Package 'MR.RGM'

August 27, 2026

Title Fitting Multivariate Bidirectional Mendelian Randomization Networks Using Bayesian Directed Cyclic Graphical Models

Version 0.1.1

Description Addressing a central challenge encountered in Mendelian randomization (MR) studies, where MR primarily focuses on discerning the effects of individual exposures on specific outcomes and establishes causal links between them. Using a network-based methodology, the intricacy involving interdependent outcomes due to numerous factors has been tackled through this routine. Based on Ni et al. (2018) <doi:10.1214/17-BA1087>, 'MR.RGM' extends to a broader exploration of the causal landscape by leveraging on network structures and involves the construction of causal graphs that capture interactions between response variables and consequently between responses and instrument variables. The resulting Graph visually represents these causal connections, showing directed edges with effect sizes labeled. 'MR.RGM' facilitates the navigation of various data availability scenarios effectively by accommodating three input formats, i.e., individual-level data and two types of summary-level data. The method also optionally incorporates measured covariates (when available) and allows flexible modeling of the error variance structure, including correlated errors that may reflect unmeasured confounding among responses. In the process, causal effects, adjacency matrices, and other essential parameters of the complex biological networks, are estimated. Besides, 'MR.RGM' provides uncertainty quantification for specific network structures among response variables. Parts of the Inverse Wishart sampler are adapted from the econ722 repository by DiTraglia (GPL-2.0).

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.3

URL https://github.com/bitansa/MR.RGM

BugReports https://github.com/bitansa/MR.RGM/issues

Imports Rcpp, stats, igraph, GIGrvg

Suggests MASS

LinkingTo Rcpp, RcppArmadillo, RcppDist, GIGrvg

NeedsCompilation yes

Author Bitan Sarkar [aut, cre],
Yang Ni [aut]

Maintainer Bitan Sarkar <bitansarkar010899@gmail.com>

Repository CRAN

Date/Publication 2026-08-27 12:10:02 UTC

Contents

NetworkMotif 2
RGM 5

Index **18**

`NetworkMotif` *Estimating the uncertainty of a specified network*

Description

The NetworkMotif function facilitates uncertainty quantification. Specifically, it determines the proportion of posterior samples that contains the given network structure. To use this function, users may use the GammaPst output obtained from the RGM function.

Usage

```
NetworkMotif(Gamma, GammaPst)
```

Arguments

`Gamma`	A matrix of dimension p * p that signifies a specific network structure among the response variables, where p represents the number of response variables. This matrix is the focus of uncertainty quantification.
`GammaPst`	An array of dimension p * p * n_pst, where n_pst is the number of posterior samples and p denotes the number of response variables. It comprises the posterior samples of the causal network among the response variables. This input might be obtained from the RGM function. Initially, execute the RGM function and save the resulting GammaPst. Subsequently, utilize this stored GammaPst as input for this function.

Value

The NetworkMotif function calculates the uncertainty quantification for the provided network structure. A value close to 1 indicates that the given network structure is frequently observed in the posterior samples, while a value close to 0 suggests that the given network structure is rarely observed in the posterior samples.

References

Ni, Y., Ji, Y., & Müller, P. (2018). Reciprocal graphical models for integrative gene regulatory network analysis. *Bayesian Analysis*, **13(4)**, 1095-1110. doi:10.1214/17BA1087.

