# miRsift: Sifting through high-throughput sequencing data to identify microRNAs with a regulatory impact

Users Guide

*Stephanie Hilz*

## Chapter 1:

### Introduction

#### 1.1 Scope

miRsift was designed to help biologists identify microRNA (miRNA) regulatory signatures in RNAseq data. miRsift does this by performing multiple linear regression analysis on the relationship between the fold change for each expressed gene in the RNAseq data and the predicted fold change response contribution for each of 107 conserved miRNA families. It has the option of first filtering the RNAseq data, retaining only genes which show post-transcriptional gene regulatory signatures, thus reducing noise in the dataset due solely to transcriptional changes. To help with the intepretation of the miRsift analysis output, it also has the option of integrating the results of small RNA sequencing data into the final output. miRsift, therefore, is a useful tool for determining if a relationship exists between changes in gene expression observed in RNA sequencing data and one or more miRNA families.

#### 1.2 Obtaining miRsift

Currently, miRsift is only available via download from github.

To install directly from github, first make sure that you have installed and loaded the package "devtools" in R:

```
install.packages("devtools")
library("devtools")
```

Then, run the command

```
install_github('miRsift','SRHilz')
```

If this fails to install the package successfully, as an alternative, you can manually download the miRsift package from github (https://github.com/SRHilz), unzip it in your current working directory, and, making sure that you are still in the parent directory that contains the now unzipped miRsift folder, run the command

```
install("miRsift")
```

#### 1.3 Getting Help

Specific usage informaiton for each of the functions can be obtained as an R help page by typing `?functionName` or `help(functionName)` into the R console. For example, for information about the importSeq funciton, type `?importSeq` or `help(importSeq)`.

If you are unable to find an answer for your question using this user guide or the R help pages for functions, you can contact the author of this script, Stephanie Hilz, at sh745@cornell.edu.

### 1.4 Quick Start

While there are many options in miRsift that you can choose to use (or not), here is an example of what a simple analysis that also utilzies optional post-transcriptional regulatory data and small RNAseq data would look like:

```
contextTable <- buildContextTable('Summary_Counts.all_predictions.txt',
  SpeciesID)
rnaseqTable <- importSeq("RNAseq_counts.txt", group, contrast)
smallrnaseqTable <- importSeq("smallRNAseq_counts.txt", group, contrast)
PTList <- importPT("PTFile.txt")
analysis.table <- miRsiftAnalyze(ContextTable, rnaseqTable,
  smallrnaseqTable=smallrnaseqTable,
  PTList=PTList)
```

# Chapter 2:

## The miRsift Approach

### 2.1 Predicting miRNAs with Regulatory Impacts

The basis of miRsift is the usage of a linear model in order to determine the extent of correlation between the changes observed in gene expresison for an RNAseq dataset and the predicted effect of miRNA families on those genes. The observed changes in gene expression for each gene in a dataset are calculated as log2 fold changes (logFC) in miRsift using edgeR. The predicted effect of miRNA families on genes used in miRsift analysis is the TargetScan context score.

In miRsift analysis, a matrix is built which, for each gene in TargetScan and our RNAseq dataset (TargetScan RNAseq), gives the log2FC for that gene in the RNAseq data, as well as the context scores for all miRNA families targeting that gene. A value of 0 is given to all miRNA families that do not target a particular gene. Here is an example of such a matrix for four miRNA families and an extremely subset of genes:

| logFC | AAAGUGC | AACACUG | AACAGUC | AACCGUU |
|---|---|---|---|---|
| 0.40 | 0 | 0 | 0 | 0 |
| 0.25 | 0 | 0 | 0 | 0 |
| -0.17 | -0.03 | 0 | 0 | 0 |
| 0.68 | 0 | -0.18 | -0.05 | 0 |
| 0.22 | 0 | -0.02 | 0 | 0 |
| -1.23 | -0.17 | -0.09 | -0.07 | 0 |

We next perform single linear regression for each miRNA family:

$y_{gene} = \beta_{miR} * x_{gene:miR} + \beta_0$

where $y_{gene}$ is the log2FC (gene-specific response), $x_{gene:miR}$ is the context score (gene-specific predicted response), and $\beta_{miR}$ is the coefficient of the predicted response, which modulates the fold impact of the predicted response and is the value estimated in our regression. For example, in an experiment where levels of a miRNA are unchanged between two conditions, even if is $x_{gene:miR} < 0$ for a particular gene, $\beta_{miR}$ for that miRNA family is expected to equal 0. And in a case where levels of an miRNA family go up, $\beta_{miR}$ is expected to be negative for that miRNA.

After single regression is performed, all miRNA families with a seed that significantly contributes to the model ($n$) are then incorporated into a multiple linear model:

$y_{gene} = \beta_{miR} * x_{gene:miR_1} + \beta_{miR} * x_{gene:miR_2} + [...] + \beta_{miR} * x_{gene:miR_n} + \beta_0$

Those that still contribute significantly after incorporation into the multiple linear model are thus miRNA families whose predicted response correlates with actual responses in RNAseq data, and therefore are candidate regulators of gene expression in the experiment analyzed.

For information on how to perform basic miRsift analysis, including information about data pre-processing, please see "3.1 Example with only RNAseq Data".

