

Package ‘surreal’

October 9, 2026

Title Create Datasets with Hidden Images in Residual Plots

Version 0.0.3

Description Implements the ``Residual (Sur)Realism" algorithm described by Stefanski (2007) <[doi:10.1198/000313007X190079](https://doi.org/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, testthat (>= 3.0.0), tiff,
withr

Config/testthat/edition 3

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 2026-10-09 16:50:02 UTC

Contents

border_augmentation	2
jackolantern_surreal_data	3
plot.surreal_path	4
plot.surreal_trace	5
r_logo_image_data	6
surreal	7
surreal_app	8
surreal_decoys	10
surreal_image	11
surreal_image_points	13
surreal_path	14
surreal_text	15
surreal_text_points	17
surreal_trace	17
Index	20

border_augmentation	<i>Transform Data by Adding a Border</i>
---------------------	--

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

Stefanski, 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)
```

plot.surreal_path	<i>Plot a Recorded Selection</i>
-------------------	----------------------------------

Description

Draws one step of a selection recorded by `surreal_path()`, as a row of panels: the share each step explained, the criterion along the path, and the residual plot of the model at the step.

Usage

```
## S3 method for class 'surreal_path'
plot(x, step = x$best, panels = c("explained", "criterion", "residuals"), ...)
```

Arguments

<code>x</code>	A <code>surreal_path</code> object.
<code>step</code>	Integer. The step to draw, from 0 to the number of predictors. Default is the best step.
<code>panels</code>	Character. The panels to draw, in order, from "explained", "coefficients", "criterion" and "residuals". Default is all but "coefficients". Four panels are drawn two to a row.
<code>...</code>	Further arguments passed to <code>plot()</code> for the residual plot.

Details

The panels are:

- "explained": a bar for each step, on a log scale, for the share of the variation left before the step that its predictor explained. A dotted line marks the criterion's charge for a predictor: the least a step has to explain for the criterion to fall. The bars above it are the steps that improved the model.

- "coefficients": the coefficient of every predictor along the path.
- "criterion": the criterion along the path.
- "residuals": the residual plot of the model at the step.

A solid violet line marks the step that is drawn, and a dashed green line the best step. A predictor is blue once it is in the model at the step, and gray until then. A decoy in the model is orange, when the path knows its decoys.

Value

x, invisibly. Called for the plot it draws.

Examples

```
set.seed(114)
hidden <- surreal(r_logo_image_data)
path <- surreal_path(surreal_decoys(hidden, n = 20))

plot(path)
plot(path, step = 25)

# The coefficient paths, with the criterion beside them
plot(path, panels = c("coefficients", "criterion"))
```

plot.surreal_trace	<i>Plot a Recorded Search</i>
--------------------	-------------------------------

Description

Draws the picture as it stood at one iteration of a search recorded by `surreal_trace()`, or the path the search took.

Usage

```
## S3 method for class 'surreal_trace'
plot(x, iteration = nrow(x$iterations), type = c("picture", "trace"), ...)
```

Arguments

x	A <code>surreal_trace</code> object.
iteration	Integer. The iteration to draw, or to mark on the path. Default is the last one.
type	Character. "picture" (default) plots the fitted values of the iteration against the residuals. "trace" plots the distance of the fitted values from their targets at every iteration, on a log scale.
...	Further arguments passed to <code>plot()</code> .

Value

x, invisibly. Called for the plot it draws.

Examples

```
set.seed(114)
trace <- surreal_trace(r_logo_image_data)

plot(trace, iteration = 2)
plot(trace, type = "trace")
```

r_logo_image_data	<i>R Logo Pixel Data</i>
-------------------	--------------------------

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 $\text{lm}(y \sim X)$ 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

<code>data</code>	A data frame or matrix with two columns representing the <code>y_hat</code> and <code>R_0</code> values.
<code>y_hat</code>	Numeric vector of desired fitted values (only used if data is not provided).
<code>R_0</code>	Numeric vector of desired residuals (only used if data is not provided).
<code>R_squared</code>	Numeric. Desired R-squared value. Default is 0.3.
<code>p</code>	Integer. Desired number of columns for matrix X . Default is 5.
<code>n_add_points</code>	Integer. Number of points to add in border transformation. Default is 40.
<code>max_iter</code>	Integer. Maximum number of iterations for convergence. Default is 100.
<code>tolerance</code>	Numeric. Criteria for detecting convergence and stopping optimization early. Default is 0.01.
<code>verbose</code>	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
- A Search tab that plays back the search that makes the data, one iteration at a time
- A Selection tab that hides the image among decoy predictors and selects the real ones one step at a time

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. [surreal_trace\(\)](#) and [surreal_path\(\)](#) for the functions behind the Search and Selection tabs.

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_decoys

*Add Decoy Predictors to a Surreal Dataset***Description**

This function adds predictors of pure noise to a dataset made by [surreal\(\)](#). The hidden image then shows only in the residuals of the model with the real predictors: leave some out and it is not there yet, and let decoys in and it blurs. Finding it becomes a variable selection problem, which [surreal_path\(\)](#) works through step by step.

Usage

