# Package 'Linnorm'

October 12, 2016

**Type** Package

**Title** Linear model and normality based transformation method (Linnorm)

**Version** 1.0.6

**Date** 2016-08-27

**Author** Shun Hang Yip <shunyip@bu.edu>, Panwen Wang <pwwang@pwwang.com>, Jean-Pierre Kocher <Kocher.JeanPierre@mayo.edu>, Pak Chung Sham <pcsham@hku.hk>, Junwen Wang <junwen@uw.edu>

**Maintainer** Ken Shun Hang Yip <shunyip@bu.edu>

**Description** Linnorm is an R package for the analysis of RNA-seq, scRNA-seq, ChIP-seq count data or any large scale count data. Its main function is to normalize and transform these datasets for parametric tests. Examples of parametric tests include using limma for differential expression analysis or differential peak detection, or calculating Pearson correlation coefficient for gene correlation study. Linnorm can work with raw count, CPM, RPKM, FPKM and TPM. Additionally, Linnorm provides the RnaXSim function for the simulation of RNA-seq raw counts for the evaluation of differential expression analysis methods. RnaXSim can simulate RNA-seq dataset in Gamma, Log Normal, Negative Binomial or Poisson distributions.

**License** MIT + file LICENSE

**Imports** Rcpp (>= 0.12.2), RcppArmadillo, MASS, limma, stats, utils, statmod

**LinkingTo** Rcpp, RcppArmadillo

**Suggests** BiocStyle, knitr, rmarkdown, seqc, gplots, RColorBrewer

**VignetteBuilder** knitr

**biocViews** Sequencing, ChIPSeq, RNASeq, DifferentialExpression, GeneExpression, Genetics, Normalization, Software, Transcription, BatchEffect

**NeedsCompilation** yes

**URL** http://www.jjwanglab.org/Linnorm/

**RoxygenNote** 5.0.1

# R topics documented:



---

`Linnorm` *Linnorm Transformation Function*

---

## Description

This function performs the Linear model and normality based transformation method (Linnorm) for RNA-seq expression data or large scale count data.

## Usage

```
Linnorm(datamatrix, showinfo = FALSE, method = "default",
  perturbation = 10, minZeroPortion = 2/3)
```

## Arguments

| | |
|---|---|
| `datamatrix` | The matrix or data frame that contains your dataset. Each row is a feature (or Gene) and each column is a sample (or replicate). Undefined values such as NA are not supported. |
| `showinfo` | Logical. Show lambda value calculated. Defaults to FALSE. |
| `method` | "default" or "lambda" The program will output the transformed matrix if the method is "default". If the method is "lambda", the program will output a lambda value. |
| `perturbation` | Integer >=2. To search for an optimal minimal deviation parameter (please see the article), Linnorm uses the iterated local search algorithm which perturbs away from the initial local minimum. The range of the area searched in each perturbation is exponentially increased as the area get further away from the initial local minimum, which is determined by their index. This range is calculated by 10 * (perturbation ^ index). |
| `minZeroPortion` | Double >=0, <= 1. Featuress without at least this portion of non-zero values will not be used in the calculation of normalizing parameter. Defaults to 2/3. |

## Details

If method is default, Linnorm outputs a transformed expression matrix. For users who wish to work with lambda instead, the output is a single lambda value. Please note that users with the lambda value can obtain a transformed Linnorm dataset by: log1p(lambda * datamatrix). There is no need to rerun the program if a lambda is already calculated.

## Value

This function returns either a transformed data matrix or a lambda value.

## Examples

```
#Obtain example matrix:
library(seqc)
SampleA <- ILM_aceview_gene_BGI[,grepl("A_",colnames(ILM_aceview_gene_BGI))]
rownames(SampleA) <- ILM_aceview_gene_BGI[,2]
#Extract a portion of the matrix for an example
expMatrix <- SampleA[,1:3]
transformedExp <- Linnorm(expMatrix)
transformedExp <- Linnorm(expMatrix, method = "lambda")
```

---

# Linnorm.limma *Linnorm-limma pipeline for Differentially Expression Analysis*

---

## Description

This function first performs Linnorm transformation on the dataset. Then, it will perform limma for DEG analysis. Finally, it will correct fold change outputs from limma results, that will be wrong otherwise. Please cite both Linnorm and limma when you use this function for publications.

## Usage

```
Linnorm.limma(datamatrix, design = NULL, output = "DEResults",
  noINF = TRUE, showinfo = FALSE, perturbation = 10,
  minZeroPortion = 2/3, robust = TRUE)
```

## Arguments

| | |
|---|---|
| datamatrix | The matrix or data frame that contains your dataset. Each row is a feature (or Gene) and each column is a sample (or replicate). Undefined values such as NA are not supported. |
| design | A design matrix required for limma. Please see limma's documentation or our vignettes for more detail. |
| output | Character. "DEResults" or "Both". Set to "DEResults" to output a matrix that contains Differential Expression Analysis Results. Set to "Both" to output a list that contains both Differential Expression Analysis Results and the transformed data matrix. |
| noINF | Logical. Prevent generating INF in the fold change column by using Linnorm's lambda and adding one. If it is set to FALSE, INF will be generated if one of the conditions has zero expression. Defaults to TRUE. |
| showinfo | Logical. Show lambda value calculated. Defaults to FALSE. |

| | |
|---|---|
| perturbation | Integer >=2. To search for an optimal minimal deviation parameter (please see the article), Linnorm uses the iterated local search algorithm which perturbs away from the initial local minimum. The range of the area searched in each perturbation is exponentially increased as the area get further away from the initial local minimum, which is determined by their index. This range is calculated by 10 * (perturbation ^ index). |
| minZeroPortion | Double >=0, <= 1. Featuress without at least this portion of non-zero values will not be used in the calculation of normalizing parameter. Defaults to 2/3. |
| robust | Logical. In the eBayes function of Limma, run with robust setting with TRUE or FALSE. Defaults to TRUE. |

## Details

This function performs both Linnorm and limma for users who are interested in differential expression analysis. Please note that if you directly use a Linnorm Nomralized dataset with limma, the output fold change and average expression with be wrong. (p values and adj.pvalues will be fine.) This is because the voom-limma pipeline assumes input to be in raw counts. This function is written to fix this problem.

