Package 'PathwaySpace'

August 24, 2026

Type Package

Title Spatial Projection of Network Signals along Geodesic Paths

Version 1.5.1

Depends R(>= 4.5), methods, RGraphSpace(>= 1.4.1)

Imports grDevices, rlang, stats, scales, RANN, igraph, ggplot2, grid, ggnewscale, ggrepel, colorspace, lifecycle

Suggests knitr, rmarkdown, testthat, RedeR, patchwork

Description For a given graph containing vertices, edges, and a signal associated with the vertices, the 'PathwaySpace' package performs a convolution operation, which involves a weighted combination of neighboring vertices and their associated signals. The package uses a decay function to project these signals, creating geodesic paths on a 2D-image space. 'PathwaySpace' has various applications, such as visualizing network data in a graphical format that highlights the relationships and signal strengths between vertices. By combining graph theory, signal processing, and visualization, 'PathwaySpace' provides a way of representing graph data on a continuous projection space. Based on methods introduced in Tercan et al. (2025) <doi:10.1016/j.xpro.2025.103681> and Ellrott et al. (2025) <doi:10.1016/j.ccell.2024.12.002>.

License Artistic-2.0

VignetteBuilder knitr

URL https://sysbiolab.github.io/PathwaySpace/, https://github.com/sysbiolab/PathwaySpace

BugReports https://github.com/sysbiolab/PathwaySpace/issues

Collate pspace-checks.R pspace-supplements.R pspace-misc.R pspace-watershed.R pspace-decay.R pspace-classes.R pspace-generics.R pspace-methods.R pspace-plots.R annotation-pspace.R

Encoding UTF-8

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Sysbiolab Team [aut],
Victor Apolonio [ctb],
Jonathan Back [ctb],
Lana Querne [ctb],
Vinicius Chagas [ctb],
Bahar Tercan [ctb],
Mauro Castro [cre] (ORCID: <https://orcid.org/0000-0003-4942-8131>)

Maintainer Mauro Castro <mauro.a.castro@gmail.com>

Repository CRAN

Date/Publication 2026-08-24 13:00:08 UTC

Contents

annotation_pspace_signal
Annotation Functions for PathwaySpace Plots

Description

annotation_pspace_signal() annotates a ggplot-based PathwaySpace plot with the projected signal layer, rendered as a raster heatmap bounded to the normalized unit space [0, 1].

Usage

```
annotation_pspace_signal(
  ps,
  title = activeFeature(ps),
  colors = pspace.cols(),
  bg.color = "grey95",
  si.color = "grey85",
  si.alpha = 1,
  zlab = "Density",
  zlim = NULL,
  slices = 25,
  interpolate = FALSE,
  ...
)
```

Arguments

ps	A PathwaySpace object containing a valid signal projection. Run circularProjection or polarProjection before calling this function.
title	A string used as the plot title. Defaults to the active feature name returned by activeFeature.
colors	A character vector of colors used to build the signal palette. Defaults to pspace.cols.
bg.color	A string specifying the background color, used for zero-signal or masked regions. Defaults to "grey95".
si.color	A single color for silhouette. (see silhouetteMapping).
si.alpha	A transparency level in [0, 1], used to adjust the opacity of the silhouette. This parameter is useful for improving the perception of a background image, when one is available.
zlab	The title for the 'z' axis of the image signal.
zlim	The 'z' limits of the plot (a numeric vector with two numbers). If NULL, limits are determined from the range of the input values.
slices	An integer specifying the number of discrete color levels used to quantize the signal. For "negpos" scale types, this is rounded up to the nearest even number.
interpolate	A logical value indicating whether to apply linear interpolation with geom_raster.
...	Additional parameters passed to the title annotation; see geom_text.

Value

A list of ggplot2 layer objects that can be added to a ggplot() call with +.

See Also

circularProjection

Examples

```
data("gtoy1", package = "RGraphSpace")
ps <- buildPathwaySpace(gtoy1, nrc = 100)
vertexSignal(ps) <- 1
ps <- circularProjection(ps)

## Not run:
ggplot(ps) +
  annotation_pspace_signal(ps, si.alpha = 0.8) +
  theme_gspace_coords(is_norm = TRUE)

## End(Not run)
```

---

buildPathwaySpace *Constructor of PathwaySpace-class Objects*

Description

buildPathwaySpace is a constructor of PathwaySpace-class objects.

Usage

```
buildPathwaySpace(gs, nrc = 500, verbose = TRUE)
```

Arguments

gs	A GraphSpace object. Alternatively, an igraph object with node coordinates assigned to x and y vertex attributes, and node labels assigned to name vertex attribute.
nrc	A single positive integer indicating the number of rows and columns (in pixels) for a square image matrix. This argument will affect the resulting image size and resolution.
verbose	A logical value specifying to display detailed messages (when verbose=TRUE) or not (when verbose=FALSE).

Value

A pre-processed PathwaySpace class object.

Author(s)

Sysbiolab Team

See Also

circularProjection, polarProjection

Examples

```
library(PathwaySpace)

# Load a demo igraph
data('gtoy1', package = 'RGraphSpace')

# Check graph validity
gs <- GraphSpace(gtoy1)

gs <- normalizeGraphSpace(gs)

# Create a new PathwaySpace object
ps <- buildPathwaySpace(gs, nrc = 100)
# note: adjust 'nrc' to increase image resolution
```

CGC_20211118 *COSMIC-CGC genes mapped to PathwaySpace images*

Description

A data frame listing 'GeneSymbol' and 'Entrez' IDs from the COSMIC-CGC database (Sondka et al., 2020). These genes are used to demonstrate the PathwaySpace's summit mapping pipeline, which assigns summits to an image space.

