Package ‘phemd’

October 27, 2023

Type Package
Title Phenotypic EMD for comparison of single-cell samples
Version 1.18.0
Description Package for comparing and generating a low-dimensional embedding of multiple single-cell samples.
License GPL-2
Encoding UTF-8
LazyData true
Depends R (>= 4.0), monocle, Seurat
Imports SingleCellExperiment, RColorBrewer, igraph, transport, pracma, cluster, Rtsne, destiny, RANN, ggplot2, maptree, pheatmap, scatterplot3d, VGAM, methods, grDevices, graphics, stats, utils, cowplot, S4Vectors, BiocGenerics, SummarizedExperiment, Biobase, phateR, reticulate
Config/reticulate list( packages = list( package = "phate" ) )
Suggests knitr, BiocStyle
VignetteBuilder knitr
biocViews Clustering, ComparativeGenomics, Proteomics, Transcriptomics, Sequencing, DimensionReduction, SingleCell, DataRepresentation, Visualization, MultipleComparison
RoxygenNote 7.0.2
git_url https://git.bioconductor.org/packages/phemd
git_branch RELEASE_3_18
git_last_commit c895e7c
git_last_commit_date 2023-10-24
Date/Publication 2023-10-26
Author William S Chen [aut, cre]
Maintainer William S Chen <wil.yum.chen@gmail.com>
R topics documented:

- aggregateSamples ............................................ 3
- all_expn_data .................................................. 4
- all_genes ....................................................... 4
- assignCellClusterNearestNode ............................... 5
- batchIDs ......................................................... 6
- bindSeuratObj .................................................. 6
- celltypeFreqs ................................................... 7
- clusterIndividualSamples ................................. 7
- compareSamples ............................................... 8
- createDataObj ................................................ 9
- drawColnames45 .............................................. 10
- embedCells ..................................................... 11
- gaussianffLocal .............................................. 12
- GDM .............................................................. 13
- generateGDM .................................................. 13
- getArithmeticCentroids .................................... 14
- getCellYield ................................................... 15
- getSampleCelltypeFreqs .................................... 16
- getSampleHistsByCluster .................................. 17
- getSampleSizes ............................................... 18
- groupSamples ............................................... 18
- heatmap_genes ............................................... 19
- identifyCentroids .......................................... 20
- monocleInfo .................................................. 21
- orderCellsMonocle ......................................... 21
- phateInfo ....................................................... 22
- Phemd .......................................................... 23
- Phemd-methods ............................................... 23
- plotCellYield ................................................. 26
- plotEmbeddings ................................................ 27
- plotGroupedSamplesDmap .................................. 28
- plotHeatmaps ................................................ 30
- plotSummaryHistograms .................................... 31
- pooledCells ................................................... 32
- printClusterAssignments .................................. 32
- rawExpn ....................................................... 33
- removeTinySamples ......................................... 34
- retrieveRefClusters ....................................... 34
- selected_genes ............................................... 35
- selectFeatures ............................................... 36
- selectMarkers ............................................... 37
- seuratInfo ..................................................... 37
- sNames ........................................................ 38
- snames_data .................................................. 38
- subsampledBool ............................................. 39
- subsampledIdx .............................................. 39
aggregateSamples

Description

Takes initial Phemd object and returns object with additional data frame in slot @data_aggregate containing cells aggregated from all samples (to be used for further analyses e.g. Monocle 2 trajectory building / pseudotime mapping / cell clustering)

Usage

aggregateSamples(obj, max_cells = 12000)

Arguments

obj 'Phemd' object containing raw expression data and associated metadata
max_cells Maximum number of cells across all samples to be included in final matrix on which Monocle 2 will be run

Details

Subsamples cells as necessary based on max_cells. If subsampling is performed, an equal number of cells are subsampled from each sample

Value

Same as input 'Phemd' object with additional slot 'data_aggregate' containing aggregated expression data (num_markers x num_cells)

Examples

my_phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
my_phemdObj_lg <- removeTinySamples(my_phemdObj, 10)
my_phemdObj_lg <- aggregateSamples(my_phemdObj_lg, max_cells=1000)
all_expn_data  
*Single-cell RNA-seq expression data for melanoma samples*

**Description**

This dataset contains normalized single-cell RNA-seq expression data for 19 melanoma samples (immune cells).

**Usage**

```r
data(all_expn_data)
```

**Format**

A list of length 19 with each element representing a distinct sample. Each list element (sample) is a matrix with dimension num_genes x num_cells.

**Source**


**References**


---

all_genes  
*All genes included in (subsampled) melanoma single-cell RNA-seq expression data*

**Description**

This object contains 100 genes measured in melanoma single-cell RNA-seq expression data.

**Usage**

```r
data(all_genes)
```

**Format**

Vector of length 100 representing row names of each matrix in melanoma expression dataset

**Source**

assignCellClusterNearestNode

Assign cells to a reference cell subtype

Description
Assigns each cell in cur_cells to a cluster based on nearest cell in Monocle 2 tree

Usage
assignCellClusterNearestNode(
cur_cells,
ref_cells,
ref_cell_labels,
cell_model = c("monocle2", "seurat", "phate")
)

Arguments
cur_cells Matrix of cells to be assigned to clusters (Dim: num_cells x num_markers)
ref_cells Matrix of cells used to build reference Monocle 2 tree (Dim: num_monocle_cells x num_markers)
ref_cell_labels Vector of length num_monocle_cells containing Monocle 2 cell branch assignments
cell_model Either "monocle2", "seurat", or "phate" depending on method used to model cell state space

Details
Private method (not exported in namespace). Uses RANN package for fast knn search

Value
Vector of length num_cells representing cluster assignments for each cell in cur_cells

Examples
## Not run:
cur_cells_cluster_labels <- assignCellClusterNearestNode(cur_cells_expn_data,
clustered_cells_expn_data, clustered_cells_cluster_labels, cell_model='monocle2')

## End(Not run)
### batchIDs

**Description**

Accessor function for batch ID for each sample

**Usage**

```r
batchIDs(obj)
```

**Arguments**

- **obj**  
  Phemd object

**Value**

Vector of length num_samples representing the experiment (batch) in which the sample was profiled

