Package 'RPointCloud'

July 21, 2025

Version 0.8.0

Date 2024-08-19

Title Visualizing Topological Loops and Voids

Description Visualizations to explain the results of a topological data analysis. The goal of topological data analysis is to identify persistent topological structures, such as loops (topological circles) and voids (topological spheres), in data sets. The output of an analysis using the 'TDA' package is a Rips diagram (named after the mathematician Eliyahu Rips). The goal of 'RPointCloud' is to fill in these holes in the data by providing tools to visualize the features that help explain the structures found in the Rips diagram. See McGee and colleagues (2024) <doi:10.1101/2024.05.16.593927>.

Depends R (>= 3.5)

Imports methods, graphics, stats, TDA, ClassDiscovery, ClassComparison (>= 3.3), Mercator, rgl, splines, circlize

Suggests knitr, rmarkdown, Polychrome, igraph, ape, PCDimension

License Artistic-2.0

URL http://oompa.r-forge.r-project.org/

VignetteBuilder knitr

NeedsCompilation no

Author Kevin R. Coombes [aut, cre],
Jake Reed [aut],
RB McGee [aut]

Maintainer Kevin R. Coombes <krc@silicovore.com>

Repository CRAN

Date/Publication 2024-08-19 21:20:02 UTC

Contents

`CLL-data` *Chronic Lymphocytic Leukemia Clinical Data*

Description

Contains 29 columns of deidentified clinical data on 266 patients with chronic lymphocytic leukemia (CLL).

Usage

```
data(CLL)
```

Format

Note that there are three distinct objects included in the data set: `clinical`, `daisydist`, and `ripDiag`.

`clinical` A numerical data matrix with 266 rows (patients) amd 29 columns (clinical features). Patients were newly diagnosed with CLL and previously untreated at the time the clinical measurements were recorded.

`daisydist` A distance matrix, stored as a `dist` object, recording pairwise distances between the 266 CLL patients. Distances were computed using the `daisy` function of Kaufmann and Rooseeuw, as implemented in the `cluster` R package.

`ripDiag` This object is a "Rips diagram". It was produced by running the the `ripsDiag` function from the `TDA` R package on the `daisydist` distance matrix.

Author(s)

Kevin R. Coombes <krc@silicovore.com>, Caitlin E. Coombes <ccoombes@stanford.edu>, Jake Reed <hreed@augusta.edu>

Source

Data were collected in the clinic of Dr. Michael Keating at the University of Texas M.D. Anderson Cancer Center to accompany patient samples sent to the laboratory of Dr. Lynne Abruzzo. Various subsets of the data have been reported previously; see the references for some publications. Computation of the daisy distance was performed by Caitlin E. Coombes, whose Master's Thesis also showed that this was the most appropriate method when dealing with heterogeneous clinical data of mixed type. Computation of the topological data analysis (TDA) using the Rips diagram was performed by Jake Reed.

References

Schweighofer CD, Coombes KR, Barron LL, Diao L, Newman RJ, Ferrajoli A, O'Brien S, Wierda WG, Luthra R, Medeiros LJ, Keating MJ, Abruzzo LV. A two-gene signature, SKI and SLAMF1, predicts time-to-treatment in previously untreated patients with chronic lymphocytic leukemia. PLoS One. 2011;6(12):e28277. doi: 10.1371/journal.pone.0028277.

Duzkale H, Schweighofer CD, Coombes KR, Barron LL, Ferrajoli A, O'Brien S, Wierda WG, Pfeifer J, Majewski T, Czerniak BA, Jorgensen JL, Medeiros LJ, Freireich EJ, Keating MJ, Abruzzo LV. LDOC1 mRNA is differentially expressed in chronic lymphocytic leukemia and predicts overall survival in untreated patients. Blood. 2011 Apr 14;117(15):4076-84. doi: 10.1182/blood-2010-09-304881.

Schweighofer CD, Coombes KR, Majewski T, Barron LL, Lerner S, Sargent RL, O'Brien S, Ferrajoli A, Wierda WG, Czerniak BA, Medeiros LJ, Keating MJ, Abruzzo LV. Genomic variation by whole-genome SNP mapping arrays predicts time-to-event outcome in patients with chronic lymphocytic leukemia: a comparison of CLL and HapMap genotypes. J Mol Diagn. 2013 Mar;15(2):196-209. doi: 10.1016/j.jmoldx.2012.09.006.

