Package: TrajectoryGeometry (via r-universe)

November 20, 2026

Title This Package Discovers Directionality in Time and Pseudo-times Series of Gene Expression Patterns

Version 1.20.0

Description Given a time series or pseudo-times series of gene expression data, we might wish to know: Do the changes in gene expression in these data exhibit directionality? Are there turning points in this directionality. Do different subsets of the data move in different directions? This package uses spherical geometry to probe these sorts of questions. In particular, if we are looking at (say) the first n dimensions of the PCA of gene expression, directionality can be detected as the clustering of points on the (n-1)-dimensional sphere.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.3.1

Imports pracma, rgl, ggplot2, stats, methods

Depends R (>= 4.1)

Suggests dplyr, knitr, RColorBrewer, rmarkdown

VignetteBuilder knitr

biocViews BiologicalQuestion, StatisticalMethod, GeneExpression, SingleCell

Config/pak/sysreqs cmake libfreetype6-dev libglu1-mesa-dev make texlive libpng-dev libuv1-dev libgl1-mesa-dev zlib1g-dev

Repository Bioconductor 3.23

Date/Publication 2026-04-28 12:54:58 UTC

RemoteUrl https://github.com/bioc/TrajectoryGeometry

RemoteRef RELEASE_3_23

RemoteSha da60a5cdac361f131f871d3b00bbd76536a015fd

Contents

analyseBranchPoint . . . 3
analyseSingleCellTrajectory . . . 4
chol_answers . . . 6
chol_attributes . . . 6
chol_branch_point_results . . . 7
chol_pseudo_time . . . 7
chol_pseudo_time_normalised . . . 8
circleOnTheUnitSphere . . . 8
crooked_path . . . 9
crooked_path_center . . . 10
crooked_path_projection . . . 10
crooked_path_radius . . . 11
distanceBetweenTrajectories . . . 11
findSphereClusterCenter . . . 12
findSphericalDistance . . . 13
generateRandomPaths . . . 13
generateRandomUnitVector . . . 14
getDistanceDataForPaths . . . 15
getSphericalData . . . 15
getStepLengths . . . 16
hep_answers . . . 17
hep_attributes . . . 17
hep_pseudo_time . . . 18
hep_pseudo_time_normalised . . . 18
orthonormalBasis . . . 19
oscillation . . . 19
pathProgression . . . 20
pathToSphericalData . . . 21
plotPathProjectionCenterAndCircle . . . 21
projectPathToSphere . . . 23
samplePath . . . 24
single_cell_matrix . . . 24
straight_path . . . 25
straight_path_center . . . 26
straight_path_projection . . . 26
straight_path_radius . . . 27
testPathForDirectionality . . . 27
visualiseBranchPointStats . . . 28
visualiseTrajectoryStats . . . 29

Index **31**

`analyseBranchPoint` *Analyse branch point.*

Description

This function takes a single cell trajectory and analyses it starting from successively later points in pseudotime, with the rationale that a more consistent directionality will be followed after the branch point.

Usage

```
analyseBranchPoint(
  attributes,
  pseudotime,
  randomizationParams,
  statistic,
  start = (max(pseudotime) - min(pseudotime)) * 0.25,
  stop = (max(pseudotime) - min(pseudotime)) * 0.75,
  step = (max(pseudotime) - min(pseudotime)) * 0.05,
  nSamples = 1000,
  nWindows = 10,
  d = ncol(attributes),
  N = 1
)
```

Arguments

`attributes`	- An n x d (cell x attribute) matrix of numeric attributes for single cell data. Rownames should be cell names.
`pseudotime`	- A named numeric vector of pseudotime values for cells.
`randomizationParams`	- A character vector which is used to control the production of randomized paths for comparison.
`statistic`	- Allowable values are 'median', 'mean' or 'max'.
`start`	- The first pseudotime value (percentage of the trajectory) from which to analyse the trajectory from. Defaults to 25% of the way through the trajectory.
`stop`	- The last pseudotime value (as a percentage of the trajectory) from which to analyse the trajectory from. Defaults to 75% of the way through the trajectory.
`step`	- The size of the step to take between successively later starting points in pseudotime. Defaults to 5% of the trajectory length.
`nSamples`	- The number of sampled paths to generate (defaults to 1000).
`nWindows`	- The number of windows pseudotime should be split into to sample cells from (defaults to 10).
`d`	- The dimension under consideration. This defaults to ncol(attributes).
`N`	- The number of random paths to generated for statistical comparison to the given path (defaults to 1000).

