

Package ‘mistyR’

September 19, 2021

Type Package

Title Multiview Intercellular SpaTial modeling framework

Version 1.0.3

Description mistyR is an implementation of the Multiview Intercellular SpaTial modeling framework (MISTy). MISTy is an explainable machine learning framework for knowledge extraction and analysis of single-cell, highly multiplexed, spatially resolved data. MISTy facilitates an in-depth understanding of marker interactions by profiling the intra- and intercellular relationships. MISTy is a flexible framework able to process a custom number of views. Each of these views can describe a different spatial context, i.e., define a relationship among the observed expressions of the markers, such as intracellular regulation or paracrine regulation, but also, the views can also capture cell-type specific relationships, capture relations between functional footprints or focus on relations between different anatomical regions. Each MISTy view is considered as a potential source of variability in the measured marker expressions. Each MISTy view is then analyzed for its contribution to the total expression of each marker and is explained in terms of the interactions with other measurements that led to the observed contribution.

URL <https://github.com/saezlab/mistyR>

BugReports <https://github.com/saezlab/mistyR/issues>

biocViews Software, BiomedicalInformatics, CellBiology, SystemsBiology, Regression, DecisionTree, SingleCell

Depends R (>= 4.0)

License GPL-3

Encoding UTF-8

VignetteBuilder knitr

Imports assertthat, caret, deldir, digest, distances, dplyr, filelock, furr (>= 0.2.0), ggplot2, MASS, purrr, ranger, readr, rlang, rlist, R.utils, stats, stringr, tibble, tidyr, withr

Suggests BiocStyle, covr, future, igraph, knitr, Matrix, progeny, rmarkdown, sctransform, SingleCellExperiment, SpatialExperiment, SummarizedExperiment, testthat (>= 3.0.0)

RoxygenNote 7.1.1

Config/testthat/edition 3

git_url <https://git.bioconductor.org/packages/mistyR>

git_branch RELEASE_3_13

git_last_commit cf6b099

git_last_commit_date 2021-07-22

Date/Publication 2021-09-19

Author Jovan Tanevski [cre, aut] (<<https://orcid.org/0000-0001-7177-1003>>),
Ricardo Omar Ramirez Flores [ctb]
(<<https://orcid.org/0000-0003-0087-371X>>)

Maintainer Jovan Tanevski <jovan.tanevski@uni-heidelberg.de>

R topics documented:

add_juxtaview	2
add_paraview	4
add_views	5
clear_cache	6
collect_results	7
create_initial_view	8
create_view	9
plot_contrast_heatmap	10
plot_contrast_results	11
plot_improvement_stats	12
plot_interaction_communities	13
plot_interaction_heatmap	14
plot_view_contributions	15
remove_views	16
run_misty	17
synthetic	18
Index	20

add_juxtaview	<i>Generate and add a juxtaview to the current view composition</i>
---------------	---

Description

The juxtaview captures the expression of all markers within the immediate neighborhood of a spatial unit.

Usage

```
add_juxtaview(  
  current.views,  
  positions,  
  neighbor.thr = 15,  
  cached = FALSE,  
  verbose = TRUE  
)
```

Arguments

<code>current.views</code>	the current view composition.
<code>positions</code>	a <code>data.frame</code> , <code>tibble</code> or a <code>matrix</code> with named coordinates in columns and rows for each spatial unit ordered as in the <code>intraview</code> .
<code>neighbor.thr</code>	a threshold value used to indicate the largest distance between two spatial units that can be considered as neighboring.
<code>cached</code>	a logical indicating whether to cache the calculated view after the first calculation and to reuse a previously cached view if it already exists for this sample.
<code>verbose</code>	a logical controlling the verbosity of the output of the function during execution.

Details

The neighborhood of each spatial unit is estimated by constructing a graph by 2D Delaunay triangulation following by removal of edges with length larger than `neighbor.thr`. For each spatial unit the `juxtaview` contains the sum of expressions across its estimated neighbors for each marker.