Examples

```
#' # ------------------------------------------------------------

# Example 1:
# Run NetworkMotif to do uncertainty quantification for a given network among the response variable

# Data Generation
set.seed(9154)

# Number of data points
n = 10000

# Number of response variables and number of instrument variables
p = 3
k = 4

# Initialize causal interaction matrix between response variables
A = matrix(sample(c(-0.1, 0.1), p^2, replace = TRUE), p, p)

# Diagonal entries of A matrix will always be 0
diag(A) = 0

# Make the network sparse
A[sample(which(A!=0), length(which(A!=0))/2)] = 0

# Create D matrix (Indicator matrix where each row corresponds to a response variable
# and each column corresponds to an instrument variable)
D = matrix(0, nrow = p, ncol = k)

# Manually assign values to D matrix
D[1, 1:2] = 1  # First response variable is influenced by the first 2 instruments
D[2, 3] = 1    # Second response variable is influenced by the 3rd instrument
D[3, 4] = 1    # Third response variable is influenced by the 4th instrument


# Initialize B matrix
B = matrix(0, p, k)  # Initialize B matrix with zeros

# Calculate B matrix based on D matrix
for (i in 1:p) {
  for (j in 1:k) {
    if (D[i, j] == 1) {
      B[i, j] = 1  # Set B[i, j] to 1 if D[i, j] is 1
    }
  }
}

Sigma = diag(p)

Mult_Mat = solve(diag(p) - A)
```

```
Variance = Mult_Mat %*% Sigma %*% t(Mult_Mat)

# Generate instrument data matrix
X = matrix(rnorm(n * k, 0, 1), nrow = n, ncol = k)

# Initialize response data matrix
Y = matrix(0, nrow = n, ncol = p)

# Generate response data matrix based on instrument data matrix
for (i in 1:n) {

    Y[i, ] = MASS::mvrnorm(n = 1, Mult_Mat %*% B %*% X[i, ], Variance)

}


# Apply RGM on individual level data with Spike and Slab Prior
Output = RGM(X = X, Y = Y, D = D, prior = "Spike and Slab", SigmaStarModel = "diagonal")

# Store GammaPst
GammaPst = Output$GammaPst

# Define a function to create smaller arrowheads
smaller_arrowheads = function(graph) {
    igraph::E(graph)$arrow.size = 1  # Adjust the arrow size value as needed
    return(graph)
}


# Plot the true graph structure between response variables
plot(smaller_arrowheads(igraph::graph_from_adjacency_matrix((A != 0) * 1,
  mode = "directed")), layout = igraph::layout_in_circle, main = "True Graph")

# Start with a random subgraph
Gamma = matrix(0, nrow = p, ncol = p)
Gamma[2, 1] = 1

# Plot the subgraph to get an idea about the causal network
plot(smaller_arrowheads(igraph::graph_from_adjacency_matrix(Gamma,
         mode = "directed")), layout = igraph::layout_in_circle,
            main = "Subgraph")


# Do uncertainty quantification for the subgraph
NetworkMotif(Gamma = Gamma, GammaPst = GammaPst)
```

RGM *Fitting Multivariate Bidirectional Mendelian Randomization Networks Using Bayesian Directed Cyclic Graphical Models*

Description

The RGM function transforms causal inference by merging Mendelian randomization and network-based methods, enabling the creation of comprehensive causal graphs within complex biological systems. RGM accommodates varied data contexts with three input options: individual-level data (X, Y matrices), summary-level data including Syy, Syx, and Sxx matrices, and intricate data with challenging cross-correlations, utilizing Sxx, Beta, and SigmaHat matrices. For the latter input, data centralization is necessary. Users can select any of these data formats to suit their needs and don't have to specify all of them, allowing flexibility based on data availability. In addition, RGM optionally allows the inclusion of covariate information. When individual-level data are provided, covariates may be supplied either through the raw covariate matrix U or through summary matrices Syu, Sxu, and Suu. When only summary-level data are used, the covariate summary matrices Syu, Sxu, and Suu must be provided explicitly. Covariate information is optional and the method remains valid in its absence. Crucial inputs encompass "D" (a matrix indicating which IV is affecting which response) and "n" (total observations, only required for summary level data), amplified by customizable parameters that refine analysis. Additionally, users can tailor the analysis by setting parameters such as "nIter" (number of MCMC iterations), "nBurnin" (number of discarded samples during burn-in for convergence), and "Thin" (thinning of posterior samples). These customizable parameters enhance the precision and relevance of the analysis. RGM provides essential causal effect/strength estimates between response variables, between response and instrument variables and between response and covariates. Moreover, it furnishes adjacency matrices, visually mapping causal graph structures. These outputs empower researchers to untangle intricate relationships within biological networks, fostering a holistic understanding of complex systems. The function also returns a graph object representing the estimated causal network between the responses. This graph consists of directed edges that indicate causal effects, with effect sizes displayed on the edges. The edges are determined based on a threshold of 0.5 for inclusion. Users can visualize the network by plotting this object, which helps in interpreting the causal relationships between response variables. The output includes AEst, BEst, CEst, A0Est, B0Est, GammaEst, TauEst, PhiEst, EtaEst, tAEst, tBEst, SigmaEst, ZEst, RhoEst, and PsiEst, representing the posterior means of the corresponding quantities. Additionally, LLPst and GammaPst represent posterior samples. The adjacency matrices zAEst and zBEst are obtained by thresholding GammaEst and TauEst, respectively. When a full error variance–covariance matrix is estimated, the off-diagonal elements of SigmaEst may be interpreted as evidence of unmeasured confounding between responses.

