Package: countland (via r-universe)

August 30, 2024

Title Analysis of Biological Count Data, Especially from Single-Cell RNA-Seq

Version 0.1.2

Description A set of functions for applying a restricted linear algebra to the analysis of count-based data. See the accompanying preprint manuscript: ``Normalizing need not be the norm: count-based math for analyzing single-cell data" Church et al (2022) <doi:10.1101/2022.06.01.494334> This tool is specifically designed to analyze count matrices from single cell RNA sequencing assays. The tools implement several count-based approaches for standard steps in single-cell RNA-seq analysis, including scoring genes and cells, comparing cells and clustering, calculating differential gene expression, and several methods for rank reduction. There are many opportunities for further optimization that may prove useful in the analysis of other data. We provide the source code freely available at <https://github.com/shchurch/countland> and encourage users and developers to fork the code for their own purposes.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.0

URL https://github.com/shchurch/countland

BugReports https://github.com/shchurch/countland/issues

Imports methods, rlang, Matrix, ggplot2

Suggests tidyverse, viridis, gridExtra, igraph, RSpectra, matrixTests, rdist, stats, Seurat, testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation no

Author Church Samuel H. [aut, cre] (<https://orcid.org/0000-0002-8451-103X>)

Maintainer Church Samuel H. <samuelhchurch@gmail.com>

Date/Publication 2024-02-01 18:00:02 UTC

Repository https://shchurch.r-universe.dev

RemoteUrl https://github.com/cran/countland

RemoteRef HEAD

RemoteSha 348e6886f6d3399cbf9af8c703bb858b689f5aea

Contents

Center *Recapitulate Seurat centering scaled and transformed data*

Description

Recapitulate Seurat centering scaled and transformed data

Usage

```
Center(C)
```

Arguments

C	countland object

Value

countland object with slots `centered_counts`

Cluster *Perform spectral clustering on dot products.*

Description

Perform spectral clustering on dot products.

Usage

```
Cluster(C, n_clusters, n_components = NULL)
```

Arguments

C	countland object
n_clusters	number of clusters, integer
n_components	number of components from spectral embedding to use (default NULL, will be set to n_clusters), integer

Value

countland object with slot `cluster_labels`

Examples

```
gold_path <- system.file("testdata", package = "countland", mustWork = TRUE)
gold.data <- Seurat::Read10X(data.dir = gold_path)
C <- countland(gold.data)
C <- Dot(C)
C <- Embed(C,n_components=5)
C <- Cluster(C,n_clusters=3)
```

CountIndex *Internal function for calculating count index.*

Description

Internal function for calculating count index.

Usage

```
CountIndex(lm)
```

Arguments

lm	column vector

Value

count index = largest n where n cells have >= n counts

---

countland *Initialize a countland object from a dgCMatrix*

Description

Initialize a countland object from a dgCMatrix

Usage

```
countland(m, remove_empty = TRUE, verbose = TRUE)
```

Arguments

m	A matrix of counts (dense or sparse)
remove_empty	filter out cells and genes with no observed counts (default=TRUE)
verbose	show stderr message statements (default=TRUE)

Value

countland object

Examples

```
gold_path <- system.file("testdata", package = "countland", mustWork = TRUE)
gold.data <- Seurat::Read10X(data.dir = gold_path)
C <- countland(gold.data)
```

`countland-class` *An S4 class to represent a countland object*

Description

An S4 class to represent a countland object

Slots

`counts` A dgCMatrix with rows as cells, columns as genes.

`names_genes` A character vector of column names.

`names_cells` A character vector of row names.

`raw_counts` The count dgCMatrix as originally loaded.

`raw_names_genes` The gene name character vector as originally loaded.

`raw_names_cells` The cell name character vector as originally loaded.

`subsample` A dgCMatrix with row sums equal.

`cell_scores` A data.frame of cell count measures.

`gene_scores` A data.frame of gene expression measures.

`dots` A similarity dgCMatrix of dot products.

`eigenvals` An vector of eigenvalues from spectral embedding

`embedding` An array of two columns (spectral embeddings).

`cluster_labels` A numeric vector of cluster assignments of length n cells.

