ManyBabies 5: The Hunter and Ames Model of Infant Looking Preference

Supplementary Materials: Data Simulation and Power Analysis

ManyBabies Analysis Team

Contents

1 Data Simulation **2**

2 Visualisation of Simulated Data **4**

2.1 Familiarization: . . . 4

2.2 Complexity: . . . 5

2.3 Age: . . . 6

2.4 Age*Familiarization: . . . 7

2.5 Age*Complexity: . . . 8

2.6 Familiarization*Complexity: . . . 9

2.7 Variability by Lab . . . 11

2.8 Variability by Item . . . 12

3 Overview of Power Simulation Results **13**

3.1 Summary Statistics for Power Calculation with Full Data and Varying Intercepts and Varying Slopes: . . . 13

3.2 Summary Statistics for Power Calculation with Full Data and Varying Intercepts: . . . 13

3.3 Summary Statistics for Power Calculation with 20 pct. Missing Data and Varying Intercepts: . . . 13

3.4 Summary Statistics for Power Calculation with 50 pct. Missing Data and Varying Intercepts: . . . 13

4 Overview of Bias Results **14**

5 Grid Search Code **14**

6 Power Calculation with Full Data and Varying Intercepts and Varying Slopes **20**

6.1 Effect Size = 0.5 . . . 20

6.2 Effect Size = 0.4 . . . 21

6.3 Effect Size = 0.3 . . . 21

6.3.1 Visualise Estimates for Fixed Effects: . . . 21

6.3.2 Visualise Estimates for Random Effects: . . . 23

6.4 Effect Size = 0.2 . . . 24

6.5 Effect Size = 0.1 . . . 24

7 Power Calculation with Full Data and Varying Intercepts **24**

7.1 Effect Size = 0.5 . . . 24

7.2 Effect Size = 0.4 . . . 25

7.3 Effect Size = 0.3 . . . 25

7.3.1 Visualise Estimates for Fixed Effects: . . . 25

7.3.2 Visualise Estimates for Random Effects: . . . 26

7.4 Effect Size = 0.2 . . . 27

7.5 Effect Size = 0.1 . . . 27

8 Power Calculation with 20 pct. Missing Data and Varying Intercepts **28**

8.1 Effect Size = 0.5 . . . 28

8.2 Effect Size = 0.4 . . . 28

8.3 Effect Size = 0.3 . . . 29

8.3.1 Visualise Estimates for Fixed Effects: . . . 29

8.3.2 Visualise Estimates for Random Effects: . . . 29

8.4 Effect Size = 0.2 . . . 29

8.5 Effect Size = 0.1 . . . 30

9 Power Calculation with 50 pct. Missing Data and Varying Intercepts **30**

9.1 Effect Size = 0.5 . . . 30

9.2 Effect Size = 0.4 . . . 31

9.3 Effect Size = 0.3 . . . 31

9.3.1 Visualise Estimates for Fixed Effects: . . . 31

9.3.2 Visualise Estimates for Random Effects: . . . 31

9.4 Effect Size = 0.2 . . . 32

9.5 Effect Size = 0.1 . . . 32

1 Data Simulation

```
my_sim_data <- function(
    n_subj      = 1280,   # number of subjects
    n_simple  =  6,   # number of complex stimuli
    n_complex =  6,   # number of complex stimuli
    n_small_fam = 4,   #small familiarization time
    n_medium_fam = 4,  #medium familiarization time
    n_high_fam = 4,    #high familiarization time
    n_lab = 40,

    beta_0  =  0, # intercept; i.e., the grand mean
    beta_c  =  0.3, # main effect for complexity
    beta_f = 0.3, # main effect for familiarization time
    beta_a = 0.3, # main effect for age

    beta_ca = 0.3,
    beta_af = 0.3,
    beta_cf = 0.3,

    beta_cfa = 0.3, #main effect for interaction between complexity and familiarization.

    subject_0   = 0.2, # by-subject random intercept sd

    subject_c   =  0.2, # by-subject slope complexity sd
    subject_f = 0.2, # by-subject slope familiarization sd
    subject_a = 0.2, # by-subject slope age sd

    subject_ca = 0.2,# by-subject slope for interaction between age and complexity sd
    subject_af = 0.2, # by-subject slope for interaction between age and familiarization sd
    subject_cf = 0.2, # by-subject slope complexity*familiarization sd

    subject_cfa = 0.2, # by-subject slope for interaction between age, complexity and familiarization sd

    subj_rho     =  .2, # correlations between by-subject random effects

    lab_0 = 0.2, # by-lab random intercept sd

    lab_c = 0.2, # by-lab slope complexity sd
    lab_f = 0.2, # by-lab slope familiarization sd
    lab_a = 0.2, # by-lab slope age sd

    lab_ca = 0.2, # by-lab slope for interaction between age and complexity sd
    lab_af = 0.2, # by-lab slope for interaction between age and familiarization sd
    lab_cf = 0.2, # by-lab random slope complexity*familiarization sd

    lab_cfa = 0.2, # by-lab slope for interaction between age, complexity and familiarization sd

    lab_rho = 0.2, # correlations between by-lab random effects

    item_0 =  0.05, # by-item random intercept sd

    item_c = 0.05, # by-item slope complexity sd
    item_f = 0.05, # by-item slope familiarization sd
    item_a = 0.05, # by-item slope age sd

    item_ca = 0.05, # by-item slope for interaction between age and complexity sd
    item_af = 0.05, # by-item slope for interaction between age and familiarization sd
    item_cf = 0.05, # by-item random slope complexity*familiarization sd

    item_cfa = 0.05, # by-item slope for interaction between age, complexity and familiarization sd
```

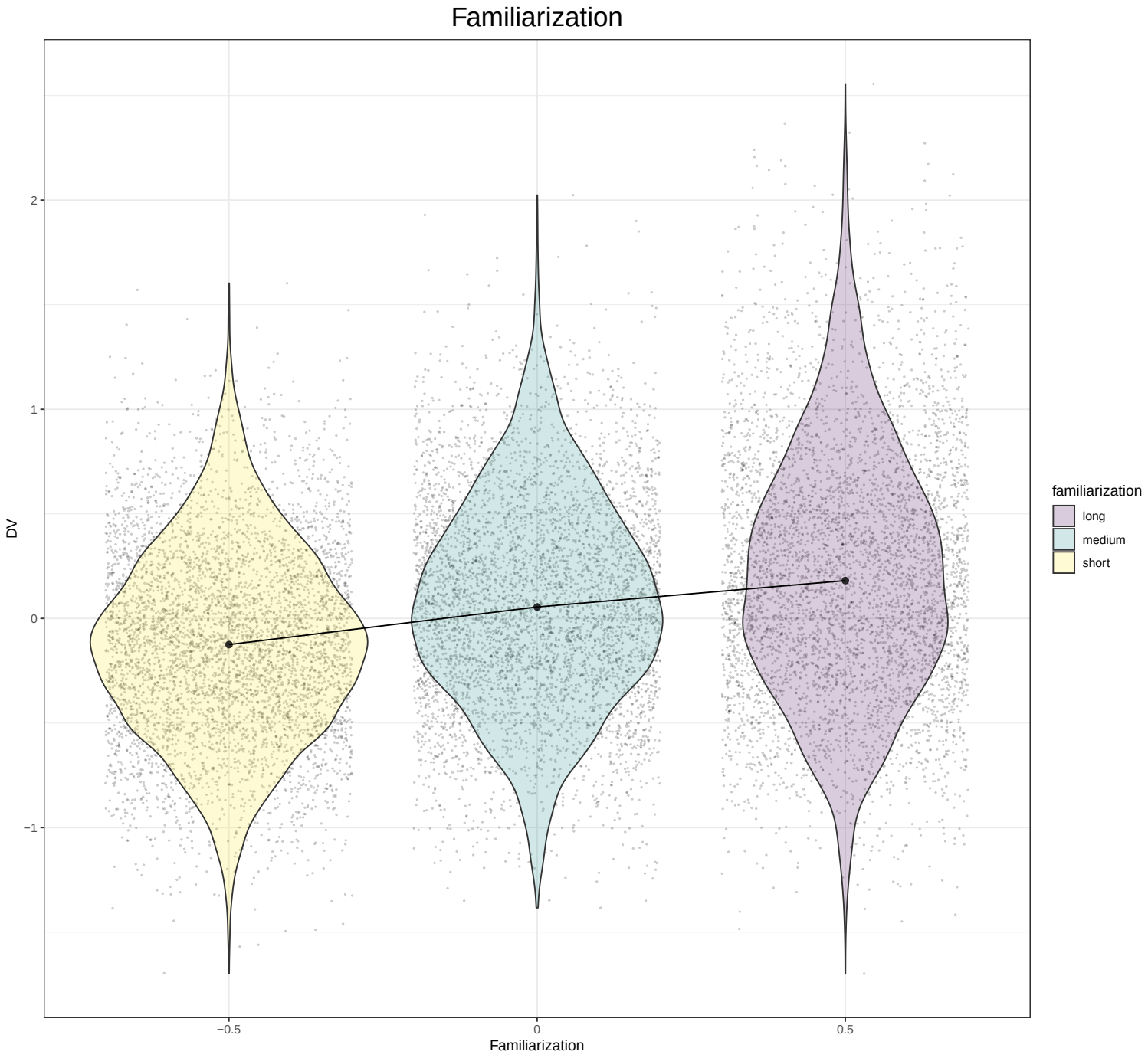
```
  item_rho = 0.2, # correlations between by-item random effects


  sigma = 0.3 # residual (error) sd
  ) { # residual (standard deviation)

# simulate a sample of items
items <- data.frame(
  category = rep(c("simple", "complex"), c(n_simple, n_complex)),
  X_c = rep(c(-0.5, 0.5), c(n_simple, n_complex)),
  familiarization = rep(c("short", "medium", "long"), (n_simple + n_complex)/3),
  X_f = rep(c(-0.5, 0, 0.5), (n_simple + n_complex)/3),
  faux::rnorm_multi(
  n = n_simple + n_complex, mu = 0, sd = c(item_0,
                              item_c,
                              item_f,
                              item_a,
                              item_ca,
                              item_af,
                              item_cf,
                              item_cfa), r = item_rho,
  varnames = c("I_0", "I_c","I_f","I_a",
               "I_ca","I_af", "I_cf",
               "I_cfa"))
  ) %>%
  mutate(item_id = faux::make_id(nrow(.), "I"))

# simulate a sample of subjects
subjects <-
  faux::rnorm_multi(
  n = n_subj, mu = 0, sd = c(subject_0,
                             subject_c,
                             subject_f,
                             subject_a,
                             subject_ca,
                             subject_af,
                             subject_cf,
                             subject_cfa), r = subj_rho,
  varnames = c("S_0", "S_c","S_f","S_a",
               "S_ca","S_af", "S_cf",
               "S_cfa")
) %>%
  mutate(subj_id = faux::make_id(nrow(.), "S")) %>%
  mutate(X_a = runif(n_subj, min = -0.5, max = 0.5))
#add subject age measure, sample from distribution from -0.5 to 0.5. #subjects$subj_id <- 1:n_subj

labs <- faux::rnorm_multi(
  n = n_lab, mu = 0, sd = c(lab_0, lab_c, lab_f, lab_a,
                            lab_ca, lab_af, lab_cf,
                            lab_cfa), r = lab_rho,
  varnames = c("L_0", "L_c","L_f","L_a",
               "L_ca","L_af", "L_cf",
               "L_cfa")
) %>%
  mutate(lab_id = faux::make_id(nrow(.), "L"))

#create lab and subj nesting structure
#Number of subjects must be a multiple of number of labs
lab_multiplier = n_subj/n_lab
lab_subj_dict <- data.frame(
  subj_id = subjects$subj_id,
```

```
    lab_id = rep(labs$lab_id,lab_multiplier)
  )

  # cross subject and item IDs
  temp <- crossing(subjects, items)  %>%
    left_join(lab_subj_dict, by = "subj_id") %>%
    left_join(labs, by = "lab_id") %>%
    group_by(subj_id, item_id) %>%
    mutate(item_id = sample(item_id)) %>%
    ungroup() %>%
    mutate(trial_num = rep(seq(n_simple + n_complex), n_subj))

  temp <- temp %>%
    mutate(
      B_0  = beta_0 + S_0 + L_0 + I_0,

      B_c  = beta_c + S_c + L_c + I_c,
      B_f  = beta_f + S_f + L_f + I_f,
      B_a = beta_a + S_a + L_a + I_a,

      B_ca = beta_ca + S_ca + L_ca + I_ca,
      B_af = beta_af + S_af + L_af + I_af,
      B_cf = beta_cf + S_cf + L_cf + I_cf,

      B_cfa = beta_cfa + S_cfa + L_cfa + I_cfa,

      e_si = rnorm(nrow(temp), mean = 0, sd = sigma),

      DV = B_0 +
        (B_a * X_a) + (B_c * X_c) + (B_f * X_f) +
        (B_cf * X_c * X_f) + (B_af * X_a * X_f) + (B_ca * X_c * X_a) +
        (B_cfa * X_c * X_f * X_a) + e_si
    )
}

dat_sim <- my_sim_data()
```

