Package: DIME (via r-universe)

August 27, 2024

Type Package

Title Differential Identification using Mixture Ensemble

Version 1.3.0

Date 2022-05-07

Author Cenny Taslim <taslim.2@osu.edu>, with contributions from Dustin Potter, Abbasali Khalili and Shili Lin <shili@stat.osu.edu>.

Maintainer Cenny Taslim <taslim.2@osu.edu>

License GPL (>= 2)

Description A robust identification of differential binding sites method for analyzing ChIP-seq (Chromatin Immunoprecipitation Sequencing) comparing two samples that considers an ensemble of finite mixture models combined with a local false discovery rate (fdr) allowing for flexible modeling of data. Methods for Differential Identification using Mixture Ensemble (DIME) is described in: Taslim et al., (2011) <doi:10.1093/bioinformatics/btr165>.

LazyLoad yes

NeedsCompilation yes

Date/Publication 2022-05-09 14:50:13 UTC

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

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

RemoteRef HEAD

RemoteSha 4c34813ded0a2333af020e6b9784f82edc648296

Contents

`DIME-package` *DIME (Differential Identification using Mixtures Ensemble)*

Description

A robust differential identification method that considers an ensemble of finite mixture models combined with a local false discovery rate *(fdr)* for analyzing ChIP-seq data comparing two samples. This package can also be used to identify differential in other high throughput data such as microarray, methylation etc.
After normalization, an Exponential-Normal(k) or a Uniform-Normal(k) mixture is fitted to the data. The (k)-normal component can represent either differential regions or non-differential regions depending on their location and spread. The exponential or uniform represent differentially sites. local *(fdr)* are computed from the fitted model. Unique features of the package:

1. Accurate modeling of data that comes from any distribution by the use of multiple normal components (any distribution can be accurately represented by mixture of normal).
2. Using ensemble of mixture models allowing data to be accurately & efficiently represented. Then two-phase selection ensure the selection of the best overall model.
3. This method can be used as a general program to fit a mixture of uniform-normal or uniform-k-normal or exponential-k-normal

Details

Package:	DIME
Type:	Package
Version:	1.0
Date:	2010-11-19
License:	GPL-2
LazyLoad:	yes

Author(s)

Cenny Taslim <taslim.2@osu.edu>, with contributions from Abbas Khalili <khalili@stat.ubc.ca>, Dustin Potter <potterdp@gmail.com>, and Shili Lin <shili@stat.osu.edu>
Maintainer: Cenny Taslim <taslim.2@osu.edu> or Shili Lin <shili@stat.osu.edu>

References

- Khalili, A., Huang, T., and Lin, S. (2009). A robust unified approach to analyzing methylation and gene expression data. *Computational Statistics and Data Analysis*, 53(5), 1701-1710.
- Dean, N. and Raftery, A. E. (2005). Normal uniform mixture differential gene expression detection for cDNA microarrays. *BMC Bioinformatics*, 6, 173.
- Taslim, C., Wu, J., Yan, P., Singer, G., Parvin, J., Huang, T., Lin, S., and Huang, K. (2009). Comparative study on chip-seq data: normalization and binding pattern characterization. *Bioinformatics*, 25(18), 2334-2340.

DIME *DIME (Differential Identification using Mixtures Ensemble)*

Description

A robust differential identification method that considers ensemble of finite mixture models combined with a local false discovery rate *(fdr)* for analyzing ChIP-seq data comparing two samples. This package can also be used to identify differential in other high throughput data such as microarray, methylation etc.

Usage

```
DIME(data, avg = NULL, gng.K = 2, gng.weights = NULL, gng.weights.cutoff= -1.345,
  gng.pi = NULL, gng.mu = NULL,
  gng.sigma = NULL, gng.beta = NULL, gng.tol = 1e-05, gng.max.iter = 2000,
  gng.th = NULL, gng.rep = 15, gng.fdr.cutoff = 0.1,
  gng.sigma.diff.cutoff = NULL, gng.mu.diff.cutoff = NULL,
  gng.var.thres = 1e2, gng.min.sd = NULL,
```

```
inudge.K = 2, inudge.weights = NULL, inudge.weights.cutoff = -1.345,
inudge.pi = NULL, inudge.mu = NULL,
inudge.sigma = NULL, inudge.tol = 1e-05, inudge.max.iter = 2000,
inudge.z = NULL, inudge.rep = 15, inudge.fdr.cutoff = 0.1,
inudge.sigma.diff.cutoff = NULL, inudge.mu.diff.cutoff = NULL,
inudge.var.thres = 1e2, inudge.min.sd = NULL,
nudge.z = NULL, nudge.tol = 1e-05, nudge.max.iter = 2000,
nudge.mu = NULL, nudge.sigma = NULL, nudge.rep = 15,
nudge.fdr.cutoff = 0.1, nudge.weights = NULL, nudge.weights.cutoff = -1.345,
nudge.pi = NULL)
```

Arguments

data	an **R list** of vector of normalized difference (log ratios). Each element can correspond to a particular chromosome. User can construct their own list containing only the chromosome(s) they want to analyze.
avg	optional **R list** of vector of mean data (or log intensities). Each element can correspond to a particular chromosome in data. Only required when any one of huber weight (lower, upper or full) is selected.
gng.K	optional maximum number of normal component that will be fitted in GNG model. For example: gng.K=2 will fit a model with 1 and 2 normal components and select the best k.
gng.weights	optional weights to be used for robust fitting. Can be a matrix the same length as data with each row correspond to weights to be used in each repetition or a character description of the huber-type method to be employed: "lower" - only value below cutoff are weighted,\ "upper" - only value above cutoff are weighted,\ "full" - both values above and below the cutoff are weighted,\ If selected, mean of data (avg) is required.
gng.weights.cutoff	optional cutoff to be used with the Huber weighting scheme.
gng.pi	optional matrix containing initial estimates for proportion of the GNG mixture components. Each row is the initial pi to be used in each repetition. Each row must have gng.K+2 entries. The first and last entries are for the estimates of negative and positive exponentials, respectively. The middle k entries are for normal components.
gng.mu	optional matrix containing initial estimates of the Gaussian means in GNG model. Each row is the initial means to be used in each repetition. Each row must have gng.K entries.
gng.sigma	optional maxtrix containing initial estimates of the Gaussian standard deviation in GNG model. Each row is the initial means to be used in each repetition. Each row must have gng.K entries.
gng.beta	optional maxtrix containing initial estimates for the shape parameter in exponential components in GNG model. Each row is the initial beta's to be used in each repetition. Each row must have 2 entries, one for negative exponential followed by another for positive exponential.
gng.tol	optional threshold for convergence for EM algorithm to estimate GNG parameters.

`gng.max.iter` optional maximum number of iterations for EM algorithm to estimate GNG parameters.

`gng.th` optional 2-column matrix of threshold for the two location exponential components. First column is the initial estimates for negative exponential and the second column is the initial estimates for positive exponential.

`gng.rep` optional number of times to repeat the GNG parameter estimation using different starting estimates.

`gng.fdr.cutoff` optional cut-off for local *fdr* for classifying regions into differential and non-differential using GNG mixture.

`gng.sigma.diff.cutoff`
optional cut-off for sigma of the normal component in GNG to be declared as representing differential. For example: gng.sigma.diff.cutoff = 2 then if a normal component has sigma > 2 then this component is considered as differential component. Default = (1.5*iqr(data)-gng$mu)/2. Where gng$mu is mean of non-differential normal components in iNUDGE.

`gng.mu.diff.cutoff`
optional cut-off for mu of the normal component in GNG to be declared as representing differential. For example: gng.mu.diff.cutoff = 2 then if a normal component has mean > 2 then this component is considered as differential component.

`gng.var.thres` optional threshold to detect huge imbalance in variance. max(gng.variance)/min(gng.variance) <= gng.var.thres.

`gng.min.sd` optional threshold to detect very small sigma. all normal components in GNG model has to have sigma > gng.min.sd. Default = 0.1 * sd(data)

`inudge.K` optional maximum number of normal component that will be fitted in iNUDGE model. For example: inudge.K=2 will fit a model with 1 and 2 normal components and select the best k.

`inudge.weights` optional weights to be used for robust fitting. Can be a matrix the same length as data with each row correspond to weights to be used in each repetition or a character description of the huber-type method to be employed: "lower" - only value below cutoff are weighted,\ "upper" - only value above cutoff are weighted,\ "full" - both values above and below the cutoff\ are weighted, any other character - "lower" are used (default). \ If selected, mean of data (avg) is required.

`inudge.weights.cutoff`
optional cutoff to be used with the Huber weighting scheme.

`inudge.pi` optional matrix of initial estimates for proportion of the iNUDGE mixture components. Each row correspond to the intial proportion to be used in each repetition. Each row must have inudge.K+1 entries corresponding to proportion of negative exponential, proportion of k-normal and proportion of exponential, respectively.

`inudge.mu` optional maxtrix of initial estimates of the Gaussian means in iNUDGE model. Each row correspond to the intial means to be used in each repetition. Each row must have inudge.K entries.

