Package 'mvfmr'

July 14, 2026

Type Package

Title Functional Multivariable Mendelian Randomization

Version 0.2.0

Description Implements Multivariable Functional Mendelian Randomization (MV-FMR) to estimate time-varying causal effects of multiple longitudinal exposures on health outcomes. Extends univariable functional Mendelian Randomisation (MR) (Tian et al., 2024 <doi:10.1002/sim.10222>) to the multivariable setting, enabling joint estimation of multiple time-varying exposures with pleiotropy and mediation scenarios. Key features include: (1) data-driven cross-validation for basis component selection, (2) handling of mediation pathways between exposures, (3) support for both continuous and binary outcomes using Generalized Method of Moments (GMM) and control function approaches, (4) one-sample and two-sample MR designs, (5) bootstrap inference and instrument diagnostics including Q-statistics for overidentification testing. Methods are described in Fontana et al. (2025) <doi:10.48550/arXiv.2512.19064>.

License MIT + file LICENSE

Encoding UTF-8

Depends R (>= 3.5.0)

Imports fdapace, ggplot2 (>= 3.0.0), parallel, doParallel, foreach, pROC, progress, glmnet, gridExtra, stats

Suggests dplyr, tidyr, testthat (>= 3.0.0), knitr, rmarkdown

VignetteBuilder knitr

RoxygenNote 7.3.1

Config/testthat/edition 3

NeedsCompilation no

Author Nicole Fontana [aut, cre], Francesca Ieva [aut, ths], Piercesare Secchi [aut, ths]

Maintainer Nicole Fontana <nicole.fontana@polimi.it>

Repository CRAN

Date/Publication 2026-07-14 07:10:02 UTC

Contents

`mvfmr-package` *mvfmr: Multivariable Functional Mendelian Randomization*

Description

Implements Multivariable Functional Mendelian randomization to estimate time-varying causal effects of multiple correlated longitudinal exposures.

Author(s)

Nicole Fontana

`AUTOMATIC_Multi_FMVMR_twosample_simple`
Two-sample joint multivariable FMR (internal)

Description

Two-sample joint multivariable FMR (internal)

Usage

```
AUTOMATIC_Multi_FMVMR_twosample_simple(
  Gmatrix,
  res_list,
  by_used,
  sy_used,
  ny_used,
  max_nPC = NA,
  XYmodels = NA,
  basis = "eigenfunction"
)
```

Arguments

Gmatrix	Genetic instrument matrix from the exposure sample (N × J)
res_list	List of length m of FPCA results, one per exposure
by_used	Vector of SNP-outcome effect estimates (betas) from the outcome GWAS, length J
sy_used	Vector of standard errors for SNP-outcome effects, length J
ny_used	Sample size of the outcome GWAS
max_nPC	Length-m vector: maximum number of principal components to retain per exposure (NA = select automatically)
XYmodels	Length-m vector: true effect model for each exposure on Y (for simulation only)
basis	Basis type for functional representation: "eigenfunction" or "polynomial"

Value

List with separate estimation results for each of the m exposures

---

cf_logit *Control function for logit model*

Description

Control function for logit model

Usage

```
cf_logit(
  X,
  Y,
  Z,
  alpha = 1,
  nfolds = 10,
  standardize = TRUE,
  use_lasso = FALSE
)
```

Arguments

X	Matrix of exposure principal components (N x K)
Y	Binary outcome vector (0/1, length N)
Z	Genetic instrument matrix (N x J)
alpha	Elastic net mixing parameter (1=lasso, 0=ridge)
nfolds	Number of cross-validation folds for lambda selection
standardize	Standardize variables before fitting
use_lasso	Use LASSO regularization in first stage. If FALSE, uses OLS.

Value

List with gmm_est, gmm_se, variance_matrix, gmm_pval

Examples

```
set.seed(1)
n <- 200; J <- 5; K <- 2
Z <- matrix(rbinom(n * J, 2, 0.3), n, J)
X <- Z[, 1:K] + matrix(rnorm(n * K, sd = 0.5), n, K)
lin_pred <- X %*% c(0.8, -0.4)
Y <- rbinom(n, 1, plogis(lin_pred))
fit <- cf_logit(X, Y, Z)
fit$gmm_est
```

---

fmvmr_separate_twosample

Two-Sample Separate Univariable Functional MR

Description

Separate estimation for each exposure using outcome GWAS summary statistics.

