Supplemental file

Last updated 2021-07-23

Contents

Cleaning

Workspace

```
library(here) # for working with files
library(tidyverse) # for cleaning
library(janitor) # for variable names
library(lme4) # for multilevel modeling
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
```

```
data_path = here("data/Wording_July 13, 2021_20.00.text.csv")

data_labels = read_csv(data_path)

data = read_csv(data_path,
                skip = 3,
                col_names = names(data_labels))
rm(data_labels)
data = clean_names(data)
```

Remove the following columns.

```
data = data %>%
  select(-end_date,
         -ip_address,
         -progress,
         -finished,
         -recorded_date,
         -external_reference,
         -distribution_channel,
         -user_language,
         -starts_with("recipient"),
         -starts_with("location"),
         -starts_with("meta_info"))
```

Recode personality item responses to numeric

We recode the responses to personality items, which we downloaded as text strings.

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

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

Next we write a simple function to recode values.

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

Now we merge this back into the data.

```
data = select(data, -all_of(p_items))
data = full_join(data, personality_items)
```

Drop bots

Based on ID

We removed 5 participants without valid Prolific IDs.

```r
data = data %>%
  mutate(proid = str_remove(proid, "Value will be set from panel or URL"),
         proid = str_remove(proid, "Value will be set from panel or UR"),
         proid = str_remove(proid, "TEST")) %>%
  filter(proid != "")
```

We removed 0 participants that do not speak english well or very well.

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.

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

# extract block 1 questions
block1 = data %>%
  select(proid, matches("^[[:alpha:]]+_[abcd]$"))

#rename variables
n = 0
for(i in adjectives){
  n = n+1
  names(block1) = str_replace(names(block1), i, paste0("trait", str_pad(n, 2, pad = "0")))
}


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

block1_runs = numeric(length = nrow(block1))

#   working on this!!!
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
}
```

```r
#add to data frame
block1$block1_runs = block1_runs
```

```r
# extract block 2 questions
block2 = data %>%
  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))

#   working on this!!!
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 frame
block2$block2_runs = block2_runs
```

```r
#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"
    ))
```

```r
#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()
```

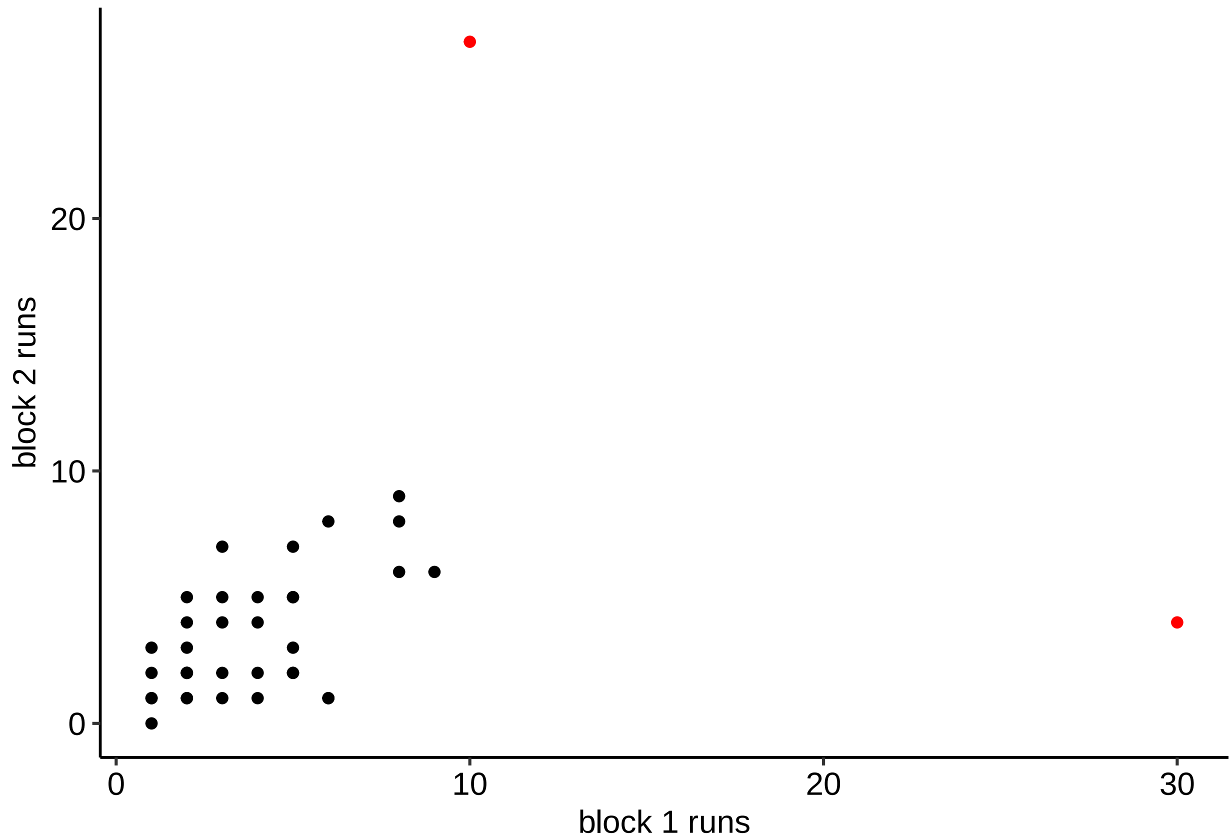

Figure 1: Maximum number of same consecutive responses in personality blocks.

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

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

rm(runs_data)
```

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.

```
in_average = data %>%
  # 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()
```

We remove 1 participants whose responses suggest inattention.

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

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 %>%
  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))
```

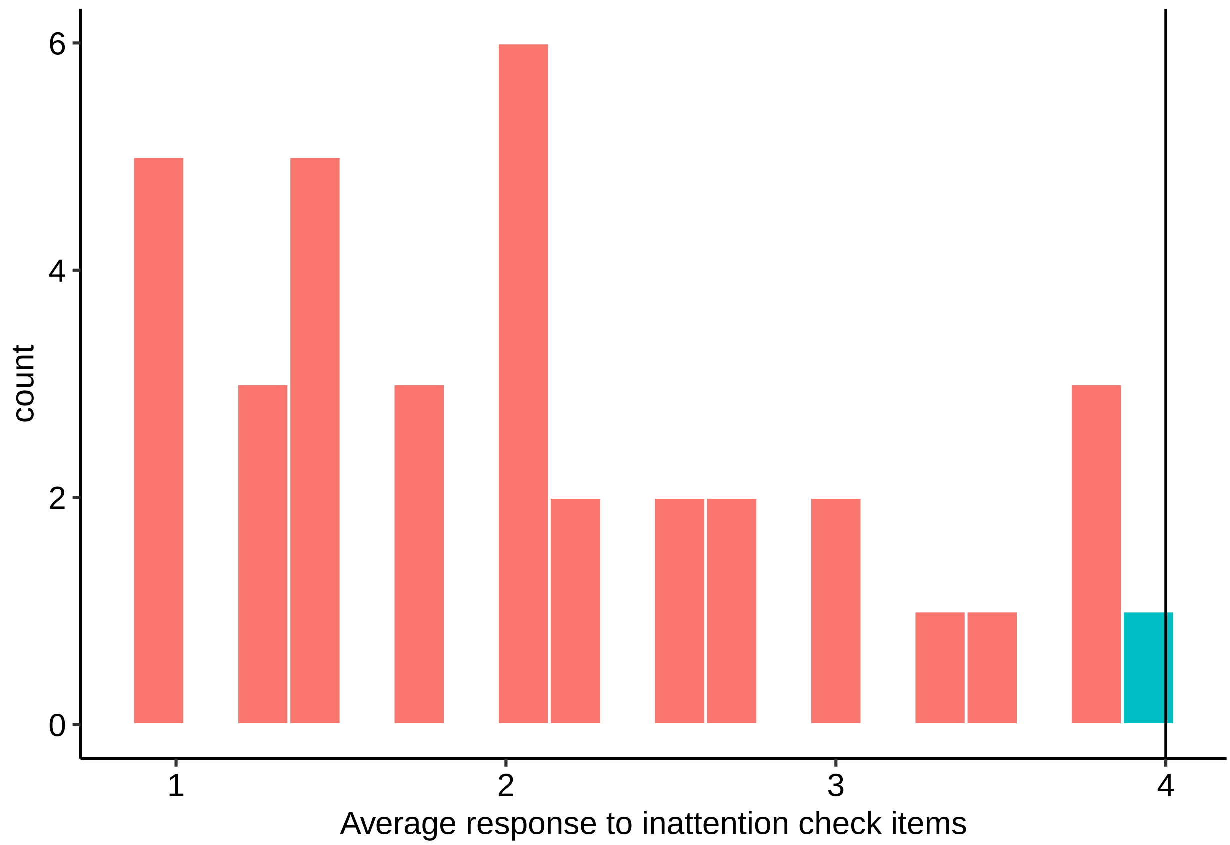

Figure 2: Average response to inattention check items

```
timing_data
```

```
## # A tibble: 2,170 x 3
##    proid                    variable                 timing
##    <chr>                    <chr>                     <dbl>
##  1 5f0227ae81bf2f3a4618c8c7 t_outgoing_a_page_submit   4.47
##  2 60eb3434de863fc43f563b0e t_outgoing_a_page_submit   4.52
##  3 60eb31222e8b2fb8dc904432 t_outgoing_a_page_submit   4.86
##  4 60e49d66c9c4f08ce7a3b789 t_outgoing_a_page_submit   4.16
##  5 60ea7b0e32a76a57b4a34664 t_outgoing_a_page_submit   4.20
##  6 60e950d879a14636c5fc286d t_outgoing_a_page_submit   4.49
##  7 60e781742748ec6401b79f86 t_outgoing_a_page_submit   4.5
##  8 60e9521961c670e718bcc4df t_outgoing_a_page_submit  85.7
##  9 60e777356e13630d745eeb49 t_outgoing_a_page_submit   2.47
## 10 60e99999b101ef725cb0b8a2 t_outgoing_a_page_submit   4.45
## # ... with 2,160 more rows
```

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()
```

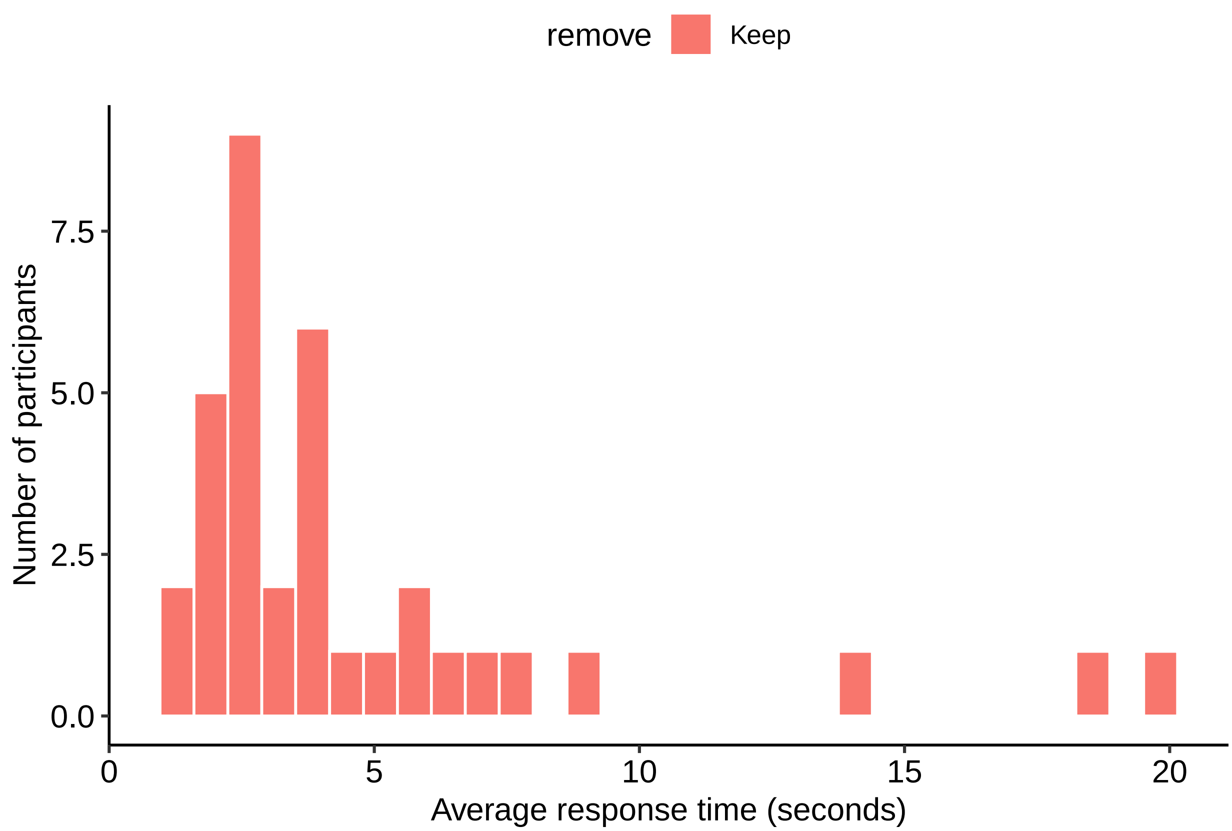

Figure 3: Distribution of average time to respond to personality items.

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

Based on timing, we removed 0 participants.

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 specturm.

```r
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))
```

Score memory task

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

```r
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.

```r
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 = ","))
```

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…

```
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 resposes, 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 resposnes, 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)
```

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()
```

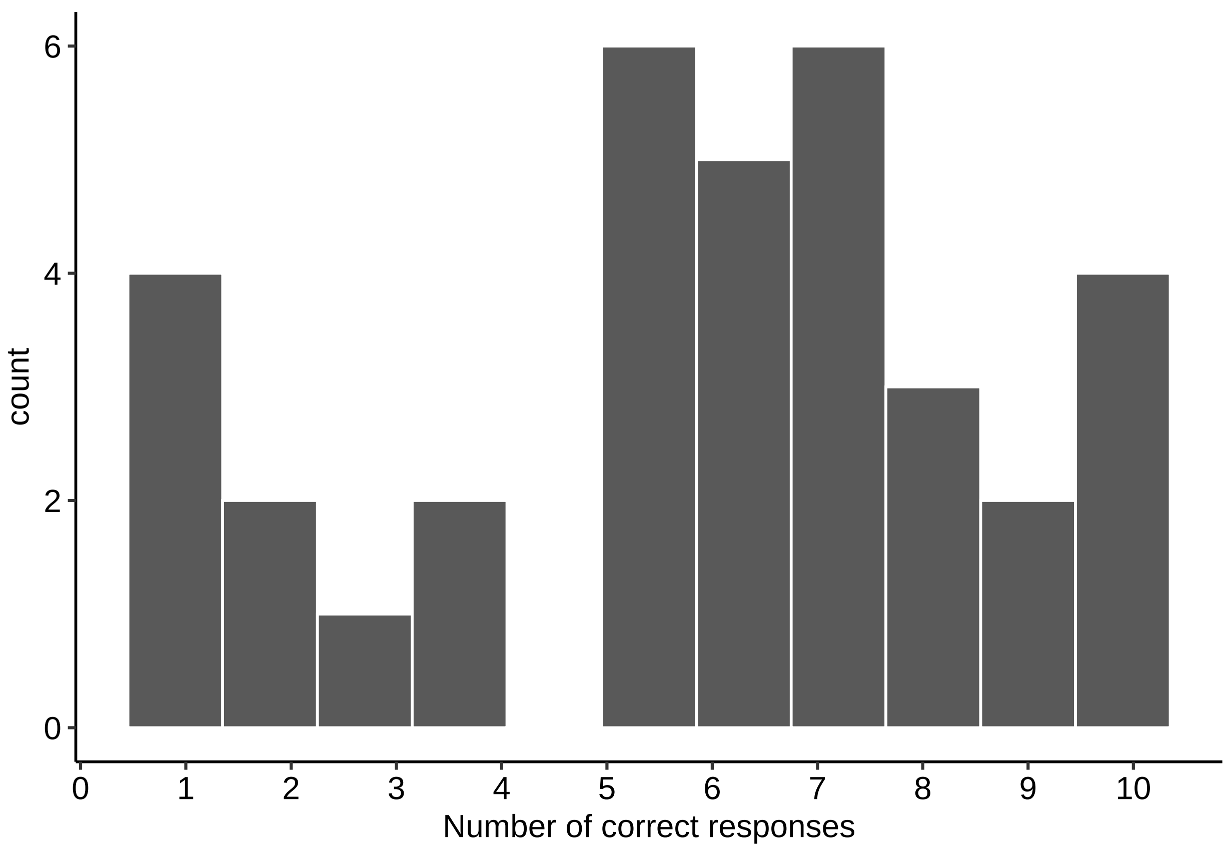

Figure 4: Correct responses on the memory task

```
data %>%
  group_by(mem_condition) %>%
  summarise(
    m = mean(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

```
    s = sd(memory),
    min = min(memory),
    max = max(memory),
    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()
```

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.