Value

This returns a list of results for analyseSingleCellTrajectory, named by trajectory starting point. Each result from analyseSingleCellTrajectory is a list which contains an entry for each sampled path. Each of these entries is a list containing information comparing the sampled path in question to random paths. The entries consist of: pValue - the p-value for the path and statistic in question; sphericalData - a list containing the projections of the path to the sphere, the center of that sphere and the statistic for distance to that center; randomDistances - the corresponding distances for randomly chosen; paths; randomizationParams - the choice of randomization parameters

Examples

```
chol_branch_point_results = analyseBranchPoint(chol_attributes[,seq_len(3)],
                        chol_pseudo_time[!is.na(chol_pseudo_time)],
                        randomizationParams = c('byPermutation',
                                        'permuteWithinColumns'),
                        statistic = "mean",
                        start = 0,
                        stop = 50,
                        step = 5,
                        nSamples = 10,
                        N = 1)
```

`analyseSingleCellTrajectory`
Analyse a single cell trajectory.

Description

This function analyses a single cell trajectory by sampling multiple paths and comparing each path to random paths. It takes vector of pseudotime values, and a matrix of attribute values (cell x attribute). It also optionally takes the number of pseudotime windows to sample a single cell from. This defaults to 10. The function returns a list of Answers for each comparison of a sampled path to a random path.

Usage

```
analyseSingleCellTrajectory(
  attributes,
  pseudotime,
  randomizationParams,
  statistic,
  nSamples = 1000,
  nWindows = 10,
  d = ncol(attributes),
  N = 1000
)
```

Arguments

attributes	- An n x d (cell x attribute) matrix of numeric attributes for single cell data. Rownames should be cell names.
pseudotime	- A named numeric vector of pseudotime values for cells.
randomizationParams	- A character vector which is used to control the production of randomized paths for comparison.
statistic	- Allowable values are 'median', 'mean' or 'max'.
nSamples	- The number of sampled paths to generate (default 1000).
nWindows	- The number of windows pseudotime should be split into to sample cells from (defaults to 10).
d	- The dimension under consideration. This defaults to ncol(attributes).
N	- The number of random paths to generated for statistical comparison to the given path (defaults to 1000).

Value

This returns a list, where each entry is itself a list containing information comparing a sampled path to random paths. These entries consist of: pValue - the p-value for the path and statistic in question; sphericalData - a list containing the projections of the path to the sphere, the center of that sphere and the statistic for distance to that center; randomDistances - the corresponding distances for randomly chosen; paths; randomizationParams - the choice of randomization parameters

Examples

```
chol_answers = analyseSingleCellTrajectory(chol_attributes[,seq_len(3)],
                                        chol_pseudo_time_normalised,
                                        nSamples = 10,
                                        randomizationParams =
                                                    c('byPermutation',
                                                  'permuteWithinColumns'),
                                        statistic = "mean",
                                        N = 1)
hep_answers = analyseSingleCellTrajectory(hep_attributes[,seq_len(3)],
                                        hep_pseudo_time_normalised,
                                        nSamples = 10,
                                        randomizationParams =
                                                  c('byPermutation',
                                                  'permuteWithinColumns'),
                                        statistic = "mean",
                                        N = 1)
```

chol_answers *chol_answers*

Description

Results of running analyseSingleCellTrajectory() on a trajectory describing the development of cholangiocytes from hepatoblasts.

Usage

```
chol_answers
```

Format

A list

Results of running analyseSingleCellTrajectory() on a trajectory describing the development of cholangiocytes from hepatoblasts.

Source

Single-cell data has been obtained from GEO (GSE90047) and the script used for upstream processing is available at https://github.com/AnnaLaddach/TrajectoryGeometryData

chol_attributes *chol_attributes*

Description

PCA projections derived from normalised gene expression values for single cells, and filtered for cells which feature in a trajectory from hepatoblast to cholangiocyte. The columns are the PCs and the rows are the cells.

Usage

```
chol_attributes
```

Format

A matrix

PCA projections derived from normalised gene expression values for single cells, and filtered for cells which feature in a trajectory from hepatoblast to cholangiocyte. The columns are the PCs and the rows are the cells.

Source

Single-cell data has been obtained from GEO (GSE90047) and the script used for upstream processing is available at https://github.com/AnnaLaddach/TrajectoryGeometryData

chol_branch_point_results *chol_branch_point_results*

Description

Results of running analyseBranchPoint() on a trajectory describing the development of cholangiocytes from hepatoblasts.