**Examples**

```r
phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
batch_metadata <- batchIDs(phemdObj)
```

---

### bindSeuratObj

**Description**

Attach 'Seurat' object to 'Phemd' object

**Usage**

```r
bindSeuratObj(phemd_obj, seurat_obj, batch.colname = "plt")
```

**Arguments**

- **phemd_obj**  
  Phemd object initialized using createDataObj
- **seurat_obj**  
  S4 'seurat' object containing batch-normalized reference cell data
- **batch.colname**  
  Name of column in Seurat object that denotes batch ID

**Value**

'Phemd' object containing with attached Seurat object
Examples

my_phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
my_seuratObj <- Seurat::CreateSeuratObject(counts = t(all_expn_data[[1]]), project = "A")
my_seuratObj <- Seurat::FindVariableFeatures(object = my_seuratObj)
my_seuratObj <- Seurat::ScaleData(object = my_seuratObj, do.scale=FALSE, do.center=FALSE)
my_seuratObj <- Seurat::RunPCA(object = my_seuratObj, pc.genes = colnames(all_expn_data[[1]]), do.print = FALSE)
my_seuratObj <- Seurat::FindNeighbors(my_seuratObj, reduction = "pca", dims.use = 1:10)
my_seuratObj <- Seurat::FindClusters(my_seuratObj, resolution = 0.6, print.output = 0, save.SNN = TRUE)
my_phemdObj <- bindSeuratObj(my_phemdObj, my_seuratObj)

celltypeFreqs

Accessor function for cell subtype distribution for each sample

Description

Accessor function for cell subtype distribution for each sample

Usage

celltypeFreqs(obj)

Arguments

obj Phemd object

Value

Matrix representing cell subtype relative frequencies for each sample (num_samples x num_genes)

Examples

phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
celltype_weights <- celltypeFreqs(phemdObj)

clusterIndividualSamples

Computes cell subtype abundances for each sample

Description

Takes as input a Phemd object with all single-cell expression data of all single-cell samples in @data slot and cell-state embedding generated by embedCells. Returns updated object with cell subtype frequencies of each sample that may be retrieved by the 'celltypeFreqs' accessor function.
Usage

```r
classIndividualSamples(
  obj,
  verbose = FALSE,
  cell_model = c("monocle2", "seurat", "phate")
)
```

Arguments

- `obj`: 'Phemd' object containing single-cell expression data of all samples in @data slot and cell-state embedding object generated and stored using the embedCells function.
- `verbose`: Boolean that determines whether progress (sequential processing of samples) should be printed. FALSE by default.
- `cell_model`: Either "monocle2", "seurat", or "phate" depending on method used to model cell state space.

Details

- `embedCells` (and `orderCellsMonocle` if using the Monocle2 embedding technique) needs to be called before calling this function.

Value

- 'Phemd' object with cell subtype frequencies of each sample that can be retrieved using the `cell-typeFreqs` accessor function.

Examples

```r
my_phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
my_phemdObj_lg <- removeTinySamples(my_phemdObj, 10)
my_phemdObj_lg <- aggregateSamples(my_phemdObj_lg, max_cells=1000)
my_phemdObj_monocle <- embedCells(my_phemdObj_lg, data_model = 'gaussianff', sigma=0.02, maxIter=2)
my_phemdObj_monocle <- orderCellsMonocle(my_phemdObj_monocle)
my_phemdObj_final <- clusterIndividualSamples(my_phemdObj_monocle)
```

---

**compareSamples**

*Computes EMD distance matrix representing pairwise dissimilarity between samples*

Description

Takes as input a Phemd object with cell subtype relative frequencies for each sample in @data_cluster_weights slot and ground distance matrix (representing cell subtype pairwise dissimilarity) in @emd_dist_mat slot. Returns distance matrix representing pairwise dissimilarity between samples.
createDataObj

Usage

compareSamples(obj)

Arguments

obj 'Phemd' object containing cell subtype relative frequencies for each sample in @data_cluster_weights slot and ground distance matrix (representing cell subtype dissimilarity) in @emd_dist_mat slot

Details

Requires 'transport' and 'pracma' packages

Value

Distance matrix of dimension num_samples x num_samples representing pairwise dissimilarity between samples

Examples

my_phemdObj <- createDataObj(all.expn_data, all.genes, as.character(snames_data))
my_phemdObj_lg <- removeTinySamples(my_phemdObj, 10)
my_phemdObj_lg <- aggregateSamples(my_phemdObj_lg, max_cells=1000)
my_phemdObj_monocle <- embedCells(my_phemdObj_lg, data_model = 'gaussianff', sigma=0.02, maxIter=2)
my_phemdObj_monocle <- orderCellsMonocle(my_phemdObj_monocle)
my_phemdObj_final <- clusterIndividualSamples(my_phemdObj_monocle)
my_phemdObj_final <- generateGDM(my_phemdObj_final)
my_EMD_mat <- compareSamples(my_phemdObj_final)

createDataObj

Create 'Phemd' object

Description

Wrapper function to create 'Phemd' object containing raw expression data and metadata

Usage

createDataObj(data, markers, snames, datatype = "list", valtype = "counts")

Arguments

data List of length num_samples containing expression data; each element is of size num_cells x num_markers. Alternately a SingleCellExperiment object.
markers Vector containing marker names (i.e. column names of all_data)
snames Vector containing sample names (i.e. names of samples contained in all_data)
datatype Either "list" or "sce" (SingleCellExperiment with genes x cells)
valtype Type of assay data (i.e. "counts", "normcounts", "logcounts", "tpm", "cpm") if datatype is "sce"

Details
Note that each element in list can have different number of rows (i.e. number of cells in each sample can vary).

Value
'Phemd' object containing raw multi-sample expression data and associated metadata

Examples
my_phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))

---

drawColnames45 Rotates heatmap marker labels 45 degrees

Description
Overwrites default draw_colnames in the pheatmap package

Usage
drawColnames45(coln, gaps, ...)

