

# Package ‘ccImpute’

May 7, 2024

**Type** Package

**Title** ccImpute: an accurate and scalable consensus clustering based approach to impute dropout events in the single-cell RNA-seq data (<https://doi.org/10.1186/s12859-022-04814-8>)

**Version** 1.6.0

**Description** Dropout events make the lowly expressed genes indistinguishable from true zero expression and different than the low expression present in cells of the same type. This issue makes any subsequent downstream analysis difficult. ccImpute is an imputation algorithm that uses cell similarity established by consensus clustering to impute the most probable dropout events in the scRNA-seq datasets. ccImpute demonstrated performance which exceeds the performance of existing imputation approaches while introducing the least amount of new noise as measured by clustering performance characteristics on datasets with known cell identities.

**License** GPL-3

**Imports** Rcpp, matrixStats, stats, SIMLR, BiocParallel

**LinkingTo** Rcpp, RcppEigen

**Encoding** UTF-8

**LazyData** FALSE

**BugReports** <https://github.com/khazum/ccImpute/issues>

**RoxygenNote** 7.2.1

**biocViews** SingleCell, PrincipalComponent, DimensionReduction, Clustering, RNASeq, Transcriptomics

**biocType** Software

**Suggests** knitr, rmarkdown, BiocStyle, sessioninfo, scRNAseq, scater, SingleCellExperiment, mclust, testthat (>= 3.0.0)

**VignetteBuilder** knitr

**Config/testthat/edition** 3

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

**git\_branch** RELEASE\_3\_19

**git\_last\_commit** ce4e4c1

**git\_last\_commit\_date** 2024-04-30

**Repository** Bioconductor 3.19

**Date/Publication** 2024-05-07

**Author** Marcin Malec [cre, aut] (<<https://orcid.org/0000-0002-2354-513X>>)

**Maintainer** Marcin Malec <mamalec@iu.edu>

## Contents

|              |          |
|--------------|----------|
| ccImpute     | 2        |
| findDropouts | 4        |
| findNDim     | 4        |
| getConsMtx   | 5        |
| getPConsMtx  | 5        |
| kmAux        | 6        |
| solveDrops   | 6        |
| wCorDist     | 7        |
| <b>Index</b> | <b>8</b> |

---

|          |   |
|----------|---|
| ccImpute | <i>Performs imputation of dropout values in scRNA-seq data using ccImpute algorithm as described in the ccImpute: an accurate and scalable consensus clustering based algorithm to impute dropout events in the single-cell RNA-seq data DOI: <a href="https://doi.org/10.1186/s12859-022-04814-8">https://doi.org/10.1186/s12859-022-04814-8</a></i> |
|----------|---|

---

## Description

Performs imputation of dropout values in scRNA-seq data using ccImpute algorithm as described in the ccImpute: an accurate and scalable consensus clustering based algorithm to impute dropout events in the single-cell RNA-seq data DOI: <https://doi.org/10.1186/s12859-022-04814-8>

## Usage

```
ccImpute(
  logX,
  useRanks = TRUE,
  pcaMin,
  pcaMax,
  k,
  consMin = 0.65,
  kmNStart,
  kmMax = 1000,
  BPPARAM = bpparam()
)
```

**Arguments**

|          |   |
|----------|---|
| logX     | A normalized and log transformed scRNA-seq expression matrix.   |
| useRanks | A Boolean specifying if non-parametric version of weighted Pearson correlation should be used. It's recommended to keep this as TRUE since this performs better as determined experimentally. However, FALSE will also provide decent results with the benefit of faster runtime.   |
| pcaMin   | This is used to establish the number of minimum PCA features used for generating subsets. For small datasets up to 500 cells this equals $\text{pcaMin} \times n$ minimum features, where $n$ is number of cells. For large datasets, this corresponds to the feature count that has proportion of variance less than $\text{pcaMin}$ . Both $\text{pcaMin}$ and $\text{pcaMax}$ must be specified to be considered. It's best to keep this value as default unless a better value was obtained experimentally. |
| pcaMax   | This is used to establish the number of maximum PCA features used for generating subsets. For small datasets up to 500 cells this equals $\text{pcaMax} \times n$ maximum features, where $n$ is number of cells. For large datasets, this corresponds to the feature count that has proportion of variance less than $\text{pcaMax}$ . Both $\text{pcaMin}$ and $\text{pcaMax}$ must be specified to be considered. It's best to keep this value as default unless a better value was obtained experimentally. |
| k        | centers parameter passed to <code>kmeans</code> function. This corresponds to a number of different cell groups in data. This can be estimated in a number of methods. If not provided we take the approach provided in the SIMLR package. ( <a href="https://www.bioconductor.org/packages/release/bioc/html/SIMLR.html">https://www.bioconductor.org/packages/release/bioc/html/SIMLR.html</a> )  |
| consMin  | the low-pass filter threshold for processing consensus matrix. This is to eliminate noise from unlikely clustering assignments. It is recommended to keep this value $>.5$ .  |
| kmNStart | nstart parameter passed to <code>kmeans</code> function. Can be set manually. By default it is 1000 for up to 2000 cells and 50 for more than 2000 cells.   |
| kmMax    | iter.max parameter passed to <code>kmeans</code> . <code>ccImpute</code> is a stochastic method, and setting the <code>rand_seed</code> allows reproducibility.   |
| BPPARAM  | - BiocParallel parameters for parallelization   |

**Value**

A normalized and log transformed scRNA-seq expression matrix with imputed missing values.

