Package 'DWLS'

May 7, 2026

Type Package

Title Gene Expression Deconvolution Using Dampened Weighted Least Squares

Version 0.1.0

Maintainer Adriana Sistig <adriana.sistig@icahn.mssm.edu>

Description The rapid development of single-cell transcriptomic technologies has helped uncover the cellular heterogeneity within cell populations. However, bulk RNA-seq continues to be the main workhorse for quantifying gene expression levels due to technical simplicity and low cost. To most effectively extract information from bulk data given the new knowledge gained from single-cell methods, we have developed a novel algorithm to estimate the cell-type composition of bulk data from a single-cell RNA-seq-derived cell-type signature. Comparison with existing methods using various real RNA-seq data sets indicates that our new approach is more accurate and comprehensive than previous methods, especially for the estimation of rare cell types. More importantly,our method can detect cell-type composition changes in response to external perturbations, thereby providing a valuable, cost-effective method for dissecting the cell-type-specific effects of drug treatments or condition changes. As such, our method is applicable to a wide range of biological and clinical investigations. Dampened weighted least squares ('DWLS') is an estimation method for gene expression deconvolution, in which the cell-type composition of a bulk RNA-seq data set is computationally inferred. This method corrects common biases towards cell types that are characterized by highly expressed genes and/or are highly prevalent, to provide accurate detection across diverse cell types. See: <https://www.nature.com/articles/s41467-019-10802-z.pdf> for more information about the development of 'DWLS' and the methods behind our functions.

URL https://github.com/sistia01/DWLS

BugReports https://github.com/sistia01/DWLS/issues

Depends R (>= 3.5.0)

Imports quadprog, reshape, Seurat, ROCR, varhandle, dplyr, stats, utils, e1071, MAST, SummarizedExperiment

License GPL-2

Encoding UTF-8

Language en-US

LazyData true

LazyDataCompression xz

RoxygenNote 7.1.2

Suggests testthat (>= 3.0.0), Matrix (>= 1.3.3)

Config/testthat/edition 3

NeedsCompilation no

Author Daphne Tsoucas [aut],
Adriana Sistig [aut, cre]

Repository CRAN

Date/Publication 2022-05-24 09:20:01 UTC

Contents

`buildSignatureMatrixMAST` *Signature Matrix Using MAST*

Description

This function builds a signature matrix using genes identified by the DEAnalysisMAST() function.

Usage

```
buildSignatureMatrixMAST(
  scdata,
  id,
  path,
  diff.cutoff = 0.5,
  pval.cutoff = 0.01,
  f = 200
)
```

Arguments

`scdata`	The data
`id`	The identities of the genes
`path`	The path to the file results
`diff.cutoff`	This is automatically set to 0.5
`pval.cutoff`	This is automatically set to 0.01
`f`	The maximum number of genes (when creating the signature matrix, need to reduce number of genes, between 50:f number of significant genes are chosen). If not set, this number is automatically set to 200.

Value

Signature Matrix built using the MAST algorithm

Examples

```
#dataSC
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/dataSC.RData"
#dest <- "data/dataSC.RData"
#load(download.file(url, tempfile(data/dataSC.RData))
#load("dataSC.RData")
#SOLUTION
load(system.file("extdata", "dataSC.RData", package = "DWLS"))

#dataBulk
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/dataBulk.RData"
```

```
#dest <- "data/dataBulk.RData"
#load(download.file(url, tempfile(dest)))
#load("data/dataBulk.RData")
load(system.file("extdata", "dataBulk.RData", package = "DWLS"))

#labels
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/labels.RData"
#dest <- "data/labels.RData"
#download.file(url, dest)
#load("data/labels.RData")
load(system.file("extdata", "labels.RData", package = "DWLS"))

#data('trueLabels', package = "DWLS")
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/trueLabels.RData"
#dest <- "data/trueLabels.RData"
#download.file(url, dest)
#load("data/trueLabels.RData")
load(system.file("extdata", "trueLabels.RData", package = "DWLS"))

labels<-trueLabels

#Change to real labels
newcat<-c("NonCycISC","CycISC","TA","Ent","PreEnt","Goblet","Paneth","Tuft",
"EE")
for (i in 1:length(newcat)){
  labels[which(labels==(i-1))]<-newcat[i]
  }

#Run on local w/ inst/extdata/results folder
#Signature <-
#buildSignatureMatrixMAST(
#dataSC,labels,"inst/extdata/results",diff.cutoff = 0.5,pval.cutoff = 0.01)
```

---

buildSignatureMatrixUsingSeurat
Signature Matrix Using Seurat

Description

This function builds a signature matrix using genes identified by the DEAnalysis() function.

