Package ‘GGMncv’

November 15, 2020

Type Package

Title Gaussian Graphical Models with Nonconvex Regularization

Version 2.0.0

Date 2020-11-15

Description

Estimate Gaussian graphical models with nonconvex penalties <doi:10.31234/osf.io/ad57p>,
including the atan Wang and Zhu (2016) <doi:10.1155/2016/6495417>,
seamless L0 Dicker, Huang, and Lin (2013) <doi:10.5705/ss.2011.074>,
exponential Wang, Fan, and Zhu <doi:10.1007/s10463-016-0588-3>,
smooth integration of counting and absolute deviation Lv and Fan (2009) <doi:10.1214/09-AOS683>,
logarithm Mazumder, Friedman, and Hastie (2011) <doi:10.1198/jasa.2011.tm09738>,
Lq, smoothly clipped absolute deviation Fan and Li (2001) <doi:10.1198/016214501753382273>,
and minimax concave penalty Zhang (2010) <doi:10.1214/09-AOS729>. There are also extensions
for computing variable inclusion probabilities, multiple regression coefficients, and
statistical inference <doi:10.1214/15-EJS1031>.

License GPL-2

Depends R (>= 4.0.0)

Imports Rcpp (>= 1.0.4.6),
Rdpack (>= 0.11-1),
reshape,
ggplot2 (>= 3.3.0),
glassoFast (>= 1.0),
numDeriv (>= 2016.8-1.1),
mathjaxr (>= 1.0-1),
MASS (>= 7.3-51.5),
methods,
stats,
utils

Suggests qgraph (>= 1.6.5)

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

LinkingTo Rcpp,
RcppArmadillo

RdMacros Rdpack,
mathjaxr

BugReports https://github.com/donaldRwilliams/GGMncv/issues

R topics documented:

`boot_eip` *Bootstrapped Edge Inclusion 'Probabilities'*

Description

Compute edge inclusion 'probabilities' with a non-parametric bootstrap.

Usage

```
boot_eip(
  Y,
  method = "pearson",
  samples = 50,
  penalty = "atan",
  ic = "bic",
  gamma = NULL,
  lambda = NULL,
  n_lambda = 50,
  n_gamma = 50,
  unreg = FALSE,
  progress = TRUE,
  ...
)
```

Arguments

- `Y` Matrix. A matrix of dimensions n by p.
- `method` Character string. Which correlation coefficient (or covariance) is to be computed. One of "pearson" (default), "kendall", or "spearman." Defaults to `pearson`.
- `samples` Numeric. How many boostrap samples (defaults to `500`).
- `penalty` Character string. Which penalty should be used (defaults to `"atan"`)?
- `ic` Character string. Which information criterion should be used (defaults to `"bic"`)? The options include `aic`, `ebic` (ebic_gamma defaults to `0.5`; see details), `ric`, or any of the generalized information criteria provided in section 5 of Kim et al. (2012). The options are `gic_1` (i.e., `bic`) to `gic_6`.
- `gamma` Numeric. Hyperparameter for the penalty function. Defaults to 3.7 (`SCAD`), 2 (`MCP`), 0.5 (`adapt`), and 0.01 otherwise with `select = "lambda"`.
- `lambda` Numeric vector. Regularization parameter. Defaults to `NULL` that provides default values with `select = "lambda"` and `sqrt(log(p)/n)` with `select = "gamma"`.
- `n_lambda` Numeric. The number of λ's to be evaluated. Defaults to 50. This is disregarded if custom values are provided in `lambda`.
- `n_gamma` Numeric. The number of γ's to be evaluated. Defaults to 50. This is disregarded if custom values are provided in `lambda`.
- `unreg` Logical. Should the models be refitted (or unregularized) with maximum likelihood (defaults to `FALSE`)? Setting to `TRUE` results in the approach of Foygel and Drton (2010), but with the regularization path obtained from nonconvex regularization, as opposed to the ℓ_1-penalty.
- `progress` Logical. Should a progress bar be included (defaults to `TRUE`) ?
- `...` Additional arguments. Currently gamma in EBIC (`ic = "ebic"`) can be set with `ebic_gamma = 1`.

