Package ‘matrixCorr’

August 26, 2025

Type Package

Title Collection of Correlation and Association Estimators

Version 0.3.1

Description Compute correlation and other association matrices from small to very large datasets with simple, 'cor()'-like functions and sensible defaults. Includes options for shrinkage and robustness to improve results in noisy or high-dimensional settings (p >= n), plus convenient print/plot methods for inspection. Implemented with optimised 'C++' backends using 'BLAS'/'OpenMP' and memory-aware symmetric updates. Works with base matrices and data frames, returning standard 'R' objects via a consistent S3 interface. Useful across genomics, agriculture, and machine-learning workflows. Methods based on Ledoit and Wolf (2004) <doi:10.1016/S0047-259X(03)00096-4>; Schäfer and Strimmer (2005) <doi:10.2202/1544-6115.1175>; Lin (1989) <doi:10.2307/2532051>.

License MIT + file LICENSE

Encoding UTF-8

LinkingTo cpp11, Rcpp, RcppArmadillo

Imports Rcpp, ggplot2 (>= 3.5.2)

Suggests knitr, rmarkdown, testthat, MASS, Matrix, viridisLite

RoxygenNote 7.3.2

URL `https://github.com/Prof-ThiagoOliveira/matrixCorr`

BugReports `https://github.com/Prof-ThiagoOliveira/matrixCorr/issues`

NeedsCompilation yes

Author Thiago de Paula Oliveira [aut, cre] (ORCID: <`https://orcid.org/0000-0002-4555-2584`>)

Maintainer Thiago de Paula Oliveira <`toliveira@abacusbio.com`>

Repository CRAN

Date/Publication 2025-08-26 14:20:12 UTC

Contents

`ccc` *Pairwise Lin's concordance correlation coefficient*

Description

Computes all pairwise Lin's Concordance Correlation Coefficients (CCC) from the numeric columns of a matrix or data frame. CCC measures both precision (Pearson correlation) and accuracy (closeness to the 45-degree line). This function is backed by a high-performance 'C++' implementation.

Usage

```
ccc(data, ci = FALSE, conf_level = 0.95, verbose = FALSE)

## S3 method for class 'ccc'
print(x, digits = 4, ...)

## S3 method for class 'ccc'
plot(
  x,
  title = "Lin's Concordance Correlation Heatmap",
  low_color = "indianred1",
  high_color = "steelblue1",
  mid_color = "white",
  value_text_size = 4,
  ...
)
```

Arguments

`data`	A numeric matrix or data frame with at least two numeric columns. Non-numeric columns will be ignored.
`ci`	Logical; if TRUE, return lower and upper confidence bounds
`conf_level`	Confidence level for CI, default = 0.95
`verbose`	Logical; if TRUE, prints how many threads are used

`x`	An object of class `"ccc"` (either a matrix or a list with confidence intervals).
`digits`	Integer; number of decimal places to print in the concordance matrix (default is 4).
`...`	Additional arguments passed to underlying functions (like `theme` or `print`).
`title`	Title for the plot.
`low_color`	Color for low CCC values.
`high_color`	Color for high CCC values.
`mid_color`	Color for mid CCC values (typically near 0).
`value_text_size`	Text size for numbers in the heatmap.

Details

Lin's CCC is defined as:

$$\rho_c = \frac{2 \cdot \mathrm{cov}(X,Y)}{\sigma_X^2 + \sigma_Y^2 + (\mu_X - \mu_Y)^2}$$

This formula combines the Pearson correlation coefficient $r = \mathrm{cov}(X,Y)/(\sigma_X \sigma_Y)$ with a bias correction factor:

$$C_b = \frac{2\sigma_X \sigma_Y}{\sigma_X^2 + \sigma_Y^2 + (\mu_X - \mu_Y)^2}$$

Confidence intervals are not provided in the matrix version to retain speed, but can be computed separately for individual variable pairs using the scalar form of Lin's CCC.

Missing values are not allowed; input must be clean numeric data.

Value

A symmetric numeric matrix with class `"ccc"` and attributes:

- `method`: The method used ("Lin's concordance")
- `description`: Description string

If `ci = FALSE`, returns matrix of class `"ccc"`. If `ci = TRUE`, returns a list with elements: `est`, `lwr.ci`, `upr.ci`.

Author(s)

Thiago de Paula Oliveira

References

Lin L (1989). A concordance correlation coefficient to evaluate reproducibility. Biometrics 45: 255-268.

Lin L (2000). A note on the concordance correlation coefficient. Biometrics 56: 324-325.

Bland J, Altman D (1986). Statistical methods for assessing agreement between two methods of clinical measurement. The Lancet 327: 307-310.

See Also

print.ccc, plot.ccc

Examples

```
# Example with multivariate normal data
Sigma <- matrix(c(1, 0.5, 0.3,
                  0.5, 1, 0.4,
                  0.3, 0.4, 1), nrow = 3)
mu <- c(0, 0, 0)
set.seed(123)
mat_mvn <- MASS::mvrnorm(n = 100, mu = mu, Sigma = Sigma)
result_mvn <- ccc(mat_mvn)
print(result_mvn)
plot(result_mvn)
```

`distance_corr` *Pairwise Distance Correlation (dCor)*

Description

Computes all pairwise *distance correlations* using the unbiased U-statistic estimator for the numeric columns of a matrix or data frame, via a high-performance 'C++' backend ('OpenMP'-parallelised). Distance correlation detects general (including non-linear and non-monotonic) dependence between variables; unlike Pearson or Spearman, it is zero (in population) if and only if the variables are independent.