### 2.2 Improving miRsift Analysis By Incorporating Post-transcriptional Regulatory Information

One limitation of comparing RNAseq data from two different conditions is that RNAseq measures changes in transcription in addition to post-transcriptional regulation. Even in simple miRNA transfection experiments, there will almost certainly be downstream changes in transcription. When the effects of transcriptional changes are large, they can dwarf the more subtle post-transcriptional changes of miRNAs.

Incorporation of post-transcriptional regulatory analysis into miRsift aims to focus the regression we perform on genes that show a post-transcriptional regulatory signature. Post-transcriptional regulatory analysis is first performed to identify these genes, which we will refer to in this guide as post-transcriptionally (PT)-regulated genes. We then remove all other genes before performing the two-step linear regression analysis described above. Thus, incorporation of post-transcriptional regulatory analysis means the miRsift analysis is only performed on a subset of genes that show a measurable post-transcriptional regulatory signature. A demonstration of the gains of adding post-transcriptional regulatory analysis to miRsift can be found in "3.2 Example incorporating post-transcriptional regulatory analysis". While not required, filtering out genes that do not show a detectable post-transcriptional regulatory signature is highly recommended when using miRsift in order to reduce the number of false positives.

One such method for post-transcriptional regulatory analysis is the EISA (exon intron split read analysis) method, developed by *Gadaitzis et al. 2015*. This method utilizes data already collected in RNAseq, and thus requires no additional datasets to be produced. Briefly, the number of intronic and exonic reads for each gene in an RNAseq dataset are counted, then edgeR is used to determine which genes show a significant difference between their change in intron vs change in exon levels, thus identifying genes whose expression changes do not appear to be solely transcriptional.

As the EISA method requires no additional data other than what is already present in an RNAseq dataset, it is the method we recommend to use with miRsift. However, as miRsift simply requires an input list of genes showing a post-transcriptional regulatory signature, the user is welcome to use whatever methods he or she prefers.

The primary limitation of focusing miRsift analysis on a subset of genes (PT-regulated genes) is that regression will be performed on a smaller dataset, resulting in less significant results. The number of PT-regulated genes, and thus the power of the statistical analysis, is determined by two factors. One, the number of genes actually undergoing post-transcriptional regulation in an experiment, is a product of the biology of that experiment and is out of the scientist's control. However, the other factor that can result in a small number of PT-regulated genes is the number of intron-mapping reads. As the number of intron mapping reads is generally lower than exon-mapping reads in RNAseq data, a problem especially likely when using poly-A selection to focus sequencing on mature mRNA transcripts, many genes may be "thrown out" simply because they do not have sufficient intron coverage to perform EISA analysis. Additionally, usage of unstranded libraries can lead to additional reads from overlapping genes being thrown out doing to ambiguity as to whether they derive from an exon on one strand, or an intron on the other. Luckily, by sequencing at a greater depth, performing rRNA depletion rather than poly-A selection, and finally, performing stranded library preparation, the number of genes that have a suitable number of intron and exon counts for EISA analysis can be maximized.

### 2.3 Synthesizing miRsift Analysis with Small RNAseq Data

Small RNAseq data is not required for miRsift analysis. However, if an interesting miRNA regulatory signature is identified in an RNAseq data set, small RNA sequencing might be the next step in better understanding what miRNAs are playing important regulatory roles. As synthesizing this data with RNAseq

data can be challenging, miRsift offers the option to incorporate small RNAseq data into the miRsift output, so that you can quickly check if miRNA family expression is inversely correlated to target expression, as would be expected, for a miRNA family with a signficant miRNA regulatory signature. For certain applications, it might be desirable to limit miRsift analysis to those miRNA families that are highly expressed based on small RNAseq analysis, and thus we have also added this an additional option. For more information on incorporating small RNAseq into miRsift analysis, see "3.3 Example incorporating post-transcriptional regulatory analysis and small RNA sequencing data".

# Chapter 3:

## Using miRsift

### 3.1 Example with only RNAseq Data

This example will provide you with an understanding of how to perform miRsift analysis on an RNAseq dataset. It will also serve as a building block for the example analyses in 3.2 and 3.3. The data used in this example is a set of transfection experiments from *Guo et al. 2010*, a publication from the Bartel lab, in which miR-1, miR-155, or a control mock miRNA were transfected into HeLa cells and RNA was harvested 32 hours later for RNAseq. Thus, in this experiment, we expect to see a clear regulatory impact for miR-1 or miR-155, respectively, for the miR-1 and miR-155 transfections.

*Input Files*

The first step is to obtain all necessary input files. For a basic miRsift analysis that will not incorporate post-transcriptional regulatory information or small RNAseq data, the required input files are:

- TargetScan Summary Counts File (downloadable from targetscan.org)
- RNAseq data as raw counts in tab-delim format (user-supplied)

The TargetScan Summary Counts File is a file that can be downloaded from http://www.targetscan.org/ under "Download data or code" after choosing the appropriate version (i.e. TargetScanHuman, TargetScanMouse, etc) for your species. Specifically, the file required is Summary_Counts.all_predictions.txt, which is referred to as Summary Counts, all predictions on the website. While only a limited number of species have their own Summary Counts files, TargetScan can still be useful for the prediction of conserved target sites in other species (more info at http://www.targetscan.org/faqs.html #4). See http://www.targetscan.org/vert_70/docs/species100.html for a complete list of supported species.

