Supplementary Materials And Methods

Selcen Ari Alper Yilmaz

2020-08-11

Contents

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 this study are developed with R and are available in Bioconductor.

```r
library(ceRNAnetsim)
```

- load *minsamp* data

```r
minsamp <- readRDS("data/minsamp.RDS")

minsamp
```

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

See Table S1 in Supplementary Tables file.

2. *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 changes_variable
##   <chr> <chr>       <int>         <dbl>     <dbl>         <dbl> <chr>
## 1 Gene1 Competing       1         10000     10000         10000 Competing
## 2 Gene2 Competing       2         10000     10000         10000 Competing
## 3 Gene3 Competing       3          5000      5000          5000 Competing
## 4 Gene4 Competing       4         10000     10000         10000 Competing
## 5 Gene5 Competing       5          5000      5000          5000 Competing
## 6 Gene6 Competing       6         10000     10000         10000 Competing
## # ... with 2 more rows
## #
## # Edge Data: 7 x 19
##    from    to Competing_name miRNA_name Competing_expre~ miRNA_expression dummy
##   <int> <int> <chr>          <chr>                 <dbl>            <dbl> <dbl>
## 1     1     7 Gene1          Mir1                  10000             1000     1
## 2     2     7 Gene2          Mir1                  10000             1000     1
## 3     3     7 Gene3          Mir1                   5000             1000     1
## # ... with 4 more rows, and 12 more variables: 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 changes_variable
##   <chr> <chr>       <int>         <dbl>     <dbl>         <dbl> <chr>
## 1 Gene1 Competing       1         10000     10000         10000 Competing
## 2 Gene2 Competing       2         10000     10000         20000 Up
## 3 Gene3 Competing       3          5000      5000          5000 Competing
## 4 Gene4 Competing       4         10000     10000         10000 Competing
## 5 Gene5 Competing       5          5000      5000          5000 Competing
## 6 Gene6 Competing       6         10000     10000         10000 Competing
## # ... with 2 more rows
## #
## # Edge Data: 7 x 19
##    from    to Competing_name miRNA_name Competing_expre~ miRNA_expression dummy
##   <int> <int> <chr>          <chr>                 <dbl>            <dbl> <dbl>
## 1     1     7 Gene1          Mir1                  10000             1000     1
## 2     2     7 Gene2          Mir1                  10000             1000     1
## 3     3     7 Gene3          Mir1                   5000             1000     1
## # ... with 4 more rows, and 12 more variables: 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 changes_variable
##   <chr> <chr>       <int>         <dbl>     <dbl>         <dbl> <chr>
## 1 Gene1 Competing       1         10000    10063.        10062. Down
## 2 Gene2 Competing       2         10000    19841.        19845. Up
## 3 Gene3 Competing       3          5000     5032.         5031. Down
## 4 Gene4 Competing       4         10000    10063.        10059. Down
## 5 Gene5 Competing       5          5000     5000          5001. Up
## 6 Gene6 Competing       6         10000    10000         10002. Up
## # ... with 2 more rows
## #
## # Edge Data: 7 x 20
##    from    to Competing_name miRNA_name Competing_expre~ miRNA_expression dummy
##   <int> <int> <chr>          <chr>                 <dbl>            <dbl> <dbl>
## 1     1     7 Gene1          Mir1                  10000             1000     1
## 2     2     7 Gene2          Mir1                  10000             1000     1
## 3     3     7 Gene3          Mir1                   5000             1000     1
## # ... with 4 more rows, and 13 more variables: 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
```

3. *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_factor = c(energy,
