Package ‘ferrn’

March 17, 2021

Title Facilitate Exploration of touRR optimisatioN

Version 0.0.1

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.

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, geozoo, tourr, stringr, ggrepel, ggforce, tidyr

RoxygenNote 7.1.1

Depends R (>= 2.10)

Suggests roxygen2, covr, pkgdown, testthat, forcats, patchwork

Language en-GB

NeedsCompilation no

Author H. Sherry Zhang [aut, cre] (<https://orcid.org/0000-0002-7122-1463>), Di 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 <huize.zhang@monash.edu>

Repository CRAN

Date/Publication 2021-03-17 13:20:16 UTC
R topics documented:

add_anchor ................................................................. 3
add_anno ................................................................. 3
add_dir_search .......................................................... 4
add_end ................................................................. 5
add_interp ............................................................... 5
add_interp_last ........................................................... 6
add_interrupt ............................................................. 7
add_search ............................................................... 8
add_space ................................................................. 9
add_start ............................................................... 10
add_theo ................................................................. 11
bind_random ............................................................... 12
bind_random_matrix ...................................................... 12
bind_theoretical .......................................................... 13
botanical_palettes ........................................................ 14
clean_method ............................................................. 15
explore_space_pca ........................................................ 15
explore_space_tour ....................................................... 17
explore_trace_interp ...................................................... 18
explore_trace_search ..................................................... 19
format_label .............................................................. 20
get_anchor ............................................................... 21
get_basis_matrix .......................................................... 21
get_best ................................................................. 22
get_dir_search ............................................................ 23
get_interp ............................................................... 23
get_interp_last ............................................................ 24
get_interrupt ............................................................. 25
get_search ............................................................... 26
get_search_count ......................................................... 26
get_space_param .......................................................... 27
get_start ............................................................... 28
get_theo ................................................................. 28
holes_1d_geo .............................................................. 29
scale_color_continuous_botanical ...................................... 31
theme_fern ............................................................... 32

Index .............................................................................. 33
add_anchor

A ggproto for drawing anchor points

Description

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

Usage

```r
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_last()`, `add_interp()`, `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 called directly by the user.

Usage

```r
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 drawing directional search points (used in pseudo derivative search) with buffer in `explore_space_pca()`

**See Also**

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

---

**Description**

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

**Usage**

```r
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 ggproto for drawing directional search points

**See Also**

Other draw functions: **add_anchor()**, **add_dir_search()**, **add_end()**, **add_interp_last()**, **add_interp()**, **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 called directly by the user.

Usage

```r
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_last()`, `add_interp()`, `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 called directly by the user.

Usage

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

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dt</td>
<td>A data object from the running the optimisation algorithm in guided tour</td>
</tr>
<tr>
<td>interp_size</td>
<td>numeric; the size of the interpolation path</td>
</tr>
<tr>
<td>interp_alpha</td>
<td>the alpha of the interpolation path</td>
</tr>
<tr>
<td>interp_color</td>
<td>the variable to be coloured by</td>
</tr>
<tr>
<td>interp_group</td>
<td>the variable to label different interpolation path</td>
</tr>
<tr>
<td>...</td>
<td>other aesthetics inherent from explore_space_pca()</td>
</tr>
</tbody>
</table>

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()

Description

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

Usage

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

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dt</td>
<td>A data object from the running the optimisation algorithm in guided tour</td>
</tr>
<tr>
<td>interp_last_size</td>
<td>numeric; the size of the last interpolation points in each iteration</td>
</tr>
<tr>
<td>interp_last_alpha</td>
<td>numeric; the alpha of the last interpolation points in each iteration</td>
</tr>
<tr>
<td>interp_last_color</td>
<td>the variable to be coloured by</td>
</tr>
<tr>
<td>...</td>
<td>other aesthetics inherent from explore_space_pca()</td>
</tr>
</tbody>
</table>
**add_interrupt**

**Value**

A ggproto for annotating the interrupted path

**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**

**Description**

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

**Usage**

```r
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 ggproto for annotating the interrupted path in `explore_space_pca()`
See Also

Other draw functions: `add_anchor()`, `add_anno()`, `add_dir_search()`, `add_end()`, `add_interp_last()`, `add_interp()`, `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**

```r
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_last()`, `add_interp()`, `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 called directly by the user.

