Package 'decone'

October 30, 2022

Type Package

Date 2022-09-18

Title Cell Type Deconvolution Evaluator

Version 1.0.1

Description The evaluation platform for cell type deconvolution. This platform provides multiple simulation and real mixture expression datasets for evaluate the performance of deconvlution tools. This toolkit also plots preety figure for the comparison of different deconvolution methods.

License GPL-3 | file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.2.1

Depends R (>= 4.1.0)

Imports Metrics,
stats,
utils,
Matrix,
tibble,
data.table,
tools,
magrittr,
ggplot2,
reshape2,
grDevices,
forcats,
stringr,
R.utils,
ggpubr,
progress

Suggests knitr,
rmarkdown

VignetteBuilder knitr

R topics documented:

`addNoise` *Add noise to gene expression data*

Description

Add noise based on negtive bionomial distribution. Please see "A benchmark for RNA-seq deconvolution analysis under dynamic testing environments" in Genome Biology.

Usage

```
addNoise(x = NULL, pt = 0.1, type = "NB")
```

Arguments

`x`	a gene expression numeric vector.
`pt`	parameter to control noised level. Default: 0.1
`type`	"NB", "N" or "LN". "NB" means Negative binomial model, "N" means normal model, "LN" means Log-normal model. Default: "NB"

Value

the sample length vector.

Examples

```
res <- addNoise(x = seq(100))
```

addNoiseExpr *Add noise to the simulated expression matrix*

Description

Add different level noise to the simulated expression matrix based on the negative binomial distribution. The related model can be found in this paper (Jin, H., Liu, Z. A benchmark for RNA-seq deconvolution analysis under dynamic testing environments. Genome Biol 22, 102 (2021). https://doi.org/10.1186/s13059-021-02290-6).

Usage

```
addNoiseExpr(
  exprFile,
  outputPath = NULL,
  prefix = NULL,
  Pt = seq(0.1, 1, 0.1),
  type = "NB"
)
```

Arguments

exprFile	The input expression file. Must be in csv format. Each row is a gene, each column is a sample. rownames and colnames are required. Please check the output of the function exprSim.
outputPath	Output path, create if not exists. Default: a new folder based on the exprFile. For example, if the exprFile is "/data/test.txt", the outputPath will be a new folder "/data/test".
prefix	The prefix of the output file. Note: the suffix will be added automatically based on the noise level.
Pt	Parameter to control noise level. Default: seq(0.1, 1, 0.1)
type	"NB", "N" or "LN". Noise type. "NB" means Negative binomial model, "N" means normal model, "LN" means Log-normal model. Default: "NB"

Value

All the information will be written in the output path, and each file is the generated data in a specific noise level.

Examples

```
res <- pseudoExpr()
write.csv(x = res$mix, file = "mix.csv", row.names = T, quote = F)
addNoiseExpr(exprFile = "mix.csv", Pt = seq(0.1, 1, 0.1), type = "NB")
```

CPM *Convert a gene expression count matrix into CPM matrix.*

Description

Convert a gene expression count matrix into CPM matrix

Usage

```
CPM(data = NULL)
```

Arguments

data	A dataframe or matrix object. Note: Must contain a column named "Length", "Length" means gene length.

Value

A dataframe or matrix object.

Examples

```
data <- data.frame(s1 = seq(100), Length = seq(100))
res <- CPM(data)
```

exprSim *Generate in silico mixture expression matrix*

Description

Generate in silico mixture expression matrix based on the internal RNA-seq database. The internal RNA-seq database is collected from multiple studies. All the samples are passed the quality filter. This function is different from pseudoExpr. Counts in pseudoExpr is randomly generated from uniform distribution. This function use the real cell type specific expression data to generate mixture data. We provide two types of simulation called "coarse" and "fine". This idea is from DREAM Challenge Tumor Deconvolution problem. https://www.synapse.org/#!Synapse:syn15589870/wiki/.

