Package ‘TDbasedUFE’

January 13, 2024

Type  Package
Title  Tensor Decomposition Based Unsupervised Feature Extraction
Version  1.2.0
Description  This is a comprehensive package to perform
 Tensor decomposition based unsupervised feature extraction.
 It can perform unsupervised feature extraction.
 It uses tensor decomposition.
 It is applicable to gene expression, DNA methylation, and
 histone modification etc.
 It can perform multiomics analysis.
 It is also potentially applicable to single cell omics data sets.

biocViews  GeneExpression, FeatureExtraction, MethylationArray,
 SingleCell

License  GPL-3
Encoding  UTF-8
LazyData  false

URL  https://github.com/tagtag/TDbasedUFE
BugReports  https://github.com/tagtag/TDbasedUFE/issues
Imports  GenomicRanges, rTensor, readr, methods, MOFAdata, tximport,
 tximportData, graphics, stats, utils, shiny

RoxygenNote  7.2.3
Roxygen  list(markdown = TRUE)
Suggests  BiocStyle, knitr, rmarkdown, testthat (>= 3.0.0)
VignetteBuilder  knitr

Config/testthat/edition  3
git_url  https://git.bioconductor.org/packages/TDbasedUFE
git_branch  RELEASE_3_18
git_last_commit  7f51f6b
git_last_commit_date  2023-10-24
Repository Bioconductor 3.18
Date/Publication 2024-01-12
Author Y-h. Taguchi [aut, cre] (<https://orcid.org/0000-0003-0867-8986>)
Maintainer Y-h. Taguchi <tag@granular.com>

R topics documented:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>TDbasedUFE-package</td>
<td>2</td>
</tr>
<tr>
<td>computeHosvd</td>
<td>3</td>
</tr>
<tr>
<td>computeHosvdSqure</td>
<td>3</td>
</tr>
<tr>
<td>convertSquare</td>
<td>4</td>
</tr>
<tr>
<td>PrepareSummarizedExperimentTensor</td>
<td>5</td>
</tr>
<tr>
<td>PrepareSummarizedExperimentTensorSquare</td>
<td>6</td>
</tr>
<tr>
<td>selectFeature</td>
<td>7</td>
</tr>
<tr>
<td>selectFeatureSquare</td>
<td>7</td>
</tr>
<tr>
<td>selectSingularValueVectorLarge</td>
<td>9</td>
</tr>
<tr>
<td>selectSingularValueVectorSmall</td>
<td>9</td>
</tr>
<tr>
<td>tableFeatures</td>
<td>10</td>
</tr>
<tr>
<td>tableFeaturesSquare</td>
<td>11</td>
</tr>
</tbody>
</table>

Index 12

---

**Description**

This is a comprehensive package to perform Tensor decomposition based unsupervised feature extraction. It can perform unsupervised feature extraction. It uses tensor decomposition. It is applicable to gene expression, DNA methylation, and histone modification etc. It can perform multiomics analysis. It is also potentially applicable to single cell omics data sets.

**Author(s)**

**Maintainer:** Y-h. Taguchi <tag@granular.com> (ORCID)

**See Also**

Useful links:

- https://github.com/tagtag/TDbasedUFE
- Report bugs at https://github.com/tagtag/TDbasedUFE/issues
computeHosvd

Title Compute higher order singular value decomposition

Description
Title Compute higher order singular value decomposition

Usage
computeHosvd(Z, dims = c(10, dim(attr(Z, "value"))[1]), scale = TRUE)

Arguments
- Z: array that includes omics data
- dims: dimensions to be computed by HOSVD
- scale: If value is scaled

Value
List that includes output from HOSVD

Examples
Z <- PrepareSummarizedExperimentTensor(
  sample=matrix(as.character(seq_len(6)),c(3,2)),
  feature=as.character(seq_len(10)),
  value=array(runif(10*3*2),c(10,3,2)))
HOSVD <- computeHosvd(Z)

computeHosvdSquare

Title Compute higher order singular value decomposition from the tensor generated from squared matrix

Description
Title Compute higher order singular value decomposition from the tensor generated from squared matrix

Usage
computeHosvdSquare(
  Z,
  dims = unlist(lapply(dim(attr(Z, "value")), function(x) {min(10, x)})),
  scale = TRUE
)
convertSquare

Description

Generate squared tensor from multiomics data

Usage

convertSquare(Multi)

Arguments

Multi A list that include multiomics data

Value

A tensor computed from multiomics data

Arguments

Z A tensor including sample names, feature values, associated with featureRange and sample properties
dims dimensions to be computed by HOSVD
scale If value is scaled

