

Additional file 7. Assessing causal links between age at menarche and adolescent mental health: a Mendelian randomisation study (Stage 1 code supplement)

Code for this project is also available at: <https://github.com/psychgen/aam-psych-adolesc-rr>

Table of Contents

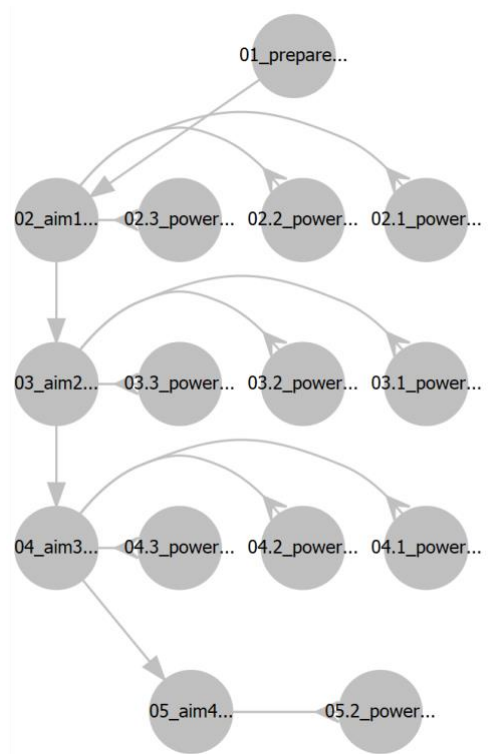
Overview of project.....	4
Data preparation for the project (01_prepare_data.R).....	5
MoBa/MBRN variables required:	5
Phenotypic:	5
Genetic:.....	5
Registry variables required:	5
Primary care diagnoses (KUHR; ICPC-2):	5
Secondary care diagnoses (NPR; ICD-10):	5
Load required packages	6
Curate dataset with variables from MoBa phenotypic data.....	6
Fetch and process polygenic scores.....	7
Join genetic and phenotypic data for complete analytic dataset.....	7
Aim 1: Investigate to what extent early age at menarche is associated with elevated rates of adolescent depressive symptoms/depression (02_aim1_analyses.R).....	8
Hypothesis 1a	8
Hypothesis 1b	9
Power analyses for hypothesis/equivalence tests in Aim 1 (02.1_power_simulation_aim1a.R) ..	11
Load required packages	11
Power analysis 1a	11
Simulation	11
Power analyses for hypothesis/equivalence tests in Aim 1 (02.2_power_simulation_aim1b.R) ..	14
Load required packages	14
Power analysis 1b	14
Simulation	14
Power analyses for hypothesis/equivalence tests in Aim 1 (02.1_power_analyses_aim1.R)	18
Power.....	18
Power analysis 1b	19
Power.....	19

36	Aim 2: Investigate to what extent effects of age at menarche extend to other domains of mental	
37	health (03_aim2_analyses.R)	22
38	Hypotheses 2.1-4a	22
39	Hypotheses 2.1-3b	24
40	Power analyses for aim 2 hypothesis/equivalence tests (03.1_power_simulation_aim2a.R)	26
41	Load required packages	26
42	Power analysis 2a	26
43	Simulation	26
44	Power analyses for aim 2 hypothesis/equivalence tests (03.2_power_simulation_aim2b.R)	30
45	Load required packages	30
46	Power analysis 2b	30
47	Simulation	30
48	Power analyses for aim 2 hypothesis/equivalence tests (03.1_power_analyses_aim2.R)	34
49	Load required packages	34
50	Power analysis 2a	34
51	Power	34
52	Power analysis 2b	35
53	Power	35
54	Aim 3: Determine whether any associations between early age at menarche and adolescent	
55	depressive symptoms/depression are likely to be causal (04_aim3_analyses.R)	38
56	Hypothesis 3a	38
57	Hypothesis 3b	39
58	Power analyses for aim 3(&4) hypothesis/equivalence tests (04.1_power_simulation_aim34a.R)	
59	41
60	Load required packages	41
61	Power analysis 3/4a	41
62	Simulation	41
63	Power analyses for aim 3(&4) hypothesis/equivalence tests (04.2_power_simulation_aim34b.R)	
64	45
65	Load required packages	45
66	Power analysis 3/4b	45
67	Simulation	45
68	Power analyses for aim 3(&4) hypothesis/equivalence tests (04.3_power_summary_aim3.R)	50
69	Load required packages	50
70	Power	50
71	Power analysis 3/4b	51
72	Power	52
73	Aim 4: Determine whether any associations between early age at menarche and adolescent	
74	symptoms/diagnoses in other domains are likely to be causal (05_aim4_analyses.R)	55
75	Hypotheses 4.1-4a	55

76	Hypotheses 4.1-3b	56
77	Summary of power analyses for aim 4 hypothesis/equivalence tests (05.1_power_summary_aim4.R)	
78	59
79	Load required packages	59
80	Power.....	59
81	Power analysis 3/4b	60
82	Power.....	61
83	Appendix 1: Psychometric properties of scales	64
84	Appendix 2: Meta-analysis of the association between age at menarche and depressive symptoms	
85	65
86	Overview	65
87	Load libraries.....	65
88	Extract data from studies	65
89	Joinson et al. (2013)	65
90	Black & Klein (2012)	66
91	Stice et al. (2001)	66
92	Lien et al. (2010)	66
93	Kaltiala-Heino et al. (2003)	67
94	Merge effect sizes	68
95	Perform meta-analysis.....	68
96	Convert effect size to <i>OR</i>	68
97	Convert effect size to Cohen's <i>D</i>	68
98	Decide equivalence bounds.....	68
99		
100		

Overview of project

The project consists of four aims, with each aim comprising several hypotheses. In terms of the structure of the code, we first prepare data (level one of the schematic below), before carrying out power analyses and providing code for the analyses on which primary inferences will be made for each hypothesis, in each aim (subsequent levels). The analytic code will be expanded to include sensitivity and exploratory analyses, as well as results processing, at Stage 2.



Data preparation for the project (01_prepare_data.R)

The purpose of this script is to source and prepare variables for analysis.

Most of the preparation of raw variables in the MoBa dataset and linked registry sources is done using the `phenotools` R package, code and documentation for which is available at <https://github.com/psychgen/phenotools>.

MoBa/MBRN variables required:

Phenotypic:

- self-rep age at menarche and pubertal stage 14 year;
- depressive sx (sMFQ) 8 & 14 year;
- anxiety sx (SCARED) 8 & 14 year;
- conduct disorder sx (RS-DBD) 8 & 14 year;
- oppositional defiant disorder (RS-DBD) sx 8 & 14 year;
- ADHD sx (RS-DBD) 8 & 14 yr;
- Covariates: BMI (derived) 8 & 14 yr; maternal/paternal age; child age at
- Q completion; parental education; parental income; parental cohabitation;
- number of children in family; financial problems; maternal prenatal
- depression (SCL); maternal postnatal depression (EPD)

Genetic:

- age at menarche PRS (GWSig hits only) 10.1038/ng.3841
- childhood body size PRS (GWSig hits only) 10.1136/bmj.m1203
- adult BMI PRS (GWSig hits only) 10.1136/bmj.m1203

Registry variables required:

Primary care diagnoses (KUHR; ICPC-2):

- depressive disorders: P76
- anxiety disorders: P74, P79, P82
- adhd: P81
- conduct/behavioural disorders: P23

Secondary care diagnoses (NPR; ICD-10):

- depressive disorders: F32-F33, F34.1
- anxiety disorders: F40-F44, F93.0-F93.2
- adhd: F90
- conduct/behavioural disorders: F91, F92

Load required packages

```
library(phenotools)
library(tidyverse)
library(genotools)
```

Curate dataset with variables from MoBa phenotypic data

```
covars <- c("KJONN",
            "age_at_q_ret_8yr",
            "parent_income_derived_q1",
            "parent_educ_derived_q1",
            "parent_cohab_18m",
            "parent_cohab_3yr",
            "p_age_at_birth",
            "m_age_at_birth",
            "financ_probs_18m",
            "scl_5item_m_q1",
            "scl_full_m_q3",
            "epds_full_m_6m",
            "n_children_8yr",
            "bmi_derived_c_8yr",
            "bmi_derived_c_14s")

q8yr_vars <- c("smfq_dep_c_8yr",
              "scared_anx_c_8yr",
              "rsdbd_adhd_c_8yr",
              "rsdbd_cd_c_8yr",
              "rsdbd_odd_c_8yr")

q14yr_vars <- c("aam_c_14s",
               "smfq_dep_c_14s",
               "scared_anx_c_14s",
               "rsdbd_adhd_c_14m",
               "rsdbd_cd_c_14s",
               "rsdbd_odd_c_14m")

if(!file.exists("data/pheno_data.RData")){
  pheno_data <- curate_dataset(
    variables_required=list(moba=c(covars,
                                   q8yr_vars,
                                   q14yr_vars),
                           npr=c("dep=F32,F33,F341",
                                   "anx=F40,F41,F42,F43,F44,F930,F931,F932",
                                   "adhd=F90",
                                   "beh=F91,F92"),
                           kuhr=c("dep=P76",
                                   "anx=P74, P79, P82",
                                   "adhd=P81",
                                   "beh=P23")),
    out_format = "merged_df",
    recursive=T,
    dx_owners="child")
  pheno_data <- pheno_data %>%
    rename("sex"=KJONN_raw)
  filter(sex ==2)
}
```

```

201   save(pheno_data, file="data/pheno_data.RData")
202 }else{
203   load(file="data/pheno_data.RData")
204 }

```

205 NB: all 14-year variables are (by design) not available during preparation/submission of Stage 1 RR

206 Fetch and process polygenic scores

```

207 #Not run
208 pgs <- fetch_prs(c("aam", "bmi", "cbmi"),
209                 thresholds = c( 1),
210                 threshold_range = FALSE,
211                 geno_data = "moba_full",
212                 maf = "0.01",
213                 clump = "10000_1_0.001")
214
215 pgs_procd <- process_prs(pgs,
216                         geno_data = "moba_full",
217                         regress_prs = TRUE,
218                         return_covs = TRUE)

```

219 NB: polygenic scores generated using PRSice will be read in to R and processed in this section
 220 using functions from the genotools package, as outlined above.

221 Join genetic and phenotypic data for complete analytic dataset

```

222 fulldata <- pheno_data %>%
223   left_join(pgs_procd)
224
225
226 #save out
227 save(fulldata, file="data/analysis_dataset.RData")

```

228

Aim 1: Investigate to what extent early age at menarche is associated with elevated rates of adolescent depressive symptoms/depression (02_aim1_analyses.R)

The purpose of this script is to run the main analyses corresponding to Aim 1. Specifically, we will:

- Run a linear regression model with NHS+equivalence tests for aam on dep_sx
- Run a logistic regression with NHS+equivalence tests for aam on dep_DX

Read in the data and load packages:

```
load(file="data/analysis_dataset.RData")
library(tidyverse)
library(parameters)
library(effectsize)
```

Hypothesis 1a

```
fulldata <- fulldata %>%
  mutate(across( c("smfq_dep_c_14s", "aam_c_14s", "smfq_dep_c_8yr" ), scale ))

modell1a_unadj <- lm(smfq_dep_c_14s ~ aam_c_14s ,
  data = fulldata)

modell1a_adj <- lm(smfq_dep_c_14s ~ aam_c_14s +
  sex + age_at_q_ret_8yr + parent_income_derived_q1 +
  parent_educ_derived_q1 + parent_cohab_18m +
  parent_cohab_3yr + p_age_at_birth + m_age_at_birth +
  financ_probs_18m + scl_5item_m_q1 + scl_full_m_q3 +
  epds_full_m_6m + n_children_8yr + bmi_derived_c_8yr +
  bmi_derived_c_14s,
  data = fulldata)

modell1a_8yr_unadj <- lm(smfq_dep_c_14s ~ aam_c_14s + smfq_dep_c_8yr,
  data = fulldata)

modell1a_8yr_adj <- lm(smfq_dep_c_14s ~ aam_c_14s + smfq_dep_c_8yr +
  sex + age_at_q_ret_8yr + parent_income_derived_q1 +
  parent_educ_derived_q1 + parent_cohab_18m +
  parent_cohab_3yr + p_age_at_birth + m_age_at_birth +
  financ_probs_18m + scl_5item_m_q1 + scl_full_m_q3 +
  epds_full_m_6m + n_children_8yr + bmi_derived_c_8yr +
  bmi_derived_c_14s,
  data = fulldata)

models <- list(modell1a_unadj, modell1a_adj,
  modell1a_8yr_unadj, modell1a_8yr_adj)
```



```

275
276 all_res <- map(models, function(res){
277
278   #Extract the results of the NHST
279
280   nhst_res <- res %>%
281     summary() %>%
282     .$coefficients %>%
283     as.data.frame() %>%
284     rownames_to_column("predictor")
285
286   #Run the equivalence test (using d=0.23 as SESOI)
287
288   equiv_res <- parameters::equivalence_test(
289     res,
290     range= c(-effectsize::d_to_r(0.23),effectsize::d_to_r(0.23)), rule="classic") %>
291 %
292   as.data.frame() %>%
293   `row.names<-`(NULL)
294
295   #Combine
296
297   comb_res <- nhst_res %>%
298     bind_cols(equiv_res) %>%
299     mutate(p_onetailed = ifelse(Estimate<0,`Pr(>|t|)`/2,1)) #One-tailed test p-value
300
301 })

```

302 Hypothesis 1b

```

303 model1b_unadj <- glm(dep_dx_10_17 ~ aam_c_14s ,
304                     family = binomial(link="logit"),
305                     data = fulldata)
306
307 model1b_adj <- glm(dep_dx_10_17 ~ aam_c_14s +
308                   sex + age_at_q_ret_8yr + parent_income_derived_q1 +
309                   parent_educ_derived_q1 + parent_cohab_18m +
310                   parent_cohab_3yr + p_age_at_birth + m_age_at_birth +
311                   financ_probs_18m + scl_5item_m_q1 + scl_full_m_q3 +
312                   epds_full_m_6m + n_children_8yr + bmi_derived_c_8yr +
313                   bmi_derived_c_14s,
314                   family = binomial(link="logit"),
315                   data = fulldata)
316
317 model1b_8yr_unadj <- glm(dep_dx_10_17 ~ aam_c_14s + dep_dx_0_8,
318                         family = binomial(link="logit"),
319                         data = fulldata)
320
321 model1b_8yr_adj <- glm(dep_dx_10_17 ~ aam_c_14s + dep_dx_0_8 +
322                      sex + age_at_q_ret_8yr + parent_income_derived_q1 +
323                      parent_educ_derived_q1 + parent_cohab_18m +
324                      parent_cohab_3yr + p_age_at_birth + m_age_at_birth +
325                      financ_probs_18m + scl_5item_m_q1 + scl_full_m_q3 +
326                      epds_full_m_6m + n_children_8yr + bmi_derived_c_8yr +
327                      bmi_derived_c_14s,
328                      family = binomial(link="logit"),
329                      data = fulldata)

```

```

330
331 models <- list(model1b_unadj, model1b_adj,
332               model1b_8yr_unadj,model1b_8yr_adj)
333
334
335 all_res <- map(models, function(res){
336
337   #Extract the results of the NHST
338
339   nhst_res <- res %>%
340     summary() %>%
341     .$coefficients %>%
342     as.data.frame() %>%
343     rownames_to_column("predictor")
344
345   #Run the equivalence test (using d=0.23 as SESOI)
346
347   equiv_res <- parameters::equivalence_test(
348     res, range= c(log(d_to_oddsratio(-0.23)),log(d_to_oddsratio(0.23))), rule="classi
349 c") %>%
350     as.data.frame() %>%
351     `row.names<-`(NULL)
352
353   #Combine
354
355   comb_res <- nhst_res %>%
356     bind_cols(equiv_res) %>%
357     mutate(p_onetailed = ifelse(Estimate<0,`Pr(>|t|)`/2,1)) #One-tailed test p-value
358
359 })