Usage

```
chol_branch_point_results
```

Format

A list

Results of running analyseBranchPoint() on a trajectory describing the development of cholangiocytes from hepatoblasts.

Source

Single-cell data has been obtained from GEO (GSE90047) and the script used for upstream processing is available at https://github.com/AnnaLaddach/TrajectoryGeometryData

chol_pseudo_time *chol_pseudo_time*

Description

A vector of pseudotime values for a trajectory describing the development of cholangiocytes from hepatoblasts. Pseudotime values have been inferred using the SlingShot package. The vector is named according to cell ID.

Usage

```
chol_pseudo_time
```

Format

A vector

A vector of pseudotime values for a trajectory describing the development of cholangiocytes from hepatoblasts. Pseudotime values have been inferred using the SlingShot package. The vector is named according to cell ID.

Source

Single-cell data has been obtained from GEO (GSE90047) and the script used for upstream processing is available at https://github.com/AnnaLaddach/TrajectoryGeometryData

---

`chol_pseudo_time_normalised` *chol_pseudo_time_normalised*

Description

A vector of pseudotime values, normalised to range from 0 to 100, for a trajectory describing the development of cholangiocytes from hepatoblasts. Pseudotime values have been inferred using the SlingShot package. The vector is named according to cell ID.

Usage

```
chol_pseudo_time_normalised
```

Format

A vector

A vector of pseudotime values, normalised to range from 0 to 100, for a trajectory describing the development of cholangiocytes from hepatoblasts. Pseudotime values have been inferred using the SlingShot package. The vector is named according to cell ID.

Source

Single-cell data has been obtained from GEO (GSE90047) and the script used for upstream processing is available at https://github.com/AnnaLaddach/TrajectoryGeometryData

---

`circleOnTheUnitSphere` *Circle on the unit sphere*

Description

Find a circle on the unit 2-sphere

Usage

```
circleOnTheUnitSphere(center, radius, N = 36)
```

Arguments

`center`	- The center of the circle.
`radius`	- The radius of the circle.
`N`	- The number of segments to approximate the circle. It defaults to 36.

Details

Given a point on the unit 2-sphere and a radius given as a spherical distance, this finds the circle.

It's not clear to me this should be exported, but it's handy to do this for testing and debugging.

Value

This returns an approximation to the the circle as a N+1 x 3 matrix

Examples

```
pole = c(1,0,0)
radius = pi / 4
circle = circleOnTheUnitSphere(pole,radius)
```

`crooked_path` *Crooked path*

Description

A path of n points in dimension d is an n x d matrix. This particular path is relatively crooked.

Usage

```
crooked_path
```

Format

A 14 x 3 matrix

This path changes direction after the 5th point.

Source

This was created by code in createSyntheticData.R available from https://github.com/AnnaLaddach/TrajectoryGeometryData
The second author wishes to categorically deny that this variable is named after the course of his own life.

crooked_path_center *Crooked path center*

Description

The point on the unit sphere minimizing mean spherical distance to the projection of crooked_path

Usage

```
crooked_path_center
```

Format

A vector of length 3

A unit vector of length 3 minimizing mean spherical distance to the points of the projection of crooked_path.

Source

Synthetic data.

crooked_path_projection *Crooked path projection*

Description

The projection of the last 8 points on crooked_path onto the unit sphere as seen from the the 6th. This is a collection of 8 unit points in dimension 3.

Usage

```
crooked_path_projection
```

Format

An 8 x 3 matrix

The projection of crooked_path[7:14,] onto the unit sphere as seen from crooked_path[6,] .

Source

This was created by code in createSyntheticData.R available from https://github.com/AnnaLaddach/TrajectoryGeometryData

crooked_path_radius *Crooked path radius*

Description

The mean spherical distance from the points of the projection of crooked_path to the point minimizing this mean distance.

Usage

```
crooked_path_radius
```

Format

Numeric

The mean spherical distance from the points of the projection of crooked_path to the point minimizing this mean distance.

Source

This was created by code in createSyntheticData.R available from https://github.com/AnnaLaddach/TrajectoryGeometryData

distanceBetweenTrajectories *Get distances between trajectories.*

Description

This function compares two single cell trajectories (representative of different lineages within the same dataset), and finds the minimum euclidean distance between the first and the second trajectory at each point in pseudotime. Please note, attributes can either be values for single cells, or attributes which have been smoothed over pseudotime. Likewise the pseudotime values should be for single cells, or for smoothed attributes over pseudotime

Usage

```
distanceBetweenTrajectories(attributes1, pseudotime1, attributes2)
```

Arguments

attributes1	- An n x d (cell x attribute) matrix of numeric attributes for the first single cell trajectory.
pseudotime1	- A named numeric vector of pseudotime values for the first single cell trajectory, names should match rownames of atrributes1.
attributes2	- An n x d (cell x attribute) matrix of numeric attributes for the sencond single cell trajectory.

