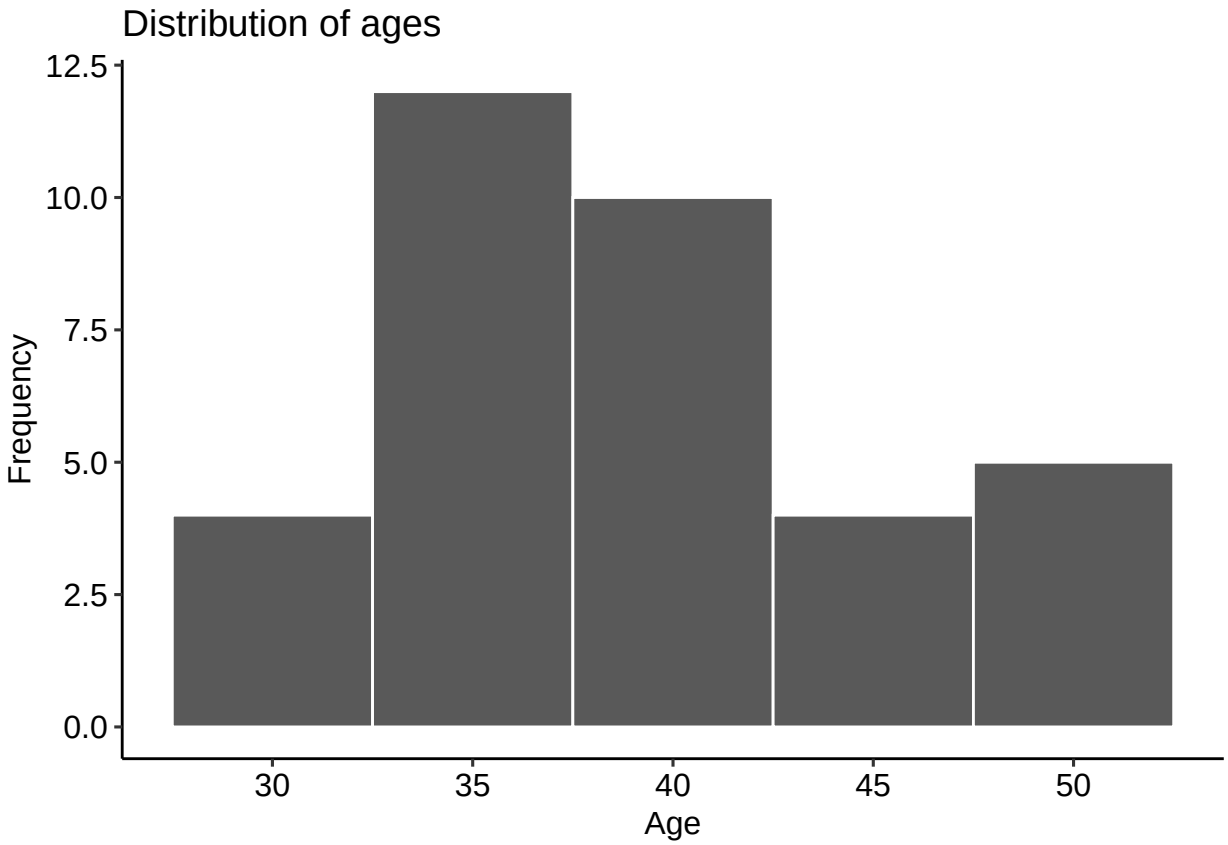
2.1 Demographics

2.1.1 Age

```
data %>%
  select(age) %>%
  summarise(across(age, list(mean = mean,
                             sd = sd,
                             min = min,
                             max = max)))
```

```
## # A tibble: 1 x 4
##   age_mean age_sd age_min age_max
##      <dbl>  <dbl>   <dbl>   <dbl>
## 1     39.4   6.02      31      51
```

```
data %>%
  ggplot(aes(x = age)) +
  geom_histogram(binwidth = 5, color = "white") +
  labs(x = "Age", y = "Frequency",
       title = "Distribution of ages") +
  theme_pubr()
```



2.1.2 Gender

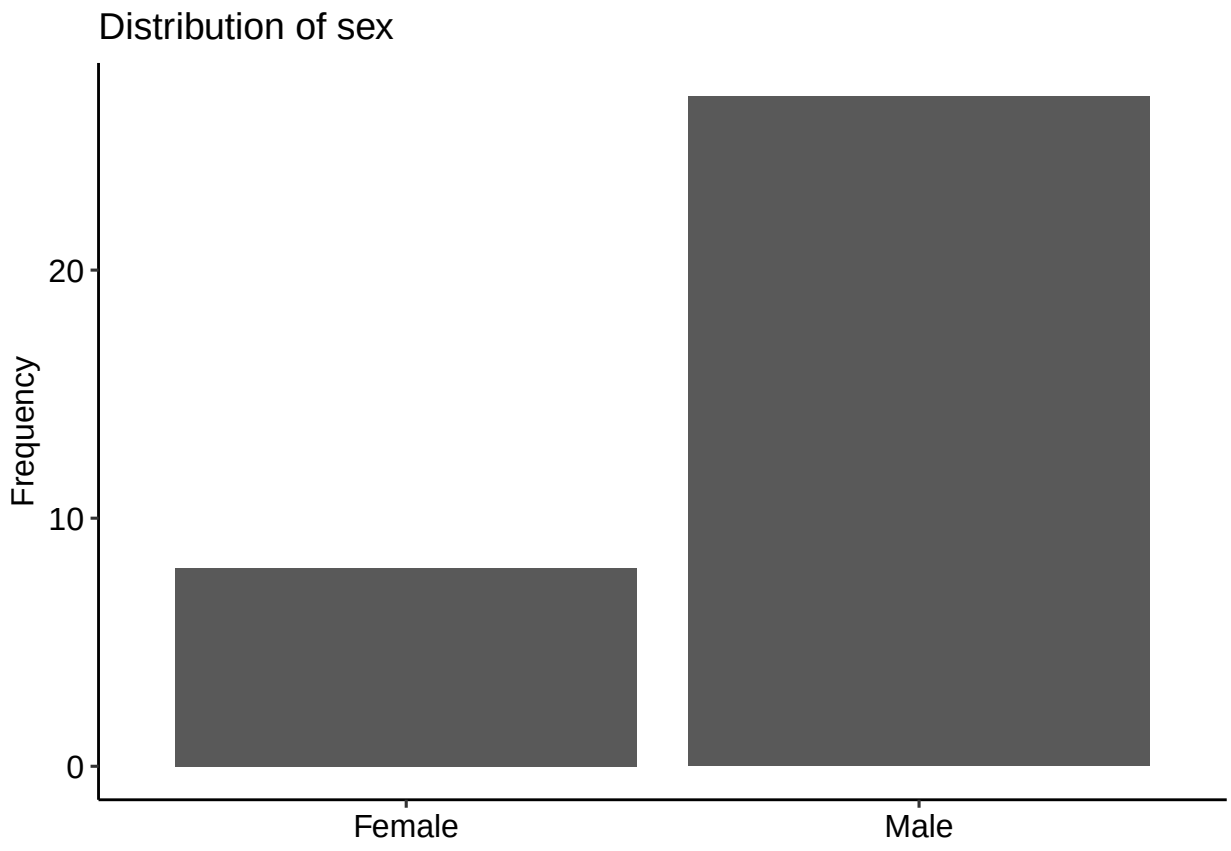
```
table(data$sex)
```

```
##
## Female   Male
##      8     27
```

```
round((table(data$sex)/nrow(data))*100,2)
```

```
##
## Female   Male
##  22.86  77.14
```

```
data %>%
  ggplot(aes(x = sex)) +
  geom_bar(stat = "count") +
  labs(x = NULL, y = "Frequency",
       title = "Distribution of sex") +
  theme_pubr()
```



2.1.3 Education

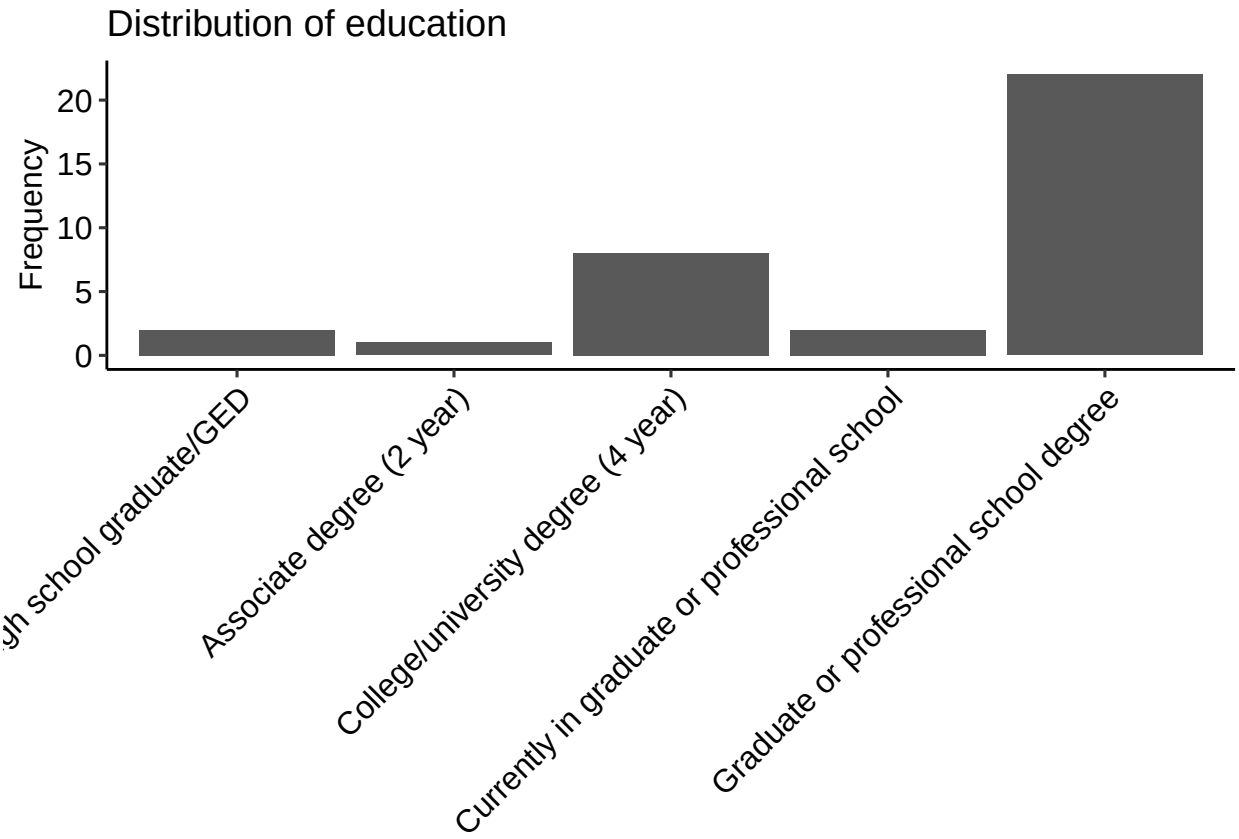
```r
table(data$edu)
```

```
##
##                            Less than 12 years
##                                             0
##                         High school graduate/GED
##                                             2
##                 Currently in college/university
##                                             0
## Some college/university, but did not graduate
##                                             0
##                         Associate degree (2 year)
##                                             1
##              College/university degree (4 year)
##                                             8
##  Currently in graduate or professional school
##                                             2
##         Graduate or professional school degree
##                                            22
```

```r
round((table(data$edu)/nrow(data))*100,2)
```

```
##
##                            Less than 12 years
##                                          0.00
##                         High school graduate/GED
##                                          5.71
##                 Currently in college/university
##                                          0.00
## Some college/university, but did not graduate
##                                          0.00
##                         Associate degree (2 year)
##                                          2.86
##              College/university degree (4 year)
##                                         22.86
##  Currently in graduate or professional school
##                                          5.71
##         Graduate or professional school degree
##                                         62.86
```

```r
data %>%
  ggplot(aes(x = edu)) +
  geom_bar(stat = "count") +
  labs(x = NULL, y = "Frequency",
       title = "Distribution of education") +
  theme_pubr() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



2.1.4 Ethnicity

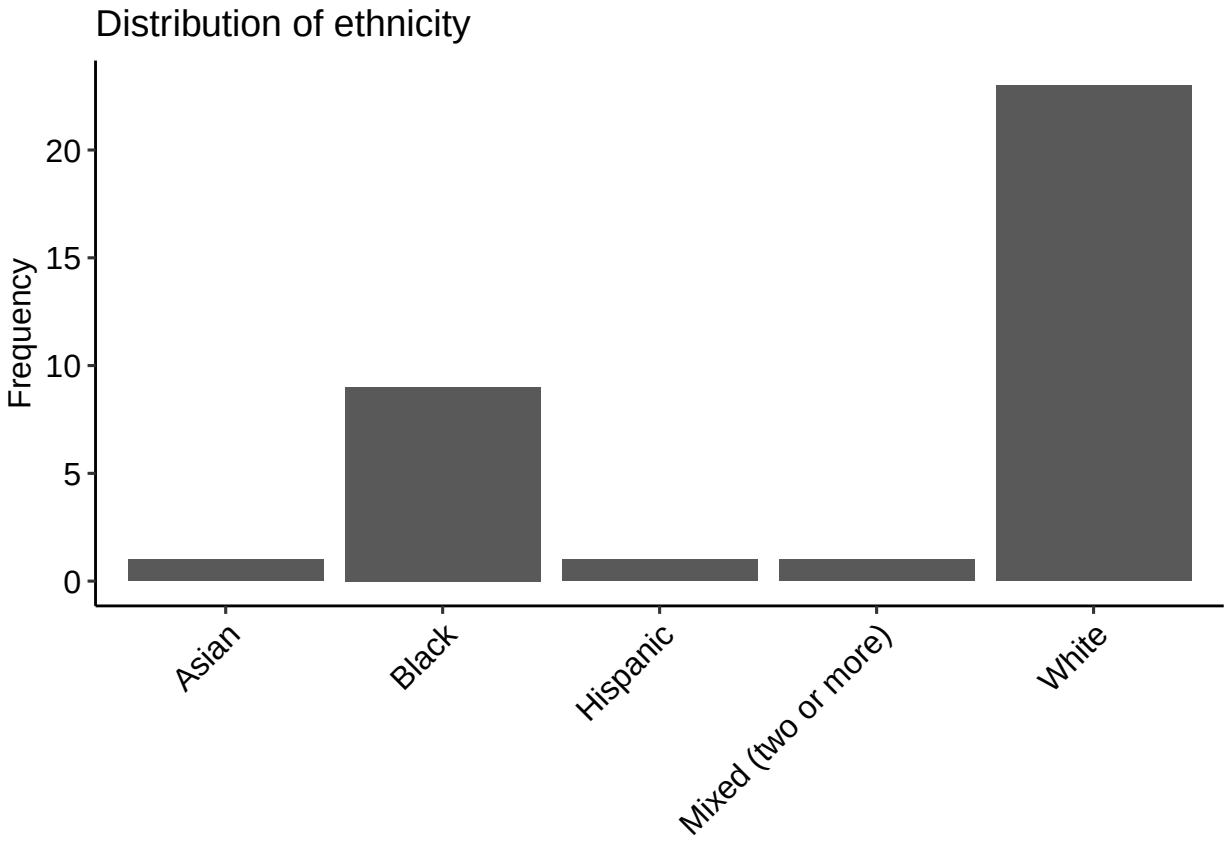
```
table(data$ethnic)
```

```
##
##                Asian                Black             Hispanic Mixed (two or more)
##                    1                    9                    1                    1
##                White
##                   23
```

```
round((table(data$ethnic)/nrow(data))*100,2)
```

```
##
##                Asian                Black             Hispanic Mixed (two or more)
##                 2.86                25.71                 2.86                 2.86
##                White
##                65.71
```

```
data %>%
  ggplot(aes(x = ethnic)) +
  geom_bar(stat = "count") +
  labs(x = NULL, y = "Frequency",
       title = "Distribution of ethnicity") +
  theme_pubr() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



2.1.5 Household income

```
table(data$hhinc)
```

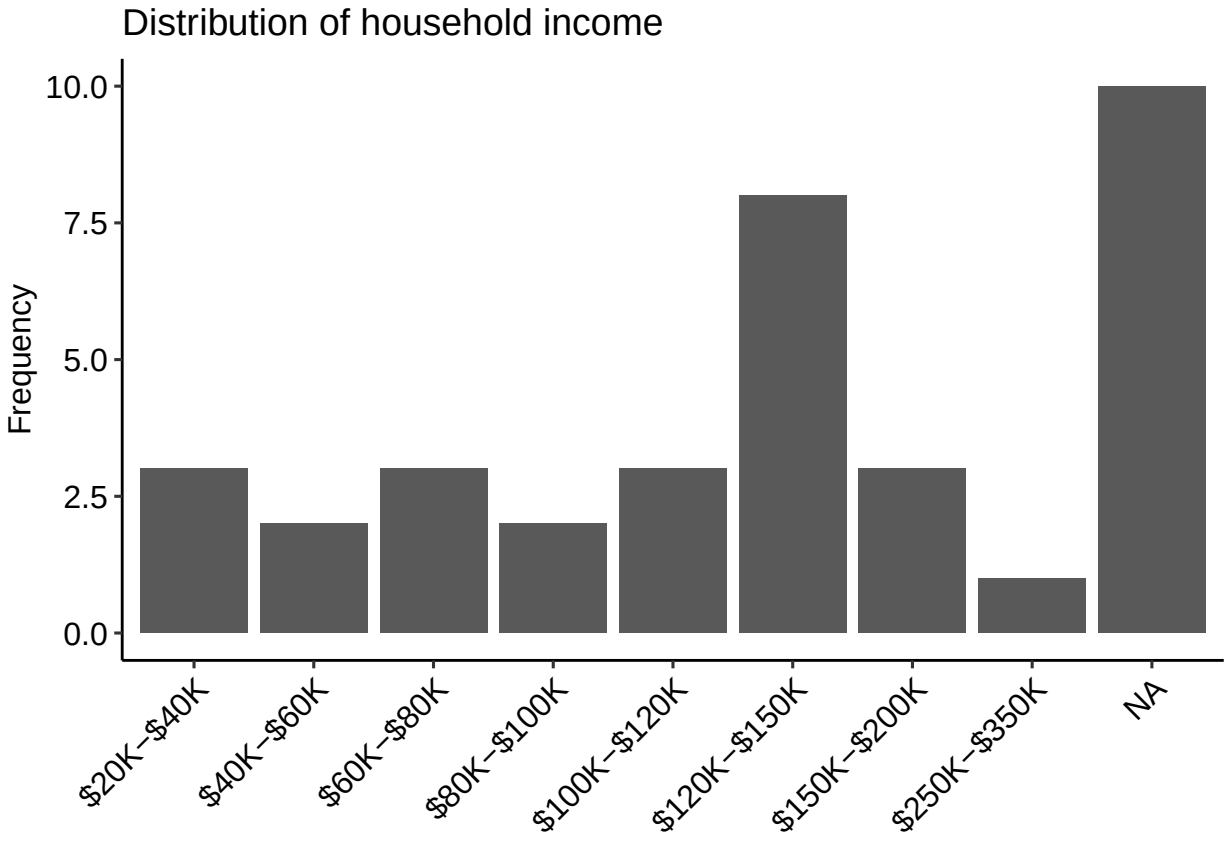
```
##
##   < $20,000   $20K-$40K   $40K-$60K   $60K-$80K  $80K-$100K $100K-$120K
##           0           3           2           3           2           3
## $120K-$150K $150K-$200K $200K-$250K $250K-$350K $350K-$500K      >$500K
##           8           3           0           1           0           0
```

```
round((table(data$hhinc)/nrow(data))*100,2)
```

```
##
##   < $20,000   $20K-$40K   $40K-$60K   $60K-$80K  $80K-$100K $100K-$120K
##        0.00        8.57        5.71        8.57        5.71        8.57
## $120K-$150K $150K-$200K $200K-$250K $250K-$350K $350K-$500K      >$500K
##       22.86        8.57        0.00        2.86        0.00        0.00
```

```
data %>%
  ggplot(aes(x = hhinc)) +
  geom_bar(stat = "count") +
  labs(x = NULL, y = "Frequency",
       title = "Distribution of household income") +
  theme_pubr() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

predictor	estimate	ci	statistic	p.value
Intercept	39.50	$[36.81$, $42.19]$	29.92	$< .001$
DevicetypeMobile	0.00	$[-4.81$, $4.81]$	0.00	$> .999$
DevicetypeTablet	-1.17	$[-8.93$, $6.60]$	-0.31	.761



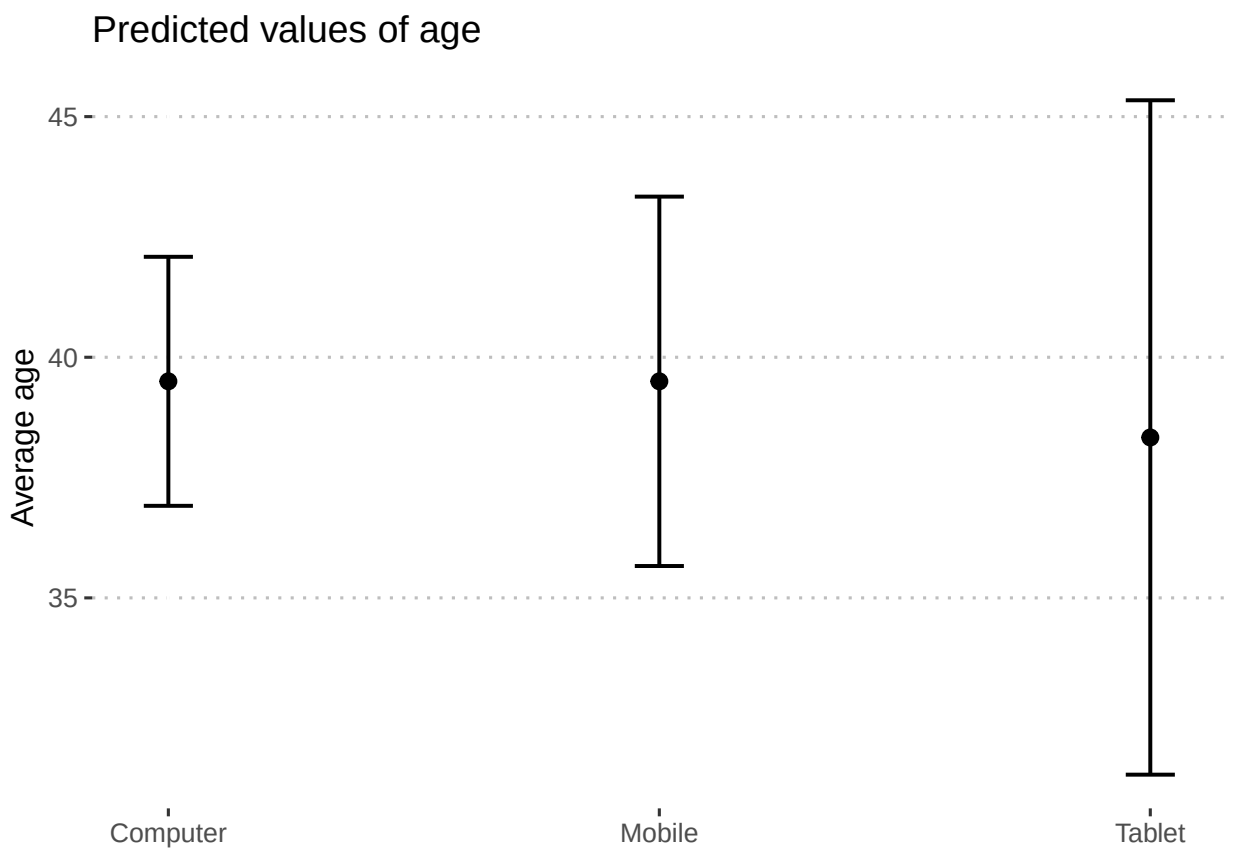
2.2 Demographics by device type

2.2.1 Age

```
age_mod = lm(age ~ devicetype, data = data)

apa_print(age_mod)$table %>%
  kable(booktabs = T) %>%
  kable_styling()
```

```
plot_model(age_mod, type = "pred")$devicetype +
  labs(x = NULL, y = "Average age") +
  theme_pubclean()
```



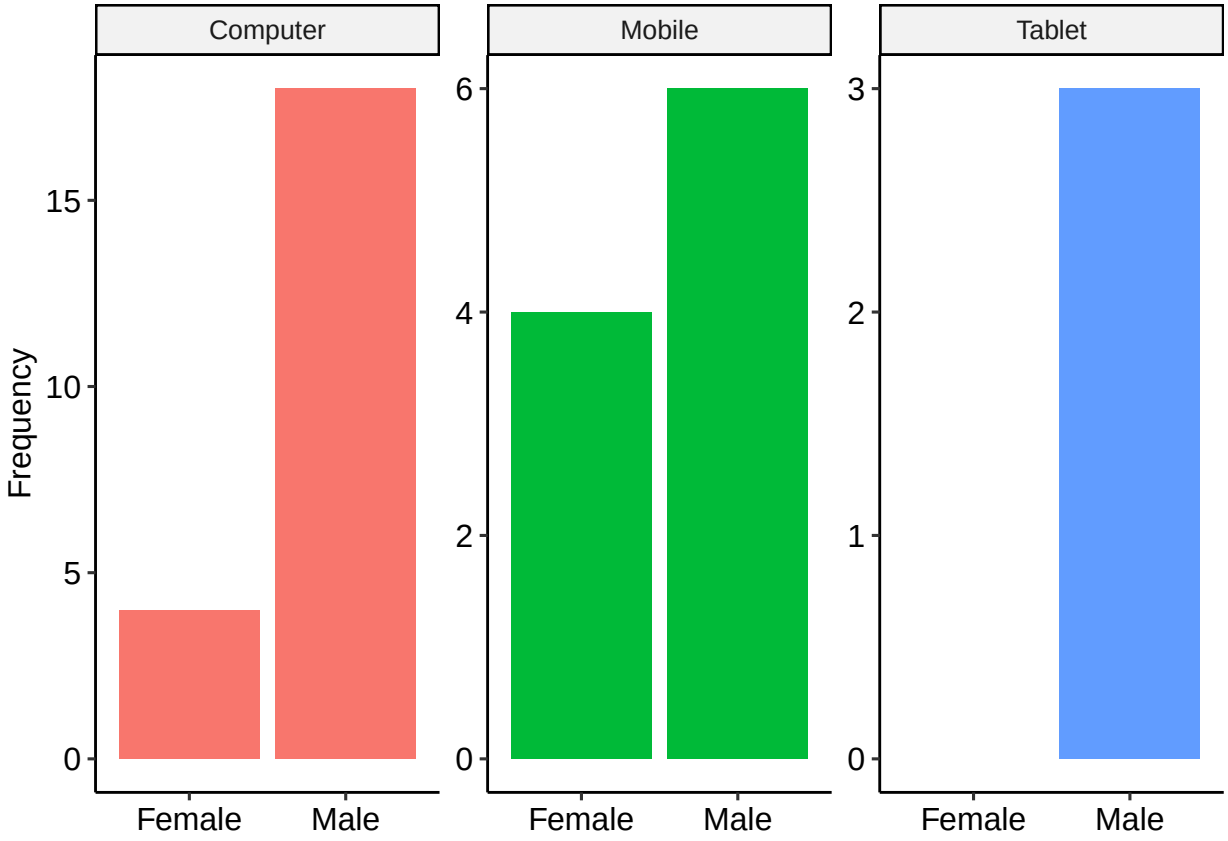
2.2.2 Sex

```
chisq.test(table(data$sex, data$devicetype))
```

```
Pearson's Chi-squared test
```

data: table(data$sex, data$devicetype) X-squared = 2.8283, df = 2, p-value = 0.2431

```
data %>%
  ggplot(aes(x = sex, fill = devicetype)) +
  geom_bar(stat = "count") +
  facet_wrap(~devicetype, scales = "free_y") +
  guides(fill = "none") +
  labs(x = NULL, y = "Frequency") +
  theme_pubr()
```



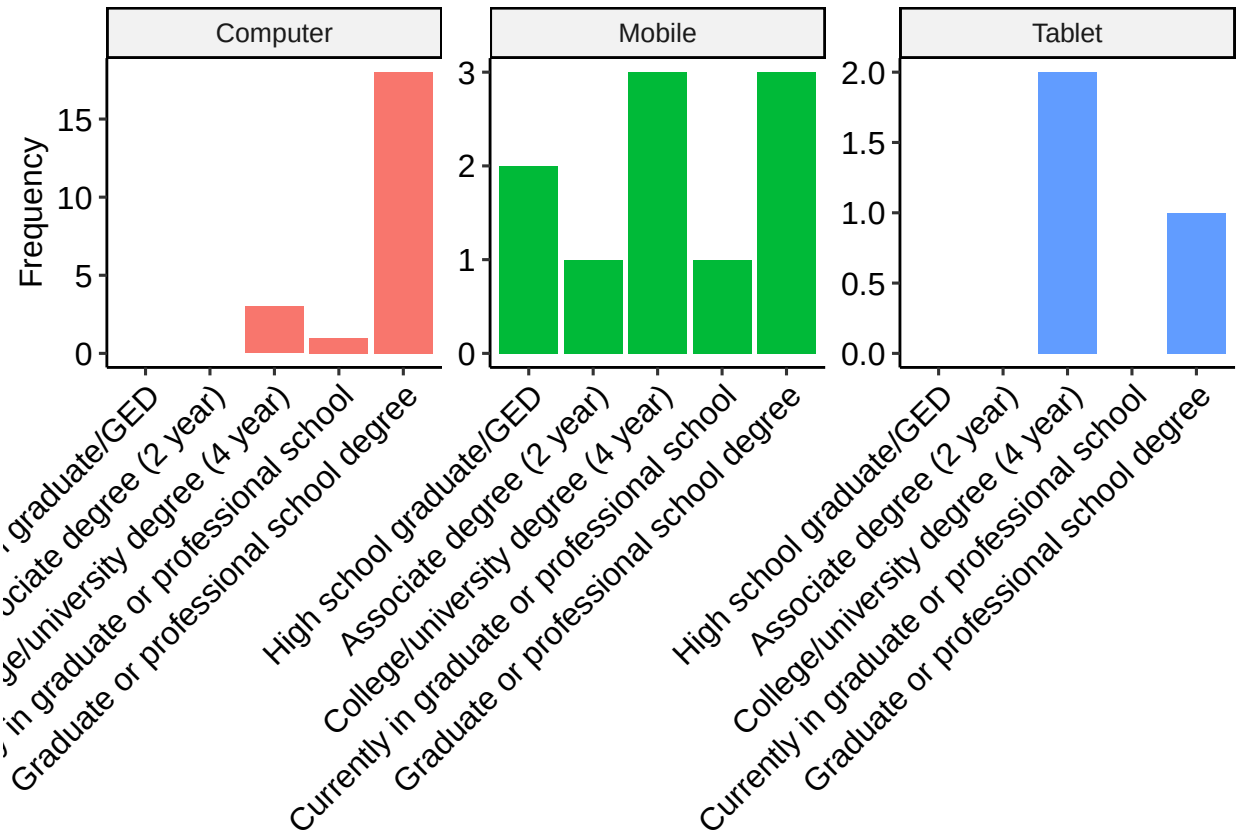
2.2.3 Education

```
chisq.test(table(data$edu, data$devicetype))
```

```
Pearson's Chi-squared test
```

data: table(data$edu, data$devicetype) X-squared = NaN, df = 14, p-value = NA

```
data %>%
  ggplot(aes(x = edu, fill = devicetype)) +
  geom_bar(stat = "count") +
  facet_wrap(~devicetype, scales = "free_y") +
  guides(fill = "none") +
  labs(x = NULL, y = "Frequency") +
  theme_pubr() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



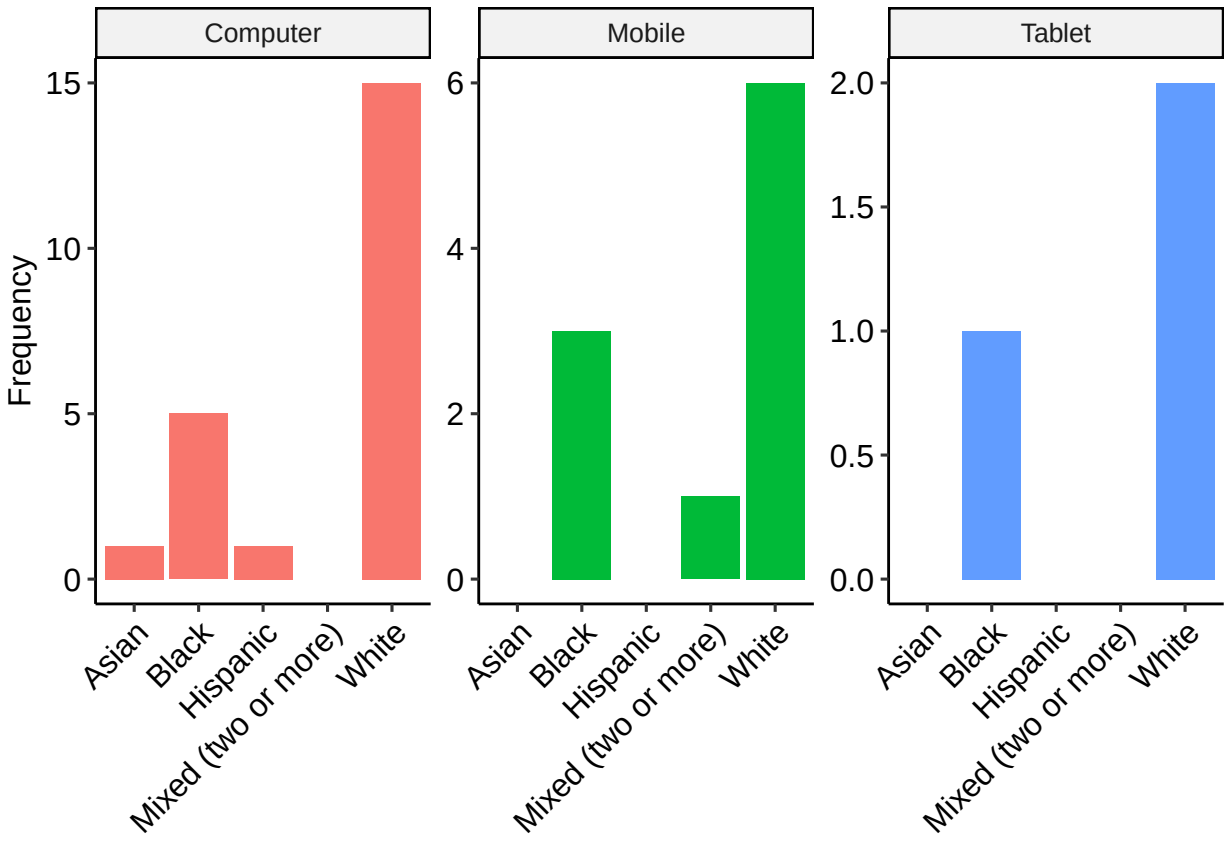
2.2.4 Ethnicity

```
chisq.test(table(data$ethnic, data$devicetype))
```

```
Pearson's Chi-squared test
```

data: table(data$ethnic, data$devicetype) X-squared = 3.9678, df = 8, p-value = 0.86

```
data %>%
  ggplot(aes(x = ethnic, fill = devicetype)) +
  geom_bar(stat = "count") +
  facet_wrap(~devicetype, scales = "free_y") +
  guides(fill = "none") +
  labs(x = NULL, y = "Frequency") +
  theme_pubr() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



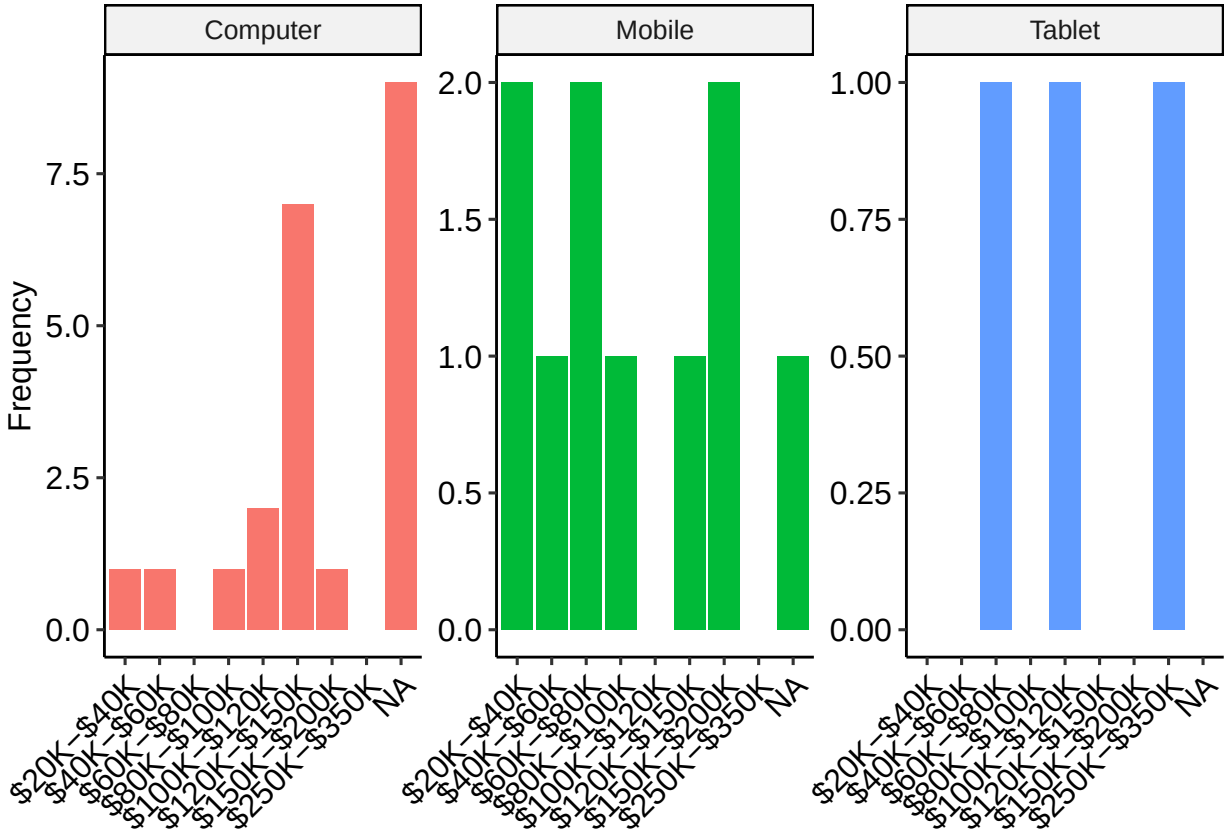
2.2.5 Household income

```
chisq.test(table(data$hhinc, data$devicetype))
```

```
Pearson's Chi-squared test
```

data: table(data$hhinc, data$devicetype) X-squared = NaN, df = 22, p-value = NA

```
data %>%
  ggplot(aes(x = hhinc, fill = devicetype)) +
  geom_bar(stat = "count") +
  facet_wrap(~devicetype, scales = "free_y") +
  guides(fill = "none") +
  labs(x = NULL, y = "Frequency") +
  theme_pubr() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



2.3 Time

How much time elapsed between assessments?

```
data = data %>%
  mutate(difference = as.numeric(start_date2-start_date))
summary(data$difference)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   11.96   12.00   12.49   13.26   14.17   17.48      13
```

How long did it take participants to complete the Time 1 survey?

```
summary(data$duration_in_seconds/60)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    6.65   13.98   17.55   20.56   25.08   44.27
```

How long did it take participants to complete the Time 2 survey?

```
summary(data$duration_in_seconds2/60)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   2.000   3.329   4.925   8.910   7.875  40.833      13
```

Table 2: Descriptives of responses to Block 1

format	mean	sd	median	N_responses	N_participants
Adjective Only	4.70	1.31	5	341	11
Am Adjective	4.52	1.32	5	279	9
Tend to be Adjective	4.63	1.22	5	248	8
Am someone who tends to be Adjective	4.73	1.32	5	217	7

2.4 Personality by block and format

2.4.1 Block 1

```
items_df %>%
  filter(block == "1") %>%
  group_by(format) %>%
  summarise(
    mean = mean(response),
    sd = sd(response),
    median = median(response),
    N_responses = n(),
    N_participants = length(unique(proid))
  ) %>%
  kable(booktabs = T, digits = c(0,2,2,0,0,0),
        caption = "Descriptives of responses to Block 1") %>%
  kable_styling()
```

```
items_df %>%
  filter(block == "1") %>%
  group_by(item, format) %>%
  summarise(
    mean = mean(response),
    sd = sd(response)
  ) %>%
  mutate(value = paste0(
    printnum(mean), " (", printnum(sd), ")"
  )) %>%
  select(-mean, -sd) %>%
  spread(format, value) %>%
  kable(booktabs = T) %>%
  kable_styling()
```

2.4.2 Block 2

```
items_df %>%
  filter(block == "2") %>%
  group_by(format) %>%
  summarise(
```

item	Adjective Only	Am Adjective	Tend to be Adjective	Am someone who tends to be Adjective
active	5.45 (1.21)	5.00 (0.87)	4.75 (1.04)	4.86 (1.77)
adventurous	4.82 (0.60)	5.00 (0.87)	4.50 (0.93)	4.57 (1.40)
broadminded	4.55 (1.29)	5.11 (1.05)	5.00 (0.93)	4.57 (0.79)
calm	5.45 (0.52)	4.67 (1.00)	4.75 (1.28)	5.00 (1.15)
careless	4.82 (1.40)	3.33 (1.94)	4.12 (1.25)	5.29 (1.11)
caring	5.36 (0.67)	4.78 (0.83)	5.00 (1.07)	5.43 (0.79)
cautious	4.91 (1.14)	4.67 (1.50)	5.00 (0.53)	4.43 (1.72)
creative	5.09 (0.94)	4.67 (1.22)	5.00 (0.93)	5.43 (0.79)
curious	4.64 (1.12)	4.89 (0.78)	4.62 (1.30)	4.57 (0.98)
friendly	5.55 (0.69)	5.33 (0.71)	5.25 (0.71)	5.29 (0.76)
hardworking	5.45 (0.69)	5.44 (0.73)	5.38 (1.06)	5.43 (0.53)
helpful	5.09 (0.94)	5.22 (0.83)	5.12 (0.64)	5.29 (1.11)
imaginative	5.00 (1.00)	5.22 (0.97)	5.25 (0.89)	5.57 (0.53)
impulsive	3.00 (1.18)	3.11 (1.17)	3.00 (1.51)	3.71 (1.80)
intelligent	5.09 (1.22)	4.78 (1.64)	5.25 (0.89)	5.43 (0.53)
lively	4.91 (0.83)	4.56 (0.73)	5.00 (1.20)	5.00 (1.41)
moody	3.91 (1.30)	2.89 (1.36)	3.38 (1.19)	3.86 (1.57)
nervous	4.09 (1.70)	3.56 (1.59)	3.88 (0.99)	4.00 (1.91)
organized	5.27 (0.79)	4.67 (1.12)	4.75 (1.28)	4.86 (1.07)
outgoing	4.91 (0.83)	4.89 (1.05)	4.38 (1.19)	4.71 (1.60)
reckless	4.18 (1.72)	4.11 (1.83)	4.38 (1.69)	5.14 (0.90)
responsible	5.64 (0.50)	5.22 (0.83)	5.50 (0.76)	5.43 (0.53)
selfdisciplined	4.91 (1.76)	5.11 (0.60)	4.75 (0.71)	5.57 (0.79)
softhearted	5.18 (0.98)	4.78 (0.97)	4.88 (0.99)	4.71 (1.80)
sophisticated	3.73 (1.27)	4.11 (1.05)	4.12 (1.73)	3.86 (1.07)
sympathetic	5.27 (1.01)	4.56 (1.01)	5.00 (0.76)	4.57 (1.27)
talkative	2.73 (1.01)	4.22 (1.72)	4.12 (1.46)	2.71 (1.38)
thorough	4.36 (1.29)	4.56 (1.13)	4.38 (1.06)	4.57 (0.98)
thrifty	3.64 (1.12)	3.89 (1.27)	4.75 (1.28)	3.43 (0.79)
warm	5.00 (1.48)	4.78 (0.83)	5.12 (0.99)	5.14 (0.69)
worrying	3.64 (1.43)	2.89 (1.69)	3.12 (1.25)	4.29 (1.80)

