Package ‘simrel’  
September 17, 2021

**Title**  Simulation of Multivariate Linear Model Data  
**Version**  2.1.0  
**Description**  Researchers have been using simulated data from a multivariate linear model to compare and evaluate different methods, ideas and models. Additionally, teachers and educators have been using a simulation tool to demonstrate and teach various statistical and machine learning concepts. This package helps users to simulate linear model data with a wide range of properties by tuning few parameters such as relevant latent components. In addition, a shiny app as an 'RStudio' gadget gives users a simple interface for using the simulation function. See more on: Sæbø, S., Almøy, T., Hel-land, I.S. (2015) <doi:10.1016/j.chemolab.2015.05.012> and Rimal, R., Almøy, T., Sæbø, S. (2018) <doi:10.1016/j.chemolab.2018.02.009>.  

**Depends**  R (>= 3.5.0)  
**License**  GPL-3  
**Encoding**  UTF-8  
**RoxygenNote**  7.1.2  
**VignetteBuilder**  knitr  
**BugReports**  https://github.com/simulatr/simrel/issues  
**URL**  https://simulatr.github.io/simrel/  
**Repository**  CRAN  
**Imports**  FrF2, ggplot2, gridExtra, jsonlite, magrittr, miniUI, purrr, reshape2, rstudioapi, scales, sfsmisc, shiny, tidyr, rlang, testthat  
**Suggests**  covr, knitr, pls, markdown, DoE.base  
**NeedsCompilation**  no  
**Author**  Raju Rimal [aut, cre] (<https://orcid.org/0000-0001-6097-0285>), Solve Sæbø [aut, ths] (Original creator of the package, <https://orcid.org/0000-0001-8699-4592>), Kristian Hovde Liland [aut] (Contributor and coauthor of the univariate version of simrel, <https://orcid.org/0000-0001-6468-9423>)  
**Maintainer**  Raju Rimal <raju.rimal@medisin.uio.no>  
**Date/Publication**  2021-09-17 15:30:02 UTC
Description

Simulation of Multivariate Linear Model Data

Usage

AppSimrel()

Value

No return value, runs the shiny interface for simulation
bisimrel

Simulation of Multivariate Linear Model data with response

Description

Simulation of Multivariate Linear Model data with response

Usage

bisimrel(
  n = 50,
  p = 100,
  q = c(10, 10, 5),
  rho = c(0.8, 0.4),
  relpos = list(c(1, 2), c(2, 3)),
  gamma = 0.5,
  R2 = c(0.8, 0.8),
  ntest = NULL,
  muY = NULL,
  muX = NULL,
  sim = NULL
)

Arguments

n       Number of training samples
p       Number of x-variables
q       Vector of number of relevant predictor variables for first, second and common
to both responses
rho     A 2-element vector, unconditional and conditional correlation between y_1 and
        y_2
relpos  A list of position of relevant component for predictor variables. The list contains
        vectors of position index, one vector or each response
gamma   A declining (decaying) factor of eigen value of predictors (X). Higher the value
        of gamma, the decrease of eigenvalues will be steeper
R2      Vector of coefficient of determination for each response
ntest   Number of test observation
muY     Vector of average (mean) for each response variable
muX     Vector of average (mean) for each predictor variable
sim     A simrel object for reusing parameters setting
Value

A simrel object with all the input arguments along with following additional items

- **X**: Simulated predictors
- **Y**: Simulated responses
- **beta**: True regression coefficients
- **beta0**: True regression intercept
- **relpred**: Position of relevant predictors
- **testX**: Test Predictors
- **testY**: Test Response
- **minerror**: Minimum model error
- **Rotation**: Rotation matrix of predictor (R)
- **type**: Type of simrel object, in this case *bivariate*
- **lambda**: Eigenvalues of predictors
- **Sigma**: Variance-Covariance matrix of response and predictors

References


Examples

```r
sobj <- bisimrel(  
  n = 100,  
  p = 10,  
  q = c(5, 5, 3),  
  rho = c(0.8, 0.4),  
  relpos = list(c(1, 2, 3), c(2, 3, 4)),  
  gamma = 0.7,  
  R2 = c(0.8, 0.8)  
)  
# Regression Coefficients from this simulation  
sobj$beta
```

cov_mat

Extract various sigma matrices

Description

Extract various sigma matrices

Usage

cov_mat(obj, which = c("xy", "zy", "zw"), use_population = TRUE)