```

360


```

404         empirical = FALSE) %>%
405     mutate(DepSx14yr = ifelse(DepSx14yr<0,0,round(DepSx14yr)),
406           #Assume a floor effect; this attenuates the correlation slightly
407           AaM = round(AaM/12) )
408
409     # Note that the possibility of AaM values > 168 represents the situation
410     # after we have imputed right censored values
411
412     #Run the model
413
414     res <- lm(scale(DepSx14yr) ~ scale(AaM), data = simdat)
415
416     #Extract the results of the NHST
417
418     nhst_res <- res %>%
419       summary() %>%
420       .$coefficients %>%
421       as.data.frame() %>%
422       `row.names<-` (NULL) %>%
423       .[2,]
424
425     #Run the equivalence test (using OR = 1.4 as SESOI)
426
427     equiv_res <- parameters::equivalence_test(
428       res, range= c(-effectsize::d_to_r(equiv_d),effectsize::d_to_r(equiv_d)), rule="cl
429 assic") %>%
430       as.data.frame() %>%
431       `row.names<-` (NULL) %>%
432       .[2,]
433
434
435     #Combine and apply decision rules - inferiority (i.e.,
436     #only use high equiv bound (negative in this case))
437
438     comb_res <- nhst_res %>%
439       bind_cols(equiv_res) %>%
440       mutate(p = ifelse(Estimate<0,`Pr(>|t|)`/2,1), #One-tailed test p-value computatio
441 n
442           iteration = iteration,
443           corr = corr,
444           sample_size=sample_size)
445
446
447     simres_full<- comb_res
448
449
450     row.names(simres_full)<- NULL
451     return(simres_full)
452 }

```

We apply this function with different parameters for the first power analysis

```

454 # Set range of parameters
455 corrs <- c(seq(0,-0.15,-0.01)) #start, end, increment
456 ns <- seq(12000,14000,1000)
457 equiv <- oddsratio_to_d(1.4)
458

```

```

459
460 nsims=1000
461
462 ## Load the checkpointing file, if it exists:
463 checkFile <- "checkpoint_aim1a.RData"
464 tempFile <- "tempCheckpoint_aim1a.RData"
465 if (file.exists(checkFile)) {
466   load(checkFile)
467   cat("Resuming after iteration", iter, "\n")
468   iter <- iter + 1
469 }else{
470   # Create some objects to make the timer work
471   iter<- 1
472   durs <- vector()
473   # Create an empty DF for the output
474   fullsim<- data.frame()
475 }
476
477 if(iter <= nsims){
478   for(iter in iter:nsims){
479
480
481     message(paste0(
482       "\nReplicate ",iter, " of ", nsims
483     ))
484     ptm <- proc.time()
485
486     # Loop over the function
487     for(r in corrs){
488       for(n in ns){
489
490         scen_temp <- sim_dep_sx(corr=r,
491                                sample_size=n,
492                                iteration=iter,
493                                equiv_d= equiv)
494         fullsim <- rbind(fullsim, scen_temp)
495         ## Save the results of the iteration. (By first saving to a temporary file
496         ## and then renaming it into the checkpointing file, we guard against being
497         ## interrupted while saving.)
498         save.image(tempFile)
499         file.rename(tempFile, checkFile)
500       }
501     }
502     dur <- proc.time()-ptm
503     durs <-c(durs,dur[3])
504     message(paste0("\nProjected finish in ",
505                    round( mean(durs)*(nsims-iter)/60,2), " mins" ))
506   }
507 }
508
509 save(fullsim, file="./output/fullsim_power1a.RData")
510 ## Clean up (_after_ saving the results)
511 if (file.exists(checkFile)) file.remove(checkFile)

```

512

Power analyses for hypothesis/equivalence tests in Aim 1

(02.2_power_simulation_aim1b.R)

The purpose of this script is to run power analyses for the hypothesis/equivalence tests in Aim 1. Specifically, we will:

- Simulate a depressive symptoms outcome variable based on a range of scenarios in which the association with (simulated) age at menarche increases from 0.01 to 0.1 in increments of 0.01, and in which availability of 14-year data varies between 12k, 13k, and 14k
- Run linear models on the simulated data for 1000 iterations of each scenario, deriving power empirically as the proportion of significant (#' at alpha = 5%) effects detected in each scenario (power analysis 1a)
- Simulate equivalent data and run equivalent power analyses for the generalized linear models of depression diagnoses, additionally varying prevalence of depression (power analysis 1b)

Load required packages

```
library(tidyverse)
library(faux)
library(effectsize)
library(parameters)
```

Power analysis 1b

Simulation

Here we create a function which simulates both the 14 yr age at menarche (AAM) and depression cases, based on: 1. The effect size 2. 14-yr data availability (as a proportion of 8-year availability) 3. Prevalence of depression in sample ...and runs a linear model and accompanying equivalence test, saving standardised betas and inferences based on decision rules in a data.frame

```
set.seed(82928)

sim_dep_dx <- function(effect_size_d,
                        sample_size,
                        case_rate,
                        iteration,
                        equiv_d){

  # Mean and SD values for AaM are from 10.1192/bjp.bp.115.168617 Sequeira et al.
  # Supp table DS1

  or <- d_to_oddsratio(effect_size_d)

  simdat <- tibble("AaM" = rnorm(n = sample_size,
                                m = 151.52,
```

```

556         sd = 14.11))
557
558     # The new way, the downside of which is that the only "error" is in the measurement
559     # of AaM, but since we specify beta1 for AaM as it is "measured", this is not propa
560 gated
561     # to the models - the net result being that we always make the same inference
562     # in each iteration of a given scenario (not realistic)
563
564     # To avoid this, we will add noise to the beta1 parameter
565
566     beta0 <- log((1/(1-case_rate))-1)
567     beta1 <- log(or) + rnorm(1, mean=0, sd=0.10)
568     pi_x <- exp(beta0 + beta1 * scale(simdat$AaM)) / (1 + exp(beta0 + beta1 * scale(sim
569 dat$AaM)))
570     simdat$"DepDx" <- rbinom(n=length(simdat$AaM), size=1, prob=pi_x)
571
572     obs_case_rate <- prop.table(table(simdat$DepDx))[[2]]
573
574
575     # Note that the possibility of AaM values > 168 represents the situation
576     # after we have imputed right censored values
577
578
579     #Run the model
580
581     res <- glm(DepDx ~ scale(AaM), family = binomial(link="logit"), data=simdat)
582
583     #Extract the results of the NHST
584
585     nhst_res <- res %>%
586       summary() %>%
587       .$coefficients %>%
588       as.data.frame() %>%
589       `row.names<-`(NULL) %>%
590       .[,2] %>%
591       mutate(oddsr = exp(Estimate), # Odds ratio/gradient
592             var.diag = diag(vcov(res))[,2], # Variance of each coefficient
593             or.se = sqrt(or^2 * var.diag)) # Odds-ratio adjusted
594
595     #Run the equivalence test (using OR = 1.4 as SESOI)
596
597     equiv_res <- parameters::equivalence_test(
598       res, range= c(log(d_to_oddsratio(-equiv_d)), log(d_to_oddsratio(equiv_d))), rule="
599 classic") %>%
600     as.data.frame() %>%
601     `row.names<-`(NULL) %>%
602     .[,2]
603
604     #Combine and apply decision rules - inferiority, i.e., only use high bound
605
606     comb_res <- nhst_res %>%
607       bind_cols(equiv_res) %>%
608       mutate(p = ifelse(Estimate<0, `Pr(>|z|)`/2, 1), #One-tailed test p-value computatio
609 n
610       iteration = iteration,

```

```

612         effect_size = or,
613         sample_size=sample_size,
614         case_rate=case_rate,
615         obs_case_rate=obs_case_rate) # We monitor this because, at a certain effec
616 t size,
617     # the case rate gets inflated for a given sample size
618
619
620     simres_full<- comb_res
621
622
623
624     row.names(simres_full)<- NULL
625     return(simres_full)
626 }

```

627 We apply this function with different parameters for the second power analysis

```

628 # Set range of parameters
629 es <- seq(0.0,-0.26,-0.02)
630 ns <- c(12000,13000)
631 rates <- c(0.02,0.06,.10,.14)
632
633 equiv=oddsratio_to_d(1.4)
634
635 nsims=1000
636
637 ## Load the checkpointing file, if it exists:
638 checkFile <- "checkpoint_aim1b.RData"
639 tempFile <- "tempCheckpoint_aim1b.RData"
640 if (file.exists(checkFile)) {
641     load(checkFile)
642     cat("Resuming after iteration", iter, "\n")
643     iter <- iter + 1
644 }else{
645     # Create some objects to make the timer work
646     iter<- 1
647     durs <- vector()
648     # Create an empty DF for the output
649     fullsim<- data.frame()
650 }
651
652 if(iter <= nsims){
653
654     for(iter in iter:nsims){
655
656
657         message(paste0(
658             "\nReplicate ",iter, " of ", nsims
659         ))
660         ptm <- proc.time()
661
662         # Loop over the function
663         for(e in es){
664             for(n in ns){
665                 for(c in rates){
666                     scen_temp <- suppressMessages(sim_dep_dx(effect_size_d=e,

```



```

667                                     sample_size=n,
668                                     case_rate = c,
669                                     equiv_d=equiv,
670                                     iteration=iter))
671     fullsim <- rbind(fullsim, scen_temp)
672     ## Save the results of the iteration. (By first saving to a temporary file
673     ## and then renaming it into the checkpointing file, we guard against being
674     ## interrupted while saving.)
675     save.image(tempFile)
676     file.rename(tempFile, checkFile)
677   }
678 }
679 }
680
681 dur <- proc.time()-ptm
682 durs <-c(durs,dur[3])
683 message(paste0("\nProjected finish in ",
684               round( mean(durs)*(nsims-iter)/60,2), " mins" ))
685 }
686 }
687
688 save(fullsim, file="./output/fullsim_power1b.RData")
689 ## Clean up (_after_ saving the results)
690 if (file.exists(checkFile)) file.remove(checkFile)
691

```

692

Power analyses for hypothesis/equivalence tests in Aim 1

(02.1_power_analyses_aim1.R)

The purpose of this script is to summarise the power analyses for the hypothesis/equivalence tests in Aim 1.

```
load(file="./output/fullsim_power1a.RData")
library(tidyverse)
library(effectsize)
```

Power

Our power in NHST in each scenario is the proportion of iterations in which a true effect is detected, given the effect exists in the population

Our power in equivalence/inferiority each scenario is the proportion of iterations in which an effect is declared equivalent to zero, given that in the population it is less than the SESOI

```
#Compute one-sided p-value
fullsim <- fullsim %>%
  mutate(p = ifelse(Estimate<0,`Pr(>|t|)`/2,1))
#Summarise at the level of scenario (i.e., across iterations)
power1a <- fullsim %>%
  group_by(corr, sample_size) %>%
  summarise(power_nhst= mean(p < 0.05),
            se_nhst=sd(p < 0.05)/sqrt(n()),
            power_equiv= mean(CI_low > ROPE_low),
            se_equiv=sd(CI_low > ROPE_low)/sqrt(n())) %>%
  rename(effect_size= corr,N = sample_size) %>%
  mutate(effect_size=r_to_d(effect_size )) %>%
  pivot_longer(cols=matches("power|se_"), names_to = "type", values_to="power") %>%
  separate(type, into=c("est","test")) %>%
  pivot_wider(names_from=est,values_from=power) %>%
  mutate( test=factor(test,
                      levels=c("nhst","equiv"),
                      labels=c("NHST\n(H1:Effect > 0)","Equivalence\n(H1:Effect = 0)"))
),
  N=factor(N))%>%
  filter(N%in%c("12000","13000"))

#Plot the power curves
ggplot(data=power1a, aes(x= effect_size, y= power, fill=N,
                        colour=N,group=N))+
  geom_errorbar(aes(ymin=power-se,ymax=power+se), width=0, size=1.2, alpha=0.4)+
  geom_point(size=2.6,
            alpha=0.4)+
  geom_line(size=0.8)+
  geom_hline(yintercept=0.95, colour="red",size=1.4, linetype=2)+
  geom_hline(yintercept=0.80, colour="orange",size=1.4, linetype=3, alpha=0.7)+
```

```

739 facet_grid(test~.)+
740 coord_cartesian(ylim=c(0,1))+
741 theme(text=element_text(size=11),
742       strip.text.y = element_text(angle=0, face="bold"),
743       legend.position = "bottom",
744       legend.direction= "horizontal")+
745 scale_x_continuous("Effect size (d)")+
746 scale_y_continuous("Power")
747 ggsave("./output/power1a.tiff")

```

748 We can summarise by giving a) the smallest effect size for which we have 80 and 95% power in
 749 NHST and b) the largest effect size for which we have 80 and 95% power to declare equivalence to
 750 zero:

```

751 # Smallest effect size for which 80% power (NHST):
752 power1a %>%
753   filter(power>.8,
754          str_detect(test,"NHST")) %>%
755   group_by(N) %>%
756   summarise(`min_es_80%power` = max(effect_size))%>%
757   mutate(sesoi = rep(-oddsratio_to_d(1.4),3))
758 # Smallest effect size for which 95% power (NHST):
759 power1a %>%
760   filter(power>.95,
761          str_detect(test,"NHST")) %>%
762   group_by(N) %>%
763   summarise(`min_es_95%power` = max(effect_size))%>%
764   mutate(sesoi = rep(-oddsratio_to_d(1.4),3))
765
766 # Largest effect size for which 80% power (equiv):
767 power1a %>%
768   filter(power>.8,
769          str_detect(test,"NHST",negate = T)) %>%
770   group_by(N) %>%
771   summarise(`max_es_80%power` = min(effect_size))%>%
772   mutate(sesoi = rep(-oddsratio_to_d(1.4),3))
773 # Largest effect size for which 95% power (equiv):
774 power1a %>%
775   filter(power>.95,
776          str_detect(test,"NHST",negate=T)) %>%
777   group_by(N) %>%
778   summarise(`max_es_95%power` = min(effect_size))%>%
779   mutate(sesoi = rep(-oddsratio_to_d(1.4),3))
780
781 #

```

782 Power analysis 1b

```
783 load(file="./output/fullsim_power1b.RData")
```

784 Power

785 Our power in NHST in each scenario is the proportion of iterations in which a true effect is detected,
 786 given the effect exists in the population

787 Our power in equivalence/inferiority each scenario is the proportion of iterations in which an effect is
 788 declared equivalent to zero, given that in the population it is less than the SESOI

```

789 fullsim <- fullsim %>%
790   mutate(p = ifelse(Estimate<0,`Pr(>|z|)`/2,1))
791 #Summarise at the level of scenario (i.e., across iterations)
792 power1b <- fullsim %>%
793   group_by(effect_size, sample_size,case_rate) %>%
794   summarise(power_nhst= mean(p < 0.05),
795             se_nhst=sd(p < 0.05)/sqrt(n()),
796             power_equiv= mean(CI_low > ROPE_low),
797             se_equiv=sd(CI_low > ROPE_low)/sqrt(n())) %>%
798   rename(N = sample_size, prevalence = case_rate) %>%
799   mutate(effect_size=oddsratio_to_d(effect_size)) %>%
800   pivot_longer(cols=matches("power|se_"), names_to = "type", values_to="power") %>%
801   separate(type, into=c("est","test")) %>%
802   pivot_wider(names_from=est,values_from=power) %>%
803   mutate( test=factor(test,
804                     levels=c("nhst","equiv"),
805                     labels=c("NHST\n(H1:Effect > 0)", "Equivalence\n(H1:Effect = 0)"
806   )),
807          N=factor(N)) %>%
808   filter(N%in%c("12000","13000"))

809 #Plot the power curves
810
811 prevlabs <- c("2% avg. \nprevalence","6% avg. \nprevalence","10% avg. \nprevalence")
812 names(prevlabs)<- c("0.02","0.06", "0.1")
813 ggplot(data=power1b, aes(x= effect_size, y= power, fill=N,
814                          colour=N,group=N))+
815   geom_errorbar(aes(ymin=power-se,ymax=power+se), width=0, size=1.2, alpha=0.4)+
816   geom_point(size=2.6,
817             alpha=0.4)+
818   geom_line(size=0.8)+
819   geom_hline(yintercept=0.95, colour="red",size=1.4, linetype=2)+
820   geom_hline(yintercept=0.80, colour="orange",size=1.4, linetype=3, alpha=0.7)+
821   facet_grid(test~prevalence, labeller=labeller(prevalence=prevlabs))+
822   coord_cartesian(ylim=c(0,1))+
823   theme(text=element_text(size=11),
824         strip.text.y = element_text(angle=0, face="bold"),
825         legend.position = "bottom",
826         legend.direction= "horizontal")+
827   scale_x_continuous("Effect size (d)")+
828   scale_y_continuous("Power")
829 ggsave("./output/power1b.tiff")

```

830 We can summarise by giving a) the smallest effect size for which we have 80 and 95% power in
831 NHST and b) the largest effect size for which we have 80 and 95% power to declare equivalence to
832 zero:

```

833 # Smallest effect size for which 80% power (NHST):
834 power1b %>%
835   filter(power>.8,
836          str_detect(test,"NHST")) %>%
837   group_by(N, prevalence) %>%
838   summarise(`min_es_80%power` = max(effect_size))%>%
839   mutate(sesoi = rep(-oddsratio_to_d(1.4),3))
840 # Smallest effect size for which 95% power (NHST):
841 power1b %>%
842   filter(power>.95,

