Package: mantar (via r-universe)

May 26, 2026

Title Missingness Alleviation for Network Analysis

Version 0.3.0

Description Provides functionality for estimating cross-sectional network structures representing partial correlations while accounting for missing data. Networks are estimated via neighborhood selection or regularization, with model selection guided by information criteria. Missing data can be handled primarily via multiple imputation or a maximum likelihood-based approach, as demonstrated by Nehler and Schultze (2025) <doi:10.1080/00273171.2025.2503833> and Nehler and Schultze (2026) <doi:10.1037/met0000828>. Deletion-based approaches are also available but play a secondary role.

License GPL (>= 3)

Depends R (>= 4.1.0)

Imports Rdpack, mathjaxr, stats, Matrix, glassoFast

Suggests numDeriv, mice, lavaan, qgraph, testthat (>= 3.0.0), knitr, rmarkdown

LazyData true

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

RdMacros Rdpack, mathjaxr

Config/testthat/edition 3

URL `https://github.com/kai-nehler/mantar`

BugReports `https://github.com/kai-nehler/mantar/issues`

VignetteBuilder knitr

Repository https://kai-nehler.r-universe.dev

Date/Publication 2026-05-26 14:46:06 UTC

RemoteUrl https://github.com/kai-nehler/mantar

RemoteRef HEAD

RemoteSha 3b800d68ec2877bd1140471a04ff1b8c1447aca8

Contents

`cor_calc` *Correlation Matrix Estimation with Support for Multiple Correlation Types*

Description

Computes a correlation matrix from raw data while accounting for missing values through several missing-data handling strategies. Supports different correlation types based on whether variables are treated as ordered.

Usage

```
cor_calc(
  data,
  ordered = FALSE,
  network_vars = NULL,
  auxiliary_vars = NULL,
  missing_handling = "two-step-em",
  nimp = 20,
  imp_method = "pmm",
  maxit = 10,
  ...
)
```

Arguments

`data` Data frame or matrix containing the variables for which the correlation matrix is to be computed. May include missing values.

`ordered` Logical vector indicating whether each variable in `data` should be treated as ordered categorical when computing the correlation matrix. If a single logical value is supplied, it is recycled to all variables.

`network_vars` Optional character or numeric vector specifying the variables for which the correlation matrix is returned, typically the variables used in subsequent network estimation. Default is `NULL`, which means that all variables are included.

`auxiliary_vars`	Optional character or numeric vector specifying variables in the data set that are additionally used for correlation estimation and missing-data handling, but are not included in the returned correlation matrix. Default is `NULL`, which means that no auxiliary variables are used.
`missing_handling`	Character string specifying how the correlation matrix is estimated from `data` in the presence of missing values. Possible values are: `"two-step-em"` Uses a classical EM algorithm to estimate the correlation matrix from `data`. `"stacked-mi"` Uses stacked multiple imputation to estimate the correlation matrix from `data`. `"pairwise"` Uses pairwise deletion to compute correlations from `data`. `"listwise"` Uses listwise deletion to compute correlations from `data`.
`nimp`	Number of imputations (default: 20) to be used when `missing_handling = "stacked-mi"`.
`imp_method`	Character string specifying the imputation method to be used when `missing_handling = "stacked-mi"` (default: `"pmm"` - predictive mean matching).
`maxit`	Maximum number of iterations for the imputation algorithm when `missing_handling = "stacked-mi"` (default: 10).
`...`	Further arguments passed to internal functions.

Details

Correlations are computed pairwise:

- Polychoric correlations for two ordered variables,
- Polyserial correlations for one ordered and one continuous variable,
- Pearson correlations for two continuous variables.

Treating variables as ordered requires the missing handling method to be either `"stacked-mi"` or `"listwise"`

Means are computed whenever Pearson correlations are used. If any variable is treated as ordered, `means` is returned as NULL.

Value

A list containing:

mat Estimated correlation matrix.

means Vector of estimated means. If any variable is treated as ordered, means is returned as NULL.

cor_method A matrix indicating the correlation method used for each variable pair.

imputed_data Imputed datasets used for estimation. For `missing_handling = "stacked-mi"`, the imputed data are returned as a `mids` object from the **mice** package; otherwise `NULL`

args List of settings used in the correlation estimation.