Usage

```
buildSignatureMatrixUsingSeurat(
  scdata,
  id,
  path,
  diff.cutoff = 0.5,
  pval.cutoff = 0.01,
```

```
  f = 200
)
```

Arguments

scdata	The data
id	The identities of the genes
path	The path to the file results
diff.cutoff	This is automatically set to 0.5
pval.cutoff	The p-value cutoff. This is automatically set to 0.01
f	The maximum number of genes (when creating the signature matrix, need to reduce number of genes, between 50:f number of significant genes are chosen). If not set, this number is automatically set to 200.

Value

Signature Matrix built using the Seurat algorithm

---

dataSC_3 *Single cell data*

Description

A subset of the dataSC dataset in inst/extdata/

Usage

```
dataSC_3
```

Format

Data

Source

https://pubmed.ncbi.nlm.nih.gov/28467820/

DEAnalysisMAST *DEAnalysisMAST*

Description

Perform DE analysis using MAST. Dampened weighted least squares (DLWS) is an estimation method for gene expression deconvolution, in which the cell-type composition of a bulk RNA-seq data set is computationally inferred. This method corrects common biases towards cell types that are characterized by highly expressed genes and/or are highly prevalent, to provide accurate detection across diverse cell types. To begin, the user must input a bulk RNA-seq data set, along with a labeled representative single-cell RNA-seq data set that will serve to generate cell-type-specific gene expression profiles. Ideally, the single-cell data set will contain cells from all cell types that may be found in the bulk data. DWLS will return the cell-type composition of the bulk data. First, solve OLS then use the solution to find a starting point for the weights. Next, the dampened weighted least squares is performed. The weights are iterated until convergence then the dampening constant for weights is found using cross-validation (with decreasing step size for convergence).

DWLS captures ISC composition changes across conditions. One of the most important applications of deconvolution methods is in the identification of cell-type composition variations across conditions.

Note: The function uses solveDampenedWLSj() and findDampeningConstant().

Usage

```
DEAnalysisMAST(scdata, id, path)
```

Arguments

scdata	The gene expression datafarme
id	The unique identities within the data
path	The path for the RData results

Value

matrix The resulting matrix is a gene by cell-type signature matrix. The cell-type signature matrix is constructed using a representative single-cell data set, such that all cell types expected in the bulk data are also represented in the single-cell data (the converse need not be true). The single-cell data is first clustered to reveal its constituent cell types.The function will return 3 different files: an RData file, an rds file, and a csv file.

Examples

```
#dataSC
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/dataSC.RData"
#dest <- "data/dataSC.RData"
```

```
#load(download.file(url, tempfile(data/dataSC.RData))
#load("dataSC.RData")
#SOLUTION
load(system.file("extdata", "dataSC.RData", package = "DWLS"))

#dataBulk
url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/dataBulk.RData"
dest <- "data/dataBulk.RData"
load(download.file(url, tempfile(dest)))
#load("data/dataBulk.RData")
load(system.file("extdata", "dataBulk.RData", package = "DWLS"))

#labels
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/labels.RData"
#dest <- "data/labels.RData"
#download.file(url, dest)
#load("data/labels.RData")
load(system.file("extdata", "labels.RData", package = "DWLS"))

#data('trueLabels', package = "DWLS")
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/trueLabels.RData"
#dest <- "data/trueLabels.RData"
#download.file(url, dest)
#load("data/trueLabels.RData")
load(system.file("extdata", "trueLabels.RData", package = "DWLS"))

labels<-trueLabels

#Old Method
#load("data/dataBulk.RData") #read in bulk data for WT1 (control condition #1)
#load("data/labels.RData") #read in single-cell labels from clustering
#data('dataSC_3', package = "DWLS")
#dataSC <- dataSC_3

labels<-trueLabels

#Old Method
#load("data/dataBulk.RData") #read in bulk data for WT1 (control condition #1)
#load("data/labels.RData") #read in single-cell labels from clustering
#data('dataSC_3', package = "DWLS")
#dataSC <- dataSC_3

labels<-trueLabels

#Change to real labels
newcat<-c("NonCycISC","CycISC","TA","Ent","PreEnt","Goblet","Paneth","Tuft",
 "EE")
for (i in 1:length(newcat)){
  labels[which(labels==(i-1))]<-newcat[i]
  }

#Run deconvolution
#Results are in inst/extdata/results folder -- run on local
```

```
#Example code below
Mast_test <- DEAnalysisMAST(dataSC, labels, "inst/extdata/results")
```