```

```

843       str_detect(test, "NHST")) %>%
844   group_by(N, prevalence) %>%
845   summarise(`min_es_95%power` = max(effect_size))%>%
846   mutate(sesoi = rep(-oddsratio_to_d(1.4), 3))
847
848   # Largest effect size for which 80% power (equiv):
849   power1b %>%
850     filter(power > .8,
851       str_detect(test, "NHST", negate = T)) %>%
852     group_by(N, prevalence) %>%
853     summarise(`max_es_80%power` = min(effect_size))%>%
854     mutate(sesoi = rep(-oddsratio_to_d(1.4), 3))
855   # Largest effect size for which 95% power (equiv):
856   power1b %>%
857     filter(power > .95,
858       str_detect(test, "NHST", negate = T)) %>%
859     group_by(N, prevalence) %>%
860     summarise(`max_es_95%power` = min(effect_size))%>%
861     mutate(sesoi = rep(-oddsratio_to_d(1.4), 3))

```

862

Aim 2: Investigate to what extent effects of age at menarche extend to other domains of mental health (03_aim2_analyses.R)

The purpose of this script is to run the primary analyses corresponding to Aim 2. Specifically, we will:

- Run a linear regression model with NHS+equivalence tests for effect of aam on all sx variables
- Run a logistic regression with NHS+equivalence tests for effect of aam on on all dx variables

Read in the data and load packages:

```
load(file="data/analysis_dataset.RData")
library(tidyverse)
library(parameters)
library(effectsize)
```

Hypotheses 2.1-4a

```
# List the symptom variables
domains <-list(anx=c("scared_anx_c_8yr","scared_anx_c_14s"),
              adhd = c("rsdbd_adhd_c_8yr","rsdbd_adhd_c_14m"),
              cd=c("rsdbd_cd_c_8yr","rsdbd_cd_c_14s"),
              odd=c("rsdbd_odd_c_8yr","rsdbd_odd_c_14m"))

# Scale for standardised effects
fulldata <- fulldata %>%
  mutate(across( c("smfq_dep_c_14s","aam_c_14s","smfq_dep_c_8yr",unlist(domains) ), s
cale ))

# Run models for all domains and save results in a list
domains_res <- list()
n=0
for(domain in domains){
  n=n+1
  model2a_unadj <- lm(get(domain[[2]]) ~ aam_c_14s ,
                    data = fulldata)

  model2a_adj <- lm(get(domain[[2]]) ~ aam_c_14s +
                    sex + age_at_q_ret_8yr + parent_income_derived_q1 +
                    parent_educ_derived_q1 + parent_cohab_18m +
                    parent_cohab_3yr + p_age_at_birth + m_age_at_birth +
                    financ_probs_18m + scl_5item_m_q1 + scl_full_m_q3 +
                    epds_full_m_6m + n_children_8yr + bmi_derived_c_8yr +
                    bmi_derived_c_14s,
                    data = fulldata)

  model2a_dep_unadj <- lm(get(domain[[2]]) ~ aam_c_14s + smfq_dep_c_14s ,
                        data = fulldata)
```

```

910
911 model2a_dep_adj <- lm(get(domain[[2]]) ~ aam_c_14s + smfq_dep_c_14s +
912     sex + age_at_q_ret_8yr + parent_income_derived_q1 +
913     parent_educ_derived_q1 + parent_cohab_18m +
914     parent_cohab_3yr + p_age_at_birth + m_age_at_birth +
915     financ_probs_18m + scl_5item_m_q1 + scl_full_m_q3 +
916     epds_full_m_6m + n_children_8yr + bmi_derived_c_8yr +
917     bmi_derived_c_14s,
918     data = fulldata)
919
920 model2a_dep_8yr_unadj <- lm(get(domain[[2]]) ~ aam_c_14s + smfq_dep_c_14s + get(dom
921 ain[[1]]),
922     data = fulldata)
923
924 model2a_dep_8yr_adj <- lm(get(domain[[2]]) ~ aam_c_14s + smfq_dep_c_14s + get(domai
925 n[[1]]) +
926     sex + age_at_q_ret_8yr + parent_income_derived_q1 +
927     parent_educ_derived_q1 + parent_cohab_18m +
928     parent_cohab_3yr + p_age_at_birth + m_age_at_birth +
929     financ_probs_18m + scl_5item_m_q1 + scl_full_m_q3 +
930     epds_full_m_6m + n_children_8yr + bmi_derived_c_8yr +
931     bmi_derived_c_14s,
932     data = fulldata)
933
934 models <- list(model2a_unadj, model2a_adj,
935     model2a_dep_unadj,model2a_dep_adj,
936     model2a_dep_8yr_unadj,model2a_dep_8yr_adj)
937
938 all_res <- map(models, function(res){
939
940     #Extract the results of the NHST
941
942     nhst_res <- res %>%
943         summary() %>%
944         .$coefficients %>%
945         as.data.frame() %>%
946         rownames_to_column("predictor")
947
948     #Run the equivalence test (using d=0.22 as SESOI)
949
950     equiv_res <- parameters::equivalence_test(
951         res, range= c(-effectsize::d_to_r(0.22),effectsize::d_to_r(0.22)), rule="classi
952 c") %>%
953         as.data.frame() %>%
954         `row.names<-`(NULL)
955
956     #Combine
957
958     comb_res <- nhst_res %>%
959         bind_cols(equiv_res)
960 })
961
962 domains_res[[names(domains)[[n]]]] <- all_res
963
964 }

```

Hypotheses 2.1-3b

List the symptom variables

```
domains <-list(anx=c("anx_dx_0_8", "anx_dx_10_17"),
              adhd = c("adhd_dx_0_8", "adhd_dx_10_17"),
              cdodd=c("cdodd_dx_0_8", "cdodd_dx_10_17"))
```

```
# Run models for all domains and save results in a list
```

```
domains_res <- list()
```

$$n=0$$

```
for(domain in domains){
```

$n = n + 1$

```
model2b_unadj <- glm(get(domain[[2]]) ~ aam_c_14s ,
                      family = binomial(link="logit"),
                      data = fulldata)
```

```
model12b_adj <- glm(get(domain[[2]]) ~ aam_c_14s +
  sex + age_at_q_ret_8yr + parent_income_derived_q1 +
  parent_educ_derived_q1 + parent_cohab_18m +
  parent_cohab_3yr + p_age_at_birth + m_age_at_birth +
  financ_probs_18m + scl_5item_m_q1 + scl_full_m_q3 +
  epds_full_m_6m + n_children_8yr + bmi_derived_c_8yr +
  bmi_derived_c_14s,
  family = binomial(link="logit"),
  data = fulldata)
```

```
model2b_dep_unadj <- glm(get(domain[[2]]) ~ aam_c_14s + dep_dx_10_17 ,
                          family = binomial(link="logit"),
                          data = fulldata)
```

```
model2b_dep_adj <- glm(get(domain[[2]]) ~ aam_c_14s + dep_dx_10_17 +
  sex + age_at_q_ret_8yr + parent_income_derived_q1 +
  parent_educ_derived_q1 + parent_cohab_18m +
  parent_cohab_3yr + p_age_at_birth + m_age_at_birth +
  financ_probs_18m + scl_5item_m_q1 + scl_full_m_q3 +
  epds_full_m_6m + n_children_8yr + bmi_derived_c_8yr +
  bmi_derived_c_14s,
  family = binomial(link="logit"),
  data = fulldata)
```

```
model2b_dep_8yr_unadj <- glm(get(domain[[2]]) ~ aam_c_14s + dep_dx_10_17 + get(domain[[1]]),
```

```
family = binomial(link="logit"),
data = fulldata)
```

```
model2b_dep_8yr_adj <- glm(get(domain[[2]]) ~ aam_c_14s + dep_dx_10_17 + get(domain
[[1]]) +
```

```
sex + age_at_q_ret_8yr + parent_income_derived_q1 +
parent_educ_derived_q1 + parent_cohab_18m +
parent_cohab_3yr + p_age_at_birth + m_age_at_birth +
financ_probs_18m + scl_5item_m_q1 + scl_full_m_q3 +
epds_full_m_6m + n_children_8yr + bmi_derived_c_8yr +
bmi_derived_c_14s,
family = binomial(link="logit"),
```



```

1021         data = fulldata)
1022
1023     models <- list(model2b_unadj, model2b_adj,
1024                   model2b_dep_unadj,model2b_dep_adj,
1025                   model2b_dep_8yr_unadj,model2b_dep_8yr_adj)
1026
1027     all_res <- map(models, function(res){
1028
1029         #Extract the results of the NHST
1030
1031         nhst_res <- res %>%
1032             summary() %>%
1033             .$coefficients %>%
1034             as.data.frame()%>%
1035             rownames_to_column("predictor")
1036
1037         #Run the equivalence test (using d=0.22 as SESOI)
1038
1039         equiv_res <- parameters::equivalence_test(
1040             res, range= c(-effectsize::d_to_r(0.22),effectsize::d_to_r(0.22)), rule="classi
1041 c") %>%
1042             as.data.frame() %>%
1043             `row.names<-`(NULL)
1044
1045         #Combine
1046
1047         comb_res <- nhst_res %>%
1048             bind_cols(equiv_res)
1049     })
1050
1051     domains_res[[names(domains)[[n]]]] <- all_res
1052
1053 }

```

1054

Power analyses for aim 2 hypothesis/equivalence tests

(03.1_power_simulation_aim2a.R)

The purpose of this script is to run the power analyses corresponding to Aim 2 hypothesis tests. Specifically, we will:

- Simulate outcomes in four new symptom domains (anx, cd, odd, adhd) in which associations with aam are randomly drawn from a normal distribution with an incrementally increasing mean, and in which availability of 14-year data (that is, sample size for the analyses) varies (as a percentage of the observed 8-year sample size) from 55-65% in increments of 5%
- Run linear models on simulated data for 1000 iterations per scenario, deriving power empirically for the NHS+Equivalence tests of a null hypotheses that effects are practically equivalent to zero (alpha = 5%)
- Run logistic regression models of aam on other diagnoses (anx, cd/odd, adhd) and test the null hypothesis that each effect is practically equivalent to zero (alpha = 5%)

Load required packages

```
library(tidyverse)
library(effectsize)
library(parameters)
library(faux)
```

Power analysis 2a

Simulation

Here we create a function which simulates both the 14 yr symptoms variables and age at menarche (AAM) variable based on two inputs: 1. The average correlation (across all domains) between with aam 2. The 14-yr data availability (as a proportion of 8-year availability) ...and runs a linear model and accompanying equivalence test, saving standardised betas and inferences based on decision rules in a data.frame

```
set.seed(91734)

sim_all_sx <- function(corr=0.1,
                      sample_size=13000,
                      iteration=1,
                      equiv_d=0.1){

  # Mean and SD values are from 10.1192/bjp.bp.115.168617 Sequeira et al.
  # Supp table DS1 (aam) and observed 8year data in MoBa:

  #anx m= 1.03 sd= 1.20
  #cd m= 0.78 sd= 1.51
  #odd m= 3.42 sd= 3.16
```

```

1098   #adhd m= 8.54  sd= 7.23
1099
1100   simdat <- rnorm_multi(n = sample_size,
1101                        mu = c(151.52, 1.03),
1102                        sd = c(14.11, 1.2),
1103                        r = rnorm(1, corr, 0.02),
1104                        varnames = c("AaM", "AnxSx14yr"),
1105                        empirical = FALSE) %>%
1106     mutate(AnxSx14yr = ifelse(AnxSx14yr<0,0,round(AnxSx14yr)),
1107            #Assume a floor effect; this attenuates the correlation slightly
1108            CDsx14yr = rnorm_pre(AaM, mu= 0.78,sd=1.51 ,
1109                                r =rnorm(1, corr, 0.02), empirical=FALSE ),
1110            ODDsx14yr = rnorm_pre(AaM, mu= 3.42,sd= 3.16,
1111                                r =rnorm(1, corr, 0.02), empirical=FALSE ),
1112            ADHDsx14yr = rnorm_pre(AaM, mu= 8.54,sd= 7.23,
1113                                r =rnorm(1, corr, 0.02), empirical=FALSE ),
1114            CDsx14yr = ifelse(CDsx14yr<0,0,round(CDsx14yr)),
1115            #Assume a floor effect; this attenuates the correlation slightly
1116            ODDsx14yr = ifelse(ODDsx14yr<0,0,round(ODDsx14yr)),
1117            #Assume a floor effect; this attenuates the correlation slightly
1118            ADHDsx14yr = ifelse(ADHDsx14yr<0,0,round(ADHDsx14yr)),
1119            #Assume a floor effect; this attenuates the correlation slightly
1120            AaM = round(AaM/12) )
1121
1122
1123   # Note that the possibility of AaM values > 168 represents the situation
1124   # after we have imputed right censored values
1125
1126   #Run the models
1127
1128   res <-list(
1129     lm(scale(AnxSx14yr)~ scale(AaM), data = simdat),
1130     lm(scale(CDsx14yr)~ scale(AaM), data = simdat),
1131     lm(scale(ODDsx14yr)~ scale(AaM), data = simdat),
1132     lm(scale(ADHDsx14yr) ~ scale(AaM), data = simdat) )
1133
1134   #Extract the results of the NHST
1135
1136   nhst_res <- map(res,function(x){
1137     x %>%
1138     summary() %>%
1139     .$coefficients %>%
1140     as.data.frame()%>%
1141     `row.names<-`(NULL) %>%
1142     .[,2,]
1143   })%>%
1144   reduce(bind_rows) %>%
1145   rename(p=`Pr(>|t|)` )
1146
1147   #Run the equivalence test
1148
1149   equiv_res <- map(res, function(x){
1150     x %>%
1151     parameters::equivalence_test( range= c(d_to_r(-equiv_d),d_to_r(equiv_d)),
1152                                   rule="classic",
1153                                   ci=1-(0.05/nrow(nhst_res)),p_values=T) %>%

```

```

1154     as.data.frame() %>%
1155     select(-p) %>%
1156     `row.names<-`(NULL) %>%
1157     .[,2,]
1158   }) %>%
1159   reduce(bind_rows)
1160
1161   #Combine and apply decision rules
1162
1163   comb_res <- nhst_res %>%
1164     bind_cols(equiv_res) %>%
1165     mutate(iteration = iteration,
1166            corr = corr,
1167            sample_size=sample_size)
1168
1169   simres_full<- comb_res
1170
1171   row.names(simres_full)<- NULL
1172   return(simres_full)
1173 }

```

1174 We apply this function with different parameters for the first power analysis

```

1175 # Set range of parameters
1176 corrs <- c(seq(-0.01,-0.15,-0.01)) #start, end, increment
1177 ns <- seq(12000,14000,1000)
1178 equiv <- oddsratio_to_d(1.49)
1179
1180
1181 nsims=1000
1182
1183 ## Load the checkpointing file, if it exists:
1184 checkFile <- "checkpoint_aim2a.RData"
1185 tempFile <- "tempCheckpoint_aim2a.RData"
1186 if (file.exists(checkFile)) {
1187   load(checkFile)
1188   cat("Resuming after iteration", iter, "\n")
1189   iter <- iter + 1
1190 }else{
1191   # Create some objects to make the timer work
1192   iter<- 1
1193   durs <- vector()
1194   # Create an empty DF for the output
1195   fullsim<- data.frame()
1196 }
1197
1198 if(iter <= nsims){
1199   for(iter in iter:nsims){
1200
1201
1202     message(paste0(
1203       "\nReplicate ",iter, " of ", nsims
1204     ))
1205     ptm <- proc.time()
1206
1207     # Loop over the function
1208     for(r in corrs){

```

```

1209   for(n in ns){
1210
1211       scen_temp <- sim_all_sx(corr=r,
1212                             sample_size=n,
1213                             iteration=iter,
1214                             equiv_d= equiv)
1215       fullsim <- rbind(fullsim, scen_temp)
1216       ## Save the results of the iteration. (By first saving to a temporary file
1217       ## and then renaming it into the checkpointing file, we guard against being
1218       ## interrupted while saving.)
1219       save.image(tempFile)
1220       file.rename(tempFile, checkFile)
1221   }
1222 }
1223 dur <- proc.time()-ptm
1224 durs <-c(durs,dur[3])
1225 message(paste0("\nProjected finish in ",
1226               round( mean(durs)*(nsims-iter)/60,2), " mins" ))
1227 }
1228 }
1229
1230 save(fullsim, file="./output/fullsim_power2a.RData")
1231 ## Clean up (_after_ saving the results)
1232 if (file.exists(checkFile)) file.remove(checkFile)