Herling CD, Coombes KR, Benner A, Bloehdorn J, Barron LL, Abrams ZB, Majewski T, Bondaruk JE, Bahlo J, Fischer K, Hallek M, Stilgenbauer S, Czerniak BA, Oakes CC, Ferrajoli A, Keating MJ, Abruzzo LV. Time-to-progression after front-line fludarabine, cyclophosphamide, and rituximab chemoimmunotherapy for chronic lymphocytic leukaemia: a retrospective, multicohort study. Lancet Oncol. 2019 Nov;20(11):1576-1586. doi: 10.1016/S1470-2045(19)30503-0.

Coombes CE, Abrams ZB, Li S, Abruzzo LV, Coombes KR. Unsupervised machine learning and prognostic factors of survival in chronic lymphocytic leukemia. J Am Med Inform Assoc. 2020 Jul 1;27(7):1019-1027. doi: 10.1093/jamia/ocaa060.

`Cycle-class` `Cycle` *Objects For Visualizing Topological Data Analysis Results*

Description

The `Cycle` object represents a cycle found by performing topological data analysis (using the `TDA` package) on a data matrix or distance matrix.

Usage

```
Cycle(rips, dimen, J, color)
getCycle(rips, dimension = 1, target = NULL)
cycleSupport(cycle, view)
## S4 method for signature 'Cycle,matrix'
plot(x, y, lwd = 2, ...)
## S4 method for signature 'Cycle'
lines(x, view, ...)
```

Arguments

rips	A Rips diagram from the TDA package.
dimen	An integer; the dimension of the cycle.
J	An integer; the index locating the cycle in the cycleLocation portion of the Rips diagram. If NULL, then looks for the most persistent cycle.
color	A character vector of length one; the color in which to display the cycle.
x	A Cycle object.
y	A (layout) matrix with coordinates showing where to plot each point supporting the cycle.
view	A (layout) matrix with coordinates showing where to plot each point supporting the cycle.
lwd	A number; the graphical line width parameter lwd.
...	The usual set of additional graphical parameters.
dimension	An integer; the dimension of the cycle.
target	An integer indexing the desired cycle. Note that the index should be relative to other cuycles of the same dimension. If NULL, gets the longest persisting cycle.
cycle	A raw cycle, meaning a simple element of the cycleLocation part of a Rips diagram.

Value

The Cycle function constructs and returns an object of the Cycle class

The plot and lines methods return (invisibly) the Cycle object that was their first argument.

The getCycle function extracts a single raw cycle from a Rips diagram and returns it. The cycleSupport function combines such a cycle with a layout/view matrix to extract a list of the coordinates of the points contained in the cycle.

Slots

index: A matrix, containing the indices into the data matrix or distance matrix defining the simplices that realize the cycle.

dimension: A matrix; the dimension of the cycle.

color: A character vector; the color in which to display the cycle.

Methods

plot(x, y, lwd = 2, ...): Produce a plot of a `Cycle` object.

lines(x, view, ...) Add a depiction of a cycle to an existing plot.

Author(s)

Kevin R. Coombes <krc@silicovore.com>

See Also

`TDA`, `ripsDiag`

Examples

```
data(CLL)
cyc1 <- Cycle(ripdiag, 1, 236, "forestgreen")
V <- cmdscale(daisydist)
plot(cyc1, V)
plot(V, pch = 16, col = "gray")
lines(cyc1, V)
```

---

cytof-data *CyTOF Data*

Description

This data set contains 18 columns of protein expression values measured in 214 early monocyte cells using mass cytometry (CyTOF).

Usage

```
data(cytof)
```

Format

Note that there are three distinct objects included in the data set: `AML10.node287`, `AML10,node287.rips`, and `Arip`.

`AML10.node287` A numerical data matrix with 214 rows (single cells) amd 18 columns (antibody-protein markers). Values were obtained using mass cytometry applied to a sample collected from a patient with acute myeloid leukemia (AML).

