Simulations

The data generating model used was from Appendix 3 of Bowden et al (ref), and was as follows

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

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

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

for participants indexed by $i = 1, ..., N$, and genetic instruments indexed by $j = 1, ..., J$. 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 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. The Monte Carlo standard error for the mean estimates was 0.002 or less, and for the power was less than 0.5% in all cases.

The standard errors used in the IVW and MR-Egger methods in this simulation correspond to those used in the simulations of Bowden et al. [Bowden et al., 2015], but differ from those previously recommended for use in the IVW [Johnson, 2013; Burgess et al., 2013] and originally recommended for the MR-Egger regression method (see Web Appendix 2 above). The reason is that the IVW method corresponds to a fixed-effect meta-analysis of the ratio estimates from each genetic instrument. In the examples of this paper, there is heterogeneity in the ratio estimates, and so a fixedeffect analysis is inappropriate, and leads to overly precise confidence intervals and inflated Type 1 error rates. A conventional random-effects meta-analysis would not be wise, as the random-effects estimate upweights outlying estimates, inflating the influence of pleiotropic genetic instruments on the analysis. Hence we have reached a compromise, similar to that suggested by

Copas et al. [Henmi and Copas, 2010], that we take the point estimate from a fixed-effect analysis, but allow confidence intervals to be inflated by heterogeneity as per a random-effects analysis. This is achieved by performing a weighted regression as described in the description of MR-Egger regression, but not setting the residual standard error in the regression to be 1, as is recommended in meta-analysis to correspond to a fixed-effect analysis (unless the estimate of the residual standard error is less than 1, in which case we divide the standard errors of the coefficients by the estimate of the residual standard error to avoid over-precision). This is equivalent to a multiplicative random-effects model [Thompson and Sharp, 1999].

```r
wme_model_sim <- 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){

  set.seed(1701)


  # Initialise blank lists to receive datasets for
  # each of U (unmeasured confounding exposures),
  # x (exposures) and Y (outcomes)
  U_list <- list()
  X_list <- list()
  Y_list <- list()


  # Assign features common to all datasets
  beta <- if_else(causal_effect == TRUE, # size of causal effect
                  0.1,
                  0)


  # Create N datasets for each of U, X and Y as matrices,
  # 1 row per participant, 1 column per genetic instrument,
  # initialise with error term drawn from standard normal
  # distribution
  for(n in 1:n_datasets){

    U_mat <-  matrix(rnorm(n = n_participants * n_instruments), # unmeasured confounders
                     nrow = n_participants,
                     ncol = n_instruments)

    X_mat <-  matrix(rnorm(n = n_participants * n_instruments), # exposures
                     nrow = n_participants,
                     ncol = n_instruments)

    Y_mat <-  matrix(rnorm(n = n_participants * n_instruments), # outcomes
                     nrow = n_participants,
                     ncol = n_instruments)

    # Create matrix of genotypes, 0 = reference,
    # 1 = effect allele. Probability of effect
    # allele for each instrument is set per dataset,
    # value set at random

    G_prob_vect <- runif(n = n_instruments,
                         min = 0,
                         max = 1)

    # G_mat <- matrix(rep(G_prob_vect, n_participants),
    #                 nrow = n_participants,
    #                 ncol = n_instruments,
    #                 byrow = TRUE)

    G_mat <- matrix(rbinom(n = n_participants * n_instruments, # genotypes
                           size = 1,
                           prob = rep(G_prob_vect, n_participants)),
                    nrow = n_participants,
                    ncol = n_instruments,
                    byrow = TRUE)

    # Set which instruments 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 = 0.03,
                        max = 0.1)


    # 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){
      alpha_vect[j] <- ifelse(invalid_instrument_vect[j] == FALSE,
                              0,
                              ifelse(balanced_pleio == TRUE,
                                     runif(n = n_instruments,
                                           min = -0.2,
                                           max = 0.2),
                                     runif(n = n_instruments,
                                           min = 0,
                                           max = 0.2)
                                    )
                              )

      # Assign default phi = 0 unless unbalanced pleiotropy &
      # InSIDE assumption not satisfied & genetic instrument invalid
```

```
      if(balanced_pleio == FALSE & InSIDE_satisfied == FALSE){
        phi_vect[j] <- ifelse(invalid_instrument_vect[j] == FALSE,
                              0,
                              runif(n = 1,
                                    min = -0.2,
                                    max = 0.2)
                             )

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

    # Complete U matrix
    U_mat[ ,j] <- phi_vect[j] + U_mat[ ,j]
    U_list[[n]] <- U_mat


    # Complete X matrix
    X_mat[ , j] <- gamma_vect[j]*G_mat[ , j] + U_mat[ , j] + X_mat[ , j]
    X_list[[n]] <- X_mat

    # Complete Y matrix
    Y_mat[ ,j] <- alpha_vect[j] + beta*X_mat[ ,j] + U_mat[ ,j] + Y_mat[ ,j]
    Y_list[[n]] <- Y_mat

  }

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

  return(combined_list)

}
```

