

Package: 11kdeconv (via r-universe)

October 8, 2026

Type Package

Title Deconvolution for LINCS L1000 Data

Version 1.2.0

Date 2017-07-05

Author Zhao Li[aut], Peng Yu[aut, cre]

Maintainer Zhao Li <lizhao.informatics@gmail.com>

Description LINCS L1000 is a high-throughput technology that allows the gene expression measurement in a large number of assays. However, to fit the measurements of ~1000 genes in the ~500 color channels of LINCS L1000, every two landmark genes are designed to share a single channel. Thus, a deconvolution step is required to infer the expression values of each gene. Any errors in this step can be propagated adversely to the downstream analyses. We present a LINCS L1000 data peak calling R package 11kdeconv based on a new outlier detection method and an aggregate Gaussian mixture model. Upon the remove of outliers and the borrowing information among similar samples, 11kdeconv shows more stable and better performance than methods commonly used in LINCS L1000 data deconvolution.

Imports stats, mixtools, ggplot2

License GPL (>= 2)

Depends R (>= 3.2.0)

RoxygenNote 6.0.1

NeedsCompilation no

Config/pak/sysreqs cmake make libicu-dev libuv1-dev libssl-dev

Repository <https://zhaoli2017.r-universe.dev>

Date/Publication 2017-07-08 03:41:45 UTC

RemoteUrl <https://github.com/cran/11kdeconv>

RemoteRef HEAD

RemoteSha 463499db63e7bd0839be883c7ccdc97cfa672200

Contents

getclusterranges	2
gmmpplot	2
multigmmmanydata	3
multigmmplot	4
multigmmamedistribu	5
multigmmamedistribulik	6
multigmmamedistribumulti	6
rmoutlier1d	7
splitgrp	8
Index	9

getclusterranges	<i>Get the Cluster Ranges in a Vector of 1D Coordinates</i>
------------------	---

Description

Get the Cluster Ranges in a Vector of 1D Coordinates

Usage

getclusterranges(x, gap)

Arguments

- x a numeric vector
- gap the size for the recognition of data free gaps

Examples

```
x = c(1:3, 11:13)
getclusterranges(x, 3)
```

gmmpplot	<i>Plot the Fit Results of 2-Component Gaussian Mixture Model</i>
----------	---

Description

Plot the Fit Results of 2-Component Gaussian Mixture Model

Usage

gmmpplot(x, mu1, mu2, sigma, lambda, nbins = 15, xlim)

Arguments

x	a numeric vector
mu1	the mean of the 1st cluster
mu2	the mean of the 2nd cluster
sigma	the common variance of both clusters
lambda	the proportion parameter
nbins	the number of bins per cluster (6*sigma)
xlim	the limitation of x scale

Examples

```

set.seed(0)
x=list(c(
  rnorm(150, mean=0)
  , rnorm(50, mean=10)
))
fit_res=multigmmmanydata(x)

with(
  as.list(fit_res$par_conv)
  , gmmplot(x[[1]]
    , mu1=mu1
    , mu2=mu2
    , sigma=sigma
    , lambda=lambda
    , xlim=range(unlist(x))
  )
)

```

multigmmmanydata	<i>Split the input dataset into several sub list to deconvolution.</i>
------------------	--

Description

Due to the limitation of optimization that too many data would dramatically slow down the speed.

Usage

```

multigmmmanydata(x, grp_size = 3, lambda_lower = 0.1, lambda_upper = 1 -
  lambda_lower, sigma_lower = 0.01, debug = F)

```

Arguments

x	a list of numeric vector
grp_size	the normal group size for each group
lambda_lower	the lower bound of λ
lambda_upper	the upper bound of λ
sigma_lower	the lower bound of σ
debug	enable the debug mode to show par and fn

Examples

```
set.seed(0)
x1=c(rnorm(150, mean=0), rnorm(50, mean=10))
x2=c(rnorm(150, mean=20), rnorm(50, mean=40))
x3=c(rnorm(150, mean=30), rnorm(50, mean=60))
x4=c(rnorm(150, mean=30), rnorm(50, mean=60))
x5=c(rnorm(150, mean=30), rnorm(50, mean=60))
x6=c(rnorm(150, mean=30), rnorm(50, mean=60))
x=list(x1, x2, x3, x4, x5, x6)
multigmmmanydata(x)
```

multigmmplot

Plot the Fit Results of aggregate 2-Component Gaussian Mixture Model

Description

Plot the Fit Results of aggregate 2-Component Gaussian Mixture Model

Usage

```
multigmmplot(x, fit_res, nbins = 15)
```

Arguments

x	a list of a numeric vector
fit_res	the result of AGMM
nbins	the number of bins per cluster

