Package 'LncMod'

June 24, 2015

Type Package

Title Predicting Modulator and Functional/Survival Analysis

Version 1.0

Author Yongsheng Li,Zishan Wang,Juan Xu*,Xia Li*

Maintainer Zishan Wang<azureskyeric@gmail.com>

Description Predict modulators regulating the ability of effectors to regulate their targets and produce modulator-effctor-target triplets followed by goterm functional enrichment and survival analysis.

License GPL-2

LazyData TRUE

Depends R (>= 2.15.2),survival,pheatmap,parallel

R topics documented:

`LncMod-package` *LncMod*

Description

Modulator prediction and functional/survival analysis

datatests *Data for Examples*

Description

This object contains data for examples.

Format

A list with 18 variables:

m_app1 a lncRNA (long non-coding RNA) name in the *M_exp* which can affect the ability of effectors to regulate their corresponding targets.

m_app2 two lncRNA names in the *M_exp* which can affect the ability of effectors to regulate their corresponding targets.

m_app3 three lncRNA names in the *M_exp* in which,one can affect the ability of effectors to regulate their corresponding targets,another expression is not in the initial expression profile,the other expression is in the initial expression profile but not in the filterd expression profile by IQR.

ET a dataframe representing the TF-target regulations in glioblastoma.It includes 200 rows standing for 200 pairs of regulations and 2 columns(the first is the TF and the second is the corresponding target).

M_exp a dataframe representing the expression profile of lncRNAs in glioblastoma.It includes 4 rows standing for 4 lncRNAs and 451 columns standing for 451 samples.

E_exp a dataframe representing the expression profile of some transcription factors (TFs) in glioblastoma.It includes 14 rows standing for 14 transcription factors (TFs) and 451 columns standing for 451 samples.

T_exp a dataframe representing the expression profile of some targets in glioblastoma.It includes 158 rows standing for 158 targets and 451 columns standing for 451 samples.

tri_basic1 a lncRNA-TF-target (modulator-effector-target) triplet whose factors are in their corresponding expression profiles(M_exp,E_exp,T_exp),respectively.

tri_basic2 two lncRNA-TF-target (modulator-effector-target) triplets whose factors are in their corresponding expression profiles(M_exp,E_exp,T_exp),respectively.

tri_enrich a dataframe representing the triplets in glioblastoma.It includes 433 rows standing for different triplets and 3 columns(the first is lncRNA(Ensemble id),the second is TF and the third is target(Entrez id)).

background a integer vector representing all genes(18292) entrezid(consistent with the id format in GOterms) in glioblastoma expression profiles.

GOterms List with 34 goterms and corresponding genes entrez id.

GOterms_mark a dataframe with two columns(the first is goterms to enrich,the second is cancerhallmarks to which the goterms belong).

tri_survival1 a lncRNA-TF-target (modulator-effector-target) triplet whose factors are in the expression and survival profile(exp_sur).

tri_survival2 two lncRNA-TF-target (modulator-effector-target) triplets whose factors are in the expression and survival profile(exp_sur).

exp_sur a dataframe representing the expression and survival information.It includes 426 rows standing for the samples (rownames are sample tags) and 6 columns(the top 4 columns are molecular names while the last 2 columns are the survival information).

train a character vector representing 213 trainning sample tags(There is no significant difference between trainning samples and testing samples).

test a character vector representing 212 test sample tags.

Details

All the data is from a study about glioblastoma.

Source

Zhou Du,T. et al. *Integrative genomic analyses reveal clinically relevant long noncoding RNAs in human cancer* nature structural & molecular biology,**20**, 908-913 (2013).

`tri.app` *Modulators Prediction*

Description

This function predicts a modulator affecting the ability of a effector to regulate its targets based on expression profiles.

