

9. Appendices

Contents

Appendix A: List of Abbreviations	1
Appendix B: Simulation Code	2
Generating Data and Models	2
Testing Generation of Data and Models	12
Appendix C: Citation Search Strategy	32

Appendix A: List of Abbreviations

Appendix B: Simulation Code

Generating Data and Models

The data generating model used was from Appendix 3 of Bowden et al¹; the relevant section describing their model is reproduced below:

“...

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

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

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

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

The error terms $\epsilon_i^U, \epsilon_i^X$ 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.”¹

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

```
# Define function to create data generating model
# Arguments/default values based on Bowden et al
simulate_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 = 0.01,    # frequency of effect allele
        allele_freq_max = 0.99,
        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

# 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()
beta_list <- list()
#prop_invalid_list <- list()

n_participants_list <- list()
n_instruments_list <- list()
n_datasets_list <- list()
prop_invalid_list <- list()
causal_effect_list <- list()
balanced_pleio_list <- list()
InSIDE_satisfied_list <- list()
rand_error_list <- list()
two_sample_list <- list()
beta_val_list <- list()
allele_freq_min_list <- list()
allele_freq_max_list <- list()
gamma_min_list <- list()
gamma_max_list <- list()
alpha_min_list <- list()
alpha_max_list <- list()

```

```

phi_min_list <- list()
phi_max_list <- list()

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

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

# --- 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 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))

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

  # 0 = reference, i.e. zero effect alleles,
  # 1 = 1 effect allele, 2 = 2 effect alleles

```

```

# 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)

# 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
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)

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

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

# 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 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 = n_instruments, # balanced
                               min = alpha_min,
                               max = alpha_max),
        alpha_vect[j] <- runif(n = n_instruments, # 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
}
}

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

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

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 parameters used in simulation for testing
alpha_list[[n]] <- alpha_vect

gamma_list[[n]] <- gamma_vect

phi_list[[n]] <- phi_vect

beta_list[[n]] <- beta

#prop_invalid_list[[n]] <- prop_invalid

n_participants_list[[n]] <- n_participants
n_instruments_list[[n]] <- n_instruments
n_datasets_list[[n]] <- n_datasets
prop_invalid_list[[n]] <- prop_invalid
causal_effect_list[[n]] <- causal_effect
balanced_pleio_list[[n]] <- balanced_pleio
InSIDE_satisfied_list[[n]] <- InSIDE_satisfied
rand_error_list[[n]] <- rand_error
two_sample_list[[n]] <- two_sample
beta_val_list[[n]] <- beta_val
allele_freq_min_list[[n]] <- allele_freq_min
allele_freq_max_list[[n]] <- allele_freq_max
gamma_min_list[[n]] <- gamma_min
gamma_max_list[[n]] <- gamma_max
alpha_min_list[[n]] <- alpha_min
alpha_max_list[[n]] <- alpha_max
phi_min_list[[n]] <- phi_min
phi_max_list[[n]] <- phi_max

}

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

# --- 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,
    #beta = beta_list,
    #prop_invalid = prop_invalid_list,

    n_participants = n_participants_list, # Inputs
    n_instruments = n_instruments_list,
    n_datasets = n_datasets_list,
    prop_invalid = prop_invalid_list,
    causal_effect = causal_effect_list,
    balanced_pleio = balanced_pleio_list,
    InSIDE_satisfied = InSIDE_satisfied_list,
    rand_error = rand_error_list,
    two_sample = two_sample_list,
    beta_val = beta_val_list,
    allele_freq_min = allele_freq_min_list,
    allele_freq_max = allele_freq_max_list,
    gamma_min = gamma_min_list,
    gamma_max = gamma_max_list,
    alpha_min = alpha_min_list,
    alpha_max = alpha_max_list,
    phi_min = phi_min_list,
    phi_max = phi_max_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 <- simulate_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 26
## $ 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] 1.12 1.59 1.76 1.49 1.56 ...
## ..$ : num [1:1000, 1] 1.84 1.7 1.6 1.66 1.5 ...
## $ Y :List of 2
## ..$ : num [1:1000, 1] -0.24 -0.311 -0.393 -0.227 -0.1 ...
## ..$ : num [1:1000, 1] -0.872 -0.901 -0.772 -0.999 -0.477 ...
## $ G_X :List of 2

```



```

## ..$ : int [1:1000, 1:25] 0 1 1 1 1 0 0 0 0 0 ...
## ..$ : int [1:1000, 1:25] 1 2 1 2 2 2 2 2 2 2 ...
## $ G_Y :List of 2
## ..$ : int [1:1000, 1:25] 0 1 1 0 1 0 0 0 0 0 ...
## ..$ : int [1:1000, 1:25] 2 2 2 2 1 2 1 1 2 1 ...
## $ alpha :List of 2
## ..$ : num [1:25] -0.106 0 -0.121 0 0 ...
## ..$ : num [1:25] 0 0 -0.0786 0 0 ...
## $ gamma :List of 2
## ..$ : num [1:25] 0.0902 0.0878 0.08 0.0832 0.084 ...
## ..$ : num [1:25] 0.0374 0.0721 0.0975 0.085 0.0322 ...
## $ 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
## $ n_datasets :List of 2
## ..$ : num 2
## ..$ : num 2
## $ prop_invalid :List of 2
## ..$ : num 0.3
## ..$ : num 0.3
## $ causal_effect :List of 2
## ..$ : logi TRUE
## ..$ : logi TRUE
## $ balanced_pleio :List of 2
## ..$ : logi TRUE
## ..$ : logi TRUE
## $ InSIDE_satisfied:List of 2
## ..$ : logi TRUE
## ..$ : logi TRUE
## $ rand_error :List of 2
## ..$ : logi FALSE
## ..$ : logi FALSE
## $ two_sample :List of 2
## ..$ : logi TRUE
## ..$ : logi TRUE
## $ beta_val :List of 2
## ..$ : num 0.1
## ..$ : num 0.1
## $ allele_freq_min :List of 2
## ..$ : num 0.01
## ..$ : num 0.01
## $ allele_freq_max :List of 2
## ..$ : num 0.99
## ..$ : num 0.99
## $ gamma_min :List of 2
## ..$ : num 0.03
## ..$ : num 0.03
## $ gamma_max :List of 2

```

```
## ..$ : num 0.1
## ..$ : num 0.1
## $ alpha_min      :List of 2
## ..$ : num -0.2
## ..$ : num -0.2
## $ alpha_max      :List of 2
## ..$ : num 0.2
## ..$ : num 0.2
## $ phi_min        :List of 2
## ..$ : num -0.2
## ..$ : num -0.2
## $ phi_max        :List of 2
## ..$ : num 0.2
## ..$ : num 0.2
```

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

```
# Create plotting tibble with Mean/SD X + Y grouped by
# Dataset + instrument
extract_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
    #X_lm <- lm(X ~ Instruments_X)
    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))$r.squared
    #R2_stat <- summary(X_lm)$r.squared

    # Model for gene:outcome
    #Y_lm <- lm(Y ~ Instruments_Y)
```

```

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,
    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 InSIDE assumption not satisfied (`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 <- extract_models(test_data_sim)

summary(test_extract_model[[1]])