Value

results - a dataframe containing pseudotime values (for the first trajectory), and distances (the minimum euclidian distance between the two trajectories at that point in pseudotime).

Examples

```
distances = distanceBetweenTrajectories(chol_attributes,
                                chol_pseudo_time[!is.na(chol_pseudo_time)],
                                hep_attributes)
```

---

findSphereClusterCenter

Find a center for points on the unit sphere

Description

This function takes a set of points on the d-1 sphere in d-space and finds a center for these. Depending on choice of statistic, this center is a point on the sphere which minimizes either the median distance, the mean distance or the maximum distance of the center to the given points. "Distance" here is taken to mean angle between the points, i.e., arccos of their dot product.

Usage

```
findSphereClusterCenter(points, statistic, normalize = FALSE)
```

Arguments

points	- A set of n points on the (d-1) sphere given as an n x d matrix.
statistic	- The statistic to be minimized. Allowable values are 'median','mean' or 'max'.
normalize	- If this is set to TRUE, the function will start by normalizing the input points.

Value

This returns a point in dimension d given as a vector.

Examples

```
projection = projectPathToSphere(straight_path)
center = findSphereClusterCenter(projection,'mean')
```

findSphericalDistance *Find the spherical distance from a given point to a set of points.*

Description

This function takes a point (typically a center) and a set of points and finds the spherical distance between the given point and each of the others. If requested, it will first normalize all of them.

Usage

```
findSphericalDistance(center, points, normalize = FALSE)
```

Arguments

center	- The proposed point from which distance to the others should be measured. This is a numerical vector of length d.
points	- The set of target points for which spherical distance to the center should be calculated. This is in the form of a n x d matrix.
normalize	- If this is set to TRUE, the function will start by normalizing the input points.

Value

This returns a vector of n spherical distances in radians.

Examples

```
distances = findSphericalDistance(straight_path_center,
    straight_path_projection)
```

generateRandomPaths *Produce random paths modeled on a given path*

Description

This function takes a path and produces N random paths of the same dimension and length based on it. This can be done either by permuting the entries in path or by taking steps from the initial point of path. Exact behaviour is controlled by randomizationParams.

Usage

```
generateRandomPaths(
  path,
  from = 1,
  to = nrow(path),
  d = ncol(path),
  randomizationParams,
  N
)
```

Arguments

path	- This is an mxn dimensional matrix. Each row is considered a point.
from	- The starting place along the path which will be treated as the center of the sphere. This defaults to 1.
to	- The end point of the path. This defaults to nrow(path).
d	- The dimension under consideration. This defaults to ncol(path)
randomizationParams	- A character vector controling the randomization method used. It's first entry must be either 'byPermutation' or 'bySteps' See the vignette for further details.
N	- The number of random paths required.

Value

This function returns a list of random paths. Each path is a matrix.

Examples

```
randomizationParams = c('byPermutation','permuteWithinColumns')
randomPaths = generateRandomPaths(crooked_path,from=6,to=nrow(crooked_path),
              d=ncol(crooked_path),randomizationParams=randomizationParams,
              N=10)
```

---

generateRandomUnitVector

Generate random unit vector.

Description

This function generates a random unit vector in in dimension d.

Usage

```
generateRandomUnitVector(d)
```

Arguments

d	- The dimension.

Value

A unit vector in dimension d.

Examples

```
randomUnitVector = generateRandomUnitVector(5)
```

getDistanceDataForPaths

Produce distance statistics for random paths

Description

This function takes a list of paths and a choice of statistic (median, mean or max) and returns that statistic for the appropriate center for each path. Each path is an n x d matrix. In use, it is assumed that these will be the randomized paths. It is therefore assumed that they are already of the correct dimensions.

Usage

```
getDistanceDataForPaths(paths, statistic)
```

Arguments

paths	- A list of paths. Each of these is an n x d matrix.
statistic	- Allowable values are 'median', 'mean' or 'max'.

Value

This returns a vector of n distances.

