

Package: DOEpro (via r-universe)

October 10, 2026

Title Analysis of Designed Agricultural Experiments

Version 2.0.1

Description A 'shiny' application and supporting functions for the analysis of designed agricultural experiments, following the procedures set out by Gomez and Gomez (1984, ISBN:9780471870920). Handles completely randomised, randomised complete block, Latin square, factorial (up to four factors), split-plot and strip-plot designs, and pooled (combined) analysis over environments including factorial treatments, after Yates and Cochran (1938) <doi:10.1017/S0021859600050978>; analyses several response variables simultaneously; recommends and applies variance-stabilising transformations using the profile likelihood of Box and Cox (1964) <doi:10.1111/j.2517-6161.1964.tb00553.x>; reports standard errors and critical differences for every legitimate comparison, including the four distinct comparisons of a split plot, the mixed ones using the approximation of Satterthwaite (1946) <doi:10.2307/3002019>; and produces publication-format tables of means, diagnostic plots and a written interpretation.

License GPL (>= 3)

Encoding UTF-8

Depends R (>= 4.0)

Imports shiny, bslib, DT, ggplot2, rlang, stats, utils, graphics, grDevices

Suggests pagedown, testthat (>= 3.0.0)

Config/testthat/edition 3

URL <https://github.com/Uzairkhan11w/DOEpro>, <https://doepro.pages.dev>

BugReports <https://github.com/Uzairkhan11w/DOEpro/issues>

Config/roxygen2/version 8.0.0

Config/pak/sysreqs cmake make libuv1-dev zlib1g-dev

Repository <https://uzairkhan11w.r-universe.dev>

Date/Publication 2026-10-10 14:42:32 UTC
RemoteUrl https://github.com/uzairkhan11w/doepro
RemoteRef HEAD
RemoteSha 466b63c16e71085765c457c21da76c06a13e10e3

Contents

analyze	2
build_report	4
check_data	5
cluster_data	5
correlate_data	7
demo_data	8
describe_data	9
DESIGNS	9
fa_data	10
fix_data	11
pca_data	12
regress_data	13
run_all	15
run_DOEpro	16
TRANS	16
Index	18

analyze	Analyse one response from a designed experiment
---------	---

Description

Fits the analysis of variance for a single response variable and returns the means, standard errors and critical differences for every legitimate comparison the design allows.

Usage

```
analyze(d, design, map, alpha = 0.05)
```

Arguments

d	A data frame in long format: one row per plot, with columns for the design factors and the response. Text factor columns are put in natural order (D0, D60, D120; T1, T2, ..., T10), with a label carrying no number, such as Control, placed first; a column that is already a factor keeps its levels, so supply a factor to fix an order such as Vegetative, Flowering, Maturity.
design	The design code. One of the values of DESIGNS , for example "RCBD", "SPLIT" or "POOLFRCBD".

map	A named list mapping roles to column names of d. Always needs response. Then, by design: treat (CRD, RCBD, LSD); block (RCBD, factorial RCBD); row, col (LSD); factors, a character vector of two to four column names (factorial); rep, main, sub (split and strip plots); env together with treat or factors, and rep for an RCBD base (pooled designs).
alpha	The significance level. It sets the critical differences, the grouping letters and the significance of every F-test. Defaults to 0.05.

Details

The error term is chosen to match the design. A split plot is fitted with `Error(rep/main)` and a strip plot with `Error(rep/(A+B))`, so the main-plot and sub-plot comparisons are each tested against their own error; the two mixed comparisons use a weighted t . In a pooled (combined) analysis over environments, each treatment effect is tested against its own interaction with the environment, and each environment by treatment interaction against the pooled error.

Unequal replication is handled exactly in the designs with a single error term (CRD, RCBD, Latin square and the factorials): each mean gets its own standard error and each pair of means its own standard error of a difference and critical difference. When blocks are incomplete or factorial cells are unequal, the means are adjusted (least-squares) means and every term is tested adjusted for all the others (Type III). Designs with several error strata (split plot, strip plot and the pooled analyses) can only be analysed exactly when they are complete; with a plot missing `analyze()` stops and says which combinations are missing.

Value

A list with, among others:

`anova` The analysis of variance table.