Arguments
coln Column names
gaps Spacing of labels
... Additional parameters to be passed to gpar

Details
To be used with pheatmap plotting function; not to be called directly. Thanks to Josh O'Brien at http://stackoverflow.com/questions/15505607

Value
Formatted marker labels in heatmap

Examples
#Not to be called directly
**Description**

Takes as input a Phemd object with aggregated data and returns updated object containing cell-state embedding

**Usage**

```r
eembedCells(
  obj,
  cell_model = c("monocle2", "seurat", "phate"),
  data_model = "negbinomial_sz",
  phate_ncluster = 8,
  phate_cluster_seed = NULL,
  ...
)
```

**Arguments**

- `obj` 'Phemd' object containing aggregated data
- `cell_model` Method to use to generate cell-state embedding. Currently supports "phate" and "monocle2". If using the Seurat to model the cell-state space, please identify cell subtypes as outlined in the Seurat software package and then use the bindSeuratObj function.
- `data_model` Only relevant if cell_model = "monocle2". One of the following: 'negbinomial', 'negbinomial_sz', 'tobit', 'uninormal', 'gaussianff'. See "Family Function" table at the following link for more details on selecting the proper one.
- `phate_ncluster` Only relevant if cell_model = "phate". Number of cell state clusters to return when using PHATE
- `phate_cluster_seed` Only relevant if cell_model = "phate". Seed to use when performing cell state clustering (optional)
- `...` Additional parameters to be passed to reduceDimension function for Monocle or phate function for PHATE

**Details**

`aggregateSamples` needs to be called before running this function.

**Value**

Same as input 'Phemd' object containing additional cell-state embedding object
Examples

my_phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
my_phemdObj_lg <- removeTinySamples(my_phemdObj_lg, 10)
my_phemdObj_lg <- aggregateSamples(my_phemdObj_lg, max_cells=1000)
my_phemdObj_lg <- embedCells(my_phemdObj_lg, cell_model='monocle2', data_model='gaussianff', sigma=0.02, maxIter=20)

---

**gaussianffLocal**  
Models expression data using generalized linear model with Gaussian error

**Description**

Useful for modeling pre-normalized single-cell expression data.

**Usage**

gaussianffLocal(dispersion = 0, parallel = FALSE, zero = NULL)

**Arguments**

- **dispersion**  
  Dispersion parameter. If 0, then estimate as described in VGAM 1.0-5 documentation.

- **parallel**  
  A logical or formula. If a formula, the response of the formula should be a logical and the terms of the formula indicates whether or not those terms are parallel.

- **zero**  
  An integer-valued vector specifying which linear/additive predictors are modelled as intercepts only. The values must be from the set 1...M where M is the number of columns of the matrix response.

**Details**

Private method (not to be called by user directly). Requires VGAM package. Obtained from VGAM v1.0-5 (https://www.rdocumentation.org/packages/VGAM/versions/1.0-5/topics/gaussianff)

**Value**

Generalized linear model with Gaussian error
### Description

Accessor function for EMD ground distance matrix

### Usage

```r
GDM(obj)
```

### Arguments

- **obj**
  
  A Phemd object

### Value

Square matrix representing pairwise distances between cell subtypes

### Examples

```r
phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
gdm <- GDM(phemdObj)
```

---

### Description

Computes ground distance matrix based on cell embedding

### Usage

```r
generateGDM(
  obj,
  cell_model = c("monocle2", "seurat", "phate"),
  expn_type = "reduced",
  ndim = 8
)
```
getArithmeticCentroids

**Argument**

- `obj`: 'Phemd' object containing cell-state embedding object
- `cell_model`: Method by which cell state was modeled (either "monocle2", "seurat", or "phate")
- `expn_type`: Data type to use to determine cell-type dissimilarities
- `ndim`: Number of embedding dimensions to be used for computing cell-type dissimilarity (optional)

**Details**

`embedCells` and `orderCellsMonocle` need to be called before calling this function. Requires 'igraph' package

**Value**

Phemd object with ground distance matrix (to be used in EMD computation) in @data_cluster_weights slot

**Examples**

```r
my_phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
my_phemdObj_lg <- removeTinySamples(my_phemdObj, 10)
my_phemdObj_lg <- aggregateSamples(my_phemdObj_lg, max_cells=1000)
my_phemdObj_monocle <- embedCells(my_phemdObj_lg, data_model = 'gaussianff', sigma=0.02, maxIter=2)
my_phemdObj_monocle <- orderCellsMonocle(my_phemdObj_monocle)
my_phemdObj_final <- clusterIndividualSamples(my_phemdObj_monocle)
my_phemdObj_final <- generateGDM(my_phemdObj_final)
```

**Description**

Takes initial list and returns a matrix with row $i$ representing the arithmetic centroid of cluster $i$

**Usage**

```r
getArithmeticCentroids(ref_clusters)
```

**Arguments**

- `ref_clusters`: list containing each cluster of interest (each list element is a matrix of dimension num_cells x num_markers)

**Details**

Private method (not exported in namespace)
getCellYield

Value
Matrix of dimension num_cluster x num_markers; row $i$ representing the arithmetic centroid of cluster $i$

Examples

```r
## Not run:
cluster_centroids <- getArithmeticCentroids(ref_clusters)

## End(Not run)
```

---

**getCellYield**

*Gets cell yield of each sample as a table*

**Description**

 Gets cell yield (number of viable cells) of each single-cell sample in decreasing order

**Usage**

```r
cellYield(myobj, cluster_assignments = NULL)
```

**Arguments**

- `myobj`: phemdObj object containing expression data for each sample in 'data' slot
- `cluster_assignments`: vector of cluster assignments to be included as additional column in output table (optional)

**Value**

Data frame representing cell yield of each sample

**Examples**

```r
my_phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
my_phemdObj_lg <- removeTinySamples(my_phemdObj, 10)
my_phemdObj_lg <- aggregateSamples(my_phemdObj_lg, max_cells=1000)
my_phemdObj_monocle <- embedCells(my_phemdObj_lg, data_model = 'gaussianff', sigma=0.02, maxIter=2)
my_phemdObj_monocle <- orderCellsMonocle(my_phemdObj_monocle)
my_phemdObj_final <- clusterIndividualSamples(my_phemdObj_monocle)
my_phemdObj_final <- generateGDM(my_phemdObj_final)
my_EMD_mat <- compareSamples(my_phemdObj_final)
cluster_assignments <- groupSamples(my_EMD_mat, distfun = 'hclust', ncluster=4)
geeYield(my_phemdObj_final, cluster_assignments)
```
getSampleCelltypeFreqs

*Returns cell subtype distribution for each sample as a table*

**Description**

Returns cell subtype distribution for each single-cell sample along with (optional) final inhibitor cluster assignment.

