Sensitivity Trial Level Analysis

Erin M. Buchanan

2022-10-19

```
# Seed for random number generation
set.seed(42)
knitr::opts_chunk$set(cache.extra = knitr::rand_seed)
```

Load Data

```
# deal with too many trials
number_trials_start <- SP_V %>%
  group_by(Subject, Match, Target) %>%
  summarize(n_trials = n())
```

```
## ‘summarise()‘ has grouped output by ’Subject’, ’Match’. You can override using
## the ‘.groups‘ argument.
```

```
# site_CAN_020_1 needs fixing
SP_V$Subject[SP_V$Subject == "site_CAN_020_1"] <-
  c(rep("site_CAN_020_1", 24), rep("site_CAN_020_1_2", 24))

# several have too many
SP_V <- SP_V %>%
  group_by(Subject, Match, Target) %>%
  filter(!duplicated(Subject))
```

```
# We will implement a minimum response latency 160
# We will use a 2*MAD criterion to eliminate long response latencies
# SP_V_tidy and PP_tidy has the variable "Outlier" denoted the outlier.
SP_V_tidy <- SP_V_tidy %>%
  group_by(Subject) %>%
  mutate(MAD = mad(response_time),
         med = median(response_time),
         Outlier = response_time <= 160 | response_time >= (med + 2*MAD))

# Integrate this into the outlier analysis table, change out for lmer criterion and say why
```

Look at the number of trials

```
number_trials <-
  SP_V_lme_data %>%
  group_by(Subject, Match) %>%
  summarize(count = n())
```

```
## ‘summarise()‘ has grouped output by ’Subject’. You can override using the
## ‘.groups‘ argument.
```

Run Models

```r
models <- list()

for (i in 4:8){

  subjects <- number_trials %>%
    filter(count >= i) %>%
    pull(Subject) %>% unique()

  temp_data <- SP_V_lme_data %>%
    filter(Subject %in% subjects)

  #only intercept
  models[[paste("intercept.model", i, sep = "_")]] <- lm(response_time ~ 1,
                        data = temp_data)
  #add random intercept of subject
  models[[paste("subject.model", i, sep = "_")]] <- lmer(response_time ~ 1 + (1|Subject),
                        control = lmerControl(optimizer = "bobyqa",
                                              optCtrl = list(maxfun = 1e6)),
                        data = temp_data)
  # add random intercept of item
  models[[paste("item.model", i, sep = "_")]] <- lmer(response_time ~ 1 + (1|Subject) + (1|Target),
                        control = lmerControl(optimizer = "bobyqa",
                                              optCtrl = list(maxfun = 1e6)),
                        data = temp_data)
  # add random intercept of lab
  models[[paste("lab.model", i, sep = "_")]] <- lmer(response_time ~ 1 + (1|Subject) + (1|Target) + (1|
                        control = lmerControl(optimizer = "bobyqa",
                                              optCtrl = list(maxfun = 1e6)),
                        data = temp_data)

  # add random intercept of language
  models[[paste("language.model", i, sep = "_")]] <- lmer(response_time ~ 1 + (1|Subject) + (1|Target)
                        control = lmerControl(optimizer = "bobyqa",
                                              optCtrl = list(maxfun = 1e6)),
                        data = temp_data)

  # add fixed effect of match
  models[[paste("fixed.four.model", i, sep = "_")]] <- lmer(response_time ~ Match + (1|Subject) + (1|Ta
                        control = lmerControl(optimizer = "bobyqa",
                                              optCtrl = list(maxfun = 1e6)),
                        data = temp_data)

}
```

View the Results

```r
AIC_values <- as.data.frame(unlist(lapply(models, AIC))) %>%
  rename("AIC" = "unlist(lapply(models, AIC))")
AIC_values$model <- rownames(AIC_values)

AIC_values <- tidyr::separate(AIC_values,
                model,
                into = c("model", "number_trials"),
                sep = "_") %>%
  pivot_wider(data = .,
              id_cols = c(number_trials),
              values_from = AIC,
              names_from = model)

AIC_values
```

```
## # A tibble: 5 x 7
##   number_trials intercept.model subject.model item.model lab.m~1 langu~2 fixed~3
##   <chr>                   <dbl>         <dbl>      <dbl>   <dbl>   <dbl>   <dbl>
## 1 4                     974894.       929623.    927110. 926836. 926834. 926833.
## 2 5                     974894.       929623.    927110. 926836. 926834. 926833.
## 3 6                     974894.       929623.    927110. 926836. 926834. 926833.
## 4 7                     974894.       929623.    927110. 926836. 926834. 926833.
## 5 8                     974211.       929095.    926583. 926312. 926309. 926308.
## # ... with abbreviated variable names 1: lab.model, 2: language.model,
## #   3: fixed.four.model
```

```
fixef(models$fixed.four.model_4)
```

```
##      (Intercept) MatchMISMATCHING
##      666.2821665        0.7457011
```

```
fixef(models$fixed.four.model_5)
```

```
##      (Intercept) MatchMISMATCHING
##      666.2821665        0.7457011
```

```
fixef(models$fixed.four.model_6)
```

```
##      (Intercept) MatchMISMATCHING
##      666.2821665        0.7457011
```

```
fixef(models$fixed.four.model_7)
```

```
##      (Intercept) MatchMISMATCHING
##      666.2821665        0.7457011
```

```
fixef(models$fixed.four.model_8)
```

```
##      (Intercept) MatchMISMATCHING
##      667.1132665        0.7398123
```