`effects` One entry per effect. Each holds means, the table of means with columns Mean, N, SD, SE (the standard error of that mean), the grouping letters and, when the means are adjusted, `Raw_mean`; `sed_mat` and `cd_mat`, the standard error of the difference and the critical difference for every pair of means; `sem`, `sed` and `cd`, single values that are NA when replication is unequal and they differ from mean to mean (`equal_rep` is then FALSE, and `se_range`, `sed_range` and `cd_range` give the range); `mse` and `df`, the error mean square and degrees of freedom the effect is tested against; `p`, `F` and `alpha`. The interaction of a split plot, strip plot or pooled analysis also has `slice`, the factor within whose levels its means are compared: its SE, `sed` and `cd` apply to those comparisons, `sed_mat` and `cd_mat` are NA across levels of `slice`, and `extra` gives the standard errors and critical differences of the other comparisons.

`mse`, `dfe` The error mean square and its degrees of freedom.

`balanced` Whether every cell of the design holds the same number of observations.

`cv`, `grand`, `resid`, `lm` The coefficient of variation, grand mean, residuals and the fitted linear model.

See Also

[run_all](#) to analyse several responses at once, and [build_report](#) to render the result.

Examples

```
d <- demo_data("RCBD")
res <- analyze(d, "RCBD", list(response = "Yield", treat = "Variety",
                                block = "Block"))

res$anova
res$effects[["Variety"]]$means
```

build_report

Render an analysis as an HTML report

Description

Turns the result of [run_all](#) into a complete, self-contained HTML report: the analysis of variance, the tables of means with their standard errors and critical differences, the checks of the assumptions, and a plain-English interpretation. The report carries the DOEpro citation in its footer.

Usage

```
build_report(rr, letters_on = TRUE, detailed = TRUE, screen = FALSE)
```

Arguments

rr	The object returned by run_all .
letters_on	Show the grouping letters. Letters are suppressed automatically for any effect whose F-test is not significant, whatever this is set to.
detailed	If TRUE, include the full per-effect tables of means. If FALSE, give the compact summary tables only.
screen	If TRUE, return a fragment styled for display inside the application. If FALSE, return a complete standalone HTML document.

Value

A character string of HTML, of length one.

Examples

```
rr <- run_all(demo_data("CRD"), "CRD", list(treat = "Treatment"), "Yield")
html <- build_report(rr)
substr(html, 1, 60)

# writeLines(html, "report.html")
```

check_data	<i>Check data for common entry problems</i>
------------	---

Description

Looks through every column for the problems that most often spoil an analysis of field data: numbers written with a decimal comma, decimal points and commas mixed in one column, thousands separators, per cent signs and units typed after numbers, numbers stored as text, entries that are not numbers, empty cells and the markers people type for them (such as n/a or Excel's #DIV/0!), labels that differ only in spacing, capitals or punctuation, and empty or repeated rows. Each problem is described with the rows it affects and, where a correction is safe, what the correction would be. Nothing is changed; see [fix_data](#).

Usage

```
check_data(d)
```

Arguments

d A data frame, as loaded.

Value

A list with issues, a list with one element per problem (column; level: "correct" for a correction [fix_data](#) will make, "check" for something only the user can settle, "note" for information; message; rows, as row numbers of the data; fixable), and numeric and labels, the columns read as numbers and as labels.

Examples

```
d <- data.frame(Variety = c("V1", "V1", "V2", "V2"),
               Yield = c("5,6", "6.1", "n/a", "12a"))
chk <- check_data(d)
vapply(chk$issues, `[`, "", "message")
```

cluster_data	<i>Hierarchical cluster analysis</i>
--------------	--------------------------------------

Description

Clusters the rows, or the means of each level of a grouping column, on the chosen numeric columns ([hclust](#)), from the rows with a value of every one. By default each variable is standardised first, so no variable dominates through its units. Text entries are read as the data check reads them; see [check_data](#).

Usage

```
cluster_data(
  d,
  vars,
  group = NULL,
  k = NULL,
  distance = c("euclidean", "manhattan"),
  linkage = c("ward", "complete", "average", "single"),
  scale = TRUE
)

## S3 method for class 'doepro_cluster'
print(x, ...)
```

Arguments

d	A data frame.
vars	The names of two or more numeric columns.
group	Optional: the name of a column whose levels are clustered by their means, instead of clustering the rows.
k	The number of clusters, or NULL for the suggested number.
distance	"euclidean" or "manhattan".
linkage	"ward" (Ward's method, which needs Euclidean distances), "complete", "average" or "single".
scale	TRUE (the default) to standardise each variable.
x	A result of cluster_data().
...	Not used.