```

1233

Power analyses for aim 2 hypothesis/equivalence tests

(03.2_power_simulation_aim2b.R)

The purpose of this script is to run the power analyses corresponding to Aim 2 hypothesis tests. Specifically, we will:

- Simulate outcomes in four new symptom domains (anx, cd, odd, adhd) in which associations with aam are randomly drawn from a normal distribution with an incrementally increasing mean, and in which availability of 14-year data (that is, sample size for the analyses) varies (as a percentage of the observed 8-year sample size) from 55-65% in increments of 5%
- Run linear models on simulated data for 1000 iterations per scenario, deriving power empirically for the NHS+Equivalence tests of null hypotheses that effects are practically equivalent to zero (alpha = 5%)
- Run logistic regression models of aam on other diagnoses (anx, cd/odd, adhd) and test the null hypothesis that each effect is practically equivalent to zero (alpha = 5%)

Load required packages

```
library(tidyverse)
library(effectsize)
library(parameters)
library(faux)
```

Power analysis 2b

Simulation

Here we create a function which simulates both the 14 yr age at menarche (AAM) and anx/cd/odd/adhd cases, based on: 1. The effect size 2. 14-yr data availability (as a proportion of 8-year availability) 3. Prevalence of anx/cd/odd/adhd in sample ...and runs a generalized linear model and accompanying equivalence test, saving standardised betas and inferences based on decision rules in a data.frame

```
set.seed(82928)

#First, Load the real data:

sim_oth_dx <- function(effect_size_d=-0.10,
                        sample_size=13000,
                        case_rate=0.06,
                        iteration=1,
                        equiv_d=oddsratio_to_d(1.49)){

  # Mean and SD values for AaM are from 10.1192/bjp.bp.115.168617 Sequeira et al.
  # Supp table DS1

  # Convert OR to r as we are simulating the underlying continuous liability
```

```

1277 #corr <- effectsize::convert_oddsratio_to_r(or, log=F)
1278
1279 or <- d_to_oddsratio(effect_size_d)
1280
1281 simdat <- tibble("AaM" = rnorm(n = sample_size,
1282                               m = 151.52,
1283                               sd = 14.11))
1284
1285 # The new way, the downside of which is that the only "error" is in the measurement
1286 # of AaM, but since we specify beta1 for AaM as it is "measured", this is not propa
1287 gated
1288 # to the models - the net result being that we always make the same inference
1289 # in each iteration of a given scenario (not realistic)
1290
1291 # To avoid this, we will add noise to the beta1 parameter
1292
1293 # Here, we also vary the case rate, with a lower bound of 1%
1294 cr_anx<- case_rate+ rnorm(1, mean=0,sd=0.02)
1295 cr_anx<-ifelse(cr_anx<0.01,0.01,cr_anx)
1296 cr_cdodd<- case_rate+ rnorm(1, mean=0,sd=0.02)
1297 cr_cdodd<-ifelse(cr_cdodd<0.01,0.01,cr_cdodd)
1298 cr_adhd <-case_rate+ rnorm(1, mean=0,sd=0.02)
1299 cr_adhd <-ifelse(cr_adhd<0.01,0.01,cr_adhd)
1300
1301 #Anx
1302 beta0 <- log((1/(1-cr_anx))-1)
1303 beta1 <- log(or) + rnorm(1, mean=0,sd=0.05)
1304 pi_x <- exp(beta0 + beta1 * scale(simdat$AaM)) / (1 + exp(beta0 + beta1 * scale(sim
1305 dat$AaM)))
1306 simdat$"AnxDx" <- rbinom(n=length(simdat$AaM), size=1, prob=pi_x)
1307 #Conduct/ODD
1308 beta0 <- log((1/(1-cr_cdodd))-1)
1309 beta1 <- log(or) + rnorm(1, mean=0,sd=0.05)
1310 pi_x <- exp(beta0 + beta1 * scale(simdat$AaM)) / (1 + exp(beta0 + beta1 * scale(sim
1311 dat$AaM)))
1312 simdat$"CdOddDx" <- rbinom(n=length(simdat$AaM), size=1, prob=pi_x)
1313 #ADHD
1314 beta0 <- log((1/(1-cr_adhd))-1)
1315 beta1 <- log(or) + rnorm(1, mean=0,sd=0.05)
1316 pi_x <- exp(beta0 + beta1 * scale(simdat$AaM)) / (1 + exp(beta0 + beta1 * scale(sim
1317 dat$AaM)))
1318 simdat$"AdhdDx" <- rbinom(n=length(simdat$AaM), size=1, prob=pi_x)
1319
1320 simdat$AaM <- scale(simdat$AaM)
1321
1322 #Run the models
1323
1324 res <-
1325   list(glm(AnxDx~ AaM, family = binomial(link="logit"), data=simdat),
1326         glm(CdOddDx~ AaM, family = binomial(link="logit"), data=simdat),
1327         glm(AdhdDx~ AaM, family = binomial(link="logit"), data=simdat))
1328
1329 #Extract the results of the NHST
1330
1331 nhst_res <- map(res,function(x){
1332   x %>%

```

```

1333     summary() %>%
1334     .$coefficients %>%
1335     as.data.frame() %>%
1336     `row.names<-`(NULL) %>%
1337     .[,2] %>%
1338     mutate(oddsr = exp(Estimate), # Odds ratio/gradient
1339            var.diag = diag(vcov(x))[,2], # Variance of each coefficient
1340            or.se = sqrt(or^2 * var.diag)) # Odds-ratio adjusted
1341   }) %>%
1342   reduce(bind_rows) %>%
1343   rename(p=Pr(>|z|))
1344
1345   #Run the equivalence test
1346
1347   equiv_res <- map(res, function(x){
1348     x %>%
1349     parameters::equivalence_test(
1350       range= c(d_to_oddsratio(-equiv_d, log=T), d_to_oddsratio(equiv_d, log=T)),
1351       rule="classic",
1352       ci=1-(0.05/nrow(nhst_res)),
1353       p_values=T ) %>%
1354     as.data.frame() %>%
1355     `row.names<-`(NULL) %>%
1356     .[,2]
1357   }) %>%
1358   reduce(bind_rows)
1359
1360   #Combine and apply decision rules
1361
1362   comb_res <- nhst_res %>%
1363     bind_cols(equiv_res) %>%
1364     mutate(iteration = iteration,
1365            effect_size = or,
1366            sample_size=sample_size,
1367            case_rate=case_rate)
1368
1369   simres_full<- comb_res
1370
1371   row.names(simres_full)<- NULL
1372   return(simres_full)
1373 }

```

1374 We apply this function with different parameters for the second power analysis

```

1375 # Set range of parameters
1376 es <- seq(0.0, -0.26, -0.02)
1377 ns <- c(12000, 13000, 14000)
1378 rates <- c(0.02, 0.06, .10)
1379
1380 equiv=oddsratio_to_d(1.49)
1381
1382
1383 nsims=1000
1384
1385 ## Load the checkpointing file, if it exists:
1386 checkFile <- "checkpoint_aim2b.RData"
1387 tempFile <- "tempCheckpoint_aim2b.RData"

```



```

1388 if (file.exists(checkFile)) {
1389   load(checkFile)
1390   cat("Resuming after iteration", iter, "\n")
1391   iter <- iter + 1
1392 }else{
1393   # Create some objects to make the timer work
1394   iter<- 1
1395   durs <- vector()
1396   # Create an empty DF for the output
1397   fullsim<- data.frame()
1398 }
1399
1400 if(iter <= nsims){
1401   # Loop over the function
1402   for(iter in iter:nsims){
1403
1404
1405     message(paste0(
1406       "\nReplicate ",iter, " of ", nsims
1407     ))
1408     ptm <- proc.time()
1409
1410     # Loop over the function
1411     for(e in es){
1412       for(n in ns){
1413         for(c in rates){
1414
1415             scen_temp <- suppressMessages(sim_oth_dx(effect_size_d=e,
1416                                                       sample_size=n,
1417                                                       case_rate = c,
1418                                                       equiv_d=equiv,
1419                                                       iteration=iter))
1420
1421             fullsim <- rbind(fullsim, scen_temp)
1422             ## Save the results of the iteration. (By first saving to a temporary file
1423             ## and then renaming it into the checkpointing file, we guard against being
1424             ## interrupted while saving.)
1425             save.image(tempFile)
1426             file.rename(tempFile, checkFile)
1427         }
1428       }
1429     }
1430     dur <- proc.time()-ptm
1431     durs <-c(durs,dur[3])
1432     message(paste0("\nProjected finish in ", round(mean(durs)*(nsims-iter)/60,2), " m
1433 ins" ))
1434   }
1435 }
1436
1437 save(fullsim, file="./output/fullsim_power2b.RData")
1438 ## Clean up (_after_ saving the results)
1439 if (file.exists(checkFile)) file.remove(checkFile)
1440

```

1441

Power analyses for aim 2 hypothesis/equivalence tests

(03.1_power_analyses_aim2.R)

The purpose of this script is to summarise the power analyses corresponding to Aim 2 hypothesis/equivalence tests.

Load required packages

```
library(tidyverse)
library(effectsize)
```

Power analysis 2a

```
load(file="./output/fullsim_power2a.RData")
```

Power

Our power in NHST in each scenario is the proportion of iterations in which a true effect is detected, given the effect exists in the population

Our power in equivalence/inferiority each scenario is the proportion of iterations in which an effect is declared equivalent to zero, given that in the population it is less than the SESOI

```
#Summarise at the level of scenario (i.e., across iterations)
power2a <- fullsim %>%
  mutate(outcome= factor(rep(c("anx","cd","odd","adhd"), nrow(fullsim)/4 ))) %>%
  group_by(corr, sample_size,outcome) %>%
  summarise(power_nhst= mean(p < 0.05),
            se_nhst=sd(p < 0.05)/sqrt(n()),
            power_equiv= mean(ROPE_Percentage==1),
            se_equiv=sd(ROPE_Percentage==1)/sqrt(n())) %>%
  rename(effect_size= corr,N = sample_size) %>%
  mutate(effect_size=r_to_d(effect_size )) %>%
  pivot_longer(cols=matches("power|se_"), names_to = "type", values_to="power") %>%
  separate(type, into=c("est","test")) %>%
  pivot_wider(names_from=est,values_from=power) %>%
  mutate( test=factor(test,
                    levels=c("nhst","equiv"),
                    labels=c("NHST\n(H1:Effect > 0)","Equivalence\n(H1:Effect = 0)")),
         N=factor(N)) %>%
  filter(N%in%c("12000","13000"))

#Plot the power curves
ggplot(data=power2a, aes(x= effect_size, y= power, fill=N,
                        colour=N,group=N))+
  geom_errorbar(aes(ymin=power-se,ymax=power+se), width=0, size=1.2, alpha=0.4)+
  geom_point(size=2.6,
            alpha=0.4)+
  geom_line(size=0.8)+
  geom_hline(yintercept=0.95, colour="red",size=1.4, linetype=2)+
```

```

1488 geom_hline(yintercept=0.80, colour="orange",size=1.4, linetype=3, alpha=0.7)+
1489 facet_grid(test~outcome)+
1490 coord_cartesian(ylim=c(0,1))+
1491 theme(text=element_text(size=11),
1492        strip.text.y = element_text(angle=0, face="bold"),
1493        legend.position = "bottom",
1494        legend.direction= "horizontal")+
1495 scale_x_continuous("Effect size (d)")+
1496 scale_y_continuous("Power")
1497 ggsave("../output/power2a.tiff")

```

1498 We can summarise by giving a) the smallest effect size for which we have 80 and 95% power in
 1499 NHST and b) the largest effect size for which we have 80 and 95% power to declare equivalence to
 1500 zero:

```

1501 # Smallest effect size for which 80% power (NHST):
1502 power2a %>%
1503   filter(power>.8,
1504          str_detect(test,"NHST")) %>%
1505   group_by(N, outcome) %>%
1506   summarise(`min_es_80%power` = max(effect_size))%>%
1507   mutate(sesoi = rep(-oddsratio_to_d(1.49),4))
1508 # Smallest effect size for which 95% power (NHST):
1509 power2a %>%
1510   filter(power>.95,
1511          str_detect(test,"NHST")) %>%
1512   group_by(N,outcome) %>%
1513   summarise(`min_es_95%power` = max(effect_size))%>%
1514   mutate(sesoi = rep(-oddsratio_to_d(1.49),4))
1515
1516 # Largest effect size for which 80% power (equiv):
1517 power2a %>%
1518   filter(power>.8,
1519          str_detect(test,"NHST",negate = T)) %>%
1520   group_by(N,outcome) %>%
1521   summarise(`max_es_80%power` = min(effect_size))%>%
1522   mutate(sesoi = rep(-oddsratio_to_d(1.49),4))
1523 # Largest effect size for which 95% power (equiv):
1524 power2a %>%
1525   filter(power>.95,
1526          str_detect(test,"NHST",negate=T)) %>%
1527   group_by(N,outcome) %>%
1528   summarise(`max_es_95%power` = min(effect_size))%>%
1529   mutate(sesoi = rep(-oddsratio_to_d(1.49),4))

```

1530 Power analysis 2b

```
1531 load(file="../output/fullsim_power2b.RData")
```

1532 Power

1533 Our power in NHST in each scenario is the proportion of iterations in which a true effect is detected,
 1534 given the effect exists in the population

1535 Our power in equivalence/inferiority each scenario is the proportion of iterations in which an effect is
 1536 declared equivalent to zero, given that in the population it is less than the SESOI

```

1537 fullsim <- fullsim %>%
1538   rename(p = `p...4`) %>%
1539   filter(!iteration %in% c(284,589,858)) # drop specific iterations that ran incomple
1540 tely
1541                                     # due to timeout
1542
1543 #Summarise at the level of scenario (i.e., across iterations)
1544 power2b <- fullsim %>%
1545   group_by(effect_size, sample_size, case_rate) %>%
1546   rename(N = sample_size, prev=case_rate) %>%
1547   summarise(power_nhst= mean(p < 0.05),
1548             se_nhst=sd(p < 0.05)/sqrt(n()),
1549             power_equiv= mean(ROPE_Percentage == 1),
1550             se_equiv=sd(ROPE_Percentage)/sqrt(n())) %>%
1551   mutate(effect_size=oddsratio_to_d(effect_size)) %>%
1552   pivot_longer(cols=matches("power|se"), names_to = "type", values_to="power") %>%
1553   separate(type, into=c("est", "test")) %>%
1554   pivot_wider(names_from=est, values_from=power) %>%
1555   mutate( test=factor(test,
1556                  levels=c("nhst", "equiv"),
1557                  labels=c("NHST\n(H1:Effect > 0)", "Equivalence\n(H1:Effect = 0)"))),
1558          N=factor(N))%>%
1559   filter(N%in%c("12000", "13000"))
1560
1561 prevlabs <- c("2% avg. \nprevalence", "6% avg. \nprevalence", "10% avg. \nprevalence")
1562 names(prevlabs)<- c("0.02", "0.06", "0.1")
1563 #Plot the power curves
1564 ggplot(data=power2b, aes(x= effect_size, y= power, fill=N,
1565                          colour=N, group=N))+
1566   geom_errorbar(aes(ymin=power-se, ymax=power+se), width=0, size=1.2, alpha=0.4)+
1567   geom_point(size=2.6,
1568             alpha=0.4)+
1569   geom_line(size=0.8)+
1570   geom_hline(yintercept=0.95, colour="red", size=1.4, linetype=2)+
1571   geom_hline(yintercept=0.80, colour="orange", size=1.4, linetype=3, alpha=0.7)+
1572   facet_grid(test~prev, labeller = labeller(prev=prevlabs))+
1573   coord_cartesian(ylim=c(0,1))+
1574   theme(text=element_text(size=11),
1575         strip.text.y = element_text(angle=0, face="bold"),
1576         legend.position = "bottom",
1577         legend.direction= "horizontal")+
1578   scale_x_continuous("Effect size (d)")
1579   scale_y_continuous("Power")
1580   ggsave("./output/power2b.tiff")
1581

```

1582 We can summarise by giving a) the smallest effect size for which we have 80 and 95% power in
 1583 NHST and b) the largest effect size for which we have 80 and 95% power to declare equivalence to
 1584 zero:

```

1585 # Smallest effect size for which 80% power (NHST):
1586 power2b %>%
1587   filter(power>.8,
1588         str_detect(test, "NHST")) %>%
1589   group_by(N, prev) %>%
1590   summarise(`min_es_80%power` = max(effect_size))%>%
1591   mutate(sesoi = rep(-oddsratio_to_d(1.49), 3))

