Package ‘snazzieR’

May 11, 2025

Type Package

Title Chic and Sleek Functions for Beautiful Statisticians

Version 0.1.1

Maintainer Aidan J. Wagner <JesusButForGayPeople@proton.me>

Description Because your linear models deserve better than console output.
A sleek color palette and kable styling to make your regression results look sharper than they are.
Includes support for Partial Least Squares (PLS) regression via both the SVD and NIPALS algorithms,
along with a unified interface for model fitting and fabulous LaTeX and console output formatting.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports ggplot2, knitr, kableExtra, dplyr, stats

RoxygenNote 7.3.2

Depends R (>= 4.1.0)

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation no

Author Aidan J. Wagner [aut, cre]

Contents

ANOVA.summary.table *Generate a Summary Table for ANOVA Results*

Description

This function creates a summary table for ANOVA results, including degrees of freedom, sum of squares, mean squares, F-values, and p-values. The table is formatted for LaTeX output using the ‘kableExtra‘ package.

Usage

```
ANOVA.summary.table(model, caption)
```

Arguments

`model`	A model object for which ANOVA results are computed (e.g., output from ‘lm()‘ or ‘aov()‘).
`caption`	A character string to be used as the caption for the table.

Value

A LaTeX-formatted table generated by ‘kableExtra::kable()‘.

Examples

```
# Fit a linear model
model <- lm(mpg ~ wt + hp, data = mtcars)

# Generate the ANOVA summary table
ANOVA.summary.table(model, caption = "ANOVA Summary")
```

color.ref *Display a Color Reference Palette*

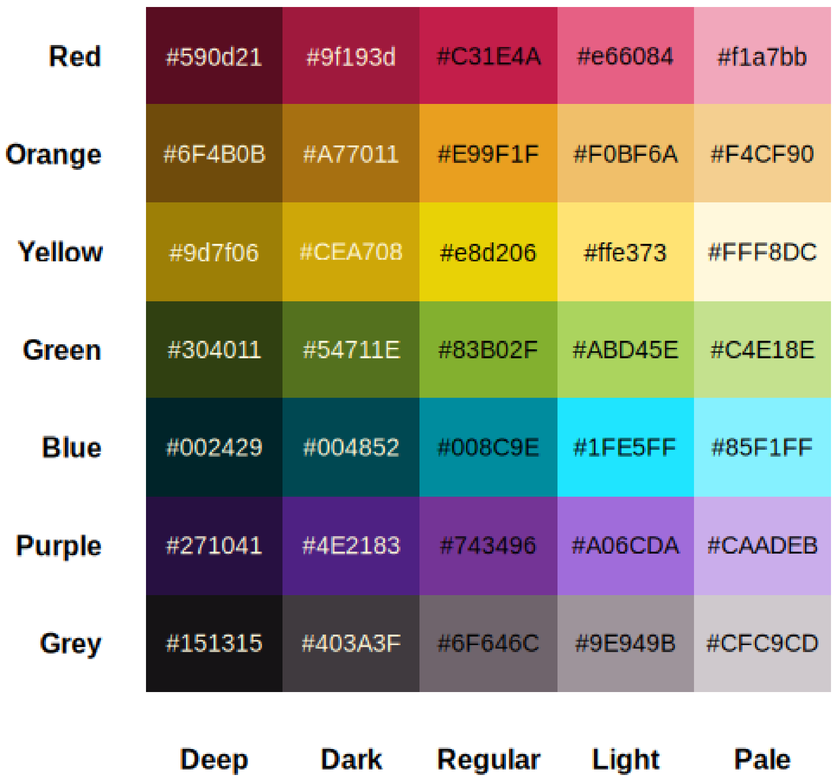
Description

This function generates a plot displaying a predefined color palette with color codes for easy reference. The palette includes shades of Red, Orange, Yellow, Green, Blue, Purple, and Grey.

Usage

```
color.ref()
```

Details



Value

A plot displaying the color palette.

Examples

```
color.ref()
```

`colors` *SnazzieR Color Palette*

Description

A collection of named hex colors grouped by hue and tone. Each color is available as an exported object (e.g., `Red`, `Dark.Red`).

Usage

```
color.list
```

Format

Each color is a character string representing a hex code.

An object of class `character` of length 1.

An object of class `list` of length 35.

Details

This palette consists of named hex colors. Each color's name (e.g., `Dark.Red`) is available as an exported object.