Value

A `mistyR` view composition with added `juxtaview`.

See Also

[create_initial_view\(\)](#) for starting a view composition with an `intraview` only.

Other view composition functions: [add_paraview\(\)](#), [add_views\(\)](#), [create_initial_view\(\)](#), [create_view\(\)](#), [remove_views\(\)](#)

Examples

```
# Create a view composition of an intraview and a juxtaview.  
  
library(dplyr)  
  
# get the expression data  
data("synthetic")  
expr <- synthetic[[1]] %>% select(-c(row, col, type))  
# get the coordinates for each cell  
pos <- synthetic[[1]] %>% select(row, col)
```

```
# compose
misty.views <- create_initial_view(expr) %>% add_juxtaview(pos, neighbor.thr = 1.5)

# preview
str(misty.views[["juxtaview.1.5"]])
```

 add_paraview

Generate and add a paraview to the current view composition

Description

The paraview captures the expression of all markers in the broader tissue structure.

Usage

```
add_paraview(
  current.views,
  positions,
  l,
  approx = 1,
  nn = NULL,
  cached = FALSE,
  verbose = TRUE
)
```

Arguments

current.views	the current view composition.
positions	a data.frame, tibble or a matrix with named coordinates in columns and rows for each spatial unit ordered as in the intraview.
l	effective radius of influence of expression in the broader tissue structure.
approx	rank of the Nyström approximation matrix. (see Details)
nn	the number of spatial units to be used for approximating the paraview using a fast nearest neighbor search. (see Details)
cached	a logical indicating whether to cache the calculated view after the first calculation and to reuse a previously cached view if it already exists for this sample.
verbose	a logical controlling the verbosity of the output of the function during execution.

Details

The paraview is generated by weighted sum of the expression of all spatial units for each marker. The weights for each spatial unit i are dependent on the distance to the spatial unit j and the parameter l and are calculated using the radial basis function

$$w_{ij} = e^{-\frac{d_{ij}^2}{l^2}}$$

Since the generation of the paraview requires the calculation of pairwise distances of all spatial units it can take a significant amount of computation time. The parameters `approx` and `nn` can be set to speed up the calculation by approximation. The approximation can be achieved by using the Nyström low-rank approximation method or by limiting the calculation of the paraview to a number of nearest neighbors around each spatial unit.

If the value of `approx` is between 0 and 1 it will be interpreted as fraction of the number of spatial units. Discrete values above 1 will be interpreted as the size of the approximation block. The number of nearest neighbors `nn` around each spatial unit are determined using a fast nearest neighbor search.

If both `approx` and `nn` have non-null values, `nn` has priority and an approximation based on fast nearest neighbor search will be used to generate the paraview.

Value

A mistyR view composition with added paraview with parameter `l`.

See Also

[create_initial_view\(\)](#) for starting a view composition with an intraview only.

Other view composition functions: [add_juxtaview\(\)](#), [add_views\(\)](#), [create_initial_view\(\)](#), [create_view\(\)](#), [remove_views\(\)](#)

Examples

```
# Create a view composition of an intraview and a paraview with radius 10.

library(dplyr)

# get the expression data
data("synthetic")
expr <- synthetic[[1]] %>% select(-c(row, col, type))
# get the coordinates for each cell
pos <- synthetic[[1]] %>% select(row, col)

# compose
misty.views <- create_initial_view(expr) %>% add_paraview(pos, l = 10)

# preview
str(misty.views[["paraview.10"]])
```

add_views

Add custom views to the current view composition

Description

Add one or more custom views to the current view composition.

Usage

```
add_views(current.views, new.views)
```

Arguments

`current.views` the current view composition.
`new.views` a view or a list of views created with `create_view()` or otherwise.

Value

A mistyR view composition containing an union of views from `current.views` and `new.views`.