Table 3: Descriptives of responses to Block 2

format	mean	sd	median	N_responses	N_participants
Adjective Only	4.68	1.13	5	273	35
Am Adjective	4.57	1.42	5	279	35
Tend to be Adjective	4.67	1.32	5	269	35
Am someone who tends to be Adjective	4.61	1.30	5	264	35

```
    mean = mean(response),
    sd = sd(response),
    median = median(response),
    N_responses = n(),
    N_participants = length(unique(proid))
  ) %>%
  kable(booktabs = T, digits = c(0,2,2,0,0,0),
        caption = "Descriptives of responses to Block 2") %>%
  kable_styling()
```

```
items_df %>%
  filter(block == "2") %>%
  group_by(item, format) %>%
  summarise(
    mean = mean(response),
    sd = sd(response)
  ) %>%
  mutate(value = paste0(
    printnum(mean), " (", printnum(sd), ")"
  )) %>%
  select(-mean, -sd) %>%
  spread(format, value) %>%
  kable(booktabs = T) %>%
  kable_styling()
```

2.4.3 Block 3

```
items_df %>%
  filter(block == "2") %>%
  group_by(format) %>%
  summarise(
    mean = mean(response),
    sd = sd(response),
    median = median(response),
    N_responses = n(),
    N_participants = length(unique(proid))
  ) %>%
  kable(booktabs = T, digits = c(0,2,2,0,0,0),
        caption = "Descriptives of responses to Block 2") %>%
  kable_styling()
```

item	Adjective Only	Am Adjective	Tend to be Adjective	Am someone who tends to be Adjective
active	4.56 (1.67)	5.22 (1.30)	4.89 (1.27)	5.12 (0.35)
adventurous	5.00 (1.05)	4.86 (0.90)	5.22 (0.67)	4.11 (1.36)
broadminded	5.29 (0.49)	4.70 (0.95)	5.22 (0.67)	4.78 (0.67)
calm	5.10 (0.57)	5.25 (0.71)	4.88 (0.83)	4.89 (1.05)
careless	4.50 (1.43)	3.62 (2.13)	4.62 (1.51)	4.89 (1.05)
caring	5.11 (0.60)	5.50 (0.76)	4.75 (0.46)	5.20 (0.79)
cautious	4.67 (0.71)	4.80 (0.92)	5.00 (0.53)	4.62 (1.30)
creative	5.00 (0.58)	5.50 (0.71)	5.10 (0.57)	4.88 (1.36)
curious	4.25 (1.58)	4.30 (1.64)	4.67 (1.32)	5.38 (0.74)
friendly	5.25 (0.71)	5.00 (1.31)	5.33 (0.71)	5.00 (0.67)
hardworking	5.44 (0.73)	4.60 (1.17)	6.00 (0.00)	5.12 (0.35)
helpful	5.33 (0.50)	5.40 (0.70)	5.43 (0.53)	5.56 (0.73)
imaginative	4.70 (0.67)	5.22 (0.44)	5.12 (1.13)	5.25 (0.71)
impulsive	3.89 (1.62)	3.22 (1.79)	3.75 (1.67)	2.89 (0.93)
intelligent	5.12 (0.64)	5.00 (1.32)	5.44 (0.53)	5.33 (0.71)
lively	4.75 (1.04)	5.00 (1.00)	5.00 (1.00)	4.00 (1.94)
moody	3.73 (0.79)	4.00 (1.94)	3.00 (1.91)	4.00 (1.69)
nervous	4.12 (1.46)	4.10 (1.52)	3.11 (1.83)	3.75 (1.67)
organized	4.62 (1.51)	4.70 (1.42)	5.50 (0.76)	4.89 (0.60)
outgoing	5.12 (0.99)	4.40 (1.35)	4.12 (1.55)	4.78 (0.97)
reckless	4.56 (1.33)	4.89 (1.54)	3.80 (1.87)	4.86 (1.46)
responsible	5.30 (0.48)	5.22 (0.83)	5.00 (0.58)	4.78 (1.72)
selfdisciplined	5.00 (0.93)	4.67 (1.41)	4.80 (1.23)	5.00 (0.76)
softhearted	4.90 (0.88)	4.78 (1.30)	5.62 (0.52)	5.00 (1.07)
sophisticated	3.89 (1.17)	4.14 (1.68)	3.40 (1.35)	4.33 (0.87)
sympathetic	5.00 (0.87)	4.78 (0.83)	4.22 (0.97)	5.00 (0.76)
talkative	3.67 (1.00)	2.22 (1.20)	3.88 (1.81)	3.11 (1.83)
thorough	4.62 (1.19)	4.89 (1.05)	4.30 (1.25)	4.88 (0.83)
thrifty	4.00 (1.41)	3.78 (1.20)	4.33 (1.12)	3.56 (1.51)
warm	5.33 (0.50)	5.00 (0.94)	5.11 (0.60)	5.29 (0.76)
worrying	3.56 (1.24)	2.43 (1.62)	4.45 (1.69)	2.88 (1.64)

Table 4: Descriptives of responses to Block 2

format	mean	sd	median	N_responses	N_participants
Adjective Only	4.68	1.13	5	273	35
Am Adjective	4.57	1.42	5	279	35
Tend to be Adjective	4.67	1.32	5	269	35
Am someone who tends to be Adjective	4.61	1.30	5	264	35

item	Adjective Only	Am Adjective	Tend to be Adjective	Am someone who tends to be Adjective
active	4.56 (1.67)	5.22 (1.30)	4.89 (1.27)	5.12 (0.35)
adventurous	5.00 (1.05)	4.86 (0.90)	5.22 (0.67)	4.11 (1.36)
broadminded	5.29 (0.49)	4.70 (0.95)	5.22 (0.67)	4.78 (0.67)
calm	5.10 (0.57)	5.25 (0.71)	4.88 (0.83)	4.89 (1.05)
careless	4.50 (1.43)	3.62 (2.13)	4.62 (1.51)	4.89 (1.05)
caring	5.11 (0.60)	5.50 (0.76)	4.75 (0.46)	5.20 (0.79)
cautious	4.67 (0.71)	4.80 (0.92)	5.00 (0.53)	4.62 (1.30)
creative	5.00 (0.58)	5.50 (0.71)	5.10 (0.57)	4.88 (1.36)
curious	4.25 (1.58)	4.30 (1.64)	4.67 (1.32)	5.38 (0.74)
friendly	5.25 (0.71)	5.00 (1.31)	5.33 (0.71)	5.00 (0.67)
hardworking	5.44 (0.73)	4.60 (1.17)	6.00 (0.00)	5.12 (0.35)
helpful	5.33 (0.50)	5.40 (0.70)	5.43 (0.53)	5.56 (0.73)
imaginative	4.70 (0.67)	5.22 (0.44)	5.12 (1.13)	5.25 (0.71)
impulsive	3.89 (1.62)	3.22 (1.79)	3.75 (1.67)	2.89 (0.93)
intelligent	5.12 (0.64)	5.00 (1.32)	5.44 (0.53)	5.33 (0.71)
lively	4.75 (1.04)	5.00 (1.00)	5.00 (1.00)	4.00 (1.94)
moody	3.73 (0.79)	4.00 (1.94)	3.00 (1.91)	4.00 (1.69)
nervous	4.12 (1.46)	4.10 (1.52)	3.11 (1.83)	3.75 (1.67)
organized	4.62 (1.51)	4.70 (1.42)	5.50 (0.76)	4.89 (0.60)
outgoing	5.12 (0.99)	4.40 (1.35)	4.12 (1.55)	4.78 (0.97)
reckless	4.56 (1.33)	4.89 (1.54)	3.80 (1.87)	4.86 (1.46)
responsible	5.30 (0.48)	5.22 (0.83)	5.00 (0.58)	4.78 (1.72)
selfdisciplined	5.00 (0.93)	4.67 (1.41)	4.80 (1.23)	5.00 (0.76)
softhearted	4.90 (0.88)	4.78 (1.30)	5.62 (0.52)	5.00 (1.07)
sophisticated	3.89 (1.17)	4.14 (1.68)	3.40 (1.35)	4.33 (0.87)
sympathetic	5.00 (0.87)	4.78 (0.83)	4.22 (0.97)	5.00 (0.76)
talkative	3.67 (1.00)	2.22 (1.20)	3.88 (1.81)	3.11 (1.83)
thorough	4.62 (1.19)	4.89 (1.05)	4.30 (1.25)	4.88 (0.83)
thrifty	4.00 (1.41)	3.78 (1.20)	4.33 (1.12)	3.56 (1.51)
warm	5.33 (0.50)	5.00 (0.94)	5.11 (0.60)	5.29 (0.76)
worrying	3.56 (1.24)	2.43 (1.62)	4.45 (1.69)	2.88 (1.64)

```
items_df %>%
  filter(block == "2") %>%
  group_by(item, format) %>%
  summarise(
    mean = mean(response),
    sd = sd(response)
  ) %>%
  mutate(value = paste0(
    printnum(mean), " (", printnum(sd), ")"
  )) %>%
  select(-mean, -sd) %>%
  spread(format, value) %>%
  kable(booktabs = T) %>%
  kable_styling()
```

3 Does item format affect response?

The primary aims of this study are to evaluate the effects of item wording in online, self-report personality assessment. Specifically, we intend to consider the extent to which incremental wording changes may influence differences in the distributions of responses, response times, and psychometric properties of the items. These wording changes will include a progression from using (1) trait-descriptive adjectives by themselves, (2) with the linking verb "to be" (Am...), (3) with the additional verb "to tend" (Tend to be...), and (4) with the pronoun "someone" (Am someone who tends to be...).

Using a protocol that administers each adjective twice to the same participant (in different combinations of item format administered randomly across participants), we will use between-person analyses to compare responses using group-level data for the different formats.

These analyses will attempt to account for delayed_memory effects by collecting data on immediate and delayed recall (5 minutes and approximately two weeks) using a delayed_memory paradigm that was developed based on a similar recall task used in the HRS (Runge et al., 2015).

3.1 Effect of format (Block 1 data)

We used a multilevel model, nesting response within participant to account for dependence. Our primary predictor was format. Here, we use only Block 1 data; in other words, effects are largely between person, although each person contributes 31 unique data points to the analysis (one for each trait). We use the `anova` function to estimate the amount of variability in response due to format.

```
item_block1 = filter(items_df, block == "1")

mod.format_b1 = lmer(response~format + (1|proid),
                     data = item_block1)
anova(mod.format_b1)
```

```
## Type III Analysis of Variance Table with Satterthwaite's method
##        Sum Sq Mean Sq NumDF DenDF F value Pr(>F)
## format 1.3559 0.45198     3    31  0.3033 0.8228
```

When examining only Block 1 data, item format was unassociated with participants' responses to personality items ($F(3, 31.00) = 0.30, p = .823$).

```
plot_b1 = plot_model(mod.format_b1, type = "pred")

plot_b1$format +
  labs(x = NULL,
       y = "Average response",
       title = "Average responses by item formatting (Block 1 Data)") +
  theme_pubclean()
```

```
means_by_group = item_block1 %>%
  group_by(format) %>%
  summarise(m = mean(response),
            s = sd(response))

item_block1 %>%
  ggplot(aes(x = response, fill = format)) +
```

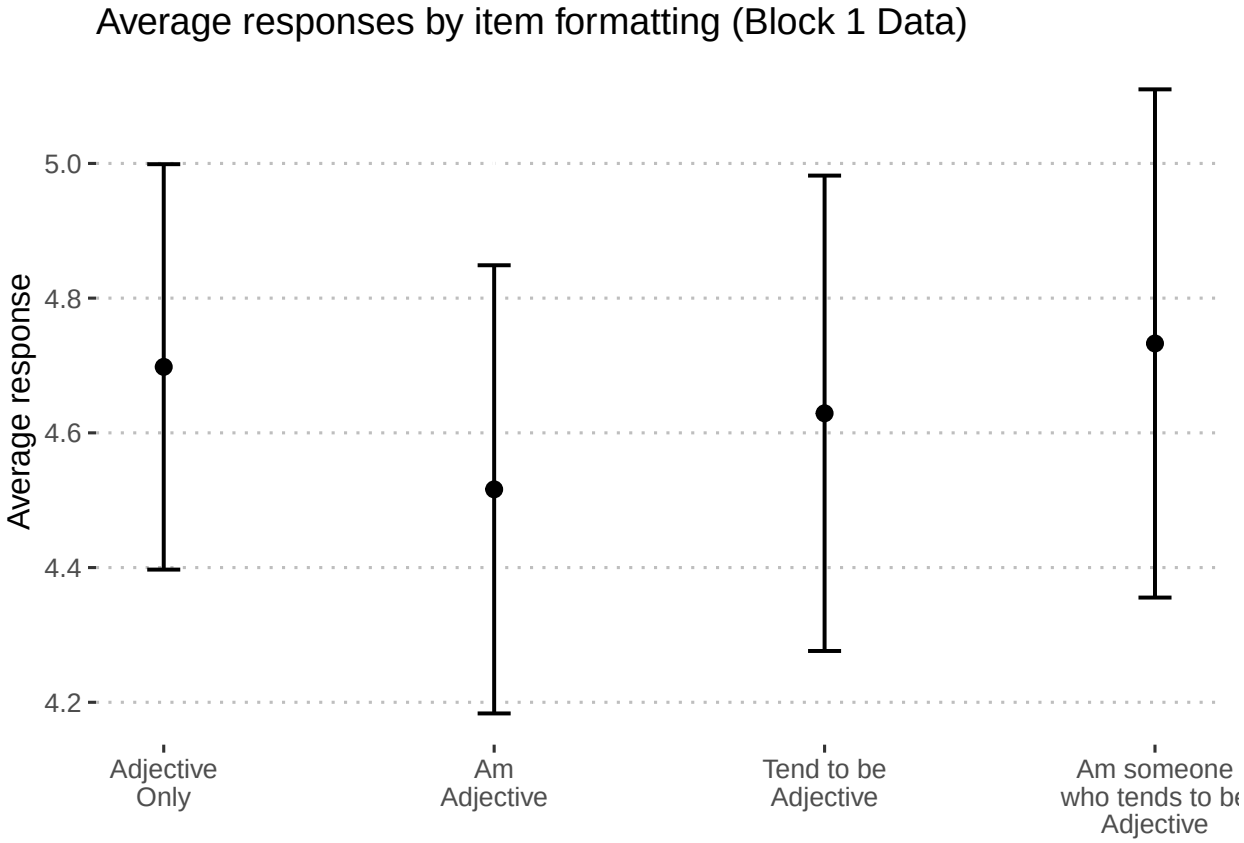


Figure 13: Predicted response on personality items by condition, using only Block 1 data.

```
geom_histogram(bins = 6, color = "white") +
geom_vline(aes(xintercept = m), data = means_by_group) +
geom_text(aes(x = 1,
              y = 125,
              label = paste("M =", round(m,2),
                            "\nSD =", round(s,2))),
          data = means_by_group,
          hjust =0,
          vjust = 1) +
facet_wrap(~format) +
guides(fill = "none") +
scale_x_continuous(breaks = 1:6) +
labs(y = "Number of particpants",
     title = "Distribution of responses by format (Block 1 data)") +
theme_pubr()
```

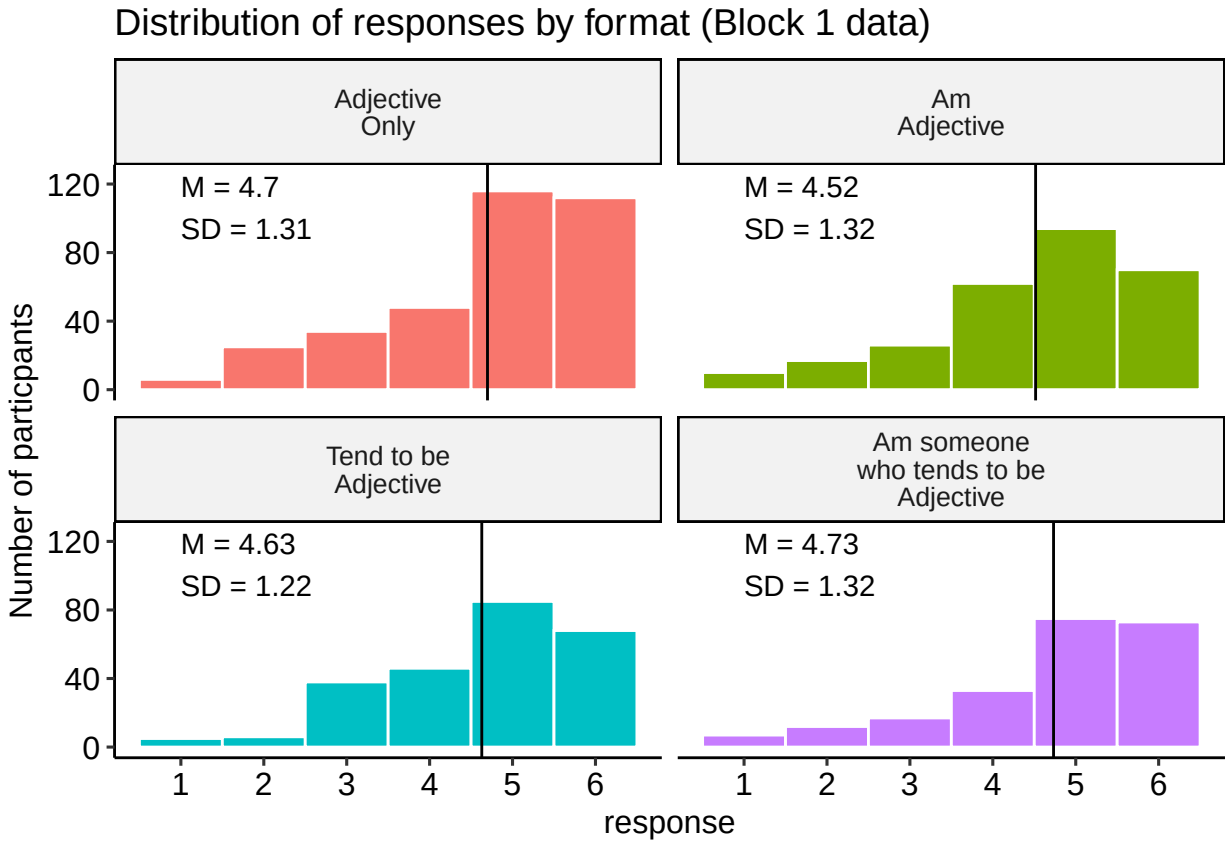


Figure 14: Distribution of responses by category, block 1 data only

3.1.1 One model for each adjective

We repeat this analysis separately for each trait. Because there is only one response per participant (when using only Block 1 data), we can drop the use of multilevel models and instead rely on a simple general linear model to test our hypothesis. Sepecfically, we test whether the proportion of variance attributable to item format is statistically significant.

```
mod_by_item_b1 = item_block1 %>%
  group_by(item) %>%
  nest() %>%
  mutate(mod = map(data, ~lm(response~format, data = .))) %>%
  mutate(aov = map(mod, anova))
```

We apply a Holm correction to the p-values extracted from these analyses, to adjust for the number of tests conducted. We present results in a table, which is organized by whether items were reverse-coded prior to analysis.

```
summary_by_item_b1 = mod_by_item_b1 %>%
  ungroup() %>%
  mutate(tidy = map(aov, broom::tidy)) %>%
  select(item, tidy) %>%
  unnest(cols = c(tidy)) %>%
  filter(term == "format") %>%
  mutate(reverse = case_when(
    item %in% reverse ~ "Y",
    TRUE ~ "N"
  )) %>%
  mutate(p.adj = p.adjust(p.value, method = "holm"))

summary_by_item_b1 %>%
  mutate(across( starts_with("p"), printp )) %>% # format p-values
  arrange(reverse, item) %>%
  select(item, reverse, sumsq, meansq, df, statistic, p.value, p.adj) %>%
  kable(digits = 2,
        booktabs = T,
        col.names = c("Item", "Reverse\nScored?", "SS", "MS", "df", "F", "raw", "adj"),
        caption = "Format effects on response by item (block 1 data only)") %>%
  kable_styling()
```

3.1.2 Pairwise t-tests for significant ANOVAs

When format was a significant predictor of response for an item (using the un-adjusted p-value here), we follow up with pairwise comparisons of format. Here we identify the items which meet this criteria. In the manuscript proper, we will only report the results for items in which format was significant, even after applying the Holm correction.

```
sig_item_b1 = summary_by_item_b1 %>%
  filter(p.value < .05)

sig_item_b1 = sig_item_b1$item
sig_item_b1
```

```
## [1] "talkative"
```

Then we create models for each adjective. We use the `emmeans` package to perform pairwise comparisons, again with a Holm correction on the p-values. We also plot the means and 95% confidence intervals of each mean.

This code will have to be changed after final data collection. It is not self-adapting!

Table 5: Format effects on response by item (block 1 data only)

Item	Reverse Scored?	SS	MS	df	F	raw	adj
active	N	2.80	0.93	3	0.61	.611	> .999
adventurous	N	1.34	0.45	3	0.50	.682	> .999
broadminded	N	2.27	0.76	3	0.66	.581	> .999
calm	N	3.77	1.26	3	1.29	.295	> .999
caring	N	2.47	0.82	3	1.17	.337	> .999
cautious	N	1.55	0.52	3	0.32	.814	> .999
creative	N	2.35	0.78	3	0.79	.507	> .999
curious	N	0.52	0.17	3	0.15	.927	> .999
friendly	N	0.52	0.17	3	0.34	.796	> .999
hardworking	N	0.03	0.01	3	0.02	.997	> .999
helpful	N	0.20	0.07	3	0.08	.968	> .999
imaginative	N	1.40	0.47	3	0.58	.630	> .999
intelligent	N	1.86	0.62	3	0.44	.725	> .999
lively	N	1.15	0.38	3	0.36	.782	> .999
organized	N	2.20	0.73	3	0.66	.583	> .999
outgoing	N	1.58	0.53	3	0.40	.755	> .999
responsible	N	0.87	0.29	3	0.65	.588	> .999
selfdisciplined	N	2.87	0.96	3	0.72	.545	> .999
softhearted	N	1.25	0.42	3	0.30	.828	> .999
sophisticated	N	1.08	0.36	3	0.21	.887	> .999
sympathetic	N	3.42	1.14	3	1.10	.363	> .999
talkative	N	18.53	6.18	3	3.19	.037	> .999
thorough	N	0.33	0.11	3	0.08	.968	> .999
thrifty	N	8.09	2.70	3	2.06	.126	> .999
warm	N	0.71	0.24	3	0.20	.897	> .999
careless	Y	18.23	6.08	3	2.77	.058	> .999
impulsive	Y	2.65	0.88	3	0.45	.716	> .999
moody	Y	6.21	2.07	3	1.14	.350	> .999
nervous	Y	1.54	0.51	3	0.20	.893	> .999
reckless	Y	5.14	1.71	3	0.65	.587	> .999
worrying	Y	8.95	2.98	3	1.25	.307	> .999

Table 6: Differences in response to Talkative by format (Block 1 data only)

Contrast	Difference in means	SE	df	t	p
Adjective Only - Am Adjective	-1.49	0.63	31	-2.39	.139
Adjective Only - Tend to be Adjective	-1.40	0.65	31	-2.16	.192
Adjective Only - Am someone who tends to be Adjective	0.01	0.67	31	0.02	> .999
Am Adjective - Tend to be Adjective	0.10	0.68	31	0.14	> .999
Am Adjective - Am someone who tends to be Adjective	1.51	0.70	31	2.15	.192
Tend to be Adjective - Am someone who tends to be Adjective	1.41	0.72	31	1.96	.192

3.1.3 Talkative

```
talkative_model_b1 = item_block1 %>%
  filter(item == "talkative") %>%
  lm(response~format, data = .)

talkative_em_b1 = emmeans(talkative_model_b1, "format")
pairs(talkative_em_b1, adjust = "holm") %>%
  as_tibble() %>%
  mutate(across( starts_with("p"), printp )) %>% # format p-values
  kable(booktabs = T,
        digits = 2,
        caption = "Differences in response to Talkative by format (Block 1 data only)",
        col.names = c("Contrast", "Difference in means", "SE", "df", "t", "p")) %>%
  kable_styling()
```

```
plot_model(talkative_model_b1, type = "pred", terms = c("format"))
```

3.2 Effect of format (Block 1 and 2)

We again test whether format is a significant predictor of response. However, here we use data from both Blocks 1 and 2. As a reminder, all participants were presented with all four formats during Block 2. We expect this model to have greater power than the previous model, due to both increased sample size (twice as many data points) and because participants now provide data to all four formats, instead of only one (i.e., a within-person analysis).

```
items_12 = items_df %>% filter(block %in% c("1","2"))
```

```
mod.format_b2 = lmer(response~format + (1|proid),
                     data = items_12)
anova(mod.format_b2)
```

```
## Type III Analysis of Variance Table with Satterthwaite's method
##        Sum Sq Mean Sq NumDF  DenDF F value Pr(>F)
## format 4.1445  1.3815     3 2104.9  0.9193 0.4307
```

When examining both Block 1 and Block 2 data, item format was unassociated with participants' responses to personality items ($F(3, 2, 104.92) = 0.92, p = .431$).

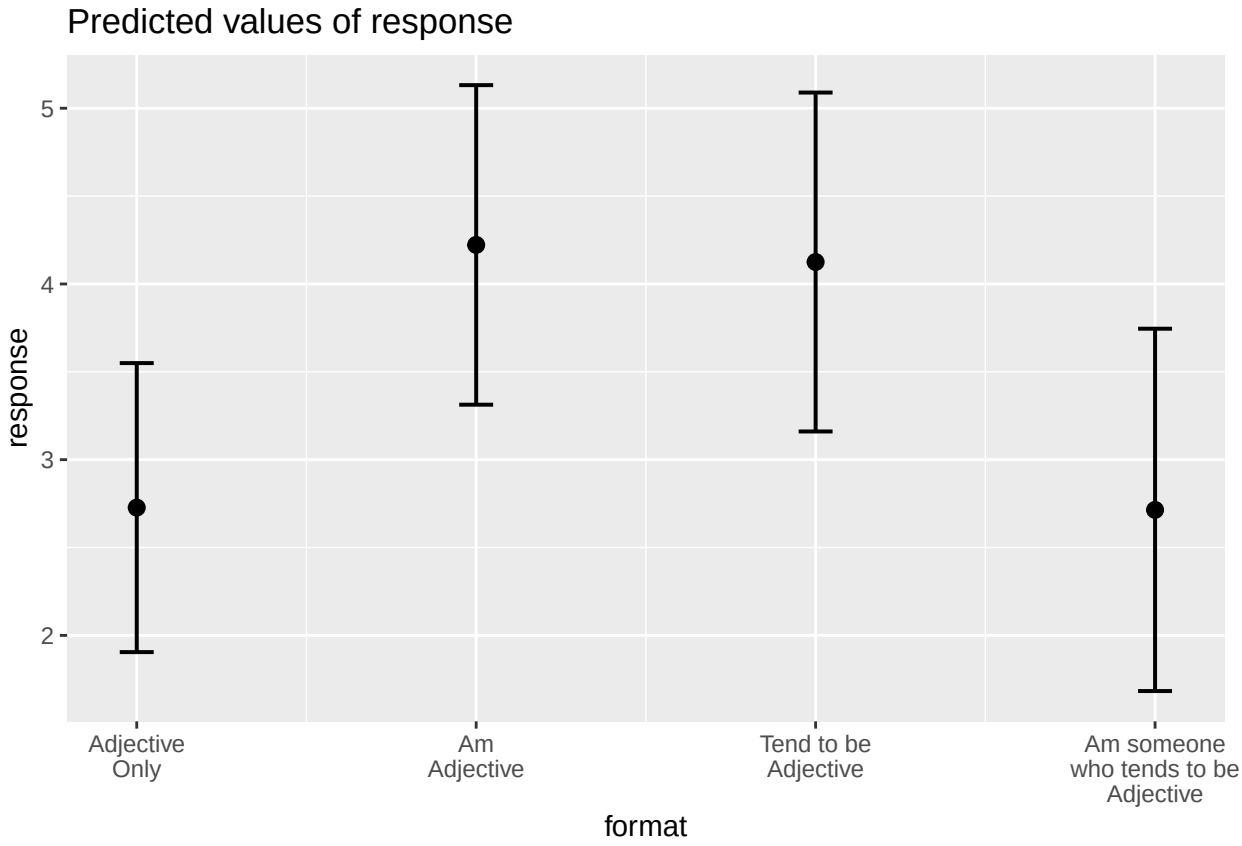


Figure 15: Average response to “talkative” by format (block 1 data only)

```
plot_b2 = plot_model(mod.format_b2, type = "pred")

plot_b2$format +
  labs(x = NULL,
       y = "Average response",
       title = "Average responses by item formatting (Block 1 and Block 2)") +
  theme_pubclean()
```

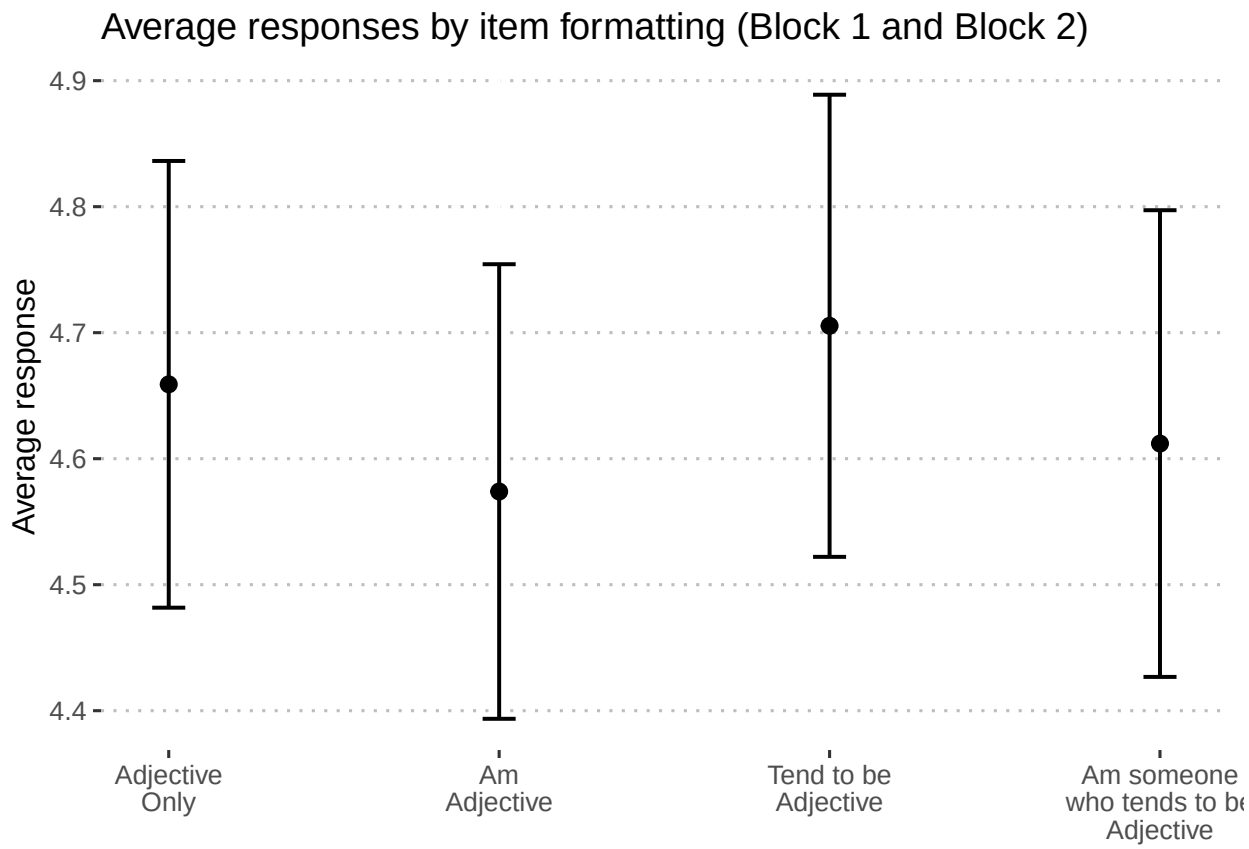


Figure 16: Predicted response on personality items by condition, using only Block 1 data.

```
means_by_group = items_12 %>%
  group_by(format) %>%
  summarise(m = mean(response),
            s = sd(response))

items_12 %>%
  ggplot(aes(x = response, fill = format)) +
  geom_histogram(bins = 6, color = "white") +
  geom_vline(aes(xintercept = m), data = means_by_group) +
  geom_text(aes(x = 1,
                y = 200,
                label = paste("M =", round(m,2),
                              "\nSD =", round(s,2))),
            data = means_by_group,
```

```
            hjust =0,
            vjust = 1) +
  facet_wrap(~format) +
  guides(fill = "none") +
  scale_x_continuous(breaks = 1:6) +
  labs(y = "Number of particpants",
       title = "Distribution of responses by format (Block 1 and Block 2)") +
  theme_pubr()
```

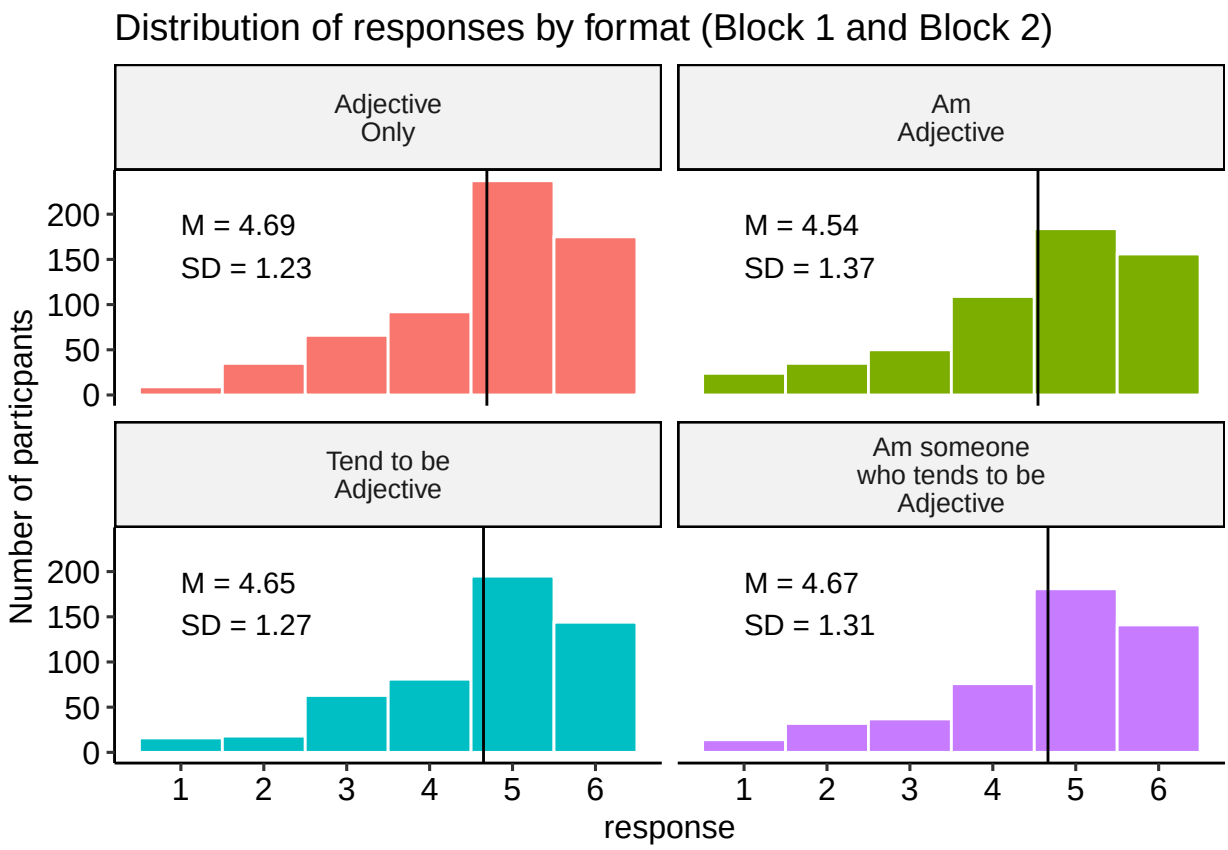


Figure 17: Distribution of responses by category, block 1 and block 2

3.2.1 One model for each adjective

We can also repeat this analysis separately for each trait. We use the `anova` function to estimate the variability due to format and print the corresponding F-test.

```
mod_by_item_b2 = items_12 %>%
  group_by(item) %>%
  nest() %>%
  mutate(mod = map(data, ~lmer(response~format + (1|proid), data = .))) %>%
  mutate(aov = map(mod, anova))
```

To present these results, we use the `tidy` function to summarise the findings and extract just the F-test associated with the format variable. We calculate adjusted p-values using a Holm correction. We also create

a column that indicates whether the item was reverse-scored; we use this to sort the table, in case a pattern emerges.

```
summary_by_item_b2 = mod_by_item_b2 %>%
  ungroup() %>%
  mutate(tidy = map(aov, broom::tidy)) %>%
  select(item, tidy) %>%
  unnest(cols = c(tidy)) %>%
  filter(term == "format") %>%
  mutate(reverse = case_when(
    item %in% reverse ~ "Y",
    TRUE ~ "N"
  )) %>%
  mutate(p.adj = p.adjust(p.value, method = "holm"))

summary_by_item_b2 %>%
  mutate(across( starts_with("p"), printp )) %>% # format p-values
  arrange(reverse, item) %>%
  select(item, reverse, sumsq, meansq, NumDF, DenDF, statistic, p.value, p.adj) %>%
  kable(digits = 2,
        col.names = c("Item", "Reverse\nScored?", "SS", "MS", "df1", "df2", "F", "raw", "adj"),
        booktabs = T, caption = "Format effects on response by item (block 1 data only)") %>%
  kable_styling() %>%
    add_header_above(c(" " = 7, "p-value" = 2))
```

3.2.2 Pairwise t-tests for significant ANOVAs

Here we identify the specific items with significant differences.

```
sig_item_b2 = summary_by_item_b2 %>%
  filter(p.value < .05)

sig_item_b2 = sig_item_b2$item
sig_item_b2
```

```
## [1] "careless" "thrifty"
```

Then we create models for each adjective. We use the `emmeans` package to perform pairwise comparisons, again with a Holm correction on the p-values. We also plot the means and 95% confidence intervals of each mean.

This code will have to be changed after final data collection. It is not self-adapting!