`marker_full` A list of data.frames with genes ranked for each cluster.

`marker_genes` A data.frame of top ten marker genes per cluster.

`matrixU` A dgCMatrix of dimensions cells x features.

`matrixV` A dgCMatrix of dimensions genes x features.

`matrixLambda` A diagonal dgCMatrix of scaling factors.

`sharedcounts` A similarity dgCMatrix of shared counts between genes.

`sum_sharedcounts` A dgCMatrix with counts summed within gene clusters.

`sum_sharedcounts_all` A dgCMatrix with counts summed and including all genes not present in any cluster.

`norm_factor` A numeric vector of cell normalization factors.

`norm_counts` A dgCMatrix of normalized counts.

`log_counts` A dgCMatrix of log transformed counts.

`scaled_counts` A dgCMatrix of counts scaled by gene unit variance.

`centered_counts` A dgCMatrix of counts centered at zero.

`verbose` A T/F object for suppressing messages

`Dot` *Calculate pairwise dot products of counts between all cells.*

Description

Calculate pairwise dot products of counts between all cells.

Usage

```
Dot(C, subsample = FALSE)
```

Arguments

`C`	countland object
`subsample`	if TRUE, use subsampled counts, otherwise use counts (default=FALSE)

Value

countland object with slot `dots`

Examples

```
gold_path <- system.file("testdata", package = "countland", mustWork = TRUE)
gold.data <- Seurat::Read10X(data.dir = gold_path)
C <- countland(gold.data)
C <- Dot(C)
```

Embed — *Perform spectral embedding on dot products.*

Description

Perform spectral embedding on dot products.

Usage

```
Embed(C, n_components = 10)
```

Arguments

C	countland object
n_components	number of components, integer (default=10)

Value

countland object with slot embedding, eigenvals

Examples

```
gold_path <- system.file("testdata", package = "countland", mustWork = TRUE)
gold.data <- Seurat::Read10X(data.dir = gold_path)
C <- countland(gold.data)
C <- Dot(C)
C <- Embed(C,n_components=5)
```

IMA — *run integer matrix approximation*

Description

run integer matrix approximation

Usage

```
IMA(X, params)
```

Arguments

X	observed data matrix
params	parameter object

Value

U, V, and Lambda matrix factors

IMA_Compute_Init_Scaled *rescale if max val is above upper bound*

Description

rescale if max val is above upper bound

Usage

```
IMA_Compute_Init_Scaled(h, l_bound, u_bound)
```

Arguments

h	matrix to be rescaled
l_bound	lower bound
u_bound	upper bound

Value

rescaled matrix

IMA_init *function to initialize U, V, and Lambda*

Description

function to initialize U, V, and Lambda

Usage

```
IMA_init(X, params)
```

Arguments

X	observed data matrix
params	parameter object

Value

initialized U, V, and Lambda matrices

IMA_params *Parameter class for IMA*

Description

Parameter class for IMA

Usage

```
IMA_params(
  rank,
  u_bounds,
  l_bounds = c(0, 0),
  maxiter = 1e+06,
  stop_crit = 1e-04
)
```

Arguments

rank	target number of features in final matrices
u_bounds	upper bounds on integers
l_bounds	lower bounds on integers (default = c(0,0))
maxiter	maximum number of iterations (default = 1000000)
stop_crit	criterion of difference at which to stop (default = 0.0001)

Value

parameter object

IMA_Update_Factor *Update factor matrix - see SUSTain code*

Description

Update factor matrix - see SUSTain code

Usage

```
IMA_Update_Factor(M, coeff, mkrp, mode, lambda_, params)
```

Arguments

M	matrix to be updated (either U or V)
coeff	matrix used in updating algorithm
mkrp	matrix used in updating algorithm
mode	whether update U or V
lambda_	scaling matrix
params	parameter object

Value

updated matrix and scaling factors

---

listCols *Split dgCMatrix into column vectors.*

Description

Split dgCMatrix into column vectors.

