Introduction to ceRNAmiRNAfun

ceRNAmiRNAfun Developers

2018-10-19

This document guides the user through all available functions of the ceRNAmiRNAfun package. ceRNAmiRNAfun aims to find out miRNA and ceRNA triplets based on both miRNA and gene expression data. Recent studies have shown that among the target genes of miRNAs, several algorithms have been developed to identify ceRNAs and their dynamic regulating systems. Most of the algorithms divide a miRNA into different groups based on its expression level and them perform the analysis accordingly. However, the expression level of a miRNA is actually a continuous variable instead of a discrete variable. To address this issue, we developed a new algorithm based on the circular binary algorithm. We got the correlation with each window, and the applied the circular binary algorithm to get the peaks from the miRNA expression level across samples.

Installation

ceRNAmiRNAfun is on Bioconductor and can be installed following standard installation procedure.

```
install.packages("BiocManager",repos = "http://cran.us.r-project.org")
BiocManager::install("ceRNAmiRNAfun")
system.file("exdata","mirtest3.csv",package="ceRNAmiRNAfun")
```

To use,

```
library(ceRNAmiRNAfun)
```

General Workflow

workflow steps:

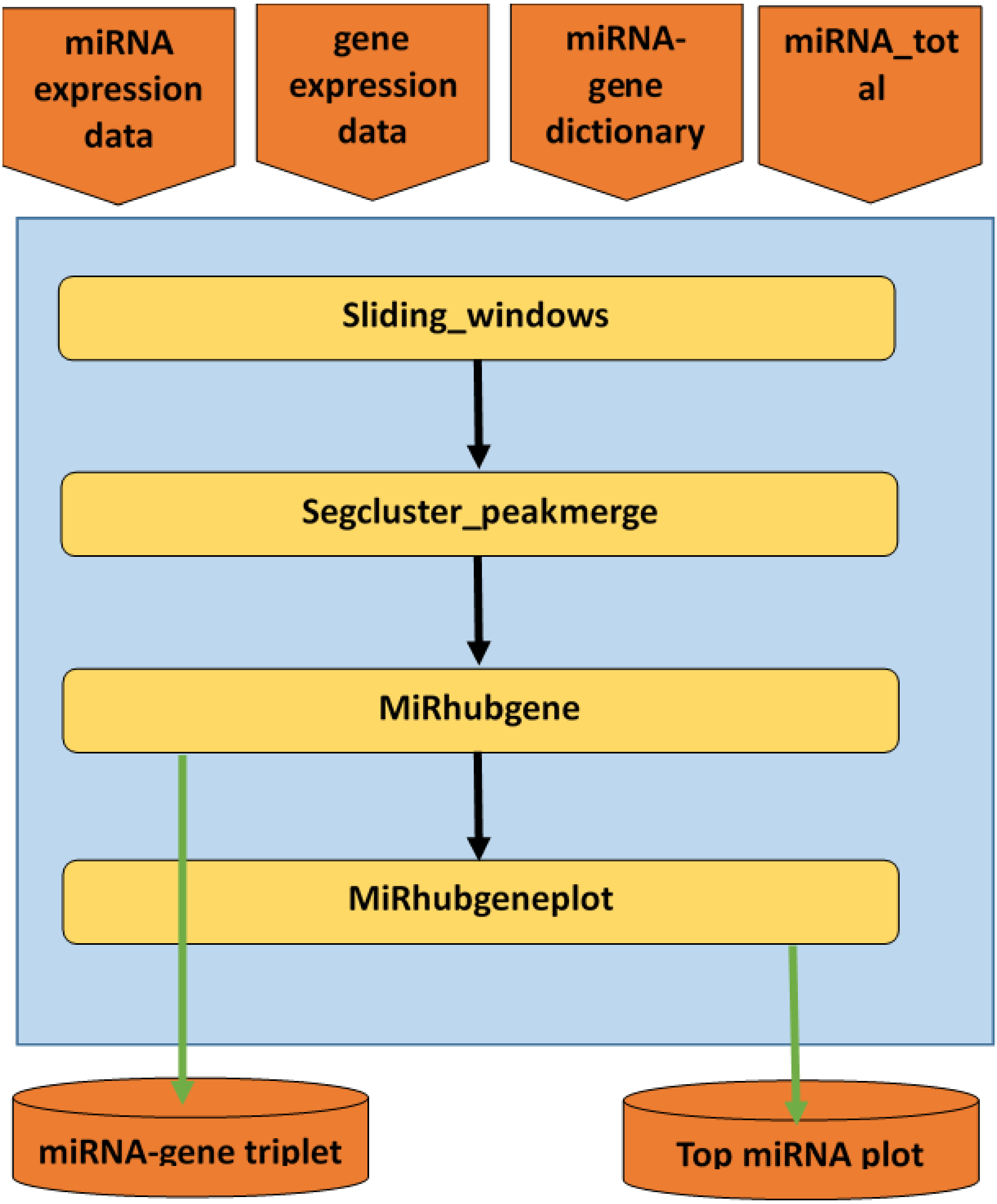
Basically, there are four steps, corresponding to four R functions, to complete the whole analysis:

- 1.Sliding_windows to get the correlation coefficients in each window.