The RNAseq data file is provided by the user. It should be in tab-delim format and contain a header line. The first column should be the name of each gene in gene symbol format. All other columns should be raw, unnormalized counts in integer format. Such a file is typically produced by first performing quality analysis on raw RNAseq data, then aligning reads to the genome with a tool such as Bowtie or BWA, and then finally counting the number of reads that align to genomic features (in our case, genes) using a tool such as HTSeq or TopHat. Here are the first few lines of our example RNAseq data file:

| gene | mockN1 | mockN2 | mockN3 | miR1N1 | miR1N2 | miR1N3 | miR155N1 | miR155N2 | miR155N3 |
|---|---|---|---|---|---|---|---|---|---|
| MTVR2 | 0 | 0 | 0 | 0 | | 0 | 2 | 0 | 0 |
| CHMP1B | 96 | 113 | 26 | 129 | 136 | 23 | 120 | 135 | 30 |
| LOC441204 | 0 | 0 | 0 | 0 | | 0 | 2 | 0 | 0 |
| TCOF1 | 375 | 484 | 126 | 669 | 637 | 136 | 565 | 556 | 132 |
| NSRP1 | 68 | 73 | 26 | 151 | 138 | 30 | 116 | 139 | 33 |

*Building the Context Table*

The next step is to create a table of context scores, which will be imported from the TargetScan Summary Counts file, organized by gene and miRNA family. Ensure that you have properly loaded the miRsift package (see "1.2 Obtaining miRsift" if you haven't already done this). The miRsift command to create a table of context scores is:

```
buildContextTable(SummaryCountsFile, SpeciesID)
```

This function reads in TargetScan's Summary Counts file, first filtering out all information not specific to your species of interest, as well as entries for miRNA families that are poorly conserved. Finally, it is restructured to prepare for miRsift regression analysis. The paramters of this function are the path to & name of the TargetScan Summary Counts file downloaded from TargetScan, and a Species ID. The species ID is a number that uniquely identifies a particular species, and is required to predict the appropriate miRNA targets for your species. See http://www.targetscan.org/vert_70/docs/species100.html for a complete list of supported species and their speciesIDs.

Running this function can take some time, as it requires a large file to be read into memory. To build a `contextTable` for the example dataset, which is from cultured human cells, we download the 'Summary_Counts.all_predictions.txt' file from TargetScan Human, and run the following command:

```
contextTable <- buildContextTable('Summary_Counts.all_predictions.txt', 9606)
```

If we wanted to build a `contextTable` for a mouse dataset, we would alternatively download the 'Summary_Counts.all_predictions.txt' file from TargetScan Mouse, and run the following command:

```
contextTable <- buildContextTable('Summary_Counts.all_predictions.txt', 10090)
```

And for a `contextTable` for a non-human or mouse dataset, such as cow, we would download the 'Summary_Counts.all_predictions.txt' file from TargetScan Human, and run the following command (in this case with the cow species ID):

```
contextTable <- buildContextTable('Summary_Counts.all_predictions.txt', 9913)
```

The first few lines and columns of `contextTable` for our example look like this:

| logFC | AAAGUGC | AACACUG | AACAGUC | AACCGUU |
|---|---|---|---|---|
| 0.40 | 0 | 0 | 0 | 0 |
| 0.25 | 0 | 0 | 0 | 0 |
| -0.17 | -0.03 | 0 | 0 | 0 |
| 0.68 | 0 | -0.18 | -0.05 | 0 |
| 0.22 | 0 | -0.02 | 0 | 0 |

Note that once created, a `contextTable` can be used for multiple analysis, so long as they are all within the same species.

*Importing RNAseq Data*

We next import our RNAseq data with the following command:

```
importSeq(seqFile, group, contrast, minDepth = 6)
```

This function reads in sequencing data from a file provided by the user, calculating the log2 fold change values and determining differential expression of expressed genes with edgeR for a given comparison. The seqFile is the RNAseq data discussed in 'Input Files'. The group is a vector which denotes sample and replicate information for the input RNAseq data file. Contrast is also a vector, and it specifies the pairwise comparison to be made. Finally, minDepth is an optional parameter which allows for the adjustment of the minimum read depth cutoff. If not provided, the default value of 6 will be used.

For our data, we run the `importSeq()` command with the following inputs:

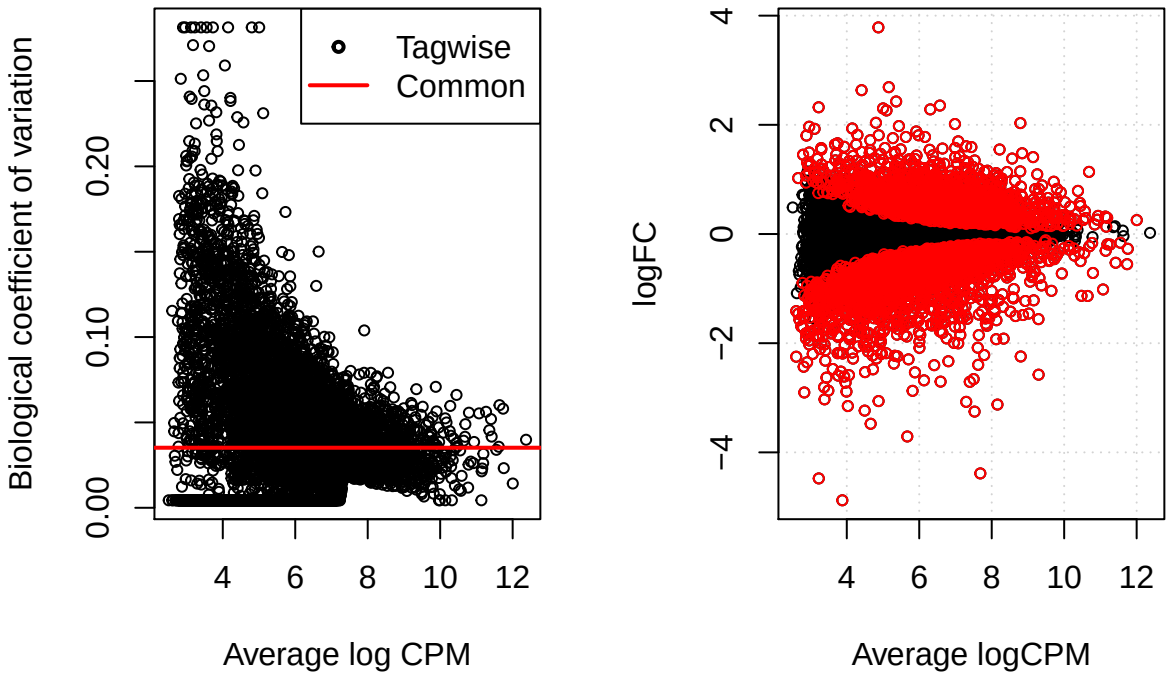
```
rnaseqTable_miR1 <- importSeq('Guo2010_merged_counts.txt', c(1,1,1,2,2,2,3,3,3), c(1,2))
```

Our parameter input for `group` specifies that, in our input RNAseq file, we have three different conditions (1:mock [first three columns of input file], 2:miR-1 [next three columns of input file], 3:miR-155 [final three columns of input file]), and three replicates for each condition. The pattern of numbers in `group` corresponds directly to the input RNAseq file raw count data columns.

Our parameter input for `contrast` specifies that we want to compare sample groups 1 vs 2, or mock vs miR-1.

In addition to calculating the log2 fold change for each gene, importSeq also produces two plots using edgeR which can help set an optimal cutoff (BCV plot, left panel) and visualize the overall change in gene expression between the two samples (MA plot, right panel). It also provides the square root of the common dispersion as calculated in edgeR, which gives an idea of the overall biological variance. Here is an example of those plots and the square root of the common dispersion for our example dataset:

```
## [1] "Square root of common dispersion:  0.035"
```




*Performing miRsift Analysis*

We now have the necessary input for miRsift analysis. This is conducted with the following command:

```
miRsiftAnalyze(contextTable, rnaseqTable, ...)
```

where `contextTable` is the output of the miRsift `buildContextTable` function, and `rnaseqTable` is the output of the miRsift `importSeq` function for RNAseq data. `...` are optional parameters which we will explore in 3.2 and 3.3 of this user's guide.

For our data, the command and its printed output look like this:

```
analysisTable_miR1 <- miRsiftAnalyze(contextTable, rnaseqTable_miR1)
```

```
## [1] "94 % of expressed genes in your dataset are predicted to be miRNA targets and will be used for r
## [1] "Performing regression on matrix of dimensions: 9727(genes) x 108(miRNA families)"
```

Looking at the first few lines of the output table, we have:

```
head(analysisTable_miR1)
```

```
##         miRfamily.logCPM miRfamily.logFC    slr.pvalue       slr.FDR
## GGAAUGU               NA              NA 3.160409e-234 3.381638e-232
## AGCUUAU               NA              NA  1.002893e-07  2.146192e-06
## AAAGUGC               NA              NA  3.678448e-09  1.311980e-07
## GAGGUAG               NA              NA  8.854016e-07  1.184225e-05
## GGAGUGU               NA              NA  3.167213e-03  1.255155e-02
## GUGCAAA               NA              NA  2.067755e-07  3.160711e-06
##         mlr.coefficient    mlr.pvalue       mlr.FDR
## GGAAUGU       2.4793893 6.411876e-276 2.564751e-274
## AGCUUAU      -0.4579558  3.769368e-06  5.025824e-05
## AAAGUGC      -0.3433725  5.962959e-06  5.962959e-05
## GAGGUAG      -0.1887281  3.633884e-04  2.026276e-03
## GGAGUGU       0.3048908  4.030569e-04  2.026276e-03
## GUGCAAA      -0.2048289  4.052553e-04  2.026276e-03
```