Details

The number of clusters, unless given, is the one from 2 to 10 (or one fewer than the items) with the largest average silhouette width (Rousseeuw, 1987): for each item, how much closer it lies to the other items of its own cluster than to those of the nearest other cluster, from -1 to 1. The cophenetic correlation measures how faithfully the tree keeps the distances between the items. Clusters are numbered in the order they appear in the tree from left to right.

Value

A list of class `doepro_cluster`: `items` (the labels of the items clustered), `X` (their values), `cluster` (each item's cluster), `sizes`, `means` (each cluster's mean of each variable), `silhouette` (each item's width), `by_k` (the average silhouette width for each number of clusters tried), `suggested`, `k`, `cophenetic`, `tree` (the `hclust` result), and the settings.

Examples

```
set.seed(1)
d <- data.frame(Variety = rep(paste0("V", 1:9), each = 3))
d$Yield <- rep(c(30, 31, 29, 40, 42, 41, 35, 36, 34), each = 3) + rnorm(27)
d$Height <- rep(c(80, 82, 79, 95, 97, 96, 88, 87, 90), each = 3) + rnorm(27)
cl <- cluster_data(d, c("Yield", "Height"), group = "Variety")
cl$k
cl$cluster
```

correlate_data	<i>Correlation between pairs of numeric variables</i>
----------------	---

Description

For every pair of the variables named: the number of rows with both values (N), the coefficient (Pearson's r, Spearman's rho or Kendall's tau-b), its two-sided p-value, a significance mark at the chosen level (* at alpha, ** at alpha / 5, NS otherwise), for Pearson's r a confidence interval at level 1 - alpha (Fisher's approximation), the strength of the association, and a one-line reading. Text entries are read as the data check reads them: an entry that is not plainly a number counts as missing; see [check_data](#).

Usage

```
correlate_data(
  d,
  vars,
  method = c("pearson", "spearman", "kendall"),
  alpha = 0.05
)
```

Arguments

d	A data frame.
vars	The names of two or more numeric columns.
method	"pearson", "spearman" or "kendall".
alpha	The significance level, between 0 and 0.5.

Details

Pearson's p-value is the usual t test. Spearman's and Kendall's are exact whenever the distinct orderings of one variable's values can be listed (up to 50,000, which covers eight rows and many more when values are tied); with ties and more orderings than that, Spearman's at any size and Kendall's up to 20 rows are estimated from 20,000 random orderings (200,000 near a significance threshold; a fixed seed, and the user's random number stream is left as it was), and Kendall's above 20 rows use the large-sample approximation, which holds its level there; without ties they come

from [cor.test](#), exact for Kendall's tau below 50 rows and Spearman's rho at nine, and a large-sample approximation beyond. Strength uses Cohen's 0.1, 0.3 and 0.5 for r, carried to rho and tau through their relation to r for bivariate normal data (0.10, 0.29, 0.48 for rho; 0.06, 0.19, 0.33 for tau).

Value

A data frame with one row per pair: Variable1, Variable2, N, Coefficient, p, Mark, CI_lower and CI_upper (Pearson only; NA otherwise), Strength, P_method ("exact", "estimate" from random orderings, or "approximation"), Approximate (TRUE unless exact), Held (rows left out only because an entry waits for a data-check correction) and Reading. A pair that cannot be computed has NA coefficient and p-value and says why in Reading.

Examples

```
d <- demo_data("RCBD")
set.seed(1)
d$Height <- 60 + 0.8 * d$Yield + rnorm(nrow(d), 0, 2)
correlate_data(d, c("Yield", "Height"))
correlate_data(d, c("Yield", "Height"), method = "spearman")
```

demo_data

Example datasets for each design

Description

Generates a small, balanced example dataset for a design, so you can try DOEpro without your own data. These are the datasets behind the **Load example** button in the application.

Usage

```
demo_data(which)
```

Arguments

which	The example to build: "CRD", "RCBD", "LSD", "FRCBD" (factorial), "SPLIT", "STRIP", "POOLRCBD" or "POOLCRD" (a single treatment factor over environments), "POOLFACT" or "POOLFACTC" (factorial treatments over environments, RCBD and CRD base respectively).
-------	---

Value

A data frame in long format, ready to pass to [analyze](#).

Examples

```
head(demo_data("RCBD"))
str(demo_data("SPLIT"))
```

describe_data	<i>Descriptive statistics for numeric variables</i>
---------------	---

Description

For each variable: the number of values (N), the number missing, the mean, standard deviation (SD), minimum, first quartile (Q1), median, third quartile (Q3), maximum, coefficient of variation (CV, per cent) and skewness, optionally for each level of a grouping column. Quartiles are calculated as Excel's QUARTILE.INC does (quantile(type = 7)) and skewness as Excel's SKEW does (the adjusted Fisher-Pearson coefficient). Text entries are read as the data check reads them: an entry that is not plainly a number (such as 5,6 or 12a) counts as missing; see [check_data](#).

