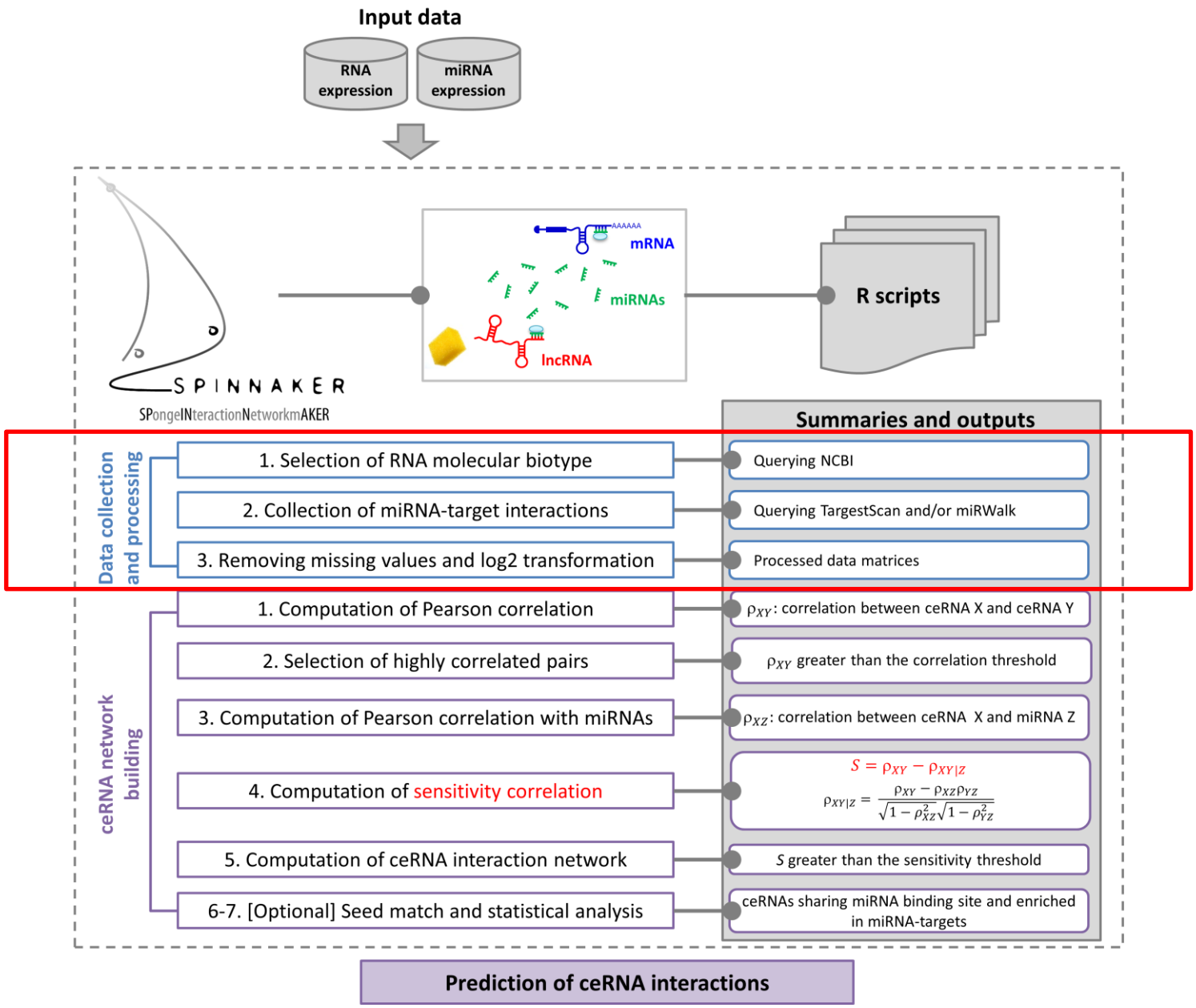
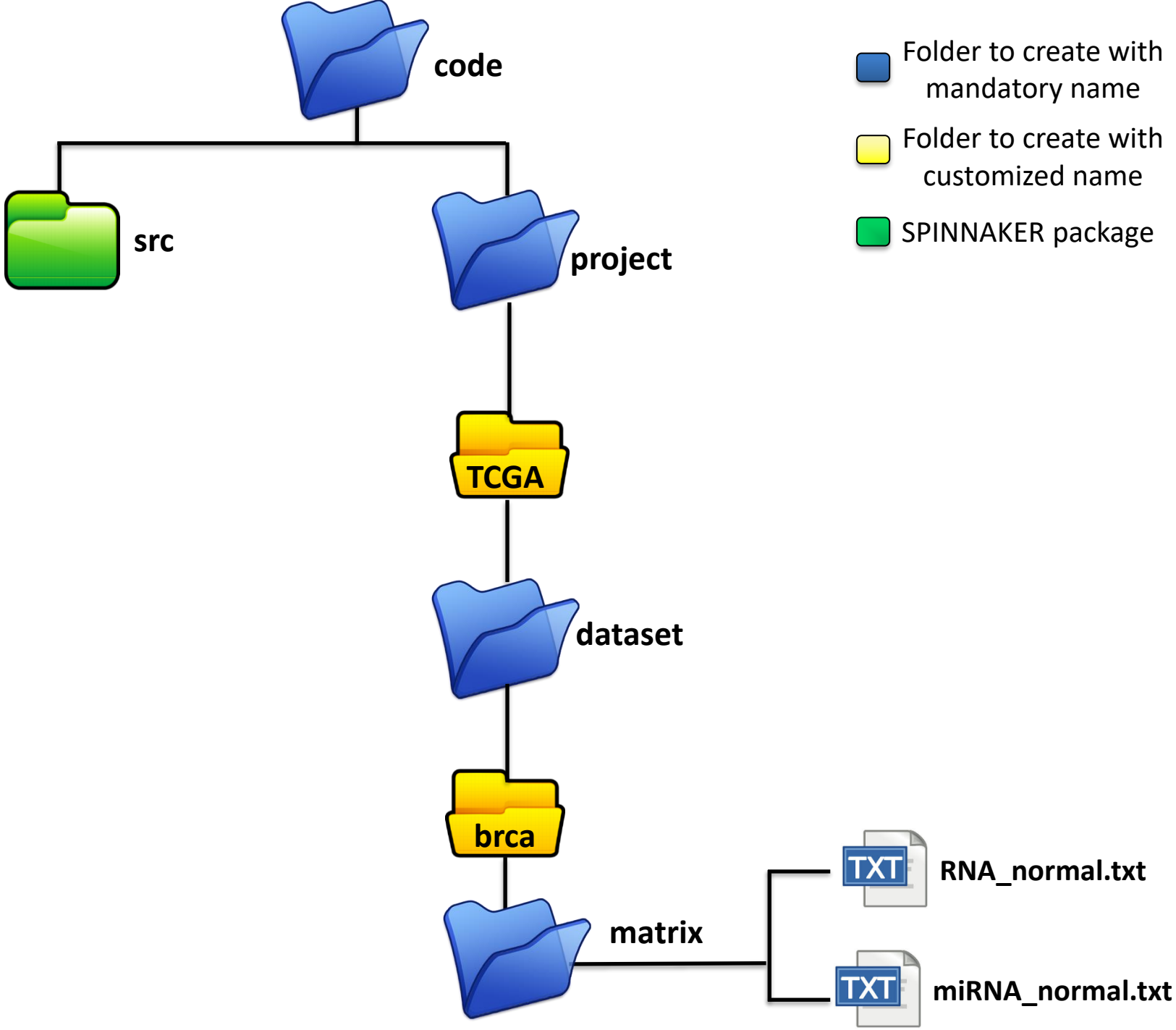
Module 1:
Data collection and processing