Examples

```
# Estimate correlation matrix from full data set
result <- cor_calc(data = mantar_dummy_full_cont,
                   ordered = FALSE)

# View estimated correlation matrix and methods used
result$mat
result$cor_method

# Estimate correlation matrix for data set with missings
result_mis <- cor_calc(data = mantar_dummy_mis_cont,
                       ordered = FALSE,
                       missing_handling = "two-step-em")

# View estimated correlation matrix and methods used
result_mis$mat
result_mis$cor_method
```

---

mantar_dummy_data   *Dummy data sets for illustration purposes in the mantar package*

Description

These simulated data sets are provided for illustration purposes. They are based on a sparse psychological network structure with a single underlying construct. The column names represent core properties of neuroticism but are purely made up to make the example more illustrative.

Format

All data sets are data frames with 400 rows and 8 columns. The columns are:

EmoReactivity Tending to feel emotions strongly in response to life events.

TendWorry Being more likely to feel concerned or uneasy.

StressSens Feeling more stressed in challenging or uncertain situations.

SelfAware Being conscious of one's own feelings and how they shift.

Moodiness Experiencing occasional changes in mood.

Cautious Being careful and thinking ahead about possible negative outcomes.

ThoughtFuture Reflecting on what might go wrong and preparing for it.

RespCriticism Being affected by others' feedback or disapproval.

Details

The following data sets are available:

- `mantar_dummy_full_cont`: A complete data set of continuous variables without missing values.

- mantar_dummy_mis_cont: A data set of continuous variables with approximately 30% missing values in each column.
- mantar_dummy_full_cat: A complete data set where variables are ordered categorical without missing values.
- mantar_dummy_mis_cat: A data set where variables are ordered categorical with approximately 25% missing values in each column.
- mantar_dummy_full_mix: A complete data set with a mix of continuous and ordered categorical variables without missing values.
- mantar_dummy_mis_mix: A data set with a mix of continuous and ordered categorical variables with approximately 25% missing values in each column.

Examples

```
# Load selected data set
data(mantar_dummy_full_cont)
data(mantar_dummy_mis_cont)

# View the first few rows of selected data sets
head(mantar_dummy_full_cont)
head(mantar_dummy_mis_cont)
```

---

neighborhood_net *Network Estimation via Neighborhood Selection using Information Criteria*

Description

Estimates a network structure through node-wise regression models, where each regression is selected via an information-criterion–based stepwise procedure. The selected regression coefficients are subsequently combined into partial correlations to form the final network.

Usage

```
neighborhood_net(
  data = NULL,
  ns = NULL,
  mat = NULL,
  network_vars = NULL,
  auxiliary_vars = NULL,
  n_calc = "individual",
  ic_type = "bic",
  ordered = FALSE,
  pcor_merge_rule = "and",
  missing_handling = "two-step-em",
  nimp = 20,
  imp_method = "pmm",
  ...
)
```

Arguments

`data`
: Optional raw data matrix or data frame containing the variables to be included in the network. May include missing values. If `data` is not provided (`NULL`), a covariance or correlation matrix must be supplied in `mat`.

`ns`
: Optional numeric sample size specification. Can be a single value (same sample size is used for all regressions) or a vector (e.g., variable-wise sample sizes). When `data` is provided and `ns` is `NULL`, sample sizes are derived automatically from `data`. When `mat` is supplied instead of raw data, `ns` must be provided and should reflect the sample size underlying `mat`.

`mat`
: Optional covariance or correlation matrix for the variables to be included in the network. Used only when `data` is `NULL`. If both `data` and `mat` are supplied, `mat` is ignored. When `mat` is used, `ns` must also be provided.

`network_vars`
: Optional character or numeric vector specifying the variables in the data set or matrix that are used for network estimation. Default is `NULL`, which means that all variables are used.

`auxiliary_vars`
: Optional character or numeric vector specifying variables in the data set that are used as auxiliary variables for correlation estimation and missing-data handling, but are not included in the network estimation. Default is `NULL`, which means that no auxiliary variables are used.

`n_calc`
: Character string specifying how per-variable sample sizes for node-wise regression models are computed when `ns` is not supplied. If `ns` is provided, its values are used directly and `n_calc` is ignored. Possible values are:

  - `"individual"` For each variable, uses the number of non-missing observations for that variable.
  - `"average"` Computes the average number of non-missing observations across all variables and uses this average as the sample size for every variable.
  - `"max"` Computes the maximum number of non-missing observations across all variables and uses this maximum as the sample size for every variable.
  - `"total"` Uses the total number of rows in `data` as the sample size for every variable.

`ic_type`
: Type of information criterion to compute for model selection in the node-wise regression models. Options are `bic` (default), `aic`, `aicc`.