Arguments

obj : A simrel object

which : A character string to specify which covariance matrix to extract, possible values are "xy", "zy" and "zw"

use_population : A boolean whether to use compute population values or to estimate from sample

Value

A matrix of covariances with column equals to the number of response and row equals to the number of predictors

Examples

set.seed(1983)
sobj <- multisimrel()
cov_mat(sobj, which = "xy", use_population = TRUE)
cov_mat(sobj, which = "xy", use_population = FALSE)

cov_plot_data

Prepare data for Plotting Covariance Matrix

Description

Prepare data for Plotting Covariance Matrix

Usage

cov_plot_data(sobj, type = "relpos", ordering = TRUE, facetting = TRUE)
Arguments

obj A simrel object
type Type of covariance matrix - can take two values relpos for relevant position of principal components and relpred for relevant position of predictor variables
ordering TRUE for ordering the covariance for block diagonal display
facetting TRUE for facetting the predictor and response space. FALSE will give a single facet plot

Value

A data frame with covariances and related values based on type argument that is ready to plot

Examples

sobj <- simrel(n = 100, p = 10, q = c(4, 5), relpos = list(c(1, 2, 3), c(4, 6, 7)), m = 3,
R2 = c(0.8, 0.7), ypos = list(c(1, 3), 2), gamma = 0.7, type = "multivariate")
head(cov_plot_data(sobj))

cov_xy Covariance between X and Y

Description

Covariance between X and Y

Usage

cov_xy(obj, use_population = TRUE)

Arguments

obj A simrel object
use_population A boolean to specify wheather to use population or sample

Value

A covariance matrix of X and Y
\textbf{cov\_zw} \hspace{1em} \textit{Covariance between Z and W}

\textbf{Description}

Helper Functions

\textbf{Usage}

\texttt{cov\_zw(obj)}

\textbf{Arguments}

\texttt{obj} \hspace{1em} A simrel object

\textbf{Value}

A covariance matrix of Z and W

\textbf{cov\_zy} \hspace{1em} \textit{Covariance between Z and Y}

\textbf{Description}

Covariance between Z and Y

\textbf{Usage}

\texttt{cov\_zy(obj, use\_population = \text{TRUE})}

\textbf{Arguments}

\texttt{obj} \hspace{1em} A simrel object

\texttt{use\_population} \hspace{1em} A boolean to specify whether to use population or sample

\textbf{Value}

A covariance matrix of Z and Y
expect_subset  Extra test functions

Description

Extra test functions

Usage

```r
expect_subset(
  object,
  expected,
  info = NULL,
  label = NULL,
  expected.label = NULL
)
```

Arguments

- `object`: object to test
- `expected`: Expected value
- `info`: extra information to be included in the message (useful when writing tests in loops).
- `label`: object label. When ‘NULL’, computed from deparsed object.
- `expected.label`: Equivalent of ‘label’ for shortcut form.

Value

Returns the object itself if expected value is found in the object as a subset else return Error

Examples

```r
expect_subset(c(1, 2, 3, 4, 5), c(2, 4, 5))
```

ggsimrelplot  Simulation Plot with ggplot: The true beta, relevant component and eigen structure

Description

Simulation Plot with ggplot: The true beta, relevant component and eigen structure
Usage

ggsimrelplot(
  obj,
  ncomp = min(obj$p, obj$n, 20),
  which = 1L:3L,
  layout = NULL,
  print.cov = FALSE,
  use_population = TRUE
)

Arguments

obj A simrel object
ncomp Number of components to plot
which A character indicating which plot you want as output, it can take TrueBeta, RelComp and EstRelComp
layout A layout matrix of how to layout multiple plots
print.cov Output estimated covariance structure
use_population Logical, TRUE if population values should be used and FALSE if sample values should be used

Value

A list of plots

Examples

sim.obj <- simrel(n = 100, p = 16, q = c(3, 4, 5),
  relpos = list(c(1, 2), c(3, 4), c(5, 7)), m = 5,
  ypos = list(c(1, 4), 2, c(3, 5)), type = "multivariate",
  R2 = c(0.8, 0.7, 0.9), gamma = 0.8)

ggsimrelplot(sim.obj, layout = matrix(c(2, 1, 3, 1), 2))

ggsimrelplot(sim.obj, which = c(1, 2), use_population = TRUE)

ggsimrelplot(sim.obj, which = c(1, 2), use_population = FALSE)

ggsimrelplot(sim.obj, which = c(1, 3), layout = matrix(c(1, 2), 1))
Description

Function to create multi-level binary replacement (MBR) design (Martens et al., 2010). The MBR approach was developed for constructing experimental designs for computer experiments. MBR makes it possible to set up fractional designs for multi-factor problems with potentially many levels for each factor. In this package it is mainly called by the mbrdsim function.