Input data
RNA expression
miRNA expression
SPINNAKER
SPongeINteractionNetworkmAKER
mRNA
miRNAs
lncRNA
R scripts
Summaries and outputs
Data collection and processing
1. Selection of RNA molecular biotype
Querying NCBI
2. Collection of miRNA-target interactions
Querying TargestScan and/or miRWalk
3. Removing missing values and log2 transformation
Processed data matrices
ceRNA network building
1. Computation of Pearson correlation
ρ_{XY}: correlation between ceRNA X and ceRNA Y
2. Selection of highly correlated pairs
ρ_{XY} greater than the correlation threshold
3. Computation of Pearson correlation with miRNAs
ρ_{XZ}: correlation between ceRNA X and miRNA Z
4. Computation of sensitivity correlation
$S = \rho_{XY} - \rho_{XY|Z}$
$\rho_{XY|Z} = \frac{\rho_{XY} - \rho_{XZ}\rho_{YZ}}{\sqrt{1-\rho_{XZ}^2}\sqrt{1-\rho_{YZ}^2}}$
5. Computation of ceRNA interaction network
S greater than the sensitivity threshold
6-7. [Optional] Seed match and statistical analysis
ceRNAs sharing miRNA binding site and enriched in miRNA-targets
Prediction of ceRNA interactions