- 2.Segcluster_peakmerge to merge the continuous samples.

- 3.MiRhubgene to sort the data and then give the output table of miRNA and ceRNA triplets.

- 4.MiRhubgeneplot to plot the top miRNA.



workflow_ceRNAmiRNAfun

Data Source

As shown in the workflow, not only samples of miRNA and gene expression data, but also the miRNA and corresponding genes library based on the same data resource for the analysis. The format of ceRNAmiRNAfun input data in expression is matrices, in dictionary is list. Data sources are platform- and technology-independent. Therefore, expression data are all acceptable for ceRNAmiRNAfun. However, the raw miRNA and gene data have to be formatted into expression matrices and the related library should be necessary before using ceRNAmiRNAfun package.

miRNA expression

Columns for samples. Rows for miRNAs

```
miRNA  SampleA  SampleB ...
A         0.1     0.5
B        -0.5     2.1
C         0.4     0.3
```

gene expression

Columns for samples. Rows for genes

```
GENE  SmapleA   SampleB ...
A       0.1       0.2
B      -0.5      -0.3
C       0.4       0.1
```

dictionary data

The first column of miRNA names, the second column of list form of candidate genes.

```
pairinformation    miRNA    genelist
Row1                 A        geneA1,geneA2…
Row2                 B        geneB1,geneB2…
Row3                 C        geneC1,geneC2
```

miRNA_total data

The column of miRNA names, extracted from the first column of dictionary data.

```
         miRNA
Row1      A
Row2      B
Row3      C
```

Usage Example

Now, we show an example using internal data for ceRNAmiRNAfun workflow.

Example Data Source

To demonstrate the usage of the ceRNAmiRNAfun package, the package contains 475 paired miRNA and gene lung cancer samples. As for dictionary data, there are 137 paired miRNA and corresponding gene data in a list form, and miRNA_total data has the first column of the dictionary data of the miRNA’s names. Both of the miRNA and gene data comes from the TCGA lung adenocarcinoma dataset.

Format of Input Data

First of all, load the internal data and check the format.

```
data(mirna_sam)
data(gene_sam)
data(dictionary)
data(miRNA_total)
```

Basically, the format of input data should be the same as the internal data. For miRNA expression data should be like,

```
mirna_sam[1:3, 1:3]
```

As for gene expression data,

```
gene_sam[1:3, 1:3]
```

As for the dictionary data, a list form of miRNA and corresponding genes.

```
dictionary[1:3,]
```

And miRNA_total

```
miRNA_total[1:3]
```

Sliding_windows

Set the size of the window, and then take the input such as dictionary, mirna_sam, gene_sam, miRNA_total, and the window size w into the function

```
Realdata=Datalist_sliding_w(w=10,dictionary=dictionary,miRNA_total=miRNA_total,mirna_sam=mirn
a_sam,gene_sam=gene_sam)
```

```
## Error in makeCluster(no_cores, outfile = ""): 沒有這個函數 "makeCluster"
```

The function will calculate correlation coefficients within each window, a sliding mover that contains putative ceRNA triplets composed of a miRNA and several genes. And the corresponding miRNA expression of each window was the average expression of samples within different windows. The output will be Realdata, a list of data frames with correlation coefficients of genes within each window for each corresponding miRNA average expression.

Segcluster_peakmerge

After getting correlation coefficients with sliding_windows, we try to merge some samples within a cluster to decrease the noise of the all correlations. We have to put cor_shreshold, dictionary, mirna_sam, gene_sam, miRNA_total and Realdata from sliding_windows into the function.

```
Result_TCGA_LUSC =segcluster_peakmerge(cor_shreshold_peak=0.85,dictionary=dictioanry,miRNA_to
tal=miRNA_total,mirna_sam=mirna_sam,gene_sam=gene_sam,Realdata=Realdata)
```

```
## Error in makeCluster(no_cores, outfile = ""): 沒有這個函數 "makeCluster"
```

The function will cluster noisy correlation into neighboring regions of distinct correlation levels. And then do the peak merging by considering low probability of multi-peaks occurring. Also, the segments with few samples detected by circular binary segmentation which are less than three and make no sense to the interactions would merge to a single peak. The output will be Result_TCGA_LUSC, a list format with miRNAs,candidate ceRNA-ceRNA pirs,peak locations, and the number of samples occuring ceRNA-ceRNA interactions.