Usage

```r
mbrd(
  l2levels = c(2, 2),
  fraction = 0,
  gen = NULL,
  fnames1 = NULL,
  fnames2 = NULL
)
```

Arguments

- **l2levels**: A vector indicating the number of log2-levels for each factor. E.g. `c(2,3)` means 2 factors, the first with $2^2 = 4$ levels, the second with $2^3 = 8$ levels.
- **fraction**: Design fraction at bit-level. Full design: fraction=0, half-fraction: fraction=1, and so on...
- **gen**: list of generators at bit-factor level. Same as generators in function FrF2.
- **fnames1**: Factor names of original multi-level factors (optional).
- **fnames2**: Factor names at bit-level (optional).

Details

The MBR design approach was developed for designing fractional designs in multi-level multi-factor experiments, typically computer experiments. The basic idea can be summarized in the following steps: 1) Choose the number of levels $L$ for each multi-level factor as a multiple of 2, that is $L \in \{2, 4, 8, \ldots\}$. 2) Replace any given multi-level factor by a set of $\log_2(L)$ two-level "bit factors". The complete bit-factor design can then be expressed as a $2^K$ design where $K$ is the total number of bit-factors across all original multi-level factors. 3) Choose a fraction level $P$ defining an fractional design $2^{(K-P)}$ (see e.g. Montgomery, 2008) as for regular two-levels factorial designs. 4) Express the reduced design in terms of the original multi-level factors.

Value

- **BitDesign**: The design at bit-factor level (inherits from FrF2). Function `design.info()` can be used to get extra design info of the bit-design, and `plot` for plotting of the bit-level design.
- **Design**: The design at original factor levels, non-randomized.
References


Examples

```r
# Two variables with 8 levels each (2^3=8), a half-fraction design.
res <- mbrd(c(3,3), fraction=1, gen=list(c(1,4)))
# plot(res$Design, pch=20, cex=2, col=2)
# Three variables with 8 levels each, a 1/16-fraction.
res <- mbrd(c(3,3,3), fraction=4)
# library(rgl)
# plot3d(res$Design, type="s", col=2)
```

**mbrdsim**  
*A function to set up a design for a given set of factors with their specific levels using the MBR-design method.*

Description

The multi-level binary replacement (MBR) design approach is used here in order to facilitate the investigation of the effects of the data properties on the performance of estimation/prediction methods. The mbrdsim function takes as input a list containing a set of factors with their levels. The output is an MBR-design with the combinations of the factor levels to be run.

Usage

```r
mbrdsim(simlist, fraction, gen = NULL)
```

Arguments

- **simlist**: A named list containing the levels of a set of (multi-level) factors.
- **fraction**: Design fraction at bit-level. Full design: fraction=0, half-fraction: fraction=1, and so on.
- **gen**: Generators for the fractioning at the bit level. Default is NULL for which the generators are chosen automatically by the FrF2 function. See documentation of FrF2 for details on how to set the generators.

Value

- **BitDesign**: The design at bit-factor level. The object is of class design, as output from FrF2. Function design.info() can be used to get extra design info of the bit-design. The bit-factors are named.numbered if the input factor list is named.
- **Design**: The design at original factor level, non-randomized. The factors are named if the input factor list is named.
Author(s)

Solve Sæbø

References


Examples

```r
# Input: A list of factors with their levels (number of levels must be a multiple of 2).
## Simrel Parameters ----
sim_list <- list(
p = c(20, 150),
gamma = seq(0.2, 1.1, length.out = 4),
relpos = list(list(c(1, 2, 3), c(4, 5, 6)), list(c(1, 5, 6), c(2, 3, 4))),
R2 = list(c(0.4, 0.8), c(0.8, 0.8)),
ypos = list(list(1, c(2, 3)), list(c(1, 3), 2))
)
## 1/8 fractional Design ----
dgn <- mbrdsim(sim_list, fraction = 3)
design <- cbind(
dgn["Design"],
q = lapply(dgn["Design"], function(x) rep(x/2, 2)),
type = "multivariate",
n = 100,
ntest = 200,
m = 3,
eta = 0.6)
## Simulation ----
sobj <- apply(design, 1, function(x) do.call(simrel, x))
names(sobj) <- paste0("Design", seq.int(sobj))

# Info about the bit-design including bit-level aliasing (and resolution if \code{gen = NULL})
if (requireNamespace("DoE.base", quietly = TRUE)) {
dgn <- mbrdsim(sim_list, fraction = 3)
DoE.base::design.info(dgn$BitDesign)
}
```