        seed_type), deg_factor = region)
```

```
## # 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 changes_variable
##   <chr> <chr>       <int>         <dbl>     <dbl>         <dbl> <chr>
## 1 Gene1 Competing       1         10000     10000         10000 Competing
## 2 Gene2 Competing       2         10000     10000         10000 Competing
## 3 Gene3 Competing       3          5000      5000          5000 Competing
## 4 Gene4 Competing       4         10000     10000         10000 Competing
## 5 Gene5 Competing       5          5000      5000          5000 Competing
## 6 Gene6 Competing       6         10000     10000         10000 Competing
## # ... with 2 more rows
## #
## # Edge Data: 7 x 22
##    from    to Competing_name miRNA_name Competing_expre~ miRNA_expression energy
##   <int> <int> <chr>          <chr>                 <dbl>            <dbl>  <dbl>
## 1     1     7 Gene1          Mir1                  10000             1000    -20
## 2     2     7 Gene2          Mir1                  10000             1000    -15
## 3     3     7 Gene3          Mir1                   5000             1000    -14
## # ... with 4 more rows, and 15 more variables: 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_factor = c(energy,
        seed_type), deg_factor = region) %>% 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 changes_variable
##   <chr> <chr>       <int>         <dbl>     <dbl>         <dbl> <chr>
## 1 Gene1 Competing       1         10000     10000         10000 Competing
## 2 Gene2 Competing       2         10000     10000         20000 Up
## 3 Gene3 Competing       3          5000      5000          5000 Competing
## 4 Gene4 Competing       4         10000     10000         10000 Competing
## 5 Gene5 Competing       5          5000      5000          5000 Competing
## 6 Gene6 Competing       6         10000     10000         10000 Competing
## # ... with 2 more rows
## #
## # Edge Data: 7 x 22
##    from    to Competing_name miRNA_name Competing_expre~ miRNA_expression energy
##   <int> <int> <chr>          <chr>                 <dbl>            <dbl>  <dbl>
## 1     1     7 Gene1          Mir1                  10000             1000    -20
## 2     2     7 Gene2          Mir1                  10000             1000    -15
## 3     3     7 Gene3          Mir1                   5000             1000    -14
## # ... with 4 more rows, and 15 more variables: 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_factor = c(energy,
        seed_type), deg_factor = region) %>% 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 changes_variable
##   <chr> <chr>       <int>         <dbl>     <dbl>         <dbl> <chr>
## 1 Gene1 Competing       1         10000    10065.        10064. Down
## 2 Gene2 Competing       2         10000    19997.        19997. Up
## 3 Gene3 Competing       3          5000     5023.         5023. Down
## 4 Gene4 Competing       4         10000    10029.        10028. Down
## 5 Gene5 Competing       5          5000     5000          5000. Up
## 6 Gene6 Competing       6         10000    10000         10000. Up
## # ... with 2 more rows
## #
## # Edge Data: 7 x 23
##    from    to Competing_name miRNA_name Competing_expre~ miRNA_expression energy
##   <int> <int> <chr>          <chr>                 <dbl>            <dbl>  <dbl>
## 1     1     7 Gene1          Mir1                  10000             1000    -20
## 2     2     7 Gene2          Mir1                  10000             1000    -15
## 3     3     7 Gene3          Mir1                   5000             1000    -14
## # ... with 4 more rows, and 16 more variables: 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 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_factor = c(energy,
        seed_type), deg_factor = region) %>% 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
```