Usage

```
data(CGC_20211118)
```

Format

data.frame

Value

A data.frame object.

Source

COSMIC-CGC database (release v95, tier 1 collection).

References

Sondka et al. The COSMIC Cancer Gene Census: describing genetic dysfunction across all human cancers. Nat Rev Cancer 18, 696-705, 2018. Doi: 10.1038/s41568-018-0060-1.

Examples

```
data(CGC_20211118)
```

---

circularProjection,PathwaySpace-method

Circular Projection of Graph-Associated Signals

Description

`circularProjection()` implements a convolution algorithm to project vertex-associated signals onto a 2D image space using a circular decay function.

Usage

```
## S4 method for signature 'PathwaySpace'
circularProjection(
  ps,
  feature = activeFeature(ps),
  decay.fun = weibullDecay(),
  aggregate.fun = signalAggregation(),
  k = gs_vcount(ps),
  rescale = TRUE,
  verbose = TRUE,
  pdist = deprecated()
)
```

Arguments

`ps`	A PathwaySpace class object.
`feature`	A single string specifying the feature to project as a signal. Must match either a feature name (see `gs_features(ps)`) or a node attribute (see `gs_names(ps)`). If a node attribute, make sure it is of numeric type. If the signal does not come from internal features, assign it directly using the `vertexSignal` accessor.
`decay.fun`	A signal decay function. Available options include 'Weibull', 'exponential', and 'linear' (see `weibullDecay`). Users may also define a custom decay model with at least two arguments, e.g., `function(x, signal) { ... }`, which should returns a vector of projected signals of the same length as `x`. Additional arguments may include any variable available as a graph vertex attribute.
`aggregate.fun`	A function used to aggregate the projected signals. It must be provided as a unary function, e.g., `function(x) { ... }`, which should aggregate a vector of signals to a scalar value. Available options include 'mean', 'wmean', 'log.wmean', and 'exp.wmean' (See `signalAggregation`).

k	A single positive integer specifying the maximum number of vertices whose signals contribute to the projection. Defaults to `gs_vcount(ps)`, i.e. all vertices are considered. Specifically, at each point in space, the *k*-top decayed signals are retained prior to aggregation. Reducing *k* focuses the projection on the strongest local signals, filtering out weaker contributions.
rescale	A logical value indicating whether to rescale the signal. If the signal `>=0`, then it will be rescaled to `[0, 1]`; if the signal `<=0`, then it will be rescaled to `[-1, 0]`; and if the signal in `(-Inf, +Inf)`, then it will be rescaled to `[-1, 1]`.
verbose	A logical value specifying to display detailed messages (when `verbose=TRUE`) or not (when `verbose=FALSE`).
pdist	Deprecated as of PathwaySpace 1.0.2; this parameter is now passed internally through `decay.fun`.

Value

A preprocessed PathwaySpace class object.

Author(s)

Sysbiolab Team

See Also

buildPathwaySpace

Examples

```
library(PathwaySpace)

# Load a demo igraph
data('gtoy1', package = 'RGraphSpace')

# Create a new PathwaySpace object
ps <- buildPathwaySpace(gtoy1, nrc = 100)
# note: adjust 'nrc' to increase image resolution

# Set '1s' as vertex signal
vertexSignal(ps) <- 1

# Create a 2D-landscape image
ps <- circularProjection(ps)
```

expDecay *Constructor of exponential decay functions*

Description

The `expDecay()` constructor either creates a decay function or returns a `ggplot` object for visualizing the decay model. It is a utility function used internally by `circularProjection` and `polarProjection`.

Usage

```
expDecay(decay = 0.001, pdist = 0.15, plot = FALSE, demo.signal = 1)
```

Arguments

`decay`	A decay factor (in `[0,1]`). This term indicates how much a `signal` decreases as a function of distance in pathway space. For example, at a specific distance defined by the `pdist` parameter, the signal intensity will be the initial signal multiplied by `decay`.
`pdist`	A distance normalization term (in (0, 1]) at which the signal reaches `signal * decay`. This parameter is used to anchor the decay to a meaningful distance (see `details`). Also, when `pdist = 1`, it will represent the diameter of the inscribed circle within the coordinate space of a `PathwaySpace` object.
`plot`	A logical value indicating whether to return a `ggplot` object.
`demo.signal`	A numeric value in `[-Inf, Inf]`, only passed when `plot = TRUE` to visualize the decay curve with a specific signal intensity. The value is ignored by the function constructor, as the decay function itself is returned without using an initial signal.

Details

The `expDecay()` constructor creates an exponential decay model. It describes how a signal decreases as a function of distance, controlled by a decay rate parameter.

The decay function is defined as:

$$y = signal \times decay^{\left(\frac{x}{pdist}\right)}$$

where $signal$ represents the initial intensity, $decay$ controls the rate of attenuation, and x is a vector of normalized distances. The $pdist$ parameter anchors the model such that:

- $y = signal$ when $x = 0$
- $y = signal \times decay$ when $x = pdist$

Value

Returns either a function of the form `function(x, signal) { ... }` or, if `plot = TRUE`, a `ggplot` object illustrating the decay model.

Author(s)

Sysbiolab Team

See Also

linearDecay, weibullDecay

Examples

```
# Return a decay function
decay_fun <- expDecay(decay = 0.25, pdist = 0.5)

# Plot decay model parameters
# expDecay(decay = 0.25, pdist = 0.5, plot = TRUE)
```

---

getNearestNode *getNearestNode*

Description

Retrieves the nearest neighbor for each node from a PathwaySpace object using Euclidean distance.

Usage

```
getNearestNode(ps)
```

Arguments

ps	Either a PathwaySpace or GraphSpace object.