Description

Simulation of Multivariate Linear Model Data
Usage

```r
msim(
    p = 15,
    q = c(5, 4, 3),
    m = 5,
    relpos = list(c(1, 2), c(3, 4, 6), c(5, 7)),
    gamma = 0.6,
    R2 = c(0.8, 0.7, 0.8),
    eta = 0,
    muX = NULL,
    muY = NULL,
    ypos = list(c(1), c(3, 4), c(2, 5))
)
```

Arguments

- **p**: Number of variables
- **q**: Vector containing the number of relevant predictor variables for each relevant response components
- **m**: Number of response variables
- **relpos**: A list of position of relevant component for predictor variables. The list contains vectors of position index, one vector or each relevant response components
- **gamma**: A declining (decaying) factor of eigen value of predictors (X). Higher the value of gamma, the decrease of eigenvalues will be steeper
- **R2**: Vector of coefficient of determination (proportion of variation explained by predictor variable) for each relevant response components
- **eta**: A declining (decaying) factor of eigenvalues of response (Y). Higher the value of eta, more will be the declining of eigenvalues of Y. `eta = 0` refers that all eigenvalues of responses (Y) are 1.
- **muX**: Vector of average (mean) for each predictor variable
- **muY**: Vector of average (mean) for each response variable
- **ypos**: List of position of relevant response components that are combined to generate response variable during orthogonal rotation

Value

A simrel object with all the input arguments along with following additional items

- **X**: Simulated predictors
- **Y**: Simulated responses
- **W**: Simulated predictor components
- **Z**: Simulated response components
- **beta**: True regression coefficients
- **beta0**: True regression intercept
multisimrel

Description

Simulation of Multivariate Linear Model Data

Usage

```r
multisimrel(
  n = 100,
  p = 15,
  q = c(5, 4, 3),
  m = 5,
  relpos = list(c(1, 2), c(3, 4, 6), c(5, 7)),
  gamma = 0.6,
  R2 = c(0.8, 0.7, 0.8),
)```

References


### Arguments

- **n**: Number of observations
- **p**: Number of variables
- **q**: Vector containing the number of relevant predictor variables for each relevant response component
- **m**: Number of response variables
- **replpos**: A list of position of relevant component for predictor variables. The list contains vectors of position index, one vector or each relevant response component
- **gamma**: A declining (decaying) factor of eigen value of predictors (X). Higher the value of gamma, the decrease of eigenvalues will be steeper
- **R2**: Vector of coefficient of determination (proportion of variation explained by predictor variable) for each relevant response components
- **eta**: A declining (decaying) factor of eigenvalues of response (Y). Higher the value of eta, more will be the declining of eigenvalues of Y. \( \eta = 0 \) refers that all eigenvalues of responses (Y) are 1.
- **ntest**: Number of test observation
- **muX**: Vector of average (mean) for each predictor variable
- **muY**: Vector of average (mean) for each response variable
- **ypos**: List of position of relevant response components that are combined to generate response variable during orthogonal rotation

### Value

A simrel object with all the input arguments along with following additional items

- **X**: Simulated predictors
- **Y**: Simulated responses
- **W**: Simulated predictor components
- **Z**: Simulated response components
- **beta**: True regression coefficients
- **beta0**: True regression intercept
- **replpred**: Position of relevant predictors
- **testX**: Test Predictors
- **testY**: Test Response
- **testW**: Test predictor components
parse_parm

Description

These function helps to parse a character string into a list object and also creates parameters for performing multiple simulations

Usage

parse_parm(character_string, in_list = FALSE)

Arguments

character_string

A character string for parameter where the items in a list is separated by semi-colon. For example: 1; 2; 3, 4

in_list

TRUE if the result need to wrap in a list, default is FALSE

Value

A list or a vector

References


Examples

```
parse_parm("1, 2; 3, 4")
parse_parm("1, 2")
```

---

### Description

Plotting Functions

### Usage