Example dataset - brca

- Data of **miRNA- and RNA-sequencing** samples of breast invasive carcinoma (**brca**), downloaded from **TCGA**:
  - 1182 RNA-sequencing samples
    - 1069 tumor samples
    - 113 normal samples
  - 1212 miRNA-sequencing samples
    - 1108 tumor samples
    - 104 normal samples
- The analysis was restricted to **103** matched-normal samples (i.e. tissues that are adjacent to the tumor and taken from the same patient) for both the RNA-sequencing and miRNA-sequencing


code
src
project
TCGA
dataset
brca
matrix
TXT
RNA_normal.txt
TXT
miRNA_normal.txt
Folder to create with mandatory name
Folder to create with customized name
SPINNAKER package

Data collection and processing

```r
DataCollectionProcessing <- function(){

  ######################################
  # input parameters

  data_RNA <- input_file$data_RNA
  data_miRNA <- input_file$data_miRNA

  ceRNA1 <- input_parameter$ceRNA1
  ceRNA2 <- input_parameter$ceRNA2

  searchSeedMatch <- input_parameter$searchSeedMatch

  threshold_perc_missing_values <- input_parameter$threshold_perc_missing_values

  filename_data <- output_file$filename_data
  ######################################

  data_RNA <- removeMissingValues(data_RNA,threshold_perc_missing_values)
  data_miRNA <- removeMissingValues(data_miRNA,threshold_perc_missing_values)

  biotype <- getBiotype(rownames(data_RNA))

  data_ceRNA1 <- createDataCerna(data_RNA,biotype,ceRNA1)
  data_ceRNA2 <- createDataCerna(data_RNA,biotype,ceRNA2)

  res <- list(data_ceRNA1 = data_ceRNA1,
              data_ceRNA2 = data_ceRNA2,
              data_miRNA = data_miRNA)

  if(searchSeedMatch == "YES"){

    miRNAtarget <- getmiRNAtarget(ceRNA1,ceRNA2)

    res_seedmatch <- list(miRNAtarget = miRNAtarget)
    res <- c(res,res_seedmatch)

  }

  save(res,file = filename_data)

  return(res)
}
```

- The goal of this module is to collect and process data for running SPINNAKER
- This module is composed of three steps:
  i. Selection of RNA molecular biotype
  ii. Collection of miRNA-target interactions
  iii. Removing missing values and log2 transformation