Value

A `data.frame` with columns `from`, `to`, and `dist`, listing each node's nearest neighbor and the Euclidean distance between them.

See Also

nn2

Examples

```
# See examples in the PathwaySpace's tutorials:
# https://sysbiolab.github.io/PathwaySpace/
```

`getPathwaySpace,PathwaySpace-method`
Accessors for Fetching Slots from a PathwaySpace Object

Description

`getPathwaySpace` retrives information from individual slots available in a PathwaySpace object.

Usage

```
## S4 method for signature 'PathwaySpace'
getPathwaySpace(ps, what = "status")
```

Arguments

`ps`	A preprocessed PathwaySpace class object
`what`	A character value specifying which information should be retrieved from the slots. Options: "nodes", "edges", "graph", "image", "pars", "misc", "signal","projection", "status", "silhouette", "summits", "summit_mask", "summit_contour"

Value

Content from slots in the PathwaySpace object.

Examples

```
library(PathwaySpace)

# Load a demo igraph
data('gtoy1', package = 'RGraphSpace')

# Create a new PathwaySpace object
ps <- buildPathwaySpace(gtoy1, nrc = 100)
# note: adjust 'nrc' to increase image resolution

# Get the 'status' slot in ps
status <- getPathwaySpace(ps, what = 'status')
```

gimage *An image matrix*

Description

An image matrix used for workflow demonstrations.

Usage

```
data(gimage)
```

Format

matrix

Value

An image matrix.

Source

This package.

Examples

```
data(gimage)
```

Hallmarks_v2023_1_Hs_symbols *A list with Hallmark gene sets (v2023.1)*

Description

A list with Human gene symbols from the MSigDB's Hallmark gene set collection (Liberzon et al., 2015). These gene sets are used to demonstrate the PathwaySpace's summit mapping pipeline, which assigns summits to an image space.

Usage

```
data(Hallmarks_v2023_1_Hs_symbols)
```

Format

list

Value

A list object.

Source

MSigDB database (v2023.1).

References

Liberzon et al. The Molecular Signatures Database (MSigDB) hallmark gene set collection. Cell Systems 1(5):417-425, 2015 Doi: 10.1016/j.cels.2015.12.004

Examples

```
data(Hallmarks_v2023_1_Hs_symbols)
```

---

linearDecay *Constructor of linear decay functions*

Description

The `linearDecay()` constructor either creates a decay function or returns a `ggplot` object for visualizing the decay model. It is a utility function used internally by `circularProjection` and `polarProjection`.

Usage

```
linearDecay(decay = 0.001, pdist = 0.15, plot = FALSE, demo.signal = 1)
```

Arguments

`decay`	A decay factor (in `[0,1]`). This term indicates how much a `signal` decreases as a function of distance in pathway space. For example, at a specific distance defined by the `pdist` parameter, the signal intensity will be the initial signal multiplied by `decay`.
`pdist`	A distance normalization term (in (0, 1]) at which the signal reaches `signal * decay`. This parameter is used to anchor the decay to a meaningful distance (see `details`). Also, when `pdist = 1`, it will represent the diameter of the inscribed circle within the coordinate space of a `PathwaySpace` object.
`plot`	A logical value indicating whether to return a `ggplot` object.
`demo.signal`	A numeric value in `[-Inf, Inf]`, only passed when `plot = TRUE` to visualize the decay curve with a specific signal intensity. The value is ignored by the function constructor, as the decay function itself is returned without using an initial signal.

Details

The `linearDecay()` constructor creates a simple linear decay model. It describes how a signal decreases proportionally with distance.

The decay function is defined as:

$$y = signal \times \left(1 - (1 - decay) \times \frac{x}{pdist}\right)$$

where $signal$ represents the initial intensity, $decay$ defines the relative signal level at $pdist$, and x is a vector of normalized distances. The signal decreases uniformly from its initial value to $pdist$, which is a reference distance that anchors the model such that:

- $y = signal$ when $x = 0$
- $y = signal \times decay$ when $x = pdist$

This makes the linear form consistent with the exponential and Weibull decay functions, both of which also reach $signal \times decay$ at the reference distance.

Value

Returns either a function of the form `function(x, signal) { ... }` or, if `plot = TRUE`, a `ggplot` object illustrating the decay model.

Author(s)

Sysbiolab Team

See Also

`expDecay`, `weibullDecay`

Examples

```
# Return a decay function
decay_fun <- linearDecay(decay = 0.5, pdist = 0.25)

# Plot decay model parameters
# linearDecay(decay = 0.5, pdist = 0.25, plot = TRUE)
```

pathDistances *Calculate a pathway space distance between two vectors*

Description

Calculate a pathway space distance between two vectors

Usage

```
pathDistances(gdist, from, to, nperm = 1000, verbose = TRUE)
```

Arguments

gdist	A distance matrix computed by the igraph's distances function. Rows and columns must be named with vertex labels as listed in the 'igraph' object.
from	A vector with valid vertex names.
to	A vector with valid vertex names.
nperm	Number of permutations.
verbose	A single logical value specifying to display detailed messages (when verbose=TRUE) or not (when verbose=FALSE).

Value

A list with pathway space distances and a 'ggplot' object.

See Also

plotPathwaySpace

Examples

```
# Load a vertex-wise distance matrix (distance between nodes in a graph)
data("gdist.toy", package = "PathwaySpace")

# Get two vertex lists
from <- sample(colnames(gdist.toy), 50)
to <- sample(colnames(gdist.toy), 50)

# Calculate distances between lists, and between random lists
res <- pathDistances(gdist.toy, from, to)
names(res)
# "p_dist"  "z_score"
```

PathwaySpace-accessors *Accessor Functions for PathwaySpace Objects*

Description

Get or set edge and vertex attributes in PathwaySpace class object.