`inudge.sigma` optional matrix of initial estimates for Gaussian standard deviation in iNUDGE model. Each row correspond to the intial means to be used in each repetition. Each row must have inudge.K entries.

`inudge.tol` optional threshold for convergence for EM algorithm to estimate iNUDGE parameters.

`inudge.max.iter` optional maximum number of iterations for EM algorithm to estimate iNUDGE parameters.

`inudge.z` optional 2-column matrix with each row giving initial estimate of probability of the region being non-differential and a starting estimate for the probability of the region being differential. Each row must sum to 1. Number of row must be equal to data length.

`inudge.rep` optional number of times to repeat the iNUDGE parameter estimation using different starting estimates.

`inudge.fdr.cutoff` optional cut-off for local *fdr* for classifying regions into differential and non-differential based on iNUDGE mixture.

`inudge.sigma.diff.cutoff` optional cut-off for sigma of the normal component in GNG to be declared as representing differential. For example: gng.sigma.diff.cutoff = 2 then if a normal component has sigma > 2 then this component is considered as differential component. Default = (1.5*iqr(data)-inudge$mu^)/2. Where inudge$mu^ is mean of non-differential normal components in iNUDGE.

`inudge.mu.diff.cutoff` optional cut-off for mu of the normal component in GNG to be declared as representing differential. For example: gng.mu.diff.cutoff = 2 then if a normal component has mean > 2 then this component is considered as differential component.

`inudge.var.thres` optional threshold to detect huge imbalance in variance. max(inudge.variance)/min(inudge.variance) <= inudge.var.thres.

`inudge.min.sd` optional threshold to detect very small sigma. all normal components in iNUDGE model has to have sigma > inudge.min.sd. Default = 0.1 * sd(data)

`nudge.z` optional 2-column matrix with each row giving initial estimate of probability of the region being non-differential and a starting estimate for the probability of the region being differential. Each row must sum to 1. Number of row must be equal to data length.

`nudge.tol` optional threshold for convergence for EM algorithm to estimate NUDGE parameters.

`nudge.max.iter` optional maximum number of iterations for EM algorithm to estimate iNUDGE parameters.

`nudge.mu` optional maxtrix of initial estimates of the Gaussian means in NUDGE model. Each row correspond to the intial means to be used in each repetition. Each row must have 1 entry.

`nudge.sigma` optional initial estimates of the Gaussian standard deviation in NUDGE model. Each row correspond to the intial standard deviation to be used in each repetition. Each row must have 1 entry.

`nudge.rep` optional number of times to repeat the NUDGE parameter estimation using different starting estimates.

`nudge.fdr.cutoff` optional cut-off for local *fdr* for classifying regions into differential and non-differential based on NUDGE mixture.

`nudge.weights` optional weights to be used for robust fitting. Can be a matrix the same length as data with each row correspond to weights to be used in each repetition or a character description of the huber-type method to be employed: "lower" - only value below cutoff are weighted,\ "upper" - only value above cutoff are weighted,\ "full" - both values above and below the cutoff\ are weighted, any other character - "lower" are used (default). \ If selected, mean of data (avg) is required.

`nudge.weights.cutoff` optional cutoff to be used with the Huber weighting scheme.

`nudge.pi` optional initial estimates for proportion of the NUDGE mixture components. Each row is the initial pi to be used in each repetition. Each row must have 2 entries: proportion of uniform and proportion of normal components, respectively .

Details

After normalization, a Gamma-Normal(k)-Gamma (GNG), a Uniform-Normal(k) (iNUDGE) and a Uniform-Normal (NUDGE) mixture are fitted to the data. Two-phase selection method is used to choose the best model. The (k)-normal component can represent either differential regions or non-differential regions depending on their location and shape, making the model more robust to different underlying distributions. The exponential or uniform represents differential sites. Local *(fdr)* is computed from the best fitted model. Parameters estimation is performed using EM algorithm.

Value

A list with 4 components (i.e. best, gng, inudge and nudge) which in itself is another list containing the estimated parameters of each model fitted correspondingly. "best" lists the model chosen as the best overall model, i.e. if the best model is inudge then best$name = "iNUDGE" and its content is the same as inudge. Thus, depending on the model, the components are:

`name` the name of the model "GNG", "iNUDGE","NUDGE" where GNG: normal(k)-exponential (a special case of gamma), iNUDGE: normal(k)-uniform, or NUDGE: normal-uniform models

`pi` a vector of estimated proportion of each components in the model

`mu` a vector of estimated Gaussian means for k-normal components.

`sigma` a vector of estimated Gaussian standard deviation for k-normal components.

`beta` a vector of estimated exponential shape values. Only available in gng.

th1	negative location parameter used to fit the negative exponential model. Only available in gng.
th2	positive location parameter used to fit the positive exponential model. Only available in gng.
a	the minimum value of the normalized data. Only available in (i)nudge.
b	the maximum value of the normalized data. Only available in (i)nudge.
K	the number of normal components in the corresponding mixture model. For inudge, K=1.
loglike	the log likelihood for the fitted mixture model.
iter	the number of iterations run by the EM algorithm until either convergence or iteration limit was reached.
fdr	the local false discover rate estimated based on the corresponding model.
class	a vector of classifications for the observations in data. A classification of 0 denotes that the regions could not be classified as differential with fdr < <model>.fdr.cutoff, 1 denotes differential.
diffPiIdx	a vector of index of the normal components that are defined as capturing differential regions based on their shape and locations.
phi	a vector of estimated mixture function
mu.diff.cutoff	normal component with mean > mu.diff.cutoff will be used to represent differential component.
sigma.diff.cutoff	normal component with standard deviation > sigma.diff.cutoff will be used to represent differential component.

Author(s)

Cenny Taslim <taslim.2@osu.edu>, with contributions from Abbas Khalili <khalili@stat.ubc.ca>, Dustin Potter <potterdp@gmail.com>, and Shili Lin <shili@stat.osu.edu>

See Also

gng.fit, inudge.fit, nudge.fit

Examples

```
library(DIME)
# generate simulated datasets with underlying exponential-normal components
N1 <- 1500; N2 <- 500; K <- 4; rmu <- c(-2.25,1.50); rsigma <- c(1,1);
rpi <- c(.05,.45,.45,.05); rbeta <- c(12,10);
set.seed(1234)
chr1 <- c(-rgamma(ceiling(rpi[1]*N1),shape = 1,scale = rbeta[1]),
  rnorm(ceiling(rpi[2]*N1),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N1),rmu[2],rsigma[2]),
  rgamma(ceiling(rpi[4]*N1),shape = 1,scale = rbeta[2]));
chr2 <- c(-rgamma(ceiling(rpi[1]*N2),shape = 1,scale = rbeta[1]),
  rnorm(ceiling(rpi[2]*N2),rmu[1],rsigma[1]),
```

```
  rnorm(ceiling(rpi[3]*N2),rmu[2],rsigma[2]),
  rgamma(ceiling(rpi[4]*N2),shape = 1,scale = rbeta[2]));
chr3 <- c(-rgamma(ceiling(rpi[1]*N2),shape = 1,scale = rbeta[1]),
  rnorm(ceiling(rpi[2]*N2),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N2),rmu[2],rsigma[2]),
  rgamma(ceiling(rpi[4]*N2),shape = 1,scale = rbeta[2]));
# analyzing only chromosome 1 and chromosome 3
data <- list(chr1,chr3);

# run DIME
set.seed(1234)
test <- DIME(data,gng.max.iter=10,gng.rep=1,inudge.max.iter=10,inudge.rep=1,
 nudge.max.iter=10,nudge.rep=1)

# Getting the best fitted model (parameters)
test$best$name # name of the best fitted model
test$best$pi # estimated proportion of each component in the best model
test$best$mu # estimated mean of the normal component(s) the best model
# estimated standard deviation of the normal component(s) in best model
test$best$sigma
# estimated shape parameter of the exponential components in best model
test$best$beta
# class indicator inferred using best model chosen. 1 means differential, 0 o.w
bestClass = test$best$class

# plot best model
DIME.plot.fit(data,test)

# Eg. getting Gaussian mean from iNUDGE model
test$inudge$mu

# Eg. getting Gaussian mean from NUDGE model
test$nudge$mu

# Eg. getting parameters from GNG model
test$gng$mu

# provide initial estimates means of Gaussian in GNG model
test <- DIME(data,gng.max.iter=5,gng.rep=1,inudge.max.iter=5,inudge.rep=1,
 nudge.max.iter=5,nudge.rep=1,gng.K=2,gng.mu=rbind(c(1,2)))
```

`DIME.classify` *Classification Based on The Best Model*

Description

Classifies observed data into differential and non-differential groups based on the model selected as the best fit to the observed data.