Usage

```
fmvmr_separate_twosample(
  G_list,
  fpca_results,
  by_outcome_list,
  sy_outcome_list,
  ny_outcome,
  max_nPC = NA,
  true_effects = NULL,
  verbose = TRUE
)
```

Arguments

G_list	List of length m of genetic instrument matrices, one per exposure (N x J_k)
fpca_results	List of length m of FPCA objects, same length as G_list
by_outcome_list	List of length m of SNP-outcome beta vectors, one per exposure
sy_outcome_list	List of length m of SNP-outcome standard error vectors, one per exposure
ny_outcome	Outcome GWAS sample size
max_nPC	Maximum number of principal components to retain per exposure (length 1 or m; NA = automatically determined)
true_effects	Length-m vector of true effect model codes, one per exposure (simulation only)
verbose	Print progress messages and diagnostics during computation

Value

fmvmr_separate_twosample object

Examples

```
set.seed(1)
sim_data <- getX_multi_exposure(N = 60, J = 8, nSparse = 5, n_exposures = 2)
outcome_data <- getY_multi_exposure(sim_data, XYmodels = c("2", "8"))
fpca_results <- lapply(sim_data$exposures, function(exp_k) {
  fdapace::FPCA(exp_k$Ly_sim, exp_k$Lt_sim,
                list(dataType = "Sparse", error = TRUE, verbose = FALSE))
})
# Simulate outcome GWAS summary statistics for the two-sample design
by_outcome <- sapply(1:8, function(j) {
  coef(lm(outcome_data$Y ~ sim_data$details$G[, j]))[2]
})
sy_outcome <- sapply(1:8, function(j) {
  summary(lm(outcome_data$Y ~ sim_data$details$G[, j]))$coefficients[2, 2]
})
result <- fmvmr_separate_twosample(
  G_list = list(sim_data$details$G, sim_data$details$G),
  fpca_results = fpca_results,
  by_outcome_list = list(by_outcome, by_outcome),
  sy_outcome_list = list(sy_outcome, sy_outcome),
  ny_outcome = 60,
  max_nPC = c(2, 2),
  verbose = FALSE
)
result$exposures[[1]]$coefficients
```

`fmvmr_twosample` *Two-Sample Joint Multivariable Functional MR*

Description

Joint estimation using outcome GWAS summary statistics. Simplified approach: only needs by, sy, ny (not individual outcome data).

Usage

```
fmvmr_twosample(
  G_exposure,
  fpca_results,
  by_outcome,
  sy_outcome,
  ny_outcome,
  max_nPC = NA,
  true_effects = NULL,
  verbose = TRUE
)
```

Arguments

`G_exposure`	Genetic instrument matrix from the exposure sample (N × J)
`fpca_results`	List of length m of FPCA objects, one per exposure
`by_outcome`	Vector of SNP-outcome effect estimates (betas) from the outcome GWAS, length J
`sy_outcome`	VVector of standard errors for SNP-outcome effects, length J
`ny_outcome`	Sample size of the outcome GWAS
`max_nPC`	Maximum number of principal components to retain per exposure (length 1 or m; NA = automatically determined)
`true_effects`	Length-m vector of true effect model codes, one per exposure (simulation only)
`verbose`	Print progress messages and diagnostics during computation

Value

fmvmr_twosample object

Examples

```
set.seed(1)
sim_data <- getX_multi_exposure(N = 60, J = 8, nSparse = 5, n_exposures = 2)
outcome_data <- getY_multi_exposure(sim_data, XYmodels = c("2", "8"))
fpca_results <- lapply(sim_data$exposures, function(exp_k) {
  fdapace::FPCA(exp_k$Ly_sim, exp_k$Lt_sim,
                list(dataType = "Sparse", error = TRUE, verbose = FALSE))
```

```
})
# Simulate outcome GWAS summary statistics for the two-sample design
by_outcome <- sapply(1:8, function(j) {
  coef(lm(outcome_data$Y ~ sim_data$details$G[, j]))[2]
})
sy_outcome <- sapply(1:8, function(j) {
  summary(lm(outcome_data$Y ~ sim_data$details$G[, j]))$coefficients[2, 2]
})
result <- fmvmr_twosample(
  G_exposure = sim_data$details$G,
  fpca_results = fpca_results,
  by_outcome = by_outcome,
  sy_outcome = sy_outcome,
  ny_outcome = 60,
  max_nPC = c(2, 2),
  verbose = FALSE
)
coef(result)
```

