

Package: ixsurface (via r-universe)

May 26, 2026

Title Interactive 3D Surface Plots for Multi-Factor Interaction
Visualization

Version 0.1.0

Description Visualize interactions between multiple experimental factors using interactive 3D surface plots powered by 'plotly'. Instead of examining combinatorial pairwise interaction plots, map factor combinations to response surfaces and use surface crossings as geometric indicators of interaction effects. Supports continuous, categorical, and mixed factor designs with automatic binning for continuous conditioning variables.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

URL <https://github.com/cjbrant/ixsurface>

BugReports <https://github.com/cjbrant/ixsurface/issues>

Imports plotly (>= 4.10.0), stats, utils, grDevices

Suggests testthat (>= 3.0.0), knitr, rmarkdown

Config/testthat/edition 3

VignetteBuilder knitr

Config/pak/sysreqs cmake make libicu-dev libuv1-dev libssl-dev

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

Date/Publication 2026-03-21 19:07:40 UTC

RemoteUrl <https://github.com/cjbrant/ixsurface>

RemoteRef HEAD

RemoteSha 4d416f81c7fcf26e55d52e48fe16d7813e0ed63d

Contents

bin_continuous	2
find_crossings	3
interaction_surface	4
interaction_surface_grid	6
make_prediction_grid	7
plot_crossings	8
sim_factorial	9

Index	11
--------------	-----------

bin_continuous	<i>Bin a continuous variable into discrete levels</i>
----------------	---

Description

Used when a continuous variable appears in facet_by to produce a manageable number of surfaces.

Usage

```
bin_continuous(x, n_bins = 3, method = c("quantile", "equal", "pretty"))
```

Arguments

x	Numeric vector to bin.
n_bins	Integer. Number of bins (default 3).
method	Character. One of "quantile" (equal-count), "equal" (equal-width), or "pretty" (round breakpoints).

Value

A factor with descriptive level labels.

Examples

```
x = rnorm(100, mean = 50, sd = 15)
bin_continuous(x, n_bins = 3, method = "quantile")
bin_continuous(x, n_bins = 4, method = "equal")
```

find_crossings *Find Crossing Regions Between Interaction Surfaces*

Description

Identifies where predicted response surfaces cross for different levels of conditioning factors. Returns a data frame of approximate crossing locations.

Usage

```
find_crossings(  
  model,  
  x,  
  y,  
  facet_by,  
  n = 50,  
  n_bins = 3,  
  bin_method = "quantile",  
  tolerance = NULL  
)
```

Arguments

model	A fitted model object.
x	Character. First focal variable.
y	Character. Second focal variable.
facet_by	Character vector. Conditioning variable(s).
n	Integer. Grid resolution (default 50).
n_bins	Integer. Bins for continuous facet_by (default 3).
bin_method	Character. Binning method.
tolerance	Numeric or NULL. Crossing detection tolerance.

Value

A data.frame with columns:

cx x-coordinate of crossing

cy y-coordinate of crossing

cz predicted response at crossing (average of both surfaces)

pair_label which surface pair crosses

Examples

```
dat = sim_factorial(design = "mixed", seed = 42)
fit = lm(y ~ temp * pressure * catalyst, data = dat)
crossings = find_crossings(fit, "temp", "pressure", "catalyst")
head(crossings)
```

interaction_surface *Interactive 3D Surface Plot for Multi-Factor Interactions*

Description

Generates an interactive plotly surface plot from a fitted model. Two focal variables are mapped to the x and y aesthetics, with the predicted response on z. Additional conditioning factors (`facet_by`) generate separate surfaces — where surfaces cross indicates interaction effects.

Usage

```
interaction_surface(  
  model,  
  x,  
  y,  
  facet_by = NULL,  
  n = 50,  
  n_bins = 3,  
  bin_method = "quantile",  
  alpha = 0.6,  
  show_points = FALSE,  
  show_crossings = TRUE,  
  show_contour = FALSE,  
  contour_z = NULL,  
  labs = NULL,  
  title = NULL,  
  theme = "default",  
  ...  
)
```

Arguments

<code>model</code>	A fitted model object with a predict method (e.g., from <code>lm</code> , <code>aov</code> , <code>glm</code> , <code>gam</code>).
<code>x</code>	Character. Variable name mapped to the x-axis.
<code>y</code>	Character. Variable name mapped to the y-axis.
<code>facet_by</code>	Character vector or NULL. Variable(s) whose levels generate separate surfaces. Continuous variables are automatically binned.

