

# Supplementary Materials And Methods

*Selcen Ari*

*7 Nisan 2019*

## 1. Functions defined for ceRNA models and workflow of method

We defined the functions that can be used with R programming. Briefly, these functions process a given miRNA:gene dataset and convert to graph object. All values that are significant in miRNA:target interactions are stored in edge variables and processed with formulations that are given in previous section. The functions and steps of approach are explained as following (Figure S1 in Supplementary Figures) :

**Conversion of dataset:** `priming_graph()` function processes the given dataset that includes competing elements in first variable and repressive element in second variable. If the affinity and/or degradation factors are specified in the function, factors are taken into account, are processed with defaults in vice versa. The formulations that are given in equations (1-4) are performed in this function. This step gives the graph object which contains efficiency values of miRNA:competing target pairs in steady-state in terms of amount. It is assumed that the initial target amounts in the dataset is observed after the repressive activity of miRNAs in steady-state.

**Transition of variables in graph:** In the previous step, the calculations are performed in the edge variables of the graph object. However, the graph object allows to use node variables, while the node features are handled to the graph. In this direction, `update_nodes` function carries the amount values to node variables. This step must be applied with “once” option because it is primary process.

**Trigger change in graph:** The dataset are assumed as steady-state in previous step and the efficiency coefficients are calculated according to this acceptance. In the network that is found in steady-state conditions, the change is applied to the graph object for distribution of steady-state. To provide the distribution in the network the workflow offer two methods: `update_variables` and `update_how`. The first, a new dataset that is contained competing and repressive element names and current values of these can be processed with `update_variables`. The second option, the amount of the given node name in `update_how` function can be changed according to “how” argument.

**Updating current values of variables:** After variables updating in edge variables, these are carried to node variables. Current and previous values of variables are stored as node variables with `update_variables` function.

**Simulation of competing behavior of targets:** After the change in the steady-state conditions, the network elements try to gain steady-state again. This process progresses as repeating of regulations after the spreading the changes in the network. In this step, simulation of regulations according to given cycle count in `simulate` function is applied. After each simulation cycle, the miRNA repression values are re-calculated and the current values of competing elements are found and saved. The process is performed in the edge data and at the same time outputs of the calculations are carried from edge to node data.

The node elements in the dataset are handled as two type; repressive (miRNAs) and competing (targets). It is assumed in approach that while targets are degrading or inhibiting by miRNAs continuously, miRNAs reversibly used. If the trigger of the network is a miRNA, it maintains the current value of amount that provides by user. On the contrary, it tries to help this process to provide steady-state through the regulations on its amount, if a competing element is used as a trigger. The functions that are used in the approach are developed with R programming so as can be used with other packages. These are can be found in the Github repository `ceRNAetsim` Github page and improved with contributions from others.

```
#install.packages("devtools")
#devtools::install_github("selcenari/ceRNAetsim")
library(ceRNAetsim)
```

- load *minsamp* data

```
data("minsamp")
```

```
minsamp
```

```
##   competing miRNA Competing_expression miRNA_expression seed_type region
## 1   Gene1  Mir1           10000           1000       0.43   0.30
## 2   Gene2  Mir1           10000           1000       0.43   0.01
## 3   Gene3  Mir1            5000           1000       0.32   0.40
## 4   Gene4  Mir1           10000           1000       0.23   0.50
## 5   Gene4  Mir2           10000           2000       0.35   0.90
## 6   Gene5  Mir2            5000           2000       0.05   0.40
## 7   Gene6  Mir2           10000           2000       0.01   0.80
##   energy
## 1    -20
## 2    -15
## 3    -14
## 4    -10
## 5    -12
## 6    -11
## 7    -25
```

See Table S1 in Supplementary Tables file.

### ***minsamp* dataset analysis in lack of interaction factors.**

Firstly, we have analysed minimal data without interaction factors between miRNA:target.

1. We have evaluated graph in the steady state conditions as followings (Figure S2 in Supplementary Figures):

```
priming_graph(minsamp, competing_count = Competing_expression, miRNA_count = miRNA_expression)
```

```
## # A tbl_graph: 8 nodes and 7 edges
## #
## # A rooted tree
## #
## # Node Data: 8 x 7 (active)
##   name  type  node_id initial_count count_pre count_current
##   <chr> <chr>   <int>      <dbl>    <dbl>    <dbl>
## 1 Gene1 Comp~     1        10000    10000    10000
## 2 Gene2 Comp~     2        10000    10000    10000
## 3 Gene3 Comp~     3         5000     5000     5000
## 4 Gene4 Comp~     4        10000    10000    10000
## 5 Gene5 Comp~     5         5000     5000     5000
## 6 Gene6 Comp~     6        10000    10000    10000
## # ... with 2 more rows, and 1 more variable: changes_variable <chr>
## #
## # Edge Data: 7 x 19
##   from    to Competing_name miRNA_name Competing_expre~ miRNA_expression
```

```
## <int> <int> <chr> <chr> <dbl> <dbl>
## 1 1 7 Gene1 Mir1 10000 1000
## 2 2 7 Gene2 Mir1 10000 1000
## 3 3 7 Gene3 Mir1 5000 1000
## # ... with 4 more rows, and 13 more variables: dummy <dbl>,
## # afff_factor <dbl>, degg_factor <dbl>, comp_count_list <list>,
## # comp_count_pre <dbl>, comp_count_current <dbl>,
## # mirna_count_list <list>, mirna_count_pre <dbl>,
## # mirna_count_current <dbl>, mirna_count_per_dep <dbl>,
## # effect_current <dbl>, effect_pre <dbl>, effect_list <list>
```