```

```

##      dataset      Instrument      coeff_G_X      coeff_G_X_SE
##  Min.   :1      Min.   : 1      Min.   :0.03006      Min.   :1.591e-16
## 1st Qu.:1      1st Qu.: 7      1st Qu.:0.03791      1st Qu.:1.702e-16
## Median :1      Median :13      Median :0.05578      Median :1.847e-16
## Mean   :1      Mean   :13      Mean   :0.06018      Mean   :2.346e-16
## 3rd Qu.:1      3rd Qu.:19      3rd Qu.:0.07998      3rd Qu.:2.441e-16
## Max.   :1      Max.   :25      Max.   :0.09140      Max.   :7.259e-16
##      gamma      coeff_G_Y      coeff_G_Y_SE      alpha
##  Min.   :0.03006      Min.   : -0.1188256      Min.   :0.0009824      Min.   : -0.120669
## 1st Qu.:0.03791      1st Qu.: 0.0006676      1st Qu.:0.0010520      1st Qu.: 0.000000
## Median :0.05578      Median : 0.0031161      Median :0.0011837      Median : 0.000000
## Mean   :0.06018      Mean   : -0.0047291      Mean   :0.0014576      Mean   : -0.008692
## 3rd Qu.:0.07998      3rd Qu.: 0.0068099      3rd Qu.:0.0015114      3rd Qu.: 0.000000
## Max.   :0.09140      Max.   : 0.1356693      Max.   :0.0040567      Max.   : 0.133513
##      phi      beta      prop_invalid
##  Min.   :0      Min.   :0.1      Min.   :0.3
## 1st Qu.:0      1st Qu.:0.1      1st Qu.:0.3

```

```
## Median :0   Median :0.1   Median :0.3
## Mean   :0   Mean     :0.1   Mean     :0.3
## 3rd Qu.:0   3rd Qu.:0.1   3rd Qu.:0.3
## Max.   :0   Max.     :0.1   Max.     :0.3
```

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.

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`

```
# Check altering proportion of invalid instruments alters
# proportion of instruments displaying pleiotropic effects
# N.B. cluster around alpha = 0 represents valid instruments with
# no pleiotropic effects

# 10% of instruments invalid
set.seed(1701)
sim_test_data_inval_0.1 <- simulate_MR_data(n_participants = 1000,
                                           n_instruments = 25,
                                           n_datasets = 1,
                                           prop_invalid = 0.1,
                                           rand_error = FALSE,
                                           causal_effect = FALSE,
                                           alpha_min = -0.2,
                                           alpha_max = 0.2)

# 30% of instruments invalid
set.seed(1701)
sim_test_data_inval_0.3 <- simulate_MR_data(n_participants = 1000,
                                           n_instruments = 25,
                                           n_datasets = 1,
                                           prop_invalid = 0.3,
                                           rand_error = FALSE,
                                           causal_effect = FALSE,
                                           alpha_max = 0.2)

# 50% of instruments invalid
```

```

set.seed(1701)
sim_test_data_inval_0.5 <- simulate_MR_data(n_participants = 1000,
                                           n_instruments = 25,
                                           n_datasets = 1,
                                           prop_invalid = 0.5,
                                           rand_error = FALSE,
                                           causal_effect = FALSE,
                                           alpha_min = -0.2,
                                           alpha_max = 0.2)

test_plot_tib_inval_0.1 <- extract_models(sim_test_data_inval_0.1)[[1]]
test_plot_tib_inval_0.3 <- extract_models(sim_test_data_inval_0.3)[[1]]
test_plot_tib_inval_0.5 <- extract_models(sim_test_data_inval_0.5)[[1]]

test_plot_inval_0.1 <- test_plot_tib_inval_0.1 %>%
  select(alpha, coeff_G_Y) %>%
  plot_template() +
  geom_point(colour = edin_bright_red_hex, alpha = 0.3) +
  aes(x = alpha, y = coeff_G_Y) +
  scale_y_continuous(limits = c(-0.2, 0.2)) +
  scale_x_continuous(limits = c(-0.2, 0.2)) +
  labs(y = "Observed Gene:Outcome Coefficient",
       title = "Test: 10% of 25 \nInstruments Invalid",
       subtitle = "Pleiotropic effects range \n-0.2 to 0.2")

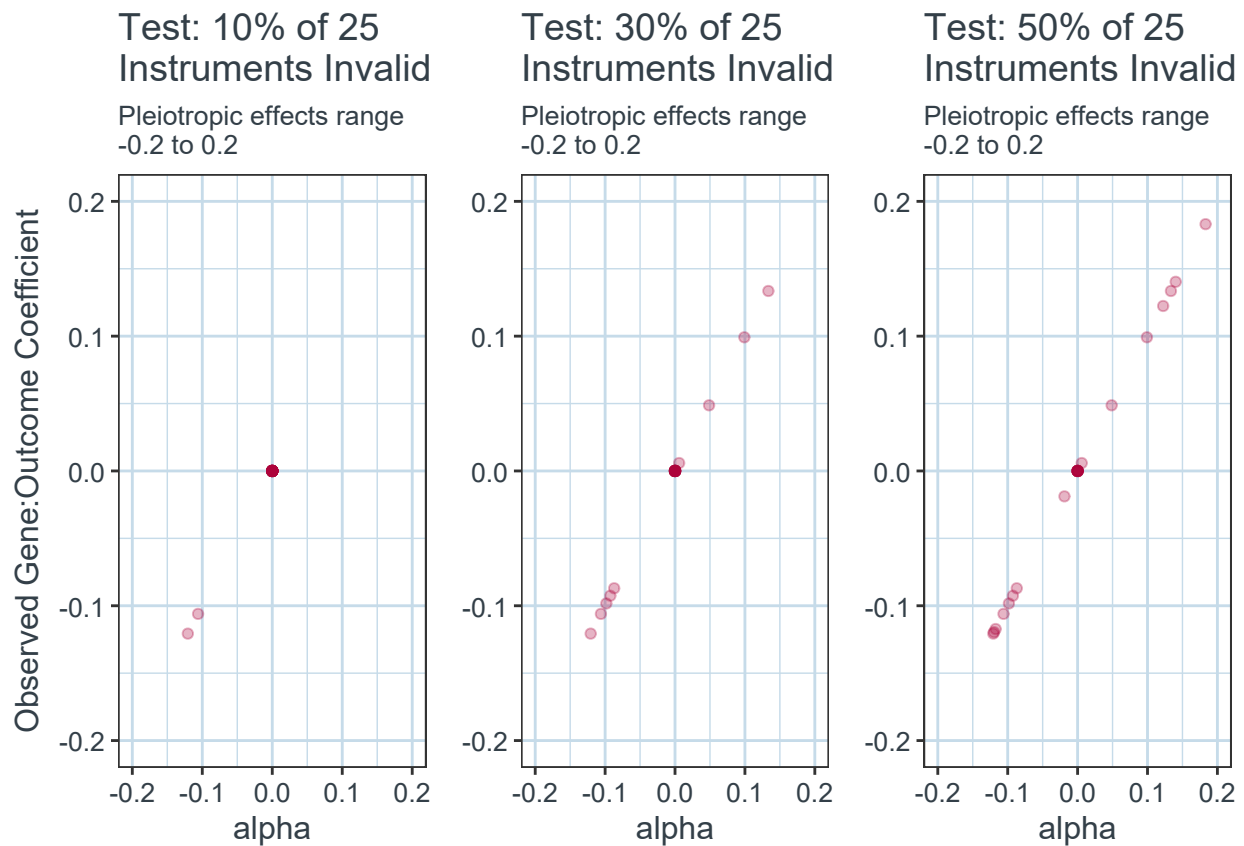
test_plot_inval_0.3 <- test_plot_tib_inval_0.3 %>%
  select(alpha, coeff_G_Y) %>%
  plot_template() +
  geom_point(colour = edin_bright_red_hex, alpha = 0.3) +
  aes(x = alpha, y = coeff_G_Y) +
  scale_y_continuous(limits = c(-0.2, 0.2)) +
  scale_x_continuous(limits = c(-0.2, 0.2)) +
  labs(y = "Observed Gene:Outcome Coefficient",
       title = "Test: 30% of 25 \nInstruments Invalid",
       subtitle = "Pleiotropic effects range \n-0.2 to 0.2") +
  theme(axis.title.y = element_blank())

test_plot_inval_0.5 <- test_plot_tib_inval_0.5 %>%
  select(alpha, coeff_G_Y) %>%
  plot_template() +
  geom_point(colour = edin_bright_red_hex, alpha = 0.3) +
  aes(x = alpha, y = coeff_G_Y) +
  scale_y_continuous(limits = c(-0.2, 0.2)) +
  scale_x_continuous(limits = c(-0.2, 0.2)) +
  labs(y = "Observed Gene:Outcome Coefficient",
       title = "Test: 50% of 25 \nInstruments Invalid",
       subtitle = "Pleiotropic effects range \n-0.2 to 0.2") +
  theme(axis.title.y = element_blank())

plot_grid(test_plot_inval_0.1,
          test_plot_inval_0.3,
          test_plot_inval_0.5,