Usage

```
## S4 replacement method for signature 'PathwaySpace'
gs_vertex_attr(x, name, ...) <- value

## S4 replacement method for signature 'PathwaySpace'
gs_edge_attr(x, name, ...) <- value
```

Arguments

x	A PathwaySpace class object.
name	Name of the attribute.
...	Additional arguments passed to igraph methods.
value	The new value of the attribute.

Value

Updated PathwaySpace object.

Examples

```
library(PathwaySpace)

# Load a demo igraph
data('gtoy1', package = 'RGraphSpace')
ps <- buildPathwaySpace(gtoy1, nrc = 100)

# Get vertex count
gs_vcount(ps)

# Get edge count
gs_ecount(ps)

# Access a specific vertex attribute
gs_vertex_attr(ps, "signal")

# Replace an entire vertex attribute
gs_vertex_attr(ps, "signal") <- 1

# Modify a single value within a vertex attribute
```

```
gs_vertex_attr(ps, "signal")["n1"] <- 1

# Access a specific edge attribute
gs_edge_attr(ps, "weight")

# Replace an entire edge attribute
gs_edge_attr(ps, "weight") <- 1
```

`PathwaySpace-class` *PathwaySpace: An S4 class for signal projection on image spaces*

Description

`PathwaySpace` extends the GraphSpace class with signal projection slots. It stores projected signal matrices, projection parameters, and workflow status, and is the main object used by the **PathwaySpace** package.

Value

A `PathwaySpace` object.

Slots

`projection` A SpaceProjection object storing the intermediate and final matrices produced by a projection method.

`pars_ps` A list with PathwaySpace parameters.

`status` A vector containing the processing status of the PathwaySpace object.

Constructor

See `buildPathwaySpace`.

Author(s)

Sysbiolab Team, Mauro Castro (<mauro.castro@ufpr.br>)

See Also

GraphSpace, SpaceProjection, `buildPathwaySpace`, `circularProjection`

PCv12_pruned_igraph *A pruned and laid out igraph object from Pathway Commons V12*

Description

This igraph object was created from a 'sif' file available from the Pathway Commons V12 (Rodchenkov et al., 2020), which was filtered to keep interactions from the following sources: CTD, Recon, HumanCyc, DrugBank, MSigDB, DIP, BioGRID, IntAct, BIND, and PhosphoSite. The igraph was additionally pruned and laid out by a force-directed algorithm aiming signal projection on PathwaySpace's images. Edges with the smallest betweenness centrality were pruned using 'backward elimination' and 'forward selection' strategies. The resulting graph represents the main connected component with the minimum number of edges.

Usage

```
data(PCv12_pruned_igraph)
```

Format

igraph

Value

An igraph object.

Author(s)

Chris Wong, Mauro Castro, and TCGA Network.

Source

Pathway Commons V12.

References

Rodchenkov et al. Pathway Commons 2019 Update: integration, analysis and exploration of pathway data. Nucleic Acids Research 48(D1):D489–D497, 2020. doi:10.1093/nar/gkz946

Examples

```
data(PCv12_pruned_igraph)
## Suggestion to vizualize this igraph in R:
library(RGraphSpace)
plotGraphSpace(PCv12_pruned_igraph)
```

plotPathDistances *Accessory function to plot pathway space distances*

Description

Accessory function to plot pathway space distances

Usage

```
plotPathDistances(pdist, z.transform = FALSE)
```

Arguments

pdist	A list generated by the pathDistances function.
z.transform	A single logical value specifying to convert pathway distances into z-score values.

Value

A 'ggplot' object.

Examples

```
# Load a vertex-wise distance matrix (distance between nodes in a graph)
data("gdist.toy", package = "PathwaySpace")

# Get two gene lists
from <- sample(colnames(gdist.toy), 50)
to <- sample(colnames(gdist.toy), 50)

# Calculate distances between lists, and between random lists
res <- pathDistances(gdist.toy, from, to)

# Plot observed and null distances
plotPathDistances(res)
```

plotPathwaySpace,PathwaySpace-method *Plotting 2D-landscape images for the PathwaySpace package*

Description

plotPathwaySpace is a wrapper function to create dedicated ggplot graphics for PathwaySpace-class objects.

Usage

```
## S4 method for signature 'PathwaySpace'
plotPathwaySpace(
  ps,
  title = activeFeature(ps),
  colors = pspace.cols(),
  bg.color = "grey95",
  si.color = "grey85",
  si.alpha = 1,
  theme = "th0",
  xlab = "Graph coordinates 1",
  ylab = "Graph coordinates 2",
  zlab = "Density",
  font.size = 1,
  font.color = "white",
  zlim = NULL,
  slices = 25,
  add.image = FALSE,
  add.grid = TRUE,
  grid.color = "white",
  marks = FALSE,
  mark.size = 3,
  mark.color = "white",
  mark.padding = 0.5,
  mark.line.width = 0.5,
  use.dotmark = FALSE,
  add.summits = TRUE,
  label.summits = TRUE,
  summit.color = "white",
  add.marks = deprecated()
)
```

Arguments

ps	A PathwaySpace class object.
title	A string for the title.
colors	A vector of colors.
bg.color	A single color for background.
si.color	A single color for silhouette. (see silhouetteMapping).
si.alpha	A transparency level in [0, 1], used to adjust the opacity of the silhouette. This parameter is useful for improving the perception of a background image, when one is available.
theme	Name of a custom PathwaySpace theme. These themes (from 'th0' to 'th3') consist mainly of preconfigured ggplot settings, which the user can subsequently refine using ggplot2.
xlab	The title for the 'x' axis of a 2D-image space.

