Package 'nda'

November 6, 2026

Type Package

Title Generalized Network-Based Dimensionality Reduction and Analysis

Version 0.3.0

Maintainer Zsolt T. Kosztyan <kosztyan.zsolt@gtk.uni-pannon.hu>

Description Constructs interpretable latent variables from communities in squared-association networks. Supports full, partial, and semipartial associations, feature selection, sparse networks, inference, prediction, visualization, and supervised regression. The generalized method is described in Kosztyan et al. (2024) <doi:10.1016/j.eswa.2023.121779>.

License GPL (>= 2)

Encoding UTF-8

LazyData true

LazyDataCompression xz

URL https://github.com/kzst/nda

BugReports https://github.com/kzst/nda/issues

Depends R (>= 4.0.0)

Imports energy, psych, stats, graphics, grDevices, grid, igraph, Matrix, MASS, mco, parallel, ppcor, visNetwork

Suggests glmnet, knitr, lm.beta, plotly, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

RoxygenNote 7.2.3

NeedsCompilation no

Author Zsolt T. Kosztyan [aut, cre],
Marcell T. Kurbucz [aut],
Attila I. Katona [aut],
Zahid Khan [aut]

Repository CRAN

Date/Publication 2026-11-06 17:40:08 UTC

Contents

nda-package . . . 2
biplot.nda . . . 3
COVID19_2020 . . . 4
CrimesUSA1990.X . . . 4
CrimesUSA1990.Y . . . 5
CWTS_2020 . . . 5
data_gen . . . 6
dCor . . . 7
dCov . . . 7
fitted.ndrlm . . . 8
fs.dimred . . . 9
fs.KMO . . . 9
GOVDB2020 . . . 10
I40_2020 . . . 11
ndr . . . 11
ndrlm . . . 14
normalize . . . 17
pdCor . . . 17
plot.nda . . . 18
plot.ndrlm . . . 20
predict.nda . . . 22
predict.ndrlm . . . 22
print.nda . . . 23
print.ndrlm . . . 24
residuals.ndrlm . . . 24
spdCor . . . 25
summary.nda . . . 26
summary.ndrlm . . . 26

Index **28**

`nda-package` *Package of Generalized Network-based Dimensionality Reduction and Analyses*

Description

The package of Generalized Network-based Dimensionality Reduction and Analysis (GNDA).

Author(s)

Zsolt T. Kosztyan*, Marcell T. Kurbucz, Attila I. Katona, Zahid Khan

e-mail*: kosztyan.zsolt@gtk.uni-pannon.hu

References

Kosztyan, Z. T., Kurbucz, M. T., & Katona, A. I. (2022). Network-based dimensionality reduction of high-dimensional, low-sample-size datasets. Knowledge-Based Systems, 109180.

Kosztyan, Z. T., Katona, A. I., Kurbucz, M. T., & Lantos, Z. (2024). Generalized network-based dimensionality analysis. Expert Systems with Applications, 238, 121779. <URL: https://doi.org/10.1016/j.eswa.2023.121779>

See Also

ndr, ndrlm, plot, biplot, summary, dCor.

`biplot.nda` *Biplot NDA Scores and Loadings*

Description

Creates a printable two-dimensional biplot or a sequence of three-factor projections.

Usage

```
## S3 method for class 'nda'
biplot(x, main = NULL, dims = c(1, 2), view_3D = FALSE,
    triples = NULL, interactive = FALSE, angle = 45,
    point_cex = 0.7, label_cex = 0.9, ...)
```

Arguments

`x`	An `nda` object fitted to row-level data.
`main`	Optional title.
`dims`	Two factors for the two-dimensional view.
`view_3D`	Display three-factor projections.
`triples`	Optional list of factor triples. By default all triples are used.
`interactive`	Use **plotly** for rotatable 3D views.
`angle`	Static axonometric projection angle in degrees.
`point_cex, label_cex`	Observation and loading-label size.
`...`	Additional two-dimensional base-graphics arguments.

Value

An `nda_biplot` object; printing it renders the selected view or views.

Examples

```
fit <- ndr(swiss)
b <- biplot(fit, dims = c(1, 2))
b
```

COVID19_2020 *COVID-19 case data for 138 countries and 18 variables*

Description

Sample datasets for Generalized Network-based Dimensionality Reduction and Analysis (GNDA)

Covid'19 of countries (2020), where the data frame has 138 observations of 18 variables.

Usage

```
data("COVID19_2020")
```

Format

A data frame with 138 observations 18 variables.

Source

Kurbucz, M. T. (2020). A joint dataset of official COVID-19 reports and the governance, trade and competitiveness indicators of World Bank group platforms. Data in brief, 31, 105881.

Examples

```
data(COVID19_2020)
```

CrimesUSA1990.X *Crimes in USA cities in 1990. Independent variables (X)*

Description

Sample datasets for Generalized Network-based Dimensionality Reduction and Analysis (GNDA)

Crimes in USA cities in 1990. Independent variables (X)

Usage

```
data("CrimesUSA1990.X")
```

Format

A data frame with 1994 observations 123 variables.