2 Visualisation of Simulated Data

2.1 Familiarization:

```
dat_sim_plot_familiarization <- dat_sim %>%
    group_by(X_f) %>%
    dplyr::summarise(med_DV = median(DV))

plot_familiarization <- dat_sim %>%
    mutate(X_f = as.factor(X_f)) %>%
    ggplot() + geom_point(aes(y = DV, x = X_f), position = "jitter",
    alpha = 0.2, size = 0.2) + geom_violin(aes(y = DV, x = X_f,
    fill = familiarization), alpha = 0.2) + geom_line(aes(y = med_DV,
    x = as.factor(X_f), group = 1), data = dat_sim_plot_familiarization) +
    geom_point(aes(y = med_DV, x = as.factor(X_f)), alpha = 0.8,
        size = 2, data = dat_sim_plot_familiarization) + scale_fill_manual(values = viridis(n = 3)) +
    ggtitle("Familiarization") + xlab("Familiarization") + theme_bw()

plot_familiarization <- plot_familiarization + theme(plot.title = element_text(hjust = 0.5,
    size = 20))
plot_familiarization
```



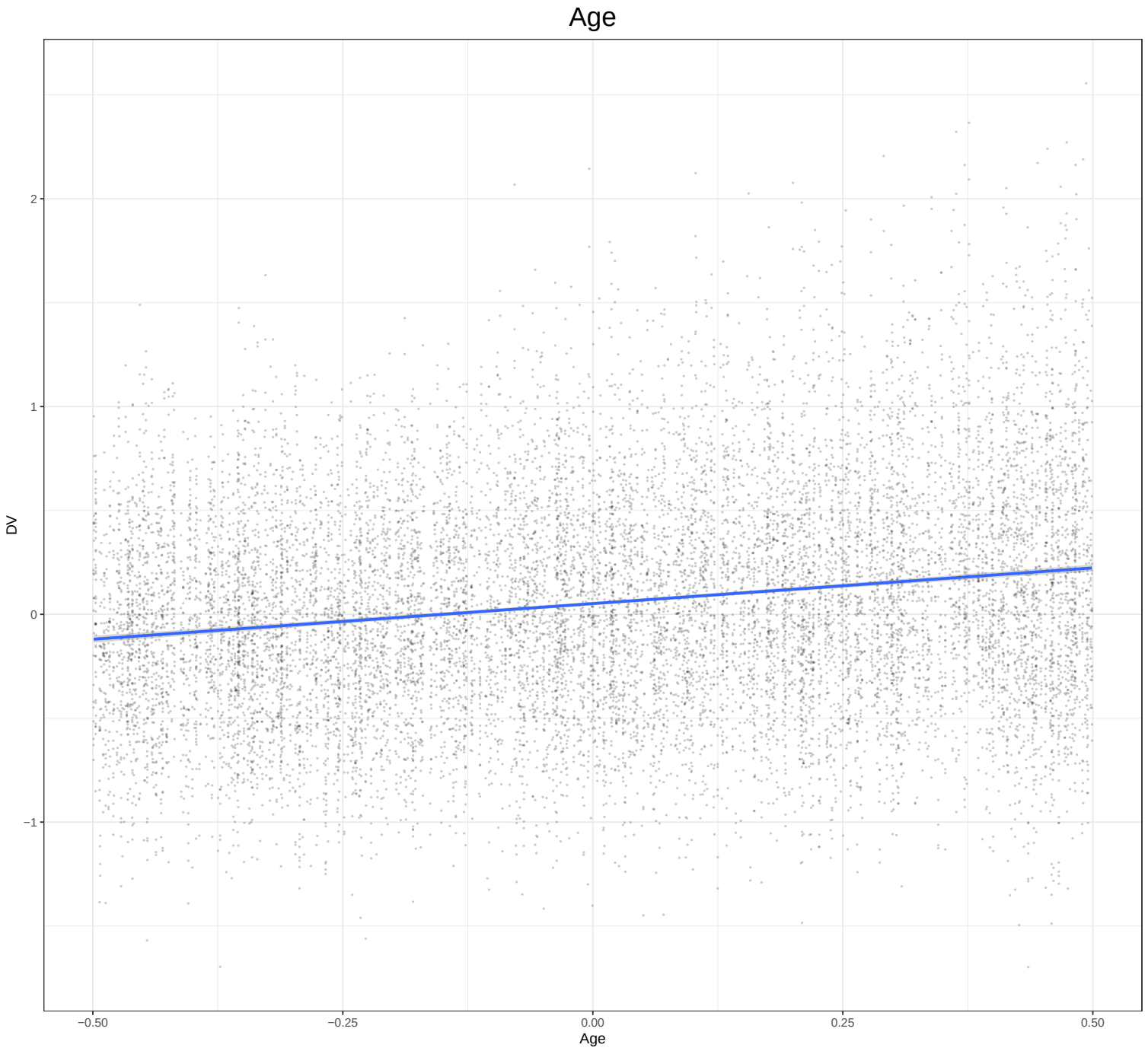
2.2 Complexity:

```
dat_sim_plot_complexity <- dat_sim %>%
    group_by(X_c) %>%
    dplyr::summarise(med_DV = median(DV))

plot_complexity <- dat_sim %>%
    mutate(X_c = as.factor(X_c)) %>%
    ggplot() + geom_point(aes(y = DV, x = X_c), position = "jitter",
    alpha = 0.2, size = 0.2) + geom_violin(aes(y = DV, x = X_c,
    fill = category), alpha = 0.2) + geom_line(aes(y = med_DV,
    x = as.factor(X_c), group = 1), data = dat_sim_plot_complexity) +
    geom_point(aes(y = med_DV, x = as.factor(X_c)), alpha = 0.8,
        size = 2, data = dat_sim_plot_complexity) + scale_fill_manual(values = viridis(n = 2)) +
    ggtitle("Stimulus Complexity") + xlab("Stimulus Complexity") +
    theme_bw()
```

```
plot_complexity <- plot_complexity + theme(plot.title = element_text(hjust = 0.5,
    size = 20))
plot_complexity
```

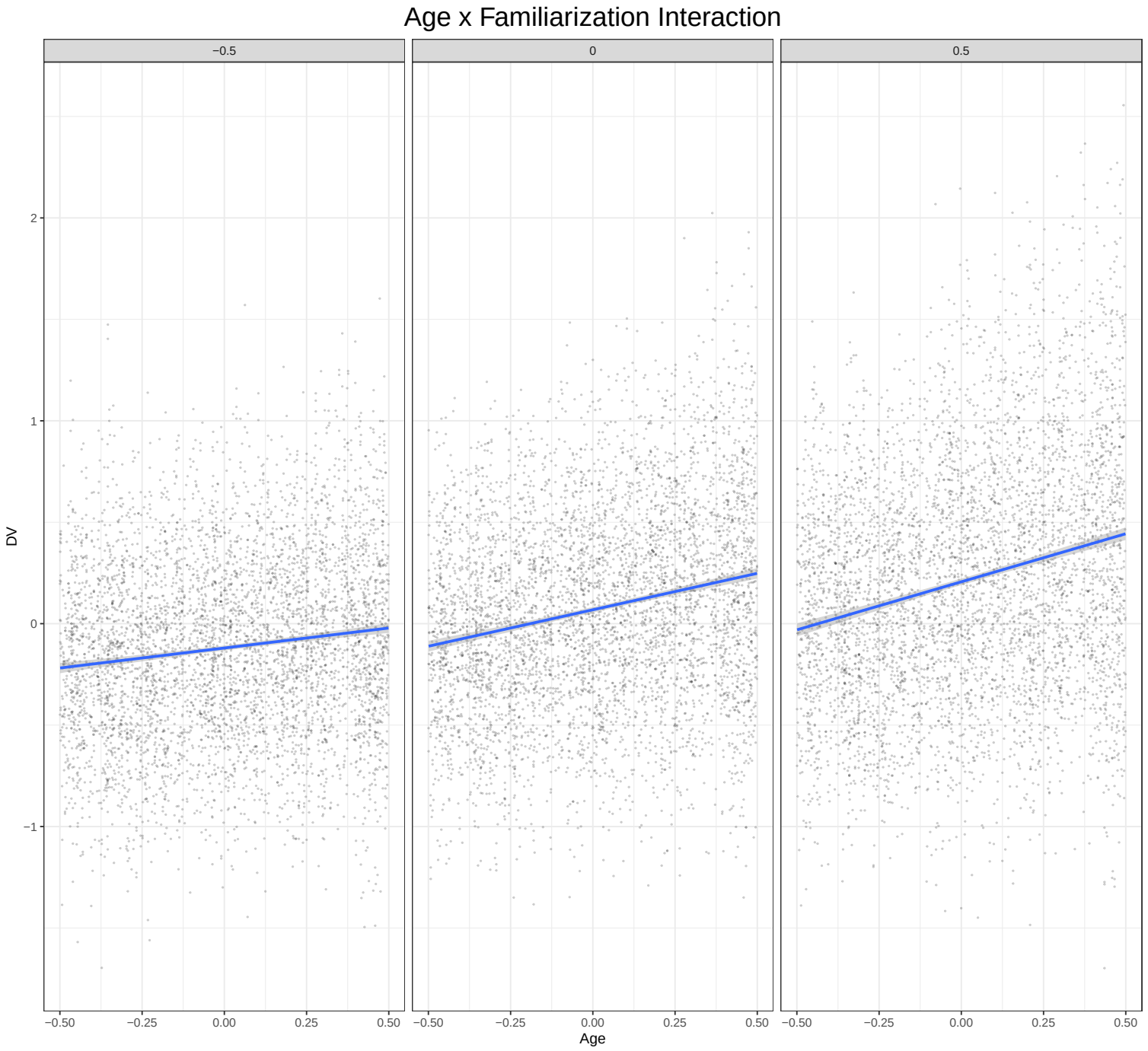
2.3 Age:

```
plot_age <- dat_sim %>%
    ggplot() + geom_point(aes(y = DV, x = X_a), position = "jitter",
    alpha = 0.2, size = 0.2) + geom_smooth(method = "lm", se = TRUE,
    formula = y ~ x, aes(y = DV, x = X_a)) + ggtitle("Age") +
    xlab("Age") + theme_bw()

plot_age <- plot_age + theme(plot.title = element_text(hjust = 0.5,
    size = 20))
plot_age
```