Usage

```
DIME.classify(data, obj, obj.cutoff = 0.1, obj.sigma.diff.cutoff = NULL,
  obj.mu.diff.cutoff = NULL)
```

Arguments

data	an **R list** of vector of normalized intensities (counts). Each element can correspond to particular chromosome. User can construct their own list containing only the chromosome(s) they want to analyze.
obj	a list object returned by DIME function.
obj.cutoff	optional local *fdr* cutoff for classifying data into differential and non-differential groups based on the best mixture model.
obj.sigma.diff.cutoff	optional cut-off for standard deviation of the normal component in the best model to be declared as representing differential.
obj.mu.diff.cutoff	optional cut-off for standard deviation of the normal component in the best model to be declared as representing differential.

Value

A list object passed as input with additional element $class containing vector of classifications for all the observations in data. A classification of 1 denotes that the data is classified as differential with fdr < obj.cutoff.

Author(s)

Cenny Taslim <taslim.2@osu.edu>, with contributions from Abbas Khalili <khalili@stat.ubc.ca>, Dustin Potter <potterdp@gmail.com>, and Shili Lin <shili@stat.osu.edu>

See Also

DIME

Examples

```
library(DIME)
# generate simulated datasets with underlying exponential-normal components
N1 <- 1500; N2 <- 500; K <- 4; rmu <- c(-2.25,1.50); rsigma <- c(1,1);
rpi <- c(.05,.45,.45,.05); rbeta <- c(12,10);
set.seed(1234)
chr1 <- c(-rgamma(ceiling(rpi[1]*N1),shape = 1,scale = rbeta[1]),
  rnorm(ceiling(rpi[2]*N1),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N1),rmu[2],rsigma[2]),
  rgamma(ceiling(rpi[4]*N1),shape = 1,scale = rbeta[2]));
chr2 <- c(-rgamma(ceiling(rpi[1]*N2),shape = 1,scale = rbeta[1]),
  rnorm(ceiling(rpi[2]*N2),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N2),rmu[2],rsigma[2]),
  rgamma(ceiling(rpi[4]*N2),shape = 1,scale = rbeta[2]));
```

```
chr3 <- c(-rgamma(ceiling(rpi[1]*N2),shape = 1,scale = rbeta[1]),
  rnorm(ceiling(rpi[2]*N2),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N2),rmu[2],rsigma[2]),
  rgamma(ceiling(rpi[4]*N2),shape = 1,scale = rbeta[2]));
# analyzing only chromosome 1 and chromosome 3
data <- list(chr1,chr3);

# run DIME with small maximum iteration and repetitions
set.seed(1234);
test <- DIME(data,gng.max.iter=10,gng.rep=1,inudge.max.iter=10,inudge.rep=1,
 nudge.max.iter=10,nudge.rep=1)
# get classification based on inudge
test$inudge <- DIME.classify(data,test$inudge,obj.cutoff=0.1);
# vector of classification. 1 represents differential, 0 denotes non-differential
inudgeClass <- test$inudge$class;
```

`DIME.plot.fit` *Plot Best Model Goodness of Fit*

Description

Plot the best mixture model fitted using `DIME` along with their estimated individual components, superimposed on the histogram of the observation data. This plot shows how good the fit of the estimated model to the data.

Usage

```
DIME.plot.fit(data, obj, ...)
```

Arguments

`data`	an **R list** of vector of normalized intensities (counts). Each element can correspond to particular chromosome. User can construct their own list containing only the chromosome(s) they want to analyze.
`obj`	a list object returned by `DIME` function.
`...`	additional graphical arguments to be passed to methods (see `par`).

Details

The components representing differential data are denoted by asterisk (*) symbol on the plot legend.

Author(s)

Cenny Taslim <taslim.2@osu.edu>, with contributions from Abbas Khalili <khalili@stat.ubc.ca>, Dustin Potter <potterdp@gmail.com>, and Shili Lin <shili@stat.osu.edu>

See Also

DIME, gng.plot.fit,inudge.plot.fit

Examples

```
library(DIME);
# generate simulated datasets with underlying exponential-normal components
N1 <- 1500; N2 <- 500; K <- 4; rmu <- c(-2.25,1.50); rsigma <- c(1,1)
rpi <- c(.05,.45,.45,.05); rbeta <- c(12,10)
set.seed(1234)
chr1 <- c(-rgamma(ceiling(rpi[1]*N1),shape = 1,scale = rbeta[1]),
  rnorm(ceiling(rpi[2]*N1),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N1),rmu[2],rsigma[2]),
  rgamma(ceiling(rpi[4]*N1),shape = 1,scale = rbeta[2]));
chr3 <- c(-rgamma(ceiling(rpi[1]*N2),shape = 1,scale = rbeta[1]),
  rnorm(ceiling(rpi[2]*N2),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N2),rmu[2],rsigma[2]),
  rgamma(ceiling(rpi[4]*N2),shape = 1,scale = rbeta[2]));
data <- list(chr1,chr3);

# run DIME with small maximum iterations and repetitions
set.seed(1234);
test <- DIME(data,gng.max.iter=10,gng.rep=1,inudge.max.iter=10,inudge.rep=1,
 nudge.max.iter=10,nudge.rep=1);

# plot best model
DIME.plot.fit(data,test);
```

---

gng.classify *Classification Based on GNG Model*

Description

Classifies observed data into differential and non-differential groups based on GNG model.

Usage

```
gng.classify(data, obj, obj.cutoff = 0.1, obj.sigma.diff.cutoff = NULL,
  obj.mu.diff.cutoff = NULL)
```

Arguments

data	an **R list** of vector of normalized intensities (counts). Each element can correspond to particular chromosome. User can construct their own list containing only the chromosome(s) they want to analyze.
obj	a list object returned by gng.fit function.
obj.cutoff	optional local *fdr* cutoff for classifying data into differential and non-differential groups based on GNG model.

`obj.sigma.diff.cutoff`
: optional cut-off for standard deviation of the normal component in the best model to be declared as representing differential.

`obj.mu.diff.cutoff`
: optional cut-off for standard deviation of the normal component in the best model to be declared as representing differential.

Value

A list object passed as input with additional element $class containing vector of classifications for all the observations in data. A classification of 1 denotes that the data is classified as differential with fdr < obj.cutoff.

`mu.diff.cutoff` normal component with mean > mu.diff.cutoff will be used to represent differential component.

`sigma.diff.cutoff`
: normal component with standard deviation > sigma.diff.cutoff will be used to represent differential component.

Author(s)

Cenny Taslim <taslim.2@osu.edu>, with contributions from Abbas Khalili <khalili@stat.ubc.ca>, Dustin Potter <potterdp@gmail.com>, and Shili Lin <shili@stat.osu.edu>

See Also

gng.fit

Examples

```
library(DIME);
# generate simulated datasets with underlying exponential-normal components
N1 <- 1500; N2 <- 500; K <- 4; rmu <- c(-2.25,1.50); rsigma <- c(1,1);
rpi <- c(.05,.45,.45,.05); rbeta <- c(12,10);
set.seed(1234);
chr1 <- c(-rgamma(ceiling(rpi[1]*N1),shape = 1,scale = rbeta[1]),
  rnorm(ceiling(rpi[2]*N1),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N1),rmu[2],rsigma[2]),
  rgamma(ceiling(rpi[4]*N1),shape = 1,scale = rbeta[2]));
chr2 <- c(-rgamma(ceiling(rpi[1]*N2),shape = 1,scale = rbeta[1]),
  rnorm(ceiling(rpi[2]*N2),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N2),rmu[2],rsigma[2]),
  rgamma(ceiling(rpi[4]*N2),shape = 1,scale = rbeta[2]));
chr3 <- c(-rgamma(ceiling(rpi[1]*N2),shape = 1,scale = rbeta[1]),
  rnorm(ceiling(rpi[2]*N2),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N2),rmu[2],rsigma[2]),
  rgamma(ceiling(rpi[4]*N2),shape = 1,scale = rbeta[2]));
# analyzing only chromosome 1 and chromosome 3
data <- list(chr1,chr3);

# fit GNG model with 2 normal components
```

```
test <- gng.fit(data, K = 2);
# vector of classification. 1 represents differential, 0 denotes non-differential
gngClass <- test$class;
```

---

gng.fit *Function for Fitting GNG model parameters*

Description

Function to estimate parameters for GNG model, mixture of exponential and *k*-normal. Parameters are estimated using EM algorithm.