```

```
ncol = 3,
rel_widths = c(1.1, 1, 1))
```



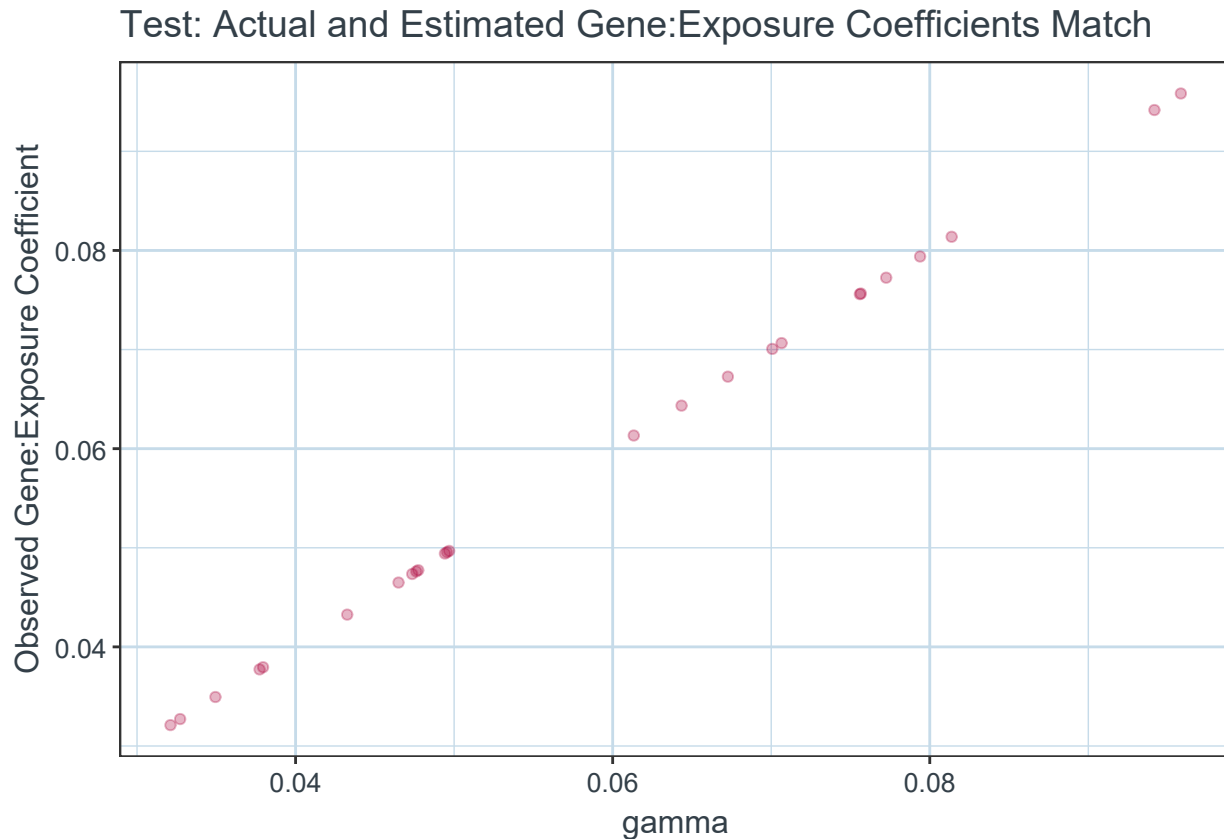
Similarly, with random error terms set to 0 (`rand_error = FALSE`) and no causal effect present (`causal_effect = FALSE`), gene:exposure coefficients estimated for each instrument should exactly match the actual values simulated, i.e. `coeff_G_X = gamma` for all instruments:

```
# Check observed gene:exposure coefficients for each instrument
# (coeff_G_X) approximate true values (gamma) when a causal effect
# is present & a large number of participants are included
set.seed(1701)
sim_test_data_gamma_1 <- simulate_MR_data(n_participants = 100,
                                         n_instruments = 25,
                                         n_datasets = 1,
                                         prop_invalid = 0.1,
                                         causal_effect = FALSE,
                                         rand_error = FALSE,
                                         balanced_pleio = TRUE,
                                         InSIDE_satisfied = TRUE)

test_plot_tib_gamma_1 <- extract_models(sim_test_data_gamma_1)[[1]]

test_plot_tib_gamma_1 %>%
  select(gamma, coeff_G_X) %>%
  plot_template() +
```

```
geom_point(colour = edin_bright_red_hex, alpha = 0.3) +
aes(x = gamma, y = coeff_G_X ) +
labs(y = "Observed Gene:Exposure Coefficient",
      title = "Test: Actual and Estimated Gene:Exposure Coefficients Match")
```



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 y-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() +
    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
) +
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 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:

```

# No causal effect present
set.seed(1701)
sim_test_data_causal_0 <- simulate_MR_data(n_participants = 10000,
                                           n_instruments = 100,
                                           n_datasets = 1,
                                           prop_invalid = 0,
                                           causal_effect = FALSE,
                                           rand_error = FALSE)

test_plot_tib_causal_0 <- extract_models(sim_test_data_causal_0)[[1]]

test_plot_causal_0 <- plot_GY_GX(test_plot_tib_causal_0, plot_title = "No Causal Effect")

# Causal effect present
set.seed(1701)
sim_test_data_causal_1 <- simulate_MR_data(n_participants = 10000,
                                           n_instruments = 100,
                                           n_datasets = 1,
                                           prop_invalid = 0,
                                           beta_val = 0.1,
                                           causal_effect = TRUE,