`getX_multi_exposure` *Generate multi-exposure data with genetic instruments*

Description

Generate multi-exposure data with genetic instruments

Usage

```
getX_multi_exposure(
  N = 10000,
  J = 30,
  ZXmodel = "A",
  nSparse = 10,
  NT = 1000,
  TT = 50,
  n_exposures = 2,
  shared_effect = TRUE,
  separate_G = FALSE,
  shared_G_proportion = 0.15
)
```

Arguments

`N`	Sample size
`J`	Number of genetic instruments (per exposure, if separate_G = TRUE)
`ZXmodel`	Model type (currently not used)
`nSparse`	Number of sparse observations per subject

NT	Number of points
TT	Max observation period
n_exposures	Number of exposures to simulate (m)
shared_effect	Whether all exposures share the same time-varying confounding
separate_G	Whether to use separate instruments for each exposure
shared_G_proportion	Proportion of shared instruments (0-1)

Value

List with per-exposure sparse data and genetic instruments

Examples

```
set.seed(1)
sim_data <- getX_multi_exposure(N = 50, J = 8, nSparse = 5, n_exposures = 2)
length(sim_data$exposures)
dim(sim_data$details$G)
```

---

getX_multi_exposure_mediation

Generate multi-exposure mediation data with genetic instruments

---

Description

Generate multi-exposure mediation data with genetic instruments

Usage

```
getX_multi_exposure_mediation(
  N = 10000,
  J = 30,
  ZXmodel = "A",
  nSparse = 10,
  n_exposures = 2,
  mediation_strength = NULL,
  separate_G = FALSE,
  shared_G_proportion = 0,
  mediation_type = "linear"
)
```

Arguments

N	Sample size
J	Number of genetic instruments per exposure
ZXmodel	Model type (currently not used, kept for compatibility)
nSparse	Number of sparse observations per subject
n_exposures	Number of exposures to simulate (m)
mediation_strength	m x m numeric matrix of pairwise mediation strengths: entry [j, k] (with j < k) is the strength with which exposure j mediates its effect onto exposure k, generated later in the sequence. Must be strictly upper triangular (entries with j >= k must be 0). Default: NULL, i.e. no mediation (all-zero matrix).
separate_G	Whether to use separate instruments for each exposure
shared_G_proportion	Proportion of shared instruments (0-1)
mediation_type	Character, or m x m character matrix mirroring mediation_strength: type of mediation effect for each pair, one of "linear" (default), "nonlinear", or "time_varying".

Value

List with same structure as getX_multi_exposure()

Examples

```
set.seed(1)
# Exposure 1 mediates onto exposure 2 with strength 0.3
mediation_strength <- matrix(c(0, 0, 0.3, 0), 2, 2)
sim_data <- getX_multi_exposure_mediation(
  N = 50, J = 8, nSparse = 5, n_exposures = 2,
  mediation_strength = mediation_strength
)
length(sim_data$exposures)
```

---

getY_multi_exposure *Generate outcome from exposures*

Description

Generate outcome from exposures

Usage

```
getY_multi_exposure(
  RES,
  XYmodels = NULL,
  X_effects = NULL,
  outcome_type = "continuous"
)
```

Arguments

RES	Output from getX_multi_exposure() or getX_multi_exposure_mediation()
XYmodels	Length-m vector of effect models per exposure, one of '0'-'9' (default: '1' for all)
X_effects	Length-m logical vector: include each exposure's effect? (default: TRUE for all)
outcome_type	"continuous" or "binary"

Value

Data frame with outcome Y

Examples

```
set.seed(1)
sim_data <- getX_multi_exposure(N = 50, J = 8, nSparse = 5, n_exposures = 2)
dat <- getY_multi_exposure(sim_data, XYmodels = c("2", "8"), outcome_type = "continuous")
head(dat$Y)
```

---

gmm_lm_onesample *GMM estimation for continuous outcome*

Description

GMM estimation for continuous outcome

Usage

```
gmm_lm_onesample(X, Y, Z, beta0 = NA)
```

Arguments

X	Matrix of exposure principal components (N x K)
Y	Outcome vector (length N)
Z	Genetic instrument matrix (N x J)
beta0	Initial values for beta (default NA, uses zero initialization)

