Package 'RGBM'

July 21, 2025

Type Package

Title LS-TreeBoost and LAD-TreeBoost for Gene Regulatory Network Reconstruction

Version 1.0-11

Date 2023-04-13

Author Raghvendra Mall [aut, cre],
Khalid Kunji [aut],
Melissa O'Neill [ctb]

Maintainer Raghvendra Mall <raghvendra5688@gmail.com>

Repository CRAN

Description
Provides an implementation of Regularized LS-TreeBoost & LAD-TreeBoost algorithm for Regulatory Network inference from any type of expression data (Microarray/RNA-seq etc).

License GPL (>= 3)

LazyLoad yes

Depends foreach, plyr, doParallel

NeedsCompilation yes

Date/Publication 2023-04-14 08:50:14 UTC

Contents

`add_names` *Add row and column names to the adjacency matrix A*

Description

Here we add the names of the transcription factors (Tfs) as rownames and names of the target genes as column names to the adjacency matrix A.

Usage

```
add_names(A, tfs, targets)
```

Arguments

`A`	Adjacency matrix A obtained as a result of GBM procedure.
`tfs`	List of names of transcription factors.
`targets`	List of names of target genes.

Details

In case of DREAM Challenge datasets list of transcription factors is same as list of target genes and are referred as G1, ..., G100.

Author(s)

Raghvendra Mall <rmall@hbku.edu.qa>

apply_row_deviation *Apply row-wise deviation on the inferred GRN*

Description

This function performs a row-wise standard deviation of network A to generate an S1 matrix which is then used to modify the weights in network A

Usage

```
apply_row_deviation(A,Ntfs,Ntargets)
```

Arguments

A	Inferred GRN in the form of Ntfs-by-Ntargets matrix
Ntfs	Total number of transcription factors used in the experiment.
Ntargets	Total number of target genes used in the experiment

Value

Refined adjacency matrix A in the form of Ntfs-by-Ntargets matrix

Author(s)

Raghvendra Mall <rmall@hbku.edu.qa>

consider_previous_information *Remember the intermediate inferred GRN while generating the final inferred GRN*

Description

This function combines the adjacency matrix A_prev obtained as a result of `first_GBM_step` with the adjacency matrix A obtained as a result of `second_GBM_step`. All the edges in the matrix A which have non-zero weights are given machine precision weights initially. We then perform a harmonic mean for each element of A_prev and A to obtain a regularized adjacency matrix (A_final). As a result of this procedure transcriptional regulations which were strong and present in both A_prev and A end up getting highest weights in A_final. We finally remove all edges whose weights are less than machine precision from A_final.

Usage

```
consider_previous_information(A, A_prev,real)
```

Arguments

A	Inferred GRN from the second_GBM_step
A_prev	Inferred GRN from the first_GBM_step
real	Numeric value 0 or 1 corresponding to simulated or real experiment respectively.

Value

Returns an adjacency matrix A_final of the form Ntfs-by-Ntargets

Author(s)

Raghvendra Mall <rmall@hbku.edu.qa>

See Also

first_GBM_step, second_GBM_step

Examples

```
## The function is currently defined as
function (A, A_prev)
{
  #Utilize Past Information also to not remove true positives
  A_prev[A_prev==0] <- .Machine$double.eps;
  A_prev <- transform_importance_to_weights(A_prev);
  A[A==0] <- .Machine$double.eps;
  epsilon <- 1/log(1/.Machine$double.eps);
  A <- transform_importance_to_weights(A);
  A_final <- 2*A*A_prev/(A+A_prev);
  A_final <- A_final - epislon;
  A_final[A_final<0] <- 0.0;
  return(A_final);
}
```

---

first_GBM_step *Perform either LS-Boost or LAD-Boost (GBM) on expression matrix E followed by the null_model_refinement_step*

Description

This function utilizes the core gradient boosting machine model (GBM) followed by the refinement step to generate the first adjacency matrix A of size p x p using the list of Tfs and the set of target genes. Several such adjacency matrices (A) are obtained based on the number of iterations to be performed. All these adjacency matrices are averaged to reduce the noise in the inferred intermediate GRN.