```



```

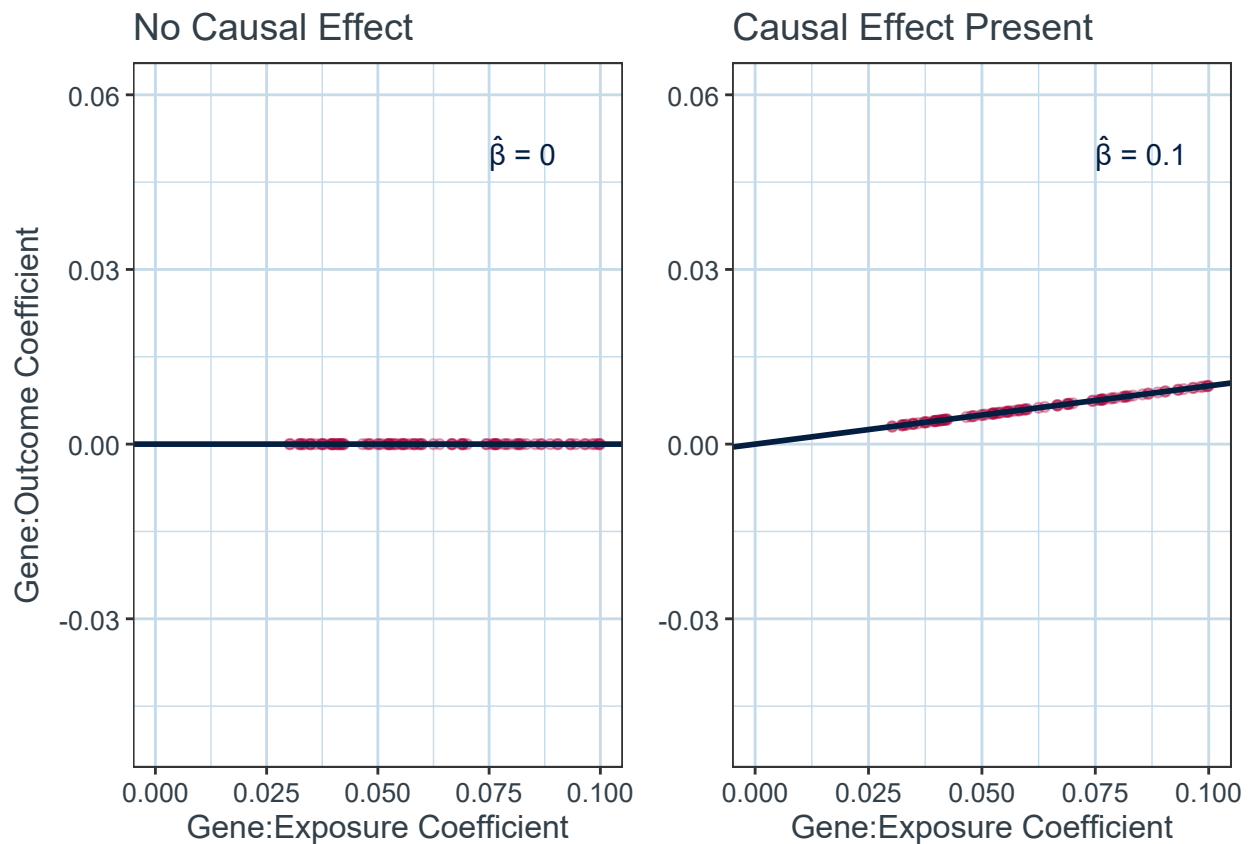
    rand_error = FALSE,
    two_sample = FALSE)

test_plot_tib_causal_1 <- extract_models(sim_test_data_causal_1)[[1]]

test_plot_causal_1 <- plot_GY_GX(test_plot_tib_causal_1, plot_title = "Causal Effect Present")+
  theme(axis.title.y = element_blank())

plot_grid(test_plot_causal_0,
  test_plot_causal_1,
  ncol = 2,
  rel_widths = c(1.05, 1))

```



Re-plotting the same graphs with non-zero random error terms 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:

```

# Causal effect not present
set.seed(1701)
sim_test_data_causal_0_errors <- simulate_MR_data(n_participants = 10000,
  n_instruments = 100,
  n_datasets = 1,
  prop_invalid = 0,
  causal_effect = FALSE,
  rand_error = TRUE,
  two_sample = FALSE)

```

```

test_plot_tib_causal_0_errors <- extract_models(sim_test_data_causal_0_errors)[[1]]

test_plot_causal_0_errors <- plot_GY_GX(test_plot_tib_causal_0_errors,
                                         plot_title = "Causal Effect Not Present")

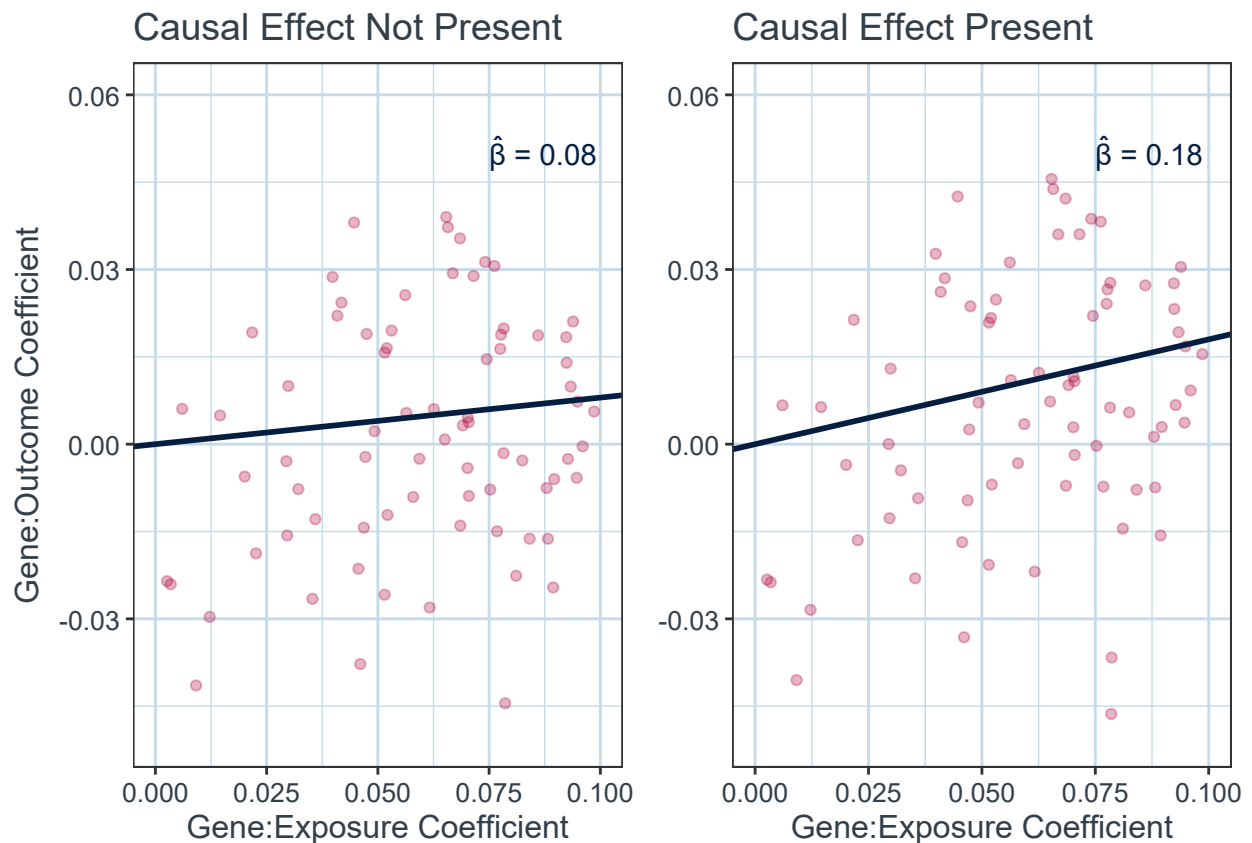
# Causal effect present
set.seed(1701)
sim_test_data_causal_1_errors <- simulate_MR_data(n_participants = 10000,
                                                  n_instruments = 100,
                                                  n_datasets = 1,
                                                  prop_invalid = 0,
                                                  causal_effect = TRUE,
                                                  rand_error = TRUE,
                                                  two_sample = FALSE)

test_plot_tib_causal_1_errors <- extract_models(sim_test_data_causal_1_errors)[[1]]

test_plot_causal_1_errors <- plot_GY_GX(test_plot_tib_causal_1_errors,
                                         plot_title = "Causal Effect Present") +
  theme(axis.title.y = element_blank())

plot_grid(test_plot_causal_0_errors,
          test_plot_causal_1_errors,
          ncol = 2,
          rel_widths = c(1.05, 1))