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

  summ_X_list <- list()
  summ_Y_list <- list()

  for(mat in 1:length(sim$X)){
    # Get mean/SD cols for X, grouped by dataset + instrument
    summ_X_list[[mat]] <- sim$X[[mat]] %>%
      as_tibble(.name_repair = "unique") %>% # unique names = ...1, ...2 etc for each instrument
      mutate(Dataset = mat) %>%
      pivot_longer(cols = starts_with("..."),
                   names_prefix = "...",
                   names_to = "Instrument") %>%
      mutate(Instrument = as.integer(Instrument)) %>%
```

```r
      group_by(Dataset,
               Instrument) %>%
      summarise(Mean_X = mean(value),
                SD_X = sd(value),
                .groups = "keep")

    # Get mean/SD cols for Y, grouped by dataset + instrument
    summ_Y_list[[mat]] <- sim$Y[[mat]] %>%
      as_tibble(.name_repair = "unique") %>%
      mutate(Dataset = mat) %>%
      pivot_longer(cols = starts_with("..."),
                   names_prefix = "...",
                   names_to = "Instrument") %>%
      mutate(Instrument = as.integer(Instrument)) %>%
      group_by(Dataset,
               Instrument) %>%
      summarise(Mean_Y = mean(value),
                SD_Y = sd(value),
                .groups = "keep")

  }

  # Combine matrices for X+Y into single tibble and return
  summ_X_tib <- summ_X_list %>%
    list_rbind()

  summ_Y_tib <- summ_Y_list %>%
    list_rbind()

  summ_XY_tib <- summ_X_tib %>%
    full_join(summ_Y_tib) %>%
    mutate(Instrument = as_factor(Instrument)) # to aid plotting

  return(summ_XY_tib)

