Title  PHATE - Potential of Heat-Diffusion for Affinity-Based Transition Embedding

Description  PHATE is a tool for visualizing high dimensional single-cell data with natural progressions or trajectories. PHATE uses a novel conceptual framework for learning and visualizing the manifold inherent to biological systems in which smooth transitions mark the progressions of cells from one state to another. To see how PHATE can be applied to single-cell RNA-seq datasets from hematopoietic stem cells, human embryonic stem cells, and bone marrow samples, check out our publication in Nature Biotechnology at <doi:10.1038/s41587-019-0336-3>.

License  GPL-2 | file LICENSE

Encoding  UTF-8

LazyData  true

Depends  R (>= 3.3), Matrix (>= 1.2-0)

Imports  methods, stats, graphics, reticulate (>= 1.8), ggplot2

Suggests  gridGraphics, cowplot

RoxygenNote  7.0.2

NeedsCompilation  no

Author  Krishnan Srinivasan [aut], Scott Gigante [cre] (<https://orcid.org/0000-0002-4544-2764>)

Maintainer  Scott Gigante <scott.gigante@yale.edu>

Repository  CRAN

Date/Publication  2019-12-05 19:00:05 UTC

R topics documented:

  as.data.frame.phate .............................................. 2
  as.matrix.phate .................................................... 2
  cluster_phate ..................................................... 3
  ggplot.phate ...................................................... 4
as.data.frame.phate: Convert a PHATE object to a data.frame

Description
Returns the embedding matrix with column names PHATE1 and PHATE2

Usage
## S3 method for class 'phate'
as.data.frame(x, ...)

Arguments
x A fitted PHATE object
...
Arguments for as.data.frame()

as.matrix.phate: Convert a PHATE object to a matrix

Description
Returns the embedding matrix. All components can be accessed using phate$embedding, phate$diff.op, etc

Usage
## S3 method for class 'phate'
as.matrix(x, ...)

Arguments
x A fitted PHATE object
...
Arguments for as.matrix()
Description

KMeans on the PHATE potential Clustering on the PHATE operator as introduced in Moon et al. This is similar to spectral clustering.

Usage

cluster_phate(phate, k = 8, seed = NULL)

Arguments

  phate       phate() output
  k           Number of clusters (default: 8)
  seed        Random seed for kmeans (default: NULL)

Value

clusters Integer vector of cluster assignments

Examples

if (reticulate::py_module_available("phate")) {

  # Load data
  # data(tree.data)
  # We use a smaller tree to make examples run faster
  data(tree.data.small)

  # Run PHATE
  phate.tree <- phate(tree.data.small$data)

  # Clustering
  cluster_phate(phate.tree)
}
ggplot.phate  
Convert a PHATE object to a data.frame for ggplot

Description

Passes the embedding matrix to ggplot with column names PHATE1 and PHATE2

Usage

```r
## S3 method for class 'phate'
ggplot(data, ...)
```

Arguments

- `data`: A fitted PHATE object
- `...`: Arguments for `ggplot()`

Examples

```r
if (reticulate::py_module_available("phate") && require(ggplot2)) {
  # data(tree.data)
  # We use a smaller tree to make examples run faster
  data(tree.data.small)
  phate.tree <- phate(tree.data.small$data)
  ggplot(phate.tree, aes(x=PHATE1, y=PHATE2, color=tree.data.small$branches)) +
  geom_point()
}
```

install.phate  
Install PHATE Python Package

Description

Install PHATE Python package into a virtualenv or conda env.

Usage

```r
install.phate(
  envname = "r-reticulate",
  method = "auto",
  conda = "auto",
  pip = TRUE,
  ...
)
```
library.size.normalize

Arguments

envname Name of environment to install packages into

method Installation method. By default, "auto" automatically finds a method that will work in the local environment. Change the default to force a specific installation method. Note that the "virtualenv" method is not available on Windows.

conda Path to conda executable (or "auto" to find conda using the PATH and other conventional install locations).

pip Install from pip, if possible.

Additional arguments passed to conda_install() or virtualenv_install().

Details

On Linux and OS X the "virtualenv" method will be used by default ("conda" will be used if virtualenv isn’t available). On Windows, the "conda" method is always used.

library.size.normalize

Performs L1 normalization on input data such that the sum of expression values for each cell sums to 1, then returns normalized matrix to the metric space using median UMI count per cell effectively scaling all cells as if they were sampled evenly.

