

Package ‘missoNet’

May 8, 2026

Type Package

Title Joint Sparse Regression & Network Learning with Missing Data

Version 1.5.1

Date 2025-09-01

Maintainer Yixiao Zeng <yixiao.zeng@mail.mcgill.ca>

Description Simultaneously estimates sparse regression coefficients and response network structure in multivariate models with missing data. Unlike traditional approaches requiring imputation, handles missingness natively through unbiased estimating equations (MCAR/MAR compatible). Employs dual L1 regularization with automated selection via cross-validation or information criteria. Includes parallel computation, warm starts, adaptive grids, publication-ready visualizations, and prediction methods. Ideal for genomics, neuroimaging, and multi-trait studies with incomplete high-dimensional outcomes. See Zeng et al. (2025) <[doi:10.48550/arXiv.2507.05990](https://doi.org/10.48550/arXiv.2507.05990)>.

License GPL-2

URL <https://github.com/yixiao-zeng/missoNet>,
<https://arxiv.org/abs/2507.05990>

BugReports <https://github.com/yixiao-zeng/missoNet/issues>

Depends R (>= 3.6.0)

Imports circlize (>= 0.4.15), ComplexHeatmap, glassoFast (>= 1.0.1), graphics, grid, mvtnorm (>= 1.2.3), pbapply (>= 1.7.2), Rcpp (>= 1.0.9), scatterplot3d (>= 0.3.44), stats, utils

Suggests ggplot2, glasso, gridExtra, igraph, knitr, parallel, RColorBrewer, reshape2, rmarkdown

LinkingTo Rcpp, RcppArmadillo

VignetteBuilder knitr

ByteCompile true

Encoding UTF-8

NeedsCompilation yes

RoxygenNote 7.3.2

Author Yixiao Zeng [aut, cre, cph],
Celia Greenwood [ths, aut]

Repository CRAN

Date/Publication 2025-09-02 20:50:07 UTC

Contents

| | |
|----------------------------|----|
| missoNet-package | 2 |
| cv.missoNet | 4 |
| generateData | 8 |
| missoNet | 11 |
| plot.missoNet | 15 |
| predict.missoNet | 17 |

| | |
|--------------|-----------|
| Index | 20 |
|--------------|-----------|

| | |
|------------------|--|
| missoNet-package | <i>missoNet: Multi-Task Regression and Conditional Network Estimation with Missing Responses</i> |
|------------------|--|

Description

missoNet fits a joint multivariate regression and conditional dependency (precision–matrix) model when some response entries are missing. The method estimates a sparse coefficient matrix B linking predictors X to multivariate responses Y , together with a sparse inverse covariance Θ for the residuals in $Y = \mathbf{1}\mu^T + XB + E$, $E \sim \mathcal{N}(0, \Theta^{-1})$. Responses may contain missing values (e.g., MCAR/MAR); predictors must be finite. The package provides cross-validation, prediction, publication-ready plotting, and simple simulation utilities.

Details**Key features**

- Joint estimation of B (regression) and Θ (conditional network).
- ℓ_1 -regularization on both B and Θ with user-controlled grids.
- K-fold cross-validation with optional 1-SE model selections.
- Heatmap and 3D surface visualizations for CV error or GoF across $(\lambda_B, \lambda_\Theta)$.
- Fast prediction for new data using stored solutions.
- Lightweight data generator for simulation studies.

Workflow

1. Fit a model across a grid of penalties with `missoNet` or select penalties via `cv.missoNet`.
2. Visualize the CV error/GoF surface with `plot.missoNet`.
3. Predict responses for new observations with `predict.missoNet`.

Main functions

`missoNet` Fit models over user-specified penalty grids for λ_B and λ_Θ ; returns estimated μ , B , Θ , and metadata (grids, GoF).

`cv.missoNet` Perform k-fold cross-validation over a penalty grid; stores `est.min` and (optionally) `est.1se.beta`, `est.1se.theta`.

`plot.missoNet` S3 plotting method; heatmap or 3D scatter of CV error or GoF.

`predict.missoNet` S3 prediction method; returns $\hat{Y} = \mathbf{1}\hat{\mu}^\top + X_{\text{new}}\hat{B}$ for a chosen solution.

`generateData` Generate synthetic datasets with controllable dimensions, signal, and missingness mechanisms for benchmarking.

License

GPL-2.