ylab	The title for the 'y' axis of a 2D-image space.
zlab	The title for the 'z' axis of the image signal.
font.size	A single numeric value passed to plot annotations.
font.color	A single color passed to plot annotations.
zlim	The 'z' limits of the plot (a numeric vector with two numbers). If NULL, limits are determined from the range of the input values.
slices	A single positive integer value used to split the image signal into equally-spaced intervals.
add.image	A logical value indicating whether to add a background image, when one is available (see GraphSpace).
add.grid	A logical value indicating whether to add gridlines to the image space. However, gridlines will only appear when the image is decorated with graph silhouettes (see silhouetteMapping).
grid.color	A color passed to geom_point.
marks	A vector of vertex names to be highlighted in the image space. This argument overrides 'add.labels'.
mark.size	A size argument passed to geom_text.
mark.color	A color passed to geom_text.
mark.padding	A box padding argument passed to geom_text_repel.
mark.line.width	A line width argument passed to geom_text_repel.
use.dotmark	A logical value indicating whether "marks" should be represented as dots.
add.summits	A logical value indicating whether to add contour lines to 'summits' (when summits are available; see summitMapping).
label.summits	A logical value indicating whether to label summits.
summit.color	A color passed to 'summits'.
add.marks	Deprecated. Use marks instead.

Value

A ggplot-class object.

Author(s)

Sysbiolab Team, Mauro Castro.

See Also

circularProjection, polarProjection

Examples

```
# Load a demo igraph
data('gtoy1', package = 'RGraphSpace')

# # Check graph validity
gs <- GraphSpace(gtoy1)

gs <- normalizeGraphSpace(gs)

# Create a PathwaySpace object
ps <- buildPathwaySpace(gs, nrc = 300)
# note: adjust 'nrc' to increase image resolution

# Set '1s' as vertex signal
vertexSignal(ps) <- 1

# Create a 2D-landscape image
ps <- circularProjection(ps, k = 2,
   decay.fun = weibullDecay(pdist = 0.4))

# Plot a 2D-landscape image
plotPathwaySpace(ps)
```

---

polarDecay *Polar transformation functions*

Description

Creates polar transformation functions for `polarProjection` internal calls. These functions are used to adjusts signal decay according to point-to-edge angular distances, with options to attenuate angular shapes.

Usage

```
polarDecay(
  method = c("power", "gaussian", "logistic"),
  s = 0.5,
  k = 10,
  m = 0.5
)
```

Arguments

`method`	String indicating the transformation to apply. Must be one of: "power", "gaussian", or "logistic".
`s`	Single numeric value in `[0, 1]`. Controls the spread around the `x` mean of the Gaussian function.

k	Single numeric value >=1. Controls the steepness of the logistic function.
m	Single numeric value in [0, 1]. Specifies the midpoint of the logistic function.

Details

The polar transformation controls how much the projected signal decays as a function of the angular distance between a point in pathway space and a reference edge axis. The function returned by `polarDecay()` expects two arguments, with the following signature: `function(x, beta) { ... }`.

Power:

$$x^{\beta}$$

where x is a vector of normalized angular distances (in [0, 1]) and $beta$ is a non-negative exponent that controls the rate of signal decay. Increasing $beta$ results in a steeper decay rate, modulating the angular span of the projection.

Gaussian:

$$\exp\left(-\frac{(1-x)^2}{2\sigma^2}\right)^{\beta}$$

where $sigma$ controls the spread around the mean, creating fuzzier effect on projections.

Logistic:

$$(1/(1+\exp(k(x-m))))^{\beta}$$

where k is the steepness and m is the function's midpoint, making more gradual transitions.

These transformations are intended to be plugged into the higher-level `polarProjection` function, allowing user control over the polar projection profiles.

Value

Returns a function of the form: `function(x, beta) { ... }`, that applies the specified shape-based transformation.

Author(s)

Sysbiolab Team

See Also

`polarProjection`

Examples

```
polar.fun <- polarDecay("power")
```

polarProjection,PathwaySpace-method

Polar Projection of Graph-Associated Signals

Description

polarProjection() implements a convolution algorithm to project vertex-associated signals onto a 2D image space along graph edges, using a polar decay function.

Usage

```
## S4 method for signature 'PathwaySpace'
polarProjection(
  ps,
  feature = activeFeature(ps),
  decay.fun = weibullDecay(pdist = 1),
  aggregate.fun = signalAggregation(),
  polar.fun = polarDecay(),
  k = gs_vcount(ps),
  beta = 10,
  directional = FALSE,
  edge.norm = TRUE,
  rescale = TRUE,
  verbose = TRUE,
  pdist = deprecated()
)
```

Arguments

ps	A PathwaySpace class object.
feature	A single string specifying the feature to project as a signal. Must match either a feature name (see gs_features(ps)) or a node attribute (see gs_names(ps)). If a node attribute, make sure it is of numeric type. If the signal does not come from internal features, assign it directly using the vertexSignal accessor.
decay.fun	A signal decay function. Available options include 'Weibull', 'exponential', and 'linear' (see weibullDecay). Users may also define a custom decay model with at least two arguments, e.g., function(x, signal) { ... }, which should returns a vector of projected signals of the same length as x. Additional arguments may include any variable available as a graph vertex attribute.
aggregate.fun	A function used to aggregate the projected signals. It must be provided as a unary function, e.g., function(x) { ... }, which should aggregate a vector of signals to a scalar value. Available options include 'mean', 'wmean', 'log.wmean', and 'exp.wmean' (See signalAggregation).
polar.fun	A polar decay function (see polarDecay).