Usage

```
gng.fit(data, avg = NULL, K = 2, weights = NULL, weights.cutoff = -1.345,
  pi = NULL, mu = NULL, sigma = NULL, beta = NULL, tol = 1e-05,
  max.iter = 2000, th = NULL)
```

Arguments

data	an **R list** of vector of normalized intensities (counts). Each element can correspond to particular chromosome. User can construct their own list containing only the chromosome(s) they want to analyze.
avg	optional vector of mean data (or log intensities). Only required when any one of huber weight (lower, upper or full) is selected.
K	optional number of normal component that will be fitted in GNG model.
weights	optional weights to be used for robust fitting. Can be a matrix the same length as data, or a character description of the huber weight method to be employed: "lower" - only value below weights.cutoff are weighted,\ "upper" - only value above weights.cutoff are weighted,\ "full" - both values above and below weights.cutoff are weighted,\ If selected, mean of data (avg) is required.
weights.cutoff	optional cutoff to be used with the Huber weighting scheme.
pi	optional vector containing initial estimates for proportion of the GNG mixture components. The first and last entries are for the estimates of negative and positive exponentials, respectively. The middle k entries are for normal components.
mu	optional vector containing initial estimates of the Gaussian means in GNG model.
sigma	optional vector containing initial estimates of the Gaussian standard deviation in GNG model. Must have K entries.
beta	optional vector containing initial estimates for the shape parameter in exponential components in GNG model. Must have 2 entries, one for negative exponential the other for positive exponential components.
tol	optional threshold for convergence for EM algorithm to estimate GNG parameters.
max.iter	optional maximum number of iterations for EM algorithm to estimate GNG parameters.
th	optional location parameter used to fit the negative and positive exponential model.

Value

A list of object:

name	the name of the model "GNG"
pi	a vector of estimated proportion of each components in the model
mu	a vector of estimated Gaussian means for k-normal components.
sigma	a vector of estimated Gaussian standard deviation for k-normal components.
beta	a vector of estimated exponential shape values.
th1	negative location parameter used to fit the negative exponential model.
th2	positive location parameter used to fit the positive exponential model.
K	the number of normal components in the corresponding mixture model.
loglike	the log likelihood for the fitted mixture model.
iter	the actual number of iterations run by the EM algorithm.
fdr	the local false discover rate estimated based on GNG model.
phi	a matrix of estimated GNG mixture component function.
AIC	Akaike Information Criteria.
BIC	Bayesian Information Criteria.

Author(s)

Cenny Taslim <taslim.2@osu.edu>, with contributions from Abbas Khalili <khalili@stat.ubc.ca>, Dustin Potter <potterdp@gmail.com>, and Shili Lin <shili@stat.osu.edu>

See Also

DIME, inudge.fit, nudge.fit

Examples

```
library(DIME)
# generate simulated datasets with underlying exponential-normal components
N1 <- 1500; N2 <- 500; K <- 4; rmu <- c(-2.25,1.50); rsigma <- c(1,1);
rpi <- c(.05,.45,.45,.05); rbeta <- c(12,10);
set.seed(1234)
chr1 <- c(-rgamma(ceiling(rpi[1]*N1),shape = 1,scale = rbeta[1]),
  rnorm(ceiling(rpi[2]*N1),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N1),rmu[2],rsigma[2]),
  rgamma(ceiling(rpi[4]*N1),shape = 1,scale = rbeta[2]));
chr2 <- c(-rgamma(ceiling(rpi[1]*N2),shape = 1,scale = rbeta[1]),
  rnorm(ceiling(rpi[2]*N2),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N2),rmu[2],rsigma[2]),
  rgamma(ceiling(rpi[4]*N2),shape = 1,scale = rbeta[2]));
chr3 <- c(-rgamma(ceiling(rpi[1]*N2),shape = 1,scale = rbeta[1]),
  rnorm(ceiling(rpi[2]*N2),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N2),rmu[2],rsigma[2]),
  rgamma(ceiling(rpi[4]*N2),shape = 1,scale = rbeta[2]));
```

```
# analyzing only chromosome 1 and chromosome 3
data <- list(chr1,chr3);

# fit GNG model with 2 normal components
test <- gng.fit(data, K = 2);

# Getting the best fitted GNG model (parameters)
test$pi # estimated proportion of each component in GNG
test$mu # estimated mean of the normal component(s) GNG
# estimated standard deviation of the normal component(s) in GNG
test$sigma
# estimated shape parameter of the exponential components in best model
test$beta
```

---

gng.plot.comp *Plot GNG Individual Components*

Description

Plot each estimated individual components of GNG model (mixture of exponential and *k*-normal) fitted using gng.fit.

Usage

```
gng.plot.comp(data, obj, new.plot = TRUE, legpos = NULL, xlim=NULL,
  ylim=NULL, xlab=NULL, ylab=NULL, main=NULL,lwd=NULL,...)
```

Arguments

data	an **R list** of vector of normalized intensities (counts). Each element can correspond to a particular chromosome. User can construct their own list containing only the chromosome(s) they want to analyze.
obj	a list object returned by gng.fit function.
new.plot	optional logical variable on whether to create new plot.
legpos	optional vector of (x,y) location for the legend position
xlim	optional x-axis limit (see par).
ylim	optional y-axis limit (see par).
xlab	optional x-axis label (see par).
ylab	optional y-axis label (see par).
main	optional plot title (see par).
lwd	optional line width for lines in the plot (see par).
...	additional graphical arguments to be passed to methods (see par).

Details

The components representing differential data are denoted by asterisk (*) symbol on the plot legend.

Author(s)

Cenny Taslim <taslim.2@osu.edu>, with contributions from Abbas Khalili <khalili@stat.ubc.ca>, Dustin Potter <potterdp@gmail.com>, and Shili Lin <shili@stat.osu.edu>

See Also

gng.plot.mix, gng.plot.comp, gng.plot.fit, gng.plot.qq, DIME.plot.fit, inudge.plot.fit.

Examples

```
library(DIME);
# generate simulated datasets with underlying exponential-normal components
N1 <- 1500; N2 <- 500; K <- 4; rmu <- c(-2.25,1.50); rsigma <- c(1,1);
rpi <- c(.05,.45,.45,.05); rbeta <- c(12,10);
set.seed(1234);
chr1 <- c(-rgamma(ceiling(rpi[1]*N1),shape = 1,scale = rbeta[1]),
  rnorm(ceiling(rpi[2]*N1),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N1),rmu[2],rsigma[2]),
  rgamma(ceiling(rpi[4]*N1),shape = 1,scale = rbeta[2]));
chr2 <- c(-rgamma(ceiling(rpi[1]*N2),shape = 1,scale = rbeta[1]),
  rnorm(ceiling(rpi[2]*N2),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N2),rmu[2],rsigma[2]),
  rgamma(ceiling(rpi[4]*N2),shape = 1,scale = rbeta[2]));
chr3 <- c(-rgamma(ceiling(rpi[1]*N2),shape = 1,scale = rbeta[1]),
  rnorm(ceiling(rpi[2]*N2),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N2),rmu[2],rsigma[2]),
  rgamma(ceiling(rpi[4]*N2),shape = 1,scale = rbeta[2]));
# analyzing only chromosome 1 and chromosome 3
data <- list(chr1,chr3);

# Fitting a GNG model with 2-normal component
bestGng <- gng.fit(data,K=2);

# plot individual components of GNG
gng.plot.comp(data,bestGng);
# plot mixture component on top of the individual components plot
gng.plot.mix(bestGng,resolution=1000,new.plot=FALSE);
```

---

gng.plot.fit *Plot GNG Goodness of Fit*

Description

Plot the estimated GNG mixture model fitted using gng.fit along with it's estimated individual components, superimposed on the histogram of the observation data. This plot shows how good the fit of the estimated model to the data.

Usage

```
gng.plot.fit(data, obj, resolution = 100, breaks = 100, legpos = NULL,
  xlim = NULL, main=NULL, ...)
```

Arguments

data	an **R list** of vector of normalized intensities (counts). Each element can correspond to a particular chromosome. User can construct their own list containing only the chromosome(s) they want to analyze.
obj	a list object returned by gng.fit function.
resolution	optional bandwidth used to estimate the density function. Higher number smoother curve.
breaks	optional see hist, breaks parameters for histogram plot.
legpos	optional vector of (x,y) location for the legend position
xlim	optional x-axis limit (see par).
main	optional plot title (see par).
...	additional graphical arguments to be passed to methods (see par).

Details

The components representing differential data are denoted by asterisk (*) symbol on the plot legend.

See Also

gng.plot.comp, gng.plot.mix, hist

Examples

```
library(DIME)
# generate simulated datasets with underlying exponential-normal components
N1 <- 1500; N2 <- 500; K <- 4; rmu <- c(-2.25,1.50); rsigma <- c(1,1);
rpi <- c(.05,.45,.45,.05); rbeta <- c(12,10);
set.seed(1234);
chr1 <- c(-rgamma(ceiling(rpi[1]*N1),shape = 1,scale = rbeta[1]),
  rnorm(ceiling(rpi[2]*N1),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N1),rmu[2],rsigma[2]),
  rgamma(ceiling(rpi[4]*N1),shape = 1,scale = rbeta[2]));
chr2 <- c(-rgamma(ceiling(rpi[1]*N2),shape = 1,scale = rbeta[1]),
  rnorm(ceiling(rpi[2]*N2),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N2),rmu[2],rsigma[2]),
  rgamma(ceiling(rpi[4]*N2),shape = 1,scale = rbeta[2]));
chr3 <- c(-rgamma(ceiling(rpi[1]*N2),shape = 1,scale = rbeta[1]),
  rnorm(ceiling(rpi[2]*N2),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N2),rmu[2],rsigma[2]),
  rgamma(ceiling(rpi[4]*N2),shape = 1,scale = rbeta[2]));
# analyzing only chromosome 1 and chromosome 3
data <- list(chr1,chr3);
```

```
# Fitting a GNG model only with 2-normal components
bestGng <- gng.fit(data,K=2);

# Goodness of fit plot
gng.plot.fit(data,bestGng);
```

---

gng.plot.mix *Plot GNG Mixture Component Function*

Description

Plot each estimated individual components of GNG mixture model fitted using gng.fit.

