Appendix: Power Analysis

2022-05-11

Design

For this power analysis we will simulate 30 labs contributing 32 infants (960 participants) from 3 to 15 months of age. (This is conservative, as MB5 estimates there will be at least 1200 participants in the final sample.)

Notes: MB1 overall effect size was 0.29 for the single-screen central fixation (CF) method, with additional effect of 0.21 for HPP, and eye-tracking (ET) yielding a slight (non-significant) decrease in effect of -0.06. We expect to have X labs running infant-controlled familiarization duration, and the other 20-X labs running a fixed familiarization procedure.

Factors:

- *stimulus_type*: indicates the type (complexity/difficulty) of the stimuli that infants are familiarized with during training (high/low stimulus type; within-infant, 12 per type)
- *familiarization_time*: indicates how long each stimulus is exposed during familiarization (5, 10, or 15 seconds; within-infant)
- *trial_num*: indicates the sequential order in which test trials were presented. Trial number thus ranges from 1 to 24.
- *age_mos*: the infants' age in months (3.0-15.0), centered in *age* column.
- *procedure*: indicates the experimental method that was used to record infants' looking to the stimuli: infant-controlled exposure (IC; total familiarization time is achieved over uncontrolled period of time) vs. fixed-duration exposure (FD; controlled period of exposure, unknown period of infant fixation)
- *test_order*: indicates which of the four pseudorandom test orders (from our provided scripts) were used to present test trials to the infant. true for MB5???

To do our power analysis, we will generate 1,000 datasets of this structure with a given effect size (e.g., .3), run the mixed-effects regression for each simulated dataset, and count the number of times that the effect is significant. Note that we generate normally-distributed looking times, assuming that they have already been log-transformed.

Simulate Datasets

```
# To use the rbeta function, we need to supply two shape functions that correspond to the mean and the
# variance of the data. To get those shape parameters, we will use the get.ab function below from
# Mijke Rhemtulla/ Minhajuddin et al. (2004)
get.ab <- function(mu, var){
  v <- var
  w <- mu/(1-mu)
```

```r
  b <- ((w/ (v*(w^2 + 2*w + 1))) - 1)  / (w + 1)
  a <- b*w
  return(list(a = a, b = b))
}

set.seed(123) # reproducible sampling

generate_dataset <- function(n_labs=30, n_per_lab=32,
                             effect_sizes=list(type = .3, familiarization = .1, age = .1, "age*type"=.1
                                               "age*familiarization"=0, "type*familiarization"=0,
                                               "type*age*familiarization"=.1)) {
  # critical test is the 3-way interaction?

  # rewrite to use expand.grid ?
  labID = rep(as.character(1:n_labs), each=n_per_lab)
  subjID = 1:(n_labs*n_per_lab)

  familiarization_times = c(5,10,15)
  stimulus_types = c(rep("high",4), rep("low",4)) # stimulus complexity
  # trials each subject gets (but randomly ordered)
  fam_by_stim = expand.grid(fam_time = familiarization_times, stimulus_type = stimulus_types)

  # assume each lab uses one procedure
  lab_procedure = sample(c("IC","FD"), n_labs, replace=T, prob=c(.5,.5)) # 50/50 IC / FD procedures?
  procedure = rep(lab_procedure, each=n_per_lab)