Value

List with gmm_est, gmm_se, variance_matrix, gmm_pval, Q_stat, Q_pval

Examples

```
set.seed(1)
n <- 200; J <- 5; K <- 2
Z <- matrix(rbinom(n * J, 2, 0.3), n, J)
X <- Z[, 1:K] + matrix(rnorm(n * K, sd = 0.5), n, K)
Y <- as.numeric(X %*% c(1, -0.5) + rnorm(n))
fit <- gmm_lm_onesample(X, Y, Z)
fit$gmm_est
```

---

gmm_twosample_simple *Two-sample GMM*

Description

Two-sample GMM

Usage

```
gmm_twosample_simple(bx, by, sy, ny)
```

Arguments

bx	Matrix J x K of first-stage coefficients (SNP -> PC associations)
by	Vector length J of outcome GWAS betas
sy	Vector length J of outcome GWAS standard errors
ny	Outcome GWAS sample size

Value

List with gmm_est, gmm_se, variance_matrix, gmm_pval, Q_stat, Q_df, Q_pval

Examples

```
set.seed(1)
J <- 10; K <- 2
bx <- matrix(rnorm(J * K, sd = 0.3), J, K)
by <- bx %*% c(0.5, -0.2) + rnorm(J, sd = 0.05)
sy <- runif(J, 0.02, 0.05)
fit <- gmm_twosample_simple(bx, by, sy, ny = 50000)
fit$gmm_est
```

IS *Calculate F-statistics and Q-statistic for instrument strength (internal)*

Description

Calculate F-statistics and Q-statistic for instrument strength (internal)

Usage

```
IS(J, K, PC, datafull, Y = NULL)
```

Arguments

J	Number of genetic instruments
K	Number of exposures
PC	Vector of indices indicating which columns in datafull correspond to the principal components
datafull	Data frame containing instruments (first J columns) and principal components (subsequent columns) [G, X]
Y	Optional outcome vector; if provided, Q-statistic for overidentification is calculated)

Value

Matrix with columns: PC (component index), RR (R-squared), FF (F-statistic), cFF (conditional F-statistic). If Y is provided, additional columns: Qvalue (Hansen's J overidentification test statistic), df (degrees of freedom for Q-test), pvalue (p-value for Q-test from chi-squared distribution).

Examples

```
set.seed(1)
n <- 200; J <- 5; K <- 2
G <- matrix(rbinom(n * J, 2, 0.3), n, J)
PCmat <- G[, 1:K] + matrix(rnorm(n * K, sd = 0.5), n, K)
Y <- as.numeric(PCmat %*% c(1, -0.5) + rnorm(n))
fstats <- IS(J = J, K = K, PC = 1:K, datafull = cbind(G, PCmat), Y = Y)
fstats
```

mvfmr *Joint Multivariable Functional Mendelian Randomization*

Description

Joint Multivariable Functional Mendelian Randomization

Usage

```
mvfmr(
  G,
  fpca_results,
  Y,
  outcome_type = c("continuous", "binary"),
  method = c("gmm", "cf", "cf-lasso"),
  nPC = NA,
  max_nPC = NA,
  improvement_threshold = 0.001,
  bootstrap = FALSE,
  n_bootstrap = 100,
  n_cores = parallel::detectCores() - 1,
  true_effects = NULL,
  X_true = NULL,
  verbose = FALSE
)
```

Arguments

G	Genetic instrument matrix (N x J)
fpca_results	List of length m of FPCA objects from fdapace, one per exposure
Y	Outcome vector
outcome_type	Type of outcome: "continuous" for numeric outcomes, "binary" for 0/1 outcomes
method	Estimation method: "gmm" (Generalized Method of Moments), "cf" (control function), or "cf-lasso" (control function with Lasso)
nPC	Fixed number of principal components to retain per exposure (length 1 or m; NA = select automatically)
max_nPC	Maximum number of principal components to retain per exposure (length 1 or m; NA = automatically determined)
improvement_threshold	Minimum cross-validation improvement required to add an additional principal component
bootstrap	Whether to compute confidence intervals using bootstrap resampling
n_bootstrap	Number of bootstrap replicates (only used if bootstrap = TRUE)

n_cores	Number of CPU cores to use for parallel computations
true_effects	Length-m vector of true effect model codes, one per exposure (simulation only)
X_true	Length-m list of true X curves, one per exposure (simulation only)
verbose	Print progress and diagnostic messages during computation

Value

mvfmr object with:

- coefficients - Estimated beta coefficients
- vcov - Variance-covariance matrix
- effects - List of length m with the estimated time-varying effect curves
- nPC_used - Components selected per exposure
- diagnostics - F-statistics, instrument diagnostics
- performance - MISE, coverage (if true effects provided)