- 2. We have obtained graph after change on Gene2 expression as following (Figure S3 in Supplementary Figures):

```
priming_graph(minsamp, competing_count = Competing_expression, miRNA_count = miRNA_expression)%>%
  update_how("Gene2", 2)
```

```
## # A tbl_graph: 8 nodes and 7 edges
## #
## # A rooted tree
## #
## # Node Data: 8 x 7 (active)
##   name type node_id initial_count count_pre count_current
##   <chr> <chr> <int> <dbl> <dbl> <dbl>
## 1 Gene1 Comp~ 1 10000 10000 10000
## 2 Gene2 Comp~ 2 10000 10000 20000
## 3 Gene3 Comp~ 3 5000 5000 5000
## 4 Gene4 Comp~ 4 10000 10000 10000
## 5 Gene5 Comp~ 5 5000 5000 5000
## 6 Gene6 Comp~ 6 10000 10000 10000
## # ... with 2 more rows, and 1 more variable: changes_variable <chr>
## #
## # Edge Data: 7 x 19
##   from to Competing_name miRNA_name Competing_expre~ miRNA_expression
##   <int> <int> <chr> <chr> <dbl> <dbl>
## 1 1 7 Gene1 Mir1 10000 1000
## 2 2 7 Gene2 Mir1 10000 1000
## 3 3 7 Gene3 Mir1 5000 1000
## # ... with 4 more rows, and 13 more variables: dummy <dbl>,
## # afff_factor <dbl>, degg_factor <dbl>, comp_count_list <list>,
## # comp_count_pre <dbl>, comp_count_current <dbl>,
## # mirna_count_list <list>, mirna_count_pre <dbl>,
## # mirna_count_current <dbl>, mirna_count_per_dep <dbl>,
## # effect_current <dbl>, effect_pre <dbl>, effect_list <list>
```

- 3. We have determined regulations after Gene2 upregulation (Figure S4 in Supplementary Figures):

```
priming_graph(minsamp, competing_count = Competing_expression, miRNA_count = miRNA_expression)%>%
  update_how("Gene2", 2)%>%
  simulate(cycle= 2)
```

```
## # A tbl_graph: 8 nodes and 7 edges
```

```
## #
## # A rooted tree
## #
## # Node Data: 8 x 7 (active)
##   name type node_id initial_count count_pre count_current
##   <chr> <chr>   <int>      <dbl>    <dbl>    <dbl>
## 1 Gene1 Comp~    1        10000    10063.    10062.
## 2 Gene2 Comp~    2        10000    19841.    19845.
## 3 Gene3 Comp~    3         5000     5032.     5031.
## 4 Gene4 Comp~    4        10000    10063.    10059.
## 5 Gene5 Comp~    5         5000     5000      5001.
## 6 Gene6 Comp~    6        10000    10000     10002.
## # ... with 2 more rows, and 1 more variable: changes_variable <chr>
## #
## # Edge Data: 7 x 20
##   from to Competing_name miRNA_name Competing_expre~ miRNA_expression
##   <int> <int> <chr>          <chr>          <dbl>          <dbl>
## 1     1     7 Gene1      Mir1            10000          1000
## 2     2     7 Gene2      Mir1            10000          1000
## 3     3     7 Gene3      Mir1             5000          1000
## # ... with 4 more rows, and 14 more variables: dummy <dbl>,
## #   afff_factor <dbl>, degg_factor <dbl>, comp_count_list <list>,
## #   comp_count_pre <dbl>, comp_count_current <dbl>,
## #   mirna_count_list <list>, mirna_count_pre <dbl>,
## #   mirna_count_current <dbl>, mirna_count_per_dep <dbl>,
## #   effect_current <dbl>, effect_pre <dbl>, effect_list <list>,
## #   mirna_count_per_comp <dbl>
```

Note that the regulations are colored according to expression changes of present and a previous value. So, it can be observed that whole gene expressions increase in comparison of initial steady-state. The overall regulations of gene expressions are as followings:

```
priming_graph(minsamp, competing_count = Competing_expression, miRNA_count = miRNA_expression)%>%
  update_how("Gene2", 2)%>%
  simulate(2)%>%
  activate(edges)%>%
  as_tibble()%>%
  select(Competing_name, comp_count_list, effect_list)%>%
  unnest()
```

```
## # A tibble: 21 x 3
##   Competing_name comp_count_list effect_list
##   <chr>          <dbl>          <dbl>
## 1 Gene1          10000          286.
## 2 Gene1          10063.          222.
## 3 Gene1          10062.          224.
## 4 Gene2          10000          286.
## 5 Gene2          19841.          444.
## 6 Gene2          19845.          441.
## 7 Gene3           5000          143.
## 8 Gene3           5032.          111.
## 9 Gene3           5031.          112.
## 10 Gene4         10000          286.
## # ... with 11 more rows
```