Usage

```
gng.plot.mix(obj, amplify = 1, resolution = 100, new.plot = TRUE, ...)
```

Arguments

obj	a list object returned by gng.fit function.
amplify	optional scaling factor for visualization purposes.
resolution	optional bandwidth used to estimate the density function. Higher number makes a smoother curve.
new.plot	optional logical variable on whether to create new plot.
...	additional graphical arguments to be passed to methods (see par).

Author(s)

Cenny Taslim <taslim.2@osu.edu>, with contributions from Abbas Khalili <khalili@stat.ubc.ca>, Dustin Potter <potterdp@gmail.com>, and Shili Lin <shili@stat.osu.edu>

See Also

gng.plot.mix, gng.plot.comp, gng.plot.fit, gng.plot.qq, DIME.plot.fit, inudge.plot.fit.

Examples

```
library(DIME)
# generate simulated datasets with underlying exponential-normal components
N1 <- 1500; N2 <- 500; K <- 4; rmu <- c(-2.25,1.50); rsigma <- c(1,1);
rpi <- c(.05,.45,.45,.05); rbeta <- c(12,10);
set.seed(1234);
chr1 <- c(-rgamma(ceiling(rpi[1]*N1),shape = 1,scale = rbeta[1]),
  rnorm(ceiling(rpi[2]*N1),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N1),rmu[2],rsigma[2]),
  rgamma(ceiling(rpi[4]*N1),shape = 1,scale = rbeta[2]));
chr2 <- c(-rgamma(ceiling(rpi[1]*N2),shape = 1,scale = rbeta[1]),
  rnorm(ceiling(rpi[2]*N2),rmu[1],rsigma[1]),
```

```
  rnorm(ceiling(rpi[3]*N2),rmu[2],rsigma[2]),
  rgamma(ceiling(rpi[4]*N2),shape = 1,scale = rbeta[2]));
chr3 <- c(-rgamma(ceiling(rpi[1]*N2),shape = 1,scale = rbeta[1]),
  rnorm(ceiling(rpi[2]*N2),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N2),rmu[2],rsigma[2]),
  rgamma(ceiling(rpi[4]*N2),shape = 1,scale = rbeta[2]));
# analyzing only chromosome 1 and chromosome 3
data <- list(chr1,chr3);

# Fitting a GNG model only
bestGng <- gng.fit(data,K=2);

# Plot the estimated GNG model imposed on the histogram of the observed data
hist(unlist(data),freq=FALSE,breaks=100,xlim=c(-20,20))
gng.plot.mix(bestGng,resolution=1000,new.plot=FALSE,col="red");
```

`gng.plot.qq` *QQ-plot of GNG model vs. observed data*

Description

Produces a QQ-plot for visual inspection of quality of fit with regards to the exponential Gaussian (GNG) mixture model estimated using the function `gng.fit`

Usage

```
gng.plot.qq(data, obj, resolution=10, xlab=NULL, ylab=NULL,
  main=NULL, pch=NULL,...)
```

Arguments

`data`	an **R list** of vector of normalized intensities (counts). Each element can correspond to a particular chromosome. User can construct their own list containing only the chromosome(s) they want to analyze.
`obj`	a list object returned by `gng.fit` function.
`resolution`	optional number of points used to sample the estimated density function.
`xlab`	optional x-axis label (see `par`).
`ylab`	optional y-axis label (see `par`).
`main`	optional plot title (see `par`).
`pch`	optional plotting symbol to use (see `par`).
`...`	additional graphical arguments to be passed to methods (see `par`).

Author(s)

Cenny Taslim <taslim.2@osu.edu>, with contributions from Abbas Khalili <khalili@stat.ubc.ca>, Dustin Potter <potterdp@gmail.com>, and Shili Lin <shili@stat.osu.edu>

See Also

gng.fit, gng.plot.fit

Examples

```
library(DIME)
# generate simulated datasets with underlying exponential-normal components
N1 <- 1500; N2 <- 500; K <- 4; rmu <- c(-2.25,1.50); rsigma <- c(1,1);
rpi <- c(.05,.45,.45,.05); rbeta <- c(12,10);
set.seed(1234);
chr1 <- c(-rgamma(ceiling(rpi[1]*N1),shape = 1,scale = rbeta[1]),
  rnorm(ceiling(rpi[2]*N1),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N1),rmu[2],rsigma[2]),
  rgamma(ceiling(rpi[4]*N1),shape = 1,scale = rbeta[2]));
chr2 <- c(-rgamma(ceiling(rpi[1]*N2),shape = 1,scale = rbeta[1]),
  rnorm(ceiling(rpi[2]*N2),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N2),rmu[2],rsigma[2]),
  rgamma(ceiling(rpi[4]*N2),shape = 1,scale = rbeta[2]));
chr3 <- c(-rgamma(ceiling(rpi[1]*N2),shape = 1,scale = rbeta[1]),
  rnorm(ceiling(rpi[2]*N2),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N2),rmu[2],rsigma[2]),
  rgamma(ceiling(rpi[4]*N2),shape = 1,scale = rbeta[2]));
# analyzing only chromosome 1 and chromosome 3
data <- list(chr1,chr3);

# Fitting a GNG model only
bestGng <- gng.fit(data,K=2);

# QQ-plot
gng.plot.qq(data,bestGng)
```

---

huber *Huber's weight function*

Description

A weight functions used to downweigh outliers.

Usage

```
huber(input, co, shape = c("full", "lower", "upper"))
```

Arguments

input	an **R list** of vector of normalized mean (log intensities).
co	cutoff used in determining weights.
shape	parameter determining which outliers are weighted: "full" - both values above and below -threshold are downweighted;\ "lower" - only values below threshold are downweighted; \ "upper" - only values above threshold are downweighted.

Value

a vector of weights.

Author(s)

Dustin Potter

References

Huber, P. J. (1981) Robust Statistics. John Wiley & Sons

`inudge.classify` *Classification Based on iNUDGE Model*

Description

Classifies observed data into differential and non-differential groups based on iNUDGE model.

Usage

```
inudge.classify(data, obj, obj.cutoff = 0.1, obj.sigma.diff.cutoff = NULL,
  obj.mu.diff.cutoff = NULL)
```

Arguments

`data`	an **R list** of vector of normalized intensities (counts). Each element can correspond to particular chromosome. User can construct their own list containing only the chromosome(s) they want to analyze.
`obj`	a list object returned by `inudge.fit` function.
`obj.cutoff`	optional local *fdr* cutoff for classifying data into differential and non-differential groups based on iNUDGE model.
`obj.sigma.diff.cutoff`	optional cut-off for standard deviation of the normal component in iNUDGE model to be designated as representing differential.
`obj.mu.diff.cutoff`	optional cut-off for standard deviation of the normal component in iNUDGE model to be designated as representing differential.

Value

A list object passed as input with additional element $class containing vector of classifications for all the observations in data. A classification of 1 denotes that the data is classified as differential with fdr < obj.cutoff.

`mu.diff.cutoff`	normal component with mean > mu.diff.cutoff was used to represent differential component.
`sigma.diff.cutoff`	normal component with standard deviation > sigma.diff.cutoff was used to represent differential component.

Author(s)

Cenny Taslim <taslim.2@osu.edu>, with contributions from Abbas Khalili <khalili@stat.ubc.ca>, Dustin Potter <potterdp@gmail.com>, and Shili Lin <shili@stat.osu.edu>

See Also

inudge.fit

Examples

```
library(DIME);
# generate simulated datasets with underlying uniform and 2-normal distributions
set.seed(1234);
N1 <- 1500; N2 <- 500; rmu <- c(-2.25,1.5); rsigma <- c(1,1);
rpi <- c(.10,.45,.45); a <- (-6); b <- 6;
chr4 <- list(c(-runif(ceiling(rpi[1]*N1),min = a,max =b),
  rnorm(ceiling(rpi[2]*N1),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N1),rmu[2],rsigma[2])));
chr9 <- list(c(-runif(ceiling(rpi[1]*N2),min = a,max =b),
  rnorm(ceiling(rpi[2]*N2),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N2),rmu[2],rsigma[2])));
# analyzing chromosome 4 and 9
data <- list(chr4,chr9);

# fit iNUDGE model with 2 normal components and maximum iterations = 20
set.seed(1234);
test <- inudge.fit(data, K = 2, max.iter=20);
# vector of classification. 1 represents differential, 0 denotes non-differential
inudgeClass <- test$class;
```

---

inudge.fit *Function for Fitting iNUDGE model parameters*

Description

Function to estimate parameters for NUDGE model, mixture of uniform and *k*-normal. Parameters are estimated using EM algorithm.