Usage

```
tri.app(ms, ET, M.exp, E.exp, T.exp, N = 0.25, method = "pearson",
       iqr.filter = c(log2(1.5), log2(1.5), log2(1.5)),
       cor.MvsET = c(0.3, 0.3), cor.EvsT.dif = 0.45, cor.EvsT.abs = 0.4,
       ET.fc.filter = log2(1.5) , ET.p.filter = 0.01 ,
       rand = 100,correction="BH",cores=1)
```

Arguments

`ms`	a character string (or a character string vector) specifying the candidate modulator names to predict.
`ET`	a dataframe representing the effector-target regulations in which factors are effector/target names.
`M.exp`	a numeric dataframe (or matrix) representing the expression profile of candidate modulator whose rownames is the names of the candidate modulators.
`E.exp`	a numeric dataframe (or matrix) representing the expression profile of effectors.Its rownames are names of the effectors.
`T.exp`	a numeric dataframe (or matrix) representing the expression profile of targets.Its rownames are names of the targets.
`N`	a numeric(rangs from 0 to 0.5) specifying the proportion of the samples to use for dividing samples into LOW/HIGH level of modulator expression.
`method`	a character string (default "pearson") indicating which correlation coefficient (or covariance) is to be computed. One of "pearson" (default), "kendall", or "spearman", can be abbreviated.

iqr.filter	a numeric vector of the form c(modulator_iqr,effector_iqr,target_iqr) which specifies the iqr threshold to filter expression profiles(default (log2(1.5), log2(1.5), log2(1.5))).
cor.MvsET	a numeric vector of the form c(cor.MvsE,corMvsT) which specifies the threshold for the correlation between modulator and effector/target(default (0.3, 0.3)).
cor.EvsT.dif	a numeric(default 0.45) representing the threshold for the difference between the effector-target correlation in the HIGH and LOW proportion of the samples.
cor.EvsT.abs	a numeric(default 0.4) representing the threshold for the effector-target correlation either in the HIGH proportion of samples or the LOW.
ET.fc.filter	a numeric representing the threshold for the fold change (fc) of the diffrent expression of the effectors in the HIGH versus LOW proportion of samples(default log2(1.5)).
ET.p.filter	a numeric representing the threshold for p value of the diffrent expression of the effectors in the HIGH versus LOW proportion of samples(default 0.01).
rand	a numeric specifying the number of disturbance (default 100).
correction	correction method (default "BH") in one of p.adjust.methods.
cores	The number of cores (default 1) to use, i.e. at most how many child processes will be run simultaneously. Must be exactly 1 on Windows (which uses the master process).

Details

Note:The arguments without default value must be assigned.

This function running a time checked whether a modulator in a sets,one by one,can affect the ability of a effector sets to regulate their corresponding targets. Please go to *Kai Wang,M. et al. Genome-wide identification of post-translational modulators of transcription factor activity in human B cells. Nature biotechnology,**27**, 829-837 (2009)* for detailed information.

The running time and the memory required was increasing as the possible triplets increased.To speed-up the analysis,the function implemented parallel computations when runned on multi-core machines. It used mclapply function in the parallel package to make use of all the CPUs available on the system, with each core simultaneously performing part of the runs.If the possible triplets are big,please work on a big memory machine.

Value

It returned a list containing following components:

- `triplets` predicted triplets and related information,a 7 columns dataframe as following:
  - `modulator effector target` represented the modulator/effector/target names,respectively;
  - `R_low R_high` the effector-target correlation in the LOW/HIGH proportion of samples,respectively;
  - `P_value` the significance of the triplet;
  - `fdr` corrected P_value by the assigned method;
- `initialnot` the names of modulators whose expression is not in the initial expression profile which is assigned by the parameter *M.exp*;
- `filterdnot` the names of modulators whose expression is in the initial expression profile which is assigned by the parameter *M.exp* but not in the filterd profile by IQR;