Usage

```
listCols(m)
```

Arguments

m	dgCMatrix

Value

list of column vectors, numeric

Log *Recapitulate Seurat log transformation*

Description

Recapitulate Seurat log transformation

Usage

```
Log(C)
```

Arguments

C	countland object

Value

countland object with slots `log_counts`

`Normalize` *Recapitulate Seurat normalization*

Description

Recapitulate Seurat normalization

Usage

```
Normalize(C)
```

Arguments

`C`	countland object

Value

countland object with slots `norm_factor`, `norm_counts`

`PlotEigengap` *Plots eigenvalues to investigate the optimal number of clusters*

Description

Plots eigenvalues to investigate the optimal number of clusters

Usage

```
PlotEigengap(C)
```

Arguments

`C`	countland object

Value

generates plot of eigenvalues by number of components

Examples

```
gold_path <- system.file("testdata", package = "countland", mustWork = TRUE)
gold.data <- Seurat::Read10X(data.dir = gold_path)
C <- countland(gold.data)
C <- Dot(C)
C <- Embed(C,n_components=5)
PlotEigengap(C)
```

PlotEmbedding *Plot cells using spectral embedding of dot products.*

Description

Plot cells using spectral embedding of dot products.

Usage

```
PlotEmbedding(C, colors = color_palette)
```

Arguments

C	countland object
colors	color palette for ggplot2, default=palette of 11 colors

Value

generates plot of cells in two spectral embedding dimensions

Examples

```
gold_path <- system.file("testdata", package = "countland", mustWork = TRUE)
gold.data <- Seurat::Read10X(data.dir = gold_path)
C <- countland(gold.data)
C <- Dot(C)
C <- Embed(C,n_components=5)
C <- Cluster(C,n_clusters=3)
PlotEmbedding(C)
```

PlotGeneCounts *Generate a strip plot for counts across selected genes*

Description

Generate a strip plot for counts across selected genes

Usage

```
PlotGeneCounts(C, gene_indices, colors = color_palette)
```

Arguments

C	countland object
gene_indices	vector of gene index values
colors	color palette for ggplot2, default=palette of 11 colors

Value

generates plot of gene count distributions

Examples

```
gold_path <- system.file("testdata", package = "countland", mustWork = TRUE)
gold.data <- Seurat::Read10X(data.dir = gold_path)
C <- countland(gold.data)
PlotGeneCounts(C,gene_indices=1:10)
```

`PlotIMA` *Plot cells using integer matrix approximation*

Description

Plot cells using integer matrix approximation

Usage

```
PlotIMA(C, x = 1, y = 2, colors = color_palette, subsample = TRUE)
```

Arguments

`C`	countland object
`x`	feature on x-axis, integer (default=1)
`y`	feature on y-axis, integer (default=2)
`colors`	color palette for ggplot2, default=palette of 11 colors
`subsample`	if TRUE, use subsampled counts (default), otherwise use counts

Value

generates plot of cells using integer matrix approximation

`PlotIMAElbow` *Plot the difference between the observed and reconstructed count matrix using integer matrix approximation and a series of total features.*

Description

Plot the difference between the observed and reconstructed count matrix using integer matrix approximation and a series of total features.

Usage

```
PlotIMAElbow(C, max_features, u_bounds, subsample = TRUE)
```

Arguments

C	countland object
max_features	maximum number of features to assess, integer
u_bounds	upper bounds for U and V matrices, vector of length 2
subsample	if TRUE, use subsampled counts (default), otherwise use counts

Value

generates elbow plot for the difference between observed and reconstructed matrices as number of features increases

---

PlotMarker *Plot cell using spectral embedding and display counts in a given gene.*

Description

Plot cell using spectral embedding and display counts in a given gene.

Usage

```
PlotMarker(C, gene_index, colors = color_palette)
```

Arguments

C	countland object
gene_index	index value for gene to visualize
colors	color palette for ggplot2, default=palette of 11 colors

Value

generates plot of cells with spectral embedding, colored by marker gene counts

---

PlotSharedCounts *Plot cells using matrix of counts summed by clusters of genes.*

Description

Plot cells using matrix of counts summed by clusters of genes.