DEAnalysisSeurat *Differential Expression without Idents (Seurat)*

Description

This function calculates the differential expression values along with identifying the Idents (through Seurat). The output is saved in an RData file for each unique identity (id).

Usage

```
DEAnalysisSeurat(scdata, id, path)
```

Arguments

scdata	The gene expression datafarme
id	The unique identities within the data
path	The path for the RData results

Value

An RData and rds file with the differential expression analysis results for each unique id.

Examples

```
#dataSC
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/dataSC.RData"
#dest <- "data/dataSC.RData"
#download.file(url, dest)
#load("data/dataSC.RData")
load(system.file("extdata", "dataSC.RData", package = "DWLS"))

#dataBulk
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/dataBulk.RData"
#dest <- "data/dataBulk.RData"
#download.file(url, dest)
#load("data/dataBulk.RData")
load(system.file("extdata", "dataBulk.RData", package = "DWLS"))

#labels
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/labels.RData"
#dest <- "data/labels.RData"
#download.file(url, dest)
#load("data/labels.RData")
```

```
load(system.file("extdata", "labels.RData", package = "DWLS"))

#data('trueLabels', package = "DWLS")
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/trueLabels.RData"
#dest <- "data/trueLabels.RData"
#download.file(url, dest)
#load("data/trueLabels.RData")
load(system.file("extdata", "trueLabels.RData", package = "DWLS"))

labels<-trueLabels

#Old Method
#load("data/dataBulk.RData") #read in bulk data for WT1 (control condition #1)
#load("data/labels.RData") #read in single-cell labels from clustering
#data('dataSC_3', package = "DWLS")
#dataSC <- dataSC_3

labels<-trueLabels
#Change to real labels
newcat<-c("NonCycISC","CycISC","TA","Ent","PreEnt","Goblet","Paneth","Tuft","EE")
for (i in 1:length(newcat)){
  labels[which(labels==(i-1))]<-newcat[i]
  }
#Run deconvolution -- run on local
#Results in inst/extdata/results
Seurat_DE <- DEAnalysisSeurat(dataSC, labels, "inst/extdata/results")
```

`DEAnalysisSeuratIdents` *Differential Expression with Idents (Seurat)*

Description

This function calculates the differential expression values along with identifying the Idents (through Seurat). The output is saved in an RData file for each unique identity (id).

Usage

```
DEAnalysisSeuratIdents(scdata, id, path)
```

Arguments

`scdata`	The gene expression datafarme
`id`	The unique identities within the data
`path`	The path for the RData results

Value

An RData file with the differential expression analysis results for each unique id.