Usage

```
RGM(
  X = NULL,
  Y = NULL,
  U = NULL,
  Syy = NULL,
  Syx = NULL,
  Sxx = NULL,
```

```
  Syu = NULL,
  Sxu = NULL,
  Suu = NULL,
  Beta = NULL,
  SigmaHat = NULL,
  D,
  n,
  nIter = 10000,
  nBurnin = 2000,
  Thin = 1,
  prior = c("Threshold", "Spike and Slab"),
  SigmaStarModel = c("diagonal", "IW", "SSSL"),
  aRho = 3,
  bRho = 1,
  nu1 = 0.001,
  aPsi = 0.5,
  bPsi = 0.5,
  nu2 = 1e-04,
  aSigma = 0.01,
  bSigma = 0.01,
  PropVarA = 0.01,
  PropVarB = 0.01
)
```

Arguments

X	A matrix of dimension n * k. Each row represents a distinct observation, and each column corresponds to a specific instrumental variable. The default value is set to NULL.
Y	A matrix of dimension n * p. Each row represents a specific observation, and each column corresponds to a particular response variable. The default value is set to NULL.
U	A matrix of dimension n * l. Each row represents a distinct observation, and each column corresponds to a covariate. The default value is set to NULL.
Syy	A matrix of dimension p * p, where "p" is the number of response variables. It is calculated as t(Y) %*% Y / n, where "Y" represents the response data matrix and "n" is the number of observations.
Syx	A matrix of dimension p * k, where "p" is the number of response variables, and "k" is the number of instrumental variables. It is calculated as t(Y) %*% X / n, where "Y" represents the response data matrix, "X" represents the instrumental data matrix, and "n" is the number of observations.
Sxx	A matrix of dimension k * k, where "k" is the number of instrumental variables. It is derived as t(X) %*% X / n, where "X" represents the instrumental data matrix and "n" is the number of observations.
Syu	A matrix of dimension p * l, where "p" is the number of response variables and "l" is the number of covariates. It is calculated as t(Y) %*% U / n, where "Y" represents the response data matrix, "U" represents the covariate matrix, and "n" is the number of observations.

`Sxu` A matrix of dimension k * l, where "k" is the number of instrumental variables and "l" is the number of covariates. It is calculated as t(X) %*% U / n, where "X" represents the instrument data matrix, "U" represents the covariate matrix, and "n" is the number of observations.

`Suu` A matrix of dimension l * l, where "l" is the number of covariates. It is calculated as t(U) %*% U / n, where "U" represents the covariate matrix and "n" is the number of observations.

`Beta` A matrix of dimension p * k, where each row corresponds to a specific response variable and each column pertains to an instrumental variable. Each entry represents the regression coefficient of the response variable on the instrumental variable. When using Beta as input, ensure that both Y (response data) and X (instrument data) are centered before calculating Beta, Sxx, and SigmaHat.

`SigmaHat` A matrix of dimension p * k. Each row corresponds to a specific response variable, and each column pertains to an instrumental variable. Each entry represents the mean square error of the regression between the response and the instrumental variable. As with Beta, ensure that both Y and X are centered before calculating SigmaHat.