Value

An object of class `eip`

Examples

```
# data
Y <- GGMncv::ptsd[,1:10]

# compute eip's
boot_samps <- boot_eip(Y, samples  = 10)
```

`coef.ggmncv` *Regression Coefficients from* ggmncv *Objects*

Description

Regression Coefficients from ggmncv Objects

Usage

```
## S3 method for class 'ggmncv'
coef(object, ...)
```

Arguments

`object` An Object of classs ggmncv

`...` Currently ignored

Value

A matrix of regression coefficients

Note

The matrix of coefficients can be accessed by removing the class from the returned object (e.g., `unclass(coefs)`).

Examples

```
# data
Y <- GGMncv::ptsd

# correlations
S <- cor(Y)

# fit model
fit <- ggmncv(S, n = nrow(Y))

coefs <- coef(fit)
```

`constrained` *Constrained Precision Matrix*

Description

Compute the maximum likelihood estimate, given certain elements are constrained to zero (e.g., an adjacency matrix). This approach is described in Hastie et al. (2015).

Usage

```
constrained(Sigma, adj)
```

Arguments

`Sigma` Covariance matrix

`adj` Matrix with contraints. A zero indicates that element should be constrained to zero.

Value

A list containing the inverse covariance matrix and the covariance matrix.

Note

The algorithm is written in c++.

Examples

```
# data
Y <- GGMncv::ptsd[,1:5]

# columns
p <- ncol(Y)

# contstraint matrix
constraints <- matrix(0,p,p)

# set one value to zero
constraints[2,3] <- 1
constraints[3,2] <-1

# estimate, given constraints
fit <- constrained(cor(Y), adj = constraints)
Theta <- fit$Theta
```

---

desparsify *De-Sparsified Graphical Lasso Estimator*

Description

Compute the de-sparsified glasso estimator with the approach described in Equation 7 of Jankova and Van De Geer (2015).

Usage

```
desparsify(object, ...)
```

Arguments

object	An object of class ggmncv
...	Currently ignored

Details

According to Jankova and Van De Geer (2015), the de-sparisifed estimator, $\hat{\mathbf{T}}$, is defined as

$\hat{\mathbf{T}} = 2\hat{\mathbf{\Theta}} - \hat{\mathbf{\Theta}}\hat{\mathbf{R}}\hat{\mathbf{\Theta}},$

where $\hat{\mathbf{\Theta}}$ denotes the graphical lasso estimator of the precision matrix and $\hat{\mathbf{R}}$ is the sample correlation matrix. Further details can be found in section 2 ("Main Results") of Jankova and Van De Geer (2015).

Value

The de-sparsified estimates, including

- `Theta`: De-sparsified precision matrix
- `P`: De-sparsified partial correlation matrix

Note

This assumes (reasonably) Gaussian data.

References

Jankova J, Van De Geer S (2015). "Confidence intervals for high-dimensional inverse covariance estimation." *Electronic Journal of Statistics*, **9**(1), 1205–1229.

Examples

```
# data
Y <- GGMncv::Sachs

# fit model
fit <- ggmncv(cor(Y), n = nrow(Y))

# remove (some) bias and sparsity
That <- desparsify(fit)

# de-sparsified partial correlations
That$P
```

---

ggmncv *GGMncv*

Description

Gaussian graphical models with nonconvex regularization. A survey of these approaches is provided in Williams (2020).

