Package 'surreal'

May 9, 2026

Title Create Datasets with Hidden Images in Residual Plots

Version 0.0.2

Description Implements the ``Residual (Sur)Realism'' algorithm described by Stefanski (2007) <doi:10.1198/000313007X190079> to generate datasets that reveal hidden images or messages in their residual plots. It offers both predefined datasets and tools to embed custom text or images into residual structures. Allowing users to create intriguing visual demonstrations for teaching model diagnostics.

License GPL (>= 3)

Depends R (>= 4.3.0)

Encoding UTF-8

RoxygenNote 7.3.3

URL https://github.com/coatless-rpkg/surreal, https://r-pkg.thecoatlessprofessor.com/surreal/

BugReports https://github.com/coatless-rpkg/surreal/issues

LazyData true

Imports cli, png

Suggests bmp, bslib, jpeg, rsvg, shiny, tiff

NeedsCompilation no

Author James Joseph Balamuta [aut, cre, cph] (ORCID: <https://orcid.org/0000-0003-2826-8458>)

Maintainer James Joseph Balamuta <james.balamuta@gmail.com>

Repository CRAN

Date/Publication 2026-01-11 06:40:02 UTC

Contents

`border_augmentation` *Transform Data by Adding a Border*

Description

This function transforms the input data by adding points around the original data to create a frame. It uses an optimization process to find the best alpha parameter for point distribution, which helps in making the fitted values and residuals orthogonal.

Usage

```
border_augmentation(x, y, n_add_points = 40, verbose = FALSE)
```

Arguments

`x`	Numeric vector of x coordinates.
`y`	Numeric vector of y coordinates.
`n_add_points`	Integer. Number of points to add on each side of the frame. Default is `40`.
`verbose`	Logical. If `TRUE`, prints optimization progress. Default is `FALSE`.

Value

A matrix with two columns representing the transformed `x` and `y` coordinates.

Examples

```
# Simulate data
x <- rnorm(100)
y <- rnorm(100)

# Append border to data
transformed_data <- border_augmentation(x, y)

# Modify par settings for plotting side-by-side
oldpar <- par(mfrow = c(1, 2))

# Graph original and transformed data
plot(x, y, pch = 16, main = "Original data")
plot(
  transformed_data[, 1], transformed_data[, 2], pch = 16,
  main = "Transformed data", xlab = 'x', ylab = 'y'
```

```
)

# Restore original par settings
par(oldpar)
```

`jackolantern_surreal_data`
Jack-o'-Lantern Surreal Data

Description

Data set containing a hidden image of a Jack-o'-Lantern lurking in the residual plot of a full model being fit.

Usage

```
jackolantern_surreal_data
```

Format

A data frame with 5,395 observations and 7 variables.

- `y`: Response variable
- `x1`: Predictor variable 1
- `x2`: Predictor variable 2
- `x3`: Predictor variable 3
- `x4`: Predictor variable 4
- `x5`: Predictor variable 5
- `x6`: Predictor variable 6

References

Stefansk, L.A. (2013). Hidden Images in the Helen Barton Lecture Series. Retrieved from https://www4.stat.ncsu.edu/~stefansk/NSF_Supported/Hidden_Images/UNCG_Helen_Barton_Lecture_Nov_2013/pumpkin_1_data_yx1x6.txt

Examples

```
# Load the Jack-o'-Lantern data
data <- jackolantern_surreal_data

# Fit a linear model to the surreal Jack-o'-Lantern data
model <- lm(y ~ ., data = data)

# Plot the residuals to reveal the hidden image
plot(model$fitted, model$resid, type = "n", main = "Residual plot from transformed data")
points(model$fitted, model$resid, pch = 16)
```

r_logo_image_data *R Logo Pixel Data*

Description

2D data set with the shape of the R Logo in x and y coordinate pairings.

Usage

```
r_logo_image_data
```

Format

A data frame with 2,000 observations and 2 variables describing the x and y coordinates of the R logo.

