Package 'dineR'

June 1, 2026

Title Differential Network Estimation in R

Version 2.0.0

Description An efficient and convenient set of functions to perform differential network estimation through the use of alternating direction method of multipliers optimization with a variety of different loss functions.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

URL https://ricsalgado.github.io/dineR/, https://github.com/RicSalgado/dineR

BugReports https://github.com/RicSalgado/dineR/issues

Imports doSNOW, foreach, MASS, Matrix, parallel, progress

Suggests doParallel, knitr, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

VignetteBuilder knitr

NeedsCompilation no

Author Ricardo Daniel Marques Salgado [aut, cre] (ORCID: <https://orcid.org/0000-0002-3125-5159>), Mohammad Arashi [ctb], Andriette Bekker [ctb]

Maintainer Ricardo Daniel Marques Salgado <ricardodansalgado@gmail.com>

Repository CRAN

Date/Publication 2026-06-01 18:10:02 UTC

Contents

`data_generator` *Data Generating Function*

Description

This function generates two multivariate normal samples, by means of simulation. The samples can be represented as follows:

$$X \sim N_p(0, \Sigma_1)$$

$$Y \sim N_p(0, \Sigma_2)$$

with the only restriction being that each sample has the same number of features p.

Usage

```
data_generator(
  n_X = NULL,
  n_Y = NULL,
  p = NULL,
  Delta = NULL,
  case = "sparse",
  seed = NULL
)
```

Arguments

`n_X`	The number of observations to be generated for the first sample.
`n_Y`	The number of observations to be generated for the second sample. • Only one of the above sample sizes need to be specified. In such a scenario, the sample size for the other sample is set to the same as the specified sample size.
`p`	The dimensions/features for the samples.
`Delta`	Optional parameter - Provides the differential network from which the sample covariance matrices must be derived.
`case`	Optional parameter - Allows for the specification of the precision matrix structure. Possible cases are: "sparse" - Sparse Case or "asymsparse"- Asymptotically Sparse Case. Defaults to "sparse". • Sparse Case: $\Omega_1 = (0.5^{\lvert i-j \rvert})^{-1}$. That is, $\{\Omega_1\}_{1,1} = \{\Omega_1\}_{p,p} = \frac{4}{3}, \{\Omega_1\}_{i,i} = \frac{5}{3}$ for all other i. $\{\Omega_1\}_{i,i+1} = \{\Omega_1\}_{i-1,i} = \frac{2}{3}$ and $\{\Omega_1\}_{i,j} = 0$ for all other i, j. • Asymptotically Sparse Case
`seed`	Optional parameter - Allows a seed to be set for reproducibility.

Value

A list of the various outputs, namely:

- case - The case used.
- seed_option - The seed used for simulation.
- X - The first multivariate normal sample.
- Y - The second multivariate normal sample.
- n_X - The number of observations simulated for X.
- n_Y - The number of observations simulated for Y.
- Sigma_X - The covariance matrix of X: Σ_X.
- Sigma_Y - The covariance matrix of Y: Σ_Y.
- Omega_X - The precision matrix of X: $\Sigma_X^{-1} = \Omega_X$.
- Omega_Y - The precision matrix of Y: $\Sigma_Y^{-1} = \Omega_Y$.
- Diff_Omega - The difference of the precision matrices: $\Omega_X - \Omega_Y$.
- Delta - The target differential network: Δ.

References

Tang, Z., Yu, Z. and Wang, C., 2020. A fast iterative algorithm for high-dimensional differential network. Computational Statistics, 35(1), pp.95-109.

Examples

```
data <- data_generator(n_X = 100, p = 50, seed = 123)
data <- data_generator(n_X = 10, p = 50, case = "asymsparse")
```

`estimation` *Estimation*

Description

This function performs alternating direction method of multipliers optimization for a variety of loss functions to estimate the differential network given two samples of multivariate normal data.

Usage

```
estimation(
  X,
  Y,
  lambdas = NULL,
  lambda_min_ratio = 0.3,
  nlambda = 10,
  a = NULL,
```

```
  loss = "lasso",
  tuning = "none",
  perturb = FALSE,
  stop_tol = 1e-05,
  max_iter = 500,
  correlation = FALSE,
  Delta_init = NULL,
  rho = NULL,
  gamma = NULL,
  cores = 1,
  verbose = FALSE
)
```

Arguments

X	The first multivariate normal sample.
Y	The second multivariate normal sample.
lambdas	Optional parameter - A list of the regularization values to be used within the loss functions.
lambda_min_ratio	Optional parameter - Defines the smallest regularization values as this proportion of the largest regularization value. Defaults to 0.3.
nlambda	Optional parameter - The number of regularization values considered. Defaults to 10.
a	Optional parameter - The thresholding parameter used in SCAD and MCP loss functions. Defaults to 3.7 with SCAD, and 3 with MCP respectively.
loss	Optional parameter - The loss function of choice to implement. The function allows for four choices, namely "lasso", "scad", "mcp" and "d-trace". Defaults to "lasso".
tuning	Optional parameter - The tuning method selected to determine the optimal value for the regularization parameter. Options are "none", "AIC", "BIC" and "EBIC". Defaults to "none".
perturb	Optional parameter - When set to TRUE perturbation as done by the CLIME software to improve performance is implemented. Options are TRUE or FALSE, with the function defaulting to FALSE.
stop_tol	Optional parameter - The stop tolerance to determine whether convergence has occurred. Defaults to 1e-5.
max_iter	Optional parameter - The maximum number of iterations that can be perform for any one regularization value. Defaults to 100.
correlation	Optional parameter - Determines whether the sample correlation matrices should be used in the place of the sample covariance matrices. Choices are TRUE and FALSE with the function defaulting to FALSE.
Delta_init	Optional parameter - Allows for the algorithm to provided an initial estimate of the differential network to ease computation.
rho	Optional parameter - Allows the user to adjust the ADMM step-size. Defaults to 1.