Prints a summary of the distance correlation matrix with optional truncation for large objects.

Generates a ggplot2 heatmap of the distance correlation matrix. Distance correlation is non-negative; the fill scale spans `[0, 1]`.

Usage

```
distance_corr(data)

## S3 method for class 'distance_corr'
print(x, digits = 4, max_rows = NULL, max_cols = NULL, ...)

## S3 method for class 'distance_corr'
plot(
  x,
  title = "Distance correlation heatmap",
  low_color = "white",
  high_color = "steelblue1",
  value_text_size = 4,
  ...
)
```

Arguments

`data`	A numeric matrix or a data frame with at least two numeric columns. All non-numeric columns are dropped. Columns must be numeric and contain no `NA`s.
`x`	An object of class `distance_corr`.
`digits`	Integer; number of decimal places to print.
`max_rows`	Optional integer; maximum number of rows to display. If `NULL`, all rows are shown.
`max_cols`	Optional integer; maximum number of columns to display. If `NULL`, all columns are shown.
`...`	Additional arguments passed to `ggplot2::theme()` or other `ggplot2` layers.
`title`	Plot title. Default is `"Distance correlation heatmap"`.
`low_color`	Colour for zero correlation. Default is `"white"`.
`high_color`	Colour for strong correlation. Default is `"steelblue1"`.
`value_text_size`	Font size for displaying values. Default is `4`.

Details

Let $x \in \mathbb{R}^n$ and $D^{(x)}$ be the pairwise distance matrix with zero diagonal: $D^{(x)}_{ii} = 0$, $D^{(x)}_{ij} = |x_i - x_j|$ for $i \neq j$. Define row sums $r^{(x)}_i = \sum_{k \neq i} D^{(x)}_{ik}$ and grand sum $S^{(x)} = \sum_{i \neq k} D^{(x)}_{ik}$. The U-centred matrix is

$$A^{(x)}_{ij} = \begin{cases} D^{(x)}_{ij} - \dfrac{r^{(x)}_i + r^{(x)}_j}{n-2} + \dfrac{S^{(x)}}{(n-1)(n-2)}, & i \neq j, \\ 0, & i = j\ . \end{cases}$$

For two variables x, y, the unbiased distance covariance and variances are

$$\widehat{\mathrm{dCov}}^2_u(x,y) = \frac{2}{n(n-3)} \sum_{i<j} A^{(x)}_{ij} A^{(y)}_{ij} \ = \ \frac{1}{n(n-3)} \sum_{i \neq j} A^{(x)}_{ij} A^{(y)}_{ij},$$

with $\widehat{\mathrm{dVar}}^2_u(x)$ defined analogously from $A^{(x)}$. The unbiased distance correlation is

$$\widehat{\mathrm{dCor}}_u(x,y) = \frac{\widehat{\mathrm{dCov}}_u(x,y)}{\sqrt{\widehat{\mathrm{dVar}}_u(x)\,\widehat{\mathrm{dVar}}_u(y)}} \in [0,1].$$

Value

A symmetric numeric matrix where the `(i, j)` entry is the unbiased distance correlation between the `i`-th and `j`-th numeric columns. The object has class `distance_corr` with attributes `method = "distance_correlation"`, `description`, and `package = "matrixCorr"`.

Invisibly returns `x`.

A `ggplot` object representing the heatmap.

Note

Requires $n \geq 4$. Columns with (near) zero unbiased distance variance yield `NA` in their row/column. Computation is $O(n^2)$ per pair.

References

Székely, G. J., Rizzo, M. L., & Bakirov, N. K. (2007). Measuring and testing dependence by correlation of distances. *Annals of Statistics*, 35(6), 2769–2794.

Székely, G. J., & Rizzo, M. L. (2013). The distance correlation t-test of independence. *Journal of Multivariate Analysis*, 117, 193-213.

Examples

```
##Independent variables -> dCor ~ 0
set.seed(1)
X <- cbind(a = rnorm(200), b = rnorm(200))
D <- distance_corr(X)
print(D, digits = 3)

## Non-linear dependence: Pearson ~ 0, but unbiased dCor > 0
set.seed(42)
n <- 200
x <- rnorm(n)
y <- x^2 + rnorm(n, sd = 0.2)
XY <- cbind(x = x, y = y)
D2 <- distance_corr(XY)
# Compare Pearson vs unbiased distance correlation
round(c(pearson = cor(XY)[1, 2], dcor = D2["x", "y"]), 3)
plot(D2, title = "Unbiased distance correlation (non-linear example)")

## Small AR(1) multivariate normal example
set.seed(7)
p <- 5; n <- 150; rho <- 0.6
Sigma <- rho^abs(outer(seq_len(p), seq_len(p), "-"))
X3 <- MASS::mvrnorm(n, mu = rep(0, p), Sigma = Sigma)
colnames(X3) <- paste0("V", seq_len(p))
D3 <- distance_corr(X3)
print(D3[1:3, 1:3], digits = 2)
```

`kendall_tau` *Pairwise Kendall's Tau Rank Correlation*

Description

Computes all pairwise Kendall's tau rank correlation coefficients for the numeric columns of a matrix or data frame using a high-performance 'C++'.

This function uses a fast and scalable algorithm implemented in 'C++' to compute both Kendall's tau-a (when no ties are present) and tau-b (when ties are detected), making it suitable for large datasets. Internally, it calls a highly optimized function that uses a combination of merge-sort-based inversion counting and a Fenwick Tree (binary indexed tree) for efficient tie handling.

Prints a summary of the Kendall's tau correlation matrix, including description and method metadata.