```

```

1592 # Smallest effect size for which 95% power (NHST):
1593 power2b %>%
1594   filter(power>.95,
1595     str_detect(test,"NHST")) %>%
1596   group_by(N,prev) %>%
1597   summarise(`min_es_95%power` = max(effect_size))%>%
1598   mutate(sesoi = rep(-oddsratio_to_d(1.49),3))
1599
1600 # Largest effect size for which 80% power (equiv):
1601 power2b %>%
1602   filter(power>.8,
1603     str_detect(test,"NHST",negate = T)) %>%
1604   group_by(N,prev) %>%
1605   summarise(`max_es_80%power` = min(effect_size))%>%
1606   mutate(sesoi = rep(-oddsratio_to_d(1.49),3))
1607 # Largest effect size for which 95% power (equiv):
1608 power2b %>%
1609   filter(power>.95,
1610     str_detect(test,"NHST",negate=T)) %>%
1611   group_by(N,prev) %>%
1612   summarise(`max_es_95%power` = min(effect_size)) %>%
1613   mutate(sesoi = rep(-oddsratio_to_d(1.49),3))
1614
1615

```

```

1616

```

Aim 3: Determine whether any associations between early age at menarche and adolescent depressive symptoms/depression are likely to be causal (04_aim3_analyses.R)

The purpose of this script is to run the primary analyses corresponding to Aim 4. Specifically, we will:

- Run a 2SLS one-sample MR with NHS+equivalence tests for causal effect of aam on dep sx
- Run a 2SLS one-sample MR with NHST+equivalence tests for causal effect of aam on earlier dep sx (negative control)
- Run a 2SLS (logistic 2nd stage) one-sample MR with NHS+equivalence tests for causal effect of aam on on dep dx

Read in the data and load packages:

```
load(file="data/analysis_dataset.RData")
library(tidyverse)
library(parameters)
library(effectsize)
library(AER)
library(modelr)
library(broom)
library(sandwich)
```

Hypothesis 3a

```
fulldata <- fulldata %>%
  mutate(across( c("smfq_dep_c_14s", "aam_c_14s", "smfq_dep_c_8yr" ), scale ))

model3a_unadj <- ivreg(smfq_dep_c_14s ~ aam_c_14s | PGS_AaM, data = fulldata) %>%
  broom::tidy(conf.int=T, conf.level=0.9) %>%
  mutate(p_one_tailed = ifelse(estimate<0,p.value/2,1),
         ROPE_low =- d_to_r(0.25)) #only ROPE_low because of directional hypothesis

model3a_adj <- ivreg(smfq_dep_c_14s ~ aam_c_14s +
  sex + age_at_q_ret_8yr + parent_income_derived_q1 +
  parent_educ_derived_q1 + parent_cohab_18m +
  parent_cohab_3yr + p_age_at_birth + m_age_at_birth +
  financ_probs_18m + scl_5item_m_q1 + scl_full_m_q3 +
  epds_full_m_6m + n_children_8yr + bmi_derived_c_8yr +
  bmi_derived_c_14s | PGS_AaM +
  sex + age_at_q_ret_8yr + parent_income_derived_q1 +
  parent_educ_derived_q1 + parent_cohab_18m +
  parent_cohab_3yr + p_age_at_birth + m_age_at_birth +
  financ_probs_18m + scl_5item_m_q1 + scl_full_m_q3 +
  epds_full_m_6m + n_children_8yr + bmi_derived_c_8yr +
  bmi_derived_c_14s, data = fulldata) %>%
  broom::tidy(conf.int=T, conf.level=0.9) %>%
  mutate(p_one_tailed = ifelse(estimate<0,p.value/2,1),
```

```

1663     ROPE_low =- d_to_r(0.25)) #only ROPE_low because of directional hypothesis
1664
1665 model3a_neg_unadj <- ivreg(smfq_dep_c_8yr ~ aam_c_14s | PGS_AaM, data = fulldata) %>%
1666 %
1667   broom::tidy(conf.int=T, conf.level=0.9) %>%
1668   mutate(ROPE_low =- d_to_r(0.25)) #only ROPE_low because of directional hypothesis
1669
1670 model3a_neg_adj <- ivreg(smfq_dep_c_8yr ~ aam_c_14s +
1671   sex + age_at_q_ret_8yr + parent_income_derived_q1 +
1672   parent_educ_derived_q1 + parent_cohab_18m +
1673   parent_cohab_3yr + p_age_at_birth + m_age_at_birth +
1674   financ_probs_18m + scl_5item_m_q1 + scl_full_m_q3 +
1675   epds_full_m_6m + n_children_8yr + bmi_derived_c_8yr +
1676   bmi_derived_c_14s | PGS_AaM +
1677   sex + age_at_q_ret_8yr + parent_income_derived_q1 +
1678   parent_educ_derived_q1 + parent_cohab_18m +
1679   parent_cohab_3yr + p_age_at_birth + m_age_at_birth +
1680   financ_probs_18m + scl_5item_m_q1 + scl_full_m_q3 +
1681   epds_full_m_6m + n_children_8yr + bmi_derived_c_8yr +
1682   bmi_derived_c_14s, data = fulldata) %>%
1683   broom::tidy(conf.int=T, conf.level=0.9) %>%
1684   mutate(ROPE_low =- d_to_r(0.25)) #only ROPE_low because of directional hypothesis

```

1685 Hypothesis 3b

```

1686 # Get PGS predicted values of AaM from first stage linear models
1687
1688 model3b_stage1_unadj <- lm(aam_c_14s ~ PGS_AaM, data = fulldata)
1689
1690 model3b_stage1_adj <- lm(aam_c_14s ~ PGS_AaM +
1691   sex + age_at_q_ret_8yr + parent_income_derived_q1 +
1692   parent_educ_derived_q1 + parent_cohab_18m +
1693   parent_cohab_3yr + p_age_at_birth + m_age_at_birth +
1694   financ_probs_18m + scl_5item_m_q1 + scl_full_m_q3 +
1695   epds_full_m_6m + n_children_8yr + bmi_derived_c_8yr +
1696   bmi_derived_c_14s , data = fulldata)
1697
1698 fulldata <- fulldata%>%
1699   add_predictions(model3b_stage1_unadj, var= "pred_unadj") %>%
1700   add_predictions(model3b_stage1_adj, var= "pred_adj")
1701
1702 # Run Logistic second stage
1703
1704 model3b_stage2_unadj <- glm(dep_dx_10_17 ~ pred_unadj,
1705   family = binomial(link="logit"),
1706   data=fulldata)
1707
1708 model3b_stage2_adj <- glm(dep_dx_10_17 ~ pred_adj +
1709   sex + age_at_q_ret_8yr + parent_income_derived_q1 +
1710   parent_educ_derived_q1 + parent_cohab_18m +
1711   parent_cohab_3yr + p_age_at_birth + m_age_at_birth +
1712   financ_probs_18m + scl_5item_m_q1 + scl_full_m_q3 +
1713   epds_full_m_6m + n_children_8yr + bmi_derived_c_8yr +
1714   bmi_derived_c_14s ,
1715   family = binomial(link="logit"),
1716   data=fulldata)
1717

```

```

1718 model3b_stage2_8yr_unadj <- glm(dep_dx_0_8 ~ pred_unadj,
1719                                family = binomial(link="logit"),
1720                                data=fulldata)
1721
1722 model3b_stage2_8yr_adj <- glm(dep_dx_0_8 ~ pred_adj +
1723                               sex + age_at_q_ret_8yr + parent_income_derived_q1 +
1724                               parent_educ_derived_q1 + parent_cohab_18m +
1725                               parent_cohab_3yr + p_age_at_birth + m_age_at_birth +
1726                               financ_probs_18m + scl_5item_m_q1 + scl_full_m_q3 +
1727                               epds_full_m_6m + n_children_8yr + bmi_derived_c_8yr +
1728                               bmi_derived_c_14s ,
1729                               family = binomial(link="logit"),
1730                               data=fulldata)
1731 # We also need to adjust the SEs to account for the uncertainty in the first stage:
1732 # Calculate robust standard errors
1733 ##(from https://stats.stackexchange.com/questions/89999/
1734 #how-to-replicate-statas-robust-binomial-glm-for-proportion-data-in-r)
1735 adjSEs <- function(x){
1736   cov.m1 <- vcovHC(x, type = "HC1")
1737   ## This is used as it mirrors stas ivreg2 robust, used in Sequeira et al)
1738   std.err <- sqrt(diag(cov.m1))
1739   q.val <- qnorm(0.95)
1740   comb_res <- cbind(
1741     Estimate = coef(x)
1742     , "Robust SE" = std.err
1743     , z = (coef(x)/std.err)
1744     , "p" = 2 * pnorm(abs(coef(x)/std.err), lower.tail = FALSE)
1745     , CI_low = coef(x) - q.val * std.err
1746     , CI_high = coef(x) + q.val * std.err
1747   ) %>%
1748   as.data.frame() %>%
1749   mutate(ROPE_low =- d_to_oddsratio(0.25, log=T))
1750   return(comb_res)
1751 }
1752
1753 model3b_stage2_unadj_res <- adjSEs(model3b_stage2_unadj) %>%
1754   mutate(p_onetailed = ifelse(Estimate<0,p/2,1))
1755 model3b_stage2_adj_res <- adjSEs(model3b_stage2_adj) %>%
1756   mutate(p_onetailed = ifelse(Estimate<0,p/2,1))
1757 model3b_stage2_8yr_unadj_res <- adjSEs(model3b_stage2_8yr_unadj) %>%
1758   mutate(p_onetailed = ifelse(Estimate<0,p/2,1))
1759 model3b_stage2_8yr_adj_res <- adjSEs(model3b_stage2_8yr_adj) %>%
1760   mutate(p_onetailed = ifelse(Estimate<0,p/2,1))

```

1761

Power analyses for aim 3(&4) hypothesis/equivalence tests

(04.1_power_simulation_aim34a.R)

The purpose of this script is to run the power analyses corresponding to Aim 3(&4) hypothesis/equivalence tests. Specifically, we will:

- Simulate a polygenic score of genome-wide significant SNPs explaining variance in (simulated) aam at different levels, with causal associations with depression symptoms simulated based on SNP-predicted values of AaM which associations with aam are randomly drawn from a normal distribution with an incrementally increasing mean, and in which availability of 14-year data (that is, genotyped sample size for the analyses) varies between 9.5k, 10.5k, and 11.5k
- Run one-sample MR (2 SLS linear models) on simulated data for 1000 iterations per scenario, deriving power empirically for the NHST/Equivalence tests (alpha = 5%) NB: we use two sets of equivalence bounds, one for depression based on the “small telescopes” procedure described in the report, and one generic one for outcomes in any other domain.
- Simulate depression diagnosis outcome and run one-sample MR (structural mean models) on simulated data for 1000 iterations per scenario, deriving power empirically for the NHST/Equivalence tests (alpha = 5%) NB: we use two sets of equivalence bounds, one for depression based on the “small telescopes” procedure described in the report, and one generic one for outcomes in any other domain.

Load required packages

```
library(tidyverse)
library(effectsize)
library(parameters)
library(faux)
library(AER)
library(modelr)
library(broom)
library(sandwich)
```

Power analysis 3/4a

Simulation

Here we create a function which simulates the age at menarche (AAM) genetic instrument and observed value, and depressive symptoms based on three inputs: 1. The explanatory power of the instrument in the exposure (aam ~ pgs) 2. The 14-yr data availability (as a proportion of 8-year availability) ...and runs a linear model and accompanying equivalence test, saving standardised betas and inferences based on decision rules in a data.frame

```
set.seed(91734)

sim_sx_mr <- function( causal_eff_d=0.25,
                        R2_inst=0.09,
                        sample_size=10500,
                        iteration=1,
```

```

1806         equiv_d_aim3=0.1,
1807         equiv_d_aim4=0.1,
1808         out_mean=5.71,
1809         out_sd=4.93){
1810
1811     # Mean and SD values are from 10.1192/bjp.bp.115.168617 Sequeira et al.
1812     # Supp table DS1
1813
1814     #Calculate b_yz based on ACE and R2_inst:
1815
1816     b_xz = sqrt(R2_inst)
1817     ace = rnorm(1, d_to_r(causal_eff_d), 0.02)
1818     b_yz = b_xz * ace
1819
1820     confound= rnorm(1, 0, d_to_r(0.05))
1821     #if(confound<0){confound=0} #Allow masking?
1822
1823     b_yx = ace + confound
1824
1825
1826     simdat <- rnorm_multi(n = sample_size,
1827                          mu = c(151.52, 0, out_mean ),
1828                          sd = c(14.11, 1, out_sd ),
1829                          r = c(b_xz, b_yx, b_yz) ,
1830                          varnames = c("AaM", "PGS_AaM", "sx"),
1831                          empirical = TRUE) %>%
1832     mutate(AaM = round(AaM/12),
1833            sx = ifelse(sx<0,0,round(sx)))
1834     #Assume a floor effect; this attenuates the correlation slightly
1835
1836     #Run the model and get 90%CIs for equivalence testing (will use bootstrapped
1837     #CIs in the analysis)
1838
1839     mr_res <- ivreg(scale(sx) ~ scale(AaM) | scale(PGS_AaM), data = simdat) %>%
1840     broom::tidy(conf.int=T, conf.level=0.9) %>%
1841     mutate(ROPE_high_aim3 = d_to_r(equiv_d_aim3),
1842            ROPE_low_aim3 =- d_to_r(equiv_d_aim3),
1843            ROPE_high_aim4 = d_to_r(equiv_d_aim4),
1844            ROPE_low_aim4 =- d_to_r(equiv_d_aim4)) %>%
1845     .[,2,]
1846
1847     #Combine and apply decision rules - inferiority (i.e., only use high equiv bound)
1848
1849     comb_res <- mr_res %>%
1850     mutate(p_one_tailed = ifelse(estimate<0,p.value/2,1), #One-tailed test p-value
1851            computation
1852            iteration = iteration,
1853            R2_inst = R2_inst,
1854            causal_eff_d=causal_eff_d,
1855            true_causal=ace,
1856            sample_size=sample_size)
1857
1858
1859     row.names(comb_res)<- NULL
1860     return(comb_res)
1861 }

```

1862 We apply this function with different parameters for the second power analysis

```

1863 # Set range of parameters
1864 r2s <- c(0.05,0.075,0.10)
1865 ns <- c(9500,10500)
1866 ces <- seq(0.0,-0.30,-0.01)
1867
1868 equiv_d_aim3=0.25
1869 equiv_d_aim4=0.20
1870
1871 nsims=1000
1872
1873 ## Load the checkpointing file, if it exists:
1874 checkFile <- "checkpoint_aim34a.RData"
1875 tempFile <- "tempCheckpoint_aim34a.RData"
1876 if (file.exists(checkFile)) {
1877   load(checkFile)
1878   cat("Resuming after iteration", iter, "\n")
1879   iter <- iter + 1
1880 }else{
1881   # Create some objects to make the timer work
1882   iter<- 1
1883   durs <- vector()
1884   # Create an empty DF for the output
1885   fullsim<- data.frame()
1886 }
1887
1888 if(iter <= nsims){
1889   # Loop over the function
1890   for(iter in iter:nsims){
1891
1892
1893     message(paste0(
1894       "\nReplicate ",iter, " of ", nsims
1895     ))
1896     ptm <- proc.time()
1897
1898     for(r2 in r2s){
1899       for(n in ns){
1900         for(ce in ces){
1901           message(paste0(
1902             "\n R2 = ",r2,
1903             "\n N = ",n,
1904             "\n CE = ",ce,
1905             "\n (replicate ",iter,")"))
1906           scen_temp <- suppressMessages(sim_sx_mr(causal_eff_d=ce,
1907                                                     R2_inst=r2,
1908                                                     sample_size=n,
1909                                                     iteration=iter,
1910                                                     equiv_d_aim3=equiv_d_aim3,
1911                                                     equiv_d_aim4=equiv_d_aim4,
1912                                                     out_mean=5.71,
1913                                                     # Mean and SD values are from 10.1192/bjp.bp.115.168617 Sequeira et al.
1914                                                     out_sd=4.93)) # Supp table DS1
1915           fullsim <- rbind(fullsim, scen_temp)
1916
1917

```

```

1918     ## Save the results of the iteration. (By first saving to a temporary file
1919 ## and then renaming it into the checkpointing file, we guard against being
1920 ## interrupted while saving.)
1921     save.image(tempFile)
1922     file.rename(tempFile, checkFile)
1923
1924   }
1925 }
1926 }
1927
1928     dur <- proc.time()-ptm
1929     durs <-c(durs,dur[3])
1930     message(paste0("\nProjected finish in ",
1931                   round( mean(durs)*(nsims-iter)/60,2), " mins" ))
1932 }
1933
1934 }
1935
1936
1937 save(fullsim, file="./output/fullsim_power34a.RData")
1938 ## Clean up (_after_ saving the results)
1939 if (file.exists(checkFile)) file.remove(checkFile)
1940
1941