Usage

```
first_GBM_step(E, K, tfs, targets, Ntfs, Ntargets, lf, M, nu,s_f, no_iterations)
```

Arguments

E	N-by-p expression matrix. Columns correspond to genes, rows correspond to experiments. E is expected to be already normalized using standard methods, for example RMA. Colnames of E is the set of all genes.
K	N-by-p initial perturbation matrix. It directly corresponds to E matrix, e.g. if K[i,j] is equal to 1, it means that gene j was knocked-out in experiment i. Single gene knock-out experiments are rows of K with only one value 1. Colnames of K is set to be the set of all genes. By default it's a matrix of zeros of the same size as E, e.g. unknown initial perturbation state of genes.
tfs	List of names of transcription factors. In case of presence of prior mechanistic network it is a subset of all the p genes whereas in absence of such a mechanistic network it is a list of names of all the p genes.
targets	List of names of target genes. In case of presence of prior mechanistic network it is a subset of all the p genes whereas in absence of such a mechanistic network it is a list of names of all the p genes.
Ntfs	Total number of transcription factors used in the experiment.
Ntargets	Total number of target genes used in the experiment.
lf	Loss Function: 1 -> Least Squares and 2 -> Least Absolute Deviation
M	Number of extensions in boosting model, e.g. number of iterations of the main loop of RGBM algorithm. By default it's 5000.
nu	Shrinkage factor, learning rate, 0<nu<=1. Each extension to boosting model will be multiplied by the learning rate. By default it's 0.001.
s_f	Sampling rate of transcription factors, 0<s_f<=1. Fraction of transcription factors from E, as indicated by tfs vector, which will be sampled without replacement to calculate each extesion in boosting model. By default it's 0.3.
no_iterations	Number of iterations to perform equivalent to building that many core LS-Boost/LAD-Boost models and then averaging them to have smooth edge-weights in the inferred intermediate GRN.

Value

Intermediate Gene Regulatory Network in form of a Ntfs-by-Ntargets adjacency matrix.

Author(s)

Raghvendra Mall <rmall@hbku.edu.qa>

See Also

second_GBM_step

GBM *Calculate Gene Regulatory Network from Expression data using either LS-TreeBoost or LAD-TreeBoost*

Description

This function calculates a Ntfs-by-Ntargets adjacency matrix A from N-by-p expression matrix E. E is expected to be given as input. E is assumed to have p columns corresponding to all the genes, Ntfs represents the number of transcription factors and Ntargets represents the number of target genes and N rows corresponding to different experiments. Additionally, GBM function takes matrix of initial perturbations of genes K of the same size as E, and other parameters including which loss function to use (LS = 1, LAD = 2). As a result, GBM returns a squared matrix A of edge confidences of size Ntfs-by-Ntargets. A subset of known transcription factors can be defined as a subset of all p genes.

Usage

```
GBM(E = matrix(rnorm(100), 10, 10), K = matrix(0, nrow(E), ncol(E)),
     tfs = paste0("G",c(1:10)), targets = paste0("G",c(1:10)),
     s_s = 1, s_f = 0.3, lf = 1,
     M = 5000,nu = 0.001, scale = TRUE,center = TRUE, optimization.stage = 2)
```

Arguments

E	N-by-p expression matrix. Columns correspond to genes, rows correspond to experiments. E is expected to be already normalized using standard methods, for example RMA. Colnames of E is the set of all genes.
K	N-by-p initial perturbation matrix. It directly corresponds to E matrix, e.g. if K[i,j] is equal to 1, it means that gene j was knocked-out in experiment i. Single gene knock-out experiments are rows of K with only one value 1. Colnames of K is set to be the set of all genes. By default it's a matrix of zeros of the same size as E, e.g. unknown initial perturbation state of genes.
tfs	List of names of transcription factors
targets	List of names of target genes
s_s	Sampling rate of experiments, 0<s_s<=1. Fraction of rows of E, which will be sampled with replacement to calculate each extension in boosting model. By default it's 1.
s_f	Sampling rate of transcription factors, 0<s_f<=1. Fraction of transcription factors from E, as indicated by tfs vector, which will be sampled without replacement to calculate each extesion in boosting model. By default it's 0.3.
lf	Loss function: 1 -> Least Squares, 2 -> Least Absolute deviation
M	Number of extensions in boosting model, e.g. number of iterations of the main loop of RGBM algorithm. By default it's 5000.
nu	Shrinkage factor, learning rate, 0<nu<=1. Each extension to boosting model will be multiplied by the learning rate. By default it's 0.001.

scale — Logical flag indicating if each column of E should be scaled to be unit standard deviation. By default it's TRUE.

center — Logical flag indicating if each column of E should be scaled to be zero mean. By default it's TRUE.

optimization.stage — Numerical flag indicating if re-evaluation of edge confidences should be applied after calculating initial V, optimization.stage={0,1,2}. If optimization.stage=0, no re-evaluation will be applied. If optimization.stage=1, variance-based optimization will be applied. If optimization.stage=2, variance-based and z-score based optimizations will be applied.