**Usage**

```r
getSampleCelltypeFreqs(myobj, cluster_assignments = NULL)
```

**Arguments**

- **myobj**: phemdObj object containing expression data for each sample in 'data' slot
- **cluster_assignments**: Vector of cluster assignments to be included as additional column in output table (optional)

**Value**

Data frame representing relative frequencies of each cell subtype along with (optional) final inhibitor cluster assignment for each single-cell sample.

**Examples**

```r
my_phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
my_phemdObj_lg <- removeTinySamples(my_phemdObj, 10)
my_phemdObj_lg <- aggregateSamples(my_phemdObj_lg, max_cells=1000)
my_phemdObj_monocle <- embedCells(my_phemdObj_lg, data_model = 'gaussianff', sigma=0.02, maxIter=2)
my_phemdObj_monocle <- orderCellsMonocle(my_phemdObj_monocle)
my_phemdObj_final <- clusterIndividualSamples(my_phemdObj_monocle)
my_phemdObj_final <- generateGDM(my_phemdObj_final)
my_EMD_mat <- compareSamples(my_phemdObj_final)
cluster_assignments <- groupSamples(my_EMD_mat, distfun = 'hclust', ncluster=4)
geSampleCelltypeFreqs(my_phemdObj_final, cluster_assignments)
```
getSampleHistsByCluster

*Gets cell subtype frequency histograms for each sample by cluster ID*

**Description**

Gets relative frequency ("weights") of cell subtypes ("bins" or "signatures") in each single-cell sample.

**Usage**

```r
getSampleHistsByCluster(
  myobj,
  cluster_assignments,
  cell_model = c("monocle2", "seurat")
)
```

**Arguments**

- `myobj`: phemdObj object containing cell subtype relative frequency in @data_cluster_weights slot.
- `cluster_assignments`: Vector containing group assignments for each sample in myobj.
- `cell_model`: Method by which cell state was modeled (either "monocle2" or "seurat").

**Details**

`groupSamples` must be called before calling this function. Saves plots in directory called "individual_inhibs".

**Value**

List of lists, with outer list representing sample cluster ID and inner list representing cell subtype frequencies of given sample.

**Examples**

```r
my_phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
my_phemdObj_lg <- removeTinySamples(my_phemdObj, 10)
my_phemdObj_lg <- aggregateSamples(my_phemdObj_lg, max_cells=1000)
my_phemdObj_monocle <- embedCells(my_phemdObj_lg, data_model = 'gaussianff', sigma=0.02, maxIter=2)
my_phemdObj_monocle <- orderCellsMonocle(my_phemdObj_monocle)
my_phemdObj_final <- clusterIndividualSamples(my_phemdObj_monocle)
my_phemdObj_final <- generateGDM(my_phemdObj_final)
my_EM Mat <- compareSamples(my_phemdObj_final)
cluster_assignments <- groupSamples(my_EM Mat, distfun = 'hclust', ncluster=4)
weights_by_cluster <- getSampleHistsByCluster(my_phemdObj_final, cluster_assignments)
```
getSampleSizes  Retrieve single-cell sample sizes

Description
Takes initial list of single-cell samples and returns vector containing number of cells in each sample.

Usage
getSampleSizes(data_list)

Arguments
data_list List of length num_samples (each element has dimension num_cells x num_markers)

Details
Private method (not exported in namespace)

Value
Vector of length num_samples representing number of cells in each sample

Examples
```r
## Not run:
sample_sizes <- getSampleSizes(all_expn_data)
## End(Not run)
```

---

groupSamples  Performs community detection on sample-sample distance matrix to identify groups of similar samples

Description
Takes sample-sample distance matrix as input and returns group assignments for each sample

Usage
groupSamples(
  distmat,
  distfun = "hclust",
  ncluster = NULL,
  method = "complete",
  ...
)
```
**Arguments**

- **distmat**
  A distance matrix of dimension num_samples x num_samples representing pairwise dissimilarity between samples
- **distfun**
  Method of partitioning network of samples (currently either 'hclust' or 'pam')
- **ncluster**
  Optional parameter specifying total number of sample groups
- **method**
  Optional parameter for hierarchical clustering (see "hclust" documentation)
- **...**
  Optional additional parameters to be passed to diffusionKmeans method

**Details**

By default, uses 'kgs' (Kelley-Gardner-Sutcliffe) method for determining optimal number of groups. Alternatively, can take user-specified number of groups). Requires 'cluster' and 'maptree' packages.

**Value**

Vector containing group assignments for each sample (same order as row-order of distmat) based on user-specified partitioning method (e.g. hierarchical clustering)

**Examples**

```r
my_phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
my_phemdObj_lg <- removeTinySamples(my_phemdObj, 10)
my_phemdObj_lg <- aggregateSamples(my_phemdObj_lg, max_cells=1000)
my_phemdObj_monocle <- embedCells(my_phemdObj_lg, cell_model = 'monocle2', data_model = 'gaussianff', sigma=0.02, 
my_phemdObj_monocle <- orderCellsMonocle(my_phemdObj_monocle)
my_phemdObj_final <- clusterIndividualSamples(my_phemdObj_monocle)
my_phemdObj_final <- generateGDM(my_phemdObj_final)
my_EMD_mat <- compareSamples(my_phemdObj_final)
cluster_assignments <- groupSamples(my_EMD_mat, distfun = 'hclust', ncluster=4)
```

**Description**

This object contains genes to be used when plotting heatmap for melanoma single-cell RNA-seq expression data.

**Usage**

```r
data(heatmap_genes)
```

**Format**

Vector of length 42 representing selected genes for plotting heatmap.
identifyCentroids

Source


References


identifyCentroids | Identify cluster centroids (cell names)

Description

Takes initial list and returns list of cell names representing centroid of cluster

Usage

identifyCentroids(ref_clusters)

Arguments

ref_clusters | list containing each cluster of interest (each list element is a matrix of dimension num_cells x num_markers)

Details

Private method (not exported in namespace)

Value

List of names; element i represents the name of the cell in cluster i that is closest to the centroid (arithmetic mean) of cluster i

Examples

```r
## Not run:
centroid_names <- identifyCentroids(ref_clusters)

## End(Not run)
```
**monocleInfo**

*Accessor function for stored Monocle object*

**Description**

Accessor function for stored Monocle object

**Usage**

monocleInfo(obj)

**Arguments**

- `obj` A Phemd object.

**Value**

An object of class ‘CellDataSet’ (from Monocle)

**Examples**