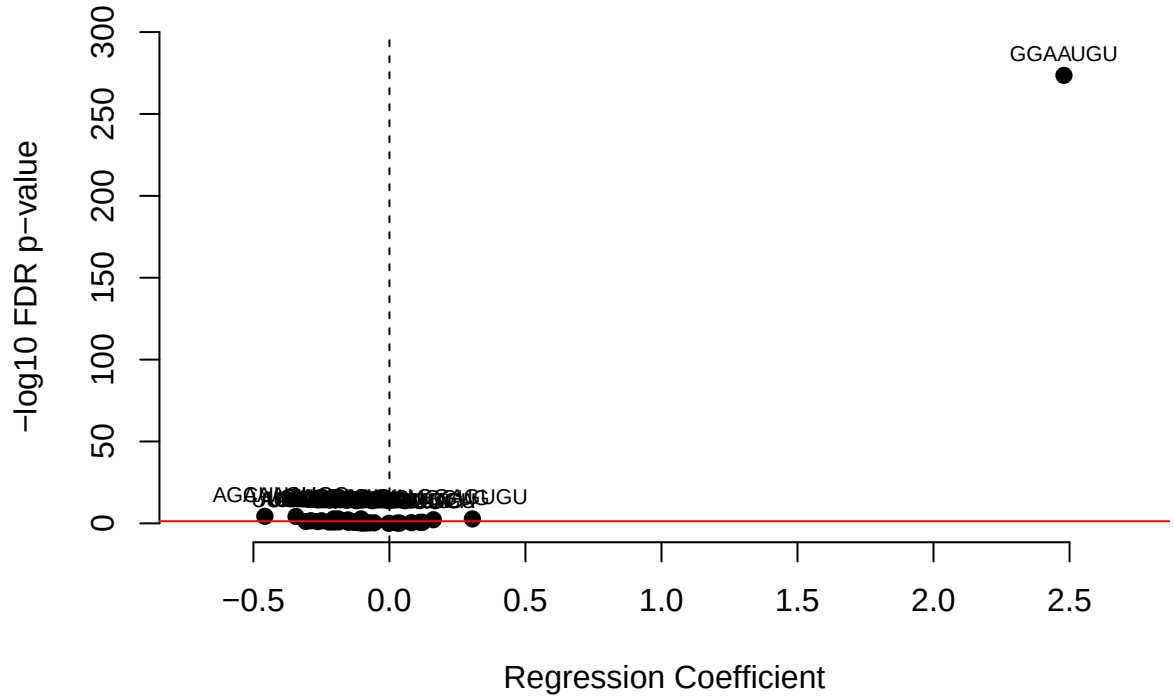
The output of this analysis produces a table with the following information for each miRNA family analyzed: `miRfamily.logCPM` is the expression level of the miRNA family. It will only be available if you have provided small RNA sequencing data (please see 3.3). `miRfamily.logFC` is the log2FC of the miRNA family. It will only be available if you have provided small RNA sequencing data (please see 3.3). `slr.pvalue` is the significance of the single linear regression performed for that specific miRNA family. `slr.FDR` is the multiple test-corrected significance of the single linear regression performed for that specific miRNA family. miRNA families with slr.FDR $< 0.05$ were selected for incorporation into the multiple linear model. `mlr.coefficient` is the value of the coefficient determined for the miRNA family in the multiple linear model. Negative values indicate derepression of miRNA targets, suggesting that levels of the miRNA family are decreasing. Positive values indicate repression, indicating an increase in that miRNA family. `mlr.pvalue` is the significance of the multiple linear regression performed for that specific miRNA family. `mlr.FDR` is the multiple test-corrected significance of the multiple linear regression performed for that specific miRNA family. miRNA families with slr.FDR $< 0.05$ are considred to be those miRNA families whose targets show a significan miRNA targeting signature.

We can see that one miRNA family, miR-1 (GGAAUGU), is very significant. This is expected, as it was the experimentally manipulated miRNA. Many other miRNA families also appear significant, which could be due to disruption of endogenous miRNA targeting, downstream effects of miR-1 upregulation on the expresison of other miRNAs, or false positives.

The name of the miRNA family can be looked up for each seed at http://www.targetscan.org/cgi-bin/targetscan/vert_71/mirna_families.cgi?db=vert_71&species=Human

We can use the `plotAnalysis` function to more easily visualize the miRsift results for our data:

```
plotAnalysis(analysisTable_miR1)
```