```
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)
```

```r
data = inner_join(data, mem2)
```

```r
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()
```

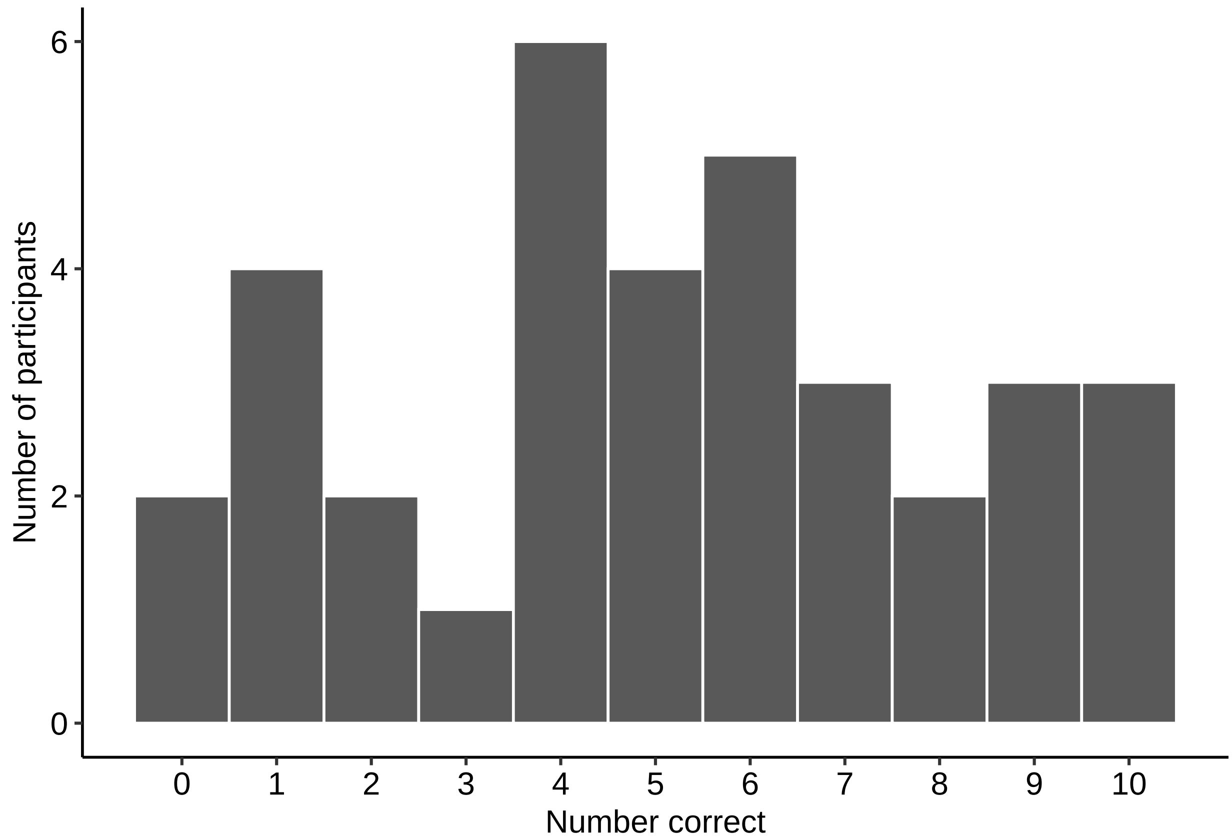

Figure 5: Distribution of delayed memory scores

```r
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()
```

Save data

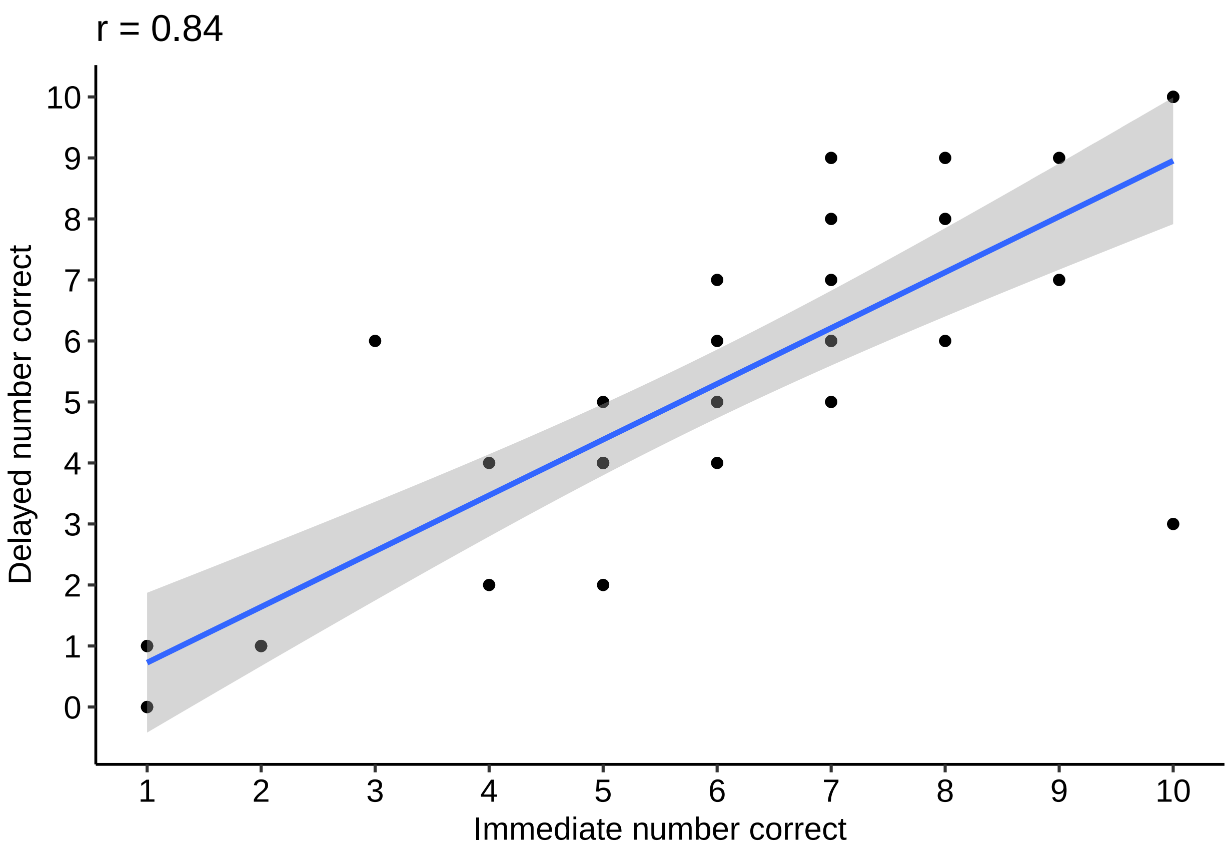

Figure 6: Relationship between immediate and delayed recall

Does item format affect response?

Using Block 1 Data only

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 memory effects by collecting data on immediate and delayed recall (5 minutes and approximately two weeks) using a memory paradigm that was developed based on a similar recall task used in the HRS (Runge et al., 2015).

Workspace

```
library(here) # for working with files
library(tidyverse) # for cleaning
library(janitor) # for variable names
library(lme4) # for mulitlevel modeling
library(lmerTest) # for p-values
library(emmeans) # for comparisons
library(sjPlot) # for figures
library(ggpubr) # for prettier plots
library(kableExtra) # for nicer tables
```

Data prep

We will use between-person analyses to compare responses using group-level data for the different formats.

First we select the responses to the items of different formats. For this set of analyses, we only use data collected in Block 1 – that is, each participant saw the same format for every item.

These variable names all have the same format: `[trait]_[abcd]` (for example, `talkative_a`). We search for these items using regular expressions.

```
items_seen_first = str_subset(
  names(data),
  "^([[:alpha:]])+_[abcd]$"
)

item_responses = data %>%
  select(proid, all_of(items_seen_first), memory)
```

Next we reshape these data into long form.

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

Response by Format

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

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

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

```
## Type III Analysis of Variance Table with Satterthwaite's method
##        Sum Sq Mean Sq NumDF DenDF F value Pr(>F)
## format  2.139 0.71299     3    31  0.4306 0.7325
```

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

plot1$format +
  labs(x = NULL,
       y = "Average response",
       title = "Average responses by item formatting") +
  theme_pubclean()
```

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

item_responses %>%
  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 = 125,
                label = paste("M =", round(m,2),
                              "\nSD =", round(s,2))),
            data = means_by_group,
            hjust =0,
            vjust = 1) +
  facet_wrap(~format) +
  guides(fill = F) +
  scale_x_continuous(breaks = 1:6) +
```

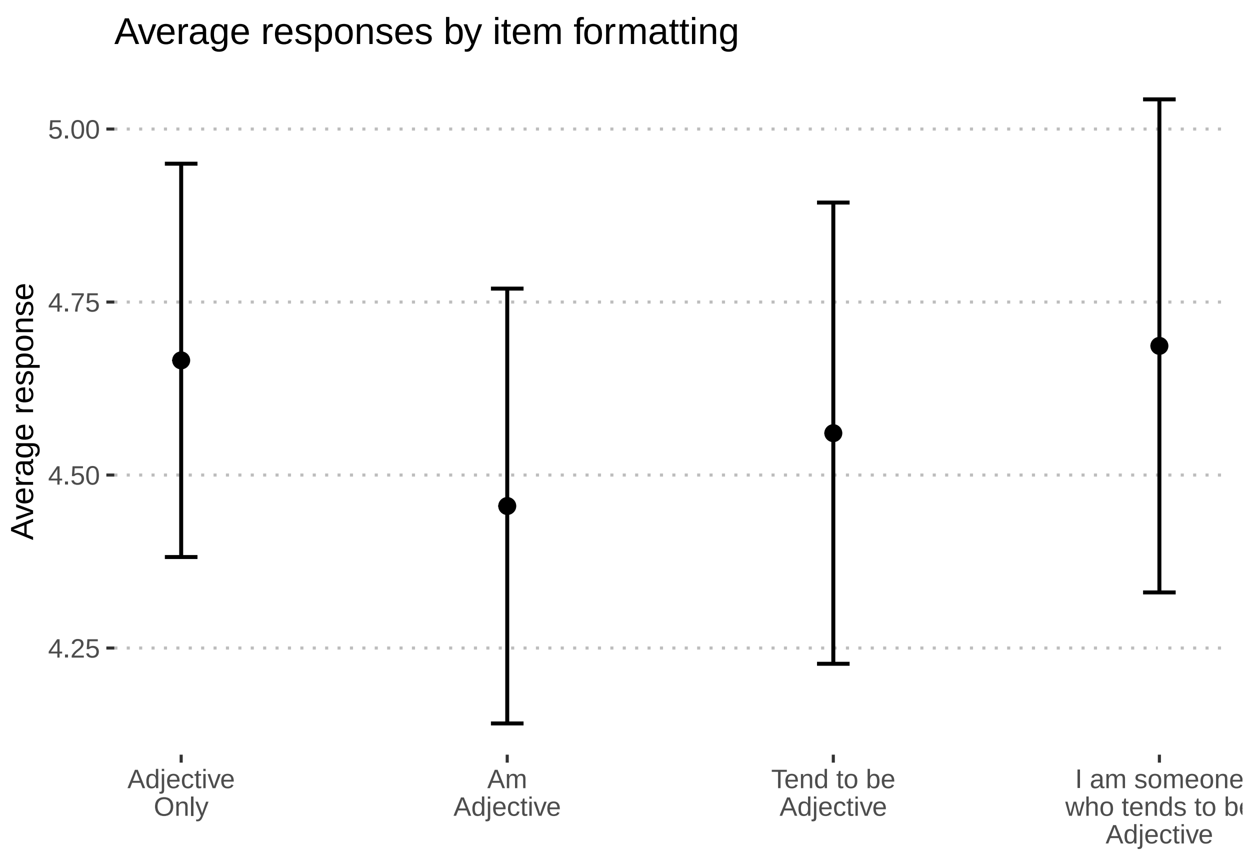

Figure 7: Predicted response on personality items by condition.

```
  labs(y = "Number of particpants",
       title = "Distribution of responses by format") +
  theme_pubr()
```

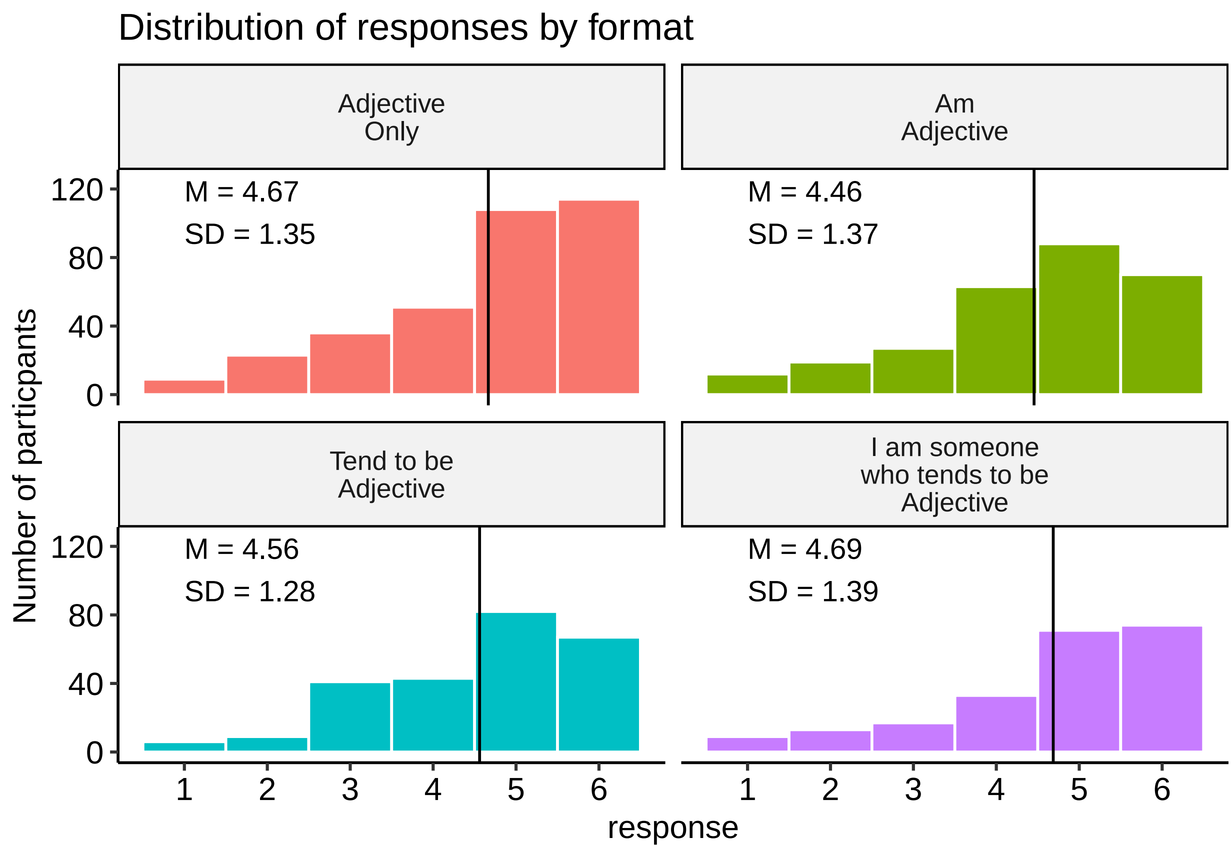

Figure 8: Distribution of responses by category

One model for each adjective

We can also repeat this analysis separately for each trait.

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

summary_by_item = mod_by_item %>%
  mutate(tidy = map(aov, broom::tidy)) %>%
  select(item, tidy) %>%
  unnest(cols = c(tidy)) %>%
  filter(term == "format") %>%
  select(-term, -df) %>%
  mutate(p.adj = p.adjust(p.value, method = "holm"))
```

item	sumsq	meansq	statistic	p.value	p.adj
outgoing	1.58	0.53	0.40	0.76	0.76
helpful	0.20	0.07	0.08	0.97	0.97
reckless	5.14	1.71	0.65	0.59	0.59
moody	6.21	2.07	1.14	0.35	0.35
organized	2.20	0.73	0.66	0.58	0.58
friendly	0.52	0.17	0.34	0.80	0.80
warm	0.71	0.24	0.20	0.90	0.90
worrying	8.95	2.98	1.25	0.31	0.31
responsible	0.87	0.29	0.65	0.59	0.59
lively	1.15	0.38	0.36	0.78	0.78
asleep	0.89	0.30	0.13	0.94	0.94
caring	2.47	0.82	1.17	0.34	0.34
nervous	1.54	0.51	0.20	0.89	0.89
creative	2.35	0.78	0.79	0.51	0.51
hardworking	0.03	0.01	0.02	1.00	1.00
imaginative	1.40	0.47	0.58	0.63	0.63
softhearted	1.25	0.42	0.30	0.83	0.83
calm	3.77	1.26	1.29	0.30	0.30
intelligent	1.86	0.62	0.44	0.73	0.73
curious	0.52	0.17	0.15	0.93	0.93
active	2.80	0.93	0.61	0.61	0.61
human	4.25	1.42	3.03	0.04	0.04
careless	18.23	6.08	2.77	0.06	0.06
impulsive	2.65	0.88	0.45	0.72	0.72
sympathetic	3.42	1.14	1.10	0.36	0.36
cautious	1.55	0.52	0.32	0.81	0.81
talkative	18.53	6.18	3.19	0.04	0.04
sophisticated	1.08	0.36	0.21	0.89	0.89
adventurous	1.34	0.45	0.50	0.68	0.68
thorough	0.33	0.11	0.08	0.97	0.97
thrifty	8.09	2.70	2.06	0.13	0.13

```
summary_by_item %>%
  mutate(across(
    starts_with("p"),
    papaja::printnum
  )) %>%
  kable(digits = 2, booktabs = T) %>%
  kable_styling()
```

Pairwise t-tests for significant ANOVAs

Here we identify the specific items with significant differences.

```
sig_item = summary_by_item %>%
  filter(p.value < .05)
```

Contrast	Difference in means	SE	df	t	p
Adjective Only - Am Adjective	0.48	0.31	31	1.58	0.499
Adjective Only - Tend to be Adjective	0.82	0.32	31	2.58	0.090
Adjective Only - I am someone who tends to be Adjective	-0.04	0.33	31	-0.12	0.907
Am Adjective - Tend to be Adjective	0.33	0.33	31	1.00	0.647
Am Adjective - I am someone who tends to be Adjective	-0.52	0.34	31	-1.52	0.499
Tend to be Adjective - I am someone who tends to be Adjective	-0.86	0.35	31	-2.42	0.107

```
sig_item = sig_item$item
sig_item
```

```
## [1] "human"     "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!

Human

```
human_model = item_responses %>%
  filter(item == "human") %>%
  lm(response~format, data = .)

human_em = emmeans(human_model, "format")
pairs(human_em, adjust = "holm") %>%
  kable(booktabs = T,
        digits = c(0,2,2,2,2,3),
        col.names = c("Contrast", "Difference in means", "SE", "df", "t", "p")) %>%
  kable_styling()
```

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

Talkative

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

talkative_em = emmeans(talkative_model, "format")
pairs(talkative_em, adjust = "holm") %>%
  kable(booktabs = T,
        digits = c(0,2,2,2,2,3),
        col.names = c("Contrast", "Difference in means", "SE", "df", "t", "p")) %>%
  kable_styling()
```