`AML10.node287.rips` This object is a "Rips diagram". It was produced by running the `ripsDiag` function from the `TDA` R package on the `AML10.node287` data matrix.

`Arip` This object is a "Rips diagram". It was produced by running the `ripsDiag` function from the `TDA` R package on the Euclidean distance matrix between the single cells.

Author(s)

Kevin R. Coombes <krc@silicovore.com>, RB McGee <mcgee@sholycross.edu>, Jake Reed <hreed@augusta.edu>

Source

Data were collected by Dr. Greg Behbehani while working in the laboratory of Dr. Garry Nolan at Stanford University. This data set is a tiny subset of the full data set, which was described previously in the paper by Behbehani et al.

References

Behbehani GK, Samusik N, Bjornson ZB, Fantl WJ, Medeiros BC, Nolan GP. Mass Cytometric Functional Profiling of Acute Myeloid Leukemia Defines Cell-Cycle and Immunophenotypic Properties That Correlate with Known Responses to Therapy. Cancer Discov. 2015 Sep;5(9):988-1003. doi: 10.1158/2159-8290.CD-15-0298.

---

disentangle *Disentangling Rips Diagrams From Their Initial Data Coordinates*

Description

The ripsDiag function in the TDA package produces very different results depending on whether you invoke it on a data matrix (expressed in terms of specific data coordinates) or a distance matrix (expressed as abstract indices). This function converts the specific coordinates into indices, allowing one to more easily plot different views of the data structures.

Usage

```
disentangle(rips, dataset)
```

Arguments

rips	A Rips diagram produced from a data matrix using the TDA package.
dataset	The original dataset used to create the rips object.

Details

The core algorithm is, quite simply, to recombine the coordinates of a point in the Rips diagram in a manner consistent with their storage in the original data set, and find the index (i.e., row number) of that point.

Value

Returns a Rips diagram nearly identical to the one that would be produced if the ripsDiag function had been invoked instead on the Euclidean distance matrix.

Author(s)

Kevin R. Coombes <krc@silicovore.com>

Examples

```
data(cytof)
fixed <- disentangle(Arip, AML10.node287)
```

EBexpo-class *The* EBexpo *Class*

Description

The EBexpo object represents the results of an Empirical Bayes approach to estimate a distribution as a mixture of a (more or less) known exponential distribution along with a completely unknown "interesting" distribution. The basic method was described by Efron and Tibshirani with an application to differential expression in microarray data.

Usage

```
EBexpo(edata, resn = 200)
cutoff(target, prior, object)
## S4 method for signature 'EBexpo,missing'
plot(x, prior = 1, significance = c(0.5, 0.8, 0.9),
                 ylim = c(-0.5, 1), xlab = "Duration",
                 ylab = "Probability(Interesting | Duration)", ...)
## S4 method for signature 'EBexpo'
hist(x, ...)
```

Arguments

edata	A numeric vector; the observed data that we think comes mainly from an exponential distribution.
resn	A numeric vector; the resolution used to estimate a histogream.
x	An EBexpo object.
prior	A numeric vector of length 1; the prior probability of an observed data point coming from the known exponential distribution.
significance	A numeric vector with values between 0 and 1; the target posterior probabiltiites.
ylim	A numeric vector of length two.
xlab	A character vector; the label for the x-axis.
ylab	A character vector; the label for the y-axis.
...	The usual set of graphical parameters.
target	The target posterior probability.
object	An EBexpo object.

Value

The EBexpo function constructs and returns an object of the EBexpo class

The plot and hist methods return (invisibly) the EBexpo object that was their first argument.

Slots

xvals: Inherited from MultiWilcoxonTest

statistics: Inherited from MultiWilcoxonTest, Here, these are the same a the edata slot from an link{ExpoFit} object.

pdf: Inherited from MultiWilcoxonTest

theoretical.pdf: Inherited from MultiWilcoxonTest

unravel: Inherited from MultiWilcoxonTest

groups: Inherited from MultiWilcoxonTest, but not used

call: Inherited from MultiWilcoxonTest

h0: See ExpoFit

lambda: See ExpoFit

mu: See ExpoFit

Methods

plot(x, prior, post = c(0.5, 0.8, 0.9), ...): Produce a plot of a EBexpo object.

hist(x, ...): Produce a histogram of the observed distibution, with overlays.