  test_order = rep(1:4, n_per_lab/4*n_labs)

  # per-subject data
  simd <- tibble(subjID, labID, procedure, test_order) %>%
    mutate(subjInt = rnorm(length(subjID), mean=0, sd=1))

  # add lab random intercept
  simd$labInt = 0.0
  for(lab in unique(labID)) {
    labInd = which(simd$labID==lab)
    simd[labInd,]$labInt = rnorm(1, mean=0, sd=1) # could increase per-lab variability ..
  }

  # uniform random vars
  simd$age_mos = runif(nrow(simd), min=3.0, max=15.0)
  simd$age = scale(simd$age_mos, center=T, scale=T)[,1]

  # generate per-subject data, put in long (row per-trial) df

  siml <- tibble()
  for(i in 1:nrow(simd)) {
    # randomized trial order (but maybe should be done according to preset pseudorandom orders?)
    tmp_sdat <- fam_by_stim[sample(1:nrow(fam_by_stim), size=nrow(fam_by_stim), replace=F),]
    # need novel and familiar looking times, to calculate prop_novel --
    # UNLESS prop_novel turns out to be normally-distributed
    # (which would be easier to generate, requiring fewer assumptions) - check empirical trial-level da
    stimulus_type = with(tmp_sdat, ifelse(stimulus_type=="high", .5, -.5))
    error_term = rnorm(nrow(tmp_sdat), 0, sd=1) + simd[i,]$labInt + simd[i,]$subjInt # add random slope
    # rescale error to be >0
    # ToDo: scale familiarization time ?
    age_effect_subj = effect_sizes$age * rep(simd[i,]$age, nrow(tmp_sdat))

    # can we assume these are z-scored proportions of novel looking? maybe truncate them?
    # ToDo: check if problems when effect sizes are 0?
    tmp_sdat$dv_zscore = effect_sizes$type * stimulus_type + # main
      age_effect_subj +  # main
      effect_sizes$familiarization * tmp_sdat$fam_time +  # main
      effect_sizes$`age*type` * stimulus_type * effect_sizes$type * age_effect_subj +
      effect_sizes$`age*familiarization` * age_effect_subj * tmp_sdat$fam_time * effect_sizes$familiari
      effect_sizes$`type*familiarization` * tmp_sdat$fam_time * stimulus_type * effect_sizes$type +
      effect_sizes$`type*age*familiarization` * stimulus_type * effect_sizes$type * age_effect_subj * tr
      error_term
    # standardize from normal means to beta mean: .5+/-.3 ??
    siml <- siml %>%
      bind_rows(tmp_sdat %>% mutate(subjID = simd[i,]$subjID,
                                    labID = simd[i,]$labID,
                                    age = simd[i,]$age,
                                    age_mos = simd[i,]$age_mos,
                                    subjInt = simd[i,]$subjInt,
                                    labInt = simd[i,]$labInt,
                                    trial_num = 1:nrow(tmp_sdat)))
        #novel_looking_time = rnorm(n = nrow(tmp_sdat), mean=0, sd=1), # = .05
        #familiar_looking_time = rnorm(n = nrow(tmp_sdat), mean=0, sd=1), # = .05
        #prop_novel = novel_looking_time / (novel_looking_time + familiar_looking_time), # use beta d
        #prop_novel = rbeta(n=nrow(tmp_sdat), shape1=??, shape2=??)
        # mean_beta = .5 + familiarization_time*age*type
             # how to choose beta parameters: more non-central = more of a novelty/familiarity effect
  }

  siml$trial_num_sc = scale(siml$trial_num, center=T, scale=T)

  siml$subjID = as.factor(siml$subjID)
  # switch from dummy-code to effects code
  siml$stimulus_type = as.factor(siml$stimulus_type)
  contrasts(siml$stimulus_type) = contr.sum(2)
  return(siml)
}
```

