

# Package ‘ctgGEM’

October 14, 2021

**Type** Package

**Title** Generating Tree Hierarchy Visualizations from Gene Expression Data

**Version** 1.4.0

**Author** Mark Block and Carrie Minette

**Maintainer** USD Biomedical Engineering <bicbioeng@gmail.com>

## Description

Cell Tree Generator for Gene Expression Matrices (ctgGEM) streamlines the building of cell-state hierarchies from single-cell gene expression data across multiple existing tools for improved comparability and reproducibility. It supports pseudotemporal ordering algorithms and visualization tools from monocle, cellTree, TSCAN, sincell, and destiny, and provides a unified output format for integration with downstream data analysis workflows and Cytoscape.

**VignetteBuilder** knitr

**License** GPL(>=2)

**Encoding** UTF-8

**LazyData** true

**biocViews** GeneExpression, Visualization, Sequencing, SingleCell, Clustering, RNASeq, ImmunoOncology, DifferentialExpression, MultipleComparison, QualityControl, DataImport

**RoxygenNote** 7.1.0

**Roxygen** list(markdown = TRUE)

**Depends** monocle, SummarizedExperiment,

**Imports** Biobase, BiocGenerics, graphics, grDevices, igraph, methods, utils, sincell, TSCAN, destiny, HSMMSingleCell

**Suggests** BiocStyle, biomaRt, irlba, knitr, VGAM

**Collate** 'ctgGEMset-class.R' 'ctgGEMset-methods.R' 'generate\_tree.R' 'makeDestiny.R' 'makeMonocle.R' 'makeSincell.R' 'makeTSCAN.R' 'plotOriginalTree.R' 'tree2igraph.R'

**git\_url** <https://git.bioconductor.org/packages/ctgGEM>

**git\_branch** RELEASE\_3\_13

**git\_last\_commit** 0e246e8

**git\_last\_commit\_date** 2021-05-19

**Date/Publication** 2021-10-14

## R topics documented:

ctgGEMset . . . . .	2
generate_tree . . . . .	3
plotOriginalTree . . . . .	4
<b>Index</b>	<b>6</b>

---

ctgGEMset	<i>The ctgGEMset class</i>
-----------	----------------------------

---

### Description

The main class used by the **cellTreeGenerator** package to hold single cell gene expression data and generated tree results. ctgGEMset extends the [SummarizedExperiment::SummarizedExperiment](#) class.

### Usage

```
ctgGEMset(exprsData, phenoData = NULL, featureData = NULL)
```

### Arguments

exprsData	expression data matrix for an experiment
phenoData	a data frame containing attributes of individual samples
featureData	a data frame containing attributes of features (genes)

### Details

This class is initialized from a matrix of gene expression values and associated metadata. Methods that operate on ctgGEMset objects comprise the ctgGEM workflow.

### Value

a new ctgGEMset object

**Fields**

- monocleInfo** A character vector of parameters used by `generate_tree(treeType = "monocle")` in the **cellTreeGenerator** workflow
- TSCANinfo** A character vector of the row name of a single gene in `exprsData()` to use for a single gene vs. pseudotime plot for `generate_tree(treeType = "TSCAN")` in the **cellTreeGenerator** workflow
- sincellInfo** A list containing named parameters used by `generate_tree(treeType = "sincell")` in the **cellTreeGenerator** workflow
- treeList** A list containing the simplified igraph representation of the trees generated by the **ctgGEM** workflow
- originalTrees** A list containing the trees generated by the **ctgGEM** workflow in their original formats for re-plotting

