Package 'ferrn'

June 24, 2024

Title Facilitate Exploration of touRR optimisatioN

Version 0.1.0

Description Diagnostic plots for optimisation, with a focus on projection pursuit. These show paths the optimiser
takes in the high-dimensional space in multiple ways: by reducing the dimension using principal component analysis, and
also using the tour to show the path on the high-
dimensional space. Several botanical colour palettes are included, reflecting the
name of the package. A paper describing the methodology can be found at <https://journal.r-project.org/archive/2021/RJ-2021-105/index.html>.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

URL https://github.com/huizezhang-sherry/ferrn/

BugReports https://github.com/huizezhang-sherry/ferrn/issues

Imports rlang (>= 0.1.2), dplyr, magrittr, scales, gganimate, ggplot2,
tibble, purrr, tourr, stringr, ggrepel, ggforce, tidyr, cli,
progress, glue, GpGp,

RoxygenNote 7.3.1

Depends R (>= 2.10)

Suggests roxygen2, covr, pkgdown, testthat, forcats, patchwork,
future.apply,

Language en-GB

NeedsCompilation no

Author H. Sherry Zhang [aut, cre] (<https://orcid.org/0000-0002-7122-1463>),
Dianne Cook [aut] (<https://orcid.org/0000-0002-3813-7155>),
Ursula Laa [aut] (<https://orcid.org/0000-0002-0249-6439>),
Nicolas Langrené [aut] (<https://orcid.org/0000-0001-7601-4618>),
Patricia Menéndez [aut] (<https://orcid.org/0000-0003-0701-6315>)

Maintainer H. Sherry Zhang <huizezhangsh@gmail.com>

Repository CRAN

Date/Publication 2024-06-23 22:30:07 UTC

Contents

`add_anchor` *A ggproto for drawing anchor points*

Description

This is a wrapper function used by `explore_space_pca()` and should be be called directly by the user

Usage

```
add_anchor(dt, anchor_size = 3, anchor_alpha = 0.5, anchor_color = NULL, ...)
```

Arguments

dt	A data object from the running the optimisation algorithm in guided tour
anchor_size	numeric; the size of the anchor points
anchor_alpha	numeric; the alpha of the anchor points
anchor_color	the variable to be coloured by
...	other aesthetics inherent from explore_space_pca()

Value

a wrapper for drawing anchor points in explore_space_pca()

See Also

Other draw functions: add_anno(), add_dir_search(), add_end(), add_interp(), add_interp_last(), add_interrupt(), add_search(), add_space(), add_start(), add_theo()

---

add_anno *A ggproto for annotating the symmetry of the starting points*

Description

This is a wrapper function used by explore_space_pca() and should be be called directly by the user

Usage

```
add_anno(dt, anno_color = "black", anno_lty = "dashed", anno_alpha = 0.1, ...)
```

Arguments

dt	A data object from the running the optimisation algorithm in guided tour
anno_color	character; the colour of the annotation line
anno_lty	character; the linetype of the annotation line
anno_alpha	numeric; the alpha of the annotation line
...	other aesthetics inherent from explore_space_pca()

Value

a wrapper for annotating the symmetry of start points in explore_space_pca()

See Also

Other draw functions: add_anchor(), add_dir_search(), add_end(), add_interp(), add_interp_last(), add_interrupt(), add_search(), add_space(), add_start(), add_theo()

add_dir_search *A ggproto for drawing directional search points*

Description

This is a wrapper function used by `explore_space_pca()` and should be be called directly by the user

Usage

```
add_dir_search(dt, dir_size = 0.5, dir_alpha = 0.5, dir_color = NULL, ...)
```

Arguments

`dt`	A data object from the running the optimisation algorithm in guided tour
`dir_size`	numeric; the size of the directional search points in pseudo derivative search
`dir_alpha`	numeric; the alpha of the directional search points in pseudo derivative search
`dir_color`	the variable to be coloured by
`...`	other aesthetics inherent from `explore_space_pca()`

Value

a wrapper for drawing directional search points (used in pseudo derivative search) with buffer in `explore_space_pca()`