Description

Performs L1 normalization on input data such that the sum of expression values for each cell sums to 1, then returns normalized matrix to the metric space using median UMI count per cell effectively scaling all cells as if they were sampled evenly.

Usage

library.size.normalize(data, verbose = FALSE)

Arguments

data matrix (n_samples, n_dimensions) 2 dimensional input data array with n cells and p dimensions

verbose boolean, default=FALSE. If true, print verbose output

Value

data_norm matrix (n_samples, n_dimensions) 2 dimensional array with normalized gene expression values
phate

Run PHATE on an input data matrix

Description

PHATE is a data reduction method specifically designed for visualizing **high** dimensional data in **low** dimensional spaces.

Usage

```r
phate(
  data,
  ndim = 2,
  knn = 5,
  decay = 40,
  n.landmark = 2000,
  gamma = 1,
  t = "auto",
  knn.dist.method = "euclidean",
  init = NULL,
  mds.solver = "sgd",
  mds.method = "metric",
  mds.dist.method = "euclidean",
  t.max = 100,
  npca = 100,
  plot.optimal.t = FALSE,
  verbose = 1,
  n.jobs = 1,
  seed = NULL,
  potential.method = NULL,
  k = NULL,
  alpha = NULL,
  use.alpha = NULL,
  ...
)
```

Arguments

data: matrix (n_samples, n_dimensions) 2 dimensional input data array with n_samples samples and n_dimensions dimensions. If knn.dist.method is ‘precomputed’, data is treated as a (n_samples, n_samples) distance or affinity matrix.

ndim: int, optional, default: 2 number of dimensions in which the data will be embedded.

t: int, optional, default: 5 number of nearest neighbors on which to build kernel.

decay: int, optional, default: 40 sets decay rate of kernel tails. If NULL, alpha decaying kernel is not used.
n. landmark int, optional, default: 2000 number of landmarks to use in fast PHATE

gamma float, optional, default: 1 Informational distance constant between -1 and 1.
gamma=1 gives the PHATE log potential, gamma=0 gives a square root potential.

t int, optional, default: 'auto' power to which the diffusion operator is powered
sets the level of diffusion

mds.solver 'sgd', 'smacof', optional, default: 'sgd' which solver to use for metric MDS.
SGD is substantially faster, but produces slightly less optimal results. Note that
SMACOF was used for all figures in the PHATE paper.

knn.dist.method string, optional, default: 'euclidean'. recommended values: 'euclidean', 'cosine', 'precomputed' Any metric from scipy.spatial.distance can be used
distance metric for building kNN graph. If 'precomputed', data should be an
n_samples x n_samples distance or affinity matrix. Distance matrices are assumed
to have zeros down the diagonal, while affinity matrices are assumed to
have non-zero values down the diagonal. This is detected automatically using
data[0,0]. You can override this detection with knn.dist.method='precomputed_distance'
or knn.dist.method='precomputed_affinity'.

init phate object, optional object to use for initialization. Avoids recomputing inter-
mediate steps if parameters are the same.

mds.method string, optional, default: 'classic', 'metric', and 'non-
metric' which MDS algorithm is used for dimensionality reduction

mds.dist.method string, optional, default: 'euclidean' recommended values: 'euclidean' and 'cosine'

t.max int, optional, default: 100. Maximum value of t to test for automatic t selection.
npca int, optional, default: 100 Number of principal components to use for calculating neighborhoods. For extremely large datasets, using n_pca < 20 allows
neighborhoods to be calculated in log(n_samples) time.

plot.optimal.t boolean, optional, default: FALSE If TRUE, produce a plot showing the Von
Neumann Entropy curve for automatic t selection.

verbose int or boolean, optional (default : 1) If TRUE or > 0, print verbose updates.
n.jobs int, optional (default: 1) The number of jobs to use for the computation. If -1
all CPUs are used. If 1 is given, no parallel computing code is used at all, which
is useful for debugging. For n_jobs below -1, (n.cpus + 1 + n.jobs) are used.
Thus for n_jobs = -2, all CPUs but one are used

seed int or NULL, random state (default: NULL)
potential.method Deprecated. For log potential, use gamma=1. For sqrt potential, use gamma=0.

k Deprecated. Use knn.