Author(s)

Kevin R. Coombes <krc@silicovore.com>

References

Efron B, Tibshirani R. Empirical bayes methods and false discovery rates for microarrays. Genet Epidemiol. 2002 Jun;23(1):70-86. doi: 10.1002/gepi.1124.

Examples

```
data(cytof)
diag <- AML10.node287.rips[["diagram"]]
persistence <- diag[, "Death"] - diag[, "Birth"]
d1 <- persistence[diag[, "dimension"] == 1]
eb <- EBexpo(d1, 200)
hist(eb)
plot(eb, prior = 0.56)
cutoff(0.8, 0.56, eb)
```

ExpoFit-class *The* ExpoFit *Class*

Description

An ExpoFit object represents a robust fit to an exponential distribution, in a form that can conveniently be used as part of an Empirical Bayes approach to decompose the distributions of cycle presistence or duration for a topological data analysis performed using the TDA package.

Usage

```
ExpoFit(edata, resn = 200)
## S4 method for signature 'ExpoFit,missing'
plot(x, y, ...)
```

Arguments

edata	A numeric vector; the observed data that we think comes mainly from an exponential distribution.
resn	A numeric vector of length 1; the resolution (number of breaks) used to estimate a histogram.
x	An ExpoFit object.
y	Ignored.
...	The usual set of graphical parameters.

Value

The ExpoFit function constructs and returns an object of the ExpoFit class.

The plot method returns (invisibly) the ExpoFit object that was its first argument.

Slots

edata: A numeric vector; the observed data that we think comes from an exponential distribution.

h0: A histogram object produced by the hist function applied to the supplied edata.

X0: A numeric vector containing the midpoints of the breaks in the histogram object.

pdf: The empirical density function extracted from the histogram object.

mu: The observed mean of the putative exponential distribution.

lambda: The robustly estimated parameter of the exponential distribution. Originally crudely repesented by the reciprocal of the mean.

Methods

plot(x, y, lwd = 2, ...): Produce a plot of a ExpoFit object.

Author(s)

Kevin R. Coombes <krc@silicovore.com>

See Also

EBexpo

Examples

```
data(cytof)
diag <- AML10.node287.rips[["diagram"]]
persistence <- diag[, "Death"] - diag[, "Birth"]
d1 <- persistence[diag[, "dimension"] == 1]
ef <- ExpoFit(d1) # should be close to log(2)/median?
plot(ef)
```

Feature-class Feature *Objects For Visualizing Topological Data Anal;ysis Results*

Description

The Feature object represents a feature from a data matrix used when performing topological data analysis with the TDA package. Features might be genes, proteins, clinical or demographic covariates, or any other item measured on a set of patient samples or cells.

Usage

```
Feature(values, name, colors, meaning, ...)
## S4 method for signature 'Feature,matrix'
plot(x, y, pch = 16, ...)
## S4 method for signature 'Feature'
points(x, view, ...)
```

Arguments

values	A numeric vector.
name	A character vector of length one.
colors	A character vector of length at least two, containing the names or hexadecimal representations of colors used to create a color ramp to display the feature.
meaning	A character vector of length two containing the interpretations of the low and high extreme values of the feature. Note that this works perfectly well for binary factors represented as numeric 0-1 vectors.
x	A Feature object.
y	A (layout) matrix with coordinates showing where to plot each point in the feature.

view	A (layout) matrix with coordinates showing where to plot each point in the feature.
pch	A number; the graphical plotting character parameter pch.
...	The usual set of additional graphical parameters.

Value

The Feature function constructs and returns an object of the Feature class

The plot and points methods return (invisibly) the Feature object that was their first argument.

Slots

name: A character vector of length one; the name of the feature..

values: A numeric vector of the values of this feature.

meaning: A character vector of length two containing the interpretations of the low and high extreme values of the feature. Note that this works perfectly well for binary factors represented as numeric 0-1 vectors.

colRamp: A function created using the colorRamp2 function from the circlize package.