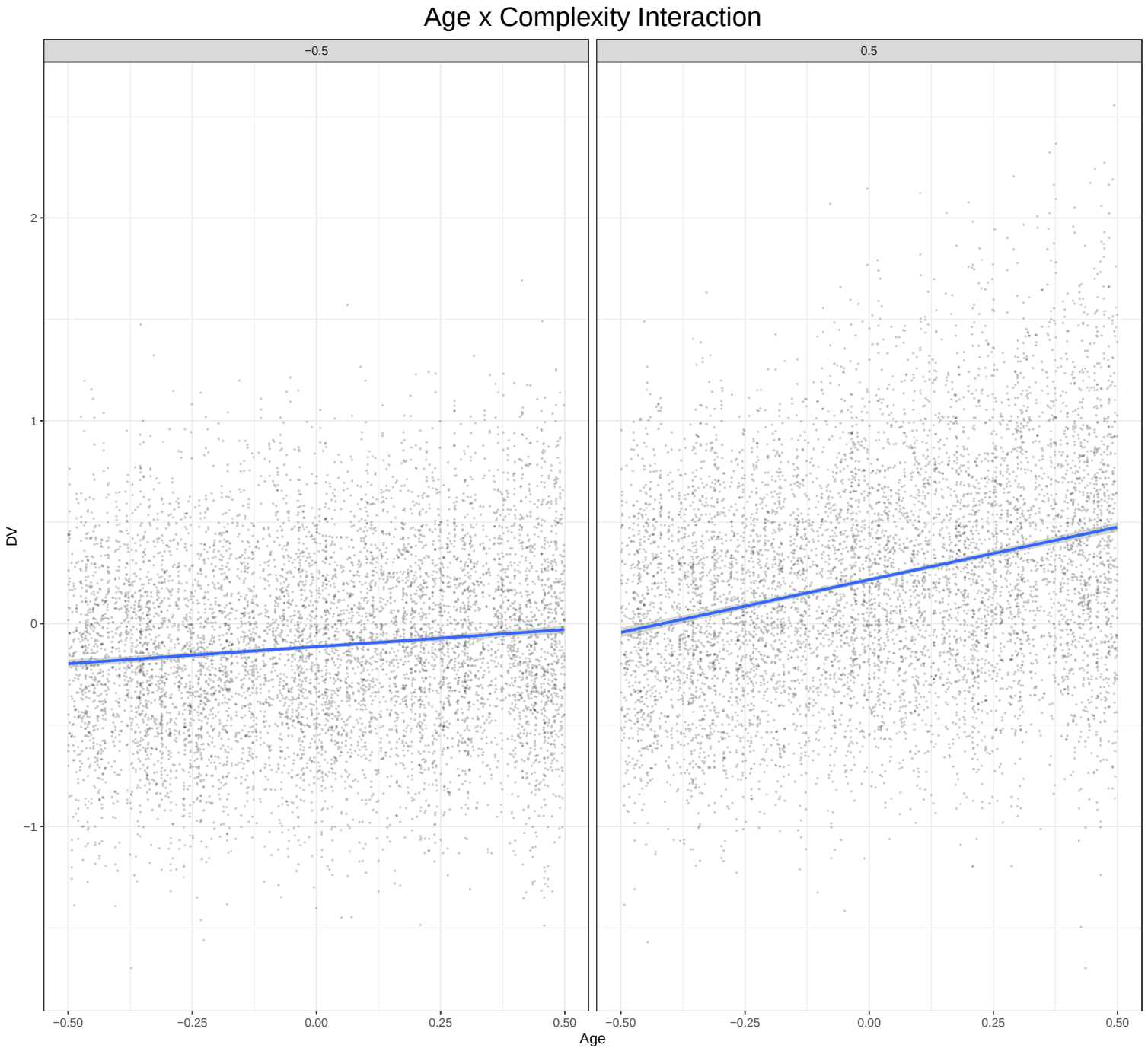
2.4 Age*Familiarization:

```
plot_age_familiarization <- dat_sim %>%
    ggplot() + geom_point(aes(y = DV, x = X_a), position = "jitter",
    alpha = 0.2, size = 0.2) + geom_smooth(method = "lm", formula = y ~
    x, se = TRUE, aes(y = DV, x = X_a)) + facet_wrap(~X_f) +
    ggtitle("Age x Familiarization Interaction") + xlab("Age") +
    theme_bw()
plot_age_familiarization <- plot_age_familiarization + theme(plot.title = element_text(hjust = 0.5,
    size = 20))
plot_age_familiarization
```



2.5 Age*Complexity:

```
plot_age_complexity <- dat_sim %>%
    ggplot() + geom_point(aes(y = DV, x = X_a), position = "jitter",
    alpha = 0.2, size = 0.2) + geom_smooth(method = "lm", formula = y ~
    x, se = TRUE, aes(y = DV, x = X_a)) + facet_wrap(~X_c) +
    ggtitle("Age x Complexity Interaction") + xlab("Age") + theme_bw()

plot_age_complexity <- plot_age_complexity + theme(plot.title = element_text(hjust = 0.5,
    size = 20))
plot_age_complexity
```



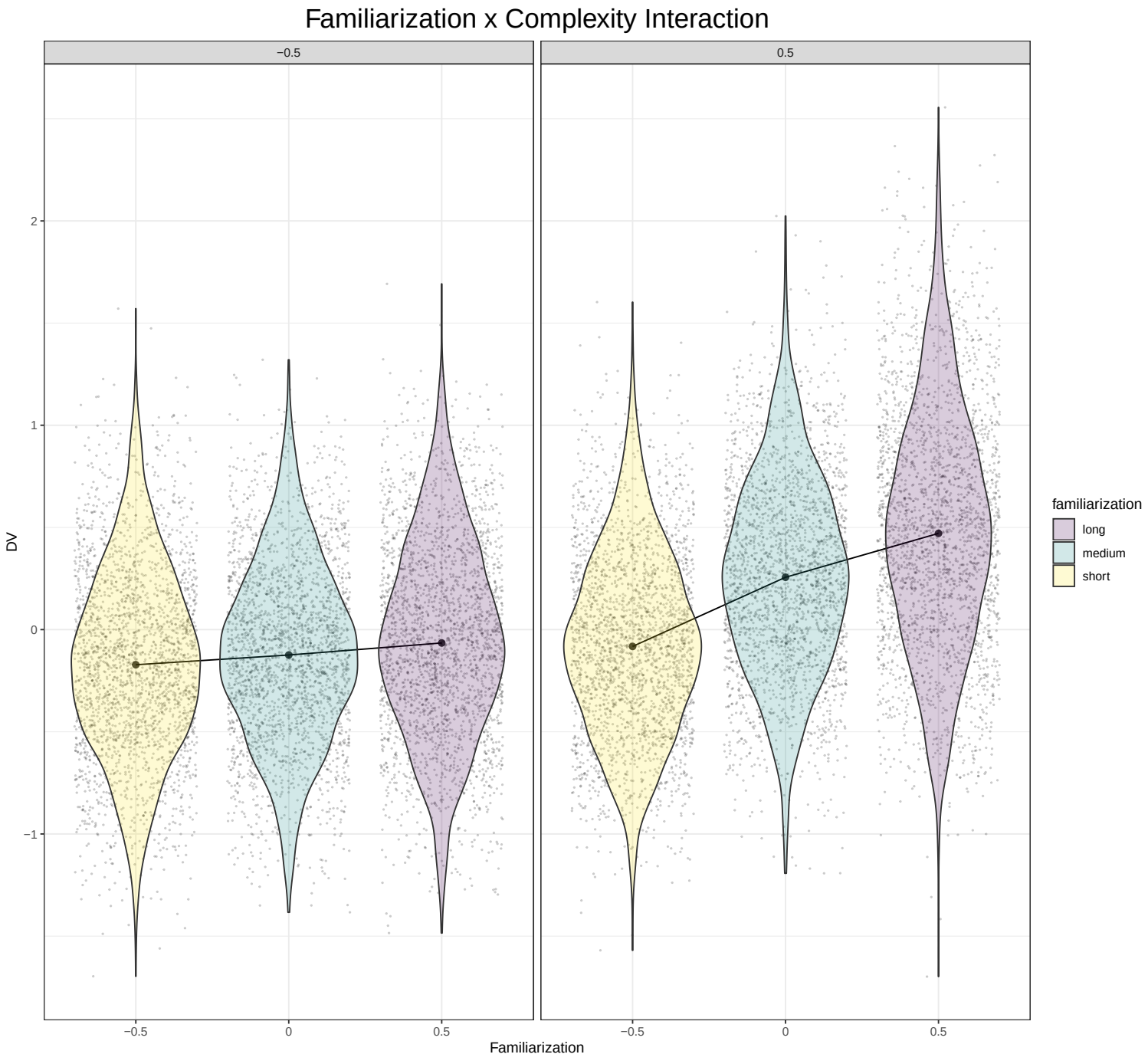
2.6 Familiarization*Complexity:

```
dat_f_c_interaction <- dat_sim %>%
    mutate(X_c = as.factor(X_c)) %>%
    mutate(X_f = as.factor(X_f)) %>%
    group_by(X_f, X_c) %>%
    dplyr::summarise(med_DV = median(DV))
```

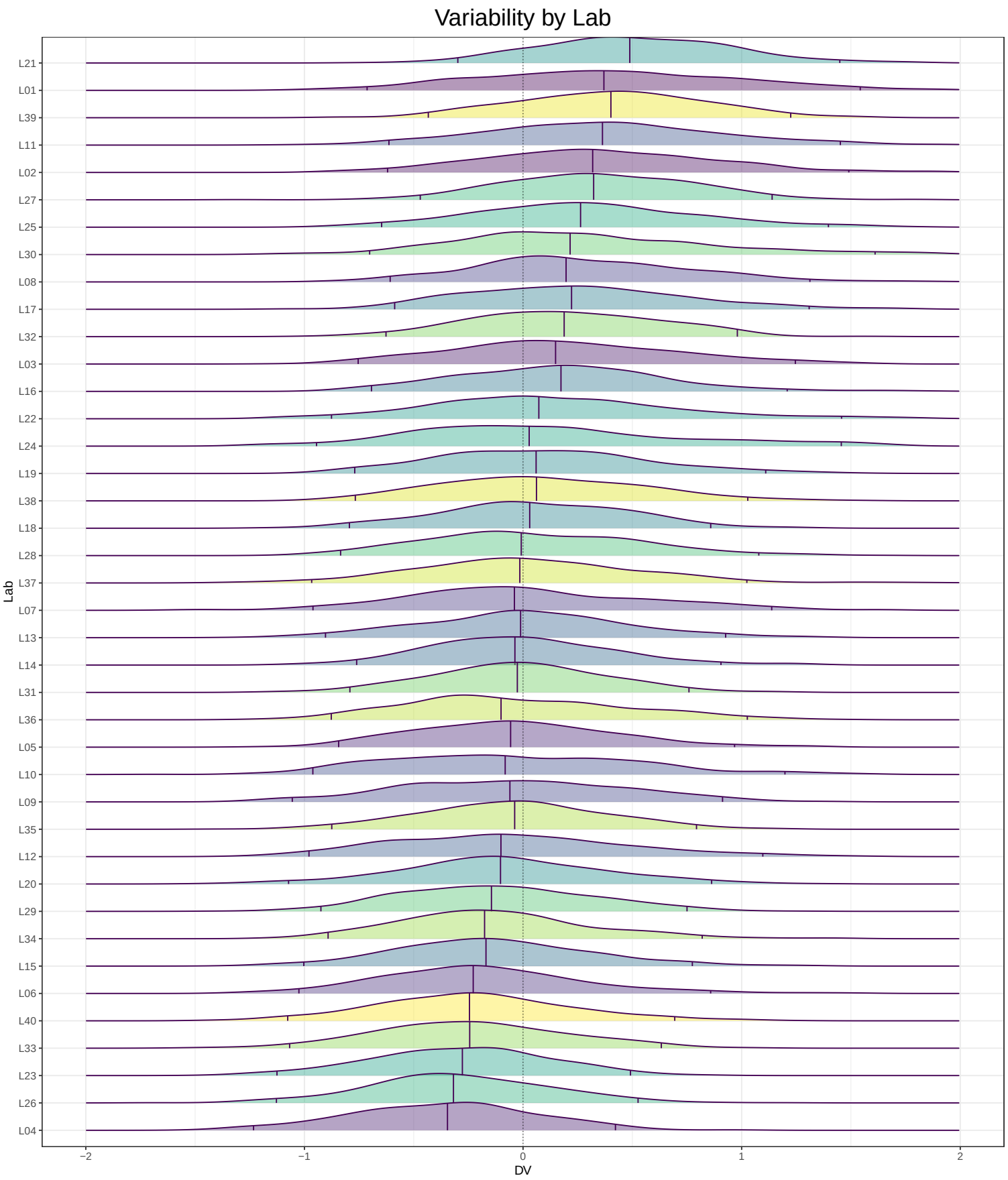
```
## `summarise()` has grouped output by 'X_f'. You can override using the `.groups`
## argument.
```

```
plot_familiarization_complexity <- dat_sim %>%
    mutate(X_c = as.factor(X_c)) %>%
    mutate(X_f = as.factor(X_f)) %>%
    ggplot() + geom_point(aes(y = DV, x = X_f), position = "jitter",
```