alpha Deprecated. Use decay.

use.alpha Deprecated To disable alpha decay, use alpha=NULL

... Additional arguments for graphtools.Graph.
Value

"phate" object containing:

- **embedding**: the PHATE embedding
- **operator**: The PHATE operator (python phate.PHATE object)
- **params**: Parameters passed to phate

Examples

```r
if (reticulate::py_module_available("phate")) {

  # Load data
  # data(tree.data)
  # We use a smaller tree to make examples run faster
  data(tree.data.small)

  # Run PHATE
  phate.tree <- phate(tree.data.small$data)
  summary(phate.tree)
  ## PHATE embedding
  ## knn = 5, decay = 40, t = 58
  ## Data: (3000, 100)
  ## Embedding: (3000, 2)

  library(graphics)
  # Plot the result with base graphics
  plot(phate.tree, col=tree.data.small$branches)
  # Plot the result with ggplot2
  if (require(ggplot2)) {
    ggplot(phate.tree) +
    geom_point(aes(x=PHATE1, y=PHATE2, color=tree.data.small$branches))
  }

  # Run PHATE again with different parameters
  # We use the last run as initialization
  phate.tree2 <- phate(tree.data.small$data, t=150, init=phate.tree)
  # Extract the embedding matrix to use in downstream analysis
  embedding <- as.matrix(phate.tree2)
}
```

**plot.phate**

Plot a PHATE object in base R

Description

Plot a PHATE object in base R
print.phate

Usage

```r
## S3 method for class 'phate'
plot(x, ...)
```

Arguments

- `x`: A fitted PHATE object
- `...`: Arguments for `plot()`

Examples

```r
if (reticulate::py_module_available("phate")) {
  library(graphics)
  # data(tree.data)
  # We use a smaller tree to make examples run faster
  data(tree.data.small)
  phate.tree <- phate(tree.data.small$data)
  plot(phate.tree, col=tree.data.small$branches)
}
```

---

print.phate

*Print a PHATE object*

Description

This avoids spamming the user’s console with a list of many large matrices

Usage

```r
## S3 method for class 'phate'
print(x, ...)
```

Arguments

- `x`: A fitted PHATE object
- `...`: Arguments for `print()`

Examples

```r
if (reticulate::py_module_available("phate")) {

  # data(tree.data)
  # We use a smaller tree to make examples run faster
  data(tree.data.small)
  phate.tree <- phate(tree.data.small$data)
  print(phate.tree)
}
```
## PHATE embedding with elements

## $embedding : (3000, 2)

## $operator : Python PHATE operator

## $params : list with elements (data, knn, decay, t, n.landmark, ndim,
## gamma, npca, mds.method,
## knn.dist.method, mds.dist.method)

---

**summary.phate**

*Summarize a PHATE object*

### Description

Summarize a PHATE object

### Usage

```r
## S3 method for class 'phate'
summary(object, ...)
```

### Arguments

- `object` A fitted PHATE object
- `...` Arguments for `summary()

### Examples

```r
if (reticulate::py_module_available("phate")) {
  # data(tree.data)
  # We use a smaller tree to make examples run faster
data(tree.data.small)
phate.tree <- phate(tree.data.small$data)
summary(phate.tree)
  ## PHATE embedding
  ## knn = 5, decay = 40, t = 58
  ## Data: (3000, 100)
  ## Embedding: (3000, 2)
}
```
**tree.data**

*Fake branching data for examples*

---

**Description**

A dataset containing high dimensional data that has 10 unique branches

**Usage**

tree.data

**Format**

A list containing data, a matrix with 3000 rows and 100 variables and branches, a factor containing 3000 elements.

**Source**

The authors

---

**tree.data.small**

*Fake branching data for running examples fast*

---

**Description**

A dataset containing high dimensional data that has 10 unique branches

**Usage**

tree.data.small

**Format**

A list containing data, a matrix with 250 rows and 50 variables and branches, a factor containing 250 elements.

**Source**

The authors
Index

*Topic datasets
  tree.data, 11
  tree.data.small, 11

as.data.frame.phate, 2
as.matrix.phate, 2

cluster_phate, 3
ggplot.phate, 4
install.phate, 4
library.size.normalize, 5

phate, 6
plot.phate, 8
print.phate, 9

summary.phate, 10

tree.data, 11
tree.data.small, 11