Package: IntNMF (via r-universe)

August 22, 2024

Type Package

Title Integrative Clustering of Multiple Genomic Dataset

Version 1.2.0

Date 2018-07-17

Author Prabhakar Chalise, Rama Raghavan, Brooke Fridley

Maintainer Prabhakar Chalise <pchalise@kumc.edu>

Description Carries out integrative clustering analysis using multiple types of genomic dataset using integrative Non-negative Matrix factorization.

License GPL-2

Depends R (>= 3.5), MASS, NMF, mclust, cluster, InterSIM

NeedsCompilation no

Repository CRAN

Date/Publication 2018-07-18 22:40:19 UTC

RoxygenNote 6.0.1

Contents

`ClusterEntropy` *A function to measure cluster entropy*

Description

Given the true clustering assignment for the subjects, this function calculates cluster entropy index comparing with clustering assignment determined by integrative NMF algorithm. Smaller value of cluster entropy indicates better cluster predictive discrimination.

Usage

```
ClusterEntropy(ComputedClusters, TrueClasses)
```

Arguments

`ComputedClusters`	Clustering assignment determined by integrative NMF algorithm
`TrueClasses`	True clustering assignment of the subjects

Value

Cluster entropy index value

Author(s)

Prabhakar Chalise, Rama Raghavan, Brooke Fridley

References

Kim Hyunsoo and Park Haesun (2007). Sparse non-negative matrix factorization via alternating non-negativity constrained least squares for microarray data analysis. Bioinformatics, 23: 1495-1502.

Examples

```
prop <- c(0.20,0.30,0.27,0.23)
effect <- 2.5
sim.D <- InterSIM(n.sample=100,cluster.sample.prop=prop,delta.methyl=effect,
delta.expr=effect,delta.protein=effect,p.DMP=0.25,p.DEG=NULL,p.DEP=NULL,
do.plot=FALSE, sample.cluster=TRUE, feature.cluster=TRUE)
dat1 <- sim.D$dat.methyl
dat2 <- sim.D$dat.expr
dat3 <- sim.D$dat.protein
true.cluster.assignment <- sim.D$clustering.assignment

## Make all data positive by shifting to positive direction.
## Also rescale the datasets so that they are comparable.
if (!all(dat1>=0)) dat1 <- pmax(dat1 + abs(min(dat1)), .Machine$double.eps)
dat1 <- dat1/max(dat1)
```

```
if (!all(dat2>=0)) dat2 <- pmax(dat2 + abs(min(dat2)), .Machine$double.eps)
dat2 <- dat2/max(dat2)
if (!all(dat3>=0)) dat3 <- pmax(dat3 + abs(min(dat3)), .Machine$double.eps)
dat3 <- dat3/max(dat3)

# The function nmf.mnnals requires the samples to be on rows and variables on columns.
dat <- list(dat1,dat2,dat3)
fit <- nmf.mnnals(dat=dat,k=length(prop),maxiter=200,st.count=20,n.ini=15,ini.nndsvd=TRUE,
seed=TRUE)
ClusterEntropy(ComputedClusters=fit$clusters, TrueClasses=true.cluster.assignment$cluster.id)
```

`ClusterPurity` *A function to measure cluster purity*

Description

Given the true clustering assignment for the subjects, this function calculates cluster purity index comparing with clustering assignment determined by integrative NMF algorithm. Higher value of cluster purity indicates better cluster predictive discrimination.

Usage

```
ClusterPurity(ComputedClusters, TrueClasses)
```

Arguments

`ComputedClusters`	Clustering assignment determined by integrative NMF algorithm
`TrueClasses`	True clustering assignment of the subjects

Value

Cluster purity index value

Author(s)

Prabhakar Chalise, Rama Raghavan, Brooke Fridley

References

Kim Hyunsoo and Park Haesun (2007). Sparse non-negative matrix factorization via alternating non-negativity constrained least squares for microarray data analysis. Bioinformatics, 23: 1495-1502.