See Also

Other draw functions: `add_anchor()`, `add_anno()`, `add_end()`, `add_interp()`, `add_interp_last()`, `add_interrupt()`, `add_search()`, `add_space()`, `add_start()`, `add_theo()`

add_end *A ggproto for drawing start points*

Description

This is a wrapper function used by `explore_space_pca()` and should be be called directly by the user

Usage

```
add_end(dt, end_size = 5, end_alpha = 1, end_color = NULL, ...)
```

Arguments

dt	A data object from the running the optimisation algorithm in guided tour
end_size	numeric; the size of the end point
end_alpha	numeric; the alpha of the end point
end_color	the variable to be coloured by
...	other aesthetics inherent from explore_space_pca()

Value

a wrapper for drawing end points in explore_space_pca()

See Also

Other draw functions: add_anchor(), add_anno(), add_dir_search(), add_interp(), add_interp_last(), add_interrupt(), add_search(), add_space(), add_start(), add_theo()

---

add_interp *A ggproto for drawing interpolation path*

Description

This is a wrapper function used by explore_space_pca() and should be be called directly by the user

Usage

```
add_interp(
  dt,
  interp_size = 1.5,
  interp_alpha = NULL,
  interp_color = NULL,
  interp_group = NULL,
  ...
)
```

Arguments

dt	A data object from the running the optimisation algorithm in guided tour
interp_size	numeric; the size of the interpolation path
interp_alpha	numeric; the alpha of the interpolation path
interp_color	the variable to be coloured by
interp_group	the variable to label different interpolation path
...	other aesthetics inherent from explore_space_pca()

Value

a wrapper for drawing the interpolation points in `explore_space_pca()`

See Also

Other draw functions: `add_anchor()`, `add_anno()`, `add_dir_search()`, `add_end()`, `add_interp_last()`, `add_interrupt()`, `add_search()`, `add_space()`, `add_start()`, `add_theo()`

`add_interp_last` *A ggproto for drawing finish points*

Description

This is a wrapper function used by `explore_space_pca()` and should be be called directly by the user

Usage

```
add_interp_last(
  dt,
  interp_last_size = 3,
  interp_last_alpha = 1,
  interp_last_color = NULL,
  ...
)
```

Arguments

`dt`	A data object from the running the optimisation algorithm in guided tour
`interp_last_size`	numeric; the size of the last interpolation points in each iteration
`interp_last_alpha`	numeric; the alpha of the last interpolation points in each iteration
`interp_last_color`	the variable to be coloured by
`...`	other aesthetics inherent from `explore_space_pca()`

Value

a wrapper for drawing the last interpolation points of each iteration in `explore_space_pca()`

See Also

Other draw functions: `add_anchor()`, `add_anno()`, `add_dir_search()`, `add_end()`, `add_interp()`, `add_interrupt()`, `add_search()`, `add_space()`, `add_start()`, `add_theo()`

add_interrupt *A ggproto for annotating the interrupted path*

Description

This is a wrapper function used by `explore_space_pca()` and should be be called directly by the user

Usage

```
add_interrupt(
  dt,
  interrupt_size = 0.5,
  interrupt_alpha = NULL,
  interrupt_color = NULL,
  interrupt_group = NULL,
  interrupt_linetype = "dashed",
  ...
)
```

Arguments

`dt`	A data object from the running the optimisation algorithm in guided tour
`interrupt_size`	numeric; the size of the interruption path
`interrupt_alpha`	numeric; the alpha of the interruption path
`interrupt_color`	the variable to be coloured by
`interrupt_group`	the variable to label different interruption
`interrupt_linetype`	character; the linetype to annotate the interruption
`...`	other aesthetics inherent from `explore_space_pca()`

Value

a wrapper for annotating the interruption in `explore_space_pca()`

See Also

Other draw functions: `add_anchor()`, `add_anno()`, `add_dir_search()`, `add_end()`, `add_interp()`, `add_interp_last()`, `add_search()`, `add_space()`, `add_start()`, `add_theo()`

add_search *A ggproto for drawing search points*

Description

This is a wrapper function used by `explore_space_pca()` and should be be called directly by the user

Usage

```
add_search(dt, search_size = 0.5, search_alpha = 0.5, search_color = NULL, ...)
```