`ordered`
: Logical vector indicating whether each variable in `data` should be treated as ordered categorical. Only used when `data` is provided. If a single logical value is supplied, it is recycled to all variables.

`pcor_merge_rule`
: Character string specifying how regression weights from the node-wise models are merged into partial correlations. Possible values are:

  - `"and"` Estimates a partial correlation only if the regression weights in both directions (e.g., from node 1 to 2 and from node 2 to 1) are non-zero in the final models.
  - `"or"` Uses the available regression weight from one direction as the partial correlation if the corresponding regression in the other direction is not included in the final model.

missing_handling
: Character string specifying how correlations are estimated from the `data` input in the presence of missing values. Possible values are:

  `"two-step-em"` Uses a classical EM algorithm to estimate the correlation matrix from `data`.

  `"stacked-mi"` Uses stacked multiple imputation to estimate the correlation matrix from `data`.

  `"pairwise"` Uses pairwise deletion to compute correlations from `data`.

  `"listwise"` Uses listwise deletion to compute correlations from `data`.

nimp
: Number of imputations (default: 20) to be used when `missing_handling = "stacked-mi"`.

imp_method
: Character string specifying the imputation method to be used when `missing_handling = "stacked-mi"` (default: `"pmm"` - predictive mean matching).

...
: Further arguments passed to internal functions.

Details

This function estimates a network structure using neighborhood selection guided by information criteria. Simulations by Williams et al. (2019) indicated that using the `"and"` rule for merging regression weights tends to yield more accurate partial correlation estimates than the `"or"` rule.

The argument `ic_type` specifies which information criterion is computed. All criteria are computed based on the log-likelihood of the maximum likelihood estimated regression model, where the residual variance determines the likelihood. The following options are available:

`"aic"`: Akaike Information Criterion (Akaike 1974); defined as $\mathrm{AIC} = -2\ell + 2k$, where ℓ is the log-likelihood of the model and k is the number of estimated parameters (including the intercept).

`"bic"`: Bayesian Information Criterion (Schwarz 1978); defined as $\mathrm{BIC} = -2\ell + k\log(n)$, where ℓ is the log-likelihood of the model, k is the number of estimated parameters (including the intercept) and n is the sample size.

`"aicc"`: Corrected Akaike Information Criterion (Hurvich and Tsai 1989); particularly useful in small samples where AIC tends to be biased. Defined as $\mathrm{AIC_c} = \mathrm{AIC} + \frac{2k(k+1)}{n-k-1}$, where k is the number of estimated parameters (including the intercept) and n is the sample size.

Missing Handling

To handle missing data, the function offers two approaches: a two-step expectation-maximization (EM) algorithm and stacked multiple imputation. According to simulations by Nehler and Schultze (2026), stacked multiple imputation performs reliably across a range of sample sizes. In contrast, the two-step EM algorithm provides accurate results primarily when the sample size is large relative to the amount of missingness and network complexity - but may still be preferred in such cases due to its much faster runtime.

Currently, the function only supports variables that are directly included in the network analysis; auxiliary variables for missing handling are not yet supported. During imputation, all variables are imputed by default using predictive mean matching (see e.g., van Buuren 2018), with all other variables in the data set serving as predictors.

Value

A list with the following elements:

pcor Partial correlation matrix estimated from the node-wise regressions.

betas Matrix of regression coefficients from the final regression models.

ns Sample sizes used for each variable in the node-wise regressions.

imputed_data The imputed data used for estimation, returned as a `mids` object from the **mice** package when `missing_handling = "stacked-mi"`; otherwise `NULL`.

args List of settings used in the network estimation.

References

Akaike H (1974). "A new look at the statistical model identification." *IEEE Transactions on Automatic Control*, **19**, 716–723. doi:10.1109/TAC.1974.1100705.

Hurvich CM, Tsai C (1989). "Regression and time series model selection in small samples." *Biometrika*, **76**(2), 297–307. doi:10.1093/biomet/76.2.297.

Nehler KJ, Schultze M (2026). "Handling Missing Values When Using Neighborhood Selection for Network Analysis." *Psychological Methods*, **Online ahead of print.**. doi:10.1037/met0000828.

Schwarz G (1978). "Estimating the dimension of a model." *Annals of Statistics*, **6**(2), 461–464. doi:10.1214/aos/1176344136.

van Buuren S (2018). *Flexible Imputation of Missing Data*, 2 edition. CRC Press, Boca Raton. doi:10.1201/9780429492259.