4. 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) .

```r
priming_graph(minsamp, competing_count = Competing_expression,
    miRNA_count = miRNA_expression, aff_factor = c(energy,
        seed_type), deg_factor = region) %>% 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 changes_variable
##   <chr> <chr>       <int>         <dbl>     <dbl>         <dbl> <chr>
## 1 Gene1 Competing       1         10000    10028.        10027. Down
## 2 Gene2 Competing       2         10000    10001.        10001. Down
## 3 Gene3 Competing       3          5000     5010.         5009. Down
## 4 Gene4 Competing       4         10000    19803.        19806. Up
## 5 Gene5 Competing       5          5000     5024.         5024. Down
## 6 Gene6 Competing       6         10000    10044.        10044. Down
## # ... with 2 more rows
## #
## # Edge Data: 7 x 23
##    from    to Competing_name miRNA_name Competing_expre~ miRNA_expression energy
##   <int> <int> <chr>          <chr>                 <dbl>            <dbl>  <dbl>
## 1     1     7 Gene1          Mir1                  10000             1000    -20
## 2     2     7 Gene2          Mir1                  10000             1000    -15
## 3     3     7 Gene3          Mir1                   5000             1000    -14
## # ... with 4 more rows, and 16 more variables: 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 at following).

```
priming_graph(minsamp, competing_count = Competing_expression,
    miRNA_count = miRNA_expression, aff_factor = c(energy,
        seed_type), deg_factor = region) %>% 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
```

5. Determination of perturbation efficiencies of elements in system.

```
sample_graph <- priming_graph(minsamp, competing_count = Competing_expression,
    miRNA_count = miRNA_expression, aff_factor = c(energy,
        seed_type), deg_factor = region)

find_node_perturbation(sample_graph, how = 2, cycle = 3,
    limit = 0.1) %>% as_tibble() %>% select(name, perturbation_efficiency,
    perturbed_count)
```

```
## # A tibble: 8 x 3
##   name  perturbation_efficiency perturbed_count
##   <chr>                   <dbl>           <dbl>
## 1 Gene1                 0.132                 2
## 2 Gene2                 0.198                 3
## 3 Gene3                 0.0555                2
## 4 Gene4                 0.197                 4
## 5 Gene5                 0.143                 1
## 6 Gene6                 0.131                 1
## 7 Mir1                  0.806                 3
## 8 Mir2                  2.80                  3
```

6. Additional data manipulation steps

6.1 Arrangement of CLASH dataset

CLASH dataset was retrieved from PubMed (Helwak et al. 2013).

```r
clashelwak <- read.table("mmc1.txt", comment.char = "#",
    header = TRUE, skip = 1, stringsAsFactors = FALSE)

# hg19
```

Query of Human Genome 19.

Human genome 19 information was handled through biomaRt package.

```r
# HG19
listEnsemblArchives()
listMarts(host = "http://grch37.ensembl.org")
ensemblgrch37 = useMart(host = "http://grch37.ensembl.org",
    biomart = "ENSEMBL_MART_ENSEMBL", dataset = "hsapiens_gene_ensembl")
hg19 <- getBM(attributes = c("ensembl_transcript_id",
    "ensembl_gene_id", "chromosome_name", "start_position",
    "end_position", "hgnc_symbol", "entrezgene_id",
    "strand"), mart = ensemblgrch37)
```

Adding miRNA and gene information

```r
clashelwak <- clashelwak %>% separate(microRNA_name,
    c("Barcode", "Database", "mirna_name", "type"),
    sep = "_") %>% separate(mRNA_name, c("Ensembl_Gene_Id",
    "Ensembl_Transcript_Id", "Hugo_Symbol", "mRNA_Type"),
    sep = "_")
```

MiRNA releases are obtained from miRBase. In this step, release 21 (in Human genome 38) was downloaded.

```r
mirbasehg38 <- read.table("mirbasehg38.txt", comment.char = "#") %>%
    filter(V3 != "miRNA_primary_transcript") %>% separate(V9,
    c("ID", "Alias", "Name", "Precusor"), sep = ";") %>%
    mutate(ID = substr(ID, 4, length(ID)), Alias = substr(Alias,
        7, length(Alias)), Name = substr(Name, 6, length(Name)),
        Precusor = substr(Precusor, 14, length(Precusor))) %>%
    dplyr::select(chr = V1, start = V4, end = V5, strand = V7,
        ID, Alias, Name, Precusor)
```

CLASH dataset is published in miRBase release 15 and Human Genome 19 version.