Usage

```
exprSim(
  n_sample = 50,
  p = 2/3,
  type = "coarse",
  transform = "TPM",
  outputPath = NULL,
  mix_name = "coarse_gene_expr.csv",
  ref_name = "coarse_ref.csv",
  prop_name = "coarse_prop.csv",
```

```
  refVar_name = NULL,
  train_name = NULL
)
```

Arguments

n_sample	Sample number to be generated, default: 50.
p	Proportion of sample in train set, default: 0.6.
type	"coarse" or "fine". "coarse" means the simulation will be performed in a coarse level. Only 8 cell types will be used, including ("B.cells", "CD4.T.cells", "CD8.T.cells", "endothelial.cells", "macrophages", "monocytes","neutrophils", "NK.cells"). "fine" means the simulation will be performed in a fine level. 14 cell types will be used, including ("memory.B.cells", "naive.B.cells", "memory.CD4.T.cells", "naive.CD4.T.cells", "regulatory.T.cells", "memory.CD8.T.cells", "naive.CD8.T.cells", "NK.cells", "neutrophils", "monocytes", "myeloid.dendritic.cells", "macrophages", "fibroblasts", "endothelial.cells")
transform	"TPM", "CPM" or "NO". Transform the data into TPM, CPM or in counts.
outputPath	output file save path.
mix_name	mixture output file name.
ref_name	reference output file name in csv.
prop_name	simulated proportion file name in csv.
refVar_name	reference variance file name in csv.
train_name	file name for all data in train set in csv. This data can be used for differential gene analysis.

Value

All the information will be written in the output path.

Examples

```
exprSim()
```

---

plot_multiple *Plotting function for comparison multiple deconvolution method.*

Description

Generating plots for multiple deconvolution method. This method is designed for comparing the results from different methods under a certain scenario or one method under different scenario. For example, comparing the deconvolution effect of different methods from a specific noise level data, or comparing the deconvolution effect of one method from different noise level. Of course, there can be many samples in this certain scenario. But for comparing the results from data with different noise level, please use plot_multiple2. Note: Function plot_multiple can reveal the deconvolution effect for celltypes as well as samples directly. Function plot_multiple2 reveals the overall deconvolution results for different scenarios. The celltype or sample specific effect can not be illustrated. Users should choose the appropriate function. Of course, users can adopt the results from each function to perform customized analysis.

Usage

```
plot_multiple(
  actual,
  predicted,
  label = NULL,
  method = NULL,
  method2 = NULL,
  type = "sample",
  figure = "boxplot",
  errbar = "SE",
  nrow = 3
)
```

Arguments

actual	The groundtruth proportion of cell types in matrix. row: cell types, column: samples. Also can be a csv matrix with row and column names.
predicted	a vector contains all the files with the predicted proportions from different methods. Must be in csv file with row and column names. row: cell types, column: samples. For example, c("method1.csv", "method2.csv", "method3.csv").
label	a vector contains the label corresponding to the predicted proportion file. Default: base file name in parameter 'predicted'. For example, c("method1", "method2", "method3").
method	One of the c("mae", "rmse", "mape", "kendall", "pearson", "spearman"). Note: for mape, cell types with real proportion 0 will be ignored. Note: for scatter plot, method must in c("kendall", "pearson", "spearman").
method2	One of the c("mae", "rmse", "mape", "kendall", "pearson", "spearman"). Generating the second metric in cheatmap. Note: should be different from parameter method.
type	One of the c("sample", "celltpye", "all"). For "sample", generate metric for each sample. Scatter plot will assign different shape and color for each sample. For "celltype", generate metric for each cell type. Scatter plot will assign different shape and color for each celltype. For "all", generate metric for all data, which means flattening all data into an vector. Scatter plot will remove shape and color. Note: Parameter 'type' cannot be set to 'all' when plotting boxplot, heatmap and cheatmap.
figure	One of the c("boxplot", "barplot", "scatterplot", "heatmap", "cheatmap") Note: Parameter type cannot be set to 'all' when plotting boxplot, heatmap and cheatmap. Note: for scatter plot, method must in c("kendall", "pearson", "spearman").
errbar	error bar type for barplot. One of the c("SD", "SE"), default: SE. SD: Standard Deviation SE: Standard Error
nrow	Only used in scatterplot. Control the layout of output figure.