Usage

```
ggmncv(
  R,
  n,
  penalty = "atan",
  ic = "bic",
  select = "lambda",
  gamma = NULL,
  lambda = NULL,
  n_lambda = 50,
  n_gamma = 50,
  initial = NULL,
  LLA = FALSE,
  unreg = FALSE,
```

```
  maxit = 10000,
  thr = 1e-04,
  store = TRUE,
  progress = TRUE,
  ...
)
```

Arguments

`R`	Matrix. A correlation matrix of dimensions *p* by *p*.
`n`	Numeric. The sample size used to compute the information criterion.
`penalty`	Character string. Which penalty should be used (defaults to `"atan"`)?
`ic`	Character string. Which information criterion should be used (defaults to `"bic"`)? The options include `aic`, `ebic` (ebic_gamma defaults to `0.5`; see details), `ric`, or any of the generalized information criteria provided in section 5 of Kim et al. (2012). The options are `gic_1` (i.e., `bic`) to `gic_6`.
`select`	Character string. Which tuning parameter should be selected (defaults to `"lambda"`)?. The options include `"lambda"` (the regularization parameter), `"gamma"` (governs the 'shape'), and `"both"`. See details.
`gamma`	Numeric vector. Hyperparameter for the penalty function. Defaults to 3.7 (`SCAD`), 2 (`MCP`), 0.5 (`adapt`), and 0.01 otherwise with `select = "lambda"`.
`lambda`	Numeric vector. Regularization parameter. Defaults to `NULL` that provides default values with `select = "lambda"` and `sqrt(log(p)/n)` with `select = "gamma"`.
`n_lambda`	Numeric. The number of λ's to be evaluated. Defaults to 50. This is disregarded if custom values are provided in `lambda`.
`n_gamma`	Numeric. The number of γ's to be evaluated. Defaults to 50. This is disregarded if custom values are provided in `lambda`.
`initial`	Matrix. The initial inverse correlation matrix for computing the penalty derivative. Defaults to `NULL` which uses the inverse of `R`.
`LLA`	Logical. Should the local linear approximation be used (default to `FALSE`)?
`unreg`	Logical. Should the models be refitted (or unregularized) with maximum likelihood (defaults to `FALSE`)? Setting to `TRUE` results in the approach of Foygel and Drton (2010), but with the regularization path obtained from nonconvex regularization, as opposed to the ℓ_1-penalty.
`maxit`	Numeric. The maximum number of iterations for determining convergence of the LLA algorithm (defaults to `1e4`). Note this can be changed to, say, `2` or `3`, which will provide two and three-step estimators without convergence check.
`thr`	Numeric. Threshold for determining convergence of the LLA algorithm (defaults to `1.0e-4`).
`store`	Logical. Should all of the fitted models be saved (defaults to `TRUE`)?.
`progress`	Logical. Should a progress bar be included (defaults to `TRUE`)?
`...`	Additional arguments. Currently gamma in EBIC (`ic = "ebic"`) can be set with `ebic_gamma = 1`.

Details

Several of the penalties are (continuous) approximations to the ℓ_0 penalty, that is, best subset selection. However, the solution does not require enumerating all possible models which results in a computationally efficient solution.

L0 Approximations

- Atan: `penalty = "atan"` (Wang and Zhu 2016). This is currently the default.
- Seamless ℓ_0: `penalty = "selo"` (Dicker et al. 2013).
- Exponential: `penalty = "exp"` (Wang et al. 2018)
- Log: `penalty = "log"` (Mazumder et al. 2011).
- Sica: `penalty = "sica"` (Lv and Fan 2009)

Additional penalties:

- SCAD: `penalty = "scad"` (Fan and Li 2001).
- MCP: `penalty = "mcp"` (Zhang 2010).
- Adaptive lasso (`penalty = "adapt"`): Defaults to $\gamma = 0.5$ (Zou 2006). Note that for consistency with the other penalties, $\gamma \rightarrow 0$ provides more penalization and $\gamma = 1$ results in ℓ_1 regularization.
- Lasso: `penalty = "lasso"` (Tibshirani 1996).

Gamma (γ):

The `gamma` argument corresponds to additional hyperparameter for each penalty. The defaults are set to the recommended values from the respective papers.