References

Staudenmayer, J. (2007). Hidden Images in R. Retrieved from https://www4.stat.ncsu.edu/~stefansk/NSF_Supported/Hidden_Images/000_R_Programs/John_Staudenmayer/logo.txt

Examples

```
# Load the R logo data
data("r_logo_image_data", package = "surreal")

# Plot the R logo
plot(r_logo_image_data$x, r_logo_image_data$y, pch = 16, main = "R Logo", xlab = '', ylab = '')
```

surreal *Find X Matrix and Y Vector for Residual Surrealism*

Description

This function implements the Residual (Sur)Realism algorithm as described by Leonard A. Stefanski (2007). It finds a matrix X and vector y such that the fitted values and residuals of lm(y ~ X) are similar to the inputs y_hat and R_0.

Usage

```
surreal(
  data,
  y_hat = data[, 1],
  R_0 = data[, 2],
  R_squared = 0.3,
  p = 5,
```

```
  n_add_points = 40,
  max_iter = 100,
  tolerance = 0.01,
  verbose = FALSE
)
```

Arguments

data	A data frame or matrix with two columns representing the y_hat and R_0 values.
y_hat	Numeric vector of desired fitted values (only used if data is not provided).
R_0	Numeric vector of desired residuals (only used if data is not provided).
R_squared	Numeric. Desired R-squared value. Default is 0.3.
p	Integer. Desired number of columns for matrix X. Default is 5.
n_add_points	Integer. Number of points to add in border transformation. Default is 40.
max_iter	Integer. Maximum number of iterations for convergence. Default is 100.
tolerance	Numeric. Criteria for detecting convergence and stopping optimization early. Default is 0.01.
verbose	Logical. If TRUE, prints progress information. Default is FALSE.

Details

To disable the border augmentation, set n_add_points = 0.

Value

A data frame containing the generated X matrix and y vector.

References

Stefanski, L. A. (2007). Residual (Sur)Realism. The American Statistician, 61(2), 163-177.

Examples

```
# Generate a 2D data set
data <- cbind(y_hat = rnorm(100), R_0 = rnorm(100))

# Display original data
plot(data, pch = 16, main = "Original data")

# Apply the surreal method
result <- surreal(data)

# View the expanded data after transformation
pairs(y ~ ., data = result, main = "Data after transformation")

# Fit a linear model to the transformed data
model <- lm(y ~ ., data = result)
```

```
# Plot the residuals
plot(model$fitted, model$resid, type = "n", main = "Residual plot from transformed data")
points(model$fitted, model$resid, pch = 16)
```

`surreal_app` *Launch the Surreal Shiny App*

Description

Opens an interactive Shiny application for exploring the surreal algorithm. The app allows you to generate datasets with hidden images in residual plots using demo data, custom text, or uploaded images.

Usage

```
surreal_app(launch.browser = TRUE, port = NULL, host = "127.0.0.1")
```

Arguments

`launch.browser`	Logical. If `TRUE` (default), opens the app in the default web browser. If `FALSE`, returns the app URL for manual opening.
`port`	Integer. The port to run the app on. If `NULL` (default), Shiny will choose an available port.
`host`	Character. The host address. Default is `"127.0.0.1"` (localhost).

Details

The app provides:

- Demo datasets (Jack-o-Lantern, R Logo)
- Custom text input to embed messages in residual plots
- Image upload support (PNG, JPEG, BMP, TIFF, SVG)
- Interactive controls for R^2, predictors, and image processing settings
- Dark/light mode toggle
- Data export to CSV

Value

This function is called for its side effect of launching the Shiny app. It does not return a value.

Requirements

The app requires the **shiny** and **bslib** packages to be installed. For image uploads, additional packages may be needed depending on the format:

- JPEG: **jpeg**
- BMP: **bmp**
- TIFF: **tiff**
- SVG: **rsvg**

See Also

`surreal()` for the core algorithm. `surreal_text()` for embedding text programmatically. `surreal_image()` for processing images programmatically.

Examples

```
## Not run:
# Launch the app in the default browser
surreal_app()

# Launch on a specific port
surreal_app(port = 3838)

# Get the app without launching browser
surreal_app(launch.browser = FALSE)

## End(Not run)
```

---

surreal_image *Apply the surreal method to an image file*

Description

This function loads an image file, extracts pixel coordinates based on a brightness threshold, and applies the surreal method to create a dataset where the image appears in the residual plot.