Arguments

`dt`	A data object from the running the optimisation algorithm in guided tour
`search_size`	numeric; the size of the search points
`search_alpha`	numeric; the alpha of the anchor points
`search_color`	the variable to be coloured by
`...`	other aesthetics inherent from `explore_space_pca()`

Value

a wrapper for drawing search points in `explore_space_pca()`

See Also

Other draw functions: `add_anchor()`, `add_anno()`, `add_dir_search()`, `add_end()`, `add_interp()`, `add_interp_last()`, `add_interrupt()`, `add_space()`, `add_start()`, `add_theo()`

add_space *A ggproto for drawing circle*

Description

This is a wrapper function used by `explore_space_pca()` and should be be called directly by the user

Usage

```
add_space(
  dt,
  space_alpha = 0.5,
  space_fill = "grey92",
  space_color = "white",
  cent_size = 1,
  cent_alpha = 1,
  cent_color = "black",
  ...
)
```

Arguments

dt	A data object from the running the optimisation algorithm in guided tour
space_alpha	numeric; the alpha of the basis space
space_fill	character; the colour of the space filling
space_color	character; the colour of the space brim
cent_size	numeric; the size of the centre point
cent_alpha	numeric; an alpha of the centre point
cent_color	character; the colour of the centre point
...	other aesthetics inherent from explore_space_pca()

Value

a wrapper for drawing the space in explore_space_pca()

See Also

Other draw functions: add_anchor(), add_anno(), add_dir_search(), add_end(), add_interp(), add_interp_last(), add_interrupt(), add_search(), add_start(), add_theo()

Examples

```
library(ggplot2)
space <- tibble::tibble(x0 = 0, y0 = 0, r = 5)
ggplot() +
  add_space(space) +
  theme_void() +
  theme(aspect.ratio = 1)
```

add_start *A ggproto for drawing start points*

Description

This is a wrapper function used by `explore_space_pca()` and should be be called directly by the user

Usage

```
add_start(dt, start_size = 5, start_alpha = 1, start_color = NULL, ...)
```

Arguments

`dt`	A data object from the running the optimisation algorithm in guided tour
`start_size`	numeric; the size of start point
`start_alpha`	numeric; the alpha of start point
`start_color`	the variable to be coloured by
`...`	other aesthetics inherent from `explore_space_pca()`

Value

a wrapper for drawing start points in `explore_space_pca()`

See Also

Other draw functions: `add_anchor()`, `add_anno()`, `add_dir_search()`, `add_end()`, `add_interp()`, `add_interp_last()`, `add_interrupt()`, `add_search()`, `add_space()`, `add_theo()`

Examples

```
library(ggplot2)
# construct the space and start df for plotting
space <- tibble::tibble(x0 = 0, y0 = 0, r = 5)
holes_1d_geo %>%
  compute_pca() %>%
  purrr::pluck("aug") %>%
  clean_method() %>%
  get_start()
```

add_theo *A ggproto for drawing the theoretical basis, if applicable*

Description

This is a wrapper function used by `explore_space_pca()` and should be be called directly by the user

Usage

```
add_theo(
  dt,
  theo_label = "*",
  theo_size = 25,
  theo_alpha = 0.8,
  theo_color = "#000000",
  ...
)
```

Arguments

`dt`	A data object from the running the optimisation algorithm in guided tour
`theo_label`	character; a symbol to label the theoretical point
`theo_size`	numeric; the size of the theoretical point
`theo_alpha`	numeric; the alpha of the theoretical point
`theo_color`	character; the colour of the theoretical point in hex
`...`	other aesthetics inherent from `explore_space_pca()`

Value

a wrapper for drawing theoretical points in `explore_space_pca()`

See Also

Other draw functions: `add_anchor()`, `add_anno()`, `add_dir_search()`, `add_end()`, `add_interp()`, `add_interp_last()`, `add_interrupt()`, `add_search()`, `add_space()`, `add_start()`

bind_random *Bind random bases in the projection bases space*

Description

Given the orthonormality constraint, the projection bases live in a high dimensional hollow sphere. Generating random points on the sphere is useful to perceive the data object in the high dimensional space.