<code>n</code>	Integer. Grid resolution per continuous axis (default 50). Analogous to <code>n</code> in <code>geom_contour</code> .
<code>n_bins</code>	Integer. Number of bins for continuous <code>facet_by</code> variables (default 3).
<code>bin_method</code>	Character. Binning method for continuous <code>facet_by</code> : "quantile" (default), "equal", or "pretty".
<code>alpha</code>	Numeric in $[\emptyset, 1]$. Surface opacity (default 0.6).
<code>show_points</code>	Logical. If TRUE, overlays the observed data points, color-coded by <code>facet_by</code> level. Analogous to adding <code>geom_point</code> .
<code>show_crossings</code>	Logical. If TRUE (default), marks regions where surfaces cross with red markers.
<code>show_contour</code>	Logical. If FALSE (default), no contour projection. If TRUE, projects crossing curves onto the x-y floor of the plot as a 2D summary of interaction regions.
<code>contour_z</code>	Numeric or NULL. The z-value at which to draw the contour projection. If NULL, uses the minimum z in the plot.
<code>labs</code>	Named list for axis labels, e.g., <code>list(x = "Temperature", y = "Pressure", z = "Yield")</code> . Analogous to <code>ggplot2::labs()</code> .
<code>title</code>	Character or NULL. Plot title.
<code>theme</code>	Character. Color theme: "default", "viridis", or "grey".
<code>...</code>	Additional arguments (reserved for future use).

Details

Geometric interpretation:

- Parallel surfaces → no interaction between `facet_by` and the focal variables
- Crossing surfaces → interaction present
- Twisted/warped surfaces → higher-order or nonlinear interaction

For categorical focal variables, the surface is constructed over an integer grid with axis tick labels showing level names.

Non-focal, non-`facet_by` variables are held at their median (continuous) or mode (categorical).

For GLM family models, predictions are returned on the response scale via `predict(..., type = "response")`.

Value

A plotly htmlwidget object.

Examples

```
dat = sim_factorial(design = "mixed", seed = 42)
fit = lm(y ~ temp * pressure * catalyst, data = dat)
interaction_surface(fit, x = "temp", y = "pressure", facet_by = "catalyst")

# with GLM
dat$success = rbinom(nrow(dat), 1, plogis(scale(dat$y)))
```

```
gfit = glm(success ~ temp * pressure * catalyst, data = dat, family = binomial)
interaction_surface(gfit, x = "temp", y = "pressure", facet_by = "catalyst")
```

interaction_surface_grid

Generate All Pairwise Interaction Surface Plots

Description

For a model with multiple factors, generates interaction surface plots for every pair of focal variables, using remaining variables as conditioning factors (facet_by). Useful for exploratory analysis.

Usage

```
interaction_surface_grid(
  model,
  factors = NULL,
  facet_max = 2,
  n = 30,
  n_bins = 3,
  alpha = 0.6,
  ...
)
```

Arguments

model	A fitted model object.
factors	Character vector or NULL. Variables to consider. If NULL, uses all predictors.
facet_max	Integer. Maximum number of facet_by variables per plot (default 2). Extra variables are held at central values.
n	Integer. Grid resolution (default 30, lower for speed).
n_bins	Integer. Bins for continuous facet_by variables.
alpha	Numeric. Surface opacity.
...	Passed to interaction_surface.

Value

A named list of plotly objects. Names use pattern "x__y".

Examples

```
dat = sim_factorial(design = "mixed", seed = 42)
fit = lm(y ~ temp * pressure * catalyst, data = dat)
plots = interaction_surface_grid(fit)
plots$temp__pressure
```

make_prediction_grid *Build a prediction grid over two focal variables*

Description

For continuous variables, generates a regular sequence across the observed range. For categorical/factor variables, uses all observed levels. Continuous facet_by variables are binned, and predictions use bin midpoints.