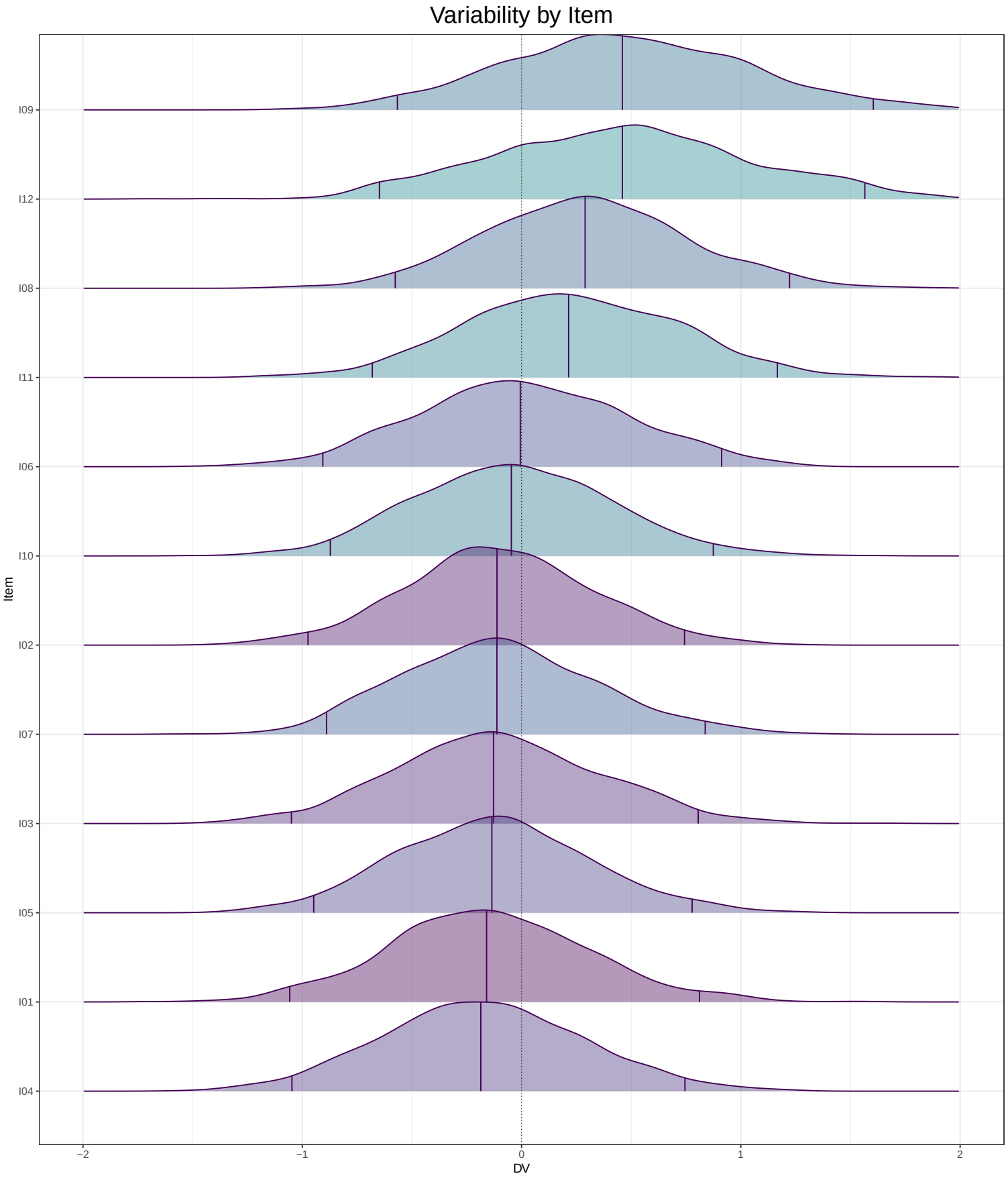
```
    alpha = 0.2, size = 0.2) + geom_point(aes(y = med_DV, x = as.factor(X_f)),
    alpha = 0.8, size = 2, data = dat_f_c_interaction) + geom_line(aes(y = med_DV,
    x = as.factor(X_f), group = 1), data = dat_f_c_interaction) +
    geom_violin(aes(y = DV, x = X_f, fill = familiarization),
        alpha = 0.2) + scale_fill_manual(values = viridis(n = 3)) +
    facet_wrap(~X_c) + ggtitle("Familiarization x Complexity Interaction") +
    xlab("Familiarization") + theme_bw()
plot_familiarization_complexity <- plot_familiarization_complexity +
    theme(plot.title = element_text(hjust = 0.5, size = 20))
plot_familiarization_complexity
```



2.7 Variability by Lab



2.8 Variability by Item



3 Overview of Power Simulation Results

3.1 Summary Statistics for Power Calculation with Full Data and Varying Intercepts and Varying Slopes:

Table 1: Power for Simulations with Full Data and Varying Intercepts and Varying Slopes

term	power, ef = 0.1	power, ef = 0.2	power, ef = 0.3	power, ef = 0.4	power, ef = 0.5
(Intercept)	0.056	0.044	0.056	0.038	0.044
X_a	0.892	1.000	1.000	1.000	1.000
X_c	0.648	0.988	1.000	1.000	1.000
X_f	0.514	0.966	1.000	1.000	1.000
X_a:X_c	0.744	0.994	1.000	1.000	1.000
X_a:X_f	0.702	0.976	0.998	1.000	1.000
X_c:X_f	0.190	0.600	0.824	0.968	1.000
X_a:X_c:X_f	0.468	0.878	0.972	1.000	1.000

3.2 Summary Statistics for Power Calculation with Full Data and Varying Intercepts:

Table 2: Power for Simulations with Full Data and Varying Intercepts

term	power, ef = 0.1	power, ef = 0.2	power, ef = 0.3	power, ef = 0.4	power, ef = 0.5
(Intercept)	0.060	0.070	0.052	0.038	0.048
X_a	0.896	1.000	1.000	1.000	1.000
X_c	0.638	0.988	1.000	1.000	1.000
X_f	0.476	0.944	1.000	1.000	1.000
X_a:X_c	0.854	0.994	1.000	1.000	1.000
X_a:X_f	0.778	0.992	1.000	1.000	1.000
X_c:X_f	0.192	0.572	0.840	1.000	0.968
X_a:X_c:X_f	0.536	0.830	0.974	1.000	0.996

3.3 Summary Statistics for Power Calculation with 20 pct. Missing Data and Varying Intercepts:

Table 3: Power for Simulations with 20 pct. Missing Data and Varying Intercepts and Slopes

term	power, ef = 0.1	power, ef = 0.2	power, ef = 0.3	power, ef = 0.4	power, ef = 0.5
(Intercept)	0.056	0.050	0.062	0.038	0.046
X_a	0.890	1.000	1.000	1.000	1.000
X_c	0.660	0.988	1.000	1.000	1.000
X_f	0.550	0.940	1.000	1.000	1.000
X_a:X_c	0.818	0.994	1.000	1.000	1.000
X_a:X_f	0.768	0.984	1.000	1.000	1.000
X_c:X_f	0.266	0.534	0.852	0.974	1.000
X_a:X_c:X_f	0.486	0.816	0.954	0.998	1.000

3.4 Summary Statistics for Power Calculation with 50 pct. Missing Data and Varying Intercepts:

Table 4: Power for Simulations with 50 pct. Missing Data and Varying Intercepts and Slopes

term	power, ef = 0.1	power, ef = 0.2	power, ef = 0.3	power, ef = 0.4	power, ef = 0.5
(Intercept)	0.044	0.036	0.056	0.042	0.046
X_a	0.858	0.998	1.000	1.000	1.000
X_c	0.630	0.990	1.000	1.000	1.000
X_f	0.528	0.944	1.000	1.000	1.000
X_a:X_c	0.756	0.990	1.000	1.000	1.000
X_a:X_f	0.666	0.976	1.000	1.000	1.000
X_c:X_f	0.170	0.480	0.840	0.972	0.998
X_a:X_c:X_f	0.370	0.696	0.914	0.994	0.998

4 Overview of Bias Results

Table 5: Bias for Simulations with Full Data and Varying Intercepts and Varying Slopes

term	bias, ef = 0.1	bias, ef = 0.2	bias, ef = 0.3	bias, ef = 0.4	bias, ef = 0.5
(Intercept)	0.001	-0.004	0.001	-0.003	-0.003
X_a	-0.001	-0.005	0.001	-0.001	0.001
X_c	0.000	-0.003	0.001	-0.003	0.000
X_f	0.003	-0.003	0.000	-0.003	-0.001
X_a:X_c	-0.009	0.002	0.002	-0.001	0.000
X_a:X_f	-0.001	-0.003	-0.005	0.002	0.000
X_c:X_f	-0.004	-0.011	0.009	0.001	-0.001
X_a:X_c:X_f	0.012	-0.009	-0.005	-0.005	-0.008

Table 6: Bias for Simulations with Full Data and Varying Intercepts

term	bias, ef = 0.1	bias, ef = 0.2	bias, ef = 0.3	bias, ef = 0.4	bias, ef = 0.5
(Intercept)	-0.001	-0.002	0.002	0.000	-0.003
X_a	0.000	0.002	0.003	-0.100	0.100
X_c	0.000	0.001	-0.002	-0.104	0.098
X_f	0.007	0.004	-0.002	-0.100	0.096
X_a:X_c	0.000	-0.002	-0.002	-0.094	0.092
X_a:X_f	-0.002	0.009	0.002	-0.100	0.098
X_c:X_f	0.003	-0.010	0.005	-0.096	0.103
X_a:X_c:X_f	0.001	-0.001	0.000	-0.103	0.092

Table 7: Bias for Simulations with 20 pct. Missing Data and Varying Intercepts and Slopes

term	bias, ef = 0.1	bias, ef = 0.2	bias, ef = 0.3	bias, ef = 0.4	bias, ef = 0.5
(Intercept)	-0.001	0.000	0.001	0.002	0.001
X_a	-0.001	-0.002	-0.002	0.001	-0.004
X_c	-0.001	0.000	0.000	-0.002	0.001
X_f	-0.001	0.006	-0.003	0.001	-0.001
X_a:X_c	0.000	0.005	-0.003	-0.008	-0.001
X_a:X_f	0.005	-0.003	0.000	0.003	0.003
X_c:X_f	-0.007	-0.005	-0.001	0.011	-0.004
X_a:X_c:X_f	-0.002	0.001	-0.003	-0.002	0.003

Table 8: Bias for Simulations with 50 pct. Missing Data and Varying Intercepts and Slopes

term	bias, ef = 0.1	bias, ef = 0.2	bias, ef = 0.3	bias, ef = 0.4	bias, ef = 0.5
(Intercept)	-0.003	-0.002	-0.002	-0.004	0.002
X_a	0.006	0.001	0.000	-0.004	0.002
X_c	0.003	-0.008	0.005	-0.002	0.004
X_f	0.000	0.003	-0.004	-0.003	0.003
X_a:X_c	0.000	0.002	-0.002	0.001	0.001
X_a:X_f	0.003	-0.002	-0.001	0.001	-0.001
X_c:X_f	0.007	0.004	-0.002	-0.009	-0.001
X_a:X_c:X_f	0.020	0.001	0.002	-0.003	-0.001

5 Grid Search Code

```
n_subjects <- c(1280, 1920, 2560)
n_trials <- c(12, 18, 24)
b_parameter <- c(0.3, 0.5, 0.7)
sd_parameter <- c(0.1, 0.2, 0.3)
```