Value

A — Gene Regulatory Network in form of a Ntfs-by-Ntargets adjacency matrix.

Author(s)

Raghvendra Mall <rmall@hbku.edu.qa>

See Also

GBM.train, GBM.test, v2l

Examples

```
# load RGBM library
library("RGBM")
# this step is optional, it helps speed up calculations, run in parallel on 2 processors
library(doParallel)
cl <- makeCluster(2)
# run network inference on a 100-by-100 dummy expression data.
V = GBM()
stopCluster(cl)
```

GBM.test *Test GBM predictor*

Description

This function tests a regression model for a given X.test feature matrix, Y.test response vector, and working parameters.

Usage

```
GBM.test(model, X.test, Y.test, M.test)
```

Arguments

model	Model returned by GBM.train function.
X.test	Input N-by-p feature matrix of unseen samples. Columns correspond to features, rows correspond to samples.
Y.test	Input N-element response vector of unseen samples.
M.test	Number of extensions of boosting model to take when predicting response. Must be not greater than M.train used when training boosting model.

Value

Result of regression

Author(s)

Raghvendra Mall <rmall@hbku.edu.qa>

See Also

GBM.train

GBM.train *Train GBM predictor*

Description

This function trains a regression model for a given X.train feature matrix, Y.train response vector, and working parameters. A model returned by this function can be used to predict response for unseen data with GBM.test function.

Usage

```
GBM.train(X.train, Y.train, s_f = 0.3, s_s = 1, lf =1, M.train = 5000, nu = 0.001)
```

Arguments

X.train	Input N-by-p feature matrix of training samples. Columns correspond to features, rows correspond to samples.
Y.train	Input N-element response vector of training samples.
s_f	Sampling rate of features, 0<s_f<=1. Fraction of columns from X.train, which will be sampled without replacement to calculate each extesion in boosting model. By default it's 0.3.
s_s	Sampling rate of samples, 0<s_s<=1. Fraction of rows from X.train, which will be sampled with replacement to calculate each extension in boosting model. By default it's 1.
lf	Loss function: 1-> Least Squares and 2 -> Least Absolute Deviation

M.train	Number of extensions in boosting model, e.g. number of iterations of the main loop of RGBM algorithm. By default it's 5000.
nu	Shrinkage factor, learning rate, 0<nu<=1. Each extension to boosting model will be multiplied by the learning rate. By default it's 0.001.

Value

Regression model is a structure containing all the information needed to predict response for unseen data

Author(s)

Raghvendra Mall <rmall@hbku.edu.qa>

See Also

GBM.test

---

get_colids *Get the indices of recitifed list of Tfs for individual target gene*

Description

This function is used to identify the recitified list of transcription factors for individual target genes after analysing the variable importance scores (where non-essential Tfs are pruned). These list of Tfs are usually different for individual target genes. Hence we maintain this in the form an adjacency matrix where the rownames correspond to all the Tfs and colnames correspond to all the target genes. Each column is a binary vector where all the values corresponding to the rectified Tfs active for that target are 1 while rest of the values are zeros.

Usage

```
get_colids(A, ideal_k, tfs, targets, Ntfs, Ntargets)
```

Arguments

A	Adjacency Matrix A obtained after the GBM and refinement step.
ideal_k	A vector containing the optimal value of k (no of active TFs) for each target gene obtained from select_ideal_k.
tfs	List of names of transcription factors.
targets	List of names of target genes.
Ntfs	Total number of transcription factors used in the experiment.
Ntargets	Total number of target genes used in the experiment.

Value

The function returns an adjacency matrix where the rownames correspond to all the Tfs and colnames correspond to all the target genes. Each column is a binary vector where all the values corresponding to the rectified Tfs active for that target are 1 while rest of the values are zeros.

Author(s)

Raghvendra Mall <rmall@hbku.edu.qa>

See Also

get_tf_indices

---

get_filepaths *Generate filepaths to maintain adjacency matrices and images*

Description

This function generates a set of filepaths which are used to keep the adjacency matrix A obtained after the first_GBM_step + null_model_refinement_step. It also generates a path where an image of the variable importance curves for several target genes can be kept.