Examples

```
set.seed(1)
sim_data <- getX_multi_exposure(N = 60, J = 8, nSparse = 5, n_exposures = 2)
outcome_data <- getY_multi_exposure(sim_data, XYmodels = c("2", "8"))
fpca_results <- lapply(sim_data$exposures, function(exp_k) {
  fdapace::FPCA(exp_k$Ly_sim, exp_k$Lt_sim,
                list(dataType = "Sparse", error = TRUE, verbose = FALSE))
})
result <- mvfmr(
  G = sim_data$details$G,
  fpca_results = fpca_results,
  Y = outcome_data$Y,
  max_nPC = c(2, 2),
  n_cores = 1,
  verbose = FALSE
)
coef(result)
```

---

mvfmr_separate *Separate Univariable Functional Mendelian Randomization*

Description

Separate Univariable Functional Mendelian Randomization

Usage

```
mvfmr_separate(
  G_list,
  fpca_results,
  Y,
  outcome_type = c("continuous", "binary"),
  method = c("gmm", "cf", "cf-lasso"),
  nPC = NA,
  max_nPC = NA,
  improvement_threshold = 0.001,
  bootstrap = FALSE,
  n_bootstrap = 100,
  n_cores = parallel::detectCores() - 1,
  true_effects = NULL,
  X_true = NULL,
  verbose = FALSE
)
```

Arguments

G_list	List of length m of genetic instrument matrices, one per exposure. Use a list of length 1 to analyze a single exposure.
fpca_results	List of length m of FPCA objects, same length as G_list
Y	Outcome vector
outcome_type	Type of outcome: "continuous" for numeric outcomes, "binary" for 0/1 outcomes
method	Estimation method: "gmm" (Generalized Method of Moments), "cf" (control function), or "cf-lasso" (control function with Lasso)
nPC	Fixed number of principal components to retain per exposure (length 1 or m; NA = select automatically)
max_nPC	Maximum number of principal components to retain per exposure (length 1 or m; NA = automatically determined)
improvement_threshold	Minimum cross-validation improvement required to add an additional principal component
bootstrap	Whether to compute confidence intervals using bootstrap resampling
n_bootstrap	Number of bootstrap replicates (only used if bootstrap = TRUE)
n_cores	Number of CPU cores to use for parallel computations
true_effects	Length-m vector of true effect model codes, one per exposure (simulation only)
X_true	Length-m list of true X curves, one per exposure (simulation only)
verbose	Print progress and diagnostic messages during computation

Value

mvfmr_separate object

Examples

```
set.seed(1)
sim_data <- getX_multi_exposure(N = 60, J = 8, nSparse = 5, n_exposures = 2)
outcome_data <- getY_multi_exposure(sim_data, XYmodels = c("2", "8"))
fpca_results <- lapply(sim_data$exposures, function(exp_k) {
  fdapace::FPCA(exp_k$Ly_sim, exp_k$Lt_sim,
                list(dataType = "Sparse", error = TRUE, verbose = FALSE))
})
result <- mvfmr_separate(
  G_list = list(sim_data$details$G, sim_data$details$G),
  fpca_results = fpca_results,
  Y = outcome_data$Y,
  max_nPC = c(2, 2),
  n_cores = 1,
  verbose = FALSE
)
coef(result, exposure = 1)
```

Separate_Multi_FMVMR_twosample_simple

Separate univariable two-sample FMR (internal)

Description

Separate univariable two-sample FMR (internal)

Usage

```
Separate_Multi_FMVMR_twosample_simple(
  Gmatrix_list,
  res_list,
  by_used_list,
  sy_used_list,
  ny_used,
  max_nPC = NA,
  XYmodels = NA,
  basis = "eigenfunction"
)
```

Arguments

Gmatrix_list	List of length m of genetic instrument matrices, one per exposure (N x J_k)
res_list	List of length m of FPCA results, one per exposure
by_used_list	List of length m of SNP-outcome effect estimate vectors, one per exposure
sy_used_list	List of length m of SNP-outcome standard error vectors, one per exposure
ny_used	Sample size of the outcome GWAS

`max_nPC`	Length-m vector: maximum number of principal components to retain per exposure (NA = select automatically)
`XYmodels`	Length-m vector: true effect model for each exposure on Y (for simulation only)
`basis`	Basis type for functional representation: "eigenfunction" or "polynomial"

Value

List with separate estimation results for each of the m exposures

Index