Generates a ggplot2-based heatmap of the Kendall's tau correlation matrix.

Usage

```
kendall_tau(data)

## S3 method for class 'kendall_matrix'
print(x, digits = 4, max_rows = NULL, max_cols = NULL, ...)

## S3 method for class 'kendall_matrix'
plot(
  x,
  title = "Kendall's Tau correlation heatmap",
  low_color = "indianred1",
  high_color = "steelblue1",
  mid_color = "white",
  value_text_size = 4,
  ...
)
```

Arguments

`data`	A numeric matrix or a data frame with at least two numeric columns. All non-numeric columns will be excluded. Each column must have at least two non-missing values and contain no NAs.
`x`	An object of class `kendall_matrix`.
`digits`	Integer; number of decimal places to print
`max_rows`	Optional integer; maximum number of rows to display. If `NULL`, all rows are shown.
`max_cols`	Optional integer; maximum number of columns to display. If `NULL`, all columns are shown.
`...`	Additional arguments passed to `ggplot2::theme()` or other `ggplot2` layers.
`title`	Plot title. Default is `"Kendall's Tau Correlation Heatmap"`.
`low_color`	Color for the minimum tau value. Default is `"indianred1"`.
`high_color`	Color for the maximum tau value. Default is `"steelblue1"`.
`mid_color`	Color for zero correlation. Default is `"white"`.
`value_text_size`	Font size for displaying correlation values. Default is `4`.

Details

Kendall's tau is a rank-based measure of association between two variables. For a dataset with n observations of two variables X and Y, Kendall's tau coefficient is defined as:

$$\tau = \frac{C - D}{\sqrt{(C + D + T_x)(C + D + T_y)}}$$

where:

- C is the number of concordant pairs defined by $(x_i - x_j)(y_i - y_j) > 0$
- D is the number of discordant pairs defined by $(x_i - x_j)(y_i - y_j) < 0$
- T_x, T_y are the number of tied pairs in X and Y, respectively

When there are no ties, the function computes the faster *tau-a* version:

$$\tau_a = \frac{C - D}{n(n-1)/2}$$

The function automatically selects *tau-a* or *tau-b* depending on the presence of ties. Performance is maximized by computing correlations in 'C++' directly from the matrix columns.

Value

A symmetric numeric matrix where the `(i, j)`-th element is the Kendall's tau correlation between the `i`-th and `j`-th numeric columns of the input.

Invisibly returns the `kendall_matrix` object.

A `ggplot` object representing the heatmap.

Note

Missing values are not allowed. Columns with fewer than two observations are excluded.

Author(s)

Thiago de Paula Oliveira <`toliveira@abacusbio.com`>

Thiago de Paula Oliveira

References

Kendall, M. G. (1938). A New Measure of Rank Correlation. Biometrika, 30(1/2), 81–93.

See Also

`print.kendall_matrix`, `print.kendall_matrix`

Examples

```
# Basic usage with a matrix
mat <- cbind(a = rnorm(100), b = rnorm(100), c = rnorm(100))
kt <- kendall_tau(mat)
print(kt)
plot(kt)

# With a large data frame
df <- data.frame(x = rnorm(1e4), y = rnorm(1e4), z = rnorm(1e4))
kendall_tau(df)

# Including ties
tied_df <- data.frame(
  v1 = rep(1:5, each = 20),
  v2 = rep(5:1, each = 20),
  v3 = rnorm(100)
)
kt <- kendall_tau(tied_df)
print(kt)
plot(kt)
```

`partial_correlation` *Partial correlation matrix (sample / ridge / OAS)*

Description

Computes the Gaussian partial correlation matrix from a numeric data frame or matrix. The covariance matrix can be estimated using:

- **Unbiased sample covariance**: the standard empirical covariance estimator.
- **Ridge-regularised covariance**: adds a positive ridge penalty to improve stability when the covariance matrix is near-singular.
- **OAS shrinkage to a scaled identity**: recommended when $p \gg n$, as it reduces estimation error by shrinking towards a scaled identity matrix.

The method uses a high-performance 'C++' backend.

Prints only the partial correlation matrix (no attribute spam), with an optional one-line header stating the estimator used.

Produces a **ggplot2**-based heatmap of the partial correlation matrix stored in `x$pcor`. Optionally masks the diagonal and/or reorders variables via hierarchical clustering of $1 - |pcor|$.

Usage

```
partial_correlation(
  data,
  method = c("oas", "ridge", "sample"),
  lambda = 0.001,
  return_cov_precision = FALSE
)

## S3 method for class 'partial_correlation'
print(x, digits = 3, show_method = TRUE, max_rows = NULL, max_cols = NULL, ...)

## S3 method for class 'partial_corr'
plot(
  x,
  title = NULL,
  low_color = "indianred1",
  high_color = "steelblue1",
  mid_color = "white",
  value_text_size = 4,
  mask_diag = TRUE,
  reorder = FALSE,
  ...
)
```

Arguments

`data`	A numeric matrix or data frame with at least two numeric columns. Non-numeric columns are ignored.
`method`	Character; one of `"oas"`, `"ridge"`, `"sample"`. Default `"oas"`.
`lambda`	Numeric ≥ 0; ridge penalty added to the covariance diagonal when `method = "ridge"`. Ignored otherwise. Default `1e-3`.
`return_cov_precision`	Logical; if `TRUE`, also return the covariance (`cov`) and precision (`precision`) matrices used to form the partial correlations. Default to `FALSE`
`x`	An object of class `partial_corr`.
`digits`	Integer; number of decimal places for display (default 3).
`show_method`	Logical; print a one-line header with `method` (and `lambda`/`rho` if available). Default `TRUE`.
`max_rows, max_cols`	Optional integer limits for display; if provided, the printed matrix is truncated with a note about omitted rows/cols.
`...`	Additional arguments passed to `ggplot2::theme()` or other **ggplot2** layers.
`title`	Plot title. By default, constructed from the estimator in `x$method`.
`low_color`	Colour for low (negative) values. Default `"indianred1"`.
`high_color`	Colour for high (positive) values. Default `"steelblue1"`.
`mid_color`	Colour for zero. Default `"white"`.
`value_text_size`	Font size for cell labels. Default `4`.
`mask_diag`	Logical; if `TRUE`, the diagonal is masked (set to `NA`) and not labelled. Default `TRUE`.