Author(s)

Maintainer: Yixiao Zeng <yixiao.zeng@mail.mcgill.ca> [copyright holder]

Authors:

- Celia Greenwood <celia.greenwood@mcgill.ca> [thesis advisor]

See Also

`missoNet`, `cv.missoNet`, `plot.missoNet`, `predict.missoNet`, `generateData`, and `browseVignettes("missoNet")` for tutorials.

Examples

```
sim <- generateData(n = 100, p = 8, q = 5, rho = 0.1, missing.type = "MCAR")
```

```
fit <- missoNet(X = sim$X, Y = sim$Z)           # fit over a grid
plot(fit)                                     # GoF heatmap
```

```
cvfit <- cv.missoNet(X = sim$X, Y = sim$Z, kfold = 5, compute.1se = TRUE)
plot(cvfit, type = "scatter", plt.surf = TRUE) # CV error surface
yhat <- predict(cvfit, newx = sim$X, s = "lambda.min")
```

`cv.missoNet`*Cross-validation for missoNet*

Description

Perform k -fold cross-validation to select the regularization pair (`lambda.beta`, `lambda.theta`) for `missoNet`. For each fold the model is trained on $k - 1$ partitions and evaluated on the held-out partition over a grid of lambda pairs; the pair with minimum mean CV error is returned, with optional 1-SE models for more regularized solutions.

Usage

```
cv.missoNet(  
  X,  
  Y,  
  kfold = 5,  
  rho = NULL,  
  lambda.beta = NULL,  
  lambda.theta = NULL,  
  lambda.beta.min.ratio = NULL,  
  lambda.theta.min.ratio = NULL,  
  n.lambda.beta = NULL,  
  n.lambda.theta = NULL,  
  beta.pen.factor = NULL,  
  theta.pen.factor = NULL,  
  penalize.diagonal = NULL,  
  beta.max.iter = 10000,  
  beta.tol = 1e-05,  
  theta.max.iter = 10000,  
  theta.tol = 1e-05,  
  eta = 0.8,  
  eps = 1e-08,  
  standardize = TRUE,  
  standardize.response = TRUE,  
  compute.1se = TRUE,  
  relax.net = FALSE,  
  adaptive.search = FALSE,  
  shuffle = TRUE,  
  seed = NULL,  
  parallel = FALSE,  
  cl = NULL,  
  verbose = 1  
)
```

Arguments

`X` Numeric matrix ($n \times p$). Predictors (no missing values).

| | |
|--|---|
| <code>Y</code> | Numeric matrix ($n \times q$). Responses. Missing values should be coded as NA/NaN. |
| <code>kfold</code> | Integer ≥ 2 . Number of folds (default 5). |
| <code>rho</code> | Optional numeric vector of length q . Working missingness probabilities (per response). If NULL (default), estimated from Y . |
| <code>lambda.beta, lambda.theta</code> | Optional numeric vectors. Candidate regularization paths for \mathbf{B} and Θ . If NULL, sequences are generated automatically from the data. Avoid supplying a single value because warm starts along a path are used. |
| <code>lambda.beta.min.ratio, lambda.theta.min.ratio</code> | Optional numerics in $(0, 1]$. Ratio of the smallest to the largest value when generating lambda sequences (ignored if the corresponding <code>lambda.*</code> is supplied). |
| <code>n.lambda.beta, n.lambda.theta</code> | Optional integers. Lengths of the automatically generated lambda paths (ignored if the corresponding <code>lambda.*</code> is supplied). |
| <code>beta.pen.factor</code> | Optional $p \times q$ non-negative matrix of element-wise penalty multipliers for \mathbf{B} . Inf = maximum penalty; 0 = no penalty for the corresponding coefficient. Default: all 1s (equal penalty). |
| <code>theta.pen.factor</code> | Optional $q \times q$ non-negative matrix of element-wise penalty multipliers for Θ . Off-diagonal entries control edge penalties; diagonal treatment is governed by <code>penalize.diagonal</code> . Inf = maximum penalty; 0 = no penalty for that element. Default: all 1s (equal penalty). |
| <code>penalize.diagonal</code> | Logical or NULL. Whether to penalize diagonal entries of Θ . If NULL (default) the choice is made automatically. |
| <code>beta.max.iter, theta.max.iter</code> | Integers. Max iterations for the \mathbf{B} update (FISTA) and Θ update (graphical lasso). Defaults: 10000. |
| <code>beta.tol, theta.tol</code> | Numerics > 0 . Convergence tolerances for the \mathbf{B} and Θ updates. Defaults: 1e-5. |
| <code>eta</code> | Numeric in $(0, 1)$. Backtracking line-search parameter for the \mathbf{B} update (default 0.8). |
| <code>eps</code> | Numeric in $(0, 1)$. Eigenvalue floor used to stabilize positive definiteness operations (default 1e-8). |
| <code>standardize</code> | Logical. Standardize columns of X internally? Default TRUE. |
| <code>standardize.response</code> | Logical. Standardize columns of Y internally? Default TRUE. |
| <code>compute.1se</code> | Logical. Also compute 1-SE solutions? Default TRUE. |
| <code>relax.net</code> | (Experimental) Logical. If TRUE, refit active edges of Θ without ℓ_1 penalty (de-biased network). Default FALSE. |
| <code>adaptive.search</code> | (Experimental) Logical. Use adaptive two-stage lambda search? Default FALSE. |