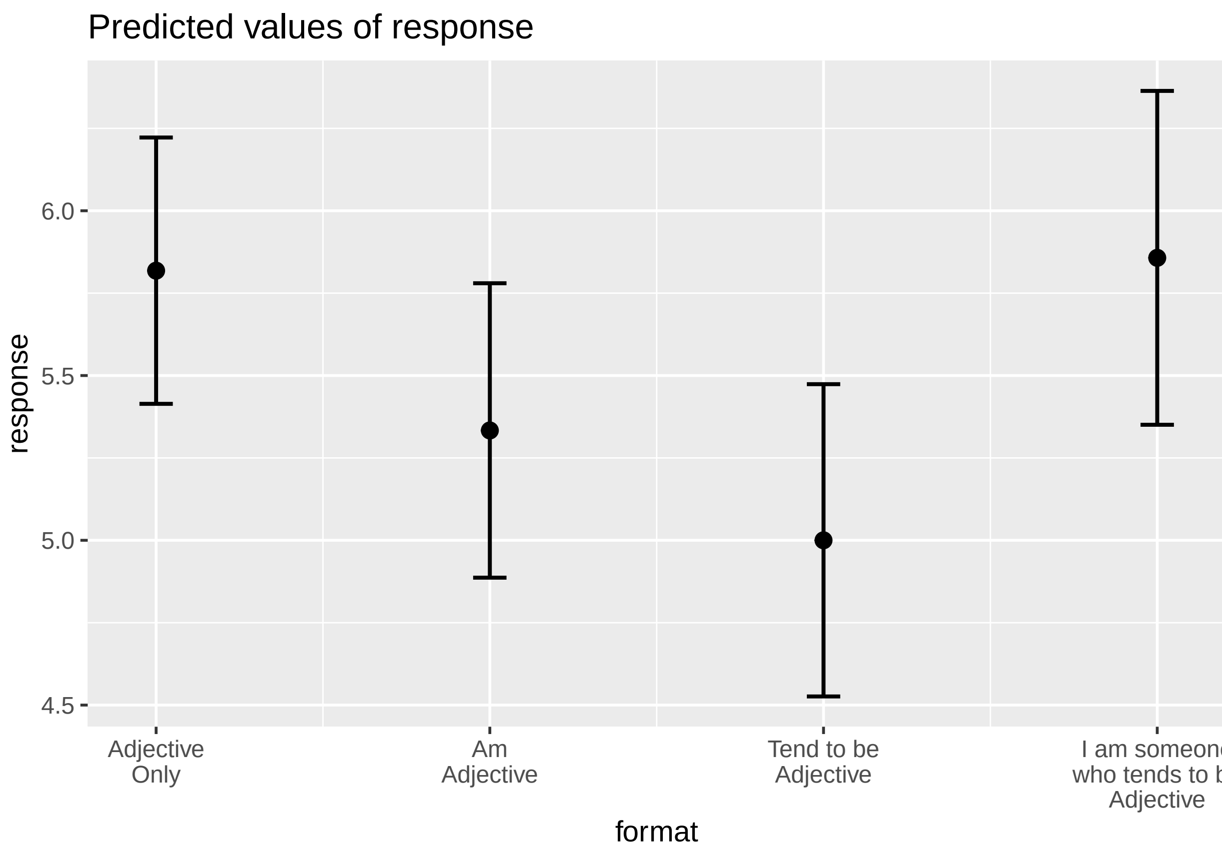

Figure 9: Average response to “human” by format

Contrast	Difference in means	SE	df	t	p
Adjective Only - Am Adjective	-1.49	0.63	31	-2.39	0.139
Adjective Only - Tend to be Adjective	-1.40	0.65	31	-2.16	0.192
Adjective Only - I am someone who tends to be Adjective	0.01	0.67	31	0.02	1.000
Am Adjective - Tend to be Adjective	0.10	0.68	31	0.14	1.000
Am Adjective - I am someone who tends to be Adjective	1.51	0.70	31	2.15	0.192
Tend to be Adjective - I am someone who tends to be Adjective	1.41	0.72	31	1.96	0.192

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

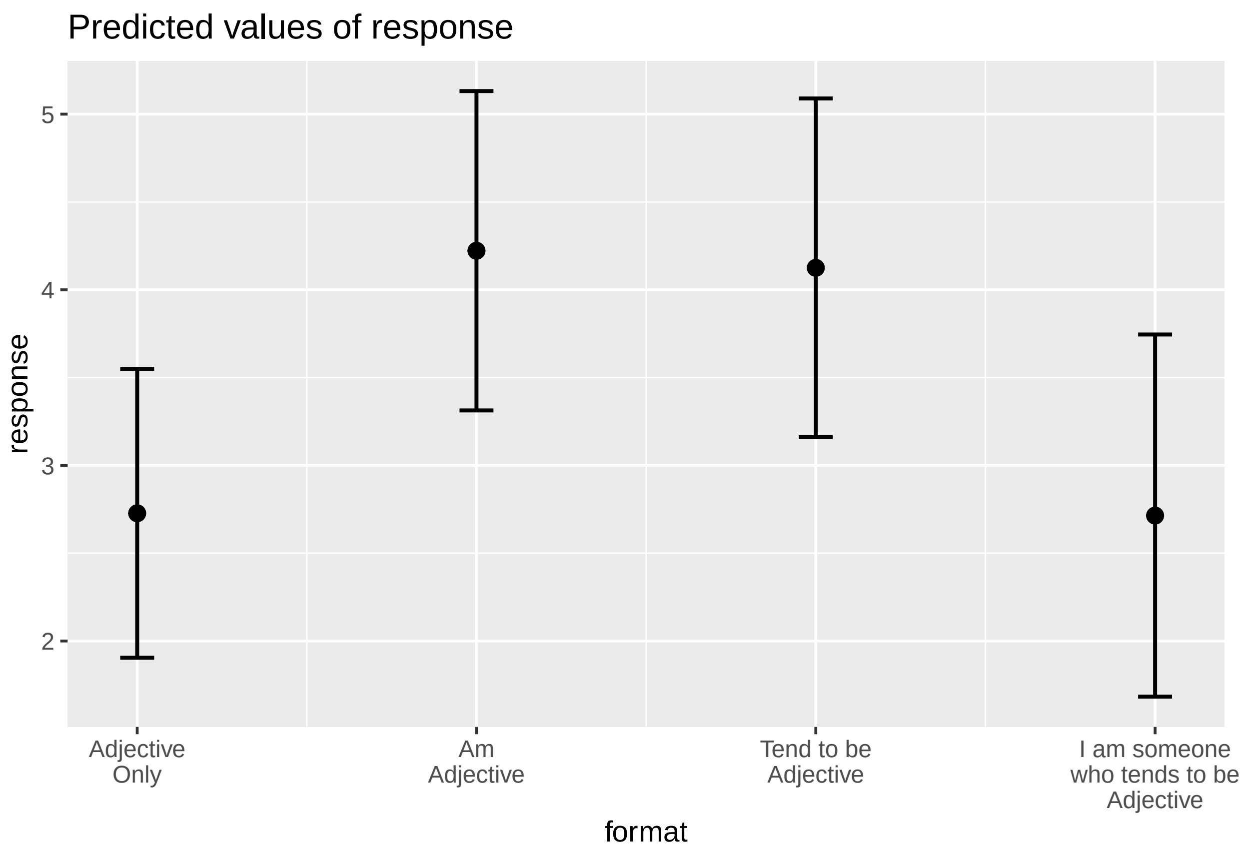

Figure 10: Average response to "talkative" by format

Response by Format + Memory

```r
mod.format_mem = lmer(response~format + memory + (1|proid),
                      data = item_responses)
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 2.4560 0.81866     3    30  0.4944 0.6889
## memory 3.1107 3.11066     1    30  1.8787 0.1807
```

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

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: response ~ format + memory + (1 | proid)
##    Data: item_responses
##
```

```
## REML criterion at convergence: 3684
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.3168 -0.5233  0.2396  0.7158  1.5598
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  proid    (Intercept) 0.1715   0.4141
##  Residual             1.6558   1.2868
## Number of obs: 1085, groups:  proid, 35
##
## Fixed effects:
##                                                  Estimate Std. Error       df
## (Intercept)                                       4.42246    0.22790 30.00000
## formatAm\nAdjective                              -0.22338    0.21337 30.00000
## formatTend to be\nAdjective                      -0.07808    0.22126 30.00000
## formatI am someone\nwho tends to be\nAdjective    0.02897    0.22938 30.00000
## memory                                            0.04116    0.03003 30.00000
##                                                  t value Pr(>|t|)
## (Intercept)                                       19.405   <2e-16 ***
## formatAm\nAdjective                               -1.047    0.304
## formatTend to be\nAdjective                       -0.353    0.727
## formatI am someone\nwho tends to be\nAdjective     0.126    0.900
## memory                                             1.371    0.181
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) frmtAA frTtbA faswttbA
## frmtAmAdjct -0.386
## frmtTndtbAd -0.475  0.429
## frmIaswttbA -0.411  0.417  0.405
## memory      -0.779 -0.044  0.089  0.026
```

```
plot_model(mod.format_mem, type = "pred", term = c("format"))
```

```
plot_model(mod.format_mem, type = "pred", term = c("memory"))
```

One model for each adjective

```
mod_by_item = item_responses %>%
  group_by(item) %>%
  nest() %>%
  mutate(mod = map(data, ~lm(response~format + memory, data = .))) %>%
  mutate(aov = map(mod, anova))

summary_by_item = mod_by_item %>%
  mutate(tidy = map(aov, broom::tidy)) %>%
  select(item, tidy) %>%
```

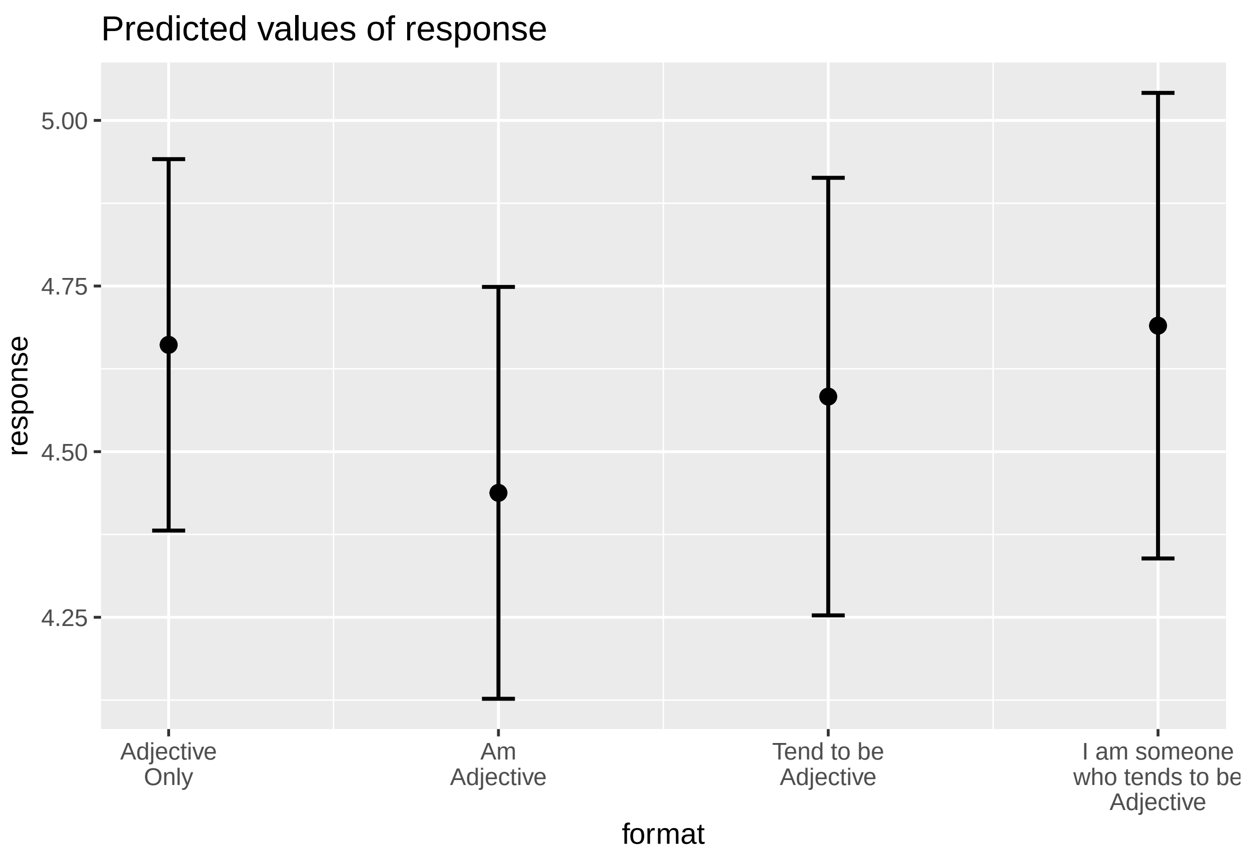

Figure 11: Predicted response on personality items by condition after controlling for memory.

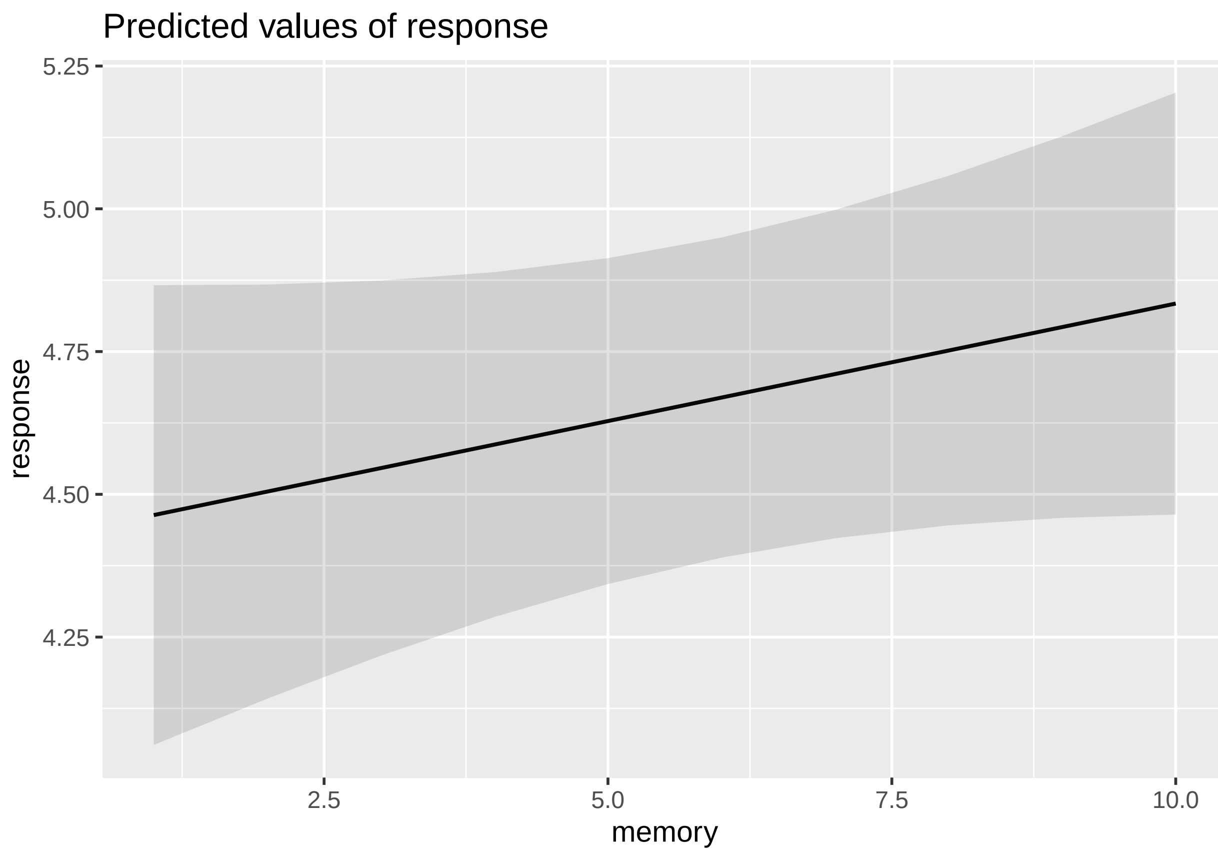

Figure 12: Predicted response on personality items by memory.

item	sumsq	meansq	statistic	p.value	p.adj
outgoing	1.58	0.53	0.39	0.76	0.76
helpful	0.20	0.07	0.08	0.97	0.97
reckless	5.14	1.71	0.63	0.60	0.60
moody	6.21	2.07	1.16	0.34	0.34
organized	2.20	0.73	0.68	0.57	0.57
friendly	0.52	0.17	0.36	0.79	0.79
warm	0.71	0.24	0.24	0.87	0.87
worrying	8.95	2.98	1.21	0.32	0.32
responsible	0.87	0.29	0.63	0.60	0.60
lively	1.15	0.38	0.35	0.79	0.79
asleep	0.89	0.30	0.13	0.94	0.94
caring	2.47	0.82	1.16	0.34	0.34
nervous	1.54	0.51	0.20	0.90	0.90
creative	2.35	0.78	0.81	0.50	0.50
hardworking	0.03	0.01	0.02	1.00	1.00
imaginative	1.40	0.47	0.60	0.62	0.62
softhearted	1.25	0.42	0.30	0.83	0.83
calm	3.77	1.26	1.30	0.29	0.29
intelligent	1.86	0.62	0.44	0.72	0.72
curious	0.52	0.17	0.15	0.93	0.93
active	2.80	0.93	0.60	0.62	0.62
human	4.25	1.42	2.94	0.05	0.05
careless	18.23	6.08	2.87	0.05	0.05
impulsive	2.65	0.88	0.44	0.73	0.73
sympathetic	3.42	1.14	1.08	0.37	0.37
cautious	1.55	0.52	0.31	0.81	0.81
talkative	18.53	6.18	3.10	0.04	0.04
sophisticated	1.08	0.36	0.21	0.89	0.89
adventurous	1.34	0.45	0.49	0.69	0.69
thorough	0.33	0.11	0.09	0.97	0.97
thrifty	8.09	2.70	1.99	0.14	0.14

```
  unnest(cols = c(tidy)) %>%
  filter(term == "format") %>%
  select(-term, -df) %>%
  mutate(p.adj = p.adjust(p.value, method = "holm"))

summary_by_item %>%
  mutate(across(
    starts_with("p"),
    papaja::printnum
  )) %>%
  kable(digits = 2, booktabs = T) %>%
  kable_styling()
```

Here we identify the specific items with significant differences.

Contrast	Difference in means	SE	df	t	p
Adjective Only - Am Adjective	0.48	0.31	30	1.54	0.533
Adjective Only - Tend to be Adjective	0.82	0.32	30	2.54	0.099
Adjective Only - I am someone who tends to be Adjective	-0.04	0.34	30	-0.11	0.912
Am Adjective - Tend to be Adjective	0.34	0.34	30	1.00	0.647
Am Adjective - I am someone who tends to be Adjective	-0.52	0.35	30	-1.48	0.533
Tend to be Adjective - I am someone who tends to be Adjective	-0.86	0.36	30	-2.39	0.116

```
sig_item = summary_by_item %>%
  filter(p.value < .05)

sig_item = sig_item$item
sig_item
```

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

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

Human

```
human_model = item_responses %>%
  filter(item == "human") %>%
  lm(response~format + memory, data = .)

human_em = emmeans(human_model, "format")
pairs(human_em, adjust = "holm") %>%
  kable(booktabs = T,
        digits = c(0,2,2,2,2,3),
        col.names = c("Contrast", "Difference in means", "SE", "df", "t", "p")) %>%
  kable_styling()
```

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

Talkative

```
talkative_model = item_responses %>%
  filter(item == "talkative") %>%
  lm(response~format + memory, data = .)