## *minsamp* dataset analysis with interaction factors.

We have made the same analysis in presence of interaction factors (Sequentially shown at Figure S5-7 in Supplementary Figures).

```
priming_graph(minsamp, competing_count = Competing_expression, miRNA_count = miRNA_expression, aff_fact
```

```
## # A tbl_graph: 8 nodes and 7 edges
## #
## # A rooted tree
## #
## # Node Data: 8 x 7 (active)
##   name type node_id initial_count count_pre count_current
##   <chr> <chr>   <int>         <dbl>    <dbl>      <dbl>
## 1 Gene1 Comp~     1         10000    10000     10000
## 2 Gene2 Comp~     2         10000    10000     10000
## 3 Gene3 Comp~     3          5000     5000      5000
## 4 Gene4 Comp~     4         10000    10000     10000
## 5 Gene5 Comp~     5          5000     5000      5000
## 6 Gene6 Comp~     6         10000    10000     10000
## # ... with 2 more rows, and 1 more variable: changes_variable <chr>
## #
## # Edge Data: 7 x 22
##   from to Competing_name miRNA_name Competing_expre~ miRNA_expression
##   <int> <int> <chr>         <chr>         <dbl>          <dbl>
## 1     1     7 Gene1      Mir1           10000          1000
## 2     2     7 Gene2      Mir1           10000          1000
## 3     3     7 Gene3      Mir1            5000          1000
## # ... with 4 more rows, and 16 more variables: energy <dbl>,
## #   seed_type <dbl>, region <dbl>, dummy <dbl>, afff_factor <dbl>,
## #   degg_factor <dbl>, comp_count_list <list>, comp_count_pre <dbl>,
## #   comp_count_current <dbl>, mirna_count_list <list>,
## #   mirna_count_pre <dbl>, mirna_count_current <dbl>,
## #   mirna_count_per_dep <dbl>, effect_current <dbl>, effect_pre <dbl>,
## #   effect_list <list>
```

```
priming_graph(minsamp, competing_count = Competing_expression, miRNA_count = miRNA_expression, aff_fact
  update_how("Gene2", 2)
```

```
## # A tbl_graph: 8 nodes and 7 edges
## #
## # A rooted tree
## #
## # Node Data: 8 x 7 (active)
##   name type node_id initial_count count_pre count_current
##   <chr> <chr>   <int>         <dbl>    <dbl>      <dbl>
## 1 Gene1 Comp~     1         10000    10000     10000
## 2 Gene2 Comp~     2         10000    10000     20000
## 3 Gene3 Comp~     3          5000     5000      5000
## 4 Gene4 Comp~     4         10000    10000     10000
## 5 Gene5 Comp~     5          5000     5000      5000
## 6 Gene6 Comp~     6         10000    10000     10000
## # ... with 2 more rows, and 1 more variable: changes_variable <chr>
```

```
## #
## # Edge Data: 7 x 22
##   from    to Competing_name miRNA_name Competing_expre~ miRNA_expression
##   <int> <int> <chr>          <chr>          <dbl>          <dbl>
## 1     1     7 Gene1          Mir1          10000          1000
## 2     2     7 Gene2          Mir1          10000          1000
## 3     3     7 Gene3          Mir1           5000          1000
## # ... with 4 more rows, and 16 more variables: energy <dbl>,
## #   seed_type <dbl>, region <dbl>, dummy <dbl>, afff_factor <dbl>,
## #   degg_factor <dbl>, comp_count_list <list>, comp_count_pre <dbl>,
## #   comp_count_current <dbl>, mirna_count_list <list>,
## #   mirna_count_pre <dbl>, mirna_count_current <dbl>,
## #   mirna_count_per_dep <dbl>, effect_current <dbl>, effect_pre <dbl>,
## #   effect_list <list>

priming_graph(minsamp, competing_count = Competing_expression, miRNA_count = miRNA_expression, aff_fact
  update_how("Gene2", 2)%>%
  simulate(cycle=2)
```

```
## # A tbl_graph: 8 nodes and 7 edges
## #
## # A rooted tree
## #
## # Node Data: 8 x 7 (active)
##   name type node_id initial_count count_pre count_current
##   <chr> <chr>   <int>      <dbl>    <dbl>    <dbl>
## 1 Gene1 Comp~    1      10000    10065.    10064.
## 2 Gene2 Comp~    2      10000    19997.    19997.
## 3 Gene3 Comp~    3       5000     5023.     5023.
## 4 Gene4 Comp~    4      10000    10029.    10028.
## 5 Gene5 Comp~    5       5000     5000.     5000.
## 6 Gene6 Comp~    6      10000    10000.    10000.
## # ... with 2 more rows, and 1 more variable: changes_variable <chr>
## #
## # Edge Data: 7 x 23
##   from    to Competing_name miRNA_name Competing_expre~ miRNA_expression
##   <int> <int> <chr>          <chr>          <dbl>          <dbl>
## 1     1     7 Gene1          Mir1          10000          1000
## 2     2     7 Gene2          Mir1          10000          1000
## 3     3     7 Gene3          Mir1           5000          1000
## # ... with 4 more rows, and 17 more variables: energy <dbl>,
## #   seed_type <dbl>, region <dbl>, dummy <dbl>, afff_factor <dbl>,
## #   degg_factor <dbl>, comp_count_list <list>, comp_count_pre <dbl>,
## #   comp_count_current <dbl>, mirna_count_list <list>,
## #   mirna_count_pre <dbl>, mirna_count_current <dbl>,
## #   mirna_count_per_dep <dbl>, effect_current <dbl>, effect_pre <dbl>,
## #   effect_list <list>, mirna_count_per_comp <dbl>
```