3.2.3 Careless

```
careless_model_b2 = items_12 %>%
  filter(item == "careless") %>%
  lmer(response~format + (1|proid),
                               data = .)
```

Table 7: Format effects on response by item (block 1 data only)

							p-value	
Item	Reverse Scored?	SS	MS	df1	df2	F	raw	adj
active	N	0.76	0.25	3	40.00	1.01	.398	> .999
adventurous	N	1.51	0.50	3	48.65	1.09	.361	> .999
broadminded	N	1.54	0.51	3	64.49	0.76	.521	> .999
calm	N	0.13	0.04	3	45.02	0.13	.940	> .999
caring	N	1.16	0.39	3	52.66	1.28	.291	> .999
cautious	N	2.30	0.77	3	54.97	0.98	.407	> .999
creative	N	0.14	0.05	3	49.22	0.13	.943	> .999
curious	N	2.35	0.78	3	52.00	1.05	.377	> .999
friendly	N	0.94	0.31	3	46.85	1.45	.239	> .999
hardworking	N	2.10	0.70	3	52.03	2.06	.117	> .999
helpful	N	0.90	0.30	3	60.54	0.82	.488	> .999
imaginative	N	1.03	0.34	3	55.07	0.86	.465	> .999
intelligent	N	1.72	0.57	3	51.16	1.15	.340	> .999
lively	N	1.08	0.36	3	42.77	0.89	.456	> .999
organized	N	0.22	0.07	3	37.39	0.41	.750	> .999
outgoing	N	0.71	0.24	3	39.34	0.83	.484	> .999
responsible	N	1.51	0.50	3	56.50	0.82	.487	> .999
selfdisciplined	N	0.32	0.11	3	41.84	0.26	.853	> .999
softhearted	N	1.95	0.65	3	57.46	0.74	.531	> .999
sophisticated	N	0.05	0.02	3	44.04	0.04	.989	> .999
sympathetic	N	0.49	0.16	3	51.48	0.51	.676	> .999
talkative	N	6.36	2.12	3	43.86	1.95	.135	> .999
thorough	N	2.23	0.74	3	40.16	2.72	.057	> .999
thrifty	N	6.39	2.13	3	51.69	3.30	.027	.849
warm	N	0.10	0.03	3	52.22	0.08	.968	> .999
careless	Y	8.29	2.76	3	53.32	2.84	.047	> .999
impulsive	Y	1.38	0.46	3	41.41	0.85	.473	> .999
moody	Y	1.21	0.40	3	42.77	0.69	.566	> .999
nervous	Y	1.35	0.45	3	46.22	0.45	.716	> .999
reckless	Y	1.14	0.38	3	48.60	0.36	.782	> .999
worrying	Y	1.72	0.57	3	41.67	0.63	.602	> .999

Table 8: Differences in response to Careless by format (Block 1 and Block 2)

Contrast	Difference in means	SE	df	t	p
Adjective Only - Am Adjective	0.82	0.45	58.04	1.83	.359
Adjective Only - Tend to be Adjective	0.14	0.44	54.32	0.32	.749
Adjective Only - Am someone who tends to be Adjective	-0.44	0.44	54.32	-1.01	.633
Am Adjective - Tend to be Adjective	-0.68	0.45	51.43	-1.52	.542
Am Adjective - Am someone who tends to be Adjective	-1.26	0.45	51.43	-2.82	.041
Tend to be Adjective - Am someone who tends to be Adjective	-0.58	0.45	49.48	-1.31	.591

```
careless_em_b2 = emmeans(careless_model_b2, "format")
pairs(careless_em_b2, adjust = "holm") %>%
  as_tibble() %>%
  mutate(across( starts_with("p"), printp )) %>% # format p-values
  kable(booktabs = T,
        digits = 2,
        caption = "Differences in response to Careless by format (Block 1 and Block 2)",
        col.names = c("Contrast", "Difference in means", "SE", "df", "t", "p")) %>%
  kable_styling()
```

```
plot_model(careless_model_b2, type = "pred", terms = c("format"))
```

3.2.4 Thrifty

```
thrifty_model_b2 = items_12 %>%
  filter(item == "thrifty") %>%
  lmer(response~format + (1|proid),
                                data = .)

thrifty_em_b2 = emmeans(thrifty_model_b2, "format")
pairs(thrifty_em_b2, adjust = "holm") %>%
  as_tibble() %>%
  mutate(across( starts_with("p"), printp )) %>% # format p-values
  kable(booktabs = T,
        digits = 2,
        caption = "Differences in response to Thifty by format (Block 1 and Block 2)",
        col.names = c("Contrast", "Difference in means", "SE", "df", "t", "p")) %>%
  kable_styling()
```

```
plot_model(thrifty_model_b2, type = "pred", terms = c("format"))
```

3.3 Account for memory effects (Blocks 1 and 2)

One limitation of the two-blocks model is that format effects may depend on a person's memory. For example, suppose that participants, in general, are more likely to respond with a 6 to items containing "tend

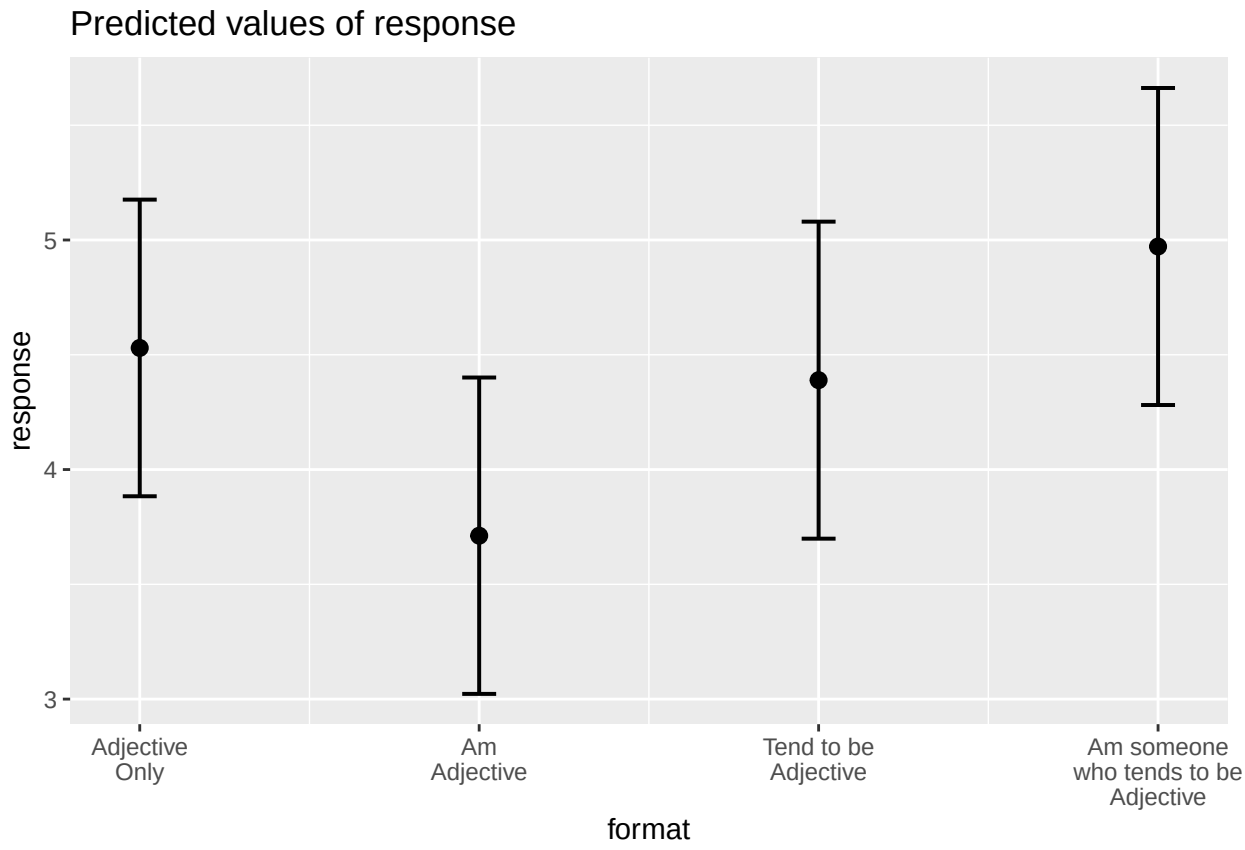


Figure 18: Average response to "careless" by format (Block 1 and Block 2)

Table 9: Differences in response to Thifty by format (Block 1 and Block 2)

Contrast	Difference in means	SE	df	t	p
Adjective Only - Am Adjective	-0.02	0.36	56.66	-0.04	> .999
Adjective Only - Tend to be Adjective	-0.92	0.36	55.47	-2.54	.083
Adjective Only - Am someone who tends to be Adjective	-0.09	0.35	52.28	-0.26	> .999
Am Adjective - Tend to be Adjective	-0.90	0.37	56.24	-2.45	.084
Am Adjective - Am someone who tends to be Adjective	-0.08	0.37	54.54	-0.21	> .999
Tend to be Adjective - Am someone who tends to be Adjective	0.83	0.33	43.22	2.49	.084

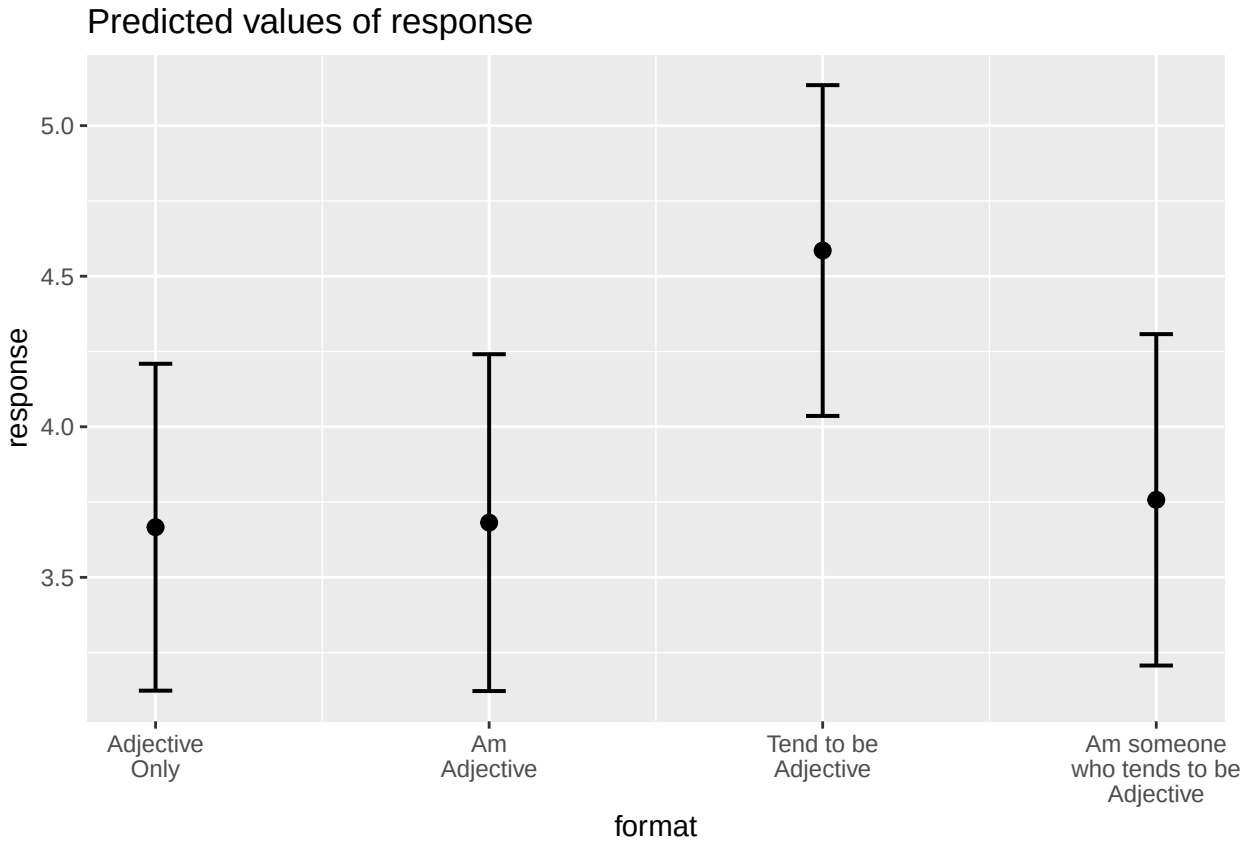


Figure 19: Average response to "thrifty" by format (Block 1 and Block 2)

to" (e.g., "tend to be outgoing") than to items that only start with "am" (e.g., "am outgoing"). However, if a participant remembers that on the first presentation of the item they selected 4, they may be more likely choose 4 again to appear consistent. This example posits that memory moderates format's effect on response. We model this possibility using participant's delayed memory scores, or their recall score 10 minutes after seeing the list of words.

```
mod.format_mem = lmer(response~format*delayed_memory + (1|proid),
                  data = items_12)
anova(mod.format_mem)
```

```
## Type III Analysis of Variance Table with Satterthwaite's method
##                        Sum Sq Mean Sq NumDF   DenDF F value Pr(>F)
## format                 4.8144 1.60479     3 2117.22  1.0675 0.3617
## delayed_memory         2.4823 2.48233     1   33.18  1.6513 0.2077
## format:delayed_memory 2.9060 0.96867     3 2118.52  0.6444 0.5865
```

```
summary(mod.format_mem)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: response ~ format * delayed_memory + (1 | proid)
##    Data: items_12
##
## REML criterion at convergence: 7144.7
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.4343 -0.4565  0.2507  0.6893  1.7916
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  proid    (Intercept) 0.1804   0.4248
##  Residual             1.5033   1.2261
## Number of obs: 2170, groups:  proid, 35
##
## Fixed effects:
##                                                                 Estimate
## (Intercept)                                                      4.58932
## formatAm\nAdjective                                             -0.27329
## formatTend to be\nAdjective                                     -0.07698
## formatAm someone\nwho tends to be\nAdjective                    -0.14891
## delayed_memory                                                   0.01320
## formatAm\nAdjective:delayed_memory                               0.03632
## formatTend to be\nAdjective:delayed_memory                       0.02519
## formatAm someone\nwho tends to be\nAdjective:delayed_memory      0.02048
##                                                                Std. Error
## (Intercept)                                                       0.18198
## formatAm\nAdjective                                               0.15843
## formatTend to be\nAdjective                                       0.16694
## formatAm someone\nwho tends to be\nAdjective                      0.17237
## delayed_memory                                                    0.03108
## formatAm\nAdjective:delayed_memory                                0.02641
## formatTend to be\nAdjective:delayed_memory                        0.02944
```

```
## formatAm someone\nwho tends to be\nAdjective:delayed_memory    0.02956
##                                                                      df t value
## (Intercept)                                                       62.17156  25.219
## formatAm\nAdjective                                             2128.91591  -1.725
## formatTend to be\nAdjective                                     2092.17673  -0.461
## formatAm someone\nwho tends to be\nAdjective                    2139.15284  -0.864
## delayed_memory                                                    63.52491   0.425
## formatAm\nAdjective:delayed_memory                              2116.48721   1.375
## formatTend to be\nAdjective:delayed_memory                      2070.60879   0.856
## formatAm someone\nwho tends to be\nAdjective:delayed_memory 2147.54209   0.693
##                                                                 Pr(>|t|)
## (Intercept)                                                       <2e-16 ***
## formatAm\nAdjective                                               0.0847 .
## formatTend to be\nAdjective                                       0.6447
## formatAm someone\nwho tends to be\nAdjective                     0.3878
## delayed_memory                                                    0.6725
## formatAm\nAdjective:delayed_memory                                0.1692
## formatTend to be\nAdjective:delayed_memory                        0.3922
## formatAm someone\nwho tends to be\nAdjective:delayed_memory       0.4885
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) frmtAA frTtbA frAswttbA dlyd_m frAA:_ fTtbA:
## frmtAmAdjct -0.439
## frmtTndtbAd -0.433  0.478
## frmtAswttbA -0.407  0.455  0.470
## delayd_mmry -0.869  0.387  0.381  0.355
## frmtAAdjc:_  0.397 -0.859 -0.433 -0.413    -0.465
## frmtTtbAd:_  0.370 -0.409 -0.865 -0.398    -0.431  0.495
## frAswttbA:_  0.354 -0.400 -0.410 -0.873    -0.410  0.484  0.460
```

When examining both Block 1 and Block 2 data, memory did not have a main effect on participant responses ($F(1, 33.18) = 1.65, p = .208$) and did not moderate differences between formats ($F(3, 2, 118.52) = 0.64, p = .586$).

```
plot_model(mod.format_mem,
           type = "pred",
           term = c("format", "delayed_memory[meansd]")) +
  geom_line() +
  labs(x = NULL,
       y = "Average response") +
  scale_color_discrete("Memory", labels = c("-1SD", "Mean", "+1SD"))+
  theme_pubclean()
```

3.3.1 One model for each adjective

Again, we test this model within each trait adjective, to determine whether the moderating effect of memory is stronger for any particular trait(s).

```
mod_by_item_mem = items_12 %>%
  group_by(item) %>%
```

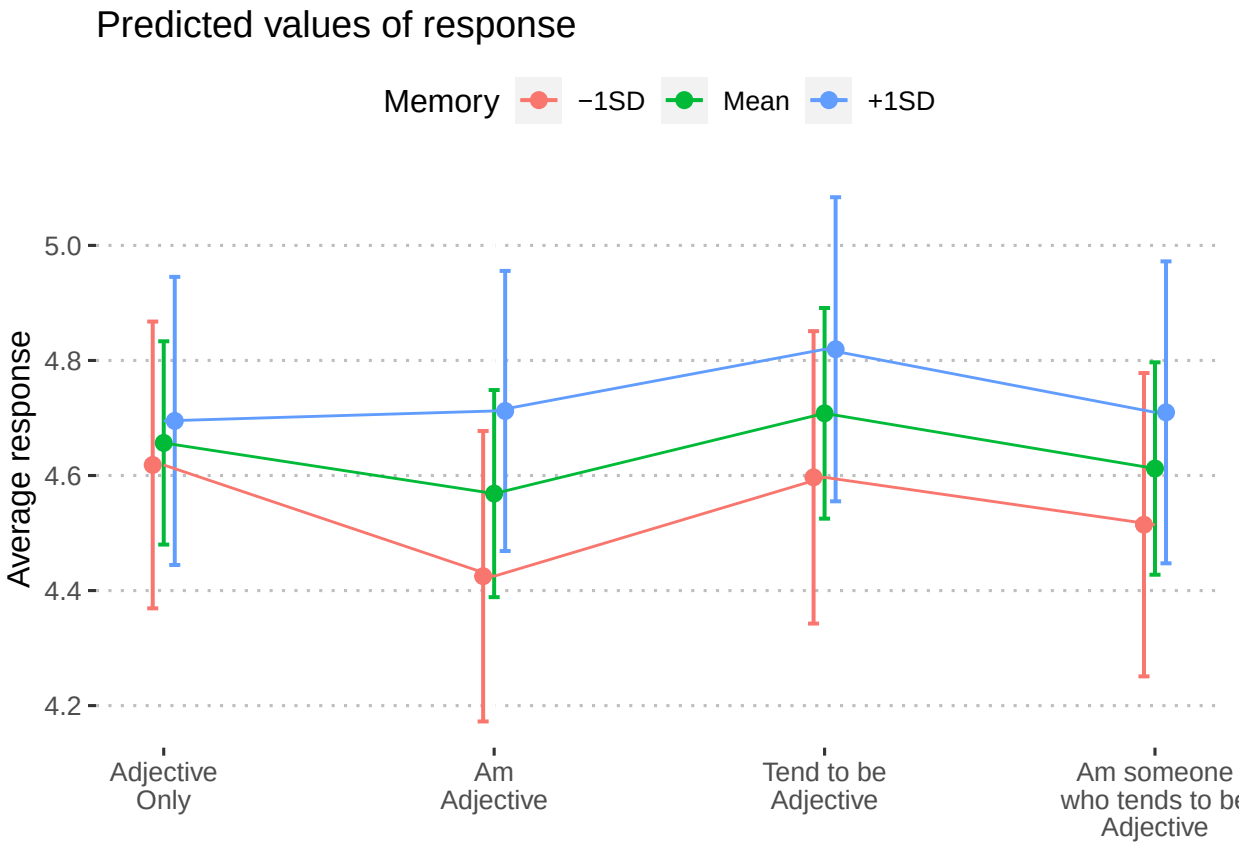


Figure 20: Predicted response on personality items by condition after controlling for delayed_memory.

```
  nest() %>%
  mutate(mod = map(data, ~lm(response~format*delayed_memory, data = .))) %>%
  mutate(aov = map(mod, anova))

summary_by_item_mem = mod_by_item_mem %>%
  ungroup() %>%
  mutate(tidy = map(aov, broom::tidy)) %>%
  select(item, tidy) %>%
  unnest(cols = c(tidy)) %>%
  filter(term == "format:delayed_memory") %>%
  mutate(reverse = case_when(
    item %in% reverse ~ "Y",
    TRUE ~ "N"
  )) %>%
  mutate(p.adj = p.adjust(p.value, method = "holm"))

summary_by_item_mem %>%
  mutate(across( starts_with("p"), printp )) %>% # format p-values
  arrange(reverse, item) %>%
  select(item, reverse, sumsq, meansq, df, statistic, p.value, p.adj) %>%
  kable(digits = 2,
        col.names = c("Item", "Reverse\nScored?", "SS", "MS", "df", "F", "raw", "adj"),
        booktabs = T) %>%
  kable_styling() %>%
  add_header_above(c(" " = 6, "p-value" = 2))
```

3.3.2 Pairwise t-tests for significant ANOVAs

Here we identify the specific items with significant differences.

```
sig_item_mem = summary_by_item_mem %>%
  filter(p.value < .05)

sig_item_mem = sig_item_mem$item
sig_item_mem
```

```
## character(0)
```

3.4 Inclusion of "I" (Block 1 and Block 3)

Finally, we test whether the inclusion of the word "I" impacts item response (e.g. "I am outgoing"). We used two multilevel models, nesting response within participant to account for dependence. Our primary predictors are format and also the presence of the word "I". Because we have no specific rationale for how or why "I" would impact responses, we test both the partialled main effect of "I" as well as the interaction with format. Here, we use data from Blocks 1 and 3.

```
items_13 = items_df %>%
  filter(block %in% c("1","3")) %>%
  filter(condition != "A") %>%
  filter(time2 == "yes")
```

						p-value	
Item	Reverse Scored?	SS	MS	df	F	raw	adj
active	N	0.29	0.10	3	0.06	.980	> .999
adventurous	N	1.17	0.39	3	0.39	.759	> .999
broadminded	N	0.71	0.24	3	0.28	.843	> .999
calm	N	0.15	0.05	3	0.06	.980	> .999
caring	N	0.16	0.05	3	0.08	.968	> .999
cautious	N	0.42	0.14	3	0.12	.948	> .999
creative	N	1.12	0.37	3	0.44	.727	> .999
curious	N	1.28	0.43	3	0.27	.843	> .999
friendly	N	1.83	0.61	3	1.01	.394	> .999
hardworking	N	0.74	0.25	3	0.37	.771	> .999
helpful	N	1.31	0.44	3	0.73	.536	> .999
imaginative	N	0.65	0.22	3	0.35	.790	> .999
intelligent	N	2.94	0.98	3	0.95	.421	> .999
lively	N	1.21	0.40	3	0.28	.840	> .999
organized	N	5.06	1.69	3	1.53	.216	> .999
outgoing	N	1.72	0.57	3	0.40	.753	> .999
responsible	N	2.53	0.84	3	1.11	.353	> .999
selfdisciplined	N	2.29	0.76	3	0.60	.616	> .999
softhearted	N	0.46	0.15	3	0.13	.943	> .999
sophisticated	N	4.21	1.40	3	0.89	.451	> .999
sympathetic	N	1.11	0.37	3	0.40	.754	> .999
talkative	N	8.53	2.84	3	1.23	.306	> .999
thorough	N	3.87	1.29	3	1.13	.344	> .999
thrifty	N	0.61	0.20	3	0.13	.940	> .999
warm	N	1.65	0.55	3	0.69	.560	> .999
careless	Y	3.74	1.25	3	0.55	.653	> .999
impulsive	Y	7.37	2.46	3	1.13	.343	> .999
moody	Y	10.15	3.38	3	1.76	.165	> .999
nervous	Y	4.40	1.47	3	0.57	.638	> .999
reckless	Y	7.65	2.55	3	1.01	.394	> .999
worrying	Y	1.77	0.59	3	0.22	.880	> .999

```
mod.format_b3_1 = lmer(response~format + i + (1|proid),
                  data = items_13)
anova(mod.format_b3_1)
```

```
## Type III Analysis of Variance Table with Satterthwaite's method
##        Sum Sq Mean Sq NumDF  DenDF F value Pr(>F)
## format 2.4558  1.2279     2  13.00  0.8151 0.4640
## i      3.4314  3.4314     1 979.33  2.2778 0.1316
```

```
mod.format_b3_2 = lmer(response~format*i + (1|proid),
                  data = items_13)
anova(mod.format_b3_2)
```

```
## Type III Analysis of Variance Table with Satterthwaite's method
##          Sum Sq Mean Sq NumDF  DenDF F value Pr(>F)
## format   3.3194 1.65969     2  14.78  1.1006 0.3584
## i        1.7404 1.74044     1 976.53  1.1542 0.2829
## format:i 1.5669 0.78345     2 976.81  0.5195 0.5950
```

```
plot_b3 = plot_model(mod.format_b3_2, type = "pred", terms = c("format", "i"))
plot_b3 +
  geom_line() +
  labs(x = NULL,
       y = "Average response",
       title = "Average responses by item formatting (Block 1 and Block 2)") +
  theme_pubclean()
```

```
means_by_group = items_13 %>%
  group_by(format, i) %>%
  summarise(m = mean(response),
            s = sd(response))

items_13 %>%
  ggplot(aes(x = response, fill = i)) +
  geom_histogram(bins = 6, color = "white") +
  geom_vline(aes(xintercept = m), data = means_by_group) +
  geom_text(aes(x = 1,
                y = 100,
                label = paste("M =", round(m,2),
                              "\nSD =", round(s,2))),
            data = means_by_group,
            hjust =0,
            vjust = 1) +
  facet_grid(i~format, scales = "free") +
  guides(fill = "none") +
  scale_x_continuous(breaks = 1:6) +
  labs(y = "Number of particpants",
       title = "Distribution of responses by format (Block 1 and Block 2)") +
  theme_pubr()
```

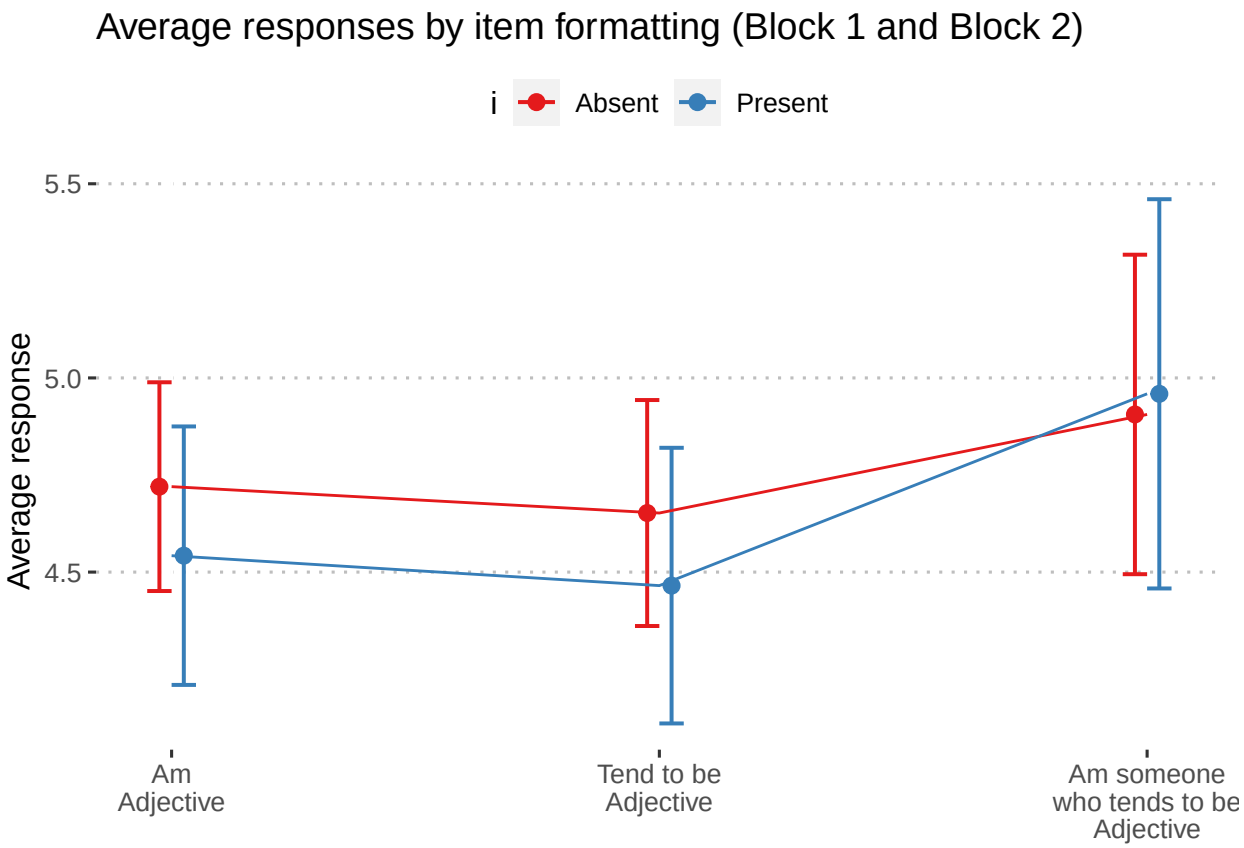


Figure 21: Predicted response on personality items by condition, using only Block 1 data.

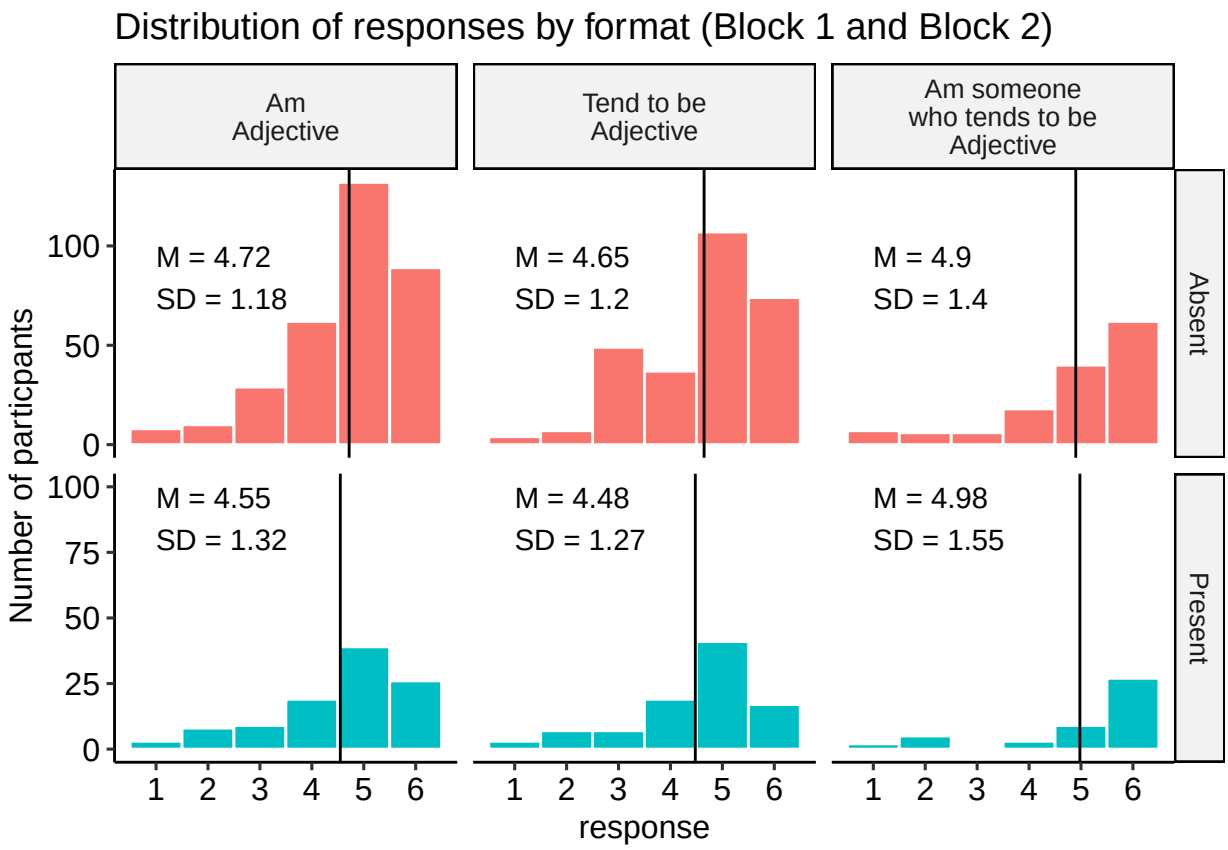


Figure 22: Distribution of responses by category, block 1 and block 2

3.4.1 One model for each adjective

```
mod_by_item_i = items_13 %>%
  group_by(item) %>%
  nest() %>%
  mutate(mod = map(data, ~lmer(response~format + i + (1|proid), data = .))) %>%
  mutate(aov = map(mod, anova))

summary_by_item_i = mod_by_item_i %>%
  ungroup() %>%
  mutate(tidy = map(aov, broom::tidy)) %>%
  select(item, tidy) %>%
  unnest(cols = c(tidy)) %>%
  filter(term == "i") %>%
  mutate(reverse = case_when(
    item %in% reverse ~ "Y",
    TRUE ~ "N"
  )) %>%
  mutate(p.adj = p.adjust(p.value, method = "holm"))

summary_by_item_i %>%
  mutate(across( starts_with("p"), printp )) %>% # format p-values
  arrange(reverse, item) %>%
  select(item, reverse, sumsq, meansq, NumDF, DenDF, statistic, p.value, p.adj) %>%
  kable(digits = 2,
        col.names = c("Item", "Reverse\nScored?", "SS", "MS", "df1", "df2", "F", "raw", "adj"),
        booktabs = T, caption = "Effect of \"I\" (block 1 and 3 data)") %>%
  kable_styling() %>%
    add_header_above(c(" " = 7, "p-value" = 2))
```

```
mod_by_item_i2 = items_13 %>%
  group_by(item) %>%
  nest() %>%
  mutate(mod = map(data, ~lmer(response~format*i + (1|proid), data = .))) %>%
  mutate(aov = map(mod, anova))

summary_by_item_i2 = mod_by_item_i2 %>%
  ungroup() %>%
  mutate(tidy = map(aov, broom::tidy)) %>%
  select(item, tidy) %>%
  unnest(cols = c(tidy)) %>%
  filter(term == "format:i") %>%
  mutate(reverse = case_when(
    item %in% reverse ~ "Y",
    TRUE ~ "N"
  )) %>%
  mutate(p.adj = p.adjust(p.value, method = "holm"))

summary_by_item_i2 %>%
  mutate(across( starts_with("p"), printp )) %>% # format p-values
  arrange(reverse, item) %>%
  select(item, reverse, sumsq, meansq, NumDF, DenDF, statistic, p.value, p.adj) %>%
```

Table 10: Effect of "I" (block 1 and 3 data)

							p-value	
Item	Reverse Scored?	SS	MS	df1	df2	F	raw	adj
active	N	0.79	0.79	1	17.53	2.15	.161	> .999
adventurous	N	0.69	0.69	1	28.00	1.31	.262	> .999
broadminded	N	0.07	0.07	1	21.24	0.13	.718	> .999
calm	N	2.26	2.26	1	21.01	2.85	.106	> .999
caring	N	0.19	0.19	1	28.00	0.63	.433	> .999
cautious	N	0.59	0.59	1	18.73	0.97	.337	> .999
creative	N	0.53	0.53	1	16.63	2.22	.155	> .999
curious	N	0.16	0.16	1	19.78	0.20	.664	> .999
friendly	N	0.00	0.00	1	28.00	0.00	.971	> .999
hardworking	N	0.00	0.00	1	17.35	0.04	.850	> .999
helpful	N	0.11	0.11	1	23.35	0.70	.410	> .999
imaginative	N	0.08	0.08	1	20.83	0.31	.582	> .999
intelligent	N	0.02	0.02	1	18.94	0.10	.755	> .999
lively	N	4.35	4.35	1	18.29	13.11	.002	.059
organized	N	0.29	0.29	1	16.69	0.91	.354	> .999
outgoing	N	0.18	0.18	1	17.41	0.62	.440	> .999
responsible	N	0.24	0.24	1	21.98	0.74	.399	> .999
selfdisciplined	N	0.17	0.17	1	17.30	1.13	.303	> .999
softhearted	N	0.91	0.91	1	21.21	0.69	.415	> .999
sophisticated	N	0.45	0.45	1	17.51	0.79	.385	> .999
sympathetic	N	0.60	0.60	1	20.31	1.17	.293	> .999
talkative	N	0.42	0.42	1	16.66	0.77	.394	> .999
thorough	N	1.64	1.64	1	20.01	5.48	.030	.893
thrifty	N	1.48	1.48	1	19.77	1.78	.197	> .999
warm	N	0.00	0.00	1	20.23	0.00	.967	> .999
careless	Y	1.34	1.34	1	26.28	0.60	.446	> .999
impulsive	Y	0.13	0.13	1	17.64	0.16	.693	> .999
moody	Y	0.39	0.39	1	20.49	0.40	.534	> .999
nervous	Y	3.24	3.24	1	26.54	1.89	.181	> .999
reckless	Y	0.85	0.85	1	21.68	0.68	.420	> .999
worrying	Y	0.01	0.01	1	20.29	0.01	.932	> .999

```
  kable(digits = 2,
        col.names = c("Item", "Reverse\nScored?", "SS", "MS", "df1", "df2", "F", "raw", "adj"),
        booktabs = T, caption = "Interaction of format and \"I\" (block 1 and 3 data)") %>%
  kable_styling() %>%
    add_header_above(c(" " = 7, "p-value" = 2))
```

3.4.2 Pairwise t-tests for significant ANOVAs

Here we identify the specific items with significant differences.

```
sig_item_i = summary_by_item_i %>%
  filter(p.value < .05)

sig_item_i = sig_item_i$item
sig_item_i
```

```
## [1] "lively"   "thorough"
```

3.4.3 Curious

```
curious_model_i = items_13 %>%
  filter(item == "curious") %>%
  lmer(response~format*i + (1|proid),
                              data = .)

curious_model_i %>%
  tidy() %>%
  mutate(term = str_replace(term, "\n", " "),
         term = str_replace(term, "format", ""),
         term = str_replace(term, "iPresent", "I")) %>%
  filter(is.na(group)) %>%
  select(-effect, -group) %>%
  mutate(across( starts_with("p"), printp )) %>% # format p-values
  kable(booktabs = T,
        digits = 2,
        caption = "Interaction of format and \"I\"",
        col.names = c("Term", "Estimate", "SE",  "t", "df", "p")) %>%
  kable_styling()
```

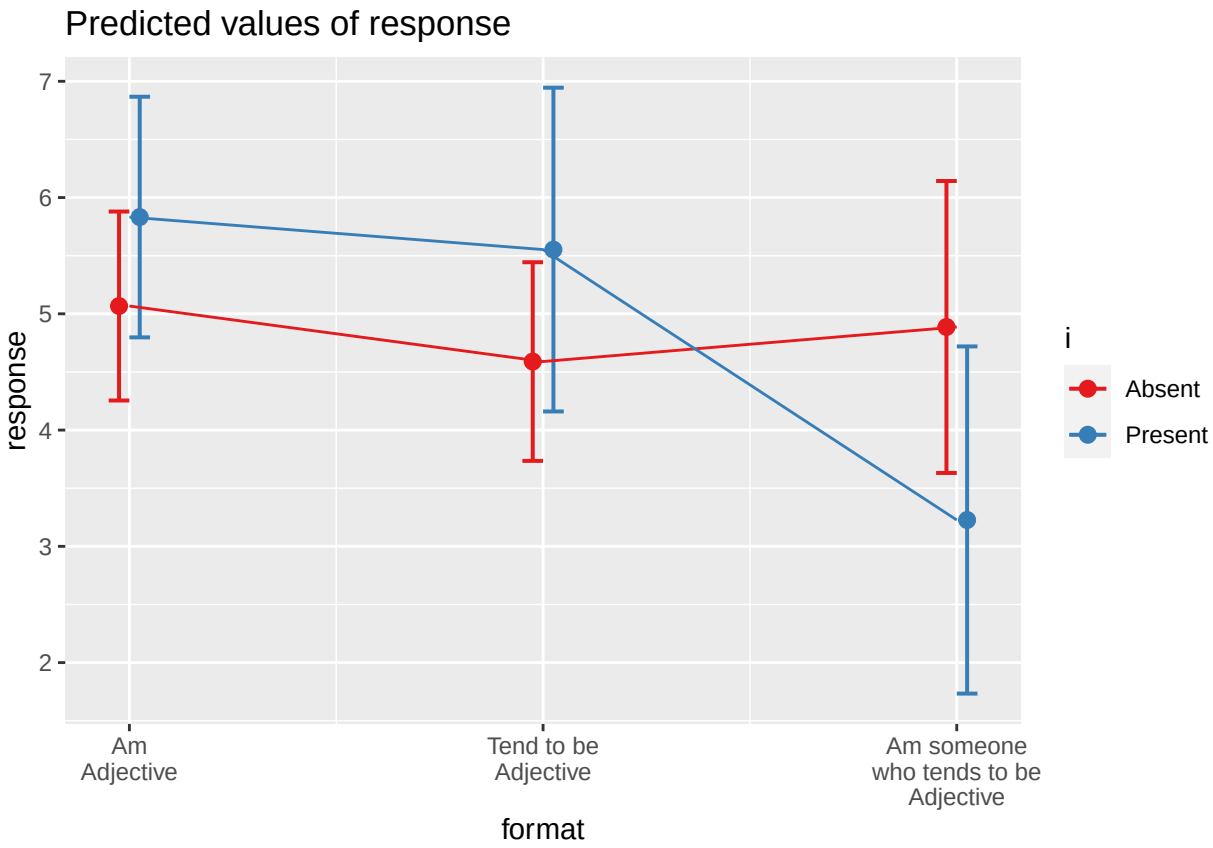
```
plot_model(curious_model_i, type = "pred",
           terms = c("format", "i")) +
  geom_line()
```

Table 11: Interaction of format and "I" (block 1 and 3 data)

							p-value	
Item	Reverse Scored?	SS	MS	df1	df2	F	raw	adj
active	N	0.65	0.32	2	15.11	0.76	.487	> .999
adventurous	N	1.49	0.75	2	22.11	1.52	.241	> .999
broadminded	N	1.95	0.97	2	20.88	1.94	.169	> .999
calm	N	0.31	0.16	2	20.67	0.18	.837	> .999
caring	N	0.27	0.13	2	26.00	0.43	.658	> .999
cautious	N	1.23	0.62	2	15.17	1.15	.344	> .999
creative	N	0.11	0.05	2	15.03	0.21	.812	> .999
curious	N	5.15	2.58	2	16.43	4.65	.025	.729
friendly	N	0.55	0.27	2	26.00	0.87	.431	> .999
hardworking	N	0.28	0.14	2	14.65	1.16	.342	> .999
helpful	N	0.23	0.12	2	21.45	0.69	.515	> .999
imaginative	N	0.29	0.15	2	19.61	0.51	.608	> .999
intelligent	N	0.07	0.04	2	16.71	0.16	.854	> .999
lively	N	2.34	1.17	2	15.12	5.78	.014	.413
organized	N	0.33	0.16	2	15.11	0.48	.629	> .999
outgoing	N	0.26	0.26	1	16.27	0.90	.356	> .999
responsible	N	2.07	1.03	2	19.58	4.17	.031	.869
selfdisciplined	N	0.47	0.23	2	14.86	1.65	.225	> .999
softhearted	N	10.15	5.08	2	20.14	3.69	.043	> .999
sophisticated	N	1.96	0.98	2	15.26	1.96	.174	> .999
sympathetic	N	3.24	1.62	2	14.45	5.91	.013	.413
talkative	N	0.17	0.08	2	14.68	0.13	.875	> .999
thorough	N	0.01	0.00	2	18.49	0.01	.988	> .999
thrifty	N	1.22	0.61	2	17.82	0.70	.509	> .999
warm	N	0.45	0.23	2	18.90	0.40	.673	> .999
careless	Y	9.07	4.54	2	24.69	2.16	.137	> .999
impulsive	Y	0.06	0.03	2	15.73	0.03	.966	> .999
moody	Y	1.12	0.56	2	17.91	0.55	.587	> .999
nervous	Y	1.35	0.67	2	23.48	0.41	.667	> .999
reckless	Y	2.37	1.18	2	18.59	1.05	.370	> .999
worrying	Y	1.95	0.97	2	18.56	0.89	.429	> .999

Table 12: Interaction of format and "I"

Term	Estimate	SE	t	df	p
(Intercept)	5.07	0.41	12.22	15.50	< .001
Tend to be Adjective	-0.48	0.60	-0.79	14.75	.440
Am someone who tends to be Adjective	-0.18	0.76	-0.24	15.96	.816
I	0.77	0.50	1.53	16.12	.145
Tend to be Adjective:I	0.20	0.85	0.23	17.12	.819
Am someone who tends to be Adjective:I	-2.43	0.87	-2.78	15.68	.013



3.4.4 Lively

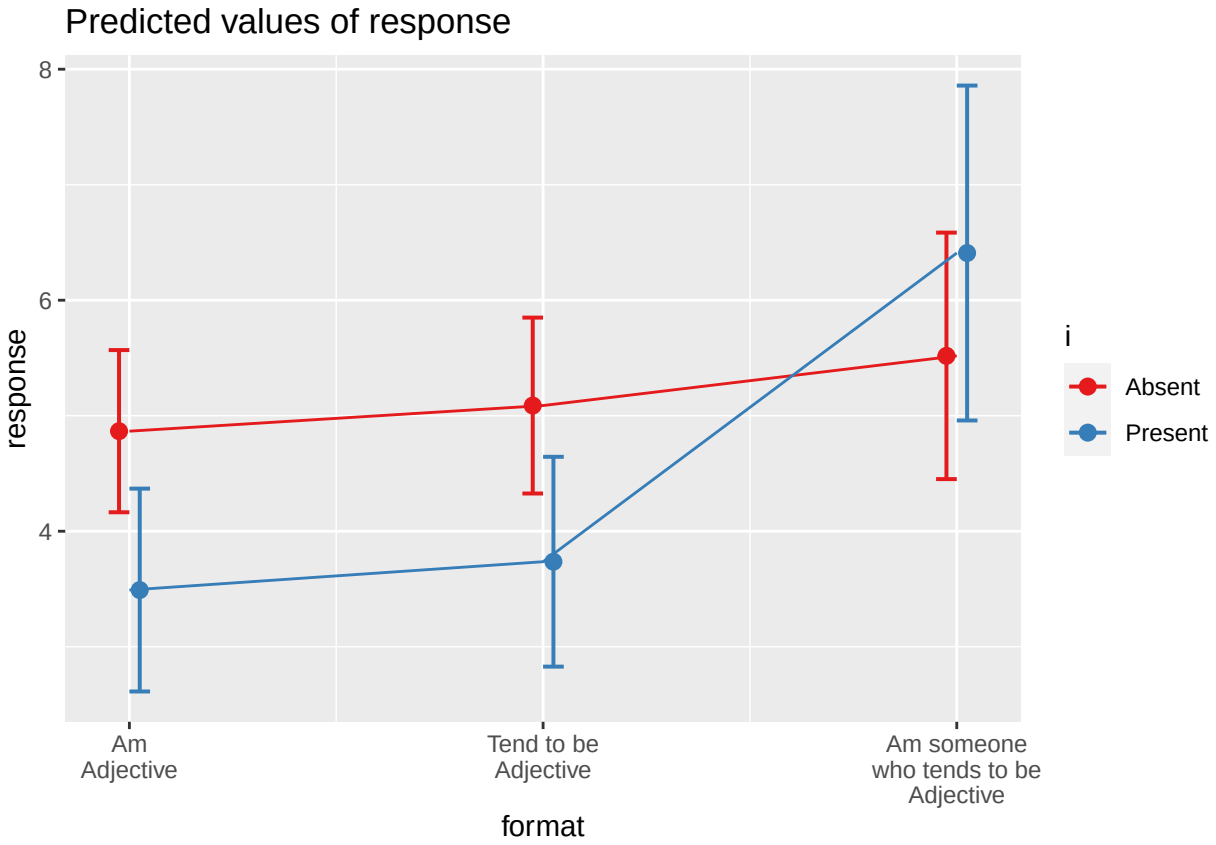
```
lively_model_i = items_13 %>%
  filter(item == "lively") %>%
  lmer(response~format*i + (1|proid),
                                data = .)

lively_model_i %>%
  tidy() %>%
  mutate(term = str_replace(term, "\n", " "),
         term = str_replace(term, "format", ""),
         term = str_replace(term, "iPresent", "I")) %>%
  filter(is.na(group)) %>%
  select(-effect, -group) %>%
  mutate(across( starts_with("p"), printp )) %>% # format p-values
  kable(booktabs = T,
        digits = 2,
        caption = "Interaction of format and \"I\"",
        col.names = c("Term", "Estimate", "SE",  "t", "df", "p")) %>%
  kable_styling()
```

```
plot_model(lively_model_i, type = "pred",
           terms = c("format", "i")) +
  geom_line()
```

Table 13: Interaction of format and "I"

Term	Estimate	SE	t	df	p
(Intercept)	4.87	0.36	13.58	14.04	< .001
Tend to be Adjective	0.22	0.53	0.42	14.15	.681
Am someone who tends to be Adjective	0.65	0.65	1.00	13.85	.334
I	-1.38	0.36	-3.86	15.06	.002
Tend to be Adjective:I	0.02	0.50	0.05	14.95	.963
Am someone who tends to be Adjective:I	2.27	0.71	3.20	15.27	.006



3.4.5 Responsible

```
responsible_model_i = items_13 %>%
  filter(item == "responsible") %>%
  lmer(response~format*i + (1|proid),
                              data = .)

responsible_model_i %>%
  tidy() %>%
  mutate(term = str_replace(term, "\n", " "),
         term = str_replace(term, "format", ""),
         term = str_replace(term, "iPresent", "I")) %>%
  filter(is.na(group)) %>%
```