talkative_em = emmeans(talkative_model, "format")
pairs(talkative_em, adjust = "holm") %>%
  kable(booktabs = T,
        digits = c(0,2,2,2,2,3),
        col.names = c("Contrast", "Difference in means", "SE", "df", "t", "p")) %>%
  kable_styling()
```

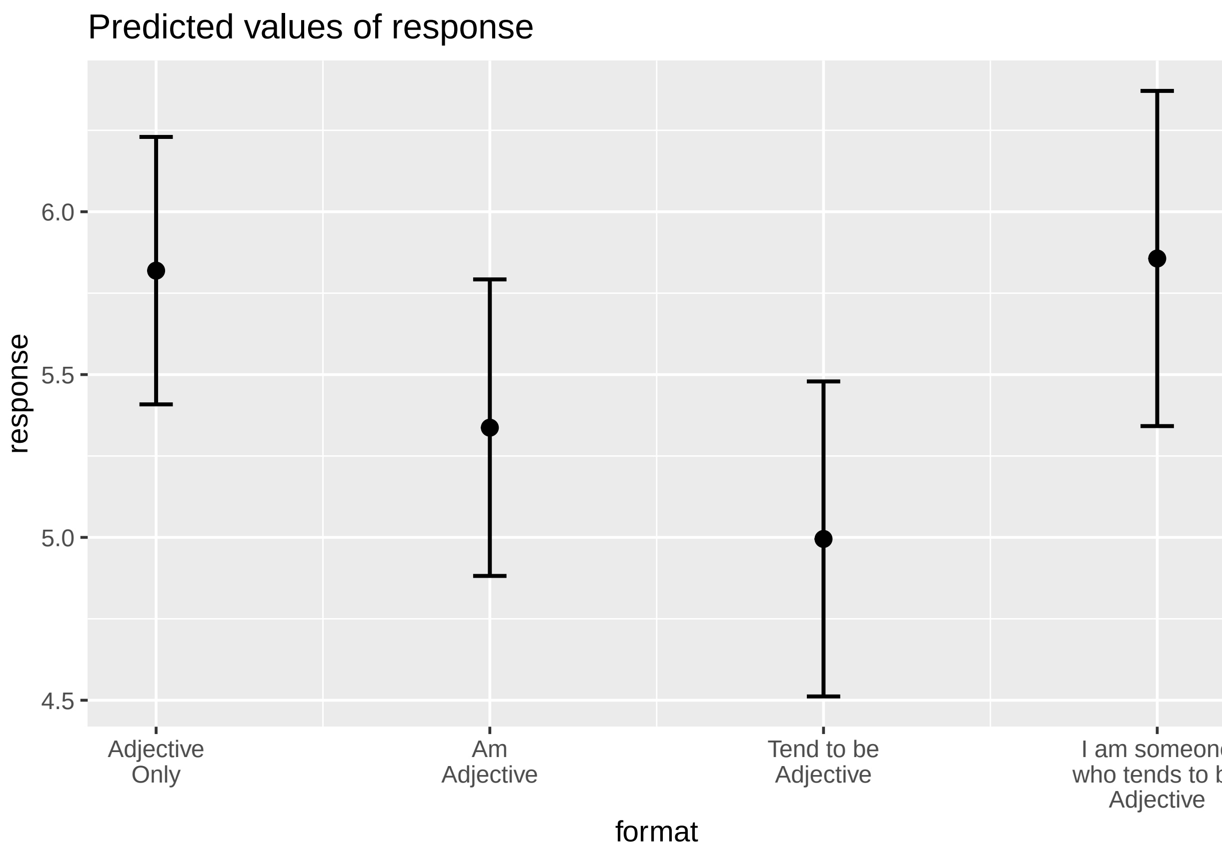

Figure 13: Average response to “human” by format

Contrast	Difference in means	SE	df	t	p
Adjective Only - Am Adjective	-1.51	0.63	30	-2.37	0.146
Adjective Only - Tend to be Adjective	-1.38	0.66	30	-2.09	0.203
Adjective Only - I am someone who tends to be Adjective	0.02	0.68	30	0.03	1.000
Am Adjective - Tend to be Adjective	0.13	0.69	30	0.19	1.000
Am Adjective - I am someone who tends to be Adjective	1.53	0.71	30	2.14	0.203
Tend to be Adjective - I am someone who tends to be Adjective	1.39	0.73	30	1.91	0.203

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

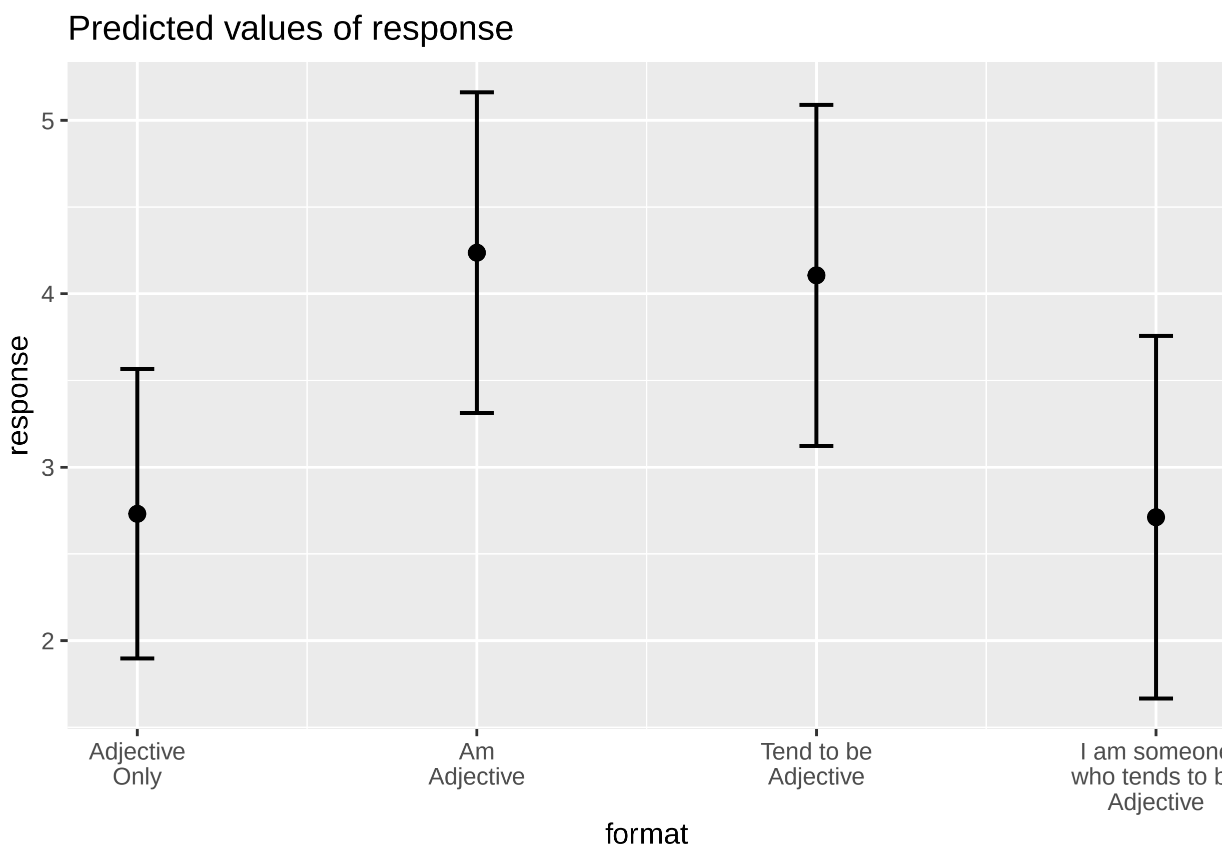

Figure 14: Average response to "talkative" by format

Questions

- I only used responses in Block 1 here. Should I merge the two blocks together (see here for code that does that)? Or I can repeat the analyses using Block 2? One thing to consider if we merge is that I can't use the simple ANOVA for the item-level analyses (we'll have two responses per person, so back to a nested design).
- When looking at individual items, should we use the p-value or the adjusted p-value to determine which to follow up with? To be clear, in supplemental analyses, we should look at all – do we report all in the manuscript or just the ones significant after the Holm correction?

Using Block 1 and Block 2

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 memory effects by collecting data on immediate and delayed recall (5 minutes and approximately two weeks) using a memory paradigm that was developed based on a similar recall task used in the HRS (Runge et al., 2015).

Workspace

```
library(here) # for working with files
library(tidyverse) # for cleaning
library(janitor) # for variable names
library(lme4) # for mulitlevel modeling
library(lmerTest) # for df and p-values
library(emmeans) # for pairwise comparisons
library(sjPlot) # for figures
library(ggpubr) # for prettier plots
library(kableExtra) # for nicer tables
```

Data prep

We will use between-person analyses to compare responses using group-level data for the different formats.

First we 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 two formats: `[trait]_[abcd]` (for example, `talkative_a`) or `[trait]_[abcd]_2` (for example, `talkative_a_2`). We search for these items using regular expressions.

```
items_seen_b1b2 = str_subset(
  names(data),
  "^([[:alpha:]])+_[abcd](_2)?$"
)

item_responses = data %>%
  select(proid, all_of(items_seen_b1b2), memory)
```

Next we reshape these data into long form.

```
item_responses = item_responses %>%
  gather(item, response, -proid, -memory) %>%
  mutate(
    block = case_when(              # which block is the item from?
      str_detect(item, "_2") ~ "2",
      TRUE ~ "1"),
    item = str_remove(item, "_2")) %>%  # remove block id from item string
  separate(item, into = c("item", "format")) %>%
  filter(!is.na(response))
```

Response by Format

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

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

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

```
## Type III Analysis of Variance Table with Satterthwaite's method
##        Sum Sq Mean Sq NumDF  DenDF F value Pr(>F)
## format 4.0209  1.3403     3 2031.2  0.7886 0.5002
```

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

plot1$format +
  labs(x = NULL,
       y = "Average response",
       title = "Average responses by item formatting") +
  theme_pubclean()
```

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

item_responses %>%
  ggplot(aes(x = response)) +
  geom_histogram(aes(fill = block),
                 position = "dodge",
                 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") +
  theme_pubr()
```

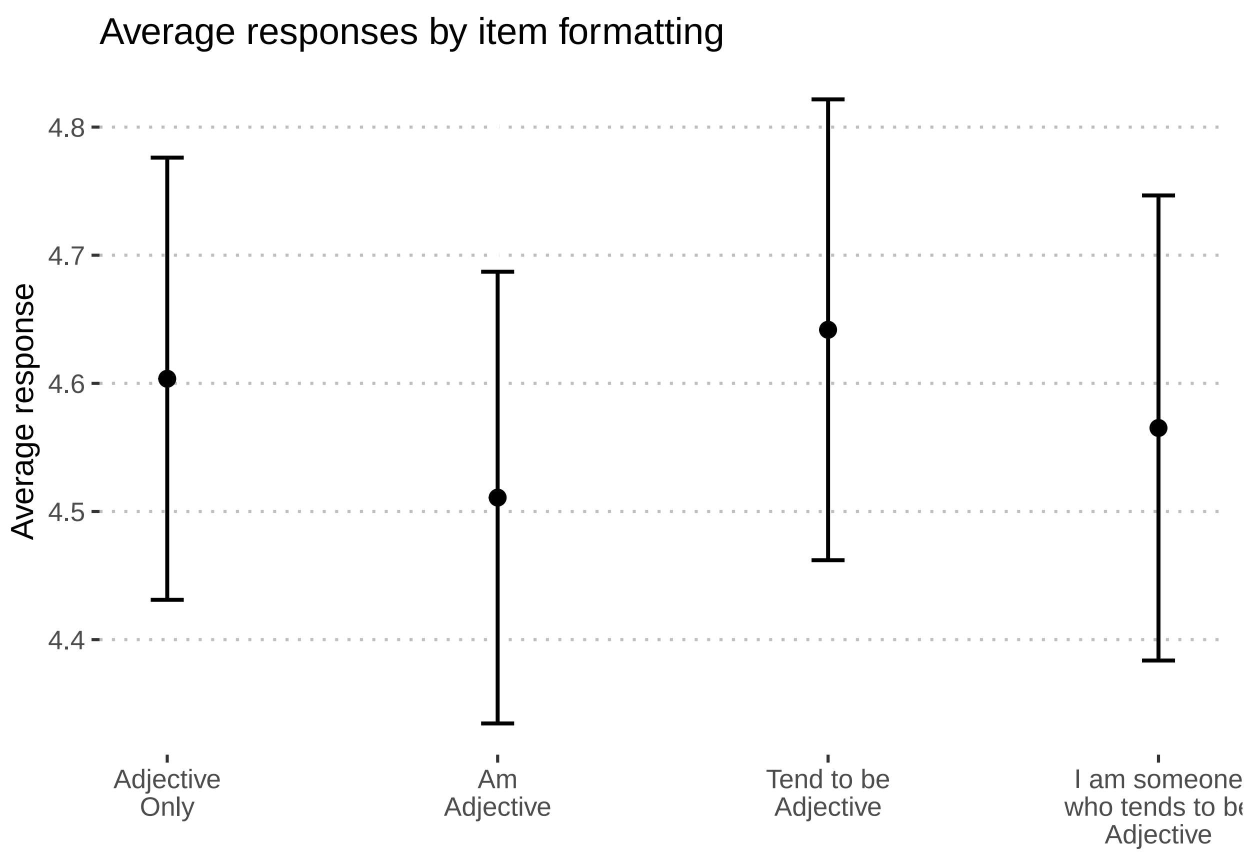

Figure 15: Predicted response on personality items by condition.

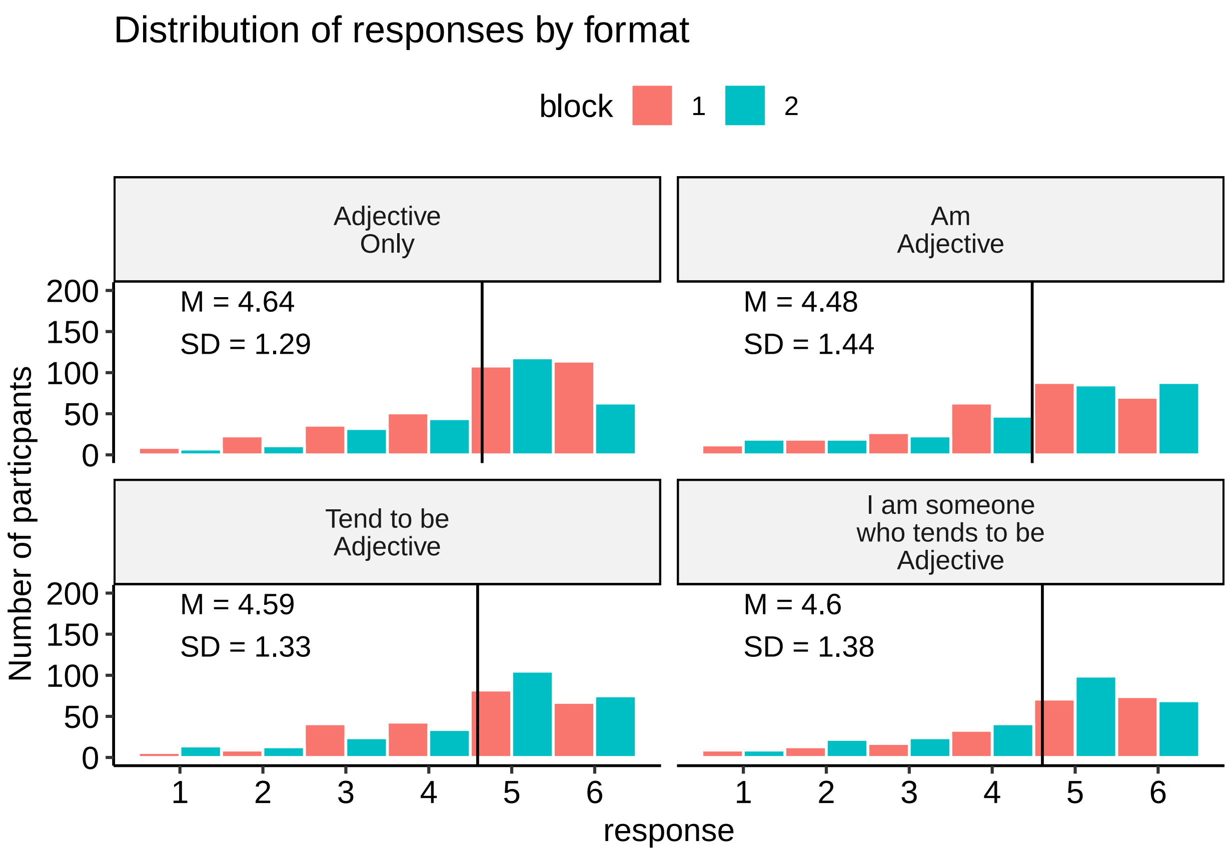

Figure 16: Distribution of responses by category

One model for each adjective

We can also repeat this analysis separately for each trait.

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

summary_by_item = mod_by_item %>%
  mutate(tidy = map(aov, broom::tidy)) %>%
  select(item, tidy) %>%
  unnest(cols = c(tidy)) %>%
  filter(term == "format") %>%
  select(-term) %>%
  mutate(p.adj = p.adjust(p.value, method = "holm"))

summary_by_item %>%
  mutate(across(
    starts_with("p"),
    papaja::printnum
  )) %>%
  kable(digits = 2, booktabs = T) %>%
  kable_styling()
```

Pairwise t-tests for significant ANOVAs

Here we identify the specific items with significant differences.

```
sig_item = summary_by_item %>%
  filter(p.value < .05)

sig_item = sig_item$item
sig_item
```

```
## [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!

Careless

```
careless_model = item_responses %>%
  filter(item == "careless") %>%
  lmer(response~format + (1|proid),
```

item	sumsq	meansq	NumDF	DenDF	statistic	p.value	p.adj
outgoing	0.71	0.24	3	39.34	0.83	0.48	0.48
helpful	0.90	0.30	3	60.54	0.82	0.49	0.49
reckless	1.14	0.38	3	48.60	0.36	0.78	0.78
moody	1.21	0.40	3	42.77	0.69	0.57	0.57
organized	0.22	0.07	3	37.39	0.41	0.75	0.75
friendly	0.94	0.31	3	46.85	1.45	0.24	0.24
warm	0.10	0.03	3	52.22	0.08	0.97	0.97
worrying	1.72	0.57	3	41.67	0.63	0.60	0.60
responsible	1.51	0.50	3	56.50	0.82	0.49	0.49
lively	1.08	0.36	3	42.77	0.89	0.46	0.46
asleep	0.53	0.18	3	42.91	0.28	0.84	0.84
caring	1.16	0.39	3	52.66	1.28	0.29	0.29
nervous	1.35	0.45	3	46.22	0.45	0.72	0.72
creative	0.14	0.05	3	49.22	0.13	0.94	0.94
hardworking	2.10	0.70	3	52.03	2.06	0.12	0.12
imaginative	1.03	0.34	3	55.07	0.86	0.47	0.47
softhearted	1.95	0.65	3	57.46	0.74	0.53	0.53
calm	0.13	0.04	3	45.02	0.13	0.94	0.94
intelligent	1.72	0.57	3	51.16	1.15	0.34	0.34
curious	2.35	0.78	3	52.00	1.05	0.38	0.38
active	0.76	0.25	3	40.00	1.01	0.40	0.40
human	0.28	0.09	3	41.14	0.82	0.49	0.49
careless	8.29	2.76	3	53.32	2.84	0.05	0.05
impulsive	1.38	0.46	3	41.41	0.85	0.47	0.47
sympathetic	0.49	0.16	3	51.48	0.51	0.68	0.68
cautious	2.30	0.77	3	54.97	0.98	0.41	0.41
talkative	6.36	2.12	3	43.86	1.95	0.13	0.13
sophisticated	0.05	0.02	3	44.04	0.04	0.99	0.99
adventurous	1.51	0.50	3	48.65	1.09	0.36	0.36
thorough	2.23	0.74	3	40.16	2.72	0.06	0.06
thrifty	6.39	2.13	3	51.69	3.30	0.03	0.03

Contrast	Difference in means	SE	df	t	p
Adjective Only - Am Adjective	0.82	0.45	58.04	1.83	0.359
Adjective Only - Tend to be Adjective	0.14	0.44	54.32	0.32	0.749
Adjective Only - I am someone who tends to be Adjective	-0.44	0.44	54.32	-1.01	0.633
Am Adjective - Tend to be Adjective	-0.68	0.45	51.43	-1.52	0.542
Am Adjective - I am someone who tends to be Adjective	-1.26	0.45	51.43	-2.82	0.041
Tend to be Adjective - I am someone who tends to be Adjective	-0.58	0.45	49.48	-1.31	0.591