See Also

`create_initial_view()` for starting a view composition, with an intraview, `create_view()` for creating a custom view.

Other view composition functions: `add_juxtaview()`, `add_paraview()`, `create_initial_view()`, `create_view()`, `remove_views()`

Examples

```
# create random views
view1 <- data.frame(marker1 = rnorm(100, 10, 2), marker2 = rnorm(100, 15, 3))
view2 <- data.frame(marker1 = rnorm(100, 10, 5), marker2 = rnorm(100, 15, 5))

misty.views <- create_initial_view(view1)

new.view <- create_view("dummyname", view2, "dname")
add_views(misty.views, new.view)

misty.views %>% add_views(create_view("dummyname", view2, "dname"))
```

clear_cache

Clear cached objects

Description

Purge the cache or clear the cached objects for a single sample.

Usage

```
clear_cache(id = NULL)
```

Arguments

`id` the unique id of the sample.

Details

The cached objects are removed from disk and cannot be retrieved. Whenever possible specifying an id is recommended. If `id = NULL` all contents of the folder `‘.misty.temp’` will be removed.

Value

None (NULL)

Examples

```
clear_cache("b98ad35f4e671871cba35f2155228612")
clear_cache()
```

<code>collect_results</code>	<i>Collect and aggregate results</i>
------------------------------	--------------------------------------

Description

Collect and aggregate performance, contribution and importance estimations of a set of raw results produced by `run_misty()`.

Usage

```
collect_results(folders)
```

Arguments

`folders` Paths to folders containing the raw results from `run_misty()`.

Value

List of collected performance, contributions and importances per sample, performance and contribution statistics and aggregated importances.

improvements Long format tibble with measurements of performance for each *target* and each *sample*. Available performance measures are RMSE and variance explained (R2) for a model containing only an intrinsic view (*intra.RMSE*, *intra.R2*), model with all views (*multi.RMSE*, *multi.R2*), gain of RMSE and gain of variance explained of multi-view model over the intrinsic model where *gain.RMSE* is the relative decrease of RMSE in percent, while *gain.R2* is the absolute increase of variance explained in percent. Each *value* represents the mean performance across folds (k-fold cross-validation). The p values of a one sided t-test of improvement of performance (*p.RMSE*, *p.R2*) are also available as a measure.

improvements.stats Long format tibble with summary statistics (mean, standard deviation and coefficient of variation) for all performance measures for each target over all samples.

contributions Long format tibble with the values of the coefficients for each *view* in the meta-model, for each *target* and each *sample*. The p values for the coefficient for each view, under the null hypothesis of zero contribution to the meta model are also available.

contributions.stats Long format tibble with summary statistics for all views per target over all samples. Including mean coefficient value, fraction of contribution, mean and standard deviation of p values.

importances List of view-specific predictor-target importance tables per sample. The importances in each table are standardized per target and weighted by the quantile of the coefficient for the target in that view. Columns other than *Predictor* represent target markers.

importances.aggregated A list of aggregated view-specific predictor-target importance tables. Aggregation is reducing by mean over all samples.

See Also

[run_misty\(\)](#) to train models and generate results.

Examples

```
# Train and collect results for 3 samples in synthetic

library(dplyr)
library(purrr)

data("synthetic")

misty.results <- synthetic[seq_len(3)] %>%
  imap_chr(~ create_initial_view(.x %>% select(-c(row, col, type))) %>%
    add_paraview(.x %>% select(row, col), l = 10) %>%
    run_misty(paste0("results/", .y))) %>%
  collect_results()
str(misty.results)
```

create_initial_view **Start here:** *create a basic view composition with an intraview*

Description

This function is the first one to be called when building a mistyR workflow, starting from view composition. The initial view describes the intraview of the sample.