```r
clashelwakfinal <- read_tsv("mirna_mature.txt", col_names = FALSE) %>%
    filter(startsWith(X2, "hsa")) %>% dplyr::select(mirna_ID = X2,
    mirbase_ID = X3) %>% inner_join(mirbasehg38 %>%
    dplyr::select(ID, Name), by = c(mirbase_ID = "ID")) %>%
    dplyr::select(mirbase_ID, Name) %>% distinct() %>%
    inner_join(clashelwak, by = c(mirbase_ID = "Barcode")) %>%
    dplyr::select(Name, miRNA_seq, Ensembl_Gene_Id,
        Ensembl_Transcript_Id, Hugo_Symbol, mRNA_seq_extended,
        chimeras_decompressed, seed_type, seed_basepairs,
        folding_class, seq_ID, folding_energy, X5.UTR,
        CDS, X3.UTR) %>% inner_join(hg19, by = c(Ensembl_Gene_Id = "ensembl_gene_id",
    Ensembl_Transcript_Id = "ensembl_transcript_id",
    Hugo_Symbol = "hgnc_symbol")) %>% mutate(region1 = ifelse(X5.UTR ==
    "1", "5UTR", " "), region2 = ifelse(X3.UTR == "1",
    "3UTR", " "), region3 = ifelse(CDS == "1", "CDS",
    " ")) %>% unite(region, c(region1, region2, region3),
    sep = "||") %>% dplyr::select(chromosome_name,
    start_position, end_position, strand, Hugo_Symbol,
    Ensembl_Gene_Id, Ensembl_Transcript_Id, mRNA_seq_extended,
    Name, miRNA_seq, seq_ID, seed_type, seed_basepairs,
    folding_class, folding_energy, region) %>% as_tibble()
```

Converting CLASH data to human genome 38 build.

There are different liftover methods for conversion among Human Genome builds. We preffered to use UCSC liftover tool

```
# Obtaining chromosomal locations from miRNA:target
# interaction dataset.

lift19 <- clashelwakfinal %>% dplyr::select(1, 2, 3) %>%
    unite(start_end, c("start_position", "end_position"),
        sep = "-") %>% mutate(Chromosome = paste0("chr",
    chromosome_name, "")) %>% unite(chromosome_name,
    c("Chromosome", "start_end"), sep = ":")

write_tsv(lift19, "lift19.txt")

# After we searched this file in UCSC browser, the
# output loaded (lift19_del deleted regions on the
# HG38 genome build; hg38clashcomp new locations of
# the genes on HG38)

lift19_del <- read_tsv("deleted_lift19.txt")
colnames(lift19_del)[1] <- "chromosome_loc"

lift19_del <- lift19_del %>% dplyr::filter(startsWith(chromosome_loc,
    "chr")) %>% separate(chromosome_loc, c("Chr", "End"),
    "-", remove = TRUE) %>% separate(Chr, c("Chr",
    "Start"), ":", remove = TRUE)

lift19_del$Start <- as.numeric(lift19_del$Start)
lift19_del$End <- as.numeric(lift19_del$End)

# removing deleted location from CLASH dataset

clashelwakfinal <- clashelwakfinal %>% mutate(Chromosome = paste0("chr",
    chromosome_name, "")) %>% dplyr::anti_join(lift19_del,
    by = c(Chromosome = "Chr", start_position = "Start",
        end_position = "End"))

hg38clash <- read.delim("hg38clashcomp.txt", header = FALSE,
```

```r
    stringsAsFactors = FALSE)

# adding new location information:

clashelwakfinal <- clashelwakfinal %>% bind_cols(hg38clash)

colnames(clashelwakfinal)[18] <- "HG38build_loc"

clashelwakfinal <- clashelwakfinal %>% dplyr::mutate(Genom_build = rep("hg19"))

# Arrangement in dataset

clashelwakfinal <- clashelwakfinal %>% dplyr::select(cluster = seq_ID,
    chromosome = Chromosome, start_position, end_position,
    strand, hgnc_symbol = Hugo_Symbol, Ensembl_Gene_Id,
    Ensembl_Transcript_Id, target_seq = mRNA_seq_extended,
    miRNA = Name, miR_seq = miRNA_seq, seed_type, seed_type2 = seed_basepairs,
    seed_type3 = folding_class, Energy = folding_energy,
    HG38build_loc, Genom_build, region)


clashelwakfinal$strand <- as.character(clashelwakfinal$strand)