Source

UCI - Machine Learning Repository: https://archive.ics.uci.edu/ml/datasets/communities+and+crime

Examples

```
data(CrimesUSA1990.X)
```

`CrimesUSA1990.Y` *Crimes in USA cities in 1990. Dependent variable (Y)*

Description

Sample datasets for Generalized Network-based Dimensionality Reduction and Analysis (GNDA)

Crimes in USA cities in 1990. Dependent variable (Y)

Usage

```
data("CrimesUSA1990.Y")
```

Format

A data frame with 1994 observations 1 variables.

Source

UCI - Machine Learning Repository: https://archive.ics.uci.edu/ml/datasets/communities+and+crime

Examples

```
data(CrimesUSA1990.Y)
```

`CWTS_2020` *CWTS Leiden University Ranking 2020 with 1,176 universities and 42 indicators*

Description

Sample datasets for Generalized Network-based Dimensionality Reduction and Analysis (GNDA)

CWTS Leiden's 2020 dataset, where the data frame has 1176 observations of 42 variables.

Usage

```
data("CWTS_2020")
```

Format

A data frame with 1176 observations of 42 variables.

Source

CWTS Leiden Ranking 2020: https://www.leidenranking.com/ranking/2020/list

Examples

```
data(CWTS_2020)
```

`data_gen` *Generate Structured NDA Example Data*

Description

Generates a noisy block matrix without constructing intermediate block-diagonal matrices.

Usage

```
data_gen(n, m, nfactors = 2, lambda = 1, sparse = FALSE,
         density = 1, seed = NULL)
```

Arguments

`n`	Positive number of rows.
`m`	Positive number of columns.
`nfactors`	Positive number of blocks.
`lambda`	Finite exponential smoothing parameter.
`sparse`	Return a sparse **Matrix** object.
`density`	Proportion in (0, 1] of block entries retained as nonzero.
`seed`	Optional random seed.

Value

A data frame, or a sparse matrix when `sparse = TRUE`.

Examples

```
dense <- data_gen(30, 10, nfactors = 2, seed = 1)
sparse <- data_gen(100, 40, nfactors = 4, sparse = TRUE, density = 0.5)
```

dCor *Distance-Correlation Matrix*

Description

Calculates distance correlation for two vectors or for every unique pair of matrix columns.

Usage

```
dCor(x, y = NULL, test = FALSE, adjust = "BH", alpha = NULL,
     parallel = FALSE, cores = 1L, R = 499L)
```

Arguments

x	Numeric vector, matrix, or data frame.
y	Optional numeric vector paired with x.
test	Run permutation tests and return adjusted p-values.
adjust	Method passed to p.adjust.
alpha	Optional adjusted-p threshold used to zero nonsignificant coefficients.
parallel	Calculate unique matrix pairs in parallel on supported systems.
cores	Requested core count.
R	Permutation count passed to energy::dcor.test.

Value

A coefficient or symmetric matrix when test = FALSE; otherwise a list with r and p.

Examples

```
x <- matrix(rnorm(80), 20, 4)
dCor(x)
## Not run: dCor(x, test = TRUE, R = 999)
```

dCov *Distance-Covariance Matrix*

Description

Calculates distance covariance for two vectors or for every unique pair of matrix columns.

Usage

```
dCov(x, y = NULL, test = FALSE, adjust = "BH", alpha = NULL,
     parallel = FALSE, cores = 1L, R = 499L)
```

Arguments

x	Numeric vector, matrix, or data frame.
y	Optional numeric vector paired with x.
test	Run permutation tests and return adjusted p-values.
adjust	Method passed to p.adjust.
alpha	Optional adjusted-p threshold used to zero nonsignificant off-diagonal values.
parallel	Calculate unique matrix pairs in parallel on supported systems.
cores	Requested core count.
R	Permutation count passed to energy::dcov.test.

Value

A covariance value or symmetric matrix when test = FALSE; otherwise a list with r and p.

Examples

```
x <- matrix(rnorm(80), 20, 4); dCov(x)
```

---

fitted.ndrlm *NDRLM Fitted Values*

Description

Extracts fitted values from every response model.

Usage

```
## S3 method for class 'ndrlm'
fitted(object, ...)
```

Arguments

object	An ndrlm object.
...	Reserved for method compatibility.

Value

A data frame with one column per fitted response.

Examples

```
X <- as.data.frame(freeny.x)
Y <- data.frame(y = freeny.y)
fit <- ndrlm(Y, X, optimize = FALSE)
fitted(fit)
```

fs.dimred *Loading-Based Feature Selection*

Description

Selects indicators from NDA, PCA, factor-analysis, and other objects that expose loadings, rotations, or variable coordinates. When the original model call is available, the model is refitted after every removal so that all later decisions use the current communalities and loadings.