When the graphs which were resulted from analyses were examined, it was observed that behaviors were same. But, when the results were analysed in terms of expression values, the regulation differences can be observed.

```
priming_graph(minsamp, competing_count = Competing_expression, miRNA_count = miRNA_expression, aff_facto
  update_how("Gene2", 2)%>%
  simulate(3)%>%
  activate(edges)%>%
  as_tibble()%>%
  select(Competing_name, comp_count_list, effect_list)%>%
  unnest()
```

```
## # A tibble: 28 x 3
##   Competing_name comp_count_list effect_list
##   <chr>          <dbl>          <dbl>
## 1 Gene1          10000          263.
## 2 Gene1          10065.          198.
## 3 Gene1          10064.          199.
## 4 Gene1          10064.          199.
## 5 Gene2          10000           6.58
## 6 Gene2          19997.           9.91
## 7 Gene2          19997.           9.88
## 8 Gene2          19997.           9.88
## 9 Gene3           5000          91.5
## 10 Gene3         5023.          68.8
## # ... with 18 more rows
```

## Common target perturbation in *minsamp* dataset.

Genes targeted by multiple miRNAs (referred to as “common target”) are of special interest since they are subject to cooperative effect. Also, they perturb more than one neighborhood. In our small dataset, minsamp, Gene4 is regulated by two miRNAs. Let’s simulate perturbation effects triggered by Gene4 (Shown at Figure S8 in Supplementary Figures .

```
priming_graph(minsamp, competing_count = Competing_expression, miRNA_count = miRNA_expression, aff_facto
  update_how("Gene4", 2)%>%
  simulate(cycle = 2)
```

```
## # A tbl_graph: 8 nodes and 7 edges
## #
## # A rooted tree
## #
## # Node Data: 8 x 7 (active)
##   name type node_id initial_count count_pre count_current
##   <chr> <chr> <int>          <dbl>    <dbl>    <dbl>
## 1 Gene1 Comp~ 1          10000    10028.    10027.
## 2 Gene2 Comp~ 2          10000    10001.    10001.
## 3 Gene3 Comp~ 3           5000     5010.     5009.
## 4 Gene4 Comp~ 4          10000    19803.    19806.
## 5 Gene5 Comp~ 5           5000     5024.     5024.
## 6 Gene6 Comp~ 6          10000    10044.    10044.
## # ... with 2 more rows, and 1 more variable: changes_variable <chr>
## #
## # Edge Data: 7 x 23
##   from      to Competing_name miRNA_name Competing_expre~ miRNA_expression
##   <int> <int> <chr>          <chr>          <dbl>          <dbl>
```

```
## 1      1      7 Gene1      Mir1      10000      1000
## 2      2      7 Gene2      Mir1      10000      1000
## 3      3      7 Gene3      Mir1      5000      1000
## # ... with 4 more rows, and 17 more variables: energy <dbl>,
## #   seed_type <dbl>, region <dbl>, dummy <dbl>, afff_factor <dbl>,
## #   degg_factor <dbl>, comp_count_list <list>, comp_count_pre <dbl>,
## #   comp_count_current <dbl>, mirna_count_list <list>,
## #   mirna_count_pre <dbl>, mirna_count_current <dbl>,
## #   mirna_count_per_dep <dbl>, effect_current <dbl>, effect_pre <dbl>,
## #   effect_list <list>, mirna_count_per_comp <dbl>
```

The common target perturbation (increasing to two fold at Gene4 expression in presence of interaction factors) resulted in more prominent efficiency at the same conditions (shown in following).

```
priming_graph(minsamp, competing_count = Competing_expression, miRNA_count = miRNA_expression, aff_factor = aff_factor,
  update_how("Gene4", 2))%>%
  simulate(3)%>%
  activate(edges)%>%
  as_tibble()%>%
  select(Competing_name, comp_count_list, effect_list)%>%
  unnest()
```

```
## # A tibble: 28 x 3
##   Competing_name comp_count_list effect_list
##   <chr>          <dbl>          <dbl>
## 1 Gene1          10000          263.
## 2 Gene1          10028.          236.
## 3 Gene1          10027.          237.
## 4 Gene1          10027.          237.
## 5 Gene2          10000           6.58
## 6 Gene2          10001.           5.89
## 7 Gene2          10001.           5.90
## 8 Gene2          10001.           5.90
## 9 Gene3           5000          91.5
## 10 Gene3         5010.          81.9
## # ... with 18 more rows
```

## Determination of perturbation efficiencies of elements in system.

```
priming_graph(minsamp, competing_count = Competing_expression, miRNA_count = miRNA_expression, aff_factor = aff_factor,
  find_node_perturbation(sample_graph, how = 2, cycle = 3, limit = 0.1))
```

```
## # A tibble: 8 x 6
##   name type node_id initial_count count_pre count_current
##   <chr> <chr>   <int>         <dbl>    <dbl>    <dbl>
## 1 Gene1 Comp~     1      10000    10000    10000
## 2 Gene2 Comp~     2      10000    10000    10000
## 3 Gene3 Comp~     3       5000     5000     5000
## 4 Gene4 Comp~     4      10000    10000    10000
## 5 Gene5 Comp~     5       5000     5000     5000
```



```
## 6 Gene6 Comp~      6      10000      10000      10000
## 7 Mir1  miRNA      7       1000       1000       1000
## 8 Mir2  miRNA      8       2000       2000       2000
## # ... with 3 more variables: changes_variable <chr>,
## #   perturbation_efficiency <dbl>, perturbed_count <dbl>
```

## 2. Obtaining breast cancer dataset and integration

This section describes how to apply `ceRnAnetsim` package on a breast cancer patient miRNA:target interaction dataset. Before the approach, we obtained three datasets and combined them.