```
                              data = .)

careless_em = emmeans(careless_model, "format")
pairs(careless_em, adjust = "holm") %>%
  kable(booktabs = T,
        digits = c(0,2,2,2,2,3),
        col.names = c("Contrast", "Difference in means", "SE", "df", "t", "p")) %>%
  kable_styling()
```

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

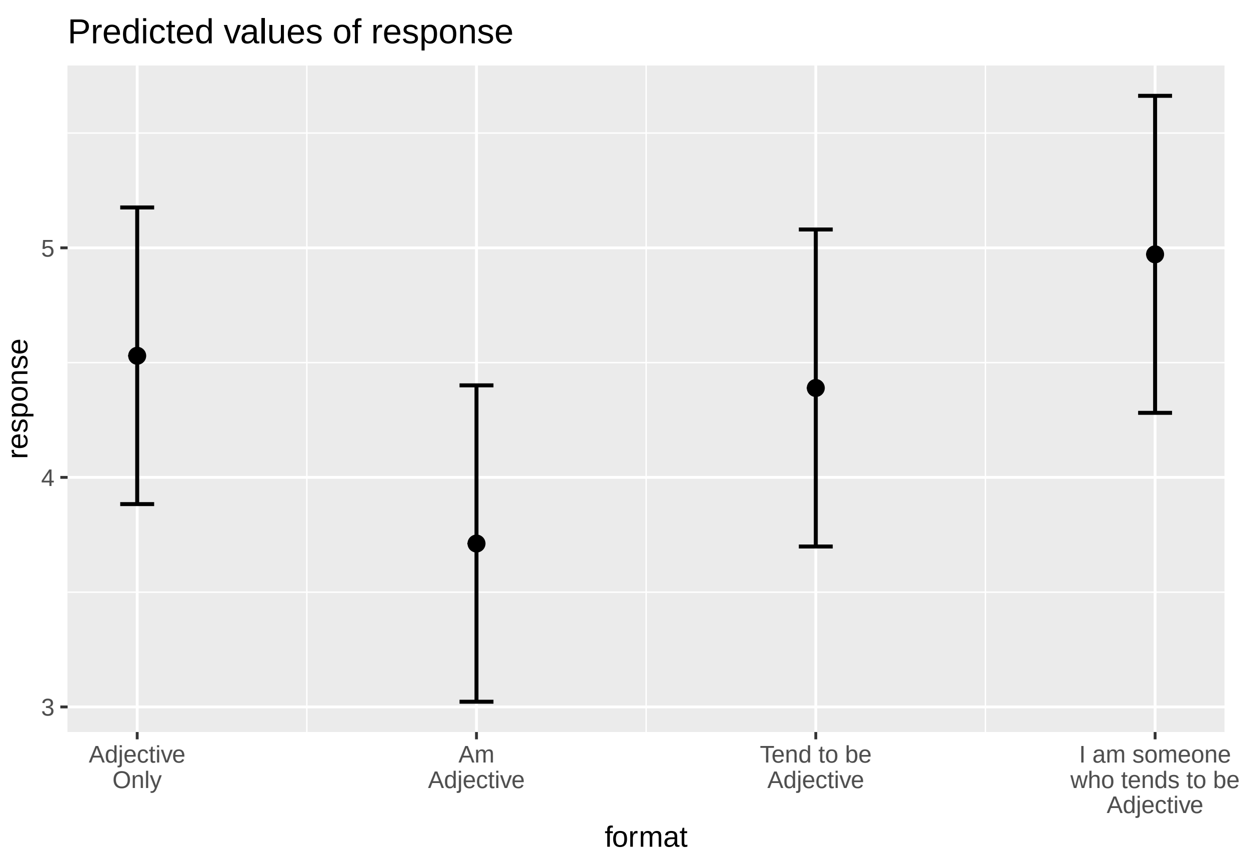

Figure 17: Average response to “careless” by format

Contrast	Difference in means	SE	df	t	p
Adjective Only - Am Adjective	-0.02	0.36	56.66	-0.04	1.000
Adjective Only - Tend to be Adjective	-0.92	0.36	55.47	-2.54	0.083
Adjective Only - I am someone who tends to be Adjective	-0.09	0.35	52.28	-0.26	1.000
Am Adjective - Tend to be Adjective	-0.90	0.37	56.24	-2.45	0.084
Am Adjective - I am someone who tends to be Adjective	-0.08	0.37	54.54	-0.21	1.000
Tend to be Adjective - I am someone who tends to be Adjective	0.83	0.33	43.22	2.49	0.084

Thrifty

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

thrifty_em = emmeans(thrifty_model, "format")
pairs(thrifty_em, adjust = "holm") %>%
  kable(booktabs = T,
        digits = c(0,2,2,2,2,3),
        col.names = c("Contrast", "Difference in means", "SE", "df", "t", "p")) %>%
  kable_styling()
```

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

Response by Format + Memory

```
mod.format_mem = lmer(response~format + memory + (1|proid),
                      data = item_responses)
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 4.1952  1.3984     3 2007.23  0.8227 0.4812
## memory 3.7526  3.7526     1   32.85  2.2078 0.1468
```

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

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: response ~ format + memory + (1 | proid)
##    Data: item_responses
##
## REML criterion at convergence: 7387.9
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
```

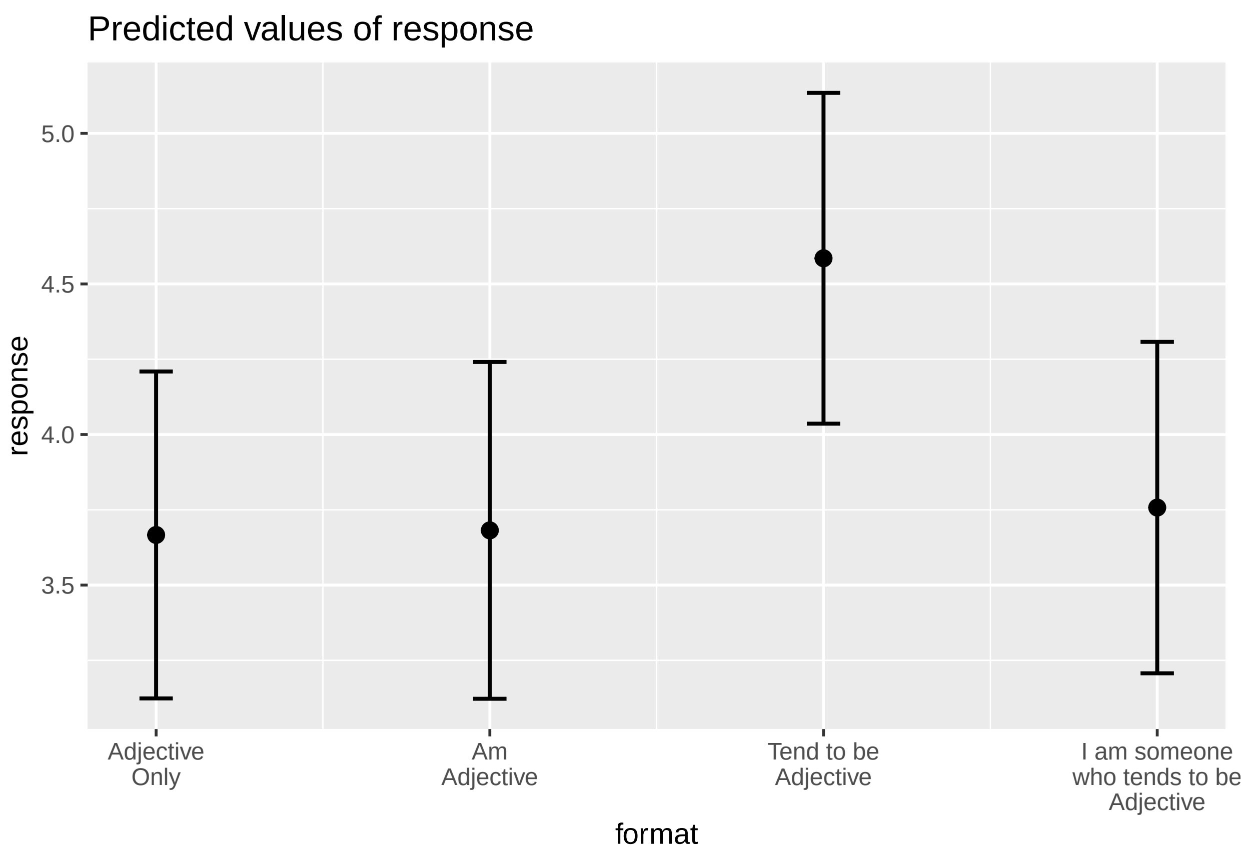

Figure 18: Average response to “thrifty” by format

```
## -3.3044 -0.4135  0.2692  0.7190  1.6316
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  proid    (Intercept) 0.1483   0.3851
##  Residual             1.6997   1.3037
## Number of obs: 2170, groups:  proid, 35
##
## Fixed effects:
##                                                     Estimate Std. Error         df
## (Intercept)                                          4.37695    0.17579   38.65739
## formatAm\nAdjective                                 -0.09504    0.08590 1999.45836
## formatTend to be\nAdjective                          0.03897    0.08886 1952.13858
## formatI am someone\nwho tends to be\nAdjective      -0.03779    0.08886 2023.87207
## memory                                               0.03912    0.02633   32.84897
##                                                   t value Pr(>|t|)
## (Intercept)                                        24.899   <2e-16 ***
## formatAm\nAdjective                                -1.106    0.269
## formatTend to be\nAdjective                         0.439    0.661
## formatI am someone\nwho tends to be\nAdjective     -0.425    0.671
## memory                                              1.486    0.147
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) frmtAA frTtbA faswttbA
## frmtAmAdjct -0.221
## frmtTndtbAd -0.241  0.466
## frmIaswttbA -0.231  0.458  0.474
## memory      -0.869 -0.013  0.011  0.005
```

```
plot_model(mod.format_mem, type = "pred", term = c("format"))
```

```
plot_model(mod.format_mem, type = "pred", term = c("memory"))
```

One model for each adjective

```
mod_by_item = item_responses %>%
  group_by(item) %>%
  nest() %>%
  mutate(mod = map(data, ~lm(response~format + memory, data = .))) %>%
  mutate(aov = map(mod, anova))

summary_by_item = mod_by_item %>%
  mutate(tidy = map(aov, broom::tidy)) %>%
  select(item, tidy) %>%
  unnest(cols = c(tidy)) %>%
  filter(term == "format") %>%
  select(-term, -df) %>%
  mutate(p.adj = p.adjust(p.value, method = "holm"))
```

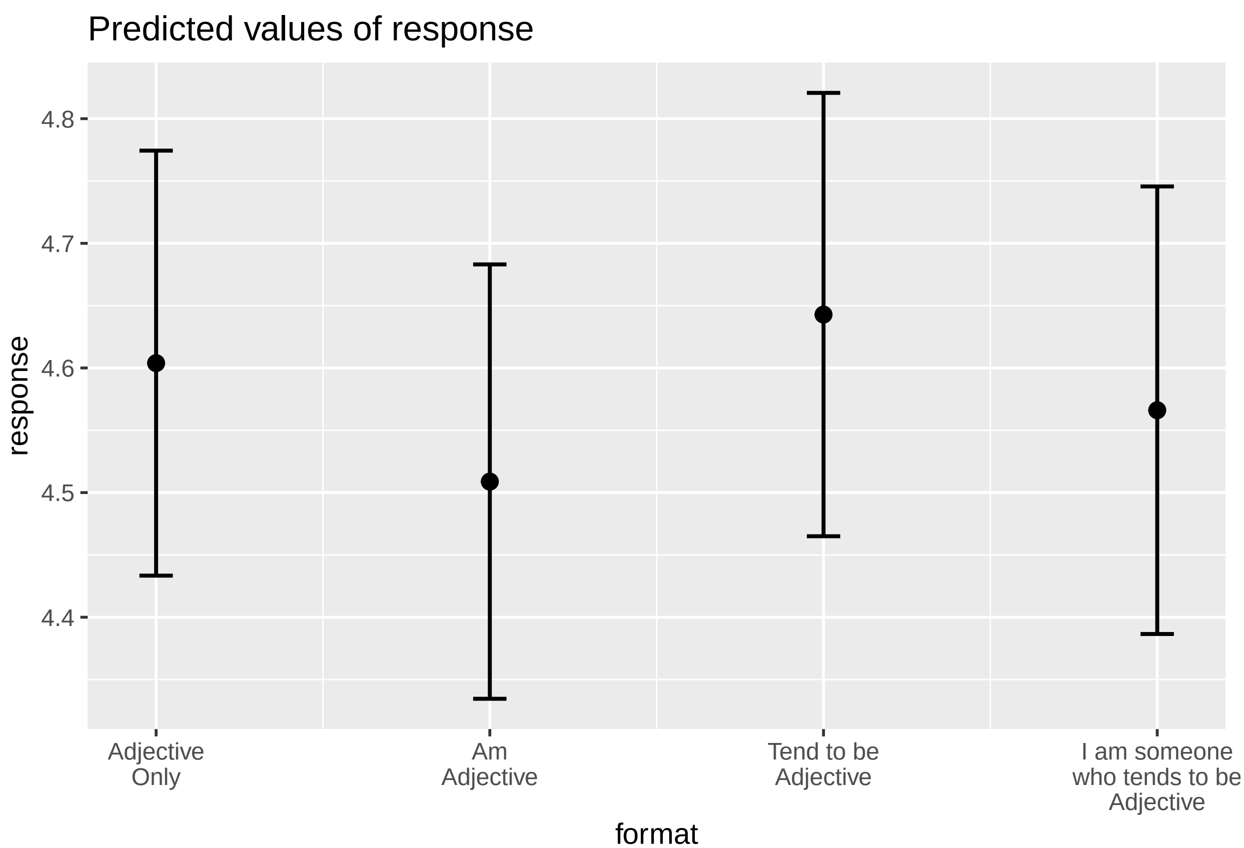

Figure 19: Predicted response on personality items by condition after controlling for memory.

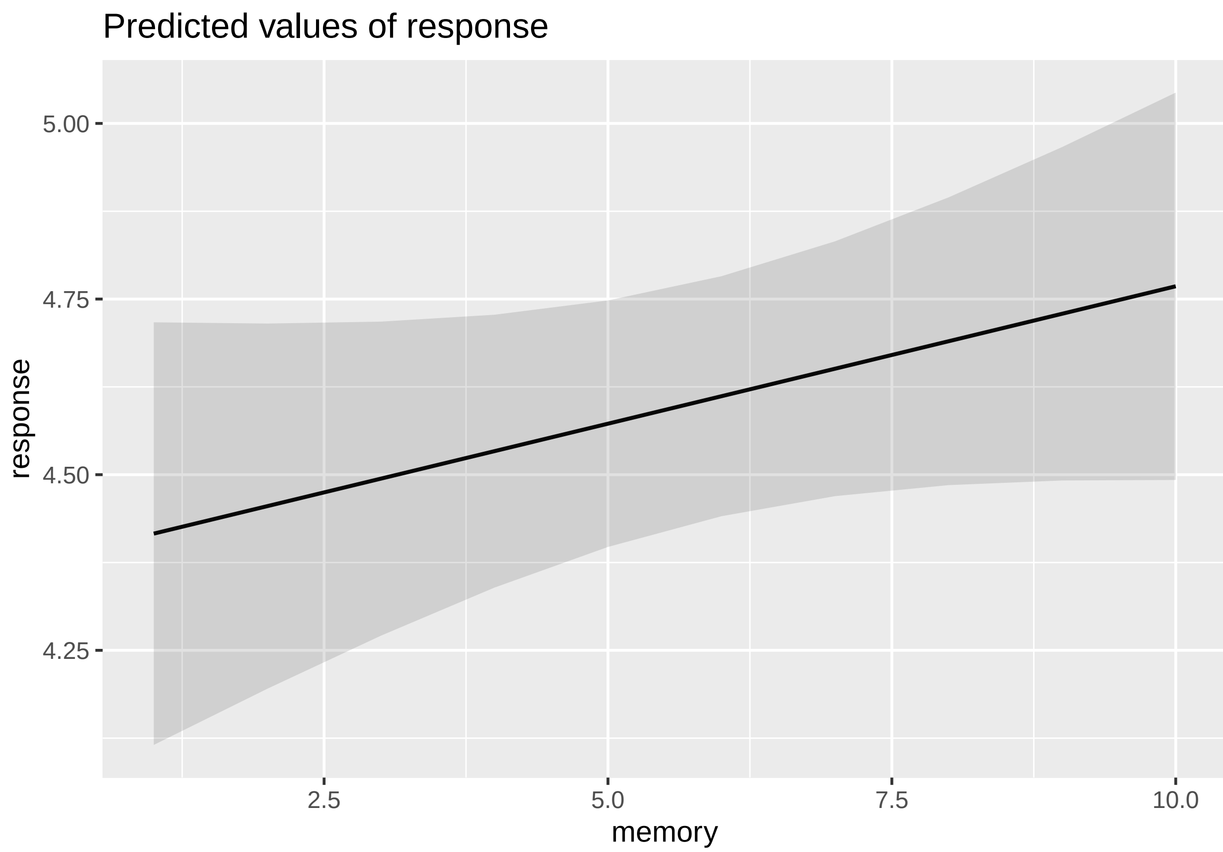

Figure 20: Predicted response on personality items by memory.

item	sumsq	meansq	statistic	p.value	p.adj
outgoing	5.02	1.67	1.20	0.32	0.32
helpful	0.52	0.17	0.30	0.83	0.83
reckless	7.28	2.43	0.96	0.42	0.42
moody	5.54	1.85	0.89	0.45	0.45
organized	1.88	0.63	0.54	0.66	0.66
friendly	0.98	0.33	0.55	0.65	0.65
warm	1.02	0.34	0.45	0.72	0.72
worrying	13.61	4.54	1.81	0.15	0.15
responsible	1.62	0.54	0.71	0.55	0.55
lively	2.77	0.92	0.67	0.57	0.57
asleep	0.23	0.08	0.03	0.99	0.99
caring	1.78	0.59	1.00	0.40	0.40
nervous	3.66	1.22	0.48	0.70	0.70
creative	0.07	0.02	0.03	0.99	0.99
hardworking	4.45	1.48	2.40	0.08	0.08
imaginative	2.85	0.95	1.56	0.21	0.21
softhearted	2.19	0.73	0.66	0.58	0.58
calm	2.38	0.79	0.99	0.40	0.40
intelligent	2.74	0.91	0.90	0.44	0.44
curious	2.52	0.84	0.56	0.64	0.64
active	0.80	0.27	0.17	0.91	0.91
human	1.46	0.49	0.84	0.48	0.48
careless	23.21	7.74	3.49	0.02	0.02
impulsive	0.65	0.22	0.10	0.96	0.96
sympathetic	3.52	1.17	1.30	0.28	0.28
cautious	1.73	0.58	0.51	0.67	0.67
talkative	10.49	3.50	1.49	0.23	0.23
sophisticated	2.33	0.78	0.48	0.70	0.70
adventurous	4.39	1.46	1.52	0.22	0.22
thorough	1.99	0.66	0.59	0.63	0.63
thrifty	9.59	3.20	2.17	0.10	0.10

```
summary_by_item %>%
  mutate(across(
    starts_with("p"),
    papaja::printnum
  )) %>%
  kable(digits = 2, booktabs = T) %>%
  kable_styling()
```

Pairwise t-tests for significant ANOVAs

Here we identify the specific items with significant differences.