Usage

```
get_filepaths(A_prev, experimentid, outputpath, sample_type)
```

Arguments

A_prev	Adjacency matrix A obtained after first_GBM_step + null_model_refinement_step.
experimentid	The id of the experiment being conducted. It takes natural numbers like 1,2,3 etc. By default it's 1.
outputpath	Location where the Adjacency_Matrix and Images folder will be created.
sample_type	String arguement representing a label for the experiment i.e. in case of DREAM3 challenge sample_type="DREAM3".

Value

Returns a data frame where the first element in the data frame is the location where the Adjacency_Matrix folder is located in the filesystem, second element represents the location where the Images folder is located in the filesystem, third element represents the path to the file where the Adjacency_Matrix will be written.

Author(s)

Raghvendra Mall <rmall@hbku.edu.qa>

get_ko_experiments *Get indices of experiments where knockout or knockdown happened*

Description

This function provides the indices of all those samples (out of N) where it is known apriori that a gene was either knocked-out or was knocked-down. This information is useful for the null_model_refinement_step which utilizes the z_score_effect technique (with the help of this information).

Usage

```
get_ko_experiments(K)
```

Arguments

K	N-by-p initial perturbation matrix. It directly corresponds to E matrix, e.g. if K[i,j] is equal to 1, it means that gene j was knocked-out in experiment i. Single gene knock-out experiments are rows of K with only one value 1. Colnames of K is set to be the set of all genes. By default it's a matrix of zeros of the same size as E, e.g. unknown initial perturbation state of genes.

Value

Return a vector containing the indices of all the samples where a gene was knocked-out/down.

Author(s)

Raghvendra Mall <rmall@hbku.edu.qa>

See Also

null_model_refinement_step, z_score_effect

get_tf_indices *Get the indices of all the TFs from the data*

Description

This function provides the indices of all the transcription factors which are present in the expression matrix. In case of DREAM Challenges it will return the indices as 1,...,p for all the p genes in the data as the transcription factors are not known beforehand.

Usage

```
get_tf_indices(E, tfs, Ntfs)
```

Arguments

E	E is the expression matrix of size N x p where N is number of examples and p is the number of genes. Here the column names of expression matrix is the list of all the genes present in the E matrix. Colnames of E is the set of all genes.
tfs	List of names of transcription factors.
Ntfs	Total number of transcription factors used in the experiment.

Value

Returns the indices of all the transcription factors present in E matrix.

Author(s)

Raghvendra Mall <rmall@hbku.edu.qa>

See Also

get_colids

---

normalize_matrix_colwise

Column normalize the obtained adjacency matrix

Description

We perform a column normalization on an adjacency matrix A equivalent to inferred GRN

Usage

```
normalize_matrix_colwise(A,Ntargets)
```

Arguments

A	Inferred GRN in the form of Ntfs-by-Ntargets matrix
Ntargets	Total number of target genes used in the experiment

Value

Column Normalized GRN of size Ntfs-by-Ntargets

Author(s)

Raghvendra Mall <rmall@hbku.edu.qa>

null_model_refinement_step
Perform the null model refinement step

Description

We used this function for refining the edge-weights in an inferred GRN (A) by utilizing matrix (S2) obtained from null-mutant zscore effect (z_score_effect) as shown in *Slawek J, Arodz T* i.e. A = A x S2.

Usage

```
null_model_refinement_step(E, A, K,tfs, targets, Ntfs, Ntargets)
```

Arguments

E	N-by-p expression matrix. Columns correspond to genes, rows correspond to experiments. E is expected to be already normalized using standard methods, for example RMA. Colnames of E is the set of all genes.
A	Intermediate GRN network in the form of a p-by-p adjacency matrix.
K	N-by-p initial perturbation matrix. It directly corresponds to E matrix, e.g. if K[i,j] is equal to 1, it means that gene j was knocked-out in experiment i. Single gene knock-out experiments are rows of K with only one value 1. Colnames of K is set to be the set of all genes. By default it's a matrix of zeros of the same size as E, e.g. unknown initial perturbation state of genes.
tfs	List of names of transcription factors
targets	List of names of target genes
Ntfs	Number of transcription factors used while building the GBM (GBM) model.
Ntargets	Number of targets used while building the GBM (GBM) model.