`D` A binary indicator matrix of dimension p * k, where each row corresponds to a response variable, and each column corresponds to an instrumental variable. The entry `D[i, j]` is 1 if instrumental variable j affects response variable i, and 0 otherwise. For each response variable, there must be at least one instrumental variable that affects only that response (i.e., for each row in D, there must be at least one column with 1, and that column must have zeros in all other rows). If you use `Syy`, `Beta`, and `SigmaHat` as inputs, this condition must be satisfied to run this algorithm. If this condition is not met, an error will be thrown. However, if using `X`, `Y` or `Syy`, `Syx`, `Sxx` as inputs, a warning will be issued if the condition is violated, but the method will still proceed.

`n` A positive integer input representing the count of data points or observations in the dataset. This input is only required when summary level data is used as input.

`nIter` A positive integer input representing the number of MCMC (Markov Chain Monte Carlo) sampling iterations. The default value is set to 10,000.

`nBurnin` A non-negative integer input representing the number of samples to be discarded during the burn-in phase of MCMC sampling. It's important that nBurnin is less than nIter. The default value is set to 2000.

`Thin` A positive integer input denoting the thinning factor applied to posterior samples. Thinning reduces the number of samples retained from the MCMC process for efficiency. Thin should not exceed (nIter - nBurnin). The default value is set to 1.

`prior` A character input specifying the prior assumption on the graph structure. Options are `"Threshold"` and `"Spike and Slab"`. The default option is `"Spike and Slab"`.

`SigmaStarModel` A character input specifying the assumed structure of the error covariance matrix in the model. It provides three options: `"SSSL"` (stochastic spike-and-slab prior for correlated errors), `"IW"` (Inverse Wishart prior for correlated errors),

or "diagonal" (diagonal error covariance for independent errors). The default option is "SSSL".

aRho — A positive scalar specifying the first shape parameter of the Beta prior on the inclusion probability of edges in matrix A under the spike-and-slab prior. The default value is set to 3.

bRho — A positive scalar specifying the second shape parameter of the Beta prior on the inclusion probability of edges in matrix A under the spike-and-slab prior. The default value is set to 1.

nu1 — A positive scalar input representing the multiplication factor in the variance of the spike part in the spike and slab distribution of matrix A. The default value is set to 0.001.

aPsi — A positive scalar specifying the first shape parameter of the Beta prior on the inclusion probability of edges in matrix B under the spike-and-slab prior. The default value is set to 0.5.

bPsi — A positive scalar specifying the second shape parameter of the Beta prior on the inclusion probability of edges in matrix B under the spike-and-slab prior. The default value is set to 0.5.

nu2 — A positive scalar input corresponding to the multiplication factor in the variance of the spike part in the spike and slab distribution of matrix B. The default value is set to 0.0001.

aSigma — A positive scalar specifying the first shape parameter of the Inverse Gamma prior for the diagonal entries of the error variance–covariance matrix when SigmaStarModel = "diagonal". The default value is set to 0.01.

bSigma — A positive scalar specifying the second shape parameter of the Inverse Gamma prior for the diagonal entries of the error variance–covariance matrix when SigmaStarModel = "diagonal". The default value is set to 0.01.

PropVarA — A positive scalar input representing the variance of the normal distribution used for proposing terms within the A matrix. The default value is set to 0.01.

PropVarB — A positive scalar input representing the variance of the normal distribution used for proposing terms within the B matrix. The default value is set to 0.01.

Value

AEst — A matrix of dimensions p * p, representing the estimated causal effects or strengths between the response variables.

BEst — A matrix of dimensions p * k, representing the estimated causal effects or strengths of the instrument variables on the response variables. Each row corresponds to a specific response variable, and each column corresponds to a particular instrument variable.

CEst — A matrix of dimensions p * l, representing the estimated effects of covariates on the response variables. Each row corresponds to a response variable and each column corresponds to a covariate. This output is returned only when covariate data are provided as input (i.e., when U or Syu, Sxu and Suu are supplied).

zAEst — A binary adjacency matrix of dimensions p * p, indicating the graph structure between the response variables. Each entry in the matrix represents the presence (1) or absence (0) of a causal link between the corresponding response variables.