LLA

The local linear approximate is noncovex penalties was described in (Fan et al. 2009). This is essentially a iteratively reweighted (g)lasso. Note that by default `LLA = FALSE`. This is due to the work of Zou and Li (2008), which suggested that, so long as the starting values are good enough, then a one-step estimator is sufficient. In the case of low-dimensional data, the sample based inverse covariance matrix is used to compute the penalty. This is expected to work well, assuming that n is sufficiently larger than p.

EBIC

When setting `ic = "ebic"` the hyperparameter that determines the additional penalty to BIC is passed via the `...` argument. This must be specified as `ebic_gamma = 1`. The default is `0.5`.

Value

An object of class `ggmncv`, including:

- `Theta` Inverse covariance matrix
- `Sigma` Covariance matrix
- `P` Weighted adjacency matrix
- `adj` Adjacency matrix
- `lambda` Tuning parameter
- `fit` glasso fitted model (a list)

References

Dicker L, Huang B, Lin X (2013). "Variable selection and estimation with the seamless-L 0 penalty." *Statistica Sinica*, 929–962.

Fan J, Feng Y, Wu Y (2009). "Network exploration via the adaptive LASSO and SCAD penalties." *The annals of applied statistics*, **3**(2), 521.

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.

Foygel R, Drton M (2010). "Extended Bayesian Information Criteria for Gaussian Graphical Models." *Advances in Neural Information Processing Systems*, 604–612. 1011.6640.

Kim Y, Kwon S, Choi H (2012). "Consistent model selection criteria on high dimensions." *The Journal of Machine Learning Research*, **13**, 1037–1057.

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.

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

Tibshirani R (1996). "Regression shrinkage and selection via the lasso." *Journal of the Royal Statistical Society: Series B (Methodological)*, **58**(1), 267–288.

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.

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

Williams DR (2020). "Beyond Lasso: A Survey of Nonconvex Regularization in Gaussian Graphical Models." *PsyArXiv*.

Zhang C (2010). "Nearly unbiased variable selection under minimax concave penalty." *The Annals of statistics*, **38**(2), 894–942.

Zou H (2006). "The adaptive lasso and its oracle properties." *Journal of the American statistical association*, **101**(476), 1418–1429.

Zou H, Li R (2008). "One-step sparse estimates in nonconcave penalized likelihood models." *Annals of statistics*, **36**(4), 1509.

Examples

```
# data
Y <- GGMncv::ptsd[,1:10]

S <- cor(Y)

# fit model
fit <- ggmncv(S, n = nrow(Y))
```

```
# plot
qgraph::qgraph(fit$P)
```

`ggm_compare` *Compare Gaussian Graphical Models*

Description

Compare Gaussian graphical models with the de-sparsified estimator of (Jankova and Van De Geer 2015).

Usage

```
ggm_compare(object_1, object_2, method = "fdr", alpha = 0.05, ...)
```

Arguments

`object_1`	An object of class ggmncv
`object_2`	An object of class ggmncv
`method`	Character string. A correction method for multiple comparison (defaults to `fdr`). Can be abbreviated. See p.adjust.
`alpha`	Numeric. Significance level (defaults to `0.05`).
`...`	Currently ignored.

Value

- `P_diff` De-sparsified partial correlation matrix differences
- `adj` Adjacency matrix based on the p-values.
- `uncorrected` Uncorrected p-values
- `corrected` Corected p-values
- `method` The approach used for multiple comparisons
- `alpha` Significance level

Examples

```
# data
Y1 <- MASS::mvrnorm(250, rep(0, 10), Sigma = diag(10))
Y2 <- MASS::mvrnorm(250, rep(0, 10), Sigma = diag(10))

# fit models
fit1 <- ggmncv(cor(Y1), n = nrow(Y1))
fit2 <- ggmncv(cor(Y2), n = nrow(Y2))

# compare
compare_ggms <- ggm_compare(fit1, fit2)
```

`inference` *Statistical Inference for Gaussian Graphical Models*

Description

Compute p-values for each relation based on the de-sparsified precision matrix (Jankova and Van De Geer 2015).