Williams DR, Rhemtulla M, Wysocki AC, Rast P (2019). "On Nonregularized Estimation of Psychological Networks." *Multivariate Behavioral Research*, **54**(5), 719–750. doi:10.1080/00273171.2019.1575716.

Examples

```
# Estimate network from full data set
# Using Akaike information criterion
result <- neighborhood_net(data = mantar_dummy_full_cont,
ic_type = "aic")

# View estimated partial correlations
result$pcor

# Estimate network for data set with missings
# Using Bayesian Information Criterion, individual sample sizes, and two-step EM
result_mis <- neighborhood_net(data = mantar_dummy_mis_cont,
n_calc = "individual",
missing_handling = "two-step-em",
ic_type = "bic")

# View estimated partial correlations
result_mis$pcor
```

ordered_suggest *Heuristic procedure for identifying ordered categorical variables*

Description

Suggests which variables in a data set may be treated as ordered categorical based on their number of unique categories and the amount of available information for estimating the network structure. This function provides a preliminary, non-binding recommendation and should be interpreted as a beta-level heuristic.

Usage

```
ordered_suggest(data, max_categories = 7)
```

Arguments

`data`	Raw data matrix or data frame containing the variables to be included in the network. May include missing values.
`max_categories`	Maximum number of categories a variable may have to be treated as ordered (default: 7).

Details

While polychoric correlations are generally more appropriate for ordered categorical data (foldness.2022), they may encounter estimation problems if the number of available observations is small relative to the number of estimated parameters (see e.g., Johal and Rhemtulla 2023). Our preliminary simulations suggest that in such cases Pearson correlations may introduce less bias, an effect that becomes even more pronounced when data are missing.

This helper function provides a recommendation on which variables to treat as ordered. In general, variables with more than `max_categories` categories are recommended to be treated as continuous, whereas for variables with fewer categories the procedure evaluates whether the amount of available information is too limited to justify polychoric estimation, in which case Pearson correlations are recommended instead. This procedure is only a helper, is still under early development, and may be refined in future versions.

Value

A logical vector of length `ncol(data)` indicating, for each variable, whether it is recommended to be treated as ordered (`TRUE`) or continuous (`FALSE`). Additionally, a message is printed to the console summarizing the recommendation in terms of which correlation methods to use.

References

Johal SK, Rhemtulla M (2023). "Comparing Estimation Methods for Psychometric Networks with Ordinal Data." *Psychological Methods*, **28**(6), 1251–1272. doi:10.1037/met0000449.

Examples

```
# Suggest ordered variables in a data set with mixed variable types
# (400 observations for 8 variables)
ordered_suggest(data = mantar_dummy_full_mix, max_categories = 7)
```

`regression_opt` *Stepwise Multiple Regression Model Search based on Information Criteria*

Description

Performs stepwise model selection for multiple regression using information criteria to identify the optimal regression model.

Usage

```
regression_opt(
  data = NULL,
  n = NULL,
  mat = NULL,
  dep_ind,
  n_calc = "individual",
  ic_type = "bic",
  ordered = FALSE,
  missing_handling = "stacked-mi",
  nimp = 20,
  imp_method = "pmm",
  ...
)
```

Arguments

`data` Raw data matrix or data frame containing the variables to be included in the regression models. May include missing values. If `data` is `NULL`, a covariance or correlation matrix must be supplied in `mat`.

`n` Numeric value specifying the sample size used in calculating the information criteria. If not provided, it is derived from `data`. When `mat` is supplied instead of raw data, `n` must be provided.

`mat` Optional covariance or correlation matrix for the variables to be included in the regression. Used only when `data` is `NULL`.

`dep_ind` Index of the column in `data` to be used as the dependent variable in the regression model.

`n_calc` Character string specifying how the sample size is calculated when `n` is not provided. Possible values are:

"individual" Uses the number of non-missing observations for the variable used as the dependent variable.

"average" Uses the average number of non-missing observations across all variables.

"max" Uses the maximum number of non-missing observations across all variables.

"total" Uses the total number of rows in data.

ic_type
: Type of information criterion to compute for model selection. Options are bic (default), aic, aicc.

ordered
: Logical vector indicating whether each variable in data should be treated as ordered categorical when computing the correlation matrix. If a single logical value is supplied, it is recycled to all variables. Only used when data is provided.

missing_handling
: Character string specifying how the correlation matrix is estimated from data in the presence of missing values. Possible values are:

"two-step-em" Uses a classical EM algorithm to estimate the correlation matrix from data.