Usage

```
make_prediction_grid(
  model,
  x,
  y,
  facet_by = NULL,
  n = 50,
  n_bins = 3,
  bin_method = "quantile"
)
```

Arguments

model	A fitted model object.
x	Character. First focal variable (mapped to x-axis).
y	Character. Second focal variable (mapped to y-axis).
facet_by	Character vector or NULL. Conditioning variable(s) whose levels generate separate surfaces.
n	Integer. Grid density for continuous axes.
n_bins	Integer. Number of bins for continuous facet_by variables.
bin_method	Character. Binning method: "quantile", "equal", or "pretty".

Value

A list with components:

grid data.frame suitable for predict()

binned_by named list of bin info for continuous facet_by variables

Examples

```

dat = sim_factorial(design = "mixed", seed = 42)
fit = lm(y ~ temp * pressure * catalyst, data = dat)
result = make_prediction_grid(fit, x = "temp", y = "pressure",
                             facet_by = "catalyst", n = 10)

str(result$grid)

```

plot_crossings

Plot Crossing Regions as a Standalone 3D Scatter

Description

Visualizes only the crossing points between interaction surfaces as an interactive 3D scatter plot, color-coded by surface pair. This isolates where interaction effects are strongest, without the surfaces themselves.

Usage

```

plot_crossings(
  model,
  x,
  y,
  facet_by,
  n = 50,
  n_bins = 3,
  bin_method = "quantile",
  tolerance = NULL,
  labs = NULL,
  title = NULL,
  marker_size = 3,
  marker_opacity = 0.7
)

```

Arguments

model	A fitted model object.
x	Character. First focal variable.
y	Character. Second focal variable.
facet_by	Character vector. Conditioning variable(s).
n	Integer. Grid resolution (default 50).
n_bins	Integer. Bins for continuous facet_by (default 3).
bin_method	Character. Binning method.
tolerance	Numeric or NULL. Crossing detection tolerance.

labs	Named list for axis labels, e.g., <code>list(x = "Temperature", y = "Pressure", z = "Yield")</code> .
title	Character or NULL. Plot title.
marker_size	Numeric. Marker size (default 3).
marker_opacity	Numeric in <code>[0, 1]</code> . Marker opacity (default 0.7).

Value

A plotly htmlwidget object. If no crossings are found, returns an empty plot with a "No crossings detected" annotation.

Examples

```
dat = sim_factorial(design = "mixed", seed = 42)
fit = lm(y ~ temp * pressure * catalyst, data = dat)
plot_crossings(fit, "temp", "pressure", "catalyst")

# with custom labels
plot_crossings(fit, "temp", "pressure", "catalyst",
               labs = list(x = "Temp (C)", y = "Press (psi)", z = "Yield"))
```

sim_factorial

Simulate a Multi-Factor Experimental Dataset

Description

Generates synthetic data from a factorial design with known interaction structure. Useful for demonstrating and testing `interaction_surface`.

Usage

```
sim_factorial(
  n = 200,
  design = c("mixed", "continuous", "categorical"),
  noise = 0.5,
  seed = NULL
)
```

Arguments

n	Integer. Total number of observations (default 200).
design	Character. One of: "mixed" Two continuous (temp, pressure) + one categorical (catalyst). "continuous" Three continuous factors (temp, pressure, speed).

	"categorical" Three categorical factors (catalyst, operator, shift).
noise	Numeric. Standard deviation of Gaussian noise (default 0.5).
seed	Integer or NULL. Random seed for reproducibility.

Details

The data generating process includes main effects for all factors, two-way interactions between the first two factors, and a three-way interaction (weaker) involving all factors. This makes it straightforward to verify that `interaction_surface` correctly detects the embedded structure.

Value

A `data.frame` with factor columns and a response column `y`.

Examples

```
dat = sim_factorial(design = "mixed", seed = 42)
head(dat)
```

Index

`bin_continuous`, 2
`find_crossings`, 3
`interaction_surface`, 4
`interaction_surface_grid`, 6
`make_prediction_grid`, 7
`plot_crossings`, 8
`sim_factorial`, 9