`reorder`	Logical; if `TRUE`, variables are reordered by hierarchical clustering of $1 - \|pcor\|$. Default `FALSE`.

Details

Statistical overview. Given an $n \times p$ data matrix X (rows are observations, columns are variables), the routine estimates a *partial correlation* matrix via the precision (inverse covariance) matrix. Let μ be the vector of column means and

$$S = (X - \mathbf{1}\mu)^\top (X - \mathbf{1}\mu)$$

be the centred cross-product matrix computed without forming a centred copy of X. Two conventional covariance scalings are formed:

$$\hat{\Sigma}_{\mathrm{MLE}} = S/n, \qquad \hat{\Sigma}_{\mathrm{unb}} = S/(n-1).$$

- *Sample:* $\Sigma = \hat{\Sigma}_{\mathrm{unb}}$.
- *Ridge:* $\Sigma = \hat{\Sigma}_{\mathrm{unb}} + \lambda I_p$ with user-supplied $\lambda \geq 0$ (diagonal inflation).
- *OAS (Oracle Approximating Shrinkage):* shrink $\hat{\Sigma}_{\mathrm{MLE}}$ towards a scaled identity target $\mu_I I_p$, where $\mu_I = \mathrm{tr}(\hat{\Sigma}_{\mathrm{MLE}})/p$. The data-driven weight $\rho \in [0,1]$ is

  $$\rho = \min\left\{1, \max\left(0,\ \frac{(1-\frac{2}{p})\,\mathrm{tr}(\hat{\Sigma}^2_{\mathrm{MLE}}) + \mathrm{tr}(\hat{\Sigma}_{\mathrm{MLE}})^2}{(n+1-\frac{2}{p})\left[\mathrm{tr}(\hat{\Sigma}^2_{\mathrm{MLE}}) - \frac{\mathrm{tr}(\hat{\Sigma}_{\mathrm{MLE}})^2}{p}\right]}\right)\right\},$$

  and

  $$\Sigma = (1-\rho)\,\hat{\Sigma}_{\mathrm{MLE}} + \rho\,\mu_I I_p.$$

The method then ensures positive definiteness of Σ (adding a very small diagonal *jitter* only if necessary) and computes the precision matrix $\Theta = \Sigma^{-1}$. Partial correlations are obtained by standardising the off-diagonals of Θ:

$$\mathrm{pcor}_{ij} \;=\; -\frac{\theta_{ij}}{\sqrt{\theta_{ii}\,\theta_{jj}}}, \qquad \mathrm{pcor}_{ii} = 1.$$

Interpretation. For Gaussian data, pcor_{ij} equals the correlation between residuals from regressing variable i and variable j on all the remaining variables; equivalently, it encodes conditional dependence in a Gaussian graphical model, where $\mathrm{pcor}_{ij} = 0$ if variables i and j are conditionally independent given the others. Partial correlations are invariant to separate rescalings of each variable; in particular, multiplying Σ by any positive scalar leaves the partial correlations unchanged.

Why shrinkage/regularisation? When $p \geq n$, the sample covariance is singular and inversion is ill-posed. Ridge and OAS both yield well-conditioned Σ. Ridge adds a fixed λ on the diagonal, whereas OAS shrinks adaptively towards $\mu_I I_p$ with a weight chosen to minimise (approximately) the Frobenius risk under a Gaussian model, often improving mean–square accuracy in high dimension.

Computational notes. The implementation forms S using 'BLAS' `syrk` when available and constructs partial correlations by traversing only the upper triangle with 'OpenMP' parallelism. Positive definiteness is verified via a Cholesky factorisation; if it fails, a tiny diagonal jitter is increased geometrically up to a small cap, at which point the routine signals an error.

Value

An object of class `"partial_corr"` (a list) with elements:

- `pcor`: $p \times p$ partial correlation matrix.
- `cov` (if requested): covariance matrix used.
- `precision` (if requested): precision matrix Θ.
- `method`: the estimator used (`"oas"`, `"ridge"`, or `"sample"`).
- `lambda`: ridge penalty (or `NA_real_`).
- `rho`: OAS shrinkage weight in $[0, 1]$ (or `NA_real_`).
- `jitter`: diagonal jitter added (if any) to ensure positive definiteness.

Invisibly returns `x`.

A `ggplot` object.

References

Chen, Y., Wiesel, A., & Hero, A. O. III (2011). Robust Shrinkage Estimation of High-dimensional Covariance Matrices. IEEE Transactions on Signal Processing.

Ledoit, O., & Wolf, M. (2004). A well-conditioned estimator for large-dimensional covariance matrices. Journal of Multivariate Analysis, 88(2), 365–411.

Schafer, J., & Strimmer, K. (2005). A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics. Statistical Applications in Genetics and Molecular Biology, 4(1), Article 32.