`zBEst` A binary adjacency matrix of dimensions p * k, illustrating the graph structure between the response variables and the instrument variables. Each row corresponds to a specific response variable, and each column corresponds to a particular instrument variable. The presence of a causal link is denoted by 1, while the absence is denoted by 0.

`Graph` A graph object that visually represents the estimated causal network by integrating AEst and zAEst. The graph consists of directed edges that indicate causal relationships between response variables. The edge weights correspond to effect sizes from AEst, while the structure of the graph is defined by zAEst, which already contain binary values (0 or 1) after thresholding at 0.5. This visualization provides an intuitive way to analyze both the strength and structure of causal effects within the network. Users can plot this graph to explore the causal relationships effectively.

`A0Est` A matrix of dimensions p * p, representing the estimated causal effects or strengths between response variables before thresholding. This output is particularly relevant for cases where the "Threshold" prior assumption is utilized.

`B0Est` A matrix of dimensions p * k, representing the estimated causal effects or strengths of the instrument variables on the response variables before thresholding. This output is particularly relevant for cases where the "Threshold" prior assumption is utilized. Each row corresponds to a specific response variable, and each column corresponds to a particular instrument variable.

`GammaEst` A matrix of dimensions p * p, representing the estimated probabilities of edges between response variables in the graph structure. Each entry in the matrix indicates the probability of a causal link between the corresponding response variables.

`TauEst` A matrix of dimensions p * p, representing the estimated variances of causal interactions between response variables. Each entry in the matrix corresponds to the variance of the causal effect between the corresponding response variables.

`PhiEst` A matrix of dimensions p * k, representing the estimated probabilities of edges between response and instrument variables in the graph structure. Each row corresponds to a specific response variable, and each column corresponds to a particular instrument variable.

`EtaEst` A matrix of dimensions p * k, representing the estimated variances of causal interactions between response and instrument variables. Each row corresponds to a specific response variable, and each column corresponds to a particular instrument variable.

`tAEst` A scalar value representing the estimated thresholding value of causal interactions between response variables. This output is relevant when using the "Threshold" prior assumption.

`tBEst` A scalar value representing the estimated thresholding value of causal interactions between response and instrument variables. This output is applicable when using the "Threshold" prior assumption.

`SigmaEst` Estimated error variance structure. If `SigmaStarModel = "diagonal"`, this is a vector of length p containing the marginal error variances of the response variables. If `SigmaStarModel` is `"IW"` or `"SSSL"`, this is a p * p error variance–covariance matrix.

`ZEst` A matrix of dimensions p * p containing posterior inclusion probabilities for the entries of `SigmaEst`. This output is returned when `SigmaStarModel = "SSSL"`.

`AccptA` The percentage of accepted entries in the A matrix, which represents the causal interactions between response variables. This metric indicates the proportion of proposed changes that were accepted during the sampling process.

`AccptB` The percentage of accepted entries in the B matrix, which represents the causal interactions between response and instrument variables. This metric indicates the proportion of proposed changes that were accepted during the sampling process.

`AccpttA` The percentage of accepted thresholding values for causal interactions between response variables when using the "Threshold" prior assumption. This metric indicates the proportion of proposed thresholding values that were accepted during the sampling process.

`AccpttB` The percentage of accepted thresholding values for causal interactions between response and instrument variables when using the "Threshold" prior assumption. This metric indicates the proportion of proposed thresholding values that were accepted during the sampling process.

`LLPst` A vector containing the posterior log-likelihoods of the model. Each element in the vector represents the log-likelihood of the model given the observed data and the estimated parameters.

`RhoEst` A matrix of dimensions p * p, representing the estimated Bernoulli success probabilities of causal interactions between response variables when using the "Spike and Slab" prior assumption. Each entry in the matrix corresponds to the success probability of a causal interaction between the corresponding response variables.

`PsiEst` A matrix of dimensions p * k, representing the estimated Bernoulli success probabilities of causal interactions between response and instrument variables when using the "Spike and Slab" prior assumption. Each row in the matrix corresponds to a specific response variable, and each column corresponds to a particular instrument variable.