Methods

plot(x, y, lwd = 2, ...): Produce a plot of a Feature object.

points(x, view, ...) Add a depiction of a feature to an existing plot.

Author(s)

Kevin R. Coombes <krc@silicovore.com>

Examples

```
data(CLL)
featSex <- Feature(clinical[,"Sex"], "Sex",
           c("pink", "skyblue"), c("Female", "Male"))
V <- cmdscale(daisydist)
plot(featSex, V)
plot(V)
points(featSex, V)
```

LDPC *Local Dimension of Point Clouds*

Description

Given a data set viewed as a point cloud in N-dimesnional space, compute the local estimate of the dimension of an underlying manifold, as defined by Ellis and McDermott, analogous to earlier related work by Takens.

Usage

```
takens(r, dists)
LDPC(CellID, dset, rg, quorum, samplesAreRows = TRUE)
```

Arguments

`CellID`	An integer indexing one of the cells-samples in the data set.
`dset`	A data set. The usual orientation is that rows are cells and columns are features.
`rg`	A numerical vector of radial distances at which to compute Takens estimates of the local dimension.
`quorum`	The minimum number of neighboring cells required for the computation to be meaningful.
`samplesAreRows`	A logical value: do rows or columns represent samples at which to compute local dimensions.
`r`	A radial distances at which to compute the Takens estimate.
`dists`	A sorted vector of distances from one cell to all other cells.

Details

"[T]he procedure is carried out as follows. A 'bin increment', A; a number, m, of bin increments; and a 'quorum', $q > 0$, are chosen and raw dimensions are calculated for $r=A$, $2A$, $\ldots$, m. Next, for each observation, x_i, let r_i be the smallest multiple of A not exceeding mA such that the ball with radius r_i centered at x_i contains at least q observations, providing that there are at least q observations within mA of x_i. Otherwise let $r_i = mA$."

The `takens` function computes the Takens estimate of the local dimension of a point cloud at radius r around a data point. For each cell-sample, we must compute and sort the distances from that cell to all other cells. (For the `takens` function, these distances are passed in as the second argument to the function.) Preliminary histograms of distance distributions may be used to inform a good set of radial distances. Note that the local dimension estimates are infinite if the radius is so small that there are no neighbors. The estmiates decrease as the radius increases os as the number of local neighbors increases. The reference paper by Ellis and McDermott says:

The `LDPC` function iterates over all cells-samples int he data sets, computes and sorts their distance to all other cells, and invokes teh `takens` function to compute local estimates os dimension.

Value

The takens function returns a list with two items: the number of neighbors k and the dimension estimate d at each value of the radius from the input vector.

The LDPC function returns a list containing vectors R, k, and d values for each cell in the data set.

Author(s)

Kevin R. Coombes <krc@silicovore.com>

References

Ellis and McDermott

Examples

```
data(cytof)
localdim <- LDPC(1, AML10.node287, seq(1, 6, length=20), 30, TRUE)
```

LoopCircos-class *The* LoopCircos *Class*

Description

A LoopCircos object represents a one-dimensional cycle (a loop, or topogical eequivalent of a circle), along with a set of features that vary around the loop and can be plotted in a "circos" plot to help explain why the loop exists. The LoopCircos class inherits from the Cycle class, so you can think of it as a cycle object with extra information (namely, the explanatory features).

Usage

```
LoopCircos(cycle, angles, colors)
angleMeans(view, rips, cycle = NULL, dset, angleWidth = 20, incr = 15)
## S4 method for signature 'LoopCircos'
image(x, na.col = "grey", ...)
```

Arguments

cycle	In the LoopCircos function, an object of the Cycle class. In the angleMeans function, an object of the Cycle class.
angles	A matrix where columns are features, rows are angles, and the value is the mean expression of that feature in a sector around that central angle. These are almost always constructed using the angleMeans function.
colors	A list of character vectors, each of length two, containing the names or hexadecimal representations of colors used to create a color ramp to display the feature. The list should be the same length as the number of features to be displayed.

view	A (layout) matrix with coordinates showing where to plot each point in the feature while computing means in sectors.
rips	A Rips diagram object.
dset	The numeric matrix from which Featuers are drawn.
angleWidth	A numeric vector of length one; the width (in degrees) of each sector.
incr	A numeric vector; the increment (in degeres) between sector centers.
x	A LoopCircos object.
na.col	A character string; the color in which to plot missing (NA) data.
...	The usual set of additional graphical parameters.