Examples

```
#dataSC
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/dataSC.RData"
#dest <- "data/dataSC.RData"
#load(download.file(url, tempfile(data/dataSC.RData))
#load("dataSC.RData")
#SOLUTION
load(system.file("extdata", "dataSC.RData", package = "DWLS"))

#dataBulk
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/dataBulk.RData"
#dest <- "data/dataBulk.RData"
#load(download.file(url, tempfile(dest)))
#load("data/dataBulk.RData")
load(system.file("extdata", "dataBulk.RData", package = "DWLS"))

#labels
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/labels.RData"
#dest <- "data/labels.RData"
#download.file(url, dest)
#load("data/labels.RData")
load(system.file("extdata", "labels.RData", package = "DWLS"))

#data('trueLabels', package = "DWLS")
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/trueLabels.RData"
#dest <- "data/trueLabels.RData"
#download.file(url, dest)
#load("data/trueLabels.RData")
load(system.file("extdata", "trueLabels.RData", package = "DWLS"))

#Old Method
#load("data/dataSC_3.RData")
#load("data/trueLabels.RData")
#load("data/dataBulk.RData") #read in bulk data for WT1 (control condition #1)
#load("data/labels.RData") #read in single-cell labels from clustering

labels<-trueLabels
#Change to real labels
newcat<-c("NonCycISC","CycISC","TA","Ent","PreEnt","Goblet","Paneth",
"Tuft","EE")
for (i in 1:length(newcat)){
  labels[which(labels==(i-1))]<-newcat[i]
  }
#Run deconvolution
#Seurat_test2 <- DEAnalysisSeuratIdents(dataSC, labels, "results")
```

findDampeningConstant *findDampeningConstant*

Description

Finds a dampening constant for the weights using cross-validation. The goldStandard is used to define the weights. Multiple values of the dampening constant (multiplier) are tried. For each attempt, the variance of the dampened weighted solution for a subset of genes is calculated (on a randomly selected half of the genes). Note that infinite weights are ignored.The dampening constant that results in least cross-validation variance is chosen. It functions in a nondeterministic manner. The dampening constant defines the maximum value that any weight can take on.

Usage

```
findDampeningConstant(S, B, goldStandard)
```

Arguments

S	List output from trimData$Sig (S)
B	List output from trimData$dataBulk (B)
goldStandard	Starting point for the weights, determined by solving OLS

Value

value (dampening constant value)

Examples

```
#Sig
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/Sig.RData"
#dest <- "data/Sig.RData"
#download.file(url, dest)
#load("data/Sig.RData")
load(system.file("extdata", "Sig.RData", package = "DWLS"))

#dataBulk
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/dataBulk.RData"
#dest <- "data/dataBulk.RData"
#download.file(url, dest)
#load("data/dataBulk.RData")
load(system.file("extdata", "dataBulk.RData", package = "DWLS"))

trimmed <- trimData(Sig, dataBulk)
S <- trimmed$sig
B <- trimmed$bulk
solution <- solveOLSInternal(S,B)
findDampeningConstant(S, B, solution)
```

m.auc *m.auc*

Description

Calculates the AUC of a dataset. The function mainly serves to support the DWLS function.

Usage

```
m.auc(dataset, grouping)
```

Arguments

dataset	Data
grouping	Data subdivision

Value

Matrix of standardized output of AUC calculation

Examples

```
#dataSC
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/dataSC.RData"
#dest <- "data/dataSC.RData"
#load(download.file(url, tempfile(data/dataSC.RData))
#load("dataSC.RData")
#SOLUTION
load(system.file("extdata", "dataSC.RData", package = "DWLS"))

#dataBulk
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/dataBulk.RData"
#dest <- "data/dataBulk.RData"
#load(download.file(url, tempfile(dest)))
#load("data/dataBulk.RData")
load(system.file("extdata", "dataBulk.RData", package = "DWLS"))

#labels
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/labels.RData"
#dest <- "data/labels.RData"
#download.file(url, dest)
#load("data/labels.RData")
load(system.file("extdata", "labels.RData", package = "DWLS"))

#data('trueLabels', package = "DWLS")
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/trueLabels.RData"
#dest <- "data/trueLabels.RData"
#download.file(url, dest)
```

```
#load("data/trueLabels.RData")
load(system.file("extdata", "trueLabels.RData", package = "DWLS"))

pseudo.count = 0.1
data.used.log2 <- log2(dataSC+pseudo.count)
colnames(data.used.log2)<-make.unique(colnames(data.used.log2))
diff.cutoff=0.5
id = labels
for (i in unique(id)){
  cells.symbol.list2 = colnames(data.used.log2)[which(id==i)]
  cells.coord.list2 = match(cells.symbol.list2, colnames(data.used.log2))
  cells.symbol.list1 = colnames(data.used.log2)[which(id != i)]
  cells.coord.list1= match(cells.symbol.list1, colnames(data.used.log2))
  data.used.log2.ordered = cbind(data.used.log2[,cells.coord.list1],
                                         data.used.log2[,cells.coord.list2])
  group.v <- c(rep(0,length(cells.coord.list1)),
                               rep(1, length(cells.coord.list2)))
  #ouput
  log2.stat.result <- stat.log2(data.used.log2.ordered,
                                    group.v, pseudo.count)
  Auc <- m.auc(data.used.log2.ordered, group.v)}
```