Table 14: Interaction of format and "I"

Term	Estimate	SE	t	df	p
(Intercept)	5.50	0.21	26.30	16.71	< .001
Tend to be Adjective	-0.04	0.31	-0.11	17.32	.910
Am someone who tends to be Adjective	0.06	0.37	0.17	15.13	.868
I	0.25	0.32	0.78	18.85	.443
Tend to be Adjective:I	-1.14	0.46	-2.50	18.26	.022
Am someone who tends to be Adjective:I	0.38	0.69	0.55	20.83	.591

```
  select(-effect, -group) %>%
  mutate(across( starts_with("p"), printp )) %>% # format p-values
  kable(booktabs = T,
        digits = 2,
        caption = "Interaction of format and \"I\"",
        col.names = c("Term", "Estimate", "SE",  "t", "df", "p")) %>%
  kable_styling()
```

```
plot_model(responsible_model_i, type = "pred",
           terms = c("format", "i")) +
  geom_line()
```

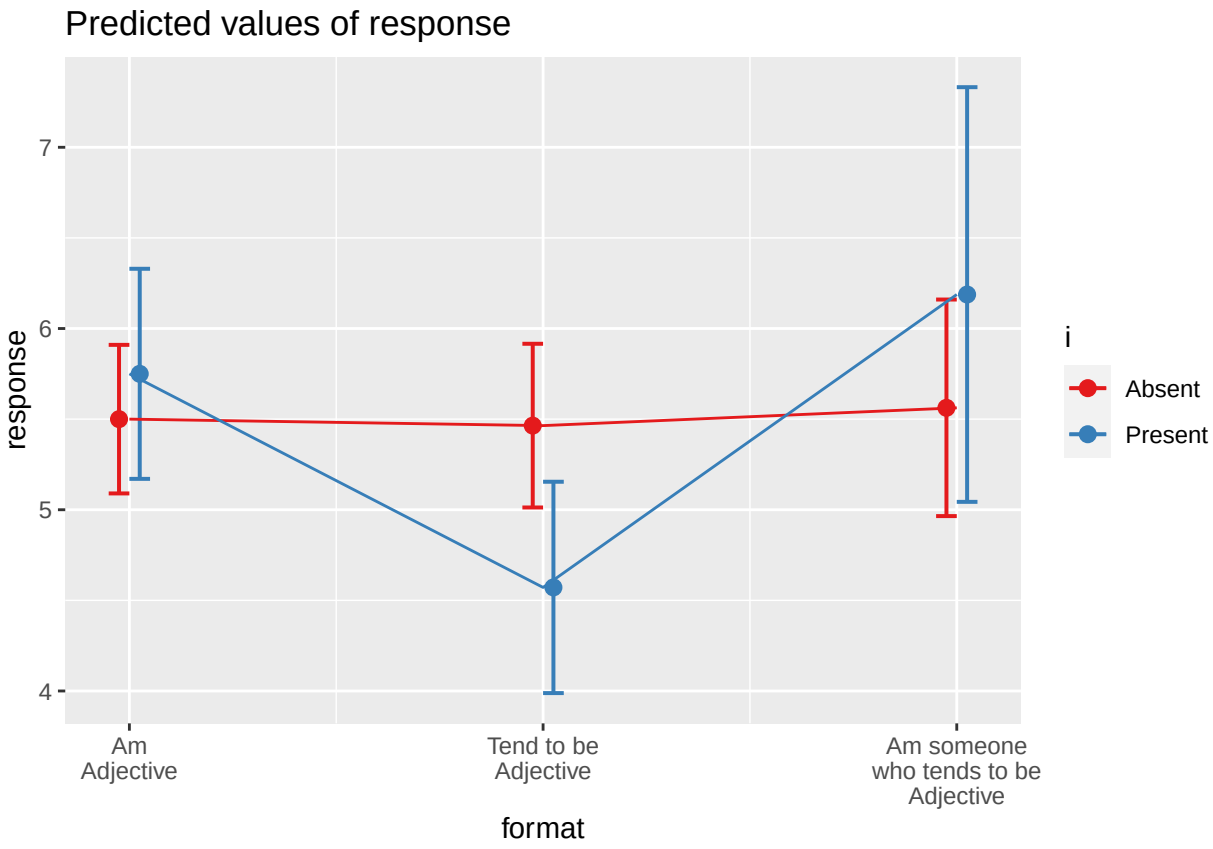


Table 15: Interaction of format and "I"

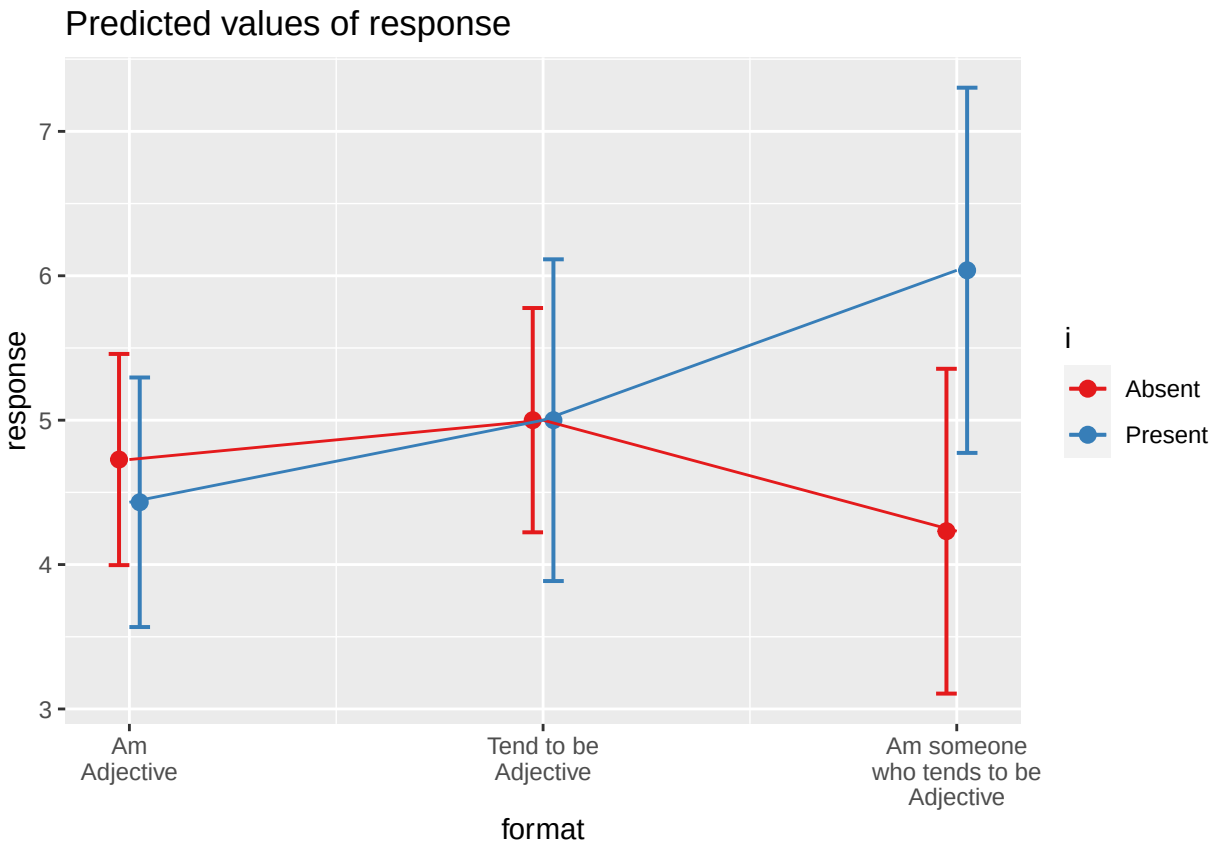
Term	Estimate	SE	t	df	p
(Intercept)	4.73	0.37	12.67	14.07	< .001
Tend to be Adjective	0.27	0.54	0.50	13.60	.624
Am someone who tends to be Adjective	-0.50	0.68	-0.73	14.34	.480
I	-0.30	0.36	-0.82	14.25	.423
Tend to be Adjective:I	0.30	0.61	0.48	14.87	.637
Am someone who tends to be Adjective:I	2.10	0.62	3.37	13.99	.005

3.4.6 Sympathetic

```
sympathetic_model_i = items_13 %>%
  filter(item == "sympathetic") %>%
  lmer(response~format*i + (1|proid),
                                data = .)

sympathetic_model_i %>%
  tidy() %>%
  mutate(term = str_replace(term, "\n", " "),
         term = str_replace(term, "format", ""),
         term = str_replace(term, "iPresent", "I")) %>%
  filter(is.na(group)) %>%
  select(-effect, -group) %>%
  mutate(across( starts_with("p"), printp )) %>% # format p-values
  kable(booktabs = T,
        digits = 2,
        caption = "Interaction of format and \"I\"",
        col.names = c("Term", "Estimate", "SE",  "t", "df", "p")) %>%
  kable_styling()
```

```
plot_model(sympathetic_model_i, type = "pred",
           terms = c("format", "i")) +
  geom_line()
```


Predicted values of response
response
7
6
5
4
3
Am
Adjective
Tend to be
Adjective
Am someone
who tends to be
Adjective
format
i
Absent
Present

4 Does the internal consistency of Big Five traits vary by item wording?

We calculate and report Cronbach's alpha for all formats using data from Block 1 only. This will include both the average split-half reliability, as well as the 95% confidence interval. Differences in internal consistency will be considered statistically significant if the confidence intervals of two formats do not overlap. We will also show the distribution of all possible split halves for each of the four formats.

4.1 Prep data

We start by creating a wide-format of the dataset using only the Block 1 data.

```
items_wide = items_df %>%
  # only block 1 responses
  filter(block == 1) %>%
  #only need these variables
  select(proid, condition, item, response) %>%
  # to wide form
  spread(item, response)
```

Next, we identify the items associated with each trait. These come from the Health and Retirement Study Psychosocial and Lifestyle Questionnaire 2006-2016 user guide, which can be found at this link.

```
Extra = c("outgoing", "friendly", "lively", "active" ,"talkative")
Agree = c("helpful", "warm", "caring", "softhearted", "sympathetic")
Consc = c("reckless", "organized", "responsible", "hardworking", "selfdisciplined",
          "careless", "impulsive", "cautious", "thorough", "thrifty")
Neuro = c("moody", "worrying", "nervous", "calm")
Openn = c("creative", "imaginative", "intelligent", "curious", "broadminded",
          "sophisticated", "adventurous")
```

4.2 Calculate Cronbach's alpha for each format

We start by grouping data by condition and then nesting, to create separate data frames for each of the four fomats.

```
format_data = items_wide %>%
  group_by(condition) %>%
  nest() %>%
  ungroup()
```

Next we create separate datasets for each of the five personality traits.

```
format_data = format_data %>%
  mutate(
    data_Extra = map(data, ~select(.x, all_of(Extra))),
    data_Agree = map(data, ~select(.x, all_of(Agree))),
    data_Consc = map(data, ~select(.x, all_of(Consc))),
    data_Neuro = map(data, ~select(.x, all_of(Neuro))),
    data_Openn = map(data, ~select(.x, all_of(Openn)))
    )
```

We gather these datasets into a single column, for ease of use.

```
format_data = format_data %>%
  select(-data) %>%
  gather(variable, data, starts_with("data")) %>%
  mutate(variable = str_remove(variable, "data_"))
```

Next we apply the alpha function to the datasets. We do not need to use the `check.keys` function, as items were reverse-scored during the cleaning process.

```
format_data = format_data %>%
  mutate(alpha = map(data, psych::alpha))
```

```
## Some items ( talkative ) were negatively correlated with the total scale and
## probably should be reversed.
## To do this, run the function again with the 'check.keys=TRUE' optionSome items ( active ) were negati
## probably should be reversed.
## To do this, run the function again with the 'check.keys=TRUE' optionSome items ( friendly ) were nega
## probably should be reversed.
## To do this, run the function again with the 'check.keys=TRUE' optionSome items ( sympathetic ) were r
## probably should be reversed.
## To do this, run the function again with the 'check.keys=TRUE' optionSome items ( softhearted sympathe
## probably should be reversed.
## To do this, run the function again with the 'check.keys=TRUE' optionSome items ( reckless impulsive
## probably should be reversed.
## To do this, run the function again with the 'check.keys=TRUE' optionSome items ( hardworking impulsiv
## probably should be reversed.
## To do this, run the function again with the 'check.keys=TRUE' optionSome items ( reckless careless in
## probably should be reversed.
## To do this, run the function again with the 'check.keys=TRUE' optionSome items ( worrying ) were nega
## probably should be reversed.
## To do this, run the function again with the 'check.keys=TRUE' optionSome items ( sophisticated ) were
## probably should be reversed.
## To do this, run the function again with the 'check.keys=TRUE' option
```

We extract the estimated confidence intervals. (Note that these estimates are unreliable in small samples. The estimates extracted based on pilot data are not expected to reflect estimates provided in the final analyses.)

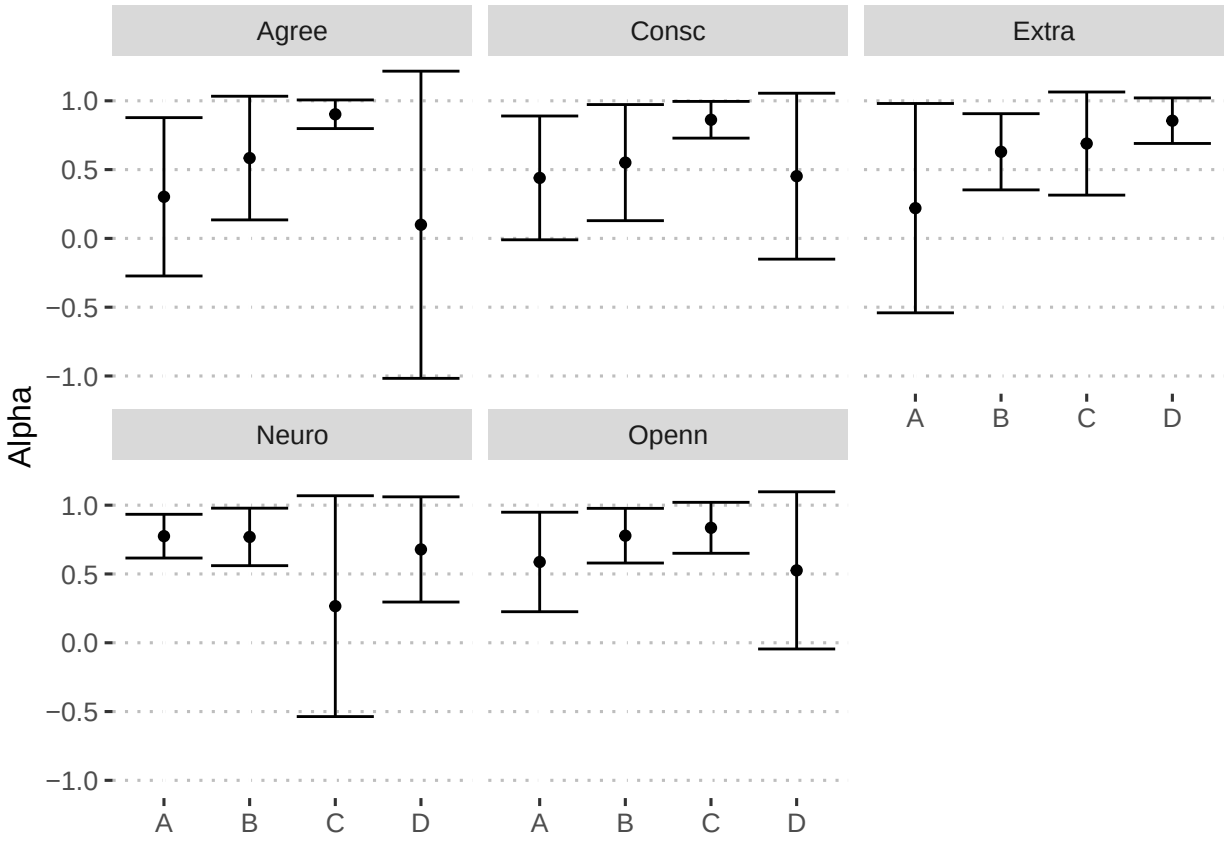
```
format_data = format_data %>%
  mutate(alpha_list = map(alpha, "total"),
         alpha_est = map_dbl(alpha_list, "raw_alpha"),
         se_est = map_dbl(alpha_list, "ase"),
         lower_est = alpha_est - (1.96*se_est),
         upper_est = alpha_est + (1.96*se_est))

format_data %>%
  select(condition, variable, alpha_est, lower_est, upper_est) %>%
  mutate(
    across(ends_with("est"), printnum),
    value = paste0(alpha_est, " [", lower_est, ", ", upper_est, "]")) %>%
  select(-contains("est")) %>%
  spread(condition, value) %>%
```

variable	A	B	C	D
Agree	0.30 [-0.27, 0.88]	0.58 [0.13, 1.03]	0.90 [0.80, 1.01]	0.10 [-1.02, 1.22]
Consc	0.44 [-0.01, 0.89]	0.55 [0.13, 0.97]	0.86 [0.73, 1.00]	0.45 [-0.15, 1.06]
Extra	0.22 [-0.54, 0.98]	0.63 [0.35, 0.91]	0.69 [0.31, 1.06]	0.86 [0.69, 1.02]
Neuro	0.77 [0.62, 0.93]	0.77 [0.56, 0.98]	0.27 [-0.54, 1.07]	0.68 [0.30, 1.06]
Openn	0.59 [0.23, 0.95]	0.78 [0.58, 0.98]	0.84 [0.65, 1.02]	0.53 [-0.05, 1.10]

```
  kable(booktabs = T) %>%
  kable_styling()
```

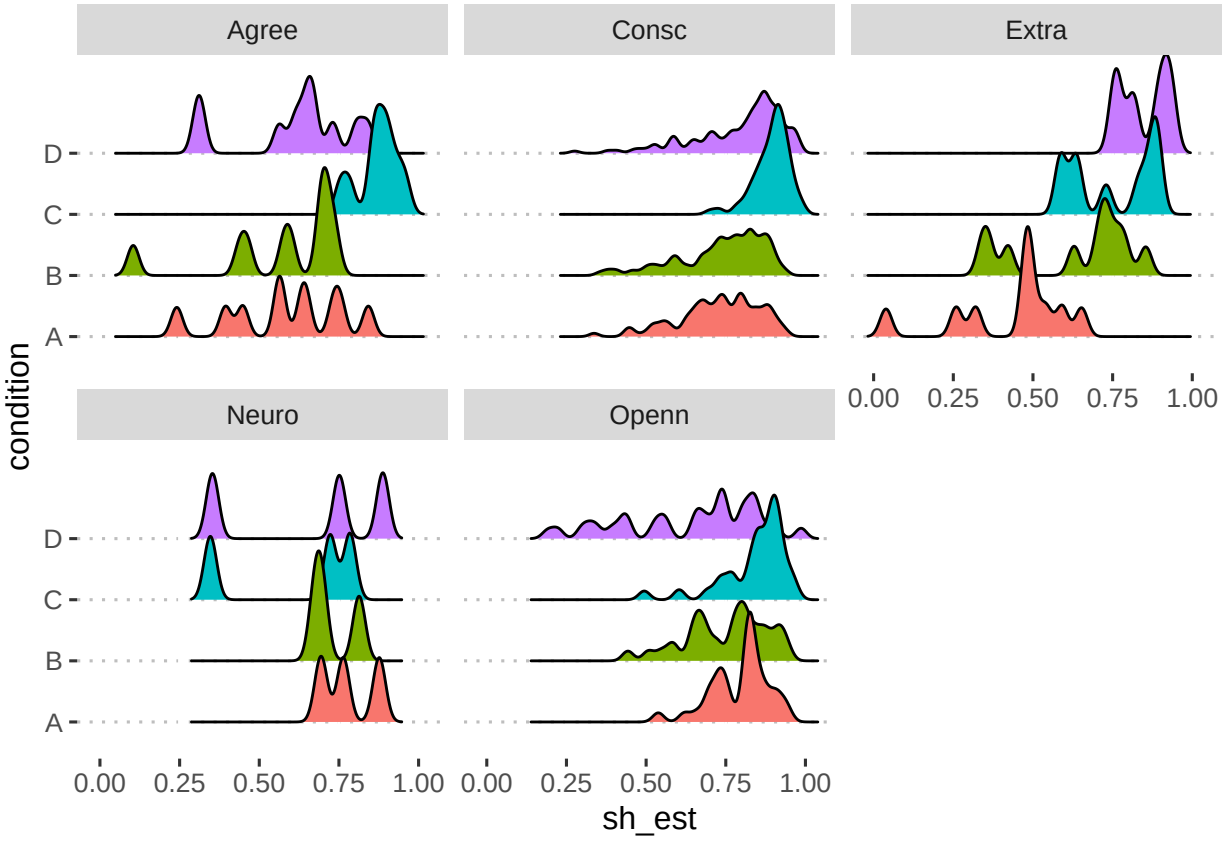
```
format_data %>%
  ggplot(aes(x = condition, y = alpha_est)) +
  geom_errorbar(aes(ymin = lower_est, ymax = upper_est)) +
  geom_point() +
  labs(x = NULL, y = "Alpha") +
  facet_wrap(~variable) +
  theme_pubclean()
```



Alpha is the average split-half reliability; given space, it can be useful to report the distribution of all split-half reliability estimates. We use the `splitHalf` function to calculate those. We use smoothed correlation matrices here because when developing code on the pilot data, we had non-positive definite correlation matrices.

```
format_data = format_data %>%
  mutate(cor_mat = map(data, cor),
         cor_mat = map(cor_mat, cor.smooth)) %>%
  mutate(splithalf = map(cor_mat, splitHalf, raw = T))

format_data %>%
  mutate(sh_est = map(splithalf, "raw")) %>%
  select(variable, condition, sh_est) %>%
  unnest(cols = c(sh_est)) %>%
  ggplot(aes(x = sh_est, y = condition, fill = condition)) +
  geom_density_ridges() +
  facet_wrap(~variable) +
  guides(fill = "none") +
  theme_pubclean()
```



5 Does the test-retest reliability of personality items change as a function of item wording?

We also plan to evaluate test-retest reliability within formats (within session and over two weeks); we expect slightly higher test-retest reliability for item wording formats that are more specific – formats #3 and #4 above vs the use of adjectives alone. In other words, we expect equal or lower retest reliability for the adjectives than for longer phrases. We will also consider the effect of performance on the word recall task on retest reliability .

5.1 Prep dataset

The data structure needed for these analyses is in wide-format. That is, we require one column for each time point. In addition, we hope to examine reliability *within* format, which requires selecting only the response options which match the original, Block 1, assessment.

```
items_df = items_df %>%
  mutate(condition = tolower(condition)) %>%
  mutate(condition = factor(condition,
                            levels = c("a", "b", "c", "d"),
                            labels = c("Adjective\nOnly", "Am\nAdjective", "Tend to be\nAdjective", "Am

items_matchb1 = items_df %>%
  mutate(across(c(format, condition), as.character)) %>%
  filter(format == condition) %>%
  mutate(block = paste0("block_", block)) %>%
  select(-timing, -seconds_log, -i) %>%
  spread(block, response)
```

We standardize responses within each block – this allows us to use a regression framework yet interpret the slopes as correlations.

```
items_matchb1 = items_matchb1 %>%
  mutate(across(
    starts_with("block"), ~(.-mean(., na.rm=T))/sd(., na.rm = T)
  ))
```

We also standardize the memory scores for ease of interpretation.

```
items_matchb1 = items_matchb1 %>%
  mutate(across(
    ends_with("memory"), ~(.-mean(., na.rm=T))/sd(., na.rm = T)
  ))
```

5.2 Test-retest reliability (all items pooled)

To estimate the reliability coefficients, we use a multilevel model, predicting the latter block from the earlier one. These models nest responses within participant, allowing us to estimate standard errors which account for the dependency of scores.

```
tr_mod1_b1b2 = lmer(block_2 ~ block_1 + (1 |proid), data = items_matchb1)
tr_mod1_b1b3 = lmer(block_3 ~ block_1 + (1 |proid), data = items_matchb1)

tab_model(tr_mod1_b1b2, tr_mod1_b1b3, show.re.var = F)
```

block 2

block 3

Predictors

Estimates

CI

p

Estimates

CI

p

(Intercept)

-0.02

-0.12 – 0.09

0.733

-0.05

-0.15 – 0.05

0.316

block_1

0.78

0.70 – 0.85

<0.001

0.64

0.58 – 0.70

<0.001

ICC

0.11

0.06

N

35 proid

22 proid

Observations

252

682

Marginal R2 / Conditional R2

0.597 / 0.642

0.389 / 0.425

5.3 Test-retest reliability (all items pooled, by memory)

Here we fit models moderated by memory – that it, perhaps the test-retest coefficient is affected by the memory of the participant.

```
tr_mod2_b1b2 = lmer(block_2 ~ block_1*delayed_memory +
                      (1 |proid),
                    data = items_matchb1)
tr_mod2_b1b3 = lmer(block_3 ~ block_1*very_delayed_memory +
                      (1 |proid),
                    data = items_matchb1)

tab_model(tr_mod2_b1b2, tr_mod2_b1b3, show.re.var = F)
```

block 2

block 3

Predictors

Estimates

CI

p

Estimates

CI

p

(Intercept)

-0.01

-0.12 – 0.09

0.826

-0.05

-0.15 – 0.05

0.325

block_1

0.77

0.68 – 0.85

<0.001

0.64

0.58 – 0.70

<0.001

delayed_memory

0.02

-0.09 – 0.13

0.746

block_1 * delayed_memory

-0.05

-0.14 – 0.04

0.301

very_delayed_memory

-0.00

-0.10 – 0.10

0.956

block_1 *very_delayed_memory

0.01

-0.06 – 0.08

0.749

ICC

0.12

0.06

N

35 proid

22 proid

Observations

252

682

Marginal R2 / Conditional R2

0.597 / 0.644

0.388 / 0.426

We also extract the simple slopes estimates of these models, which allow us to more explicitly identify and compare the test-retest correlations.

5.3.1 Block 1/Block 2

```
mem_list = list(delayed_memory = c(-1,0,1))

emtrends(tr_mod2_b1b2,
         pairwise~delayed_memory,
         var = "block_1",
         at = mem_list)
```

```
## $emtrends
##  delayed_memory block_1.trend     SE  df lower.CL upper.CL
##              -1         0.814 0.0565 246    0.703    0.925
##               0         0.765 0.0414 247    0.684    0.847
##               1         0.717 0.0686 247    0.582    0.852
##
## Degrees-of-freedom method: kenward-roger
## Confidence level used: 0.95
##
## $contrasts
##  contrast estimate     SE  df t.ratio p.value
##  (-1) - 0   0.0486 0.0473 247 1.028   0.5599
##  (-1) - 1   0.0972 0.0946 247 1.028   0.5599
##  0 - 1      0.0486 0.0473 247 1.028   0.5599
##
## Degrees-of-freedom method: kenward-roger
## P value adjustment: tukey method for comparing a family of 3 estimates
```

5.3.2 Block 1/Block 3

This chunk is turned off due to low coverage. Be sure to turn on with real data.

```
mem_list = list(very_delayed_memory = c(-1,0,1))

emtrends(tr_mod2_b1b3,
         pairwise~very_delayed_memory,
         var = "block_1",
         at = mem_list)
```

```
## $emtrends
##  very_delayed_memory block_1.trend     SE  df lower.CL upper.CL
##                   -1         0.631 0.0444 677    0.544    0.718
##                    0         0.642 0.0312 678    0.581    0.703
##                    1         0.653 0.0502 674    0.555    0.752
##
## Degrees-of-freedom method: kenward-roger
## Confidence level used: 0.95
##
## $contrasts
##  contrast estimate     SE  df t.ratio p.value
##  (-1) - 0  -0.0114 0.0356 676 -0.319  0.9456
##  (-1) - 1  -0.0227 0.0713 676 -0.319  0.9456
##  0 - 1     -0.0114 0.0356 676 -0.319  0.9456
##
## Degrees-of-freedom method: kenward-roger
## P value adjustment: tukey method for comparing a family of 3 estimates
```

5.4 Test-retest reliability (all items pooled, by format)

We fit these same models, except now we moderate by format, to determine whether the test-retest reliabilty differs as a function of item wording.

```
tr_mod3_b1b2 = lmer(block_2 ~ block_1*condition + (1 |proid),
                    data = items_matchb1)
tr_mod3_b1b3 = lmer(block_3 ~ block_1*condition + (1 |proid),
                    data = items_matchb1)

anova(tr_mod3_b1b2)
```

Type III Analysis of Variance Table with Satterthwaite's method Sum Sq Mean Sq NumDF DenDF F value Pr(>F)
block_1 129.868 129.868 1 242.83 368.1068 < 2e-16 *** condition 0.961 0.320 3 27.78 0.9082 0.44968
block_1:condition 2.518 0.839 3 238.81 2.3793 0.07039 .
— Signif. codes: 0 '*' ***0.001*** '*' *0.01* '' 0.05 '.' 0.1 ' ' 1

```
anova(tr_mod3_b1b3)
```

Type III Analysis of Variance Table with Satterthwaite's method Sum Sq Mean Sq NumDF DenDF F value Pr(>F)
block_1 240.879 240.88 1 646.33 416.9331 < 2e-16 ** *condition 8.459 2.82 3 17.56 4.8803 0.01212*
block_1:condition 1.201 0.40 3 644.48 0.6927 0.55673
— Signif. codes: 0 '*' ***0.001*** '*' *0.01* '' 0.05 '.' 0.1 ' ' 1

We also extract the simple slopes estimates of these models, which allow us to more explicitly identify and compare the test-retest correlations.

5.4.1 Block 1/Block 2

```
emtrends(tr_mod3_b1b2, pairwise ~ condition, var = "block_1")
```

```
## $emtrends
##   condition                               block_1.trend     SE  df lower.CL
##  Adjective\nOnly                                  0.611 0.0739 230    0.465
##  Am\nAdjective                                    0.822 0.0706 240    0.683
##  Am someone\nwho tends to be\nAdjective           0.877 0.0986 223    0.683
##  Tend to be\nAdjective                            0.843 0.0856 240    0.674
##  upper.CL
##     0.757
##     0.961
##     1.072
##     1.011
##
## Degrees-of-freedom method: kenward-roger
## Confidence level used: 0.95
##
## $contrasts
##    contrast                                                 estimate    SE
##  Adjective\nOnly - Am\nAdjective                             -0.2108 0.102
##  Adjective\nOnly - Am someone\nwho tends to be\nAdjective    -0.2663 0.123
##  Adjective\nOnly - Tend to be\nAdjective                     -0.2316 0.113
##  Am\nAdjective - Am someone\nwho tends to be\nAdjective      -0.0555 0.121
##  Am\nAdjective - Tend to be\nAdjective                       -0.0208 0.111
##  Am someone\nwho tends to be\nAdjective - Tend to be\nAdjective  0.0347 0.131
```

```
##   df t.ratio p.value
##  236 -2.062  0.1686
##  241 -2.161  0.1372
##  244 -2.048  0.1734
##  238 -0.457  0.9681
##  244 -0.187  0.9977
##  232  0.266  0.9934
##
## Degrees-of-freedom method: kenward-roger
## P value adjustment: tukey method for comparing a family of 4 estimates
```

5.4.2 Block 1/Block 3

```
emtrends(tr_mod3_b1b3, pairwise ~ condition, var = "block_1")
```

```
## $emtrends
##   condition                              block_1.trend     SE  df lower.CL
##  Adjective\nOnly                                 0.611 0.0573 673    0.499
##  Am\nAdjective                                   0.609 0.0531 673    0.505
##  Am someone\nwho tends to be\nAdjective          0.646 0.0745 671    0.499
##  Tend to be\nAdjective                           0.721 0.0678 400    0.588
##  upper.CL
##     0.724
##     0.713
##     0.792
##     0.854
##
## Degrees-of-freedom method: kenward-roger
## Confidence level used: 0.95
##
## $contrasts
##    contrast                                                  estimate     SE
##  Adjective\nOnly - Am\nAdjective                              0.00221 0.0781
##  Adjective\nOnly - Am someone\nwho tends to be\nAdjective    -0.03459 0.0940
##  Adjective\nOnly - Tend to be\nAdjective                     -0.10986 0.0888
##  Am\nAdjective - Am someone\nwho tends to be\nAdjective      -0.03680 0.0915
##  Am\nAdjective - Tend to be\nAdjective                       -0.11207 0.0861
##  Am someone\nwho tends to be\nAdjective - Tend to be\nAdjective -0.07526 0.1007
##   df t.ratio p.value
##  674  0.028  1.0000
##  673 -0.368  0.9830
##  543 -1.238  0.6031
##  672 -0.402  0.9779
##  543 -1.302  0.5621
##  607 -0.747  0.8778
##
## Degrees-of-freedom method: kenward-roger
## P value adjustment: tukey method for comparing a family of 4 estimates
```

5.5 Test-retest reliability (items separated, by format)

To assess test-retest reliability for each item, we can rely on more simple correlation analyses, as each participant only contributed one response to each item in each block. We first not the sample size coverage for these comparisons:

```
items_matchb1 %>%
  group_by(item, condition) %>%
  count() %>%
  ungroup() %>%
  full_join(expand_grid(item = unique(items_matchb1$item),
                        condition = unique(items_matchb1$condition))) %>%
  mutate(n = ifelse(is.na(n), 0, n)) %>%
  summarise(
    min = min(n),
     max = max(n),
    mean = mean(n),
    median = median(n)
  )
```

```
## # A tibble: 1 x 4
##     min   max  mean median
##   <int> <int> <dbl>  <dbl>
## 1     7    11  8.75    8.5
```

```
items_matchb1 %>%
  group_by(item, condition) %>%
  count() %>%
  ungroup() %>%
  full_join(expand_grid(item = unique(items_matchb1$item),
                        condition = unique(items_matchb1$condition))) %>%
  mutate(n = ifelse(is.na(n), 0, n)) %>%
  ggplot(aes(x = n)) +
  geom_histogram(bins = 50) +
  labs(x = "Sample size",
       y = "Number of tests") +
  facet_wrap(~condition)
```

```
items_cors = items_matchb1 %>%
  select(item, condition, contains("block")) %>%
  group_by(item, condition) %>%
  nest() %>%
   mutate(cors = map(data, psych::corr.test, use = "pairwise"),
          cors = map(cors, print, short = F),
          cors = map(cors, ~.x %>% mutate(comp = rownames(.)))) %>%
  select(item, condition, cors) %>%
  unnest(cols = c(cors))
```

```
items_cors %>%
  mutate(raw.r = printnum(raw.r),
         raw.r = case_when(
           is.na(raw.p) ~ NA_character_,
```

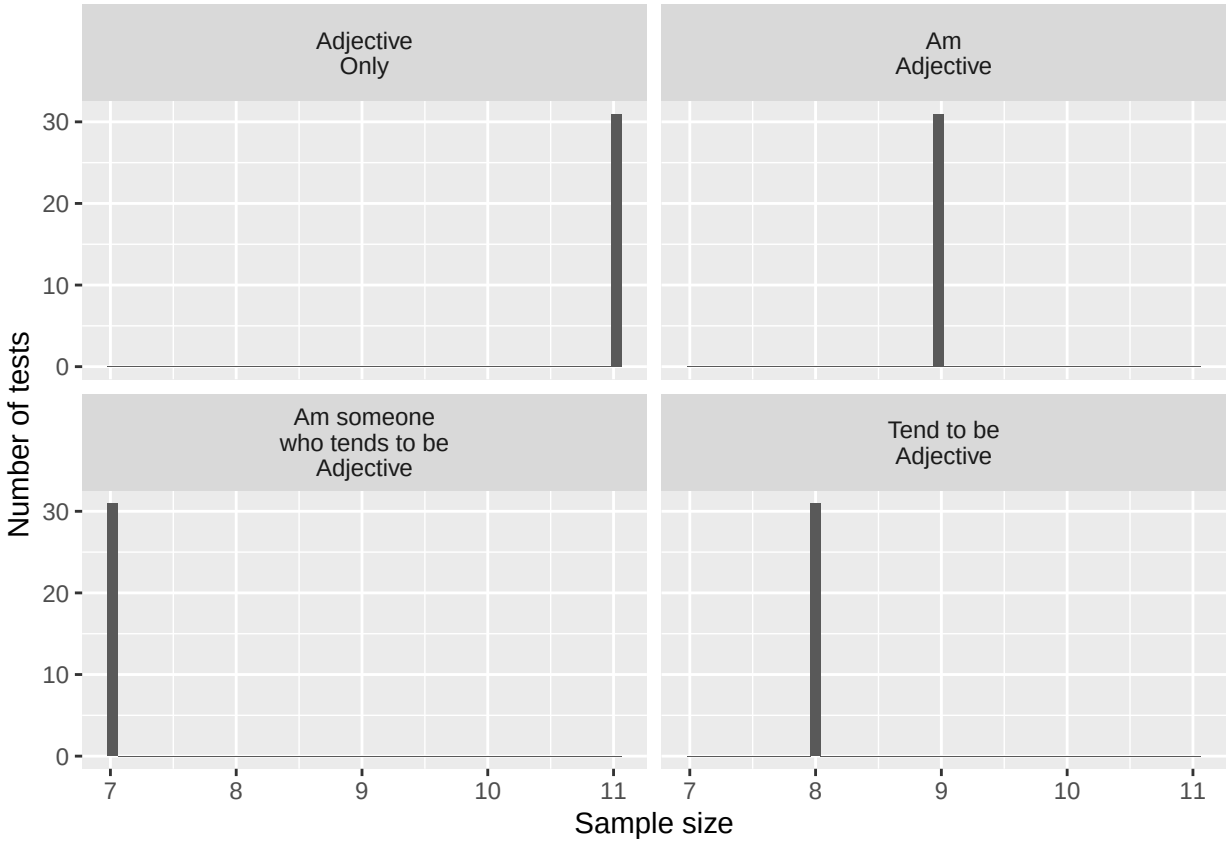


Figure 23: Sample sizes for item-level test-retest correlations.

```
          raw.p < .05 ~ paste0(raw.r, "*"),
          TRUE ~ raw.r)) %>%
  select(item, condition, comp, raw.r) %>%
  mutate(reverse = case_when(
    item %in% reverse ~ "Y",
    TRUE ~ "N"
  )) %>%
  filter(comp != "blc_2-blc_3") %>%
  mutate(condition = case_when(
    condition == "Adjective\nOnly" ~ "a",
    condition == "Am\nAdjective" ~ "b",
    condition == "Tend to be\nAdjective" ~ "c",
    condition == "Am someone\nwho tends to be\nAdjective" ~ "d"
  )) %>%
  unite(comp, condition, comp) %>%
  spread(comp, raw.r) %>%
  arrange(reverse, item) %>%
  kable(caption = "Test-retest correlations for each item and condition. Preregistration note: given th
        col.names = c("Item", "Reverse scored?", rep(c("5 min", "2 weeks"), 4)),
        booktabs = T) %>%
  kable_styling() %>%
  add_header_above(c(" " = 2, "Adjective Only" = 2, "Am Adjective" = 2,
                     "Tend to be" = 2, "Am someone who tends to be" = 2))
```

```
items_cors %>%
  mutate(comp_num = case_when(
    comp == "blc_1-blc_2" ~ 1,
    comp == "blc_1-blc_3" ~ 2,
    comp == "blc_2-blc_3" ~ NA_real_,
  )) %>%
  filter(!is.na(comp_num)) %>%
  ggplot(aes(x = comp_num, y = raw.r, color = condition)) +
  geom_jitter(width = .1) +
  scale_x_continuous(breaks = c(1:2),
                     labels = c("1-2", "1-3")) +
  labs(x = NULL, y = "Correlation") +
  theme_pubclean()
```

Table 16: Test-retest correlations for each item and condition. Preregistration note: given the low sample size for the pilot data, we are missing observations for many of these comparisons. Correlations which could not be computed are blank in this table, but we expect them to be reported in the final manuscript.

		Adjective Only		Am Adjective		Tend to be		Am someone who tends to be	
Item	Reverse scored?	5 min	2 weeks	5 min	2 weeks	5 min	2 weeks	5 min	2 weeks
active	N	0.87	-0.32	0.93	0.84*	1.00*	0.75		-0.50
adventurous	N	0.50	0.00		0.00		0.17	0.97	-0.87
broadminded	N		0.29	0.50	-0.44	0.87	0.82*	0.87	1.00*
calm	N	0.58	0.63		0.71		0.40		-0.19
caring	N	-0.50	0.79		-0.54	1.00*	0.34		
cautious	N		0.59		0.48		-0.42		0.91
creative	N	0.87	0.54	1.00*	0.64	0.50	0.71		
curious	N		0.66	0.30	0.00		0.55		0.87
friendly	N		0.50		0.42		-0.54	1.00*	
hardworking	N		0.25		0.88*		0.50		
helpful	N	0.00	0.00	1.00*	0.09		0.25		
imaginative	N	0.90	0.77		0.24	0.98	0.87*		
intelligent	N		0.59		0.50		0.70		0.50
lively	N		0.97*		0.61		0.69		
organized	N	0.97*	0.70		0.77*		0.98*		
outgoing	N		0.72	0.77	0.80*		0.95*		
responsible	N		0.71		0.42		0.48		-0.50
selfdisciplined	N		0.00		0.71	1.00*	0.96*		1.00*
softhearted	N		0.94*		0.31		0.49		0.98
sophisticated	N	1.00*	-0.40	0.96*	0.87*		0.62		0.97
sympathetic	N		0.00	0.50	0.79*	0.50		0.43	1.00*
talkative	N		0.26		0.92*		0.90*		-0.19
thorough	N		0.41	0.50	-0.09	0.50	0.81		-0.50
thrifty	N	0.93	0.46	0.91	0.75		0.90*		0.00
warm	N		0.61		-0.42		0.81		-0.50
careless	Y	0.87	0.05	0.69	0.65		0.16		
impulsive	Y		0.72		0.57	0.96	0.87*		0.62
moody	Y	0.79	0.96*		0.76		0.10		0.84
nervous	Y		0.22	-1.00*	0.48		-0.58		-0.87
reckless	Y	0.87	0.59		0.64	-0.18	-0.08		-0.87
worrying	Y	0.00	0.84*		0.61		0.80		

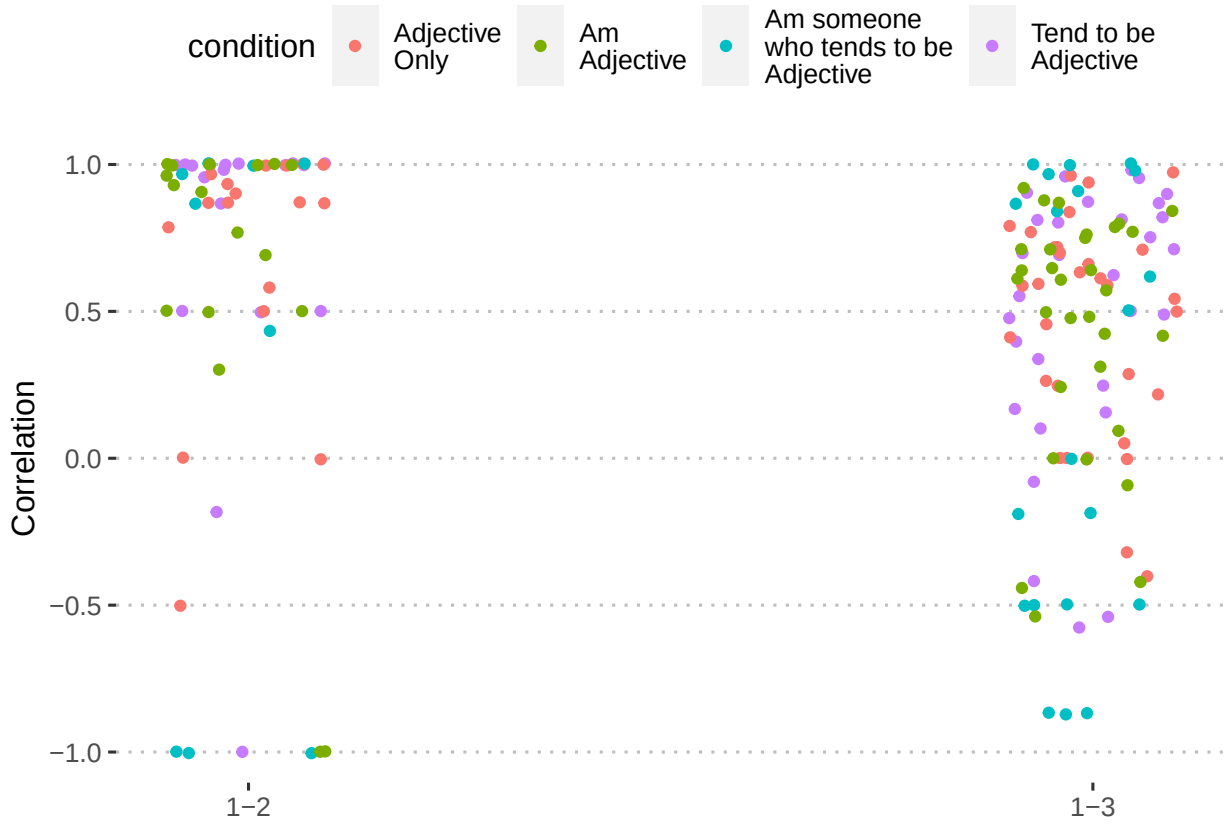


Figure 24: Test-retest correlations of specific items across word format. Each dot represents the test-retest correlation within a specific item.