Examples

```
paths =
    generateRandomPaths(path=straight_path,randomizationParam='bySteps',N=5)
distance = getDistanceDataForPaths(paths=paths,statistic='max')
```

getSphericalData *This is a simplified wrapper for pathToSphericalData*

Description

It handles the case in which from, to and d are all given by the dimensions of the path

Usage

```
getSphericalData(path, statistic)
```

Arguments

path	- an m x n matrix. Each row is considered a point
statistic	- one of 'mean','median' or 'max'

Value

This function returns a list whose elements are the projections of the path to the sphere, the center for those projections, the median, mean or max distance from the center to those projections and the name of the statistic used.

Examples

```
sphericalData = getSphericalData(straight_path,'max')
```

`getStepLengths` *Find the step lengths:*

Description

This finds the lengths of the steps along a path

Usage

```
getStepLengths(path, from = 1, to = nrow(path), d = ncol(path))
```

Arguments

`path`	- This is an mxn dimensional matrix. Each row is considered a point.
`from`	- The starting place along the path which will be treated as the center of the sphere. This defaults to 1.
`to`	- The end point of the path. This defaults to nrow(path).
`d`	- The dimension under consideration. This defaults to ncol(path)

Value

This function returns the length of each step in a path.

Examples

```
stepLengths = getStepLengths(path=crooked_path)
stepLengths = getStepLengths(path=crooked_path,from=4)
```

hep_answers *hep_answers*

Description

Results of running analyseSingleCellTrajectory() on a trajectory describing the development of hepatocytes from hepatoblasts.

Usage

```
hep_answers
```

Format

A list

Results of running analyseSingleCellTrajectory() on a trajectory describing the development of hepatocytes from hepatoblasts.

Source

Single-cell data has been obtained from GEO (GSE90047) and the script used for upstream processing is available at https://github.com/AnnaLaddach/TrajectoryGeometryData

hep_attributes *hep_attributes*

Description

PCA projections derived from normalised gene expression values for single cells, and filtered for cells which feature in a trajectory from hepatoblast to hepatocyte. The columns are the PCs and the rows are the cells.

Usage

```
hep_attributes
```

Format

A matrix

PCA projections derived from normalised gene expression values for single cells, and filtered for cells which feature in a trajectory from hepatoblast to hepatocyte. The columns are the PCs and the rows are the cells.

Source

Single-cell data has been obtained from GEO (GSE90047) and the script used for upstream processing is available at https://github.com/AnnaLaddach/TrajectoryGeometryData

`hep_pseudo_time` *hep_pseudo_time*

Description

A vector of pseudotime values for a trajectory describing the development of hepatocytes from hepatoblasts. Pseudotime values have been inferred using the SlingShot package. The vector is named according to cell ID.

Usage

```
hep_pseudo_time
```

Format

A vector

A vector of pseudotime values for a trajectory describing the development of hepatocytes from hepatoblasts. Pseudotime values have been inferred using the SlingShot package. The vector is named according to cell ID.

Source

Single-cell data has been obtained from GEO (GSE90047) and the script used for upstream processing is available at https://github.com/AnnaLaddach/TrajectoryGeometryData

---

`hep_pseudo_time_normalised` *hep_pseudo_time_normalised*

Description

A vector of pseudotime values, normalised to range from 0 to 100, for a trajectory describing the development of hepatocytes from hepatoblasts. Pseudotime values have been inferred using the SlingShot package. The vector is named according to cell ID.

Usage

```
hep_pseudo_time_normalised
```

Format

A vector

A vector of pseudotime values for a trajectory describing the development of hepatocytes from hepatoblasts. Pseudotime values have been inferred using the SlingShot package. The vector is named according to cell ID.

Source

Single-cell data has been obtained from GEO (GSE90047) and the script used for upstream processing is available at https://github.com/AnnaLaddach/TrajectoryGeometryData

---

`orthonormalBasis` *Find an orthonormal basis in dimension 3*

Description

Given a vector in R3, this normalizes it and then uses it as the first basis vector in an orthonormal basis. We'll use this to find circles around points on the sphere.

Usage

```
orthonormalBasis(x)
```

Arguments

x	- A vector of length 3

Value

This function returns an orthonormal basis in the the form of a 3 x 3 matrix in which the first vector is parallel to v

Examples

```
anOrthonormalBasis = orthonormalBasis(c(1,1,1))
```

`oscillation` *Oscillation*

Description

This a path which prepends small oscillations to straight path. Its purpose is to illustrate instability of spherical projection near the beginning of a path.

Usage

```
oscillation
```

Format

A matrix

This a path which prepends small oscillations to straight path. Its purpose is to illustrate instability of spherical projection near the beginning of a path.