```



Where gene:exposure coefficients and gene:outcome coefficients are estimated from two separate samples rather than one (i.e. 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:

```
# Causal effect not present
set.seed(1701)
sim_test_data_causal_0_2SMR <- simulate_MR_data(n_participants = 10000,
                                                n_instruments = 100,
                                                n_datasets = 1,
                                                prop_invalid = 0,
                                                causal_effect = FALSE,
                                                rand_error = FALSE,
                                                two_sample = TRUE)

test_plot_tib_causal_0_2SMR <- extract_models(sim_test_data_causal_0_2SMR)[[1]]

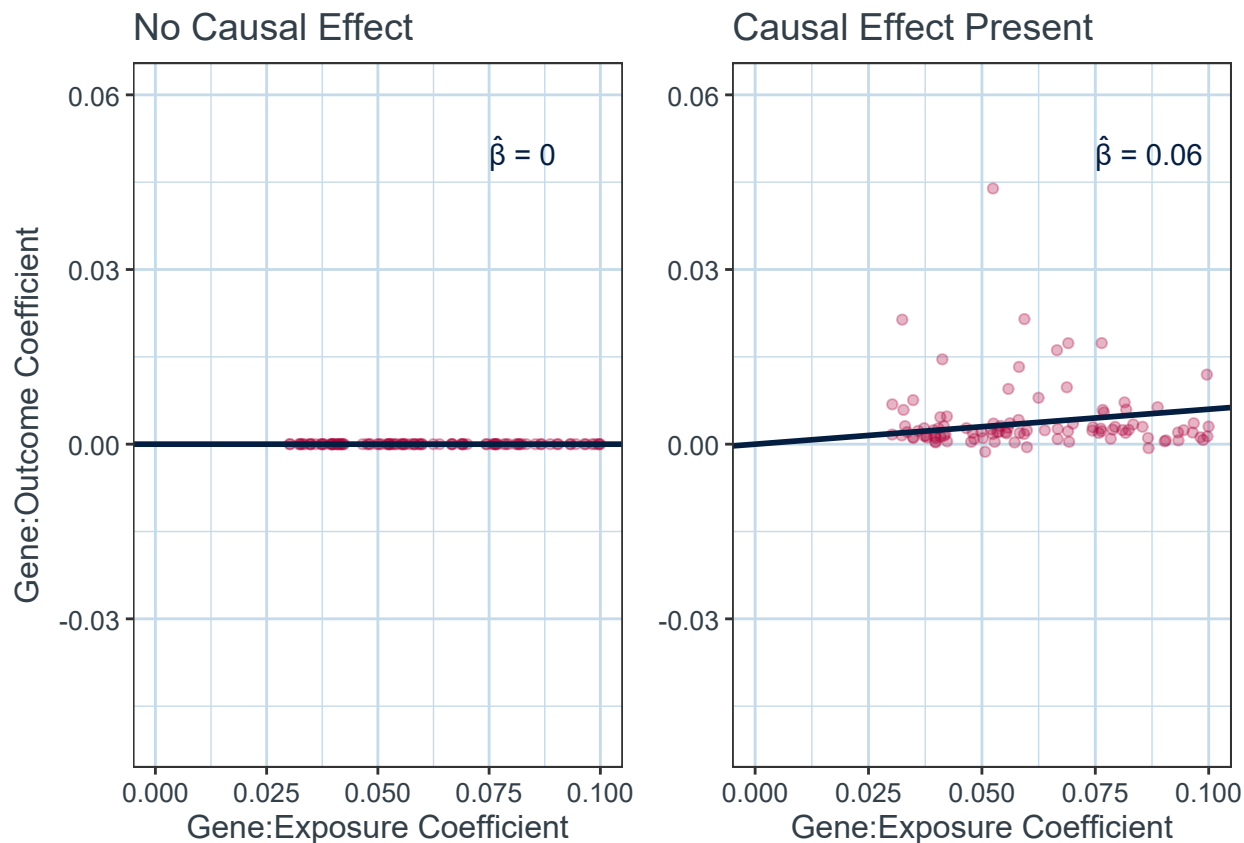
test_plot_causal_0_2SMR <- plot_GY_GX(test_plot_tib_causal_0_2SMR,
                                     plot_title = "No Causal Effect")

# Causal effect present
set.seed(1701)
sim_test_data_causal_1_2SMR <- simulate_MR_data(n_participants = 10000,
                                                n_instruments = 100,
                                                n_datasets = 1,
                                                prop_invalid = 0,
                                                causal_effect = TRUE,
                                                rand_error = FALSE,
                                                two_sample = TRUE)

test_plot_tib_causal_1_2SMR <- extract_models(sim_test_data_causal_1_2SMR)[[1]]

test_plot_causal_1_2SMR <- plot_GY_GX(test_plot_tib_causal_1_2SMR,
                                     plot_title = "Causal Effect Present") +
  theme(axis.title.y = element_blank())

plot_grid(test_plot_causal_0_2SMR,
          test_plot_causal_1_2SMR,
          ncol = 2,
          rel_widths = c(1.05, 1))
```



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:

```
# Causal effect not present
set.seed(1701)
sim_test_data_causal_0_inval <- simulate_MR_data(n_participants = 10000,
  n_instruments = 100,
  n_datasets = 1,
  prop_invalid = 0.1,
  causal_effect = FALSE,
  rand_error = FALSE,
  two_sample = FALSE)

test_plot_tib_causal_0_inval <- extract_models(sim_test_data_causal_0_inval)[[1]]

test_plot_causal_0_inval <- plot_GY_GX(test_plot_tib_causal_0_inval, plot_title = "No Causal Effect")

# Causal effect present
set.seed(1701)
sim_test_data_causal_1_inval <- simulate_MR_data(n_participants = 10000,
  n_instruments = 100,
  n_datasets = 1,
  prop_invalid = 0.1,
  causal_effect = TRUE,
  rand_error = FALSE,
```

```

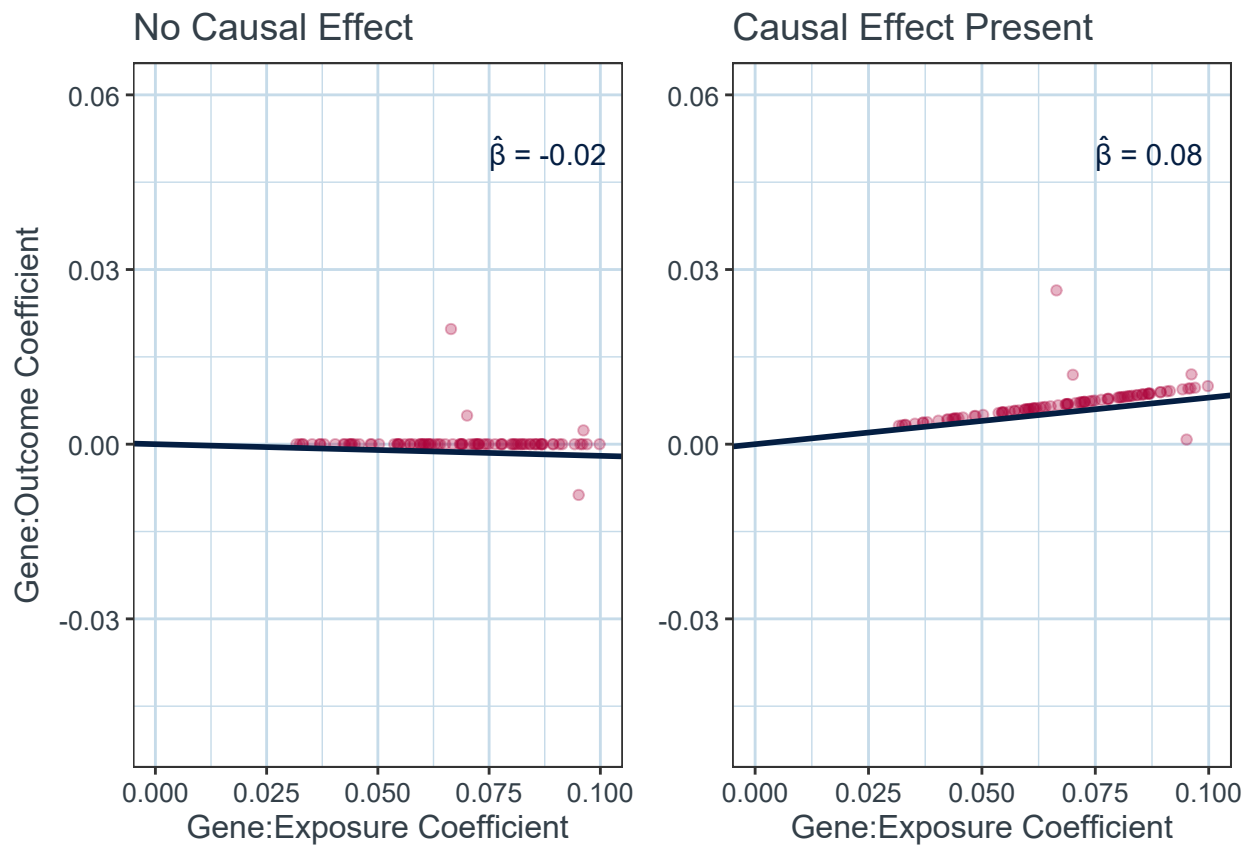
two_sample = FALSE)

test_plot_tib_causal_1_inval <- extract_models(sim_test_data_causal_1_inval)[[1]]

test_plot_causal_1_inval <- plot_GY_GX(test_plot_tib_causal_1_inval, plot_title = "Causal Effect Present",
  theme(axis.title.y = element_blank()))

plot_grid(test_plot_causal_0_inval,
  test_plot_causal_1_inval,
  ncol = 2,
  rel_widths = c(1.05, 1))

