**Basic usage of MoNET**

MoNET is an R package providing network analysis of -omics findings. It is built on top of an integrated network, including metabolite-protein interactions, protein interactions and relationships between genetic variations and transcription factor binding sites. For details on the integration of the multi-omic network, please refer to section Text S1 in the supplementary material. MoNET has two major functional modules enabling trans-omic interaction query of disease biomarkers and analysis of trans-omic paths. These two modules are enabled with 6 R functions defined for network construction, path search, subnetwork extraction, node annotation, pathway enrichment and subnetwork visualization respectively. Below, we demonstrate the usage of these two modules. All the files included can be downloaded from <https://github.com/Feng-GitHub-Feng/MoNET>. For each function, detailed information for all parameters can be found through ‘help’ function in R.

**Module 1: Trans-omic interaction query of disease markers**

1.1 Build a filtered multi-omic network

Before starting the query, we first filter out the protein interactions with low confidence score using function BuildNet(). It helps build a new multi-omic network by filtering out the protein interactions in the integrated network with the score less than the given threshold. The original integrated network comes with the R package, thus the only parameter needed for this function is the threshold. Protein-protein interaction score from the STRING database (<https://string-db.org/>) was normalized to [0, 1], with 1 indicating most confident interaction. The threshold can be any value between 0 and 1.

> MulOmicNet<-BuildNet(threshold=0)

[1] "The Multi-Omic Network with gene interaction score larger than 0.000 has been constructed."

1.2 Trans-omic path search

1.2.1 Path search with single type of query molecule

With the filtered multi-imic network, we can search the trans-omic interactions of molecule(s) of interest. Depending on the input, we can use the corresponding function to perform the search. More specifically, SNPQuery() is used for SNP input, Genequery()for gene input and Metaquery() for metabolite input. For example, given a list of SNPs, their interaction paths to other -omic layers (i.e., genes and metabolites) will be generated by functions SNPquery().

Function:

SNPquery(MulOmicNet,snplist,node=c("gene","metabolite"),

minpath=1,maxpath=3)

Parameters:

*MulOmicNet* - filtered multi-omic network generated from 1.1

*snplist* – data frame with a list of snps ( rs ID or chromosome:position)

*node* – target node type for the paths to search

*minpath* – minimum length for the paths to search

*maxpath* - maximum length for the paths to search

Output:

A text file with all returned paths named as “Path.txt”. Each line represents a path with a list of SNPs (rs- or chr:pos), genes/proteins (IDs), reactions (IDs) and metabolites (IDs).

Example:

> MulOmicNet<-BuildNet(threshold=0)

[1] "The Multi-Omic Network with gene interaction score larger than 0.000 has been constructed."

>snplist<-read.delim("IGAP\_snp.txt ",header = FALSE)

> head(snplist,10)

V1

1 10:59940129

2 6:32436350

3 6:32512457

4 rs10000546

>PathList<-SNPquery(MulOmicNet=MulOmicNet,snplist=snplist,node="metabolite",minpath=3,maxpath=4)

[1] "Identification of paths from 257 snps to 4901 metabolites are about to start."

[1] 12

[1] 13

[1] 18

[1] 26

.......

[1] 256

[1] 257

[1] "End of searching. Writing paths with length between 3 and 4 into text file."

> Paths<-read.delim(file="Paths.txt",header = FALSE)

> head(Paths,5)

V1

1 rs11039149,10062,3295,R4076,M44

2 rs11039149,10062,1962,R7092,M44

3 rs11039149,10062,3295,R7092,M44

4 rs11039149,10062,1962,R7078,M44

5 rs11039149,10062,51,R7078,M44

1.2.2 Path search with two different types of query molecules

MoNET also supports queries between two types of entities by functions SNP\_Gene\_query(), SNP\_Meta\_query() and Gene\_Meta\_query() with similar parameters.