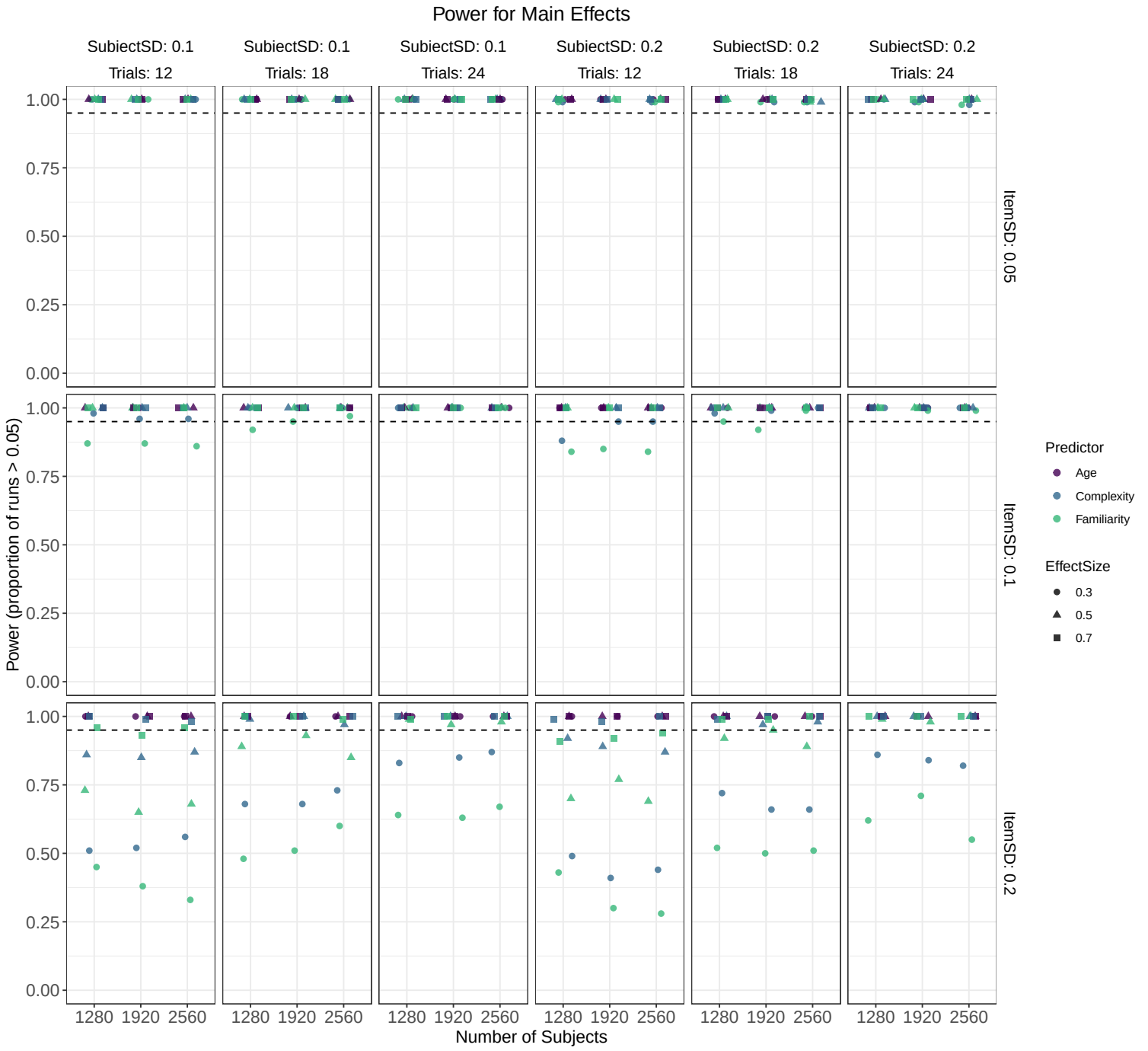
```r
sigma_parameter <- c(0.1)


run_sims_grid_point  <- function(filename_full, subj_n, trial_n, ef, residual_sd, random_var, random_var_item, rh

  participants_per_lab<-32

  n_simple<- trial_n/2
  n_complex<- trial_n/2
  n_small_fam<- trial_n/3
  n_medium_fam<- trial_n/3
  n_high_fam<- trial_n/3
  n_lab<-floor(subj_n/participants_per_lab)

  dat_sim <- my_sim_data(n_subj = subj_n,
                         n_simple = n_simple,
                         n_complex = n_complex,
                         n_small_fam = n_small_fam,
                         n_medium_fam = n_medium_fam,
                         n_high_fam = n_high_fam,
                         n_lab = n_lab,
                         beta_c  =  ef,
                         beta_f = ef,
                         beta_a = ef,

                         beta_ca = ef,
                         beta_af = ef,
                         beta_cf = ef,
                         beta_cfa = ef,

                         subject_0 = random_var,
                         subject_c = random_var,
                         subject_f = random_var,
                         subject_a = random_var,
                         subject_ca = random_var,
                         subject_af = random_var,
                         subject_cf = random_var,
                         subject_cfa = random_var,
                         subj_rho = rho_val,

                         lab_0 = random_var,
                         lab_c = random_var,
                         lab_f = random_var,
                         lab_a = random_var,
                         lab_ca = random_var,
                         lab_af = random_var,
                         lab_cf = random_var,
                         lab_cfa = random_var,
                         lab_rho = rho_val,

                         item_0 = random_var_item,
                         item_c = random_var_item,
                         item_f = random_var_item,
                         item_a = random_var_item,
                         item_ca = random_var_item,
                         item_af = random_var_item,
                         item_cf = random_var_item,
                         item_cfa = random_var_item,
                         item_rho = rho_val,

                         sigma = residual_sd,
                         )
```

```
  mod_sim <- lmer(DV ~ 1 + X_a * X_c * X_f +
                    (1 | subj_id) +
                    (1 | lab_id) +
                    (1 | item_id),
                  data=dat_sim)

  sim_results <- broom.mixed::tidy(mod_sim)

  # append the results to a file
  append <- file.exists(filename_full)
  write_csv(sim_results, filename_full, append = append)

  # return the tidy table
  sim_results
}

reps <- 50

n_subj_values <- c(1280, 1920, 2560)
trial_n_values <- c(12, 18, 24)
random_var <- c(0.1, 0.2)
random_var_item <- c(0.05, 0.1, 0.2)
b_parameter <- c(0.3, 0.5, 0.7)
residual_sd <- c(0.1, 0.3)

param_combinations <- expand.grid(n_subj = n_subj_values,
                                  trial_n = trial_n_values,
                                  random_var = random_var,
                                  random_var_item = random_var_item,
                                  b_parameter = b_parameter,
                                 residual_sd = residual_sd)

for (i in seq_len(nrow(param_combinations))) {
  start_time <- Sys.time()

  sim_params <- param_combinations[i, ]
  filename_full <- paste0('sims_grid_search/test_grid_search_',
                          sim_params$n_subj, '_',
                          sim_params$trial_n, '_',
                          sim_params$random_var, '_',
                          sim_params$random_var_item, '_',
                          sim_params$b_parameter, '_',
                          sim_params$residual_sd, '.csv')

  sims <- purrr::map_df(1:reps, ~run_sims_grid_point(filename_full = filename_full,
                                                     subj_n = sim_params$n_subj,
                                                     trial_n = sim_params$trial_n,
                                                     ef = sim_params$b_parameter,
                                                     random_var = sim_params$random_var,
                                                     random_var_item = sim_params$random_var_item,
                                                     residual_sd = sim_params$residual_sd,
                                                     rho_val = 0.2))

  end_time <- Sys.time()
  cat("Simulation", i, "Time elapsed:", end_time - start_time, "\n")
}
```

```
# set the directory where the CSV files are located
setwd("/work/sims_grid_search")

# get the file names for CSV files
```

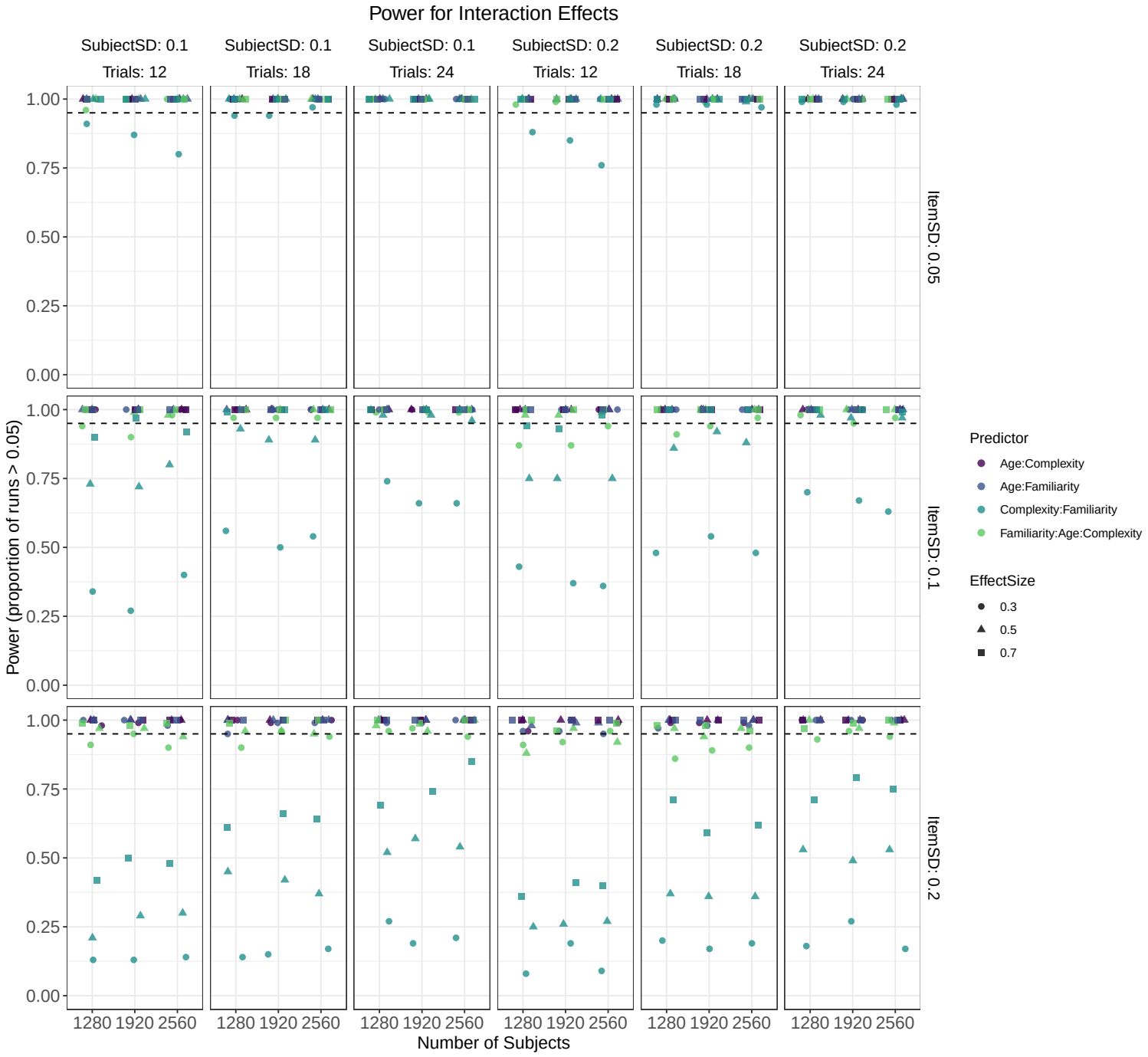
```
file_names <- list.files(pattern = "*.csv")

# read in all CSV files into a list of dataframes
df_list <- purrr::map(file_names, ~{
    df <- read.csv(.x)
    df$filename <- .x
    df
})

df <- purrr::reduce(df_list, dplyr::bind_rows)

df_per_sim <- df %>%
    filter(effect == "fixed") %>%
    group_by(filename, term) %>%
    summarise(median_estimate = median(estimate), median_se = median(std.error),
        power = mean(p.value < 0.05)) %>%
    mutate(n_subj = sapply(strsplit(filename, "_"), `[`, 4),
        Trials = sapply(strsplit(filename, "_"), `[`, 5), SubjectSD = sapply(strsplit(filename,
            "_"), `[`, 6), ItemSD = sapply(strsplit(filename,
            "_"), `[`, 7), EffectSize = sapply(strsplit(filename,
            "_"), `[`, 8), residual_sd = as.numeric(str_replace(sapply(strsplit(filename,
            "_"), `[`, 9), pattern = ".csv", "")))

error_bars <- df_per_sim %>%
    filter(term != "(Intercept)") %>%
    group_by(EffectSize, SubjectSD, Trials, ItemSD, n_subj, term) %>%
    dplyr::summarise(power_mean = median(power), power_lci = quantile(power,
        prob = 0.025), power_hci = quantile(power, prob = 0.975)) %>%
    filter(term %in% c("X_a", "X_c", "X_f")) %>%
    dplyr::rename(Predictor = term) %>%
    mutate(Predictor = ifelse(Predictor == "X_a", "Age", Predictor)) %>%
    mutate(Predictor = ifelse(Predictor == "X_c", "Complexity",
        Predictor)) %>%
    mutate(Predictor = ifelse(Predictor == "X_f", "Familiarity",
        Predictor))

MainEffectGridSearchPlot <- ggplot() + geom_point(aes(x = n_subj,
    y = power_mean, color = Predictor, shape = EffectSize), data = error_bars,
    size = 2, alpha = 0.8, position = position_jitter(width = 0.2,
        height = 0)) + geom_hline(yintercept = 0.95, linetype = 2) +
    scale_color_manual(values = viridis(n = 4)) + #scale_fill_manual(values = brewer.pal(n=8, name = 'Dark2')) +
    scale_color_manual(values = viridis(n = 4)) + #scale_fill_manual(values = brewer.pal(n=8, name = 'Dark2')) +
    scale_color_manual(values = viridis(n = 4)) + #scale_fill_manual(values = brewer.pal(n=8, name = 'Dark2')) +
    scale_color_manual(values = viridis(n = 4)) + #scale_fill_manual(values = brewer.pal(n=8, name = 'Dark2')) +
    scale_color_manual(values = viridis(n = 4)) + #scale_fill_manual(values = brewer.pal(n=8, name = 'Dark2')) +
    scale_color_manual(values = viridis(n = 4)) + #scale_fill_manual(values = brewer.pal(n=8, name = 'Dark2')) +
    scale_color_manual(values = viridis(n = 4)) + #scale_fill_manual(values = brewer.pal(n=8, name = 'Dark2')) +
    scale_color_manual(values = viridis(n = 4)) + #scale_fill_manual(values = brewer.pal(n=8, name = 'Dark2')) +
    scale_color_manual(values = viridis(n = 4)) + #scale_fill_manual(values = brewer.pal(n=8, name = 'Dark2')) +
    scale_color_manual(values = viridis(n = 4)) + #scale_fill_manual(values = brewer.pal(n=8, name = 'Dark2')) +
    scale_color_manual(values = viridis(n = 4)) + #scale_fill_manual(values = brewer.pal(n=8, name = 'Dark2')) +
    scale_color_manual(values = viridis(n = 4)) + #scale_fill_manual(values = brewer.pal(n=8, name = 'Dark2')) +
    scale_color_manual(values = viridis(n = 4)) + #scale_fill_manual(values = brewer.pal(n=8, name = 'Dark2')) +
scale_y_continuous(breaks = c(0, 0.25, 0.5, 0.75, 1), limits = c(0,
    1)) + facet_grid(ItemSD ~ SubjectSD + Trials, labeller = label_both) +
    ggtitle("Power for Main Effects") + xlab("Number of Subjects") +
    ylab("Power (proportion of runs > 0.05)") + theme_bw() +
    theme(strip.background = element_rect(color = "white", fill = "white",
        size = 5, linetype = "solid"), plot.title = element_text(hjust = 0.5,
        size = 15), strip.text.x = element_text(size = 12, color = "black"),
        strip.text.y = element_text(size = 12, color = "black"),
        axis.text.x = element_text(size = 13), axis.title.x = element_text(size = 13),
```

```
        axis.text.y = element_text(size = 13), axis.title.y = element_text(size = 13))
MainEffectGridSearchPlot
```




```
error_bars <- df_per_sim %>%
    filter(term != "(Intercept)") %>%
    group_by(EffectSize, SubjectSD, Trials, ItemSD, n_subj, term) %>%
    dplyr::summarise(power_mean = median(power), power_lci = quantile(power,
        prob = 0.025), power_hci = quantile(power, prob = 0.975)) %>%
    filter(term %in% c("X_a:X_c", "X_a:X_f", "X_c:X_f", "X_a:X_c:X_f")) %>%
    dplyr::rename(Predictor = term) %>%
    mutate(Predictor = ifelse(Predictor == "X_a:X_c", "Age:Complexity",
        Predictor)) %>%
    mutate(Predictor = ifelse(Predictor == "X_a:X_f", "Age:Familiarity",
        Predictor)) %>%
    mutate(Predictor = ifelse(Predictor == "X_c:X_f", "Complexity:Familiarity",
        Predictor)) %>%
    mutate(Predictor = ifelse(Predictor == "X_a:X_c:X_f", "Familiarity:Age:Complexity",
```

```
        Predictor))

InteractionEffectGridSearchPlot <- ggplot() + geom_point(aes(x = n_subj,
    y = power_mean, color = Predictor, shape = EffectSize), data = error_bars,
    size = 2, alpha = 0.8, position = position_jitter(width = 0.25,
        height = 0)) + geom_hline(yintercept = 0.95, linetype = 2) +
    scale_color_manual(values = viridis(n = 5)) + #scale_fill_manual(values = brewer.pal(n=8, name = 'Dark2')) +
    scale_color_manual(values = viridis(n = 5)) + #scale_fill_manual(values = brewer.pal(n=8, name = 'Dark2')) +
    scale_color_manual(values = viridis(n = 5)) + #scale_fill_manual(values = brewer.pal(n=8, name = 'Dark2')) +
    scale_color_manual(values = viridis(n = 5)) + #scale_fill_manual(values = brewer.pal(n=8, name = 'Dark2')) +
    scale_color_manual(values = viridis(n = 5)) + #scale_fill_manual(values = brewer.pal(n=8, name = 'Dark2')) +
    scale_color_manual(values = viridis(n = 5)) + #scale_fill_manual(values = brewer.pal(n=8, name = 'Dark2')) +
    scale_color_manual(values = viridis(n = 5)) + #scale_fill_manual(values = brewer.pal(n=8, name = 'Dark2')) +
    scale_color_manual(values = viridis(n = 5)) + #scale_fill_manual(values = brewer.pal(n=8, name = 'Dark2')) +
    scale_color_manual(values = viridis(n = 5)) + #scale_fill_manual(values = brewer.pal(n=8, name = 'Dark2')) +
    scale_color_manual(values = viridis(n = 5)) + #scale_fill_manual(values = brewer.pal(n=8, name = 'Dark2')) +
    scale_color_manual(values = viridis(n = 5)) + #scale_fill_manual(values = brewer.pal(n=8, name = 'Dark2')) +
    scale_color_manual(values = viridis(n = 5)) + #scale_fill_manual(values = brewer.pal(n=8, name = 'Dark2')) +
    scale_color_manual(values = viridis(n = 5)) + #scale_fill_manual(values = brewer.pal(n=8, name = 'Dark2')) +
scale_y_continuous(breaks = c(0, 0.25, 0.5, 0.75, 1), limits = c(0,
    1)) + facet_grid(ItemSD ~ SubjectSD + Trials, labeller = label_both) +
    ggtitle("Power for Interaction Effects") + xlab("Number of Subjects") +
    ylab("Power (proportion of runs > 0.05)") + theme_bw() +
    theme(strip.background = element_rect(color = "white", fill = "white",
        size = 5, linetype = "solid"), plot.title = element_text(hjust = 0.5,
        size = 15), strip.text.x = element_text(size = 12, color = "black"),
        strip.text.y = element_text(size = 12, color = "black"),
        axis.text.x = element_text(size = 13), axis.title.x = element_text(size = 13),
        axis.text.y = element_text(size = 13), axis.title.y = element_text(size = 13))

InteractionEffectGridSearchPlot
```




```
ggsave(plot = MainEffectGridSearchPlot, file = "MainEffectGridSearchPlot.png",
    height = 10, width = 15)
ggsave(plot = InteractionEffectGridSearchPlot, file = "InteractionEffectGridSearchPlot.png",
    height = 10, width = 15)
```