"stacked-mi" Uses stacked multiple imputation to estimate the correlation matrix from data.

"pairwise" Uses pairwise deletion to compute correlations from data.

"listwise" Uses listwise deletion to compute correlations from data.

nimp
: Number of imputations (default: 20) to be used when missing_handling = "stacked-mi".

imp_method
: Character string specifying the imputation method to be used when missing_handling = "stacked-mi" (default: "pmm" - predictive mean matching).

...
: Further arguments passed to internal functions.

Details

This function performs stepwise model selection for multiple regression using information criteria. It was originally developed as a component of the neighborhood selection framework for network estimation (Nehler and Schultze 2026), where each node-wise regression model is selected individually. However, the procedure can also be used as a standalone tool for exploratory regression model search, particularly in settings with missing data. Unlike standard stepwise regression functions, this implementation explicitly supports missing-data handling strategies, making it suitable for situations in which classical methods fail or produce biased results.

The argument ic_type specifies which information criterion is computed. All criteria are computed based on the log-likelihood of the maximum likelihood estimated regression model, where the residual variance determines the likelihood. The following options are available:

"aic": Akaike Information Criterion (Akaike 1974); defined as $\mathrm{AIC} = -2\ell + 2k$, where ℓ is the log-likelihood of the model and k is the number of estimated parameters (including the intercept).

"bic": Bayesian Information Criterion (Schwarz 1978); defined as $\mathrm{BIC} = -2\ell + k\log(n)$, where ℓ is the log-likelihood of the model, k is the number of estimated parameters (including the intercept) and n is the sample size.

"aicc": Corrected Akaike Information Criterion (Hurvich and Tsai 1989); particularly useful in small samples where AIC tends to be biased. Defined as $\mathrm{AIC_c} = \mathrm{AIC} + \frac{2k(k+1)}{n-k-1}$, where k is the number of estimated parameters (including the intercept) and n is the sample size.

Value

A list with the following elements:

regression Named vector of regression coefficients for the dependent variable.

R2 R-squared value of the regression model.

n Sample size used in the regression model.

args List of settings used in the regression model.

References

Akaike H (1974). "A new look at the statistical model identification." *IEEE Transactions on Automatic Control*, **19**, 716–723. doi:10.1109/TAC.1974.1100705.

Hurvich CM, Tsai C (1989). "Regression and time series model selection in small samples." *Biometrika*, **76**(2), 297–307. doi:10.1093/biomet/76.2.297.

Nehler KJ, Schultze M (2026). "Handling Missing Values When Using Neighborhood Selection for Network Analysis." *Psychological Methods*, **Online ahead of print.**. doi:10.1037/met0000828.

Schwarz G (1978). "Estimating the dimension of a model." *Annals of Statistics*, **6**(2), 461–464. doi:10.1214/aos/1176344136.

Examples

```
# For full data using AIC
# First variable of the data set as dependent variable
result <- regression_opt(
  data = mantar_dummy_full_cont,
  dep_ind = 1,
  ic_type = "aic"
)

# View regression coefficients and R-squared
result$regression
result$R2

# For data with missingess using BIC
# Second variable of the data set as dependent variable
# Using individual sample size of the dependent variable and stacked Multiple Imputation

result_mis <- regression_opt(
 data = mantar_dummy_mis_cont,
```

```
dep_ind = 2,
n_calc = "individual",
missing_handling = "two-step-em",
ic_type = "bic"
)

# View regression coefficients and R-squared
result_mis$regression
result_mis$R2
```

---

regularization_net *Regularized Network Estimation*

Description

Estimate cross-sectional network structures using regularization. The function first computes the correlations (if needed), constructs a grid of tuning parameters tailored to the chosen penalty, and then selects the final network by minimizing a user-specified information criterion.

Usage

```
regularization_net(
  data = NULL,
  ns = NULL,
  mat = NULL,
  likelihood = "obs_based",
  means = NULL,
  network_vars = NULL,
  auxiliary_vars = NULL,
  n_calc = "average",
  count_diagonal = TRUE,
  ic_type = NULL,
  extended_gamma = 0.5,
  penalty = "atan",
  vary = "lambda",
  n_lambda = NULL,
  lambda_min_ratio = 0.01,
  n_gamma = 50,
  pen_diag = FALSE,
  lambda = NULL,
  gamma = NULL,
  ordered = FALSE,
  missing_handling = "two-step-em",
  nimp = 20,
  imp_method = "pmm",
  ...
)
```

Arguments

`data` Optional raw data matrix or data frame containing the variables. May include missing values. If `data` is provided and `mat` is `NULL`, a correlation matrix is estimated from `data` and used for network estimation. If both `data` and `mat` are supplied, `mat` is used for network estimation, while `data` is used to compute `ns` (if not provided) and for model selection when `likelihood = "obs_based"`.