k	A single positive integer specifying the maximum number of vertices whose signals contribute to the projection. Defaults to `gs_vcount(ps)`, i.e. all vertices are considered. Specifically, at each point in space, the *k*-top decayed signals are retained prior to aggregation. Reducing *k* focuses the projection on the strongest local signals, filtering out weaker contributions.
beta	An exponent (in `>=0`)) used in the polar projection functions (see `polarDecay`). It controls the shape of the polar projection by modulating the angular span. For example, $beta = 0$ yields a circular projection, $beta = 1$ produces a cardioid-like shape, and `beta > 1` progressively narrows the projection along a reference edge axis.
directional	If directional edges are available, this argument can be used to orientate the signal projection on directed graphs.
edge.norm	Scale distances based on edge lengths (when `edge.norm=TRUE`) or based on full coordinate space (when `edge.norm=FALSE`).
rescale	A logical value indicating whether to rescale the signal. If the signal `>=0`, then it will be rescaled to `[0, 1]`; if the signal `<=0`, then it will be rescaled to `[-1, 0]`; and if the signal in `(-Inf, +Inf)`, then it will be rescaled to `[-1, 1]`.
verbose	A logical value specifying to display detailed messages (when `verbose=TRUE`) or not (when `verbose=FALSE`).
pdist	Deprecated as of PathwaySpace 1.0.2; this parameter is now passed internally through `decay.fun`.

Details

Nodes without edges (isolated nodes) still receive a projection: their signal is spread as a circle whose area matches that of a minimally connected (degree-1) node, so isolated nodes do not appear disproportionately large or small relative to connected ones. This treatment is the same whether `directional` is `TRUE` or `FALSE`. The area-matching guarantee is derived for the default `polarDecay("power")` method; it is not guaranteed to hold exactly for the `"gaussian"` or `"logistic"` alternatives.

Value

A preprocessed PathwaySpace class object.

Author(s)

Sysbiolab Team

See Also

`buildPathwaySpace`

Examples

```
library(PathwaySpace)

# Load a demo igraph
```

```
data('gtoy2', package = 'RGraphSpace')

# Create a new PathwaySpace object
ps <- buildPathwaySpace(gtoy2, nrc = 300)
# note: adjust 'nrc' to increase image resolution

# Set '1s' as vertex signal
vertexSignal(ps) <- 1

# Set edge weight
# gs_edge_attr(ps, "weight") <- c(-1, 1, 1, 1, 1, 1)

# Set a decay function for all vertices
vertexDecay(ps) <- weibullDecay(shape=2, pdist = 0.2)

# Create a 2D-landscape image
ps <- polarProjection(ps, beta = 5)

plotPathwaySpace(ps)
```

`pspace.cols` *A simple vector of colors for PathwaySpace images*

Description

A simple vector of colors for PathwaySpace images

Usage

```
pspace.cols(n = 25)
```

Arguments

`n`	The number of colors to generate in the output palette.

Value

A vector with hexadecimal color codes.

See Also

`plotPathwaySpace`, `pspace.pals`

Examples

```
pspace.cols()
```

`pspace.pals` *Create interpolated color palettes for PathwaySpace images*

Description

Creates mixed color palettes by interpolating and offsetting hues, useful for generating transitions between hues.

Usage

```
pspace.pals(
  colors = c("#303f9d", "#578edb", "#63b946", "#f3930c", "#a60d0d"),
  trim.colors = c(3, 2, 1, 2, 3),
  offset = 0.5,
  n = 25
)
```

Arguments

`colors`	A vector of five base colors used to construct the custom diverging palette. These colors are interpolated according to the `trim.colors` values.
`trim.colors`	A vector of five positive integers that control the relative weight of each hue in the five-color diverging palette.
`offset`	Adjusts brightness by shifting hues toward the center, either brighter (`offset > 0`) or darker (`offset < 0`).
`n`	The number of colors to generate in the output palette.

Value

A vector with hexadecimal color codes.

See Also

`plotPathwaySpace`

Examples

```
pspace.pals()
```

`signalAggregation` *Signal aggregation functions*

Description

Signal aggregation functions for `circularProjection` and `polarProjection` internal calls. The aggregation should be symmetric with respect to signal polarity, ensuring that opposite signals produce corresponding outputs.

Usage

```
signalAggregation(method = c("mean", "wmean", "log.wmean", "exp.wmean"))
```

Arguments

`method`	A character string specifying the method for signal aggregation, returning either a customized `mean` or `weighted.mean` function.

Details

At each point in pathway space, multiple vertices may each contribute a decayed signal value; `method` controls how these contributions are combined into a single value:

- **mean**: a plain, unweighted average. Because the same number of potential contributors is assumed throughout the image, points reached by few vertices can show a diluted value compared to points reached by many, even when the underlying signals are equally strong.
- **wmean**: each contribution is weighted by its own magnitude, so the strongest nearby signal dominates the result regardless of how many (or how weak) the other contributions are.
- **log.wmean**: like `wmean`, but the weighting is compressed, giving moderate signals comparatively more influence relative to the single strongest one.
- **exp.wmean**: like `wmean`, but the weighting is sharpened, so the strongest nearby signal dominates the result even more than under `wmean`.

Unlike `mean`, the weighted variants are not affected by the dilution described above, since contributions with zero weight do not affect their result. `mean` is a reasonable default for simple or binary signals, such as those used in introductory examples; for continuous, more nuanced analyses, `wmean` (or one of its variants) is generally preferable. For other aggregation rules, see *fuzzy logic* functions in the online tutorials: https://sysbiolab.github.io/PathwaySpace/

Value

Returns a function of the form: `function(x) { ... }`

Author(s)

Sysbiolab Team

See Also

circularProjection, polarProjection, weighted.mean

Examples

```
aggregate.fun <- signalAggregation()
```

---

silhouetteMapping,PathwaySpace-method

Decorating PathwaySpace Images with Graph Silhouettes

Description

silhouetteMapping constructs an image baseline used to outline the graph layout in a PathwaySpace image.