| | |
|----------|--|
| shuffle | Logical. Randomly shuffle fold assignments? Default TRUE. |
| seed | Optional integer seed (used when shuffle = TRUE). |
| parallel | Logical. Evaluate folds in parallel using a provided cluster? Default FALSE. |
| cl | Optional cluster from parallel::makeCluster() (required if parallel = TRUE). |
| verbose | Integer in 0, 1, 2. 0 = silent, 1 = progress (default), 2 = detailed tracing (not supported in parallel mode). |

Details

Internally, predictors X and responses Y can be standardized for optimization; all reported estimates are re-scaled back to the original data scale. Missingness in Y is handled via unbiased estimating equations using column-wise observation probabilities estimated from Y (or supplied via ρ). This is appropriate when the missingness of each response is independent of its unobserved value (e.g., MCAR).

If `adaptive.search = TRUE`, a fast two-stage pre-optimization narrows the lambda grid before computing fold errors on a focused neighborhood; this can be substantially faster on large grids but may occasionally miss the global optimum.

When `compute.1se = TRUE`, two additional solutions are reported: the largest `lambda.beta` and the largest `lambda.theta` whose CV error is within one standard error of the minimum (holding the other lambda fixed at its optimal value). At the end, three special lambda pairs are identified:

- **lambda.min**: Parameters giving minimum CV error
- **lambda.1se.beta**: Largest λ_B within 1 SE of minimum (with λ_Θ fixed at optimum)
- **lambda.1se.theta**: Largest λ_Θ within 1 SE of minimum (with λ_B fixed at optimum)

The 1SE rules provide more regularized models that may generalize better.

Value

A list of class "missoNet" with components:

est.min List of estimates at the CV minimum: Beta ($p \times q$), Theta ($q \times q$), intercept μ (length q), `lambda.beta`, `lambda.theta`, `lambda.beta.idx`, `lambda.theta.idx`, and (if requested) `relax.net`.

est.1se.beta List of estimates at the 1-SE `lambda.beta` (if `compute.1se = TRUE`); NULL otherwise.

est.1se.theta List of estimates at the 1-SE `lambda.theta` (if `compute.1se = TRUE`); NULL otherwise.

rho Length- q vector of working missingness probabilities.

kfold Number of folds used.

fold.index Integer vector of length n giving fold assignments (names are "fold-k").

lambda.beta.seq, lambda.theta.seq Unique lambda values explored along the grid for \mathbf{B} and Θ .

penalize.diagonal Logical indicating whether the diagonal of Θ was penalized.

beta.pen.factor, theta.pen.factor Penalty factor matrices actually used.

param_set List with CV diagnostics: n , p , q , `standardize`, `standardize.response`, mean errors `cv.errors.mean`, bounds `cv.errors.upper/lower`, and the evaluated grids `cv.grid.beta`, `cv.grid.theta` (length equals number of fitted models).

Author(s)

Yixiao Zeng <yixiao.zeng@mail.mcgill.ca>, Celia M. T. Greenwood

References

Zeng, Y., et al. (2025). *Multivariate regression with missing response data for modelling regional DNA methylation QTLs*. arXiv:2507.05990.

See Also

[missoNet](#) for model fitting; generic methods such as `plot()` and `predict()` for objects of class "missoNet".