```



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:

```

# Causal effect not present
set.seed(1701)
sim_test_data_causal_0_unbal <- simulate_MR_data(n_participants = 10000,
  n_instruments = 100,
  n_datasets = 1,
  prop_invalid = 0.2,
  causal_effect = FALSE,
  rand_error = FALSE,
  balanced_pleio = FALSE,
  two_sample = FALSE)

```

```

test_plot_tib_causal_0_unbal <- extract_models(sim_test_data_causal_0_unbal)[[1]]

test_plot_causal_0_unbal <- plot_GY_GX(test_plot_tib_causal_0_unbal, plot_title = "No Causal Effect")

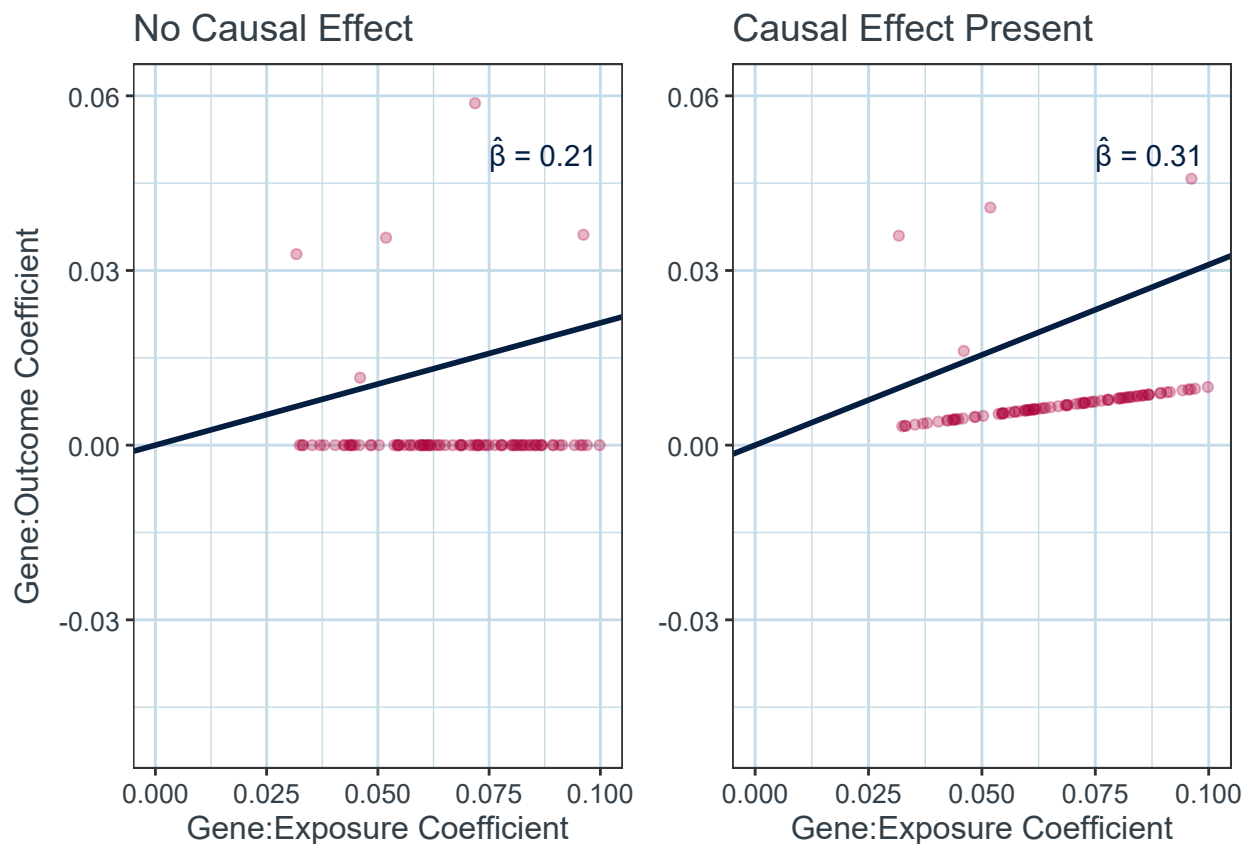
# Causal effect present
set.seed(1701)
sim_test_data_causal_1_unbal <- simulate_MR_data(n_participants = 10000,
                                                n_instruments = 100,
                                                n_datasets = 1,
                                                prop_invalid = 0.2,
                                                causal_effect = TRUE,
                                                rand_error = FALSE,
                                                balanced_pleio = FALSE,
                                                two_sample = FALSE)

test_plot_tib_causal_1_unbal <- extract_models(sim_test_data_causal_1_unbal)[[1]]

test_plot_causal_1_unbal <- plot_GY_GX(test_plot_tib_causal_1_unbal, plot_title = "Causal Effect Present",
                                       theme(axis.title.y = element_blank()))

plot_grid(test_plot_causal_0_unbal,
          test_plot_causal_1_unbal,
          ncol = 2,
          rel_widths = c(1.05, 1))

```



The variable ϕ represents additional pleiotropic effects of each invalid instrument when the InSIDE assumption (INstrument Strength Independent of Direct Effect) 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 al¹ modeled ϕ 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 α versus gene:outcome coefficients with random error terms set to zero:

```
# Check violating InSIDE assumption results in distorted
# estimation of pleiotropic effects
# N.B. cluster around alpha = 0 represents valid instruments with
# no pleiotropic effects
set.seed(1701)
sim_test_data_phi_0 <- simulate_MR_data(n_participants = 100000,
                                       n_instruments = 100,
                                       n_datasets = 1,
                                       prop_invalid = 0.3,
                                       causal_effect = FALSE,
                                       rand_error = FALSE,
                                       balanced_pleio = FALSE,
                                       InSIDE_satisfied = TRUE)

set.seed(1701)
sim_test_data_phi_1 <- simulate_MR_data(n_participants = 100000,
                                       n_instruments = 100,
                                       n_datasets = 1,
                                       prop_invalid = 0.3,
                                       causal_effect = FALSE,
                                       rand_error = FALSE,
                                       balanced_pleio = FALSE,
                                       InSIDE_satisfied = FALSE)

test_plot_tib_phi_0 <- extract_models(sim_test_data_phi_0)[[1]]
test_plot_tib_phi_1 <- extract_models(sim_test_data_phi_1)[[1]]

test_plot_phi_0 <- test_plot_tib_phi_0 %>%
  plot_template() +
```