Examples

```
prop <- c(0.20,0.30,0.27,0.23)
effect <- 2.5
sim.D <- InterSIM(n.sample=100,cluster.sample.prop=prop,delta.methyl=effect,
delta.expr=effect,delta.protein=effect,p.DMP=0.25,p.DEG=NULL,p.DEP=NULL,
do.plot=FALSE, sample.cluster=TRUE, feature.cluster=TRUE)

dat1 <- sim.D$dat.methyl
dat2 <- sim.D$dat.expr
dat3 <- sim.D$dat.protein
true.cluster.assignment <- sim.D$clustering.assignment

## Make all data positive by shifting to positive direction.
## Also rescale the datasets so that they are comparable.
if (!all(dat1>=0)) dat1 <- pmax(dat1 + abs(min(dat1)), .Machine$double.eps)
dat1 <- dat1/max(dat1)
if (!all(dat2>=0)) dat2 <- pmax(dat2 + abs(min(dat2)), .Machine$double.eps)
dat2 <- dat2/max(dat2)
if (!all(dat3>=0)) dat3 <- pmax(dat3 + abs(min(dat3)), .Machine$double.eps)
dat3 <- dat3/max(dat3)

# The function nmf.mnnals requires the samples to be on rows and variables on columns.
dat <- list(dat1,dat2,dat3)
fit <- nmf.mnnals(dat=dat,k=length(prop),maxiter=200,st.count=20,n.ini=15,ini.nndsvd=TRUE,
seed=TRUE)
ClusterPurity(ComputedClusters=fit$clusters, TrueClasses=true.cluster.assignment$cluster.id)
```

ConsensusMatPlot *A function to create image plot of the consensus matrix*

Description

Given the integrative NMF fit object, the function creates image plot of the consensus matrix ordered according to clusters groups. Cleaner block structure indicates stronger clusters.

Usage

```
ConsensusMatPlot(fit, rowLab = TRUE, colLab = TRUE)
```

Arguments

fit	A nmf.mnnals fit object
rowLab	If true row label is displayed. Default is TRUE.
colLab	If true column label is displayed. Default is TRUE.

Value

Image plot of the consensus matrix ordered according to cluster groups is returned.

Author(s)

Prabhakar Chalise, Rama Raghavan, Brooke Fridley

References

Brunnet J, Tamayo P Golub, T and Mesirov J (2004) Metagene and molecular pattern discovery using matrix factorization. PNAS, 101, 4164-4169

Monti S, Tamayo P, Mesirov J and Golup T (2003). Consesus Clustering: A resampling based method for class discovery and visualization of gene expression microarray data. Machine Learning J, 52:91-118.

Examples

```
prop <- c(0.20,0.30,0.27,0.23)
effect <- 2.5
sim.D <- InterSIM(n.sample=100,cluster.sample.prop=prop,delta.methyl=effect,
delta.expr=effect,delta.protein=effect,p.DMP=0.25,p.DEG=NULL,p.DEP=NULL,
do.plot=FALSE, sample.cluster=TRUE, feature.cluster=TRUE)
dat1 <- sim.D$dat.methyl
dat2 <- sim.D$dat.expr
dat3 <- sim.D$dat.protein
true.cluster.assignment <- sim.D$clustering.assignment

## Make all data positive by shifting to positive direction.
## Also rescale the datasets so that they are comparable.
if (!all(dat1>=0)) dat1 <- pmax(dat1 + abs(min(dat1)), .Machine$double.eps)
dat1 <- dat1/max(dat1)
if (!all(dat2>=0)) dat2 <- pmax(dat2 + abs(min(dat2)), .Machine$double.eps)
dat2 <- dat2/max(dat2)
if (!all(dat3>=0)) dat3 <- pmax(dat3 + abs(min(dat3)), .Machine$double.eps)
dat3 <- dat3/max(dat3)

# The function nmf.mnnals requires the samples to be on rows and variables on columns.
dat <- list(dat1,dat2,dat3)
fit <- nmf.mnnals(dat=dat,k=length(prop),maxiter=200,st.count=20,n.ini=15,ini.nndsvd=TRUE,
seed=TRUE)
ConsensusMatPlot(fit,rowLab=TRUE,colLab=TRUE)
```

`nmf.mnnals` *Nonnegative Matrix Factorization of Multiple data using Nonnegative Alternating Least Square*

Description

Given a single or multiple types of datasets (e.g. DNA methylation, mRNA expression, protein expression, DNA copy number) measured on same set of samples and pre-defined number of clusters, the function carries out clustering of the samples together with cluster membership assignment to the samples utilizing all the data set in a single comprehensive step.

Usage

```
nmf.mnnals(dat = dat, k = k, maxiter = 200, st.count = 20, n.ini = 30, ini.nndsvd = TRUE,
seed = TRUE,wt=if(is.list(dat)) rep(1,length(dat)) else 1)
```

Arguments

dat	A single data or a list of multiple data matrices measured on same set of samples. For each data matrix in the list, samples should be on rows and genomic features should be on columns.
k	Number of clusters
maxiter	Maximum number of iteration, default is 200.
st.count	Count for stability in connectivity matrix, default is 20.
n.ini	Number of initializations of the random matrices, default is 30.
ini.nndsvd	Initialization of the Hi matrices using non negative double singular value decomposition (NNDSVD). If true, one of the initializations of algorithm will use NNDSVD. Default is TRUE.
seed	Random seed for initialization of algorithm, default is TRUE
wt	Weight, default is 1 for each data.