Data collection

```
DataCollectionProcessing <- function(){

  ######################################
  # input parameters

  data_RNA <- input_file$data_RNA
  data_miRNA <- input_file$data_miRNA

  ceRNA1 <- input_parameter$ceRNA1
  ceRNA2 <- input_parameter$ceRNA2

  searchSeedMatch <- input_parameter$searchSeedMatch

  threshold_perc_missing_values <- input_parameter$threshold_perc_missing_values

  filename_data <- output_file$filename_data
  ######################################

  data_RNA <- removeMissingValues(data_RNA,threshold_perc_missing_values)
  data_miRNA <- removeMissingValues(data_miRNA,threshold_perc_missing_values)

  biotype <- getBiotype(rownames(data_RNA))

  data_ceRNA1 <- createDataCerna(data_RNA,biotype,ceRNA1)
  data_ceRNA2 <- createDataCerna(data_RNA,biotype,ceRNA2)

  res <- list(data_ceRNA1 = data_ceRNA1,
              data_ceRNA2 = data_ceRNA2,
              data_miRNA = data_miRNA)

  if(searchSeedMatch == "YES"){

    miRNAtarget <- getmiRNAtarget(ceRNA1,ceRNA2)

    res_seedmatch <- list(miRNAtarget = miRNAtarget)
    res <- c(res,res_seedmatch)

  }

  save(res,file = filename_data)

  return(res)
}
```

Step i: the molecular entities competing for the miRNA binding are retrieved

⚠ **Caveat:** SPINNAKER offers the possibility of choosing among different pool of RNAs acting as ceRNAs, as long as the total number of triplets to be tested is within the order of magnitude of $O(10^6)$, otherwise it collides with a huge computation complexity.

Data collection

getBiotype

```
getBiotype <- function(gene_symbol){
  ######################################
  # Query NCBI

  fileURL <- "ftp://ftp.ncbi.nlm.nih.gov/gene/DATA/GENE_INFO/Mammalia/Homo_sapiens.gene_info.gz"
  destfile <- "Homo_sapiens_gene_info.txt.gz"
  download.file(fileURL, destfile, method="auto")
  gunzip(destfile)
  filename <- gsub(".gz", "", destfile)

  NCBI_gene_info <- read.table(filename, sep ="\t", quote = "", header = T,
                               stringsAsFactors = F, comment.char = "", check.names = F)

  file.remove(filename)
  ######################################

  df_gene_symbol <- data.frame(Symbol = gene_symbol)

  df <- merge(df_gene_symbol, NCBI_gene_info, by = "Symbol", all.x = TRUE, sort = F)

  # class <- data.frame(table(df$type_of_gene))

  res <- df[,c("Symbol","type_of_gene")]

  return(res)
}
```

Step i: the molecular entities competing for the miRNA binding are retrieved

- SPINNAKER automatically queries **NCBI's Gene** database

Data collection

createDataCerna

```
createDataCerna <- function(data_RNA,biotype,ceRNA){

  if( length(ceRNA) == 1 ){

    ind <- which(biotype[,"type_of_gene"] == ceRNA)
    symbol <- biotype[ind,"Symbol"]

  }else{

    list <- lapply(ceRNA, function(x){

      ind <- which(biotype[,"type_of_gene"] == x)
      symbol <- biotype[ind,"Symbol"]

    })

    symbol <- unlist(list)

  }

  data_ceRNA <- data_RNA[symbol,]

  return(data_ceRNA)
}
```

Step i: the molecular entities competing for the miRNA binding are retrieved

- SPINNAKER automatically queries **NCBI's Gene** database
- SPINNAKER separates the two selected classes of candidate ceRNAs to be tested (e.g., **protein coding** versus **long non-coding RNAs**)