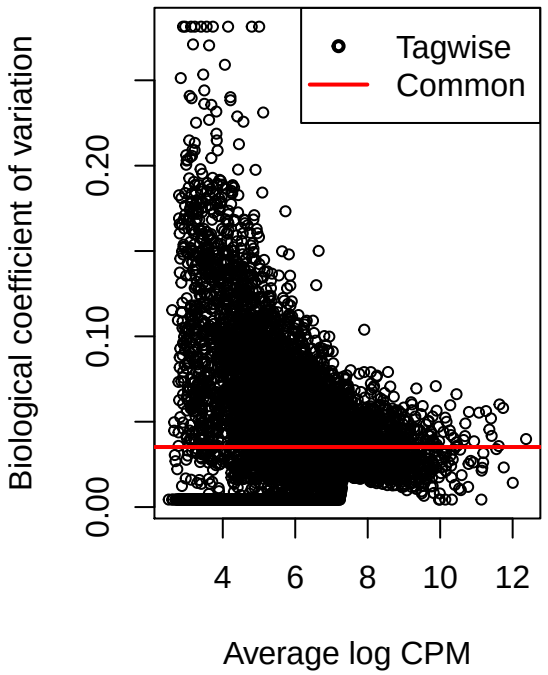


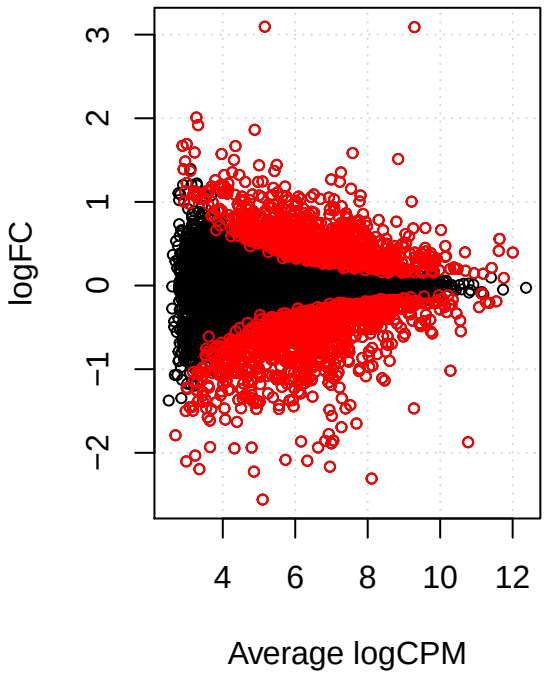

The black dotted line denotes the null hypothesis (regression coefficient equals zero), and the red line is the significance threshold ($p = 0.05$).

And finally, we can repeat the analysis again for the other transfected miRNA in our example dataset, miR-155:

```
rnaseqTable_miR155 <- importSeq('Guo2010_merged_counts.txt', c(1,1,1,2,2,2,3,3,3), c(1,3))
```

```
## [1] "Square root of common dispersion:  0.035"
```

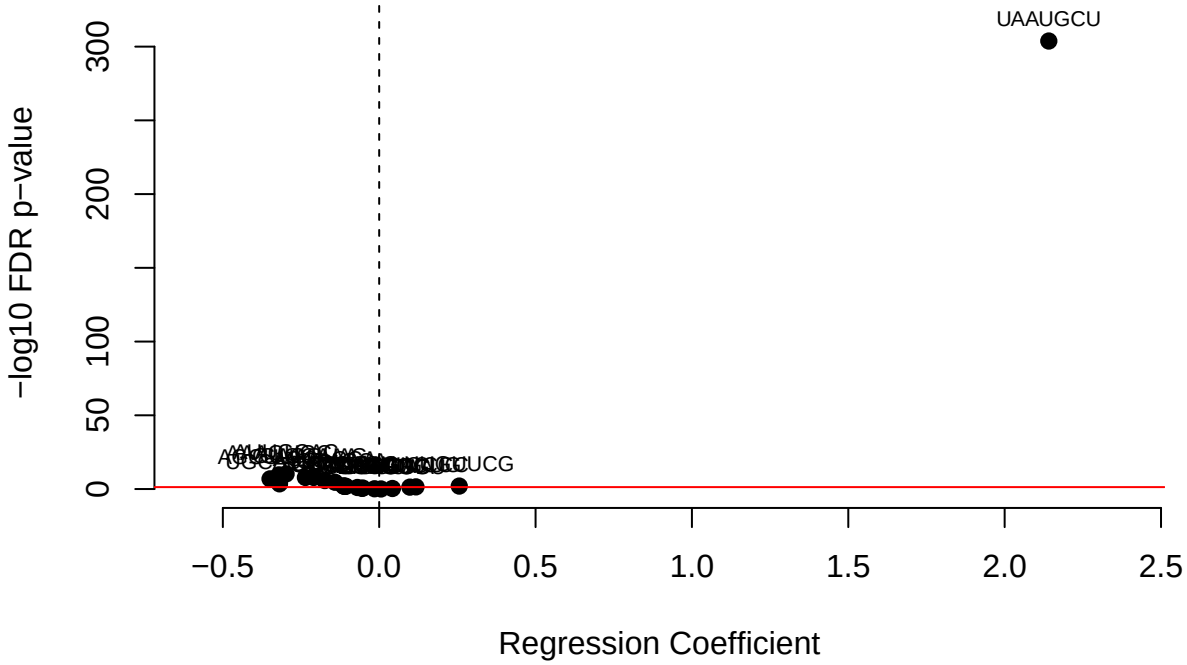



```
analysisTable_miR155 <- miRsiftAnalyze(contextTable, rnaseqTable_miR155)
```

```
## [1] "94 % of expressed genes in your dataset are predicted to be miRNA targets and will be used for r
## [1] "Performing regression on matrix of dimensions: 9727(genes) x 108(miRNA families)"
```

```
plotAnalysis(analysisTable_miR155)
```




miR-155 (UAAUGCU) appears as the most significant miRNA regulatory signature in this data set.

*Section Command Summary from Start to Finish*

```
contextTable <- buildContextTable(SummaryCountsFile, SpeciesID) #build context table
rnaseqTable <- importSeq(seqFile, group, contrast, minDepth = 6) #import RNAseq data
analysisTable <- miRsiftAnalyze(contextTable, rnaseqTable) #perform analysis
plotAnalysis(analysisTable) #visualize results
```

### 3.2 Example incorporating post-transcriptional regulatory analysis

*Incorporating post-transcriptional regulatory analysis*

In this section, we will demonstrate how to add a filter to `miRsiftAnalyze` in order to restrict the analysis to only genes which show evidence of post-transcriptional regulation, thus decreasing the noise in the dataset due to changes that are solely transcriptional. We will use the same dataset as in 3.1.