```

1942


```

1993         varnames = c("AaM", "PGS_AaM"),
1994         empirical = TRUE) %>%
1995     mutate(AaM = round(AaM/12))
1996
1997     # Note that the possibility of AaM values > 168 represents the situation
1998     # after we have imputed right censored values
1999
2000     # Get PGS predicted values of AaM
2001
2002     res <- lm(scale(AaM) ~ PGS_AaM, data = simdat)
2003
2004     simdat <- simdat %>%
2005         add_predictions(res) %>%
2006         add_residuals(res)
2007
2008
2009     # Simulate the dx variable
2010     beta0 <- log((1/(1-case_rate))-1)
2011
2012
2013     beta1 <- b_yx
2014     pi_x <- exp(beta0 + beta1 * scale(simdat$AaM)) / (1 + exp(beta0 + beta1 * scale(
2015     simdat$AaM)))
2016     simdat$`dx` <- rbinom(n=length(simdat$AaM), size=1, prob=pi_x)
2017     obs_case_rate <- prop.table(table(simdat$dx))[[2]]
2018
2019     #Run the model both ways, adjust SEs and get 90%CIs for equivalence testing (will
2020     use bootstrapped
2021     #CIs in the analysis)
2022
2023     mr_res_logs2 <- glm(dx ~ pred, family = binomial(link="logit"), data=simdat)
2024
2025     mr_res_ivreg <- ivreg(dx ~ scale(AaM)|PGS_AaM, data=simdat)
2026
2027     #And we also need to adjust the SEs to account for the uncertainty in the first s
2028     tage:
2029
2030     # Calculate robust standard errors (from https://stats.stackexchange.com/
2031     # questions/89999/how-to-replicate-stata-robust-binomial-glm-for-proportion-data
2032     -in-r)
2033
2034     adjSEs <- function(x){
2035         cov.m1 <- vcovHC(x, type = "HC1")
2036         ## This is used as it mirrors stata ivreg2 robust, used in Sequeira et al)
2037         std.err <- sqrt(diag(cov.m1))
2038         q.val <- qnorm(0.95)
2039         comb_res <- cbind(
2040             Estimate = coef(x)
2041             , "Robust SE" = std.err
2042             , z = (coef(x)/std.err)
2043             , "p" = 2 * pnorm(abs(coef(x)/std.err), lower.tail = FALSE)
2044             , CI_low = coef(x) - q.val * std.err
2045             , CI_high = coef(x) + q.val * std.err
2046         ) %>%
2047         as.data.frame() %>%
2048         .[,] %>%

```

```

2049     mutate(ROPE_high_aim3 = d_to_oddsratio(equiv_d_aim3, log=T),
2050            ROPE_low_aim3 = d_to_oddsratio(equiv_d_aim3, log=T),
2051            ROPE_high_aim4 = d_to_oddsratio(equiv_d_aim4, log=T),
2052            ROPE_low_aim4 = d_to_oddsratio(equiv_d_aim4, log=T))
2053   return(comb_res)
2054 }
2055
2056 convert_ivreg_out <- function(x){
2057   y_x0 = coef(mr_res_ivreg)[[1]]
2058   y_x1 = coef(mr_res_ivreg)[[1]] + x
2059   odds <- log((y_x1 * (1-y_x0)) / ((1-y_x1)* y_x0))
2060   return(odds)
2061 }
2062
2063 mr_log_res <- adjSEs(mr_res_logs2) %>%
2064   mutate(method="logistic_2nd_stage")
2065 mr_ivr_res <- adjSEs(mr_res_ivreg) %>%
2066   mutate(across(c("Estimate", "CI_low", "CI_high"), ~convert_ivreg_out(.x)),
2067          method="standard_2SLS")
2068
2069
2070 #Combine and apply decision rules - inferiority, i.e., only use high bound
2071
2072 comb_res <- mr_log_res %>%
2073   bind_rows(mr_ivr_res) %>%
2074   mutate(p_onetailed = ifelse(Estimate<0,p/2,1), #One-tailed test p-value computa
2075 tion
2076         iteration = iteration,
2077         r2_inst = R2_inst,
2078         effect_size = d_to_oddsratio(causal_eff_d),
2079         est_COR = exp(Estimate),
2080         est_COR_lci90 = exp(CI_low),
2081         est_COR_uci90 = exp(CI_high),
2082         sample_size=sample_size,
2083         case_rate=case_rate,
2084         obs_case_rate=obs_case_rate)
2085
2086
2087 simres_full<- comb_res
2088
2089
2090 row.names(simres_full)<- NULL
2091 return(simres_full)
2092 }

```

2093 We apply this function with different parameters for the second power analysis

```

2094 # Set range of parameters
2095 r2s <- c(0.05,0.075,0.10) # start, end, increment
2096 ns <- c(9500,10500)
2097 ces <- seq(0.0,-0.30,-0.02)
2098 rates <- c(0.02,0.06,.10,.14)
2099
2100 equiv_d_aim3=0.25
2101 equiv_d_aim4=0.20
2102
2103 nsims = 1000

```

```

2104
2105 ## Load the checkpointing file, if it exists:
2106 checkFile <- "checkpoint_aim34b.RData"
2107 tempFile <- "tempCheckpoint_aim34b.RData"
2108 if (file.exists(checkFile)) {
2109   load(checkFile)
2110   cat("Resuming after iteration", iter, "\n")
2111   iter <- iter + 1
2112 }else{
2113   # Create some objects to make the timer work
2114   iter<- 1
2115   durs <- vector()
2116   # Create an empty DF for the output
2117   fullsim<- data.frame()
2118 }
2119
2120 if(iter <= nsims){
2121   # Loop over the function
2122   for(iter in iter:nsims){
2123
2124
2125     message(paste0(
2126       "\nReplicate ",iter, " of ", nsims
2127     ))
2128     ptm <- proc.time()
2129
2130     # Loop over the function
2131     for(r2 in r2s){
2132       for(n in ns){
2133         for(ce in ces){
2134           for(rate in rates){
2135             message(paste0(
2136               "\n R2 = ",r2,
2137               "\n N = ",n,
2138               "\n CE = ",ce,
2139               "\n rate = ",rate,
2140               "\n (replicate ",iter,")"))
2141             scen_temp <- sim_dx_mr(causal_eff_d=ce,
2142                                   R2_inst=r2,
2143                                   sample_size=n,
2144                                   iteration=iter,
2145                                   equiv_d_aim3=equiv_d_aim3,
2146                                   equiv_d_aim4=equiv_d_aim4,
2147                                   case_rate= rate )
2148             fullsim <- rbind(fullsim, scen_temp)
2149
2150
2151             ## Save the results of the iteration. (By first saving to a temporary fi
2152 Le
2153 ## and then renaming it into the checkpointing file, we guard against bei
2154 ng
2155 ## interrupted while saving.)
2156             save.image(tempFile)
2157             file.rename(tempFile, checkFile)
2158           }
2159         }

```



```

2160     }
2161   }
2162
2163   dur <- proc.time()-ptm
2164   durs <-c(durs,dur[3])
2165   message(paste0("\nProjected finish in ",
2166                 round( mean(durs)*(nsims-iter)/60,2), " mins" ))
2167 }
2168
2169 }
2170
2171
2172 save(fullsim, file="./output/fullsim_power34b.RData")
2173 ## Clean up (_after_ saving the results)
2174 if (file.exists(checkFile)) file.remove(checkFile)
2175

```

2176

Power analyses for aim 3(&4) hypothesis/equivalence tests

(04.3_power_summary_aim3.R)

The purpose of this script is to summarise the power analyses corresponding to Aim 3 hypothesis/equivalence tests.

Load required packages

```
library(tidyverse)
library(effectsize)
library(parameters)

load(file="./output/fullsim_power34a.RData")
```

Power

Our power in NHST in each scenario is the proportion of iterations in which a true effect is detected, given the effect exists in the population

Our power in equivalence/inferiority each scenario is the proportion of iterations in which an effect is declared equivalent to zero, given that in the population it is less than the SESOI

```
#Summarise at the level of scenario (i.e., across iterations)
power3a <- fullsim %>%
  group_by(causal_eff_d, R2_inst, sample_size) %>%
  summarise(power_nhst= mean(p_one_tailed < 0.05),
            se_nhst=sd(p_one_tailed < 0.05)/sqrt(n()),
            power_equiv= mean(conf.low > ROPE_low_aim3),
            se_equiv=sd(conf.low > ROPE_low_aim3)/sqrt(n())) %>%
  ## In these lines, add "&CI Low> ROPE_Low" for full equivalence test
  rename(effect_size= causal_eff_d, N = sample_size) %>%
  pivot_longer(cols=matches("power|se_"), names_to = "type", values_to="power") %>%
  separate(type, into=c("est", "test")) %>%
  pivot_wider(names_from=est, values_from=power) %>%
  mutate( test=factor(test,
                      levels=c("nhst", "equiv"),
                      labels=c("NHST\n(H1:Effect > 0)", "Equivalence\n(H1:Effect = 0)"))),
         N=factor(N))%>%
  filter(N%in%c("9500", "10500"))

r2labs <- c("R2 inst. 5%", "R2 inst. 7.5%", "R2 inst. 10%")
names(r2labs)<- c("0.05", "0.075", "0.1")
#Plot the power curves
ggplot(data=power3a, aes(x= effect_size, y= power, fill=N,
                        colour=N, group=N))+
  geom_errorbar(aes(ymin=power-se, ymax=power+se), width=0, size=1.2, alpha=0.4)+
  geom_point(size=2.6,
            alpha=0.4)+
  geom_line(size=0.8)+
  geom_hline(yintercept=0.95, colour="red", size=1.4, linetype=2)+
  geom_hline(yintercept=0.80, colour="orange", size=1.4, linetype=3, alpha=0.7)+
```

```

2225 facet_grid(test~R2_inst, labeller = labeller(R2_inst=r2labs))+
2226 coord_cartesian(ylim=c(0,1))+
2227 theme(text=element_text(size=11),
2228        strip.text.y = element_text(angle=0, face="bold"),
2229        legend.position = "bottom",
2230        legend.direction= "horizontal")+
2231 scale_x_continuous("Effect size (d)")+
2232 scale_y_continuous("Power")
2233 ggsave("./output/power3a.tiff")

```

2234 We can summarise by giving a) the smallest effect size for which we have 80 and 95% power in
 2235 NHST and b) the largest effect size for which we have 80 and 95% power to declare equivalence to
 2236 zero:

```

2237 # Smallest effect size for which 80% power (NHST):
2238 power3a %>%
2239   filter(power>.8,
2240          str_detect(test,"NHST")) %>%
2241   group_by(N, R2_inst) %>%
2242   summarise(`min_es_80%power` = max(effect_size))%>%
2243   ungroup()
2244 print(paste0("sesoi = ",-r_to_d(unique(fullsim$ROPE_high_aim3))))
2245 # Smallest effect size for which 95% power (NHST):
2246 power3a %>%
2247   filter(power>.95,
2248          str_detect(test,"NHST")) %>%
2249   group_by(N, R2_inst) %>%
2250   summarise(`min_es_95%power` = max(effect_size))%>%
2251   ungroup()
2252 print(paste0("sesoi = ",-r_to_d(unique(fullsim$ROPE_high_aim3))))
2253
2254 # Largest effect size for which 80% power (equiv):
2255 power3a %>%
2256   filter(power>.8,
2257          str_detect(test,"NHST",negate=T)) %>%
2258   group_by(N, R2_inst) %>%
2259   summarise(`max_es_80%power` = min(effect_size))%>%
2260   ungroup()
2261 print(paste0("sesoi = ",-r_to_d(unique(fullsim$ROPE_high_aim3))))
2262 # Largest effect size for which 95% power (equiv):
2263 power3a %>%
2264   filter(power>.95,
2265          str_detect(test,"NHST",negate=T)) %>%
2266   group_by(N, R2_inst) %>%
2267   summarise(`max_es_95%power` = min(effect_size))%>%
2268   ungroup()
2269 print(paste0("sesoi = ",-r_to_d(unique(fullsim$ROPE_high_aim3))))
2270
2271 #

```

2272 Power analysis 3/4b

```

2273 load(file="./output/fullsim_power34b.RData")
2274 max(fullsim$iteration) # This simulation takes a long time so may have fewer iterations
2275
2276 # if the walltime runs out

```

2277 **Power**

2278 Our power in NHST in each scenario is the proportion of iterations in which a true effect is detected,
 2279 given the effect exists in the population

2280 Our power in equivalence/inferiority each scenario is the proportion of iterations in which an effect is
 2281 declared equivalent to zero, given that in the population it is less than the SESOI

```

2282 #First, compare the two MR approaches
2283 methodcomp <- fullsim %>%
2284   group_by(effect_size,r2_inst, sample_size, case_rate, method) %>%
2285   summarise(bias = mean(est_COR-effect_size, na.rm=T),
2286             mse = mean(est_COR-effect_size,na.rm=T)^2,
2287             coverage = mean(effect_size > est_COR_lci90 & effect_size < est_COR_uci90
2288 ,na.rm=T),
2289             precision = mean(est_COR_uci90 - est_COR_lci90, na.rm =T))
2290
2291 methodcomp %>%
2292   group_by(method,case_rate) %>%
2293   summarise(bias = mean(bias),
2294             mse = mean(mse),
2295             coverage = mean(coverage),
2296             precision = mean(precision) )
2297
2298 ## Logistic 2nd stage seems to be less biased, have lower mse and equivalent coverage
2299 methodcomp %>%
2300   group_by(method,sample_size) %>%
2301   summarise(bias = mean(bias),
2302             mse = mean(mse),
2303             coverage = mean(coverage),
2304             precision = mean(precision) )
2305 methodcomp %>%
2306   group_by(method,effect_size) %>%
2307   summarise(bias = mean(bias),
2308             mse = mean(mse),
2309             coverage = mean(coverage),
2310             precision = mean(precision) )
2311
2312 #Summarise at the level of scenario (i.e., across iterations)
2313 power3b <- fullsim %>%
2314   filter(method=="logistic_2nd_stage") %>%
2315   group_by(effect_size,r2_inst, sample_size, case_rate) %>%
2316   summarise(power_nhst= mean(p_onetailed < 0.05),
2317             se_nhst=sd(p_onetailed < 0.05)/sqrt(n()),
2318             power_equiv= mean(CI_low > ROPE_low_aim3),
2319             se_equiv=sd(CI_low > ROPE_low_aim3)/sqrt(n())) %>%
2320   mutate(effect_size=oddsratio_to_d(effect_size) ) %>%
2321   rename(N = sample_size, prev=case_rate) %>%
2322   pivot_longer(cols=matches("power|se_"), names_to = "type", values_to="power") %>%
2323   separate(type, into=c("est","test")) %>%
2324   pivot_wider(names_from=est,values_from=power) %>%
2325   mutate( test=factor(test,
2326                     levels=c("nhst","equiv"),
2327                     labels=c("NHST\n(H1:Effect > 0)","Equivalence\n(H1:Effect = 0)"
2328 )),
2329           N=factor(N),
2330           r2_inst=as.character(r2_inst))

```

```

2331
2332 r2labs <- c("R2 inst. 5%", "R2 inst. 7.5%", "R2 inst. 10%")
2333 names(r2labs) <- c("0.05", "0.075", "0.1")
2334
2335 #Plot the power curves
2336 ggplot(data=power3b, aes(x= effect_size, y= power, fill=factor(prev),
2337                          colour=factor(prev), group=interaction(N,prev), shape=N))+
2338   geom_errorbar(aes(ymin=power-se, ymax=power+se), width=0, size=1.2, alpha=0.4)+
2339   geom_point(size=2.6,
2340             alpha=0.4)+
2341   geom_line(size=0.8)+
2342   geom_hline(yintercept=0.95, colour="red", size=1.4, linetype=2)+
2343   geom_hline(yintercept=0.80, colour="orange", size=1.4, linetype=3, alpha=0.7)+
2344   facet_grid(test~r2_inst, labeller = labeller(r2_inst= r2labs) )+
2345   coord_cartesian(ylim=c(0,1))+
2346   theme(text=element_text(size=11),
2347         strip.text.y = element_text(angle=0, face="bold"),
2348         legend.position = "bottom",
2349         legend.direction= "horizontal")+
2350   scale_x_continuous("Effect size (d)")+
2351   scale_y_continuous("Power")
2352 ggsave("./output/power3b.tiff")

