TR2PATH R package

Installation and DEMO Manual

1. Overview of TR2PATH algorithm

TR2PATH generates and mines/interrogates a tissue-specific mechanism-centric regulatory network that integrates molecular pathways with their upstream transcriptional regulatory programs (TRs). Nodes in this network are mechanisms: either molecular pathways or TRs; connected by edges that reflect the strength and robustness of their association.

In order to generate the network, TR2PATH estimates the activity levels of molecular pathways and transcriptional regulatory programs in each patient (sample) and then elucidates their association using linear regression (where TR is a predictor variable and pathway is a response variable). To estimate robustness of these associations, bootstrapping analysis is performed (where sampling with replacement is performed 100 times and weight for each edge is estimated as the number of times the edge is recovered across the bootstrapped networks). In the current manuscript, TR2PATH was utilized to reconstruct castration-resistant prostate cancer (CRPC) specific mechanism-centric regulatory network, and mined/interrogated with signatures of resistance to Enzalutamide.

In *mining Step 1* TR2PATH identifies subnetworks of the mechanism-centric network that differentially change between phenotypes of interest. In *mining Step 2*, for each pathway (or for a particular pathway of interest) in a differentially changed sub network, TR2PATH identifies and prioritizes upstream transcriptional regulatory programs and their co llinear groups, based on the strength of their association with the pathway across the phenotypes.

2. Workflow

(A) In ***Preparation Step***, *computePathwayActivity* function and *computeTRActivity* function are utilized to estimate the activity levels/vectors of molecular pathways and TRs in each patient/sample in the cohort.

(B) In ***Reconstruction Step***, pathway and TR activity vectors are utilized to reconstruct a weighted mechanism-centric network using *reconstNetwork* function.

(C) In ***Mining Step 1***, TR2PATH utilizes the *identifySubnetworks* function to identify subnetworks that are differentially altered between phenotypes of interest.

(D) In ***Mining Step 2***, TR2PATH utilizes *computeScores* function, to identify and prioritize upstream TR programs that affect the pathway in a differentially altered subnetwork of interest.

3. Installation of *TR2PATH* package

Dependencies:

R Bioconductor dependency packages required for TR2PATH installation: *reshape2*, *rCAT*, *psych*, *stats*, *foreach*, *doParallel*, *devtools*, *Rcpp*, *parallel*, *viper*, *Hmisc* and *dplyr*

To install these packages, run in RStudio:

```
if (!require("BiocManager", quietly = TRUE))
+     install.packages("BiocManager")

BiocManager::install("viper")
install.packages("reshape2")
install.packages("rCAT")
install.packages("psych")
install.packages("stats")
install.packages("foreach")
install.packages("doParallel")
install.packages("devtools")
install.packages("Rcpp")
install.packages("parallel")
install.packages("dplyr")
```

`install.packages("Hmisc")`**Download:**

(1) Create a folder (for example, "*package*" folder)

(2) Download the source file for TR2PATH (i.e., `TR2PATH_0.2.8.tar.gz` file) to the "*package*" folder, based on either Windows or Mac/LINUX operating system.

(3) Set the working directory of R to the "*package*" folder:

```
      setwd("<path to "package" folder>")
```

*note that while copying a directory in Windows, the path is copied with backward slashes, however R requires them to be changed to forward slashes.

Installation:

TR2PATH can be installed using the following command in R:

```
install.packages("TR2PATH_0.2.8.tar.gz", repos=NULL, type = "source",
dependencies=TRUE)
```

4. Loading *TR2PATH* library and manuscript data objects

After installation, TR2PATH can be loaded as:

```
library(TR2PATH)
```

Data objects generated for the manuscript are automatically loaded as TR2PATH package and could be accessed as TR2PATH::<object name> (see examples of their structures through this manual):

1. TR2PATH::sysdata, contains list of molecular pathways and their corresponding genes from KEGG, BioCarta, REACTOME, and Hallmarks collections.
2. TR2PATH::interactome, contains a prostate-cancer specific transcriptional regulatory network (interactome) reconstructed in Aytes et al.
3. TR2PATH::*TestNESPathwayData* is a data matrix comprising of the activity levels of molecular pathways for all samples in the SU2C East Coast cohort.
4. TR2PATH::*TestTRActivityData* is a data matrix comprising of activity level of each TR program for all samples in the SU2C East Coast cohort.
5. TR2PATH::*FinalWeightedRegNetwork* is a data table, which corresponds to weighted mechanism-centric regulatory network reconstructed from SU2C East Coast cohort; each row corresponds to an edge between a pathway and a TR, with accompanied p-values, FDR, and weight.
6. TR2PATH::*TestSubnetworkList* is a list of subnetworks that are differentially altered between phenotypes that describe Enzalutamide resistance.
7. TR2PATH::*PrioritizedTRsinMYCSubnetwork* is a data table, which corresponds to the priority of each cluster of transcriptional regulatory programs affecting MYC pathway in MYC-centered subnetwork.