The first step is to create a tab-delimited file, which we will refer to as PT file, containing a list of genes whose expression changes post-transcriptionally. As mentioned before, one way to do this is using the EISA method (Gaidatzis et al. 2015). The PT file should contain no header, and have one gene symbol per line. Importantly, you would need a seperate PT file for each comparison you wish to do.

Briefly, to determine post-transcriptionally altered genes from the Guo et al. 2015 data, we used BWA for genome alignment, then counted the number of reads aligning to introns, as well as exons using custom scripts and the hg19 iGenomes genes.gtf file. We then input this data into the EISA analysis pipeline, which can be downloaded from the supplementary materials of Gaidatzis et al. 2015.

To import our PT file, first for miR-1, we use the following miRsift command:

```
PTList <- importPT('miR1eisa.txt')
```

We then repeat miRsift analysis, this time only on those genes which show evidence of post-transcriptional regulation:

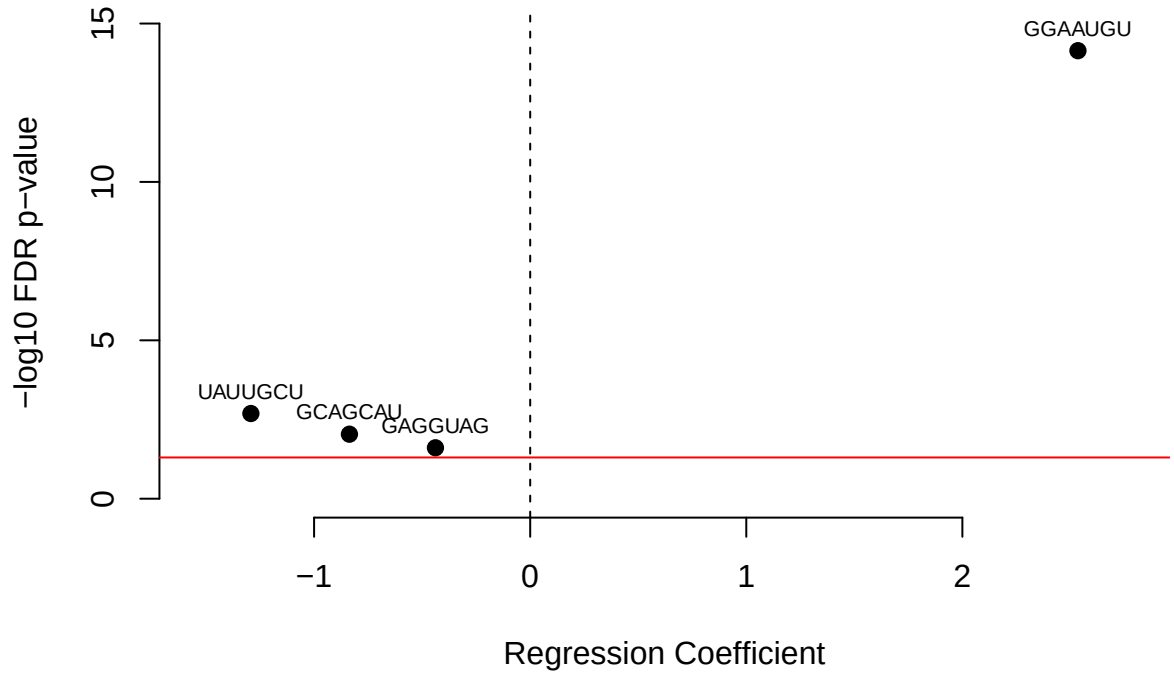
```
analysisTable_miR1 <- miRsiftAnalyze(contextTable, rnaseqTable_miR1, PTList=PTList)
```

```
## [1] "94 % of expressed genes in your dataset are predicted to be miRNA targets and will be used for r
## [1] "Performing regression on matrix of dimensions: 281(genes) x 108(miRNA families)"
```

```
head(analysisTable_miR1)
```

```
##         miRfamily.logCPM miRfamily.logFC   slr.pvalue      slr.FDR
## GGAAUGU               NA              NA 3.433657e-16 3.674013e-14
## UAUUGCU               NA              NA 2.598662e-04 1.390284e-02
## GCAGCAU               NA              NA 5.606597e-04 1.999686e-02
## GAGGUAG               NA              NA 1.602462e-03 4.286587e-02
## AAAGUGC               NA              NA 1.862505e-02 1.720710e-01
## AACACUG               NA              NA 2.932491e-01 6.275530e-01
##         mlr.coefficient   mlr.pvalue      mlr.FDR
## GGAAUGU       2.5348372 1.437502e-15 7.187511e-15
## UAUUGCU      -1.2925404 1.229149e-03 2.048582e-03
## GCAGCAU      -0.8365396 7.364488e-03 9.205610e-03
## GAGGUAG      -0.4383605 2.471688e-02 2.471688e-02
## AAAGUGC              NA           NA           NA
## AACACUG              NA           NA           NA
```

```
plotAnalysis(analysisTable_miR1)
```




We can see that the number of miRNA families that appear significant are now reduced, though the signal from miR-1 is still strong.