Examples

```
##One candidate modulator to predict
tri.app(ms=datatests[["m_app1"]],ET=datatests[["ET"]],M.exp=datatests[["M_exp"]],
        E.exp=datatests[["E_exp"]],T.exp=datatests[["T_exp"]])
##Two candidate modulators(or more) to predict
tri.app(ms=datatests[["m_app2"]],ET=datatests[["ET"]],M.exp=datatests[["M_exp"]],
        E.exp=datatests[["E_exp"]],T.exp=datatests[["T_exp"]])
##Different types of candidate modulators to predict
##Here we use 3 modulators for examle,one for one type of modulator
tri.app(ms=datatests[["m_app3"]],ET=datatests[["ET"]],M.exp=datatests[["M_exp"]],
        E.exp=datatests[["E_exp"]],T.exp=datatests[["T_exp"]])
```

---

tri.basic *Overview of Triplets' Expression*

Description

Plot to show the expression for a factor in a triplet in the prediction algorithm.This can also work on triplets at a time.

Usage

```
tri.basic(tri, M.exp, E.exp, T.exp, index = 1, N = 0.25)
```

Arguments

tri	a dataframe or matrix representing the triplets to plot in which the first column is the modulator,the second is the effector and the third column is the target.
M.exp	a numeric dataframe (or matrix) representing the expression profile of modulators whose rownames is the names of the modulators.
E.exp	a numeric dataframe (or matrix) representing the expression profile of effectors whose rownames is the names of the effectors.
T.exp	a numeric dataframe (or matrix) representing the expression profile of targets whose rownames is the names of the targets.
index	a numeric vector (default 1) specifying the rowindex of candidate triplets to plot.
N	a numeric(rangs from 0 to 0.5,default 0.25) specifying the proportion of the samples to use for dividing samples into LOW/HIGH level of modulator expression.

Details

Note:The arguments without default value must be assigned.

For each triplet,the plot consists of 3 parts.

The first is a barplot showing expression of the effector in HIGH and LOW proportion of the samples. The second is a scatter diagram with a linear fitted line using function *lm*.The formmer shows the expression of target versus effector in the HIGH and LOW proportion of the samples in which red is HIGH while green is LOW.The latter is a linear model result. The third is a barplot showing expression of the target in HIGH and LOW proportion of the samples.

Value

tri.basic returns a dataframe whose rownames is the row index of the triplets and columns represent 6 kinds of value of the plot (P_effector_target,P_effector,P_target,R_low,R_high,P_low,P_high).

- `P_effector_target` the significance of the linear fitted lines;
- `P_effector` the significance of the difference between the effector expression in the HIGH and LOW proportion of the samples;
- `P_target` the significance of the difference between the target expression in the HIGH and LOW proportion of the samples;
- `R_low` the effector-target correlation in the LOW proportion of the samples;
- `R_high` the effector-target correlation in the HIGH proportion of the samples;
- `P_low` the significance of R_low;
- `P_high` the significance of R_high;

Examples

```
#One triplet to show
tri.basic(tri=datatests[["tri_basic1"]],M.exp=datatests[["M_exp"]],
          E.exp=datatests[["E_exp"]],T.exp=datatests[["T_exp"]])
#Two triplets(or more) to show
tri.basic(tri=datatests[["tri_basic2"]],M.exp=datatests[["M_exp"]],
          E.exp=datatests[["E_exp"]],T.exp=datatests[["T_exp"]],index=c(1,2))
```

---

tri.enrich *Modulator Functional Enrichment*

Description

Targets of a modulator in the triplets is enriched to GOterms based on the hypergeometric distribution.It can also do enrichment analysis on disease hallmarks at the same time.

Usage

```
tri.enrich(tri, GOterms, inter.thr = 2, background = NULL, GOterms.mark = NULL,correction="BH")
```

Arguments

`tri`	a dataframe or matrix representing the triplets used to enrich.The first column is modulator;the second column is effetor;the third column is target.
`GOterms`	a list.The name of each variable is a GOterm name and the content is genes in the GOterm.
`inter.thr`	a numeric(default 2) representing the min numerber of intersection between modulator's targets and the GOterms genes.
`background`	a vector (default NULL) containing a set of targets from which the GOterm must be filterd.
`GOterms.mark`	a dataframe or matrix (default NULL) with 2 columns in which the first represents the GOterm sets to be enriched while the second represents hallmark to which the GOterm belongs;
`correction`	correction method(default "BH") in one of `p.adjust.methods`.