Value

The computed metrics and plot data.

plot_multiple2 *Plotting function for comparison multiple deconvolution method.*

Description

Generating plots for multiple deconvolution method. This method is designed for comparing the results from different methods under different scenario. For example, comparing the deconvolution effect of different methods from various noise level data. Of course, there can be many samples in a certain scenario. But for comparing the results from data with a certain noise level, please use plot_multiple. Note: Function plot_multiple can reveal the deconvolution effect for celltypes as well as samples directly. Function plot_multiple2 reveals the overall deconvolution results for different scenarios. The celltype or sample specific effect can not be illustrated. Users should choose the appropriate function. Of course, users can adopt the results from each function to perform customized analysis.

Usage

```
plot_multiple2(
  actual,
  predicted,
  condition = NULL,
  method = NULL,
  method2 = NULL,
  type = "sample",
  figure = "boxplot",
  errbar = NULL
)
```

Arguments

actual	The groundtruth proportion of cell types in matrix. row: cell types, column: samples. Also can be a csv matrix with row and column names.
predicted	a list of vectors. Each vector contains all the files with the predicted proportions in different scenarios. Must be in csv file with row and column names. row: cell types, column: samples. For example, list(method1 = c("m1_noise0.csv", "m1_noise1.csv", "m1_noise2.csv"), method2 = c("m2_noise0.csv", "m2_noise1.csv", "m2_noise2.csv")). Note: the name of each method should be specified. This name will be used for generating the label when plotting.
condition	a vector contains the condition labels corresponding to the predicted proportion file. For example, c("noise0", "noise1", "noise2").
method	One of the c("mae", "rmse", "mape", "kendall", "pearson", "spearman"). Note: for mape, cell types with real proportion 0 will be ignored.
method2	One of the c("mae", "rmse", "mape", "kendall", "pearson", "spearman"). Generating the second metric in cheatmap.
type	One of the c("sample", "celltpye", "all"). Note: method2 will be disabled when figure is in c("boxplot", "barplot", "heatmap")
figure	One of the c("boxplot", "barplot", "heatmap", "cheatmap")
errbar	error bar type for barplot. One of the c("SD", "SE"), default: SE. SD: Standard Deviation SE: Standard Error

Value

The computed metrics and plot data.

`plot_rare` *Scatter plot for rare component.*

Description

Generate scatter plot for rare component. Note: only the rare proportion will be considered. The 'scatterplot' function can be used to estimate the deconvolution power for each cell type. The 'heatmap' and 'cheatmap' functions can be used to compare the deconvolution power for each method.

Usage

```
plot_rare(
  actual,
  predicted,
  p_rare = c(0.001, 0.003, 0.005, 0.008, 0.01, 0.03, 0.05),
  method = NULL,
  method2 = NULL,
  type = NULL,
  celltype = TRUE,
  figure = NULL
)
```