Usage

```
inudge.fit(data, avg = NULL, K = 2, weights = NULL, weights.cutoff = -1.345,
 pi = NULL, mu = NULL, sigma = NULL, tol = 1e-5, max.iter = 2000, z = NULL)
```

Arguments

data	an **R list** of vector of normalized intensities (counts). Each element can correspond to particular chromosome. User can construct their own list containing only the chromosome(s) they want to analyze.

avg	optional vector of mean data (or log intensities). Only required when any one of huber weight (lower, upper or full) is selected.
K	optional number of normal component that will be fitted in iNUDGE model.
weights	optional weights to be used for robust fitting. Can be a matrix the same length as data, or a character description of the huber weight method to be employed: "lower" - only value below weights.cutoff are weighted,\ "upper" - only value above weights.cutoff are weighted,\ "full" - both values above and below weights.cutoff are weighted,\ If selected, mean of data (avg) is required.
weights.cutoff	optional cutoff to be used with the Huber weighting scheme.
pi	optional vector containing initial estimates for proportion of the iNUDGE mixture components. The first entry is for the uniform component, the middle k entries are for normal components.
mu	optional vector containing initial estimates of the Gaussian means in iNUDGE model.
sigma	optional vector containing initial estimates of the Gaussian standard deviation in (i)NUDGE model. Must have K entries.
tol	optional threshold for convergence for EM algorithm to estimate iNUDGE parameters.
max.iter	optional maximum number of iterations for EM algorithm to estimate iNUDGE parameters.
z	optional 2-column matrix with each row giving initial estimate of probability of the region being non-differential and a starting estimate for the probability of the region being differential. Each row must sum to 1. Number of row must be equal to data length.

Value

A list of object:

name	the name of the model "iNUDGE"
pi	a vector of estimated proportion of each components in the model
mu	a vector of estimated Gaussian means for k-normal components.
sigma	a vector of estimated Gaussian standard deviation for k-normal components.
K	the number of normal components in the corresponding mixture model.
loglike	the log likelihood for the fitted mixture model.
iter	the actual number of iterations run by the EM algorithm.
fdr	the local false discover rate estimated based on iNUDGE model.
phi	a matrix of estimated iNUDGE mixture component function.
AIC	Akaike Information Criteria.
BIC	Bayesian Information Criteria.

Author(s)

Cenny Taslim <taslim.2@osu.edu>, with contributions from Abbas Khalili <khalili@stat.ubc.ca>, Dustin Potter <potterdp@gmail.com>, and Shili Lin <shili@stat.osu.edu>

See Also

DIME, gng.fit, nudge.fit

Examples

```
library(DIME);

# generate simulated datasets with underlying uniform and 2-normal distributions
set.seed(1234);
N1 <- 1500; N2 <- 500; rmu <- c(-2.25,1.5); rsigma <- c(1,1);
rpi <- c(.10,.45,.45); a <- (-6); b <- 6;
chr4 <- list(c(-runif(ceiling(rpi[1]*N1),min = a,max =b),
  rnorm(ceiling(rpi[2]*N1),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N1),rmu[2],rsigma[2])));
chr9 <- list(c(-runif(ceiling(rpi[1]*N2),min = a,max =b),
  rnorm(ceiling(rpi[2]*N2),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N2),rmu[2],rsigma[2])));
# analyzing chromosome 4 and 9
data <- list(chr4,chr9);

# fit iNUDGE model with 2 normal components and maximum iterations = 20
set.seed(1234);
test <- inudge.fit(data, K = 2, max.iter=20);

# Getting the best fitted iNUDGE model (parameters)
test$best$pi # estimated proportion of each component in iNUDGE
test$best$mu # estimated mean of the normal component(s) in iNUDGE
# estimated standard deviation of the normal component(s) in iNUDGE
test$best$sigma
```

---

inudge.plot.comp *Plot iNUDGE Individual Components*

Description

Plot each estimated individual components of iNUDGE model (mixture of uniform and *k*-normal) fitted using inudge.fit.

Usage

```
inudge.plot.comp(data, obj, new.plot = TRUE, legpos = NULL, xlim = NULL,
  ylim = NULL, xlab = NULL, ylab = NULL, main = NULL, lwd = NULL,...)
```

Arguments

data	an **R list** of vector of normalized intensities (counts). Each element can correspond to particular chromosome. User can construct their own list containing only the chromosome(s) they want to analyze.
obj	a list object returned by inudge.fit function.

new.plot	optional logical variable on whether to create new plot.
legpos	optional vector of (x,y) location for the legend position
xlim	optional x-axis limit (see par).
ylim	optional y-axis limit (see par).
xlab	optional x-axis label (see par).
ylab	optional y-axis label (see par).
main	optional plot title (see par).
lwd	optional line width for lines in the plot (see par).
...	additional graphical arguments to be passed to methods (see par).

Details

The components representing differential data are denoted by asterisk (*) symbol on the plot legend.

Author(s)

Cenny Taslim <taslim.2@osu.edu>, with contributions from Abbas Khalili <khalili@stat.ubc.ca>, Dustin Potter <potterdp@gmail.com>, and Shili Lin <shili@stat.osu.edu>

See Also

inudge.plot.mix, inudge.plot.comp, inudge.plot.fit, inudge.plot.qq, DIME.plot.fit, gng.plot.fit.

Examples

```
library(DIME);
# generate simulated datasets with underlying uniform and 2-normal distributions
set.seed(12);
N1 <- 1500; N2 <- 500; rmu <- c(-2.25,1.5); rsigma <- c(1,1);
rpi <- c(.10,.45,.45); a <- (-6); b <- 6;
chr4 <- list(c(-runif(ceiling(rpi[1]*N1),min = a,max =b),
  rnorm(ceiling(rpi[2]*N1),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N1),rmu[2],rsigma[2])));
chr9 <- list(c(-runif(ceiling(rpi[1]*N2),min = a,max =b),
  rnorm(ceiling(rpi[2]*N2),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N2),rmu[2],rsigma[2])));
# analyzing chromosome 4 and 9
data <- list(chr4,chr9);

# fit iNUDGE model with 2-normal components and maximum iterations = 20
set.seed(12);
bestInudge <- inudge.fit(data, K = 2, max.iter=20);

# plot individual components of iNUDGE
inudge.plot.comp(data,bestInudge);
# plot individual components of iNUDGE an it's mixture component on the same plot
inudge.plot.mix(bestInudge,resolution=1000,new.plot=FALSE);
```

inudge.plot.fit *Plot iNUDGE Goodness of Fit*

Description

Plot the estimated iNUDGE mixture model fitted using inudge.fit along with it's estimated individual components, superimposed on the histogram of the observation data. This plot shows how good the fit of the estimated model to the data.

Usage

```
inudge.plot.fit(data, obj, resolution = 100, breaks = 100,
legpos = NULL, xlim = NULL, main = NULL,...)
```

Arguments

data	an **R list** of vector of normalized intensities (counts). Each element can correspond to particular chromosome. User can construct their own list containing only the chromosome(s) they want to analyze.
obj	a list object returned by inudge.fit function.
resolution	optional bandwidth used to estimate the density function. Higher number smoother curve.
breaks	optional see hist, breaks parameters for histogram plot.
legpos	optional vector of (x,y) location for the legend position
xlim	optional x-axis limit (see par).
main	optional plot title (see par).
...	additional graphical arguments to be passed to methods (see par).

Details

The components representing differential data are denoted by asterisk (*) symbol on the plot legend.

See Also

gng.plot.comp, gng.plot.mix, hist

Examples

```
library(DIME);
# generate simulated datasets with underlying uniform and 2-normal distributions
set.seed(1234);
N1 <- 1500; N2 <- 500; rmu <- c(-2.25,1.5); rsigma <- c(1,1);
rpi <- c(.10,.45,.45); a <- (-6); b <- 6;
chr4 <- list(c(-runif(ceiling(rpi[1]*N1),min = a,max =b),
  rnorm(ceiling(rpi[2]*N1),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N1),rmu[2],rsigma[2])));
```

```
chr9 <- list(c(-runif(ceiling(rpi[1]*N2),min = a,max =b),
  rnorm(ceiling(rpi[2]*N2),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N2),rmu[2],rsigma[2])));
# analyzing chromosome 4 and 9
data <- list(chr4,chr9);

# fit iNUDGE model with 2-normal components and maximum iterations = 20
set.seed(1234);
bestInudge <- inudge.fit(data, K = 2, max.iter=20);

# Goodness of fit plot
inudge.plot.fit(data,bestInudge,legpos=c(-6,0.3),ylim=c(0,0.3),breaks=40);
```

`inudge.plot.mix` *Plot iNUDGE Mixture Component Function*

Description

Plot each estimated individual components of iNUDGE mixture model fitted using `inudge.fit`.