Value

The angleMeans function computes the mean expression of each numeric feature in a two-dimensional sector swept out around the centroid of the support of a cycle. It returns these values as a matrix, with each row corresponding to the angle (in degrees) around the centroid of the cycle.

The LoopCircos function constructs and returns an object of the LoopCircos class

The image method returns (invisibly) the LoopCircos object that was its first argument.

Slots

angles: A matrix of angle means.

colors: A list of color-pairs for displayign features.

Methods

image(x, ...): Produce a circos plot of a LoopCircos object.

Author(s)

Kevin R. Coombes <krc@silicovore.com>

See Also

Cycle, LoopFeature, ripsDiag

Examples

```
data(CLL)
cyc1 <- Cycle(ripdiag, 1, 236, "forestgreen")
V <- cmdscale(daisydist)
poof <- angleMeans(V, ripdiag, cyc1, clinical)
poof[is.na(poof)] <- 0
angle.df <- poof[, c("mutation.status", "CatCD38", "CatB2M", "CatRAI",
                     "Massive.Splenomegaly", "Hypogammaglobulinemia")]
colorScheme <- list(c(M = "green", U = "magenta"),
                    c(Hi = "cyan", Lo ="red"),
                    c(Hi = "blue", Lo = "yellow"),
                    c(Hi = "#dddddd", Lo = "#111111"),
```

```
                    c(No = "#dddddd", Yes = "brown"),
                    c(No = "#dddddd", Yes = "purple"))
annote <- LoopCircos(cyc1, angle.df, colorScheme)
image(annote)
```

LoopFeature-class LoopFeature *Objects For Visualizing Features That Define Loops*

Description

The LoopFeature class is a tool for understanding and visualizing loops (topological circles) and the features that can be used to define and interpret them. Having found a (statistically significant) loop, we investigate a feature by computing its mean expression in sectors of a fixed width (usually 20 degrees) at a grid of angles around the circle (usually multiples of 15 degrees from 0 to 360). We model these data using the function

$$f(\theta) = A + B\sin(\theta) + C\cos(\theta).$$

We then compute the "fraction of unexplained variance" by dividing the residual sum of squares from this model by the total variance of the feature. Smaller values of this statistic are more likely to identify features that vary sytematically around the circle with a single peak and a single trough.

Usage

```
LoopFeature(circMeans)
## S4 method for signature 'LoopFeature,missing'
plot(x, y, ...)
## S4 method for signature 'LoopFeature,character'
plot(x, y, ...)
## S4 method for signature 'LoopFeature'
image(x, ...)
```

Arguments

circMeans	A matrix, assumed to be the output from a call to the angleMeans function. Columns are features and rows are angles.
x	A LoopFeature object.
y	A character vector; the set of features to plot.
...	The usual set of additional graphical parameters.

Value

The LoopFeature function constructs and returns an object of the LoopFeature class

The plot and image methods return (invisibly) the LoopFeature object that was their first argument.

Slots

`data`: The input `circMeans` data matrix.

`fitted`: A matrix that is the same size as `data`; the results of fitting a model for each feature as a linear combination of sine and cosine.

`RSS`: A numeric vector; the residual sum of squares for each model.

`V`: A numeric vector; the total variance for each feature.

`Kstat`: A numeric vector, the unexplained variance statistic, RSS/V.

Methods

plot(x, y, ...): For the selected features listed in y (which can be missing or "all" to plot all features), plots the fitted model as a curve along with the observed data.

image(x, ...) Produce a 2D image of all the features, with each one scaled to the range [0,1] and with the rows ordered by where around the loop the maximum value occurs.