6 How does format affect timining of responses?

6.1 Analysis: Block 1 data only

We used a multilevel model, nesting log-seconds within participant to account for dependence. Our primary predictor was format. Here, we use only Block 1 data.

```
item_block1 = filter(items_df, block == "1")

mod.format_b1 = lmer(seconds_log~format + (1|proid),
                     data = item_block1)
anova(mod.format_b1)
```

```
## Type III Analysis of Variance Table with Satterthwaite's method
##        Sum Sq Mean Sq NumDF DenDF F value  Pr(>F)
## format 7.1362  2.3788     3    31  4.0608 0.01522 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
plot_b1 = plot_model(mod.format_b1, type = "pred")

plot_b1$format +
  labs(x = NULL,
       y = "Average time (log seconds)",
       title = "Average time by item formatting (Block 1 Data)") +
  theme_pubclean()
```

```
plot_b1$format$data %>%
  mutate(predicted = exp(predicted),
         conf.low = exp(conf.low),
         conf.high = exp(conf.high)) %>%
  mutate(x = factor(x,
                    labels = c("Adjective\nOnly",
                               "Am\nAdjective",
                               "Tend to be\nAdjective",
                               "I am someone\nwho tends to be\nAdjective"))) %>%
  ggplot(aes(x = x, y = predicted)) +
  geom_point() +
  geom_errorbar(aes(ymin = conf.low, ymax = conf.high)) +
  labs(x = NULL, y = "seconds", title = "Average time by item formatting (Block 1 Data)") +
  theme_pubclean()
```

```
means_by_group = item_block1 %>%
  group_by(format) %>%
  summarise(m = mean(seconds_log),
            s = sd(seconds_log))

item_block1 %>%
  ggplot(aes(x = seconds_log, fill = format)) +
  geom_histogram(bins = 50, color = "white") +
  geom_vline(aes(xintercept = m), data = means_by_group) +
```

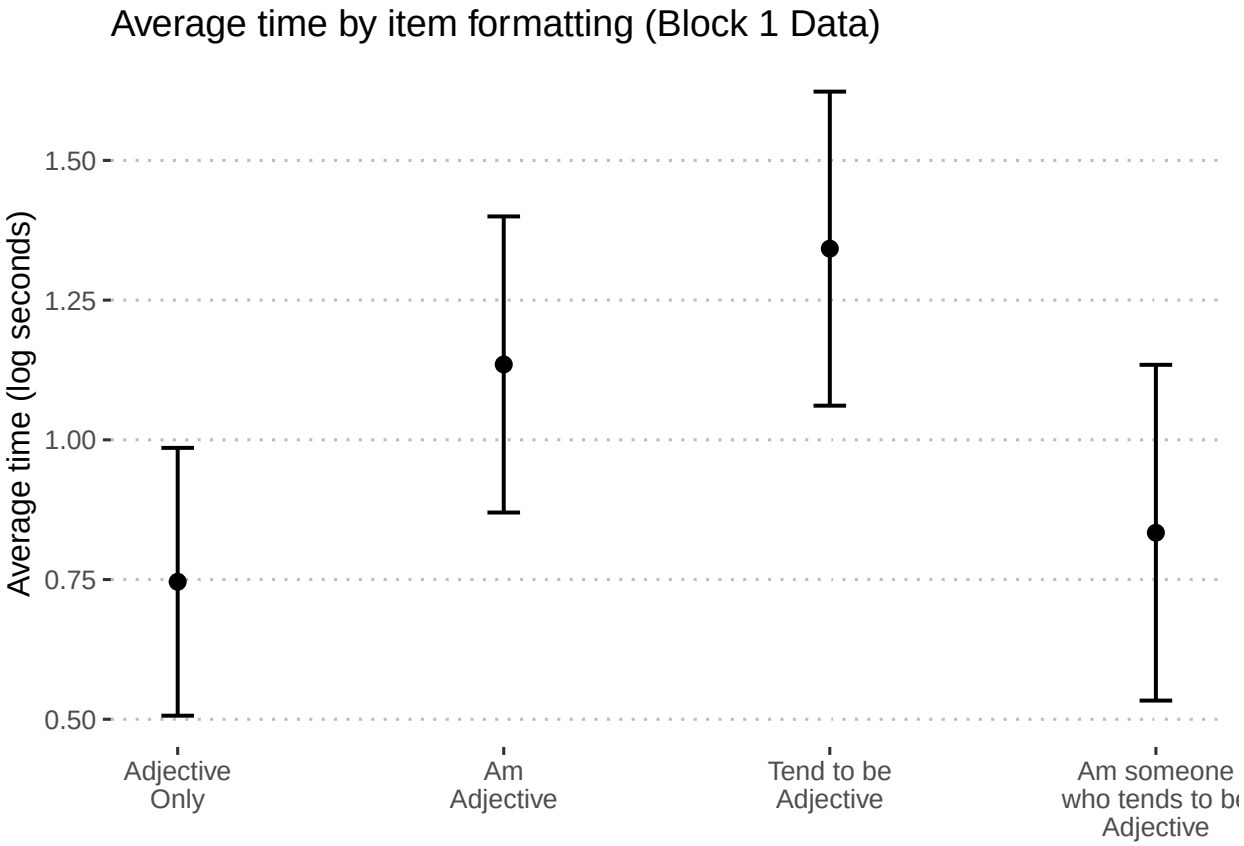


Figure 25: Predicted seconds (log) on personality items by condition, using only Block 1 data.

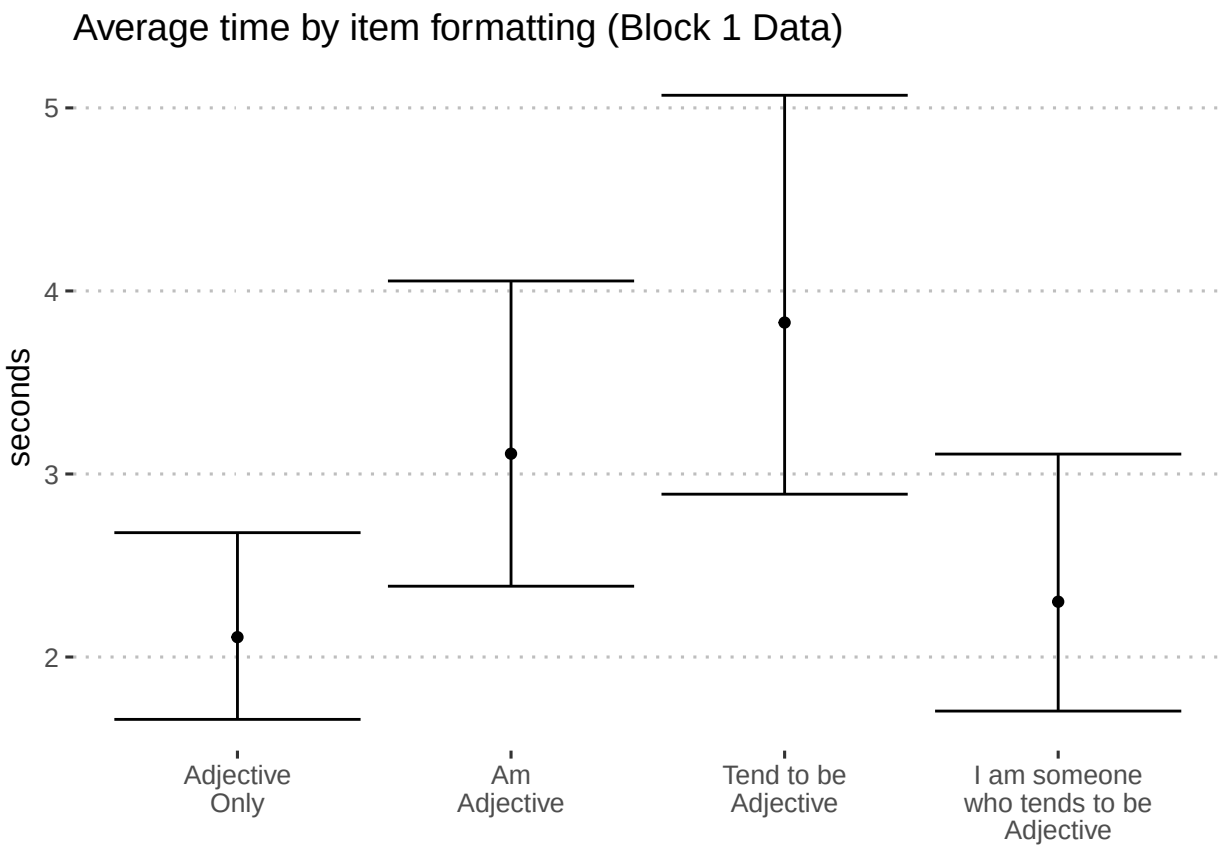


Figure 26: Predicted seconds on personality items by condition, using only Block 1 data.

```
  geom_text(aes(x = 1,
                y = 40,
                label = paste("M =", round(m,2),
                              "\nSD =", round(s,2))),
            data = means_by_group,
            hjust =0,
            vjust = 1) +
  facet_wrap(~format) +
  guides(fill = "none") +
  labs(x = "Log-seconds",
    y = "Number of particpants",
       title = "Distribution of log-seconds by format (Block 1 data)") +
  theme_pubr()
```

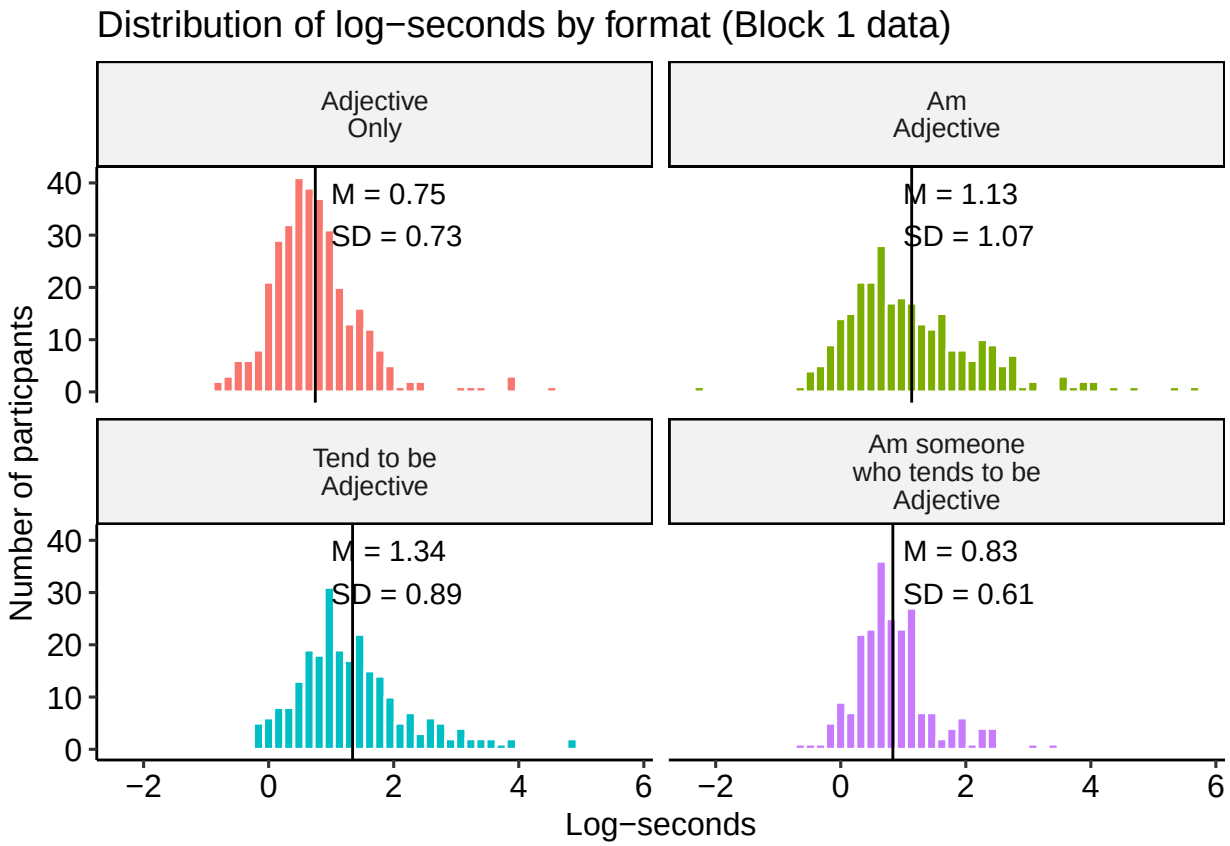


Figure 27: Distribution of time by category, block 1 data only

```
pairs(emmeans(mod.format_b1, "format"), adjust = "holm") %>%
  kable(booktabs = T, digits = c(0, 2,2,1,2,3)) %>%
  kable_styling()
```

6.1.0.1 Pairwise comparisons

contrast	estimate	SE	df	t.ratio	p.value
Adjective Only - Am Adjective	-0.39	0.18	31	-2.13	0.164
Adjective Only - Tend to be Adjective	-0.60	0.19	31	-3.16	0.021
Adjective Only - Am someone who tends to be Adjective	-0.09	0.20	31	-0.45	0.657
Am Adjective - Tend to be Adjective	-0.21	0.20	31	-1.05	0.602
Am Adjective - Am someone who tends to be Adjective	0.30	0.20	31	1.47	0.452
Tend to be Adjective - Am someone who tends to be Adjective	0.51	0.21	31	2.42	0.107

6.1.1 One model for each adjective

We can also repeat this analysis separately for each trait.

```
mod_by_item_b1 = item_block1 %>%
  group_by(item) %>%
  nest() %>%
  mutate(mod = map(data, ~lm(seconds_log~format, data = .))) %>%
  mutate(aov = map(mod, anova)) %>%
  ungroup()

summary_by_item_b1 = mod_by_item_b1 %>%
  mutate(tidy = map(aov, broom::tidy)) %>%
  select(item, tidy) %>%
  unnest(cols = c(tidy)) %>%
  filter(term == "format") %>%
  mutate(reverse = case_when(
    item %in% reverse ~ "Y",
    TRUE ~ "N"
  )) %>%
  mutate(p.adj = p.adjust(p.value, method = "holm"))

summary_by_item_b1 %>%
  mutate(across( starts_with("p"), printp )) %>% # format p-values
  arrange(reverse, item) %>%
  select(item, reverse, sumsq, meansq, df, statistic, p.value, p.adj) %>%
  kable(digits = 2,
        booktabs = T,
        col.names = c("Item", "Reverse\nScored?", "SS", "MS", "df", "F", "raw", "adj"),
        caption = "Format effects on log-seconds by item (block 1 data only)") %>%
  kable_styling()
```

6.1.2 Pairwise t-tests for significant ANOVAs

Here we identify the specific items with significant differences.

```
sig_item_b1 = summary_by_item_b1 %>%
  filter(p.value < .05)

sig_item_b1 = sig_item_b1$item
sig_item_b1
```

Table 17: Format effects on log-seconds by item (block 1 data only)

Item	Reverse Scored?	SS	MS	df	F	raw	adj
active	N	2.02	0.67	3	0.71	.556	> .999
adventurous	N	3.39	1.13	3	3.21	.037	.913
broadminded	N	1.26	0.42	3	0.78	.513	> .999
calm	N	8.79	2.93	3	3.39	.030	.789
caring	N	5.72	1.91	3	4.45	.010	.300
cautious	N	0.12	0.04	3	0.07	.974	> .999
creative	N	8.30	2.77	3	2.80	.056	> .999
curious	N	2.70	0.90	3	2.02	.132	> .999
friendly	N	1.17	0.39	3	0.66	.582	> .999
hardworking	N	6.54	2.18	3	2.55	.074	> .999
helpful	N	2.17	0.72	3	3.44	.029	.773
imaginative	N	2.50	0.83	3	1.31	.288	> .999
intelligent	N	4.15	1.38	3	1.15	.344	> .999
lively	N	3.64	1.21	3	1.42	.255	> .999
organized	N	3.42	1.14	3	1.72	.183	> .999
outgoing	N	1.97	0.66	3	1.02	.398	> .999
responsible	N	4.78	1.59	3	2.79	.057	> .999
selfdisciplined	N	6.81	2.27	3	4.20	.013	.372
softhearted	N	8.16	2.72	3	5.51	.004	.113
sophisticated	N	2.34	0.78	3	0.71	.556	> .999
sympathetic	N	12.92	4.31	3	6.31	.002	.056
talkative	N	0.20	0.07	3	0.21	.890	> .999
thorough	N	6.75	2.25	3	2.18	.111	> .999
thrifty	N	3.15	1.05	3	0.97	.420	> .999
warm	N	3.70	1.23	3	2.63	.068	> .999
careless	Y	0.57	0.19	3	0.29	.832	> .999
impulsive	Y	4.16	1.39	3	1.05	.386	> .999
moody	Y	0.27	0.09	3	0.25	.860	> .999
nervous	Y	6.11	2.04	3	2.54	.074	> .999
reckless	Y	2.88	0.96	3	2.09	.122	> .999
worrying	Y	0.85	0.28	3	0.67	.575	> .999

Table 18: Differences in log-seconds to Helpful by format (Block 1 data only)

Contrast	Difference in means	SE	df	t	p
Adjective Only - Am Adjective	-0.40	0.21	31	-1.96	.236
Adjective Only - Tend to be Adjective	-0.46	0.21	31	-2.18	.184
Adjective Only - Am someone who tends to be Adjective	-0.66	0.22	31	-3.00	.032
Am Adjective - Tend to be Adjective	-0.06	0.22	31	-0.27	.812
Am Adjective - Am someone who tends to be Adjective	-0.26	0.23	31	-1.13	.803
Tend to be Adjective - Am someone who tends to be Adjective	-0.20	0.24	31	-0.84	.812

```
## [1] "adventurous"     "calm"            "caring"          "helpful"
## [5] "selfdisciplined" "softhearted"     "sympathetic"
```

Then we create models for each adjective. We use the `emmeans` package to perform pairwise comparisons, again with a Holm correction on the p-values. We also plot the means and 95% confidence intervals of each mean.

This code will have to be changed after final data collection. It is not self-adapting!

6.1.3 Helpful

```
helpful_model_b1 = item_block1 %>%
  filter(item == "helpful") %>%
  lm(seconds_log~format, data = .)

helpful_em_b1 = emmeans(helpful_model_b1, "format")
pairs(helpful_em_b1, adjust = "holm") %>%
  as_tibble() %>%
  mutate(across( starts_with("p"), printp )) %>% # format p-values
  kable(booktabs = T,
        digits = 2,
        caption = "Differences in log-seconds to Helpful by format (Block 1 data only)",
        col.names = c("Contrast", "Difference in means", "SE", "df", "t", "p")) %>%
  kable_styling()
```

```
plot_model(helpful_model_b1, type = "pred", terms = c("format"))
```

6.1.4 Caring

```
caring_model_b1 = item_block1 %>%
  filter(item == "caring") %>%
  lm(seconds_log~format, data = .)

caring_em_b1 = emmeans(caring_model_b1, "format")
pairs(caring_em_b1, adjust = "holm") %>%
  as_tibble() %>%
```

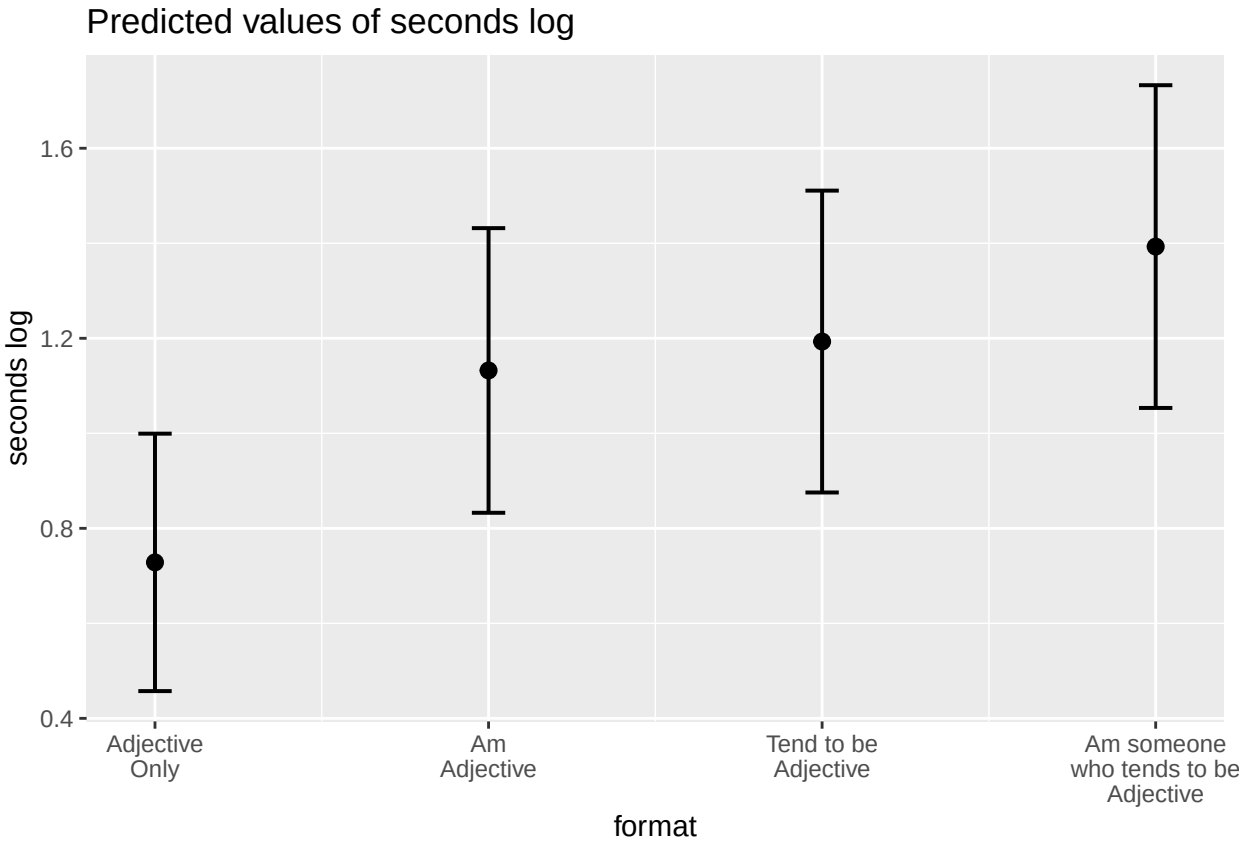


Figure 28: Average log-seconds to "helpful" by format (block 1 data only)

Table 19: Differences in log-seconds to Caring by format (Block 1 data only)

Contrast	Difference in means	SE	df	t	p
Adjective Only - Am Adjective	-1.03	0.29	31	-3.51	.008
Adjective Only - Tend to be Adjective	-0.66	0.30	31	-2.17	.189
Adjective Only - Am someone who tends to be Adjective	-0.32	0.32	31	-1.02	.750
Am Adjective - Tend to be Adjective	0.37	0.32	31	1.17	.750
Am Adjective - Am someone who tends to be Adjective	0.71	0.33	31	2.15	.189
Tend to be Adjective - Am someone who tends to be Adjective	0.34	0.34	31	0.99	.750

```
  mutate(across( starts_with("p"), printp )) %>% # format p-values
  kable(booktabs = T,
        digits = 2,
        caption = "Differences in log-seconds to Caring by format (Block 1 data only)",
        col.names = c("Contrast", "Difference in means", "SE", "df", "t", "p")) %>%
  kable_styling()
```

```
plot_model(caring_model_b1, type = "pred", terms = c("format"))
```

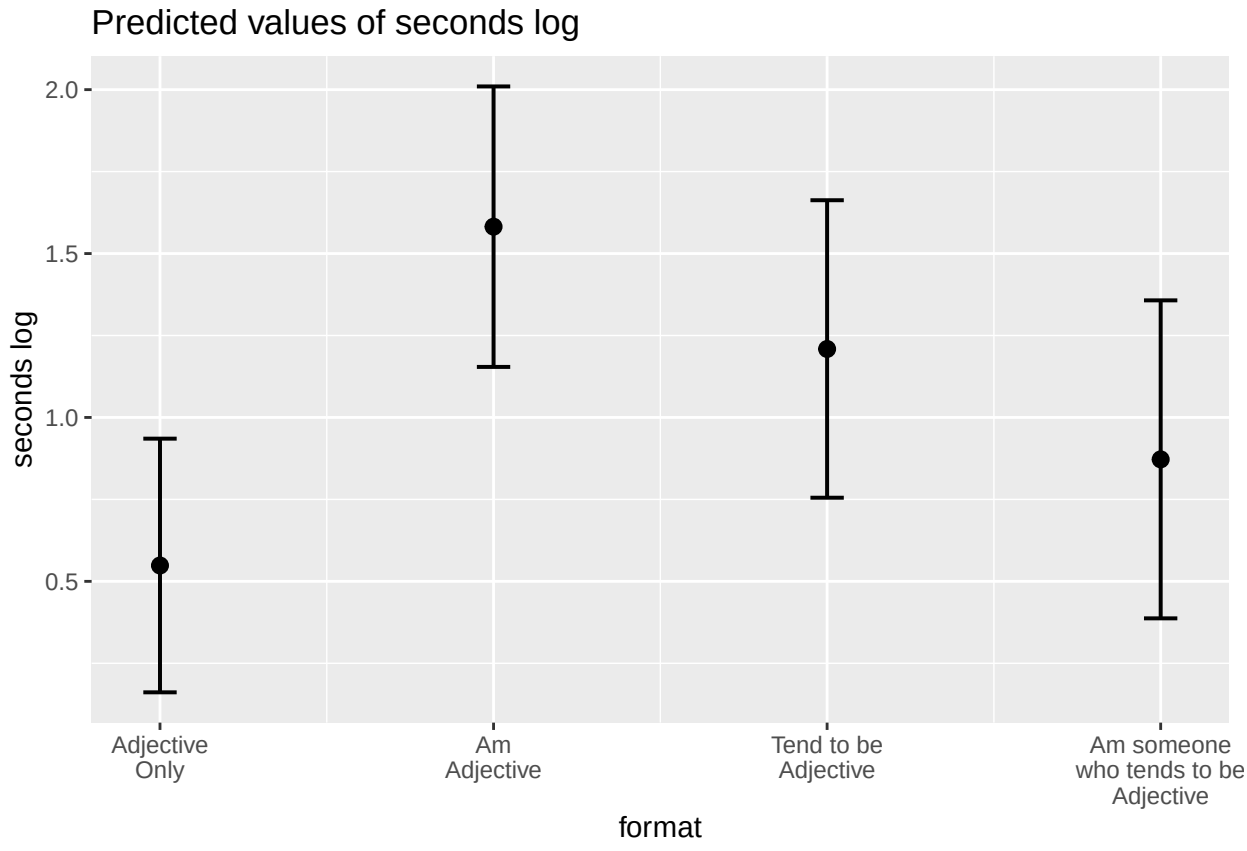


Figure 29: Average log-seconds to "caring" by format (block 1 data only)

Table 20: Differences in log-seconds to Soft-hearted by format (Block 1 data only)

Contrast	Difference in means	SE	df	t	p
Adjective Only - Am Adjective	-1.01	0.32	31	-3.21	.019
Adjective Only - Tend to be Adjective	-0.96	0.33	31	-2.94	.031
Adjective Only - Am someone who tends to be Adjective	-0.06	0.34	31	-0.18	> .999
Am Adjective - Tend to be Adjective	0.05	0.34	31	0.15	> .999
Am Adjective - Am someone who tends to be Adjective	0.95	0.35	31	2.69	.046
Tend to be Adjective - Am someone who tends to be Adjective	0.90	0.36	31	2.47	.057

6.1.5 Soft-hearted

```
softhearted_model_b1 = item_block1 %>%
  filter(item == "softhearted") %>%
  lm(seconds_log~format, data = .)

softhearted_em_b1 = emmeans(softhearted_model_b1, "format")
pairs(softhearted_em_b1, adjust = "holm") %>%
  as_tibble() %>%
  mutate(across( starts_with("p"), printp )) %>% # format p-values
  kable(booktabs = T,
        digits = 2,
        caption = "Differences in log-seconds to Soft-hearted by format (Block 1 data only)",
        col.names = c("Contrast", "Difference in means", "SE", "df", "t", "p")) %>%
  kable_styling()
```

```
plot_model(softhearted_model_b1, type = "pred", terms = c("format"))
```

6.1.6 Calm

```
calm_model_b1 = item_block1 %>%
  filter(item == "calm") %>%
  lm(seconds_log~format, data = .)

calm_em_b1 = emmeans(calm_model_b1, "format")
pairs(calm_em_b1, adjust = "holm") %>%
  as_tibble() %>%
  mutate(across( starts_with("p"), printp )) %>% # format p-values
  kable(booktabs = T,
        digits = 2,
        caption = "Differences in log-seconds to Calm by format (Block 1 data only)",
        col.names = c("Contrast", "Difference in means", "SE", "df", "t", "p")) %>%
  kable_styling()
```

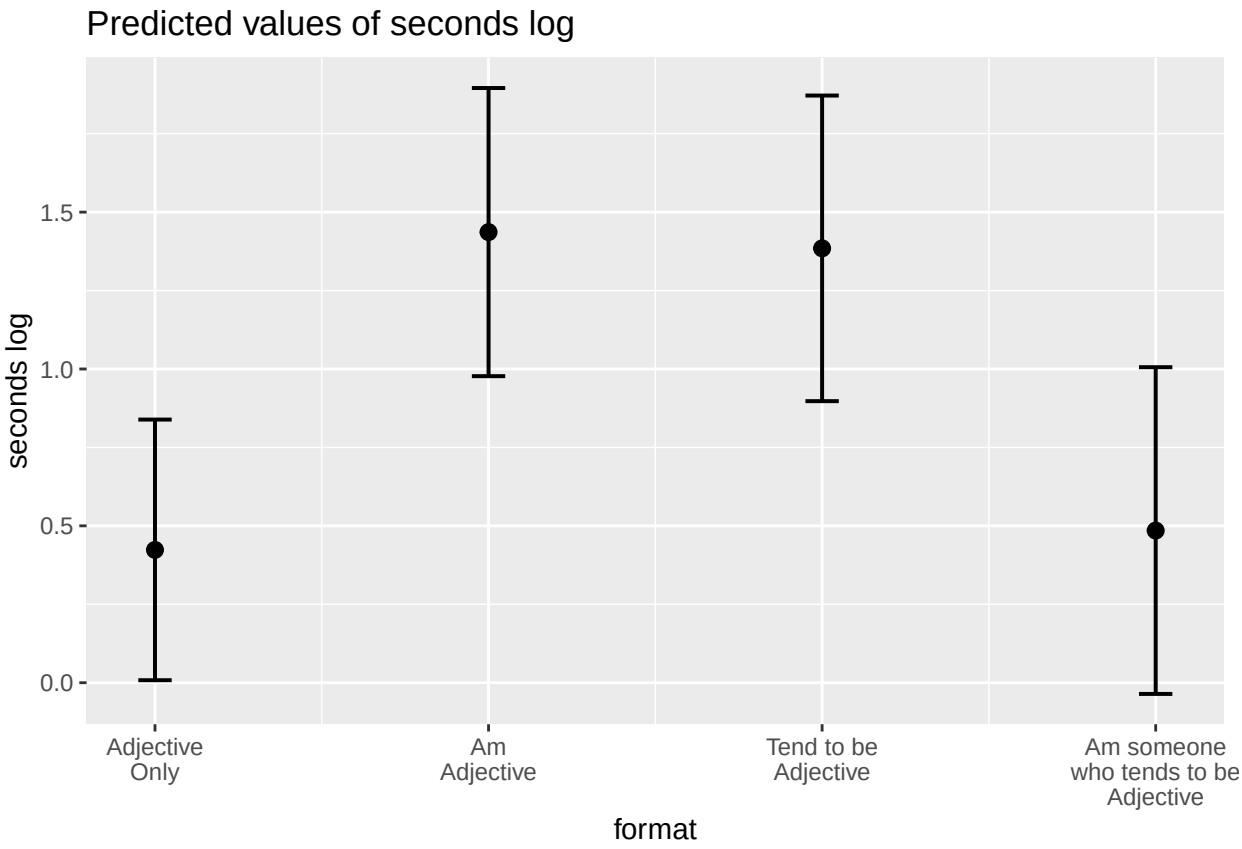


Figure 30: Average log-seconds to "softhearted" by format (block 1 data only)

Table 21: Differences in log-seconds to Calm by format (Block 1 data only)

Contrast	Difference in means	SE	df	t	p
Adjective Only - Am Adjective	-1.07	0.42	31	-2.55	.095
Adjective Only - Tend to be Adjective	-0.91	0.43	31	-2.10	.175
Adjective Only - Am someone who tends to be Adjective	0.01	0.45	31	0.02	$> .999$
Am Adjective - Tend to be Adjective	0.16	0.45	31	0.35	$> .999$
Am Adjective - Am someone who tends to be Adjective	1.08	0.47	31	2.30	.142
Tend to be Adjective - Am someone who tends to be Adjective	0.92	0.48	31	1.91	.197

```
plot_model(calm_model_b1, type = "pred", terms = c("format"))
```

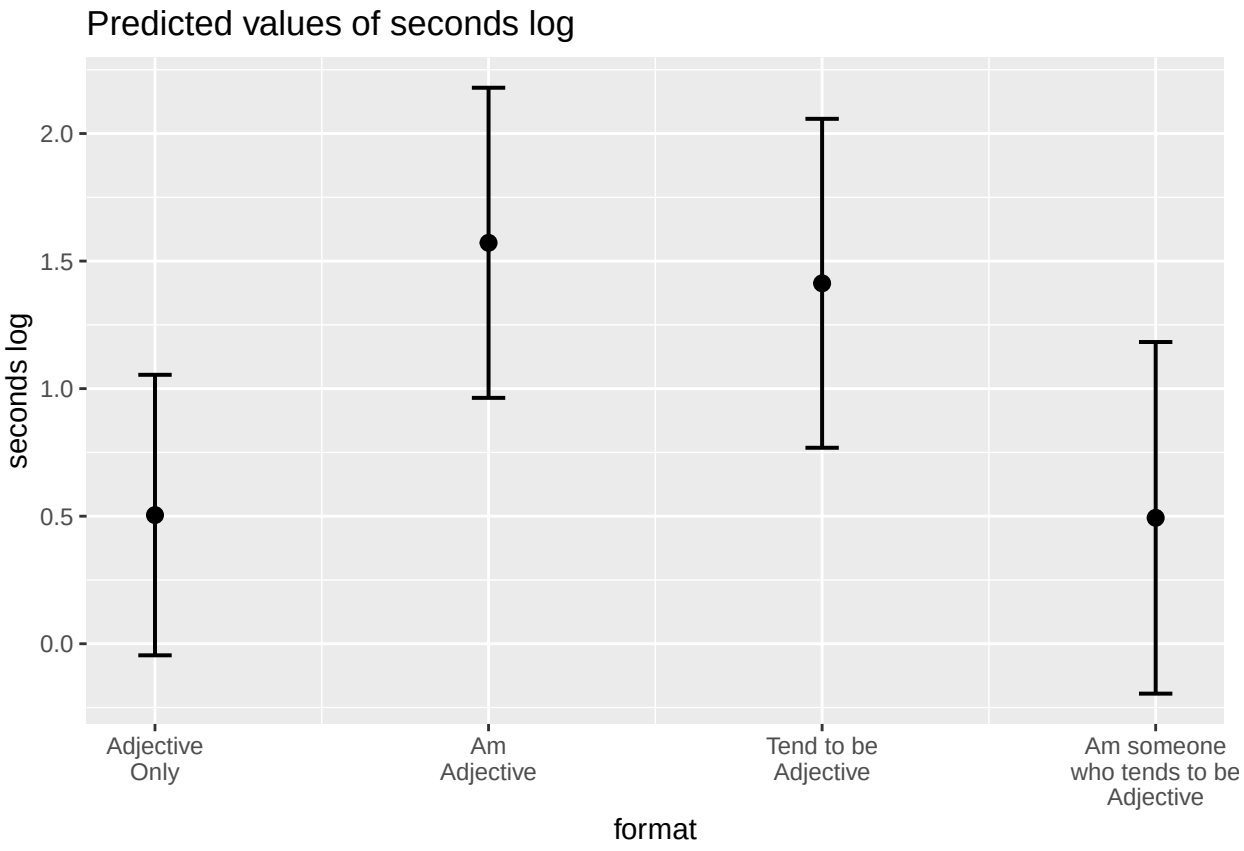


Figure 31: Average log-seconds to "calm" by format (block 1 data only)

6.1.7 Sympathetic

```
sympathetic_model_b1 = item_block1 %>%
  filter(item == "sympathetic") %>%
  lm(seconds_log~format, data = .)

sympathetic_em_b1 = emmeans(sympathetic_model_b1, "format")
pairs(sympathetic_em_b1, adjust = "holm") %>%
   as_tibble() %>%
  mutate(across( starts_with("p"), printp )) %>% # format p-values
  kable(booktabs = T,
        digits = 2,
        caption = "Differences in log-seconds to Sympathetic by format (Block 1 data only)",
        col.names = c("Contrast", "Difference in means", "SE", "df", "t", "p")) %>%
  kable_styling()
```

Table 22: Differences in log-seconds to Sympathetic by format (Block 1 data only)

Contrast	Difference in means	SE	df	t	p
Adjective Only - Am Adjective	-0.52	0.37	31	-1.39	.350
Adjective Only - Tend to be Adjective	-1.45	0.38	31	-3.77	.004
Adjective Only - Am someone who tends to be Adjective	0.18	0.40	31	0.44	.663
Am Adjective - Tend to be Adjective	-0.93	0.40	31	-2.32	.109
Am Adjective - Am someone who tends to be Adjective	0.69	0.42	31	1.66	.320
Tend to be Adjective - Am someone who tends to be Adjective	1.62	0.43	31	3.79	.004

```
plot_model(sympathetic_model_b1, type = "pred", terms = c("format"))
```

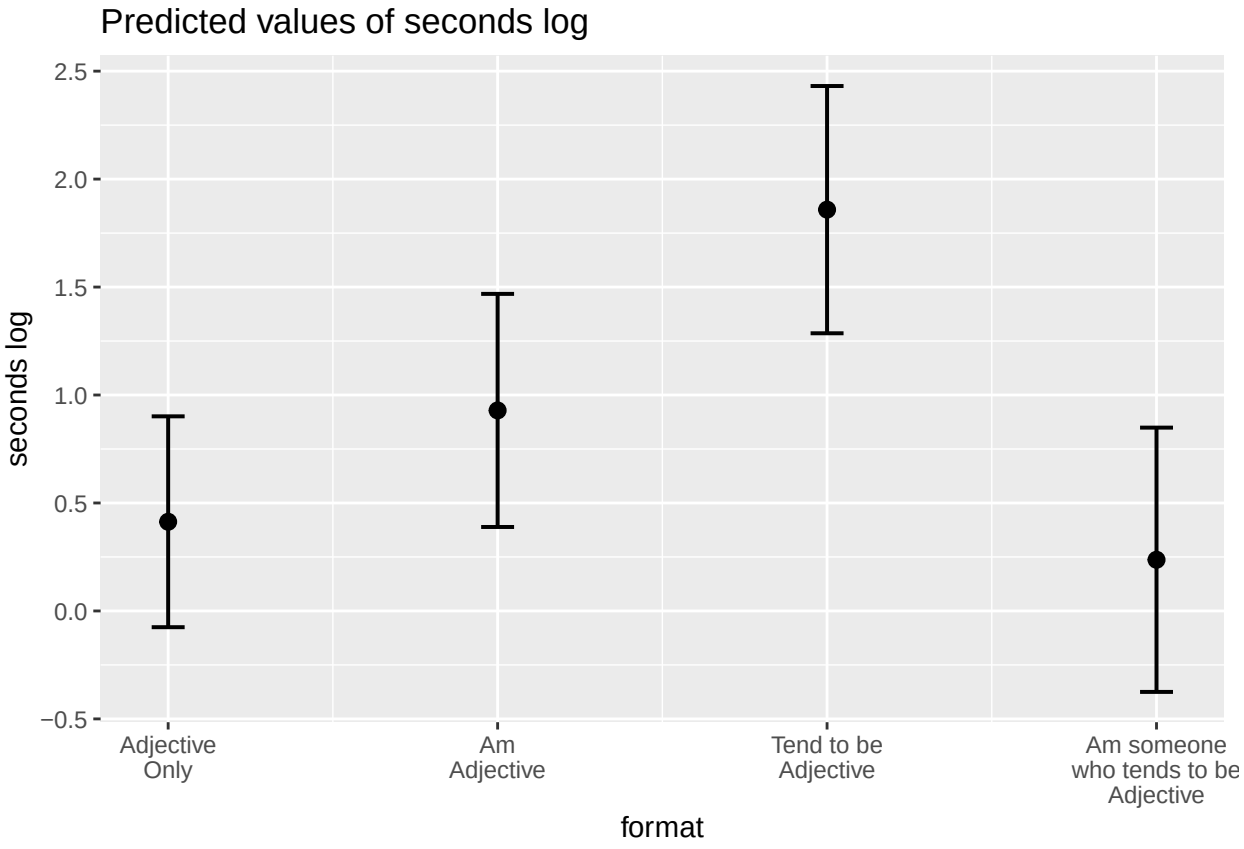


Figure 32: Average log-seconds to "sympathetic" by format (block 1 data only)

6.1.8 Adventurous

```
adventurous_model_b1 = item_block1 %>%
  filter(item == "adventurous") %>%
  lm(seconds_log~format, data = .)
```

Table 23: Differences in log-seconds to Adventurous by format (Block 1 data only)

Contrast	Difference in means	SE	df	t	p
Adjective Only - Am Adjective	-0.12	0.27	31	-0.46	> .999
Adjective Only - Tend to be Adjective	-0.82	0.28	31	-2.96	.035
Adjective Only - Am someone who tends to be Adjective	-0.27	0.29	31	-0.93	> .999
Am Adjective - Tend to be Adjective	-0.69	0.29	31	-2.41	.111
Am Adjective - Am someone who tends to be Adjective	-0.14	0.30	31	-0.48	> .999
Tend to be Adjective - Am someone who tends to be Adjective	0.55	0.31	31	1.79	.332

```
adventurous_em_b1 = emmeans(adventurous_model_b1, "format")
pairs(adventurous_em_b1, adjust = "holm") %>%
   as_tibble() %>%
  mutate(across( starts_with("p"), printp )) %>% # format p-values
  kable(booktabs = T,
        digits = 2,
        caption = "Differences in log-seconds to Adventurous by format (Block 1 data only)",
        col.names = c("Contrast", "Difference in means", "SE", "df", "t", "p")) %>%
  kable_styling()
```

```
plot_model(adventurous_model_b1, type = "pred", terms = c("format"))
```

6.2 Analysis: Block 1 and Block 2

We used a multilevel model, nesting log-seconds within participant to account for dependence. Our primary predictor was format. Here, we use data from blocks 1 and 2.

```
items_12 = items_df %>% filter(block %in% c("1","2"))
```

```
mod.format_b2 = lmer(seconds_log~format + (1|proid),
                     data = items_12)
anova(mod.format_b2)
```

```
## Type III Analysis of Variance Table with Satterthwaite's method
##        Sum Sq Mean Sq NumDF  DenDF F value    Pr(>F)
## format 25.706  8.5685     3 2159.4  12.973 2.128e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
plot_b2 = plot_model(mod.format_b2, type = "pred")

plot_b2$format +
  labs(x = NULL,
       y = "Average log-seconds",
       title = "Average responses by item formatting (Block 1 and Block 2)") +
  theme_pubclean()
```