Plot Example Dataset

We generate and plot an example dataset with stimulus *type* main effect size of .3, *age* main effect size of -.2, and an age*trial_type interaction effect size of .3.

```
siml = generate_dataset(effect_sizes=list(type = .3, familiarization = .3, age = .3, "age*type"=.3,
                                          "age*familiarization"=.3, "type*familiarization"=.3,
                                          "type*age*familiarization"=.3))

d_sub <- siml %>% mutate(age_group = cut_interval(age_mos, length=3)) %>%
  group_by(subjID, stimulus_type, fam_time, age_group) %>%
  summarise(dv_zscore = mean(dv_zscore))
```

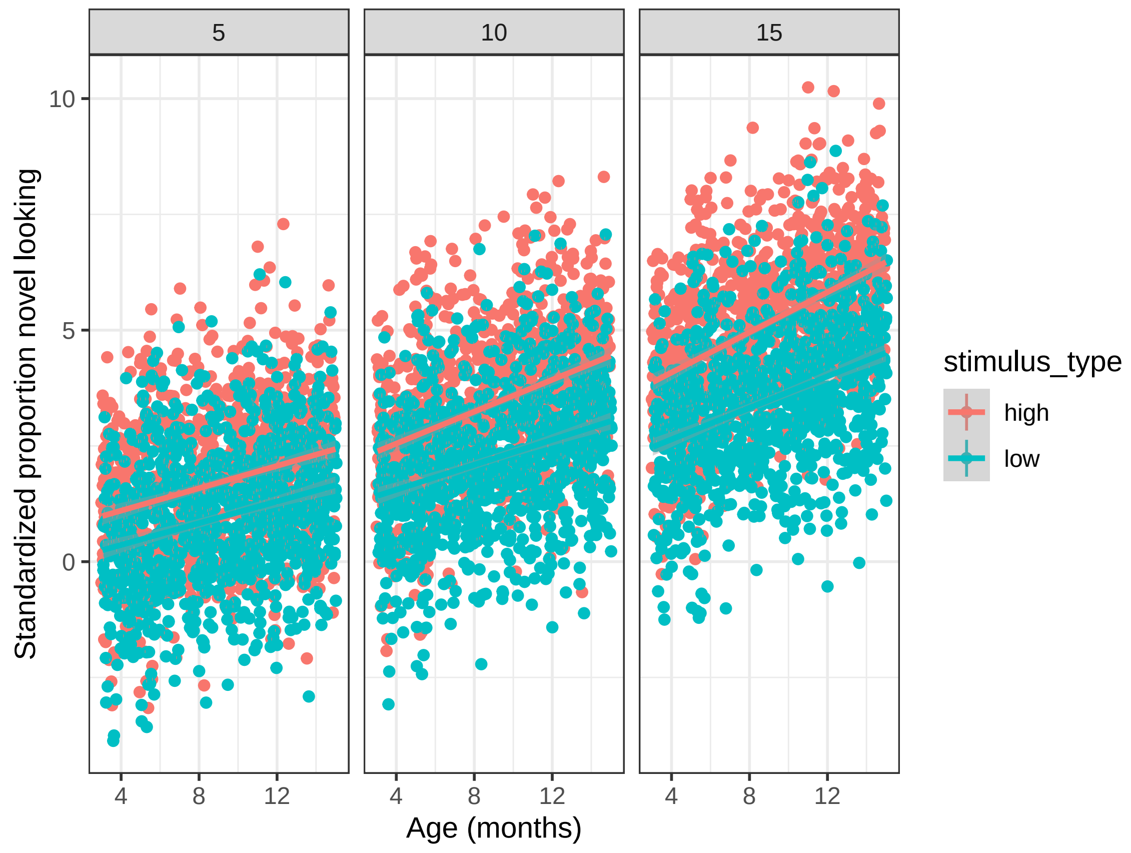
```
## `summarise()` has grouped output by 'subjID', 'stimulus_type', 'fam_time'. You
## can override using the `.groups` argument.
```

```
pos = position_dodge(width=.2)
# age group faceting
# dag <- siml %>% mutate(age_group = cut_interval(age_mos, length=3)) %>%
#   group_by(subjID, stimulus_type, fam_time, age_group) %>%
#   summarise(dv_zscore = mean(dv_zscore)) %>%
#   group_by(stimulus_type, fam_time, age_group) %>%
#   tidyboot::tidyboot_mean(dv_zscore) # quite slow..
#
# ggplot(dag, aes(x=fam_time, y=mean, group=stimulus_type, color=stimulus_type)) +
#   facet_wrap(. ~ age_group) +
#   ylab("Standardized proportion novel looking") + xlab("Familiarization Time") +
#   geom_linerange(aes(ymin=ci_lower, ymax=ci_upper), pos=pos) +
#   geom_point(data=d_sub, aes(x=jitter(fam_time), y=dv_zscore), alpha=.2) +
#   theme_bw() + geom_smooth(method="lm")

dag <- siml %>%
  group_by(subjID, stimulus_type, fam_time, age_mos) %>%
  summarise(dv_zscore = mean(dv_zscore)) %>%
  group_by(stimulus_type, fam_time, age_mos) %>%
  tidyboot::tidyboot_mean(dv_zscore)
```

```
## `summarise()` has grouped output by 'subjID', 'stimulus_type', 'fam_time'. You
## can override using the `.groups` argument.

## Warning: `as_data_frame()` was deprecated in tibble 2.0.0.
## Please use `as_tibble()` instead.
## The signature and semantics have changed, see `?as_tibble`.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was generated.

## Warning: `cols` is now required when using unnest().
## Please use `cols = c(strap)`
```

```
ggplot(dag, aes(x=age_mos, y=mean, group=stimulus_type, color=stimulus_type)) +
  facet_wrap(. ~ fam_time) +
  geom_point(aes(y=mean, x=age_mos), pos=pos) +
  ylab("Standardized proportion novel looking") + xlab("Age (months)") +
  geom_linerange(aes(ymin=ci_lower, ymax=ci_upper), pos=pos) +
  theme_bw() + geom_smooth(method="lm")
```

```
## `geom_smooth()` using formula 'y ~ x'

## Warning: position_dodge requires non-overlapping x intervals

## Warning: position_dodge requires non-overlapping x intervals
## position_dodge requires non-overlapping x intervals
## position_dodge requires non-overlapping x intervals
## position_dodge requires non-overlapping x intervals
## position_dodge requires non-overlapping x intervals
```