## Value

If output is set to "DEResults", this function will output a matrix with Differntial Expression Analysis Results with the following columns:

- logFC: Log 2 Fold Change
- XPM: Average Expression. If input is raw count or CPM, this column has the CPM unit. If input is RPKM, FPKM or TPM, this column has the TPM unit.
- t: moderated t-statistic
- P.Value: p value
- adj.P.Val: Adjusted p value. This is also called False Discovery Rate or q value.
- B: log odds that the feature is differential

If output is set to Both, this function will output a list with the following objects:

- DEResults: Differntial Expression Analysis Results as described above.
- TMatrix: A Linnorm Transformed Expression Matrix.

## Examples

```
#Obtain example matrix:
library(seqc)
SampleA <- ILM_aceview_gene_BGI[,grepl("A_",colnames(ILM_aceview_gene_BGI))]
rownames(SampleA) <- ILM_aceview_gene_BGI[,2]
SampleB <- ILM_aceview_gene_BGI[,grepl("B_",colnames(ILM_aceview_gene_BGI))]
rownames(SampleB) <- ILM_aceview_gene_BGI[,2]
#Extract a portion of the matrix for an example
expMatrix <- cbind(SampleA[,1:3], SampleB[,1:3])
designmatrix <- c(1,1,1,2,2,2)
designmatrix <- model.matrix(~ 0+factor(designmatrix))
```

```
colnames(designmatrix) <- c("group1", "group2")
rownames(designmatrix) <- colnames(expMatrix)

#Example 1
DEGResults <- Linnorm.limma(expMatrix, designmatrix)
#Example 2
DEGResults <- Linnorm.limma(expMatrix, designmatrix, output="Both")
```

---

RnaXSim *This function simulates a RNA-seq dataset based on a given distribution.*

---

## Description

This function simulates a RNA-seq dataset based on a given distribution.

## Usage

```
RnaXSim(thisdata, distribution = "Poisson", NumRep = 3, NumDiff = 2000,
  NumFea = 20000, showinfo = FALSE, DEGlog2FC = "Auto",
  MaxLibSizelog2FC = 0.5)
```

## Arguments

| | |
|---|---|
| thisdata | Matrix: The matrix or data frame that contains your dataset. Each row is a gene and each column is a replicate. Undefined values such as NA are not supported. This program assumes that all columns are replicates of the same sample. |
| distribution | Character: Defaults to "Poisson". This parameter controls the output distribution of the simulated RNA-seq dataset. It can be one of "Gamma" (Gamma distribution), "Poisson" (Poisson distribution), "LogNorm" (Log Normal distribution) or "NB" (Negative Binomial distribution). |
| NumRep | Integer: The number of replicates. This is half of the number of output samples. Defaults to 3. |
| NumDiff | Integer: The number of Differentially Changed Features. Defaults to 2000. |
| NumFea | Integer: The number of Total Features. Defaults to 20000. |
| showinfo | Logical: should we show data information on the console? Defaults to FALSE. |
| DEGlog2FC | "Auto" or Double: log 2 fold change threshold that defines differentially expressed genes. If set to "Auto," DEGlog2FC is defined at the level where ANOVA can get a q value of 0.05 with the median mean, where the data values are log1p transformed. Defaults to "Auto". |
| MaxLibSizelog2FC | Double: The maximum library size difference from the mean that is allowed, in terms of log 2 fold change. Set to 0 to prevent program from generating library size differences. Defaults to 0.5. |

## Value

This function returns a list that contains a matrix of count data in integer raw count and a vector that shows which genes are differentially expressed. In the matrix, each row is a gene and each column is a replicate. The first NumRep (see parameter) of the columns belong to sample 1, and the last NumRep (see parameter) of the columns belong to sample 2. There will be NumFea (see parameter) number of rows.

## Examples

```
#Obtain example matrix:
library(seqc)
SampleA <- ILM_aceview_gene_BGI[,grepl("A_",colnames(ILM_aceview_gene_BGI))]
rownames(SampleA) <- ILM_aceview_gene_BGI[,2]
#Extract a portion of the matrix for an example
expMatrix <- SampleA[,1:10]
#Example for Negative Binomial distribution
simulateddata <- RnaXSim(expMatrix, distribution="NB", NumRep=3, NumDiff = 200, NumFea = 2000)
#Example for Poisson distribution
simulateddata <- RnaXSim(expMatrix, distribution="Poisson", NumRep=3, NumDiff = 200, NumFea = 2000)
#Example for Log Normal distribution
simulateddata <- RnaXSim(expMatrix, distribution="LogNorm", NumRep=3, NumDiff = 200, NumFea = 2000)
#Example for Gamma distribution
simulateddata <- RnaXSim(expMatrix, distribution="Gamma", NumRep=3, NumDiff = 200, NumFea = 2000)
```

# Index