Examples

```
## Structured MVN with known partial correlations
set.seed(42)
p <- 12; n <- 1000

## Build a tri-diagonal precision (Omega) so the true partial correlations
## are sparse
phi <- 0.35
Omega <- diag(p)
for (j in 1:(p - 1)) {
  Omega[j, j + 1] <- Omega[j + 1, j] <- -phi
}
## Strict diagonal dominance
diag(Omega) <- 1 + 2 * abs(phi) + 0.05
Sigma <- solve(Omega)

## Upper Cholesky
L <- chol(Sigma)
Z <- matrix(rnorm(n * p), n, p)
X <- Z %*% L
colnames(X) <- sprintf("V%02d", seq_len(p))

pc <- partial_correlation(X, method = "oas")
```

```
## True partial correlation from Omega
pcor_true <- -Omega / sqrt(diag(Omega) %o% diag(Omega))
diag(pcor_true) <- 1

## Quick visual check (first 5x5 block)
round(pc$pcor[1:5, 1:5], 2)
round(pcor_true[1:5, 1:5], 2)

## Plot method
plot(pc)

## High-dimensional case p >> n
set.seed(7)
n <- 60; p <- 120

ar_block <- function(m, rho = 0.6) rho^abs(outer(seq_len(m), seq_len(m), "-"))

## Two AR(1) blocks on the diagonal
if (requireNamespace("Matrix", quietly = TRUE)) {
  Sigma_hd <- as.matrix(Matrix::bdiag(ar_block(60, 0.6), ar_block(60, 0.6)))
} else {
  Sigma_hd <- rbind(
    cbind(ar_block(60, 0.6), matrix(0, 60, 60)),
    cbind(matrix(0, 60, 60), ar_block(60, 0.6))
  )
}

L <- chol(Sigma_hd)
X_hd <- matrix(rnorm(n * p), n, p) %*% L
colnames(X_hd) <- paste0("G", seq_len(p))

pc_oas   <-
 partial_correlation(X_hd, method = "oas",   return_cov_precision = TRUE)
pc_ridge <-
 partial_correlation(X_hd, method = "ridge", lambda = 1e-2,
                     return_cov_precision = TRUE)
pc_samp  <-
 partial_correlation(X_hd, method = "sample", return_cov_precision = TRUE)

## Show how much diagonal regularisation was used
c(oas_jitter = pc_oas$jitter,
  ridge_lambda = pc_ridge$lambda,
  sample_jitter = pc_samp$jitter)

## Compare conditioning of the estimated covariance matrices
c(kappa_oas = kappa(pc_oas$cov),
  kappa_ridge = kappa(pc_ridge$cov),
  kappa_sample = kappa(pc_samp$cov))

## Simple conditional-dependence graph from partial correlations
pcor <- pc_oas$pcor
vals <- abs(pcor[upper.tri(pcor, diag = FALSE)])
```

```
thresh <- quantile(vals, 0.98)  # top 2%
edges  <- which(abs(pcor) > thresh & upper.tri(pcor), arr.ind = TRUE)
head(data.frame(i = colnames(pcor)[edges[,1]],
                j = colnames(pcor)[edges[,2]],
                pcor = round(pcor[edges], 2)))
```

`partial_correlation_cpp` *Partial correlation matrix with sample / ridge / OAS covariance*

Description

Partial correlation matrix with sample / ridge / OAS covariance

Usage

```
partial_correlation_cpp(
  X_,
  method = "oas",
  lambda = 0.001,
  return_cov_precision = TRUE
)
```

Arguments

`X_`	Numeric double matrix (n x p). No NAs.
`method`	One of "sample", "ridge", "oas". Default "oas" (recommended for p » n).
`lambda`	Ridge penalty for "ridge" method (added to diagonal). Ignored otherwise.
`return_cov_precision`	If TRUE, return covariance and precision matrices.

Value

A list with elements: `pcor`, and optionally `cov`, `precision`, `method`, `lambda`, `rho` (for OAS).

`pearson_corr` *Pairwise Pearson correlation*

Description

Computes all pairwise Pearson correlation coefficients for the numeric columns of a matrix or data frame using a high-performance 'C++' backend.

This function uses a direct Pearson formula implementation in 'C++' to achieve fast and scalable correlation computations, especially for large datasets.

Prints a summary of the Pearson correlation matrix, including description and method metadata.

Generates a ggplot2-based heatmap of the Pearson correlation matrix.

Usage

```
pearson_corr(data)

## S3 method for class 'pearson_corr'
print(x, digits = 4, max_rows = NULL, max_cols = NULL, ...)

## S3 method for class 'pearson_corr'
plot(
  x,
  title = "Pearson correlation heatmap",
  low_color = "indianred1",
  high_color = "steelblue1",
  mid_color = "white",
  value_text_size = 4,
  ...
)
```

Arguments

`data`	A numeric matrix or a data frame with at least two numeric columns. All non-numeric columns will be excluded. Each column must have at least two non-missing values and contain no NAs.
`x`	An object of class `pearson_corr`.
`digits`	Integer; number of decimal places to print in the concordance
`max_rows`	Optional integer; maximum number of rows to display. If `NULL`, all rows are shown.
`max_cols`	Optional integer; maximum number of columns to display. If `NULL`, all columns are shown.
`...`	Additional arguments passed to `ggplot2::theme()` or other `ggplot2` layers.
`title`	Plot title. Default is `"Pearson correlation heatmap"`.
`low_color`	Color for the minimum correlation. Default is `"indianred1"`.