Value

List that includes output from HOSVD

Examples

omics1 <- matrix(runif(100),10)
dimnames(omics1) <- list(seq_len(10),seq_len(10))
omics2 <- matrix(runif(100),10)
dimnames(omics2) <- dimnames(omics1)
Multi <- list(omics1,omics2)
Z <- PrepareSummarizedExperimentTensorSquare(sample=matrix(colnames(omics1),1),
feature=list(omics1=rownames(omics1),
omics2=rownames(omics2)),
value=convertSquare(Multi),
sampleData=list(NA))
HOSVD <- computeHosvdSquare(Z)
Examples

```r
omics1 <- matrix(runif(100),10)
dimnames(omics1) <- list(seq_len(10),seq_len(10))
omics2 <- matrix(runif(100),10)
dimnames(omics2) <- dimnames(omics1)
Multi <- list(omics1,omics2)
Z <- convertSquare(Multi)
```

Description

Title Generate feature values formatted as a tensor format

Usage

```r
PrepareSummarizedExperimentTensor(
  sample,
  feature,
  value,
  featureRange = GRanges(NULL),
  sampleData = list(NULL)
)
```

Arguments

- `sample`: Sample names
- `feature`: Feature id names
- `value`: Feature values
- `featureRange`: Genomic coordinate attributed to feature id (if any)
- `sampleData`: Sample property (labels etc)

Value

A tensor including sample names, feature id, feature values, associated with featureRange and sample properties

Examples

```r
require(GenomicRanges)
Z <- PrepareSummarizedExperimentTensor(
  sample=matrix(as.character(seq_len(6)),c(3,2)),
  feature=as.character(seq_len(10)),
  value=as.character(seq_len(10)),
  featureRange = GRanges(NULL),
  sampleData = list(NULL)
)
```
PrepareSummarizedExperimentTensorSquare

Title Generate feature values formatted as a tensor format from Squared matrix

Description
Title Generate feature values formatted as a tensor format from Squared matrix

Usage

PrepareSummarizedExperimentTensorSquare(
  sample = list(NULL),
  feature, 
  value, 
  featureRange = GRanges(NULL),
  sampleData = list(NULL)
)

Arguments

sample Sample names
feature Feature id names
value Squared Feature values
featureRange Genomic coordinate attributed to feature id (if any)
sampleData Sample property (labels etc)

Value

A tensor including sample names, feature values, associated with featureRange and sample properties

Examples

omics1 <- matrix(runif(100),10)
dimnames(omics1) <- list(seq_len(10),seq_len(10))
omics2 <- matrix(runif(100),10)
dimnames(omics2) <- dimnames(omics1)
Multi <- list(omics1,omics2)
Z <- PrepareSummarizedExperimentTensorSquare(
  sample=matrix(colnames(omics1),1),
  feature=list(omics1=rownames(omics1),
    omics2=rownames(omics2)),
  value=convertSquare(Multi),
  sampleData=list(NA))
selectFeature

Title Select features

Description

Title Select features

Usage

selectFeature(HOSVD, input_all, de = 1e-04, p0 = 0.01, breaks = 100)

Arguments

- HOSVD: output from HOSVD
- input_all: Selected singular value IDs
- de: Initial value for optimization of standard deviation
- p0: Threshold P-value
- breaks: The number of bins

Value

List that includes selected features and computed P-value

Examples

```r
set.seed(2)
require(rTensor)
HOSVD <- hosvd(as.tensor(array(runif(10000*3*3),c(10000,3,3))),c(10,3,3))
input_all <- c(2,2)
index <- selectFeature(HOSVD,input_all,de=0.01,p0=0.01)
```

selectFeatureSquare

Title Select features (for tensor generated from squared matrix)

Description

Title Select features (for tensor generated from squared matrix)
selectFeatureSquare

Usage

```
selectFeatureSquare(
    HOSVD,
    input_all,
    Multi,
    de = rep(1e-04, dim(HOSVD$U[[3]])[2]),
    p0 = 0.01,
    breaks = 100,
    interact = TRUE
)
```

Arguments

- **HOSVD**: output from HOSVD applied to tensor generated from squared matrix
- **input_all**: Selected singular value vector IDs
- **Multi**: Multimomics data
- **de**: Initial value for optimization of standard deviation
- **p0**: Threshold P-value
- **breaks**: The number of bins
- **interact**: if interact mode or not

Value

List that includes selected features and computed P-value

Examples

```r
omics1 <- matrix(runif(100000), ncol=10)
dimnames(omics1) <- list(seq_len(10000), seq_len(10))
omics2 <- matrix(runif(100000), ncol=10)
dimnames(omics2) <- dimnames(omics1)
Multi <- list(omics1, omics2)
Z <- PrepareSummarizedExperimentTensorSquare(
    sample=matrix(colnames(omics1),1),
    feature=list(omics1=rownames(omics1),
                 omics2=rownames(omics2)),
    value=convertSquare(Multi),
    sampleData=list(NA))
HOSVD <- computeHosvdSquare(Z)
cond <- list(0, rep(seq_len(2), each=5), c("A", "B"))
input_all <- selectSingularValueVectorLarge(HOSVD, cond, input_all=c(1,1))
index <- selectFeatureSquare(HOSVD, input_all, Multi, de=c(0.1, 0.1),
                           interact=FALSE)
```
selectSingularValueVectorLarge