Arguments

`actual`	The groundtruth proportion of cell types in matrix or csv file. row: cell types, column: samples. Also can be a csv matrix with row and column names.
`predicted`	The predicted proportion of cell types in matrix or csv file. row: cell types, column: samples. Also can be a csv matrix with row and column names. Note: For parameter 'figure' is scatterplot, 'predicted' must be one matrix or filename. This means scatterplot is designed for generating plot for one method. For example, predicted = 'method1_predicted.csv' Note: For parameter 'figure' is in c("heatmap", "cheatmap"), 'predicted' must be a vector of filename. This means heatmap is designed for generating plot for multiple methods. For example, predicted = c('method1_predicted.csv', 'method2_predicted.csv', 'method3_predicted.csv')
`p_rare`	A vector of proportions. should be the same with function `rareExprSim`. Default: c(0.001, 0.003, 0.005, 0.008, 0.01, 0.03, 0.05)
`method`	One of the c("mae", "rmse", "mape", "kendall", "pearson", "spearman"). For parameter 'figure' is scatterplot, method must be one of c("kendall", "pearson", "spearman") Note: kendall, pearson and spearman is not supported by heatmap and cheatmap.
`method2`	One of the c("mae", "rmse", "mape", "kendall", "pearson", "spearman"). Generating the second metric in cheatmap. Note: method2 will be disabled when figure is in c("scatterplot", "heatmap") Note: kendall, pearson and spearman is not supported by heatmap and cheatmap.

type	Not used. All metrics will be computed for each rare component, regardless of sample and cell type.
celltype	TRUE or FALSE. Assign different type and color for different cell types. Default: TRUE. Note: This parameter only supported by the scatterplot.
figure	One of the c("scatterplot", "heatmap", "cheatmap"). For different type of figure, users should pass the correct parameters.
label	a vector contains the label corresponding to the predicted proportion file from different methods. Default: base file name in parameter 'predicted'. For example, c("method1", "method2", "method3"). Note: this parameter is only supported by the heatmap and cheatmap functions.

Value

The plot data.

Examples

```
rareExprSim()
# after the simulation, just pass the predicted file
# to the parameter "actual" and "predicted"
```

---

plot_single *Plotting function for a single deconvolution method.*

Description

Gnerating plots for a specific deconvolution method.

Usage

```
plot_single(
  actual,
  predicted,
  method = NULL,
  type = "sample",
  figure = "boxplot",
  errbar = "SE"
)
```

Arguments

actual	The groundtruth proportion of cell types. Can be a matrix or csv file with row and column names. row: cell types, column: samples.
predicted	The predicted proportion of cell types. Can be a matrix or csv file with row and column names. row: cell types, column: samples.
method	One of the c("mae", "rmse", "mape", "kendall", "pearson", "spearman"). Note: for mape, cell types with real proportion 0 will be ignored. Note: for scatter plot, method must in c("kendall", "pearson", "spearman").

type	One of the c("sample", "celltpye", "all"). For "sample", generate metric for each sample. Scatter plot will assign different shape and color for each sample. For "celltype", generate metric for each cell type. Scatter plot will assign different shape and color for each celltype. For "all", generate metric for all data, which means flattening all data into an vector. Scatter plot will remove shape and color. Note: boxplot is not supported for type="all".
figure	One of the c("boxplot", "barplot", "scatterplot") Note: boxplot is not supported for type="all". Note: for scatter plot, method must in c("kendall", "pearson", "spearman").
errbar	error bar type for barplot. One of the c("SD", "SE"), default: SE. SD: Standard Deviation SE: Standard Error

Value

The computed metrics as well as the plot data.

Examples

```
res <- pseudoData(type = 1)
res <- plot_single(actual = res$actual, predicted = res$predicted, method = "mape", type = "sample", figure = "b
```

`pseudoData` *Generate pseudo data for plotting*

Description

Generate pseudo data (prediction values) for plotting

Usage

```
pseudoData(type = 1, outputPath = "test_file")
```

Arguments

type	1, 2 or 3 for different usage
outputPath	path to save tmp file

Value

The pseudo data

Examples

```
res <- pseudoData(type = 1)
```

pseudoExpr *Generate pseudo mixture expression matrix*

Description

Generate pseudo mixture expression matrix based on uniform distribution, without any noise.