Examples

```
sim <- generateData(n = 120, p = 12, q = 6, rho = 0.1)
X <- sim$X; Y <- sim$Z

# Basic 5-fold cross-validation
cvfit <- cv.missoNet(X = X, Y = Y, kfold = 5, verbose = 0)

# Extract optimal estimates
Beta.min <- cvfit$est.min$Beta
Theta.min <- cvfit$est.min$Theta

# Extract 1SE estimates (if computed)
if (!is.null(cvfit$est.1se.beta)) {
  Beta.1se <- cvfit$est.1se.beta$Beta
}
if (!is.null(cvfit$est.1se.theta)) {
  Theta.1se <- cvfit$est.1se.theta$Theta
}

# Make predictions
newX <- matrix(rnorm(10 * 12), 10, 12)
pred.min <- predict(cvfit, newx = newX, s = "lambda.min")
pred.1se <- predict(cvfit, newx = newX, s = "lambda.1se.beta")

# Parallel cross-validation
library(parallel)
cl <- makeCluster(min(detectCores() - 1, 2))
cvfit2 <- cv.missoNet(X = X, Y = Y, kfold = 5,
  parallel = TRUE, cl = cl)
stopCluster(cl)

# Adaptive search for efficiency
cvfit3 <- cv.missoNet(X = X, Y = Y, kfold = 5,
  adaptive.search = TRUE)

# Reproducible CV with specific lambdas
cvfit4 <- cv.missoNet(X = X, Y = Y, kfold = 5,
```

```

lambda.beta = 10^seq(0, -2, length = 20),
lambda.theta = 10^seq(0, -2, length = 20),
seed = 486)

# Plot CV results
plot(cvfit, type = "heatmap")
plot(cvfit, type = "scatter")

```

generateData

Generate synthetic data with missing values for missoNet

Description

Generates synthetic data from a conditional Gaussian graphical model with user-specified missing data mechanisms. This function is designed for simulation studies and testing of the missoNet package, supporting three types of missingness: Missing Completely At Random (MCAR), Missing At Random (MAR), and Missing Not At Random (MNAR).

Usage

```

generateData(
  n,
  p,
  q,
  rho,
  missing.type = "MCAR",
  X = NULL,
  Beta = NULL,
  E = NULL,
  Theta = NULL,
  Sigma.X = NULL,
  Beta.row.sparsity = 0.2,
  Beta.elm.sparsity = 0.2,
  seed = NULL
)

```

Arguments

| | |
|--------------|--|
| n | Integer. Sample size (number of observations). Must be at least 2. |
| p | Integer. Number of predictor variables. Must be at least 1. |
| q | Integer. Number of response variables. Must be at least 2. |
| rho | Numeric scalar or vector of length q. Proportion of missing values for each response variable. Values must be in [0, 1). If scalar, the same missing rate is applied to all responses. |
| missing.type | Character string specifying the missing data mechanism. One of: |

| | |
|-------------------|---|
| | <ul style="list-style-type: none"> • "MCAR" (default): Missing Completely At Random • "MAR": Missing At Random (depends on predictors) • "MNAR": Missing Not At Random (depends on response values) |
| X | Optional $n \times p$ matrix. User-supplied predictor matrix. If NULL (default), predictors are simulated from a multivariate normal distribution with mean zero and covariance $\Sigma.X$. |
| Beta | Optional $p \times q$ matrix. Regression coefficient matrix. If NULL (default), a sparse coefficient matrix is generated with sparsity controlled by <code>Beta.row.sparsity</code> and <code>Beta.elm.sparsity</code> . |
| E | Optional $n \times q$ matrix. Error/noise matrix. If NULL (default), errors are simulated from a multivariate normal distribution with mean zero and precision matrix Θ . |
| Theta | Optional $q \times q$ positive definite matrix. Precision matrix (inverse covariance) for the response variables. If NULL (default), a block-structured precision matrix is generated with four types of graph structures. Only used when $E = \text{NULL}$. |
| Sigma.X | Optional $p \times p$ positive definite matrix. Covariance matrix for the predictors. If NULL (default), an AR(1) covariance structure with correlation 0.7 is used. Only used when $X = \text{NULL}$. |
| Beta.row.sparsity | Numeric in $[0, 1]$. Proportion of rows in Beta that contain at least one non-zero element. Default is 0.2. Only used when $\text{Beta} = \text{NULL}$. |
| Beta.elm.sparsity | Numeric in $[0, 1]$. Proportion of non-zero elements within active rows of Beta. Default is 0.2. Only used when $\text{Beta} = \text{NULL}$. |
| seed | Optional integer. Random seed for reproducibility. |