Usage

```
describe_data(d, vars, group = NULL)
```

Arguments

- d A data frame.
- vars The names of the numeric columns to describe.
- group Optionally, the name of a column whose levels split each variable into groups. Rows without a label in it are left out.

Value

A data frame with one row per variable (and group): Variable, Group (when grouped), N, Missing, Mean, SD, Min, Q1, Median, Q3, Max, CV and Skewness. CV is NA when any value is negative or the mean is not positive, where it has no useful meaning; skewness needs three values that are not all equal. Missing counts empty cells, entries that are not numbers and infinite values.

Examples

```
describe_data(demo_data("RCBD"), "Yield")
describe_data(demo_data("RCBD"), "Yield", group = "Variety")
```

DESIGNS	<i>The designs DOEpro supports</i>
---------	------------------------------------

Description

A named character vector: the names are the labels shown in the application and the values are the design codes passed to [analyze](#).

Usage

```
DESIGNS
```

Format

A named character vector of length 11.

Value

A named character vector of length 11. Each element is the design code passed to the design argument of [analyze](#) or [run_all](#); the corresponding name is the label shown in the application.

Examples

```
DESIGNS
DESIGNS[["Split Plot Design"]]
```

fa_data

Factor analysis of numeric variables

Description

Maximum-likelihood factor analysis ([factanal](#)) of the chosen columns, from the rows with a value of every one, with the Kaiser-Meyer-Olkin measure of sampling adequacy (overall and for each variable) and Bartlett's test of sphericity. The number of factors, unless given, comes from Horn's parallel analysis of the eigenvalues of the correlation matrix (the same as on the principal components tab), within the most the data allow. Text entries are read as the data check reads them; see [check_data](#).

Usage

```
fa_data(
  d,
  vars,
  factors = NULL,
  rotation = c("varimax", "promax", "none"),
  alpha = 0.05
)

## S3 method for class 'doepro_fa'
print(x, ...)
```

Arguments

d	A data frame.
vars	The names of three or more numeric columns.
factors	The number of factors, or NULL for the number parallel analysis suggests.
rotation	"varimax" (the default, uncorrelated factors), "promax" (factors allowed to correlate) or "none".
alpha	The significance level for Bartlett's test and the test that the number of factors is enough.
x	A result of fa_data().
...	Not used.

Details

Each factor's sign is arbitrary; here it is chosen so that its largest loading is positive, and the factors are ordered by the variance they carry. With `rotation = "promax"` the loadings are pattern loadings and the factors may correlate; their correlations are returned.

Value

A list of class `doepro_fa`: `kmo` (overall) and `msa` (each variable), `bartlett` (statistic, df, p), `eigen` (eigenvalues of the correlation matrix with their parallel-analysis thresholds), `suggested` and `factors`, `loadings` (variables by factors), `communality`, `uniqueness`, `variance` (each factor's sum of squared loadings, proportion and cumulative proportion), `phi` (the factor correlations), `fit` (the test that the number of factors is enough), `scores` (regression scores, rows by factors), and the settings.

Examples

```
set.seed(1)
f1 <- rnorm(60); f2 <- rnorm(60)
d <- data.frame(Height = f1 + rnorm(60, 0, 0.5), Tillers = f1 + rnorm(60, 0, 0.5),
                Yield = f1 + rnorm(60, 0, 0.6), Protein = f2 + rnorm(60, 0, 0.5),
                Oil = f2 + rnorm(60, 0, 0.5), Moisture = f2 + rnorm(60, 0, 0.6))
fa <- fa_data(d, names(d))
fa$kmo
round(fa$loadings, 2)
```

Description

Makes only the corrections whose meaning is certain, entry by entry: decimal commas become points, thousands separators are removed, a per cent sign or a unit carried by every value in a column is dropped, entries such as n/a or - are emptied, extra spaces in labels are removed, and empty rows are deleted. A column becomes a column of numbers once every entry in it is a number. Entries whose meaning is not certain, such as 12a, nil, or 1,250 where the column does not show which notation it uses, are left exactly as they are, and labels that differ in capitals or punctuation are not merged.

Usage

```
fix_data(d)
```

Arguments

d A data frame, as loaded.