Usage

```
## S4 method for signature 'PathwaySpace'
silhouetteMapping(
  ps,
  pdist = 0.05,
  baseline = 0.01,
  fill.cavity = TRUE,
  verbose = TRUE
)
```

Arguments

ps	A PathwaySpace class object.
pdist	A term (in (0,1]) determining a distance unit for the silhouette projection.
baseline	A fraction (in [0,1]) of the silhouette projection, representing the level over which a silhouette will outline the graph layout. When baseline = 0 (i.e. lower level of the projection), the silhouette will extend over the entire image space, so no outline will be visible.
fill.cavity	A logical value specifying to fill cavities in the silhouette mask (when fill.cavity=TRUE) or not (when fill.cavity=FALSE).
verbose	A logical value specifying to display detailed messages (when verbose=TRUE) or not (when verbose=FALSE).

Value

A preprocessed PathwaySpace class object.

Author(s)

Sysbiolab Team

See Also

circularProjection

Examples

```
library(PathwaySpace)

# Load a demo igraph
data('gtoy1', package = 'RGraphSpace')

# Create a new PathwaySpace object
ps <- buildPathwaySpace(gtoy1, nrc = 100)
# note: adjust 'nrc' to increase image resolution

# Set '1s' as vertex signal
vertexSignal(ps) <- 1

# Map graph silhouette
ps <- silhouetteMapping(ps, pdist = 0.1)
```

SpaceProjection-class *Projection Data Container for PathwaySpace Objects*

Description

SpaceProjection is an S4 class that stores the intermediate and final matrices produced during signal projection in a PathwaySpace object. It is created internally by circularProjection or polarProjection and is not intended to be constructed directly by the user.

Value

A SpaceProjection object.

Slots

coordinates A numeric matrix with one row per graph node and four columns (X, Y, Xint, Yint), storing continuous and integer grid coordinates for each node.

floor A numeric matrix of dimensions nrc x nrc representing the projection floor, used to mask regions outside the graph silhouette.

signal A numeric matrix of dimensions nrc x nrc storing the smoothed signal values before final scaling.

result A numeric matrix of dimensions nrc x nrc containing the final projected signal, scaled to [0, 1] (or [-1, 1] for "negpos" scale types).

See Also

PathwaySpace

---

summitMapping,PathwaySpace-method

Mapping Summits on PathwaySpace Images

Description

The summitMapping method implements a segmentation strategy to identify summits on a 2D-landscape image (see summitWatershed).

Usage

```
## S4 method for signature 'PathwaySpace'
summitMapping(
  ps,
  maxset = 30,
  minsize = 30,
  threshold = 0.5,
  segm.fun = summitWatershed,
  ...
)
```

Arguments

ps	A PathwaySpace class object.
maxset	A single positive integer indicating the maximum number of summits to be returned by the segmentation function.
minsize	A single positive integer indicating the minimum size of the summits.
threshold	A threshold provided as a fraction (in [0,1]) of the max signal intensity.
segm.fun	A segmentation function used to detect summits (see summitWatershed).
...	Additional arguments passed to the segmentation function.

Value

A preprocessed PathwaySpace class object.

Author(s)

Sysbiolab Team

See Also

circularProjection

Examples

```
library(PathwaySpace)

# Load a large igraph
data("PCv12_pruned_igraph", package = "PathwaySpace")

# Continue this example from the PathwaySpace vignette,
# in the 'PathwaySpace decoration' section
```

---

summitWatershed *Variation of the watershed algorithm for summit detection*

Description

The summitWatershed function implements a segmentation strategy to identify summits within a landscape image generated by the PathwaySpace package. This function is entirely coded in R, which helps alleviating users from the task of loading an excessive number of dependencies. Nonetheless, while this novel implementation prevents the burden a 'dependency heaviness', it still requires optimization as it currently exhibits slower performance compared to well-established implementations such as the watershed function from the EBImage package. The summitWatershed maintain a certain level of compatibility with the EBImage's watershed function, and both can be used in the PathwaySpace package.

Usage

```
summitWatershed(x, tolerance = 0.1, ext = 1)
```

Arguments

x	A 2D-numeric array in which each point represents the coordinates of a signal in a landscape image.
tolerance	The minimum signal intensity of a summit (in [0,1]), representing a fraction of the maximum signal intensity.
ext	Radius (in pixels) for detecting neighboring objects.

Value

A matrix with labeled summits.

Author(s)

Sysbiolab Team, Mauro Castro.

See Also

summitMapping

Examples

```
# Load a demo landscape image
data('gimage', package = 'PathwaySpace')

# Scale down the image for a quicker demonstration
gimage <- gimage[200:300, 200:300]

# Check signal range
range(gimage, na.rm = TRUE)
# [1] 0 1

# Check image

image(gimage)


# Threshold the signal intensity, for example:
gimage[gimage < 0.5] <- 0

# Run summit segmentation
gmask <- summitWatershed(x = gimage)

# Check resulting image mask

image(gimage)
```

---

updatePathwaySpace,PathwaySpace-method
Update a PathwaySpace object

Description

Updates outdated PathwaySpace objects serialized from previous package versions, adding any missing slots with default values.

Usage

```
## S4 method for signature 'PathwaySpace'
updatePathwaySpace(x, verbose = TRUE)
```

Arguments

x	A PathwaySpace object.
verbose	Logical; if TRUE, reports which slots were added.