```
surreal_decoys(data, n = 30, sd = NULL, shuffle = TRUE)
```

Arguments

data	A data frame from surreal() , surreal_text() or surreal_image() , with a response named y.
n	Integer. Number of decoy predictors to add. Default is 30.
sd	Numeric or NULL. Standard deviation of the decoys. If NULL (default), the average standard deviation of the predictors in data.
shuffle	Logical. If TRUE (default), the decoys are mixed in among the real predictors and every predictor is renamed X.1, X.2, ..., so that neither position nor name gives a decoy away. If FALSE, the decoys are added after the real predictors as D.1, D.2,

Value

The data frame with n more columns. The names of the decoys are kept in its "decoys" attribute, which is the answer key. The attribute is not written out by [write.csv\(\)](#).

See Also

[surreal_path\(\)](#) to select the predictors one step at a time.

Examples

```
set.seed(114)
hidden <- surreal(r_logo_image_data)

# Mix in 30 decoys
decoyed <- surreal_decoys(hidden)
names(decoyed)

# The answer key
attr(decoyed, "decoys")
```

```
# The model with every predictor no longer shows a clean image
model <- lm(y ~ ., data = decoyed)
plot(model$fitted, model$resid, pch = 16)
```

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. [surreal_image_points\(\)](#) for the points of the image on their own.

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_image_points *Turn an image into the points that draw it*

Description

This function loads an image file and returns the position of every pixel that passes a brightness threshold. These are the points that `surreal_image()` hides. Getting them first lets you look at them, change them, or combine them with other points before handing them to `surreal()`.

Usage

```
surreal_image_points(
  image_path,
  mode = "auto",
  threshold = NULL,
  max_points = NULL,
  invert_y = TRUE,
  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.
verbose	Logical. If TRUE, prints progress information. Default is FALSE.

Details

The mode, the threshold and the number of points are chosen automatically unless you set them, as described for `surreal_image()`.

Value

A data.frame with one row for each point of the image and two columns, x and y.

See Also

[surreal_image\(\)](#) to go from an image to a dataset in one step. [surreal_text_points\(\)](#) for the points of a text message.

Examples

```
## Not run:
# The points of the R logo
points <- surreal_image_points("https://www.r-project.org/logo/Rlogo.png")
plot(points, pch = 16, asp = 1)

# Hide them in a dataset, as surreal_image() does
result <- surreal(points)

## End(Not run)
```

surreal_path

Select Predictors One Step at a Time

Description

This function runs forward selection on a dataset made by [surreal\(\)](#) and keeps the model that stood at every step, so the selection can be plotted or played back. It starts with no predictors and adds, at each step, the one that explains most of what is left.

Usage

```
surreal_path(data, criterion = c("BIC", "AIC"))
```

Arguments

data	A data frame from surreal() , surreal_text() , surreal_image() or surreal_decoys() , with a response named y.
criterion	Character. The score that picks the best step, "BIC" (default) or "AIC". Lower is better.

Details

On data straight from [surreal\(\)](#) every predictor is real, so the criterion falls at each step and the hidden image appears at the last one. With decoys from [surreal_decoys\(\)](#) the criterion is lowest at the model of the real predictors, where the image is clear, and rises as decoys enter and blur it.

The criterion is computed as [BIC\(\)](#) and [AIC\(\)](#) compute it for a linear model.

Value

An object of class `surreal_path`, a list with:

steps A data frame with a row for each step, from 0 (no predictors) to the number of predictors: the predictor that entered, the criterion of the model and its `r_squared`.

coefficients A matrix with a row of coefficients for each step. A predictor that has not entered yet has a coefficient of 0.

fitted, residuals Matrices with a column for each step.

criterion The criterion that was used.

best The step with the lowest criterion.

decoys The names of the decoy predictors, when data came from `surreal_decoys()`.

The rows of coefficients and the columns of fitted and residuals are named by step, so `path$residuals[, "5"]` holds the residuals at step 5.

See Also

`surreal_decoys()` to add the decoys that make this a real search.

Examples

```
set.seed(114)
hidden <- surreal(r_logo_image_data)
decoyed <- surreal_decoys(hidden, n = 20)

path <- surreal_path(decoyed)
path

# The coefficient paths, the criterion and the residuals at the best step
plot(path)

# One step too early
plot(path, step = path$best - 1)
```

surreal_text

Apply the surreal method to a text string

Description

This function applies the surreal method to a text string. It first finds the points that draw the text with `surreal_text_points()`, and then applies the surreal method to them.

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. [surreal_text_points\(\)](#) for the points of the text on their own.