Usage

```
pseudoExpr(n_sample = 50, n_gene = 1000, n_ct = 10)
```

Arguments

n_sample	Sample number to be generated, default: 50
n_gene	Gene number to be generated, default: 1000
n_ct	Cell type number to be generated, default: 10

Value

a list, prop means the pseudo proportion, ref means the pseudo external reference, mix is the output of ref

Examples

```
res <- pseudoExpr()
```

rareExprSim *Generate in silico expression dataset with rare component.*

Description

Generate in silico mixture expression matrix based on the internal RNA-seq database. This function is different from the function exprSim. exprSim generates all the proportion randomly, while this function takes one cell type as rare component and all Other cell type proportion will be set randomly from the uniform distribution. The internal RNA-seq database is collected from multiple studies. All the samples are passed the quality filter. We provide two types of simulation called "coarse" and "fine". This idea is from DREAM Challenge Tumor Deconvolution problem. https://www.synapse.org/#!Synapse:syn15589870/wiki/.

Usage

```
rareExprSim(
  p_rare = c(0.001, 0.003, 0.005, 0.008, 0.01, 0.03, 0.05),
  p = 0.6,
  type = "coarse",
  transform = "TPM",
  outputPath = NULL,
  mix_name = "coarse_gene_expr.csv",
```

```
  ref_name = "coarse_ref.csv",
  prop_name = "coarse_prop.csv",
  refVar_name = NULL,
  train_name = NULL
)
```

Arguments

p_rare	A vector of proportions. Default: c(0.001, 0.003, 0.005, 0.008, 0.01, 0.03, 0.05) Note: Every cell type will be treated as rare component. For example, if 8 cell types need to be tested, this function will generate 7 * 8 = 56 samples. 7 mean 7 rare proportions and 8 means 8 cell types.
p	Proportion of sample in train set, default: 0.6.
type	"coarse" or "fine". "coarse" means the simulation will be performed in a coarse level. Only 8 cell types will be used, including ("B.cells", "CD4.T.cells", "CD8.T.cells", "endothelial.cells", "macrophages", "monocytes","neutrophils", "NK.cells"). "fine" means the simulation will be performed in a fine level. 14 cell types will be used, including ("memory.B.cells", "naive.B.cells", "memory.CD4.T.cells", "naive.CD4.T.cells", "regulatory.T.cells", "memory.CD8.T.cells", "naive.CD8.T.cells", "NK.cells", "neutrophils", "monocytes", "myeloid.dendritic.cells", "macrophages", "fibroblasts", "endothelial.cells")
transform	"TPM", "CPM" or "NO". Transform the data into TPM, CPM or in counts.
outputPath	output file save path.
mix_name	mixture output file name.
ref_name	reference output file name in csv.
prop_name	simulated proportion file name in csv.
refVar_name	reference variance file name in csv.
train_name	file name for all data in train set in csv. This data can be used for differential gene analysis.

Value

All the information will be written in the output path.

Examples

```
rareExprSim()
```

`rarescExprSim` *Generate in silico mixture expression matrix based on scRNA-seq with rare component.*

Description

Generate in silico mixture expression matrix based on the internal scRNA-seq database. The internal RNA-seq database is collected from PMID:29474909. All the samples are passed the quality filter. This function use the real cell type specific expression data to generate mixture data. The cell type include c("FetalStomach", "FetalLung", "FetalLiver", "FetalKidney", "FetalIntestine", "FetalBrain", "Female.fetal.Gonad")

Usage

```
rarescExprSim(
  p_rare = c(0.001, 0.003, 0.005, 0.008, 0.01, 0.03, 0.05),
  p = 2/3,
  transform = "TPM",
  outputPath = NULL,
  mix_name = "scMouse_gene_expr.csv",
  ref_name = "scMouse_ref.csv",
  prop_name = "scMouse_prop.csv",
  train_name = "scMouse_ref_rawCount.csv",
  type = "mouse_tissue"
)
```

Arguments

p_rare	A vector of proportions. Default: c(0.001, 0.003, 0.005, 0.008, 0.01, 0.03, 0.05) Note: Every cell type will be treated as rare component. For example, if 8 cell types need to be tested, this function will generate 7 * 8 = 56 samples. 7 mean 7 rare proportions and 8 means 8 cell types.
p	Proportion of sample in train set, default: 2 / 3.
transform	"TPM", "CPM" or "NO". Transform the data into TPM, CPM or in counts.
outputPath	output file save path.
mix_name	mixture output file name.
ref_name	reference output file name in csv.
prop_name	simulated proportion file name in csv.
train_name	file name for all data in train set in csv. This data can be used for differential gene analysis.
type	'mouse_tissue' or 'human_PBMC'. Default: 'mouse_tissue'

Value

All the information will be written in the output path.