Swatch images are embedded below (not selectable):

Color	Swatch	Color	Swatch	Color	Swatch
Deep.Red		Deep.Green		Deep.Grey	
Dark.Red		Dark.Green		Dark.Grey	
Red		Green		Grey	
Light.Red		Light.Green		Light.Grey	
Pale.Red		Pale.Green		Pale.Grey	
Deep.Orange		Deep.Blue			
Dark.Orange		Dark.Blue			
Orange		Blue			
Light.Orange		Light.Blue			
Pale.Orange		Pale.Blue			
Deep.Yellow		Deep.Purple			
Dark.Yellow		Dark.Purple			
Yellow		Purple			
Light.Yellow		Light.Purple			
Pale.Yellow		Pale.Purple			

For the full list and hex codes, use `names(color.list)` or see `?color.list`.

See Also

`color.list`, `color.ref`

`eigen.summary` *Summarize Eigenvalues and Eigenvectors of a Covariance Matrix*

Description

This function computes the eigenvalues and eigenvectors of a given covariance matrix, ensures sign consistency in the eigenvectors, and outputs a formatted LaTeX table displaying the results.

Usage

```
eigen.summary(
  cov.matrix,
  caption = "Eigenvectors of Covariance Matrix",
  space_after_caption = "5mm"
)
```

Arguments

`cov.matrix`	A square numeric matrix representing the covariance matrix.
`caption`	A character string specifying the table caption (default: "Eigenvectors of Covariance Matrix").
`space_after_caption`	A character string specifying the space after the caption in LaTeX (default: "5mm").

Value

A LaTeX formatted table displaying the eigenvectors and eigenvalues.

Examples

```
cov_matrix <- matrix(c(4, 2, 2, 3), nrow = 2)
eigen.summary(cov_matrix)
```

`format.pls` *Format PLS Model Output as LaTeX or Console Tables*

Description

Formats and displays Partial Least Squares (PLS) model output from `pls.regression()` as either LaTeX tables (for PDF rendering) or console-friendly output.

Usage

```
## S3 method for class 'pls'
format(x, ..., include.scores = TRUE, latex = FALSE)
```

Arguments

`x`	A list returned by `pls.regression()` (class `"pls"`) containing PLS model components.
`...`	Further arguments passed to or from methods (unused).
`include.scores`	Logical. Whether to include score matrices (T and U). Default is `TRUE`.
`latex`	Logical. If `TRUE`, produces LaTeX output (for PDF rendering). If `FALSE`, prints to console. Default is `FALSE`.

Value

When `latex = TRUE`, returns a `knitr::asis_output` object (LaTeX code). When `FALSE`, prints formatted tables to console.

`model.equation` *Generate a Model Equation from a Linear Model*

Description

This function extracts and formats the equation from a linear model object. It includes an option to return the equation as a LaTeX-formatted string or print it to the console.

Usage

```
model.equation(model, latex = TRUE)
```

Arguments

`model`	A linear model object (e.g., output from ‘lm()‘).
`latex`	A logical value indicating whether to return a LaTeX-formatted equation (default: TRUE). If FALSE, the equation is printed to the console.

Value

If ‘latex‘ is TRUE, the equation is returned as LaTeX code using ‘knitr::asis_output()‘. If FALSE, the equation is printed to the console.

Examples

```
# Fit a linear model
model <- lm(mpg ~ wt + hp, data = mtcars)

# Get LaTeX equation
model.equation(model)

# Print equation to console
model.equation(model, latex = FALSE)
```

model.summary.table — *Generate a Summary Table for a Linear Model*

Description

This function creates a summary table for a linear model, including coefficients, standard errors, p-values, and model statistics (e.g., MSE, R-squared). The table is formatted for LaTeX output using the ‘kableExtra‘ package.

Usage

```
model.summary.table(model, caption)
```

Arguments

`model`	A linear model object (e.g., output from ‘lm()‘).
`caption`	A character string to be used as the caption for the table.

Value

A LaTeX-formatted table generated by ‘kableExtra::kable()‘.

Examples

```
# Fit a linear model
model <- lm(mpg ~ wt + hp, data = mtcars)

# Generate the summary table
model.summary.table(model, caption = "Linear Model Summary")
```

NIPALS.pls — *Partial Least Squares Regression via NIPALS (Internal)*

Description

This function is called internally by `pls.regression` and is not intended to be used directly. Use `pls.regression(..., calc.method = "NIPALS")` instead.