Usage

```
fs.dimred(fn, DF, min_comm = 0.25, com_comm = 0.25, ...)
```

Arguments

fn	Fitted dimensionality-reduction object. Supported structures include nda, prcomp, princomp, **psych** PCA/FA, and objects with loadings, rotation, or var$coord.
DF	Original numeric data or a symmetric correlation matrix.
min_comm	Minimum indicator communality.
com_comm	Minimum separation between the two largest absolute loadings. An indicator is treated as common only when its largest loading is both less than twice and less than com_comm above its second-largest loading.
...	Reserved for future model-specific extraction.

Value

The final refitted model, preserving its original class, with dropped_low, dropped_com, and retained_DF (also available as remain_DF) added. The original fitted model is available as model.

Examples

```
pca <- prcomp(swiss, scale. = TRUE)
selected <- fs.dimred(pca, swiss)
colnames(selected$retained_DF)
```

fs.KMO *KMO-Based Feature Selection*

Description

Iteratively removes the indicator with the lowest measure of sampling adequacy until the requested overall KMO is reached.

Usage

```
fs.KMO(data, min_MSA = 0.5, cor.mtx = FALSE,
       max_iter = ncol(data) - 1L, ...)
```

Arguments

data	Numeric row-level data or a correlation matrix.
min_MSA	Target overall Kaiser-Meyer-Olkin value in [0, 1].
cor.mtx	Treat data as a symmetric correlation matrix.
max_iter	Maximum number of removals.
...	Reserved for backward compatibility.

Value

Filtered data or correlation matrix. The final KMO result and removed variable names are stored in "KMO" and "removed" attributes.

Examples

```
filtered <- fs.KMO(swiss, min_MSA = 0.6)
```

GOVDB2020 *Governmental and economic data for 138 countries and 2,161 variables*

Description

Sample datasets for Generalized Network-based Dimensionality Reduction and Analysis (GNDA)

Governmental and economic data of countries (2020), where the data frame has 138 observations of 2161 variables.

Usage

```
data("GOVDB2020")
```

Format

A data frame with 138 observations of 2161 variables.

Source

Kurbucz, M. T. (2020). A joint dataset of official COVID-19 reports and the governance, trade and competitiveness indicators of World Bank group platforms. Data in brief, 31, 105881.

Examples

```
data(GOVDB2020)
```

I40_2020 *NUTS2 regional Industry 4.0 readiness data with 414 observations and 101 variables*

Description

Sample datasets for Generalized Network-based Dimensionality Reduction and Analysis (GNDA)

NUTS2 regional development data (2020), where the data frame has 414 observations of 101 variables.

Usage

```
data("I40_2020")
```

Format

A data frame with 414 observations of 101 variables.

Source

Honti, G., Czvetko, T., and Abonyi, J. (2020). Data describing the regional Industry 4.0 readiness index. Data in Brief, 33, 106464.

Examples

```
data(I40_2020)
```

ndr *Generalized Network-Based Dimensionality Reduction and Analysis*

Description

Builds a squared-association network, detects indicator communities, and estimates one centrality-weighted latent variable per retained community.

Usage

```
ndr(r, covar = FALSE, cor_method = 1, cor_type = 1, min_R = 0,
    min_comm = 2, Gamma = 1, null_model_type = 4, mod_mode = 6,
    min_evalue = 0, min_communality = 0, com_communalities = 0,
    use_rotation = FALSE, rotation = "oblimin", weight = NULL,
    seed = NULL, parallel = FALSE, cores = 1L, sparsecalc = FALSE,
    membership = NULL,
    centrality = c("eigenvector", "authority", "hub", "pagerank"),
    standardized = TRUE, P_mat = NULL, test = FALSE,
    p_adjust = "BH", alpha = NULL, bootstrap = FALSE,
    n_boot = 200L, conf = 0.95)
```

Arguments

r	Numeric row-level data, or a square association matrix when covar = TRUE. Sparse **Matrix** inputs are accepted.
covar	Use r as an association matrix instead of estimating associations from rows.
cor_method	Association method. Numeric values 1–6 retain compatibility; textual values are "pearson", "spearman", "kendall", "distance", "gaussian_rank", and "biweight". Gaussian-rank correlation is a fast monotone robust alternative; biweight correlation limits outlier influence.
cor_type	Association type: "full", "partial", or "semipartial"; 1–3 are accepted aliases.
min_R	Minimum squared association retained as an edge.
min_comm	Minimum preferred community size. A singleton is retained when it is the only feasible result.
Gamma	Positive resolution multiplier used by community detection and null-model subtraction.
null_model_type	Null model: 1 configuration-model expectation, 2 mean positive similarity, 3 min_R, or 4 no matrix modification (default).
mod_mode	Community method: Louvain, fast greedy, leading eigenvector, Infomap, Walktrap, or Leiden; either names or values 1–6.
min_evalue	Minimum selected centrality weight for periphery filtering.
min_communality	Minimum indicator communality.
com_communalities	Minimum separation between the two largest squared loadings for common-indicator filtering. Zero disables this filter.
use_rotation	Rotate a multi-factor solution.
rotation	Rotation understood by psych::principal.
weight	One finite observation-score weight per indicator.
seed	Optional random seed.
parallel	Use multicore calculation for pairwise distance associations and bootstrap replicates where supported.
cores	Number of requested cores.
sparsecalc	Store and process the thresholded similarity network as a sparse matrix.
membership	Optional supplied community membership, one value per indicator. Zero excludes an indicator.
centrality	Latent-variable weighting: eigenvector, authority, hub, or PageRank centrality.
standardized	Standardize input indicators using training means and standard deviations before weighting.
P_mat	Optional finite expected-edge matrix. When supplied it overrides null_model_type.
test	Calculate association p-values.