### 2.1 How to get gene expression counts of TCGA-E9-A1N5 patient.

We have obtained the gene expression values of patient using `TCGAbiolinks` package from Bioconductor. For this process, we have followed the instructions of the package. `TCGAbiolinks` package provides to obtain data for whole number of given barcode(s) at once. But, we preferred to download them separately to show datasets.

- Obtain to gene expression counts of tumor tissue.

```
BCP_tumor <- GDCquery(project = "TCGA-BRCA",
                      data.category = "Transcriptome Profiling",
                      data.type = "Gene Expression Quantification",
                      workflow.type = "HTSeq - Counts",
                      barcode = "TCGA-E9-A1N5-01A-11R-A14D-07")

GDCdownload(BCP_tumor)
BCPGE_tumor <- GDCprepare(BCP_tumor)

as.data.frame(assay(BCPGE_tumor))%>%
  mutate(ensembl_gene_id = rownames(.))%>%
  dplyr::inner_join(as.data.frame(rowData(BCPGE_tumor)), by = "ensembl_gene_id")%>%
  dplyr::select(ensembl_gene_id, external_gene_name, 1)-> TCGA_E9_A1N5_tumor

colnames(TCGA_E9_A1N5_tumor)[3] <- "GE_tumor"

head(TCGA_E9_A1N5_tumor)
```

- Obtain to gene expression counts of normal tissue.

```
BCP_normal <- GDCquery(project = "TCGA-BRCA",
                      data.category = "Transcriptome Profiling",
                      data.type = "Gene Expression Quantification",
                      workflow.type = "HTSeq - Counts",
                      barcode = "TCGA-E9-A1N5-11A-41R-A14D-07")

GDCdownload(BCP_normal)

BCPGE_normal <- GDCprepare(BCP_normal)

as.data.frame(assay(BCPGE_normal))%>%
```

```
mutate(ensembl_gene_id = rownames(.))%>%
dplyr::inner_join(as.data.frame(rowData(BCPGE_normal)), by = "ensembl_gene_id")%>%
dplyr::select(ensembl_gene_id, external_gene_name, 1)-> TCGA_E9_A1N5_normal

colnames(TCGA_E9_A1N5_normal)[3] <- "GE_normal"

head(TCGA_E9_A1N5_normal)
```

## 2.2 How to get miRNA expression counts of TCGA-E9-A1N5 patient.

We have used TCGAbiolinks package to obtain miRNA expression quantification. The query gives read count of miRNA as isoform chromosome coordination. The data also contains mature miRNA information. So, we processed data to attain -5p -3p isoform information using mirbase release21 dataset.

- Get the mirbase id of mature miRNA:

We downloaded the mirbase release 21 dataset from mirbase and processed the patient mirna expression datasets as following:

```
library(readr)
read_tsv("hsa_mirna.txt", comment = "#", col_names = FALSE)%>%
  dplyr::select(mirna_type= X3, definition = X9)%>%
  filter(!endsWith(mirna_type, "primary_transcript"))%>%
  tidyr::separate(definition, c("ID", "Alias", "Name", "Derivated"), sep = ";")%>%
  dplyr::select(Alias, Name)%>%
  tidyr::separate(Alias, c("trash1", "ID"), sep = "=")%>%
  tidyr::separate(Name, c("trash2", "Name"), sep = "=")%>%
  dplyr::select(-trash1, -trash2)-> mirbase_id_conv

head(mirbase_id_conv)
```

```
## # A tibble: 6 x 2
##   ID          Name
##   <chr>      <chr>
## 1 MIMAT0027618 hsa-miR-6859-5p
## 2 MIMAT0027619 hsa-miR-6859-3p
## 3 MIMAT0005890 hsa-miR-1302
## 4 MIMAT0027618 hsa-miR-6859-5p
## 5 MIMAT0027619 hsa-miR-6859-3p
## 6 MIMAT0049032 hsa-miR-12136
```

- Obtain the miRNA expression of tumor tissue of patient:

```
BCP_mirnatumor <- GDCQuery(project = "TCGA-BRCA",
  data.category = "Transcriptome Profiling",
  data.type = "Isoform Expression Quantification",
  workflow.type = "BCGSC miRNA Profiling",
  barcode = "TCGA-E9-A1N5-01A-11R-A14C-13")

GDCdownload(BCP_mirnatumor)
```

```
GDCprepare(BCP_mirnatumor)%>%
  as.data.frame()%>%
  dplyr::select(miRNA_ID, read_count, reads_per_million_miRNA_mapped, miRNA_region)%>%
  dplyr::filter(startsWith(miRNA_region, "mature"))%>%
  dplyr::mutate(mirbase_id =str_remove(miRNA_region, "mature,"))%>%
  dplyr::select(-miRNA_region)%>%
  dplyr::inner_join(mirbase_id_conv, by = c("mirbase_id"="ID"))%>%
  dplyr::select(miRNA_name = Name, read_count, reads_per_million_miRNA_mapped)%>%
  dplyr::group_by(miRNA_name)%>%
  mutate(read_count= sum(read_count), reads_per_million_miRNA_mapped = sum(reads_per_million_miRNA_mapped))%>%
  dplyr::ungroup()%>%
  distinct() -> BCPME_mirnatumor

head(BCPME_mirnatumor)
```