`high_color`	Color for the maximum correlation. Default is `"steelblue1"`.
`mid_color`	Color for zero correlation. Default is `"white"`.
`value_text_size`	Font size for displaying correlation values. Default is `4`.

Details

Statistical formulation. Let $X \in \mathbb{R}^{n\times p}$ be a numeric matrix with rows as observations and columns as variables, and let $\mu = \frac{1}{n}\mathbf{1}^\top X$ be the vector of column means. Define the centred cross-product matrix

$$S \;=\; (X-\mathbf{1}\mu)^\top(X-\mathbf{1}\mu) \;=\; X^\top X \;-\; n\,\mu\,\mu^\top.$$

The (unbiased) sample covariance is then

$$\widehat{\Sigma} \;=\; \tfrac{1}{n-1}\,S,$$

and the vector of sample standard deviations is

$$s_i \;=\; \sqrt{\widehat{\Sigma}_{ii}}, \qquad i=1,\ldots,p.$$

The Pearson correlation matrix R is obtained by standardising $\widehat{\Sigma}$, given by:

$$R \;=\; D^{-1/2}\,\widehat{\Sigma}\,D^{-1/2}, \qquad D \;=\; \mathrm{diag}(\widehat{\Sigma}_{11},\ldots,\widehat{\Sigma}_{pp}),$$

equivalently, entrywise

$$R_{ij} \;=\; \frac{\widehat{\Sigma}_{ij}}{s_i\,s_j}, \quad i\neq j, \qquad R_{ii} \;=\; 1.$$

If $s_i = 0$ (zero variance), the i-th row and column are set to `NA`. Tiny negative values on the covariance diagonal caused by floating-point rounding are reduced to zero before taking square roots. No missing values are permitted in X.

The implementation forms $X^\top X$ via a symmetric rank-k update ('BLAS' 'SYRK') on the upper triangle, then applies the rank-1 correction $-\,n\,\mu\,\mu^\top$ to obtain S without explicitly materialising $X - \mathbf{1}\mu$. After scaling by $1/(n-1)$, triangular normalisation by $D^{-1/2}$ yields R, which is then symmetrised. The dominant cost is $O(np^2)$ flops with $O(p^2)$ memory.

This implementation avoids calculating means explicitly and instead uses a numerically stable form.

Value

A symmetric numeric matrix where the `(i, j)`-th element is the Pearson correlation between the `i`-th and `j`-th numeric columns of the input.

Invisibly returns the `pearson_corr` object.

A `ggplot` object representing the heatmap.

Note

Missing values are not allowed. Columns with fewer than two observations are excluded.

Author(s)

Thiago de Paula Oliveira <toliveira@abacusbio.com>

Thiago de Paula Oliveira

References

Pearson, K. (1895). "Notes on regression and inheritance in the case of two parents". Proceedings of the Royal Society of London, 58, 240–242.

See Also

print.pearson_corr, plot.pearson_corr

Examples

```
## MVN with AR(1) correlation
set.seed(123)
p <- 6; n <- 300; rho <- 0.5
# true correlation
Sigma <- rho^abs(outer(seq_len(p), seq_len(p), "-"))
L <- chol(Sigma)
# MVN(n, 0, Sigma)
X <- matrix(rnorm(n * p), n, p) %*% L
colnames(X) <- paste0("V", seq_len(p))

pr <- pearson_corr(X)
print(pr, digits = 2)
plot(pr)

## Compare the sample estimate to the truth
Rhat <- cor(X)
# estimated
round(Rhat[1:4, 1:4], 2)
# true
round(Sigma[1:4, 1:4], 2)
off <- upper.tri(Sigma, diag = FALSE)
# MAE on off-diagonals
mean(abs(Rhat[off] - Sigma[off]))

## Larger n reduces sampling error
n2 <- 2000
X2 <- matrix(rnorm(n2 * p), n2, p) %*% L
Rhat2 <- cor(X2)
off <- upper.tri(Sigma, diag = FALSE)
## mean absolute error (MAE) of the off-diagonal correlations
mean(abs(Rhat2[off] - Sigma[off]))
```

`schafer_corr` *Schafer-Strimmer shrinkage correlation*

Description

Computes a shrinkage correlation matrix using the Schafer-Strimmer approach with an analytic, data-driven intensity $\hat{\lambda}$. The off-diagonals of the sample Pearson correlation R are shrunk towards zero, yielding $R_{\mathrm{shr}} = (1 - \hat{\lambda})R + \hat{\lambda}I$ with $\mathrm{diag}(R_{\mathrm{shr}}) = 1$, stabilising estimates when $p \geq n$.

This function uses a high-performance 'C++' backend that forms $X^\top X$ via 'BLAS' 'SYRK', applies centring via a rank-1 update, converts to Pearson correlation, estimates $\hat{\lambda}$, and shrinks the off-diagonals: $R_{\mathrm{shr}} = (1 - \hat{\lambda})R + \hat{\lambda}I$.

Prints a summary of the shrinkage correlation matrix with optional truncation for large objects.

Heatmap of the shrinkage correlation matrix with optional hierarchical clustering and triangular display. Uses **ggplot2** and `geom_raster()` for speed on larger matrices.

Usage

```
schafer_corr(data)

## S3 method for class 'schafer_corr'
print(x, digits = 4, max_rows = NULL, max_cols = NULL, ...)

## S3 method for class 'schafer_corr'
plot(
  x,
  title = "Schafer-Strimmer shrinkage correlation",
  cluster = TRUE,
  hclust_method = "complete",
  triangle = "upper",
  show_values = FALSE,
  value_text_limit = 60,
  value_text_size = 3,
  palette = c("diverging", "viridis"),
  ...
)
```

Arguments

`data`	A numeric matrix or a data frame with at least two numeric columns. All non-numeric columns will be excluded. Columns must be numeric and contain no `NA`s.
`x`	An object of class `schafer_corr`.
`digits`	Integer; number of decimal places to print.
`max_rows`	Optional integer; maximum number of rows to display. If `NULL`, all rows are shown.