```
plot_beta(obj, base_theme = theme_grey, lab_list = NULL, theme_list = NULL)
```

### Arguments

- **obj** A simrel object
- **base_theme** Base ggplot theme to apply
- **lab_list** List of labs arguments such as x, y, title, subtitle
- **theme_list** List of theme arguments to apply in the plot

### Value

A plot of true regression coefficients for the simulated data

### Examples

```
sobj <- multisimrel()
sobj %>%
  plot_beta(
    base_theme = ggplot2::theme_bw,
    lab_list = list(
      title = "Regression Coefficients",
      subtitle = "From Simulation",
      y = "True Regression Coefficients"
    ),
    theme_list = list(
      legend.position = "bottom"
    )
  )
```
plot_cov

Plotting Covariance Matrix

Description
Plotting Covariance Matrix

Usage
plot_cov(sobj, type = "relpos", ordering = TRUE, facetting = TRUE)

Arguments

- **sobj**: A simrel object
- **type**: Type of covariance matrix - can take two values relpos for relevant position of principal components and relpred for relevant position of predictor variables
- **ordering**: TRUE for ordering the covariance for block diagonal display
- **facetting**: TRUE for facetting the predictor and response space. FALSE will give a single facet plot

Value
A covariance plot

References


Examples
sobj <- simrel(n = 100, p = 10, q = c(4, 5), relpos = list(c(1, 2, 3), c(4, 6, 7)), m = 3, R2 = c(0.8, 0.7), ypos = list(c(1, 3), 2), gamma = 0.7, type = "multivariate")
p1 <- plot_cov(sobj, type = "relpos", facetting = FALSE)
p2 <- plot_cov(sobj, type = "rotation", facetting = FALSE)
p3 <- plot_cov(sobj, type = "relpred", facetting = FALSE)
gridExtra::grid.arrange(p1, p2, p3, ncol = 3)
plot_covariance

Plot Covariance between predictor (components) and response (components)

Description

Plot Covariance between predictor (components) and response (components)

Usage

plot_covariance(
  sigma_df,
  lambda_df = NULL,
  base_theme = theme_grey,
  lab_list = NULL,
  theme_list = NULL
)

Arguments

sigma_df A data.frame generated by tidy_sigma
lambda_df A data.frame generated by tidy_lambda
base_theme Base ggplot theme to apply
lab_list List of labs arguments such as x, y, title, subtitle
theme_list List of theme arguments to apply in the plot

Value

A plot of true regression coefficients for the simulated data

Examples

sobj <- bisimrel(p = 12)
sigma_df <- sobj %>%
  cov_mat(which = "zy") %>%
  tidy_sigma() %>%
  abs_sigma()
lambda_df <- sobj %>%
  tidy_lambda()
plot_covariance(
  sigma_df,
  lambda_df,
  base_theme = ggplot2::theme_bw,
  lab_list = list(
    title = "Covariance between Response and Predictor Components",
    subtitle = "The bar represents the eigenvalues predictor covariance",
    y = "Absolute covariance",
  ),
)


plot_simrel

A wrapper function for a simrel object

Description

A wrapper function for a simrel object

Usage

plot_simrel(
  obj,
  ncomp = min(obj$p, obj$n, 20),
  which = c(1L:4L),
  layout = NULL,
  print.cov = FALSE,
  use.population = TRUE,
  palette = "Set1",
  base_theme = ggplot2::theme_grey,
  lab_list = NULL,
  theme_list = NULL
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>obj</td>
<td>A simrel object</td>
</tr>
<tr>
<td>ncomp</td>
<td>Number of components to show in x-axis</td>
</tr>
<tr>
<td>which</td>
<td>An integer specifying which simrel plot to obtain</td>
</tr>
<tr>
<td>layout</td>
<td>A layout matrix for arranging the simrel plots</td>
</tr>
<tr>
<td>print.cov</td>
<td>A boolean where to print covariance matrices</td>
</tr>
<tr>
<td>use.population</td>
<td>A boolean specifying weather to get plot for population or sample</td>
</tr>
<tr>
<td>palette</td>
<td>Name of color palette compatible with RColorBrewer</td>
</tr>
<tr>
<td>base_theme</td>
<td>Base ggplot theme to apply</td>
</tr>
<tr>
<td>lab_list</td>
<td>List of labs arguments such as x, y, title, subtitle. A nested list if the argument which has length greater than 1.</td>
</tr>
<tr>
<td>theme_list</td>
<td>List of theme arguments to apply in the plot. A nested list if the argument which has length greater than 1.</td>
</tr>
</tbody>
</table>
prepare_design

Value

Value

Simrel Plot(s)

Examples