Usage

```
PlotSharedCounts(C, x = 1, y = 2, colors = color_palette)
```

Arguments

C	countland object
x	gene cluster to plot on x-axis, integer (default=1)
y	gene cluster to plot on y-axis, integer (default=2)
colors	color palette for ggplot2, default=palette of 11 colors

Value

generates plot of cells using shared counts

PrintGeneNumber *Restore count matrix to original state*

Description

Restore count matrix to original state

Usage

```
PrintGeneNumber(C)
```

Arguments

C	countland object

Value

countland object

RankMarkerGenes *Rank the top marker genes for each cluster from spectral clustering.*

Description

Rank the top marker genes for each cluster from spectral clustering.

Usage

```
RankMarkerGenes(C, method = "prop-zero", subsample = FALSE)
```

Arguments

C	countland object
method	prop-zero to rank by proportion of cells that are non-zero (default), or rank-sums to rank using Wilcoxon rank-sums test
subsample	if TRUE, use subsampled counts, otherwise use counts (default=FALSE)

Value

countland object with slots `marker_genes` and `marker_full`

Examples

```
gold_path <- system.file("testdata", package = "countland", mustWork = TRUE)
gold.data <- Seurat::Read10X(data.dir = gold_path)
C <- countland(gold.data)
C <- Dot(C)
C <- Embed(C,n_components=5)
C <- Cluster(C,n_clusters=3)
C <- RankMarkerGenes(C,method='prop-zero',subsample=FALSE)
```

RemoveEmpty *Internal function to remove empty columns and rows*

Description

Internal function to remove empty columns and rows

Usage

```
RemoveEmpty(C)
```

Arguments

C	countland object

Value

countland object, count matrix updated

RescaleVariance *Recapitulate Seurat scaling to unit variance*

Description

Recapitulate Seurat scaling to unit variance

Usage

```
RescaleVariance(C)
```

Arguments

C	countland object

Value

countland object with slots `scaled_counts`

`RestoreCounts` *Restore count matrix to original state*

Description

Restore count matrix to original state

Usage

```
RestoreCounts(C)
```

Arguments

`C`	countland object

Value

countland object

Examples

```
gold_path <- system.file("testdata", package = "countland", mustWork = TRUE)
gold.data <- Seurat::Read10X(data.dir = gold_path)
C <- countland(gold.data)
C <- SubsetGenes(C,gene_indices=1:200)
C <- SubsetCells(C,cell_indices=1:50)
C <- RestoreCounts(C)
```

`RunIMA` *Perform integer matrix approximation on count matrix.*

Description

Perform integer matrix approximation on count matrix.

Usage

```
RunIMA(
  C,
  features,
  u_bounds,
  l_bounds = c(0, 0),
  maxiter = 1e+06,
  stop_crit = 1e-04,
  subsample = TRUE
)
```

Arguments

C	countland object
features	target number of features, integer
u_bounds	upper bounds for U and V matrices, vector of length 2
l_bounds	lower bounds for U and V matrices, vector of length 2 (default=c(0,0))
maxiter	maximum number of iterations, integer (default=1000000)
stop_crit	criterion for stopping based on difference between iterations, numeric (default=0.0001)
subsample	if TRUE, use subsampled counts (default), otherwise use counts

Value

countland object with slots matrixU, matrixV, matrixLambda

Examples

```
gold_path <- system.file("testdata", package = "countland", mustWork = TRUE)
gold.data <- Seurat::Read10X(data.dir = gold_path)
C <- countland(gold.data)
C <- RunIMA(C,features=10,u_bounds=c(10,10),subsample=FALSE)
```

ScikitManifoldSpectralEmbedding

Recapitulate scikit.manifold.spectral_embedding from python.

Description

Recapitulate scikit.manifold.spectral_embedding from python.