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)
```

surreal_text_points	<i>Turn text into the points that draw it</i>
---------------------	---

Description

This function draws the text on a temporary bitmap and returns the position of every pixel the text covers. These are the points that [surreal_text\(\)](#) hides. Getting them first lets you look at them, change them, or combine them with other points before handing them to [surreal\(\)](#).

Usage

```
surreal_text_points(text = "hello world", cex = 4)
```

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.

Value

A data.frame with one row for each point of the text and two columns, x and y.

See Also

[surreal_text\(\)](#) to go from text to a dataset in one step. [surreal_image_points\(\)](#) for the points of an image.

Examples

```
# The points that draw "R is fun"
points <- surreal_text_points("R is fun")
plot(points, pch = 16, asp = 1)

# Hide them in a dataset, as surreal_text() does
result <- surreal(points)
```

surreal_trace	<i>Record the Search Behind a Surreal Dataset</i>
---------------	---

Description

This function runs the same search as [surreal\(\)](#) and keeps what it had at every iteration, so the search can be plotted or played back. The data it ends on is the data [surreal\(\)](#) returns for the same seed.

Usage

```
surreal_trace(
  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,
  step = 1
)
```

Arguments

<code>data</code>	A data frame or matrix with two columns representing the <code>y_hat</code> and <code>R_0</code> values.
<code>y_hat</code>	Numeric vector of desired fitted values (only used if <code>data</code> is not provided).
<code>R_0</code>	Numeric vector of desired residuals (only used if <code>data</code> is not provided).
<code>R_squared</code>	Numeric. Desired R-squared value. Default is 0.3.
<code>p</code>	Integer. Desired number of columns for matrix <code>X</code> . Default is 5.
<code>n_add_points</code>	Integer. Number of points to add in border transformation. Default is 40.
<code>max_iter</code>	Integer. Maximum number of iterations for convergence. Default is 100.
<code>tolerance</code>	Numeric. Criteria for detecting convergence and stopping optimization early. Default is 0.01.
<code>step</code>	Numeric. The fraction of its proposed update that each iteration takes, from above 0 to 1. The default of 1 is the step <code>surreal()</code> takes. A smaller step slows the search down and gives more iterations to watch.

Details

The search starts from random predictors. The picture's vertical positions, the residuals, are exact from the first iteration: the predictors are built to be unrelated to them. The search moves the fitted values, the picture's horizontal positions. Each iteration rebuilds one predictor so that the fitted values land on their targets, which shifts the model slightly, so the next iteration corrects again. With the default step this settles in a handful of iterations.

Plotting the fitted values of an iteration against the residuals shows the picture as it stood then, which `plot()` does.

Value

An object of class `surreal_trace`, a list with:

data The data frame the search ended on.

iterations A data frame with a row for each iteration: its number, the change it proposed (what tolerance is compared with) and the distance of the fitted values from their targets.

fitted A matrix with a column of fitted values for each iteration.

residuals The residuals, which are the same at every iteration.

target The fitted values the search is aiming for.

step The step that was used.

converged Whether the search stopped because the change fell below tolerance, rather than running out of iterations.

See Also

[surreal\(\)](#) for the method itself.

Examples

```
set.seed(114)
trace <- surreal_trace(r_logo_image_data)
trace

# The picture at the first iteration, and where the search ended
oldpar <- par(mfrow = c(1, 2))
plot(trace, iteration = 1)
plot(trace)
par(oldpar)

# The distance of the fitted values from their targets, by iteration
plot(trace, type = "trace")

# A smaller step gives a longer search to watch
set.seed(114)
slow <- surreal_trace(r_logo_image_data, step = 0.25)
nrow(slow$iterations)
```

Index

* datasets

- jackolantern_surreal_data, [3](#)
- r_logo_image_data, [6](#)

AIC(), [14](#)

BIC(), [14](#)

border_augmentation, [2](#)

jackolantern_surreal_data, [3](#)

plot(), [4](#), [5](#), [18](#)

plot.surreal_path, [4](#)

plot.surreal_trace, [5](#)

r_logo_image_data, [6](#)

surreal, [7](#)

surreal(), [9](#), [10](#), [12–14](#), [16–19](#)

surreal_app, [8](#)

surreal_decoys, [10](#)

surreal_decoys(), [14](#), [15](#)

surreal_image, [11](#)

surreal_image(), [9](#), [10](#), [13](#), [14](#)

surreal_image_points, [13](#)

surreal_image_points(), [12](#), [17](#)

surreal_path, [14](#)

surreal_path(), [4](#), [9](#), [10](#)

surreal_text, [15](#)

surreal_text(), [9](#), [10](#), [12](#), [14](#), [17](#)

surreal_text_points, [17](#)

surreal_text_points(), [14–16](#)

surreal_trace, [17](#)

surreal_trace(), [5](#), [9](#)

write.csv(), [10](#)