`GammaPst` An array containing the posterior samples of the network structure among the response variables.

References

Ni, Y., Ji, Y., & Müller, P. (2018). Reciprocal graphical models for integrative gene regulatory network analysis. *Bayesian Analysis*, **13(4)**, 1095-1110. doi:10.1214/17BA1087.

Examples

```
# ------------------------------------------------------
#
# Example 1:
# Run RGM based on individual-level data with Threshold prior
# based on the model Y = AY + BX + CU + E
#
# Data Generation
```

```
set.seed(9154)

# Number of data points
n = 10000

# Number of response variables, instrument variables, and covariates
p = 3
k = 4
l = 2

# Initialize causal interaction matrix between response variables
A = matrix(sample(c(-0.1, 0.1), p^2, replace = TRUE), p, p)

# Diagonal entries of A matrix will always be 0
diag(A) = 0

# Make the network sparse
A[sample(which(A != 0), length(which(A != 0)) / 2)] = 0

# Create D matrix (Indicator matrix: rows = responses, cols = instruments)
D = matrix(0, nrow = p, ncol = k)

# Manually assign values to D matrix
D[1, 1:2] = 1  # First response variable is influenced by the first 2 instruments
D[2, 3] = 1    # Second response variable is influenced by the 3rd instrument
D[3, 4] = 1    # Third response variable is influenced by the 4th instrument

# Initialize B matrix (effects of instruments on responses) based on D
B = matrix(0, p, k)
for (i in 1:p) {
  for (j in 1:k) {
    if (D[i, j] == 1) {
      B[i, j] = 1
    }
  }
}

# Generate covariate data matrix U (n x l)
U = matrix(rnorm(n * l, 0, 1), nrow = n, ncol = l)

# Initialize C matrix (effects of covariates on responses), similar to B
C = matrix(0, p, l)
C[sample(length(C), size = ceiling(length(C) / 2))] = 1  # simple sparse pattern

# Define Sigma matrix (diagonal error variance)
Sigma = diag(p)

# Compute Mult_Mat
Mult_Mat = solve(diag(p) - A)

# Calculate Variance
Variance = Mult_Mat %*% Sigma %*% t(Mult_Mat)
```

```
# Generate instrument data matrix X (n x k)
X = matrix(rnorm(n * k, 0, 1), nrow = n, ncol = k)

# Initialize response data matrix Y (n x p)
Y = matrix(0, nrow = n, ncol = p)

# Generate response data matrix based on the model
for (i in 1:n) {
  mu_i = Mult_Mat %*% (B %*% X[i, ] + C %*% U[i, ])
  Y[i, ] = MASS::mvrnorm(n = 1, mu = mu_i, Sigma = Variance)
}

# Define a function to create smaller arrowheads
smaller_arrowheads = function(graph) {
  igraph::E(graph)$arrow.size = 1
  return(graph)
}

# Print true causal interaction matrices
A
B
C

# Plot the true graph structure between response variables
plot(smaller_arrowheads(igraph::graph_from_adjacency_matrix((A != 0) * 1,
  mode = "directed")), layout = igraph::layout_in_circle, main = "True Graph")

# Apply RGM on individual-level data with covariates for Threshold prior
Output = RGM(X = X, Y = Y, U = U, D = D,
             prior = "Threshold", SigmaStarModel = "diagonal")

# Get the graph structure between response variables
Output$zAEst

# Get the estimated causal strength matrix between response variables
Output$AEst

# Get the graph structure between response and instrument variables
Output$zBEst

# Get the estimated causal strength matrix between response and instrument variables
Output$BEst

# (Only when covariates are provided) Get the estimated covariate effects
Output$CEst

# Get the estimated causal network between responses
Output$Graph

# Plot the estimated causal network
plot(Output$Graph, main = "Estimated Causal Network")

# Plot posterior log-likelihood
```

```
plot(Output$LLPst, type = "l", xlab = "Number of Iterations", ylab = "Log-likelihood")