`MASTSignatureMatrixGivenDE` *Build Signature matrix using MAST given a differential expression matrix*

Description

This function builds a signature matrix using a pre-created differential expression matrix. The input matrix must have the same format as the DEAnalysisMAST() function and must be saved as an RData file ending with _MIST. The file must be named identity_MIST.RData. See exampledata_MIST.RData for more information (inst/man).

Usage

```
MASTSignatureMatrixGivenDE(
  scdata,
  id,
  path,
  diff.cutoff = 0.5,
  pval.cutoff = 0.01
)
```

Arguments

`scdata`	The data

id	The identities of the genes
path	The path to the file results
diff.cutoff	This is automatically set to 0.5
pval.cutoff	This is automatically set to 0.01

Value

Signature Matrix built using the MAST algorithm

Examples

```
#dataSC
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/dataSC.RData"
#dest <- "data/dataSC.RData"
#load(download.file(url, tempfile(data/dataSC.RData))
#load("dataSC.RData")
#SOLUTION
load(system.file("extdata", "dataSC.RData", package = "DWLS"))

#dataBulk
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/dataBulk.RData"
#dest <- "data/dataBulk.RData"
#load(download.file(url, tempfile(dest)))
#load("data/dataBulk.RData")
load(system.file("extdata", "dataBulk.RData", package = "DWLS"))

#labels
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/labels.RData"
#dest <- "data/labels.RData"
#download.file(url, dest)
#load("data/labels.RData")
load(system.file("extdata", "labels.RData", package = "DWLS"))

#data('trueLabels', package = "DWLS")
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/trueLabels.RData"
#dest <- "data/trueLabels.RData"
#download.file(url, dest)
#load("data/trueLabels.RData")
load(system.file("extdata", "trueLabels.RData", package = "DWLS"))

labels<-trueLabels

#Change to real labels
newcat<-c("NonCycISC","CycISC","TA","Ent","PreEnt","Goblet",
"Paneth","Tuft","EE")
for (i in 1:length(newcat)){
  labels[which(labels==(i-1))]<-newcat[i]
  }
#Results in inst/extdata/results -- run on local
```

```
#Signature<-buildSignatureMatrixMAST(dataSC,labels,"results",
# diff.cutoff=0.5,pval.cutoff=0.01)
```

`Mean.in.log2space` *Mean.in.log2space*

Description

Applies the log2 to the mean of ((2^x - pseudo count) + pseudo count).

Usage

```
Mean.in.log2space(x, pseudo.count)
```

Arguments

`x`	Data
`pseudo.count`	A pseudocount value

Value

Values

Examples

```
#dataBulk
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/dataBulk.RData"
#dest <- "data/dataBulk.RData"
#download.file(url, dest)
#load("data/dataBulk.RData")
load(system.file("extdata", "dataBulk.RData", package = "DWLS"))

Mean.in.log2space(dataBulk, 0.1)
```

solveDampenedWLS *solveDampenedWLS*

Description

Dampened weighted least squares (DLWS) is an estimation method for gene expression deconvolution, in which the cell-type composition of a bulk RNA-seq data set is computationally inferred. This method corrects common biases towards cell types that are characterized by highly expressed genes and/or are highly prevalent, to provide accurate detection across diverse cell types. To begin, the user must input a bulk RNA-seq data set, along with a labeled representative single-cell RNA-seq data set that will serve to generate cell-type-specific gene expression profiles. Ideally, the single-cell data set will contain cells from all cell types that may be found in the bulk data. DWLS will return the cell-type composition of the bulk data. First, solve OLS then use the solution to find a starting point for the weights. Next, the dampened weighted least squares is performed. The weights are iterated until convergence then the dampening constant for weights is found using cross-validation (with decreasing step size for convergence).

