Package 'ExploreModelMatrix'

November 12, 2026

Type Package

Title Graphical Exploration of Design Matrices

Version 1.24.0

Date 2026-03-16

Description Given a sample data table and a design formula, ExploreModelMatrix generates an interactive application for exploration of the resulting design matrix. This can be helpful for interpreting model coefficients and constructing appropriate contrasts in (generalized) linear models. Static visualizations can also be generated.

License MIT + file LICENSE

Encoding UTF-8

Imports shiny (>= 1.5.0), shinydashboard, DT, cowplot, utils, dplyr, magrittr, tidyr, ggplot2, stats, methods, rintrojs, scales, tibble, MASS, limma, S4Vectors, shinyjs, rlang

RoxygenNote 7.3.3

Suggests testthat (>= 2.1.0), knitr, rmarkdown, htmltools, BiocStyle

VignetteBuilder knitr

URL https://github.com/csoneson/ExploreModelMatrix

BugReports https://github.com/csoneson/ExploreModelMatrix/issues

biocViews ExperimentalDesign, Regression, DifferentialExpression, ShinyApps

git_url https://git.bioconductor.org/packages/ExploreModelMatrix

git_branch RELEASE_3_23

git_last_commit fe30945

git_last_commit_date 2026-04-28

Repository Bioconductor 3.23

Date/Publication 2026-11-11

Author Charlotte Soneson [aut, cre] (ORCID:
<https://orcid.org/0000-0003-3833-2169>),
Federico Marini [aut] (ORCID: <https://orcid.org/0000-0003-3252-7758>),
Michael Love [aut] (ORCID: <https://orcid.org/0000-0001-8401-0545>),
Florian Geier [aut] (ORCID: <https://orcid.org/0000-0002-9076-9264>),
Michael Stadler [aut] (ORCID: <https://orcid.org/0000-0002-2269-4934>)

Maintainer Charlotte Soneson <charlottesoneson@gmail.com>

Contents

ExploreModelMatrix-package 2
.AddNewLine 3
.CalculateVIFsLM 3
.ExampleDesigns 4
.IsValidFormula 4
ExploreModelMatrix 5
VisualizeDesign 6

Index **8**

`ExploreModelMatrix-package`
ExploreModelMatrix

Description

ExploreModelMatrix is an R package for visualizing design matrices generated by the `model.matrix()` R function. Provided with a sample data table and a design formula, the `ExploreModelMatrix()` function launches a shiny app where the user can explore the fitted values (in terms of the model coefficients) for each combination of predictor values.

Author(s)

Maintainer: Charlotte Soneson <charlottesoneson@gmail.com> (ORCID)

Authors:

- Federico Marini <marinif@uni-mainz.de> (ORCID)
- Michael Love <michaelisaiahlove@gmail.com> (ORCID)
- Florian Geier <florian.geier@unibas.ch> (ORCID)
- Michael Stadler <michael.stadler@fmi.ch> (ORCID)

See Also

Useful links:

- https://github.com/csoneson/ExploreModelMatrix
- Report bugs at https://github.com/csoneson/ExploreModelMatrix/issues

.AddNewLine *Split a string into multiple lines if it's longer than a certain length*

Description

Split a string into multiple lines if it's longer than a certain length

Usage

```
.AddNewLine(st, lineWidth)
```

Arguments

st	A string
lineWidth	The maximum length of a line

Value

A string

.CalculateVIFsLM *Calculate variance inflation factors from a design matrix*

Description

The calculation of VIFs is done by fitting a linear model for each column in the design matrix, with all the other columns (except the intercept) as predictors. If there is an intercept (a column named "(Intercept)") in the original design matrix, each linear model will be fit with an intercept. If there is no such column in the original design matrix, the linear models will be fit without an intercept. After fitting the linear model for column i, the corresponding VIF is calculated as 1/(1-R^2), where R^2 is the coefficient of determination of the model. "Inf" results (obtained when R^2=1) are replaced by NAs.

Usage

```
.CalculateVIFsLM(mm)
```

Arguments

mm	A model.matrix object

Value

A data.frame with estimated VIFs for each coefficient, or NULL if the calculations could not be performed (there are no non-intercept columns in the design matrix, or the linear model fitting fails).

Author(s)

Charlotte Soneson

.ExampleDesigns *Define example designs*

Description

Define example designs

Usage

```
.ExampleDesigns(exampleID)
```

Arguments

exampleID	The name of the example design. One of "One factor, unpaired samples", "One factor, paired samples", "Two crossed factors", "Two crossed, one blocking factor", "Two crossed, one nested factor", "Two crossed, one nested factor, dummy coded", "Two crossed, one nested factor (manuscript example)"

Value

A list with two elements: the sample data table and the design formula

Author(s)

Charlotte Soneson

.IsValidFormula *Check whether a design formula is valid*

Description

Checks whether the object is indeed a formula, and whether all specified factors are present in the experimental metadata provided

Usage

```
.IsValidFormula(design, expdata)
```

Arguments

design	The specified formula
expdata	The experimental metadata data.frame

Value

Logical value

`ExploreModelMatrix` *Explore model matrix*

Description

Given a sample data table and a design formula, explore the resulting design matrix graphically in an interactive application.