str(clashelwakfinal)
```

Interpreting the CLASH seed structures in dataset

```r
clashelwakfinal <- clashelwakfinal %>% mutate(seed_type = ifelse(seed_type ==
    "noncanonical_seed" & seed_type2 > 4 & seed_type3 ==
    "I", paste0(seed_type2, "-mer"), seed_type), seed_type = ifelse(seed_type ==
    "noncanonical_seed" & seed_type2 > 4 & seed_type3 ==
    "II", paste0(seed_type2, "-mer_noncanonical"),
    seed_type), seed_type = ifelse(seed_type == "noncanonical_seed" &
    seed_type2 > 4 & seed_type3 == "III", paste0(seed_type2,
    "-mer_noncanonical"), seed_type), seed_type = ifelse(seed_type ==
    "noncanonical_seed" & seed_type2 > 4 & seed_type3 ==
    "IV", paste0(seed_type2, "-mer_noncanonical"),
    seed_type), seed_type = ifelse(startsWith(seed_type,
    "no"), "none", seed_type)) %>% dplyr::select(-seed_type2,
    -seed_type3)
```

6.2 Arrangement of CLEAR-CLiP Dataset (Moore et al. 2015)

CLASH dataset was retrieved from Nature article

```r
clearclip <- read_xlsx("CLEAR-CLIP.xlsx")

# Clearclip hg18
```

Query of Human Genome 18

```r
# HG18
listEnsemblArchives()
listMarts(host = "may2009.archive.ensembl.org")
ensembl54 = useMart(host = "may2009.archive.ensembl.org",
    biomart = "ENSEMBL_MART_ENSEMBL", dataset = "hsapiens_gene_ensembl")

hg18 <- getBM(attributes = c("ensembl_transcript_id",
    "ensembl_gene_id", "chromosome_name", "start_position",
    "end_position", "hgnc_symbol", "entrezgene", "strand"),
    mart = ensembl54)
```

Adding Genome Information to dataset

```r
clearclipfinal <- hg18 %>% inner_join(clearclip, by = c(entrezgene = "gene.id",
    hgnc_symbol = "gene.symbol")) %>% distinct()
```

Converting human genome build

```r
# Obtaining chromosomal locations from miRNA:target
# interaction dataset.

lift18 <- clearclipfinal %>% unite(start_end, c("start_position",
    "end_position"), sep = "-") %>% unite(location,
    c("chr", "start_end"), sep = ":") %>% dplyr::select(location)

write_tsv(lift18, "lift18.txt")

# After we searched this file in UCSC browser, the
# output loaded (deleted_lift18 deleted regions on
# the HG38 genome build; hg38clearclip new
# locations of the genes on HG38)

deleted_lift18 <- read_tsv("deleted_lift18.txt")

colnames(deleted_lift18)[1] <- "Chromosome_loc"

deleted_lift18 <- deleted_lift18 %>% dplyr::filter(startsWith(Chromosome_loc,
    "chr")) %>% separate(Chromosome_loc, c("Chr", "End"),
    "-", remove = TRUE) %>% separate(Chr, c("Chr",
    "Start"), ":", remove = TRUE)

deleted_lift18$Start <- as.numeric(deleted_lift18$Start)
deleted_lift18$End <- as.numeric(deleted_lift18$End)

# removing deleted location from CLEAR-CLiP dataset

clearclipfinal <- clearclipfinal %>% dplyr::anti_join(deleted_lift18,
    by = c(chr = "Chr", start_position = "Start", end_position = "End"))

hg38clearclip <- read.delim("hg38clearclip.txt", header = FALSE,
    stringsAsFactors = FALSE)

clearclipfinal <- clearclipfinal %>% bind_cols(hg38clearclip)

colnames(clearclipfinal)[28] <- "HG38build_loc"

# adding new location information:

clearclipfinal <- clearclipfinal %>% dplyr::mutate(Genom_build = rep("hg18"))

# Arrangement in dataset

clearclipfinal <- clearclipfinal %>% dplyr::select(cluster = cluster.ID,
    chromosome = chr, start_position, end_position,
    strand = strand.y, hgnc_symbol, Ensembl_Gene_Id = ensembl_gene_id,
    Ensembl_Transcript_Id = ensembl_transcript_id,
    target_seq = target.map, miRNA, miR_seq = miR.map,
    seed_type = "seed match", Energy = MFE, HG38build_loc,
    Genom_build, region)
```

Seed type manipulation in CLEAR-CLiP dataset

In CLEAR-CLiP dataset, seed types were shown in detail. We adjusted as canonical and non-canonical.