Value

A list with data, the corrected data frame (row names kept, so row numbers still match the data as loaded), and log, a character vector saying what was changed.

Examples

```
d <- data.frame(Variety = c("V1", "V1", "V2", "V2"),
                Yield = c("5,6", "6.1", "n/a", "7,2"))
fix_data(d)
```

pca_data

Principal component analysis of numeric variables

Description

Principal components of the chosen columns, from the rows with a value of every one ([prcomp](#)). By default each variable is standardised first, so the analysis works on their correlations and no variable dominates through its units; with `scale = FALSE` it works on their covariances, which suits variables measured in the same units. Text entries are read as the data check reads them: an entry that is not plainly a number counts as missing; see [check_data](#).

Usage

```
pca_data(d, vars, scale = TRUE)

## S3 method for class 'doepro_pca'
print(x, ...)
```

Arguments

d	A data frame.
vars	The names of two or more numeric columns.
scale	TRUE (the default) to standardise each variable, FALSE to analyse them as measured.
x	A result of <code>pca_data()</code> .
...	Not used.

Details

Each component's sign is arbitrary (reversing it changes nothing); here it is chosen so that the variable with the largest loading on it loads positively. The loadings are the correlations between each variable and each component. Components are kept by Horn's parallel analysis: those, from the first, whose eigenvalue exceeds the 95th percentile of the eigenvalues of random normal data of the same size (with the same variances for a covariance analysis), simulated with a fixed seed. Kaiser's rule (an eigenvalue above the average, which is 1 for correlations) is shown beside it.

Value

A list of class `doepro_pca`: `eigen` (a data frame: Component, Eigenvalue, Proportion, Cumulative, PA_threshold), `loadings` (variables by components, correlations), `vectors` (the eigenvectors), `scores` (rows by components), `rows` (the rows used), `kaiser` and `retained` (the number of components each rule keeps), and the settings.

Examples

```
set.seed(1)
d <- data.frame(Height = rnorm(30, 100, 10))
d$Tillers <- 0.1 * d$Height + rnorm(30)
d$Yield <- 0.3 * d$Height + 0.5 * d$Tillers + rnorm(30, 0, 3)
d$Protein <- rnorm(30, 12, 1)
p <- pca_data(d, c("Height", "Tillers", "Yield", "Protein"))
p$eigen
round(p$loadings, 2)
```

regress_data

Linear regression of one numeric variable on one or more others

Description

Fits response on the predictors by least squares ([lm](#)), using the rows with a value of every variable. One predictor gives a simple linear regression, two or more a multiple regression. Text entries are read as the data check reads them: an entry that is not plainly a number counts as missing; see [check_data](#).

Usage

```
regress_data(d, response, predictors, alpha = 0.05)

## S3 method for class 'doepro_regression'
print(x, ...)
```

Arguments

d	A data frame.
response	The name of the numeric column to be predicted.
predictors	The names of one or more numeric columns to predict it from.
alpha	The significance level, between 0 and 0.5.
x	A result of regress_data().
...	Not used.

Details

The coefficient table gives each estimate with its standard error, t statistic, two-sided p-value, a significance mark (* at alpha, ** at alpha / 5, NS otherwise) and a confidence interval at level $1 - \alpha$; with two or more predictors, each slope's variance inflation factor. The model summary gives N, R-squared, adjusted R-squared, the residual mean square (MSE, the residual sum of squares divided by its $N - k - 1$ degrees of freedom), its square root (RMSE, the residual standard error), the mean absolute residual (MAE), and the F test of the whole model. Three checks on the residuals go with them: Shapiro-Wilk's test of normality; the Breusch-Pagan test (Koenker's studentised form) of a spread that changes with the predictors (the squared residuals on the predictors, k degrees of freedom); and a test of a curve, Ramsey's RESET (with one predictor its square added to the model, with several the square of the fitted values). With up to 200 rows the p-values of Shapiro-Wilk and Breusch-Pagan come from 2,000 sets of normal data simulated on the same predictor values (10,000 near a significance threshold), since for residuals from few rows the usual ones can be far off; with one residual degree of freedom both are not available.

