9. Appendices

Contents

A Appendix: List of Abbreviations **2**

B Appendix: Bootstrapping **3**

B.1 Bootstrapping - General Method 3

B.2 3

C Appendix: Simulation Code **3**

C.1 Generating Data and Models 3

C.2 Testing Generation of Data and Models 14

C.3 Summary Table 24

D Appendix: R Packages Used **28**

A Appendix: List of Abbreviations

CI confidence interval
IV instrumental variable
InSIDE Instrument Strength Independent of Direct Effect
MR Mendelian randomisation
RCT randomised-controlled trial
SD standard deviation
SE standard error
WME weighted median estimator

B Appendix: Bootstrapping

B.1 Bootstrapping - General Method

The typical process for “bootstrap” generating an estimate, standard error (SE) and confidence interval (CI)s of a population parameter (e.g. population mean μ) from a sample x is as follows[?]:

1. A sample x of n individuals is selected from a total population X of N individuals
2. This sample x is then treated as the “bootstrap population”; the empirical distribution of values in the n individuals in the bootstrap population is taken to be broadly representative of the distribution of values in the underlying population X of N individuals
3. A “bootstrap sample” x^* is then obtained by sampling individuals from the bootstrap population with replacement n times per bootstrap sample, i.e. the new bootstrap sample comprises n sampled individuals $x_1^*, x_2^*, ...x_n^*$. As such, individuals from the original bootstrap population x may contribute once, more than once or not at all to each bootstrap sample x^*.
4. A total of k bootstrap samples are generated $x^{*1}, x^{*2}, ...x^{*k}$, and the statistic of interest (e.g. sample mean $\bar{x}$) is estimated in each $\bar{x}^{*1}, \bar{x}^{*2}, ...\bar{x}^{*k}$
5. The set of k statistics are combined to form a “bootstrap distribution”; as expected from central limit theorem[?], this is typically closer to a normal distribution than the underlying distribution of values in either the bootstrap population x or the total population X. (See Figure 1 for an example of this)
6. The final values are derived as follows:

- the parameter estimate (e.g. $\hat{\mu}$) is taken as the mean of the bootstrap distribution of k estimates $(\sum_{i=1}^{k} \bar{x}) \div n$
- the CI s are taken as the values at the appropriate centiles at the edges of the sampling distribution, e.g. a 95% CI would be generated using values at the 2.5th and 97.5th centiles
- the SE of the estimate is taken as the standard deviation (SD) of the sampling distribution

B.2

C Appendix: Simulation Code

C.1 Generating Data and Models

The data generating model used was from Appendix 3 of Bowden et al[2]; the relevant section describing their model is reproduced below:

“...

$$U_i = \sum_{j=1}^{J} \phi_j G_{ij} + \epsilon_i^U \tag{1}$$

$$X_i = \sum_{j=1}^{J} \gamma_j G_{ij} + U_i + \epsilon_i^X \tag{2}$$

$$Y_i = \sum_{j=1}^{J} \alpha_j G_{ij} + \beta X_i + U_i + \epsilon_i^Y \tag{3}$$

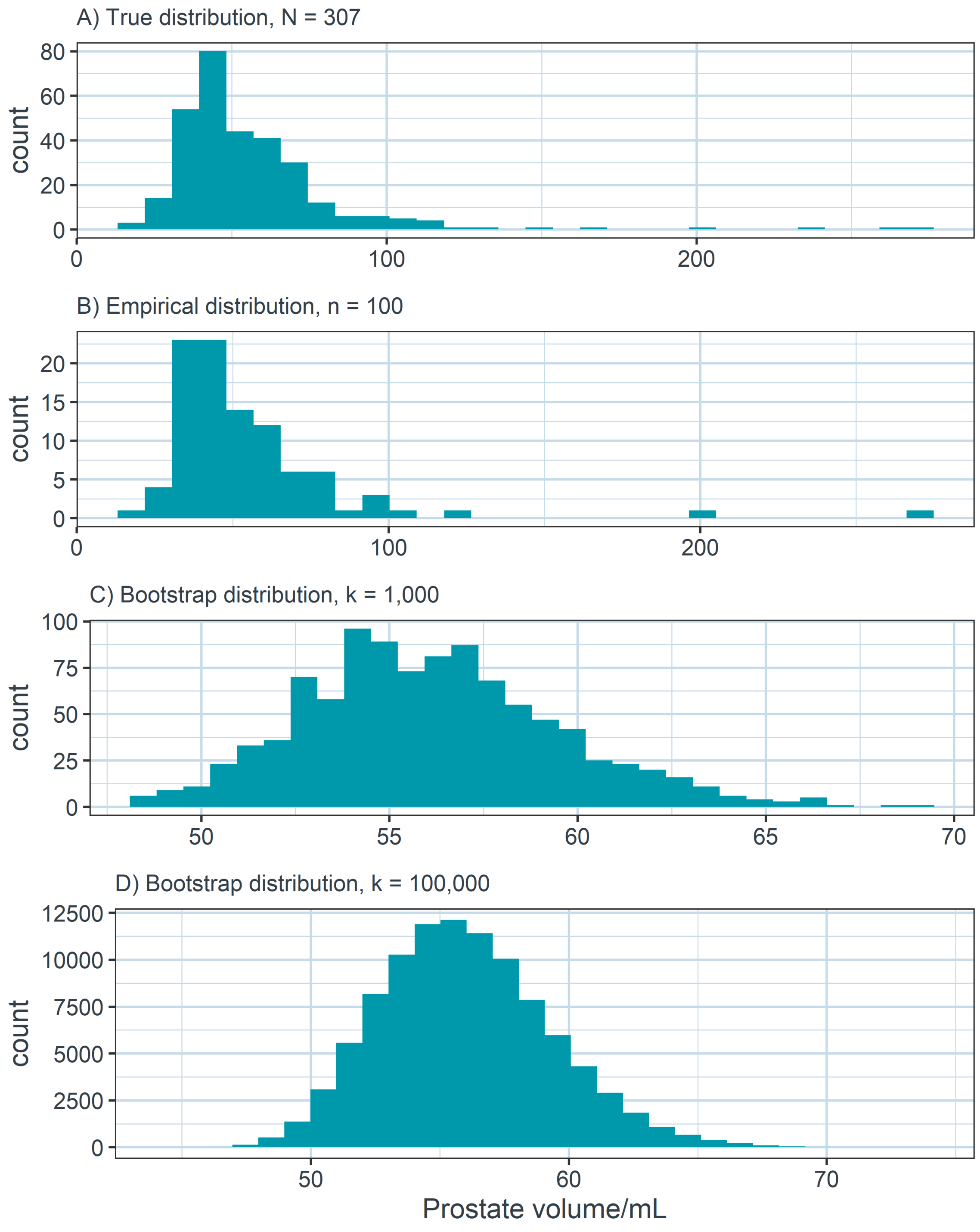


Figure 1: Density histograms demonstrating distribution of prostate volumes in patients with prostatic cancer, taken from Cata et al 2011? via the R package `medicaldata`[1]. A) Distribution from whole study population of 307 patients with non-missing data, exhibiting right-skew. B) Distribution from random sample of 100 patients, still exhibiting right-skew. C) Bootstrap distribution generated by re-sampling 1,000 bootstrap samples from the original sample of 100 patients, right-skew less apparent. D) Bootstrap distribution generated by re-sampling 100,000 bootstrap samples from the original sample of 100 patients, approaching normality.

for participants indexed by $i = 1, ..., N$, and genetic instruments indexed by $j = 1, ..., J$.

The error terms ϵ_i^U, ϵ_i^X and ϵ_i^Y were each drawn independently from standard normal distributions. The genetic effects on the exposure j are drawn from a uniform distribution between 0.03 and 0.1. Pleiotropic effects α_j and ϕ_j were set to zero if the genetic instrument was a valid instrumental variable. Otherwise (with probability 0.1, 0.2, or 0.3):

1. In Scenario 1 (balanced pleiotropy, InSIDE satisfied), the α_j parameter was drawn from a uniform distribution between −0.2 and 0.2.

2. In Scenario 2 (directional pleiotropy, InSIDE satisfied), the α_j parameter was drawn from a uniform distribution between 0 and 0.2.

3. In Scenario 3 (directional pleiotropy, InSIDE not satisfied), the ϕ_j parameter was drawn from a uniform distribution between −0.2 and 0.2.