`max_cols`	Optional integer; maximum number of columns to display. If `NULL`, all columns are shown.
`...`	Additional arguments passed to `ggplot2::theme()`.
`title`	Plot title.
`cluster`	Logical; if TRUE, reorder rows/cols by hierarchical clustering on distance $1 - r$.
`hclust_method`	Linkage method for `hclust`; default `"complete"`.
`triangle`	One of `"full"`, `"upper"`, `"lower"`. Default to `upper`.
`show_values`	Logical; print correlation values inside tiles (only if matrix dimension $\leq$ `value_text_limit`).
`value_text_limit`	Integer threshold controlling when values are drawn.
`value_text_size`	Font size for values if shown.
`palette`	Character; `"diverging"` (default) or `"viridis"`.

Details

Let R be the sample Pearson correlation matrix. The Schafer-Strimmer shrinkage estimator targets the identity in correlation space and uses $\hat{\lambda} = \frac{\sum_{i<j} \widehat{\mathrm{Var}}(r_{ij})}{\sum_{i<j} r_{ij}^2}$ (clamped to $[0, 1]$), where $\widehat{\mathrm{Var}}(r_{ij}) \approx \frac{(1-r_{ij}^2)^2}{n-1}$. The returned estimator is $R_{\mathrm{shr}} = (1 - \hat{\lambda})R + \hat{\lambda}I$.

Value

A symmetric numeric matrix of class `schafer_corr` where entry `(i, j)` is the shrunk correlation between the `i`-th and `j`-th numeric columns. Attributes:

- `method` = `"schafer_shrinkage"`
- `description` = `"Schafer-Strimmer shrinkage correlation matrix"`
- `package` = `"matrixCorr"`

Columns with zero variance are set to `NA` across row/column (including the diagonal), matching `pearson_corr()` behaviour.

Invisibly returns `x`.

A `ggplot` object.

Note

No missing values are permitted. Columns with fewer than two observations or zero variance are flagged as `NA` (row/column).

Author(s)

Thiago de Paula Oliveira `<toliveira@abacusbio.com>`

References

Schafer, J. & Strimmer, K. (2005). A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics. *Statistical Applications in Genetics and Molecular Biology*, 4(1).

See Also

`print.schafer_corr`, `plot.schafer_corr`, `pearson_corr`

Examples

```
## Multivariate normal with AR(1) dependence (Toeplitz correlation)
set.seed(1)
n <- 80; p <- 40; rho <- 0.6
d <- abs(outer(seq_len(p), seq_len(p), "-"))
Sigma <- rho^d

X <- MASS::mvrnorm(n, mu = rep(0, p), Sigma = Sigma)
colnames(X) <- paste0("V", seq_len(p))

Rshr <- schafer_corr(X)
print(Rshr, digits = 2, max_rows = 6, max_cols = 6)
plot(Rshr)

## Shrinkage typically moves the sample correlation closer to the truth
Raw <- stats::cor(X)
off  <- upper.tri(Sigma, diag = FALSE)
mae_raw <- mean(abs(Rraw[off] - Sigma[off]))
mae_shr <- mean(abs(Rshr[off] - Sigma[off]))
print(c(MAE_raw = mae_raw, MAE_shrunk = mae_shr))
plot(Rshr, title = "Schafer-Strimmer shrinkage correlation")
```

`spearman_rho` *Pairwise Spearman's rank correlation*

Description

Computes all pairwise Spearman's rank correlation coefficients for the numeric columns of a matrix or data frame using a high-performance 'C++' backend.

This function ranks the data and computes Pearson correlation on ranks, which is equivalent to Spearman's rho. It supports large datasets and is optimized in 'C++' for performance.

Prints a summary of the Spearman's correlation matrix, including description and method metadata.

Generates a ggplot2-based heatmap of the Spearman's rank correlation matrix.

Usage

```
spearman_rho(data)

## S3 method for class 'spearman_rho'
print(x, digits = 4, max_rows = NULL, max_cols = NULL, ...)

## S3 method for class 'spearman_rho'
plot(
  x,
  title = "Spearman's rank correlation heatmap",
  low_color = "indianred1",
  high_color = "steelblue1",
  mid_color = "white",
  value_text_size = 4,
  ...
)
```

Arguments

`data`	A numeric matrix or a data frame with at least two numeric columns. All non-numeric columns will be excluded. Each column must have at least two non-missing values and contain no NAs.
`x`	An object of class `spearman_rho`.
`digits`	Integer; number of decimal places to print.
`max_rows`	Optional integer; maximum number of rows to display. If `NULL`, all rows are shown.
`max_cols`	Optional integer; maximum number of columns to display. If `NULL`, all columns are shown.
`...`	Additional arguments passed to `ggplot2::theme()` or other `ggplot2` layers.
`title`	Plot title. Default is `"Spearman's rank correlation heatmap"`.
`low_color`	Color for the minimum rho value. Default is `"indianred1"`.
`high_color`	Color for the maximum rho value. Default is `"steelblue1"`.
`mid_color`	Color for zero correlation. Default is `"white"`.
`value_text_size`	Font size for displaying correlation values. Default is `4`.