Usage

```r
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_last()`, `add_interp()`, `add_interrupt()`, `add_search()`, `add_start()`, `add_theo()`
**Examples**

```r
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 called directly by the user.

**Usage**

```r
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_last()`, `add_interp()`, `add_interrupt()`, `add_search()`, `add_space()`, `add_theo()`.

**Examples**

```r
library(ggplot2)
# construct the space and start df for plotting
space <- tibble::tibble(x0 = 0, y0 = 0, r = 5)
start <- holes_1d_geo %>%
  compute_pca() %>%
purrr::pluck("aug") %>%
  clean_method() %>%
  get_start()
```
ggplot() +
add_space(dt = space) +
add_start(dt = start, start_color = info) +
theme_void() +
theme(aspect.ratio = 1)

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, ...)

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
...
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_last(), add_interp(), 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

```r
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

```r
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

```r
bind_random_matrix(basis, n = 500, 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`
- `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

```r
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

```r
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
botanical_palettes

A customised colour palette based on Australian botanies

Description

Available colours in the palettes

Usage

```r
botanical_palettes

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

Arguments

- **palette**: Colour palette from the `botanical_palettes`
- **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

```r
clean_method(dt)
```

### Arguments

- `dt`: a data object

### Value

a tibble with method cleaned

### Examples

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

## explore_space_pca

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

### Description

The set of functions returns a primary ggplot object that plots the data object in a space reduced by PCA. `compute_pca()` computes the PCA and `explore_space_pca()` plots the bases in the PCA-projected space

### Usage

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

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

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

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

- `explore_space_pca()` a ggplot object for diagnosing the optimisers in the PCA-projected basis space.
- `flip_sign()` a list containing:
  - a matrix of all the bases
  - a logical value whether a flip of sign is performed
  - a dataframe of the original dataset
- `compute_pca()` a list containing:
  - the PCA summary
  - a dataframe with PC coordinates augmented

**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()
dplyr::bind_rows(holes_1d_geo, holes_1d_better) %>%
  flip_sign(group = method) %>%
  str(max = 1)
dplyr::bind_rows(holes_1d_geo, holes_1d_better) %>%
  compute_pca(group = method)
explore_space_tour

### Description

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

### Usage

```r
explore_space_tour(...)
```

```r
prep_space_tour(
  dt,
  group = NULL,
  flip = FALSE,
  color = NULL,
  rand_size = 1,
  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()`
- `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
- `color` the variable to be coloured by
- `rand_size` numeric; the size of 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
explore_trace_interp

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_pca(), 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
explore_trace_search

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_pca(), 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_pca(), 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

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_anchor

Extract the anchor points on the geodesic path

Description
Extract the anchor points on the geodesic path

Usage
get_anchor(dt, group = NULL)

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)

Value
a tibble object containing the target bases in each iteration

See Also
Other get functions: get_basis_matrix(), get_best(), get_dir_search(), get_interp_last(), get_interp(), get_interrupt(), get_search_count(), get_search(), get_space_param(), get_start(), get_theo()

Examples
holes_1d_better %>% get_anchor()
holes_1d_geo %>% get_anchor()

get_basis_matrix

Extract all the bases as a matrix

Description
Extract all the bases as a matrix

Usage
get_basis_matrix(dt)

Arguments
- dt: a data object collected by the projection pursuit guided tour optimisation in the tourr package
Value

a matrix that flattens each basis into a row

See Also

Other get functions: `get_anchor()`, `get_best()`, `get_dir_search()`, `get_interp_last()`, `get_interp()`, `get_interrupt()`, `get_search_count()`, `get_search()`, `get_space_param()`, `get_start()`, `get_theo()`

Examples

```r
head(get_basis_matrix(holes_1d_better), 5)
```

---

### get_best

* **Description**
  
  Extract the record with the largest index value

* **Usage**
  
  ```r
get_best(dt, group = NULL)
  ```

* **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)