Details

The function generates data through the following model:

$$Y = XB + E$$

where:

- $X \in \mathbb{R}^{n \times p}$ is the predictor matrix
- $B \in \mathbb{R}^{p \times q}$ is the coefficient matrix
- $E \sim \mathcal{MVN}(0, \Theta^{-1})$ is the error matrix
- $Y \in \mathbb{R}^{n \times q}$ is the complete response matrix

Missing values are then introduced to create Z (the observed response matrix with NAs) according to the specified mechanism:

MCAR: Each element has probability `rho[j]` of being missing, independent of all variables.

MAR: Missingness depends on the predictors through a logistic model:

$$P(Z_{ij} = NA) = \text{logit}^{-1}(XB)_{ij} \times c_j$$

where c_j is calibrated to achieve the target missing rate.

MNAR: The lowest `rho[j]` proportion of values in each column are set as missing.


```

# Example 5: Use generated data with missoNet
library(missoNet)
sim.dat <- generateData(n = 400, p = 50, q = 10, rho = 0.15)

# Split into training and test sets
train.idx <- 1:300
test.idx <- 301:400

# Fit missoNet model
fit <- missoNet(X = sim.dat$X[train.idx, ],
               Y = sim.dat$Z[train.idx, ],
               lambda.beta = 0.1,
               lambda.theta = 0.1)

# Evaluate on test set
pred <- predict(fit, newx = sim.dat$X[test.idx, ])

```

 missoNet

Fit missoNet models with missing responses

Description

Fit a penalized multi-task regression with a response-network (Θ) under missing responses. The method jointly estimates the coefficient matrix \mathbf{B} and the precision matrix Θ via penalized likelihood with ℓ_1 penalties on \mathbf{B} and the off-diagonal entries of Θ .

Usage

```

missoNet(
  X,
  Y,
  rho = NULL,
  GoF = "eBIC",
  lambda.beta = NULL,
  lambda.theta = NULL,
  lambda.beta.min.ratio = NULL,
  lambda.theta.min.ratio = NULL,
  n.lambda.beta = NULL,
  n.lambda.theta = NULL,
  beta.pen.factor = NULL,
  theta.pen.factor = NULL,
  penalize.diagonal = NULL,
  beta.max.iter = 10000,
  beta.tol = 1e-05,
  theta.max.iter = 10000,

```

```

theta.tol = 1e-05,
eta = 0.8,
eps = 1e-08,
standardize = TRUE,
standardize.response = TRUE,
relax.net = FALSE,
adaptive.search = FALSE,
parallel = FALSE,
c1 = NULL,
verbose = 1
)

```

Arguments

X Numeric matrix ($n \times p$). Predictors (no missing values).

Y Numeric matrix ($n \times q$). Responses, may contain NA/NaN.

rho Optional numeric vector of length q . Working missingness probabilities; if NULL (default), estimated from Y .

GoF Character. Goodness-of-fit criterion: "AIC", "BIC", or "eBIC" (default).

lambda.beta, lambda.theta Optional numeric vectors (or scalars). Candidate regularization paths for \mathbf{B} and Θ . If NULL, paths are generated automatically.

lambda.beta.min.ratio, lambda.theta.min.ratio Optional numerics in $(0, 1]$. Ratio of the smallest to largest lambda when generating paths (ignored if the corresponding `lambda.*` is supplied).

n.lambda.beta, n.lambda.theta Optional integers. Lengths of automatically generated lambda paths (ignored if the corresponding `lambda.*` is supplied).

beta.pen.factor Optional $p \times q$ non-negative matrix of element-wise penalty multipliers for \mathbf{B} . Inf = maximum penalty; 0 = no penalty for that coefficient. Default: all 1s (equal penalty).

theta.pen.factor Optional $q \times q$ non-negative matrix of element-wise penalty multipliers for Θ . Off-diagonal entries control edge penalties; diagonal treatment is governed by `penalize.diagonal`. Inf = maximum penalty; 0 = no penalty for that coefficient. Default: all 1s (equal penalty).

penalize.diagonal Logical or NULL. Whether to penalize the diagonal of Θ . If NULL (default) the choice is made automatically.