Examples

```
rarescExprSim()
```

---

regMetrics *Compute regression Metrics for matrix results.*

Description

Compute several regression Metrics for cell type deconcolution.

Usage

```
regMetrics(actual, predicted, method = NULL, type = NULL)
```

Arguments

actual	The groundcloth proportion of cell types in matrix. row: cell types, column: samples.
predicted	The predicted proportion of cell types in matrix. row: cell types, column: samples.
method	One of c("rmse", "mape", "mae", "kendall", "pearson", "spearman"),
type	One of c("sample.", "celltype", "all"). For "sample.", generate metric for each sample. For "celltype", generate metric for each cell type. For "all", generate metric for all data, which means flattening all data into an vector. Note: for mape, cell types with read proportion 0 will be ignored.

Value

a vector with the value for each sample.

Examples

```
res <- regMetrics(actual = matrix(data = seq(9), nrow = 3),
                  predicted = matrix(data = seq(9), nrow = 3),
                  method = "rmse")
```

---

scExprSim *Generate in silico mixture expression matrix based on scRNA-seq*

Description

Generate in silico mixture expression matrix based on the internal scRNA-seq database. The internal RNA-seq database is collected from PMID:29474909. All the samples are passed the quality filter. This function use the real cell type specific expression data to generate mixture data. The cell type include c("FetalStomach", "FetalLung", "FetalLiver", "FetalKidney", "FetalIntestine", "FetalBrain", "Female.fetal.Gonad")

Usage

```
scExprSim(
  n_sample = 50,
  p = 2/3,
  transform = "TPM",
  outputPath = NULL,
  mix_name = "scMouse_gene_expr.csv",
  ref_name = "scMouse_ref.csv",
  prop_name = "scMouse_prop.csv",
  train_name = "scMouse_ref_rawCount.csv",
  type = "mouse_tissue"
)
```

Arguments

n_sample	Sample number to be generated, default: 50.
p	Proportion of sample in train set, default: 2 / 3.
transform	"TPM", "CPM" or "NO". Transform the data into TPM, CPM or in counts.
outputPath	output file save path.
mix_name	mixture output file name.
ref_name	reference output file name in csv.
prop_name	simulated proportion file name in csv.
train_name	file name for all data in train set in csv. This data can be used for differential gene analysis.
type	'mouse_tissue' or 'human_PBMC'. Default: 'mouse_tissue'

Value

All the information will be written in the output path.

Examples

```
res <- scExprSim()
```

TPM *Convert a gene expression count matrix into TPM matrix.*

Description

Convert a gene expression count matrix into TPM matrix

Usage

```
TPM(data = NULL)
```

Arguments

data	A dataframe or matrix object. Note: Must contain a column named "Length", "Length" means gene length.

Value

A dataframe or matrix object.

Examples

```
data <- data.frame(s1 = seq(100), Length = seq(100))
res <- TPM(data)
```

unExprSim *Generate in silico expression dataset with unknown component.*

Description

Generate in silico mixture expression matrix based on the internal RNA-seq database. This function is different from the function exprSim. exprSim generates all the proportion randomly, while this function takes one cell type as unknown component. The internal RNA-seq database is collected from multiple studies. All the samples are passed the quality filter. We provide two types of simulation called "coarse" and "fine". This idea is from DREAM Challenge Tumor Deconvolution problem. https://www.synapse.org/#!Synapse:syn15589870/wiki/.