Usage

```
inudge.plot.mix(obj, amplify = 1, resolution = 100, new.plot = TRUE, ...)
```

Arguments

`obj`	a list object returned by `inudge.fit` function.
`amplify`	optional scaling factor for visualization purposes.
`resolution`	optional bandwidth used to estimate the density function. Higher number makes a smoother curve.
`new.plot`	optional logical variable on whether to create new plot.
`...`	additional graphical arguments to be passed to methods (see `par`).

See Also

`inudge.plot.comp`, `inudge.plot.fit`, `inudge.plot.qq`, `DIME.plot.fit`, `gng.plot.fit`.

Examples

```
library(DIME)

# generate simulated datasets with underlying uniform and 2-normal distributions
set.seed(1234);
N1 <- 1500; N2 <- 500; rmu <- c(-2.25,1.5); rsigma <- c(1,1);
rpi <- c(.10,.45,.45); a <- (-6); b <- 6;
chr4 <- list(c(-runif(ceiling(rpi[1]*N1),min = a,max =b),
  rnorm(ceiling(rpi[2]*N1),rmu[1],rsigma[1]),
```

```
  rnorm(ceiling(rpi[3]*N1),rmu[2],rsigma[2])));
chr9 <- list(c(-runif(ceiling(rpi[1]*N2),min = a,max =b),
  rnorm(ceiling(rpi[2]*N2),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N2),rmu[2],rsigma[2])));
# analyzing chromosome 4 and 9
data <- list(chr4,chr9);

# fit iNUDGE model
set.seed(1234);
bestInudge <- inudge.fit(data, K = 2, max.iter=20);

# plot estimated iNUDGE model imposed on the histogram of observed data
hist(unlist(data),freq=FALSE,breaks=40);
inudge.plot.mix(bestInudge,resolution=1000,new.plot=FALSE,col="red");
```

`inudge.plot.qq` *QQ-plot of GNG model vs. observed data*

Description

Produces a QQ-plot for visual inspection of quality of fit with regards to the uniform Gaussian (iNUDGE) mixture model estimated using the function `inudge.fit`

Usage

```
inudge.plot.qq(data, obj, resolution = 10, xlab = NULL, ylab = NULL,
  main = NULL, pch = NULL, ...)
```

Arguments

`data`	an **R list** of vector of normalized intensities (counts). Each element can correspond to particular chromosome. User can construct their own list containing only the chromosome(s) they want to analyze.
`obj`	a list object returned by `gng.fit` function.
`resolution`	optional number of points used to sample the estimated density function.
`xlab`	optional x-axis label (see `par`).
`ylab`	optional y-axis label (see `par`).
`main`	optional plot title (see `par`).
`pch`	optional plotting symbol to use (see `par`).
`...`	additional graphical arguments to be passed to methods (see `par`).

See Also

`inudge.fit`, `qqplot`

Examples

```
library(DIME);

# generate simulated datasets with underlying uniform and 2-normal distributions
set.seed(1234);
N1 <- 1500; N2 <- 500; rmu <- c(-2.25,1.5); rsigma <- c(1,1);
rpi <- c(.10,.45,.45); a <- (-6); b <- 6;
chr4 <- list(c(-runif(ceiling(rpi[1]*N1),min = a,max =b),
  rnorm(ceiling(rpi[2]*N1),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N1),rmu[2],rsigma[2])));
chr9 <- list(c(-runif(ceiling(rpi[1]*N2),min = a,max =b),
  rnorm(ceiling(rpi[2]*N2),rmu[1],rsigma[1]),
  rnorm(ceiling(rpi[3]*N2),rmu[2],rsigma[2])));
# analyzing chromosome 4 and 9
data <- list(chr4,chr9);

# fit iNUDGE model with 2-normal components and maximum iteration =20
set.seed(1234);
bestInudge <- inudge.fit(data, K=2, max.iter=20)

# QQ-plot
inudge.plot.qq(data,bestInudge);
```

`nudge.classify` *Classification Based on NUDGE Model*

Description

Classifies observed data into differential and non-differential groups based on NUDGE model.

Usage

```
nudge.classify(data, obj, obj.cutoff = 0.1, obj.sigma.diff.cutoff = NULL,
  obj.mu.diff.cutoff = NULL)
```

Arguments

`data`	an **R list** of vector of normalized intensities (counts). Each element can correspond to particular chromosome. User can construct their own list containing only the chromosome(s) they want to analyze.
`obj`	a list object returned by `nudge.fit` function.
`obj.cutoff`	optional local *fdr* cutoff for classifying data into differential and non-differential groups based on NUDGE model.
`obj.sigma.diff.cutoff`	optional cut-off for standard deviation of the normal component in NUDGE model to be designated as representing differential.

`obj.mu.diff.cutoff`
: optional cut-off for standard deviation of the normal component in NUDGE model to be designated as representing differential.

Value

A list object passed as input with additional element $class containing vector of classifications for all the observations in data. A classification of 1 denotes that the data is classified as differential with fdr < obj.cutoff.

`mu.diff.cutoff` normal component with mean > mu.diff.cutoff was used to represent differential component.

`sigma.diff.cutoff`
: normal component with standard deviation > sigma.diff.cutoff was used to represent differential component.

Author(s)

Cenny Taslim <taslim.2@osu.edu>, with contributions from Abbas Khalili <khalili@stat.ubc.ca>, Dustin Potter <potterdp@gmail.com>, and Shili Lin <shili@stat.osu.edu>

See Also

nudge.fit

Examples

```
library(DIME);
# generate simulated datasets with underlying uniform and 1-normal components
set.seed(1234);
N1 <- 1500; N2 <- 500; rmu <- c(1.5); rsigma <- c(1);
rpi <- c(.10,.90); a <- (-6); b <- 6;
chr1 <- c(-runif(ceiling(rpi[1]*N1),min = a,max =b),
  rnorm(ceiling(rpi[2]*N1),rmu[1],rsigma[1]));
chr4 <- c(-runif(ceiling(rpi[1]*N2),min = a,max =b),
  rnorm(ceiling(rpi[2]*N2),rmu[1],rsigma[1]));
# analyzing chromosome 1 and 4
data <- list(chr1,chr4);

# fit NUDGE model with maximum iterations = 20 only
set.seed(1234);
test <- nudge.fit(data, max.iter=20)
# vector of classification. 1 represents differential, 0 denotes non-differential
nudgeClass <- test$class;
```

nudge.fit *Function for Fitting NUDGE model parameters*

Description

Function to estimate parameters for both NUDGE model, mixture of uniform and 1-normal. Parameters are estimated using EM algorithm.

Usage

```
nudge.fit(data, avg = NULL, weights = NULL, weights.cutoff = -1.345,
  pi = NULL, mu = NULL, sigma = NULL, tol = 1e-5, max.iter = 2000, z = NULL)
```

Arguments

data	an **R list** of vector of normalized intensities (counts). Each element can correspond to a particular chromosome. User can construct their own list containing only the chromosome(s) they want to analyze.
avg	optional vector of mean data (or log intensities). Only required when any one of huber weight (lower, upper or full) is selected.
weights	optional weights to be used for robust fitting. Can be a matrix the same length as data, or a character description of the huber weight method to be employed: "lower" - only value below weights.cutoff are weighted,\ "upper" - only value above weights.cutoff are weighted,\ "full" - both values above and below weights.cutoff are weighted,\ If selected, mean of data (avg) is required.
weights.cutoff	optional cutoff to be used with the Huber weighting scheme.
pi	optional vector containing initial estimates for proportion of the NUDGE mixture components. The first entry is for the uniform component, the middle k entries are for normal components.
mu	optional vector containing initial estimates of the Gaussian means in NUDGE model.
sigma	optional vector containing initial estimates of the Gaussian standard deviation in (i)NUDGE model. Must have K entries.
tol	optional threshold for convergence for EM algorithm to estimate NUDGE parameters.
max.iter	optional maximum number of iterations for EM algorithm to estimate NUDGE parameters.
z	optional 2-column matrix with each row giving initial estimate of probability of the region being non-differential and a starting estimate for the probability of the region being differential. Each row must sum to 1. Number of row must be equal to data length.

Value

A list of object:

name	the name of the model "NUDGE"
pi	a vector of estimated proportion of each components in the model
mu	a vector of estimated Gaussian means for k-normal components.
sigma	a vector of estimated Gaussian standard deviation for k-normal components.
loglike	the log likelihood for the fitted mixture model.
iter	the actual number of iterations run by the EM algorithm.
fdr	the local false discover rate estimated based on NUDGE model.
phi	a matrix of estimated NUDGE mixture component function.
AIC	Akaike Information Criteria.
BIC	Bayesian Information Criteria.