Value

Returns a refined adjacency matrix A in the form of a Ntfs-by-Ntargets matrix.

Author(s)

Raghvendra Mall <rmall@hbku.edu.qa>

References

Slawek J, Arodz T. ENNET: inferring large gene regulatory networks from expression data using gradient boosting. BMC systems biology. 2013 Oct 22;7(1):1.

See Also

z_score_effect

`regularized_GBM_step` *Perform the regularized GBM modelling once the initial GRN is inferred*

Description

This function undertakes all the proposed steps for regularizing the list of transcription factors for individual target gene followed by re-iterating through the core GBM model and the refinement step to produce the final reverse engineered GRN.

Usage

```
regularized_GBM_step(E, A_prev, K, tfs, targets, Ntfs, Ntargets, lf, M, nu, s_f,
                     experimentid, outputpath, sample_type, mink=0,real=0)
```

Arguments

`E`	N-by-p expression matrix. Columns correspond to genes, rows correspond to experiments. E is expected to be already normalized using standard methods, for example RMA. Colnames of E is the set of all genes.
`A_prev`	An intermediate inferred GRN obtained from `first_GBM_step`
`K`	N-by-p initial perturbation matrix. It directly corresponds to E matrix, e.g. if K[i,j] is equal to 1, it means that gene j was knocked-out in experiment i. Single gene knock-out experiments are rows of K with only one value 1. Colnames of K is set to be the set of all genes. By default it's a matrix of zeros of the same size as E, e.g. unknown initial perturbation state of genes.
`tfs`	List of names of transcription factors.
`targets`	List of names of target genes.
`Ntfs`	Total number of transcription factors used in the experiment.
`Ntargets`	Total number of target genes used in the experiment
`lf`	Loss Function: 1 -> Least Squares and 2 -> Least Absolute Deviation
`M`	Number of extensions in boosting model, e.g. number of iterations of the main loop of RGBM algorithm. By default it's 5000.
`nu`	Shrinkage factor, learning rate, 0<nu<=1. Each extension to boosting model will be multiplied by the learning rate. By default it's 0.001.
`s_f`	Sampling rate of transcription factors, 0<s_f<=1. Fraction of transcription factors from E, as indicated by `tfs` vector, which will be sampled without replacement to calculate each extesion in boosting model. By default it's 0.3.
`experimentid`	The id of the experiment being conducted. It takes natural numbers like 1,2,3 etc. By default it's 1.
`outputpath`	Location where the Adjacency_Matrix and Images folder will be created.
`sample_type`	String arguement representing a label for the experiment i.e. in case of DREAM3 challenge sample_type="DREAM3".

mink	User specified threshold i.e. the minimum number of Tfs to be considered while optimizing the L-curve criterion. By default it's 0.
real	Numeric value 0 or 1 corresponding to simulated or real experiment respectively.

Value

Returns the final inferred GRN in form of Ntfs-by-Ntargets matrix

Author(s)

Raghvendra Mall <rmall@hbku.edu.qa>

See Also

first_GBM_step

---

regulate_regulon_size *Regulate the size of the regulon for each TF*

Description

We control the size of the regulon for each TF by using a heuristic to remove the edges whose weights are small

Usage

```
regulate_regulon_size(A)
```

Arguments

A	Inferred GRN in the form of Ntfs-by-Ntargets matrix

Value

Refined adjacency matrix A in the form of Ntfs-by-Ntargets matrix

Author(s)

Raghvendra Mall <rmall@hbku.edu.qa>

RGBM *Regularized Gradient Boosting Machine for inferring GRN*

Description

This function performs the proposed regularized gradient boosting machines for reverse engineering GRN. It allows the user to provide prior information in the form of a mechanistic network g_M and after generation of an initially inferred GRN using the core GBM model undergoes a pruning step. Here we detect and remove isolated nodes using the `select_ideal_k` function along with identification of the optimal set of transcription factors for each target gene. We then re-iterate through the GBM followed by the refinement step to generate the final re-constructed GRN.