Usage

```
create_initial_view(data, unique.id = NULL)
```

Arguments

data	A data.frame or a tibble containing expression information for all markers of interest (in named columns) for each spatial unit (in rows).
unique.id	A character vector. Identifier of the current sample. If not provided (unique.id = NULL) then an id is automatically generated by calculating the md5 hash of table.

Value

An initial mistyR view composition containing an *intraview* list item named described with abbreviation "intra" and *data* as provided in *data* and a *misty.uniqueid* list item containing the provided or automatically calculated *unique.id*. A cache folder for the sample will be automatically created in the working directory as a subfolder of `'.misty.temp/'` with the same name as *unique.id*.

See Also

Other view composition functions: [add_juxtaview\(\)](#), [add_paraview\(\)](#), [add_views\(\)](#), [create_view\(\)](#), [remove_views\(\)](#)

Examples

```
# Create an intrinsic view from the first sample in the dataset synthetic.

library(dplyr)

# get the expression data
data("synthetic")
expr <- synthetic[[1]] %>% select(-c(row, col, type))

create_initial_view(expr)
```

create_view	<i>Create a custom view</i>
-------------	-----------------------------

Description

Create a custom view from a `data.frame` or a `tibble`.

Usage

```
create_view(name, data, abbrev = name)
```

Arguments

name	Name of the view. A character vector.
data	A <code>data.frame</code> or a <code>tibble</code> with named variables in columns and rows for each spatial unit ordered as in the <i>intraview</i> .
abbrev	Abbreviated name. A character vector.

Details

Creating a custom view does not add it to the current view composition.

Value

A new mistyR view. A list with a single named item described by the provided abbreviation and *data* containing the provided data.

See Also

[add_views\(\)](#) for adding created views to a view composition.

Other view composition functions: [add_juxtaview\(\)](#), [add_paraview\(\)](#), [add_views\(\)](#), [create_initial_view\(\)](#), [remove_views\(\)](#)

Examples

```
# Create a view from the mean expression of the 10 nearest neighbors of
# each cell.

library(dplyr)
library(purrr)
library(distances)

# get the expression data
data("synthetic")
expr <- synthetic[[1]] %>% select(-c(row, col, type))
# get the coordinates for each cell
pos <- synthetic[[1]] %>% select(row, col)

# find the 10 nearest neighbors
neighbors <- nearest_neighbor_search(distances(as.matrix(pos)), k = 11)[-1, ]

# calculate the mean expression of the nearest neighbors for all markers
# for each cell in expr
nexpr <- seq_len(nrow(expr)) %>%
  map_dfr(~ expr %>%
    slice(neighbors[, .x]) %>%
    colMeans())

create_view("nearest", nexpr, "nn")
```

plot_contrast_heatmap *Plot heatmap of local contrast between two views*

Description

The heatmap shows the interactions that are present and have importance above a cutoff value in the to.view but but not in the from.view.

Usage

```
plot_contrast_heatmap(misty.results, from.view, to.view, cutoff = 1)
```

Arguments

`misty.results` a results list generated by [collect_results\(\)](#).

`from.view, to.view`

abbreviated name of the view.

cutoff importance threshold. Importances below this value will be colored white in the heatmap and considered as not relevant.

Value

The `misty.results` list (invisibly).

See Also

[collect_results\(\)](#) to generate a results list from raw results.

Other plotting functions: [plot_contrast_results\(\)](#), [plot_improvement_stats\(\)](#), [plot_interaction_communities\(\)](#), [plot_interaction_heatmap\(\)](#), [plot_view_contributions\(\)](#)

Examples

```
all.samples <- list.dirs("results", recursive = FALSE)

misty.results <- collect_results(all.samples)

misty.results %>%
  plot_contrast_heatmap("intra", "para.10")

misty.results %>%
  plot_contrast_heatmap("intra", "para.10", cutoff = 0.5)
```

`plot_contrast_results` *Plot heatmap of contrast between two result lists*

Description

Plot interexperiment contrast of views.