p_adjust	Method passed to p.adjust for false-discovery-rate control.
alpha	Optional adjusted-p threshold; nonsignificant associations are set to zero.
bootstrap	Bootstrap loading intervals from row-level data.
n_boot	Number of bootstrap replicates.
conf	Bootstrap confidence level.

Details

The method follows three stages: construct a squared-association graph, detect network communities, and form standardized centrality-weighted latent scores. Partial and semipartial networks can be asymmetric. Null-model types 1–3 operate on nonnegative residual similarities; type 4 leaves the similarity matrix unchanged.

Value

An object of class nda. Important elements include scores, loadings, communality, uniqueness, membership, EVCs, R, correlation, modularity, reconstruction, and stats. The stats list contains indicator p-values, optional bootstrap intervals, retained and dropped indicators, community sizes, and null-model metadata. Scores are unavailable for association-matrix input, but reconstruction-consistent loadings, communalities, and a loading-based reconstructed association matrix are returned.

References

Kosztyan, Z. T., Kurbucz, M. T., and Katona, A. I. (2022). Network-based dimensionality reduction of high-dimensional, low-sample-size datasets. *Knowledge-Based Systems*, 109180. doi:10.1016/j.knosys.2022.109180.

Kosztyan, Z. T., Katona, A. I., Kurbucz, M. T., and Lantos, Z. (2024). Generalized network-based dimensionality analysis. *Expert Systems with Applications*, 238, 121779. doi:10.1016/j.eswa.2023.121779.

Examples

```
fit <- ndr(swiss, cor_method = "spearman", centrality = "pagerank")
summary(fit)
scores <- predict(fit, swiss[1:3, ])

association <- cor(swiss)
fit_matrix <- ndr(association, covar = TRUE, membership = c(1, 1, 1, 2, 2, 2))
fit_matrix$loadings

## Not run:
fit_boot <- ndr(swiss, cor_method = "distance", test = TRUE,
                bootstrap = TRUE, n_boot = 500, parallel = TRUE, cores = 4)

## End(Not run)
```

ndrlm *Network-Based Dimensionality Reduction and Regression*

Description

Fits supervised models to original or NDA-reduced input and output variables, with optional NSGA-II tuning.

Usage

```
ndrlm(Y, X, latents = "in", dircon = FALSE, optimize = TRUE,
      target = "adj.r.square", rel_weight = FALSE, cor_method = 1,
      cor_type = 1, min_comm = 2, Gamma = 1, null_model_type = 4,
      mod_mode = 1, use_rotation = FALSE, rotation = "oblimin",
      pareto = FALSE, fit_weights = NULL,
      lower.bounds.x = rep(-100, ncol(X)),
      upper.bounds.x = rep(100, ncol(X)),
      lower.bounds.latentx = c(0, 0, 0, 0),
      upper.bounds.latentx = c(0.6, 0.6, 0.6, 0.3),
      lower.bounds.y = rep(-100, ncol(Y)),
      upper.bounds.y = rep(100, ncol(Y)),
      lower.bounds.latenty = c(0, 0, 0, 0),
      upper.bounds.latenty = c(0.6, 0.6, 0.6, 0.3),
      popsize = 20, generations = 30, cprob = 0.7, cdist = 5,
      mprob = 0.2, mdist = 10, seed = NULL,
      regression = c("lm", "ridge", "loess", "elastic_net"),
      model = NULL, alpha = 0.5, lambda = NULL, loess_span = 0.75,
      restarts = 1L, parallel = FALSE, cores = 1L,
      sparsecalc = FALSE, centrality = "eigenvector",
      standardized = TRUE, membership.X = NULL, membership.Y = NULL)
```

Arguments

`Y`	Numeric dependent-variable data.
`X`	Numeric independent-variable data with the same rows as `Y`.
`latents`	Use latent variables for inputs (`"in"`), outputs (`"out"`), both, or neither.
`dircon`	Retain NDA-excluded variables as direct predictors or responses.
`optimize`	Tune indicator weights and four NDA thresholds with NSGA-II.
`target`	One of adjusted R-squared, R-squared, MAE, MAPE, MASE, MSE, RMSE, AIC, or BIC.
`rel_weight`	Constrain and normalize indicator weights to be nonnegative relative weights.
`cor_method, cor_type, min_comm, Gamma, null_model_type, mod_mode, use_rotation, rotation`	Arguments forwarded to `ndr`.