Source

This was created by code in createSyntheticData.R available from https://github.com/AnnaLaddach/TrajectoryGeometryData

---

`pathProgression` *Measure a path's progression*

Description

This function measures the progress of a path in a specified direction. This direction will typically be the center of its projection onto the sphere as revealed using your favorite statistic.

Usage

```
pathProgression(path, from = 1, to = nrow(path), d = ncol(path), direction)
```

Arguments

`path`	- An n x d matrix
`from`	- The point along the path to be taken as the starting point. This defaults to 1.
`to`	- The point along the path to be used as the end point. This defaults to nrow(path).
`d`	- The dimension to be used. This defaults to ncol(path).
`direction`	- A non-zero numeric whose length is the the dimension.

Value

This returns a numeric given the signed distance projection of the path along the line through its starting point in the given direction.

Examples

```
progress =
    pathProgression(straight_path,direction=straight_path_center)
progress =
    pathProgression(crooked_path,from=6,direction=crooked_path_center)
```

pathToSphericalData *Find the spherical data for a given path*

Description

This function takes a path and returns a list containing its projection to the sphere, the center for that projection, the spherical distance from the center to the points of the projection and the name of the statistic used.

Usage

```
pathToSphericalData(path, from, to, d, statistic)
```

Arguments

path	- This is an mxn dimensional matrix. Each row is considered a point.
from	- The starting place along the path which will be treated as the center of the sphere. This defaults to 1.
to	- The end point of the path. This defaults to nrow(path).
d	- The dimension under consideration. This defaults to ncol(path)
statistic	- One of 'median', 'mean' or 'max'

Value

This function returns a list whose elements are the projections of the path to the sphere, the center for those projections, the median, mean or max distance from the center to those projections and the name of the statistic used.

Examples

```
sphericalData = pathToSphericalData(straight_path,from=1,
                                    to=nrow(straight_path), d=3,
                                    statistic='median')
```

plotPathProjectionCenterAndCircle *Plot a path, its projection, its center and its circle*

Description

This function assumes you have a path in dimension 3 and you have found the projection for the portion under consideration, the center for its projection and the circle (i.e., radius) for the appropriate statistic. Scales the path to keep it comparable to the sphere and plots all this in your favorite color. It can be called repeatedly to add additional paths in different colors.

Usage

```
plotPathProjectionCenterAndCircle(
  path,
  from = 1,
  to = nrow(path),
  projection,
  center,
  radius,
  color,
  circleColor = "white",
  pathPointSize = 8,
  projectionPointSize = 8,
  scale = 1.5,
  newFigure = TRUE
)
```

Arguments

path	- A path of dimension 3 in the form of an N x 3 matrix.
from	- The starting place of the section under consideration. This is used for marking the relevant portion. It defaults to 1.
to	- Likewise. It defaults to nrow(path).
projection	- The projection of the relevant portion of the path.
center	- The center of the projection points.
radius	- The radius of the circle.
color	- The color to use for this path and its associated data.
circleColor	- Sets the colour of the circle. Defaults to white.
pathPointSize	- Sets the size of points which represent the path. Defaults to 8.
projectionPointSize	- Sets the size of points which represent the projected path. Defaults to 8.
scale	- The path will be start (its actual start) at 0 and will be scaled so that its most distant point will be at this distance from the origin. This is to keep it comparable in size to the sphere. It defaults to 1.5. Caution should be used here when plotting multiple paths.
newFigure	- When plotting a single figure or the first of multiple figures, this should be set to TRUE which is its default. Otherwise, set this to FALSE in order to add additional paths to the same figure.

Value

This returns 0.

Examples

```
plotPathProjectionCenterAndCircle(path=straight_path,
                                  projection=straight_path_projection,
                                  center=straight_path_center,
                                  radius=straight_path_radius,
                                  color='red',
                                  newFigure=TRUE)
```

`projectPathToSphere` *Project a path onto the unit sphere*

Description

This function takes a path in d dimensional space and projects it onto the d-1 sphere. It takes as additional arguments the starting and ending points under consideration and the dimension to be considered.

Usage

```
projectPathToSphere(path, from = 1, to = nrow(path), d = ncol(path))
```

Arguments

`path`	- This is an mxn dimensional matrix. Each row is considered a point.
`from`	- The starting place along the path which will be treated as the center of the sphere. This defaults to 1.
`to`	- The end point of the path. This defaults to nrow(path).
`d`	- The dimension under consideration. This defaults to ncol(path)

Value

This returns a projection of the path onto the d-1 sphere in the form of a (to - from) x d matrix.