**Examples**

```
exp_matrix <- log(abs(matrix(rnorm(1000000), nrow=10000))+1)
ccImpute(exp_matrix, k = 2)
```

---

|              |  |
|--------------|--|
| findDropouts | <i>Establishes which zero values in x are dropout events based on weighted cell voting with weights derived from processed consensus matrix consMtx.</i> |
|--------------|--|

---

**Description**

Establishes which zero values in x are dropout events based on weighted cell voting with weights derived from processed consensus matrix consMtx.

**Usage**

```
findDropouts(x, consMtx)
```

**Arguments**

|         |   |
|---------|---|
| x       | transpose of log normalized expression matrix |
| consMtx | processed consensus matrix                    |

**Value**

list of indices in x that are dropout events

---

|          |  |
|----------|--|
| findNDim | <i>Establish what subsets of loadings from PCA distance measure are used for for measuring cluster instability</i> |
|----------|--|

---

**Description**

Establish what subsets of loadings from PCA distance measure are used for for measuring cluster instability

**Usage**

```
findNDim(n, distPCA, pcaMin, pcaMax)
```

**Arguments**

|         |  |
|---------|--|
| n       | number of samples  |
| distPCA | PCA reduced distance matrix  |
| pcaMin  | This is used to establish the number of minimum PCA features used for generating subsets. For small datasets up to 500 cells this equals $pcaMin * n$ minimum features, where n is number of cells. For large datasets, this corresponds to the feature count that has proportion of variance less than pcaMin. Both pcaMin and pcaMax must be specified to be considered. |

pcaMax This is used to establish the number of maximum PCA features used for generating subsets. For small datasets up to 500 cells this equals  $\text{pcaMax} * n$  maximum features, where  $n$  is number of cells. For large datasets, this corresponds to the feature count that has proportion of variance less than  $\text{pcaMax}$ . Both  $\text{pcaMin}$  and  $\text{pcaMax}$  must be specified to be considered.

**Value**

list of numbers with each number corresponding to the number of loadings to use for clustering.

---

getConsMtx *Computes consensus matrix given cluster labels*

---

**Description**

Computes consensus matrix given cluster labels

**Usage**

```
getConsMtx(dat)
```

**Arguments**

dat a matrix containing clustering solutions in columns

**Value**

consensus matrix

---

getPConsMtx *Get processed consensus matrix.*

---

**Description**

This function gets consensus matrix based on the clustering solutions contained in the `kmResults` input parameter and does the processing to use it for imputation.

**Usage**

```
getPConsMtx(kmResults, consMin)
```

**Arguments**

`kmResults` list of k-means clustering assignments on the PCA loadings sub-datasets.  
`consMin` the low-pass filter threshold value for processed consensus matrix.

**Value**

a processed consensus matrix.

---

|       |  |
|-------|--|
| kmAux | <i>This function performs <a href="#">kmeans</a> clustering of the subdataset corresponding to a given range <i>i</i> of PCA loadings as contained in input parameter.</i> |
|-------|--|

---

### Description

This function performs [kmeans](#) clustering of the subdataset corresponding to a given range *i* of PCA loadings as contained in input parameter.

### Usage

```
kmAux(i, input, k, kmNStart, kmMax)
```

### Arguments

|                  |  |
|------------------|--|
| <i>i</i>         | number of loadings to use.   |
| <i>input</i>     | the matrix of all variable loadings.   |
| <i>k</i>         | centers (integer) parameter passed to <a href="#">kmeans</a> function.   |
| <i>kmNStart</i>  | nstart parameter passed to <a href="#">kmeans</a> function. Can be set manually. By default it is 1000 for up to 2000 cells and 50 for more than 2000 cells. |
| <i>kmMax</i>     | iter.max parameter passed to <a href="#">kmeans</a> function.  |
| <i>nCores</i>    | defines the number of cores to be used on the user's machine. If not set, 'ccImpute' will use all but one cores of your machine.                             |
| <i>nDim</i>      | the list of containing a number of PCA loadings to use for each sub-dataset.   |
| <i>rand_seed</i> | sets the seed of the random number generator. ccImpute is a stochastic method, and setting the <i>rand_seed</i> allows reproducibility.                      |

### Value

a list of clustering assignments for all the sub-datasets.

---

|            |   |
|------------|---|
| solveDrops | <i>Computes imputed expression matrix using linear eq solver.</i> |
|------------|---|

---

### Description

Computes imputed expression matrix using linear eq solver.

### Usage

```
solveDrops(cm, em, ids, n_cores)
```

**Arguments**

|         |  |
|---------|--|
| cm      | processed consensus matrix                         |
| em      | expression matrix                                  |
| ids     | location of values determined to be dropout events |
| n_cores | number of cores to use for parallel computation.   |

**Value**

imputed expression matrix

---

|          |  |
|----------|--|
| wCorDist | <i>Computes a weighted Pearson distance measure matrix. If ranks are used this measure turns into weighted Spearman distance measure matrix.</i> |
|----------|--|

---

**Description**

Computes a weighted Pearson distance measure matrix. If ranks are used this measure turns into weighted Spearman distance measure matrix.

**Usage**

```
wCorDist(x, w, useRanks, n_cores)
```

**Arguments**

|          |  |
|----------|--|
| x        | input with columns containing each observation             |
| w        | weights for all values in a observation                    |
| useRanks | indicates if Pearson should be computed on weighted ranks. |
| n_cores  | number of cores to use for parallel computation.           |

**Value**

weighted Pearson distance measure matrix. If ranks are used this measure turns into weighted Spearman distance measure matrix.

# Index

## \* **internal**

- findDropouts, 4
- findNDim, 4
- getPConsMtx, 5
- kmAux, 6

ccImpute, 2

findDropouts, 4  
findNDim, 4

getConsMtx, 5  
getPConsMtx, 5

kmAux, 6  
kmeans, 3, 6

solveDrops, 6

wCorDist, 7