Data collection

```r
DataCollectionProcessing <- function(){

  ######################################
  # input parameters

  data_RNA <- input_file$data_RNA
  data_miRNA <- input_file$data_miRNA

  ceRNA1 <- input_parameter$ceRNA1
  ceRNA2 <- input_parameter$ceRNA2

  searchSeedMatch <- input_parameter$searchSeedMatch

  threshold_perc_missing_values <- input_parameter$threshold_perc_missing_values

  filename_data <- output_file$filename_data
  ######################################

  data_RNA <- removeMissingValues(data_RNA,threshold_perc_missing_values)
  data_miRNA <- removeMissingValues(data_miRNA,threshold_perc_missing_values)

  biotype <- getBiotype(rownames(data_RNA))

  data_ceRNA1 <- createDataCerna(data_RNA,biotype,ceRNA1)
  data_ceRNA2 <- createDataCerna(data_RNA,biotype,ceRNA2)

  res <- list(data_ceRNA1 = data_ceRNA1,
              data_ceRNA2 = data_ceRNA2,
              data_miRNA = data_miRNA)

  if(searchSeedMatch == "YES"){

    miRNAtarget <- getmiRNAtarget(ceRNA1,ceRNA2)

    res_seedmatch <- list(miRNAtarget = miRNAtarget)
    res <- c(res,res_seedmatch)

  }

  save(res,file = filename_data)

  return(res)
}
```

Step ii: the miRNA target interactions are retrieved

⚠ **Caveat:** This step will be performed only if SearchSeedMatch = “YES” in the config.R

Data collection

getmiRNAtarget

```
getmiRNAtarget <- function(ceRNA1,ceRNA2){

  condition1 <- all(ceRNA1 == "protein-coding" & ceRNA2 == "protein-coding")
  condition2 <- any(ceRNA1 != ceRNA2) &
    any(ceRNA1 == "protein-coding" | ceRNA2 == "protein-coding")

  if(condition1){
    miRNAtarget <- queryTargetScan()
  }else if(condition2){
    mRNA <- queryTargetScan()
    lncRNA <- querymiRWalk()

    mir_common <- intersect(names(mRNA),names(lncRNA))

    miRNAtarget <- mapply(c, mRNA[mir_common], lncRNA[mir_common], SIMPLIFY = F)

  }else{
    miRNAtarget <- querymiRWalk()
  }

  return(miRNAtarget)
}
```

Step ii: the miRNA target interactions are retrieved

- If the ceRNA1 and ceRNA 2 are both *protein coding* SPINNAKER retrieves only the predictions of **miRNA-mRNA** target interactions
- If one of the ceRNA 1 or ceRNA 2 is *protein coding* and the other one is *non-coding*, SPINNAKER retrieves both the predictions of **miRNA-mRNA** target interactions and the **miRNA-lncRNA** target interaction
- If ceRNA1 and ceRNA 2 are both *non-coding*, SPINNAKER retrieves only the predictions of **miRNA-lncRNA** target interactions

Data collection

queryTargetScan

```
queryTargetScan <- function(miRBase,TargetScan){

  ######################################
  # Query miRBase

  fileURL <- "ftp://mirbase.org/pub/mirbase/CURRENT/mature.fa.zip"
  destfile <- "mature.fa.zip"
  download.file(fileURL, destfile, method="auto")
  unzip(destfile)

  fastaFile <- readRNAStringSet("mature.fa")

  ID <- unlist(lapply(names(fastaFile),function(x){strsplit(x," ")[[1]][1]}))
  sequence <- paste(fastaFile)

  ind <- grep('hsa-', ID)

  ID <- ID[ind]
  sequence <- sequence[ind]

  seed  <- sapply(sequence, function(x){substr(x, 2, 8)})

  miRBase <- data.frame(ID, seed, row.names = NULL)

  unlink("mature.fa", force = TRUE)
  file.remove(destfile)
  ######################################
  # Query TargetScan

  fileURL <- "http://www.targetscan.org/vert_72/vert_72_data_download/Summary_Counts.default_predictions.txt.zip"
  destfile <- "Summary_Counts.default_predictions.txt.zip"
  download.file(fileURL, destfile, method="auto")
  unzip(destfile)
  filename <- gsub(".zip", "", destfile)

  TargetScan <- read.table(filename, sep = "\t", header = T, quote = "", check.names = F)

  TargetScan <- TargetScan[grep('9606', TargetScan$`Species ID`), c('Gene Symbol', 'miRNA family')]
  TargetScan <- unique(TargetScan)
  rownames(TargetScan) <- NULL

  file.remove(destfile)
  file.remove(filename)
  ######################################

  df <- merge(TargetScan,miRBase, by.x  = "miRNA family", by.y = "seed", all = F)

  mRNAtarget <- split(df$`Gene Symbol`, df$ID)

  return(mRNAtarget)
}
```