beta.max.iter, theta.max.iter Integers. Max iterations for the \mathbf{B} update (FISTA) and Θ update (graphical lasso). Defaults: 10000.

beta.tol, theta.tol Numerics > 0 . Convergence tolerances for the \mathbf{B} and Θ updates. Defaults: 1e-5.

| | |
|----------------------|--|
| eta | Numeric in $(0, 1)$. Backtracking line-search parameter for the \mathbf{B} update (default 0.8). |
| eps | Numeric in $(0, 1)$. Eigenvalue floor used to stabilize positive definiteness operations (default 1e-8). |
| standardize | Logical. Standardize columns of X internally? Default TRUE. |
| standardize.response | Logical. Standardize columns of Y internally? Default TRUE. |
| relax.net | (Experimental) Logical. If TRUE, refit active edges of Θ without ℓ_1 penalty (de-biased network). Default FALSE. |
| adaptive.search | (Experimental) Logical. Use adaptive two-stage lambda search? Default FALSE. |
| parallel | Logical. Evaluate parts of the grid in parallel using a provided cluster? Default FALSE. |
| cl | Optional cluster from <code>parallel::makeCluster()</code> (required if <code>parallel = TRUE</code>). |
| verbose | Integer in $0, 1, 2$. 0 = silent, 1 = progress (default), 2 = detailed tracing (not supported in parallel mode). |

Details

The conditional Gaussian model is

$$Y_i = \mu + X_i \mathbf{B} + E_i, \quad E_i \sim \mathcal{N}_q(0, \Theta^{-1}).$$

where:

- Y_i is the i -th observation of q responses
- X_i is the i -th observation of p predictors
- \mathbf{B} is the $p \times q$ coefficient matrix
- Θ is the $q \times q$ precision matrix
- μ is the intercept vector

The parameters are estimated by solving:

$$\min_{\mathbf{B}, \Theta > 0} g(\mathbf{B}, \Theta) + \lambda_B \|\mathbf{B}\|_1 + \lambda_\Theta \|\Theta\|_{1, \text{off}}$$

where g is the negative log-likelihood.

Missing values in Y are accommodated through unbiased estimating equations using column-wise observation probabilities. Internally, X and Y may be standardized for numerical stability; returned estimates are re-scaled back to the original units.

The grid search spans `lambda.beta` and `lambda.theta`. The optimal pair is selected by the user-chosen goodness-of-fit criterion `GoF`: "AIC", "BIC", or "eBIC" (default). If `adaptive.search = TRUE`, a two-stage pre-optimization narrows the grid before the main search (faster on large problems, with a small risk of missing the global optimum).

Value

A list of class "missoNet" with components:

est.min List at the selected lambda pair: Beta ($p \times q$), Theta ($q \times q$), intercept mu (length q), lambda.beta, lambda.theta, lambda.beta.idx, lambda.theta.idx, scalar gof (AIC/BIC/eBIC at optimum), and (if requested) relax.net.

rho Length- q vector of working missingness probabilities.

lambda.beta.seq, lambda.theta.seq Unique lambda values explored along the grid for \mathbf{B} and Θ .

penalize.diagonal Logical indicating whether the diagonal of Θ was penalized.

beta.pen.factor, theta.pen.factor Penalty factor matrices actually used.

param_set List with fitting diagnostics: n, p, q, standardize, standardize.response, the vector of criterion values gof, and the evaluated grids gof.grid.beta, gof.grid.theta (length equals number of fitted models).

Author(s)

Yixiao Zeng <yixiao.zeng@mail.mcgill.ca>, Celia M. T. Greenwood

References

Zeng, Y., et al. (2025). *Multivariate regression with missing response data for modelling regional DNA methylation QTLs*. arXiv:2507.05990.

See Also

[cv.missoNet](#) for cross-validated selection; generic methods such as `plot()` and `predict()` for objects of class "missoNet".

