Package 'nonnet'

September 15, 2026

Title Generate and Analyze Nonlinear Networks

Version 1.0.0

Maintainer Lindley Slipetz <ddj6tu@virginia.edu>

Description Creates and detects nonlinear relations using the methods described in Slipetz, Qiu, Sun, and Henry (2026) <doi:10.48550/arXiv.2411.02763>. Use the netgen() function to generate a nonlinear network and the dcor_res() function for a residualization procedure for detecting nonlinear relations.

License MIT + file LICENSE

Encoding UTF-8

Imports energy, mgcv, stats

Config/roxygen2/version 8.1.0

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation no

Author Lindley Slipetz [aut, cre],
Teague Henry [ctb]

Repository CRAN

Date/Publication 2026-09-15 10:40:11 UTC

Contents

dcor_res *Distance Correlation Test After Residualization*

Description

Residualizes two random variables with respect to a set of covariates using generalized additive models, then tests the association between the residualized variables using a permutation-based distance correlation test.

Usage

```
dcor_res(dat, A, C, B_variables, nperm)
```

Arguments

dat	A data frame
A	Character string specifying the name of the first variable to be used in the distance correlation.
C	Character string specifying the name of the second variable to be used in the distance correlation.
B_variables	Character vector containing the names of the covariates used as predictors when residualizing A and C.
nperm	Integer. Number of permutations used in the distance correlation test.

Value

A numeric value giving the permutation p-value from `energy::dcor.test()`.

Examples

```
dat <- data.frame("A" = c(2,3,4), "C" = c(4,9,16), "B" = c(1,1,1), "D" = c(1,1,1))
res <- dcor_res(dat, "A", "C", c("B", "D"), 1000)
res
```

netgen *Generate Simulated Nonlinear Network Data*

Description

Simulates a dataset containing a central variable, peripheral variables, and a target variable under different functional relationships.

Usage

```
netgen(
  n,
  mean_cent,
  sd_cent,
  mean_peri_non,
  mean_peri_lin,
  sd_peri_non,
  sd_peri_lin,
  mean_target,
  sd_target,
  beta_peri_lin,
  beta_cent_lin,
  beta_lin,
  beta_non,
  beta_cent_non,
  beta_peri_non,
  inter_cent_lin,
  beta_con,
  func,
  con_func
)
```

Arguments

n	Integer. Number of observations to generate.
mean_cent	Mean of the central variable A.
sd_cent	Standard deviation of the central variable A.
mean_peri_non	Numeric vector. Means of the peripheral nonlinear variables.
mean_peri_lin	Numeric vector. Means of the peripheral linear variables.
sd_peri_non	Numeric vector. Standard deviations of the peripheral nonlinear variables.
sd_peri_lin	Numeric vector. Standard deviations of the peripheral linear variables.
mean_target	Mean of the random error term for the target variable C.
sd_target	Standard deviation of the random error term for the target variable C.
beta_peri_lin	Numeric vector. Regression coefficients for effects of the peripheral linear variables on C.
beta_cent_lin	Numeric vector. Regression coefficients for effects of A on the peripheral linear variables.
beta_lin	Coefficient for the linear effect of A on C.
beta_non	Coefficient for the nonlinear effect of A on C.
beta_cent_non	Numeric vector. Regression coefficients for effects of A on the peripheral nonlinear variables.
beta_peri_non	Numeric vector. Regression coefficients for effects of the peripheral nonlinear variables on C.

inter_cent_lin	Integer vector. Indices identifying peripheral variables involved in interactions with A.
beta_con	Regression coefficient for the confounding effect.
func	Character. Functional form relating A to C.
con_func	Character. Functional form relating A to the nonlinear peripheral variables.

Value

A data frame containing A, C, linear peripheral variables B1, B2, ..., and nonlinear peripheral variables D1, D2, ...

Examples

```
set.seed(123)

dat <- netgen(
  n = 100,
  mean_cent = 0,
  sd_cent = 1,
  mean_peri_non = rep(0, 4),
  mean_peri_lin = rep(0, 3),
  sd_peri_non = rep(1, 4),
  sd_peri_lin = rep(1, 3),
  mean_target = 0,
  sd_target = 1,
  beta_peri_lin = c(0.3, 0.4, 0.5),
  beta_cent_lin = rep(0.5, 3),
  beta_lin = 0.5,
  beta_non = 0.75,
  beta_cent_non = c(0.5, 0.6, 0.7, 0.8),
  beta_peri_non = c(0.3, 0.4, 0.5, 0.6),
  inter_cent_lin = c(1, 2, 3),
  beta_con = 0,
  func = "quad",
  con_func = "quad"
)

head(dat)
```

Index