```r
sobj <- bisimrel(p = 12)
plot_simrel(sobj, layout = matrix(1:4, 2, 2))
```

---

**Description**

Prepare design for experiment from a list of simulation parameter

**Usage**

```r
prepare_design(option_list, tabular = TRUE)
```

**Arguments**

- `option_list` A list of options that is to be parsed
- `tabular` logical if output is needed in tabular form or list format

**Value**

A list of parsed parameters for simulatr

**Examples**

```r
opts <- list(
  n = rep(100, 2),
  p = c(20, 40),
  q = c("5, 5, 4", "10, 5, 5"),
  m = c(5, 5),
  relpos = c("1; 2, 4; 3", "1, 2; 3, 4; 5"),
  gamma = c(0.2, 0.4),
  R2 = c("0.8, 0.9, 0.7", "0.6, 0.8, 0.7"),
  ypos = c("1, 4; 2, 5; 3", "1; 2, 4; 3, 5"),
  ntest = rep(1000, 2)
)
design <- prepare_design(opts)
design
```
Description

Simulation of Multivariate Linear Model Data

Usage

simrel(n, p, q, relpos, gamma, R2, type = "univariate", ...)

Arguments

n Number of observations.

p Number of variables.

q An integer for univariate, a vector of 3 integers for bivariate and 3 or more for multivariate simulation (for details see Notes).

relpos A list (vector in case of univariate simulation) of position of relevant component for predictor variables corresponding to each response.

gamma A declining (decaying) factor of eigenvalues of predictors (X). Higher the value of gamma, the decrease of eigenvalues will be steeper.

R2 Vector of coefficient of determination (proportion of variation explained by predictor variable) for each relevant response components.

type Type of simulation - univariate, bivariate and multivariate

... Since this is a wrapper function to simulate univariate, bivariate or multivariate, it calls their respective function. This parameter should contain all the necessary arguments for respective simulations. See unisimrel, bisimrel and multisimrel

Value

A simrel object with all the input arguments along with following additional items.

X Simulated predictors

Y Simulated responses

W Simulated predictor components

Z Simulated response components

beta True regression coefficients

beta0 True regression intercept

relpred Position of relevant predictors

testX Test Predictors

testY Test Response
testW  Test predictor components

minerror  Minimum model error

Xrotation  Rotation matrix of predictor (R)

Yrotation  Rotation matrix of response (Q)

type  Type of simrel object *univariate* or *multivariate*

lambda  Eigenvalues of predictors

SigmaWZ  Variance-Covariance matrix of components of response and predictors

SigmaWX  Covariance matrix of response components and predictors

SigmaYZ  Covariance matrix of response and predictor components

Sigma  Variance-Covariance matrix of response and predictors

RsqW  Coefficient of determination corresponding to response components

RsqY  Coefficient of determination corresponding to response variables

References


**Description**

Simulation Plot: The true beta, relevant component and eigen structure

**Usage**

```r
simrelplot(
  obj,
  ncomp = min(obj$p, obj$n, 20),
  ask = TRUE,
  print.cov = FALSE,
  which = 1L:3L
)
```
Arguments

- obj: A simrel object
- ncomp: Number of components to plot
- ask: logical, TRUE: functions ask for confirmation FALSE: function layout plot on predefined format
- print.cov: Output estimated covariance structure
- which: A character indicating which plot you want as output, it can take TrueBeta, RelComp and EstRelComp

Value

A list of plots

**tidy_beta**

*Tidy Functions to make plotting easy*

Description

Tidy Functions to make plotting easy

Absolute value of sigma scaled by the overall maximum absolute value

Usage

```r
tidy_beta(obj)
abs_sigma(sigma_df)
```

Arguments

- obj: A Simrel Object
- sigma_df: A tidy covariance data frame generated by tidy_sigma function

Value

A tibble with three columns: Predictor, Response and BetaCoef

Another data.frame (tibble) of same dimension with absolute covariance scaled by overall maximum absolute values

Examples