Usage

```
ScikitManifoldSpectralEmbedding(A, n_components)
```

Arguments

A	similarity matrix, dgCMatrix
n_components	number of eigenvectors to retain, integer

Value

matrix of eigenvectors

ScoreCells *Calculate several scores for counts across cells*

Description

Calculate several scores for counts across cells

Usage

```
ScoreCells(C, gene_string = NULL)
```

Arguments

C	countland object
gene_string	string with regular expression expression matching gene names of interest (default=NULL)

Value

countland object with slot cell_scores

Examples

```
gold_path <- system.file("testdata", package = "countland", mustWork = TRUE)
gold.data <- Seurat::Read10X(data.dir = gold_path)
C <- countland(gold.data)
C <- ScoreCells(C,gene_string="*149932$")
```

ScoreGenes *Calculate several scores for count-based gene expression.*

Description

Calculate several scores for count-based gene expression.

Usage

```
ScoreGenes(C, subsample = FALSE)
```

Arguments

C	countland object
subsample	if TRUE, use subsampled counts, otherwise use counts (default=FALSE)

Value

countland object with slot gene_scores

Examples

```
gold_path <- system.file("testdata", package = "countland", mustWork = TRUE)
gold.data <- Seurat::Read10X(data.dir = gold_path)
C <- countland(gold.data)
C <- ScoreGenes(C)
```

SharedCounts *Combine groups of genes with similar counts by clustering and summing.*

Description

Combine groups of genes with similar counts by clustering and summing.

Usage

```
SharedCounts(C, n_clusters, n_cells = 100, subsample = TRUE)
```

Arguments

C	countland object
n_clusters	number of clusters
n_cells	number of cells to sample for gene clustering
subsample	if TRUE, use subsampled counts (default), otherwise use counts

Value

countland object with slots shared_counts, sum_sharedcounts, sum_sharedcounts_all

Examples

```
gold_path <- system.file("testdata", package = "countland", mustWork = TRUE)
gold.data <- Seurat::Read10X(data.dir = gold_path)
C <- countland(gold.data)
C <- SharedCounts(C,n_clusters=10,subsample=FALSE)
```

`Subsample` *Subsample cells to a standard number of counts by randomly sampling observations without replacement.*

Description

Subsample cells to a standard number of counts by randomly sampling observations without replacement.

Usage

```
Subsample(C, gene_counts = NA, cell_counts = NA)
```

Arguments

`C`	countland object
`gene_counts`	maximum total counts for genes
`cell_counts`	sequencing depth for all cells, or "min" to use the minimum cell total

Value

countland object with slot `subsample`

Examples

```
gold_path <- system.file("testdata", package = "countland", mustWork = TRUE)
gold.data <- Seurat::Read10X(data.dir = gold_path)
C <- countland(gold.data)
C <- Subsample(C,gene_counts=250,cell_counts=100)
```

`SubsampleCol` *Internal function for subsampling a column from a sparse matrix.*

Description

Internal function for subsampling a column from a sparse matrix.

Usage

```
SubsampleCol(lm, li, j, n_counts)
```

Arguments

`lm`	column vector
`li`	row positions
`j`	column index
`n_counts`	count to sample

Value

subsampled column as dgTMatrix components

`SubsetCells` *Subsets cells using a vector of cell indices*

Description

Subsets cells using a vector of cell indices

Usage

```
SubsetCells(C, cell_indices, remove_empty = TRUE)
```

Arguments

`C`	countland object
`cell_indices`	vector of cell index values
`remove_empty`	filter out cells and genes with no observed counts (default=TRUE)

Value

countland object, count matrix updated

Examples

```
gold_path <- system.file("testdata", package = "countland", mustWork = TRUE)
gold.data <- Seurat::Read10X(data.dir = gold_path)
C <- countland(gold.data)
C <- SubsetCells(C,cell_indices=1:50)
```

`SubsetGenes` *Subsets genes using a vector of gene indices*

Description

Subsets genes using a vector of gene indices

Usage

```
SubsetGenes(C, gene_indices, remove_empty = TRUE)
```

Arguments

C	countland object
gene_indices	vector of gene index values
remove_empty	filter out cells and genes with no observed counts (default=TRUE)

Value

countland object, count matrix updated

Examples

```
gold_path <- system.file("testdata", package = "countland", mustWork = TRUE)
gold.data <- Seurat::Read10X(data.dir = gold_path)
C <- countland(gold.data)
C <- SubsetGenes(C,gene_indices=1:200)
```

Index