Author(s)

Kevin R. Coombes <krc@silicovore.com>

Examples

```
data(CLL)
view <- cmdscale(daisydist)
circular <- angleMeans(view, ripdiag, NULL, clinical)
lf <- LoopFeature(circular)
sort(lf@Kstat)
plot(lf, "Serum.beta.2.microglobulin")
opar <- par(mai = c(0.82, 0.2, 0.82, 1.82))
image(lf, main = "Clinical Factors in CLL")
par(opar)
```

`Projection-class` `Projection` *Objects For Visualizing 3D Topological Data Analysis Results in 2D*

Description

The `Projection` class is a tool for understanding and visualizing voids (topological spheres) and the features that can be used to define and interpret them. The core idea is to first project all data points in dimension (at least) 3 radially onto the surface of the sphere. We then use spherical coordinates on the sphere (the latitude, `psi` from 0 to 180 degrees and the longitude, `phi`, from -180 to 180 degrees) to locate the points. The `plot` method shows these points in two dimensions, colored based on the expression of the chosen `Feature`. The `image` method uses a loess fit to smooth the observed data points across the entire surface of the sphere and displays the resulting image. Using

latitude and longitude in this way is similar to a Mercator projection of the surface of the Earth onto a two-dimensional plot.

It is worth noting that, for purposes of computing the smoothed version, we actually take two "trips" around the "equator" to fit the model, but only display the middle section of that fit. The point of this approach is to avoid edge effects in smoothing the data.

Usage

```
Projection(cycle, view, feature, span = 0.3, resn = 25)
## S4 method for signature 'Projection,missing'
plot(x, y, pch = 16, ...)
## S4 method for signature 'Projection'
image(x, ...)
```

Arguments

cycle	A Cycle object representing a void, or 2-cycle.
view	A (layout) matrix with coordinates showing where to plot each point in the feature. Must be at least three-dimensional.
feature	A Feature object, representing the expression levels of one of the features in the underlying data set.
span	Parameter used for a loess fit.
resn	The number of steps used to create the vectors psi and half-phi.
x	A Projection object.
y	ignored.
pch	A number; the graphical plotting character parameter pch.
...	The usual set of additional graphical parameters.

Value

The Projection function constructs and returns an object of the Projection class

The plot and image methods return (invisibly) the Projection object that was their first argument.

Slots

phi: A numeric vector

psi: A numeric vector

displayphi: A numeric vector

displaypsi: A numeric vector

value: A matrix.

feature: A Feature object.

Methods

plot(x, y, lwd = 2, ...): Produce a 2-dimensional scatter plot of a Projection object.

image(x, ...) Produce a 2D heatmap image of a Projection object.

Author(s)

Kevin R. Coombes <krc@silicovore.com>

Examples

```
data(CLL)
cyc <- Cycle(ripdiag, 2, NULL, "black")
V <- cmdscale(daisydist, 3)
feat <- Feature(clinical[,"stat13"], "Deletion 13q",
               c("green", "magenta"), c("Abnormal", "Normal"))
ob <- Projection(cyc, V, feat, span = 0.2)
plot(ob)
image(ob, col = colorRampPalette(c("green", "gray", "magenta"))(64))
```

---

tailHisto *Function to extract the right tail of a histogram for display purposes.*

Description

Sometimes, the most interesting part of a histogram lies in the tail, where details are obscured because of the scale of earlier peaks in the distribution. This function uses a user-defined target cutoff to extract the right tail of a histogram, preserving its structure as a histogram object.

Usage

```
tailHisto(H, target)
```

Arguments

H	A `histogram` object.
target	A real number; the target cutoff defining the portion of the tail of the histogram to be extracted.

Details

There is nothing special going on. The only sanity check is to ensure that the `target` is small enough that there is actually a part of the histogram that can be extracted. After that, we simply cut out the approprioate pieces, make sure they are structured properly, and return them.

Value

Returns another `histogram` object that only contains the portion of the histogram beyond the target cutoff.

Author(s)

Kevin R. Coombes <krc@silicovore.com>

Examples

```
set.seed(12345)
fakeData <- rexp(2000, rate = 10)
H <- hist(fakeData, breaks = 123, plot = FALSE)
H2 <- tailHisto(H, 0.3)
opar <- par(mai=c(0.9, 0.9, 0.6, 0.2))
plot(H, freq = FALSE)
par(mai=c(3.4, 3.0, 0.6, 0.6), new=TRUE)
plot(H2, freq = FALSE, main = "", col = "skyblue")
par(opar)
```

---

treg-data *Single Cell Data on T Regulatory (Treg) Cells*

Description

This data set contains mRNA and protein-antibody data on T-regulatory immune cells. It is a subset of a much larger data set collected from the peripheral blood of patients with a variety of health comnditions.