**Query with SNPs and metabolites**

Function:

SNP\_Meta\_query(MulOmicNet, snplist, metalist, filter="",

minpath=1,maxpath=3)

Parameters:

*MulOmicNet* - filtered multi-omic network generated from 1.1

*snplist* – a data frame with a list of snps ( rs ID or chromosome:position)

*metalist -* a data frame of metabolites with one column

*filter –* type of ID used for metabolite, can have values “” Names“, “Formulas”,

“KEGG\_ID”, “HMDB\_ID”, “CHEBI\_ID”, “PubChem\_ID”.

*minpath* – minimum length for the paths to search

*maxpath* - maximum length for the paths to search

Output:

A text file with all returned paths named as “Path.txt”. Each line represents a path with a list of SNPs (rs- or chr:pos), genes/proteins (IDs), reactions (IDs) and metabolites (IDs).

**Query with SNPs and genes**

Function:

SNP\_Gene\_query(MulOmicNet,snplist,genelist,filter="",

minpath=1,maxpath=3)

Parameters:

*MulOmicNet* - filtered multi-omic network generated from 1.1

*snplist* – a data frame with a list of snps ( rs ID or chromosome:position)

*genelist -* a data frame of metabolites with one column

*filter –* type of ID used for genes, can have values “Gene\_Entrez\_ID”,

“Gene\_Name”, “Ensembl\_Gene\_ID”, “Enzyme\_IDs” and “RefSeq\_IDs”.

*minpath* – minimum length for the paths to search

*maxpath* - maximum length for the paths to search

Output:

A text file with all returned paths named as “Path.txt”. Each line represents a path with a list of SNPs (rs- or chr:pos), genes/proteins (IDs), reactions (IDs) and metabolites (IDs).

**Query with genes and metabolites**

Function:

Gene\_Meta\_query(MulOmicNet,genelist,metalist,genefilter="",

metafilter="",minpath=1,maxpath=3)

Parameters:

*MulOmicNet* - filtered multi-omic network generated from 1.1

*genelist -* a data frame of metabolites with one column

*metalist -* a data frame of metabolites with one column

*metafilter –* type of ID used for metabolite, can have values “Names“, “Formulas”,

“KEGG\_ID”, “HMDB\_ID”, “CHEBI\_ID”, “PubChem\_ID”.

*genefilter –* type of ID used for genes, can have values “Gene\_Entrez\_ID”,

“Gene\_Name”, “Ensembl\_Gene\_ID”, “Enzyme\_IDs” and “RefSeq\_IDs”.

*minpath* – minimum length for the paths to search

*maxpath* - maximum length for the paths to search

Output:

A text file with all returned paths named as “Path.txt”. Each line represents a path with a list of SNPs (rs- or chr:pos), genes/proteins (IDs), reactions (IDs) and metabolites (IDs).

**Module 2: Analysis of trans-omic paths**

With all the paths identified between query molecules and the -omic layer of interest, a subnetwork can be built on top of these paths, which can be used in the subsequent network analysis.

2.1 Subnetwork extraction

Function SubNetRFW() helps generate a subnetwork given a list of specified paths. It generates a text file “SubNet.txt” to save the subnetwork where each line representing one interaction among SNPs, genes/proteins, reactions and metabolites.

Function:

SubNetRFW(MulOmicNet,PathList,Anno=FALSE,enrich=FALSE,

simplot=FALSE)

Parameters:

*MulOmicNet* - filtered multi-omic network generated from 1.1

*PathList -* A list of paths that generated in 1.2.

*Anno -* Logical scalar, whether to annotate the nodes in the generated subnetwork.

*enrich –* Logical scalar, whether to perform the gene enrichment analysis

*simplot - Logical scalar, whether to plot the subnetwork, the default is FALSE.* Output:

“SubNet.txt”: each line represents one interaction in the subnetwork.

“SNPAnno.rds”, “GeneAnno.rds”, “MetaAnno.rds”: annotation files if *Anno=True*

"Enrichment.txt": pathway enrichment results of all the genes in the subnetwork, if *enrich=True*.

“SubNet.pdf”: plot of the subnetwork if *Simplot=True*

Example:

>SubNetRFW(MulOmicNet=MulOmicNet,PathList=PathList,Anno=FALSE,enrich=FALSE,simplot=TRUE)

[1] "Writing subnetwork file."

[1] "Plotting subnetwork."