```r
phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
monocle_obj <- monocleInfo(phemdObj)
```

**orderCellsMonocle**

*Compute Monocle2 cell state and pseudotime assignments*

**Description**

Takes as input a Phemd object with Monocle2 object and returns updated object with Monocle2 object containing cell state and pseudotime assignments

**Usage**

orderCellsMonocle(obj, ...)

**Arguments**

- `obj` 'Phemd' object containing Monocle2 object initialized using embedCells
- `...` Additional parameters to be passed into `orderCells` function

**Details**

Wrapper function for `orderCells` in Monocle2 package. `embedCells` needs to be called before calling this function.
Value

Same as input 'Phemd' object with updated cell-state embedding object containing cell state assignments

Examples

```r
demo <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
demo_lg <- removeTinySamples(demo, 10)
demo_lg <- aggregateSamples(demo_lg, max_cells=1000)
demo_monocle <- embedCells(demo_lg, cell_model='monocle2', data_model='gaussianff', sigma=0.02, maxIter=20)
demo_monocle <- orderCellsMonocle(demo_monocle)
```

---

**phateInfo**

Accessor function for stored phate object

**Description**

Accessor function for stored phate object

**Usage**

```r
phateInfo(obj)
```

**Arguments**

- `obj` A Phemd object.

**Value**

An object of class 'phate' (from phateR)

**Examples**

```r
phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
phateObj <- phateInfo(phemdObj)
```
Phemd class

Description

The main PhEMD class to store single-cell expression data.

Fields

- **data**: List of matrices, each of which represents a single-cell sample (num_cells x num_genes)
- **markers**: Column names (e.g. genes) for each element (i.e. data matrix) in "data"
- **snames**: Sample ID for each element in "data"
- **data_aggregate**: Numeric matrix representing expression data for cells from all experimental conditions (rows = markers, cols = cells)
- **data_subsample_idx**: List of vectors each representing the indices of elements in "data" that were subsampled and combined to form "data_aggregate"
- **subsampled_bool**: Boolean represent whether or not subsampling was performed in the data aggregation process
- **monocle_obj**: Data object of type "CellDataSet" that is the core Monocle data structure
- **data_cluster_weights**: Matrix representing cell subtype relative frequencies for each sample (num_samples x num_genes)
- **emd_dist_mat**: Matrix representing pairwise distances between each pair of cell subtypes
- **seurat_obj**: Object of class "Seurat" that is the core Seurat data structure
- **phate_obj**: Object of class "phate" that is the core PHATE data structure
- **experiment_ids**: Vector of length num_samples representing the experiment (batch) in which the sample was profiled

Phemd-methods

**Setter function for protein / gene markers**

Description

Setter function for protein / gene markers
Setter function for stored expression data
Setter function for single-cell expression data aggregated from multiple samples
Setter function for indices of cells subsampled from each sample during aggregation
Setter function for boolean denoting whether cells were subsampled from each sample during aggregation
Setter function for Monocle2 CellDataSet object for experiment
Setter function for Seurat object for experiment
Setter function for `phate` object for experiment
Setter function for cell subtype frequencies of each single-cell sample
Setter function for batch IDs of each single-cell sample
Setter function for EMD ground distance matrix

**Usage**

```r
selectMarkers(obj) <- value
## S4 replacement method for signature 'Phemd'
selectMarkers(obj) <- value
rawExpn(obj) <- value
## S4 replacement method for signature 'Phemd'
rawExpn(obj) <- value
pooledCells(obj) <- value
## S4 replacement method for signature 'Phemd'
pooledCells(obj) <- value
subsampledIdx(obj) <- value
## S4 replacement method for signature 'Phemd'
subsampledIdx(obj) <- value
subsampledBool(obj) <- value
## S4 replacement method for signature 'Phemd'
subsampledBool(obj) <- value
monocleInfo(obj) <- value
## S4 replacement method for signature 'Phemd'
monocleInfo(obj) <- value
seuratInfo(obj) <- value
## S4 replacement method for signature 'Phemd'
seuratInfo(obj) <- value
phateInfo(obj) <- value
## S4 replacement method for signature 'Phemd'
phateInfo(obj) <- value
celltypeFreqs(obj) <- value
```
## S4 replacement method for signature 'Phemd'

celltypeFreqs(obj) <- value

batchIDs(obj) <- value

## S4 replacement method for signature 'Phemd'

batchIDs(obj) <- value

GDM(obj) <- value

## S4 replacement method for signature 'Phemd'

GDM(obj) <- value

### Arguments

- **obj**  
  A Phemd object

- **value**  
  Assignment object

### Value

- Updated Phemd object
- Updated Phemd object
- Updated Phemd object
- Updated Phemd object
- Updated Phemd object
- Updated Phemd object containing Seurat object
- Updated Phemd object containing phate object
- Updated Phemd object
- Updated Phemd object
- Updated Phemd object

### Examples

```r
phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
new_genes <- all_genes
new_genes[1] <- 'IL2R'
selectMarkers(phemdObj) <- new_genes

phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
new_expn_data <- all_expn_data
new_expn_data <- lapply(new_expn_data, function(x) {log2(x+1)})
rawExpn(phemdObj) <- new_expn_data

phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
aggregated_data <- t(do.call(rbind,all_expn_data))
pooledCells(phemdObj) <- aggregated_data
```
phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
subsampledIdxList <- rep(list(1:10), length(all_expn_data)) # subsampled cells 1-10 from each sample
subsampledIdx(phemdObj) <- subsampledIdxList

phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
subsampledBool(phemdObj) <- TRUE

phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
mydata <- pooledCells(phemdObj)
myCellDataSet <- newCellDataSet(mydata, phenoData=NULL, expressionFamily=VGAM::negbinomial.size())
monocleInfo(phemdObj) <- myCellDataSet

phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
my_seuratObj <- Seurat::CreateSeuratObject(counts = t(all_expn_data[[1]]), project = "A")
seuratInfo(phemdObj) <- my_seuratObj

phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
# my_phateObj <- phateR::phate(all_expn_data[[1]])
phateInfo(phemdObj) <- list()

phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
myCellTypeFreqs <- matrix(rexp(length(all_expn_data)*10, rate=.1), ncol=10)
myCellTypeFreqs <- apply(myCellTypeFreqs, 1, function(x) {x / sum(x)})
celltypeFreqs(phemdObj) <- myCellTypeFreqs

phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
my_seuratObj <- Seurat::CreateSeuratObject(counts = t(all_expn_data[[1]]), project = "A")
seuratInfo(phemdObj) <- my_seuratObj
batchIDs(phemdObj) <- rep('A', length(all_expn_data))

phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
cluster_locs <- 1:10
myGDM <- as.matrix(dist(cluster_locs))
GDM(phemdObj) <- myGDM

---

plotCellYield  
*Plot cell yield of each sample as bar plot*

**Description**

Plots cell yield (number of viable cells) of each single-cell sample in decreasing order as horizontal bar plot

**Usage**

plotCellYield(myobj, labels = NULL, cmap = NULL, font_sz = 0.6, w = 8, h = 9.5)
plotEmbeddings

Arguments

myobj Phmed object containing expression data for each sample in 'data' slot
labels Vector containing group labels for samples (optional). If not provided, bars will be of uniform color (blue)
cmap Vector containing colors by which histogram bars should be colored (optional)
font_sz Scaling factor for font size of sample names in barplot
w Width of plot in inches
h Height of plot in inches

Value

None

Examples

my_phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
my_phemdObj_lg <- removeTinySamples(my_phemdObj, 10)
my_phemdObj_lg <- aggregateSamples(my_phemdObj_lg, max_cells=1000)
my_phemdObj_monocle <- embedCells(my_phemdObj_lg, data_model = 'gaussianff', sigma=0.02, maxIter=2)
my_phemdObj_monocle <- orderCellsMonocle(my_phemdObj_monocle)
my_phemdObj_final <- clusterIndividualSamples(my_phemdObj_monocle)
my_phemdObj_final <- generateGDM(my_phemdObj_final)
my_EMD_mat <- compareSamples(my_phemdObj_final)
cluster_assignments <- groupSamples(my_EMD_mat, distfun = 'hclust', ncluster=4)
plotCellYield(my_phemdObj_final, labels=cluster_assignments, font_sz = 0.8)

Description

Plots Monocle2 cell embedding plots

Usage

plotEmbeddings(
  obj,
  cell_model = c("monocle2", "seurat", "phate"),
  cmap = NULL,
  w = 4,
  h = 5,
  pt_sz = 1,
  ndims = NULL
)
plotGroupedSamplesDmap

**Arguments**

- **obj**: 'Phemd' object containing Monocle 2 object
- **cell_model**: Method by which cell state was modeled (either "monocle2", "seurat", or "phate")
- **cmap**: User-specified colormap to use to color cell state embedding (optional)
- **w**: Width of plot in inches
- **h**: Height of plot in inches
- **pt_sz**: Scalar factor for point size
- **ndims**: Number of dimensions to use for dimensionality reduction in case it hasn’t been performed yet (only relevant when using Seurat data as input)

**Details**

embedCells and orderCellsMonocle need to be called before calling this function. Required additional packages: 'RColorBrewer', 'cowplot'

**Value**

Colormap (vector of colors) used to color Monocle2 cell state embedding

**Examples**

```r
my_phemdObj <- createDataObj(all_exprn_data, all_genes, as.character(snames_data))
my_phemdObj_lg <- removeTinySamples(my_phemdObj, 10)
my_phemdObj_lg <- aggregateSamples(my_phemdObj_lg, max_cells=1000)
my_phemdObj_monocle <- embedCells(my_phemdObj_lg, data_model='gaussianff', sigma=0.02, maxIter=2)
my_phemdObj_monocle <- orderCellsMonocle(my_phemdObj_monocle)
cmap <- plotEmbeddings(my_phemdObj_monocle)
```

plotGroupedSamplesDmap

*Plot diffusion map embedding of samples based on distance matrix*

**Description**

Visualizes diffusion map for network of samples based on square distance matrix (sample-sample pairwise dissimilarity)

**Usage**

```r
plotGroupedSamplesDmap(
  my_distmat,
  cluster_assignments = NULL,
  pt_sz = 1,
  n_dim = 3,
  pt_label = NULL,
  cmap = NULL,
)```

Arguments

- **my_distmat**: phemdObj object containing sample names in @snames slot
- **cluster_assignments**: Vector containing group assignments for each sample
- **pt_sz**: Size of points representing samples in plot (scaling factor)
- **n_dim**: Number of dimensions for embedding (either 2 or 3)
- **pt_label**: Vector of sample names corresponding to each point (same order as samples in my_distmat and cluster_assignments)
- **cmap**: Vector containing colors by which points should be colored (corresponding to cluster_assignments)
- **w**: Width of plot in inches
- **h**: Height of plot in inches
- **scale.y**: Scaling factor for diffusion map y-axis
- **angle**: Rotation factor for diffusion map plot
- **autosave**: Boolean denoting whether or not to save output diffusion map
- **...**: Additional parameters to be passed to DiffusionMap function

Details

Requires 'destiny' package

Value

DiffusionMap object containing biological sample embedding and associated metadata

Examples

```r
my_phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
my_phemdObj_lg <- removeTinySamples(my_phemdObj, 10)
my_phemdObj_lg <- aggregateSamples(my_phemdObj_lg, max_cells=1000)
my_phemdObj_monocle <- embedCells(my_phemdObj_lg, data_model = 'gaussianff', sigma=0.02, maxIter=2)
my_phemdObj_monocle <- orderCellsMonocle(my_phemdObj_monocle)
my_phemdObj_final <- clusterIndividualSamples(my_phemdObj_monocle)
my_phemdObj_final <- generateGDM(my_phemdObj_final)
my_EMD_mat <- compareSamples(my_phemdObj_final)
cluster_assignments <- groupSamples(my_EMD_mat, distfun = 'hclust', ncluster=4)
printClusterAssignments(cluster_assignments, my_phemdObj_final, '.', overwrite=TRUE)
dm <- plotGroupedSamplesDmap(my_EMD_mat, cluster_assignments, pt_sz=2)
```
plotHeatmaps

Plot heatmap of cell subtypes

Description
Takes as input a Phemd object containing either a Monocle2, Seurat, or PHATE object (already embedded and clustered) and plots heatmap characterizing cell subtypes.