`pareto` Keep one objective per original dependent variable during NSGA-II optimization.

`fit_weights` Optional weights for aggregating response-specific objectives when their number matches the fitted responses.

`lower.bounds.x`, `upper.bounds.x`
Lower and upper NSGA-II bounds for input-indicator weights.

`lower.bounds.latentx`, `upper.bounds.latentx`
Bounds for input-side `min_evalue`, `min_communality`, `com_communalities`, and `min_R`.

`lower.bounds.y`, `upper.bounds.y`
Lower and upper NSGA-II bounds for output-indicator weights.

`lower.bounds.latenty`, `upper.bounds.latenty`
Bounds for the four output-side NDA thresholds.

`popsize`, `generations`, `cprob`, `cdist`, `mprob`, `mdist`
NSGA-II population, generation, crossover, and mutation settings passed to `mco::nsga2`.

`seed` Optional seed; repeated optimization runs use consecutive seeds.

`regression` Regression engine: ordinary least squares (`"lm"`), ridge (`"ridge"`), LOESS (`"loess"`), or elastic net (`"elastic_net"`).

`model` Backward-friendly alias that overrides `regression`.

`alpha` Elastic-net mixing value in [0, 1]. Ridge always uses zero.

`lambda` Optional penalization value. If omitted, cross-validation selects lambda.

`loess_span` LOESS smoothing span.

`restarts` Number of independent NSGA-II runs; the best feasible candidate is refitted.

`parallel`, `cores`, `sparsecalc`, `centrality`, `standardized`
Performance and latent-score settings forwarded to `ndr`.

`membership.X`, `membership.Y`
Optional fixed NDA community memberships for the input and output sides.

Details

NDRLM is predictive and exploratory; its paths do not by themselves establish causal effects. Ridge and elastic net use **glmnet**. LOESS is intended for small predictor sets and does not provide linear-path significance tests. AIC and BIC for non-likelihood engines are residual-based approximations.

The class and structure of each element of `fits` depend on the selected regression engine. With `regression = "lm"`, each `fits[[i]]` is a genuine `lm` object and can be passed directly to `summary`, `coef`, `AIC`, `BIC`, `predict.lm`, and `lm.beta::lm.beta`. The **lm.beta** package is meaningful only for ordinary least-squares fits; it must not be applied to ridge, elastic-net, or LOESS fits.

With `regression = "ridge"`, `"elastic_net"`, or `"loess"`, each `fits[[i]]` is an NDRLM engine wrapper. It contains the engine name in `method`, the underlying fitted object in `fit`, and common fitted-value, residual, coefficient, and predictor metadata. Penalized fits additionally store the selected `lambda`, effective `alpha`, and standardized coefficients calculated on the complete training observations. The plot method applies the same standardization to ordinary least-squares fits, including compatible legacy `lm` objects. For engine-independent code, use `summary`, `predict`,

`fitted`, `residuals`, and `plot` on the enclosing `ndrlm` object instead of relying on these internal fields.

The NDRLM methods retain backward compatibility with both legacy objects that stored ordinary least-squares fits directly and early 0.3.0 objects that wrapped ordinary least-squares fits. Newly fitted ordinary least-squares models use direct `lm` objects, while the newer non-`lm` engines retain their wrappers.

Value

An `ndrlm` object containing response-specific metrics, fitted NDA objects, optimized hyperparameters, training designs, NSGA-II results and an `NSGA_trace` evaluation history when optimization is enabled, and a response-named `fits` list. Elements of `fits` are `lm` objects for ordinary least squares and NDRLM engine wrappers for ridge, elastic net, and LOESS.

References

Kosztyan, Z. T., Kurbucz, M. T., Katona, A. I., and Khan, Z. Network-based dimensionality reduction and structural modeling.

Examples

```
X <- as.data.frame(freeny.x)
Y <- data.frame(income = freeny.y)

fit_lm <- ndrlm(Y, X, latents = "none", regression = "lm")
inherits(fit_lm$fits[[1]], "lm")
summary(fit_lm$fits[[1]])
predict(fit_lm, X[1:3, ])

if (requireNamespace("lm.beta", quietly = TRUE)) {
  summary(lm.beta::lm.beta(fit_lm$fits[[1]]))
}

if (requireNamespace("glmnet", quietly = TRUE)) {
  fit_ridge <- ndrlm(Y, X, latents = "none", regression = "ridge",
                     lambda = 0.1)
  fit_ridge$fits[[1]]$method
  predict(fit_ridge, X[1:3, ])
}

if (requireNamespace("glmnet", quietly = TRUE)) {
  fit_en <- ndrlm(Y, X, latents = "none", regression = "elastic_net",
                  alpha = 0.5, lambda = 0.1)
  predict(fit_en, X[1:3, ])
}
```

normalize *Fast Min-Max Normalization*

Description

Rescales a numeric matrix or data frame globally, by row, or by column.