Usage

```
unExprSim(
  unknown = c(0.001, 0.005, 0.01, 0.05, 0.1, 0.15, 0.2),
  n_sample = 50,
  p = 0.6,
  type = "coarse",
  transform = "TPM",
  outputPath = NULL,
  mix_name = "coarse_gene_expr.csv",
  ref_name = "coarse_ref.csv",
  prop_name = "coarse_prop.csv",
  refVar_name = NULL,
  train_name = NULL
)
```

Arguments

unknown	a numeric vector defines the proportion of unknown content. Default: c(0.001, 0.005, 0.01, 0.05, 0.1, 0.15, 0.2)
n_sample	Sample number to be generated, default: 50.
p	Proportion of sample in train set, default: 0.6.
type	"coarse" or "fine". "coarse" means the simulation will be performed in a coarse level. Only 8 cell types will be used, including ("B.cells", "CD4.T.cells", "CD8.T.cells", "endothelial.cells", "macrophages", "monocytes","neutrophils", "NK.cells"). "fine" means the simulation will be performed in a fine level. 14 cell types will be used, including ("memory.B.cells", "naive.B.cells", "memory.CD4.T.cells", "naive.CD4.T.cells", "regulatory.T.cells", "memory.CD8.T.cells", "naive.CD8.T.cells", "NK.cells", "neutrophils", "monocytes", "myeloid.dendritic.cells", "macrophages", "fibroblasts", "endothelial.cells")
transform	"TPM", "CPM" or "NO". Transform the data into TPM, CPM or in counts.
outputPath	output file save path.
mix_name	mixture output file name.
ref_name	reference output file name in csv.
prop_name	simulated proportion file name in csv, unknown component will be removed.
refVar_name	reference variance file name in csv.
train_name	file name for all data in train set in csv. This data can be used for differential gene analysis.

Value

All the information will be written in the output path.

Examples

```
res <- unExprSim()
```

---

unscExprSim *Generate in silico mixture expression matrix based on scRNA-seq*

Description

Generate in silico mixture expression matrix based on the internal scRNA-seq database. The internal RNA-seq database is collected from PMID:29474909. All the samples are passed the quality filter. This function use the real cell type specific expression data to generate mixture data. The cell type include c("FetalStomach", "FetalLung", "FetalLiver", "FetalKidney", "FetalIntestine", "FetalBrain", "Female.fetal.Gonad")

Usage

```
unscExprSim(
  unknown = c(0.001, 0.005, 0.01, 0.05, 0.1, 0.15, 0.2),
  n_sample = 50,
  p = 2/3,
  transform = "TPM",
  outputPath = NULL,
  mix_name = "scMouse_gene_expr.csv",
  ref_name = "scMouse_ref.csv",
  prop_name = "scMouse_prop.csv",
  train_name = "scMouse_ref_rawCount.csv",
  type = "mouse_tissue"
)
```

Arguments

unknown	a numeric vector defines the proportion of unknown content. Default: c(0.001, 0.005, 0.01, 0.05, 0.1, 0.15, 0.2)
n_sample	Sample number to be generated, default: 50.
p	Proportion of sample in train set, default: 2 / 3.
transform	"TPM", "CPM" or "NO". Transform the data into TPM, CPM or in counts.
outputPath	output file save path.
mix_name	mixture output file name.
ref_name	reference output file name in csv.
prop_name	simulated proportion file name in csv.
train_name	file name for all data in train set in csv. This data can be used for differential gene analysis.
type	'mouse_tissue' or 'human_PBMC'. Default: 'mouse_tissue'

Value

All the information will be written in the output path.

Examples

```
res <- unscExprSim()
```

`v_norm` *Normalize a vector*

Description

Normalize a vector

Usage

```
v_norm(x = NULL, scale = NULL)
```

Arguments

`x`	a numeric vector
`scale`	a number to scaled

Value

Normalized numeric vector

Examples

```
res <- v_norm(seq(100))
```

Index