Usage

```
inference(object, method = "fdr", alpha = 0.05, ...)
```

Arguments

`object`	An object of class ggmncv
`method`	Character string. A correction method for multiple comparison (defaults to `fdr`). Can be abbreviated. See p.adjust.
`alpha`	Numeric. Significance level (defaults to `0.05`).
`...`	Currently ignored.

Value

- `Theta` De-sparsified precision matrix
- `adj` Adjacency matrix based on the p-values.
- `uncorrected` Uncorrected p-values
- `corrected` Corected p-values
- `method` The approach used for multiple comparisons
- `alpha` Significance level

Note

This assumes the Gaussian data.

References

Jankova J, Van De Geer S (2015). "Confidence intervals for high-dimensional inverse covariance estimation." *Electronic Journal of Statistics*, **9**(1), 1205–1229.

Examples

```
# data
Y <- GGMncv::ptsd

# fit model
fit <- ggmncv(cor(Y), n = nrow(Y))


# statistical inference
inference(fit)
```

penalty_derivative *Penalty Derivative*

Description

Compute the derivative for a nonconvex penalty.

Usage

```
penalty_derivative(
  theta = seq(-5, 5, length.out = 1e+05),
  penalty = "atan",
  lambda = 1,
  gamma = c(0.01, 0.05)
)
```

Arguments

theta	Numeric vector. Values for which the derivative is computed.
penalty	Character string. Character string. Which penalty should be used (defaults to "atan")? See ggmncv for the available penalties.
lambda	Numeric. Regularization parameter (defaults to 1).
gamma	Numeric vector. Hyperparameter(s) for the penalty function.

Value

A list of class penalty_derivative

Examples

```
deriv <- penalty_derivative(theta =  seq(-5,5,length.out = 10000),
                            lambda = 1,
                            gamma = c(0.01, 0.05, 0.1))
```

penalty_function *Penalty Function*

Description

Compute the penalty function for nonconvex penalties.

Usage

```
penalty_function(
  theta = seq(-5, 5, length.out = 1e+05),
  penalty = "atan",
  lambda = 1,
  gamma = c(0.01, 0.05)
)
```

Arguments

theta	Numeric vector. Values for which the derivative is computed.
penalty	Character string. Character string. Which penalty should be used (defaults to "atan")? See ggmncv for the available penalties.
lambda	Numeric. Regularization parameter (defaults to 1).
gamma	Numeric vector. Hyperparameter(s) for the penalty function.

Value

An object of class `penalty_function`

Examples

```
func <- penalty_function(theta =  seq(-5,5,length.out = 10000),
                         lambda = 1,
                         gamma = c(0.01, 0.05, 0.1))
```

---

plot.eip — *Plot Edge Inclusion 'Probabilities'*

Description

Plot Edge Inclusion 'Probabilities'

Usage

```
## S3 method for class 'eip'
plot(x, color = "black", size = 1, ...)
```

Arguments

x	An object of class `eip`
color	Character string. Color for `geom_point`.
size	Numeric. Size of `geom_point`.
...	Currently ignored.

Value

An object of class `ggplot`

Examples

```
# data
Y <- GGMncv::ptsd[,1:10]

# compute eip's
boot_samps <- boot_eip(Y, B = 10)


plot(boot_samps)
```

---

plot.ggmncv *Plot* ggmncv *Objects*

Description

Plot the solution path for the partial correlations.

Usage

```
## S3 method for class 'ggmncv'
plot(x, size = 1, alpha = 0.5, ...)
```

Arguments

x	An object of class ggmncv
size	Numeric. The size of the points (eip).
alpha	Numeric. The transparency of the lines. Only for the solution path options.
...	Currently ignored.

Value

A ggplot object

Examples

```
# data
Y <- GGMncv::ptsd[,1:10]

# correlations
S <- cor(Y, method = "spearman")

# fit model
fit <- ggmncv(R = S, n = nrow(Y))

# plot
plot(fit)
```

---

plot.penalty_derivative
Plot penalty_derivative *Objects*

Description

Plot penalty_derivative Objects

Usage

```
## S3 method for class 'penalty_derivative'
plot(x, size = 1, ...)
```

Arguments

`x`	An object of class `penalty_derivative`.
`size`	Numeric. Line size in `geom_line`.
`...`	Currently ignored.