The causal effect of the exposure on the outcome was either $\beta X = 0$ (null causal effect) or $\beta X = 0.1$ (positive causal effect). A total of 10 000 simulated datasets were generated for sample sizes of N = 10 000 and 20 [sic] participants. Only the summary data, that is genetic associations with the exposure and with the outcome and their standard errors as estimated by univariate regression on the genetic instruments in turn, were used by the analysis methods. In the two-sample setting, data were generated on 2N participants, and genetic associations with the exposure were estimated in the first N participants, and genetic associations with the outcome in the second N participants."[2]

To reproduce this model, code was written in R to generate the relevant participant level data. First, a function (`get_simulated_MR_data`) was written which included parameters specified by Bowden et al, and also to allow testing of data simulation:

```r
# Define function to create data generating model
# Arguments/default values based on Bowden et al
get_simulated_MR_data <- function(n_participants = as.integer(),
                                  n_instruments = as.integer(),
                                  n_datasets = as.integer(),
                                  prop_invalid = 0.1,
                                  causal_effect = TRUE,
                                  balanced_pleio = TRUE,
                                  InSIDE_satisfied = TRUE,
                                  rand_error = TRUE,      # remove random errors, for testing
                                  two_sample = TRUE,      # 1- or 2-sample MR toggle, for testing
                                  beta_val = 0.1,         # size of causal effect
                                  allele_freq_min = 0.4,  # frequency of effect allele 0.01/0.99
                                  allele_freq_max = 0.6,  #?0.4/0.6
                                  gamma_min = 0.03,       # size of pleiotropic effects on exposure
                                  gamma_max = 0.1,
                                  alpha_min = -0.2,       # size of pleiotropic effects on outcome
                                  alpha_max = 0.2,
                                  phi_min = -0.2,         # size of additional pleiotropic effects
                                  phi_max = 0.2,          # when InSIDE not satisfied
                                  seed = 14101583){       # Set seed for reproducibility

  # Set seed to ensure comparability across scenarios
```

```r
set.seed(seed)

# Initialise blank lists to receive datasets for
# each of:
#     U (vector: unmeasured confounding exposures per participant),
#     X (vector: exposure:outcome associations estimated per participant)
#     Y (vector: gene:outcome association estimated per participant),
#     G (Matrices: Genotype data)
#
#     gamma (vector: pleiotropic effects of each instrument on exposure)
#     alpha (vector: pleiotropic effects of each instrument on outcome)
#     phi (vector: additional pleiotropic effects of each instrument when InSIDE
#     assumption not satisfied)
U_list <- list()
X_list <- list()
Y_list <- list()
G_X_list <- list()
G_Y_list <- list()

gamma_list <- list()
alpha_list <- list()
phi_list <- list()


n_participants_list <- list()
n_instruments_list <- list()
prop_invalid_list <- list()
beta_val_list <- list()



# --- Assign features common to all datasets --- #

# size of causal effect
beta <- if_else(causal_effect == TRUE,
                beta_val,
                0)

# create vector of participant indices for 1st n participants
# i.e. participants used for estimating gene:exposure coefficient
sample_1_ref <- 1:n_participants


# Default is to estimate gene:outcome coefficient from different sample
# to gene:exposure coefficient (i.e. simulating 2-sample MR)
# two_sample == FALSE toggles to single sample for testing simulation
ifelse(two_sample == FALSE,
       sample_2_ref <- sample_1_ref, # 1 sample MR
       sample_2_ref <- (n_participants+1):(2*n_participants)) # 2 sample MR

# --- Set characteristics for each genetic instrument --- #

# Set genetic effects of each instrument on the exposure,
# drawn from uniform distribution, min/max as per Bowden
# et al
gamma_vect <- runif(n = n_instruments,
                    min = gamma_min,
                    max = gamma_max)



# Set which instruments invalid, 0 = valid, 1 = invalid
invalid_instrument_vect <- rbinom(n = n_instruments,
                                  size = 1,
                                  prob = prop_invalid)


# Probability of effect allele set per dataset
# for each instrument, default value set at
# random between 0.01-0.99 (i.e. both effect +
# reference are common alleles)
allele_freq_vect <- runif(n = n_instruments,
                          min = allele_freq_min,
                          max = allele_freq_max)

# Set pleiotropic effects on outcome, Scenarios and
# min/max from Bowden et al
alpha_vect <- double() # Pleiotropic effects of instruments on outcome
phi_vect <- double() # Pleiotropic effects of confounders on outcome


for(j in 1:n_instruments){
  ifelse(invalid_instrument_vect[j] == 0, # alpha = 0 if valid
         alpha_vect[j] <- 0,
         ifelse(balanced_pleio == TRUE,
                alpha_vect[j] <- runif(n = 1, # balanced
                                       min = alpha_min,
                                       max = alpha_max),
                alpha_vect[j] <- runif(n = 1, # directional
                                       min = 0,
                                       max = alpha_max)
         )
  )


  # Assign default phi = 0 unless directional pleiotropy &
  # InSIDE assumption not satisfied & genetic instrument invalid
  if(balanced_pleio == FALSE & InSIDE_satisfied == FALSE){
    ifelse(invalid_instrument_vect[j] == 0,
           phi_vect[j] <- 0,
           phi_vect[j] <- runif(n = 1,
                                min = phi_min,
                                max = phi_max)
    )

  }
  else{
    phi_vect[j] <- 0
  }
}

# Re-set seed to ensure consistency across datasets
# N.B. above two if/ifelse statements cause de-sync
# of number of randomised functions between valid/invalid
set.seed(seed)

# --- Create separate datasets --- #

# Create N datasets by simulating genotype matrices with
# 1 row per participant, 1 column per genetic instrument
# Use these to estimate U, X + Y

for(n in 1:n_datasets){

  # --- Create matrix of genotypes --- #

  # Assign genotypes by sampling from binomial distribution
  # twice (as two alleles) per participant with probability
  # equal to frequency of effect allele
  # Create twice as many genotypes as participants in sample
  # to simulate 2 sample MR, i.e. first half used to estimate
  # Gene:Exposure, second half used to estimate Gene:Outcome

  # Matrix where columns are instruments, rows are participants
  # Values 0, 1 or 2
  # 0 = reference, i.e. zero effect alleles,
  # 1 = 1 effect allele, 2 = 2 effect alleles

  G_mat <- matrix(rbinom(n = 2 * n_participants * n_instruments,
                         size = 2,
                         prob = rep(allele_freq_vect, 2 * n_participants)),
                  nrow = 2 * n_participants,
                  ncol = n_instruments,
                  byrow = TRUE)


  # Create error terms for U, X + Y per participant,
  # each drawn from standard normal distribution
  # unless random error turned off (for testing)

  ifelse(rand_error == TRUE,
         U_epsilon_vect <- rnorm(n = 2 * n_participants),
         U_epsilon_vect <- rep(0, 2 * n_participants))

  ifelse(rand_error == TRUE,
         X_epsilon_vect <- rnorm(n = n_participants),
         X_epsilon_vect <- rep(0, n_participants))

  ifelse(rand_error == TRUE,
         Y_epsilon_vect <- rnorm(n = n_participants),
         Y_epsilon_vect <- rep(0, n_participants))


  # --- Combine Gene matrix/parameters to recreate model --- #

  # Create vectors of estimates for U, X and Y per individual,
  # i.e. Ui, Xi and Yi. Uses matrix inner product operator " %*%"
  # https://stackoverflow.com/questions/22060515/the-r-operator
  # http://matrixmultiplication.xyz/

  #     U (vector: unmeasured confounding exposures per participant),
  #     X (vector: exposure:outcome associations estimated per participant)
  #     Y (vector: gene:outcome association estimated per participant)

  Ui_vect <-  G_mat %*% phi_vect + U_epsilon_vect

  Xi_vect <-  G_mat[sample_1_ref, ] %*% gamma_vect +
    Ui_vect[sample_1_ref, ] +
    X_epsilon_vect

  Yi_vect <-  G_mat[sample_2_ref, ] %*% alpha_vect +
    beta * Xi_vect +
    Ui_vect[sample_2_ref, ] +
    Y_epsilon_vect


  # Add vectors of estimates from this dataset to lists of
  # estimates from all datasets
  U_list[[n]] <- Ui_vect

  X_list[[n]] <- Xi_vect

  Y_list[[n]] <- Yi_vect

  G_X_list[[n]] <- G_mat[sample_1_ref, ]

  G_Y_list[[n]] <- G_mat[sample_2_ref, ]


  # Include actual parameter values generated for simulation
  alpha_list[[n]] <- alpha_vect

  gamma_list[[n]] <- gamma_vect

  phi_list[[n]] <- phi_vect

  # Include inputs for reference/testing
  n_participants_list[[n]] <- n_participants
  n_instruments_list[[n]] <- n_instruments
  prop_invalid_list[[n]] <- prop_invalid
  beta_val_list[[n]] <- beta_val
```

```
  }



  # --- Combine all outputs to return --- #

  combined_list <- list(U = U_list,         # Estimates
                        X = X_list,
                        Y = Y_list,

                        G_X = G_X_list,     # Genotypes of 1st sample
                        G_Y = G_Y_list,     # Genotypes of 2nd sample

                        alpha = alpha_list, # Actual values for validating simulation
                        gamma = gamma_list,
                        phi = phi_list,

                        n_participants = n_participants_list, # Inputs
                        n_instruments = n_instruments_list,
                        prop_invalid = prop_invalid_list,
                        beta_val = beta_val_list
  )

  return(combined_list)
}
```