```r
sobj <- multisimrel()
beta_df <- tidy_beta(sobj)
beta_df
sobj <- multisimrel()
sobj %>%
cov_mat("zy") %>%
tidy_sigma() %>%
abs_sigma()
```
tidy_lambda

Extract Eigenvalues of predictors

Description
Extract Eigenvalues of predictors

Usage
	tidy_lambda(obj, use_population = TRUE)

Arguments
obj 
A simrel Object

use_population 
A boolean to specify where to use population value or calculate from sample

Value
A dataframe of eigenvalues for each predictors

Examples
sobj <- multisimrel()
sobj %>%
tidy_lambda()

tidy_sigma

Tidy covariance matrix

Description
Tidy covariance matrix

Usage

tidy_sigma(covs)

Arguments
covs 
A sigma matrix obtained from cov_mat function

Value
A tibble with three columns: Predictor, Response and Covariance

Examples
sobj <- multisimrel()
unisimrel  

Function for data simulation

Description

Functions for data simulation from a random regression model with one response variable where the data properties can be controlled by a few input parameters. The data simulation is based on the concept of relevant latent components and relevant predictors, and was developed for the purpose of testing methods for variable selection for prediction.

Usage

unisimrel(
  n,  
p,  
q,  
relpos,  
gamma,  
R2,  
ntest = NULL,  
muY = NULL,  
muX = NULL,  
lambda.min = .Machine$double.eps,  
sim = NULL
)

Arguments

n  
The number of (training) samples to generate.

p  
The total number of predictor variables to generate.

q  
The number of relevant predictor variables (as a subset of \( p \)).

relpos  
A vector indicating the position (between 1 and \( p \)) of the \( m \) relevant components, e.g. \( c(1, 2) \) means that the first two latent components should be relevant. The length of relpos must be equal to \( m \).

gamma  
A number defining the speed of decline in eigenvalues (variances) of the latent components. The eigenvalues are assumed to decline according to an exponential model. The first eigenvalue is set equal to 1.

R2  
The theoretical R-squared according to the true linear model. A number between 0 and 1.

ntest  
The number of test samples to be generated (optional).

muY  
The true mean of the response variable (optional). Default is \( \text{muY} = \text{NULL} \).

muX  
The \( p \)-vector of true means of the predictor variables (optional). Default is \( \text{muX} = \text{NULL} \).

lambda.min  
Lower bound of the eigenvalues. Defaults to \( \text{.Machine$double.eps} \).
A fitted simrel object. If this is given, the same regression coefficients will be used to simulate a new data set of requested size. Default is NULL, for which new regression coefficients are sampled.

Details

The data are simulated according to a multivariate normal model for the vector \( (y, z_1, z_2, z_3, \ldots, z_p)^t \) where \( y \) is the response variable and \( z = (z_1, \ldots, z_p)^t \) is the vector of latent (principal) components. The ordered principal components are uncorrelated variables with declining variances (eigenvalues) defined for component \( j \) as \( e^{-\gamma^*j} / e^{-\gamma} \). Hence, the variance (eigenvalue) of the first principal component is equal to 1, and a large value of \( \gamma \) gives a rapid decline in the variances. The variance of the response variable is by default fixed equal to 1.

Some of the principal components (ordered by their decreasing variances) are assumed to be relevant for the prediction of the response. The indices of the positions of the relevant components are set by the \texttt{relpos} argument. The joint degree of relevance for the relevant components is determined by the population R-squared defined by \texttt{R2}.

In order to obtain predictor variables \( x = (x_1, x_2, \ldots, x_p)^t \) for \( y \), a random rotation of the principal components is performed. Hence, \( x = R^t * z \) for some random rotation matrix \( R \). For values of \( q \) satisfying \( m \leq q < p \) only a subspace of dimension \( q \) containing the \( m \) relevant component(s) is rotated. This facilitates the possibility to generate \( q \) relevant predictor variables \( (x's) \). The indices of the relevant predictors is randomly selected with the only restriction that the index set contains the indices in \texttt{relpos}. The final index set of the relevant predictors is saved in the output argument \texttt{relpred}. If \( q=p \) all \( p \) predictor variables are relevant for the prediction of \( y \).

For further details on the simulation approach, please see Søbø, Almøy and Helland (2015).

Value

A simrel object with list of following items,

- \texttt{call} The call to simrel.
- \texttt{X} The \((n \times p)\) simulated predictor matrix.
- \texttt{Y} The \(n\)-vector of simulated response values.
- \texttt{beta} The vector of true regression coefficients.
- \texttt{beta0} The true intercept. This is zero if \texttt{muY=NULL} and \texttt{muX=NULL}
- \texttt{muY} The true mean of the response variable.
- \texttt{muX} The \(p\)-vector of true means of the predictor variables.
- \texttt{relpred} The index of the true relevant predictors, that is the \(x\)-variables with non-zero true regression coefficients.
- \texttt{TESTX} The \((ntest \times p)\) matrix of optional test samples.
- \texttt{TESTY} The \(ntest\)-vector of responses of the optional test samples.
- \texttt{n} The number of simulated samples.
- \texttt{p} The number of predictor variables.
- \texttt{m} The number of relevant components.
The number of relevant predictors.

The decline parameter in the exponential model for the true eigenvalues.

The true eigenvalues of the covariance matrix of the p predictor variables.

The true R-squared value of the linear model.

The positions of the relevant components.

The minimum achievable prediction error. Also the variance of the noise term in the linear model.

The sampled correlations between the principal components and the response.

The true covariance matrix of \((y, z_1, z_2, \ldots, z_p)^T\).

The random rotation matrix which is used to achieve the predictor variables as rotations of the latent components. Equals the transposed of the eigenvector-matrix of the covariance matrix of \((x_1, \ldots, x_p)^T\).

The type of response generated, either "univariate" as returned from simrel, or "bivariate" as returned from simrel2.

**Author(s)**

Solve S<e6>b<f8> and Kristian H. Liland

**References**

Helland, I. S. and Alm<f8>y, T., 1994, Comparison of prediction methods when only a few components are relevant, *J. Amer. Statist. Ass.*, 89(426), 583 – 591.


**Examples**