Usage

```
bind_random(dt, n = 500, seed = 1)
```

Arguments

dt	a data object collected by the projection pursuit guided tour optimisation in the tourr package
n	numeric; the number of random bases to generate in each dimension by geozoo
seed	numeric; a seed for generating reproducible random bases from geozoo

Value

a tibble object containing both the searched and random bases

See Also

Other bind: bind_random_matrix(), bind_theoretical()

Examples

```
bind_random(holes_1d_better) %>% tail(5)
```

bind_random_matrix *Bind random bases in the projection bases space as a matrix*

Description

Bind random bases in the projection bases space as a matrix

Usage

```
bind_random_matrix(basis, n = 500, d = 1, front = FALSE, seed = 1)
```

Arguments

basis	a matrix returned by get_basis_matrix()
n	numeric; the number of random bases to generate in each dimension by geozoo
d	numeric; dimension of the basis, d = 1, 2, ...
front	logical; if the random bases should be bound before or after the original bases
seed	numeric; a seed for generating reproducible random bases from geozoo

Value

matrix

a matrix containing both the searched and random bases

See Also

Other bind: bind_random(), bind_theoretical()

Examples

```
data <- get_basis_matrix(holes_1d_geo)
bind_random_matrix(data) %>% tail(5)
```

---

bind_theoretical *Bind the theoretical best record*

Description

The theoretical best basis is usually known for a simulated problem. Augment this information into the data object allows for evaluating the performance of optimisation against the theory.

Usage

```
bind_theoretical(dt, matrix, index, raw_data)
```

Arguments

dt	a data object collected by the projection pursuit guided tour optimisation in the tourr package
matrix	a matrix of the theoretical basis
index	the index function used to calculate the index value
raw_data	a tibble of the original data used to calculate the index value

Value

a tibble object containing both the searched and theoretical best bases

See Also

Other bind: `bind_random()`, `bind_random_matrix()`

Examples

```
best <- matrix(c(0, 1, 0, 0, 0), nrow = 5)
tail(holes_1d_better %>% bind_theoretical(best, tourr::holes(), raw_data = boa5), 1)
```

---

botanical_palettes *A customised colour palette based on Australian botanies*

Description

Available colours in the palettes

Usage

```
botanical_palettes

botanical_pal(palette = "fern", reverse = FALSE)
```

Arguments

`palette`	Colour palette from the botanical_palette
`reverse`	logical, if the colour should be reversed

Format

An object of class `list` of length 5.

Value

a function for interpolating colour in the botanical palette

clean_method *Clean method names*

Description

Clean method names

Usage

```
clean_method(dt)
```

Arguments

dt	a data object

Value

a tibble with method cleaned

Examples

```
head(clean_method(holes_1d_better), 5)
```

explore_space_start *Plot the PCA projection of the projection bases space*

Description

Plot the PCA projection of the projection bases space

Usage

```
explore_space_start(dt, group = NULL, pca = TRUE, ...)

explore_space_end(dt, group = NULL, pca = TRUE, ...)

explore_space_pca(
  dt,
  details = FALSE,
  pca = TRUE,
  group = NULL,
  color = NULL,
  facet = NULL,
  ...,
  animate = FALSE
)
```

Arguments

dt	a data object collected by the projection pursuit guided tour optimisation in tourr
group	the variable to label different runs of the optimiser(s)
pca	logical; if PCA coordinates need to be computed for the data
...	other arguments passed to add_*() functions
details	logical; if components other than start, end and interpolation need to be shown
color	the variable to be coloured by
facet	the variable to be faceted by
animate	logical; if the interpolation path needs to be animated

Value

a ggplot2 object

See Also

Other main plot functions: explore_space_tour(), explore_trace_interp(), explore_trace_search()

Examples

```
dplyr::bind_rows(holes_1d_geo, holes_1d_better) %>%
  bind_theoretical(matrix(c(0, 1, 0, 0, 0), nrow = 5),
    index = tourr::holes(), raw_data = boa5
  ) %>%
  explore_space_pca(group = method, details = TRUE) +
  scale_color_discrete_botanical()

## Not run:
best <- matrix(c(0, 1, 0, 0, 0), nrow = 5)
dt <- bind_theoretical(holes_1d_jellyfish, best, tourr::holes(), raw_data = boa5)
explore_space_start(dt)
explore_space_end(dt, group = loop, theo_size = 10, theo_color = "#FF0000")
explore_space_pca(
  dt, facet = loop, interp_size = 0.5, theo_size = 10,
  start_size = 1, end_size = 3
  )

## End(Not run)
```