gamma	Optional parameter - Allows the user to adjust the EBIC value when EBIC is the selected tuning method. Defaults to 0.5.
cores	Optional parameter - Allows the user to specify the number of cores used by the optimization. Defaults to 1, i.e sequential solving however appropriate values range from 2 to the minimum of the number of lambdas and the total number of available cores.
verbose	Optional parameter - Allows the user to obtain a summary of the estimation results. Options are TRUE or FALSE, where FALSE indicates the summary is not provided. Defaults to FALSE.

Value

A list of various outputs, namely:

- n_X - The number of observations in X.
- n_Y - The number of observations in Y.
- Sigma_X - The covariance matrix of X.
- Sigma_Y - The covariance matrix of Y.
- loss - The loss function implemented.
- tuning - The tuning method utilized.
- lip - The value of the lipschitz constant.
- iter - The iterations until convergence for each of the regularization values.
- elapse - The total system time (in seconds) elapsed from initialization to completion of the optimization.
- lambdas - The regularization parameter values used.
- sparsity - The level of sparsity of the differential network for each regularization value.
- path - The set of all differential networks for all regularization values considered.
- ic - The output obtained from any possible tuning.
- ic_index - The index at which the tuning is optimized.
- ic_value - The tuning method optimal value.
- chosen_lambda_ic - The regularization value that occurs at **ic_index**.
- loss_index - The index at which the loss function is optimized.
- loss_value - The loss function optimal value.
- chosen_lambda_loss - The regularization value that occurs at **loss_index**.

References

Boyd, S., Parikh, N., Chu, E., Peleato, B. and Eckstein, J., 2011. Distributed optimization and statistical learning via the alternating direction method of multipliers. Foundations and Trends® in Machine learning, 3(1), pp.1-122.

Chen, J. and Chen, Z., 2008. Extended Bayesian information criteria for model selection with large model spaces. Biometrika, 95(3), pp.759-771.

Friedman, J., Hastie, T. and Tibshirani, R., 2008. Sparse inverse covariance estimation with the graphical lasso. Biostatistics, 9(3), pp.432-441.

Tang, Z., Yu, Z. and Wang, C., 2020. A fast iterative algorithm for high-dimensional differential network. Computational Statistics, 35(1), pp.95-109.

Yuan, H., Xi, R., Chen, C. and Deng, M., 2017. Differential network analysis via lasso penalized D-trace loss. Biometrika, 104(4), pp.755-770.

Zhang, T. and Zou, H., 2014. Sparse precision matrix estimation via lasso penalized D-trace loss. Biometrika, 101(1), pp.103-120.

Examples

```
data <- data_generator(n_X = 100, p = 50, seed = 123)
X <- data$X
Y <- data$Y

# Sequential (default)
result_seq <- estimation(X, Y, nlambda = 5, cores = 1)
result_seq$elapse


# Parallel - set cores to a value greater than 1
# (detectCores() - 1 is recommended on your own machine)
result_par <- estimation(X, Y, nlambda = 5, cores = 2)
result_par$elapse
```

---

npn *NPN - Non paranormal Transformation*

Description

This functions allows us to transform non-normal multivariate data to that of non paranormal data.

Usage

```
npn(x, npn_func = "shrinkage", npn_thresh = NULL, verbose = TRUE)
```

Arguments

x	The multivariate non-normal data to be transformed.
npn_func	Optional parameter - The method of transformation to be applied. Can either be "shrinkage" or "truncation" but defaults to "shrinkage".
npn_thresh	Optional parameter - The truncation threshold that is used when making use of truncation.
verbose	Optional parameter - Prints additional output of the selected approach. Can either be "TRUE" or "FALSE" and defaults to "TRUE".

Value

Returns the transformed data matrix.

References

Liu, H., Han, F., Yuan, M., Lafferty, J. and Wasserman, L., 2012. The nonparanormal skeptic. arXiv preprint arXiv:1206.6488.

Liu, H., Lafferty, J. and Wasserman, L., 2009. The nonparanormal: Semiparametric estimation of high dimensional undirected graphs. Journal of Machine Learning Research, 10(10).

Xue, L. and Zou, H., 2012. Regularized rank-based estimation of high-dimensional nonparanormal graphical models. The Annals of Statistics, 40(5), pp.2541-2571.

Examples

```
data <- data_generator(n_X = 100, p = 50, seed = 123)
X <- data$X
X_transformed <- npn(X, npn_func = "truncation")
```

Index