Usage

plotHeatmaps(
  obj,
  cell_model = c("monocle2", "seurat", "phate"),
  selected_genes = NULL,
  w = 8,
  h = 5,
  ...
)

Arguments

obj 'Phemd' object containing cell-state embedding object

cell_model Method by which cell state was modeled ("monocle2", "seurat", or "phate")

selected_genes Vector containing gene names to include in heatmap (optional)

w Width of plot in inches

h Height of plot in inches

... Additional parameters to be passed on to pheatmap function

Details
embedCells (and orderCellsMonocle if using Monocle2) need to be called before calling this function. Required additional package: 'pheatmap'

Value
Heatmap containing expression values for each cell subtype. If cell_model is 'seurat', then returns a list of heatmaps (1 for each batch) that may be subsequently plotted individually.

Examples

my_phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
my_phemdObj_lg <- removeTinySamples(my_phemdObj, 10)
my_phemdObj_lg <- aggregateSamples(my_phemdObj_lg, max_cells=1000)
my_phemdObj_lg <- selectFeatures(my_phemdObj_lg, selected_genes)
my_phemdObj_monocle <- embedCells(my_phemdObj_lg, data_model = 'gaussianff',
  pseudo_expr=0, sigma=0.02, maxIter=2)
my_phemdObj_monocle <- orderCellsMonocle(my_phemdObj_monocle)
myheatmap <- plotHeatmaps(my_phemdObj_monocle, cell_model='monocle2')

### plotSummaryHistograms

Plots cell subtype frequency histograms summarizing each group of samples

**Description**

Visualizes plots of relative frequency ("weights") of cell subtypes ("bins" or "signatures") summarizing each group of single-cell samples. Each summary histogram is computed by taking the bin-wise mean of all samples in the group

**Usage**

```r
plotSummaryHistograms(
  myobj,
  cluster_assignments,
  cell_model = c("monocle2", "seurat", "phate"),
  cmap = NULL,
  ncol.plot = 4,
  ax.lab.sz = 2.5,
  title.sz = 3
)
```

**Arguments**

- **myobj**: Phemd object containing cell subtype relative frequency in @data_cluster_weights slot
- **cluster_assignments**: Vector containing group assignments for each sample in myobj
- **cell_model**: Method by which cell state was modeled (either "monocle2", "seurat", or "phate")
- **cmap**: Vector containing colors by which histogram bars should be colored (optional)
- **ncol.plot**: Number of columns to use to plot multi-panel histogram plot
- **ax.lab.sz**: Scaling factor for axis labels (default 2.5)
- **title.sz**: Scaling factor for plot title (default 3)

**Details**

`groupSamples` must be called before calling this function. Saves plots in directory called "summary_inhibs"

**Value**

None
**pooledCells**

*Accessor function for aggregated cells used for cell subtype definition*

**Description**

Accessor function for aggregated cells used for cell subtype definition

**Usage**

`pooledCells(obj)`

**Arguments**

- `obj` Phemd object

**Value**

Numeric matrix representing expression data for cells from all experimental conditions (rows = markers, cols = cells)

**Examples**

```r
phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
cells_aggregated <- pooledCells(phemdObj)
```

---

**printClusterAssignments**

*Writes samples to file based on community detection group assignments*

**Description**

Takes vector of cluster assignments and phemdObj containing sample names and writes sample groups to file

**Usage**

`printClusterAssignments(cluster_assignments, obj, dest, overwrite = FALSE)`

**Arguments**

- `cluster_assignments` Vector containing group assignments for each sample
- `obj` phemdObj object containing sample names in @snames slot
- `dest` Path to existing directory where output should be saved
- `overwrite` Boolean representing whether or not to overwrite contents of "dest" with output of printClusterAssignments
Details
Order of samples in obj@snames is assumed to be the same as the order of group assignments in cluster_assignments

Value
None

Examples
my_phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
my_phemdObj_lg <- removeTinySamples(my_phemdObj, 10)
my_phemdObj_lg <- aggregateSamples(my_phemdObj_lg, max_cells=1000)
my_phemdObj_monocle <- embedCells(my_phemdObj_lg, data_model = 'gaussianff', sigma=0.02, maxIter=2)
my_phemdObj_monocle <- orderCellsMonocle(my_phemdObj_monocle)
my_phemdObj_final <- clusterIndividualSamples(my_phemdObj_monocle)
my_phemdObj_final <- generateGDM(my_phemdObj_final)
my_EMN_mat <- compareSamples(my_phemdObj_final)
cluster_assignments <- groupSamples(my_EMN_mat, distfun = 'hclust', ncluster=4)
printClusterAssignments(cluster_assignments, my_phemdObj_final, '.', overwrite=TRUE)
removeTinySamples  Remove samples with too few cells

Description
Removes samples from Phemd that have fewer cells than min_sz

Usage
```
removeTinySamples(obj, min_sz = 20)
```

Arguments
- **obj**  'Phemd' object containing raw expression data and associated metadata
- **min_sz**  Minimum number of cells in each sample to be retained

Details
Note: If used, this function must be called before (and not after) the aggregateSamples function is called

Value
'Phemd' object containing raw multi-sample expression data and associated metadata (same as input minus removed samples)

Examples
```
my_phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
my_phemdObj_lg <- removeTinySamples(my_phemdObj, 10)  #removes samples with fewer than 10 cells
```

retrieveRefClusters  Retrieve reference cell clusters

Description
Takes initial Phemd struct and returns cell clusters as assigned by clustering algorithm (e.g. PHATE or Monocle2)

Usage
```
retrieveRefClusters(
  obj,
  cell_model = c("monocle2", "seurat", "phate"),
  expn_type = "reduced",
  ndim = 10
)
```
selected_genes

Arguments

- obj: Phemd struct containing cell-state embedding object and underlying expression data
- cell_model: String representing data model for cell-state space ("seurat", "monocle2", or "phate")
- expn_type: String representing whether to return raw expression values or coordinates in dimensionality-reduced feature space
- ndim: Number of dimensions in reduced dimensionality space (e.g. PHATE / CCA) to use (only relevant in reduced dimensionality space)

Details

Private method (not exported in namespace)

Value

List of data matrices; each list element is of size num_cells_in_cluster x num_markers and represents a distinct cell cluster

Examples

```r
## Not run:
cluster_expression_data <- retrieveRefClusters(my_phemdObj)
## End(Not run)
```

---

**selected_genes**  
*Genes to be used when performing clustering and trajectory analyses on melanoma single-cell RNA-seq expression data*

Description

This object contains genes to be used when performing clustering and trajectory analyses on melanoma single-cell RNA-seq expression data.