Value

consensus	Consensus matrix
W	Common basis matrix across the multiple data sets
H	List of data specific coefficient matrices.
convergence	Matrix with five columns and number of rows equal to number of iterative steps required to converge the algorithm or number of maximum iteration. The five columns represent number of iterations, count for stability in connectivity matrix, stability indicator (1/0), absolute difference in reconstruction error between ith and (i-1)th iteration and value of the objective function respectively.
min.f.WH	Collection of values of objective function at convergence for each initialization of the algorithm.
clusters	Cluster membership assignment to samples.

Author(s)

Prabhakar Chalise, Rama Raghavan, Brooke Fridley

References

Chalise P and Fridley B (2017). Integrative clustering of multi-level 'omic data based on non-negative matrix factorization algorithm. PLOS ONE, 12(5), e0176278.

Chalise P, Raghavan R and Fridley B (2016). InterSIM: Simulation tool for multiple integrative 'omic datasets. Computer Methods and Programs in Biomedicine, 128:69-74.

Examples

```
#### Simulation of three interrelated dataset
#prop <- c(0.65,0.35)
#prop <- c(0.30,0.40,0.30)
prop <- c(0.20,0.30,0.27,0.23)
effect <- 2.5

library(InterSIM)
sim.D <- InterSIM(n.sample=100, cluster.sample.prop=prop, delta.methyl=effect,
delta.expr=effect, delta.protein=effect, p.DMP=0.25, p.DEG=NULL, p.DEP=NULL,
do.plot=FALSE, sample.cluster=TRUE, feature.cluster=TRUE)
dat1 <- sim.D$dat.methyl
dat2 <- sim.D$dat.expr
dat3 <- sim.D$dat.protein
true.cluster.assignment <- sim.D$clustering.assignment

## Make all data positive by shifting to positive direction.
## Also rescale the datasets so that they are comparable.
if (!all(dat1>=0)) dat1 <- pmax(dat1 + abs(min(dat1)), .Machine$double.eps)
dat1 <- dat1/max(dat1)
if (!all(dat2>=0)) dat2 <- pmax(dat2 + abs(min(dat2)), .Machine$double.eps)
dat2 <- dat2/max(dat2)
if (!all(dat3>=0)) dat3 <- pmax(dat3 + abs(min(dat3)), .Machine$double.eps)
dat3 <- dat3/max(dat3)
# The function nmf.mnnals requires the samples to be on rows and variables on columns.
dat1[1:5,1:5]
dat2[1:5,1:5]
dat3[1:5,1:5]
dat <- list(dat1,dat2,dat3)

# Find optimum number of clusters for the data
#opt.k <- nmf.opt.k(dat=dat, n.runs=5, n.fold=5, k.range=2:7, result=TRUE,
#make.plot=TRUE, progress=TRUE)
# Find clustering assignment for the samples
fit <- nmf.mnnals(dat=dat, k=length(prop), maxiter=200, st.count=20, n.ini=15,
ini.nndsvd=TRUE, seed=TRUE)
table(fit$clusters)
fit$clusters[1:10]
```

`nmf.opt.k` *Selection of optimum number of clusters (k)*

Description

Given a single or multiple types of datasets (e.g. DNA methylation, mRNA expression, protein expression, DNA copy number) measured on same set of samples, the function finds optimum number of clusters for the data or dataset.

Usage

```
nmf.opt.k(dat = dat, n.runs = 30, n.fold = 5, k.range = 2:8, result = TRUE,
make.plot = TRUE, progress = TRUE, st.count = 10, maxiter = 100,
wt=if(is.list(dat)) rep(1,length(dat)) else 1)
```

Arguments

dat	A single data or list of multiple types of data set measured on same set of samples. For each data matrix in the list, samples should be on rows and genomic features should be on columns.
n.runs	Number of runs of algorithm in order to find optimum number of clusters, default is 30.
n.fold	Number of folds for k-fold cross-validation, default is 5.
k.range	Search range for optimum number of clusters, default is 2:8
result	Logical, to display the result-matrix, default is TRUE.
make.plot	Logical, to display the plot of cluster prediction index vs search range of clusters, default is TRUE
progress	Logical, to display the progress (in percentage) of the algorithm, default is TRUE
st.count	Count for stability in connectivity matrix, default is 10.
maxiter	Maximum number of iteration, default is 100.
wt	Weight, default is 1 for each data.