Note: The function uses solveDampenedWLSj() and findDampeningConstant().

Usage

```
solveDampenedWLS(S, B)
```

Arguments

S	List output from trimData
B	List output from trimData

Value

value (Dampened weighted least squares estimation values)

Examples

```
#Sig
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/Sig.RData"
#dest <- "data/Sig.RData"
#download.file(url, dest)
#load("data/Sig.RData")
load(system.file("extdata", "Sig.RData", package = "DWLS"))

#dataBulk
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/dataBulk.RData"
#dest <- "data/dataBulk.RData"
#download.file(url, dest)
#load("data/dataBulk.RData")
load(system.file("extdata", "dataBulk.RData", package = "DWLS"))

trimmed <- trimData(Sig, dataBulk)
```

```
S <- trimmed$sig
B <- trimmed$bulk
solveDampenedWLS(S, B)
```

`solveDampenedWLSj` *solveDampenedWLSj*

Description

Solve dampened weighted least squares given a dampening constant.

Note: The function uses solveDampenedWLS() and findDampeningConstant().

Usage

```
solveDampenedWLSj(S, B, goldStandard, j)
```

Arguments

`S`	List output from trimData$sig (S)
`B`	List output from trimData$bulk (B)
`goldStandard`	Starting point for the weights, this can be determined using solveOLSInternal(S,B)
`j`	The dampening constant, this can be determined using findDampeningConstant(S,B,goldStandard)

Value

value (Dampened weighted least squares estimation values)

Examples

```
#Sig
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/Sig.RData"
#dest <- "data/Sig.RData"
#download.file(url, dest)
#load("data/Sig.RData")
load(system.file("extdata", "Sig.RData", package = "DWLS"))

#dataBulk
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/dataBulk.RData"
#dest <- "data/dataBulk.RData"
#download.file(url, dest)
#load("data/dataBulk.RData")
load(system.file("extdata", "dataBulk.RData", package = "DWLS"))

trimmed <- trimData(Sig, dataBulk)
S <- trimmed$sig
```

```
B <- trimmed$bulk
solution <- solveOLSInternal(S,B)
j <- findDampeningConstant(S,B,solution)
goldStandard <- solveOLSInternal(S,B)
solveDampenedWLSj(S,B,goldStandard,j)
```

---

solveOLS *solveOLS*

Description

This function solves or the unknown parameters using ordinary least squares (OLS). It is constrained such that cell type numbers are greater than 0.

Usage

```
solveOLS(S, B)
```

Arguments

S	List output from trimData$sig (S)
B	List output from trimData$bulk (B)

Value

Cell-type proportion

Examples

```
#Sig
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/Sig.RData"
#dest <- "data/Sig.RData"
#download.file(url, dest)
#load("data/Sig.RData")
load(system.file("extdata", "Sig.RData", package = "DWLS"))

#dataBulk
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/dataBulk.RData"
#dest <- "data/dataBulk.RData"
#download.file(url, dest)
#load("data/dataBulk.RData")
load(system.file("extdata", "dataBulk.RData", package = "DWLS"))

trimmed <- trimData(Sig, dataBulk)
S <- trimmed$sig
B <- trimmed$bulk
solveOLS(S, B)
```

solveOLSInternal *solveOLSInternal*

Description

This function solves or the unknown parameters using ordinary least squares (OLS) without printing the output. It returns the cell numbers, not the proportions (see solveOLS).