Usage

```
ExploreModelMatrix(
  sampleData = NULL,
  designFormula = NULL,
  addStopButton = TRUE
)
```

Arguments

`sampleData`	(optional) A `data.frame` or `DataFrame` with sample information. If set to `NULL`, the user can upload the sample information from a tab-separated text file inside the app, or choose among a collection of example designs provided in the app.
`designFormula`	(optional) A `formula`. All components of the terms must be present as columns in `sampleData`. If set to `NULL`, the design formula can be specified after launching the app.
`addStopButton`	Logical scalar. If `TRUE` (default), will add a button to stop the app (by calling `shiny::stopApp`).

Value

A Shiny app object

Author(s)

Charlotte Soneson, Federico Marini, Michael I Love, Florian Geier, Michael B Stadler

Examples

```
app <- ExploreModelMatrix(
  sampleData = data.frame(genotype = rep(c("A", "B"), each = 4),
                          treatment = rep(c("treated", "untreated"), 4)),
  designFormula = ~genotype + treatment
)
if (interactive()) shiny::runApp(app)
```

VisualizeDesign *Visualize design matrix*

Description

Given a sample table and a design formula, generate a collection of static plots for exploring the resulting design matrix graphically. This function is called internally by `ExploreModelMatrix()`, but can also be used directly if interactivity is not required.

Usage

```
VisualizeDesign(
  sampleData,
  designFormula = NULL,
  flipCoordFitted = FALSE,
  flipCoordCoocc = FALSE,
  textSizeFitted = 5,
  textSizeCoocc = 5,
  textSizeLabsFitted = 12,
  textSizeLabsCoocc = 12,
  lineWidthFitted = 25,
  addColorFitted = TRUE,
  colorPaletteFitted = scales::hue_pal(),
  dropCols = NULL,
  designMatrix = NULL
)
```

Arguments

`sampleData`	A `data.frame` of `DataFrame` with sample information.
`designFormula`	A `formula`. All components of the terms must be present as columns in `sampleData`.
`flipCoordFitted, flipCoordCoocc`	A `logical`, whether to flip the coordinate axes in the fitted values/co-occurrence plot, respectively.
`textSizeFitted, textSizeCoocc`	A `numeric` scalar giving the text size in the fitted values/co-occurrence plot, respectively.
`textSizeLabsFitted, textSizeLabsCoocc`	A `numeric` scalar giving the text size for the axis labels in the fitted values/co-occurrence plot, respectively.
`lineWidthFitted`	A `numeric` scalar giving the maximal length of a row in the fitted values plot, before it is split and printed on multiple lines
`addColorFitted`	A `logical` scalar indicating whether the terms in the fitted values plot should be shown in different colors.

`colorPaletteFitted`
: A `function` returning a color palette to use for coloring the model coefficients in the fitted values plot.

`dropCols`
: A character vector with columns to drop from the design matrix, or NULL if no columns should be dropped.

`designMatrix`
: A `numeric` matrix, which can be supplied as an alternative to `designFormula`. Rows must be in the same order as the rows in `sampleData`.

Details

Note that if a design matrix is supplied (via the `designMatrix` argument), caution is required in order to interpret especially the cooccurrence plot in the situation where the provided `sampleData` contains additional columns not used to generate the design matrix (or when it does not contain all the relevant columns).

Value

A list with the following elements:

- `sampledata`: A `data.frame`, expanded from the input `sampleData`
- `plotlist`: A list of plots, displaying the fitted values for each combination of predictor values, in terms of the model coefficients.
- `designmatrix`: The design matrix, after removing any columns in `dropCols`
- `pseudoinverse`: The pseudoinverse of the design matrix
- `vifs`: A `data.frame` with calculated variance inflation factors
- `colors`: A vector with colors to use for different model coefficients
- `cooccurrenceplots`: A list of plots, displaying the co-occurrence pattern for the predictors (i.e., the number of observations for each combination of predictor values)
- `totnbrrows`: The total number of "rows" in the list of plots of fiitted values. Useful for deciding the required size of the plot canvas.

Author(s)

Charlotte Soneson

Examples

```
VisualizeDesign(
  sampleData = data.frame(genotype = rep(c("A", "B"), each = 4),
                          treatment = rep(c("treated", "untreated"), 4)),
  designFormula = ~genotype + treatment
)
```

Index

∗ **internal**
.AddNewLine, 3
.CalculateVIFsLM, 3
.ExampleDesigns, 4
.IsValidFormula, 4
ExploreModelMatrix-package, 2
.AddNewLine, 3
.CalculateVIFsLM, 3
.ExampleDesigns, 4
.IsValidFormula, 4

ExploreModelMatrix, 5
ExploreModelMatrix-package, 2

VisualizeDesign, 6