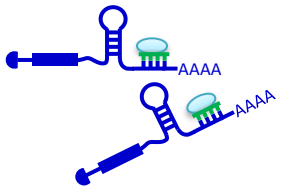
Step ii: the miRNA target interactions are retrieved

- miRNA **seed sequences** and **miRNA identifiers** are retrieved from **miRBase**
- miRNA-mRNA **target** interactions are retrieved from **TargetScan**

>hsa-miR-200a-3p
UAACACUGUCUGGUAACGAUGU

hsa-miR-200a-3p
AACACUG

AACACUG ZEB1
AACACUG ZEB2



hsa-miR-200a-3p ZEB1
hsa-miR-200a-3p ZEB2

Data collection

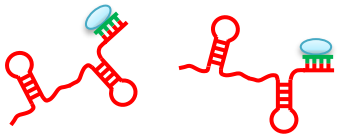
querymiRWalk

```
querymiRWalk <- function(){
  ########################################
  # Query miRWalk

  fileURL <- "http://zmf.umm.uni-heidelberg.de/apps/zmf/mirwalk2/downloads/lncrna/shsalncRNAgr.zip"
  destfile <- "shsalncRNAgr.zip"
  download.file(fileURL, destfile, method="auto")
  unzip(destfile)
  fileRdata <- "hsaLncRNA-gene.rdata"

  load(fileRdata) # load "hsa" large list

  file.remove(fileRdata)
  file.remove(destfile)
  ########################################

  return(hsa)
```

Step ii: the miRNA target interactions are retrieved

- miRNA-lncRNA **target** interactions are retrieved from **miRWalk**

hsa-miR-200a-3p PVT1
hsa-miR-200a-3p MALAT1



Data processing

```
DataCollectionProcessing <- function(){

  ########################################
  # input parameters

  data_RNA <- input_file$data_RNA
  data_miRNA <- input_file$data_miRNA

  ceRNA1 <- input_parameter$ceRNA1
  ceRNA2 <- input_parameter$ceRNA2

  searchSeedMatch <- input_parameter$searchSeedMatch

  threshold_perc_missing_values <- input_parameter$threshold_perc_missing_values

  filename_data <- output_file$filename_data
  ########################################

  data_RNA <- removeMissingValues(data_RNA,threshold_perc_missing_values)
  data_miRNA <- removeMissingValues(data_miRNA,threshold_perc_missing_values)

  biotype <- getBiotype(rownames(data_RNA))

  data_ceRNA1 <- createDataCerna(data_RNA,biotype,ceRNA1)
  data_ceRNA2 <- createDataCerna(data_RNA,biotype,ceRNA2)

  res <- list(data_ceRNA1 = data_ceRNA1,
              data_ceRNA2 = data_ceRNA2,
              data_miRNA = data_miRNA)

  if(searchSeedMatch == "YES"){

    miRNAtarget <- getmiRNAtarget(ceRNA1,ceRNA2)

    res_seedmatch <- list(miRNAtarget = miRNAtarget)
    res <- c(res,res_seedmatch)

  }

  save(res,file = filename_data)

  return(res)
}
```

Step iii: genes that have a number of **missing values** greater than a chosen threshold are removed and the **log2** transformation on the data is performed

Data processing

removeMissingValues

```
removeMissingValues <- function(data,thr){
  perc_missing_values <- apply(data, 1, function(x){
    length( which(is.na(x)) ) / length(x)
    })
  ind <- which(perc_missing_values > thr)
  if(length(ind) > 0){
    data  <- data[-ind,]
  }
  data <- log2(data + 1)
  return(data)
}
```

Step iii: genes that have a number of **missing values** greater than a chosen threshold are removed and the **log2** transformation on the data is performed

At the end of Module 1, you will obtain the processed data matrices