Examples

```
sim <- generateData(n = 120, p = 10, q = 6, rho = 0.1)
X <- sim$X; Y <- sim$Z
```

```
# Fit with defaults (criterion = eBIC)
fit1 <- missoNet(X, Y)
# Extract the optimal estimates
Beta.hat <- fit1$est.min$Beta
Theta.hat <- fit1$est.min$Theta
```

```
# Plot missoNet results
plot(fit1, type = "heatmap")
plot(fit1, type = "scatter")
```

```
# Provide short lambda paths
fit2 <- missoNet(
  X, Y,
  lambda.beta = 10^seq(0, -2, length.out = 5),
  lambda.theta = 10^seq(0, -2, length.out = 5),
```

```

    GoF = "BIC"
  )

# Test single lambda choice
fit3 <- missoNet(
  X, Y,
  lambda.beta = 0.1,
  lambda.theta = 0.1,
)

# De-biased network on the active set
fit4 <- missoNet(X, Y, relax.net = TRUE, verbose = 0)

# Adaptive search for large problems
fit5 <- missoNet(X = X, Y = Y, adaptive.search = TRUE, verbose = 0)

# Parallel (requires a cluster)
library(parallel)
cl <- makeCluster(2)
fit_par <- missoNet(X, Y, parallel = TRUE, cl = cl, verbose = 0)
stopCluster(cl)

```

plot.missoNet

Plot methods for missoNet and cross-validated fits

Description

Visualize either the cross-validation (CV) error surface or the goodness-of-fit (GoF) surface over the λ_B - λ_θ grid for objects returned by `missoNet` or `cv.missoNet`. Two display types are supported: a 2D heatmap (default) and a 3D scatter surface.

Usage

```

## S3 method for class 'missoNet'
plot(
  x,
  type = c("heatmap", "scatter"),
  detailed.axes = TRUE,
  plt.surf = TRUE,
  ...
)

```

Arguments

| | |
|------|---|
| x | A fitted object returned by <code>missoNet</code> or <code>cv.missoNet</code> . |
| type | Character string specifying the plot type. One of "heatmap" (default) or "scatter". |

| | |
|---------------|---|
| detailed.axes | Logical; if TRUE (default) show dense axis labels. If FALSE, a sparser labeling is used to avoid clutter. |
| plt.surf | Logical; for type = "scatter" only, draw light surface grid lines and highlight the minimum point. Ignored for heatmaps. Default TRUE. |
| ... | Additional graphical arguments forwarded to Heatmap when type = "heatmap", or to scatterplot3d when type = "scatter". |

Details

This S3 method detects whether `x` contains cross-validation results and chooses an appropriate plotting backend:

- **Heatmap:** uses [Heatmap](#) with a viridis-like color ramp (via [colorRamp2](#)). The selected $(\lambda_B, \lambda_\Theta)$ is outlined in white; 1-SE choices (if present) are highlighted with dashed/dotted outlines.
- **Scatter:** uses [scatterplot3d](#) to draw the error/GoF surface on \log_{10} scales. When `plt.surf = TRUE`, light lattice lines are added, and the minimum is marked.

Value

- For type = "heatmap": a `ComplexHeatmap` `Heatmap` object (invisibly drawn by `ComplexHeatmap`).
- For type = "scatter": a "scatterplot3d" object, returned *invisibly*.

What gets plotted

- **CV objects** (created by [cv.missoNet](#) or any `missoNet` object that carries CV results): the color encodes the mean CV error for each $(\lambda_B, \lambda_\Theta)$ pair. The *minimum-error* solution is outlined; if 1-SE solutions were computed, they are also marked (dashed/dotted outlines).
- **Non-CV objects** (created by [missoNet](#) without CV): the color encodes the GoF value over the grid; the selected *minimum* (best) solution is outlined.

Axes and scales

For heatmaps, axes are the raw λ values; rows are λ_Θ and columns are λ_B . For 3D scatter plots, both λ axes are shown on the \log_{10} scale for readability.

Color mapping

A viridis-like palette is used. Breaks are based on distribution quantiles of the CV error or GoF values to enhance contrast across the grid.

Dependencies

Requires **`ComplexHeatmap`**, **`circlize`**, **`scatterplot3d`**, and **`grid`**.