Details

Note:The arguments without default value must be assigned.

If the background is NULL,then the targets of a modulator is enriched to the GOterms genes passed in; If the background is not NULL,then the targets of a modulator is enriched to the GOterms genes filterd by the background.

If the GOterms.mark is NULL,it only do the GOterms enrichment; If the GOterms.mark is not NULL,it do enrichment analysis on both GOterms and disease hallmarks.

Value

If the GOterms.mark is NULL,it is a 6 column dataframe as following:

- `modulator` the modulator name;
- `goterm` the GOterm name;
- `mtarnum` the target number of a modulator;
- `gotarnum` the gene number of a GOterm;
- `internum` the number of the intersected factor between a GOterm genes and a modulator targets;
- `P_value` the significance of the enrichment;
- `fdr` corrected P_value by the assigned method;

If the GOterms.mark is not NULL,it added a seventh column(named "mark" representing the disease mark) besides six columns above.

See Also

`phyper`

Examples

```
#Functional enrichment without disease hallmarks
tri.enrich(tri=datatests[["tri_enrich"]],GOterms=datatests[["GOterms"]],
           background=datatests[["background"]])
#Funtional enrichment with disease hallmarks
tri.enrich(tri=datatests[["tri_enrich"]],GOterms=datatests[["GOterms"]],
          background=datatests[["background"]],GOterms.mark=datatests[["GOterms_mark"]])
```

---

tri.surv *Survival Analysis for Triplets*

Description

It will generate plots describing the expression and survival comparison for train/test sample groups of a triplet. This can also work on triplets at a time.

Usage

```
tri.surv(tri, exp.sur, train, test, index = 1)
```

Arguments

`tri`	a character matrix or dataframe specify the triplets in which the first column is the modulator,the second is the effector,the third is the target.
`exp.sur`	a dataframe whose rownames are the sample names and the colnames are the factor names in the triplets.It must contain the survival information additionally(see the example data in details).
`train`	a character string vector specifying the train sample names.
`test`	a character string vector specifying the test sample names.
`index`	a numeric vector (default 1) reprsenting the rowindex of the triplets to be analyed.

Details

Note:The arguments without default value must be assigned.

For the output,a triplet corresponds 6 plots,3 for train samples and 3 for test samples.For train sameples,one plot is to show the expression of triplet,another is to show the risk scores,the other is the comparison of survivorship curve between highrisk and lowrisk sameples.

Value

A dataframe whose rows represent different triplets while columns represent 15 kinds of information on the triplet. The columns are:

- `modulator` the modulator name;
- `effector` the effector name;
- `target` the target name;
- `coef_modulator` the *coxph* coefficient of modulator;
- `p_modulator` the significance of *coef_lnc*;
- `coef_effector` the *coxph* coefficient of effector;
- `p_effector` the significance of *coef_effector*;
- `coef_target` the *coxph* coefficient of target;
- `p_target` the significance of *coef_target*;
- `N_train1` the sample number with low risk score in train sameples;
- `N_train2` the sample number with high risk score in train sameples;
- `dif_train` the significance of survival difference between low/high risk sameples in train samples;
- `N_test1` the sample number with low risk score in test sameples;
- `N_test2` the sample number with high risk score in test sameples;
- `dif_test` the significance of survival difference between low/high risk sameples in test samples;

See Also

`coxph`,`Surv`,`survdiff`

Examples

```
#a triplet to do survival analysis
tri.surv(tri=datatests[["tri_survival1"]],exp.sur=datatests[["exp_sur"]],
         train=datatests[["train"]],test=datatests[["test"]])
#two triplets(or more)
tri.surv(tri=datatests[["tri_survival2"]],exp.sur=datatests[["exp_sur"]],
         train=datatests[["train"]],test=datatests[["test"]],index=c(1,2))
```

Index