- Obtain the miRNA expression of normal tissue of patient:

```
BCP_mirnanormal <- GDCquery(project = "TCGA-BRCA",
  data.category = "Transcriptome Profiling",
  data.type = "Isoform Expression Quantification",
  workflow.type = "BCGSC miRNA Profiling",
  barcode = "TCGA-E9-A1N5-11A-41R-A14C-13")

GDCdownload(BCP_mirnanormal)
# a616435d-0b69-48ac-813d-5d75ad9b85eb.mirbase21.isoforms.quantification.txt

GDCprepare(BCP_mirnanormal)%>%
  as.data.frame()%>%
  dplyr::select(miRNA_ID, read_count, reads_per_million_miRNA_mapped, miRNA_region)%>%
  dplyr::filter(startsWith(miRNA_region, "mature"))%>%
  dplyr::mutate(mirbase_id =str_remove(miRNA_region, "mature,"))%>%
  dplyr::select(-miRNA_region)%>%
  dplyr::inner_join(mirbase_id_conv, by = c("mirbase_id"="ID"))%>%
  dplyr::select(miRNA_name = Name, read_count, reads_per_million_miRNA_mapped)%>%
  dplyr::group_by(miRNA_name)%>%
  mutate(read_count= sum(read_count), reads_per_million_miRNA_mapped = sum(reads_per_million_miRNA_mapped))%>%
  dplyr::ungroup()%>%
  distinct() -> BCPME_mirnanormal

head(BCPME_mirnanormal)
```

## 2.3 Get the high-throughput experimental miRNA:target dataset.

There are various datasets about miRNA:target pairs such as miRTarBase, DianaTools, miRecords, miRWalk etc. Some of these present the experimentally supported miRNA target pairs or only predicted ones. The experimentally supported datasets generally provides weak evidence for interactions. For these reasons, we obtained the high-throughput experimental miRNA:target dataset from two studies performed by Helwak et al. and Moore et al. These steps were not handle in this file because they contain many processes.

Briefly these datasets contain various common information about miRNA:target interactions such as the miRNA name, miRNAsequence, target name, target sequence, their chromosomal locations, binding location on the target sequence, binding free energy, seed structure. But these datasets provides the informations with different data structures. So we followed the steps:

- The datasets were directly downloaded from supplementary data files of the studies.
- It was provided that the datasets are converted to same human genome build.
- The seed type information was organized as the same style.
- The datasets were combined.
- We committed the interaction factors as numeric values according to previous studies. (We added the interaction factors and their numeric values at Supplementary files, Process on miRNA:target pairs dataset for process codes and Supplementary Tables for used numeric values)

Finally, we have obtained the experimentally supported miRNA:target dataset.

```
experimentalmirnagene<-readRDS("data/experimentalmirnagene.rda")
```

```
head(experimentalmirnagene)
```

```
##   Hugo_Symbol Ensembl_Gene_Id      mRNA_seq      miRNA_name
## 1      AGL ENSG00000162688  |||||----W|||W|||  hsa-miR-16-5p
## 2      AGL ENSG00000162688 -|||||---||-|||---||WW|  hsa-miR-92a-3p
## 3      AGL ENSG00000162688  -||||---|||  hsa-miR-30a-5p
## 4    SLC35A3 ENSG00000117620  |||||----|||W-  hsa-miR-500a-5p
## 5    SLC35A3 ENSG00000117620 -|||||---W|||---|||  hsa-miR-181a-5p
## 6     HIAT1 ENSG00000156875 -|||||---|---|---|||  hsa-miR-26b-5p
##           miRNA_seq      seed_type energy      region seed_type_effect
## 1  |||||---W|||W|---|  7merm8  -19.5      intron           0.350
## 2  -|||||---|||||||WW| 7merA1.indel -27.2      CDS             0.096
## 3  -----|---|---|  6merA1.mm   -9.4      3UTR             0.028
## 4  |||||---|---|W----  7merA1  -22.0      3UTR             0.240
## 5  -|||||---W|||---|--- 6mer_off -23.6 3UTRintron 0.010
## 6  -|||||---|---|--- 6mer_off -18.5      intron           0.010
##      region_effect
## 1           0.0
## 2           0.2
## 3           0.8
## 4           0.8
## 5           0.8
## 6           0.0
```