This initial simulation function generated data in the following format:

```
# Check data produced in expected format
#set.seed(1701)
test_data_sim <- get_simulated_MR_data(n_participants = 1000,
                                       n_instruments = 25,
                                       n_datasets = 2,
                                       prop_invalid = 0.3,
                                       rand_error = FALSE,
                                       causal_effect = TRUE,
                                       balanced_pleio = TRUE,
                                       InSIDE_satisfied = TRUE)

str(test_data_sim)
```

```
## List of 12
##  $ U             :List of 2
##   ..$ : num [1:2000, 1] 0 0 0 0 0 0 0 0 0 0 ...
##   ..$ : num [1:2000, 1] 0 0 0 0 0 0 0 0 0 0 ...
##  $ X             :List of 2
##   ..$ : num [1:1000, 1] 1.59 1.29 1.02 1.89 1.49 ...
##   ..$ : num [1:1000, 1] 1.85 1.84 2.18 1.16 1.44 ...
##  $ Y             :List of 2
##   ..$ : num [1:1000, 1] 0.0704 0.1351 0.1589 0.0944 0.0161 ...
##   ..$ : num [1:1000, 1] 0.59017 0.10039 0.00743 0.27896 0.27746 ...
##  $ G_X           :List of 2
##   ..$ : int [1:1000, 1:25] 2 0 1 2 1 0 1 0 0 2 ...
##   ..$ : int [1:1000, 1:25] 0 1 2 1 1 1 0 2 0 2 ...
##  $ G_Y           :List of 2
##   ..$ : int [1:1000, 1:25] 1 1 0 1 2 0 1 0 1 2 ...
##   ..$ : int [1:1000, 1:25] 1 1 1 1 0 1 2 0 2 0 ...
##  $ alpha         :List of 2
##   ..$ : num [1:25] 0 0 0.1157 0 -0.0634 ...
##   ..$ : num [1:25] 0 0 0.1157 0 -0.0634 ...
##  $ gamma         :List of 2
##   ..$ : num [1:25] 0.0938 0.0808 0.0755 0.0342 0.0443 ...
##   ..$ : num [1:25] 0.0938 0.0808 0.0755 0.0342 0.0443 ...
##  $ phi           :List of 2
##   ..$ : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##   ..$ : num [1:25] 0 0 0 0 0 0 0 0 0 0 ...
##  $ n_participants:List of 2
##   ..$ : num 1000
##   ..$ : num 1000
##  $ n_instruments :List of 2
##   ..$ : num 25
##   ..$ : num 25
##  $ prop_invalid  :List of 2
##   ..$ : num 0.3
##   ..$ : num 0.3
##  $ beta_val      :List of 2
##   ..$ : num 0.1
##   ..$ : num 0.1
```

A function (`get_models`) was then written to create linear models from each dataset generated as per Bowden et al:

```r
# Create plotting tibble with Mean/SD X + Y grouped by
# Dataset + instrument
get_models <- function(sim){

  output_list <- list()

  # Create linear models per dataset to get coefficients
  # for gene:exposure association (coeff_G_X) and gene:outcome
  # association (coeff_G_Y)
  for(dataset in 1:length(sim$X)){

    X <- sim$X[[dataset]]
    Y <- sim$Y[[dataset]]
    Instruments_X <- sim$G_X[[dataset]]
    Instruments_Y <- sim$G_Y[[dataset]]

    alpha <- sim$alpha[[dataset]]
    gamma <- sim$gamma[[dataset]]
    phi <- sim$phi[[dataset]]
    beta <- sim$beta_val[[dataset]]
    prop_invalid <- sim$prop_invalid[[dataset]]
    n_instruments <- sim$n_instruments[[dataset]]
    n_participants<- sim$n_participants[[dataset]]
    # Model for gene:exposure ***0 + deleted from X_lm and Y_lm***
    X_lm <- lm(X ~ 0 + Instruments_X)
    coeff_G_X_vect <- coef(summary(X_lm))[1:(ncol(Instruments_X)), 1]
    SE_coeff_G_X_vect <- coef(summary(X_lm))[1:(ncol(Instruments_X)), 2]

    R2_stat <- summary(lm(X ~ Instruments_X))$r.squared
    F_stat <- summary(lm(X ~ Instruments_X))$fstatistic[[1]]


    # Model for gene:outcome
    Y_lm <- lm(Y ~ 0 + Instruments_Y)
    coeff_G_Y_vect <- coef(summary(Y_lm))[1:(ncol(Instruments_Y)), 1]
    SE_coeff_G_Y_vect <- coef(summary(Y_lm))[1:(ncol(Instruments_Y)), 2]

    output_list[[dataset]] <- as_tibble(list(dataset = dataset,
                                         Instrument = c(1:ncol(Instruments_X)),
                                         coeff_G_X = coeff_G_X_vect,
                                         coeff_G_X_SE = SE_coeff_G_X_vect,
                                         gamma = gamma,
                                         F_stat = F_stat,
                                         R2_stat = R2_stat,
                                         coeff_G_Y = coeff_G_Y_vect,
                                         coeff_G_Y_SE = SE_coeff_G_Y_vect,
                                         alpha = alpha,
                                         phi = phi,
                                         beta = beta,
                                         prop_invalid = prop_invalid,
                                         n_instruments = n_instruments,
                                         n_participants = n_participants),
                                    .name_repair = "unique")
  }

  return(output_list)

}
```

These models generated estimates of the coefficient of gene-exposure association (`coeff_G_X`), coefficient of gene-outcome association (`coeff_G_Y`), and the relevant standard errors of these estimates. The values of parameters inputted were also returned to aid in further testing of data/model generation, i.e. actual gene-exposure associations (`gamma`), pleiotropic effects of invalid instruments (`alpha`), additional pleiotropic effects when Instrument Strength Independent of Direct Effect (InSIDE) assumption not satified (`phi`), causal effect of exposure on outcome (`beta`) and the proportion of invalid genetic instruments with pleiotropic effects on the outcome (`prop_invalid`).

```
test_extract_model <- get_models(test_data_sim)

summary(test_extract_model[[1]])
```

```
##     dataset     Instrument    coeff_G_X         coeff_G_X_SE
##  Min.   :1   Min.   : 1   Min.   :0.03419   Min.   :6.659e-17
##  1st Qu.:1   1st Qu.: 7   1st Qu.:0.05645   1st Qu.:6.807e-17
##  Median :1   Median :13   Median :0.06823   Median :6.873e-17
##  Mean   :1   Mean   :13   Mean   :0.06791   Mean   :6.889e-17
##  3rd Qu.:1   3rd Qu.:19   3rd Qu.:0.08594   3rd Qu.:6.935e-17
##  Max.   :1   Max.   :25   Max.   :0.09379   Max.   :7.313e-17
##      gamma              F_stat             R2_stat    coeff_G_Y
##  Min.   :0.03419   Min.   :6.224e+27   Min.   :1   Min.   :-0.109067
##  1st Qu.:0.05645   1st Qu.:6.224e+27   1st Qu.:1   1st Qu.: 0.004951
##  Median :0.06823   Median :6.224e+27   Median :1   Median : 0.006555
##  Mean   :0.06791   Mean   :6.224e+27   Mean   :1   Mean   : 0.013297
##  3rd Qu.:0.08594   3rd Qu.:6.224e+27   3rd Qu.:1   3rd Qu.: 0.008890
##  Max.   :0.09379   Max.   :6.224e+27   Max.   :1   Max.   : 0.162629
##   coeff_G_Y_SE          alpha                phi         beta      prop_invalid
##  Min.   :0.001550   Min.   :-0.115363   Min.   :0   Min.   :0.1   Min.   :0.3
##  1st Qu.:0.001579   1st Qu.: 0.000000   1st Qu.:0   1st Qu.:0.1   1st Qu.:0.3
##  Median :0.001591   Median : 0.000000   Median :0   Median :0.1   Median :0.3
##  Mean   :0.001598   Mean   : 0.006717   Mean   :0   Mean   :0.1   Mean   :0.3
##  3rd Qu.:0.001624   3rd Qu.: 0.000000   3rd Qu.:0   3rd Qu.:0.1   3rd Qu.:0.3
##  Max.   :0.001653   Max.   : 0.156224   Max.   :0   Max.   :0.1   Max.   :0.3
##  n_instruments n_participants
##  Min.   :25    Min.   :1000
##  1st Qu.:25    1st Qu.:1000
##  Median :25    Median :1000
##  Mean   :25    Mean   :1000
##  3rd Qu.:25    3rd Qu.:1000
##  Max.   :25    Max.   :1000
```