* **Value**

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

* **See Also**

  Other get functions: `get_anchor()`, `get_best()`, `get_dir_search()`, `get_interp_last()`, `get_interp()`, `get_interrupt()`, `get_search_count()`, `get_search()`, `get_space_param()`, `get_start()`, `get_theo()`

* **Examples**

  ```r
dplyr::bind_rows(holes_1d_better, holes_1d_geo) %>% get_best(group = method)
  ```
**get_dir_search**

*Extract directional search points during the optimisation*

**Description**

Extract directional search points during the optimisation

**Usage**

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

**Arguments**

- `dt`: a data object collected by the projection pursuit guided tour optimisation in the **tourr** package
- `ratio`: numeric; a buffer value to deviate directional search points from the anchor points
- `...`: arguments passed to `compute_pca()`

**Value**

a tibble object containing the directional search bases in pseudo derivative search

**See Also**

Other get functions: `get_anchor()`, `get_basis_matrix()`, `get_best()`, `get_interp_last()`, `get_interp()`, `get_interrupt()`, `get_search_count()`, `get_search()`, `get_space_param()`, `get_start()`, `get_theo()`

**Examples**

```r
holes_1d_geo %>%
  compute_pca() %>%
  purrr::pluck("aug") %>%
  get_dir_search()
```

---

**get_interp**

*Extract interpolated records*

**Description**

Extract interpolated records

**Usage**

```r
get_interp(dt, group = NULL)
```
get_interp_last

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)

Value

a tibble object containing the interpolating bases

See Also

Other get functions: get_anchor(), get_basis_matrix(), get_best(), get_dir_search(), get_interp_last(), get_interrupt(), get_search_count(), get_search(), get_space_param(), get_start(), get_theo()

Examples

holes_1d_better %>%
  get_interp() %>%
  head()

get_interp(dplyr::bind_rows(holes_1d_better, holes_1d Geo), group = method) %>% head()

get_interp_last Extract the end point at each interpolation

Description

Extract the end point at each interpolation

Usage

get_interp_last(dt, group = NULL)

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)

Value

a tibble object containing the last interpolating basis in each iteration

See Also

Other get functions: get_anchor(), get_basis_matrix(), get_best(), get_dir_search(), get_interp(), get_interrupt(), get_search_count(), get_search(), get_space_param(), get_start(), get_theo()
**get_interrupt**

**Examples**

```r
holes_1d_better %>% get_interp_last()
get_interp_last(dplyr::bind_rows(holes_1d_better, holes_1d_geo), group = method)
```

**get_interrupt**

Extract the end point of the interpolation and the target point in the iteration when an interruption happens

**Description**

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. You should not use geodesic search on this function.

**Usage**

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

**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)
- `precision` numeric; if the index value of the last interpolating point and the anchor point differ by precision, an interruption is registered

**Value**

a tibble object containing the target and anchor bases for the iteration when an interruption happens

**See Also**

Other get functions: `get_anchor()`, `get_basis_matrix()`, `get_best()`, `get_dir_search()`, `get_interp_last()`, `get_interp()`, `get_search_count()`, `get_search()`, `get_space_param()`, `get_start()`, `get_theo()`

**Examples**

```r
holes_1d_better %>% get_interrupt()
holes_1d_geo %>% get_interrupt()
```
get_search

Extract search points during the optimisation

Description
Extract search points during the optimisation

Usage
get_search(dt)

Arguments
dt a data object collected by the projection pursuit guided tour optimisation in the tourr package

Value
a tibble object containing the search bases

See Also
Other get functions: get_anchor(), get_basis_matrix(), get_best(), get_dir_search(), get_interp_last(), get_interp(), get_interrupt(), get_search_count(), get_space_param(), get_start(), get_theo()

Examples
holes_1d_better %>% get_search()
holes_1d_geo %>% get_search()

get_search_count

Extract the count in each iteration

Description
Extract the count in each iteration

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

Arguments
dt a data object collected by the projection pursuit guided tour optimisation in the tourr package
iter the variable to be counted by
group the variable to label different runs of the optimiser(s)
**Value**

A tibble object of the number of searches conducted by the optimiser(s) in each iteration.