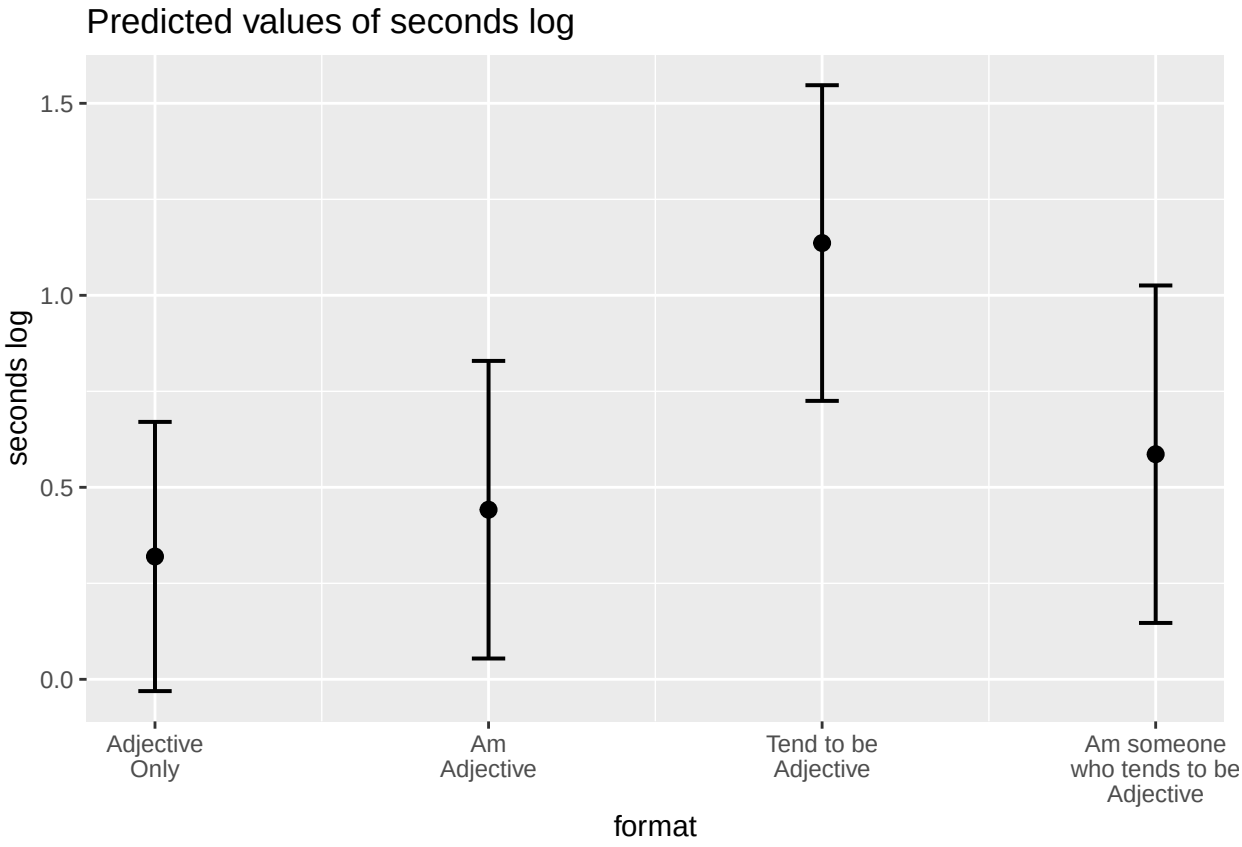


Figure 33: Average log-seconds to "adventurous" by format (block 1 data only)

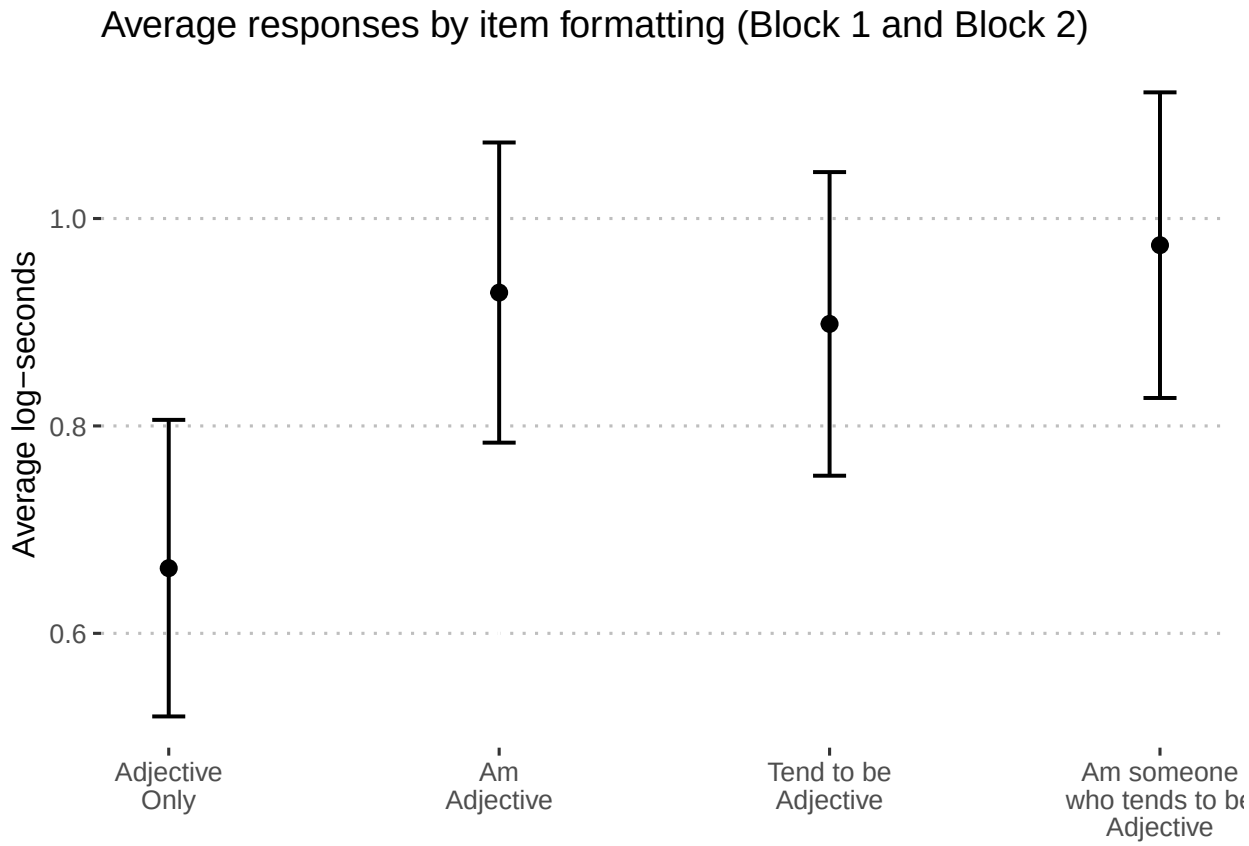


Figure 34: Predicted log-seconds on personality items by condition, using only Block 1 data.

```
plot_b2$format$data %>%
  mutate(predicted = exp(predicted),
         conf.low = exp(conf.low),
         conf.high = exp(conf.high)) %>%
  mutate(x = factor(x,
                    labels = c("Adjective\nOnly",
                               "Am\nAdjective",
                               "Tend to be\nAdjective",
                               "I am someone\nwho tends to be\nAdjective"))) %>%
  ggplot(aes(x = x, y = predicted)) +
  geom_point() +
  geom_errorbar(aes(ymin = conf.low, ymax = conf.high)) +
  labs(x = NULL, y = "seconds", title = "Average time by item formatting (Block 1 and Block 2)") +
  theme_pubclean()
```

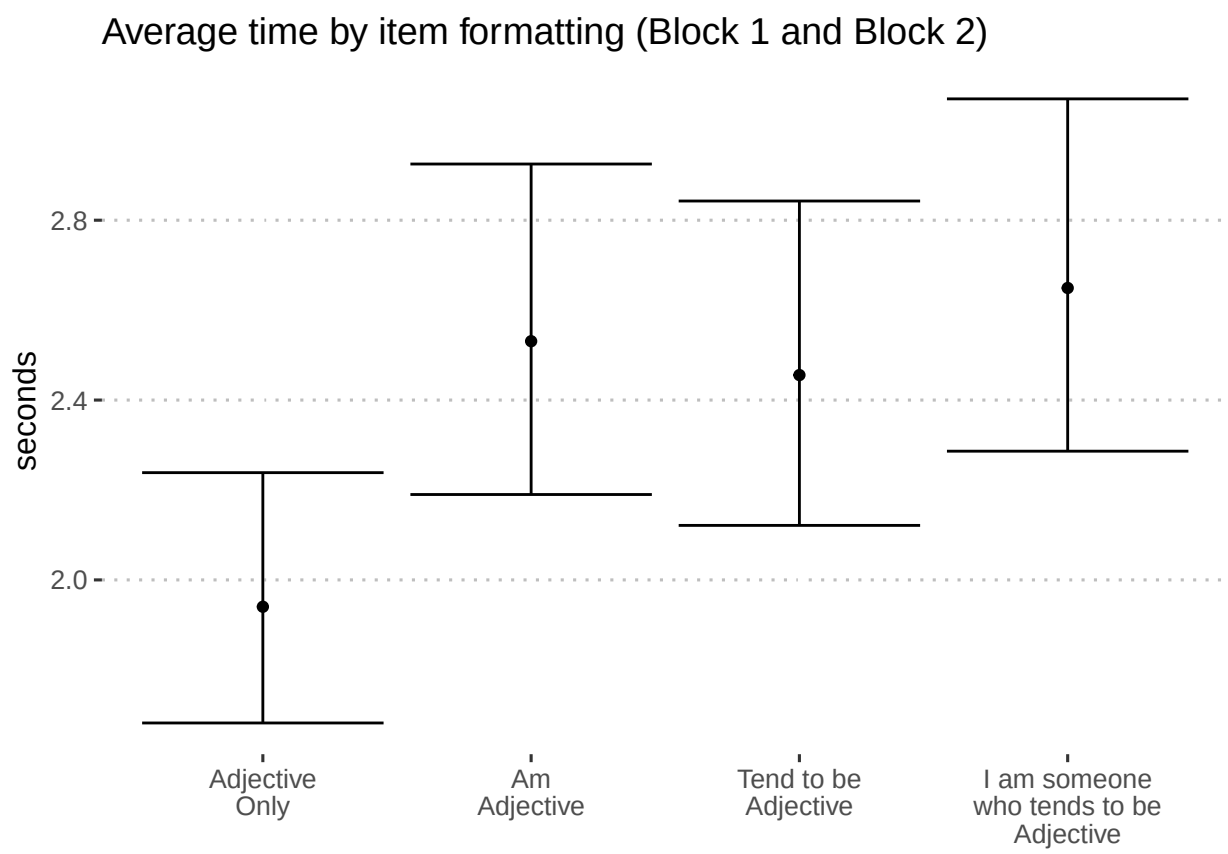


Figure 35: Predicted seconds on personality items by condition, using only Block 1 data.

```
means_by_group = items_12 %>%
  group_by(format) %>%
  summarise(m = mean(seconds_log),
            s = sd(seconds_log))

items_12 %>%
  ggplot(aes(x = seconds_log, fill = format)) +
  geom_histogram(bins = 100, color = "white") +
```

```
  geom_vline(aes(xintercept = m), data = means_by_group) +
  geom_text(aes(x = 1,
                y = 50,
                label = paste("M =", round(m,2),
                              "\nSD =", round(s,2))),
            data = means_by_group,
            hjust =0,
            vjust = 1) +
  facet_wrap(~format) +
  guides(fill = "none") +
  labs(y = "Number of particpants",
       title = "Distribution of responses by format (Block 1 and Block 2)") +
  theme_pubr()
```

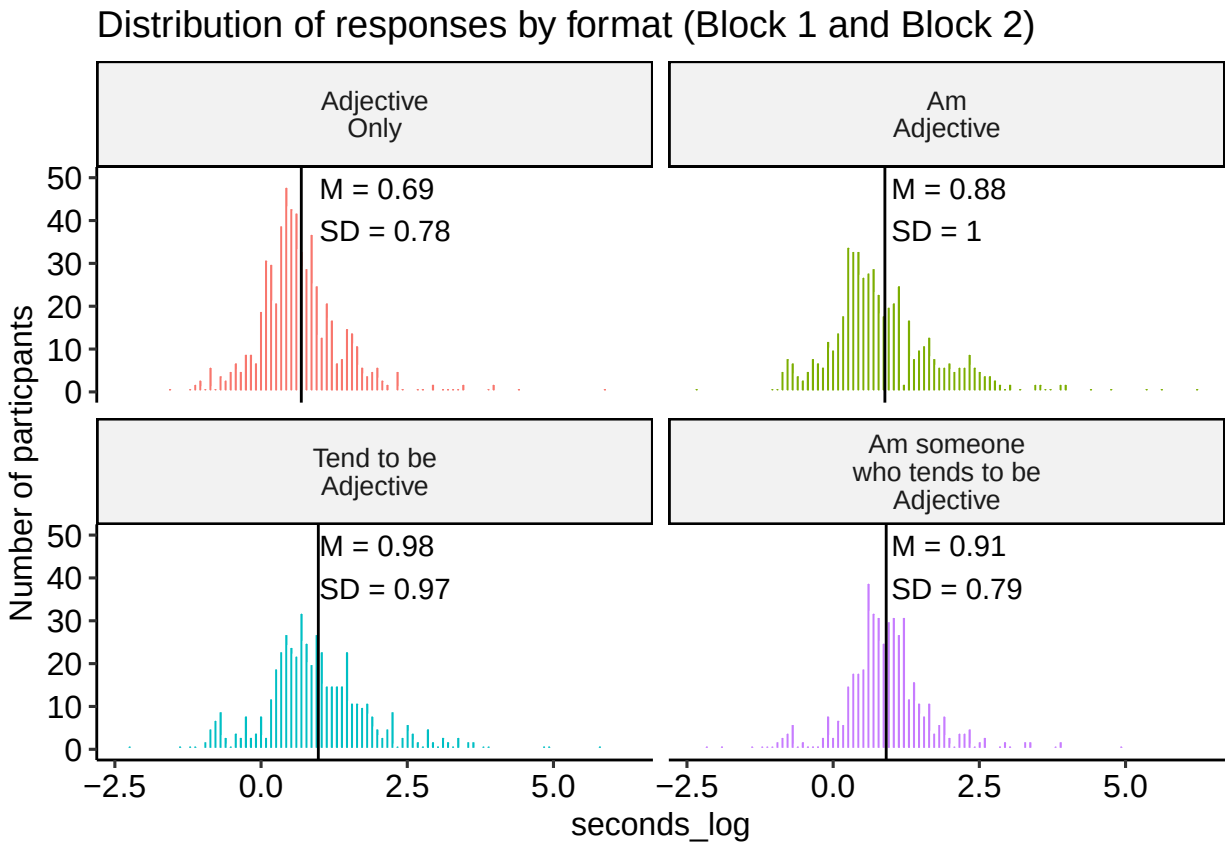


Figure 36: Distribution of log-seconds by category, block 1 and block 2

```
pairs(emmeans(mod.format_b2, "format"), adjust = "holm") %>%
  kable(booktabs = T, digits = c(0, 2,2,1,2,3)) %>%
  kable_styling()
```

6.2.0.1 Pairwise comparisons

contrast	estimate	SE	df	t.ratio	p.value
Adjective Only - Am Adjective	-0.27	0.05	2159.5	-4.91	0.000
Adjective Only - Tend to be Adjective	-0.24	0.06	2152.3	-4.20	0.000
Adjective Only - Am someone who tends to be Adjective	-0.31	0.06	2160.7	-5.55	0.000
Am Adjective - Tend to be Adjective	0.03	0.06	2156.8	0.53	0.854
Am Adjective - Am someone who tends to be Adjective	-0.05	0.06	2160.1	-0.79	0.854
Tend to be Adjective - Am someone who tends to be Adjective	-0.08	0.06	2164.2	-1.32	0.565

6.2.1 One model for each adjective

We can also repeat this analysis separately for each trait.

```
mod_by_item_b2 = items_12 %>%
  group_by(item) %>%
  nest() %>%
  mutate(mod = map(data, ~lmer(seconds_log~format + (1|proid), data = .))) %>%
  mutate(aov = map(mod, anova)) %>%
  ungroup()

summary_by_item_b2 = mod_by_item_b2 %>%
  mutate(tidy = map(aov, broom::tidy)) %>%
  select(item, tidy) %>%
  unnest(cols = c(tidy)) %>%
  filter(term == "format") %>%
  mutate(reverse = case_when(
    item %in% reverse ~ "Y",
    TRUE ~ "N"
  )) %>%
  mutate(p.adj = p.adjust(p.value, method = "holm"))

summary_by_item_b2 %>%
  mutate(across( starts_with("p"), printp )) %>% # format p-values
  arrange(reverse, item) %>%
  select(item, reverse, sumsq, meansq, NumDF, DenDF, statistic, p.value, p.adj) %>%
  kable(digits = 2,
        col.names = c("Item", "Reverse\nScored?", "SS", "MS", "df1", "df2", "F", "raw", "adj"),
        booktabs = T,
        caption = "Format effects on log-seconds by item (block 1 data only)") %>%
  kable_styling()
```

6.2.2 Pairwise t-tests for significant ANOVAs

Here we identify the specific items with significant differences.

```
sig_item_b2 = summary_by_item_b2 %>%
  filter(p.value < .05)

sig_item_b2 = sig_item_b2$item
sig_item_b2
```

```
## [1] "adventurous" "caring"      "helpful"
```

Table 24: Format effects on log-seconds by item (block 1 data only)

Item	Reverse Scored?	SS	MS	df1	df2	F	raw	adj
active	N	1.82	0.61	3	66.00	0.81	.493	$> .999$
adventurous	N	4.20	1.40	3	59.34	2.99	.038	$> .999$
broadminded	N	1.31	0.44	3	58.18	1.04	.381	$> .999$
calm	N	5.06	1.69	3	66.00	2.02	.119	$> .999$
caring	N	5.00	1.67	3	66.00	3.01	.036	$> .999$
cautious	N	2.13	0.71	3	51.97	1.14	.342	$> .999$
creative	N	1.97	0.66	3	66.00	0.70	.555	$> .999$
curious	N	2.31	0.77	3	62.52	1.40	.252	$> .999$
friendly	N	0.41	0.14	3	65.49	0.18	.909	$> .999$
hardworking	N	5.53	1.84	3	66.00	2.74	.050	$> .999$
helpful	N	2.04	0.68	3	65.91	2.92	.040	$> .999$
imaginative	N	4.68	1.56	3	66.00	1.98	.126	$> .999$
intelligent	N	3.20	1.07	3	66.00	1.02	.389	$> .999$
lively	N	2.25	0.75	3	66.00	0.70	.558	$> .999$
organized	N	2.16	0.72	3	66.00	1.02	.392	$> .999$
outgoing	N	2.85	0.95	3	60.19	1.46	.233	$> .999$
responsible	N	2.67	0.89	3	66.00	1.57	.205	$> .999$
selfdisciplined	N	1.32	0.44	3	66.00	0.66	.580	$> .999$
softhearted	N	2.34	0.78	3	66.00	1.08	.364	$> .999$
sophisticated	N	1.95	0.65	3	66.00	0.67	.572	$> .999$
sympathetic	N	4.19	1.40	3	66.00	1.75	.166	$> .999$
talkative	N	0.22	0.07	3	45.63	0.34	.794	$> .999$
thorough	N	1.54	0.51	3	60.65	0.66	.580	$> .999$
thrifty	N	2.16	0.72	3	55.08	1.48	.230	$> .999$
warm	N	0.81	0.27	3	66.00	0.42	.738	$> .999$
careless	Y	2.03	0.68	3	66.00	1.10	.356	$> .999$
impulsive	Y	2.81	0.94	3	62.76	0.96	.418	$> .999$
moody	Y	1.40	0.47	3	65.34	0.96	.416	$> .999$
nervous	Y	6.47	2.16	3	66.00	2.42	.074	$> .999$
reckless	Y	1.60	0.53	3	63.75	0.53	.662	$> .999$
worrying	Y	0.76	0.25	3	66.00	0.45	.717	$> .999$

Table 25: Differences in log-seconds to Helpful by format (Block 1 and Block 2)

Contrast	Difference in means	SE	df	t	p
Adjective Only - Am Adjective	-0.28	0.17	65.88	-1.67	.414
Adjective Only - Tend to be Adjective	-0.32	0.18	65.98	-1.76	.414
Adjective Only - Am someone who tends to be Adjective	-0.50	0.18	65.98	-2.80	.040
Am Adjective - Tend to be Adjective	-0.04	0.18	65.77	-0.19	.848
Am Adjective - Am someone who tends to be Adjective	-0.21	0.18	65.08	-1.20	.702
Tend to be Adjective - Am someone who tends to be Adjective	-0.18	0.19	65.85	-0.94	.702

Then we create models for each adjective. We use the `emmeans` package to perform pairwise comparisons, again with a Holm correction on the p-values. We also plot the means and 95% confidence intervals of each mean.

This code will have to be changed after final data collection. It is not self-adapting!

6.2.3 Helfpul

```
helpful_model_b2 = items_12 %>%
  filter(item == "helpful") %>%
  lmer(seconds_log~format + (1|proid),
                               data = .)

helpful_em_b2 = emmeans(helpful_model_b2, "format")
pairs(helpful_em_b2, adjust = "holm") %>%
  as_tibble() %>%
  mutate(across( starts_with("p"), printp )) %>% # format p-values
  kable(booktabs = T,
        digits = 2,
        caption = "Differences in log-seconds to Helpful by format (Block 1 and Block 2)",
        col.names = c("Contrast", "Difference in means", "SE", "df", "t", "p")) %>%
  kable_styling()
```

```
plot_model(helpful_model_b2, type = "pred", terms = c("format"))
```

6.2.4 Caring

```
caring_model_b2 = items_12 %>%
  filter(item == "caring") %>%
  lmer(seconds_log~format + (1|proid),
                               data = .)

caring_em_b2 = emmeans(caring_model_b2, "format")
pairs(caring_em_b2, adjust = "holm") %>%
  as_tibble() %>%
  mutate(across( starts_with("p"), printp )) %>% # format p-values
```

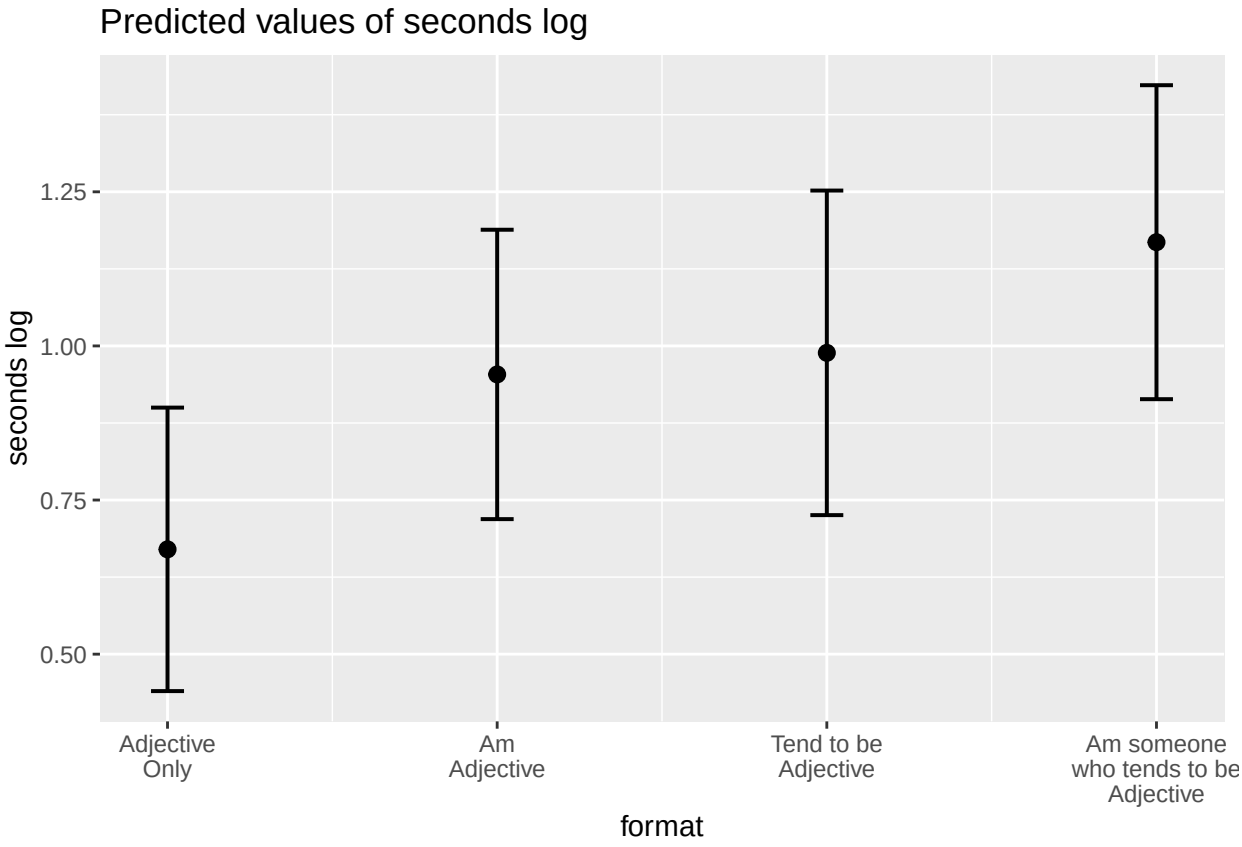


Figure 37: Average log-seconds to "helpful" by format (Block 1 and Block 2)

Table 26: Differences in log-seconds to Caring by format (Block 1 and Block 2)

Contrast	Difference in means	SE	df	t	p
Adjective Only - Am Adjective	-0.65	0.25	65.21	-2.56	.076
Adjective Only - Tend to be Adjective	-0.60	0.26	63.58	-2.34	.113
Adjective Only - Am someone who tends to be Adjective	-0.29	0.25	65.99	-1.15	.757
Am Adjective - Tend to be Adjective	0.05	0.27	65.96	0.18	.860
Am Adjective - Am someone who tends to be Adjective	0.36	0.26	65.29	1.35	.723
Tend to be Adjective - Am someone who tends to be Adjective	0.31	0.27	64.65	1.16	.757

```
kable(booktabs = T,
      digits = 2,
      caption = "Differences in log-seconds to Caring by format (Block 1 and Block 2)",
      col.names = c("Contrast", "Difference in means", "SE", "df", "t", "p")) %>%
kable_styling()
```

```
plot_model(caring_model_b2, type = "pred", terms = c("format"))
```

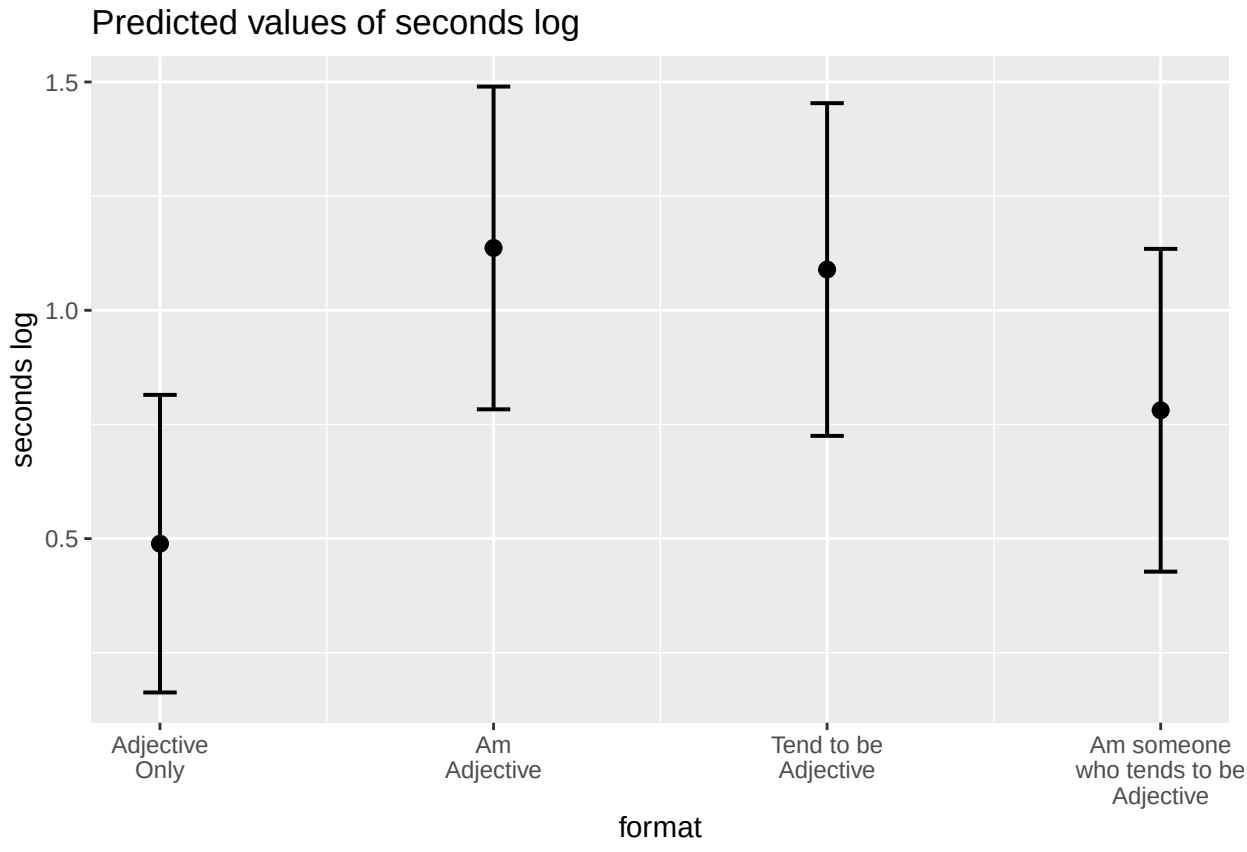


Figure 38: Average log-seconds to "caring" by format (Block 1 and Block 2)

Table 27: Differences in log-seconds to Adventurous by format (Block 1 and Block 2)

Contrast	Difference in means	SE	df	t	p
Adjective Only - Am Adjective	-0.31	0.26	59.39	-1.20	.724
Adjective Only - Tend to be Adjective	-0.68	0.25	55.91	-2.69	.057
Adjective Only - Am someone who tends to be Adjective	-0.57	0.27	65.51	-2.13	.185
Am Adjective - Tend to be Adjective	-0.36	0.27	55.75	-1.35	.724
Am Adjective - Am someone who tends to be Adjective	-0.26	0.28	62.04	-0.93	.724
Tend to be Adjective - Am someone who tends to be Adjective	0.10	0.28	63.89	0.37	.724

6.2.5 Adventurous

```
adventurous_model_b2 = items_12 %>%
  filter(item == "adventurous") %>%
  lmer(seconds_log~format + (1|proid),
                               data = .)

adventurous_em_b2 = emmeans(adventurous_model_b2, "format")
pairs(adventurous_em_b2, adjust = "holm") %>%
  as_tibble() %>%
  mutate(across( starts_with("p"), printp )) %>% # format p-values
  kable(booktabs = T,
        digits = 2,
        caption = "Differences in log-seconds to Adventurous by format (Block 1 and Block 2)",
        col.names = c("Contrast", "Difference in means", "SE", "df", "t", "p")) %>%
  kable_styling()
```

```
plot_model(adventurous_model_b2, type = "pred", terms = c("format"))
```

6.3 Analysis: Account for memory effects

```
mod.format_mem = lmer(seconds_log~format*delayed_memory + (1|proid),
                  data = items_12)
anova(mod.format_mem)
```

```
## Type III Analysis of Variance Table with Satterthwaite's method
##                         Sum Sq Mean Sq NumDF   DenDF F value    Pr(>F)
## format                 12.2415  4.0805     3 2158.60  6.2430 0.0003226 ***
## delayed_memory          0.0753  0.0753     1   33.06  0.1151 0.7365052
## format:delayed_memory 16.5149  5.5050     3 2159.06  8.4223 1.453e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(mod.format_mem)
```

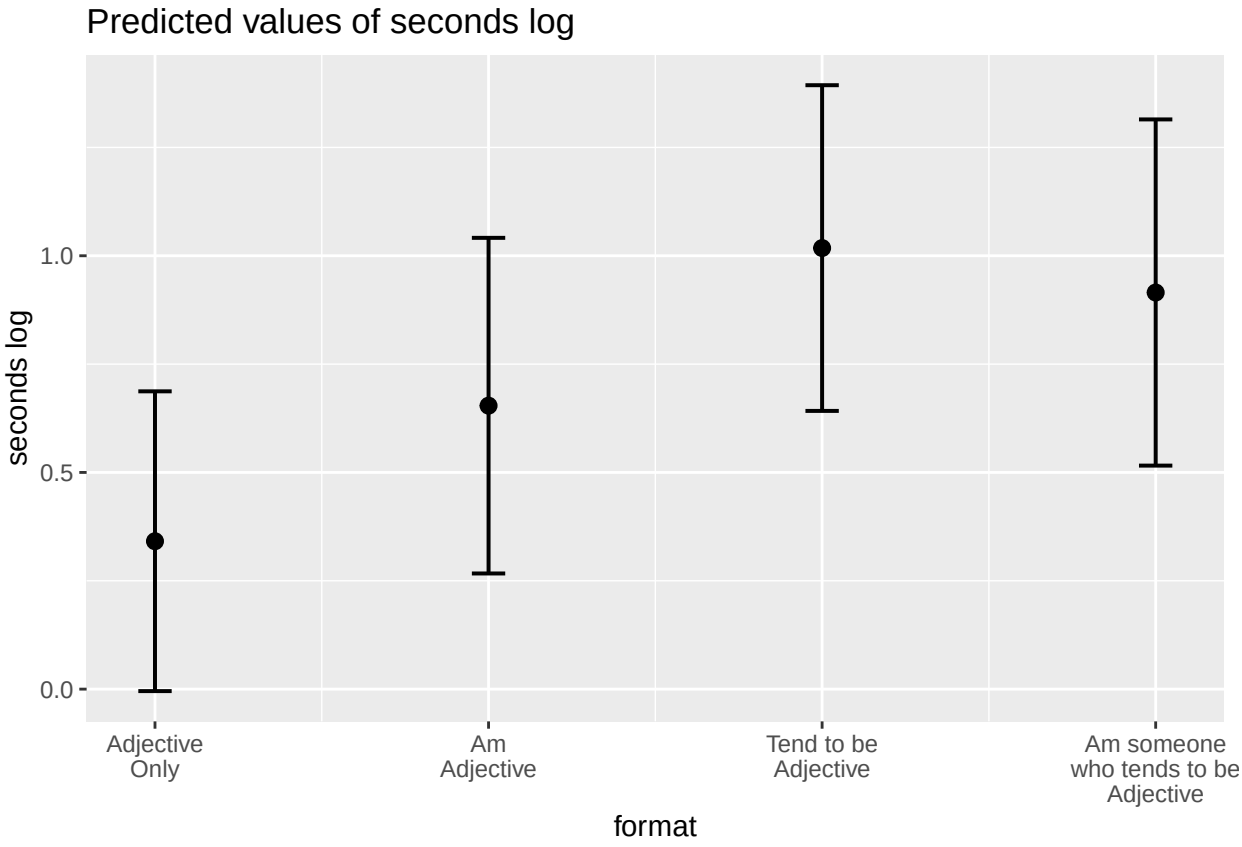


Figure 39: Average log-seconds to "adventurous" by format (Block 1 and Block 2)

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: seconds_log ~ format * delayed_memory + (1 | proid)
##    Data: items_12
##
## REML criterion at convergence: 5362.1
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.9500 -0.5434 -0.1473  0.3649  6.9573
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  proid    (Intercept) 0.1438   0.3792
##  Residual             0.6536   0.8085
## Number of obs: 2170, groups:  proid, 35
##
## Fixed effects:
##                                                          Estimate
## (Intercept)                                             7.227e-01
## formatAm\nAdjective                                    -1.157e-01
## formatTend to be\nAdjective                             1.567e-01
## formatAm someone\nwho tends to be\nAdjective            3.587e-01
## delayed_memory                                         -1.258e-02
## formatAm\nAdjective:delayed_memory                      7.284e-02
## formatTend to be\nAdjective:delayed_memory              1.629e-02
## formatAm someone\nwho tends to be\nAdjective:delayed_memory -7.931e-03
##                                                        Std. Error
## (Intercept)                                             1.484e-01
## formatAm\nAdjective                                     1.051e-01
## formatTend to be\nAdjective                             1.110e-01
## formatAm someone\nwho tends to be\nAdjective            1.143e-01
## delayed_memory                                          2.530e-02
## formatAm\nAdjective:delayed_memory                      1.754e-02
## formatTend to be\nAdjective:delayed_memory              1.958e-02
## formatAm someone\nwho tends to be\nAdjective:delayed_memory  1.959e-02
##                                                               df t value
## (Intercept)                                             4.932e+01   4.870
## formatAm\nAdjective                                     2.161e+03  -1.101
## formatTend to be\nAdjective                             2.155e+03   1.412
## formatAm someone\nwho tends to be\nAdjective            2.162e+03   3.137
## delayed_memory                                          5.010e+01  -0.497
## formatAm\nAdjective:delayed_memory                      2.160e+03   4.153
## formatTend to be\nAdjective:delayed_memory              2.151e+03   0.832
## formatAm someone\nwho tends to be\nAdjective:delayed_memory  2.162e+03  -0.405
##                                                        Pr(>|t|)
## (Intercept)                                            1.19e-05 ***
## formatAm\nAdjective                                     0.27123
## formatTend to be\nAdjective                             0.15806
## formatAm someone\nwho tends to be\nAdjective            0.00173 **
## delayed_memory                                          0.62103
## formatAm\nAdjective:delayed_memory                     3.40e-05 ***
## formatTend to be\nAdjective:delayed_memory              0.40538
## formatAm someone\nwho tends to be\nAdjective:delayed_memory  0.68568
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) frmtAA frTtbA frAswttbA dlyd_m frAA:_ fTtbA:
## frmtAmAdjct -0.357
## frmtTndtbAd -0.353  0.477
## frmtAswttbA -0.331  0.454  0.472
## delayd_mmry -0.868  0.315  0.311  0.289
## frmtAAdjc:_  0.323 -0.859 -0.432 -0.413    -0.379
## frmtTtbAd:_  0.301 -0.408 -0.864 -0.399    -0.352  0.494
## frAswttbA:_  0.288 -0.399 -0.411 -0.873    -0.334  0.484  0.462
```

```
plot_mem = plot_model(mod.format_mem,
           type = "pred",
           term = c("format", "delayed_memory[meansd]")) +
  geom_line() +
  labs(x = NULL,
       y = "Average log-seconds") +
  scale_color_discrete("Memory", labels = c("-1SD", "Mean", "+1SD"))+
  theme_pubclean()

plot_mem
```

```
plot_mem$data %>% as_tibble %>%
  mutate(predicted = exp(predicted),
         conf.low = exp(conf.low),
         conf.high = exp(conf.high),
         group_col = factor(group_col, labels = c("Memory\n-1SD", "Memory\nMean", "Memory\n+1SD"))) %>%
  mutate(x = factor(x,
                    labels = c("Adjective\nOnly",
                               "Am\nAdjective",
                               "Tend to be\nAdjective",
                               "I am someone\nwho tends to be\nAdjective"))) %>%
  ggplot(aes(x = x, y = predicted, color = group_col)) +
  geom_point() +
  geom_errorbar(aes(ymin = conf.low, ymax = conf.high)) +
  labs(x = NULL, y = "seconds", title = "Average time by item formatting (Block 1 and Block 2)") +
  facet_wrap(~group_col) +
  guides(color = "none") +
  theme_pubclean()
```

6.3.1 One model for each adjective

```
mod_by_item_mem = items_12 %>%
  group_by(item) %>%
  nest() %>%
  mutate(mod = map(data, ~lmer(seconds_log~format*delayed_memory + (1|proid), data = .))) %>%
  mutate(aov = map(mod, anova)) %>%
  ungroup()
```

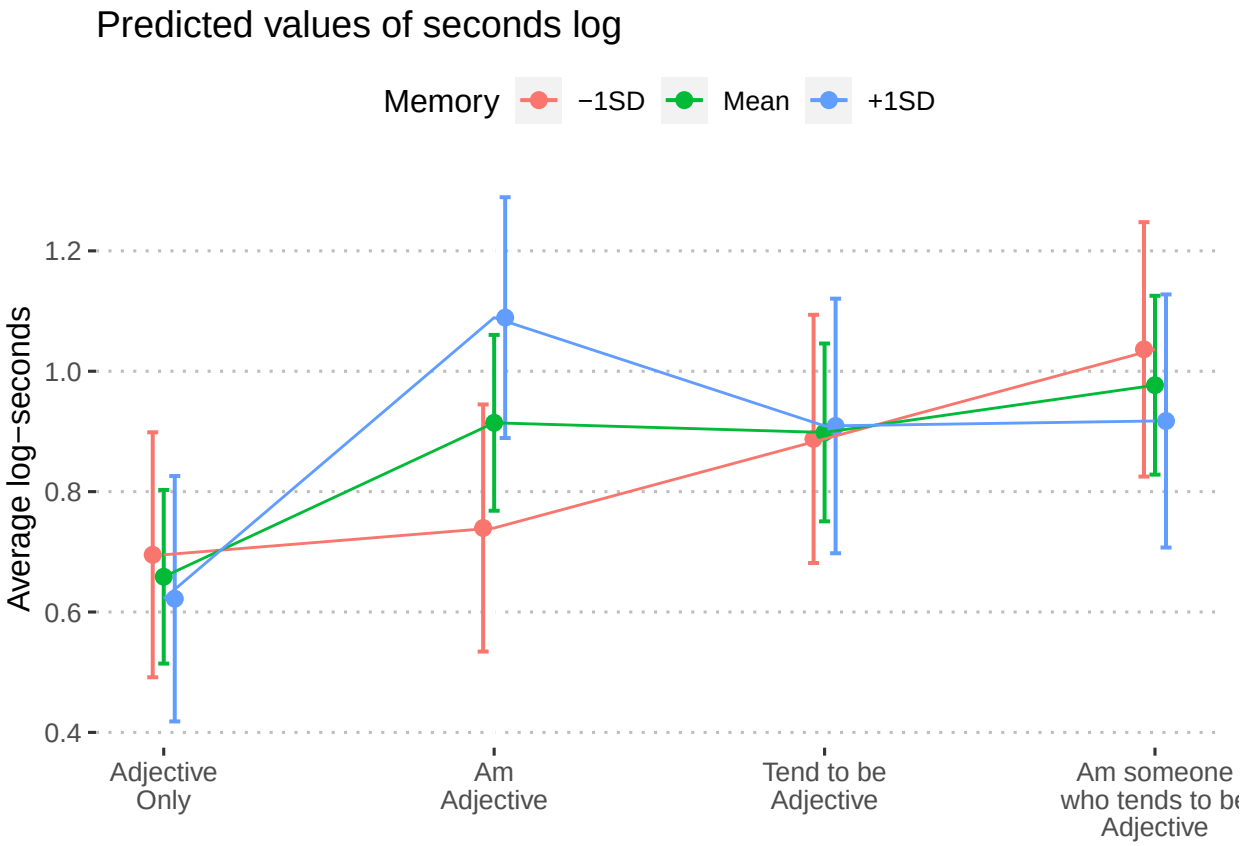


Figure 40: Predicted log-seconds on personality items by condition after controlling for delayed_memory.

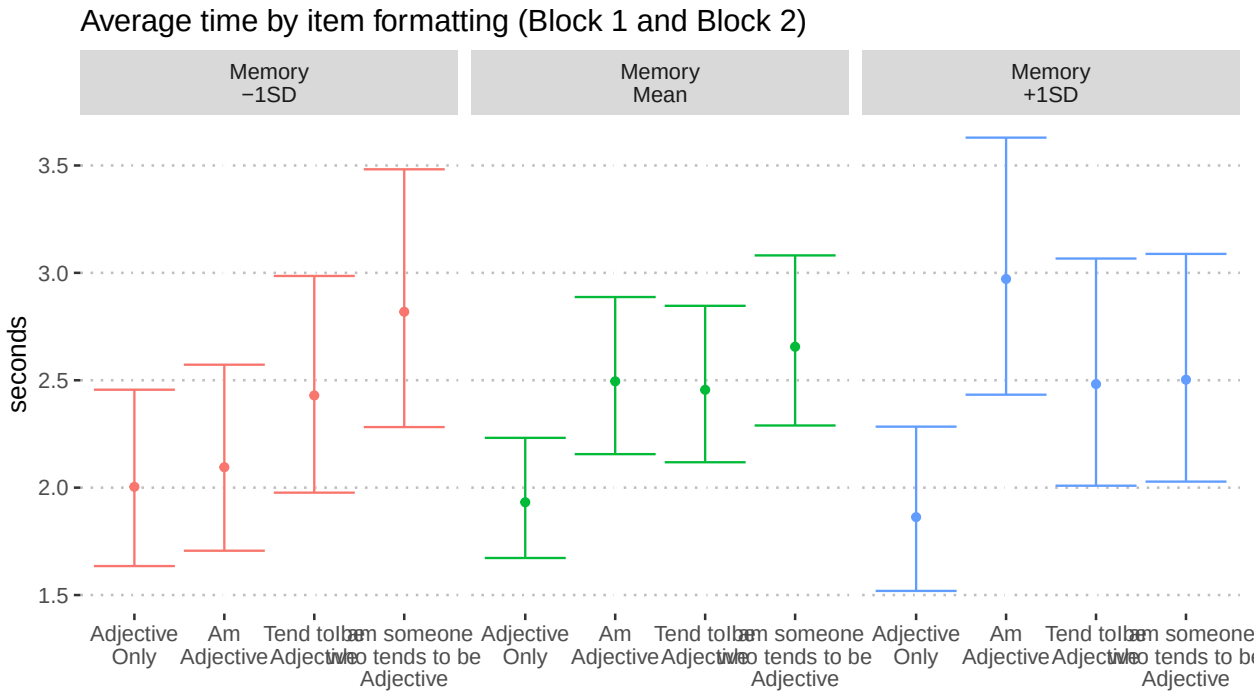


Figure 41: Predicted seconds on personality items by condition after controlling for delayed_memory.