Run times to generate these (or new) objects are listed at the end of this document.

6. Tutorial for a sample algorithm run

Sample Data matrix:

For this tutorial, the TR2PATH package includes a sample gene expression matrix (*demoNetwrkReconstExpressionData*) with 26445 genes (rows) and 15 samples (columns).

Examples of the code and outputs are below:

```
expression_mat <- TR2PATH::demoNetwrkReconstExpressionData
expression_mat[1:2,1:5]
```

```
##          PCA0001   PCA0002   PCA0003   PCA0005  PCA0007
## A1BG    8.056938  7.955632  8.287924  7.930118 7.949072
## A1CF    5.387556  5.573194  5.781781  5.338465 5.431128
```

(A) Preparation Step:

*T*o estimate the activity levels of each molecular pathway for each patient/sample in this cohort, TR2PATH utilizes *computePathwayActivity* function.

computePathwayActivity{
Inputs:
(1) scaled gene expression matrix with rows as samples and columns as genes
(2) pathway gene list (if `pathwayGenelist = NULL`, the function will utilize TR2PATH::sysdata pathway object; customized pathway list can be assigned to `pathwayGenelist`).
(3) number of permutations to estimate normalized enrichment scores (NESs) (default = 10).

Output:
pathway activity matrix, where each row is a patient/sample and each column is a molecular pathway.
}

Examples of the code and outputs are below:

```
scaled_Expression_matrix <- scale(t(expression_mat))

Pathway_Activity <- TR2PATH::computePathwayActivity(scaled_Expression_matrix,
pathwayGenelist = NULL, nperms = 10)
Pathway_Activity[1:2,1:2]
```

```
##          KEGG_GLYCOLYSIS_GLUCONEOGENESIS KEGG_CITRATE_CYCLE_TCA_CYCLE
## PCA0001                       -1.036965                   -0.8460733
## PCA0002                       -6.791604                   -1.0427827
```

*note if pathway gene list is NULL, the pathway gene list for KEGG, BioCarta and REACTOME, and Hallmark collections are accessed directly from within the function through TR2PATH::sysdata object. Also note that the pathway related genes in user defined pathway gene list should be their corresponding entrez id.

To estimate the activity level of each TR program in each patient/sample, TR2PATH utilizes *computeTRActivity* function, which utilizes a *viper* function from the *viper* Bioconductor package as a dependency.

computeTRActivity{
Inputs:
(1) scaled gene expression matrix with rows are samples and columns as genes
(2) transcriptional regulatory network (interactome) object

Output:
TR activity matrix, where each column is a patient/sample and each row is a TR program.
}

Examples of the code and outputs are below:

```
interactome <- TR2PATH::interactome
scaled_Expression_matrix <- scale(t(expression_mat))
TR_activity <- TR2PATH::computeTRActivity(scaled_Expression_matrix,
interactome)

TR_activity[1:2,1:5]

##          PCA0001   PCA0002     PCA0003    PCA0005   PCA0007
## AATF   1.2821135 5.2224329 -3.24345675  5.3168063 -3.376217
## ABCG1 -0.2236779 0.8171694 -0.03817329 -1.2147933  2.033459
```

(B) Reconstruction Step:

To reconstruct mechanism-centric regulatory network *reconstNetwork* function is utilized.

reconstNetwork{
Inputs:
(1) pathway activity matrix from step (A)
(2) TR activity matrix from step (A)
(3) FDR cutoff for significance of association between a pathway and a TR (default = 0.05)
(4) TRUE or FALSE for bootstrapping (default = TRUE)
(5) number of bootstraps (default = 10)

Output:
a 6-column data table that represents weighted mechanisms-centric regulatory network (where each row is an edge in the network – see example below)
}