**See Also**

Other get functions: `get_anchor()`, `get_basis_matrix()`, `get_best()`, `get_dir_search()`, `get_interp_last()`, `get_interp()`, `get_interrupt()`, `get_search()`, `get_space_param()`, `get_start()`, `get_theo()`

**Examples**

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

---

**get_space_param**

*Estimate the radius of the background circle based on the randomly generated points*

**Description**

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.

**Usage**

```r
get_space_param(dt, ...)
```

**Arguments**

- `dt` a data object collected by the projection pursuit guided tour optimisation in the tourr package
- `...` other arguments passed to `compute_pca()`

**Details**

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

**Value**

A tibble object of the centre and radius of the basis space.

**See Also**

Other get functions: `get_anchor()`, `get_basis_matrix()`, `get_best()`, `get_dir_search()`, `get_interp_last()`, `get_interp()`, `get_interrupt()`, `get_search_count()`, `get_search()`, `get_start()`, `get_theo()`
## get_start

### Description

Extract the starting records

### Usage

```r
get_start(dt)
```

### Arguments

- **dt**
  - a data object collected by the projection pursuit guided tour optimisation in the `tourr` package

### Value

a tibble object containing the start basis

### See Also

Other get functions: `get_anchor()`, `get_basis_matrix()`, `get_best()`, `get_dir_search()`, `get_interp_last()`, `get_interp()`, `get_interrupt()`, `get_search_count()`, `get_search()`, `get_space_param()`, `get_theo()`

### Examples

```r
holes_1d_better %>% get_start()
```

## get_theo

### Description

Extract the theoretical best basis, if applicable

### Usage

```r
get_theo(dt)
```

### Arguments

- **dt**
  - a data object collected by the projection pursuit guided tour optimisation in the `tourr` package
**Value**

a tibble object containing the theoretical bases

**See Also**

Other get functions: `get_anchor()`, `get_basis_matrix()`, `get_best()`, `get_dir_search()`, `get_interp_last()`, `get_interp()`, `get_interrupt()`, `get_search_count()`, `get_search()`, `get_space_param()`, `get_start()`

**Examples**

```r
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**

*Simulated data*

**Description**

The `boa` data is simulated using different Gaussian mixtures with varied centres and weights (see section format for the simulation code). These data are simulated for demonstrating 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_2d_better`
- `holes_2d_better_max_tries`
- `boa`
- `boa5`
- `boa6`

**Format**

The code for simulating each data object is as follows:
```
set.seed(123456);
holes_1d_geo <-
  animate_dist(boa5, tour_path = guided_tour(holes(), d = 1,
              search_f = search_geodesic),
              rescale = FALSE)

set.seed(123456)
holes_1d_better <-
  animate_dist(boa5, tour_path = guided_tour(holes(), d = 1,
              search_f = search_better),
              rescale = FALSE)

set.seed(123456)
holes_2d_better <-
  animate_xy(boa6, tour_path = guided_tour(holes(), d = 2,
              search_f = search_better),
              rescale = FALSE)

set.seed(123456)
holes_2d_better_max_tries <-
  animate_xy(boa6, tour_path = guided_tour(holes(), d = 2,
              search_f = search_better, max.tries = 500),
              rescale = FALSE)

library(tidyverse)
set.seed(1234)
x1 <- rnorm(1000, 0, 1)
x2 <- sample(c(rnorm(500, -3, 1), rnorm(500, 3, 1)), size = 1000)
x3 <- sample(c(rep(-1, 500), rep(1, 500)), size = 1000)
x4 <- sample(c(rnorm(250, -3, 1), rnorm(750, 3, 1)), size = 1000)
x5 <- sample(c(rnorm(330, -5, 1), rnorm(340, 0, 1), rnorm(330, 5, 1)), size = 1000)
x6 <- sample(c(rnorm(450, -5, 1), rnorm(100, 0, 1), rnorm(450, 5, 1)), size = 1000)
x7 <- sample(c(rnorm(500, -5, 1), rnorm(500, 5, 1)), size = 1000)
x8 <- rnorm(1000, 0, 1)
x9 <- rnorm(1000, 0, 1)
x10 <- rnorm(1000, 0, 1)

boa <- tibble(x1 = x1, x2 = x2, x3 = x3, x4 = x4, x5 = x5,
              x6 = x6, x7 = x7, x8 = x8, x9 = x9, x10 = x10)
boa <- as_tibble(scale(boa))
boa5 <- select(boa, x1, x2, x8: x10)
boa6 <- select(boa, x1, x2, x7: x10)