```
summary_by_item_mem = mod_by_item_mem %>%
  mutate(tidy = map(aov, broom::tidy)) %>%
  select(item, tidy) %>%
  unnest(cols = c(tidy)) %>%
  filter(term == "format:delayed_memory") %>%
  mutate(reverse = case_when(
    item %in% reverse ~ "Y",
    TRUE ~ "N"
  )) %>%
  mutate(p.adj = p.adjust(p.value, method = "holm"))

summary_by_item_mem %>%
  arrange(reverse, item) %>%
  mutate(across( starts_with("p"), printp )) %>% # format p-values
  select(item, reverse, sumsq, meansq, NumDF, DenDF, statistic, p.value, p.adj) %>%
  kable(digits = 2, booktabs = T) %>%
  kable_styling()
```

6.3.2 Pairwise t-tests for significant ANOVAs

Here we identify the specific items with significant differences.

```
sig_item_mem = summary_by_item_mem %>%
  filter(p.value < .05)
```

item	reverse	sumsq	meansq	NumDF	DenDF	statistic	p.value	p.adj
active	N	0.80	0.27	3	62.00	0.35	.789	> .999
adventurous	N	1.47	0.49	3	54.52	1.06	.376	> .999
broadminded	N	0.65	0.22	3	51.30	0.51	.677	> .999
calm	N	2.43	0.81	3	62.00	0.96	.417	> .999
caring	N	1.25	0.42	3	62.00	0.73	.535	> .999
cautious	N	1.31	0.44	3	47.18	0.70	.559	> .999
creative	N	6.90	2.30	3	62.00	2.61	.059	> .999
curious	N	2.25	0.75	3	53.87	1.48	.231	> .999
friendly	N	2.29	0.76	3	56.38	1.09	.361	> .999
hardworking	N	1.54	0.51	3	62.00	0.74	.531	> .999
helpful	N	1.05	0.35	3	56.98	1.43	.243	> .999
imaginative	N	6.19	2.06	3	62.00	2.79	.048	> .999
intelligent	N	3.11	1.04	3	62.00	0.97	.410	> .999
lively	N	1.36	0.45	3	62.00	0.40	.750	> .999
organized	N	1.62	0.54	3	62.00	0.75	.524	> .999
outgoing	N	6.13	2.04	3	58.90	3.48	.021	.621
responsible	N	7.33	2.44	3	62.00	5.05	.003	.106
selfdisciplined	N	4.25	1.42	3	62.00	2.21	.096	> .999
softhearted	N	3.03	1.01	3	62.00	1.43	.242	> .999
sophisticated	N	8.05	2.68	3	62.00	3.09	.034	.907
sympathetic	N	1.31	0.44	3	62.00	0.53	.664	> .999
talkative	N	2.19	0.73	3	39.67	4.15	.012	.359
thorough	N	3.48	1.16	3	55.00	1.48	.231	> .999
thrifty	N	1.23	0.41	3	49.59	0.84	.479	> .999
warm	N	5.23	1.74	3	56.82	3.30	.027	.750
careless	Y	2.81	0.94	3	61.60	1.68	.180	> .999
impulsive	Y	7.30	2.43	3	61.99	2.74	.051	> .999
moody	Y	0.43	0.14	3	54.42	0.29	.835	> .999
nervous	Y	2.86	0.95	3	62.00	1.06	.374	> .999
reckless	Y	6.95	2.32	3	60.74	2.61	.059	> .999
worrying	Y	1.49	0.50	3	62.00	0.87	.460	> .999

```
sig_item_mem = sig_item_mem$item
sig_item_mem
```

```
## [1] "imaginative"   "outgoing"      "responsible"   "sophisticated"
## [5] "talkative"     "warm"
```

6.3.3 Outgoing

```
outgoing_model_mem = items_12 %>%
  filter(item == "outgoing") %>%
  lmer(seconds_log~format*delayed_memory + (1|proid),
                                data = .)
```

```
plot_model(outgoing_model_mem, type = "pred", terms = c("format", "delayed_memory[meansd]"))
```

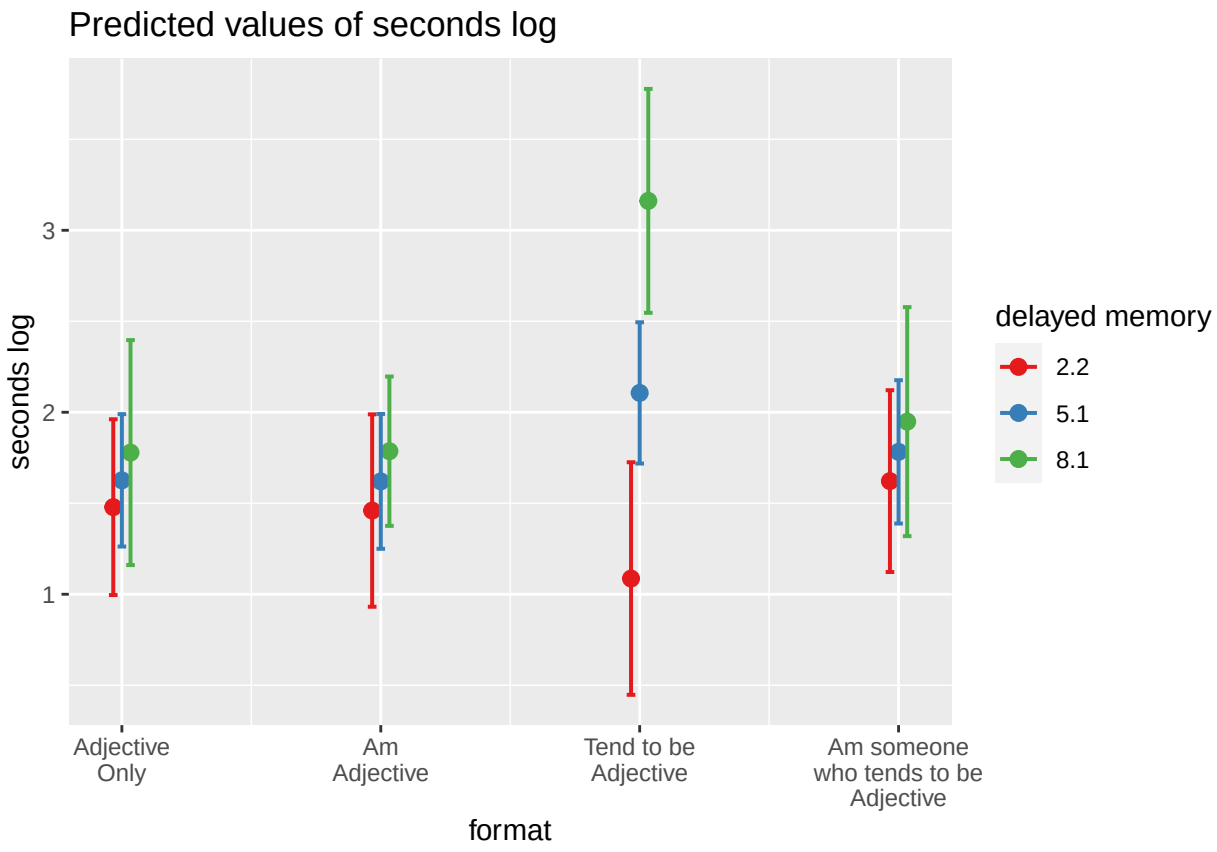


Figure 42: Average log-seconds to "outgoing" by format (Block 1 and Block 2)

6.3.4 Warm

```
warm_model_mem = items_12 %>%
  filter(item == "warm") %>%
  lmer(seconds_log~format*delayed_memory + (1|proid),
                                data = .)
```

```
plot_model(warm_model_mem, type = "pred", terms = c("format", "delayed_memory[meansd]"))
```

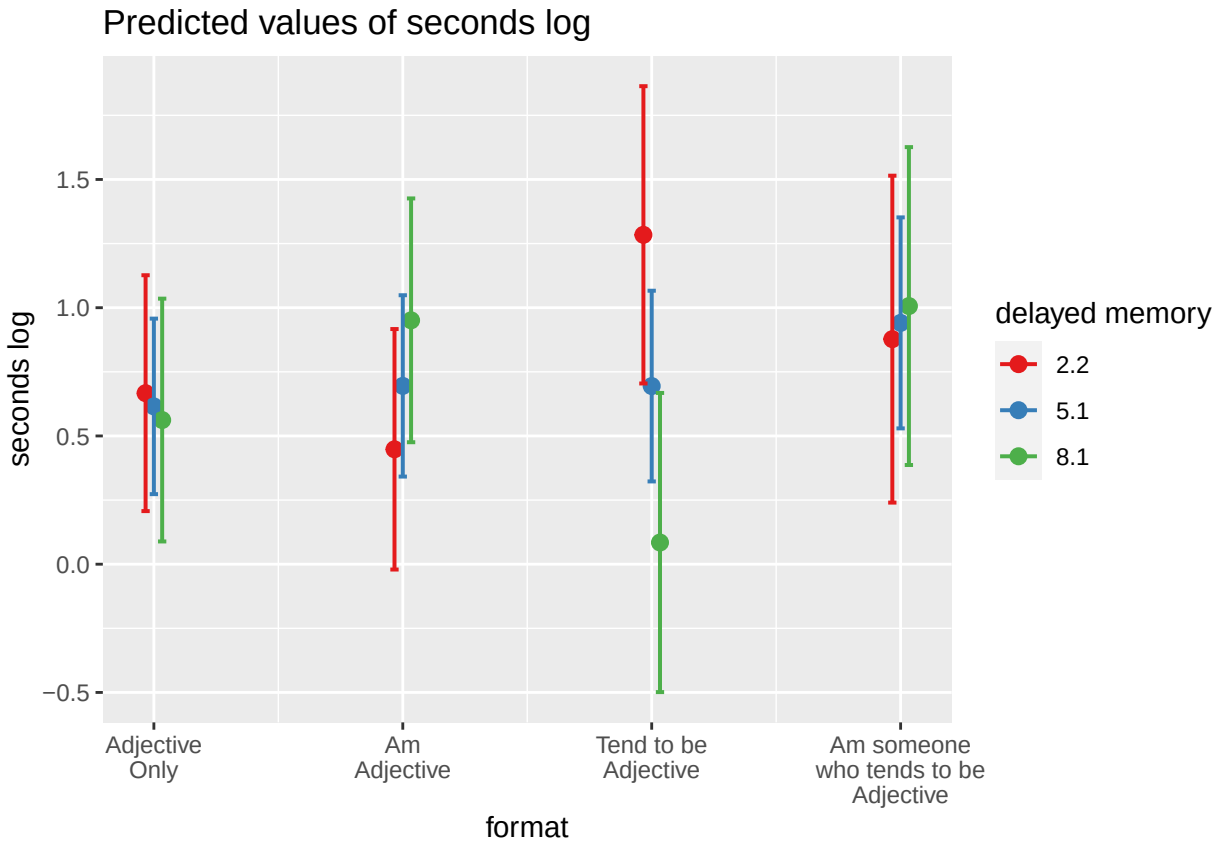


Figure 43: Average log-seconds to "warm" by format (Block 1 and Block 2)

6.3.5 Responsible

```
responsible_model_mem = items_12 %>%
  filter(item == "responsible") %>%
  lmer(seconds_log~format*delayed_memory + (1|proid),
                                data = .)
```

```
plot_model(responsible_model_mem, type = "pred", terms = c("format", "delayed_memory[meansd]"))
```

6.3.6 Imaginative

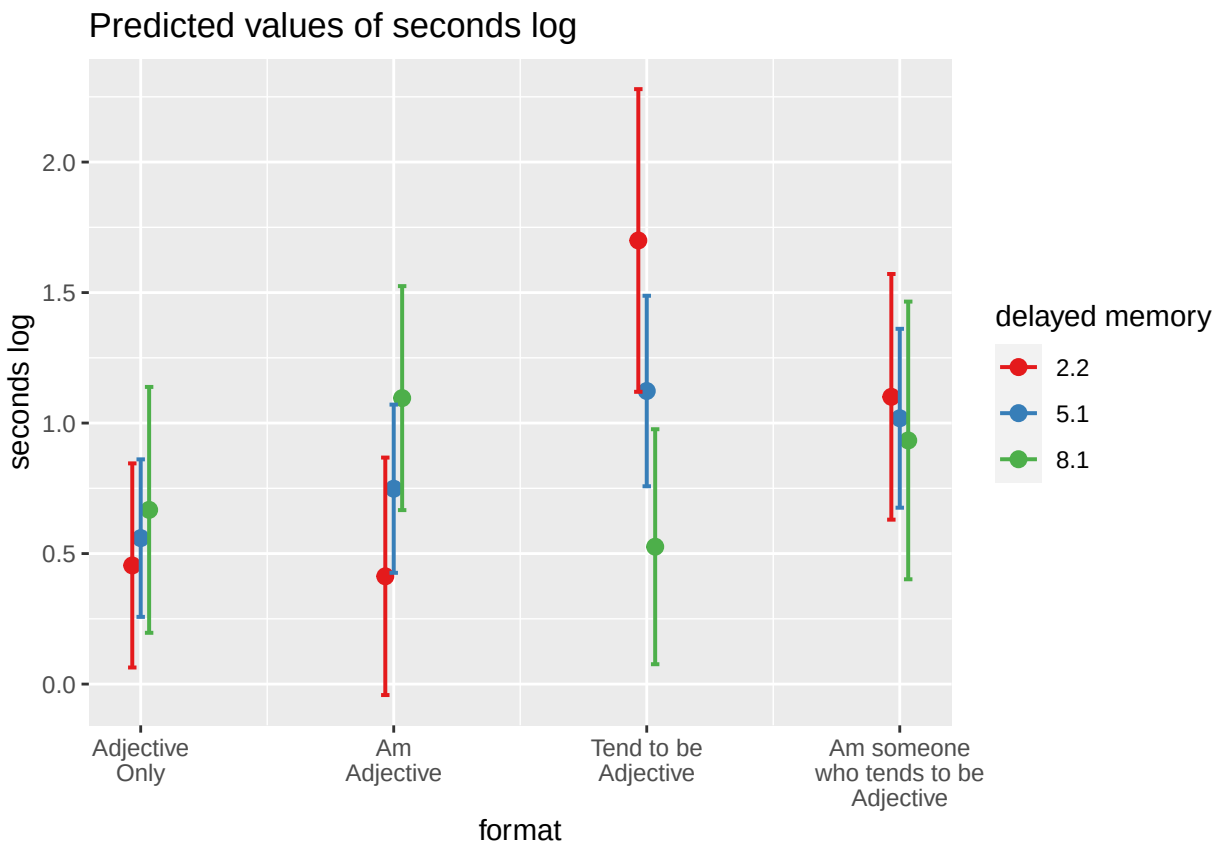


Figure 44: Average log-seconds to "responsible" by format (Block 1 and Block 2)

```
imaginative_model_mem = items_12 %>%
  filter(item == "imaginative") %>%
  lmer(seconds_log~format*delayed_memory + (1|proid),
                                data = .)
```

```
plot_model(imaginative_model_mem, type = "pred", terms = c("format", "delayed_memory[meansd]"))
```

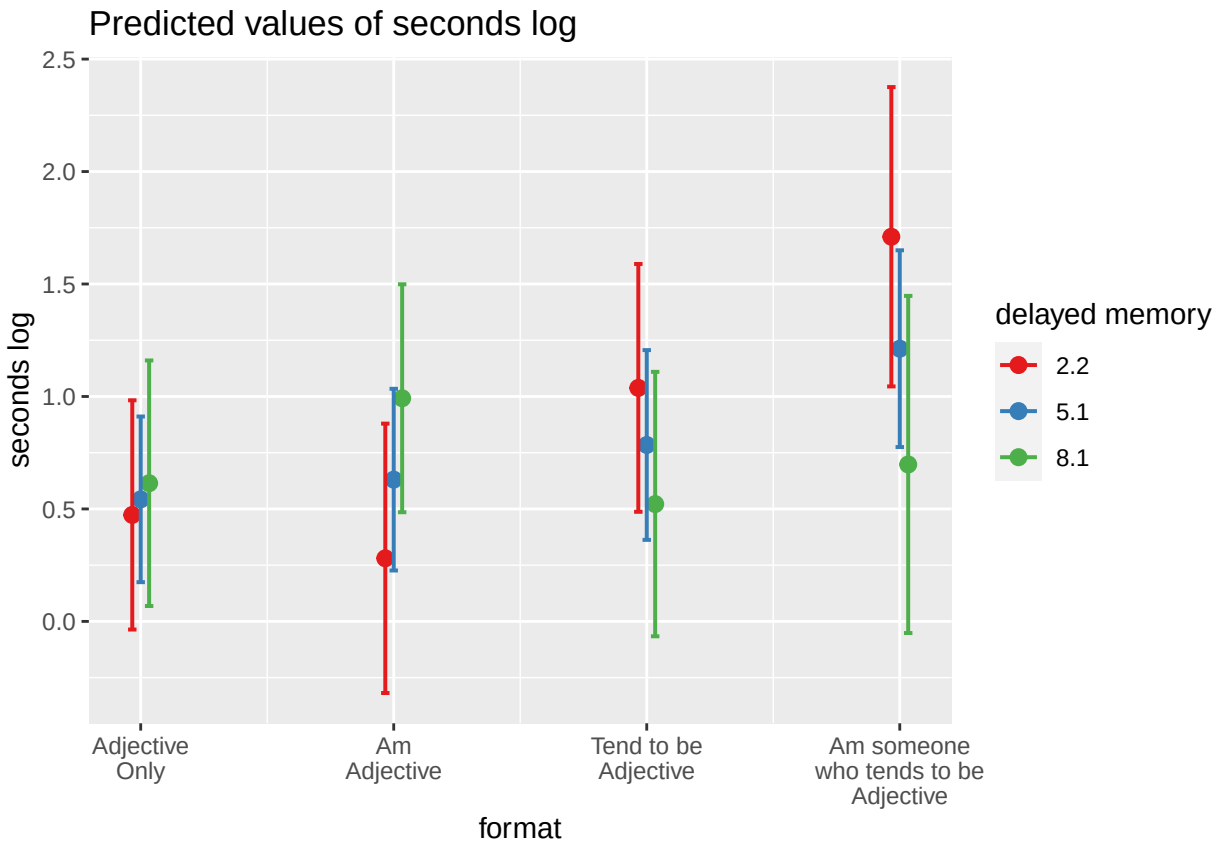


Figure 45: Average log-seconds to "imaginative" by format (Block 1 and Block 2)

6.3.7 Talkative

```
talkative_model_mem = items_12 %>%
  filter(item == "talkative") %>%
  lmer(seconds_log~format*delayed_memory + (1|proid),
                                data = .)
```

```
plot_model(talkative_model_mem, type = "pred", terms = c("format", "delayed_memory[meansd]"))
```

6.3.8 Sophisticated

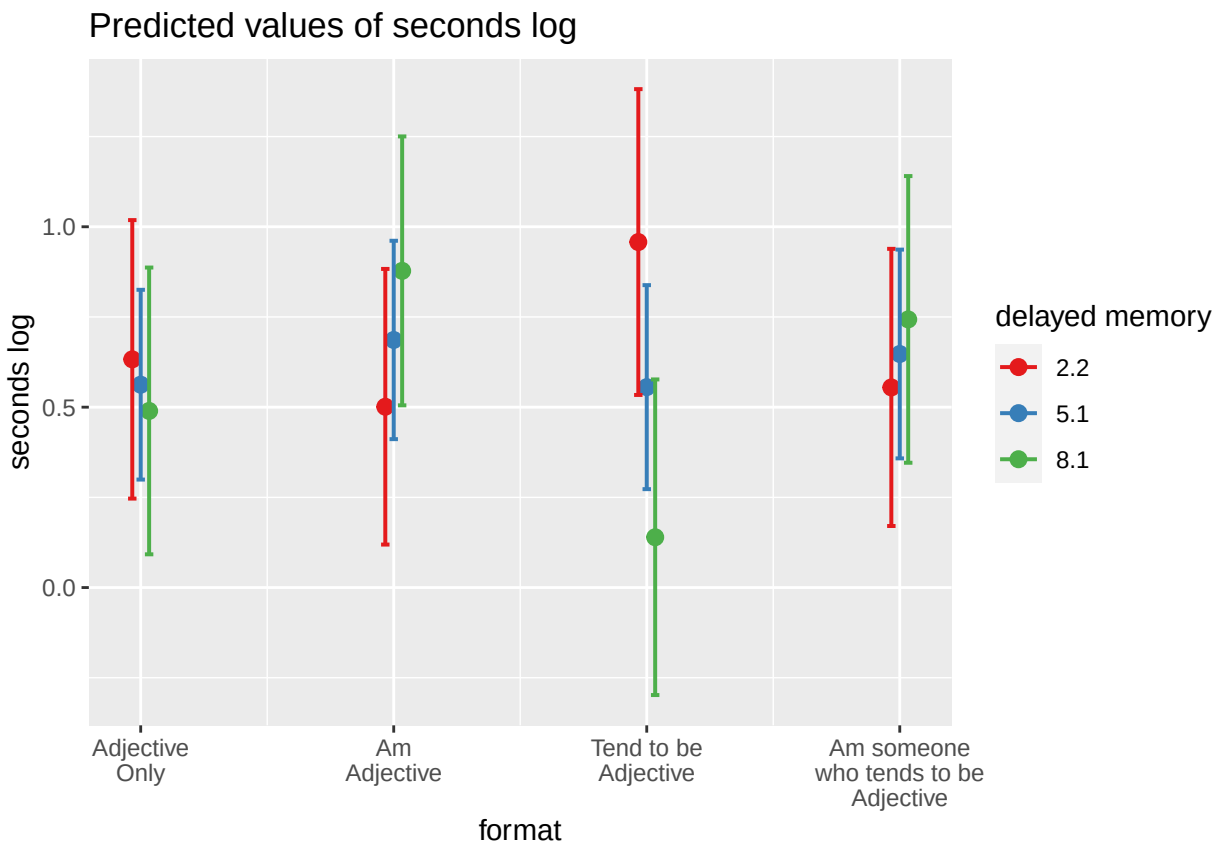


Figure 46: Average log-seconds to "talkative" by format (Block 1 and Block 2)

```
sophisticated_model_mem = items_12 %>%
  filter(item == "sophisticated") %>%
  lmer(seconds_log~format*delayed_memory + (1|proid),
                               data = .)
```

```
plot_model(sophisticated_model_mem, type = "pred", terms = c("format", "delayed_memory[meansd]"))
```

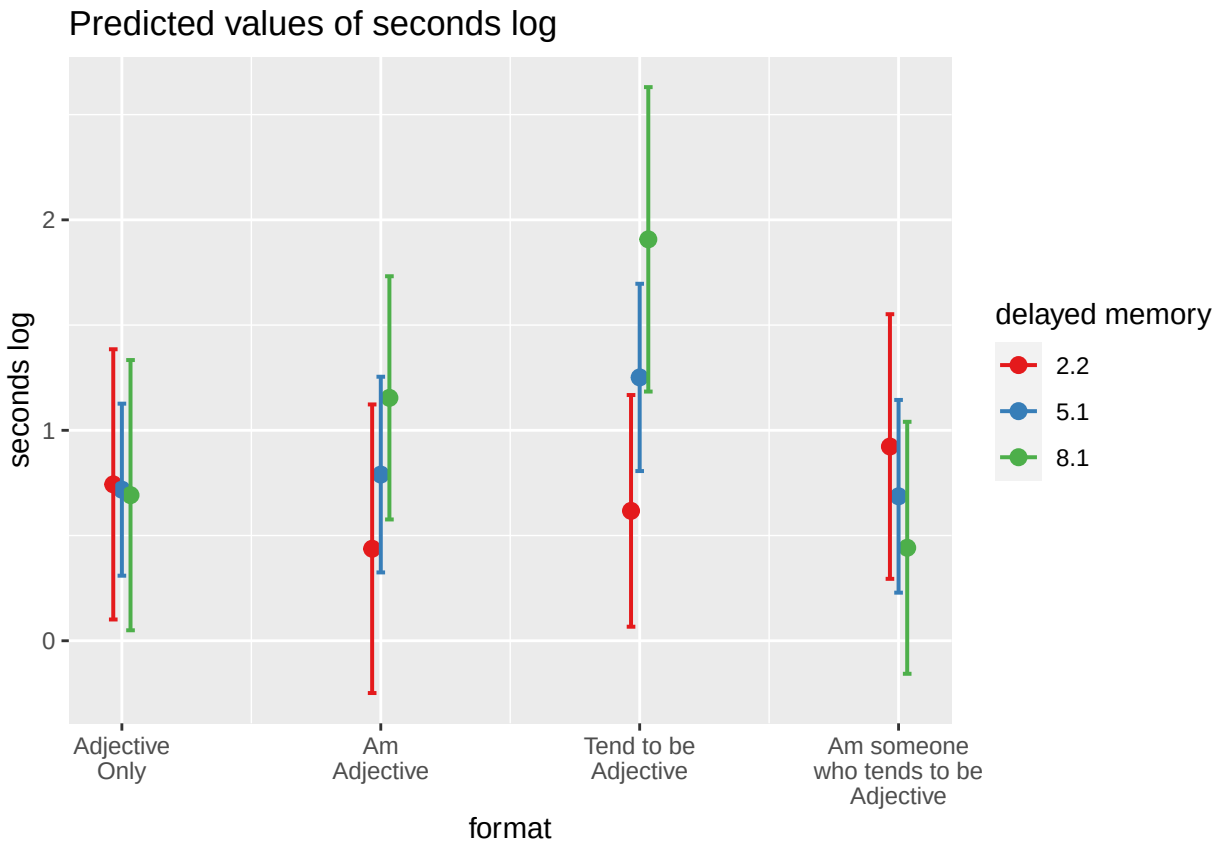


Figure 47: Average log-seconds to "sophisticated" by format (Block 1 and Block 2)

6.4 Inclusion of "I" (Block 1 and Block 3)

We used a multilevel model, nesting response within participant to account for dependence. Our primary predictors are format and also the presence of the word "I". Here, we use data from blocks 1 and 3.

```
items_13 = items_df %>%
  filter(block %in% c("1","3")) %>%
  filter(condition != "A") %>%
  filter(time2 == "yes")
```

```
mod.format_b3_1 = lmer(seconds_log~format + i + (1|proid),
                   data = items_13)
anova(mod.format_b3_1)
```

```
## Type III Analysis of Variance Table with Satterthwaite's method
##         Sum Sq Mean Sq NumDF  DenDF F value Pr(>F)
## format 1.81935 0.90968     2  13.00  1.6587 0.2282
## i      0.04319 0.04319     1 976.92  0.0788 0.7790
```

```
mod.format_b3_2 = lmer(seconds_log~format*i + (1|proid),
                  data = items_13)
anova(mod.format_b3_2)
```

```
## Type III Analysis of Variance Table with Satterthwaite's method
##           Sum Sq Mean Sq NumDF  DenDF F value Pr(>F)
## format   2.05702 1.02851     2  13.74  1.8753 0.1905
## i        0.00785 0.00785     1 974.55  0.0143 0.9048
## format:i 1.10566 0.55283     2 974.69  1.0080 0.3653
```

```
plot_b2 = plot_model(mod.format_b3_2, type = "pred", terms = c("format", "i"))

plot_b2 +
  labs(x = NULL,
       y = "Average log-seconds",
       title = "Average responses by item formatting (Block 1 and Block 3)",
       color = "I") +
  theme_pubclean()
```

6.4.1 One model for each adjective

```
mod_by_item_i1 = items_13 %>%
  group_by(item) %>%
  nest() %>%
  mutate(mod = map(data, ~lmer(seconds_log~format+i + (1|proid), data = .))) %>%
  mutate(aov = map(mod, anova)) %>%
  ungroup()

summary_by_item_i1 = mod_by_item_i1 %>%
  mutate(tidy = map(aov, broom::tidy)) %>%
  select(item, tidy) %>%
  unnest(cols = c(tidy)) %>%
  filter(term == "i") %>%
  mutate(reverse = case_when(
    item %in% reverse ~ "Y",
    TRUE ~ "N"
  )) %>%
  mutate(p.adj = p.adjust(p.value, method = "holm"))

summary_by_item_i1 %>%
  arrange(reverse, item) %>%
  mutate(across( starts_with("p"), printp )) %>% # format p-values
  select(item, reverse, sumsq, meansq, NumDF, DenDF, statistic, p.value, p.adj) %>%
  kable(digits = 2, booktabs = T) %>%
  kable_styling()
```

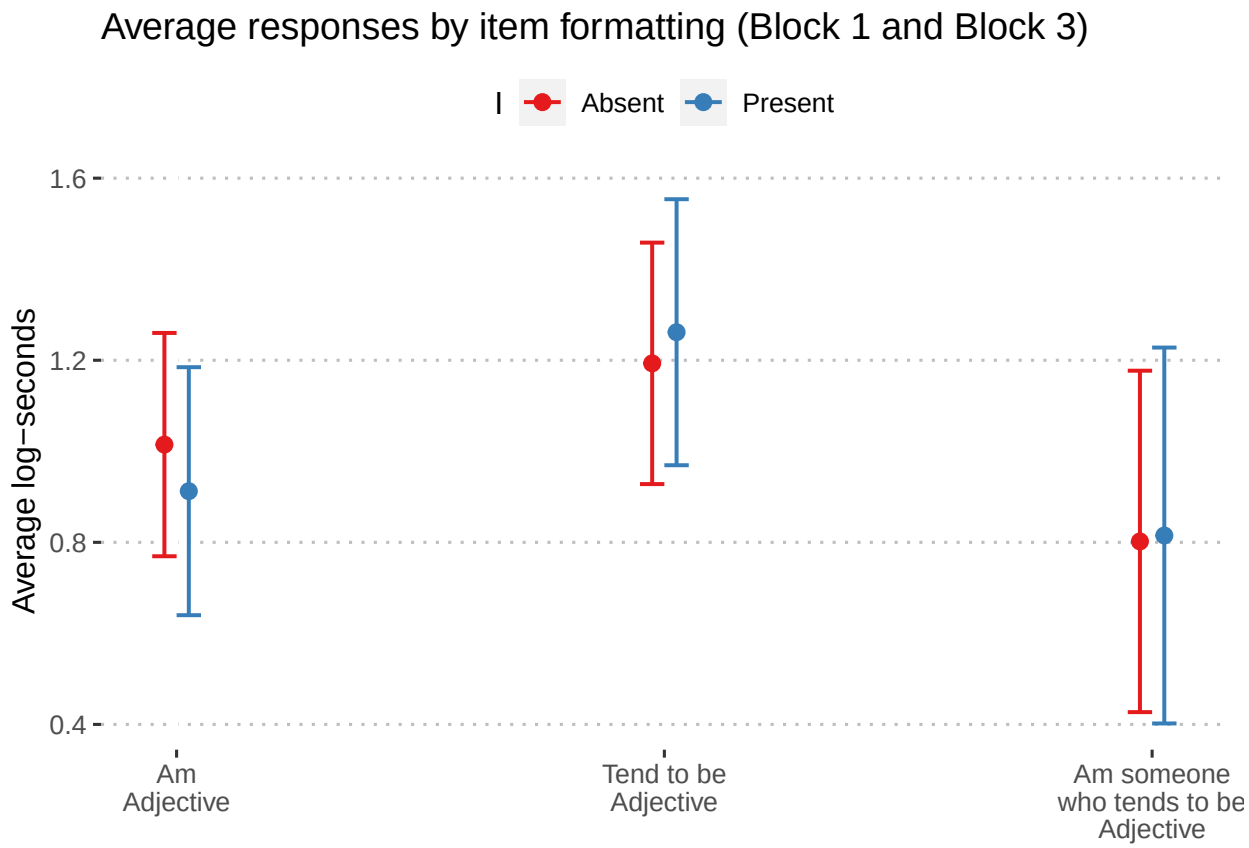


Figure 48: Predicted log-seconds on personality items by condition and I, using Block 1 and Block 3 data.

item	reverse	sumsq	meansq	NumDF	DenDF	statistic	p.value	p.adj
active	N	0.01	0.01	1	24.16	0.06	.801	> .999
adventurous	N	0.06	0.06	1	18.43	0.48	.498	> .999
broadminded	N	0.16	0.16	1	25.32	0.36	.556	> .999
calm	N	0.07	0.07	1	25.15	0.10	.759	> .999
caring	N	0.99	0.99	1	21.87	1.85	.188	> .999
cautious	N	1.44	1.44	1	21.88	3.29	.083	> .999
creative	N	0.64	0.64	1	22.13	0.79	.384	> .999
curious	N	0.77	0.77	1	28.00	1.11	.302	> .999
friendly	N	0.02	0.02	1	19.44	0.11	.740	> .999
hardworking	N	0.35	0.35	1	23.10	0.31	.582	> .999
helpful	N	0.02	0.02	1	19.54	0.19	.666	> .999
imaginative	N	0.01	0.01	1	27.24	0.02	.897	> .999
intelligent	N	0.15	0.15	1	27.01	0.16	.693	> .999
lively	N	0.85	0.85	1	24.06	2.27	.145	> .999
organized	N	0.18	0.18	1	17.14	1.94	.181	> .999
outgoing	N	0.13	0.13	1	28.00	0.21	.648	> .999
responsible	N	0.35	0.35	1	28.00	0.84	.367	> .999
selfdisciplined	N	0.05	0.05	1	19.48	0.16	.695	> .999
softhearted	N	0.04	0.04	1	18.60	0.12	.735	> .999
sophisticated	N	0.09	0.09	1	28.00	0.17	.683	> .999
sympathetic	N	0.04	0.04	1	22.50	0.07	.795	> .999
talkative	N	0.10	0.10	1	21.74	0.43	.518	> .999
thorough	N	0.10	0.10	1	25.88	0.13	.717	> .999
thrifty	N	1.52	1.52	1	28.00	1.78	.193	> .999
warm	N	0.33	0.33	1	18.12	2.14	.161	> .999
careless	Y	0.01	0.01	1	28.00	0.01	.904	> .999
impulsive	Y	0.84	0.84	1	19.81	0.84	.369	> .999
moody	Y	0.04	0.04	1	21.65	0.22	.647	> .999
nervous	Y	0.30	0.30	1	21.17	0.95	.340	> .999
reckless	Y	0.05	0.05	1	21.13	0.16	.689	> .999
worrying	Y	0.15	0.15	1	22.98	0.30	.588	> .999

```r
mod_by_item_i2 = items_13 %>%
  group_by(item) %>%
  nest() %>%
  mutate(mod = map(data, ~lmer(seconds_log~format*i + (1|proid), data = .))) %>%
  mutate(aov = map(mod, anova)) %>%
  ungroup()

summary_by_item_i2 = mod_by_item_i2 %>%
  mutate(tidy = map(aov, broom::tidy)) %>%
  select(item, tidy) %>%
  unnest(cols = c(tidy)) %>%
  filter(term == "format:i") %>%
  mutate(reverse = case_when(
    item %in% reverse ~ "Y",
    TRUE ~ "N"
  )) %>%
  mutate(p.adj = p.adjust(p.value, method = "holm"))

summary_by_item_i2 %>%
  arrange(reverse, item) %>%
  mutate(across( starts_with("p"), printp )) %>% # format p-values
  select(item, reverse, sumsq, meansq, NumDF, DenDF, statistic, p.value, p.adj) %>%
  kable(digits = 2, booktabs = T) %>%
  kable_styling()
```

item	reverse	sumsq	meansq	NumDF	DenDF	statistic	p.value	p.adj
active	N	0.21	0.10	2	21.20	0.51	.607	> .999
adventurous	N	0.06	0.03	2	16.40	0.20	.823	> .999
broadminded	N	0.03	0.01	2	24.26	0.03	.970	> .999
calm	N	0.15	0.07	2	26.00	0.09	.910	> .999
caring	N	2.15	1.07	2	17.27	2.64	.100	> .999
cautious	N	0.38	0.19	2	18.83	0.39	.681	> .999
creative	N	1.24	0.62	2	21.57	0.70	.506	> .999
curious	N	0.97	0.48	2	26.00	0.68	.516	> .999
friendly	N	0.21	0.11	2	17.58	0.60	.562	> .999
hardworking	N	1.20	0.60	2	20.93	0.51	.605	> .999
helpful	N	0.37	0.18	2	17.48	1.64	.222	> .999
imaginative	N	0.04	0.02	2	26.00	0.03	.969	> .999
intelligent	N	0.39	0.20	2	25.13	0.20	.822	> .999
lively	N	0.54	0.27	2	22.83	0.68	.518	> .999
organized	N	0.14	0.07	2	15.55	0.71	.509	> .999
outgoing	N	0.09	0.09	1	27.00	0.14	.713	> .999
responsible	N	0.29	0.15	2	26.00	0.34	.718	> .999
selfdisciplined	N	0.86	0.43	2	16.51	1.57	.237	> .999
softhearted	N	1.17	0.59	2	15.63	2.18	.146	> .999
sophisticated	N	0.07	0.04	2	26.00	0.06	.938	> .999
sympathetic	N	2.59	1.30	2	21.37	2.49	.106	> .999
talkative	N	0.03	0.01	2	19.92	0.05	.947	> .999
thorough	N	2.10	1.05	2	26.00	1.49	.244	> .999
thrifty	N	2.27	1.13	2	26.00	1.37	.273	> .999
warm	N	0.47	0.24	2	15.12	1.82	.196	> .999
careless	Y	0.81	0.41	2	26.00	0.69	.508	> .999
impulsive	Y	1.22	0.61	2	17.47	0.58	.571	> .999
moody	Y	0.00	0.00	2	20.19	0.01	.994	> .999
nervous	Y	0.49	0.25	2	19.41	0.80	.466	> .999
reckless	Y	0.87	0.44	2	17.67	1.81	.192	> .999
worrying	Y	0.59	0.29	2	20.70	0.54	.590	> .999

7 How does device type affect means and timing of responses?

In this section, we present exploratory analyses which test the effect of device type More specifically, we're interested to know whether the time it takes to complete personality assessments differs by device type (computer, phone, tablet). We also test whether the typical response to personality items is associated with device type, although we have no theoretical reasons to suspect this is the case.

For these analyses, we use only data collected in Blocks 1 and 2.

```
items_12 = items_df %>% filter(block %in% c("1","2"))
```

7.1 Timing

7.1.1 Timing by device

We used a multilevel model, nesting timing within participant to account for dependence. Our primary predictor was device type. As a reminder, our outcome variable (seconds) has been log-transformed, as it was strongly skewed.

```
mod.timing = lmer(seconds_log~devicetype + (1|proid),
                  data = items_12)
anova(mod.timing)
```

```
## Type III Analysis of Variance Table with Satterthwaite's method
##            Sum Sq Mean Sq NumDF DenDF F value Pr(>F)
## devicetype 2.4781  1.2391     2    32  1.8444 0.1745
```

When examining both Block 1 and Block 2 data, device type was unassociated with the time it took to respond to personality items ($F(2, 32.00) = 1.84, p = .175$).

```
plot1 = plot_model(mod.timing, type = "pred")

plot1$devicetype +
  labs(x = NULL,
       y = "Log-seconds (per item)",
       title = "Average time per personality item\nby device type",
       caption = "Bars represent 95% confidence intervals") +
  theme_pubclean()
```

```
means_by_group = items_12 %>%
  group_by(devicetype) %>%
  summarise(m = mean(timing),
            s = sd(timing))

items_12 %>%
  ggplot(aes(x = timing, fill = devicetype)) +
  geom_histogram(bins = 100) +
  geom_vline(aes(xintercept = m), data = means_by_group) +
  facet_wrap(~devicetype, scales = "free_y") +
  guides(fill = "none") +
  scale_x_log10() +
```

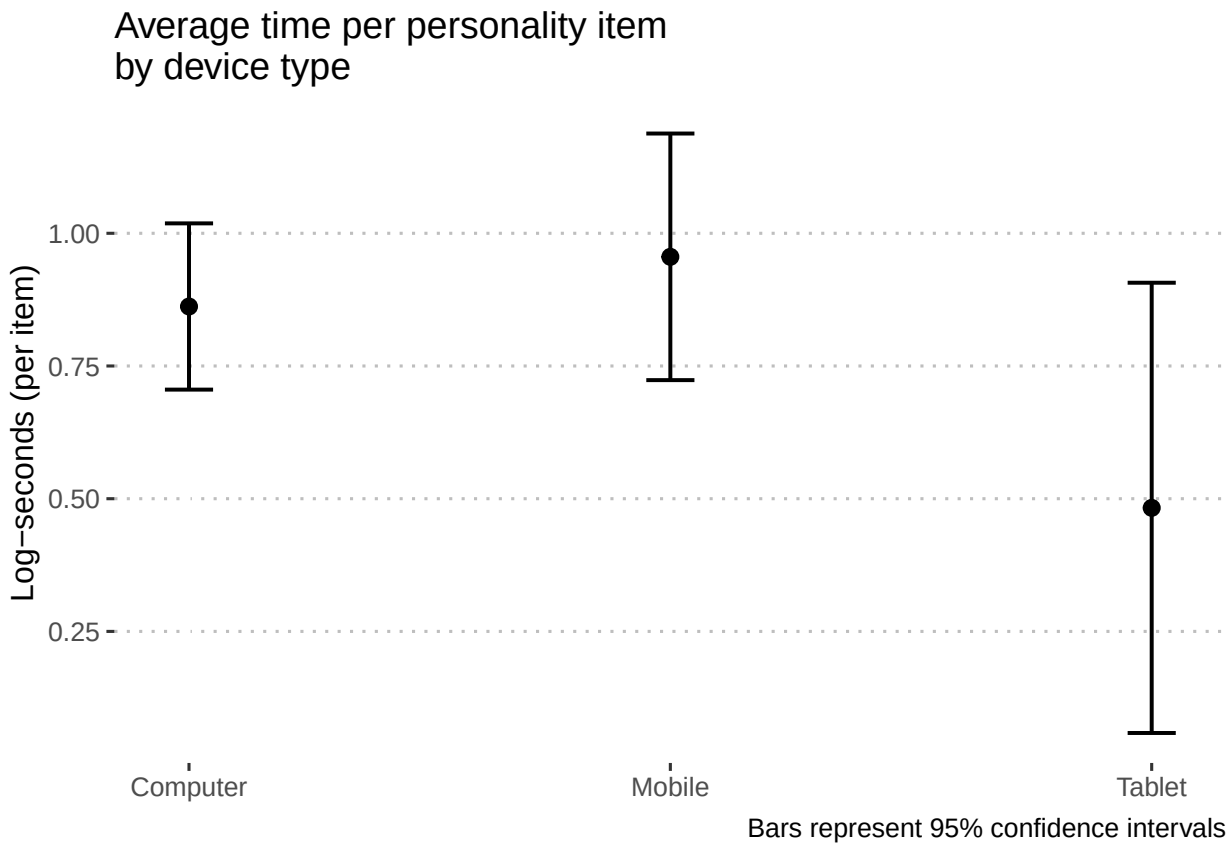


Figure 49: Predicted timing on personality items by condition.