Usage

```r
data(selected_genes)
```

Format

Vector of length 44 representing selected genes for performing computational analyses such as generating cell embeddings and clustering cell subtypes.

Source

selectFeatures

Perform feature selection on aggregated data

Description
Takes as input a Phemd object with aggregated data and returns updated object after performing feature selection on aggregated data.

Usage
selectFeatures(obj, selected_genes)

Arguments
obj 'Phemd' object containing aggregated data
selected_genes Vector containing names of genes to use for downstream analyses

Details
aggregateSamples needs to be called before running this function.

Value
Same as input 'Phemd' object after performing feature-selection based dimensionality reduction on aggregated expression data.

Examples
my_phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
my_phemdObj_lg <- removeTinySamples(my_phemdObj, 10)
my_phemdObj_lg <- aggregateSamples(my_phemdObj_lg, max_cells=1000)
my_phemdObj_lg <- selectFeatures(my_phemdObj_lg, selected_genes=c('TP53', 'EGFR', 'KRAS', 'FOXP3', 'LAG3'))

References
**selectMarkers**

**Accessor function for gene/protein markers measured in experiment**

**Description**

Accessor function for gene/protein markers measured in experiment

**Usage**

```r
selectMarkers(obj)
```

**Arguments**

- `obj` Phemd object

**Value**

Vector representing gene/protein markers corresponding to expression matrices

**Examples**

```r
phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
genes <- selectMarkers(phemdObj)
```

---

**seuratInfo**

**Accessor function for stored Seurat object within Phemd object**

**Description**

Accessor function for stored Seurat object within Phemd object

**Usage**

```r
seuratInfo(obj)
```

**Arguments**

- `obj` A Phemd object.

**Value**

An object of class 'Seurat'

**Examples**

```r
phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
seurat_obj <- seuratInfo(phemdObj)
```
sNames | Accessor function for identifiers of all single-cell samples in experiment

**Description**
Accessor function for identifiers of all single-cell samples in experiment

**Usage**
sNames(obj)

**Arguments**
obj | Phemd object

**Value**
Vector representing sample names corresponding to expression matrices

**Examples**
phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
sampleIDs <- sNames(phemdObj)

**snames_data** | Sample names for melanoma single-cell RNA-seq expression data

**Description**
This object contains sample names corresponding to samples contained in melanoma expression data.

**Usage**
data("snames_data")

**Format**
Vector of length 19 representing sample names corresponding to order of samples in all_expn_data in melanomaData dataset.

**Source**
References

---

subsampledBool

**Accessor function for whether or not cells were subsampled when aggregated for cell subtype analysis**

**Description**
Accessor function for whether or not cells were subsampled when aggregated for cell subtype analysis.

**Usage**
subsampledBool(obj)

**Arguments**

- **obj**
  Phemd object

**Value**
Boolean represent whether or not subsampling was performed in the data aggregation process.

**Examples**
phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
subsampled <- subsampledBool(phemdObj)

---

subsampledIdx

**Accessor function for aggregated cells used for cell subtype definition**

**Description**
Accessor function for aggregated cells used for cell subtype definition.

**Usage**
subsampledIdx(obj)

**Arguments**

- **obj**
  Phemd object
Value

List of vectors each representing the indices of elements in \texttt{rawExpn(obj)} that were subsampled and combined to form "data_aggregate"

Examples

\begin{verbatim}
phemdObj <- createDataObj(all_expn_data, all_genes, as.character(snames_data))
subsampled_idx_list <- subsampledIdx(phemdObj)
\end{verbatim}
Index

* datasets
  all_expn_data, 4
  all_genes, 4
  heatmap_genes, 19
  selected_genes, 35
  snames_data, 38

aggregateSamples, 3
all_expn_data, 4
all_genes, 4
assignCellClusterNearestNode, 5

batchIDs, 6
batchIDs<-(Phemd-methods), 23
batchIDs<-,Phemd-method
(Phemd-methods), 23
bindSeuratObj, 6

celltypeFreqs, 7
celltypeFreqs<-(Phemd-methods), 23
celltypeFreqs<-,Phemd-method
(Phemd-methods), 23
clusterIndividualSamples, 7
compareSamples, 8
createDataObj, 9

drawColnames45, 10
embedCells, 11

gaussianffLocal, 12
GDM, 13
GDM<-(Phemd-methods), 23
GDM<-,Phemd-method(Phemd-methods), 23
generateGDM, 13
getArithmeticCentroids, 14
getCellYield, 15
getSampleCelltypeFreqs, 16
getSampleHistByCluster, 17
getSampleSizes, 18
groupSamples, 18

heatmap_genes, 19
identifyCentroids, 20
monocleInfo, 21
monocleInfo<-(Phemd-methods), 23
monocleInfo<-,Phemd-method
(Phemd-methods), 23

orderCellsMonocle, 21
phateInfo, 22
phateInfo<-(Phemd-methods), 23
phateInfo<-,Phemd-method
(Phemd-methods), 23
Phemd, 23
Phemd,ANY,ANY-method(Phemd-methods), 23
Phemd,character,ANY-method
(Phemd-methods), 23
Phemd-class(Phemd), 23
Phemd-methods, 23
plotCellYield, 26
plotEmbeddings, 27
plotGroupedSamplesDmap, 28
plotHeatmaps, 30
plotSummaryHistograms, 31
pooledCells, 32
pooledCells<-(Phemd-methods), 23
pooledCells<-,Phemd-method
(Phemd-methods), 23
printClusterAssignments, 32

rawExpn, 33
rawExpn<-(Phemd-methods), 23
rawExpn<-,Phemd-method(Phemd-methods),
23
removeTinySamples, 34
retrieveRefClusters, 34

selected_genes, 35
selectFeatures, 36
selectMarkers, 37
selectMarkers<-(Phemd-methods), 23
selectMarkers<-,Phemd-method
   (Phemd-methods), 23
seuratInfo, 37
seuratInfo<-(Phemd-methods), 23
seuratInfo<-,Phemd-method
   (Phemd-methods), 23
sNames, 38
snames_data, 38
subsampledBool, 39
subsampledBool<-(Phemd-methods), 23
subsampledBool<-,Phemd-method
   (Phemd-methods), 23
subsampledIdx, 39
subsampledIdx<-(Phemd-methods), 23
subsampledIdx<-,Phemd-method
   (Phemd-methods), 23