```r
#Linear model data, large n, small p
mydata <- unisimrel(n = 250, p = 20, q = 5, relpos = c(2, 4), gamma = 0.25, R2 = 0.75)

#Estimating model parameters using ordinary least squares
lmfit <- lm(mydata$Y ~ mydata$X)
summary(lmfit)

#Comparing true with estimated regression coefficients
plot(mydata$beta, lmfit$coef[-1], xlab = "True regression coefficients",
     ylab = "Estimated regression coefficients")
abline(0,1)

#Linear model data, small n, large p
mydata <- unisimrel(n = 50, p = 200, q = 25, relpos = c(2, 4), gamma = 0.25, R2 = 0.8 )

#Simulating more samples with identical distribution as previous simulation
mydata2 <- unisimrel(n = 2500, sim = mydata)
```
#Estimating model parameters using partial least squares regression with cross-validation to determine the number of relevant components.
if (requireNamespace("pls", quietly = TRUE)) {
  require(pls)
  plsfit <- plsr(mydata$Y ~ mydata$X, 15, validation = "CV")

  #Validation plot and finding the number of relevant components.
  plot(0:15, c(plsfit$validation$PRESS0, plsfit$validation$PRESS),
       type = "b", xlab = "Components", ylab = "PRESS")
  mincomp <- which(plsfit$validation$PRESS == min(plsfit$validation$PRESS))

  #Comparing true with estimated regression coefficients.
  plot(mydata$beta, plsfit$coef[, 1, mincomp], xlab = "True regression coefficients",
       ylab = "Estimated regression coefficients")
  abline(0, 1)
}
Index

* Design
  mbrdsim, 11
* MBRD
  mbrdsim, 11
* Model
  unisimrel, 26
* Simulation
  unisimrel, 26
* covariance plot
  cov_plot_data, 5
* datagen
  bisimrel, 3
  msim, 12
  multisimrel, 14
  simrel, 22
* design
  mbrd, 9
* experimental design
  mbrd, 9
* intgernal
  cov_xy, 6
  cov_zw, 7
  cov_zy, 7
* linear model data
  bisimrel, 3
  msim, 12
  multisimrel, 14
  simrel, 22
* linear model
  cov_plot_data, 5
* multivariate
  multisimrel, 14
* simulation plot
  ggsimrelplot, 8
  simrelplot, 23
* simulation
  bisimrel, 3
  cov_plot_data, 5
  msim, 12
  multisimrel, 14
  simrel, 22
  abs_sigma (tidy_beta), 24
  AppSimrel, 2
  bisimrel, 3, 22
  cov_mat, 5
  cov_plot_data, 5
  cov_xy, 6
  cov_zw, 7
  cov_zy, 7
  expect_subset, 8
  ggsimrelplot, 8
  mbrd, 9
  mbrdsim, 11
  msim, 12
  multisimrel, 14, 22
  parse_parm, 16
  plot_beta, 17
  plot_cov, 18
  plot_covariance, 19
  plot_simrel, 20
  prepare_design, 21
  simrel, 22
  simrelplot, 23
  tidy_beta, 24
  tidy_lambda, 25
  tidy_sigma, 25
  unisimrel, 22, 26