Value

A list of class `doepro_regression`: equation (the fitted equation as printed), coefficients (a data frame: Term, Estimate, SE, t, p, Mark, CI_lower, CI_upper and, with two or more predictors, VIF), model (a one-row data frame: N, R2, Adj_R2, MSE, RMSE, MAE, F, df1, df2, p, Mark), anova (the regression, residual and total sums of squares), checks (the three residual checks: Check, Test, Statistic, df, p, Simulated (whether the p-value came from simulation) and Verdict, which gives the reason when a check is not available), rows (the rows used), fit (the lm fit, on columns y, x1, x2, ...: the predictors centred and scaled by centre and scale), built (predictors that are powers or products of others), and the settings. With an exact fit (every residual zero) the standard errors, tests and intervals are NA.

Examples

```
d <- demo_data("RCBD")
set.seed(1)
d$Height <- 60 + 0.8 * d$Yield + rnorm(nrow(d), 0, 2)
```

```

r <- regress_data(d, "Height", "Yield")
r$equation
r$coefficients
r$model

```

run_all

*Analyse several responses from a designed experiment***Description**

Runs [analyze](#) for every response variable in turn, using the same design and mapping, and optionally applying a variance-stabilising transformation to each. This is what the application calls when more than one response is selected.

Usage

```
run_all(d, design, map, responses, alpha = 0.05, trans = NULL, dtype = "auto")
```

Arguments

d	A data frame in long format.
design	The design code; see DESIGNS .
map	A named list mapping roles to columns; see analyze . The response element is set for each response in turn and need not be supplied.
responses	A character vector of response column names.
alpha	The significance level. Defaults to 0.05.
trans	Either NULL (analyse every response untransformed) or a named list or character vector giving a transformation for each response. The names are the response columns and the values are keys of TRANS , for example <code>list(Incidence = "arcsine")</code> .
dtype	Passed to the transformation adviser; "auto" lets DOEpro judge the type of each response from its name and its mean-variance behaviour.

Value

A list with `fits` (one entry per response, each containing the fitted analysis in `final`), together with the design, the mapping and the significance level.

Examples

```

d <- demo_data("FRCBD")
rr <- run_all(d, "FRCBD",
              list(factors = c("Nitrogen", "Variety"), block = "Block"),
              responses = "Yield")
rr$fits[["Yield"]]$final$anova

```

run_DOEpro

Run the DOEpro application

Description

Launches the DOEpro Shiny application, which analyses designed agricultural experiments: paste or upload your data, choose the design, and press **Run analysis**.

Usage

```
run_DOEpro(...)
```

Arguments

... Further arguments passed to [shinyApp](#), for example `options = list(port = 8080)`.

Details

The application returns the analysis of variance, tables of means with standard errors and critical differences, checks of the assumptions with advice on transformations, post-hoc comparisons, diagnostic plots, a plain-English interpretation, and a downloadable report.

Value

An object of class `shiny.appobj`. Called for its side effect of starting the application.

Examples

```
if (interactive()) {
  run_DOEpro()
}
```

TRANS

The transformations DOEpro offers

Description

A named list of the variance-stabilising transformations. Each entry holds a label (`lab`), the transformation (`f`) and its inverse (`inv`), which is what lets DOEpro report means on the original scale of measurement alongside the transformed ones.

Usage

```
TRANS
```

Format

A named list of length 9. The names are the keys used in the `trans` argument of `run_all`: "none", "log", "log1", "sqrt", "sqrt0.5", "arcsine", "arcsine01", "reciprocal" and "boxcox".

Value

A named list of length 9. Each element is itself a list with `lab`, the label shown in the application; `f`, a function applying the transformation; and `b`, a function applying its inverse, which is what allows means to be reported on the original scale of measurement.

Examples

```
names(TRANS)
vapply(TRANS, `[`, character(1), "lab")
```

Index

analyze, [2](#), [8–10](#), [15](#)

build_report, [3](#), [4](#)

check_data, [5](#), [5](#), [7](#), [9](#), [10](#), [12](#), [13](#)

cluster_data, [5](#)

cor.test, [8](#)

correlate_data, [7](#)

demo_data, [8](#)

describe_data, [9](#)

DESIGNS, [2](#), [9](#), [15](#)

fa_data, [10](#)

factanal, [10](#)

fix_data, [5](#), [11](#)

hclust, [5](#)

lm, [13](#)

pca_data, [12](#)

prcomp, [12](#)

print.doepr_cluster(cluster_data), [5](#)

print.doepr_fa(fa_data), [10](#)

print.doepr_pca(pca_data), [12](#)

print.doepr_regression(regress_data),
[13](#)

regress_data, [13](#)

run_all, [3](#), [4](#), [10](#), [15](#), [17](#)

run_D0Epro, [16](#)

shinyApp, [16](#)

TRANS, [15](#), [16](#)