Performs Partial Least Squares (PLS) regression using the NIPALS (Nonlinear Iterative Partial Least Squares) algorithm. This method estimates the latent components (scores, loadings, weights) by iteratively updating the X and Y score directions until convergence. It is suitable for cases where the number of predictors is large or predictors are highly collinear.

Usage

```
NIPALS.pls(x, y, n.components = NULL)
```

Arguments

`x` A numeric matrix or data frame of predictors (X). Should have dimensions n × p.

`y` A numeric matrix or data frame of response variables (Y). Should have dimensions n × q.

`n.components` Integer specifying the number of PLS components to extract. If NULL, it defaults to `qr(x)$rank`.

Details

The algorithm standardizes both `x` and `y` using z-score normalization. It then performs the following for each of the `n.components` latent variables:

1. Initializes a random response score vector u.
2. Iteratively:
   - Updates the X weight vector $w = E^\top u$, normalized.
   - Computes the X score $t = Ew$, normalized.
   - Updates the Y loading $q = F^\top t$, normalized.
   - Updates the response score $u = Fq$.
   - Repeats until t converges below a tolerance threshold.
3. Computes scalar regression coefficient $b = t^\top u$.
4. Deflates residual matrices E and F to remove current component contribution.

After component extraction, the final regression coefficient matrix $B_{original}$ is computed and rescaled to the original data units. Explained variance is also computed component-wise and cumulatively.

Value

A list with the following elements:

model.type Character string indicating the model type ("PLS Regression").

T Matrix of X scores (n × H).

U Matrix of Y scores (n × H).

W Matrix of X weights (p × H).

C Matrix of normalized Y weights (q × H).

P_loadings Matrix of X loadings (p × H).

Q_loadings Matrix of Y loadings (q × H).

B_vector Vector of regression scalars (length H), one for each component.

coefficients Matrix of regression coefficients in original data scale (p × q).

intercept Vector of intercepts (length q). Always zero here due to centering.

X_explained Percent of total X variance explained by each component.

Y_explained Percent of total Y variance explained by each component.

X_cum_explained Cumulative X variance explained.

Y_cum_explained Cumulative Y variance explained.

References

Wold, H., & Lyttkens, E. (1969). Nonlinear iterative partial least squares (NIPALS) estimation procedures. *Bulletin of the International Statistical Institute*, 43, 29–51.

Examples

```
## Not run:
X <- matrix(rnorm(100 * 10), 100, 10)
Y <- matrix(rnorm(100 * 2), 100, 2)
model <- pls.regression(X, Y, n.components = 3, calc.method = "NIPALS")
model$coefficients

## End(Not run)
```

`pls.regression` *Partial Least Squares (PLS) Regression Interface*

Description

Performs Partial Least Squares (PLS) regression using either the NIPALS or SVD algorithm for component extraction. This is the main user-facing function for computing PLS models. Internally, it delegates to either `NIPALS.pls()` or `SVD.pls()`.

Usage

```
pls.regression(x, y, n.components = NULL, calc.method = c("SVD", "NIPALS"))
```

Arguments

`x`	A numeric matrix or data frame of predictor variables (X), with dimensions n × p.
`y`	A numeric matrix or data frame of response variables (Y), with dimensions n × q.
`n.components`	Integer specifying the number of latent components (H) to extract. If NULL, defaults to the rank of `x`.
`calc.method`	Character string indicating the algorithm to use. Must be either `"SVD"` (default) or `"NIPALS"`.

Details

This function provides a unified interface for Partial Least Squares regression. Based on the value of `calc.method`, it computes latent variables using either:

- `"SVD"` — A direct method using the singular value decomposition of the cross-covariance matrix ($X^\top Y$).
- `"NIPALS"` — An iterative method that alternately estimates predictor and response scores until convergence.

The outputs from both methods include scores, weights, loadings, regression coefficients, and explained variance.

Value

A list (from either `SVD.pls()` or `NIPALS.pls()`) containing:

model.type Character string ("PLS Regression").

T, U Score matrices for X and Y.

W, C Weight matrices for X and Y.

P_loadings, Q_loadings Loading matrices.

B_vector Component-wise regression weights.

coefficients Final regression coefficient matrix (rescaled).

intercept Intercept vector (typically zero due to centering).

X_explained, Y_explained Variance explained by each component.

X_cum_explained, Y_cum_explained Cumulative variance explained.