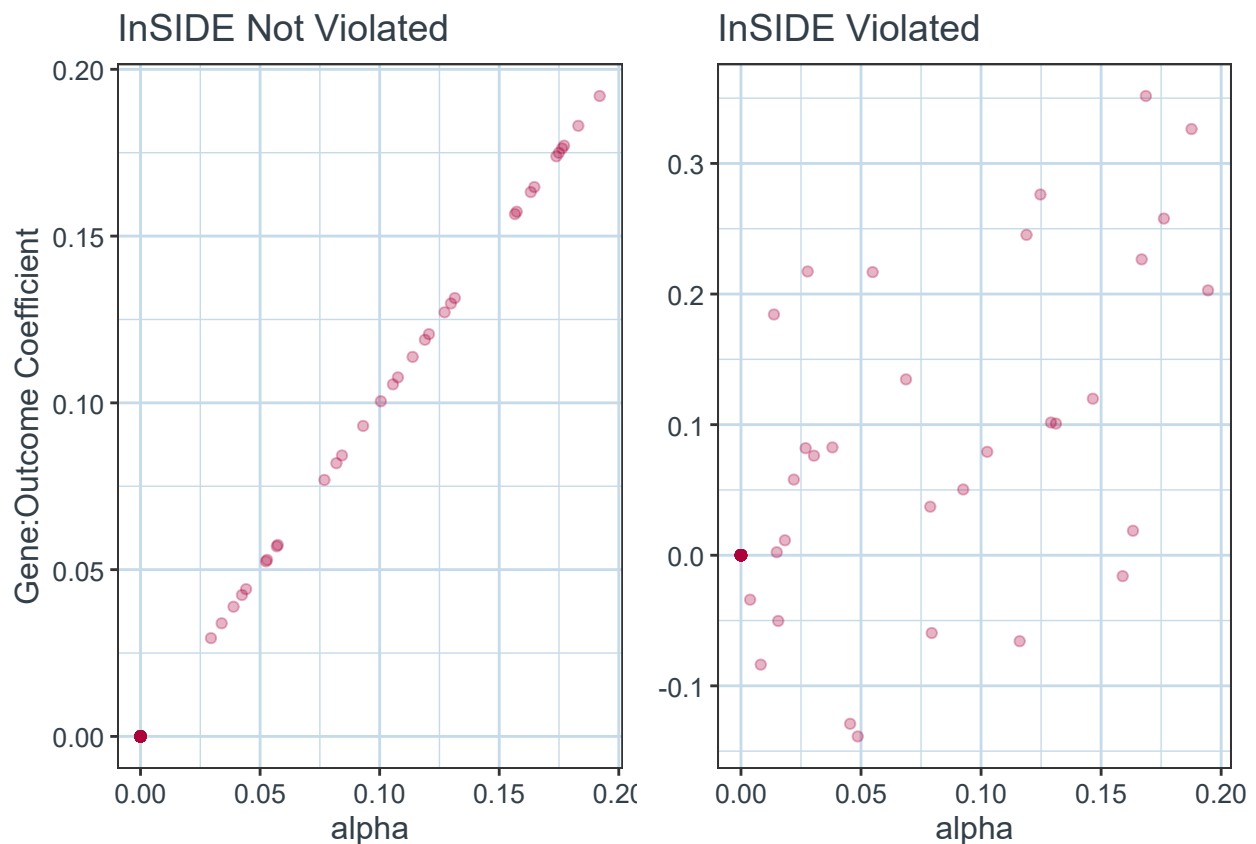
```

aes(x = alpha, y = coeff_G_Y) +
geom_point(colour = edin_bright_red_hex, alpha = 0.3) +
labs(title = "InSIDE Not Violated",
      y = "Gene:Outcome Coefficient")

test_plot_phi_1 <- test_plot_tib_phi_1 %>%
  plot_template() +
  aes(x = alpha, y = coeff_G_Y) +
  geom_point(colour = edin_bright_red_hex, alpha = 0.3) +
labs(title = "InSIDE Violated",
      y = "Gene:Outcome Coefficient") +
  theme(axis.title.y = element_blank())

plot_grid(test_plot_phi_0,
          test_plot_phi_1,
          ncol = 2,
          rel_widths = c(1.05, 1))

```



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


```

# Check violating InSIDE assumption results in distorted
# estimation of pleiotropic effects
# N.B. cluster around alpha = 0 represents valid instruments with
# no pleiotropic effects
set.seed(1701)
sim_test_data_phi_0 <- simulate_MR_data(n_participants = 100000,
                                       n_instruments = 100,
                                       n_datasets = 1,
                                       prop_invalid = 0.3,
                                       causal_effect = FALSE,
                                       rand_error = FALSE,
                                       balanced_pleio = FALSE,
                                       InSIDE_satisfied = TRUE)

set.seed(1701)
sim_test_data_phi_1 <- simulate_MR_data(n_participants = 100000,
                                       n_instruments = 100,
                                       n_datasets = 1,
                                       prop_invalid = 0.3,
                                       causal_effect = FALSE,
                                       rand_error = FALSE,
                                       balanced_pleio = FALSE,
                                       InSIDE_satisfied = FALSE)

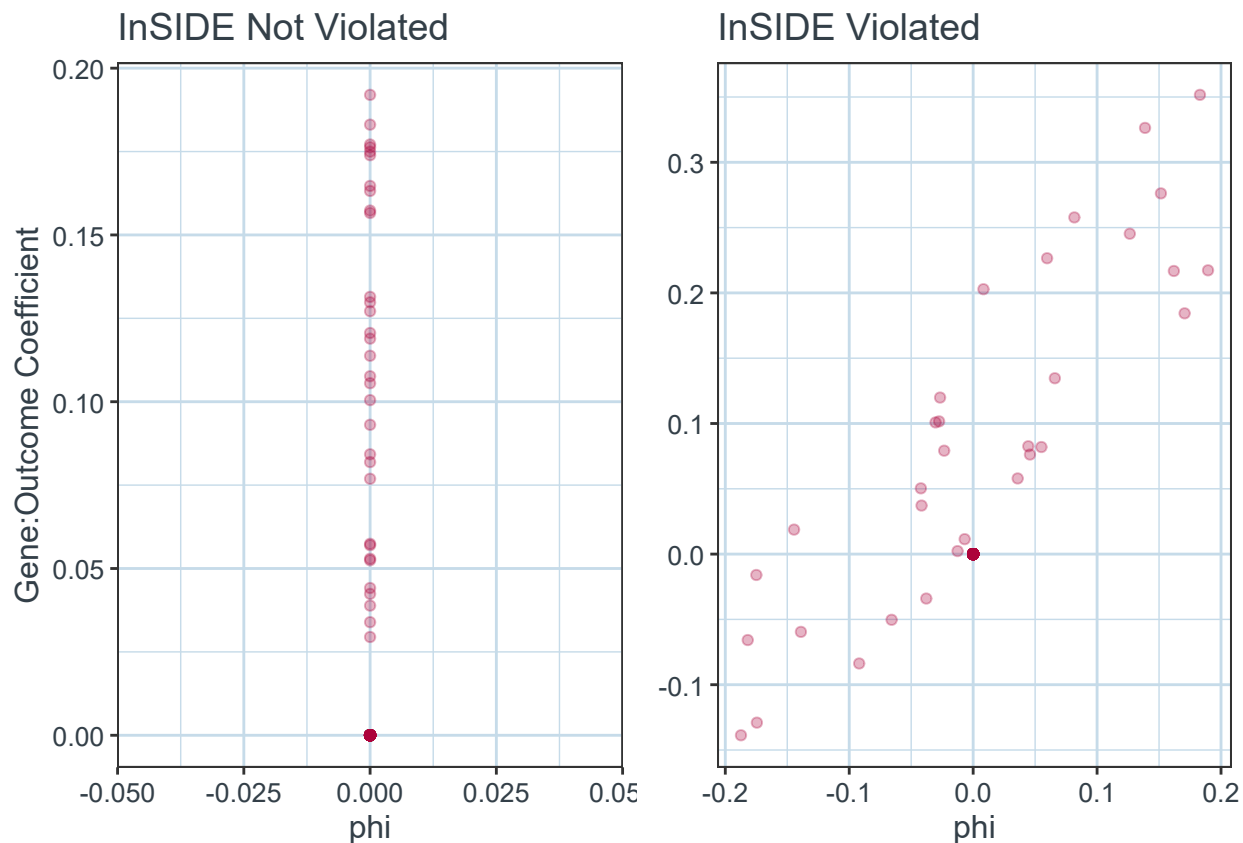
test_plot_tib_phi_0 <- extract_models(sim_test_data_phi_0)[[1]]
test_plot_tib_phi_1 <- extract_models(sim_test_data_phi_1)[[1]]

test_plot_phi_0 <- test_plot_tib_phi_0 %>%
  plot_template() +
  aes(x = phi, y = coeff_G_Y) +
  geom_point(colour = edin_bright_red_hex, alpha = 0.3) +
labs(title = "InSIDE Not Violated",
     y = "Gene:Outcome Coefficient")

test_plot_phi_1 <- test_plot_tib_phi_1 %>%
  plot_template() +
  aes(x = phi, y = coeff_G_Y) +
  geom_point(colour = edin_bright_red_hex, alpha = 0.3) +
labs(title = "InSIDE Violated",
     y = "Gene:Outcome Coefficient") +
  theme(axis.title.y = element_blank())

plot_grid(test_plot_phi_0,
          test_plot_phi_1,
          ncol = 2,
          rel_widths = c(1.05, 1))