Usage

```
solveOLSInternal(S, B)
```

Arguments

S	List output from trimData$sig (S)
B	List output from trimData$bulk (B)

Value

Cell numbers

Examples

```
#Sig
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/Sig.RData"
#dest <- "data/Sig.RData"
#download.file(url, dest)
#load("data/Sig.RData")
load(system.file("extdata", "Sig.RData", package = "DWLS"))

#dataBulk
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/dataBulk.RData"
#dest <- "data/dataBulk.RData"
#download.file(url, dest)
#load("data/dataBulk.RData")
load(system.file("extdata", "dataBulk.RData", package = "DWLS"))

trimmed <- trimData(Sig, dataBulk)
S <- trimmed$sig
B <- trimmed$bulk
solveOLSInternal(S, B)
```

solveSVR *solveSVR*

Description

Performs a support vector regression (SVR). First, the data is scaled then it solves for the SVR. An svm model is used with the following specifications nu=0.5,scale = TRUE, type = "nu-regression", kernel ="linear",cost = 1.

Nu-support vector regression was performed using the svm function in the e1071 package in R. Parameters were set to nu = 0.5, type = “nu-regression”, kernel = “linear”, cost = 1, and all others to the default values. Bulk data and signature matrices were scaled to -1, 1. These parameter and scaling choices match those specified in Schelker et al. in their MATLAB code, accessed through https://figshare.com/s/865e694ad06d5857db4b. As in Newman et al., model coefficients are extracted from the svm model using t(model$coefs) model$SV, and any negative coefficients are set to zero. The coefficients are then scaled by the sum of the coefficients, such that the scaled coefficients will sum to one.

Citations: Newman, A. M. et al. Robust enumeration of cell subsets from tissue expression profiles. Nat. Methods 12, 453–457 (2015).

Schelker, M. et al. Estimation of immune cell content in tumor tissue using single-cell RNA-seq data. Nat. Commun. 8, 2032 (2017).

Usage

```
solveSVR(S, B)
```

Arguments

S	List output from trimData$sig (S)
B	List output from trimData$bulk (B)

Value

Value (SVR)

Examples

```
#Sig
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/Sig.RData"
#dest <- "data/Sig.RData"
#download.file(url, dest)
#load("data/Sig.RData")
load(system.file("extdata", "Sig.RData", package = "DWLS"))

#dataBulk
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/dataBulk.RData"
#dest <- "data/dataBulk.RData"
#download.file(url, dest)
```

```
#load("data/dataBulk.RData")
load(system.file("extdata", "dataBulk.RData", package = "DWLS"))

trimmed <- trimData(Sig, dataBulk)
S <- trimmed$sig
B <- trimmed$bulk
solveSVR(S, B)
```

---

stat.log2 *stat.log2*

Description

One of the functions required for the differential expression analysis using MAST (DEAnalysisMast()) function.

Usage

```
stat.log2(data.m, group.v, pseudo.count)
```

Arguments

data.m	Data
group.v	Groupings
pseudo.count	A pseudocount value

Value

A dataframe of the log2 applied results

Examples

```
#dataSC
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/dataSC.RData"
#dest <- "data/dataSC.RData"
#load(download.file(url, tempfile(data/dataSC.RData))
#load("dataSC.RData")
#SOLUTION
load(system.file("extdata", "dataSC.RData", package = "DWLS"))

#dataBulk
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/dataBulk.RData"
#dest <- "data/dataBulk.RData"
#load(download.file(url, tempfile(dest)))
#load("data/dataBulk.RData")
load(system.file("extdata", "dataBulk.RData", package = "DWLS"))
```

```
#labels
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/labels.RData"
#dest <- "data/labels.RData"
#download.file(url, dest)
#load("data/labels.RData")
load(system.file("extdata", "labels.RData", package = "DWLS"))

#data('trueLabels', package = "DWLS")
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/trueLabels.RData"
#dest <- "data/trueLabels.RData"
#download.file(url, dest)
#load("data/trueLabels.RData")
load(system.file("extdata", "trueLabels.RData", package = "DWLS"))

pseudo.count = 0.1
data.used.log2 <- log2(dataSC+pseudo.count)
colnames(data.used.log2)<-make.unique(colnames(data.used.log2))
diff.cutoff=0.5
id = labels
for (i in unique(id)){
  cells.symbol.list2 = colnames(data.used.log2)[which(id==i)]
  cells.coord.list2 = match(cells.symbol.list2, colnames(data.used.log2))
  cells.symbol.list1 = colnames(data.used.log2)[which(id != i)]
  cells.coord.list1 = match(cells.symbol.list1, colnames(data.used.log2))
  data.used.log2.ordered = cbind(data.used.log2[,cells.coord.list1],
                                 data.used.log2[,cells.coord.list2])}
group.v <- c(rep(0,length(cells.coord.list1)),
                                  rep(1, length(cells.coord.list2)))
```