Details

The prefix holes_* indicates the use of holes index in the guided tour. The suffix *_better/geo indicates the optimiser used: search_better and search_geodesic.
The name boa comes from the fact that the density plot of each variable in the data looks like boa constrictors swallowing multiple French baguettes, rather than elephants, as in the novella the little prince.

Examples

```r
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))
```

---

**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**

```r
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

Description

A specific theme for trace plots

Usage

theme_fern()

Value

a ggplot2 theme for explore_trace_interp()
Index

* **bind**
  * bind_random, 12
  * bind_random_matrix, 12
  * bind_theoretical, 13

* **datasets**
  * botanical_palettes, 14
  * holes_1d_geo, 29

* **draw functions**
  * add_anchor, 3
  * add_anno, 3
  * add_dir_search, 3, 4–11
  * add_end, 3, 4–11
  * add_interp, 3–5, 7–11
  * add_interp_last, 3–6, 6, 8–11
  * add_interrupt, 3–7, 7, 8–11
  * add_search, 3–8, 9, 10, 11
  * add_start, 3–9, 10, 11
  * add_theo, 3–10, 11

* **get functions**
  * get_anchor, 21
  * get_basis_matrix, 21
  * get_best, 22
  * get_dir_search, 23
  * get_interp, 23
  * get_interp_last, 24
  * get_interrupt, 25
  * get_search, 26
  * get_search_count, 26
  * get_space_param, 27
  * get_start, 28
  * get_theo, 28

* **main plot functions**
  * explore_space_pca, 15
  * explore_space_tour, 17
  * explore_trace_interp, 18
  * explore_trace_search, 19

add_anchor, 3, 4–11
add_anno, 3, 4–11
add_dir_search, 3, 4, 4, 5–11
add_end, 3, 4, 5, 6–11
add_interp, 3–5, 5, 7–11
add_interp_last, 3–6, 6, 8–11
add_interrupt, 3–7, 7, 8–11
add_search, 3–8, 9, 10, 11
add_start, 3–9, 10, 11
add_theo, 3–10, 11
bind_random, 12, 13, 14
bind_random_matrix, 12, 12, 14
bind_theoretical, 12, 12, 14
boa(holes_1d_geo), 29
boa5(holes_1d_geo), 29
boa6(holes_1d_geo), 29
botanical_pal(botanical_palettes), 14
botanical_palettes, 14
clean_method, 15
compute_pca(explore_space_pca), 15
explore_space_pca, 15, 18–20
explore_space_tour, 16, 17, 19, 20
explore_trace_interp, 16, 18, 18, 20
explore_trace_search, 16, 18, 19, 19
flip_sign(explore_space_pca), 15
format_label, 20
get_anchor, 21, 22–29
get_basis_matrix, 21, 21, 22–29
get_best, 21, 22, 22, 23–29
get_dir_search, 21, 22, 23, 24–29
get_interp, 21–23, 23, 24–29
get_interp_last, 21–24, 24, 25–29
get_interrupt, 21–24, 25, 26–29
get_search, 21–25, 26, 27–29
get_search_count, 21–26, 26, 27–29
get_space_param, 21–27, 27, 28, 29
get_start, 21–27, 28, 29
get_theo, 27–28, 28
holes_1d_better (holes_1d_geo), 29
holes_1d_geo, 29
holes_2d_better (holes_1d_geo), 29
holes_2d_better_max_tries
(holes_1d_geo), 29
prep_space_tour (explore_space_tour), 17
scale_color_continuous_botanical, 31
scale_color_discrete_botanical
(scale_color_continuous_botanical), 31
scale_fill_continuous_botanical
(scale_color_continuous_botanical), 31
scale_fill_discrete_botanical
(scale_color_continuous_botanical), 31
theme_fern, 32