References

Abdi, H., & Williams, L. J. (2013). Partial least squares methods: Partial least squares correlation and partial least square regression. *Methods in Molecular Biology (Clifton, N.J.)*, 930, 549–579. doi:10.1007/9781627030595_23

de Jong, S. (1993). SIMPLS: An alternative approach to partial least squares regression. *Chemometrics and Intelligent Laboratory Systems*, 18(3), 251–263. doi:10.1016/01697439(93)85002X

See Also

`SVD.pls`, `NIPALS.pls`

Examples

```
## Not run:
X <- matrix(rnorm(100 * 10), 100, 10)
Y <- matrix(rnorm(100 * 2), 100, 2)

# Using SVD (default)
model1 <- pls.regression(X, Y, n.components = 3)

# Using NIPALS
model2 <- pls.regression(X, Y, n.components = 3, calc.method = "NIPALS")

## End(Not run)
```

`snazzieR.theme` *A Custom ggplot2 Theme for Publication-Ready Plots*

Description

This theme provides a clean, polished look for ggplot2 plots, with a focus on readability and aesthetics. It includes a custom color palette and formatting for titles, axes, and legends.

Usage

```
snazzieR.theme()
```

Value

A ggplot2 theme object.

Examples

```
library(ggplot2)
ggplot(mtcars, aes(x = wt, y = mpg)) +
  geom_point() +
  snazzieR.theme()
```

`SVD.pls` *Partial Least Squares Regression via SVD (Internal)*

Description

This function is called internally by `pls.regression` and is not intended to be used directly. Use `pls.regression(..., calc.method = "SVD")` instead.

Performs Partial Least Squares (PLS) regression using the Singular Value Decomposition (SVD) of the cross-covariance matrix. This method estimates the latent components by identifying directions in the predictor and response spaces that maximize their covariance, using the leading singular vectors of the matrix $R = X^\top Y$.

Usage

```
SVD.pls(x, y, n.components = NULL)
```

Arguments

`x`	A numeric matrix or data frame of predictors (X). Should have dimensions n × p.
`y`	A numeric matrix or data frame of response variables (Y). Should have dimensions n × q.
`n.components`	Integer specifying the number of PLS components to extract. If NULL, defaults to `qr(x)$rank`.

Details

The algorithm begins by z-scoring both `x` and `y` (centering and scaling to unit variance). The initial residual matrices are set to the scaled values: `E = X_scaled`, `F = Y_scaled`.

For each component h = 1, ..., H:

1. Compute the cross-covariance matrix $R = E^\top F$.
2. Perform SVD on $R = UDV^\top$.
3. Extract the first singular vectors: $w = U[,1]$, $q = V[,1]$.
4. Compute scores: $t = Ew$ (normalized), $u = Fq$.

5. Compute loadings: $p = E^\top t$, regression scalar $b = t^\top u$.
6. Deflate residuals: $E \leftarrow E - tp^\top$, $F \leftarrow F - btq^\top$.

After all components are extracted, a post-processing step removes components with zero regression weight. The scaled regression coefficients are computed using the Moore–Penrose pseudoinverse of the loading matrix P, and then rescaled to the original variable units.

Value

A list containing:

model.type Character string indicating the model type ("PLS Regression").

T Matrix of predictor scores (n × H).

U Matrix of response scores (n × H).

W Matrix of predictor weights (p × H).

C Matrix of normalized response weights (q × H).

P_loadings Matrix of predictor loadings (p × H).

Q_loadings Matrix of response loadings (q × H).

B_vector Vector of scalar regression weights (length H).

coefficients Matrix of final regression coefficients in the original scale (p × q).

intercept Vector of intercepts (length q). All zeros due to centering.

X_explained Percent of total X variance explained by each component.

Y_explained Percent of total Y variance explained by each component.

X_cum_explained Cumulative X variance explained.

Y_cum_explained Cumulative Y variance explained.

References

Abdi, H., & Williams, L. J. (2013). Partial least squares methods: Partial least squares correlation and partial least square regression. *Methods in Molecular Biology (Clifton, N.J.)*, 930, 549–579. doi:10.1007/9781627030595_23

de Jong, S. (1993). SIMPLS: An alternative approach to partial least squares regression. *Chemometrics and Intelligent Laboratory Systems*, 18(3), 251–263. doi:10.1016/01697439(93)85002X

Examples

```
## Not run:
X <- matrix(rnorm(100 * 10), 100, 10)
Y <- matrix(rnorm(100 * 2), 100, 2)
model <- pls.regression(X, Y, n.components = 3, calc.method = "SVD")
model$coefficients

## End(Not run)
```

Index