Usage

```
plot_contrast_results(
  misty.results.from,
  misty.results.to,
  views = NULL,
  cutoff.from = 1,
  cutoff.to = 1
)
```

Arguments

`misty.results.from`, `misty.results.to`
a results list generated by [collect_results\(\)](#).

`views`
one or more abbreviated names of views.

`cutoff.from`, `cutoff.to`
importance thresholds respective to the result lists.

Details

The heatmaps show the interactions that are present and have importance above a `cutoff.to` value in the views of `misty.results.to` but not present or have importance below `cutoff.from` in the views of `misty.results.from`.

Value

The `misty.results.from` list (invisibly).

See Also

[collect_results\(\)](#) to generate a results list from raw results.

Other plotting functions: [plot_contrast_heatmap\(\)](#), [plot_improvement_stats\(\)](#), [plot_interaction_communities\(\)](#), [plot_interaction_heatmap\(\)](#), [plot_view_contributions\(\)](#)

Examples

```
# if for example the available samples come from different grades of tumors

grade1.results <- collect_results(c("results/synthetic1", "results/synthetic2"))
grade3.results <- collect_results("results/synthetic10")

# highlight interactions present in grade 1 tumors but not in grade 3 tumors
# in the paraview

grade3.results %>% plot_contrast_results(grade1.results, views = "para.10")

# see the loss of interactions in all views with lower sensitivity

plot_contrast_results(grade3.results, grade1.results, cutoff.from = 0.75, cutoff.to = 0.5)
```

plot_improvement_stats

Plot observed performance and improvement per target

Description

Generates a plot of the mean (+ standard deviation) of the performance value per target across all samples from the results.

Usage

```
plot_improvement_stats(misty.results, measure = "gain.R2", trim = -Inf)
```

Arguments

`misty.results` a results list generated by `collect_results()`.
`measure` performance measure to be plotted (See `collect_results()`).
`trim` display targets with performance value above (if R2 or gain) or below (otherwise) this value only.

Value

The `misty.results` list (invisibly).

See Also

`collect_results()` to generate a results list from raw results.

Other plotting functions: `plot_contrast_heatmap()`, `plot_contrast_results()`, `plot_interaction_communities()`, `plot_interaction_heatmap()`, `plot_view_contributions()`

Examples

```
all.samples <- list.dirs("results", recursive = FALSE)

collect_results(all.samples) %>% plot_improvement_stats()

misty.results <- collect_results(all.samples)
misty.results %>% plot_improvement_stats(measure = "gain.RMSE")
misty.results %>% plot_improvement_stats(measure = "intra.R2")
```

`plot_interaction_communities`

Plot marker interaction communities

Description

Identify and plot a graph of marker interaction communities.

Usage

```
plot_interaction_communities(misty.results, view, cutoff = 1)
```

Arguments

`misty.results` a results list generated by `collect_results()`.
`view` abbreviated name of the view.
`cutoff` importance threshold. Importances below this value will be colored white in the heatmap and considered as not relevant.

Details

The communities are identified using the Louvain algorithm. Communities can be extracted only from views that have the same predictor and target markers.

Value

The `misty.results` list (invisibly).

See Also

[collect_results\(\)](#) to generate a results list from raw results.

Other plotting functions: [plot_contrast_heatmap\(\)](#), [plot_contrast_results\(\)](#), [plot_improvement_stats\(\)](#), [plot_interaction_heatmap\(\)](#), [plot_view_contributions\(\)](#)

Examples

```
all.samples <- list.dirs("results", recursive = FALSE)

misty.results <- collect_results(all.samples)

misty.results %>%
  plot_interaction_communities("intra") %>%
  plot_interaction_communities("para.10")

misty.results %>%
  plot_interaction_communities("para.10", cutoff = 0.5)
```

plot_interaction_heatmap

Plot importance heatmap for a view

Description

Generate a heatmap with importances of predictor-target interaction.