```
  labs(y = "Number of particpants",
       title = "Distribution of timing by format",
       x = "timing (logrithmic scale)") +
  theme_pubr()
```

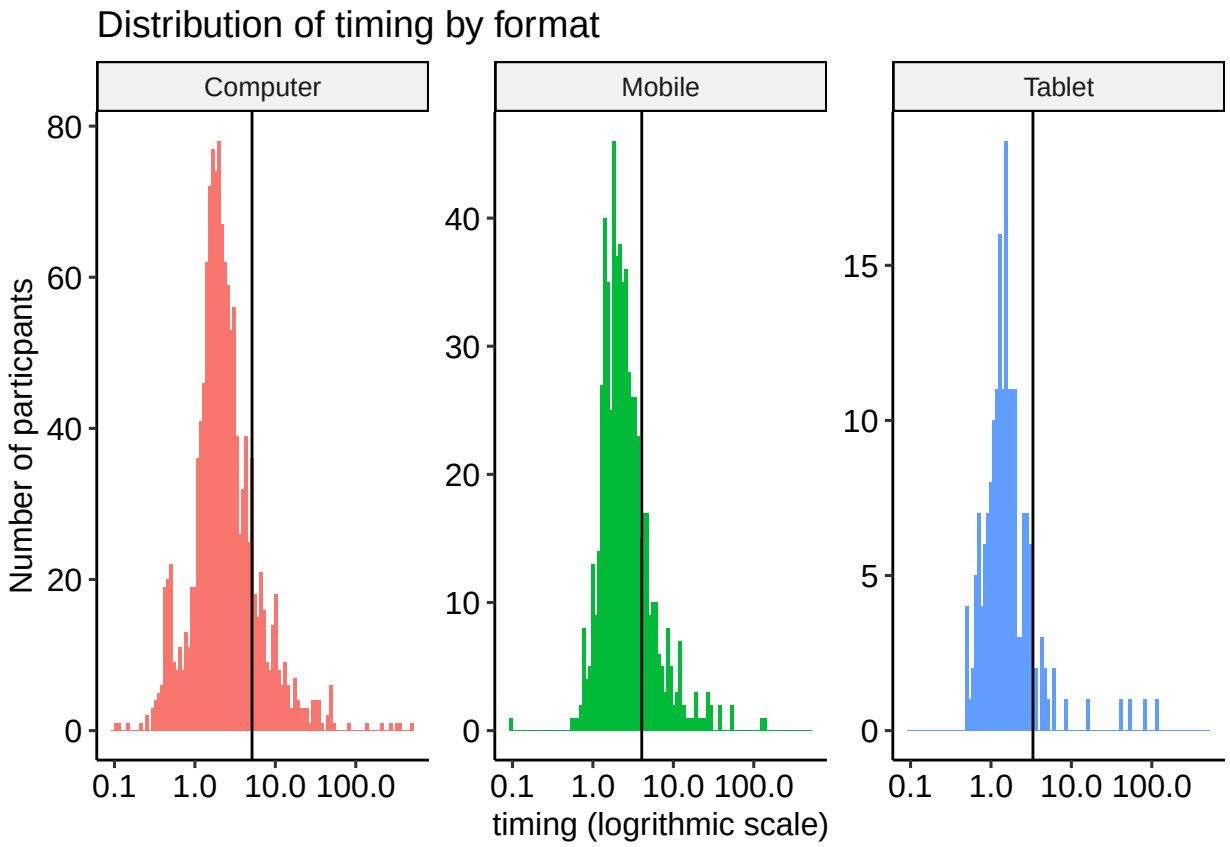


Figure 50: Distribution of secondss by category

7.1.2 Device by format

We also check to see whether device type and format interact in the prediction of time to answer personality items.

```
mod.timing2 = lmer(seconds_log~devicetype*format + (1|proid),
                   data = items_12)
anova(mod.timing2)
```

```
## Type III Analysis of Variance Table with Satterthwaite's method
##                    Sum Sq Mean Sq NumDF   DenDF F value    Pr(>F)
## devicetype          2.919  1.4595     2   32.31  2.2275 0.1240954
## format             11.022  3.6741     3 2146.65  5.6074 0.0007915 ***
## devicetype:format 14.153  2.3589     6 2148.23  3.6001 0.0014831 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The interaction between device type and format was associated with the time it took to respond to personality items ($F(6, 2, 148.23) = 3.60, p = .001$).

```
plot1 = plot_model(mod.timing2, type = "pred", terms = c("format", "devicetype"))

plot1 +
  geom_line() +
  labs(x = NULL,
       y = "Log-seconds (per item)",
       title = "Average time per personality item\nby device type and item format",
       color = "Device",
       caption = "Bars represent 95% confidence intervals") +
  theme_pubclean()
```

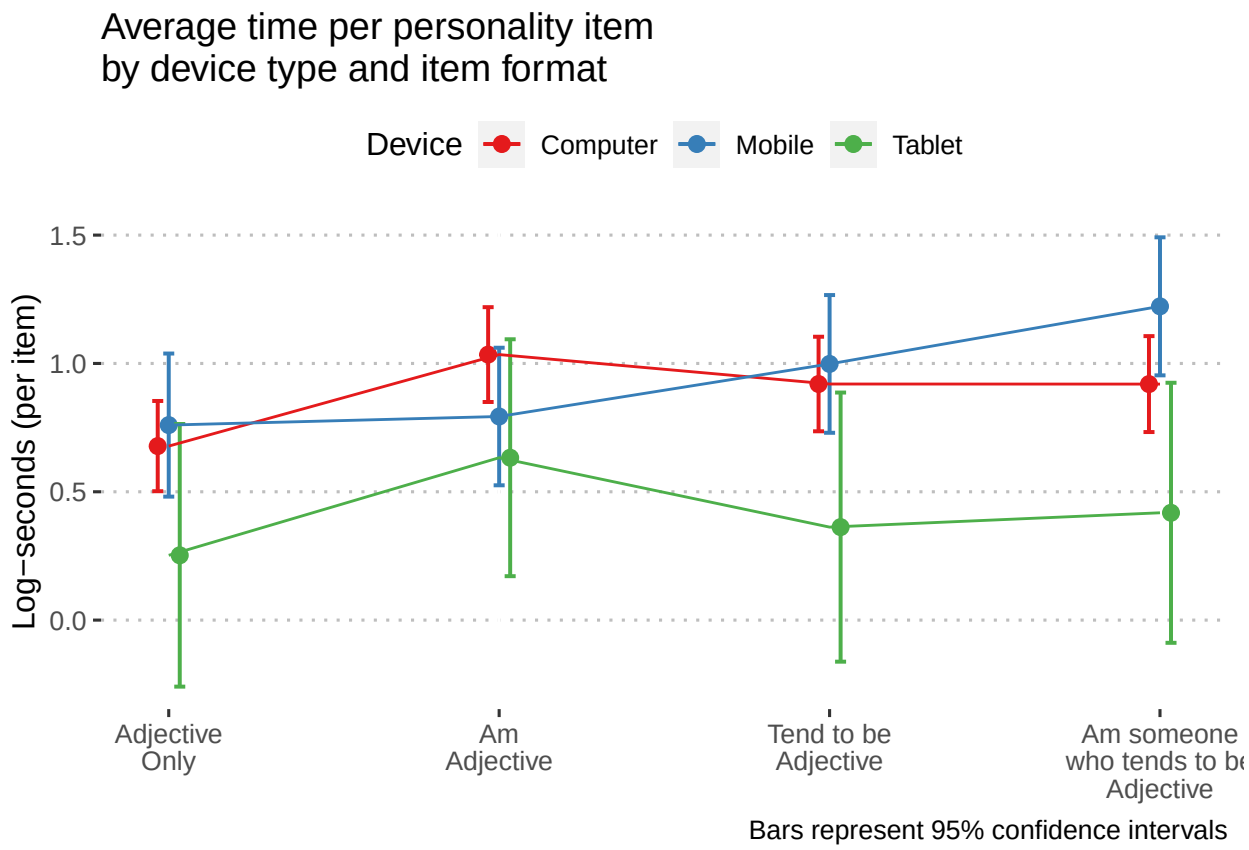


Figure 51: Predicted timing on personality items by condition.

7.2 Responses

Here we estimate the differences in response to personality items items by device. Again, we have no theoretical rationale for these models – these are purely exploratory.

7.2.1 Response by device

We used a multilevel model, nesting response within participant to account for dependence. Our primary predictor was device type.

```
mod.responseD = lmer(response~devicetype + (1|proid),
                     data = items_12)
anova(mod.responseD)
```

```
## Type III Analysis of Variance Table with Satterthwaite's method
##            Sum Sq Mean Sq NumDF DenDF F value Pr(>F)
## devicetype 2.9109  1.4554     2    32  0.9685 0.3905
```

When examining both Block 1 and Block 2 data, device type was unassociated with the time it took to respond to personality items ($F(2, 32.00) = 0.97, p = .391$).

```
plot1 = plot_model(mod.responseD, type = "pred")

plot1$devicetype +
  labs(x = NULL,
       y = "Expected response",
       title = "Expected responses by device",
       caption = "Bars represent 95% confidence intervals") +
  theme_pubclean()
```

```
means_by_group = items_12 %>%
  group_by(devicetype) %>%
  summarise(m = mean(response),
            s = sd(response))

items_12 %>%
  ggplot(aes(x = response)) +
  geom_histogram(aes(fill = block),
                 position = "dodge",
                 bins = 6, color = "white") +
  geom_vline(aes(xintercept = m),
             data = means_by_group) +
  facet_wrap(~devicetype, scales = "free_y") +
  #guides(fill = "none") +
  scale_x_continuous(breaks = 1:6) +
  labs(y = "Number of particpants",
       title = "Distribution of responses by format") +
  theme_pubr()
```

7.2.2 Device by format

We also check whether item format moderated the relationship between device type and response.

```
mod.responseD2 = lmer(response~devicetype*format + (1|proid),
                     data = items_12)
anova(mod.responseD2)
```

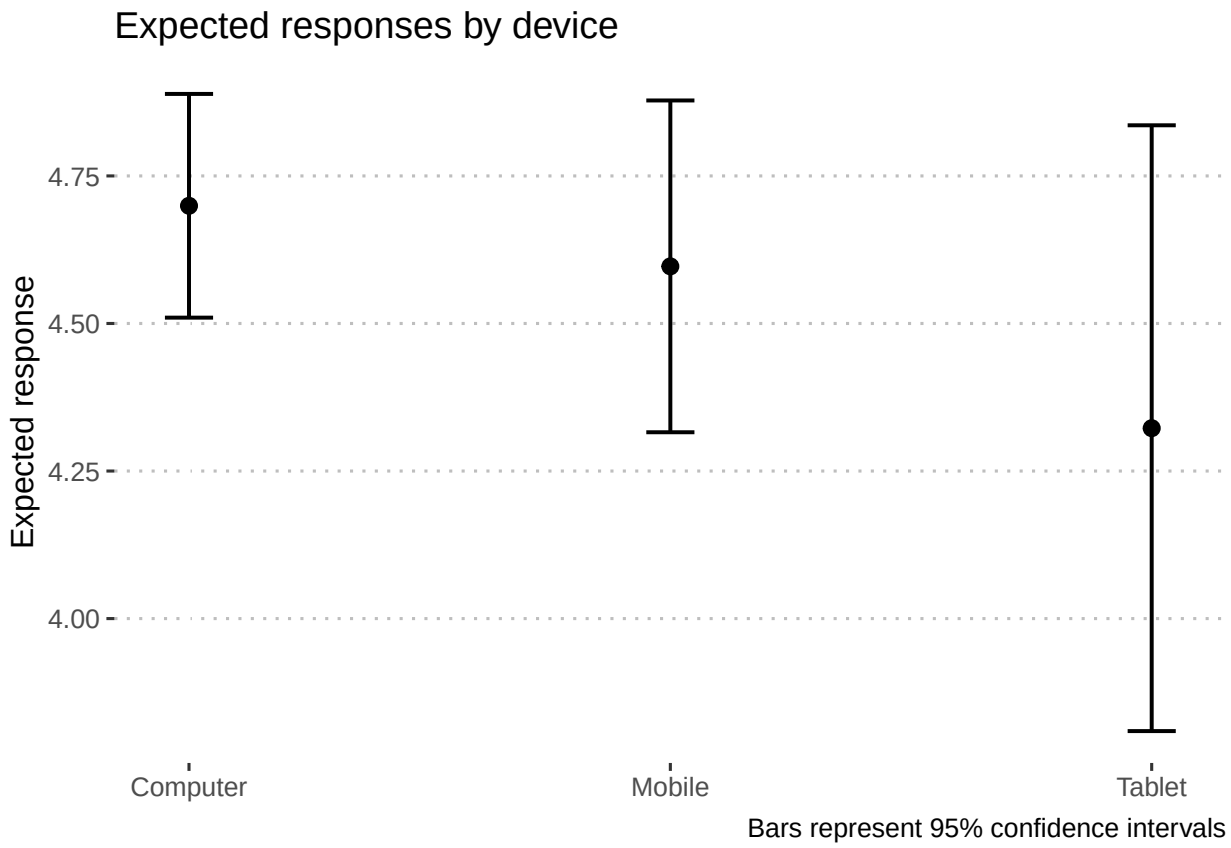


Figure 52: Predicted response on personality items by condition.

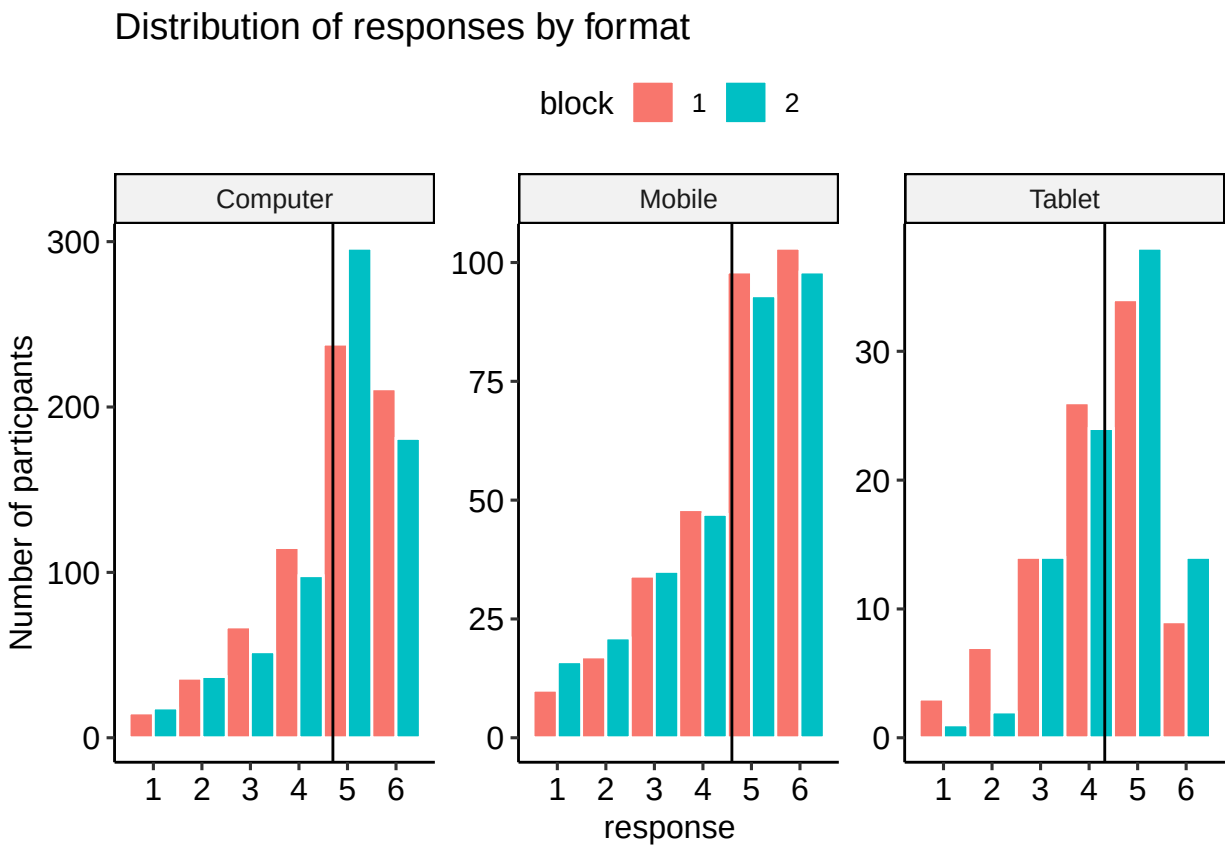


Figure 53: Distribution of responses by category

```
## Type III Analysis of Variance Table with Satterthwaite's method
##                     Sum Sq Mean Sq NumDF   DenDF F value Pr(>F)
## devicetype          1.8061 0.90304     2   32.83  0.6016 0.5539
## format              5.1709 1.72363     3 2105.79  1.1483 0.3283
## devicetype:format 13.6298 2.27163     6 2102.81  1.5133 0.1697
```

The interaction between device type and format was unassociated with the time it took to respond to personality items ($F(6, 2, 102.81) = 1.51, p = .170$).

```
plot2 = plot_model(mod.responseD2, type = "pred", terms = c("format", "devicetype"))

plot2 +
  geom_line() +
  labs(x = NULL,
       y = "Average response",
       title = "Average responses by device",
       caption = "Bars represent 95% confidence intervals") +
  theme_pubclean()
```

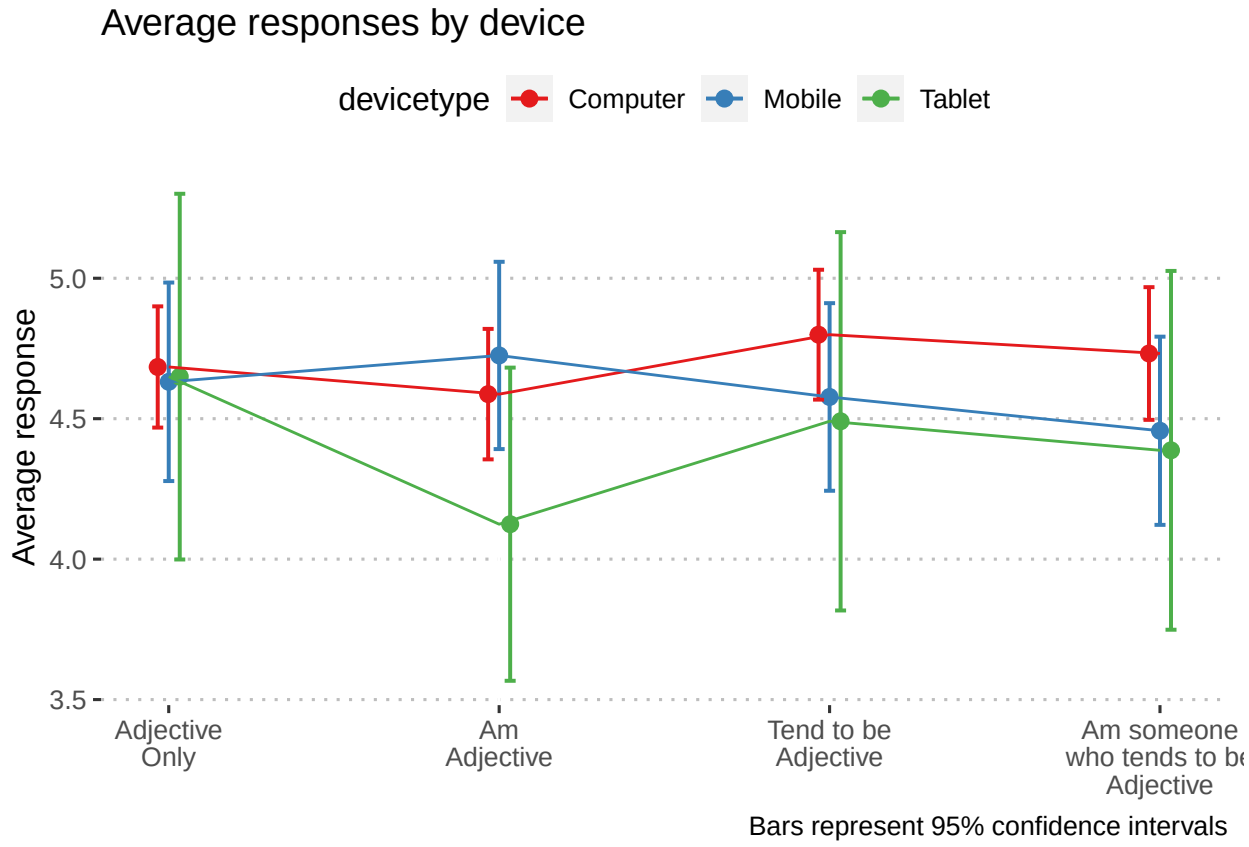


Figure 54: Predicted response on personality items by condition.

8 Power analysis

We conduct power analyses for the main research question – does formatting affect response to personality items – using a simulation method. That is, we generate datasets of varying sample sizes (from as few as 50 participants per condition to as many as 100), then simulate responses based on the models fit to the pilot data.

Here we set the sample sizes we'll test, as well as the number of simulations we'll run for each sample size.

```
sample_sizes = seq(50, 500, 25)

n_sims = 1000
```

8.1 Model 1

To simplify our code, we write a function that simulates responses to model 1 based on a given sample size, N, and number of repititions.

```
# function to simulate mod.format_b1

sim_format_b1 = function(n, sims){
  p_vals = numeric(length = sims)

  sim_a = expand_grid(
    proid = as.character(1:n),
    item = c(1:33),
    format = "Adjective\nOnly"
  )

  sim_b = expand_grid(
    proid = as.character((n+1):(2*n)),
    item = c(1:33),
    format = "Am\nAdjective"
  )

  sim_c = expand_grid(
    proid = as.character(((2*n)+1):(3*n)),
    item = c(1:33),
    format = "Tend to be\nAdjective"
  )
  sim_d = expand_grid(
    proid = as.character(((3*n)+1):(4*n)),
    item = c(1:33),
    format = "Am someone\nwho tends to be\nAdjective"
  )

  sim_data = rbind(sim_a, sim_b) %>% rbind(sim_c) %>% rbind(sim_d)
  for (i in 1:sims){
    sim_data$response = simulate(mod.format_b1, newdata = sim_data, allow.new.levels = T)[,1]
    sim_mod = lmer(response~format + (1|proid), data = sim_data)
    p_vals[i] = anova(sim_mod)["format", 6]}
  return(p_vals)
}
```

Next we identify the sample sizes for simulation (from 50 to 500 by 25) and create a data frame to hold the results. Power represents the proportion of simulations for which p is less than .05.

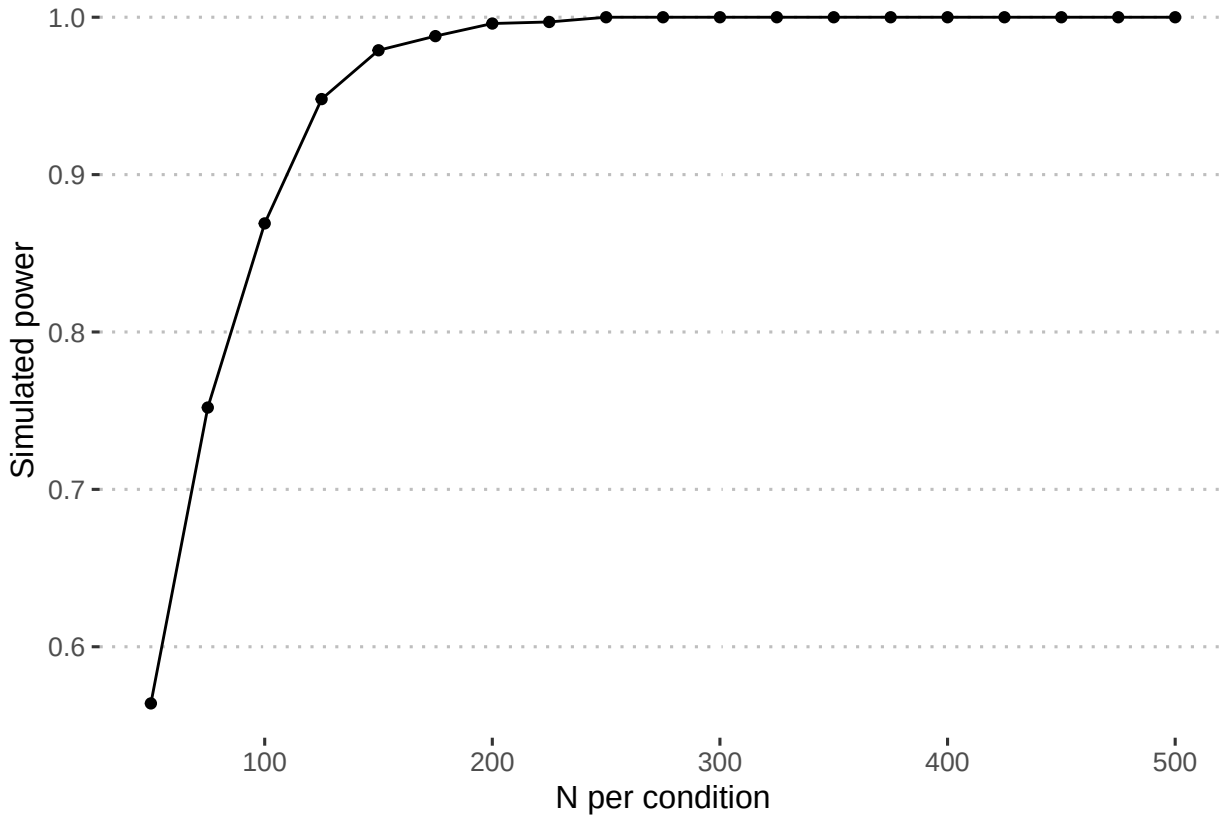
```
# simulate at various sample sizes
power_df = data.frame(
  N = sample_sizes,
  power = 0
)
```

Here we (inefficiently) loop through all sample sizes and calculate power.

```
set.seed(20210729)
for(i in sample_sizes){
  pvalues = sim_format_b1(i, n_sims)
  sig = ifelse(pvalues < .05, 1, 0)
  power_df$power[power_df$N == i] <- sum(sig)/n_sims
}
```

Finally, we plot these effects to determine needed sample size.

```
power_df %>%
  ggplot(aes(x = N, y = power)) +
  geom_line() +
  geom_point() +
  labs(
    x = "N per condition",
    y = "Simulated power"
  ) +
  theme_pubclean()
```




```
#identify minimum sample size

power_df_min = power_df %>%
  filter(power > .95)

N_min = min(power_df_min$N)
```

The simulation suggests that power would be over the threshold of .95 with a sample size of 150 participants per condition.

8.2 Model 2

Here we repeat the process for our second model, which uses both blocks of data from Time 1 (i.e., Blocks 1 and 2).

```
# function to simulate mod.format_b2

sim_format_b2 = function(n, sims){
  p_vals = numeric(length = sims)

  sim_a_b2 = expand_grid(
    proid = as.character(1:n),
    item = c(1:33),
    format = "Adjective\nOnly",
    block = "1"
  )
```

```
  sim_b_b2 = expand_grid(
    proid = as.character((n+1):(2*n)),
    item = c(1:33),
    format = "Am\nAdjective",
    block = "1"
  )

  sim_c_b2 = expand_grid(
    proid = as.character(((2*n)+1):(3*n)),
    item = c(1:33),
    format = "Tend to be\nAdjective",
    block = "1"
  )
  sim_d_b2 = expand_grid(
    proid = as.character(((3*n)+1):(4*n)),
    item = c(1:33),
    format = "Am someone\nwho tends to be\nAdjective",
    block = "1"
  )
  sim_b2 = expand_grid(
    proid = as.character(1:(4*n)),
    item = c(1:33),
    block = "2"
  )
  sim_b2$format = sample(
    x = c("Adjective\nOnly",
          "Am\nAdjective",
          "Tend to be\nAdjective",
          "Am someone\nwho tends to be\nAdjective"),
    size = 33*n*4,
    replace = TRUE
  )

  sim_data = full_join(sim_a_b2, sim_b_b2) %>%
    full_join(sim_c_b2) %>%
    full_join(sim_d_b2) %>%
    full_join(sim_b2)

  for (i in 1:sims){
    sim_data$response = simulate(mod.format_b2,
                                 newdata = sim_data,
                                 allow.new.levels = T)[,1]
    sim_mod = lmer(response~format + (1|proid),
                   data = sim_data)
    p_vals[i] = anova(sim_mod)["format", 6]}
  return(p_vals)
}
```

We use the same sample sizes and simulation length for these analyses, so we start by creating a new data frame.

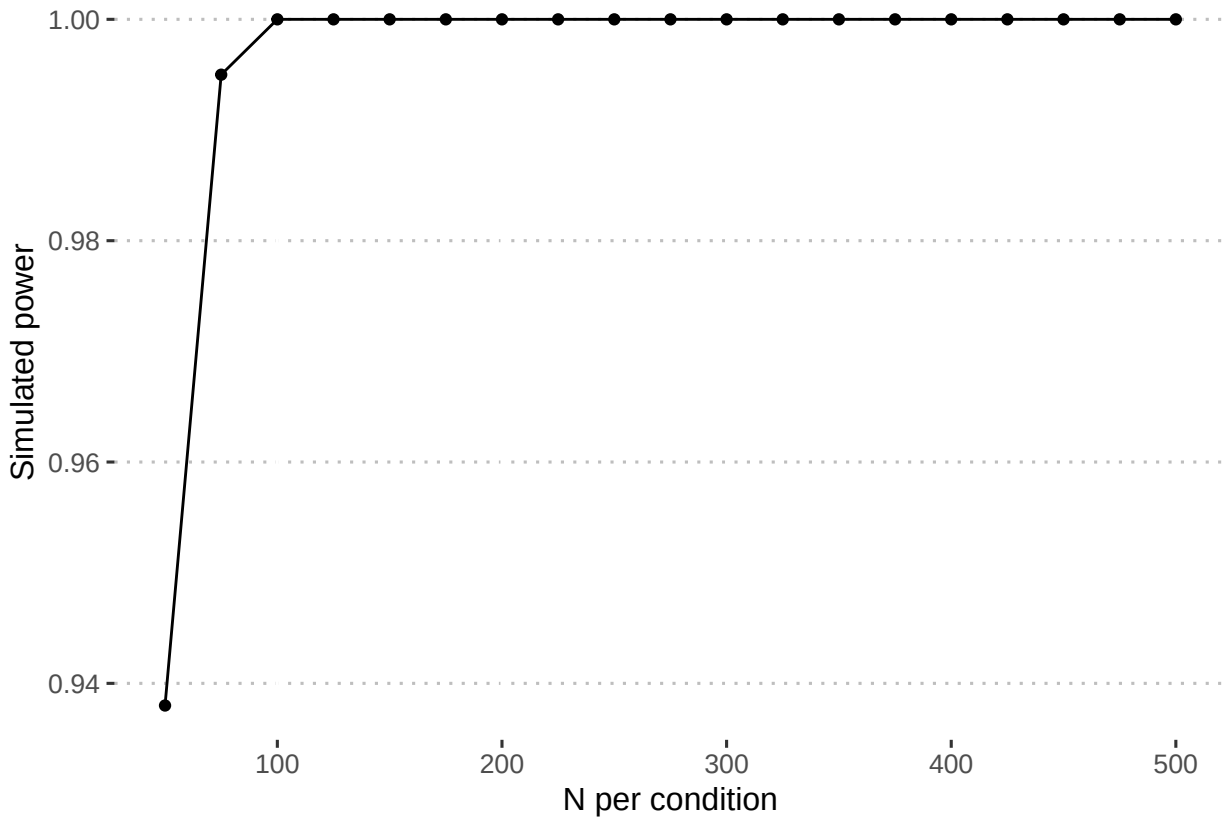
```
power_df_2 = data.frame(
  N = sample_sizes,
  power = 0
)
```

Here we (inefficiently) loop through all sample sizes and calculate power.

```
set.seed(20210729)
for(i in sample_sizes){
  pvalues = sim_format_b2(i, n_sims)
  sig = ifelse(pvalues < .05, 1, 0)
  power_df_2$power[power_df_2$N == i] <- sum(sig)/n_sims
}
```

Finally, we plot these effects to determine needed sample size.

```
power_df_2 %>%
  ggplot(aes(x = N, y = power)) +
  geom_line() +
  geom_point() +
  labs(
    x = "N per condition",
    y = "Simulated power"
  ) +
  theme_pubclean()
```



```r
#identify minimum sample size

power_df2_min = power_df_2 %>%
  filter(power > .95)

N_min2 = min(power_df2_min$N)
```

The simulation suggests that power would be over the threshold of .95 with a sample size of 75 participants per condition.

9 R version and packages

All data cleaning and analyses were completed using R version 4.1.0 (2021-05-18) (Camp Pontanezen). Below we list the packages (and versions) used in these analyses.

package	version	author
knitr	1.33	Yihui Xie [aut, cre] (<https://orcid.org/0000-0003-0645-5666>), Abhraneel Sarma [ctb], Adam Vogt [ctb], Alastair Andrew [ctb], Alex Zvoleff [ctb], Andre Simon [ctb] (the CSS files under inst/themes/ were derived from the Highlight package http://www.andre-simon.de), Aron Atkins [ctb], Aaron Wolen [ctb], Ashley Manton [ctb], Atsushi Yasumoto [ctb] (<https://orcid.org/0000-0002-8335-495X>), Ben Baumer [ctb], Brian Diggs [ctb], Brian Zhang [ctb], Bulat Yapparov [ctb], Cassio Pereira [ctb], Christophe Dervieux [ctb], David Hall [ctb], David Hugh-Jones [ctb], David Robinson [ctb], Doug Hemken [ctb], Duncan Murdoch [ctb], Elio Campitelli [ctb], Ellis Hughes [ctb], Emily Riederer [ctb], Fabian Hirschmann [ctb], Fitch Simeon [ctb], Forest Fang [ctb], Frank E Harrell Jr [ctb] (the Sweavel package at inst/misc/Sweavel.sty), Garrick Aden-Buie [ctb], Gregoire Detrez [ctb], Hadley Wickham [ctb], Hao Zhu [ctb], Heewon Jeon [ctb], Henrik Bengtsson [ctb], Hiroaki Yutani [ctb], Ian Lyttle [ctb], Hodges Daniel [ctb], Jake Burkhead [ctb], James Manton [ctb], Jared Lander [ctb], Jason Punyon [ctb], Javier Luraschi [ctb], Jeff Arnold [ctb], Jenny Bryan [ctb], Jeremy Ashkenas [ctb, cph] (the CSS file at inst/misc/docco-classic.css), Jeremy Stephens [ctb], Jim Hester [ctb], Joe Cheng [ctb], Johannes Ranke [ctb], John Honaker [ctb], John Muschelli [ctb], Jonathan Keane [ctb], JJ Allaire [ctb], Johan Toloe [ctb], Jonathan Sidi [ctb], Joseph Larmarange [ctb], Julien Barnier [ctb], Kaiyin Zhong [ctb], Kamil Slowikowski [ctb], Karl Forner [ctb], Kevin K. Smith [ctb], Kirill Mueller [ctb], Kohske Takahashi [ctb], Lorenz Walthert [ctb], Lucas Gallindo [ctb], Marius Hofert [ctb], Martin Modrák [ctb], Michael Chirico [ctb], Michael Friendly [ctb], Michal Bojanowski [ctb], Michel Kuhlmann [ctb], Miller Patrick [ctb], Nacho Caballero [ctb], Nick Salkowski [ctb], Niels Richard Hansen [ctb], Noam Ross [ctb], Obada Mahdi [ctb], Pavel N. Krivitsky [ctb] (<https://orcid.org/0000-0002-9101-3362>), Qiang Li [ctb], Ramnath Vaidyanathan [ctb], Richard Cotton [ctb], Robert Krzyzanowski [ctb], Romain Francois [ctb], Ruaridh Williamson [ctb], Scott Kostyshak [ctb], Sebastian Meyer [ctb], Sietse Brouwer [ctb], Simon de Bernard [ctb], Sylvain Rousseau [ctb], Taiyun Wei [ctb], Thibaut Assus [ctb], Thibaut Lamadon [ctb], Thomas Leeper [ctb], Tim Mastny [ctb], Tom Torsney-Weir [ctb], Trevor Davis [ctb], Viktoras Veitas [ctb], Weicheng Zhu [ctb], Wush Wu [ctb], Zachary Foster [ctb]
ggridges	0.5.3	Claus O. Wilke [aut, cre] (<https://orcid.org/0000-0002-7470-9261>)
emmeans	1.6.0	Russell V. Lenth [aut, cre, cph], Paul Buerkner [ctb], Maxime Herve [ctb], Jonathon Love [ctb], Hannes Riebl [ctb], Henrik Singmann [ctb]
broom.mixed	0.2.7	Ben Bolker [aut, cre] (<https://orcid.org/0000-0002-2127-0443>), David Robinson [aut], Dieter Menne [ctb], Jonah Gabry [ctb], Paul Buerkner [ctb], Christopher Hua [ctb], William Petry [ctb] (<https://orcid.org/0000-0002-5230-5987>), Joshua Wiley [ctb] (<https://orcid.org/0000-0002-0271-6702>), Patrick Kennedy [ctb], Eduard Szöcs [ctb] (<https://orcid.org/0000-0001-5376-1194>, BASF SE), Indrajeet Patil [ctb], Vincent Arel-Bundock [ctb] (<https://orcid.org/0000-0003-2042-7063>)
psych	2.1.6	William Revelle [aut, cre] (<https://orcid.org/0000-0003-4880-9610>)
papaja	0.1.0.9997	Frederik Aust [aut, cre] (<https://orcid.org/0000-0003-4900-788X>), Marius Barth [aut] (<https://orcid.org/0000-0002-3421-6665>), Birk Diedenhofen [ctb], Christoph Stahl [ctb], Joseph V. Casillas [ctb], Rudolf Siegel [ctb]

(continued)

package	version	author
stringdist	0.9.6.3	Mark van der Loo [aut, cre] (<https://orcid.org/0000-0002-9807-4686>), Jan van der Laan [ctb], R Core Team [ctb], Nick Logan [ctb], Chris Muir [ctb], Johannes Gruber [ctb]
kableExtra	1.3.4	Hao Zhu [aut, cre] (<https://orcid.org/0000-0002-3386-6076>), Thomas Travison [ctb], Timothy Tsai [ctb], Will Beasley [ctb], Yihui Xie [ctb], GuangChuang Yu [ctb], Stéphane Laurent [ctb], Rob Shepherd [ctb], Yoni Sidi [ctb], Brian Salzer [ctb], George Gui [ctb], Yeliang Fan [ctb], Duncan Murdoch [ctb], Bill Evans [ctb]
ggpubr	0.4.0	Alboukadel Kassambara [aut, cre]
sjPlot	2.8.8	Daniel Lüdecke [aut, cre] (<https://orcid.org/0000-0002-8895-3206>), Alexander Bartel [ctb] (<https://orcid.org/0000-0002-1280-6138>), Carsten Schwemmer [ctb], Chuck Powell [ctb] (<https://orcid.org/0000-0002-3606-2188>), Amir Djalovski [ctb], Johannes Titz [ctb] (<https://orcid.org/0000-0002-1102-5719>)
lmerTest	3.1-3	Alexandra Kuznetsova [aut], Per Bruun Brockhoff [aut, ths], Rune Haubo Bojesen Christensen [aut, cre], Sofie Pødenphant Jensen [ctb]
lme4	1.1-27	Douglas Bates [aut] (<https://orcid.org/0000-0001-8316-9503>), Martin Maechler [aut] (<https://orcid.org/0000-0002-8685-9910>), Ben Bolker [aut, cre] (<https://orcid.org/0000-0002-2127-0443>), Steven Walker [aut] (<https://orcid.org/0000-0002-4394-9078>), Rune Haubo Bojesen Christensen [ctb] (<https://orcid.org/0000-0002-4494-3399>), Henrik Singmann [ctb] (<https://orcid.org/0000-0002-4842-3657>), Bin Dai [ctb], Fabian Scheipl [ctb] (<https://orcid.org/0000-0001-8172-3603>), Gabor Grothendieck [ctb], Peter Green [ctb] (<https://orcid.org/0000-0002-0238-9852>), John Fox [ctb], Alexander Bauer [ctb], Pavel N. Krivitsky [ctb, cph] (<https://orcid.org/0000-0002-9101-3362>, shared copyright on simulate.formula)
Matrix	1.3-3	Douglas Bates [aut], Martin Maechler [aut, cre] (<https://orcid.org/0000-0002-8685-9910>), Timothy A. Davis [ctb] (SuiteSparse and 'cs' C libraries, notably CHOLMOD, AMD; collaborators listed in dir(pattern = '^[A-Z]+[.]txt$', full.names=TRUE, system.file('doc', 'SuiteSparse', package='Matrix'))), Jens Oehlschlägel [ctb] (initial nearPD()), Jason Riedy [ctb] (condest() and onenormest() for octave, Copyright: Regents of the University of California), R Core Team [ctb] (base R matrix implementation)
stringi	1.6.2	Marek Gagolewski [aut, cre, cph] (<https://orcid.org/0000-0003-0637-6028>), Bartek Tartanus [ctb], and others (stringi source code); IBM, Unicode, Inc. and others (ICU4C source code, Unicode Character Database)
janitor	2.1.0	Sam Firke [aut, cre], Bill Denney [ctb], Chris Haid [ctb], Ryan Knight [ctb], Malte Grosser [ctb], Jonathan Zadra [ctb]
forcats	0.5.1	Hadley Wickham [aut, cre], RStudio [cph, fnd]
stringr	1.4.0	Hadley Wickham [aut, cre, cph], RStudio [cph, fnd]
dplyr	1.0.6	Hadley Wickham [aut, cre] (<https://orcid.org/0000-0003-4757-117X>), Romain François [aut] (<https://orcid.org/0000-0002-2444-4226>), Lionel Henry [aut], Kirill Müller [aut] (<https://orcid.org/0000-0002-1416-3412>), RStudio [cph, fnd]
purrr	0.3.4	Lionel Henry [aut, cre], Hadley Wickham [aut], RStudio [cph, fnd]
readr	1.4.0	Hadley Wickham [aut], Jim Hester [aut, cre], Romain Francois [ctb], R Core Team [ctb] (Date time code adapted from R), RStudio [cph, fnd], Jukka Jylänki [ctb, cph] (grisu3 implementation), Mikkel Jørgensen [ctb, cph] (grisu3 implementation)
tidyr	1.1.3	Hadley Wickham [aut, cre], RStudio [cph]
tibble	3.1.2	Kirill Müller [aut, cre], Hadley Wickham [aut], Romain Francois [ctb], Jennifer Bryan [ctb], RStudio [cph]

(continued)

package	version	author
ggplot2	3.3.5	Hadley Wickham [aut] (<https://orcid.org/0000-0003-4757-117X>), Winston Chang [aut] (<https://orcid.org/0000-0002-1576-2126>), Lionel Henry [aut], Thomas Lin Pedersen [aut, cre] (<https://orcid.org/0000-0002-5147-4711>), Kohske Takahashi [aut], Claus Wilke [aut] (<https://orcid.org/0000-0002-7470-9261>), Kara Woo [aut] (<https://orcid.org/0000-0002-5125-4188>), Hiroaki Yutani [aut] (<https://orcid.org/0000-0002-3385-7233>), Dewey Dunnington [aut] (<https://orcid.org/0000-0002-9415-4582>), RStudio [cph, fnd]
tidyverse	1.3.1	Hadley Wickham [aut, cre], RStudio [cph, fnd]
here	1.0.1	Kirill Müller [aut, cre] (<https://orcid.org/0000-0002-1416-3412>), Jennifer Bryan [ctb] (<https://orcid.org/0000-0002-6983-2759>)