6 Power Calculation with Full Data and Varying Intercepts and Varying Slopes

6.1 Effect Size = 0.5

```
# Number of simulations:
reps <- 500

# Simulation function:
run_sims <- function(filename_full, ef) {
```

```
  dat_sim <- my_sim_data(beta_c  =  ef,
                         beta_f = ef,
                         beta_a = ef,

                         beta_ca = ef,
                         beta_af = ef,
                         beta_cf = ef,

                         beta_cfa = ef)

  mod_sim <- lmer(DV ~ 1 + X_a * X_c * X_f +
                    (1 + X_c * X_f | subj_id) +
                    (1 | lab_id) +
                    (1 | item_id),
                  data=dat_sim)

  sim_results <- broom.mixed::tidy(mod_sim)

  # append the results to a file
  append <- file.exists(filename_full)
  write_csv(sim_results, filename_full, append = append)

  # return the tidy table
  sim_results
}

filename_full_0.5 = 'run_sims_full_0.5.csv'
start_time <- Sys.time()
sims <- purrr::map_df(1:reps, ~run_sims(filename_full = filename_full_0.5, ef = 0.5))
end_time <- Sys.time()
end_time - start_time
```

6.2 Effect Size = 0.4

```
filename_full_0.4 = "run_sims_full_0.4.csv"
start_time <- Sys.time()
sims <- purrr::map_df(1:reps, ~run_sims(filename_full = filename_full_0.4,
    ef = 0.4))
end_time <- Sys.time()
end_time - start_time
```

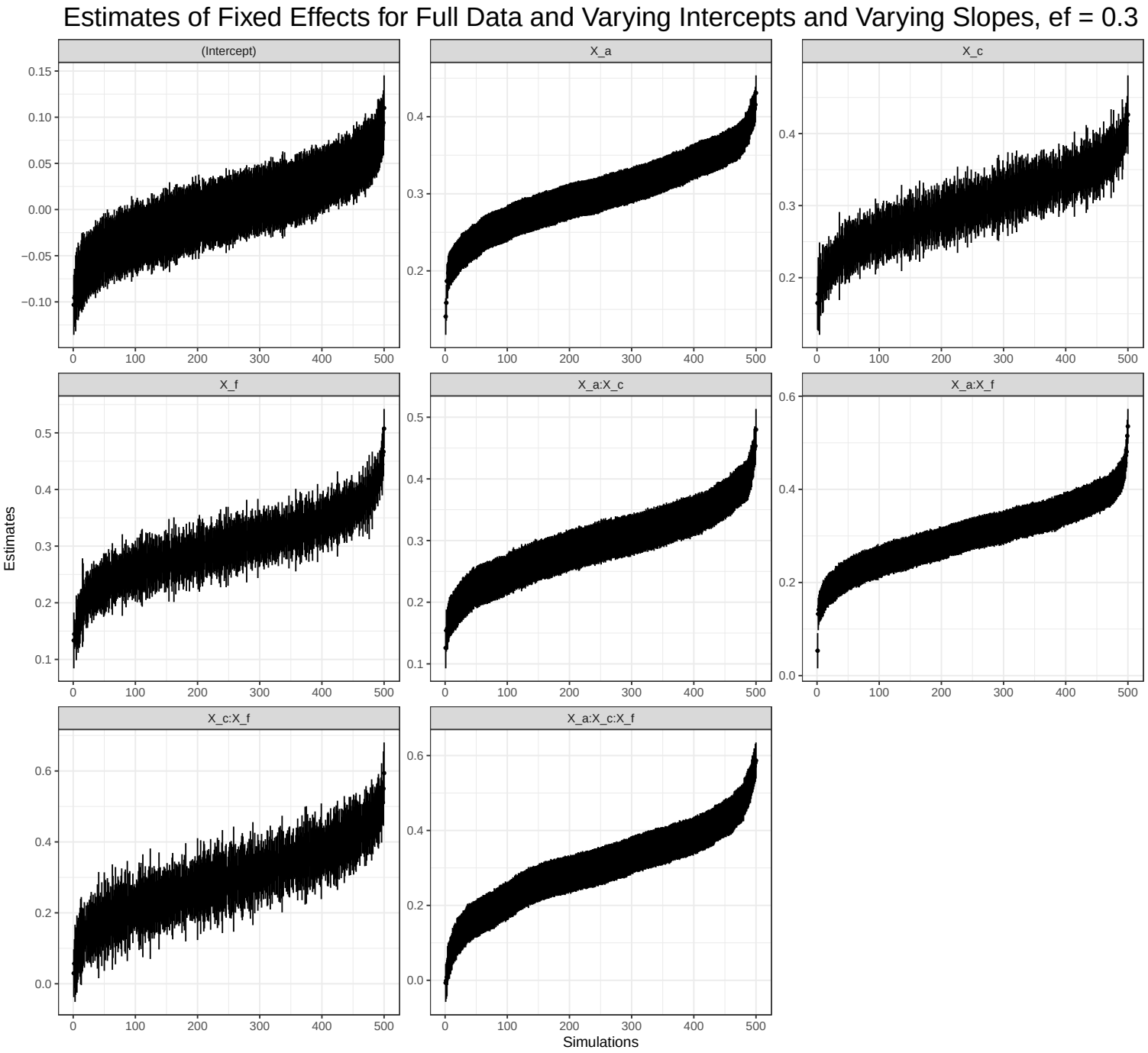
6.3 Effect Size = 0.3

```
filename_full_0.3 = "run_sims_full_0.3.csv"
start_time <- Sys.time()
sims <- purrr::map_df(1:reps, ~run_sims(filename_full = filename_full_0.3,
    ef = 0.3))
end_time <- Sys.time()
end_time - start_time
```