Usage

```
surreal_image(
  image_path,
  mode = "auto",
  threshold = NULL,
  max_points = NULL,
  invert_y = TRUE,
  R_squared = 0.3,
  p = 5,
```

```
  n_add_points = 40,
  max_iter = 100,
  tolerance = 0.01,
  verbose = FALSE
)
```

Arguments

image_path	Character. Path to an image file or a URL (PNG, JPEG, BMP, TIFF, or SVG).
mode	Character. Either "auto" (default) to automatically detect, "dark" to select dark pixels, or "light" to select light pixels.
threshold	Numeric or NULL. Value between 0 and 1 for grayscale threshold. If NULL (default), automatically calculated using Otsu's method. For "dark" mode, pixels below threshold are selected. For "light" mode, pixels above threshold are selected.
max_points	Integer or NULL. Maximum number of points to use. If NULL (default), automatically estimated based on image size (typically 2000-5000 points). Set to Inf to use all points without downsampling.
invert_y	Logical. If TRUE, flip y-coordinates so image appears right-side up in residual plot. Default is TRUE.
R_squared	Numeric. Desired R-squared value. Default is 0.3.
p	Integer. Desired number of columns for matrix X. Default is 5.
n_add_points	Integer. Number of points to add in border transformation. Default is 40.
max_iter	Integer. Maximum number of iterations for convergence. Default is 100.
tolerance	Numeric. Criteria for detecting convergence and stopping optimization early. Default is 0.01.
verbose	Logical. If TRUE, prints progress information. Default is FALSE.

Details

By default, all parameters are automatically detected:

- **mode**: Detected from image histogram (dark subject on light background or vice versa)
- **threshold**: Calculated using Otsu's method to optimally separate foreground/background
- **max_points**: Estimated based on image dimensions (2000-5000 points)

You can override any of these by specifying explicit values.

Input Support:

- Local file paths
- URLs (http:// or https://) - images are downloaded to a temporary file

Format Support:

- PNG: Supported via the png package (included)
- JPEG: Requires the jpeg package

- BMP: Requires the bmp package
- TIFF: Requires the tiff package
- SVG: Requires the rsvg package (renders vector graphics to bitmap)

Value

A data.frame containing the results of the surreal method application with columns y, X1, X2, ..., Xp.

See Also

surreal() for details on the surreal method parameters. surreal_text() for embedding text instead of images.

Examples

```
## Not run:
# Simplest usage - everything auto-detected
result <- surreal_image("https://www.r-project.org/logo/Rlogo.png")
model <- lm(y ~ ., data = result)
plot(model$fitted, model$residuals, pch = 16)

# Override specific parameters
result <- surreal_image("image.png", mode = "dark", threshold = 0.3)

# Use all points (no downsampling)
result <- surreal_image("image.png", max_points = Inf)

## End(Not run)
```

---

surreal_text *Apply the surreal method to a text string*

Description

This function applies the surreal method to a text string. It first creates a temporary plot with the text, processes the image, and then applies the surreal method to the data.

Usage

```
surreal_text(
  text = "hello world",
  cex = 4,
  R_squared = 0.3,
  p = 5,
  n_add_points = 40,
  max_iter = 100,
```

```
  tolerance = 0.01,
  verbose = FALSE
)
```

Arguments

text	Character. A plain text message to be plotted. Default is "hello world".
cex	Numeric. A value specifying the relative size of the text. Default is 4.
R_squared	Numeric. Desired R-squared value. Default is 0.3.
p	Integer. Desired number of columns for matrix X. Default is 5.
n_add_points	Integer. Number of points to add in border transformation. Default is 40.
max_iter	Integer. Maximum number of iterations for convergence. Default is 100.
tolerance	Numeric. Criteria for detecting convergence and stopping optimization early. Default is 0.01.
verbose	Logical. If TRUE, prints progress information. Default is FALSE.

Value

A data.frame containing the results of the surreal method application.

See Also

surreal() for details on the surreal method parameters.

Examples

```
# Create a surreal plot of the text "R is fun" appearing on one line
r_is_fun_result <- surreal_text("R is fun", verbose = TRUE)

# Create a surreal plot of the text "Statistics Rocks" by using an escape
# character to create a second line between "Statistics" and "Rocks"
stat_rocks_result <- surreal_text("Statistics\nRocks", verbose = TRUE)
```

Index