`explore_space_tour` *Plot the grand tour animation of the bases space in high dimension*

Description

Plot the grand tour animation of the bases space in high dimension

Usage

```
explore_space_tour(..., axes = "bottomleft")

prep_space_tour(
  dt,
  group = NULL,
  flip = FALSE,
  n_random = 2000,
  color = NULL,
  rand_size = 1,
  rand_color = "#D3D3D3",
  point_size = 1.5,
  end_size = 5,
  theo_size = 3,
  theo_shape = 17,
  theo_color = "black",
  palette = botanical_palettes$fern,
  ...
)
```

Arguments

`...`	other argument passed to `tourr::animate_xy()` and `prep_space_tour()`
`axes`	see [tourr::animate_xy()]
`dt`	a data object collected by the projection pursuit guided tour optimisation in `tourr`
`group`	the variable to label different runs of the optimiser(s)
`flip`	logical; if the sign flipping need to be performed
`n_random`	numeric; the number of random basis to generate
`color`	the variable to be coloured by
`rand_size`	numeric; the size of random points
`rand_color`	character; the color hex code for random points
`point_size`	numeric; the size of points searched by the optimiser(s)
`end_size`	numeric; the size of end points
`theo_size`	numeric; the size of theoretical point(s)

theo_shape	numeric; the shape symbol in the basic plot
theo_color	character; the color of theoretical point(s)
palette	the colour palette to be used

Value

explore_space_tour() an animation of the search path in the high-dimensional sphere

prep_space_tour() a list containing various components needed for producing the animation

See Also

Other main plot functions: explore_space_start(), explore_trace_interp(), explore_trace_search()

Examples

```
explore_space_tour(dplyr::bind_rows(holes_1d_better, holes_1d_geo),
  group = method, palette = botanical_palettes$fern[c(1, 6)]
)
```

---

explore_trace_interp *Plot the trace the search progression*

Description

Trace the index value of search/ interpolation points in guided tour optimisation

Usage

```
explore_trace_interp(
  dt,
  iter = NULL,
  color = NULL,
  group = NULL,
  cutoff = 50,
  target_size = 3,
  interp_size = 1,
  accuracy_x = 5,
  accuracy_y = 0.01
)
```

Arguments

dt	a data object collected by the projection pursuit guided tour optimisation in tourr
iter	the variable to be plotted on the x-axis
color	the variable to be coloured by

group	the variable to label different runs of the optimiser(s)
cutoff	numeric; if the number of interpolating points is smaller than cutoff, all the interpolation points will be plotted as dots
target_size	numeric; the size of target points in the interpolation
interp_size	numeric; the size of interpolation points
accuracy_x	numeric; If the difference of two neighbour x-labels is smaller than accuracy_x, only one of them will be displayed. Used for better axis label
accuracy_y	numeric; the precision of y-axis label

Value

a ggplot object for diagnosing how the index value progresses during the interpolation

See Also

Other main plot functions: explore_space_start(), explore_space_tour(), explore_trace_search()

Examples

```
# Compare the trace of interpolated points in two algorithms
holes_1d_better %>%
  explore_trace_interp(interp_size = 2) +
  scale_color_continuous_botanical(palette = "fern")
```

---

explore_trace_search *Plot the count in each iteration*

Description

Plot the count in each iteration

Usage

```
explore_trace_search(
  dt,
  iter = NULL,
  color = NULL,
  cutoff = 15,
  extend_lower = 0.95,
  ...
)
```

Arguments

dt	a data object collected by the projection pursuit guided tour optimisation in tourr
iter	the variable to be plotted on the x-axis
color	the variable to be coloured by
cutoff	numeric; if the number of searches in one iteration is smaller than cutoff, a point geom, rather than boxplot geom, will be used.
extend_lower	a numeric for extending the y-axis to display text labels
...	arguments passed into geom_label_repel() for displaying text labels