Examples

```
params=list(
  c(mu1=0, mu2=10, sd = 1)
  , c(mu1=10, mu2=20, sd = 1)
)

set.seed(0)
```

```

x=lapply(
  params
  , function(v) {
    c(
      rnorm(100, mean=v[['mu1']], sd = v[['sd']])
      , rnorm(50, mean=v[['mu2']], sd = v[['sd']])
    )
  }
)
multigmmplot(x, multigmmssamedistribu(x))

```

multigmmssamedistribu	<i>Fit Multi 2-Component Gaussian Mixture Model in same distribution with a Fixed Proportion</i>
-----------------------	--

Description

Fit Multi 2-Component Gaussian Mixture Model in same distribution with a Fixed Proportion

Usage

```
multigmmssamedistribu(x, lambda_lower = 0.1, lambda_upper = 1 - lambda_lower,
  sigma_lower = 0.01, debug = F)
```

Arguments

x	a list of numeric vector
lambda_lower	the lower bound of λ
lambda_upper	the upper bound of λ
sigma_lower	the lower bound of σ
debug	enable the debug mode to show par and fn

Examples

```

set.seed(0)
x1=c(rnorm(150, mean=0), rnorm(50, mean=10))
x2=c(rnorm(150, mean=20), rnorm(50, mean=40))
x3=c(rnorm(150, mean=30), rnorm(50, mean=60))
x=list(x1, x2, x3)
multigmmssamedistribu(x)

```

multigmmssamedistribulik

The sum of Log-Likelihoods of 1D Multi Same Distribution Gaussian Mixture Model

Description

The sum of Log-Likelihoods of 1D Multi Same Distribution Gaussian Mixture Model

Usage

```
multigmmssamedistribulik(x)
```

Arguments

x a list of numeric vectors

Examples

```
set.seed(0)
x1=c(
  rnorm(100, mean=0)
  , rnorm(100, mean=1)
)
x=list(x1)
multigmmssamedistribulik(x)(c(0.5, 1, 0.5, 1))
```

multigmmssamedistribumulti

Split the input dataset into several sub list to deconvolution.

Description

Due to the limitation of optimization that too many data would dramatically slow down the speed.

Usage

```
multigmmssamedistribumulti(x, lambda_lower = 0.1, lambda_upper = 1 -
  lambda_lower, sigma_lower = 0.01, debug = F)
```

Arguments

x	a list of numeric vector
lambda_lower	the lower bound of λ
lambda_upper	the upper bound of λ
sigma_lower	the lower bound of σ
debug	enable the debug mode to show par and fn

Examples

```

set.seed(0)
x1=c(rnorm(150, mean=0), rnorm(50, mean=10))
x2=c(rnorm(150, mean=20), rnorm(50, mean=40))
x3=c(rnorm(150, mean=30), rnorm(50, mean=60))
x4=c(rnorm(150, mean=30), rnorm(50, mean=60))
x5=c(rnorm(150, mean=30), rnorm(50, mean=60))
x6=c(rnorm(150, mean=30), rnorm(50, mean=60))
x=list(x1, x2, x3, x4, x5, x6)
multigmmmanydata(x)

```

rmoutlier1d

*Remove the Outliers in a Vector of 1D Coordinates***Description**

Remove the Outliers in a Vector of 1D Coordinates

Usage

```
rmoutlier1d(x, dy_thr = dnorm(4), clustersize_thr = 3, gapsize = 10)
```

Arguments

x	a numeric vector
dy_thr	the threshold for dy
clustersize_thr	the threshold for cluster size
gapsize	the threshold of points in recognizing data free gap

Examples

```

x=c(1,10:30,50)
par(mfrow=c(2,1))
plot(density(x))
plot(density(rmoutlier1d(x)))

```

`splitgrp`*Split a list with size n into groups with at least m elements*

Description

Split a list with size n into groups with at least m elements

Usage

```
splitgrp(n, m)
```

Arguments

n	an integer indicating the total length
m	the min group size

Examples

```
splitgrp(1, 2)  
splitgrp(2, 2)  
splitgrp(3, 2)
```

Index

- * **category**

- splitgrp, [8](#)

- * **distribution**

- getclusterranges, [2](#)

- gmmpplot, [2](#)

- multigmmmanydata, [3](#)

- multigmmplot, [4](#)

- multigmmssamedistribu, [5](#)

- multigmmssamedistribulik, [6](#)

- multigmmssamedistribumulti, [6](#)

- rmoutlier1d, [7](#)

getclusterranges, [2](#)

gmmpplot, [2](#)

multigmmmanydata, [3](#)

multigmmplot, [4](#)

multigmmssamedistribu, [5](#)

multigmmssamedistribulik, [6](#)

multigmmssamedistribumulti, [6](#)

rmoutlier1d, [7](#)

splitgrp, [8](#)