6.3.1 Visualise Estimates for Fixed Effects:

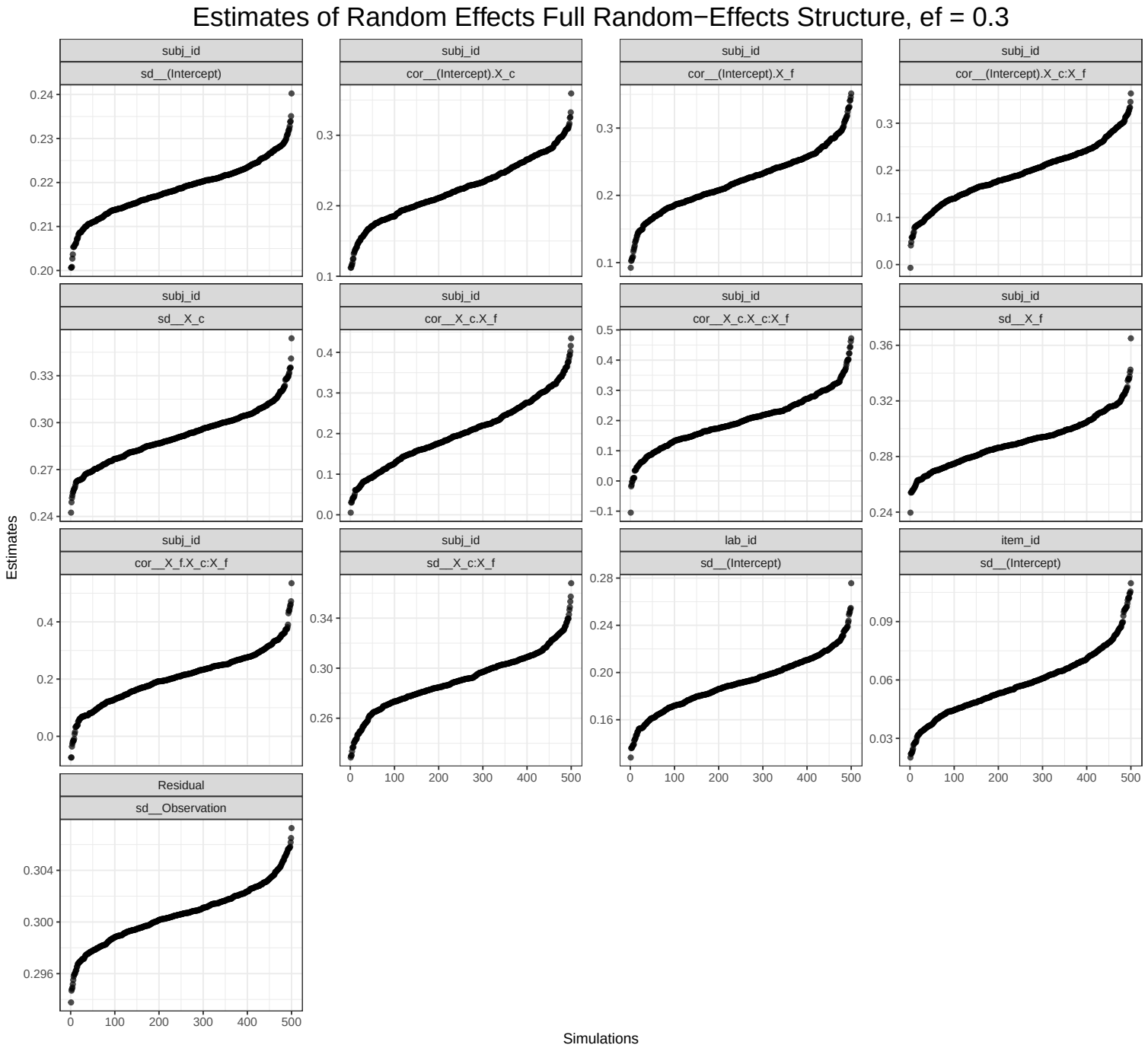
```
sims_full_0.3 <- read_csv(filename_full_0.3, col_types = cols(group = col_factor(ordered = TRUE),
    term = col_factor(ordered = TRUE)))
```

```
fixed_full_plot <- sims_full_0.3 %>%
    filter(effect == "fixed") %>%
    ungroup() %>%
    arrange(term, estimate) %>%
    mutate(row = rep(seq(1:reps), 8)) %>%
    ggplot(aes(x = row, y = estimate, ymin = estimate - std.error,
        ymax = estimate + std.error)) + facet_wrap(~term, scales = "free") +
    geom_pointrange(fatten = 1/2) + ylab("Estimates") + xlab("Simulations") +
    ggtitle("Estimates of Fixed Effects for Full Data and Varying Intercepts and Varying Slopes, ef = 0.3") +
    theme_bw()

fixed_full_plot <- fixed_full_plot + theme(plot.title = element_text(hjust = 0.5,
    size = 20))
fixed_full_plot
```



6.3.2 Visualise Estimates for Random Effects:

```
ran_full_plot <- sims_full_0.3 %>%
    filter(effect == "ran_pars") %>%
    ungroup() %>%
    arrange(group, term, estimate) %>%
    mutate(row = rep(seq(1:reps), 13)) %>%
    ggplot(aes(x = row, y = estimate)) + geom_point(alpha = 0.7) +
    facet_wrap(~group + term, scales = "free_y") + theme_bw() +
    ylab("Estimates") + xlab("Simulations") + ggtitle("Estimates of Random Effects Full Random-Effects Structure,
    theme_bw()
ran_full_plot <- ran_full_plot + theme(plot.title = element_text(hjust = 0.5,
    size = 20))
ran_full_plot
```



6.4 Effect Size = 0.2

```
filename_full_0.2 = "run_sims_full_0.2.csv"
start_time <- Sys.time()
sims <- purrr::map_df(1:reps, ~run_sims(filename_full = filename_full_0.2,
    ef = 0.2))
end_time <- Sys.time()
end_time - start_time
```

6.5 Effect Size = 0.1

```
filename_full_0.1 = "run_sims_full_0.1.csv"
start_time <- Sys.time()
sims <- purrr::map_df(1:reps, ~run_sims(filename_full = filename_full_0.1,
    ef = 0.1))
end_time <- Sys.time()
end_time - start_time
```

7 Power Calculation with Full Data and Varying Intercepts

7.1 Effect Size = 0.5

```
# Simulation function:
run_sims <- function(filename_full, ef) {

  dat_sim <- my_sim_data(beta_c  =  ef,
                         beta_f = ef,
                         beta_a = ef,

                         beta_ca = ef,
                         beta_af = ef,
                         beta_cf = ef,

                         beta_cfa = ef)

  mod_sim <- lmer(DV ~ 1 + X_a * X_c * X_f +
                    (1 | subj_id) +
                    (1 | lab_id) +
                    (1 | item_id),
                  data=dat_sim)

  sim_results <- broom.mixed::tidy(mod_sim)

  # append the results to a file
  append <- file.exists(filename_full)
  write_csv(sim_results, filename_full, append = append)

  # return the tidy table
  sim_results
}

filename_full_int_0.5 = 'run_sims_full_int_0.5.csv'
start_time <- Sys.time()
sims <- purrr::map_df(1:reps, ~run_sims(filename_full = filename_full_int_0.5, ef = 0.5))
end_time <- Sys.time()
end_time - start_time
```

7.2 Effect Size = 0.4

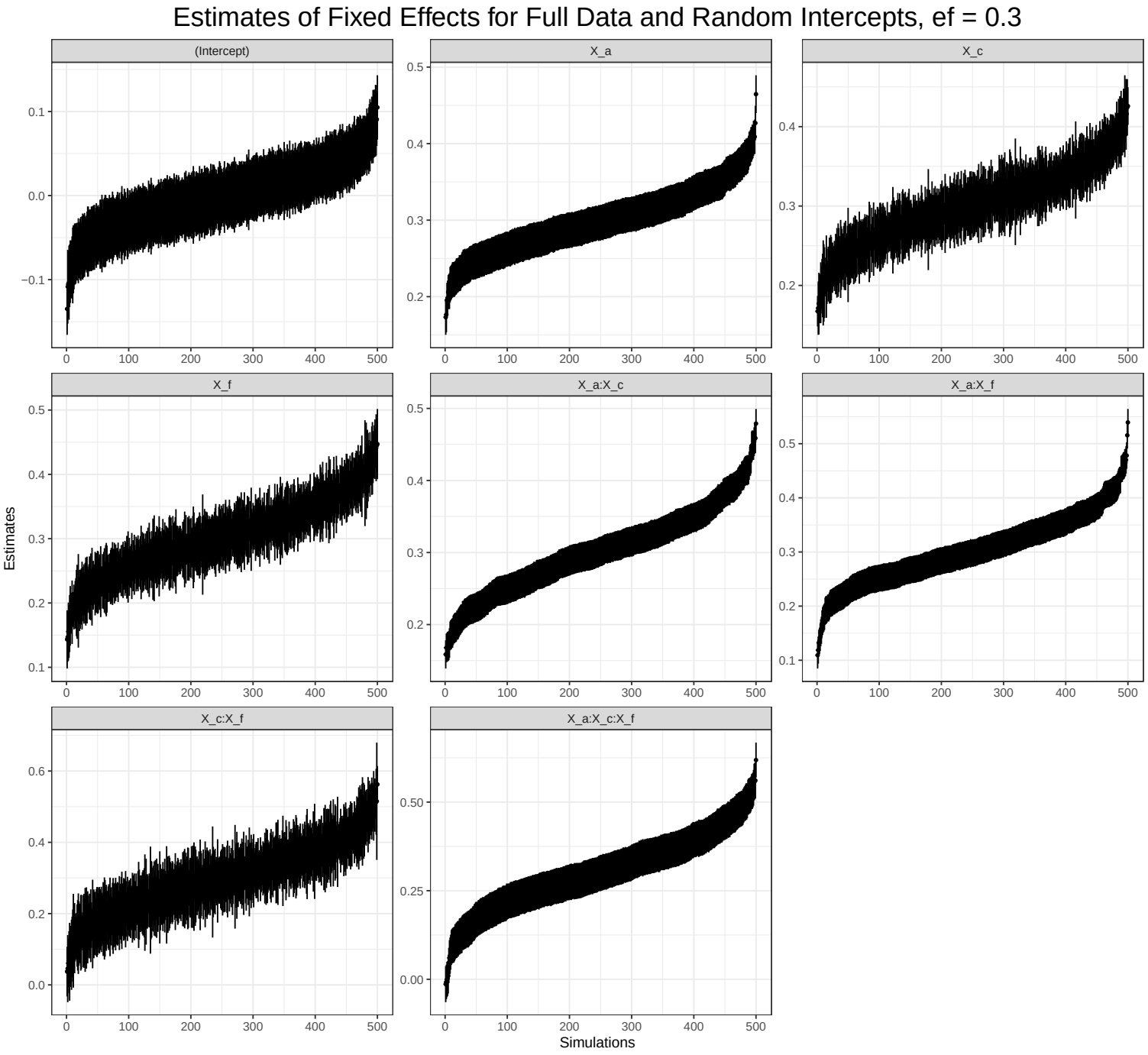
```
filename_full_int_0.4 = "run_sims_full_int_0.4.csv"
start_time <- Sys.time()
sims <- purrr::map_df(1:reps, ~run_sims(filename_full = filename_full_int_0.4,
    ef = 0.4))
end_time <- Sys.time()
end_time - start_time
```

7.3 Effect Size = 0.3

```
filename_full_int_0.3 = "run_sims_full_int_0.3.csv"
start_time <- Sys.time()
sims <- purrr::map_df(1:reps, ~run_sims(filename_full = filename_full_int_0.3,
    ef = 0.3))
end_time <- Sys.time()
end_time - start_time
```

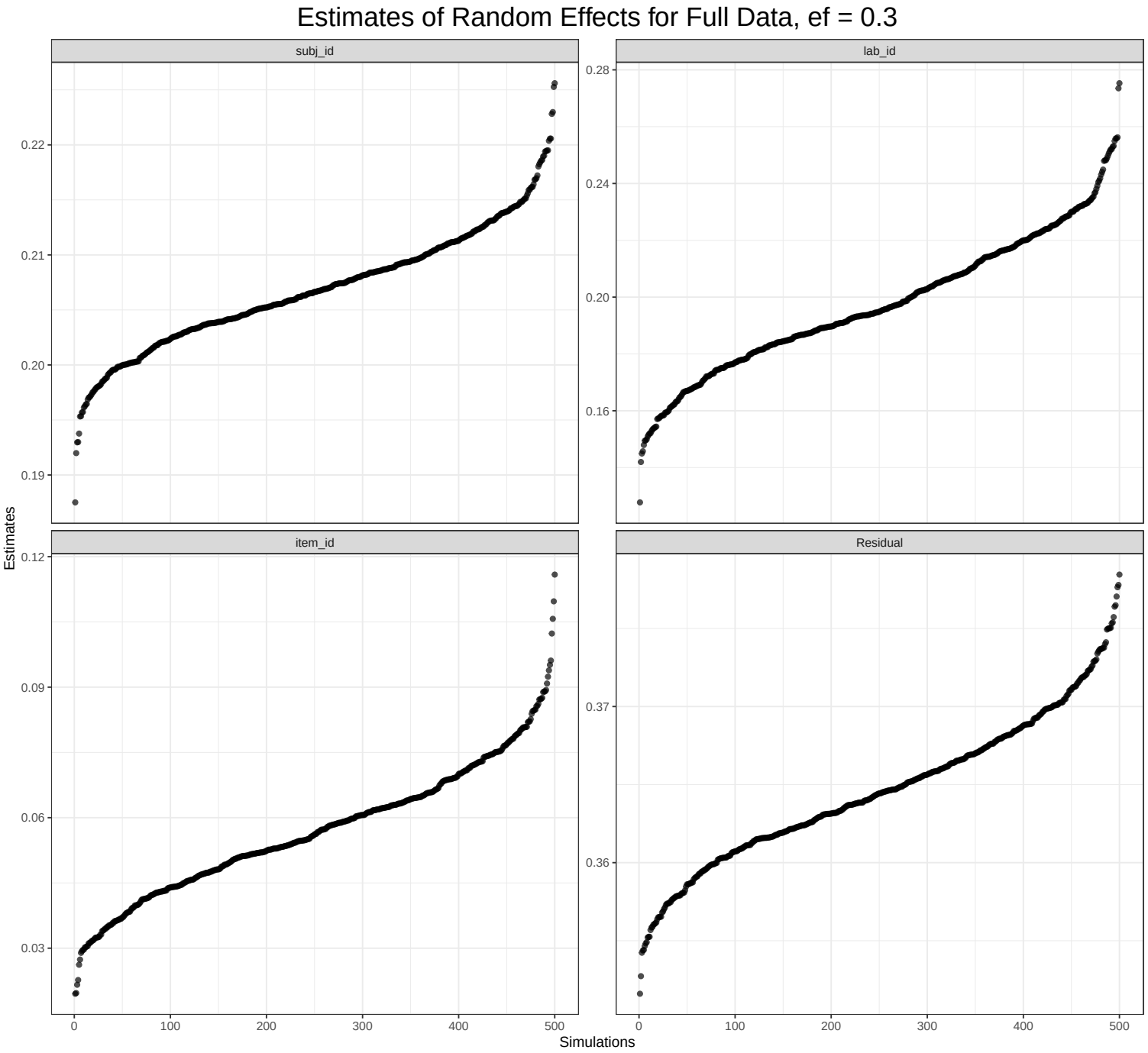
7.3.1 Visualise Estimates for Fixed Effects:

```
sims_full_int_0.3 <- read_csv(filename_full_int_0.3, col_types = cols(group = col_factor(ordered = TRUE),
    term = col_factor(ordered = TRUE)))

fixed_full_int_plot <- sims_full_int_0.3 %>%
    filter(effect == "fixed") %>%
    ungroup() %>%
    arrange(term, estimate) %>%
    mutate(row = rep(seq(1:reps), 8)) %>%
    ggplot(aes(x = row, y = estimate, ymin = estimate - std.error,
        ymax = estimate + std.error)) + facet_wrap(~term, scales = "free") +
    geom_pointrange(fatten = 1/2) + ylab("Estimates") + xlab("Simulations") +
    ggtitle("Estimates of Fixed Effects for Full Data and Random Intercepts, ef = 0.3") +
    theme_bw()

fixed_full_int_plot <- fixed_full_int_plot + theme(plot.title = element_text(hjust = 0.5,
    size = 20))
fixed_full_int_plot
```