Usage

```
normalize(x, type = c("all", "row", "col"), range = c(0, 1),
          na.rm = TRUE)
```

Arguments

x	Numeric matrix or data frame.
type	Normalize all values together, each row, or each column.
range	Two increasing finite output limits.
na.rm	Remove missing values when finding limits.

Details

A constant slice is mapped to the midpoint of range.

Value

A normalized object with the same matrix or data-frame form as x.

Examples

```
m <- matrix(rnorm(20), 5, 4)
normalize(m)
normalize(m, type = "row", range = c(-1, 1))
```

pdCor *Partial Distance-Correlation Matrix*

Description

Forms a partial distance-correlation matrix by inverting the distance-covariance matrix.

Usage

```
pdCor(x, test = FALSE, adjust = "BH", alpha = NULL,
      parallel = FALSE, cores = 1L)
```

Arguments

x	Numeric matrix or data frame.
test	Return approximate coefficient p-values.
adjust	Method passed to p.adjust.
alpha	Optional adjusted-p filtering threshold.
parallel	Calculate distance-covariance pairs in parallel where supported.
cores	Requested core count.

Details

When the distance-covariance matrix is singular, a Moore-Penrose generalized inverse is used. The optional p-values use a t approximation and should be interpreted as screening diagnostics.

Value

A matrix, or a list with matrices r and p when test = TRUE.

Examples

```
x <- matrix(rnorm(100), 20, 5); pdCor(x)
```

---

plot.nda *Plot an NDA Association Network*

Description

Creates a printable static grid or interactive **visNetwork** network object. Assignment does not draw; printing the result draws it. The legacy **igraph** renderer remains available explicitly. Communities use a light qualitative node palette by default.

Usage

```
## S3 method for class 'nda'
plot(x, cuts = 0.3, interactive = FALSE, edgescale = 1,
    labeldist = -1.5, show_weights = FALSE, shapes = NULL,
    shape = "automatic", static.engine = c("grid", "igraph"),
    edge.color.links = TRUE, edge.colors = NULL,
    positive.color = "#009E73", negative.color = "#C62828",
    neutral.color = "grey65", positive.lty = 1, negative.lty = 1,
    neutral.lty = 1, edge.width.range = c(1, 6),
    edge.label.size = 10, edge.label.color = "#333333",
    edge.label.background = "#FFFFFFE6", edge.label.offset = 0.022,
    vertex.label.size = 12, vertex.label.color = "automatic",
    node.border.color = "#25313C",
    layout = "nicely", igraph_args = list(), visnetwork_args = list(), ...)
```

Arguments

x	An nda object.
cuts	Minimum squared correlation, or a two-value retained range. For association-matrix models a scalar is interpreted relative to the nonzero minimum and maximum.
interactive	Create a **visNetwork** widget instead of a static graph.
edgescale	Edge-width multiplier.
labeldist	Static vertex-label distance.
show_weights	Label edges with unsquared correlations for row-level models and supplied similarities for association-matrix models.
shape	"automatic" uses one common rectangle for every indicator. Alternatively, supply one shape or one shape per node. Community membership is shown by colour, not shape.
shapes	Backward-compatible alias for shape.
static.engine	Static renderer. The default "grid" renderer uses layered labels with opaque backgrounds; "igraph" retains the legacy renderer.
edge.color.links	Colour links by the sign of the unsquared association.
edge.colors	Optional legacy low/middle/high colour scale. When NULL, positive, negative, and zero associations use the three explicit colours below.
positive.color, negative.color, neutral.color	Default signed edge colours.
positive.lty, negative.lty, neutral.lty	Line types for signed edges. All are solid by default.
edge.width.range	Minimum and maximum edge widths. Absolute association magnitude is rescaled to this interval.
edge.label.size, edge.label.color, edge.label.background, edge.label.offset	Edge-label appearance, background, and perpendicular displacement from the edge.
vertex.label.size, vertex.label.color	Vertex-label appearance. The "automatic" colour selects black or white for readable contrast.
node.border.color	Node-border colour.
layout	An igraph layout suffix such as "nicely", "fr", or "circle".
igraph_args	Named legacy static plot arguments, used when static.engine = "igraph".
visnetwork_args	Named lists called network, nodes, edges, options, interaction, and layout, forwarded to matching visNetwork functions.
...	Additional legacy static plot.igraph arguments.

Value

An object of class `nda_plot`; print it to render.

Examples

```
fit <- ndr(swiss)
p <- plot(fit, interactive = FALSE, show_weights = TRUE)
p
```

`plot.ndrlm` *Plot an NDRLM Path Network*

Description

Creates a layered, printable network of indicators, latent variables, and fitted regression links. A latent variable and its indicators share one community colour. A light qualitative node palette is used by default. Structural path labels and widths use standardized coefficients by default, including ridge and elastic-net fits. Legacy unnamed fits recover their response and standardization data from the fitted model.