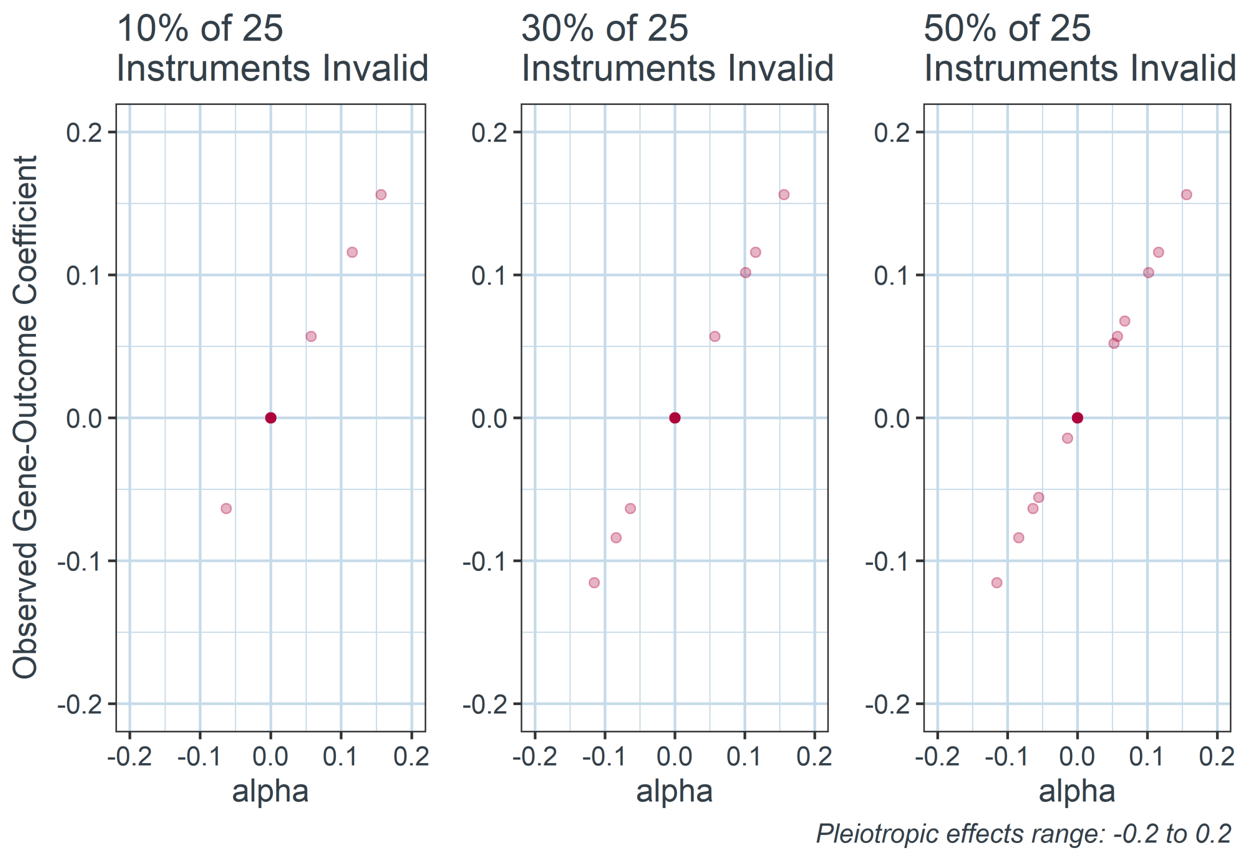
C.2 Testing Generation of Data and Models

A series of test plots were used to verify that data were simulated as intended under the various conditions specified by input parameters. Test plots were not created for the parameters `n_participants`, `n_instruments` or `n_datasets`, as the functioning of these parameters could be readily inferred from the structure of the datasets outputted, as above.

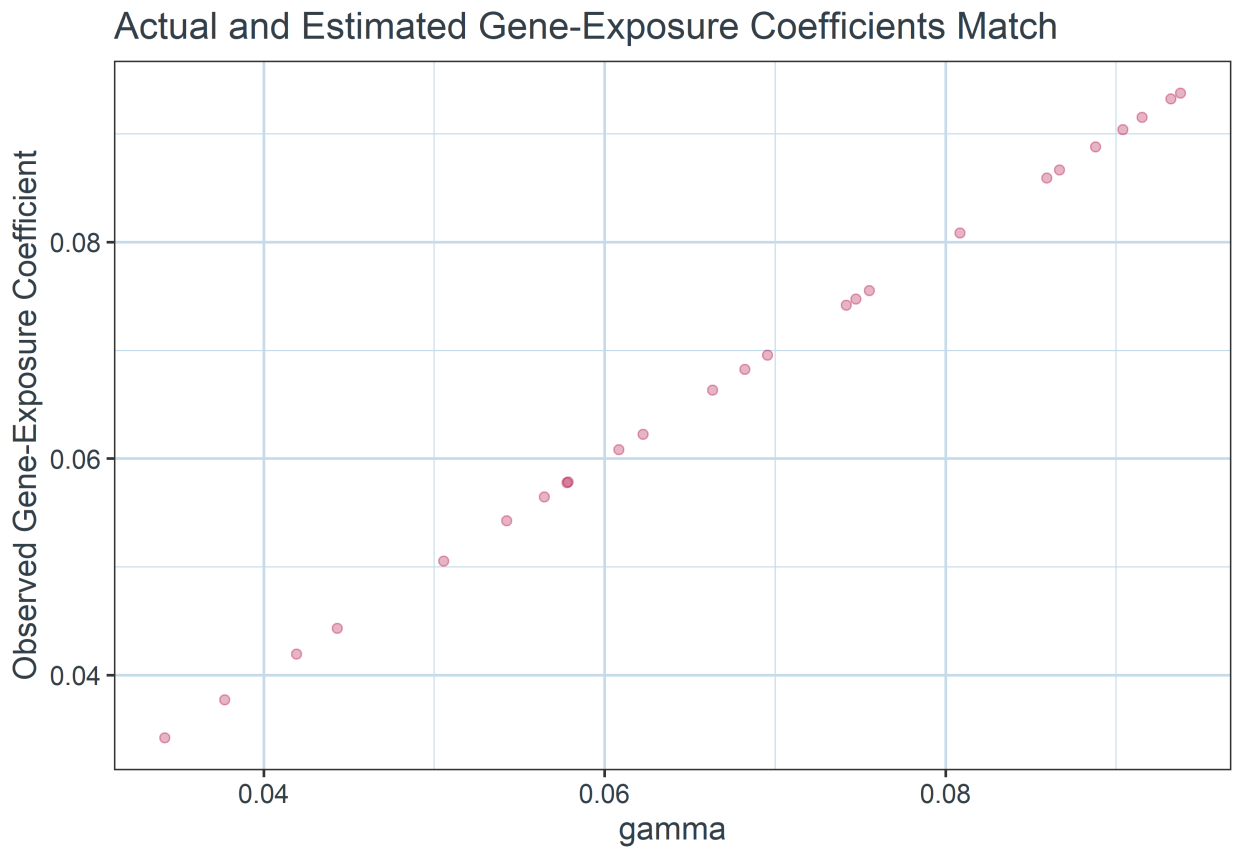
C.2.1 Proportion of Invalid Instruments

The `prop_invalid` parameter specifies the proportion of invalid genetic instruments simulated, i.e. the proportion of genetic instruments affecting the outcome via direct/pleiotropic effects, and thus not solely via the exposure of interest. If simulated correctly, increasing the value of `prop_invalid` should increase the number of instruments with pleiotropic effects, i.e. instruments with `alpha` $\neq 0$. With random error terms set to 0 and no causal effect present (i.e. `rand_error = FALSE` and `causal_effect = FALSE`), the estimated gene-outcome coefficient estimated using any given instrument will equal the pleiotropic effects of that instrument (i.e. `coeff_G_Y = alpha`), and therefore will only be non-zero for invalid instruments with non-zero pleiotropic effects on the outcome . Plotting `coeff_G_Y` against `alpha` for simulated data with no causal effect or random error should therefore yield a graph where

- For valid instruments: gene-outcome coefficient = alpha = 0
- For invalid instruments: gene-outcome coefficient = alpha $\neq 0$, with values spread uniformly between `alpha_min` and `alpha_max`



Similarly, with random error terms set to 0 (`rand_error = FALSE`) and no causal effect present (`causal_effect = FALSE`), gene-exposure coefficients estimated for each instrument should exactly match the actual values simulated, i.e. `coeff_G_X = gamma` for all instruments:

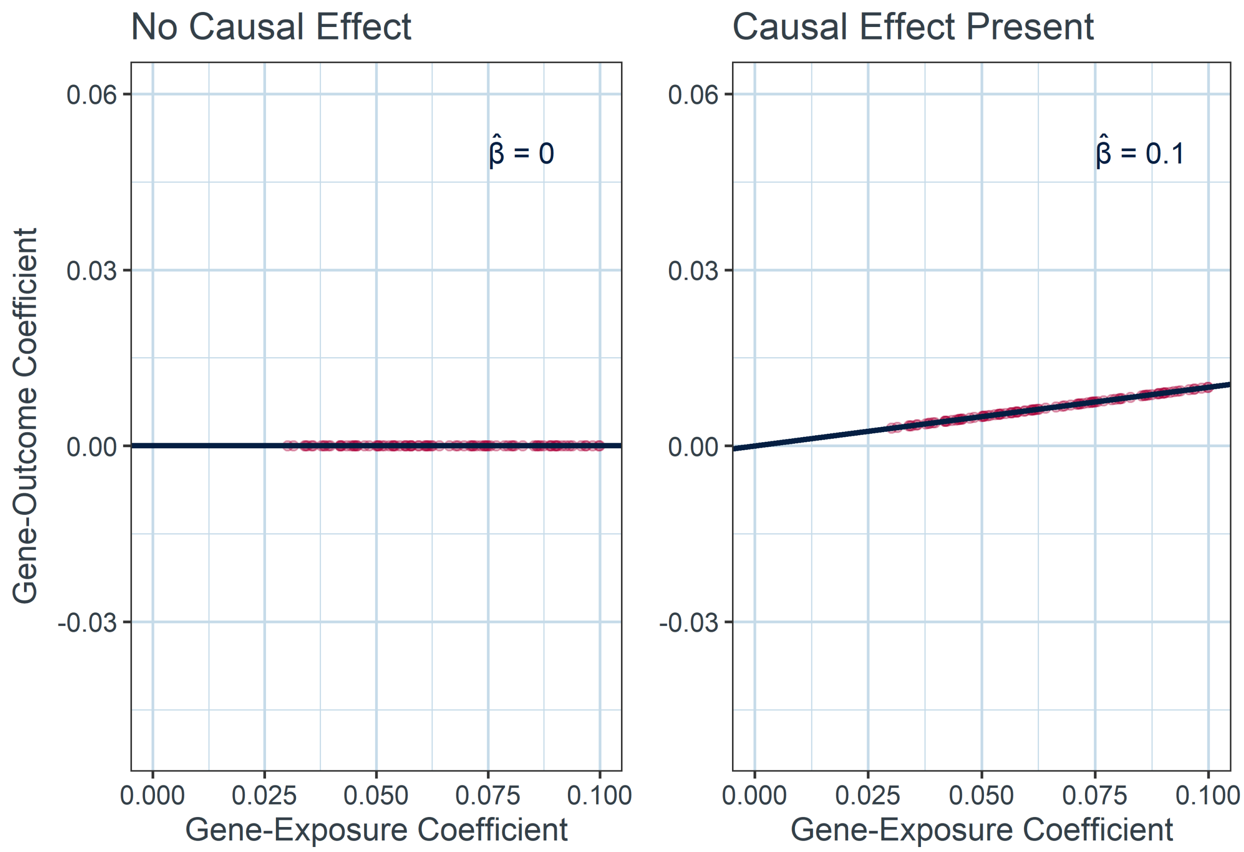


C.2.2 Gene-Exposure Coefficient Versus Gene-Outcome Coefficient Plots

For the next phase of testing, a function (`plot_GY_GX`) was written to plot the coefficients for gene-exposure versus gene-outcome as estimated using the previously created linear models:

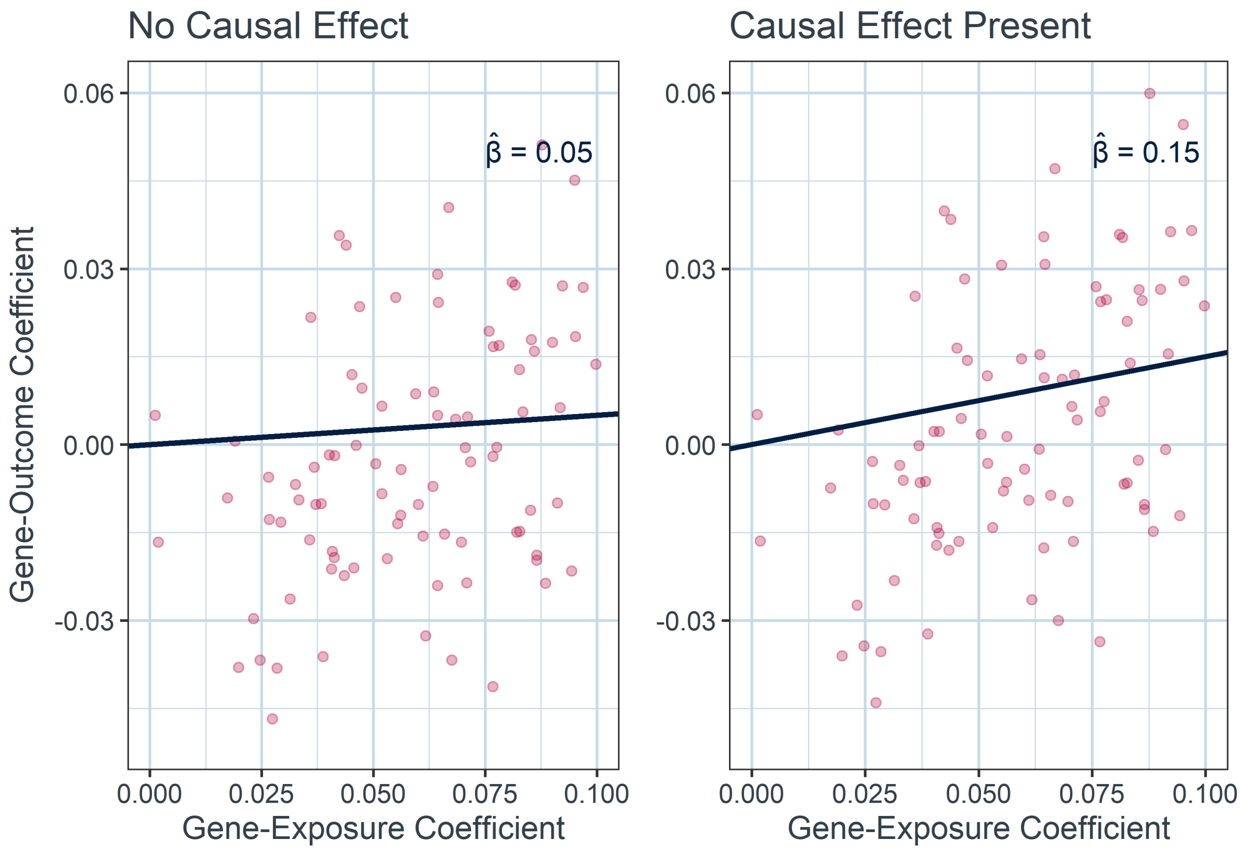
```
plot_GY_GX <- function(model_tib,
                       plot_title = as.character(NA),
                       x_min = 0,                     # set x-axis limits
                       x_max = 0.1,
                       y_min = -0.05,                 # set x-axis limits
                       y_max = 0.06,
                       beta_x = 0.075,                # set beta-hat position
                       beta_y = 0.05,
                       hat_offset = 0.003
)
{

  model_tib %>%
    mutate(Gradient = round(coefficients(lm(coeff_G_Y ~ 0 + coeff_G_X)[1], 5),
                            digits = 2)) %>%
    plot_template() + # pre-formatted plot template - call to ggplot with UoE colours
    aes(x = coeff_G_X, y = coeff_G_Y) +
    geom_point(colour = edin_bright_red_hex, alpha = 0.3) +
    geom_abline(aes(intercept = 0,
                    slope = Gradient),
                size = 1,
                colour = edin_uni_blue_hex) +
    geom_text(aes(label = paste0("\U03B2 = ", as.character(Gradient))), #beta
              x = beta_x, # labels with gradient (causal effect estimate)
              y = beta_y,
              colour = edin_uni_blue_hex,
              hjust = 0,
              data = . %>% slice_head()# prevent over-printing
    ) +
    #label = expression("True" ~ hat(beta)~ "= 0.25"),
    annotate("text",
             x = beta_x,       # add hat to beta
             y = beta_y + hat_offset,
             label = paste("\U02C6"),
             colour = edin_uni_blue_hex,
             hjust = -0.4,
             vjust = 0.9
    ) +
    labs(title = plot_title,
         x = "Gene-Exposure Coefficient",
         y = "Gene-Outcome Coefficient") +
    xlim(x_min, x_max) +
    ylim(y_min, y_max)

}
```

With random error terms set to 0 (`rand_error = FALSE`) and no causal effect present, a graph of gene-exposure coefficients versus gene-outcome coefficients should be a straight line through the origin with gradient = 0; causal effect of $\beta = 0.1$ present (`beta_val = 0.1`, `causal_effect = TRUE`), the slope of a graph of gene-exposure coefficients versus gene-outcome coefficients from the same sample should be a straight line through the origin with gradient = 0.1:



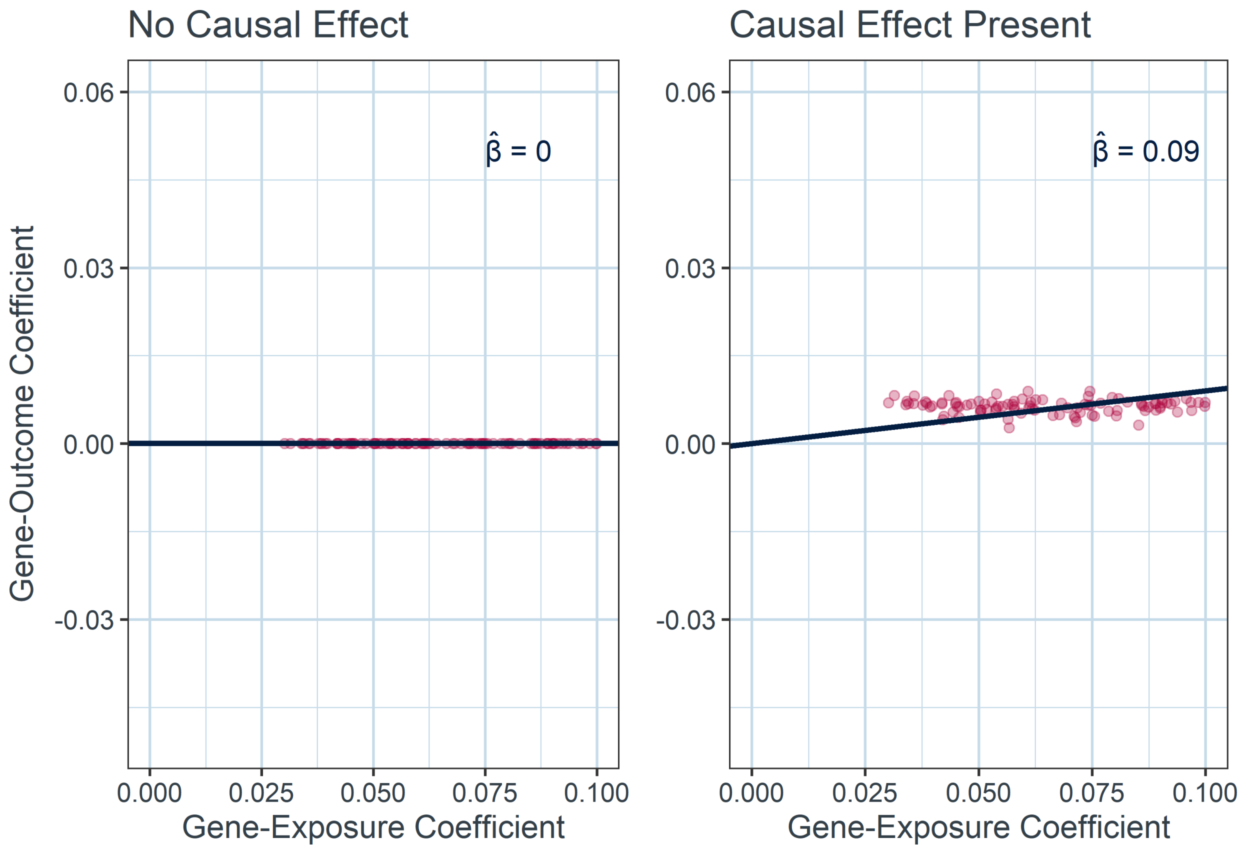
C.2.3 Random Errors

Re-plotting the same graphs with non-zero random error terms (`rand_error = TRUE`) should produce similar graphs with Gaussian spread around lines passing through the origin with gradients of 0 and 0.1 for no causal effect and causal effect, respectively:



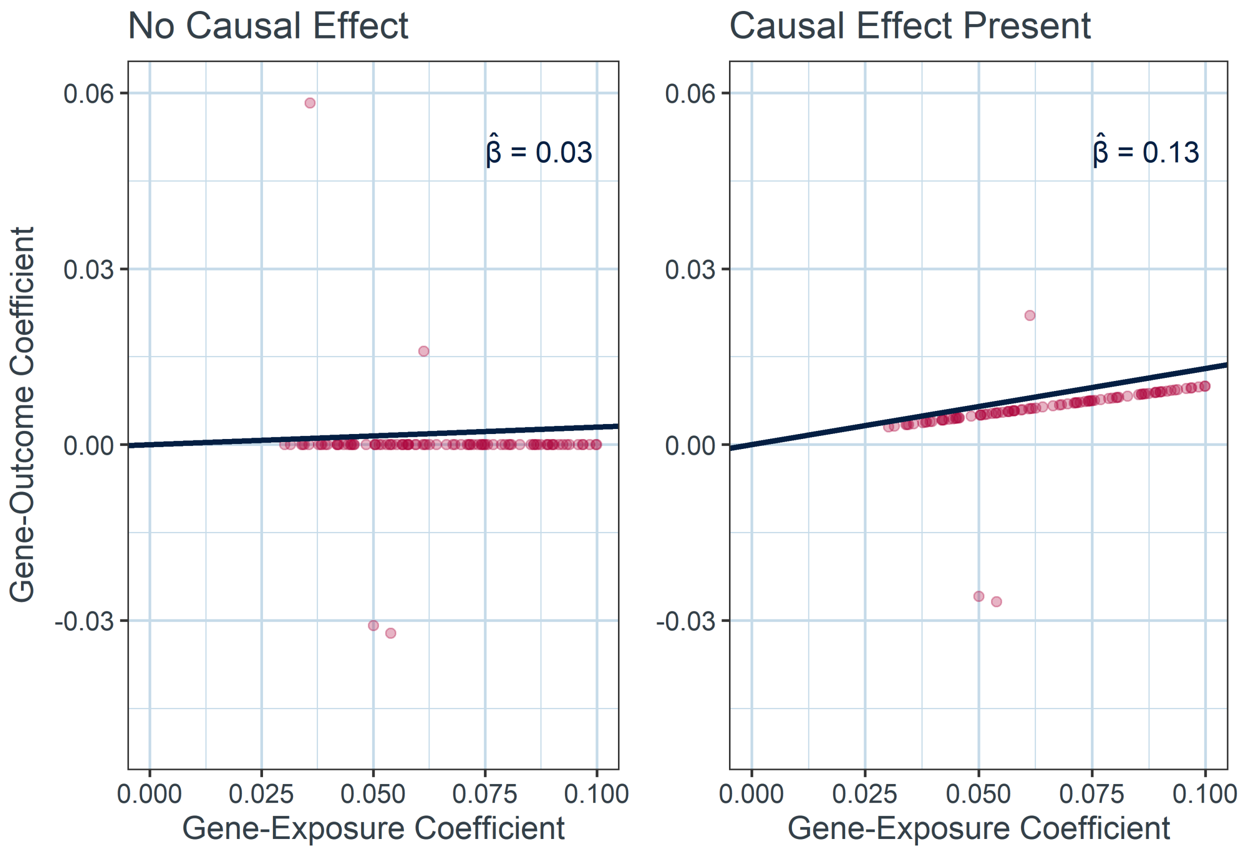
C.2.4 One versus Two Sample MR

Where gene-exposure coefficients and gene-outcome coefficients are estimated from two separate samples rather than one (i.e. `two_sample = TRUE`, simulating 2 sample MR), even with random error terms set to zero, error will be introduced into causal effect estimation through random sampling of different combinations of effect alleles. However, where a causal effect is not present, the effect estimated will consistently be zero regardless of the combinations of alleles sampled, so random error should not be introduced:



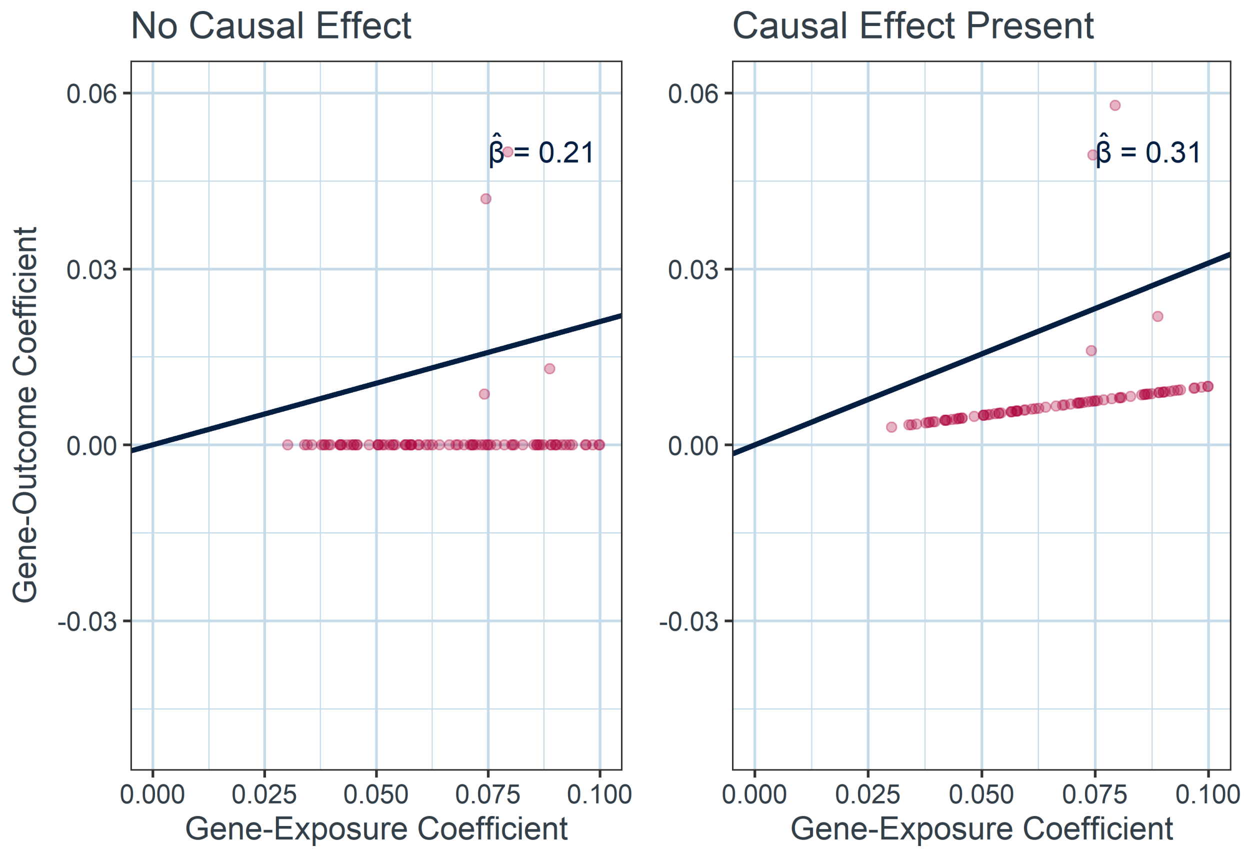
C.2.5 Invalid Instruments

Where invalid instruments are present (i.e. `prop_invalid` $\neq 0$) and random error terms are set to 0, graphs of gene-exposure coefficients versus gene-outcome coefficients should be straight lines through the origin and all points representing valid instruments; the invalid instruments should appear as outliers to this line:



C.2.6 Balanced Versus Directional Pleiotropy

Replotting the above with unbalanced pleiotropy present (`balanced_pleio = FALSE`), the invalid instruments should all appear as outliers in the positive direction, i.e. steepening the line of best fit and leading to overestimation of the causal effect:



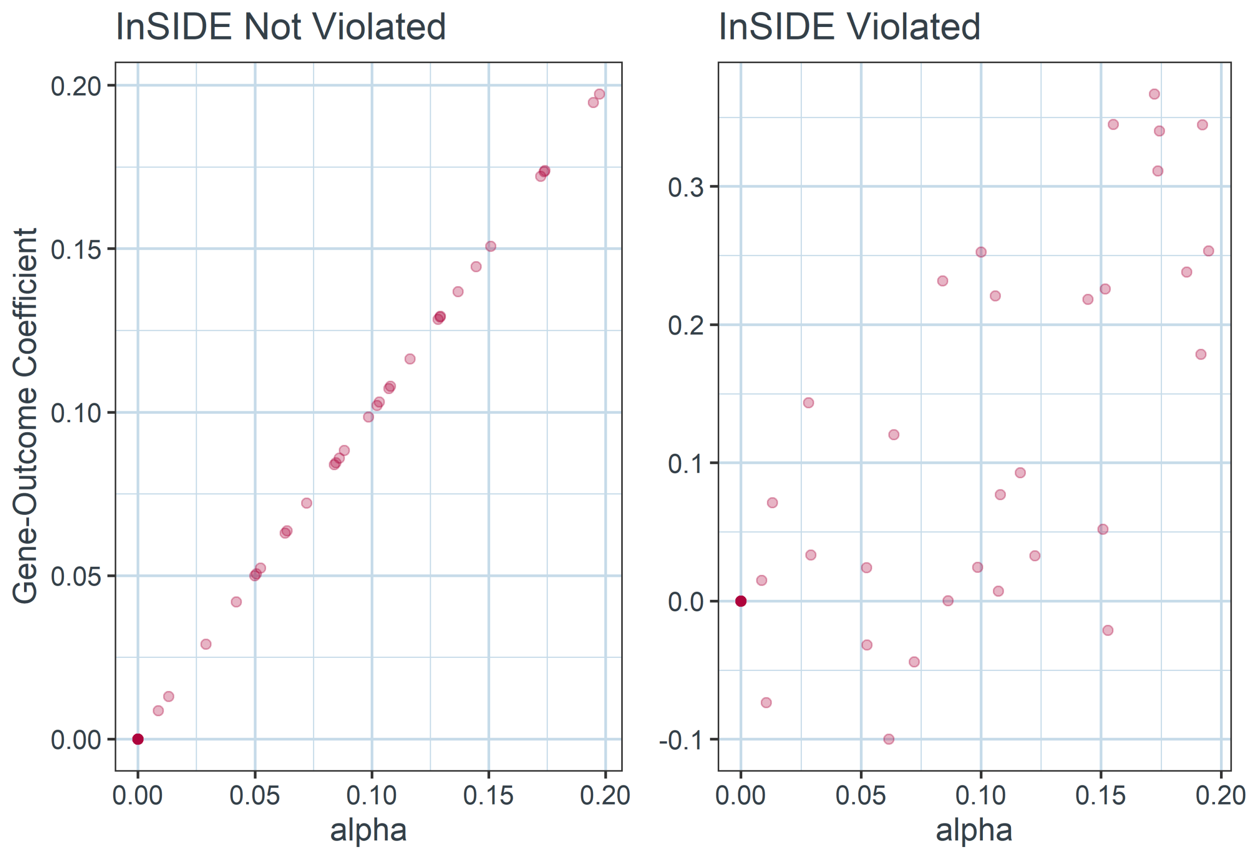
C.2.7 InSIDE Assumption and Phi

The variable phi represents additional pleiotropic effects of each invalid instrument when the InSIDE assumption is not satisfied. The InSIDE assumption states that the gene-exposure association is not correlated with the pleiotropic path gene-outcome path of any invalid genetic instruments. This assumption can be violated if e.g.:

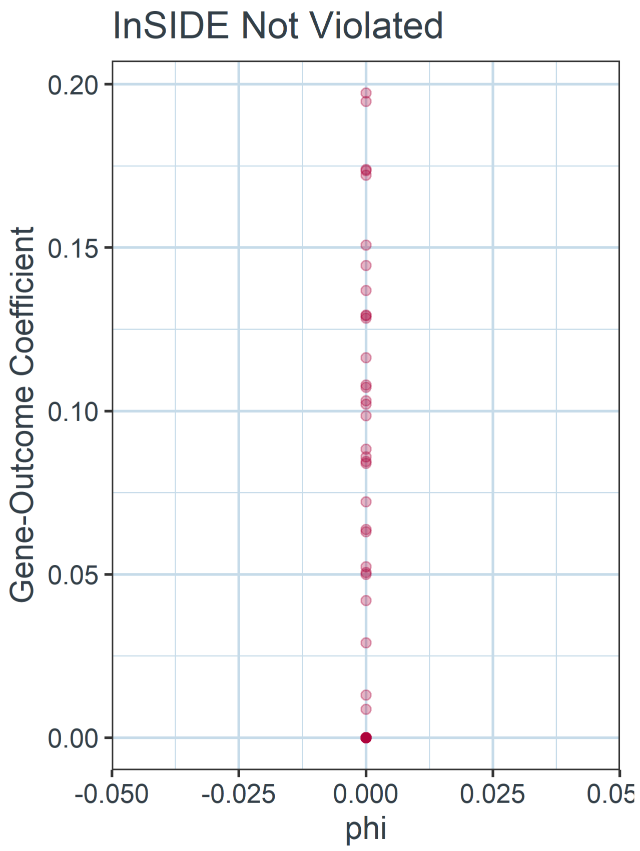
- several invalid genetic instruments influence the outcome via the same pleiotropic path
- several invalid genetic instruments are related to the same (unmeasured) confounders of the exposure:outcome relationship, aka correlated pleiotropy.

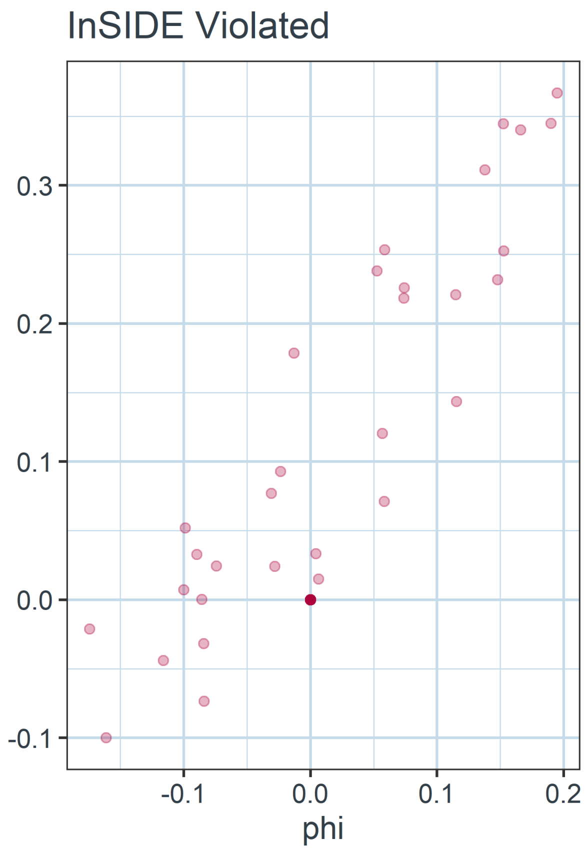
As such, when the InSIDE assumption is violated, even "strong" instruments (i.e. those with a strong gene-exposure relationship) may not allow accurate estimation of the true causal effect, as pleiotropic effects may scale with instrument strength. If pleiotropic effects are balanced, InSIDE assumption violation may lead to greater imprecision in causal effect estimation; if pleiotropic effects are directional, InSIDE assumption violation may lead to bias.