# ------------------------------------------------------------------
# Example 2:
# Run RGM based on summary-level data with Spike and Slab prior
# based on the model Y = AY + BX + CU + E

# Data Generation
set.seed(9154)

# Number of data points
n = 10000

# Number of response variables, instrument variables, and covariates
p = 3
k = 4
l = 2

# Initialize causal interaction matrix between response variables
A = matrix(sample(c(-0.1, 0.1), p^2, replace = TRUE), p, p)

# Diagonal entries of A matrix will always be 0
diag(A) = 0

# Make the network sparse
A[sample(which(A != 0), length(which(A != 0)) / 2)] = 0

# Create D matrix (Indicator matrix where each row corresponds to a response variable
# and each column corresponds to an instrument variable)
D = matrix(0, nrow = p, ncol = k)

# Manually assign values to D matrix
D[1, 1:2] = 1  # First response variable is influenced by the first 2 instruments
D[2, 3] = 1    # Second response variable is influenced by the 3rd instrument
D[3, 4] = 1    # Third response variable is influenced by the 4th instrument

# Initialize B matrix
B = matrix(0, p, k)
for (i in 1:p) {
  for (j in 1:k) {
    if (D[i, j] == 1) {
      B[i, j] = 1
    }
  }
}

# Generate covariate data matrix U (n x l)
U = matrix(rnorm(n * l, 0, 1), nrow = n, ncol = l)

# Initialize C matrix (effects of covariates on responses)
C = matrix(0, p, l)
C[sample(length(C), size = ceiling(length(C) / 2))] = 1  # simple sparse pattern
```

```
# Use a full (non-diagonal) Sigma matrix
Sigma = crossprod(matrix(rnorm(p * p), p, p))  # SPD matrix

Mult_Mat = solve(diag(p) - A)
Variance = Mult_Mat %*% Sigma %*% t(Mult_Mat)

# Generate instrument data matrix
X = matrix(rnorm(n * k, 0, 1), nrow = n, ncol = k)

# Initialize response data matrix
Y = matrix(0, nrow = n, ncol = p)

# Generate response data matrix based on instrument + covariate effects
for (i in 1:n) {
  mu_i = Mult_Mat %*% (B %*% X[i, ] + C %*% U[i, ])
  Y[i, ] = MASS::mvrnorm(n = 1, mu = mu_i, Sigma = Variance)
}

# Calculate summary-level data (including covariate summaries)
Syy = t(Y) %*% Y / n
Syx = t(Y) %*% X / n
Sxx = t(X) %*% X / n
Syu = t(Y) %*% U / n
Sxu = t(X) %*% U / n
Suu = t(U) %*% U / n

# Print true causal interaction matrices
A
B
C

# Plot the true graph structure between response variables
plot(smaller_arrowheads(igraph::graph_from_adjacency_matrix(((A != 0) * 1),
  mode = "directed")), layout = igraph::layout_in_circle, main = "True Graph")

# Apply RGM on summary-level data for Spike and Slab Prior + SSSL SigmaStarModel
Output = RGM(Syy = Syy, Syx = Syx, Sxx = Sxx,
             Syu = Syu, Sxu = Sxu, Suu = Suu,
             D = D, n = n, prior = "Spike and Slab",
             SigmaStarModel = "SSSL")

# Get the graph structure between response variables
Output$zAEst

# Get the estimated causal strength matrix between response variables
Output$AEst

# Get the graph structure between response and instrument variables
Output$zBEst

# Get the estimated causal strength matrix between response and instrument variables
Output$BEst
```

```
# Get the estimated covariate effect matrix
Output$CEst

# (Only when SigmaStarModel = "SSSL") inclusion probabilities for Sigma*
Output$ZEst

# Get the estimated causal network between responses
Output$Graph

# Plot the estimated causal network
plot(Output$Graph, main = "Estimated Causal Network")

# Plot posterior log-likelihood
plot(Output$LLPst, type = "l", xlab = "Number of Iterations", ylab = "Log-likelihood")


# ------------------------------------------------------------------
# Example 3:
# Run RGM based on Sxx, Beta and SigmaHat with Spike and Slab prior
# based on the model Y = AY + BX + E

# Data Generation
set.seed(9154)