`ns` Optional numeric sample size specification. Can be either a single value or a matrix containing pairwise sample sizes, with variable-specific sample sizes on the diagonal. In the matrix case, `ns` must be symmetric and have dimensions equal to the number of variables. When `data` is provided and `ns` is `NULL`, sample sizes are derived automatically from `data`. When `mat` is supplied instead of raw data, `ns` must be provided.

`mat` Optional covariance or correlation matrix for the variables to be included in the network. If `mat` is provided without `data`, `ns` must also be specified. If both `data` and `mat` are supplied, `mat` is used for network estimation (see `data`).

`likelihood` Character string specifying how the log-likelihood is computed. Possible values are:

- `"obs_based"` Uses the observed-data log-likelihood.
- `"mat_based"` Uses log-likelihood based on the sample correlation matrix.

`means` Optional vector of variable means of the scaled raw data when no variables are treated as ordered. This argument is primarily intended for internal integration purposes (e.g., with **bootnet**) and should usually not be specified by users directly. It is only used when `likelihood = "obs_based"` and both raw data via `data` and a user-supplied matrix via `mat` are provided, as the means are not estimated from the data in this setting. Note that in the presence of missing data, the means of the scaled raw data are not necessarily zero. Ignored otherwise.

`network_vars` Optional character or numeric vector specifying the variables in the data set or matrix that are used for network estimation. Default is `NULL`, which means that all variables are used.

`auxiliary_vars` Optional character or numeric vector specifying variables in the data set that are used as auxiliary variables for correlation estimation and missing-data handling, but are not included in the network estimation. Default is `NULL`, which means that no auxiliary variables are used.

`n_calc` Character string specifying how the effective sample size is determined. When `data` are provided, it controls how the observation counts across variables are aggregated. When `ns` is a matrix, it controls how the entries of `ns` are combined. If both `data` and `ns` are supplied, the values in `ns` take precedence. This argument is ignored when `ns` is a single numeric value. Possible values are:

- `"average"` Uses the average of the pairwise and variable-specific sample sizes across the entries of `ns`. Importantly, although `ns` is required as a matrix, off-diagonal elements are only considered once.
- `"max"` Uses the maximum pairwise sample size across variable pairs.
- `"total"` Uses the total number of observations. Only applicable when `ns` is not provided and raw data are supplied via `data`.

count_diagonal
: Logical; should observations contributing to the diagonal elements be included when computing the sample size? Only relevant when n_calc = "average" and data or ns are provided.

ic_type
: Character string specifying the type of information criterion used for model selection. Possible values are: "aic", "bic", and "ebic". If no input is provided, defaults to "ebic" when penalty = "glasso" and "bic" otherwise.

extended_gamma
: Numeric gamma parameter used in the extended information criterion calculation. Only relevant when ic_type = "ebic".

penalty
: Character string indicating the type of penalty used for regularization. Available options are described in the Details section.

vary
: Character string specifying which penalty parameter(s) are varied during regularization to determine the optimal network. Possible values are "lambda", "gamma", or "both".

n_lambda
: Number of lambda values to be evaluated. If not specified, the default is 100 when penalty = "glasso" and 50 if lambda is varied for. If vary == "gamma", n_lambda is set to 1.

lambda_min_ratio
: Ratio of the smallest to the largest lambda value.

n_gamma
: Number of gamma values to be evaluated. Is set to 1 if vary == "lambda".

pen_diag
: Logical; should the diagonal elements be penalized in the regularization process?

lambda
: Optional user-specified vector of lambda values.

gamma
: Optional user-specified vector of gamma values.

ordered
: Logical vector indicating which variables in data are treated as ordered (ordinal). Only used when data is provided. If a single logical value is supplied, it is recycled to the length of data.

missing_handling
: Character string specifying how correlations are estimated from the data input in the presence of missing values. Possible values are:

  "two-step-em" Uses a classical EM algorithm to estimate the correlation matrix from data.

  "stacked-mi" Uses stacked multiple imputation to estimate the correlation matrix from data.