Model Structure

Infants' proportion of looking at novel object (DV) ~ 1 + familiarization time (5, 10, 15) * stimulus type (high/low complexity) * age + (fam_time*stim_type | subject) + (fam_time*stim_type*age | lab)

```
# power for either just main effects, or just the 3-way
main_efs <- c("stimulus_type1","age_mos","fam_time")
                  #"stimulus_type1:fam_time", "stimulus_type1:age_mos", "fam_time:age_mos",
                  #"stimulus_type1:fam_time:age_mos")

# power for just the 3-way
fit_simple_model <- function(siml) {
  m1 <- lmer(dv_zscore ~ 1 + stimulus_type * fam_time * age_mos + (1 | subjID) + (1 | labID), data=siml
  return(summary(m1)$coefficients[c(main_efs,"stimulus_type1:fam_time:age_mos"),"Pr(>|t|)"]) # "Estimat
}

fit_simple_model(siml)
```

```
##                  stimulus_type1                         age_mos
##                    1.903245e-05                    1.285029e-12
##                        fam_time stimulus_type1:fam_time:age_mos
##                    0.000000e+00                    2.754550e-06
```

```r
# check both
fit_model <- function(siml) {
  m1 <- lmer(dv_zscore ~ 1 + stimulus_type * fam_time * age_mos +
               (fam_time*stimulus_type | subjID) + (fam_time*stimulus_type | labID), data=siml)
  sig =c(#summary(m1)$coefficients["stimulus_type1","Pr(>|t|)"],
       summary(m1)$coefficients[c(main_efs,"stimulus_type1:fam_time:age_mos"),"Pr(>|t|)"])
  return(sig) # "Estimate","t value",
}

#fit_model(siml) # boundary (singular) fit -- and is quite slow
```

Power Analysis

We use this simplified model for the power analysis: y ~ 1 + * stimulus_type * age * fam_time + (1 | subjID) + (1 | labID)

To do the power analysis, we simply generate 1000 datasets with main effect sizes of 0.1, 0.2, and 0.3 for trial type, age, and their interaction, run the above linear mixed-effects model, and report how many times 1) the trial type main effect and 2) the trial type * age interaction is significant.

```r
# repeatedly generate data and  significance of trial_typesame
get_power <- function(effect_sizes, N=100, alpha=.05, verbose=F) {
  #p = data.frame(type=numeric(), "stimulus_type*age_mos*fam_time"=numeric())
  p = tibble()
  # parallelize
  #colnames(p) = c("stimulus_type","age_mos","fam_time","stimulus_type*age_mos*fam_time")
  for(i in 1:N) {
    p <- p %>% bind_rows(fit_simple_model(generate_dataset(effect_sizes=effect_sizes)))
  }
  if(verbose) {
    #print(paste(length(which(p$type<alpha)), "of",N, "simulations had p <",alpha, "for trial type"))
    print(paste(length(which(p<alpha)), "of",N, "simulations had p <",alpha, "for stimulus_type*age_mos
  }
  return(p)
}

N = 1000

effect_size_pt1 = list(type = .1, familiarization = .1, age = .1, "age*type"=.1,
                                               "age*familiarization"=.1, "type*familiarization"=.1,
                                               "type*age*familiarization"=.1)
effect_size_pt2 = list(type = .2, familiarization = .2, age = .2, "age*type"=.2,
                                               "age*familiarization"=.2, "type*familiarization"=.2,
                                               "type*age*familiarization"=.2)
effect_size_pt3 = list(type = .3, familiarization = .3, age = .3, "age*type"=.3,
                                               "age*familiarization"=.3, "type*familiarization"=.3,
                                               "type*age*familiarization"=.3)

pvalues_pt1 = get_power(effect_sizes=effect_size_pt1, N=N)

pvalues_pt2 = get_power(effect_sizes=effect_size_pt2, N=N)

pvalues_pt3 = get_power(effect_sizes=effect_size_pt3, N=N)
```

Effect sizes = .1

147 of 1000 simulations had $p < 0.05$ for stimulus type. 775 of 1000 simulations had $p < 0.05$ for age. 1000 of 1000 simulations had $p < 0.05$ for familiarization time. 52 of 1000 simulations had $p < 0.05$ for age * stimulus type * familiarization time. 13 of 100 simulations had $p < 0.05$ for stimulus type. 73 of 100 simulations had $p < 0.05$ for age. 100 of 100simulations had $p < 0.05$ for familiarization time. 4 of 100 simulations had $p < 0.05$ for age * stimulus type* familiarization time.

Effect sizes = .2

431 of 1000 simulations had $p < 0.05$ for stimulus type. 1000 of 1000 simulations had $p < 0.05$ for age. 1000 of 1000 simulations had $p < 0.05$ for familiarization time. 68 of 1000 simulations had $p < 0.05$ for age * stimulus type * familiarization time.

Effect sizes = .3

667 of 1000 simulations had $p < 0.05$ for stimulus type. 1000 of 1000 simulations had $p < 0.05$ for age. 1000 of 1000 simulations had $p < 0.05$ for familiarization time. 720 of 1000 simulations had $p < 0.05$ for age * stimulus type * familiarization time.

For context, .3 is the average effect size across all published developmental experiments. (Any idea of the average empirical effect size (of age, complexity, or familiarization time) for habituation experiments??)