Usage

```
## S3 method for class 'ndrlm'
plot(x, sig = 0.05, interactive = FALSE,
    show_edge_labels = TRUE, edge.label.size = 9,
    vertex.label.size = 11, edgescale = 1, shape = "automatic",
    standardized.paths = TRUE, layer.positions = NULL,
    node.lighten = 0.18, node.alpha = 0.90,
    static.engine = c("grid", "igraph"),
    positive.color = "#009E73", negative.color = "#C62828",
    loading.color = "grey60", positive.lty = 1, negative.lty = 1,
    loading.lty = 2, edge.width.range = c(1, 6),
    edge.label.color = "#333333",
    edge.label.background = "#FFFFFFE6", edge.label.offset = 0.022,
    node.border.color = "#25313C",
    igraph_args = list(), visnetwork_args = list(), ...)
```

Arguments

`x`	An `ndrlm` object.
`sig`	Significance threshold for ordinary least-squares paths. Penalized and LOESS paths do not have p-values and are not filtered by `sig`.
`interactive`	Create a **visNetwork** widget.
`show_edge_labels`	Show loading estimates and structural path coefficients. Structural path labels are standardized by default.

`edge.label.size`, `vertex.label.size`
: Label sizes.

`edgescale`
: Absolute-weight edge-width multiplier.

`shape`
: `"automatic"` uses rectangles for observed variables and circles for latent variables. Community membership is encoded by colour. Alternatively, supply one shape or one shape per node.

`standardized.paths`
: If `TRUE`, use standardized structural coefficients. For linear, ridge, and elastic-net fits these are calculated as $\beta_j s_{x_j} / s_y$ from the complete training observations. Set to `FALSE` to display raw coefficients.

`layer.positions`
: Optional horizontal layer positions. Supply four strictly increasing values for the input-indicator, input-latent, output-latent, and output-indicator layers, or one value per active layer. The default closes gaps left by absent latent layers.

`node.lighten`, `node.alpha`
: Amount of white mixed into node fills and their opacity. Both accept values between zero and one.

`static.engine`
: Static renderer. The default `"grid"` renderer uses a centred layered layout and labels with opaque backgrounds; `"igraph"` retains the legacy renderer.

`positive.color`, `negative.color`, `loading.color`
: Colours for positive paths, negative paths, and measurement loadings.

`positive.lty`, `negative.lty`, `loading.lty`
: Line types for the three edge roles. Structural paths are solid and measurement links are dashed by default.

`edge.width.range`
: Minimum and maximum widths to which absolute estimates are rescaled separately within measurement loadings and structural paths. Structural widths use the same standardized or raw coefficients as their labels.

`edge.label.color`, `edge.label.background`, `edge.label.offset`
: Edge-label colour, opaque background, and perpendicular displacement from the edge.

`node.border.color`
: Node-border colour.

`igraph_args`
: Named legacy static plotting arguments, used when `static.engine = "igraph"`.

`visnetwork_args`
: Named `network`, `nodes`, `edges`, `options`, `interaction`, and `physics` argument lists.

`...`
: Additional legacy static `plot.igraph` arguments.

Value

An `ndrlm_plot` object; printing it renders the graph.

Examples

```
X <- as.data.frame(freeny.x)
Y <- data.frame(y = freeny.y)
p <- plot(ndrlm(Y, X, optimize = FALSE))
p
```

---

predict.nda *Predict NDA Scores*

Description

Applies training feature standardization, indicator weights, retained centralities, latent-score scaling, and rotation to new rows.

Usage

```
## S3 method for class 'nda'
predict(object, newdata, ...)
```

Arguments

object	An nda object fitted to row-level data.
newdata	Numeric data containing every original training indicator.
...	Reserved for method compatibility.

Value

A matrix with one predicted score column per latent variable. If newdata is NULL, training scores are returned.

Examples

```
fit <- ndr(swiss); predict(fit, swiss[1:4, ])
```

---

predict.ndrlm *Predict from an NDRLM Model*

Description

Transforms new independent variables with the fitted NDA model and predicts every fitted response.

Usage

```
## S3 method for class 'ndrlm'
predict(object, newdata = NULL, se.fit = FALSE,
    interval = c("none", "confidence", "prediction"), level = 0.95, ...)
```

Arguments

object	An ndrlm object.
newdata	New original independent variables. If omitted, fitted values are returned.
se.fit	Request standard errors for ordinary least-squares fits.
interval	No interval, a confidence interval, or a prediction interval. Intervals are available for ordinary least squares only.
level	Confidence level.
...	Additional arguments passed to predict.lm.

Value

A response-named data frame for ordinary predictions, or a response-named list when standard errors or intervals are requested.

Examples

```
X <- as.data.frame(freeny.x); Y <- data.frame(income = freeny.y)
fit <- ndrlm(Y, X, optimize = FALSE)
predict(fit, X[1:3, ])
```

---

print.nda *Print an NDA Model*

Description

Prints the NDA call, settings, modularity, community sizes, loadings, communalities, and dropped indicators.