Author(s)

Yixiao Zeng <yixiao.zeng@mail.mcgill.ca>, Celia M. T. Greenwood

See Also

[missoNet](#), [cv.missoNet](#), [Heatmap](#), [scatterplot3d](#)

Examples

```
sim <- generateData(n = 150, p = 10, q = 8, rho = 0.1, missing.type = "MCAR")

## Fit a model without CV (plots GoF surface)
fit <- missoNet(X = sim$X, Y = sim$Z, verbose = 0)
plot(fit, type = "heatmap")           # GoF heatmap
plot(fit, type = "scatter", plt.surf = TRUE)   # GoF 3D scatter

## Cross-validation (plots CV error surface)
cvfit <- cv.missoNet(X = sim$X, Y = sim$Z, verbose = 0)
plot(cvfit, type = "heatmap", detailed.axes = FALSE)
plot(cvfit, type = "scatter", plt.surf = FALSE)
```

| | |
|------------------|---|
| predict.missoNet | <i>Predict method for missoNet models</i> |
|------------------|---|

Description

Generate predicted responses for new observations from a fitted [missoNet](#) (or cross-validated) model. The prediction at a given regularization choice $(\lambda_B, \lambda_\Theta)$ uses the fitted intercept(s) $\hat{\mu}$ and coefficient matrix \hat{B} :

$$\hat{Y} = \mathbf{1}_n \hat{\mu}^\top + X_{\text{new}} \hat{B}.$$

Usage

```
## S3 method for class 'missoNet'
predict(
  object,
  newx,
  s = c("lambda.min", "lambda.1se.beta", "lambda.1se.theta"),
  ...
)

## S3 method for class 'cv.missoNet'
predict(
  object,
  newx,
  s = c("lambda.min", "lambda.1se.beta", "lambda.1se.theta"),
  ...
)
```

Arguments

| | |
|--------|--|
| object | A fitted <code>missoNet</code> (or cross-validated <code>missoNet</code>) object that contains the components <code>\$est.min</code> (and optionally <code>\$est.1se.beta</code> , <code>\$est.1se.theta</code>), each with numeric fields <code>\$mu</code> (length q) and <code>\$Beta</code> ($p \times q$). |
| newx | Numeric matrix of predictors with p columns (no intercept column of 1s). Missing or non-finite values are not allowed. Columns must be in the same order/scale used to fit <code>object</code> . |
| s | Character string selecting the stored solution; one of <code>c("lambda.min", "lambda.1se.beta", "lambda.1se.theta")</code> . |
| ... | Ignored; included for S3 compatibility. |

Details

This method does not modify or standardize `newx`. If the model was trained with standardization, ensure that `newx` has been prepared in the same way as the training data (same centering/scaling and column order).

Value

A numeric matrix of predicted responses of dimension $n_{\text{new}} \times q$. Row names are taken from `newx` (if any), and column names are inherited from the fitted coefficient matrix (if any).

Which solution is used

The `s` argument selects the stored solution:

- `"lambda.min"` (default): the minimum CV error or selected GoF solution, stored in `object$est.min`.
- `"lambda.1se.beta"`: the 1-SE solution favoring larger λ_B , stored in `object$est.1se.beta`.
- `"lambda.1se.theta"`: the 1-SE solution favoring larger λ_Θ , stored in `object$est.1se.theta`.

1-SE solutions are available only if the model was fit with `compute.1se = TRUE` during training or cross-validation.

See Also

[missoNet](#), [cv.missoNet](#), [plot.missoNet](#)

Examples

```
sim <- generateData(n = 200, p = 8, q = 6, rho = 0.1,
                  missing.type = "MCAR", seed = 123)
tr <- 1:150
tst <- 151:200

## Cross-validated fit, keeping 1-SE solutions
cvfit <- cv.missoNet(X = sim$X[tr, ], Y = sim$Z[tr, ], kfold = 5,
                   compute.1se = TRUE, verbose = 0)

## Predict on held-out set
```

```
yhat_min <- predict(cvfit, newx = sim$X[tst, ], s = "lambda.min")  
yhat_b1se <- predict(cvfit, newx = sim$X[tst, ], s = "lambda.1se.beta")  
yhat_t1se <- predict(cvfit, newx = sim$X[tst, ], s = "lambda.1se.theta")  
dim(yhat_min) # 50 x q
```

Index

- * **hplot**

- plot.missoNet, 15

- * **models**

- plot.missoNet, 15

- predict.missoNet, 17

- * **regression**

- predict.missoNet, 17

colorRamp2, 16

cv.missoNet, 2, 3, 4, 10, 14–18

generateData, 3, 8

Heatmap, 16, 17

missoNet, 2–4, 7, 10, 11, 15–18

missoNet-package, 2

package (missoNet-package), 2

plot.cv.missoNet (plot.missoNet), 15

plot.missoNet, 2, 3, 15, 18

predict.cv.missoNet (predict.missoNet),
17

predict.missoNet, 2, 3, 17

scatterplot3d, 16, 17