```
sig_item = summary_by_item %>%
  filter(p.value < .05)
```

Contrast	Difference in means	SE	df	t	p
Adjective Only - Am Adjective	0.80	0.45	57.31	1.79	0.396
Adjective Only - Tend to be Adjective	0.11	0.44	53.22	0.25	0.800
Adjective Only - I am someone who tends to be Adjective	-0.47	0.44	53.00	-1.07	0.588
Am Adjective - Tend to be Adjective	-0.69	0.45	50.79	-1.53	0.524
Am Adjective - I am someone who tends to be Adjective	-1.27	0.45	50.66	-2.84	0.039
Tend to be Adjective - I am someone who tends to be Adjective	-0.59	0.45	49.06	-1.31	0.588

```
sig_item = sig_item$item
sig_item
```

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

Careless

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

careless_em = emmeans(careless_model, "format")
pairs(careless_em, adjust = "holm") %>%
  kable(booktabs = T,
        digits = c(0,2,2,2,2,3),
        col.names = c("Contrast", "Difference in means", "SE", "df", "t", "p")) %>%
  kable_styling()
```

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

Questions

- I only used responses in Block 1 here. Should I merge the two blocks together? Or I can repeat the analyses using Block 2? One thing to consider if we merge is that I can't use the simple ANOVA for the item-level analyses (we'll have two responses per person, so back to a nested design).

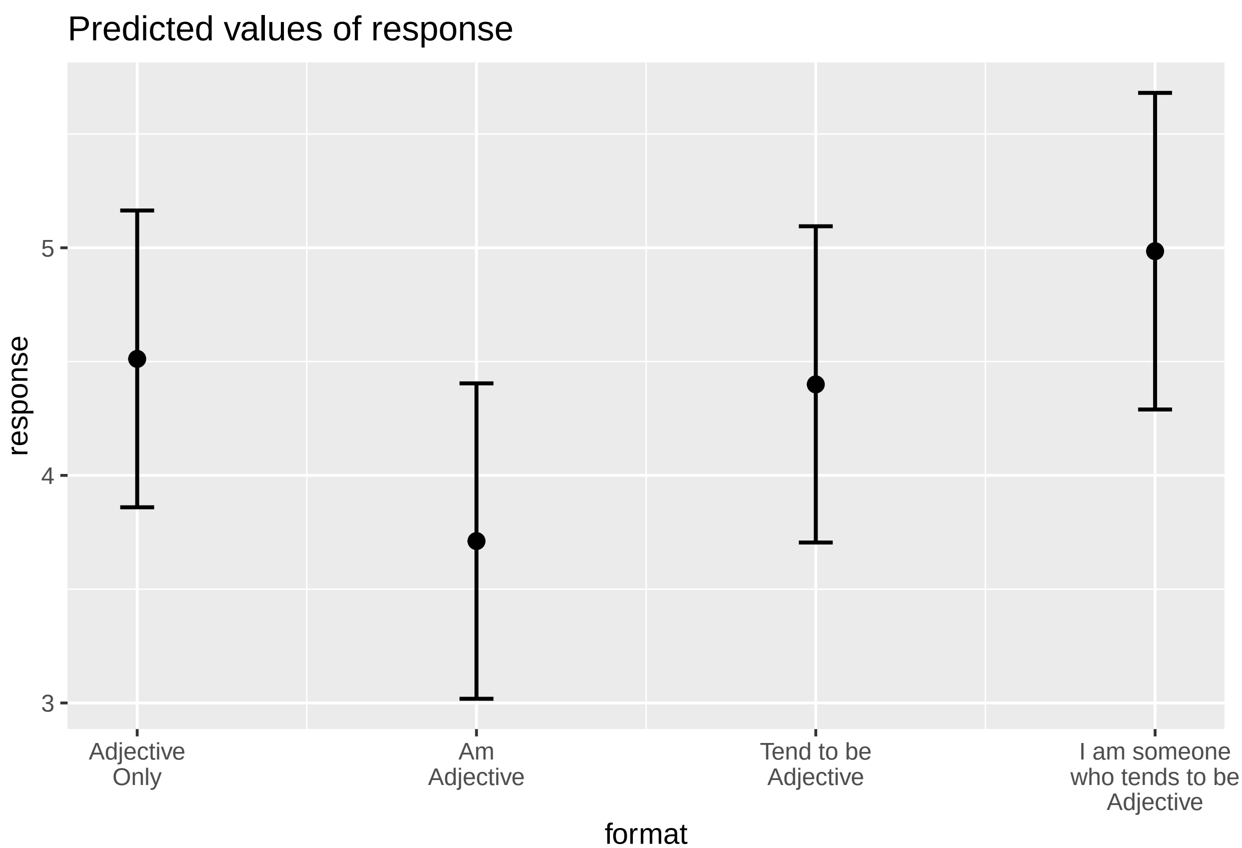

Figure 21: Average response to “careless” by format

How is variance in response attributable to participant, adjective, and format?

Within-person analyses will model the proportions of variance attributable to item format, stems of the items (i.e., the content of the adjectives), and the respondent-level variance.

Workspace

```
library(here) # for working with files
library(tidyverse) # for cleaning
library(janitor) # for variable names
library(lme4) # for mulitlevel modeling
library(sjPlot) # for figures
library(ggpubr) # for prettier plots
library(kableExtra) # for nicer tables
```

Data prep

First we select the responses to the items of different formats. These variable names all have the same format: `[trait]_[abcd]` (for example, `talkative_a`). We search for these items using regular expressions.

```
personality_items = str_subset(
  names(data),
  "^([[:alpha:]])+_[abcd]_?2?$"
)

item_responses = data %>%
  select(proid, all_of(personality_items), memory)
```

Next we reshape these data into long form.

```
item_responses = item_responses %>%
  gather(item, response, -proid, -memory) %>%
  mutate(
    time = ifelse(str_detect(item, "_2"), "2", "1"),
    item = str_remove(item, "_2")) %>%
  separate(item, into = c("item", "format")) %>%
  filter(!is.na(response))
```

Model

We estimate variance attributable to particpant (`proid`), adjective (`item`), and format (`format`) using a nested model.

```
mod_within_full = lmer(response ~ 1
                  + (1 | item)
                  + (1 | format)
                  + (1 |proid),
                  data = item_responses)
summary(mod_within_full)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: response ~ 1 + (1 | item) + (1 | format) + (1 | proid)
##    Data: item_responses
##
## REML criterion at convergence: 6724.8
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -4.1453 -0.5052  0.0946  0.6176  3.1789
##
## Random effects:
##  Groups   Name        Variance  Std.Dev.
##  proid    (Intercept) 0.1622567 0.40281
##  item     (Intercept) 0.5149790 0.71762
##  format   (Intercept) 0.0005179 0.02276
##  Residual             1.1924370 1.09199
## Number of obs: 2170, groups:  proid, 35; item, 31; format, 4
##
## Fixed effects:
##             Estimate Std. Error      df t value Pr(>|t|)
## (Intercept)   4.5802     0.1481 44.3232   30.93   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
variances = VarCorr(mod_within_full, comp="Variance")
var_proid  = variances$proid[[1]]
var_item   = variances$item[[1]]
var_format = variances$format[[1]]
var_resid = attr(variances, "sc")^2
var_total = var_proid + var_item + var_format + var_resid
```

Participants account for 8.68 percent of the variability in response.

Items account for 27.54 percent of the variability in response.

Format accounts for 0.03 percent of the variability in response.

In total, 36.24 percent of the variability in response is explained.

How does format affect timining of responses?

Using Block 1 Data only

We will further compare seconds times as a function of item format using both within- and between-person data in order to evaluate the presumption that adjective ratings require less time than phrased items.

Workspace

```
library(here) # for working with files
library(tidyverse) # for cleaning
library(janitor) # for variable names
library(lme4) # for mulitlevel modeling
library(lmerTest) # for p-values
library(emmeans) # for comparisons
library(sjPlot) # for figures
library(ggpubr) # for prettier plots
library(kableExtra) # for nicer tables
```

First, we select the timing to the items of different formats. For this set of analyses, we only use data collected in Block 1 – that is, each participant saw the same format for every item.

These variable names all have the same format: `t_[trait]_[abcd]_page_submit` (for example, `t_talkative_a_page_submit`). We search for these items using regular expressions.

```
timing_block1 = str_subset(
  names(data),
  "^t_([[:alpha:]])+_[abcd]_page_submit$"
)

timing_block1 = data %>%
  select(proid, all_of(timing_block1), memory)
```

Next we reshape these data into long form.

```
timing_block1 = timing_block1 %>%
  gather(item, seconds, -proid, -memory) %>%
  mutate(item = str_remove(item, "^t_"),
         item = str_remove(item, "_page_submit$")) %>%
  separate(item, into = c("item", "format")) %>%
  filter(!is.na(seconds))
```

Timing by Format

We used a multilevel model, nesting seconds within participant to account for dependence. Our primary predictor was format.

```
timing_block1$format = as.factor(timing_block1$format)
timing_block1$format = relevel(timing_block1$format, ref = "a")
timing_block1$format = factor(timing_block1$format,
```

```
                                levels = c("a","b","c","d"),
                                labels = c("Adjective\nOnly", "Am\nAdjective", "Tend to be\nAdjective",

mod.format = lmer(seconds~format + (1|proid),
                  data = timing_block1)
anova(mod.format)
```

```
## Type III Analysis of Variance Table with Satterthwaite's method
##        Sum Sq Mean Sq NumDF DenDF F value Pr(>F)
## format 1582.1  527.36     3    31  2.4217 0.0847 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

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

plot1$format +
  labs(x = NULL,
       y = "Seconds",
       title = "Average seconds time by item formatting") +
  theme_pubclean()
```

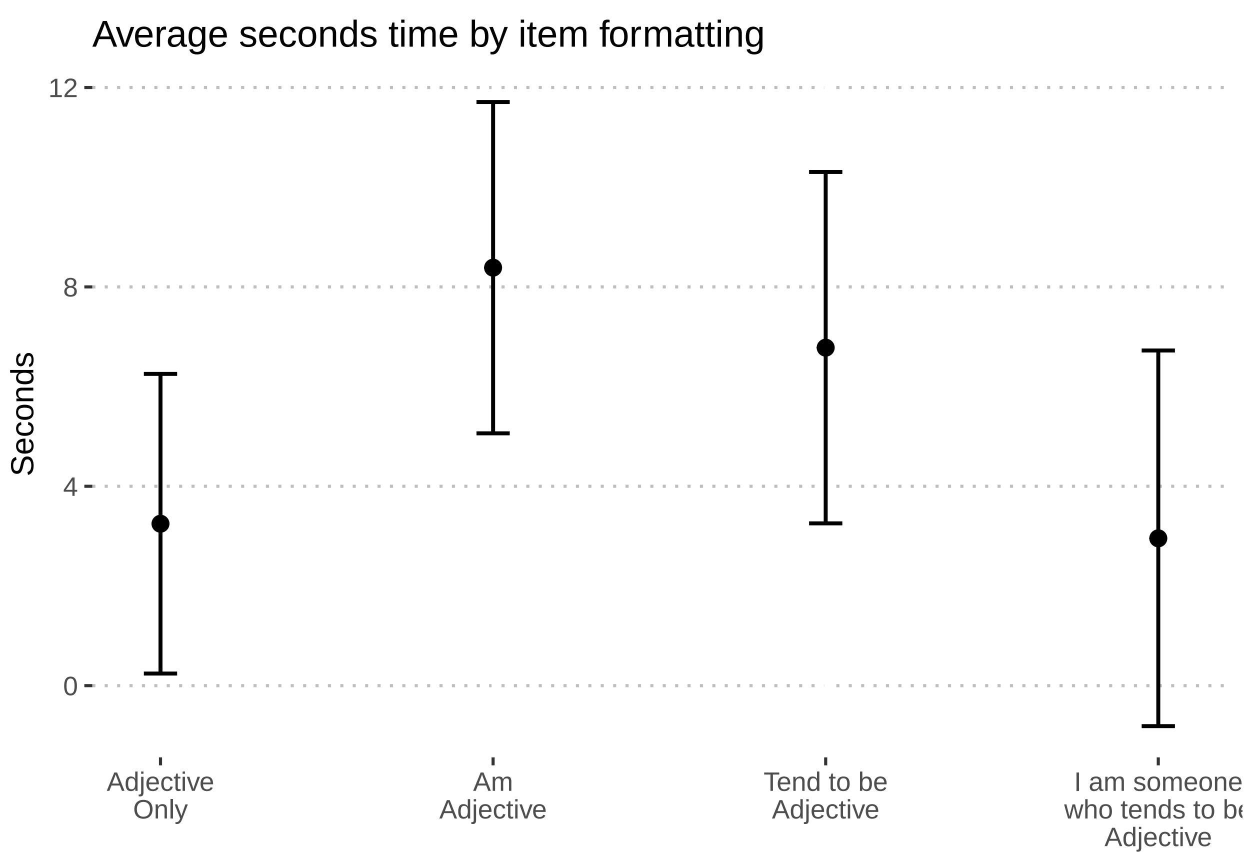

Figure 22: Predicted seconds on personality items by condition.

```
means_by_group = timing_block1 %>%
  group_by(format) %>%
  summarise(m = mean(seconds),
            s = sd(seconds))

timing_block1 %>%
  ggplot(aes(x = seconds, fill = format)) +
  geom_histogram(bins = 100) +
  geom_vline(aes(xintercept = m), data = means_by_group) +
  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, scales = "free_y") +
  guides(fill = "none") +
  scale_x_log10() +
  labs(y = "Number of particpants",
       title = "Distribution of seconds by format",
       x = "Seconds (logrithmic scale)") +
  theme_pubr()
```

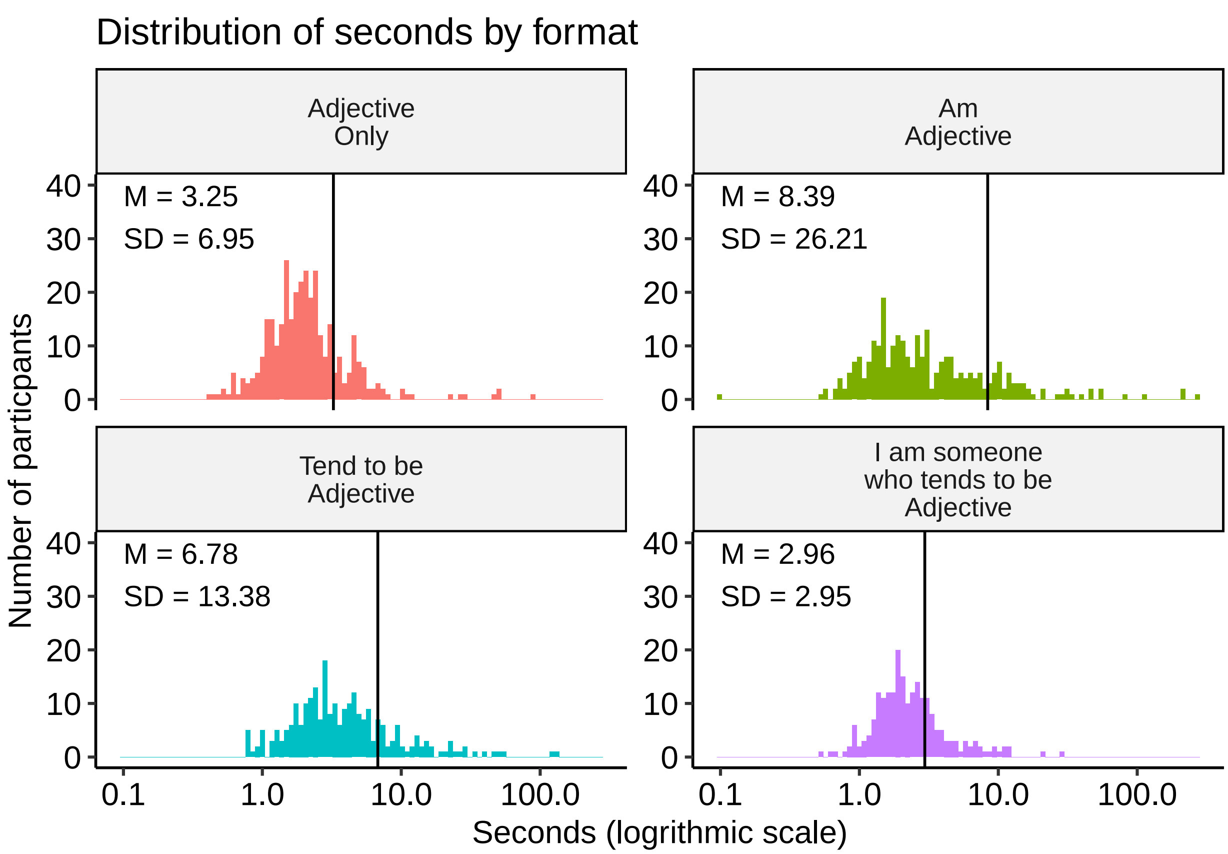

Figure 23: Distribution of secondss by category

Seconds by Format x Memory

```
mod.format_mem = lmer(seconds~format*memory + (1|proid),
                  data = timing_block1)
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          13.129   4.376     3    27  0.0201 0.9960
## memory         232.857 232.857     1    27  1.0693 0.3103
## format:memory  223.837  74.612     3    27  0.3426 0.7947
```

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

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: seconds ~ format * memory + (1 | proid)
##    Data: timing_block1
##
## REML criterion at convergence: 8943.9
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -1.4779 -0.1738 -0.0843 -0.0054 17.2126
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  proid    (Intercept)  20.61    4.54
##  Residual             217.76   14.76
## Number of obs: 1085, groups:  proid, 35
##
## Fixed effects:
##                                                         Estimate Std. Error
## (Intercept)                                              3.13042    3.92372
## formatAm\nAdjective                                      1.16198    5.94163
## formatTend to be\nAdjective                             -0.09808    5.70755
## formatI am someone\nwho tends to be\nAdjective          -0.14949    5.96873
## memory                                                   0.02004    0.60742
## formatAm\nAdjective:memory                               0.63783    0.89656
## formatTend to be\nAdjective:memory                       0.69397    0.93110
## formatI am someone\nwho tends to be\nAdjective:memory -0.02441    0.93144
##                                                             df t value Pr(>|t|)
## (Intercept)                                           27.00000   0.798    0.432
## formatAm\nAdjective                                   27.00000   0.196    0.846
## formatTend to be\nAdjective                           27.00000  -0.017    0.986
## formatI am someone\nwho tends to be\nAdjective        27.00000  -0.025    0.980
## memory                                                27.00000   0.033    0.974
## formatAm\nAdjective:memory                            27.00000   0.711    0.483
## formatTend to be\nAdjective:memory                    27.00000   0.745    0.463
## formatI am someone\nwho tends to be\nAdjective:memory 27.00000  -0.026    0.979
##
## Correlation of Fixed Effects:
```