Value

a ggplot object for diagnosing how many points the optimiser(s) have searched

See Also

Other main plot functions: explore_space_start(), explore_space_tour(), explore_trace_interp()

Examples

```
# Summary plots for search points in two algorithms
library(patchwork)
library(dplyr)
library(ggplot2)
p1 <- holes_1d_better %>% explore_trace_search() +
  scale_color_continuous_botanical(palette = "fern")
p2 <- holes_2d_better_max_tries %>% explore_trace_search() +
  scale_color_continuous_botanical(palette = "daisy")
p1 / p2
```

---

flip_sign *Helper functions for 'explore_space_pca()'*

Description

Helper functions for 'explore_space_pca()'

Usage

```
flip_sign(dt, group = NULL, ...)

compute_pca(dt, group = NULL, random = TRUE, flip = TRUE, ...)
```

Arguments

dt	a data object collected by the projection pursuit guided tour optimisation in tourr
group	the variable to label different runs of the optimiser(s)
...	other arguments received from explore_space_pca()
random	logical; if random bases from the basis space need to be added to the data
flip	logical; if the sign flipping need to be performed

Value

flip_sign(): a list containing a matrix of all the bases, a logical value indicating whether a flip of sign is performed, and a data frame of the original dataset.

compute_pca(): a list containing the PCA summary and a data frame with PC coordinates augmented.

Examples

```
dt <- dplyr::bind_rows(holes_1d_geo, holes_1d_better)
 flip_sign(dt, group = method) %>% str(max = 1)
compute_pca(dt, group = method)
```

---

format_label *Better label formatting to avoid overlapping*

Description

Better label formatting to avoid overlapping

Usage

```
format_label(labels, accuracy)
```

Arguments

labels	a numerical vector of labels
accuracy	the accuracy of the label

Value

a vector of adjusted labels

Examples

```
format_label(c(0.87, 0.87, 0.9, 0.93, 0.95), 0.01)
format_label(c(0.87, 0.87, 0.9, 0.93, 0.95, 0.96, 0.96), 0.01)
```

get_best *Functions to get components from the data collecting object*

Description

Functions to get components from the data collecting object

Usage

```
get_best(dt, group = NULL)

get_start(dt, group = NULL)

get_interp(dt, group = NULL)

get_interp_last(dt, group = NULL)

get_anchor(dt, group = NULL)

get_search(dt)

get_dir_search(dt, ratio = 5, ...)

get_space_param(dt, ...)

get_theo(dt)

get_interrupt(dt, group = NULL, precision = 0.001)

get_search_count(dt, iter = NULL, group = NULL)

get_basis_matrix(dt)
```

Arguments

dt	a data object collected by the projection pursuit guided tour optimisation in the tourr package
group	the variable to label different runs of the optimiser(s)
ratio	numeric; a buffer value to deviate directional search points from the anchor points
...	other arguments passed to compute_pca()
precision	numeric; if the index value of the last interpolating point and the anchor point differ by precision, an interruption is registered
iter	the variable to be counted by

Details

`get_best`: extract the best basis found by the optimiser(s)

`get_start`: extract the start point of the optimisation

`get_interp`: extract the interpolation points

`get_interp_last`: extract the last point in each interpolation

`get_anchor`: extract the anchor points on the geodesic path

`get_search`: extract search points in the optimisation (for `search_geodesic`)

`get_dir_search`: extract directional search points (for `search_geodesic`)

`get_space_param`: estimate the radius of the background circle based on the randomly generated points. The space of projected bases is a circle when reduced to 2D. A radius is estimated using the largest distance from the bases in the data object to the centre point.

`get_theo`: extract the theoretical basis, if exist

`get_interrupt`: extract the end point of the interpolation and the target point in the iteration when an interruption happens. The optimiser can find better basis on the interpolation path, an interruption is implemented to stop further interpolation from the highest point to the target point. This discrepancy is highlighted in the PCA plot.