Value

An object of class `ggplot`

Examples

```
pen_deriv <- penalty_derivative(theta =  seq(-5,5,length.out = 10000),
                                lambda = 1,
                                gamma = c(0.01, 0.05, 0.1))
plot(pen_deriv)
```

`plot.penalty_function` *Plot* `penalty_function` *Objects*

Description

Plot `penalty_function` Objects

Usage

```
## S3 method for class 'penalty_function'
plot(x, size = 1, ...)
```

Arguments

`x`	An object of class `penalty_function`.
`size`	Numeric. Line size in `geom_line`.
`...`	Currently ignored.

Value

An object of class `ggplot`

Examples

```
func <- penalty_function(theta =  seq(-5,5,length.out = 10000),
                                lambda = 1,
                                gamma = c(0.01, 0.05, 0.1))
plot(func)
```

predict.ggmncv *Predict method for* ggmncv *Objects*

Description

Predicted values based on a ggmncv object

Usage

```
## S3 method for class 'ggmncv'
predict(object, train_data, newdata = NULL, ...)
```

Arguments

object	An object of class ggmncv
train_data	Data used for model fitting.
newdata	An optional data frame in which to look for variables with which to predict. If omitted, the fitted values are used.
...	Currently ignored

Value

A matrix of predicted values

Examples

```
# data
Y <- scale(Sachs)

# test data
Ytest <- Y[1:100,]

# training data
Ytrain <- Y[101:nrow(Y),]

fit <- ggmncv(cor(Ytrain), n = nrow(Ytrain))

pred <- predict(fit, train_data = Y,
                newdata = Ytest)

round(apply((pred - Ytest)^2, 2, mean), 2)
```

print.eip — *Print* eip *Objects*

Description

Print eip Objects

Usage

```
## S3 method for class 'eip'
print(x, ...)
```

Arguments

x	An object of class eip
...	Currently ignored.

print.ggmncv — *Print* ggmncv *Objects*

Description

Print ggmncv Objects

Usage

```
## S3 method for class 'ggmncv'
print(x, ...)
```

Arguments

x	An object of class ggmncv
...	Currently ignored

ptsd — *Data: Post-Traumatic Stress Disorder*

Description

A dataset containing items that measure Post-traumatic stress disorder symptoms (Armour et al. 2017). There are 20 variables (p) and 221 observations (n).

Usage

```
data("ptsd")
```

Format

A dataframe with 221 rows and 20 variables

Details

- Intrusive Thoughts
- Nightmares
- Flashbacks
- Emotional cue reactivity
- Psychological cue reactivity
- Avoidance of thoughts
- Avoidance of reminders
- Trauma-related amnesia
- Negative beliefs
- Negative trauma-related emotions
- Loss of interest
- Detachment
- Restricted affect
- Irritability/anger
- Self-destructive/reckless behavior
- Hypervigilance
- Exaggerated startle response
- Difficulty concentrating
- Sleep disturbance

References

Armour C, Fried EI, Deserno MK, Tsai J, Pietrzak RH (2017). "A network analysis of DSM-5 posttraumatic stress disorder symptoms and correlates in US military veterans." *Journal of anxiety disorders*, **45**, 49–59.

`Sachs` *Data: Sachs Network*

Description

Protein expression in human immune system cells

Usage

```
data("Sachs")
```

Format

A data frame containing 7466 cells (n = 7466) and flow cytometry measurements of 11 (p = 11) phosphorylated proteins and phospholipids (Sachs et al. 2002)

References

Sachs K, Gifford D, Jaakkola T, Sorger P, Lauffenburger DA (2002). "Bayesian network approach to cell signaling pathway modeling." *Science's STKE*, **2002**(148), pe38–pe38.

Examples

```
data("Sachs")
```

Index