Details

For each column $j = 1, \dots, p$, let $R_{\cdot j} \in \{1, \dots, n\}^n$ denote the (mid-)ranks of $X_{\cdot j}$, assigning average ranks to ties. Write $\bar{R}_j = (n+1)/2$ for the mean rank and define the centred rank vectors $\tilde{R}_{\cdot j} = R_{\cdot j} - \bar{R}_j \mathbf{1}$. The Spearman correlation between columns i and j is the Pearson correlation of their rank vectors, given by

$$\rho_S(i,j) \;=\; \frac{\sum_{k=1}^n \big(R_{ki} - \bar{R}_i\big)\big(R_{kj} - \bar{R}_j\big)}{\sqrt{\sum_{k=1}^n \big(R_{ki} - \bar{R}_i\big)^2}\,\sqrt{\sum_{k=1}^n \big(R_{kj} - \bar{R}_j\big)^2}}.$$

In matrix form, with $R = [R_{\cdot 1}, \ldots, R_{\cdot p}]$, $\mu = (n+1)\mathbf{1}_p/2$ and $S_R = \big(R - \mathbf{1}\mu^\top\big)^\top \big(R - \mathbf{1}\mu^\top\big)/(n-1)$, the Spearman correlation matrix is

$$\widehat{\rho}_S \;=\; D^{-1/2} S_R D^{-1/2}, \qquad D \;=\; \operatorname{diag}(S_R).$$

When there are no ties, the familiar rank-difference formula obtains

$$\rho_S(i,j) \;=\; 1 - \frac{6}{n(n^2-1)} \sum_{k=1}^{n} d_k^2, \quad d_k \;=\; R_{ki} - R_{kj},$$

but this expression does *not* hold under ties; computing Pearson correlation on mid-ranks (as above) is the standard tie-robust approach.

$\rho_S(i,j) \in [-1,1]$ and the matrix $\widehat{\rho}_S$ is symmetric positive semi-definite. Spearman's correlation is invariant to any strictly monotone transformation applied separately to each variable, and to common translations and positive rescalings prior to ranking.

The implementation ranks each column to form R, then evaluates $R^\top R$ using a symmetric rank update ('BLAS' 'SYRK') and centres it via the identity

$$(R - \mathbf{1}\mu^\top)^\top (R - \mathbf{1}\mu^\top) \;=\; R^\top R \;-\; n\,\mu\mu^\top,$$

avoiding an explicit centred copy. Division by $n-1$ yields the sample covariance of ranks, which is finally standardised by $D^{-1/2}$ to obtain $\widehat{\rho}_S$. Columns with zero rank variance (all values equal) are returned as `NA` along their row/column, with the diagonal set to `NA`.

Ranking costs $O\big(p\,n\log n\big)$; forming and normalising $R^\top R$ costs $O\big(np^2\big)$ with $O(p^2)$ additional memory. 'OpenMP' parallelism is used across columns for ranking, and a 'BLAS' 'SYRK' kernel is used for the matrix product when available.

Value

A symmetric numeric matrix where the `(i, j)`-th element is the Spearman correlation between the `i`-th and `j`-th numeric columns of the input.

Invisibly returns the `spearman_rho` object.

A `ggplot` object representing the heatmap.

Note

Missing values are not allowed. Columns with fewer than two observations are excluded.

Author(s)

Thiago de Paula Oliveira `<toliveira@abacusbio.com>`

References

Spearman, C. (1904). The proof and measurement of association between two things. International Journal of Epidemiology, 39(5), 1137-1150.

See Also

`print.spearman_rho`, `plot.spearman_rho`

Examples

```
## Monotone transformation invariance (Spearman is rank-based)
set.seed(123)
n <- 400; p <- 6; rho <- 0.6
# AR(1) correlation
Sigma <- rho^abs(outer(seq_len(p), seq_len(p), "-"))
L <- chol(Sigma)
X <- matrix(rnorm(n * p), n, p) %*% L
colnames(X) <- paste0("V", seq_len(p))

# Monotone transforms to some columns
X_mono <- X
# exponential
X_mono[, 1] <- exp(X_mono[, 1])
# softplus
X_mono[, 2] <- log1p(exp(X_mono[, 2]))
# odd monotone polynomial
X_mono[, 3] <- X_mono[, 3]^3

sp_X <- spearman_rho(X)
sp_m <- spearman_rho(X_mono)

# Spearman should be (nearly) unchanged under monotone transformations
round(max(abs(sp_X - sp_m)), 3)
# heatmap of Spearman correlations
plot(sp_X)

## Ties handled via mid-ranks
tied <- cbind(
  # many ties
  a = rep(1:5, each = 20),
  # noisy reverse order
  b = rep(5:1, each = 20) + rnorm(100, sd = 0.1),
  # ordinal with ties
  c = as.numeric(gl(10, 10))
)
sp_tied <- spearman_rho(tied)
print(sp_tied, digits = 2)

## Bivariate normal, theoretical Spearman's rho
## For BVN with Pearson correlation r, rho_S = (6/pi) * asin(r/2).
r_target <- c(-0.8, -0.4, 0, 0.4, 0.8)
n2 <- 200
est <- true_corr <- numeric(length(r_target))
for (i in seq_along(r_target)) {
  R2 <- matrix(c(1, r_target[i], r_target[i], 1), 2, 2)
  Z  <- matrix(rnorm(n2 * 2), n2, 2) %*% chol(R2)
  s  <- spearman_rho(Z)
  est[i]  <- s[1, 2]
  true_corr[i] <- (6 / pi) * asin(r_target[i] / 2)
}
cbind(r_target, est = round(est, 3), theory = round(true_corr, 3))
```

Index