Usage

```
plot_interaction_heatmap(misty.results, view, cutoff = 1, clean = FALSE)
```

Arguments

<code>misty.results</code>	a results list generated by collect_results() .
<code>view</code>	abbreviated name of the view.
<code>cutoff</code>	importance threshold. Importances below this value will be colored white in the heatmap and considered as not relevant.
<code>clean</code>	a logical indicating whether to remove rows and columns with all importances are below cutoff from the heatmap.

Value

The `misty.results` list (invisibly).

See Also

[collect_results\(\)](#) to generate a results list from raw results.

Other plotting functions: [plot_contrast_heatmap\(\)](#), [plot_contrast_results\(\)](#), [plot_improvement_stats\(\)](#), [plot_interaction_communities\(\)](#), [plot_view_contributions\(\)](#)

Examples

```
all.samples <- list.dirs("results", recursive = FALSE)

collect_results(all.samples) %>%
  plot_interaction_heatmap("intra") %>%
  plot_interaction_heatmap("para.10", cutoff = 0.5)
```

plot_view_contributions

Plot view contributions per target

Description

Generate a stacked barplot of the average view contribution fraction per target across all samples from the results.

Usage

```
plot_view_contributions(misty.results, trim = -Inf, trim.measure = "gain.R2")
```

Arguments

<code>misty.results</code>	a results list generated by collect_results() .
<code>trim</code>	display targets with performance value above (if R2 or gain) or below (otherwise) this value only.
<code>trim.measure</code>	the measure used for trimming.

Value

The `misty.results` list (invisibly).

See Also

[collect_results\(\)](#) to generate a results list from raw results.

Other plotting functions: [plot_contrast_heatmap\(\)](#), [plot_contrast_results\(\)](#), [plot_improvement_stats\(\)](#), [plot_interaction_communities\(\)](#), [plot_interaction_heatmap\(\)](#)

Examples

```
all.samples <- list.dirs("results", recursive = FALSE)

collect_results(all.samples) %>% plot_view_contributions()
```

remove_views	<i>Remove views from the current view composition</i>
--------------	---

Description

Remove one or more views from the view composition.

Usage

```
remove_views(current.views, view.names)
```

Arguments

`current.views` the current view composition.
`view.names` the names of one or more views to be removed.

Details

The intraview and the unique id cannot be removed with this function.

Value

A mistyR view composition with `view.names` views removed.

See Also

Other view composition functions: [add_juxtaview\(\)](#), [add_paraview\(\)](#), [add_views\(\)](#), [create_initial_view\(\)](#), [create_view\(\)](#)

Examples

```
library(dplyr)

# get the expression data
data("synthetic")
expr <- synthetic[[1]] %>% select(-c(row, col, type))
# get the coordinates for each cell
pos <- synthetic[[1]] %>% select(row, col)

# compose
misty.views <- create_initial_view(expr) %>%
  add_juxtaview(pos, neighbor.thr = 1.5) %>%
  add_paraview(pos, l = 10)
```



```

# preview
str(misty.views)

# remove juxtaview and preview
misty.views %>%
  remove_views("juxtaview.1.5") %>%
  str()

# remove juxtaview and paraview and preview
misty.views %>%
  remove_views(c("juxtaview.1.5", "paraview.10")) %>%
  str()

```

run_misty

Train MISTy models

Description

Trains multi-view models for all target markers, estimates the performance, the contributions of the view specific models and the importance of predictor markers for each target marker.