Examples

```
projection1 = projectPathToSphere(straight_path)
projection2 = projectPathToSphere(crooked_path,from=6)
```

samplePath *Sample a path from single cell data*

Description

This function takes vector of pseudotime values, and a matrix of attribute values (cell x attribute). It also optionally takes the number of pseudotime windows to sample a single cell from. This defaults to 10. The function returns a matrix of sampled attribute values which form the coordinates of the sampled path. The matrix of attribute values should consist of numeric values relevant to a pseudotime trajectory i.e. gene expression values or PCA projections. The vector of pseudotime values should be named according to cell names. Simarly the row names of the matrix of attribute values should be cell names. Row names for the returned matrix of the sampled path give the window number a cell was sampled from.

Usage

```
samplePath(attributes, pseudotime, nWindows = 10)
```

Arguments

attributes	- An n x d (cell x attribute) matrix of numeric attributes for single cell data. Rownames should be cell names.
pseudotime	- A named numeric vector of pseudotime values for cells.
nWindows	- The number of windows pseudotime should be split into to sample cells from. Defaults to 10.

Value

sampledPath - A path consisting of a matrix of attributes of sampled cells. The rownames refer to the pseudotime windows cell was sampled from.

Examples

```
samplePath(chol_attributes, chol_pseudo_time_normalised)
samplePath(hep_attributes, hep_pseudo_time_normalised)
```

single_cell_matrix *single_cell_matrix*

Description

PCA projections derived from normalised gene expression values for single cells. The columns are the PCs and the rows are the cells.

Usage

```
single_cell_matrix
```

Format

A matrix

PCA projections derived from normalised gene expression values for single cells. The columns are the PCs and the rows are the cells.

Source

Single-cell data has been obtained from GEO (GSE90047) and the script used for upstream processing is available at https://github.com/AnnaLaddach/TrajectoryGeometryData

---

straight_path *Straight path*

Description

A path of n points in dimension d is an n x d matrix. This particular path is relatively straight.

Usage

```
straight_path
```

Format

A 14 x 3 matrix

The path is roughly in the (1,0,0) direction.

Source

This was created by code in createSyntheticData.R available from https://github.com/AnnaLaddach/TrajectoryGeometryData

straight_path_center *Straight path center*

Description

The point on the unit sphere minimizing mean spherical distance to the projection so straight_path

Usage

```
straight_path_center
```

Format

A vector of length 3

A unit vector of length 3 minimizing distance to the points of the projection of straight_path.

Source

This was created by code in createSyntheticData.R available from https://github.com/AnnaLaddach/TrajectoryGeometryData

straight_path_projection *Straight path projection*

Description

The projection of straight_path onto the unit sphere. This is a collection of 13 unit points in dimension 3

Usage

```
straight_path_projection
```

Format

A 13 x 3 matrix

The projection of straight_path[2:14,] onto the unit sphere as seen from straight_path[1,] .

Source

This was created by code in createSyntheticData.R available from https://github.com/AnnaLaddach/TrajectoryGeometryData

straight_path_radius *Straight path radius*

Description

The mean spherical distance from the points of the projection of straight_path to the point minimizing this mean distance.

Usage

```
straight_path_radius
```

Format

Numeric

The mean spherical distance from the points of the projection of straight_path to the point minimizing this mean distance.

Source

This was created by code in createSyntheticData.R available from https://github.com/AnnaLaddach/TrajectoryGeometryData

testPathForDirectionality *Test a path for directionality*

Description

This is the core function of this package. It takes a path, and a choice of statistical measure and computes a statistical significance for the directionality of that path.

Usage

```
testPathForDirectionality(
  path,
  from = 1,
  to = nrow(path),
  d = ncol(path),
  randomizationParams,
  statistic,
  N
)
```

Arguments

path	- An n x m matrix representing a series of n points in dimension m.
from	- The starting place along the path which will be treated as the center of the sphere. This defaults to 1.
to	- The end point of the path. This defaults to nrow(path).
d	- The dimension under consideration. This defaults to ncol(path)
randomizationParams	- A character vector which is used to control the production of randomized paths for comparison.
statistic	- Allowable values are 'median', 'mean' or 'max'
N	- The number of random paths to generated for statistical comparison to the given path.