`get_search_count`: summarise the number of search points in each iteration

`get_basis_matrix`: extract all the bases as a matrix

Value

a tibble object containing the best basis found by the optimiser(s)

Examples

```
get_search(holes_1d_geo)

get_anchor(holes_1d_geo)

get_start(holes_1d_better)

get_interrupt(holes_1d_better)

get_interp(holes_1d_better) %>% head()

get_basis_matrix(holes_1d_better) %>% head()

get_best(dplyr::bind_rows(holes_1d_better, holes_1d_geo), group = method)

get_search_count(holes_1d_better)
get_search_count(dplyr::bind_rows(holes_1d_better, holes_1d_geo), group = method)

get_interp_last(holes_1d_better)
get_interp_last(dplyr::bind_rows(holes_1d_better, holes_1d_geo), group = method)

res <- holes_1d_geo %>% compute_pca() %>% purrr::pluck("aug")
get_dir_search(res)
```

```
best <- matrix(c(0, 1, 0, 0, 0), nrow = 5)
holes_1d_better %>%
  bind_theoretical(best, tourr::holes(), raw_data = boa5) %>%
  get_theo()
```

---

holes_1d_geo *Data objects collected during the projection pursuit optimisation*

Description

Simulated data to demonstrate the usage of four diagnostic plots in the package, users can create their own guided tour data objects and diagnose with the visualisation designed in this package.

Usage

```
holes_1d_geo

holes_1d_better

holes_1d_jellyfish

holes_2d_better

holes_2d_better_max_tries
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 416 rows and 8 columns.

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 79 rows and 8 columns.

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 2500 rows and 8 columns.

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 98 rows and 8 columns.

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 1499 rows and 8 columns.

Details

The prefix `holes_*` indicates the use of holes index in the guided tour. The suffix `*_better/geo/jellyfish` indicates the optimiser used: `search_better`, `search_geodesic`, `search_jellyfish`.

Examples

```
holes_1d_better %>%
explore_trace_interp(interp_size = 2) +
  scale_color_continuous_botanical(palette = "fern")
```

plot_projection *Plot the projection from the optimisation data collected from projection pursuit*

Description

Plot the projection from the optimisation data collected from projection pursuit

Usage

```
plot_projection(dt, data, cols = NULL)

compute_projection(dt, data, cols = NULL)
```

Arguments

dt	a data object collected by the projection pursuit guided tour optimisation in tourr
data	the original data
cols	additional columns to include in the plot

Value

a ggplot object

Examples

```
holes_1d_jellyfish |> get_best() |> plot_projection(data = boa5)
```

sample_bases *Function to calculate smoothness and squintability*

Description

Function to calculate smoothness and squintability

Usage

```
sample_bases(
  idx,
  data = sine1000,
  n_basis = 300,
  parallel = FALSE,
  best = matrix(c(0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1), nrow = 6),
  min_proj_dist = NA,
```

```
  step_size = NA,
  seed = 123
)

## S3 method for class 'basis_df'
print(x, width = NULL, ...)

## S3 method for class 'basis_df'
tbl_sum(x)

calc_smoothness(
  basis_df,
  start_params = c(0.001, 0.5, 2, 2),
  other_gp_params = NULL,
  verbose = FALSE
)

## S3 method for class 'smoothness_res'
print(x, width = NULL, ...)

## S3 method for class 'smoothness_res'
tbl_sum(x)

calc_squintability(
  basis_df,
  method = c("ks", "nls"),
  scale = TRUE,
  bin_width = 0.005,
  other_params = NULL
)

## S3 method for class 'squintability_res'
print(x, width = NULL, ...)

## S3 method for class 'squintability_res'
tbl_sum(x)

fit_ks(basis_df, idx, other_params = NULL)