Bowden et al[2] modeled phi as the pleiotropic effects of unmeasured genetic confounders of the exposure:outcome relationship. Phi adds additional error to causal effect estimation in scenarios with directional pleiotropic effects (`0 < alpha < 0.2`) and InSIDE assumption violation. As such, switching `InSIDE_satisfied` from `TRUE` to `FALSE` should add scatter to the linear association expected when plotting alpha versus gene-outcome coefficients with random error terms set to zero:



Setting `InSIDE_satisfied = TRUE` should mean `phi = 0`; `InSIDE_satisfied=FALSE` should result in `phi` $\propto$ gene-outcome coefficient, with scatter only in the positive direction of gene-outcome coefficients given the model also requires directional pleiotropy before `phi` is used:





C.3 Summary Table

A function (`get_summary_MR_tib_row`) was written to take models generated from each simulated dataset, estimate causal effect using both weighted median and MR-Hevo methodologies, then output a summary formatted as per Tables 2 & 3 in Bowden et al[2]:

```r
# Load WME functions
library(TwoSampleMR)

# Load RStan - needed for MR-Hevo
library(rstan)


# Run local copy of MR-Hevo functions
# Not using full package due to conflicts with Windows
source(here("MSc_Thesis_Split", "Script", "Hevo", "functions.mrhevo.R"))

# Standard set-up for RStan
options(mc.cores = parallel::detectCores())
rstan_options(auto_write = TRUE, save_dso = TRUE)


# Compile model for MR-Hevo
mr.stanmodel <- stan_model(file= here("MSc_Thesis_Split",
                                      "Script",
                                      "Hevo",
                                      "MRHevo_summarystats.stan"),
                           model_name="MRHevo.summarystats",
                           verbose=FALSE,
                           save_dso = TRUE,
                           auto_write = TRUE)

get_summary_MR_tib_row <- function(model_list){


  # Create output tibble in same format as Table 2/3 from
  # Bowden et al
  output_tib_row <- tibble(N = as.integer(),
                           Prop_Invalid = as.double(),
                           F_stat = as.double(),
                           R2_stat = as.double(),
                           WME_Av = as.double(),
                           WME_SE = as.double(),
                           WME_Pos_Rate = as.double(),
                           Hevo_Av = as.double(),
                           Hevo_SE = as.double(),
                           Hevo_Pos_Rate = as.double())

  n_datasets <- length(model_list)

  # Create blank tibble to receive results of Weighted
  # Median Estimator function from MR-Base

  results_tib <-  tibble(WME_est = as.double(),
                         WME_se = as.double(),
                         WME_pval = as.double(),
                         WME_nsnp = as.integer(),
                         Hevo_est = as.double(),
                         Hevo_se = as.double(),
                         Hevo_sd = as.double(),
                         Hevo_est_lower_CI = as.double(),
                         Hevo_est_upper_CI = as.double(),
                         Hevo_causal_detected = as.logical()
)


# Run WME and MR-Hevo for each dataset
for(dataset in 1:n_datasets){

  # Stored as individual vectors for MR-Hevo/RStan - not
  # Tidyverse compatible
  coeff_G_X_vect <- model_list[[dataset]]$coeff_G_X
  coeff_G_Y_vect <- model_list[[dataset]]$coeff_G_Y
  coeff_G_X_SE_vect <- model_list[[dataset]]$coeff_G_X_SE
  coeff_G_Y_SE_vect <- model_list[[dataset]]$coeff_G_Y_SE
  prop_invalid <- min(model_list[[dataset]]$prop_invalid)
  F_stat <- min(model_list[[dataset]]$F_stat)
  R2_stat <- min(model_list[[dataset]]$R2_stat)
  n_instruments <- max(model_list[[dataset]]$Instrument)
  n_participants <- min(model_list[[dataset]]$n_participants)


  # N.B. MR-Hevo terminology vs WME paper/other code:
  # alpha = effects of instruments on exposure, i.e. coeff_G_X
  # beta = pleiotropic effects of instruments on outcome, i.e. alpha in WME
  # gamma = effects of instruments on outcome, i.e. coeff_G_Y
  # theta = causal effect X on Y, i.e. b

  # Results from weighted median estimator method
  WME_results <- mr_weighted_median(b_exp = coeff_G_X_vect,
                                    b_out = coeff_G_Y_vect,
                                    se_exp = coeff_G_X_SE_vect,
                                    se_out = coeff_G_Y_SE_vect,
                                    parameters = list(nboot = 1000))

  # Results from MR-Hevo method
  Hevo_results<- run_mrhevo.sstats(alpha_hat = coeff_G_X_vect,
                                  se.alpha_hat = coeff_G_X_SE_vect,
                                  gamma_hat = coeff_G_Y_vect,
                                  se.gamma_hat = coeff_G_Y_SE_vect) %>%
   summary()


  # Extract WME Results
  results_tib[dataset, ]$WME_est <- WME_results$b
  results_tib[dataset, ]$WME_se <- WME_results$se
  results_tib[dataset, ]$WME_pval <- WME_results$pval
```

```r
    results_tib[dataset, ]$WME_nsnp <- WME_results$nsnp

    # Extract MR-Hevo Results
    results_tib[dataset, ]$Hevo_est <- Hevo_results$summary["theta","mean"]
    results_tib[dataset, ]$Hevo_se <- Hevo_results$summary["theta","se_mean"]
    results_tib[dataset, ]$Hevo_sd <- Hevo_results$summary["theta","sd"]
    results_tib[dataset, ]$Hevo_est_lower_CI <- Hevo_results$summary["theta","2.5%"]
    results_tib[dataset, ]$Hevo_est_upper_CI <- Hevo_results$summary["theta","97.5%"]

  }

  # Add causality Boolean to MR-Hevo
  results_tib <- results_tib %>%
   mutate(Hevo_est_causal_detected = (Hevo_est_lower_CI > 0  | Hevo_est_upper_CI < 0))


  output_tib_row <- results_tib %>%
    summarise(N = n_participants,
              Prop_Invalid = prop_invalid,
              F_stat = mean(F_stat),
              R2_stat = mean(R2_stat),
              WME_Av = mean(WME_est),
              WME_SE = mean(WME_se),
              WME_Pos_Rate = length(WME_pval[WME_pval < 0.05]) / n_datasets,
              Hevo_Av = mean(Hevo_est),
              Hevo_SE = mean(Hevo_se),
              Hevo_Lower_CI = mean(Hevo_est_lower_CI),
              Hevo_Upper_CI = mean(Hevo_est_upper_CI),
              Hevo_Pos_Rate = sum(Hevo_est_causal_detected) / n_datasets
    ) %>%
    mutate(across(where(is.double), round, 3))

  return(output_tib_row)

}
```

```r
test_tib_summ_MR_data <-  get_simulated_MR_data(n_participants = 10000,
                                                n_instruments = 25,
                                                n_datasets = 2,
                                                prop_invalid = 0.1,
                                                beta_val = 0.1,
                                                causal_effect = TRUE,
                                                rand_error = TRUE,
                                                two_sample = TRUE,
                                                balanced_pleio = TRUE,
                                                InSIDE_satisfied = TRUE)

test_tib_summ_MR_models <- get_models(test_tib_summ_MR_data)

test_tib_summ_MR_row <- get_summary_MR_tib_row(test_tib_summ_MR_models)
```

```
##
## CHECKING DATA AND PREPROCESSING FOR MODEL 'MRHevo.summarystats' NOW.
```

N	Prop_Invalid	F_stat	R2_stat	WME_Av	WME_SE	WME_Pos_Rate	Hevo_Av	Hevo_SE	Hevo_Lower_CI	Hevo_Upper_CI	Hevo_Pos_Rate
10000	0.1	14.739	0.036	0.091	0.09	0.5	0.121	0.001	-0.052	0.302	0

```
##
## COMPILING MODEL 'MRHevo.summarystats' NOW.
##
## STARTING SAMPLER FOR MODEL 'MRHevo.summarystats' NOW.
```

```
##
## CHECKING DATA AND PREPROCESSING FOR MODEL 'MRHevo.summarystats' NOW.
##
## COMPILING MODEL 'MRHevo.summarystats' NOW.
##
## STARTING SAMPLER FOR MODEL 'MRHevo.summarystats' NOW.
```

```
test_tib_summ_MR_row %>%
  kable() %>%
  kable_styling(latex_options="scale_down")
```

D Appendix: R Packages Used

```
## [1] "\\begin{table}[t]\n\\fontsize{12.0pt}{14.4pt}\\selectfont\n\\begin{tabular*}{\\linewidth}{@{\\e
```

1. Higgins P. medicaldata: Data package for medical datasets [Internet]. 2021. Available from: https://CRAN.R-project.org/package=medicaldata

2. Bowden J, Smith GD, Haycock PC, Burgess S. Consistent Estimation in Mendelian Randomization with Some Invalid Instruments Using a Weighted Median Estimator. Genetic Epidemiology [Internet]. 2016 Apr [cited 2024 Oct 22];40(4):304. Available from: https://pmc.ncbi.nlm.nih.gov/articles/PMC4849733/