```
##              (Intr) frmtAA frTtbA frIaswttbA memory frmAA: fTtbA:
## frmtAmAdjct -0.660
## frmtTndtbAd -0.687  0.454
## frmIaswttbA -0.657  0.434  0.452
## memory      -0.915  0.604  0.629  0.601
## frmtAAdjct:  0.620 -0.917 -0.426 -0.407     -0.677
## frmtTtbAdj:  0.597 -0.394 -0.902 -0.392     -0.652  0.442
## frIaswttbA:  0.597 -0.394 -0.410 -0.905     -0.652  0.442  0.425
```

```
plot_model(mod.format_mem, type = "pred", term = c("memory", "format"))
```

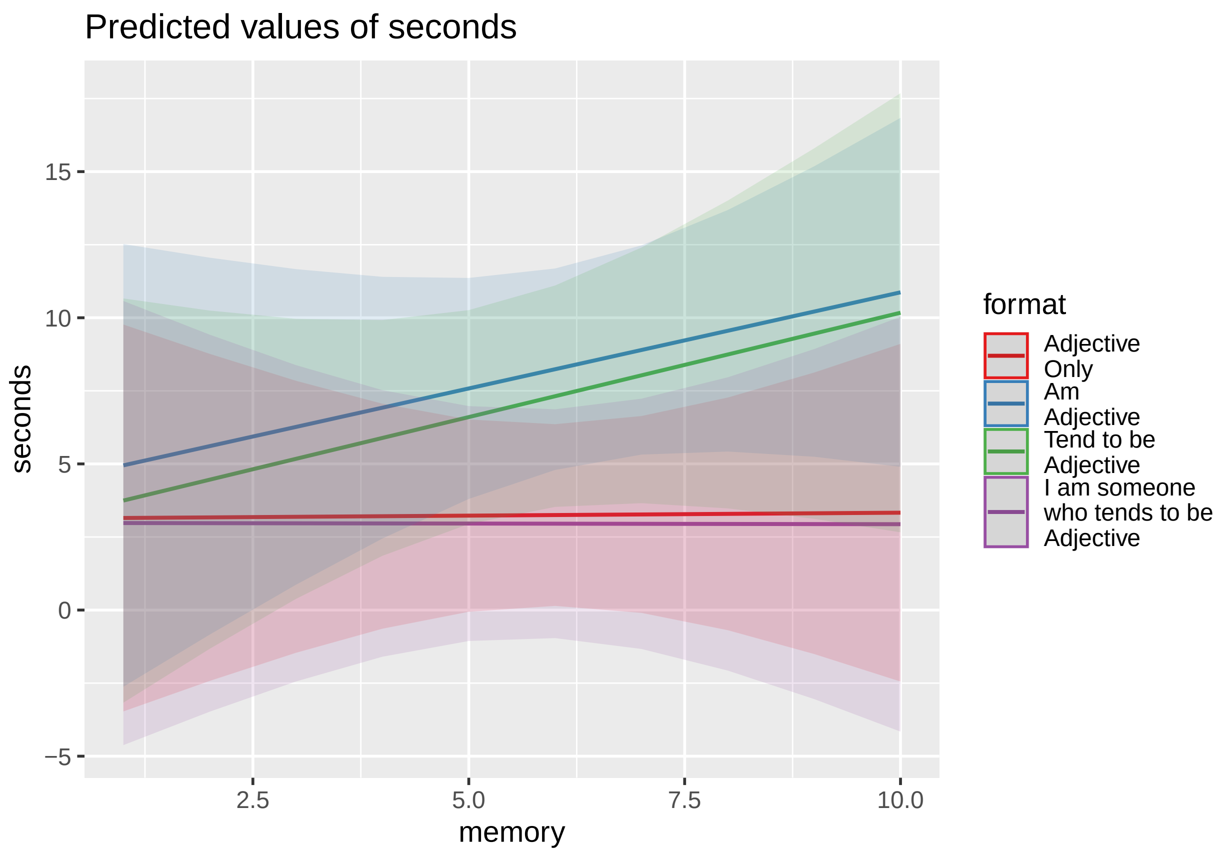

Figure 24: Predicted seconds on personality items by condition after controlling for memory.

Using Block 1 and Block 2

We will further compare seconds times as a function of item format using both within- and between-person data in order to evaluate the presumption that adjective ratings require less time than phrased items.

Workspace

```
library(here) # for working with files
library(tidyverse) # for cleaning
```

```r
library(janitor) # for variable names
library(lme4) # for mulitlevel modeling
library(lmerTest) # for p-values
library(emmeans) # for comparisons
library(sjPlot) # for figures
library(ggpubr) # for prettier plots
library(kableExtra) # for nicer tables
```

First, we select the timing to the items of different formats. For this set of analyses, we only use data collected in Blocks 1 and 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 two formats: `t_[trait]_[abcd]_page_submit` (for example, `t_talkative_a_page_submit`) or `t_[trait]_[abcd]_2_page_submit` (for example, `t_talkative_a_2_page_submit`). We search for these items using regular expressions.

```r
timing_block12 = str_subset(
  names(data),
  "^t_([[:alpha:]])+_[abcd]_(2_)?page_submit$"
)

timing_block12 = data %>%
  select(proid, all_of(timing_block12), memory)
```

Next we reshape these data into long form.

```r
timing_block12 = timing_block12 %>%
  gather(item, seconds, -proid, -memory) %>%
  mutate(item = str_remove(item, "^t_"),
         item = str_remove(item, "_2"),
         item = str_remove(item, "_page_submit$")) %>%
  separate(item, into = c("item", "format")) %>%
  filter(!is.na(seconds))
```

Timing by Format

We used a multilevel model, nesting seconds within participant to account for dependence. Our primary predictor was format.

```r
timing_block12$format = as.factor(timing_block12$format)
timing_block12$format = relevel(timing_block12$format, ref = "a")
timing_block12$format = factor(timing_block12$format,
                               levels = c("a","b","c","d"),
                               labels = c("Adjective\nOnly", "Am\nAdjective", "Tend to be\nAdjective",

mod.format = lmer(seconds~format + (1|proid),
                  data = timing_block12)
anova(mod.format)
```

```
## Type III Analysis of Variance Table with Satterthwaite's method
##        Sum Sq Mean Sq NumDF  DenDF F value Pr(>F)
## format 2076.1  692.02     3 1560.1  1.9518 0.1193
```

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

plot1$format +
  labs(x = NULL,
       y = "Seconds",
       title = "Average seconds time by item formatting") +
  theme_pubclean()
```

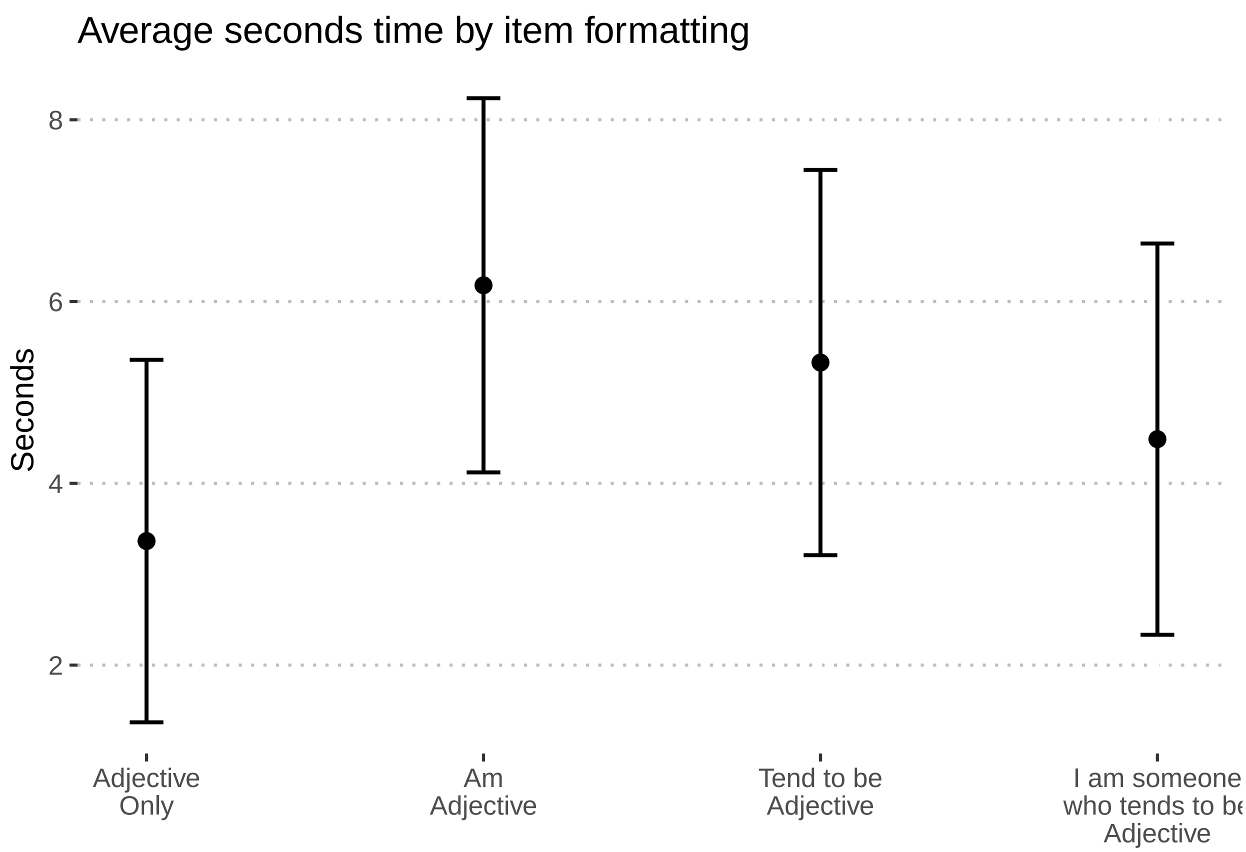

Figure 25: Predicted seconds on personality items by condition.

```r
means_by_group = timing_block12 %>%
  group_by(format) %>%
  summarise(m = mean(seconds),
            s = sd(seconds))

timing_block12 %>%
  ggplot(aes(x = seconds, fill = format)) +
  geom_histogram(bins = 100) +
  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, scales = "free_y") +
  guides(fill = "none") +
  scale_x_log10() +
  labs(y = "Number of particpants",
       title = "Distribution of seconds by format",
       x = "Seconds (logrithmic scale)") +
  theme_pubr()
```

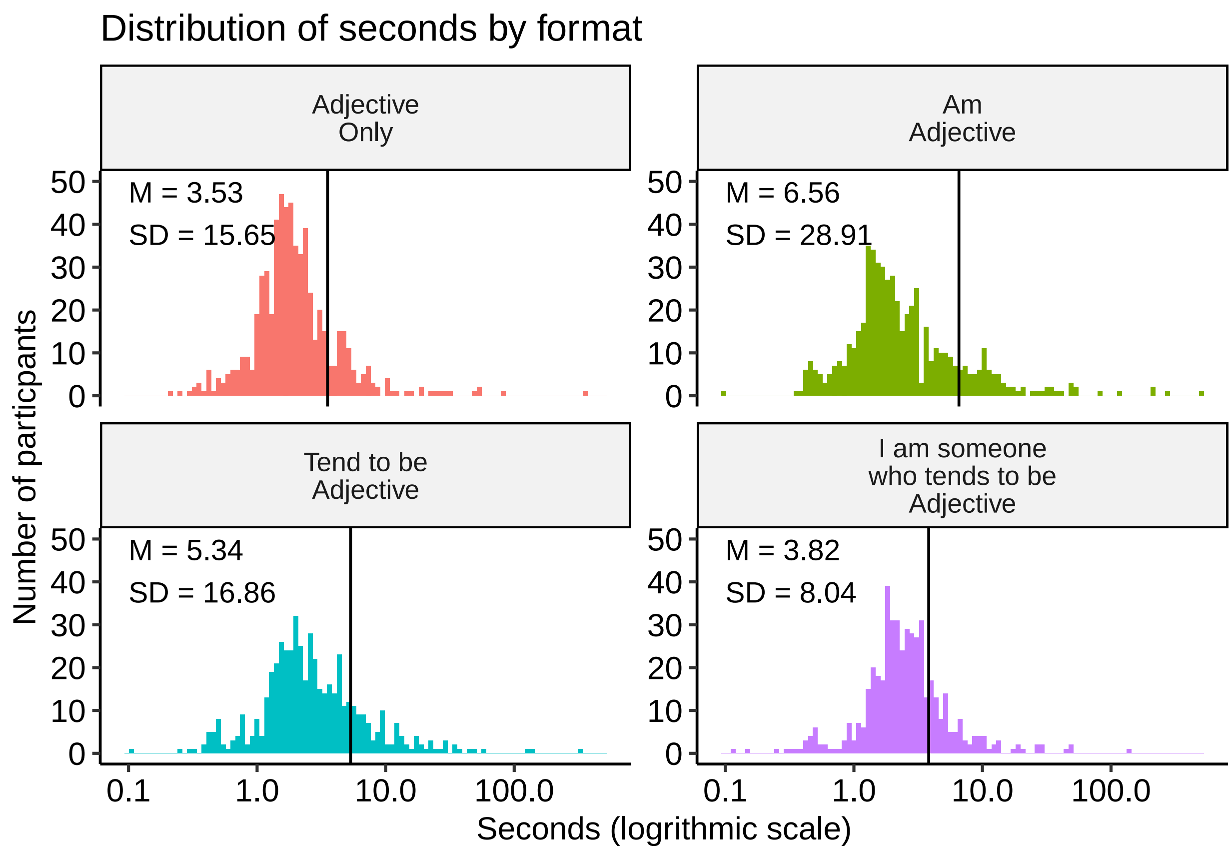

Figure 26: Distribution of secondss by category

Seconds by Format x Memory

```
mod.format_mem = lmer(seconds~format*memory + (1|proid),
                      data = timing_block12)
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         366.37  122.12     3 1633.89  0.3442 0.7934
## memory         453.69  453.69     1   32.53  1.2786 0.2664
## format:memory 507.29  169.10     3 1628.00  0.4765 0.6986
```

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

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: seconds ~ format * memory + (1 | proid)
##    Data: timing_block12
##
## REML criterion at convergence: 18925.6
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -0.8463 -0.1500 -0.0679 -0.0025 26.8076
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  proid    (Intercept)  12.58    3.548
##  Residual             354.84   18.837
## Number of obs: 2170, groups:  proid, 35
##
## Fixed effects:
##                                                         Estimate Std. Error
## (Intercept)                                               1.9198     2.4424
## formatAm\nAdjective                                       1.9008     2.9089
## formatTend to be\nAdjective                               0.2748     2.9430
## formatI am someone\nwho tends to be\nAdjective            2.4565     2.9280
## memory                                                    0.2482     0.3849
## formatAm\nAdjective:memory                                0.1501     0.4522
## formatTend to be\nAdjective:memory                        0.3036     0.4691
## formatI am someone\nwho tends to be\nAdjective:memory    -0.2304     0.4599
##                                                               df t value
## (Intercept)                                             111.6272   0.786
## formatAm\nAdjective                                    1788.8184   0.653
## formatTend to be\nAdjective                            1477.2554   0.093
## formatI am someone\nwho tends to be\nAdjective         1687.5420   0.839
## memory                                                  113.0304   0.645
## formatAm\nAdjective:memory                             1703.6493   0.332
## formatTend to be\nAdjective:memory                     1436.8229   0.647
## formatI am someone\nwho tends to be\nAdjective:memory 1692.8062  -0.501
##                                                       Pr(>|t|)
## (Intercept)                                              0.434
## formatAm\nAdjective                                      0.514
## formatTend to be\nAdjective                              0.926
## formatI am someone\nwho tends to be\nAdjective           0.402
## memory                                                   0.520
## formatAm\nAdjective:memory                               0.740
## formatTend to be\nAdjective:memory                       0.518
## formatI am someone\nwho tends to be\nAdjective:memory    0.617
##
## Correlation of Fixed Effects:
##             (Intr) frmtAA frTtbA frIaswttbA memory frmAA: fTtbA:
## frmtAmAdjct -0.564
## frmtTndtbAd -0.583  0.473
## frmIaswttbA -0.573  0.463  0.494
```

```
## memory      -0.910  0.517  0.533  0.523
## frmtAAdjct:  0.524 -0.909 -0.439 -0.431     -0.581
## frmtTtbAdj:  0.527 -0.428 -0.905 -0.447     -0.581  0.482
## frIaswttbA:  0.525 -0.427 -0.453 -0.904     -0.578  0.483  0.495
```

```
plot_model(mod.format_mem, type = "pred", term = c("memory", "format"))
```

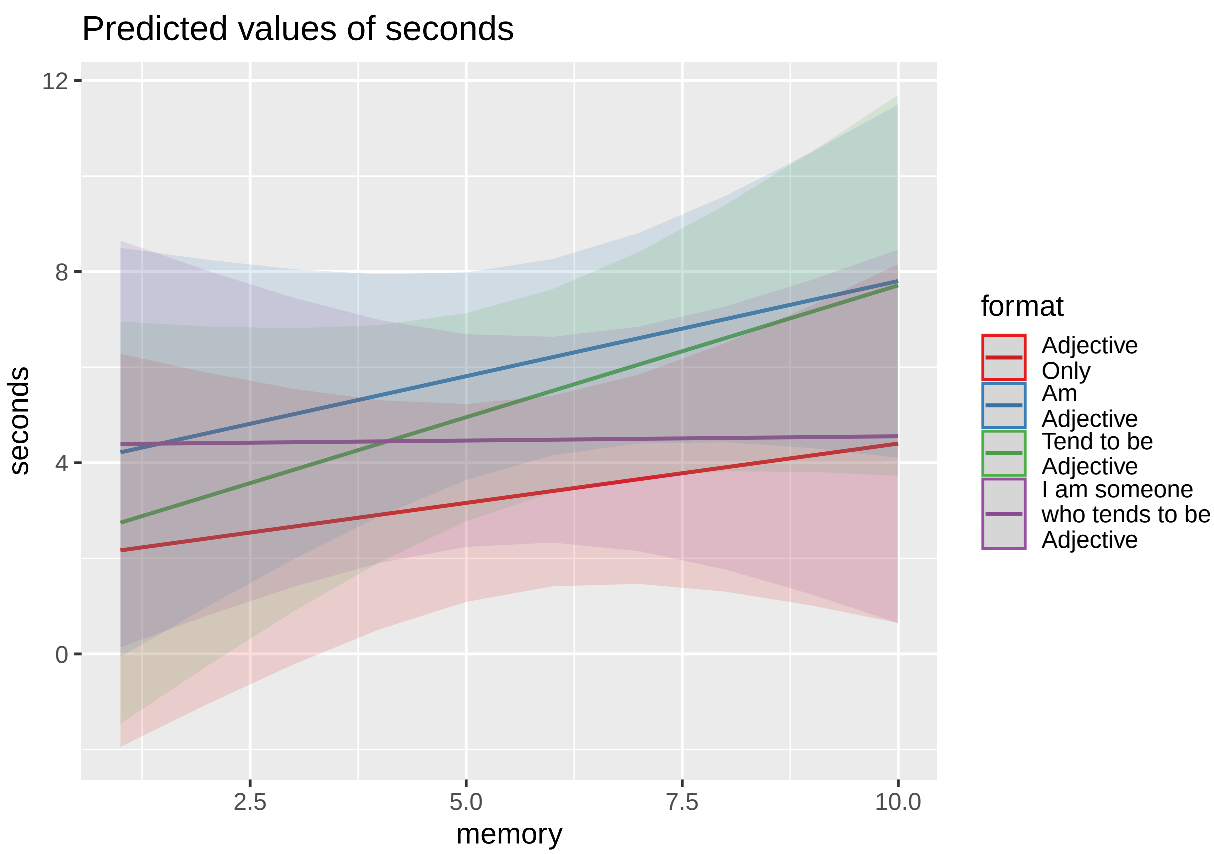

Figure 27: Predicted seconds on personality items by condition after controlling for memory.

How does device type affect means and timining of responses?

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 memory effects by collecting data on immediate and delayed recall (5 minutes and approximately two weeks) using a memory paradigm that was developed based on a similar recall task used in the HRS (Runge et al., 2015).

Workspace

```
library(here) # for working with files
library(tidyverse) # for cleaning
library(janitor) # for variable names
library(lme4) # for mulitlevel modeling
library(lmerTest) # for df and p-values
library(emmeans) # for pairwise comparisons
library(sjPlot) # for figures
library(ggpubr) # for prettier plots
library(kableExtra) # for nicer tables
```

Data prep

We will use between-person analyses to compare responses using group-level data for the different formats.

First we 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 two formats: `[trait]_[abcd]` (for example, `talkative_a`) or `[trait]_[abcd]_2` (for example, `talkative_a_2`). We search for these items using regular expressions.