  "pairwise" Uses pairwise deletion to compute correlations from data.

  "listwise" Uses listwise deletion to compute correlations from data.

nimp
: Number of imputations (default: 20) to be used when missing_handling = "stacked-mi".

imp_method
: Character string specifying the imputation method to be used when missing_handling = "stacked-mi" (default: "pmm" - predictive mean matching).

...
: Further arguments passed to internal functions.

Details

Penalties

This function supports a range of convex and nonconvex penalties for regularized network estimation.

For convex penalties, the graphical lasso can be used via `penalty = "glasso"` (Friedman et al. 2008).

Another option is the adaptive lasso, specified with `penalty = "adapt"`. By default, it employs $\gamma = 0.5$ (Zou and Li 2008). Smaller values of γ (i.e., $\gamma \to 0$) correspond to stronger penalization, whereas $\gamma = 1$ yields standard ℓ_1 regularization.

The available nonconvex penalties follow the work of Williams (2020), who identified the atan penalty as particularly promising. It serves as the default in this implementation because it has desirable theoretical properties, including consistency in recovering the true model as $n \to \infty$. Additional nonconvex penalties are included for completeness. These were originally implemented in the now–deprecated `R` package **GGMncv** (Williams 2021), and the implementation in **mantar** is based on the corresponding methods from that package.

Several algorithms exist for nonconvex regularized network estimation. In **mantar**, we use the one-step estimator of Zou and Li (2008) because of its computational efficiency and its good performance in settings where $n > p$, which is typically the case in psychological research.

- **Atan**: `penalty = "atan"` (Wang and Zhu 2016). This is currently the default.
- **Exponential**: `penalty = "exp"` (Wang et al. 2018).
- **Log**: `penalty = "log"` (Mazumder et al. 2011).
- **MCP**: `penalty = "mcp"` (Zhang 2010).
- **SCAD**: `penalty = "scad"` (Fan and Li 2001).
- **Seamless** ℓ_0: `penalty = "selo"` (Dicker et al. 2013).
- **SICA**: `penalty = "sica"` (Lv and Fan 2009).

Information Criteria

The argument `ic_type` specifies which information criterion is computed. All criteria are computed based on the log-likelihood of the estimated model.

`"aic"`**:** Akaike Information Criterion (Akaike 1974); defined as $\mathrm{AIC} = -2\ell + 2k$, where ℓ is the log-likelihood of the model and k is the number of freely estimated edge parameters (non-zero edges).

`"bic"`**:** Bayesian Information Criterion (Schwarz 1978); defined as $\mathrm{BIC} = -2\ell + k\log(n)$, where ℓ is the log-likelihood of the model, k is the number of freely estimated edge parameters (non-zero edges), and n is the sample size.

`"ebic"`**:** Extended Bayesian Information Criterion (Chen and Chen 2008); particularly useful in high-dimensional settings. Defined as $\mathrm{EBIC} = -2\ell + k\log(n) + 4\gamma k\log(p)$, where ℓ is the log-likelihood, k is the number of freely estimated edges (non-zero edges), n is the sample size, p is the number of variables, and γ is the extended-penalty parameter.

Conditional Defaults

By default, some tuning parameters depend on the chosen penalty. Specifically, when `penalty = "glasso"`, the number of lambda values `n_lambda` defaults to `100` and `ic_type` defaults to `"ebic"`. For all other penalties, the defaults are `n_lambda = 50` and `ic_type = "bic"`. These defaults can be overridden by specifying `n_lambda` and/or `ic_type` explicitly.

Missing Handling

To handle missing data, the function offers two approaches: a two-step expectation-maximization (EM) algorithm and stacked multiple imputation. According to simulations by Nehler and Schultze (2025), stacked multiple imputation performs reliably across a range of sample sizes. In contrast, the two-step EM algorithm provides accurate results primarily when the sample size is large relative to the amount of missingness and network complexity - but may still be preferred in such cases due to its much faster runtime. Currently, the function only supports variables that are directly included in the network analysis; auxiliary variables for missing handling are not yet supported. During imputation, all variables are imputed by default using predictive mean matching (see e.g., van Buuren 2018), with all other variables in the data set serving as predictors.