Value

An updated PathwaySpace object.

vertexSignal-accessors

Accessor Functions for PathwaySpace Objects

Description

Get or set vertex signals, decay functions, and the active feature in a PathwaySpace object.

`vertexSignal()` gets or sets the numeric signal assigned to each vertex, used as input for spatial projection.

`vertexDecay()` gets or sets the decay function assigned to each vertex, controlling how the signal attenuates with distance.

`activeFeature()` gets or sets the active feature name, which automatically extracts the corresponding signal from the `fdata` slot or node attributes and assigns it to `vertexSignal()`.

Usage

```
## S4 method for signature 'PathwaySpace'
vertexSignal(x)

## S4 replacement method for signature 'PathwaySpace'
vertexSignal(x) <- value

## S4 method for signature 'PathwaySpace'
vertexDecay(x)

## S4 replacement method for signature 'PathwaySpace'
vertexDecay(x) <- value

## S4 method for signature 'PathwaySpace'
activeFeature(x)

## S4 replacement method for signature 'PathwaySpace'
activeFeature(x) <- value
```

Arguments

x	A PathwaySpace class object.
value	The new value to assign: • For `vertexSignal()`: a numeric vector or scalar. • For `vertexDecay()`: a decay function or list of decay functions (see `linearDecay`, `weibullDecay`). • For `activeFeature()`: a single string matching a feature name (see `gs_features`) or a node attribute (see `gs_names`).

Value

The updated PathwaySpace object.

Examples

```
library(PathwaySpace)

# Load a demo igraph
data('gtoy1', package = 'RGraphSpace')
ps <- buildPathwaySpace(gtoy1, nrc = 100)

# Check vertex names
names(ps)

##--------------------------------------
## 'vertexSignal' accessor

# Access signal values from all vertices
vertexSignal(ps)

# Modify signal value of a specific vertex
vertexSignal(ps)[1] <- 1

# Modify signal value of specific vertices
vertexSignal(ps)[c("n2","n3")] <- 1

# Set '1s' to all vertices
vertexSignal(ps) <- 1

##--------------------------------------
## 'activeFeature' accessor

# Assign a signal feature matrix
signal_mtx <- matrix(
  rep(rnorm(gs_vcount(ps)), 2),
  ncol = 2,
  dimnames = list(names(ps), c("feature1", "feature2"))
)
gs_fdata(ps) <- signal_mtx

# Set the active feature — automatically updates vertexSignal()
activeFeature(ps) <- "feature1"

##--------------------------------------
## 'vertexDecay' accessor

# Access decay function of a specific vertex
vertexDecay(ps)[["n3"]]

# Modify decay function of a specific vertex
vertexDecay(ps)[["n3"]] <- linearDecay()
```

```
# Modify decay functions of two vertices
vertexDecay(ps)[c("n1","n3")] <- list( weibullDecay() )

# Modify decay functions of all vertices
vertexDecay(ps) <- weibullDecay(shape = 2)
```

`weibullDecay` *Constructor of Weibull decay functions*

Description

The `weibullDecay()` constructor either creates a decay function or returns a `ggplot` object for visualizing the decay model. It is a utility function used internally by `circularProjection` and `polarProjection`.

Usage

```
weibullDecay(
  decay = 0.001,
  pdist = 0.15,
  shape = 1.05,
  plot = FALSE,
  demo.signal = 1
)
```

Arguments

`decay`	A decay factor (in `[0,1]`). This term indicates how much a `signal` decreases as a function of distance in pathway space. For example, at a specific distance defined by the `pdist` parameter, the signal intensity will be the initial signal multiplied by `decay`.
`pdist`	A distance normalization term (in (0, 1]) at which the signal reaches `signal * decay`. This parameter is used to anchor the decay to a meaningful distance (see `details`). Also, when `pdist = 1`, it will represent the diameter of the inscribed circle within the coordinate space of a `PathwaySpace` object.
`shape`	A parameter (>=1) of a Weibull function. When `shape=1` the Weibull decay follows an exponential decay. When `shape>1` the function is first convex, then concave with an inflection point.
`plot`	A logical value indicating whether to return a `ggplot` object.
`demo.signal`	A numeric value in `[-Inf, Inf]`, only passed when `plot = TRUE` to visualize the decay curve with a specific signal intensity. The value is ignored by the function constructor, as the decay function itself is returned without using an initial signal.

Details

The `weibullDecay()` constructor creates a decay model based on the Weibull distribution. It describes how a signal decreases as a function of distance, controlled by both a decay rate and a shape parameter.

The decay function is defined as:

$$y = signal \times decay^{\left(\frac{x}{pdist}\right)^{shape}}$$

where $signal$ represents the initial intensity, $decay$ controls the rate of attenuation, x is a vector of normalized distances, and $shape$ adjusts the curvature of the decay. When $shape = 1$, the function follows an exponential decay. For $shape > 1$, the curve transitions from convex to concave, exhibiting an inflection point. The $pdist$ parameter anchors the model such that:

- $y = signal$ when $x = 0$
- $y = signal \times decay$ when $x = pdist$

Value

Returns either a function of the form `function(x, signal) { ... }` or, if `plot = TRUE`, a `ggplot` object illustrating the decay model.

Author(s)

Sysbiolab Team

See Also

`linearDecay`, `expDecay`

Examples

```
# Return a decay function
decay_fun <- weibullDecay(decay = 0.5, pdist = 0.4, shape = 2)

# Plot decay model parameters
# weibullDecay(decay = 0.5, pdist = 0.4, shape = 2, plot = TRUE)
```

Index