```
items_seen_b1b2 = str_subset(
  names(data),
  "^([[:alpha:]])+_[abcd](_2)?$"
)

item_responses = data %>%
  select(proid, all_of(items_seen_b1b2), devicetype)
```

Next we reshape these data into long form.

```r
item_responses = item_responses %>%
  gather(item, response, -proid, -devicetype) %>%
  mutate(
    block = case_when(              # which block is the item from?
      str_detect(item, "_2") ~ "2",
      TRUE ~ "1"),
    item = str_remove(item, "_2")) %>%  # remove block id from item string
  separate(item, into = c("item", "format")) %>%
  filter(!is.na(response))
```

We also prepare a dataframe to examine the amount of time needed to respond to items. These variable names have one of two formats: `t_[trait]_[abcd]_page_submit` (for example, `t_talkative_a_page_submit`) or `t_[trait]_[abcd]_2_page_submit` (for example, `t_talkative_a_2_page_submit`). We search for these items using regular expressions.

```r
timing_block12 = str_subset(
  names(data),
  "^t_([[:alpha:]])+_[abcd]_(2_)?page_submit$"
)

timing_block12 = data %>%
  select(proid, all_of(timing_block12), devicetype)
```

Next we reshape these data into long form.

```r
timing_block12 = timing_block12 %>%
  gather(item, seconds, -proid, -devicetype) %>%
  mutate(item = str_remove(item, "^t_"),
         item = str_remove(item, "_2"),
         item = str_remove(item, "_page_submit$")) %>%
  separate(item, into = c("item", "format")) %>%
  filter(!is.na(seconds))
```

Response by Device

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

```r
mod.format = lmer(response~devicetype + (1|proid),
                  data = item_responses)
anova(mod.format)
```

```
## Type III Analysis of Variance Table with Satterthwaite's method
##            Sum Sq Mean Sq NumDF DenDF F value Pr(>F)
## devicetype  4.008   2.004     2    32  1.1793 0.3205
```

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

plot1$devicetype +
  labs(x = NULL,
       y = "Average response",
```

```
      title = "Average responses by device") +
  theme_pubclean()
```

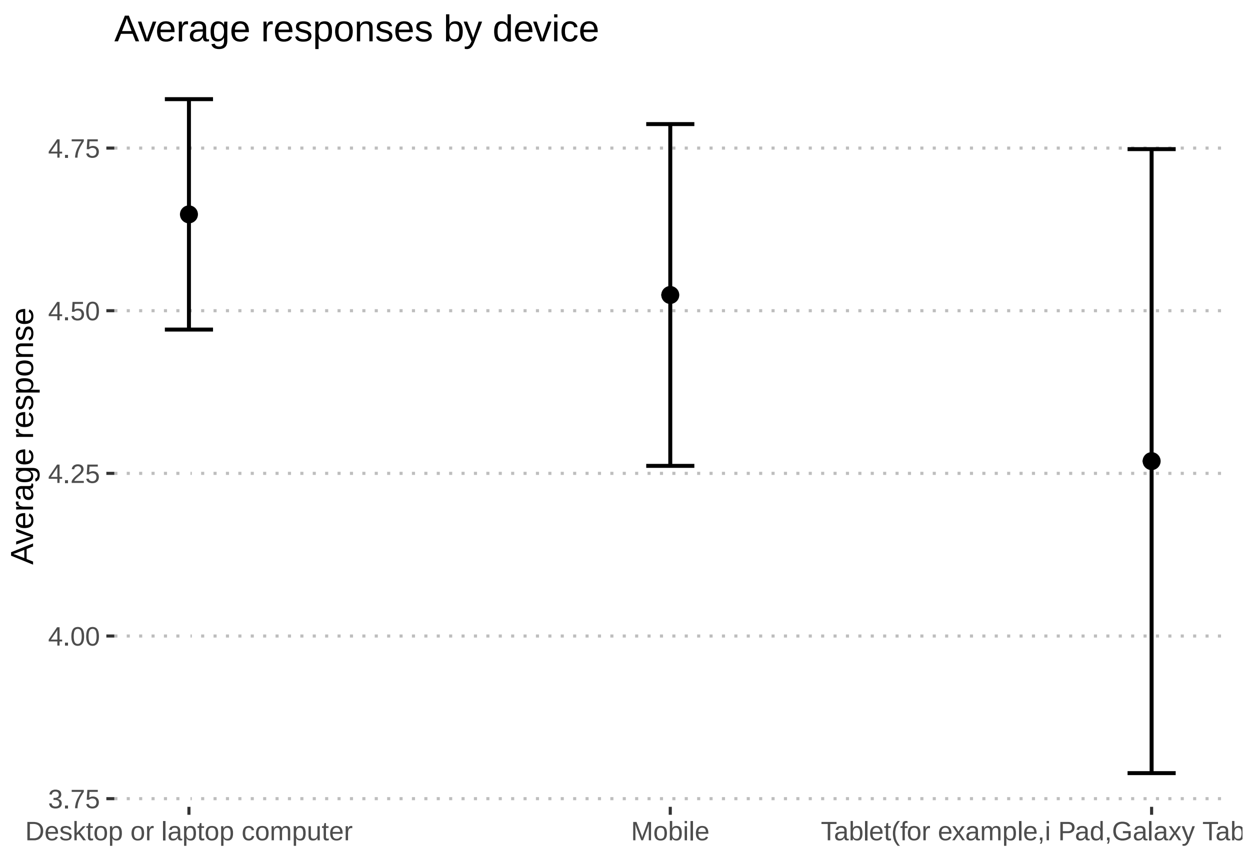

Figure 28: Predicted response on personality items by condition.

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

item_responses %>%
  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()
```

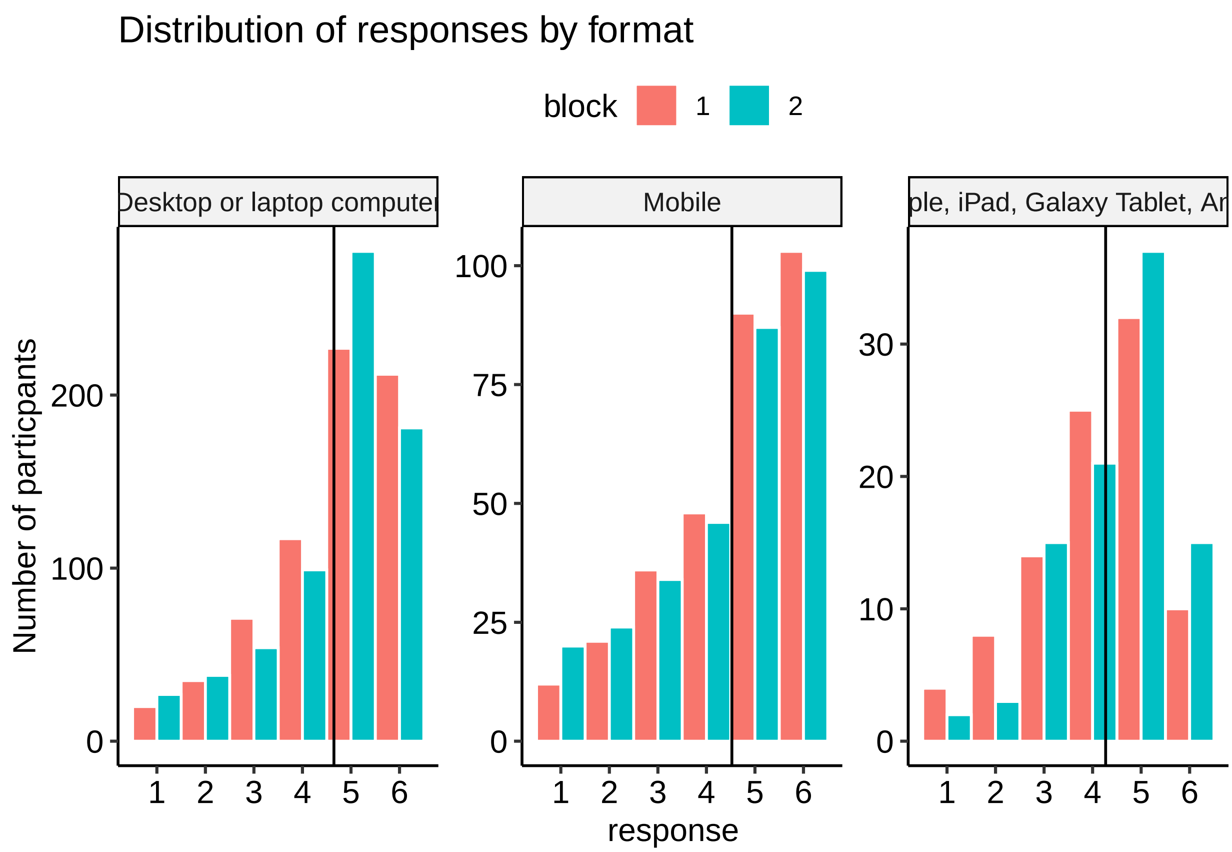

Figure 29: Distribution of responses by category

Timing by device

We used a multilevel model, nesting seconds within participant to account for dependence. Our primary predictor was format.

```
mod.format = lmer(seconds~devicetype + (1|proid),
                  data = timing_block12)
anova(mod.format)
```

```
## Type III Analysis of Variance Table with Satterthwaite's method
##            Sum Sq Mean Sq NumDF DenDF F value Pr(>F)
## devicetype 286.48  143.24     2    32  0.4036 0.6712
```

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

plot1$devicetype +
  labs(x = NULL,
       y = "Seconds",
       title = "Average seconds time by device type") +
  theme_pubclean()
```

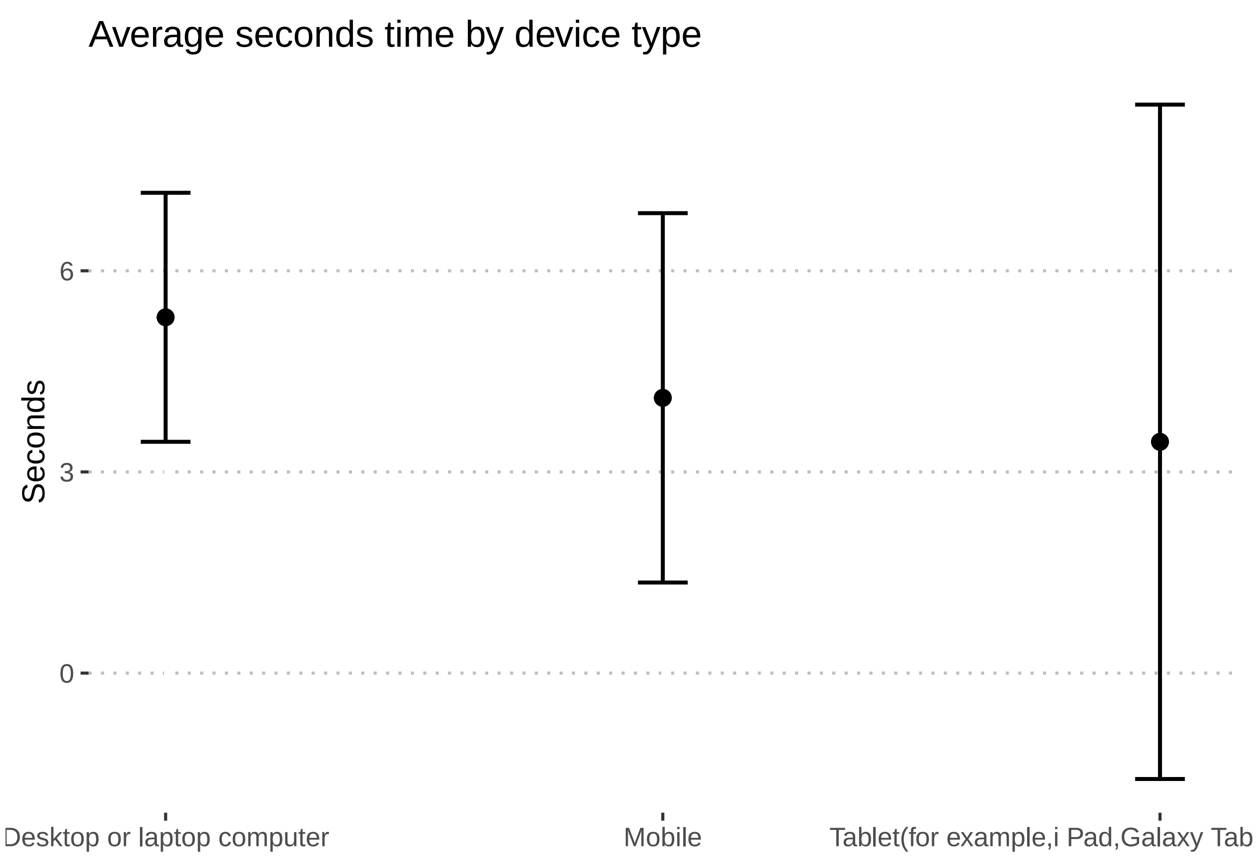

Figure 30: Predicted seconds on personality items by condition.

```
means_by_group = timing_block12 %>%
  group_by(devicetype) %>%
  summarise(m = mean(seconds),
            s = sd(seconds))

timing_block12 %>%
  ggplot(aes(x = seconds, 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() +
  labs(y = "Number of particpants",
       title = "Distribution of seconds by format",
       x = "Seconds (logrithmic scale)") +
  theme_pubr()
```

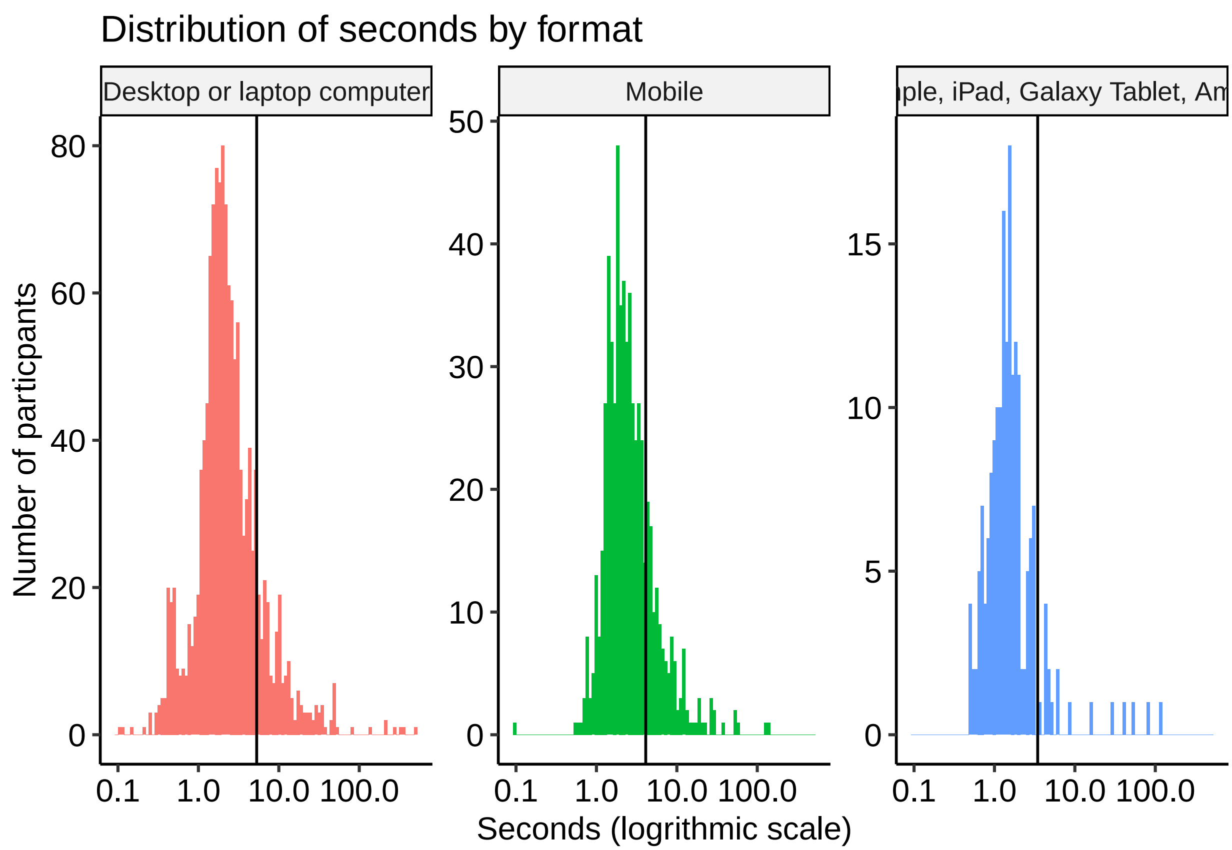

Figure 31: Distribution of secondss by category