}
```

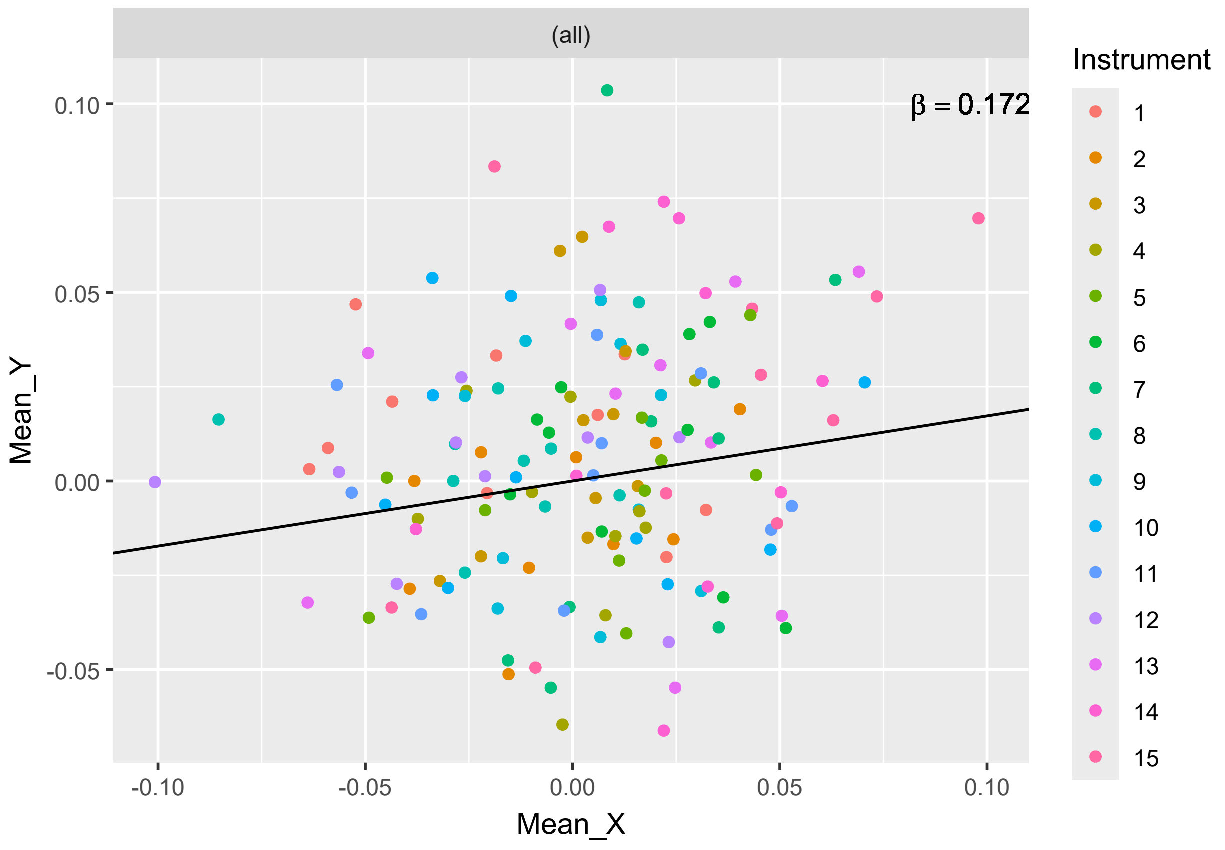
```r
sim_test_data <- wme_model_sim(n_participants = 1000,
                               n_instruments = 15,
                               n_datasets = 10,
                               prop_invalid = 0.1,
                               causal_effect = TRUE,
                               balanced_pleio = TRUE,
                               InSIDE_satisfied = TRUE)


test_plot_tib <- extract_plotting_XY(sim_test_data)
```

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

## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## Joining with `by = join_by(Dataset, Instrument)`
## * `` -> `...1`
## * `` -> `...2`
## * `` -> `...3`
## * `` -> `...4`
## * `` -> `...5`
## * `` -> `...6`
## * `` -> `...7`
## * `` -> `...8`
## * `` -> `...9`
## * `` -> `...10`
## * `` -> `...11`
## * `` -> `...12`
## * `` -> `...13`
## * `` -> `...14`
## * `` -> `...15`
```

```
test_plot_tib %>%
  group_by(NULL) %>% # needed for Gradient across all datasets combined
  #group_by(Dataset) %>%
  mutate(Gradient = round(coefficients(lm(Mean_Y ~ 0 + Mean_X))[1], 5),
         Intercept = 0) %>%
  ggplot() +
  aes(x = Mean_X, y = Mean_Y) +
  geom_point(aes(colour = Instrument)) +
  geom_abline(aes(intercept = 0,
                  slope = Gradient),) +
  geom_text(aes(x = 0.1, # labels with gradient (causal effect estimate)
                y = 0.1,
                label = paste0("beta == ", Gradient)),
            parse = TRUE) +
  facet_wrap(~NULL) #+
```



```
#facet_wrap(~Dataset) #+
```