The methods about miRNA:target interactions are based a basic principle that is reading after isolation of miRNA:target chimerics. The datasets contain all the chimeric miRNA:target structures found in the medium during the experiment. On the other hand, it could be said that the reading is performed as snapshot. Because of that, the methods can provide different chimeric interactions the same miRNA:target pair. We have preferred to select most effective interaction parameters for the same miRNA:target pairs that can exhibit various interactions. The step is performed as:

```
experimentalmirnagene%>%
```

```
  dplyr::select(miRNA_name, Ensembl_Gene_Id, Hugo_Symbol, energy, seed_type_effect, region_effect)%>%
  distinct()%>%
  group_by(Ensembl_Gene_Id, miRNA_name)%>%
  mutate(seed_type_effect= ifelse(seed_type_effect==max(seed_type_effect), seed_type_effect, max(seed_t
  distinct()-> tocombine_mirnagene
```

```
head(tocombine_mirnagene)
```

## 2.4 Combine the dataset

```
BCPME_mirnanormal%>%
  dplyr::inner_join(tocombine_mirnagene, by = c("miRNA_name"))%>%
  dplyr::inner_join(TCGA_E9_A1N5_normal, by = c("Ensembl_Gene_Id"="ensembl_gene_id", "Hugo_Symbol"="external_gene_name"))%>%
  distinct()%>%
  dplyr::select(Hugo_Symbol, miRNA_name, mirna_RPM= reads_per_million_miRNA_mapped, GE_normal, energy, seed_type_effect)

#saveRDS(E9GE_mirnagenenormal, "data/E9GE_mirnagenenormal.rda")

E9GE_mirnagenenormal<-readRDS("data/E9GE_mirnagenenormal.rda")
head(E9GE_mirnagenenormal)
```

```
## # A tibble: 6 x 7
##   Hugo_Symbol miRNA_name mirna_RPM GE_normal energy seed_type_effect
##   <chr>      <chr>      <dbl>   <dbl> <dbl>      <dbl>
## 1 ENAH      hsa-let-7~    111204.    7540 -22.7      0.01
## 2 GALNT2    hsa-let-7~    111204.    2824 -20.5      0.14
## 3 RLF       hsa-let-7~    111204.    1144 -18        0.028
## 4 MAST2     hsa-let-7~    111204.    2640 -22.6      0.24
## 5 DOCK1     hsa-let-7~    111204.    4826 -20.5      0.01
## 6 ZBTB16    hsa-let-7~    111204.     315 -24.3      0.14
## # ... with 1 more variable: region_effect <dbl>
```

```
BCPME_mirnatumor%>%
  dplyr::inner_join(tocombine_mirnagene, by = c("miRNA_name"))%>%
  dplyr::inner_join(TCGA_E9_A1N5_tumor, by = c("Ensembl_Gene_Id"="ensembl_gene_id", "Hugo_Symbol"="external_gene_name"))%>%
  distinct()%>%
  dplyr::select(Hugo_Symbol, miRNA_name, mirna_RPM= reads_per_million_miRNA_mapped, GE_tumor, energy, seed_type_effect)

#saveRDS(E9GE_mirnagenetumor, "data/E9GE_mirnagenetumor.rda")

E9GE_mirnagenetumor<-readRDS("data/E9GE_mirnagenetumor.rda")
head(E9GE_mirnagenetumor)
```

```
## # A tibble: 6 x 7
##   Hugo_Symbol miRNA_name mirna_RPM GE_tumor energy seed_type_effect
##   <chr>      <chr>      <dbl>   <dbl> <dbl>      <dbl>
## 1 ENAH      hsa-let-7~    62406.    9674 -22.7      0.01
## 2 GALNT2    hsa-let-7~    62406.    4481 -20.5      0.14
## 3 RLF       hsa-let-7~    62406.    1524 -18        0.028
## 4 MAST2     hsa-let-7~    62406.    4262 -22.6      0.24
## 5 DOCK1     hsa-let-7~    62406.    5455 -20.5      0.01
## 6 ZBTB16    hsa-let-7~    62406.     183 -24.3      0.14
## # ... with 1 more variable: region_effect <dbl>
```

## 2.5 Selection of trigger node

We have compared two datasets that are obtained for the tumor and normal tissue samples of same patient. We tried to change expression of a gene in normal tissue as the same level in the tumor tissue.

For this step, we have determined the changes of the gene expression in terms of fold change:

```
E9GE_mirnagenetumor%>%
  dplyr::select(Hugo_Symbol, GE_tumor)%>%
  dplyr::inner_join((E9GE_mirnagenenormal%>%dplyr::select(Hugo_Symbol, GE_normal)), by = "Hugo_Symbol")%>%
  dplyr::mutate(FC= GE_tumor/GE_normal)%>%
  distinct()%>%
  filter(FC>2.5, FC<3.5) -> three_fold_change

# SERPINE2 gene has 2982 read count in tumor tissue although 1083 in normal tissue (FC=2.75)
```

Secondly, we have determined the most important nodes of network. We applied `find_node_perturbation` function. We only defined nodes that affect the other nodes more than 1.05 fold change with 10 iteration when they increase 3 fold.

This dataset, `perturbationofnetwork`, includes 382 genes of totally 423 nodes.

Result of this, we obtained common nodes of these two datasets (i.e `perturbationofnetwork` and `three_fold_change`) and selected a gene, `SERPINE2`.

```
three_fold_change%>%
  inner_join(perturbationofnetwork, by = c("Hugo_Symbol"="name"))

#Selected node is SERPINE2
```

## 2.5 Approach of Method into Combined Datasets

We selected `SERPINE2` gene for simulation of regulation on network.