```
clipdata_seed <- data_frame(seed_type = c("5mer_1",
    "5mer_2", "5mer_3", "6mer", "6mer.indel", "6mer.mm",
    "6mer_off.mm", "6merA1", "6merA1.indel", "6merA1.mm",
    "7merA1", "7merA1.indel", "7merA1.mm", "7merm8",
    "7merm8.indel", "7merm8,mm", "8mer", "8mer.indel",
    "8mer.mm", "NA"), seed_type_com = c("5-mer", "5-mer_noncanonical",
    "5-mer_noncanonical", "6-mer", "6-mer_noncanonical",
    "6-mer_noncanonical", "6-mer_noncanonical", "6-merA1",
    "6-merA1_noncanonical", "6-merA1_noncanonical",
    "7-merA1", "7-merA1_noncanonical", "7-merA1_noncanonical",
    "7-mer-8m", "7-mer-8m_noncanonical", "7-mer-8m_noncanonical",
    "8-mer", "8-mer_noncanonical", "8-mer_noncanonical",
    "none"))

clearclipfinal <- clearclipfinal %>% inner_join(clipdata_seed,
    by = "seed_type") %>% dplyr::select(1:11, seed_type = seed_type_com,
    Energy, HG38build_loc, Genom_build, region)
```

6.3 Integration of two experimental dataset

```
experimentalmirnagene <- bind_rows(clashelwakfinal,
    clearclipfinal) %>% distinct()
```

Adding Coefficients of Interaction factors

Energy values in miRNA:target pairs are represented by high-throughput studies (Helwak et al. 2013; Moore et al. 2015) which are utilized in this study. On the other hand, we have specified the other interaction factors, seed type and location of binding region on the target, as numeric values based on the previous studies.(Grimson et al. 2007) have compared the seed types' effect on target repression with few miRNA had canonical seed pairing in their study. Additionally, (Bartel 2009) and (Betel et al. 2010) have studied on functional and non-functional seed interactions. Based on results of these studies we have arranged seed types of miRNA:target interactions as numeric values. We also have redefined location of binding region on the target as numeric values, based on studies of (Hausser et al. 2013) and (Helwak et al. 2013). With this process, we have handled this integrated dataset in context of competitor behaviors and functionality of interactions.

In this step we added numeric intraction values at followings

Fistly, we organized these values due to the fact that the regions were defined differently in two datasets. After that, region effect was added as numeric values (shown in Table S3).

```
experimentalmirnagene <- experimentalmirnagene %>%
    mutate(region2 = str_replace_all(region, "NA",
        ""), region3 = str_replace_all(region2, "\\|",
        ""), region = str_replace_all(region3, c(‘3’UTR‘ = "3UTR",
        ‘5’UTR‘ = "5UTR"))) %>% dplyr::select(-region2,
    -region3) %>% mutate(region_effect = as.double(ifelse(region %in%
    c("3UTRCDS", "CDS3UTR", "5UTR3UTR", "CDS5UTR3UTR",
        "CDS3UTRintron"), "0.93", ifelse(region %in%
    c("CDS", "CDSintron"), "0.42", ifelse(region %in%
    c("3UTR", "3UTRintron"), "0.84", ifelse(region %in%
    c("5UTR", "5UTRintron"), "0.01", ifelse(region %in%
    c("5UTRCDS", "CDS5UTR"), "0.42", ifelse(region %in%
    c("intron", ""), "0.01", ifelse(region %in% c("exon_unclassified",
    ""), "0.2", NA)))))))))
```

Secondly, we organized seed type interactions in *Seed type manipulation* section for CLEAR-CLiP dataset to show as found in CLASH dataset. Same type formatted values added dataset as numeric values (shown in Table S2).