```

2353 We can summarise by giving a) the smallest effect size for which we have 80 and 95% power in
 2354 NHST and b) the largest effect size for which we have 80 and 95% power to declare equivalence to
 2355 zero:

```

2356 # Smallest effect size for which 80% power (NHST):
2357 power3b %>%
2358   filter(power>.8,
2359          str_detect(test, "NHST")) %>%
2360   group_by(N, r2_inst, prev) %>%
2361   summarise(`min_es_80%power` = max(effect_size))%>%
2362   ungroup()
2363 print(paste0("sesoi = ", -oddsratio_to_d(exp(unique(fullsim$ROPE_high_aim3)))))
2364 # Smallest effect size for which 95% power (NHST):
2365 power3b %>%
2366   filter(power>.95,
2367          str_detect(test, "NHST")) %>%
2368   group_by(N, r2_inst, prev) %>%
2369   summarise(`min_es_95%power` = max(effect_size))%>%
2370   ungroup()
2371 print(paste0("sesoi = ", -oddsratio_to_d(exp(unique(fullsim$ROPE_high_aim3)))))
2372
2373 # Largest effect size for which 80% power (equiv):
2374 power3b %>%
2375   filter(power>.8,
2376          str_detect(test, "NHST", negate=T)) %>%
2377   group_by(N, r2_inst, prev) %>%
2378   summarise(`max_es_80%power` = min(effect_size))%>%
2379   ungroup()
2380 print(paste0("sesoi = ", -oddsratio_to_d(exp(unique(fullsim$ROPE_high_aim3)))))
2381 # Largest effect size for which 95% power (equiv):
2382 power3b %>%
2383   filter(power>.95,
2384          str_detect(test, "NHST", negate=T)) %>%
2385   group_by(N, r2_inst, prev) %>%

```

```
2386   summarise(`max_es_95%power` = min(effect_size))%>%  
2387   ungroup()  
2388  
2389   print(paste0("sesoi = ", -oddsratio_to_d(exp(unique(fullsim$ROPE_high_aim3)))))  
  
2390
```

Aim 4: Determine whether any associations between early age at menarche and adolescent symptoms/diagnoses in other domains are likely to be causal (05_aim4_analyses.R)

The purpose of this script is to run the primary analyses corresponding to Aim 4. Specifically, we will:

- Run a 2SLS one-sample MR with NHS+equivalence tests for causal effect of aam on sx
- Run a 2SLS one-sample MR with NHST+equivalence tests for causal effect of aam on earlier sx (negative control)
- Run a 2SLS (logistic 2nd stage) one-sample MR with NHS+equivalence tests for causal effect of aam on on all dx variables

Read in the data and load packages:

```
load(file="data/analysis_dataset.RData")
library(tidyverse)
library(parameters)
library(effectsize)
library(AER)
library(modelr)
library(broom)
library(sandwich)
```

Hypotheses 4.1-4a

List the symptom variables

```
domains <- list(anx=c("scared_anx_c_8yr", "scared_anx_c_14s"),
               adhd = c("rsdbd_adhd_c_8yr", "rsdbd_adhd_c_14m"),
               cd=c("rsdbd_cd_c_8yr", "rsdbd_cd_c_14s"),
               odd=c("rsdbd_odd_c_8yr", "rsdbd_odd_c_14m"))
```

Scale for standardised effects

```
fulldata <- fulldata %>%
  mutate(across( c("smfq_dep_c_14s", "aam_c_14s", "smfq_dep_c_8yr", unlist(domains) ), scale ))
```

Run models for all domains and save results in a list

```
domains_res <- list()
n=0
for(domain in domains){
  n=n+1

  model4a_unadj <- ivreg(get(domain[[2]]) ~ aam_c_14s | PGS_AaM, data = fulldata) %>%
  broom::tidy(conf.int=T, conf.level=0.9) %>%
  mutate(ROPE_high= d_to_r(0.2),
         ROPE_low =- d_to_r(0.2))
```

```

2437 model4a_adj <- ivreg(get(domain[[2]]) ~ aam_c_14s +
2438   sex + age_at_q_ret_8yr + parent_income_derived_q1 +
2439   parent_educ_derived_q1 + parent_cohab_18m +
2440   parent_cohab_3yr + p_age_at_birth + m_age_at_birth +
2441   financ_probs_18m + scl_5item_m_q1 + scl_full_m_q3 +
2442   epds_full_m_6m + n_children_8yr + bmi_derived_c_8yr +
2443   bmi_derived_c_14s | PGS_AaM +
2444   sex + age_at_q_ret_8yr + parent_income_derived_q1 +
2445   parent_educ_derived_q1 + parent_cohab_18m +
2446   parent_cohab_3yr + p_age_at_birth + m_age_at_birth +
2447   financ_probs_18m + scl_5item_m_q1 + scl_full_m_q3 +
2448   epds_full_m_6m + n_children_8yr + bmi_derived_c_8yr +
2449   bmi_derived_c_14s, data = fulldata) %>%
2450   broom::tidy(conf.int=T, conf.level=0.9) %>%
2451   mutate(ROPE_high= d_to_r(0.2),
2452     ROPE_low =- d_to_r(0.2))
2453
2454 model4a_neg_unadj <- ivreg(get(domain[[1]]) ~ aam_c_14s | PGS_AaM, data = fulldata
2455 ) %>%
2456   broom::tidy(conf.int=T, conf.level=0.9) %>%
2457   mutate(ROPE_high= d_to_r(0.2),
2458     ROPE_low =- d_to_r(0.2))
2459
2460 model4a_neg_adj <- ivreg(get(domain[[1]]) ~ aam_c_14s +
2461   sex + age_at_q_ret_8yr + parent_income_derived_q1 +
2462   parent_educ_derived_q1 + parent_cohab_18m +
2463   parent_cohab_3yr + p_age_at_birth + m_age_at_birth +
2464   financ_probs_18m + scl_5item_m_q1 + scl_full_m_q3 +
2465   epds_full_m_6m + n_children_8yr + bmi_derived_c_8yr +
2466   bmi_derived_c_14s | PGS_AaM +
2467   sex + age_at_q_ret_8yr + parent_income_derived_q1 +
2468   parent_educ_derived_q1 + parent_cohab_18m +
2469   parent_cohab_3yr + p_age_at_birth + m_age_at_birth +
2470   financ_probs_18m + scl_5item_m_q1 + scl_full_m_q3 +
2471   epds_full_m_6m + n_children_8yr + bmi_derived_c_8yr +
2472   bmi_derived_c_14s, data = fulldata) %>%
2473   broom::tidy(conf.int=T, conf.level=0.9) %>%
2474   mutate(ROPE_high= d_to_r(0.2),
2475     ROPE_low =- d_to_r(0.2))
2476
2477 all_res <-list (model4a_unadj,model4a_adj,model4a_neg_unadj,model4a_neg_adj)
2478
2479 domains_res[[names(domains)[[n]]]] <- all_res
2480
2481 }

```

Hypotheses 4.1-3b

Get PGS predicted values of AaM from first stage linear models

```

2485 model4b_stage1_unadj <-lm(aam_c_14s ~ PGS_AaM, data = fulldata)
2486
2487 model4b_stage1_adj <-lm(aam_c_14s ~ PGS_AaM +
2488   sex + age_at_q_ret_8yr + parent_income_derived_q1 +
2489   parent_educ_derived_q1 + parent_cohab_18m +
2490   parent_cohab_3yr + p_age_at_birth + m_age_at_birth +
2491   financ_probs_18m + scl_5item_m_q1 + scl_full_m_q3 +

```



```

2492         epds_full_m_6m + n_children_8yr + bmi_derived_c_8yr +
2493         bmi_derived_c_14s , data = fulldata)
2494
2495 fulldata <- fulldata%>%
2496   add_predictions(model3b_stage1_unadj, var= "pred_unadj") %>%
2497   add_predictions(model3b_stage1_adj, var= "pred_adj")
2498
2499 # Run Logistic second stage
2500
2501 # List the symptom variables
2502 domains <-list(anx=c("anx_dx_0_8","anx_dx_10_17"),
2503               adhd = c("adhd_dx_0_8","adhd_dx_10_17"),
2504               cdodd=c("cdodd_dx_0_8","cdodd_dx_10_17"))
2505
2506 # Run models for all domains and save results in a list
2507 domains_res <- list()
2508 n=0
2509 for(domain in domains){
2510   n=n+1
2511
2512   model4b_stage2_unadj <- glm(get(domain[[2]]) ~ pred_unadj,
2513                             family = binomial(link="logit"),
2514                             data=fulldata)
2515
2516   model4b_stage2_adj <- glm(get(domain[[2]]) ~ pred_adj +
2517                             sex + age_at_q_ret_8yr + parent_income_derived_q1 +
2518                             parent_educ_derived_q1 + parent_cohab_18m +
2519                             parent_cohab_3yr + p_age_at_birth + m_age_at_birth +
2520                             financ_probs_18m + scl_5item_m_q1 + scl_full_m_q3 +
2521                             epds_full_m_6m + n_children_8yr + bmi_derived_c_8yr +
2522                             bmi_derived_c_14s ,
2523                             family = binomial(link="logit"),
2524                             data=fulldata)
2525
2526   model4b_stage2_8yr_unadj <- glm(get(domain[[1]]) ~ pred_unadj,
2527                                   family = binomial(link="logit"),
2528                                   data=fulldata)
2529
2530   model4b_stage2_8yr_adj <- glm(get(domain[[1]]) ~ pred_adj +
2531                                   sex + age_at_q_ret_8yr + parent_income_derived_q1 +
2532                                   parent_educ_derived_q1 + parent_cohab_18m +
2533                                   parent_cohab_3yr + p_age_at_birth + m_age_at_birth
2534   +
2535                                   financ_probs_18m + scl_5item_m_q1 + scl_full_m_q3 +
2536                                   epds_full_m_6m + n_children_8yr + bmi_derived_c_8yr
2537   +
2538                                   bmi_derived_c_14s ,
2539                                   family = binomial(link="logit"),
2540                                   data=fulldata)
2541
2542   # We also need to adjust the SEs to account for the uncertainty in the first stage:
2543   # Calculate robust standard errors (from https://stats.stackexchange.com/
2544   #questions/89999/how-to-replicate-statas-robust-binomial-glm-for-proportion-data-in
2545   -r)
2546   adjSEs <- function(x){
2547     cov.m1 <- vcovHC(x, type = "HC1")

```

```

2548 ## This is used as it mirrors stata's ivreg2 robust, used in Sequeira et al)
2549 std.err <- sqrt(diag(cov.m1))
2550 q.val <- qnorm(0.95)
2551 comb_res <- cbind(
2552   Estimate = coef(x)
2553   , "Robust SE" = std.err
2554   , z = (coef(x)/std.err)
2555   , "p"= 2 * pnorm(abs(coef(x)/std.err), lower.tail = FALSE)
2556   , CI_low = coef(x) - q.val * std.err
2557   , CI_high = coef(x) + q.val * std.err
2558 ) %>%
2559   as.data.frame() %>%
2560   mutate(ROPE_low =- d_to_oddsratio(0.25, log=T))
2561 return(comb_res)
2562 }
2563
2564 all_res= list (adjSEs(model4b_stage2_unadj),
2565               adjSEs(model4b_stage2_adj),
2566               adjSEs(model4b_stage2_8yr_unadj),
2567               adjSEs(model4b_stage2_8yrs_adj))
2568 domains_res[[names(domains)[[n]]]] <- all_res
2569
2570 }

```

2571

Summary of power analyses for aim 4 hypothesis/equivalence tests

(05.1_power_summary_aim4.R)

The purpose of this script is to summarise the power analyses corresponding to Aim 4 hypothesis/equivalence tests.

Load required packages

```
library(tidyverse)
library(effectsize)
library(parameters)

load(file="./output/fullsim_power34a.RData")
```

Power

Our power in NHST in each scenario is the proportion of iterations in which a true effect is detected, given the effect exists in the population

Our power in equivalence/inferiority each scenario is the proportion of iterations in which an effect is declared equivalent to zero, given that in the population it is less than the SESOI

```
#Summarise at the level of scenario (i.e., across iterations)
power4a <- fullsim %>%
  group_by(causal_eff_d,R2_inst, sample_size) %>%
  summarise(power_nhst= mean(p.value < 0.05),
            se_nhst=sd(p.value < 0.05)/sqrt(n()),
            power_equiv= mean(conf.low > ROPE_low_aim4 & conf.high < ROPE_high_aim4
  ),
            se_equiv=sd(conf.low > ROPE_low_aim4 & conf.high < ROPE_high_aim4)/sqrt(
n())) %>%
  rename(effect_size= causal_eff_d,N = sample_size) %>%
  pivot_longer(cols=matches("power|se_"), names_to = "type", values_to="power") %>%
  separate(type, into=c("est","test")) %>%
  pivot_wider(names_from=est,values_from=power) %>%
  mutate( test=factor(test,
                    levels=c("nhst","equiv"),
                    labels=c("NHST\n(H1:Effect > 0)","Equivalence\n(H1:Effect = 0)"
  )),
         N=factor(N))%>%
  filter(N%in%c("9500","10500"))

r2labs <- c("R2 inst. 5%","R2 inst. 7.5%","R2 inst. 10%")
names(r2labs)<- c("0.05","0.075", "0.1")
#Plot the power curves
ggplot(data=power4a, aes(x= effect_size, y= power, fill=N,
                        colour=N,group=N))+
  geom_errorbar(aes(ymin=power-se,ymax=power+se), width=0, size=1.2, alpha=0.4)+
  geom_point(size=2.6,
            alpha=0.4)+
  geom_line(size=0.8)+
```

```

2620 geom_hline(yintercept=0.95, colour="red",size=1.4, linetype=2)+
2621 geom_hline(yintercept=0.80, colour="orange",size=1.4, linetype=3, alpha=0.7)+
2622 facet_grid(test~R2_inst, labeller = labeller(R2_inst=r2labs))+
2623 coord_cartesian(ylim=c(0,1))+
2624 theme(text=element_text(size=11),
2625       strip.text.y = element_text(angle=0, face="bold"),
2626       legend.position = "bottom",
2627       legend.direction= "horizontal")+
2628 scale_x_continuous("Effect size (d)")+
2629 scale_y_continuous("Power")
2630 ggsave("../output/power4a.tiff")

```

2631 We can summarise by giving a) the smallest effect size for which we have 80 and 95% power in
 2632 NHST and b) the largest effect size for which we have 80 and 95% power to declare equivalence to
 2633 zero:

```

2634 # Smallest effect size for which 80% power (NHST):
2635 power4a %>%
2636   filter(power>.8,
2637         str_detect(test,"NHST")) %>%
2638   group_by(N, R2_inst) %>%
2639   summarise(`min_es_80%power` = max(effect_size))%>%
2640   ungroup()
2641 print(paste0("sesoi = ",-r_to_d(unique(fullsim$ROPE_high_aim4))))
2642 # Smallest effect size for which 95% power (NHST):
2643 power4a %>%
2644   filter(power>.95,
2645         str_detect(test,"NHST")) %>%
2646   group_by(N, R2_inst) %>%
2647   summarise(`min_es_95%power` = max(effect_size))%>%
2648   ungroup()
2649 print(paste0("sesoi = ",-r_to_d(unique(fullsim$ROPE_high_aim4))))
2650
2651 # Largest effect size for which 80% power (equiv):
2652 power4a %>%
2653   filter(power>.8,
2654         str_detect(test,"NHST",negate=T)) %>%
2655   group_by(N, R2_inst) %>%
2656   summarise(`max_es_80%power` = min(effect_size))%>%
2657   ungroup()
2658 print(paste0("sesoi = ",-r_to_d(unique(fullsim$ROPE_high_aim4))))
2659 # Largest effect size for which 95% power (equiv):
2660 power4a %>%
2661   filter(power>.95,
2662         str_detect(test,"NHST",negate=T)) %>%
2663   group_by(N, R2_inst) %>%
2664   summarise(`max_es_95%power` = min(effect_size))%>%
2665   ungroup()
2666 print(paste0("sesoi = ",-r_to_d(unique(fullsim$ROPE_high_aim4))))
2667
2668 #

```

2669 Power analysis 3/4b

```

2670 load(file="../output/fullsim_power34b.RData")

```

2671 **Power**

2672 Our power in NHST in each scenario is the proportion of iterations in which a true effect is detected,
 2673 given the effect exists in the population