Usage

```
RGBM(E = matrix(rnorm(100), 10, 10), K = matrix(0, nrow(E), ncol(E)),
     g_M = matrix(1, 10, 10), tfs = paste0("G", c(1:10)),
     targets = paste0("G", c(1:10)), lf = 1, M = 5000, nu = 0.001, s_f = 0.3,
     no_iterations = 2, mink = 0, experimentid = 1, outputpath= "DEFAULT",
     sample_type = "Exp1_", real = 0)
```

Arguments

E	N-by-p expression matrix. Columns correspond to genes, rows correspond to experiments. E is expected to be already normalized using standard methods, for example RMA. Colnames of E is the set of all p genes and Ntfs represents the number of transcription factors and Ntargets represents the number of target genes.
K	N-by-p initial perturbation matrix. It directly corresponds to E matrix, e.g. if K[i,j] is equal to 1, it means that gene j was knocked-out in experiment i. Single gene knock-out experiments are rows of K with only one value 1. Colnames of K is set to be the set of all genes. By default it's a matrix of zeros of the same size as E, e.g. unknown initial perturbation state of genes.
g_M	Initial mechanistic network in the form of an adajcency matrix (Ntf-by-Ntargets). Here each column is a binary vector where only those elements are 1 when the corresponding transcription factor has a connection with that target gene. Colnames of g_M should be same as names of targets and Rownames of g_M should be same as names of Tfs. By default it's a matrix of ones of size Ntfs x Ntargets.
tfs	List of names of transcription factors
targets	List of names of target genes
lf	Loss Function: 1 -> Least Squares and 2 -> Least Absolute Deviation
M	Number of extensions in boosting model, e.g. number of iterations of the main loop of RGBM algorithm. By default it's 5000.
nu	Shrinkage factor, learning rate, 0<nu<=1. Each extension to boosting model will be multiplied by the learning rate. By default it's 0.001.

s_f	Sampling rate of transcription factors, 0<s_f<=1. Fraction of transcription factors from E, as indicated by tfs vector, which will be sampled without replacement to calculate each extesion in boosting model. By default it's 0.3.
no_iterations	Number of times initial GRN to be constructed and then averaged to generate smooth edge weights for the initial GRN as shown in first_GBM_step
mink	specified threshold i.e. the minimum number of Tfs to be considered while optimizing the L-curve criterion. By default it's 0.
experimentid	The id of the experiment being conducted. It takes natural numbers like 1,2,3 etc. By default it's 1.
outputpath	Location where intermediate Adjacency_Matrix and Images folder will be created. By default it's a temp directory (e.g. /tmp/Rtmp...)
sample_type	String arguement representing a label for the experiment i.e. in case of DREAM3 challenge sample_type="DREAM3".
real	Numeric value 0 or 1 corresponding to simulated or real experiment respectively.

Value

Returns the final inferred GRN of form Ntfs-by-Ntargets adjacency matrix.

Author(s)

Raghvendra Mall <rmall@hbku.edu.qa>

See Also

select_ideal_k, first_GBM_step

Examples

```
# load RGBM library
library("RGBM")
# this step is optional, it helps speed up calculations, run in parallel on 2 processors
library(doParallel)
cl <- makeCluster(2)
# run network inference on a 100-by-100 dummy expression data.
A = RGBM()
stopCluster(cl)
```

RGBM.test *Test rgbm predictor*

Description

This function tests a regression model for a given X.test feature matrix, Y.test response vector, and working parameters.

Usage

```
RGBM.test(model, X.test, Y.test, M.test)
```

Arguments

model	Model returned by RGBM.train function.
X.test	Input S-by-P feature matrix of unseen samples. Columns correspond to features, rows correspond to samples.
Y.test	Input S-element response vector of unseen samples.
M.test	Number of extensions of boosting model to take when predicting response. Must be not greater than M.train used when training boosting model.

Value

Result of regression

Author(s)

Raghvendra Mall <raghvendra5688@gmail.com>

RGBM.train *Train RGBM predictor*

Description

This function trains a regression model for a given X.train feature matrix, Y.train response vector, and working parameters. A model returned by this function can be used to predict response for unseen data with RGBM.test function.

Usage

```
RGBM.train(X.train, Y.train, s_f = 0.3, s_s = 1, lf = 1, M.train = 5000, nu = 0.001)
```

Arguments

X.train	Input S-by-P feature matrix of training samples. Columns correspond to features, rows correspond to samples.
Y.train	Input S-element response vector of training samples.
s_f	Sampling rate of features, 0<s_f<=1. Fraction of columns from X.train, which will be sampled without replacement to calculate each extesion in boosting model. By default it's 0.3.
s_s	Sampling rate of samples, 0<s_s<=1. Fraction of rows from X.train, which will be sampled with replacement to calculate each extension in boosting model. By default it's 1.
lf	Loss function: 1-> Least Squares and 2 -> Least Absolute Deviation

M.train	Number of extensions in boosting model, e.g. number of iterations of the main loop of RGBM algorithm. By default it's 5000.
nu	Shrinkage factor, learning rate, 0<nu<=1. Each extension to boosting model will be multiplied by the learning rate. By default it's 0.001.