Usage

```

run_misty(
  views,
  results.folder = "results",
  seed = 42,
  target.subset = NULL,
  cv.folds = 10,
  cached = FALSE,
  append = FALSE,
  ...
)

```

Arguments

views	view composition.
results.folder	path to the top level folder to store raw results.
seed	seed used for random sampling to ensure reproducibility.
target.subset	subset of targets to train models for. If NULL, models will be trained for markers in the intraview.
cv.folds	number of cross-validation folds to consider for estimating the performance of the multi-view models.
cached	a logical indicating whether to cache the trained models and to reuse previously cached ones if they already exist for this sample.

append	a logical indicating whether to append the performance and coefficient files in the <code>results.folder</code> . Consider setting to <code>TRUE</code> when rerunning a workflow with different <code>target.subset</code> parameters.
...	all additional parameters are passed to <code>ranger()</code> for training the view-specific models (see Details for defaults).

Details

Default values passed to `ranger()` for training the view-specific models: `num.trees = 100`, `importance = "impurity"`, `num.threads = 1`, `seed = seed`.

Value

Path to the results folder that can be passed to `collect_results()`.

See Also

`create_initial_view()` for starting a view composition.

Examples

```
# Create a view composition of an intraview and a paraview with radius 10 then
# run MISTy for a single sample.

library(dplyr)

# get the expression data
data("synthetic")
expr <- synthetic[[1]] %>% select(-c(row, col, type))
# get the coordinates for each cell
pos <- synthetic[[1]] %>% select(row, col)

# compose
misty.views <- create_initial_view(expr) %>% add_paraview(pos, l = 10)

# run with default parameters
run_misty(misty.views)
```

synthetic

Synthetic benchmark data for mistyR

Description

Data generated from 10 random layouts of four cell types and empty space on 100-by-100 grid by simulating a two-dimensional cellular automata model that focuses on signaling events. Cell growth, division, motility and death are neglected. The intracellular processes involve two layers, first the ligand activation of signaling hubs and ligand production/secretion regulated by proteins. The model simulates the production, diffusion, degradation and interactions of 11 molecular species. Ligands are produced in each cell-type based on the activity level of their production nodes

and then freely diffuse, degrade or interact with other cells on the grid. Other molecular species involved in signaling are localised in the intracellular space and their activity depends on ligand binding and intracellular wiring.

Usage

```
data("synthetic")
```

Format

A named list of length 10. Each list item is a tibble that corresponds to a simulation of one random layout with information about each cell in rows described by the following 14 variables:

row, col location of the cell on the grid

ligA, ligB, ligC, ligD expression of ligands

protE, protF expression of intracellular proteins

prodA, prodB, prodC, prodD expression of regulatory proteins

type cell type id

Source

https://github.com/saezlab/misty_pipelines/

Index

* datasets

synthetic, [18](#)

* plotting functions

plot_contrast_heatmap, [10](#)

plot_contrast_results, [11](#)

plot_improvement_stats, [12](#)

plot_interaction_communities, [13](#)

plot_interaction_heatmap, [14](#)

plot_view_contributions, [15](#)

* view composition functions

add_juxtview, [2](#)

add_paraview, [4](#)

add_views, [5](#)

create_initial_view, [8](#)

create_view, [9](#)

remove_views, [16](#)

[add_juxtview, 2, 5, 6, 9, 10, 16](#)

[add_paraview, 3, 4, 6, 9, 10, 16](#)

[add_views, 3, 5, 5, 9, 10, 16](#)

[clear_cache, 6](#)

[collect_results, 7, 10–15, 18](#)

[create_initial_view, 3, 5, 6, 8, 10, 16, 18](#)

[create_view, 3, 5, 6, 9, 9, 16](#)

[plot_contrast_heatmap, 10, 12–15](#)

[plot_contrast_results, 11, 11, 13–15](#)

[plot_improvement_stats, 11, 12, 12, 14, 15](#)

[plot_interaction_communities, 11–13, 13, 15](#)

[plot_interaction_heatmap, 11–14, 14, 15](#)

[plot_view_contributions, 11–15, 15](#)

[ranger, 18](#)

[remove_views, 3, 5, 6, 9, 10, 16](#)

[run_misty, 7, 8, 17](#)

[synthetic, 18](#)