Value

A list with the following elements:

pcor Estimated partial correlation matrix corresponding to the selected (optimal) network.

n Effective sample size used, either supplied directly via `n` or derived based on `n_calc`.

cor_method Correlation estimation method used for each variable pair.

imputed_data The imputed data used for estimation, returned as a `mids` object from the **mice** package when `missing_handling = "stacked-mi"`; otherwise `NULL`.

full_results Full set of results returned by the model selection procedure, including all evaluated networks and their fit statistics.

args A list of settings used in the estimation procedure.

References

Akaike H (1974). "A new look at the statistical model identification." *IEEE Transactions on Automatic Control*, **19**, 716–723. doi:10.1109/TAC.1974.1100705.

Chen J, Chen Z (2008). "Extended Bayesian information criteria for model selection." *Biometrika*, **95**(3), 759–771. doi:10.1093/biomet/asn034.

Dicker L, Huang B, Lin X (2013). "Variable selection and estimation with the seamless-L 0 penalty." *Statistica Sinica*, 929–962. `http://www.jstor.org/stable/24310368`.

Fan J, Li R (2001). "Variable selection via nonconcave penalized likelihood and its oracle properties." *Journal of the American Statistical Association*, **96**(456), 1348–1360. doi:10.1198/016214501753382273.

Friedman J, Hastie T, Tibshirani R (2008). "Sparse inverse covariance estimation with the graphical lasso." *Biostatistics*, **9**(3), 432–441. doi:10.1093/biostatistics/kxm045.

Lv J, Fan Y (2009). "A unified approach to model selection and sparse recovery using regularized least squares." *The Annals of Statistics*, **37**(6A), 3498–3528. doi:10.1214/09AOS683.

Mazumder R, Friedman JH, Hastie T (2011). "Sparsenet: Coordinate descent with nonconvex penalties." *Journal of the American Statistical Association*, **106**(495), 1125–1138. doi:10.1198/jasa.2011.tm09738.

Nehler KJ, Schultze M (2025). "Missing Data Handling via EM and Multiple Imputation in Network Analysis Using Glasso and Atan Regularization." *Multivariate Behavioral Research*, **60**(5), 990-1012. doi:10.1080/00273171.2025.2503833.

Schwarz G (1978). "Estimating the dimension of a model." *Annals of Statistics*, **6**(2), 461–464. doi:10.1214/aos/1176344136.

van Buuren S (2018). *Flexible Imputation of Missing Data*, 2 edition. CRC Press, Boca Raton. doi:10.1201/9780429492259.

Wang Y, Fan Q, Zhu L (2018). "Variable selection and estimation using a continuous approximation to the L0 penalty." *Annals of the Institute of Statistical Mathematics*, **70**(1), 191–214. doi:10.1007/s1046301605883.

Wang Y, Zhu L (2016). "Variable selection and parameter estimation with the Atan regularization method." *Journal of Probability and Statistics*. doi:10.1155/2016/6495417.

Williams D (2021). *GGMncv: Gaussian Graphical Models with Nonconvex Regularization*. R package version 2.1.1, commit 5be489dc3b20df1b16d1e50f20833ab4e43f94e8, `https://github.com/donaldRwilliams/GGMncv`.

Williams DR (2020). "Beyond Lasso: A Survey of Nonconvex Regularization in Gaussian Graphical Models." doi:10.31234/osf.io/ad57p. PsyArXiv Preprint.

Zhang C (2010). "Nearly unbiased variable selection under minimax concave penalty." *The Annals of Statistics*, **38**(2), 894–942. doi:10.1214/09AOS729.

Zou H, Li R (2008). "One-step sparse estimates in nonconcave penalized likelihood models." *Annals of Statistics*, **36**(4), 1509–1533. doi:10.1214/009053607000000802.

Examples

```
# Estimate regularized network from full data set
# Using observed-data loglikelihood and atan penalty
result <- regularization_net(mantar_dummy_full_cont,
                             likelihood = "obs_based",
                             penalty = "atan")

# View estimated partial correlation network
result$pcor

# Estimate regularized network from data set with missings
# Using correlation-matrix-based loglikelihood, glasso penalty,
# and stacked multiple imputation to handle missings
# set nimp to 10 for faster computation to in this example (not recommended
```

```
# in practice)
result <- regularization_net(mantar_dummy_mis_mix,
                             likelihood = "mat_based",
                             penalty = "glasso",
                             missing_handling = "stacked-mi",
                             nimp = 10,
                             ordered = c(FALSE,FALSE,TRUE,TRUE,
                                         FALSE,FALSE,TRUE,TRUE))

# View used correlation method and effective sample size
result$cor_method
result$n
# View estimated partial correlation network
result$pcor
```

Index