**Examples**

```
# load HSMMSingleCell package
library(HSMMSingleCell)

# load the data
data(HSMM_expr_matrix)
data(HSMM_sample_sheet)
data(HSMM_gene_annotation)

# construct a ctgGEMset
dataSet <- ctgGEMset(exprsData = HSMM_expr_matrix,
                    phenoData = HSMM_sample_sheet,
                    featureData = HSMM_gene_annotation)

# load HSMMSingleCell package
library(HSMMSingleCell)

# load the data
data(HSMM_expr_matrix)
data(HSMM_sample_sheet)
data(HSMM_gene_annotation)

# construct a ctgGEMset
dataSet <- ctgGEMset(exprsData = HSMM_expr_matrix,
                    phenoData = HSMM_sample_sheet,
                    featureData = HSMM_gene_annotation)
```

---

generate\_tree

*Function to Generate Cell Trees*


---

**Description**

This function builds a cell hierarchy tree of a chosen supported type with a given data set, contained in a `ctgGEMset` object. Different tree types require data from corresponding slots of the `ctgGEMset` object. See vignette for examples, usage details, and instructions on building a `ctgGEMset` object.

**Usage**

```
generate_tree(dataSet, treeType, outputDir = NULL)
```

**Arguments**

dataSet	the ctgGEMset object for creating the cell tree
treeType	the type of tree generated
outputDir	the directory where output should be saved, defaults to the temporary location returned by tempdir()

**Value**

An updated ctgGEMset object. The generated tree is placed in @treeList[treeType] slot, and can be accessed via treeList(dataSet)\$treeType. The function also creates a directory named "treeType-Output" and writes the plot(s) of the generated tree(s) and its SIF file to that directory.

**Examples**

```
# load HSMMSingleCell package
library(HSMMSingleCell)

# load the data
data(HSMM_expr_matrix)
data(HSMM_sample_sheet)
data(HSMM_gene_annotation)

# construct a ctgGEMset
dataSet <- ctgGEMset(exprsData = HSMM_expr_matrix,
                    phenoData = HSMM_sample_sheet,
                    featureData = HSMM_gene_annotation)

TSCANinfo(dataSet) <- "ENSG00000000003.10"

# choose output directory
od <- getwd()
# run generate_tree()
dataSet <- generate_tree(dataSet = dataSet, treeType = "TSCAN",
                        outputDir = od)
```

---

plotOriginalTree

*Display Original ctgGEM Plots*


---

**Description**

Displays the original plots created by the ctgGEM package and stored in the [originalTrees](#) slot of a ctgGEMset object.

**Usage**

```
plotOriginalTree(dataSet, treeType)
```

**Arguments**

dataSet	a ctgGEMset object
treeType	the type of tree to display. Must be one of names(originalTrees(dataSet))

**Value**

For treeType = "destinyDM" and treeType = "destinyDPT", returns NULL. For all other tree types, a ggplot2::ggplot object.

**Note**

In order to reproduce original oncoNEM trees or destiny plots, the respective package(s) must be installed.

**Examples**

```
# load HSMMSingleCell package
library(HSMMSingleCell)

# load the data for TSCAN and monocle:
data(HSMM_expr_matrix)
data(HSMM_sample_sheet)
data(HSMM_gene_annotation)

# construct a ctgGEMset
dataSet <- ctgGEMset(exprsData = HSMM_expr_matrix,
                    phenoData = HSMM_sample_sheet,
                    featureData = HSMM_gene_annotation)

TSCANinfo(dataSet) <- "ENSG00000000003.10"

# run generate_tree()
dataSet <- generate_tree(dataSet = dataSet, treeType = "TSCAN")

# view names of original trees
names(originalTrees(dataSet))

# plot original trees
plotOriginalTree(dataSet, "TSCANclustering")
plotOriginalTree(dataSet, "TSCANsingleGene")
```

# Index

- \* **cell**
  - generate\_tree, [3](#)
- \* **tree**
  - generate\_tree, [3](#)
  - .ctgGEMset (ctgGEMset), [2](#)
- ctgGEMset, [2](#)
- ctgGEMset-class (ctgGEMset), [2](#)
- generate\_tree, [3](#)
- originalTrees, [4](#)
- plotOriginalTree, [4](#)
- SummarizedExperiment::SummarizedExperiment,  
[2](#)