Value

Regression model is a structure containing all the information needed to predict response for unseen data

Author(s)

Raghvendra Mall <raghvendra5688@gmail.com>

`second_GBM_step` *Re-iterate through the core GBM model building with optimal set of Tfs for each target gene*

Description

This function re-performs the core GBM model building (only one time) using the optimal set of transcription factors obtained from `select_ideal_k` followed by `get_colids` for individual target gene to return a regularized GRN.

Usage

```
second_GBM_step(E, K, df_colids, tfs, targets, Ntfs, Ntargets, lf,  M,  nu, s_f)
```

Arguments

E	N-by-p expression matrix. Columns correspond to genes, rows correspond to experiments. E is expected to be already normalized using standard methods, for example RMA. Colnames of E is the set of all genes.
K	N-by-p initial perturbation matrix. It directly corresponds to E matrix, e.g. if K[i,j] is equal to 1, it means that gene j was knocked-out in experiment i. Single gene knock-out experiments are rows of K with only one value 1. Colnames of K is set to be the set of all genes. By default it's a matrix of zeros of the same size as E, e.g. unknown initial perturbation state of genes.
df_colids	A matrix made up of column vectors where each column vector represents the optimal set of active Tfs which regulate each target gene and obtained from `get_colids`. Some column vectors are just made up of zeros indicating that corresponding target genes are isolated and not regulated by any Tf
tfs	List of names of transcription factors.
targets	List of names of target genes.
Ntfs	Total number of transcription factors used in the experiment.

Ntargets	Total number of target genes used in the experiment
lf	Loss Function: 1 -> Least Squares and 2 -> Least Absolute Deviation
M	Number of extensions in boosting model, e.g. number of iterations of the main loop of RGBM algorithm. By default it's 5000.
nu	Shrinkage factor, learning rate, 0<nu<=1. Each extension to boosting model will be multiplied by the learning rate. By default it's 0.001.
s_f	Sampling rate of transcription factors, 0<s_f<=1. Fraction of transcription factors from E, as indicated by tfs vector, which will be sampled without replacement to calculate each extesion in boosting model. By default it's 0.3.

Value

Returns a regularized GRN of the form Ntfs-by-Ntargets

Author(s)

Raghvendra Mall <rmall@hbku.edu.qa>

See Also

first_GBM_step

---

select_ideal_k *Identifies the optimal value of k i.e. top k Tfs for each target gene*

Description

This function detects the optimal number of transcription factors which are regulating each target gene. This number is different for different target genes. It utilizes a heuristic to also detect the isolated targets which are not regulated by any transcription factor. To the detect the optimal number of Tfs for each target gene, it uses a notion similar to that used for optimization of the L-curve criterion for Tikonov regularization by evaluating the variable importance curve for each target gene.

Usage

```
select_ideal_k(experimentid, mink, filepath, imagepath, adjacency_matrix_path)
```

Arguments

experimentid	The id of the experiment being conducted. It takes natural numbers like 1,2,3 etc. By default it's 1.
mink	User specified threshold i.e. the minimum number of Tfs to be considered while optimizing the L-curve criterion. By default it's 0.
filepath	Path where some intermediate files will be written and provided by the function get_filepaths.

`imagepath` Path where an image of the variable importance curves for first 16 target genes will be written and provided by the function `get_filepaths`.

`adjacency_matrix_path`
Path where an intermediate adjacency matrix will be written and provided by the function `get_filepaths`.

Value

Returns a vector where each element represents the optimal number of transcription factors for each target gene.