Value

The function returns a matrix of cluster prediction index (CPI) values for each run (columns) over the search range of number of clusters (rows). The function also generates plot of CPI over the search range of number of clusters.

Author(s)

Prabhakar Chalise, Rama Raghavan, Brooke Fridley

References

Chalise P and Fridley B (2017). Integrative clustering of multi-level 'omic data based on non-negative matrix factorization algorithm. PLOS ONE, 12(5), e0176278.

Chalise P, Raghavan R and Fridley B (2016). InterSIM: Simulation tool for multiple integrative 'omic datasets. Computer Methods and Programs in Biomedicine, 128:69-74.

Examples

```
#### Simulation of three interrelated dataset
#prop <- c(0.65,0.35)
#prop <- c(0.30,0.40,0.30)
prop <- c(0.20,0.30,0.27,0.23)
```

```
effect <- 2.5

library(InterSIM)
sim.D <- InterSIM(n.sample=100, cluster.sample.prop=prop, delta.methyl=effect,
delta.expr=effect, delta.protein=effect, p.DMP=0.25, p.DEG=NULL, p.DEP=NULL,
do.plot=FALSE, sample.cluster=TRUE, feature.cluster=TRUE)
dat1 <- sim.D$dat.methyl
dat2 <- sim.D$dat.expr
dat3 <- sim.D$dat.protein
true.cluster.assignment <- sim.D$clustering.assignment

## Make all data positive by shifting to positive direction.
## Also rescale the datasets so that they are comparable.
if (!all(dat1>=0)) dat1 <- pmax(dat1 + abs(min(dat1)), .Machine$double.eps)
dat1 <- dat1/max(dat1)
if (!all(dat2>=0)) dat2 <- pmax(dat2 + abs(min(dat2)), .Machine$double.eps)
dat2 <- dat2/max(dat2)
if (!all(dat3>=0)) dat3 <- pmax(dat3 + abs(min(dat3)), .Machine$double.eps)
dat3 <- dat3/max(dat3)
# The function nmf.mnnals requires the samples to be on rows and variables on columns.
dat1[1:5,1:5]
dat2[1:5,1:5]
dat3[1:5,1:5]
dat <- list(dat1,dat2,dat3)

# Find optimum number of clusters for the data
#opt.k <- nmf.opt.k(dat=dat, n.runs=5, n.fold=5, k.range=2:7, result=TRUE,
#make.plot=TRUE,progress=TRUE)
```

`SilhouettePlot` *A function to plot silhouette width*

Description

Given the integrative NMF fit object, the function creates silhouette width plot with different colors for different cluster groups.

Usage

```
SilhouettePlot(fit, cluster.col = NULL)
```

Arguments

`fit`	A nmf.mnnals fit object
`cluster.col`	Colors for the cluster groups. If NULL, default colors are used.

Value

Silhouette width plot is returned together with mean silhouette width for each group, overall silhouette width and summary statistics.

Author(s)

Prabhakar Chalise, Rama Raghavan, Brooke Fridley

References

Rousseeue PJ (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. Journal of Computational and Applied Mathematics, 20:53-65

Examples

```
prop <- c(0.20,0.30,0.27,0.23)
effect <- 2.5
sim.D <- InterSIM(n.sample=100,cluster.sample.prop=prop,delta.methyl=effect,
delta.expr=effect,delta.protein=effect,p.DMP=0.25,p.DEG=NULL,p.DEP=NULL,
do.plot=FALSE, sample.cluster=TRUE, feature.cluster=TRUE)

dat1 <- sim.D$dat.methyl
dat2 <- sim.D$dat.expr
dat3 <- sim.D$dat.protein
true.cluster.assignment <- sim.D$clustering.assignment

## Make all data positive by shifting to positive direction.
## Also rescale the datasets so that they are comparable.
if (!all(dat1>=0)) dat1 <- pmax(dat1 + abs(min(dat1)), .Machine$double.eps)
dat1 <- dat1/max(dat1)
if (!all(dat2>=0)) dat2 <- pmax(dat2 + abs(min(dat2)), .Machine$double.eps)
dat2 <- dat2/max(dat2)
if (!all(dat3>=0)) dat3 <- pmax(dat3 + abs(min(dat3)), .Machine$double.eps)
dat3 <- dat3/max(dat3)

# The function nmf.mnnals requires the samples to be on rows and variables on columns.
dat <- list(dat1,dat2,dat3)
fit <- nmf.mnnals(dat=dat,k=length(prop),maxiter=200,st.count=20,n.ini=15,ini.nndsvd=TRUE,
seed=TRUE)
SilhouettePlot(fit,cluster.col=NULL)
```

Index