MiRhubgene

After the peak merge, we will sort the information of miRNA and ceRNA to find out triplets. We should put dictionary and Result_TCGA_LUSC from segcluster_peakmerge into the function.

```
miRhubgeneoutput=miRhubgene(dictionary=dictionary,Result_TCGA_LUSC= Result_TCGA_LUSC)
```

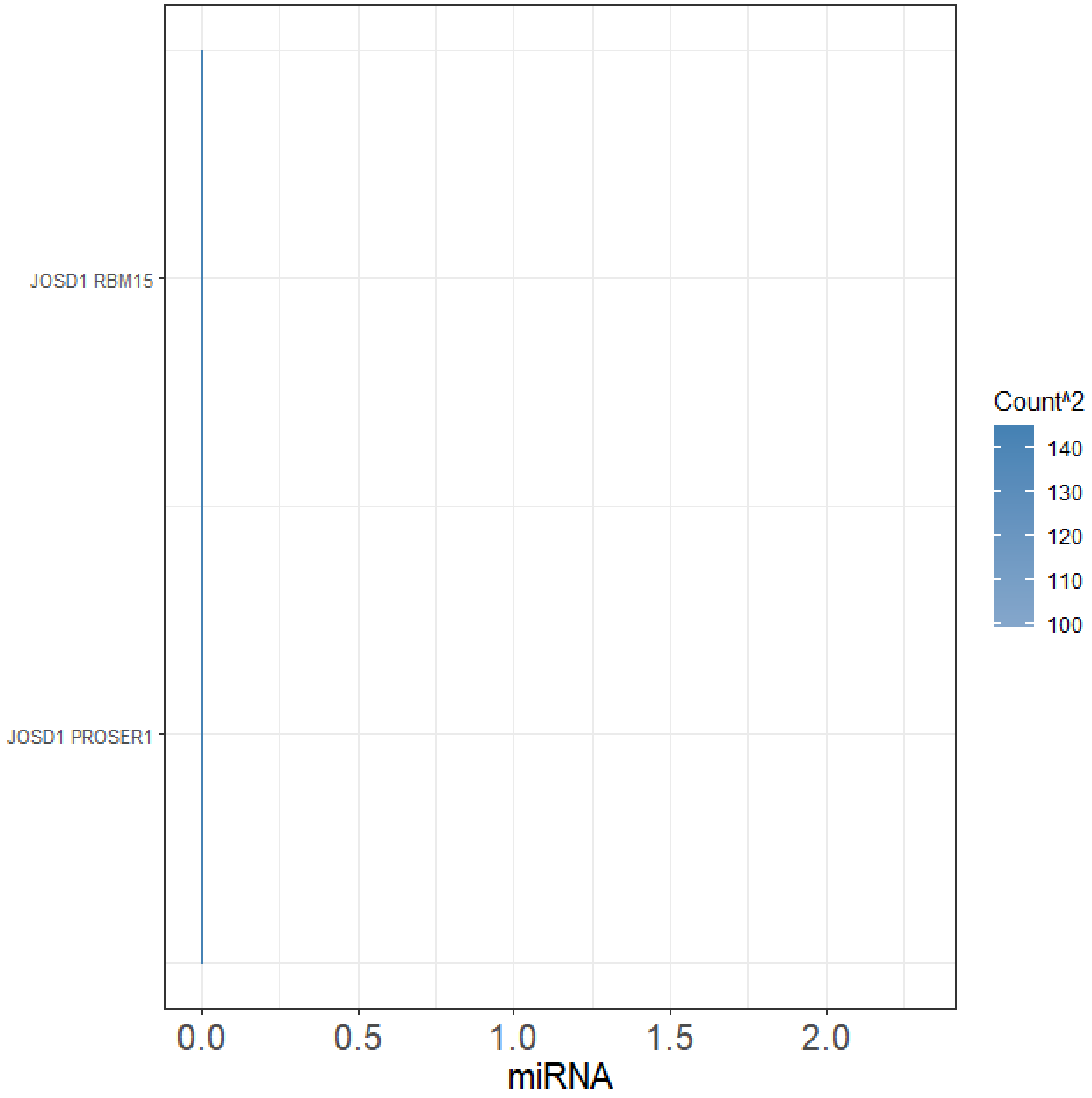
The output will be miRhubgeneoutput, a dataframe formats with miRNA names with the number of bridging ceRNA triplets and the corresponding genes with the number of ceRNA triplets.

MiRhubgeneplot

Last,.we do the plot of the top miRNA to see its expression situation. The input should be dictionary, mirna_sam, gene_sam, Result_TCGA_LUSC from segcluster_peakmerge, miRhubgeneoutput from miRhubgene, and also the window size w and the number of samples N into the function.

```
plotp=miRhubgeneplot(dictionary=dictionary,Result_TCGA_LUSC=Result_TCGA_LUSC,mirna_sam=mirna_
sam,miRhubgeneoutput=miRhubgeneoutput,w=10,N=475)
```

The function will plot the top miRNA, for visualizing the result of the triplets. The output will be plotp a plot format with miRNA expression in x-axis and the interaction ceRNA in y-axis.


JOSD1 RBM15
JOSD1 PROSER1
0.0
0.5
1.0
1.5
2.0
miRNA
Count^2
140
130
120
110
100

plotp