Author(s)

Raghvendra Mall <rmall@hbku.edu.qa>

`test_regression_stump_R`
Test the regression model

Description

Test the regression model for each target gene

Format

The format is: List of 4 $ name : chr "test_regression_stump_R" $ address :Class 'RegisteredNativeSymbol' <externalptr> $ dll :List of 5 ..$ name : chr "RGBM" ..$ path : chr "/home/raghvendra/R/x86_64-pc-linux-gnu-library/3.3/RGBM/libs/RGBM.so" ..$ dynamicLookup: logi TRUE ..$ handle :Class 'DLLHandle' <externalptr> ..$ info :Class 'DLLInfoReference' <externalptr> ..- attr(*, "class")= chr "DLLInfo" $ numParameters: int 15 - attr(*, "class")= chr [1:2] "CRoutine" "NativeSymbolInfo"

Author(s)

Raghvendra Mall <rmall@hbku.edu.qa>

`train_regression_stump_R` *Train the regression stump*

Description

Train the regression stump for each target gene

Format

The format is: List of 4 $ name : chr "train_regression_stump_R" $ address :Class 'RegisteredNativeSymbol' <externalptr> $ dll :List of 5 ..$ name : chr "RGBM" ..$ path : chr "/home/raghvendra/R/x86_64-pc-linux-gnu-library/3.3/RGBM/libs/RGBM.so" ..$ dynamicLookup: logi TRUE ..$ handle :Class 'DLLHandle' <externalptr> ..$ info :Class 'DLLInfoReference' <externalptr> ..- attr(*, "class")= chr "DLLInfo" $ numParameters: int 15 - attr(*, "class")= chr [1:2] "CRoutine" "NativeSymbolInfo"

Author(s)

Raghvendra Mall <rmall@hbku.edu.qa>

`transform_importance_to_weights` *Log transforms the edge-weights in the inferred GRN*

Description

This function performs an inverse absolute log-transformation of the non-zero edge weights in the final inferred GRN (A) to make the edge-weights more comprehensible and understandable.

Usage

```
transform_importance_to_weights(A)
```

Arguments

`A`	Inferred GRN in the form of Ntfs-by-Ntargets matrix

Value

Refined adjacency matrix A in the form of Ntfs-by-Ntargets matrix

Author(s)

Raghvendra Mall <rmall@hbku.edu.qa>

v2l — *Convert adjacency matrix to a list of edges*

Description

This function converts adjacency matrix A to a sorted list of edges, e.g. a list in which edges are sorted by decreasing confidence.

Usage

```
v2l(A, max = 1e+05, check.names = TRUE)
```

Arguments

A	Input adjacency matrix.
max	Maximal length of the resulting list. This number may be lower than the number of all the edges from adjacency matrix. Then only top max edges will be returned.
check.names	Checks name of the gene ids

Value

A data frame of sorted edges: (1) list of sources (2) list of destinations (3) list of confidences. Elements in all the lists correspond to each other.

Author(s)

Raghvendra Mall <rmall@hbku.edu.qa>

z_score_effect — *Generates a matrix S2 of size Ntfs x Ntargets using the null-mutant zscore algorithm Prill, Robert J., et al*

Description

This function generates a matrix of the form Ntfs-by-Ntargets using the steps proposed in null-mutant zscore method and acts as a refinement step for the inferred GRN where this matrix is multiplied element by element with the inferred adjacency matrix A. However, this step is only effective in presence of additional source of information like knockout, knockdown or which genes are intially perturbed in time-series expression data.

Usage

```
z_score_effect(E, K, tfs, targets, Ntfs, Ntargets)
```

Arguments

E	N-by-p expression matrix. Columns correspond to genes, rows correspond to experiments. E is expected to be already normalized using standard methods, for example RMA. Colnames of E is the set of all genes.
K	N-by-p initial perturbation matrix. It directly corresponds to E matrix, e.g. if K[i,j] is equal to 1, it means that gene j was knocked-out in experiment i. Single gene knock-out experiments are rows of K with only one value 1. Colnames of K is set to be the set of all genes. By default it's a matrix of zeros of the same size as E, e.g. unknown initial perturbation state of genes.
tfs	List of names of transcription factors
targets	List of names of target genes
Ntfs	Total number of transcription factors used in the experiment.
Ntargets	Total number of target genes used in the experiment.

Value

Returns an S2 matrix of form Ntfs-by-Ntargets. In absence of any additional knockout/knockdown/perturbation information the S2 matrix is a matrix of ones.

Author(s)

Raghvendra Mall <rmall@hbku.edu.qa>

References

Prill, Robert J., et al. "Towards a rigorous assessment of systems biology models: the DREAM3 challenges." PloS one 5.2 (2010): e9202.

See Also

null_model_refinement_step

Index