2674 Our power in equivalence/inferiority each scenario is the proportion of iterations in which an effect is
 2675 declared equivalent to zero, given that in the population it is less than the SESOI

```

2676 #First, compare the two MR approaches
2677 methodcomp <- fullsim %>%
2678   group_by(effect_size,r2_inst, sample_size, case_rate, method) %>%
2679   summarise(bias = mean(est_COR-effect_size, na.rm=T),
2680             mse = mean(est_COR-effect_size,na.rm=T)^2,
2681             coverage = mean(effect_size > est_COR_lci90 & effect_size < est_COR_uci90
2682 ,na.rm=T),
2683             precision = mean(est_COR_uci90 - est_COR_lci90, na.rm =T))
2684
2685 methodcomp %>%
2686   group_by(method,case_rate) %>%
2687   summarise(bias = mean(bias),
2688             mse = mean(mse),
2689             coverage = mean(coverage),
2690             precision = mean(precision) )
2691
2692 ## Logistic 2nd stage seems to be less biased, have lower mse and equivalent coverage
2693 methodcomp %>%
2694   group_by(method,sample_size) %>%
2695   summarise(bias = mean(bias),
2696             mse = mean(mse),
2697             coverage = mean(coverage),
2698             precision = mean(precision) )
2699 methodcomp %>%
2700   group_by(method,effect_size) %>%
2701   summarise(bias = mean(bias),
2702             mse = mean(mse),
2703             coverage = mean(coverage),
2704             precision = mean(precision) )
2705
2706 #Summarise at the level of scenario (i.e., across iterations)
2707 power4b <- fullsim %>%
2708   filter(method=="logistic_2nd_stage") %>%
2709   group_by(effect_size,r2_inst, sample_size, case_rate) %>%
2710   summarise(power_nhst= mean(p < 0.05),
2711             se_nhst=sd(p < 0.05)/sqrt(n()),
2712             power_equiv= mean(CI_low > ROPE_low_aim4 & CI_high < ROPE_high_aim4, na.
2713 rm=T),
2714             se_equiv=sd(CI_low > ROPE_low_aim4 & CI_high < ROPE_high_aim4, na.rm=T)/
2715 sqrt(n())) %>%
2716   mutate(effect_size=oddsratio_to_d(effect_size) ) %>%
2717   rename(N = sample_size, prev=case_rate) %>%
2718   pivot_longer(cols=matches("power|se_"), names_to = "type", values_to="power") %>%
2719   separate(type, into=c("est","test")) %>%
2720   pivot_wider(names_from=est,values_from=power) %>%
2721   mutate( test=factor(test,
2722                     levels=c("nhst","equiv"),
2723                     labels=c("NHST\n(H1:Effect > 0)","Equivalence\n(H1:Effect = 0)"
2724 ))),

```

```

2725     N=factor(N),
2726     r2_inst=as.character(r2_inst))
2727
2728 r2labs <- c("R2 inst. 5%", "R2 inst. 7.5%", "R2 inst. 10%")
2729 names(r2labs) <- c("0.05", "0.075", "0.1")
2730
2731 #Plot the power curves
2732 ggplot(data=power4b, aes(x= effect_size, y= power, fill=factor(prev),
2733                          colour=factor(prev), group=interaction(N,prev), shape=N))+
2734   geom_errorbar(aes(ymin=power-se, ymax=power+se), width=0, size=1.2, alpha=0.4)+
2735   geom_point(size=2.6,
2736             alpha=0.4)+
2737   geom_line(size=0.8)+
2738   geom_hline(yintercept=0.95, colour="red", size=1.4, linetype=2)+
2739   geom_hline(yintercept=0.80, colour="orange", size=1.4, linetype=3, alpha=0.7)+
2740   facet_grid(test~r2_inst, labeller = labeller(r2_inst= r2labs) )+
2741   coord_cartesian(ylim=c(0,1))+
2742   theme(text=element_text(size=11),
2743         strip.text.y = element_text(angle=0, face="bold"),
2744         legend.position = "bottom",
2745         legend.direction= "horizontal")+
2746   scale_x_continuous("Effect size (d)")+
2747   scale_y_continuous("Power")
2748 ggsave("./output/power4b.tiff")

```

2749 We can summarise by giving a) the smallest effect size for which we have 80 and 95% power in
 2750 NHST and b) the largest effect size for which we have 80 and 95% power to declare equivalence to
 2751 zero:

```

2752 # Smallest effect size for which 80% power (NHST):
2753 power4b %>%
2754   filter(power>.8,
2755          str_detect(test, "NHST")) %>%
2756   group_by(N, r2_inst, prev) %>%
2757   summarise(`min_es_80%power` = max(effect_size)) %>%
2758   ungroup()
2759 print(paste0("sesoi = ", -oddsratio_to_d(exp(unique(fullsim$ROPE_high_aim4)))))
2760 # Smallest effect size for which 95% power (NHST):
2761 power4b %>%
2762   filter(power>.95,
2763          str_detect(test, "NHST")) %>%
2764   group_by(N, r2_inst, prev) %>%
2765   summarise(`min_es_95%power` = max(effect_size)) %>%
2766   ungroup()
2767 print(paste0("sesoi = ", -oddsratio_to_d(exp(unique(fullsim$ROPE_high_aim4)))))
2768
2769 # Largest effect size for which 80% power (equiv):
2770 power4b %>%
2771   filter(power>.8,
2772          str_detect(test, "NHST", negate=T)) %>%
2773   group_by(N, r2_inst, prev) %>%
2774   summarise(`max_es_80%power` = min(effect_size)) %>%
2775   ungroup()
2776 print(paste0("sesoi = ", -oddsratio_to_d(exp(unique(fullsim$ROPE_high_aim4)))))
2777 # Largest effect size for which 95% power (equiv):
2778 power4b %>%
2779   filter(power>.95,

```

```
2780       str_detect(test, "NHST", negate=T)) %>%
2781     group_by(N, r2_inst, prev) %>%
2782     summarise(`max_es_95%power` = min(effect_size))%>%
2783     ungroup()
2784
2785 print(paste0("sesoi = ", -oddsratio_to_d(exp(unique(fullsim$ROPE_high_aim4)))))
```

```
2786
```

2787 Appendix 1: Psychometric properties of scales

2788 Load libraries

```
2789 library(phenotools)
2790 library(psych)
2791 library(tidyverse)
```

2792 Curate data

```
2793 mylistdata <- curate_dataset(variables_required=
2794                             c("rsdbd_adhd_c_8yr", "KJONN",
2795                               "rsdbd_cd_c_8yr", "rsdbd_odd_c_8yr",
2796                               "smfq_dep_c_8yr", "scared_anx_c_8yr"),
2797                             pheno_data_root_dir="N:/data/durable/data/MoBaPhenoData/
2798 PDB2306_MoBa_V12/SPSS/",
2799                             PDB="2306",
2800                             moba_data_version=12,
2801                             completion_threshold=0.5,
2802                             return_items=TRUE,
2803                             consistent_items = FALSE,
2804                             out_format="list")
```

2805 Separate items and scales

```
2806 myitemdata <- mylistdata$moba$items
2807 myscaledata <- mylistdata$moba$scales
```

2808 Filter by sex (2 = female)

```
2809 myitemdata <- myitemdata %>%
2810   filter(KJONN_raw==2)
```

2811 Create lists by scales

```
2812 adhd <- select(myitemdata, 24, 25, 26, 27, 28, 29, 30, 31, 32,
2813                33, 34, 35, 36, 37, 38, 39, 40, 41)
2814 cd <- select(myitemdata, 50, 51, 52, 53, 54, 55, 56, 57)
2815 odd <- select(myitemdata, 66, 67, 68, 69, 70, 71, 72, 73)
2816 dep <- select(myitemdata, 97, 98, 99, 100, 101, 102, 103,
2817                104, 105, 106, 107, 108, 109)
2818 anx <- select(myitemdata, 79, 80, 81, 82, 83)
```

2819 Create function for ordinal data

```
2820 ordinal_alpha <- function(x){
2821   psych::alpha(psych::polychoric(x)$rho)
2822 }
```

2823 Calculate Cronbach's alpha (N ~ 20,000)

```
2824 ordinal_alpha(adhd)
2825 ordinal_alpha(cd)
2826 ordinal_alpha(odd)
2827 ordinal_alpha(dep)
2828 ordinal_alpha(anx)
```


Appendix 2: Meta-analysis of the association between age at menarche and depressive symptoms

Overview

To determine the smallest effect size of interest (SESOI) for hypothesis 1a, we performed a meta-analysis of the association between age at menarche and depressive symptoms in early-to-mid adolescence. After screening all studies included in the meta-analysis by Ullsperger & Nikolas (2017), we included 5 studies of adolescents where this association was reported with sufficient detail:

- Joinson, C., Heron, J., Araya, R. & Lewis, G. Early menarche and depressive symptoms from adolescence to young adulthood in a UK cohort. *J. Am. Acad. Child Adolesc. Psychiatry* **52**, 591–598 (2013)
- Black, S. R. & Klein, D. N. Early menarcheal age and risk for later depressive symptomatology: the role of childhood depressive symptoms. *J. Youth Adolesc.* **41**, 1142–1150 (2012).
- Stice, E., Presnell, K. & Bearman, S. K. Relation of early menarche to depression, eating disorders, substance abuse, and comorbid psychopathology among adolescent girls. *Dev. Psychol.* **37**, 608–619 (2001).
- Lien, L., Haavet, O. R. & Dalgard, F. Do mental health and behavioural problems of early menarche persist into late adolescence? A three year follow-up study among adolescent girls in Oslo, Norway. *Soc. Sci. Med.* **71**, 529–533 (2010).
- Kaltiala-Heino, R., Marttunen, M., Rantanen, P. & Rimpelä, M. Early puberty is associated with mental health problems in middle adolescence. *Soc. Sci. Med.* **57**, 1055–1064 (2003).

The code for the meta-analysis is below. Our approach was inspired by [Baldwin et al.](#)

Load libraries

```
library(effects) # for effect size conversions
library(compute.es) # for effect size conversions
library(metafor) # for meta-analysis
```

Extract data from studies

Joinson et al. (2013)

Estimates were taken from Table 2, early menarche compared to late menarche at age 14 (whole sample, unadjusted).

```
OR <- 1.94
OR_low_CI <- 1.39
OR_up_CI <- 2.71
log_OR <- log(OR) # convert odds ratios to Log odds
log_SE <- ( log(OR_up_CI) - log(OR_low_CI) ) / 3.92 # convert odds ratio SE to Log odds SE
var <- log_SE^2 # derive sampling variance
id <- "joinson"
N1 <- 449 # depression at age 14 based on table 1
N2 <- 2843 - N1 # no depression

# transform Log odds ratios to standardized beta/correlation coefficients
conversion <- lores(log_OR, var, N1, N2, dig=20, verbose=FALSE)
B <- conversion$r
B_var <- conversion$var.r
```

```

2874
2875 # put extracted data into dataframe
2876 joinson <- data.frame(B, B_var, id)
2877 joinson

```

2878 Black & Klein (2012)

2879 Estimates were taken from Table 4, age at menarche onset predicting postpubertal depressive
 2880 symptoms at age 13 (adjusted, since unadjusted was not reported).

```

2881 b <- 0.37
2882 b_se <- 0.09
2883 B <- 0.14
2884 B_se <- B*(b_se/b) # derive SE
2885 B_var <- B_se^2 # derive sampling variance
2886 id <- "black_klein"
2887
2888 # put extracted data into dataframe
2889 black_klein <- data.frame(B, B_var, id)
2890 black_klein

```

2891 Stice et al. (2001)

2892 Estimates were taken from Table 3, early menarche compared to nonearly menarche, at age 13
 2893 (unadjusted).

```

2894 B <- 0.14
2895 b <- 1.61
2896 b_low_ci <- 0.61
2897 b_up_ci <- 2.62
2898 B_low_ci <- B*(b_low_ci/b)
2899 B_up_ci <- B*(b_up_ci/b)
2900 B_var <- ( (B_up_ci - B_low_ci) / 3.92 )^2 # derive sampling variance
2901 id <- "stice"
2902
2903 # put extracted data into dataframe
2904 stice <- data.frame(B, B_var, id)
2905 stice

```

2906 Lien et al. (2010)

2907 Estimates were taken from Table 4, time of menarche below 11 years, assessed at age 15
 2908 (unadjusted).

```

2909 OR <- 2.5
2910 OR_low_CI <- 1.6
2911 OR_up_CI <- 3.9
2912 log_OR <- log(OR) # convert odds ratios to log odds
2913 log_SE <- ( log(OR_up_CI) - log(OR_low_CI) ) / 3.92 # convert odds ratio SE to log odds SE
2914
2915 var <- log_SE^2 # derive sampling variance
2916 id <- "lien"
2917 N1 <- 353 # depression at age 15 based on table 1
2918 N2 <- 1013 # no depression at age 15 based on table 1
2919
2920 # transform log odds ratios to standardized beta/correlation coefficients
2921 conversion <- lores(log_OR, var, N1, N2, dig=20, verbose=FALSE)

```

```

2922 B <- conversion$r
2923 B_var <- conversion$var.r
2924
2925 # put extracted data into dataframe
2926 lien <- data.frame(B, B_var, id)
2927 lien

```

2928 Kaltiala-Heino et al. (2003)

2929 Early age at menarche (AAM) was defined by authors as 11 years or lower. They reported results for
 2930 those with AAM 10 years or lower and 11 years separately. Therefore, we combined the 10 years or
 2931 lower and 11 years estimates. Estimates were derived based on values reported in Table 1
 2932 (frequency distribution of adolescents by year of menarche), Table 2 (prevalence of depression), and
 2933 Table 3 (prevalence of depression in girls according to age at menarche), comparing AAM at 11
 2934 years and lower to AAM at age 15, assessed at age 15.

```

2935 # derive numbers in those with AAM 11yrs or less
2936 n10less <- 19196*0.031
2937 n11 <- 19196*0.157
2938
2939 n11less <- n10less+n11
2940
2941 ndep11less <- n11*0.156 + n10less*.206
2942
2943 # derive numbers in reference group
2944 nref <- 19196*0.021
2945 ndepref <- nref*0.089
2946
2947 # odds of depression in those with AAM 11yrs or less
2948 probdep11less <- ndep11less/n11less
2949 probNdep11less <- 1-probdep11less
2950
2951 odds <- probdep11less / probNdep11less
2952
2953 oddsref <- 0.089/(1-0.089)
2954
2955 or <- odds/oddsref
2956
2957 # calculate SE
2958 se <- sqrt(1/ndep11less+
2959           1/(n11less-ndep11less)+
2960           1/ndepref+
2961           1/(nref-ndepref))
2962
2963 # calculate CIs
2964 lci = or - 1.96*se
2965 uci = or + 1.96*se
2966
2967 # extract data derived based on the above calculations
2968 OR <- or
2969 OR_low_CI <- lci
2970 OR_up_CI <- uci
2971 log_OR <- log(OR) # convert odds ratios to log odds
2972 log_SE <- ( log(OR_up_CI) - log(OR_low_CI) ) / 3.92 # convert odds ratio SE to log odds SE
2973
2974 var <- log_SE^2 # derive sampling variance

```

```

2975 id <- "kaltiala"
2976 N1 <- 2567 # depression based on table 2
2977 N2 <- 16529 # no depression based on table 2
2978
2979 # transform log odds ratios to standardized beta/correlation coefficients
2980 conversion <- lores(log_OR, var, N1, N2, dig=20, verbose=FALSE)
2981 B <- conversion$r
2982 B_var <- conversion$var.r
2983
2984 # put extracted data into dataframe
2985 kaltiala <- data.frame(B, B_var, id)
2986 kaltiala

```

2987 Merge effect sizes

```

2988 all_data <- list(joinson, black_klein, stice, lien, kaltiala)
2989 combined <- Reduce(function(...) merge(..., all=T), all_data)
2990 combined

```

2991 Perform meta-analysis

```

2992 res <- rma.mv(B, B_var, data = combined)
2993 res

```

2994 Convert effect size to OR

2995 First, convert meta-analytic effect size to odds ratio (OR) using a function based on equations
 2996 presented [here](#).

```

2997 beta_to_or <- function(beta) {
2998   d <- (2*beta) / sqrt ( 1 - (beta^2) )
2999   log_odds <- d * (3.14159 / sqrt(3))
3000   or <- exp(log_odds)
3001   return(or)
3002 }
3003
3004 beta_to_or(c(res$beta, res$ci.lb, res$ci.ub))

```

3005 Convert effect size to Cohen's D

3006 Then convert effect size from OR to D.

```

3007 oddsratio_to_d(c(1.657245, 1.511574, 1.818713))

```

3008 Decide equivalence bounds

3009 **This gives a meta-analytic estimate of $D = 0.28(0.23-0.33)$.** According to best practice
 3010 recommendations, the smallest effect size of interest is determined based on the lower confidence
 3011 interval of this meta-analytic estimate ($D = 0.23$). The equivalence bounds are based on this value.

3012 **The lower and upper equivalence bounds are therefore Cohen's D -0.23 - 0.23.**