### 2.5.1 Find iteration of simulation

The node amount of changed gene on the system in terms of percentage were shown at Figure S9 in Supplementary Figures. As seen, firstly, the changed gene count increase. The system which contains the hundreds of miRNAs and thousands of genes can slowly gain the steady-state again. At first glance, it can be assumed that when all nodes in the system are reached, stable state will be provided. However, although all nodes are reached, the nodes competing with each other cause the edits to continue for a while.

The dynamics of the approach are shown in package vignettes.

So, we offered an approach about to find iteration. `find_iteration` function does not give the iteration to gain steady-state, but it gives the iteration which has maximum affected node counts. The function is applied as following:

```
as.data.frame(E9GE_mirnagenenormal)%>%
  priming_graph(competing_count = GE_normal, miRNA_count = mirna_RPM, aff_factor = c(energy, seed_type_e
  update_how("SERPINE2",2.75) %>%
  simulate(100) %>%
  find_iteration(limit=1, plot= FALSE)
```

```
## Warning in priming_graph(., competing_count = GE_normal, miRNA_count = mirna_RPM, : First column is p
```

```
## [1] NA
```

#31

## 2.5.2 Simulation of dataset

We tried to apply two fold of the point that SERPINE2 has maximum affected genes on network.

```
as.data.frame(E9GE_mirnagenenormal)%>%
  priming_graph(competing_count = GE_normal, miRNA_count = mirna_RPM, aff_factor = c(energy, seed_type_
  update_how("SERPINE2",2.75)%>%
  simulate(62)
```

```
## Warning in priming_graph(., competing_count = GE_normal, miRNA_count = mirna_RPM, : First column is p

## # A tbl_graph: 7891 nodes and 23936 edges
## #
## # A directed acyclic simple graph with 13 components
## #
## # Node Data: 7,891 x 7 (active)
##   name type node_id initial_count count_pre count_current
##   <chr> <chr>   <int>         <dbl>     <dbl>      <dbl>
## 1 ENAH  Comp~     1           7540      7547.      7547.
## 2 GALN~ Comp~     2           2824      2824.      2824.
## 3 RLF   Comp~     3           1144      1168.      1168.
## 4 MAST2 Comp~     4           2640      2640.      2640.
## 5 DOCK1 Comp~     5           4826      4827.      4827.
## 6 ZBTB~ Comp~     6            315       315       315
## # ... with 7,885 more rows, and 1 more variable: changes_variable <chr>
## #
## # Edge Data: 23,936 x 23
##   from to Competing_name miRNA_name GE_normal mirna_RPM energy
##   <int> <int> <chr>         <chr>         <dbl>     <dbl> <dbl>
## 1     1   7589 ENAH          hsa-let-7~     7540    111204. -22.7
## 2     2   7589 GALNT2       hsa-let-7~     2824    111204. -20.5
## 3     3   7589 RLF          hsa-let-7~     1144    111204. -18
## # ... with 2.393e+04 more rows, and 16 more variables:
## #   seed_type_effect <dbl>, region_effect <dbl>, dummy <dbl>,
## #   affff_factor <dbl>, degg_factor <dbl>, comp_count_list <list>,
## #   comp_count_pre <dbl>, comp_count_current <dbl>,
## #   mirna_count_list <list>, mirna_count_pre <dbl>,
## #   mirna_count_current <dbl>, mirna_count_per_dep <dbl>,
## #   effect_current <dbl>, effect_pre <dbl>, effect_list <list>,
## #   mirna_count_per_comp <dbl>

as.data.frame(E9GE_mirnagenenormal)%>%
  priming_graph(competing_count = GE_normal, miRNA_count = mirna_RPM, aff_factor = c(energy, seed_type_
  update_how("SERPINE2",2.75)%>%
  simulate(62)%>%
  as_tibble()%>%
  select(name, initial_count, count_current)->simulation_results
```

```
## Warning in priming_graph(., competing_count = GE_normal, miRNA_count = mirna_RPM, : First column is p
```

### 2.5.3 Comparison of simulation results and tumor tissue expression values

```
E9GE_mirnagenetumor%>%
  dplyr::select(Hugo_Symbol, GE_tumor)%>%
  dplyr::inner_join((E9GE_mirnagenenormal%>%dplyr::select(Hugo_Symbol, GE_normal)), by = "Hugo_Symbol")%
  inner_join(simulation_results, by= c("Hugo_Symbol"="name"))%>%
  distinct()
```

```
## # A tibble: 7,465 x 5
##   Hugo_Symbol GE_tumor GE_normal initial_count count_current
##   <chr>       <dbl>    <dbl>         <dbl>         <dbl>
## 1 ENAH        9674      7540          7540          7547.
## 2 GALNT2      4481      2824          2824          2824.
## 3 RLF         1524      1144          1144          1168.
## 4 MAST2       4262      2640          2640          2640.
## 5 DOCK1       5455      4826          4826          4827.
## 6 ZBTB16      183       315           315           315
## 7 ZNF215       20       51            51            51.3
## 8 SESN3       4260      7220          7220          12402.
## 9 PAH          7       12            12            12
## 10 WNK1      15671     8514          8514          8547.
## # ... with 7,455 more rows
```

Actually, we have developed to provide a new approach mirna mediated regulation networks. This approach may not explain the whole regulation behaviors between miRNAs and targets but can be first step to more detailed and coherent miRNA:target regulation approach.

## 3. REFERENCES