Examples of the code and outputs are below:

```
weighted_network <- TR2PATH::reconstNetwork(Pathway_Activity, TR_activity,
              fdr.cutoff = 0.05,bootstrap = TRUE, bootstrap_count = 10)

weighted_network[1,]

## TRs     Pathway                         Slope  pvalue    fdr  edge_weights
## ABCG4   KEGG_CITRATE_CYCLE_TCA_CYCLE -0.257  2.29-05  0.00126     9
```

(C) Mining Step 1:

Mining of the mechanism-centric regulatory network requires query gene expression matrix with well-defined phenotypes (for example: intact, Enzalutamide sensitive, and Enzalutamide resistant). We provide `demoQueryExpressionData` as an example of such query data matrix, with 34668 genes as rows and 12 samples as columns, and its three distinct phenotypes as `demoQueryPhenos` object.

Examples of the code and outputs are below:

```
queryExpressionMat <- TR2PATH::demoQueryExpressionData
queryExpressionMat[1:2,1:5]
```

```
##           GSM2069494 GSM2069495 GSM2069496 GSM2069497 GSM2069498
## LOC649563       -0.1        3.7        3.2       -1.0       -5.8
## LOC646331       -1.1        1.1       -0.5        3.9        3.2
```

```
phenos <- TR2PATH::demoQueryPhenos
print(phenos)
```

```
## [[1]]
## [1] "GSM2069494" "GSM2069495" "GSM2069496" "GSM2069497"
##
## [[2]]
## [1] "GSM2069498" "GSM2069499" "GSM2069500" "GSM2069501"
##
## [[3]]
## [1] "GSM2069502" "GSM2069503" "GSM2069504" "GSM2069505"
```

First step of the network mining includes identification of differentially altered subnetworks through *identifySubnetworks* function.

identifySubnetworks{
Inputs:
(1) mechanism-centric network obtained from *reconstNetwork function* in step (B)
(2) query gene expression matrix
(3) phenotypes
(4) interactome
(5) pathway gene list (if pathwayGenesets = NULL, TR2PATH::sysdata object is utilized; otherwise, a customized pathway list could be assigned to `pathwayGenesets`)

Output:
Subnetworks of the mechanism-centric network, consisting of the molecular pathways and their upstream TR programs along with their slope, p-value, fdr and edge weights.
}

Examples of the code and outputs are below:

```
subnetworklist = TR2PATH::identifySubnetworks(weighted_network, queryExpressi
onMat, phenos, interactome, pathwayGenesets = NULL)
```

```
subnetworklist[[1]][1,]
```

```
##  TRs             Pathway      Slope       pvalue      fdr   edge_weights
## PCBC1 HALLMARK_COMPLEMENT -0.6827294 0.0008107698 0.02688715       5
```

(D) Mining Step 2:

To identify and rank transcriptional regulatory programs and their groups that affect a pathway in a differentially altered subnetwork of interest, TR2PATH utilizes *computeScores* function.

computeScores{
Inputs:
(1) mechanism-centric network obtained from *reconstNetwork function* in step (B)
(2) subnetworks from *identifySubnetworks* function in step (C)
(3) name of the pathway for the subnetwork of interest in " " (should be copied as one of the pathways from the `subnetworklist`)
(4) TR activity matrix from step (A)
(5) pathway activity matrix from step (A)

Output:
(1) a 3-column data table with identified TR clusters/groups and their ranking based on their effect on the pathway in the subnetwork of interest.
}

Examples of the code and outputs are below:

```
effectscores = TR2PATH::computeScores(weighted_network,subnetworklist,"HALLMA
RK_COMPLEMENT",TR_activity,Pathway_Activity)

effectscores[1,]
```

```
##          clusters    TRs       effectscore
## cluster1 cluster1  PCBD1, AR    2.000000
```

Running time estimates
(to generate new objects based on SU2C East Coast cohort):

Function		**Runtime [mins]**
computePathwayActivity()	10 randomizations	4.01 min
	50 randomizations	5.43 min
	100 randomizations	5.93 min
	1000 randomizations	20.5 min
computeTRActivity()		0.06 min
reconstNetwork()	10 bootstraps	6.13 min
	50 bootstraps	17.2 min
	100 bootstraps	30.11 min
identifySubnetworks()		0.51min
computeScores()		0.0005 min