```



```
#phi on y, not alpha
```

```
# 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)

# --- Need to comment out model.dir in Hevo_results --- #

get_summary_MR_tib_row <- function(model_list){

# get_summary_MR_tib_row <- function(n_participants = 1000,                # update to 10,000 for final
#                                n_instruments = 25,                      # update to 10,000 for final
#                                n_datasets = 100,
#                                prop_invalid = 0.1,
#                                causal_effect = TRUE,
#                                balanced_pleio = TRUE,
```

```

#                               InSIDE_satisfied = TRUE,
#                               rand_error = TRUE,
#                               two_sample = TRUE,
#                               beta_val = 0.1,
#                               allele_freq_min = 0.01,
#                               allele_freq_max = 0.99,
#                               gamma_min = 0.03,
#                               gamma_max = 0.1,
#                               alpha_min = -0.2,
#                               alpha_max = 0.2,
#                               phi_min = -0.2,
#                               phi_max = 0.2){

# Generate data as per Bowden models with helper function
# dataset_list <- simulate_MR_data(n_participants = n_participants,
#                                   n_instruments = n_instruments,
#                                   n_datasets = n_datasets,
#                                   prop_invalid = prop_invalid,
#                                   causal_effect = causal_effect,
#                                   balanced_pleio = balanced_pleio,
#                                   InSIDE_satisfied = InSIDE_satisfied,
#                                   rand_error = rand_error,
#                                   two_sample = two_sample,
#                                   beta_val = beta_val,
#                                   allele_freq_min = allele_freq_min,
#                                   allele_freq_max = allele_freq_max,
#                                   gamma_min = gamma_min,
#                                   gamma_max = gamma_max,
#                                   alpha_min = alpha_min,
#                                   alpha_max = alpha_max,
#                                   phi_min = phi_min,
#                                   phi_max = phi_max)

# Use data to create model per dataset
# model_list <- extract_models(dataset_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)

```

```

output_tib_row$N <- n_datasets

# 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_Q = as.double(),
                      WME_Q_df = as.double(),
                      WME_Q_pval = as.double(),
                      WME_nsnp = as.integer(),
                      Hevo_est = as.double(),
                      Hevo_se = as.double(),
                      Hevo_sd = as.double(),
                      Hevo_2.5 = as.double(),
                      Hevo_25 = as.double(),
                      Hevo_50 = as.double(),
                      Hevo_75 = as.double(),
                      Hevo_97.5 = as.double(),
                      Hevo_n_eff = as.double(),
                      Hevo_n_Rhat = as.double(),
                      Hevo_z_stat = as.double(),
                      Hevo_pval = 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]]$prop_invalid)
  R2_stat <- min(model_list[[dataset]]$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()

results_tib[dataset, ]$WME_est <- WME_results$b
results_tib[dataset, ]$WME_se <- WME_results$se
results_tib[dataset, ]$WME_pval <- WME_results$pval
results_tib[dataset, ]$WME_Q <- WME_results$Q
results_tib[dataset, ]$WME_Q_df <- WME_results$Q_df
results_tib[dataset, ]$WME_Q_pval <- WME_results$Q_pval
results_tib[dataset, ]$WME_nsnpp <- WME_results$nsnpp

# 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, ]$Hevo_2.5 <- Hevo_results$summary["theta", "2.5%"]
results_tib[dataset, ]$Hevo_25 <- Hevo_results$summary["theta", "25%"]
results_tib[dataset, ]$Hevo_50 <- Hevo_results$summary["theta", "50%"]
results_tib[dataset, ]$Hevo_75 <- Hevo_results$summary["theta", "75%"]
results_tib[dataset, ]$Hevo_97.5 <- Hevo_results$summary["theta", "97.5%"]
results_tib[dataset, ]$Hevo_n_eff <- Hevo_results$summary["theta", "n_eff"]
results_tib[dataset, ]$Hevo_n_Rhat <- Hevo_results$summary["theta", "Rhat"]

}

results_tib <- results_tib %>%
  mutate(Hevo_causal_detected = !(Hevo_2.5 < 0 & Hevo_97.5 > 0))

# https://pmc.ncbi.nlm.nih.gov/articles/PMC10616660/
# https://mr-dictionary.mrcieu.ac.uk/term/r-squared/#:~:text=The%20R%2Dsquared%20can%20also,and%20sam
output_tib_row <- results_tib %>%
  summarise(N = n_datasets,
            Prop_Invalid = prop_invalid,
            F_Stat = R2_stat * n_participants / n_instruments,
            R2_stat = 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_Pos_Rate = sum(Hevo_causal_detected) / n_datasets#[Hevo_pval < (0.05/n_datasets) ] )
      #Hevo_Pos_Rate = length(Hevo_pval[Hevo_pval < (0.05/n_datasets) ] ) / n_datasets
    ) %>%
    mutate(across(where(is.double), round, 3))

  return(output_tib_row)
  #return(results_tib)
}

```

```

set.seed(14101583)
test_tib_summ_MR_data <- simulate_MR_data(n_participants = 10000,
                                           n_instruments = 25,
                                           n_datasets = 5,
                                           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 <- extract_models(test_tib_summ_MR_data)

# first element of each list element
# map(test_tib_summ_MR_data, first) %>%
#   map(., first)

#test_tib_summ_MR_models[[1]]

#test_tib_summ_MR_row <- get_summary_MR_tib_row(test_tib_summ_MR_models)

#
# # Causal effect present
# set.seed(14101583)
# sim_test_data_causal_2 <- simulate_MR_data(n_participants = 100000,
#                                             n_instruments = 25,
#                                             n_datasets = 1,
#                                             prop_invalid = 0.3,
#                                             beta_val = 1, # size of causal effect
#                                             causal_effect = TRUE,
#                                             rand_error = FALSE,
#                                             two_sample = FALSE,
#                                             balanced_pleio = TRUE,
#                                             InSIDE_satisfied = TRUE,
#                                             allele_freq_min = 0.01,
#                                             allele_freq_max = 0.99,
#                                             gamma_min = 0.03, # size of pleiotropic effect
#                                             gamma_max = 0.1,
#                                             alpha_min = -0.2, # size of pleiotropic effect
#                                             alpha_max = 0.2)

```

```

#                                     alpha_max = 0.2,
#                                     phi_min = 0,
#                                     phi_max = 0)
#
# #sim_test_data_causal_2
#
# test_plot_tib_causal_2 <- extract_models(sim_test_data_causal_2)[[1]]
#
# plot_GY_GX(test_plot_tib_causal_2, plot_title = "Causal Effect Present", x_max = 0.3, y_max = 0.3)
#
# test_tib_summ_MR
#
#test_tib_summ_MR_row

```

Appendix C: Citation Search Strategy

1. 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/>