Value

This returns a list giving whose entries are: pValue - the p-value for the path and statistic in question; sphericalData - a list containing the projections of the path to the sphere, the center of that sphere and the statistic for distance to that center; randomDistances - the corresponding distances for randomly chosen; paths; randomizationParams - the choice of randomization parameters

Examples

```
randomizationParams = c('byPermutation','permuteWithinColumns')
p = testPathForDirectionality(path=straight_path,
                              randomizationParams=randomizationParams,
                              statistic='median',N=100)
q = testPathForDirectionality(path=crooked_path,from=6,
                              randomizationParams=randomizationParams,
                              statistic='median',N=100)
```

---

visualiseBranchPointStats

Visualise Branch Point Stats

Description

This function creates plots and extracts statistics for analysing branch points. It returns plots and underlying data for visualising distance metrics and -log10 transformed pvalues (comparison to random trajectories) for trajectories with different starting points.

Usage

```
visualiseBranchPointStats(branchPointData, average = "mean")
```

Arguments

branchPointData	- the result of analyseBranchPoint
average	- if there are multiple distances available for each sampled trajectory, calculate the average using "mean" or "median" (defaults to "mean").

Value

a list containing: branchPointValues - dataframe containing data underlying distance plot in long format pValues- dataframe containing data underlying p-value plot in long format distancePlot - ggplot object, violin plots of distance metric for sampled paths for different trajectory different starting points pValue - ggplot object, line plot of -log10 transformed p-values for comparing sampled paths to random paths for different trajectory starting points

Examples

```
cholBranchPointStats = visualiseBranchPointStats(chol_branch_point_results)
```

---

visualiseTrajectoryStats

Visualise Trajectory Stats

Description

This function creates plots and extracts statistics for comparisons of metrics for sampled paths to random paths. It can also create plots for comparing two sets of sampled paths by providing the traj2Data argument.

Usage

```
visualiseTrajectoryStats(
  traj1Data,
  metric,
  average = "mean",
  traj2Data = list()
)
```

Arguments

traj1Data	- the result of analyseSingleCellTrajectory
metric	- either "pValue" or "distance"
average	- if there are multiple distances available for each sampled trajectory, calculate the average using "mean" or "median" (defaults to "mean").
traj2Data	- traj2Data either an empty list or the result of analyseSingleCellTrajectory

Value

a list containing: stats - output of wilcox test (paired if comparing sampled to random paths, unpaired if comparing sampled paths for two different trajectories) values - dataframe containing plotted data in long format plot - ggplot object

Examples

```
cholResultDistance = visualiseTrajectoryStats(chol_answers, "distance")
hepResultDistance = visualiseTrajectoryStats(hep_answers, "distance")
distanceComparison = visualiseTrajectoryStats(chol_answers, "distance",
    traj2Data = hep_answers)
```

Index

∗ **datasets**
  chol_answers, 6
  chol_attributes, 6
  chol_branch_point_results, 7
  chol_pseudo_time, 7
  chol_pseudo_time_normalised, 8
  crooked_path, 9
  crooked_path_center, 10
  crooked_path_projection, 10
  crooked_path_radius, 11
  hep_answers, 17
  hep_attributes, 17
  hep_pseudo_time, 18
  hep_pseudo_time_normalised, 18
  oscillation, 19
  single_cell_matrix, 24
  straight_path, 25
  straight_path_center, 26
  straight_path_projection, 26
  straight_path_radius, 27

analyseBranchPoint, 3
analyseSingleCellTrajectory, 4

chol_answers, 6
chol_attributes, 6
chol_branch_point_results, 7
chol_pseudo_time, 7
chol_pseudo_time_normalised, 8
circleOnTheUnitSphere, 8
crooked_path, 9
crooked_path_center, 10
crooked_path_projection, 10
crooked_path_radius, 11

distanceBetweenTrajectories, 11

findSphereClusterCenter, 12
findSphericalDistance, 13

generateRandomPaths, 13
generateRandomUnitVector, 14
getDistanceDataForPaths, 15
getSphericalData, 15
getStepLengths, 16

hep_answers, 17
hep_attributes, 17
hep_pseudo_time, 18
hep_pseudo_time_normalised, 18

orthonormalBasis, 19
oscillation, 19

pathProgression, 20
pathToSphericalData, 21
plotPathProjectionCenterAndCircle, 21
projectPathToSphere, 23

samplePath, 24
single_cell_matrix, 24
straight_path, 25
straight_path_center, 26
straight_path_projection, 26
straight_path_radius, 27

testPathForDirectionality, 27

visualiseBranchPointStats, 28
visualiseTrajectoryStats, 29