Title Select singular value vectors from HOSVD (boxplot version)

Description
Title Select singular value vectors from HOSVD (boxplot version)

Usage
selectSingularValueVectorLarge(HOSVD, cond, input_all = NULL)

Arguments
HOSVD output from HOSVD
cond Labels to select singular value vector number
input_all if list is not null, no interactive mode is activated but provided values are used.

Value
Selected singular value vector IDs

Examples
Z <- PrepareSummarizedExperimentTensor(
sample=matrix(as.character(seq_len(6)),c(3,2)),
feature=as.character(seq_len(10)),
value=array(runif(10*3*2),c(10,3,2)))
HOSVD <- computeHosvd(Z)
cond <- list(0,c("A","B","C"),c("A","B"))
input_all <- selectSingularValueVectorLarge(HOSVD,cond,input_all=c(1,1))

selectSingularValueVectorSmall

Title Select singular value vectors from HOSVD

Description
Title Select singular value vectors from HOSVD

Usage
selectSingularValueVectorSmall(HOSVD, input_all = NULL)
Arguments

- **HOSVD**: output from HOSVD
- **input_all**: if ist is no null, no interactive mode is activated but provided values are used.

Value

Selected singular value vector IDs

Examples

```r
Z <- PrepareSummarizedExperimentTensor(
  sample=matrix(as.character(seq_len(6)),c(3,2)),
  feature=as.character(seq_len(10)),
  value=array(runif(10*3*2),c(10,3,2)))
HOSVD <- computeHosvd(Z)
input_all <- selectSingularValueVectorSmall(HOSVD,input_all=c(1,1))

head(tableFeatures(Z, index))
```

Description

Title Show selected features as Table

Usage

```r
tableFeatures(Z, index)
```

Arguments

- **Z**: Tensor of features
- **index**: List that includes selected features and P-values

Value

Table list of selected features

Examples

```r
set.seed(2)
require(rTensor)
HOSVD <- hosvd(as.tensor(array(runif(10000*3*3),c(10000,3,3))),c(10,3,3))
input_all <- c(2,2)
index <- selectFeature(HOSVD,input_all,de=0.01,p0=0.01)
index$index[seq_len(100)] <- TRUE
Z <- PrepareSummarizedExperimentTensor(
  sample=matrix(as.character(seq_len(9)),c(3,3)),
  feature=as.character(seq_len(10000)),
  value=array(runif(10000*3*3),c(10,3,3)))
head(tableFeatures(Z,index))
```
**Description**

Title Show selected features as Table (for Squared one)

**Usage**

```r
tableFeaturesSquare(Z, index, id)
```

**Arguments**

- `Z`: Tensor of features
- `index`: List that includes selected features and P-values
- `id`: feature to be shown

**Value**

Table list of selected features

**Examples**

```r
omics1 <- matrix(runif(100000), ncol=10)
dimnames(omics1) <- list(seq_len(10000), seq_len(10))
omics2 <- matrix(runif(100000), ncol=10)
dimnames(omics2) <- dimnames(omics1)
Multi <- list(omics1, omics2)
Z <- PrepareSummarizedExperimentTensorSquare(
sample=matrix(colnames(omics1), 1),
feature=list(omics1=rownames(omics1),
omics2=rownames(omics2)),
value=convertSquare(Multi),
sampleData=list(NA))
HOSVD <- computeHosvdSquare(Z)
cond <- list(0, rep(seq_len(2), each=5), c("A", "B"))
input_all <- selectSingularValueVectorLarge(HOSVD, cond, input_all=c(1, 1))
index <- selectFeatureSquare(HOSVD, input_all, Multi, de=c(0.1, 0.1), interact=FALSE)
index[[1]]$index[1:100] <- TRUE
index[[1]]$p.value[1:100] <- 1e-3
tableFeaturesSquare(Z, index, 1)
```
Index

* internal
  TDbasedUFE-package, 2

computeHosvd, 3
computeHosvdSqr, 3
convertSquare, 4

PrepareSummarizedExperimentTensor, 5
PrepareSummarizedExperimentTensorSquare, 6

selectFeature, 7
selectFeatureSquare, 7
selectSingularValueVectorLarge, 9
selectSingularValueVectorSmall, 9

tableFeatures, 10
tableFeaturesSquare, 11
TDbasedUFE (TDbasedUFE-package), 2
TDbasedUFE-package, 2