Usage

```
data(treg)
```

Format

Note that there are three distinct objects included in the data set: `treg`, `tmat`, and `rip`.

`treg` A numerical data matrix with 538 rows and 255 columns. Each column represents a single cell from one of 61 samples that were assayed by (mixed-omics) single cell sequencing. Each row represents one of the features that was measured in the assay. Of these, 51 are antibodies that were tagged with an RNA barcode to identify them; their names all end with the string `pAbO`. The remaining 487 features are mRNA measurements, named by their official gene symbol at the time the experiment was performed. Each column represents a different single cell. This matrix is a subset of a more complete data set of T regulatory cells (Tregs). It was produced using the `downsample` function from the `Mercator` package, which was in turn inspired by a similar routine used in the SPADE algorithm by Peng Qiu. A key point of the algorithm is to make sampling less likely from the densest part of the distribution in order to preserve rare cell types in the population.

`tmat` A distance matrix, stored as a `dist` object, produced using Pearson correlation as a measure of distance between sigle-cell vectors in the `treg` data set.

`rip` This object is a "Rips diagram". It was produced by running the the `ripsDiag` function from the `TDA` R package on the `treg` subset of single cells.

Author(s)

Kevin R. Coombes <krc@silicovore.com>, Jake Reed <hreed@augusta.edu>

Source

Data were kindly provided by Dr. Klaus Ley, director of the Georgia Immunolgy Center at Augusta University. Analysis to perform cell typing with Seurat and isolate the subset of 769 T regulatory cells found in `dset` was performed by Jake Reed, as was the downsampling to select the random subset of 255 cells in the `treg` subset and use them for topolgical data analysis to compute the `rip` object.

References

Qiu P, Simonds EF, Bendall SC, Gibbs KD Jr, Bruggner RV, Linderman MD, Sachs K, Nolan GP, Plevritis SK. Extracting a cellular hierarchy from high-dimensional cytometry data with SPADE. Nat Biotechnol. 2011 Oct 2;29(10):886-91. doi: 10.1038/nbt.1991.

`voids` *3D Scatter Plots of Cycles and Features*

Description

Voids are two-dimensional cycles and should be thought of as the topological equivalents of (the surface) of a sphere. These are typically displayed as a point cloud in space, with wire-frame edges outlining the component (triangular) simplices.

Usage

```
voidPlot(cycle, view, feature = NULL, radius = 0.01, ...)
voidFeature(feature, view, radius = 0.01, ...)
```

Arguments

`cycle`	An object of the `Cycle` class.
`view`	A matrix (with three columns, x, y, and, z) specifying the coordinates to be used to display the data.
`feature`	An object of the `Feature` class.
`radius`	The radius of spheres used to display the points in three dimensions.
`...`	Additional graphical parameters.

Details

The 3D displays are implemented using the `rgl` package. The `voidPlot` function adds a wire-frame two-cycle to the 3d scatter plot of the underlying data. If a feature is present, then the point are colored by the expression lecvel of that feature. If not present, they are simply colored gray. However, you can always add the expression levels of a feature to an existing plot using the `voidFeature` function.

In an interactive session, everything is displayed in an OpenGL window, where the graph can be manipulated with a mouse. Programmatically, you can use the `rgl::rglwidget` function to write out the code for an interactive HTML display, and you can save the widget to a file using the `htmlwidgets::saveWidget` function.

Value

Both `voidPlot` and `voidFeature` invisibly return their first argument.

Author(s)

Kevin R. Coombes <krc@silicovore.com>

Examples

```
data(CLL)
featRai <- Feature(clinical[,"CatRAI"], "Rai Stage", c("green", "magenta"), c("High", "Low"))
vd <- Cycle(ripdiag, 2, 95, "blue")
mds <- cmdscale(daisydist, k = 3)
voidPlot(vd, mds)
voidFeature(featRai, mds, radius = 0.011)
```

Index