fit_nls(basis_df, other_params = NULL)
```

Arguments

idx	character, the name of projection pursuit index function, e.g. "holes"
data	a matrix or data frame, the high dimensional data to be projected
n_basis	numeric, the number of random bases to generate
parallel	logic, whether to use parallel computing for calculating the index. Recommend for the stringy index.

best	a matrix, the theoretical/ empirical best projection matrix to calculate the projection distance from the simulated random bases.
min_proj_dist	only for squintability, the threshold for projection distance for the random basis to be considered in sampling
step_size	numeric, step size for interpolating from each random basis to the best basis, recommend 0.005
seed	numeric, seed for sampling random bases
x	objects with specialised printing methods
width	only used when max.levels is NULL, see above.
...	further arguments passed to or from other methods.
basis_df	the basis data frame returned from sample_bases
start_params	list, the starting parameters for the Gaussian process for smoothness
other_gp_params	list, additional parameters to be passed to [GpGp::fit_model()] for calculating smoothness
verbose	logical, whether to print optimisation progression when fitting the Gaussian process
method	either "ks" (kernel smoothing) or "nls" (non-linear least square) for calculating squintability.
scale	logic, whether to scale the index value to 0-1 in squintability
bin_width	numeric, the bin width to average the index value before fitting the kernel, recommend to set as the same as ‘step‘ parameter
other_params	list additional parameters for fitting kernel smoothing or non-linear least square, see [stats::ksmooth()] and [stats::nls()] for details

Examples

```
## Not run:
library(GpGp)
library(fields)
library(tourr)
basis_smoothness <- sample_bases(idx = "holes")
calc_smoothness(basis_smoothness)
basis_squint <- sample_bases(idx = "holes", n_basis = 100, step_size = 0.01, min_proj_dist = 1.5)
calc_squintability(basis_squint, method = "ks", bin_width = 0.01)

## End(Not run)
```

`scale_color_continuous_botanical` *continuous scale colour function*

Description

continuous scale colour function

Discrete scale colour function

continuous scale fill function

discrete scale fill function

Usage

```
scale_color_continuous_botanical(palette = "fern", reverse = FALSE, ...)

scale_color_discrete_botanical(palette = "fern", reverse = FALSE, ...)

scale_fill_continuous_botanical(palette = "fern", reverse = FALSE, ...)

scale_fill_discrete_botanical(palette = "fern", reverse = FALSE, ...)
```

Arguments

`palette`	colour palette from the botanical_palette
`reverse`	logical; if the colour should be reversed
`...`	other arguments passed into scale_color_gradientn

Value

a wrapper for continuous scales in the botanical palette

a wrapper for discrete scales in the botanical palette

a wrapper for continuous fill in the botanical palette

a wrapper for discrete fill in the botanical palette

sine1000 *Simulated sine, pipe, and gaussian mixture*

Description

Simulated sine and pipe data for calculating optimisation features. Each dataset has 1000 observations and the last two columns contain the intended structure with the rest being noise.

Usage

```
sine1000

sine1000_8d

pipe1000

pipe1000_8d

pipe1000_10d

pipe1000_12d

boa

boa5

boa6
```

Format

An object of class `matrix` (inherits from `array`) with 1000 rows and 6 columns.

An object of class `matrix` (inherits from `array`) with 1000 rows and 8 columns.

An object of class `matrix` (inherits from `array`) with 1000 rows and 6 columns.

An object of class `matrix` (inherits from `array`) with 1000 rows and 8 columns.

An object of class `matrix` (inherits from `array`) with 1000 rows and 10 columns.

An object of class `matrix` (inherits from `array`) with 1000 rows and 12 columns.

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 1000 rows and 10 columns.

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 1000 rows and 5 columns.

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 1000 rows and 6 columns.

Examples

```
library(ggplot2)
library(tidyr)
library(dplyr)
```

```
boa %>%
  pivot_longer(cols = x1:x10, names_to = "var", values_to = "value") %>%
  mutate(var = forcats::fct_relevel(as.factor(var), paste0("x", 1:10))) %>%
  ggplot(aes(x = value)) +
  geom_density() +
  facet_wrap(vars(var))

sine1000 |> ggplot(aes(x = V5, y = V6)) + geom_point() + theme(aspect.ratio = 1)
pipe1000_8d |> ggplot(aes(x = V5, y = V6)) + geom_point() + theme(aspect.ratio = 1)
pipe1000_8d |> ggplot(aes(x = V7, y = V8)) + geom_point() + theme(aspect.ratio = 1)
```

`theme_fern` *A specific theme for trace plots*

Description

A specific theme for trace plots

Usage

```
theme_fern()
```

Value

a ggplot2 theme for `explore_trace_interp()`

Index