# Number of datapoints
n = 10000

# Number of response variables and number of instrument variables
p = 3
k = 4

# Initialize causal interaction matrix between response variables
A = matrix(sample(c(-0.1, 0.1), p^2, replace = TRUE), p, p)

# Diagonal entries of A matrix will always be 0
diag(A) = 0

# Make the network sparse
A[sample(which(A != 0), length(which(A != 0)) / 2)] = 0

# Create D matrix (Indicator matrix where each row corresponds to a response variable
# and each column corresponds to an instrument variable)
D = matrix(0, nrow = p, ncol = k)

# Manually assign values to D matrix
D[1, 1:2] = 1  # First response variable is influenced by the first 2 instruments
D[2, 3] = 1    # Second response variable is influenced by the 3rd instrument
D[3, 4] = 1    # Third response variable is influenced by the 4th instrument

# Initialize B matrix
B = matrix(0, p, k)  # Initialize B matrix with zeros

# Calculate B matrix based on D matrix
```

```
for (i in 1:p) {
  for (j in 1:k) {
    if (D[i, j] == 1) {
      B[i, j] = 1  # Set B[i, j] to 1 if D[i, j] is 1
    }
  }
}

# --- Full (non-diagonal) SPD error covariance matrix Sigma ---
R = matrix(rnorm(p * p), p, p)
Sigma = crossprod(R)                # SPD
Sigma = Sigma / mean(diag(Sigma))   # scale so diagonal magnitudes are ~1

# Compute Mult_Mat
Mult_Mat = solve(diag(p) - A)

# Calculate Variance of Y given X
Variance = Mult_Mat %*% Sigma %*% t(Mult_Mat)

# Generate DNA expressions
X = matrix(rnorm(n * k, 0, 1), nrow = n, ncol = k)

# Initialize response data matrix
Y = matrix(0, nrow = n, ncol = p)

# Generate response data matrix based on instrument data matrix
for (i in 1:n) {
  Y[i, ] = MASS::mvrnorm(n = 1, Mult_Mat %*% B %*% X[i, ], Variance)
}

# Centralize Data (required for Beta/SigmaHat input option)
Y = t(t(Y) - colMeans(Y))
X = t(t(X) - colMeans(X))

# Calculate Sxx
Sxx = t(X) %*% X / n

# Generate Beta matrix and SigmaHat
Beta = matrix(0, nrow = p, ncol = k)
SigmaHat = matrix(0, nrow = p, ncol = k)

for (i in 1:p) {
  for (j in 1:k) {
    fit = lm(Y[, i] ~ X[, j])
    Beta[i, j] = fit$coefficients[2]
    SigmaHat[i, j] = sum(fit$residuals^2) / n
  }
}

# Print true causal interaction matrices between response variables
# and between response and instrument variables
A
B
```

```
# Plot the true graph structure between response variables
smaller_arrowheads = function(graph) {
  igraph::E(graph)$arrow.size = 1
  return(graph)
}

plot(smaller_arrowheads(
  igraph::graph_from_adjacency_matrix(((A != 0) * 1), mode = "directed")
), layout = igraph::layout_in_circle, main = "True Graph")

# Apply RGM based on Sxx, Beta and SigmaHat for Spike and Slab Prior
# Use IW structure for SigmaStarModel
Output = RGM(Sxx = Sxx, Beta = Beta, SigmaHat = SigmaHat,
             D = D, n = n, prior = "Spike and Slab",
             SigmaStarModel = "IW")

# Get the graph structure between response variables
Output$zAEst

# Get the estimated causal strength matrix between response variables
Output$AEst

# Get the graph structure between response and instrument variables
Output$zBEst

# Get the estimated causal strength matrix between response and instrument variables
Output$BEst

# Get the estimated causal network between responses
Output$Graph

# Plot the estimated causal network
plot(Output$Graph, main = "Estimated Causal Network")

# Plot posterior log-likelihood
plot(Output$LLPst, type = 'l', xlab = "Number of Iterations", ylab = "Log-likelihood")
```

Index

NetworkMotif, 2

RGM, 5