7.3.2 Visualise Estimates for Random Effects:

```
ran_full_int_plot <- sims_full_int_0.3 %>%
    filter(effect == "ran_pars") %>%
    ungroup() %>%
    arrange(group, estimate) %>%
    mutate(row = rep(seq(1:reps), 4)) %>%
    ggplot(aes(x = row, y = estimate)) + geom_point(alpha = 0.7) +
    facet_wrap(~group, scales = "free_y") + theme_bw() + ylab("Estimates") +
    xlab("Simulations") + ggtitle("Estimates of Random Effects for Full Data, ef = 0.3") +
    theme_bw()
ran_full_int_plot <- ran_full_int_plot + theme(plot.title = element_text(hjust = 0.5,
    size = 20))
ran_full_int_plot
```



7.4 Effect Size = 0.2

```
filename_full_int_0.2 = "run_sims_full_int_0.2.csv"
start_time <- Sys.time()
sims <- purrr::map_df(1:reps, ~run_sims(filename_full = filename_full_int_0.2,
    ef = 0.2))
end_time <- Sys.time()
end_time - start_time
```

7.5 Effect Size = 0.1

```
filename_full_int_0.1 = "run_sims_full_int_0.1.csv"
start_time <- Sys.time()
sims <- purrr::map_df(1:reps, ~run_sims(filename_full = filename_full_int_0.1,
```

```
    ef = 0.1))
end_time <- Sys.time()
end_time - start_time
```

8 Power Calculation with 20 pct. Missing Data and Varying Intercepts

8.1 Effect Size = 0.5

```
run_sims_missing <- function(filename_missing, ef) {

  dat_sim <- my_sim_data(beta_c  =  ef,
                         beta_f = ef,
                         beta_a = ef,

                         beta_ca = ef,
                         beta_af = ef,
                         beta_cf = ef,

                         beta_cfa = ef)

  missing_samples <- dat_sim %>%
    mutate(nas = rbinom(nrow(dat_sim), 1, 1 - .20)) %>%
    mutate(DV = ifelse(nas == 1, DV, NA)) %>%
    drop_na()

  mod_sim <- lmer(DV ~ 1 + X_a * X_c * X_f +
                    (1 | subj_id) +
                    (1 | lab_id) +
                    (1 | item_id),
                  data=missing_samples)

  sim_results <- broom.mixed::tidy(mod_sim)

  # append the results to a file
  append <- file.exists(filename_missing)
  write_csv(sim_results, filename_missing, append = append)

  # return the tidy table
  sim_results
}

filename_20_missing_0.5 = 'run_sims_20_missing_0.5.csv'
start_time <- Sys.time()
sims_missing <- purrr::map_df(1:reps, ~run_sims_missing(filename_missing = filename_20_missing_0.5, ef = 0.5))
end_time <- Sys.time()
end_time - start_time
```

8.2 Effect Size = 0.4

```
filename_20_missing_0.4 = "run_sims_20_missing_0.4.csv"
start_time <- Sys.time()
sims_missing <- purrr::map_df(1:reps, ~run_sims_missing(filename_missing = filename_20_missing_0.4,
    ef = 0.4))
end_time <- Sys.time()
end_time - start_time
```

8.3 Effect Size = 0.3

```
filename_20_missing_0.3 = "run_sims_20_missing_0.3.csv"
start_time <- Sys.time()
sims_missing <- purrr::map_df(1:reps, ~run_sims_missing(filename_missing = filename_20_missing_0.3,
    ef = 0.3))
end_time <- Sys.time()
end_time - start_time
```

8.3.1 Visualise Estimates for Fixed Effects:

```
# read saved simulation data
sims_20_missing_0.3 <- read_csv(filename_20_missing_0.3, col_types = cols(
  # makes sure plots display in this order
  group = col_factor(ordered = TRUE),
  term = col_factor(ordered = TRUE)
))

fixed_missing_plot <- sims_20_missing_0.3 %>%
  filter(effect == "fixed") %>%
  ungroup() %>%
  arrange(term, estimate) %>%
  mutate(row = rep(seq(1:reps), 8)) %>%
  ggplot(aes(x = row, y = estimate, ymin = estimate-std.error, ymax = estimate+std.error)) +
  facet_wrap(~term, scales = "free") +
  geom_pointrange(fatten = 1/2) +
  ylab("Estimates") +
  xlab("Simulations") +
  ggtitle('Estimates of Fixed Effects for 20 pct. Missing Data, ef = 0.3') +
  theme_bw()

fixed_missing_plot <- fixed_missing_plot + theme(plot.title = element_text(hjust = 0.5, size=20))
fixed_missing_plot
```

8.3.2 Visualise Estimates for Random Effects:

```
ran_missing_plot <- sims_20_missing_0.3 %>%
    filter(effect == "ran_pars") %>%
    ungroup() %>%
    arrange(group, term, estimate) %>%
    mutate(row = rep(seq(1:reps), 13)) %>%
    ggplot(aes(x = row, y = estimate)) + geom_point(alpha = 0.7) +
    facet_wrap(~group + term, scales = "free_y") + theme_bw() +
    ylab("Estimates") + xlab("Simulations") + ggtitle("Estimates of Random Effects for 20 pct. Missing Data, ef =
    theme_bw()
ran_missing_plot <- ran_missing_plot + theme(plot.title = element_text(hjust = 0.5,
    size = 20))
ran_missing_plot
```

8.4 Effect Size = 0.2

```
filename_20_missing_0.2 = "run_sims_20_missing_0.2.csv"
start_time <- Sys.time()
sims_missing <- purrr::map_df(1:reps, ~run_sims_missing(filename_missing = filename_20_missing_0.2,
    ef = 0.2))
```

```
end_time <- Sys.time()
end_time - start_time
```

8.5 Effect Size = 0.1

```
filename_20_missing_0.1 = "run_sims_20_missing_0.1.csv"
start_time <- Sys.time()
sims_missing <- purrr::map_df(1:reps, ~run_sims_missing(filename_missing = filename_20_missing_0.1,
    ef = 0.1))
end_time <- Sys.time()
end_time - start_time
```

9 Power Calculation with 50 pct. Missing Data and Varying Intercepts

9.1 Effect Size = 0.5

```
run_sims_missing <- function(filename_missing, ef) {

  dat_sim <- my_sim_data(beta_c  =  ef,
                         beta_f = ef,
                         beta_a = ef,

                         beta_ca = ef,
                         beta_af = ef,
                         beta_cf = ef,

                         beta_cfa = ef)

  missing_samples <- dat_sim %>%
    mutate(nas = rbinom(nrow(dat_sim), 1, 1 - .50)) %>%
    mutate(DV = ifelse(nas == 1, DV, NA)) %>%
    drop_na()

  mod_sim <- lmer(DV ~ 1 + X_a * X_c * X_f +
                    (1 | subj_id) +
                    (1 | lab_id) +
                    (1 | item_id),
                  data=missing_samples)

  sim_results <- broom.mixed::tidy(mod_sim)

  # append the results to a file
  append <- file.exists(filename_missing)
  write_csv(sim_results, filename_missing, append = append)

  # return the tidy table
  sim_results
}

filename_50_missing_0.5 = 'run_sims_50_missing_0.5.csv'
start_time <- Sys.time()
sims_missing <- purrr::map_df(1:reps, ~run_sims_missing(filename_missing = filename_50_missing_0.5, ef = 0.5))
end_time <- Sys.time()
end_time - start_time
```

9.2 Effect Size = 0.4

```
filename_50_missing_0.4 = "run_sims_50_missing_0.4.csv"
start_time <- Sys.time()
sims_missing <- purrr::map_df(1:reps, ~run_sims_missing(filename_missing = filename_50_missing_0.4,
    ef = 0.4))
end_time <- Sys.time()
end_time - start_time
```

9.3 Effect Size = 0.3

```
filename_50_missing_0.3 = "run_sims_50_missing_0.3.csv"
start_time <- Sys.time()
sims_missing <- purrr::map_df(1:reps, ~run_sims_missing(filename_missing = filename_50_missing_0.3,
    ef = 0.3))
end_time <- Sys.time()
end_time - start_time
```

9.3.1 Visualise Estimates for Fixed Effects:

```
# read saved simulation data
sims_50_missing_0.3 <- read_csv(filename_50_missing_0.3, col_types = cols(
  # makes sure plots display in this order
  group = col_factor(ordered = TRUE),
  term = col_factor(ordered = TRUE)
))

fixed_missing_plot <- sims_50_missing_0.3 %>%
  filter(effect == "fixed") %>%
  ungroup() %>%
  arrange(term, estimate) %>%
  mutate(row = rep(seq(1:reps), 8)) %>%
  ggplot(aes(x = row, y = estimate, ymin = estimate-std.error, ymax = estimate+std.error)) +
  facet_wrap(~term, scales = "free") +
  geom_pointrange(fatten = 1/2) +
  ylab("Estimates") +
  xlab("Simulations") +
  ggtitle('Estimates of Fixed Effects for 50 pct. Missing Data, ef = 0.3') +
  theme_bw()

fixed_missing_plot <- fixed_missing_plot + theme(plot.title = element_text(hjust = 0.5, size=20))
fixed_missing_plot
```

9.3.2 Visualise Estimates for Random Effects:

```
ran_missing_plot <- sims_20_missing_0.3 %>%
    filter(effect == "ran_pars") %>%
    ungroup() %>%
    arrange(group, term, estimate) %>%
    mutate(row = rep(seq(1:reps), 13)) %>%
    ggplot(aes(x = row, y = estimate)) + geom_point(alpha = 0.7) +
    facet_wrap(~group + term, scales = "free_y") + theme_bw() +
    ylab("Estimates") + xlab("Simulations") + ggtitle("Estimates of Random Effects for 50 pct. Missing Data, ef =
    theme_bw()
ran_missing_plot <- ran_missing_plot + theme(plot.title = element_text(hjust = 0.5,
```

```
    size = 20))
ran_missing_plot
```

9.4 Effect Size = 0.2

```
filename_50_missing_0.2 = "run_sims_50_missing_0.2.csv"
start_time <- Sys.time()
sims_missing <- purrr::map_df(1:reps, ~run_sims_missing(filename_missing = filename_50_missing_0.2,
    ef = 0.2))
end_time <- Sys.time()
end_time - start_time
```

9.5 Effect Size = 0.1

```
filename_50_missing_0.1 = "run_sims_50_missing_0.1.csv"
start_time <- Sys.time()
sims_missing <- purrr::map_df(1:reps, ~run_sims_missing(filename_missing = filename_50_missing_0.1,
    ef = 0.1))
end_time <- Sys.time()
end_time - start_time
```