```
seed_type_effect <- data_frame(seed_type = c("5-mer",
    "5-mer_noncanonical", "6-mer", "6-mer_noncanonical",
    "6-merA1", "6-merA1_noncanonical", "7-mer", "7-mer_noncanonical",
    "7-merA1", "7-merA1_noncanonical", "7-mer-8m",
    "7-mer-8m_noncanonical", "8-mer", "8-mer_noncanonical",
    "9-mer", "9-mer_noncanonical", "none"), seed_type_effect = c(0.05,
    0.04, 0.07, 0.05, 0.07, 0.05, 0.23, 0.19, 0.19,
    0.16, 0.25, 0.21, 0.43, 0.35, 0.43, 0.35, 0.01))

experimentalmirnagene <- experimentalmirnagene %>%
    inner_join(seed_type_effect, by = "seed_type")
```

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

```
## # A tibble: 45,340 x 18
##    cluster chromosome start_position end_position strand hgnc_symbol
##    <chr>   <chr>              <int>       <int> <chr>  <chr>
##  1 0727A-~ chr5           162864575   162873157 1      CCNG1
##  2 L1HS-1~ chr14           95552565    95624347 -1     DICER1
##  3 L2HS-8~ chr6           109307640   109416022 -1     SESN1
##  4 L2HS-1~ chr5            36876861    37066515 1      NIPBL
##  5 L2-407~ chr4           106603784   106817143 -1     INTS12
##  6 L1HS-7~ chr5           130977407   131132710 -1     FNIP1
##  7 L1HS-4~ chr11          134123389   134135749 1      ACAD8
##  8 0727A-~ chr15           59397277    59417244 1      CCNB2
##  9 L2HS-1~ chr19           37001597    37019562 -1     ZNF260
## 10 L2HS-9~ chr11           64889252    64902004 -1     SYVN1
## # ... with 45,330 more rows, and 12 more variables: Ensembl_Gene_Id <chr>,
## #   Ensembl_Transcript_Id <chr>, target_seq <chr>, miRNA <chr>, miR_seq <chr>,
## #   seed_type <chr>, Energy <dbl>, HG38build_loc <chr>, Genom_build <chr>,
## #   region <chr>, region_effect <dbl>, seed_type_effect <dbl>
```

The context of dataset is shown in Table S5 in Supplementary Tables.

REFERENCES

Bartel, David P. 2009. "MicroRNAs: Target Recognition and Regulatory Functions." *Cell* 136 (2): 215–33. https://doi.org/10.1016/j.cell.2009.01.002.

Betel, Doron, Anjali Koppal, Phaedra Agius, Chris Sander, and Christina Leslie. 2010. "Comprehensive Modeling of microRNA Targets Predicts Functional Non-Conserved and Non-Canonical Sites." *Genome Biology* 11 (8): R90.

Grimson, Andrew, Kyle Kai-How Farh, Wendy K. Johnston, Philip Garrett-Engele, Lee P. Lim, and David P. Bartel. 2007. "MicroRNA Targeting Specificity in Mammals: Determinants Beyond Seed Pairing." *Molecular Cell* 27 (1): 91–105. https://doi.org/10.1016/j.molcel.2007.06.017.

Hausser, J., A. P. Syed, B. Bilen, and M. Zavolan. 2013. "Analysis of CDS-Located miRNA Target Sites Suggests That They Can Effectively Inhibit Translation." *Genome Research* 23 (4): 604–15. https://doi.org/10.1101/gr.139758.112.

Helwak, Aleksandra, Grzegorz Kudla, Tatiana Dudnakova, and David Tollervey. 2013. "Mapping the Human miRNA Interactome by CLASH Reveals Frequent Noncanonical Binding." *Cell* 153 (3): 654–65. https://doi.org/10.1016/j.cell.2013.03.043.

Moore, Michael J., Troels K. H. Scheel, Joseph M. Luna, Christopher Y. Park, John J. Fak, Eiko Nishiuchi, Charles M. Rice, and Robert B. Darnell. 2015. "miRNA-Target Chimeras Reveal miRNA 3'-End Pairing as a Major Determinant of Argonaute Target Specificity." *Nature Communications* 6 (November): 8864. https://doi.org/10.1038/ncomms9864.