`trimData` *trimData*

Description

This function trims bulk and single-cell data to contain the same genes. The result is a list of the intersecting genes within the two datasets.

Usage

```
trimData(Signature_Matrix, bulkdata)
```

Arguments

`Signature_Matrix`	A single-cell signature matrix
`bulkdata`	A bulk dataset

Value

A list of trimmed bulk and single-cell data.

Examples

```
#Sig
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/Sig.RData"
#dest <- "data/Sig.RData"
#download.file(url, dest)
#load("data/Sig.RData")
load(system.file("extdata", "Sig.RData", package = "DWLS"))

#dataBulk
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/dataBulk.RData"
#dest <- "data/dataBulk.RData"
#download.file(url, dest)
#load("data/dataBulk.RData")
load(system.file("extdata", "dataBulk.RData", package = "DWLS"))

trimData(Signature_Matrix = Sig, bulkdata = dataBulk)
```

`v.auc` *v.auc*

Description

Uses the prediction() function in order to create standardized output from the data in order to perform an AUC calculation. The calculation results are rounded to the third decimal place. This function serves mainly to support the DWLS function.

Usage

```
v.auc(data.v, group.v)
```

Arguments

`data.v`	Data
`group.v`	Data subdivision

Value

Matrix of standardized output of AUC calculation

Examples

```
#dataSC
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/dataSC.RData"
#dest <- "data/dataSC.RData"
#load(download.file(url, tempfile(data/dataSC.RData))
#load("dataSC.RData")
#SOLUTION
```

```
load(system.file("extdata", "dataSC.RData", package = "DWLS"))

#dataBulk
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/dataBulk.RData"
#dest <- "data/dataBulk.RData"
#load(download.file(url, tempfile(dest)))
#load("data/dataBulk.RData")
load(system.file("extdata", "dataBulk.RData", package = "DWLS"))

#labels
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/labels.RData"
#dest <- "data/labels.RData"
#download.file(url, dest)
#load("data/labels.RData")
load(system.file("extdata", "labels.RData", package = "DWLS"))

#data('trueLabels', package = "DWLS")
#url <- "https://github.com/sistia01/DWLS/raw/main/inst/extdata/trueLabels.RData"
#dest <- "data/trueLabels.RData"
#download.file(url, dest)
#load("data/trueLabels.RData")
load(system.file("extdata", "trueLabels.RData", package = "DWLS"))

pseudo.count = 0.1
data.used.log2   <- log2(dataSC+pseudo.count)
colnames(data.used.log2)<-make.unique(colnames(data.used.log2))
diff.cutoff=0.5
id = labels
for (i in unique(id)){
  cells.symbol.list2 = colnames(data.used.log2)[which(id==i)]
  cells.coord.list2 = match(cells.symbol.list2, colnames(data.used.log2))
  cells.symbol.list1 = colnames(data.used.log2)[which(id != i)]
  cells.coord.list1= match(cells.symbol.list1, colnames(data.used.log2))
  data.used.log2.ordered = cbind(data.used.log2[,cells.coord.list1],
                                       data.used.log2[,cells.coord.list2])
  group.v <- c(rep(0,length(cells.coord.list1)),
                               rep(1, length(cells.coord.list2)))
  #ouput
  log2.stat.result <- stat.log2(data.used.log2.ordered,
                                    group.v, pseudo.count)
m.auc=function(data.used.log2.ordered,group.v)
{AUC=apply(data.used.log2.ordered, 1, function(x) v.auc(x,group.v))
AUC[is.na(AUC)]=0.5
return(AUC)}

}
```

Index