Usage

```
## S3 method for class 'nda'
print(x, digits = getOption("digits"), ...)
```

Arguments

x	An nda object.
digits	Printing precision.
...	Additional arguments passed to the summary printer.

Value

x, invisibly.

Examples

```
fit <- ndr(swiss); print(fit)
```

print.ndrlm *Print an NDRLM Model*

Description

Prints the reduction mode, regression engine, target metrics, and response-specific fits.

Usage

```
## S3 method for class 'ndrlm'
print(x, digits = getOption("digits"), ...)
```

Arguments

x	An ndrlm object.
digits	Printing precision.
...	Additional arguments passed to the summary printer.

Value

x, invisibly.

Examples

```
X <- as.data.frame(freeny.x)
Y <- data.frame(y = freeny.y)
print(ndrlm(Y, X, optimize = FALSE))
```

residuals.ndrlm *NDRLM Residuals*

Description

Extracts observed-minus-fitted residuals from every response model.

Usage

```
## S3 method for class 'ndrlm'
residuals(object, ...)
```

Arguments

object	An ndrlm object.
...	Reserved for method compatibility.

Value

A data frame with one residual column per fitted response.

Examples

```
X <- as.data.frame(freeny.x)
Y <- data.frame(y = freeny.y)
fit <- ndrlm(Y, X, optimize = FALSE)
residuals(fit)
```

---

spdCor *Semipartial Distance-Correlation Matrix*

Description

Forms an asymmetric semipartial distance-correlation matrix from distance covariances.

Usage

```
spdCor(x, test = FALSE, adjust = "BH", alpha = NULL,
       parallel = FALSE, cores = 1L)
```

Arguments

x	Numeric matrix or data frame.
test	Return approximate coefficient p-values.
adjust	Method passed to p.adjust.
alpha	Optional adjusted-p filtering threshold.
parallel	Calculate distance-covariance pairs in parallel where supported.
cores	Requested core count.

Details

When the distance-covariance matrix is singular, a Moore-Penrose generalized inverse is used. The optional p-values use a t approximation and should be interpreted as screening diagnostics.

Value

A matrix, or a list with matrices r and p when test = TRUE.

Examples

```
x <- matrix(rnorm(100), 20, 5); spdCor(x)
```

summary.nda *Summarize an NDA Model*

Description

Returns a printable summary without producing output during assignment.

Usage

```
## S3 method for class 'nda'
summary(object, digits = getOption("digits"), ...)
```

Arguments

object	An nda object.
digits	Preferred printing precision.
...	Additional summary arguments.

Value

An object of class summary.nda containing settings, modularity, community sizes, loadings, communalities, uniquenesses, and inference statistics.

Examples

```
fit <- ndr(swiss); s <- summary(fit); s
```

summary.ndrlm *Summarize an NDRLM Model*

Description

Returns a printable regression and reduction summary without printing during assignment.

Usage

```
## S3 method for class 'ndrlm'
summary(object, digits = getOption("digits"), ...)
```

Arguments

object	An ndrlm object.
digits	Printing precision.
...	Additional summary arguments.

Value

A `summary.ndrlm` object containing settings, variables, response metrics, fitted NDA objects, and regression fits.

Examples

```
X <- as.data.frame(freeny.x)
Y <- data.frame(y = freeny.y)
s <- summary(ndrlm(Y, X, optimize = FALSE))
s
```

Index

∗ **array**
data_gen, 6
∗ **datasets**
COVID19_2020, 4
CrimesUSA1990.X, 4
CrimesUSA1990.Y, 5
CWTS_2020, 5
GOVDB2020, 10
I40_2020, 11
∗ **matrix**
normalize, 17
∗ **multivariate**
dCor, 7
dCov, 7
fitted.ndrlm, 8
fs.dimred, 9
fs.KMO, 9
ndr, 11
ndrlm, 14
pdCor, 17
plot.nda, 18
plot.ndrlm, 20
predict.nda, 22
predict.ndrlm, 22
print.nda, 23
print.ndrlm, 24
residuals.ndrlm, 24
spdCor, 25
summary.nda, 26
summary.ndrlm, 26
∗ **package**
nda-package, 2
∗ **plot**
biplot.nda, 3

biplot, *3*
biplot.nda, 3

COVID19_2020, 4
CrimesUSA1990.X, 4
CrimesUSA1990.Y, 5
CWTS_2020, 5

data_gen, 6
dCor, *3*, 7
dCov, 7

fitted.ndrlm, 8
fs.dimred, 9
fs.KMO, 9

GOVDB2020, 10

I40_2020, 11

nda *(*nda-package*)*, 2
nda-package, 2
ndr, *3*, 11
ndrlm, *3*, 14
normalize, 17

pdCor, 17
plot, *3*
plot.nda, 18
plot.ndrlm, 20
predict.nda, 22
predict.ndrlm, 22
print.nda, 23
print.ndrlm, 24

residuals.ndrlm, 24

spdCor, 25
summary, *3*
summary.nda, 26
summary.ndrlm, 26