Author(s)

Cenny Taslim <taslim.2@osu.edu>, with contributions from Abbas Khalili <khalili@stat.ubc.ca>, Dustin Potter <potterdp@gmail.com>, and Shili Lin <shili@stat.osu.edu>

See Also

DIME, gng.fit, inudge.fit

Examples

```
library(DIME);
# generate simulated datasets with underlying uniform and 1-normal components
set.seed(1234);
N1 <- 1500; N2 <- 500; rmu <- c(1.5); rsigma <- c(1);
rpi <- c(.10,.90); a <- (-6); b <- 6;
chr1 <- c(-runif(ceiling(rpi[1]*N1),min = a,max =b),
  rnorm(ceiling(rpi[2]*N1),rmu[1],rsigma[1]));
chr4 <- c(-runif(ceiling(rpi[1]*N2),min = a,max =b),
  rnorm(ceiling(rpi[2]*N2),rmu[1],rsigma[1]));
# analyzing chromosome 1 and 4
data <- list(chr1,chr4);

# fit NUDGE model with maximum iterations = 20 only
set.seed(1234);
bestNudge <- nudge.fit(data, max.iter=20);

# Getting the best fitted NUDGE model (parameters)
bestNudge$pi # estimated proportion of each component in NUDGE
bestNudge$mu # estimated mean of the normal component(s) in NUDGE
# estimated standard deviation of the normal component(s) in NUDGE
bestNudge$sigma
```

nudge.plot.comp *Plot NUDGE Individual Components*

Description

Plot each estimated individual components of NUDGE model (mixture of uniform and 1-normal) fitted using nudge.fit.

Usage

```
nudge.plot.comp(data, obj, new.plot = TRUE, legpos = NULL, xlim = NULL,
  ylim = NULL, xlab = NULL, ylab = NULL, main = NULL, lwd = NULL, ...)
```

Arguments

data	an **R list** of vector of normalized intensities (counts). Each element can correspond to a particular chromosome. User can construct their own list containing only the chromosome(s) they want to analyze.
obj	a list object returned by nudge.fit function.
new.plot	an **R list** of vector of normalized intensities (counts). Each object can correspond to particular chromosome that one want to fit.
legpos	optional vector of (x,y) location for the legend position
xlim	optional x-axis limit (see par).
ylim	optional y-axis limit (see par).
xlab	optional x-axis label (see par).
ylab	optional y-axis label (see par).
main	optional plot title (see par).
lwd	optional line width for lines in the plot (see par).
...	additional graphical arguments to be passed to methods (see par).

Details

The components representing differential data are denoted by asterisk (*) symbol on the plot legend.

See Also

nudge.plot.mix, inudge.plot.comp, nudge.plot.fit, nudge.plot.qq, DIME.plot.fit, gng.plot.fit.

Examples

```
library(DIME);
# generate simulated datasets with underlying uniform and 1-normal components
set.seed(1234);
N1 <- 1500; N2 <- 500; rmu <- c(1.5); rsigma <- c(1);
rpi <- c(.10,.90); a <- (-6); b <- 6;
chr1 <- c(-runif(ceiling(rpi[1]*N1),min = a,max =b),
  rnorm(ceiling(rpi[2]*N1),rmu[1],rsigma[1]));
chr4 <- c(-runif(ceiling(rpi[1]*N2),min = a,max =b),
  rnorm(ceiling(rpi[2]*N2),rmu[1],rsigma[1]));
# analyzing chromosome 1 and 4
data <- list(chr1,chr4);

# fit NUDGE model with maximum iterations = 20
set.seed(1234);
bestNudge <- nudge.fit(data, max.iter=20);

# plot individual components of NUDGE
nudge.plot.comp(data,bestNudge);
```

`nudge.plot.fit` *Plot NUDGE Goodness of Fit*

Description

Plot the estimated NUDGE mixture model fitted using `nudge.fit` along with it's estimated individual components, superimposed on the histogram of the observation data. This plot shows how good the fit of the estimated model to the data.

Usage

```
nudge.plot.fit(data, obj, resolution = 100, breaks = 100,
xlim = NULL, legpos = NULL, ...)
```

Arguments

`data`	an **R list** of vector of normalized intensities (counts). Each element can correspond to a particular chromosome. User can construct their own list containing only the chromosome(s) they want to analyze.
`obj`	a list object returned by `nudge.fit` function.
`resolution`	optional bandwidth used to estimate the density function. Higher number smoother curve.
`breaks`	optional see `hist`, breaks parameters for histogram plot.
`xlim`	optional limit for the x-axis.
`legpos`	optional vector of (x,y) location for the legend position
`...`	additional graphical arguments to be passed to methods (see `par`).

Details

The components representing differential data are denoted by asterisk (*) symbol on the plot legend.

See Also

nudge.plot.comp, nudge.plot.mix, hist

Examples

```
library(DIME);
# generate simulated datasets with underlying uniform and 1-normal components
set.seed(1234);
N1 <- 1500; N2 <- 500; rmu <- c(1.5); rsigma <- c(1);
rpi <- c(.10,.90); a <- (-6); b <- 6;
chr1 <- c(-runif(ceiling(rpi[1]*N1),min = a,max =b),
  rnorm(ceiling(rpi[2]*N1),rmu[1],rsigma[1]));
chr4 <- c(-runif(ceiling(rpi[1]*N2),min = a,max =b),
  rnorm(ceiling(rpi[2]*N2),rmu[1],rsigma[1]));
# analyzing chromosome 1 and 4
data <- list(chr1,chr4);

# fit NUDGE model with maximum iterations = 20
set.seed(1234);
bestNudge <- nudge.fit(data, max.iter=20);

# goodness of fit plot
nudge.plot.fit(data,bestNudge,breaks=40);
```

---

nudge.plot.mix *Plot NUDGE Mixture Component Function*

Description

Plot each estimated individual components of NUDGE mixture model fitted using nudge.fit.

Usage

```
nudge.plot.mix(obj, amplify = 1, resolution = 100, new.plot = TRUE, ...)
```

Arguments

obj	a list object returned by nudge.fit function.
amplify	optional scaling factor for visualization purposes.
resolution	optional bandwidth used to estimate the density function. Higher number makes a smoother curve.
new.plot	optional logical variable on whether to create new plot.
...	additional graphical arguments to be passed to methods (see par).

See Also

nudge.plot.comp, nudge.plot.fit, nudge.plot.qq, DIME.plot.fit, gng.plot.fit.

Examples

```
library(DIME);
# generate simulated datasets with underlying uniform and 1-normal components
set.seed(1234);
N1 <- 1500; N2 <- 500; rmu <- c(1.5); rsigma <- c(1);
rpi <- c(.10,.90); a <- (-6); b <- 6;
chr1 <- c(-runif(ceiling(rpi[1]*N1),min = a,max =b),
  rnorm(ceiling(rpi[2]*N1),rmu[1],rsigma[1]));
chr4 <- c(-runif(ceiling(rpi[1]*N2),min = a,max =b),
  rnorm(ceiling(rpi[2]*N2),rmu[1],rsigma[1]));
# analyzing chromosome 1 and 4
data <- list(chr1,chr4);

# fit NUDGE model with maximum iterations = 20 only
bestNudge <- nudge.fit(data, max.iter=20);

# plot estimated iNUDGE model imposed on the histogram of observed data
hist(unlist(data),freq=FALSE,breaks=40);
nudge.plot.mix(bestNudge,resolution=1000,new.plot=FALSE,col="red");
```

---

nudge.plot.qq *QQ-plot of GNG model vs. observed data*

Description

Produces a QQ-plot for visual inspection of quality of fit with regards to the uniform Gaussian (NUDGE) mixture model estimated using the function nudge.fit

Usage

```
nudge.plot.qq(data, obj, resolution = 10, xlab = NULL, ylab = NULL,
  main = NULL, pch = NULL, ...)
```

Arguments

data	an **R list** of vector of normalized intensities (counts). Each element can correspond to a particular chromosome. User can construct their own list containing only the chromosome(s) they want to analyze.
obj	a list object returned by gng.fit function.
resolution	optional number of points used to sample the estimated density function.
xlab	optional x-axis label (see par).
ylab	optional y-axis label (see par).

main	optional plot title (see par).
pch	optional plotting character, i.e., symbol to use (see par).
...	additional graphical arguments to be passed to methods (see par).

See Also

nudge.fit, qqplot

Examples

```
library(DIME)

# generate simulated datasets with underlying uniform and 1-normal components
set.seed(1234);
N1 <- 1500; N2 <- 500; rmu <- c(1.5); rsigma <- c(1);
rpi <- c(.10,.90); a <- (-6); b <- 6;
chr1 <- c(-runif(ceiling(rpi[1]*N1),min = a,max =b),
  rnorm(ceiling(rpi[2]*N1),rmu[1],rsigma[1]));
chr4 <- c(-runif(ceiling(rpi[1]*N2),min = a,max =b),
  rnorm(ceiling(rpi[2]*N2),rmu[1],rsigma[1]));
# analyzing chromosome 1 and 4
data <- list(chr1,chr4);

# fit NUDGE model with maximum iterations = 20
set.seed(1234);
bestNudge <- nudge.fit(data, max.iter=20);

# QQ-plot
nudge.plot.qq(data,bestNudge);
```

Index