Repeating this then for miR-155:

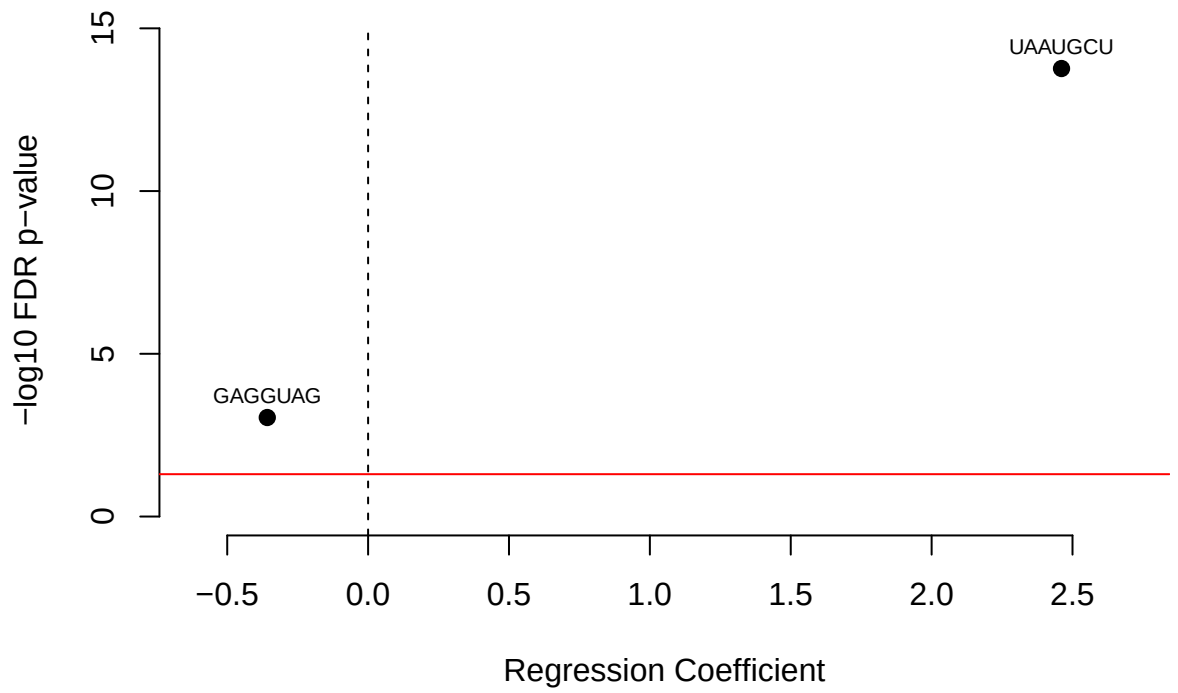
```
analysisTable_miR155 <- miRsiftAnalyze(contextTable, rnaseqTable_miR155, PTList=PTList)
```

```
## [1] "94 % of expressed genes in your dataset are predicted to be miRNA targets and will be used for r
## [1] "Performing regression on matrix of dimensions: 281(genes) x 108(miRNA families)"
```

```
head(analysisTable_miR155)
```

```
##         miRfamily.logCPM miRfamily.logFC   slr.pvalue      slr.FDR
## UAAUGCU               NA              NA 7.165638e-15 7.667232e-13
## GAGGUAG               NA              NA 8.443190e-04 4.517107e-02
## AAAGUGC               NA              NA 5.620962e-03 1.208829e-01
## AACACUG               NA              NA 8.565511e-01 9.448554e-01
## AACAGUC               NA              NA 1.177725e-01 5.891112e-01
## AACCGUU               NA              NA 5.743747e-01 9.054484e-01
##         mlr.coefficient   mlr.pvalue      mlr.FDR
## UAAUGCU       2.4609693 5.724558e-15 1.717367e-14
## GAGGUAG      -0.3578665 6.049859e-04 9.074789e-04
## AAAGUGC              NA           NA           NA
## AACACUG              NA           NA           NA
## AACAGUC              NA           NA           NA
## AACCGUU              NA           NA           NA
```

```
plotAnalysis(analysisTable_miR155)
```




We see the same, with miR-155 and let-7 now being the only miRNA families to appear significant, with miR-155 targting increasing, and let-7, a highly expressed miRNA family in HeLa cells, showing disrupted targeting.

*Gains of incorporating post-transcriptional regulatory information*

To explore whether incorporation of post-transcriptional regulatory information increases the signature of miRNA-specific targeting, we have included in miRsift the following function:

```
miRsiftSimAnalyze(contextTable, rnaseqTable, subsetSize, reps, ...)
```

This function draws random samples of expressed genes from our RNAseq data of size equal to `subsetSize`, performing a miRsift analysis on each of these subsets. Importantly, the `subsetSize` should be equivalent to the number of post-transcriptionally regulated genes analyzed using miRsift. In this way, we can determine if the observed p-value for a miRsift analysis using a list of genes with a post-transcriptional regulatory signature is more significant than we would expect using randomly selected genes of the same sample size.

The value for `subsetSize` can be quickly found in the printed output of the `miRsiftAnalyze` function, in the statement specifying the dimensions of the matrix miRsift analysis was performed on.

The parameter `reps` specifies the number of random samples we want to perform miRsift analysis on. 10,000 is recommended. Importantly, this function can take a very long time to run, proportional to the size of `reps`. `contextTable` is again the output of the miRsift `buildContextTable` function, and `rnaseqTable` the output of the miRsift `importSeq` function for RNAseq data.

```
simAnalysisTable_miR1 <- miRsiftSimAnalyze(contextTable, rnaseqTable_miR1, 281, 10000)
```

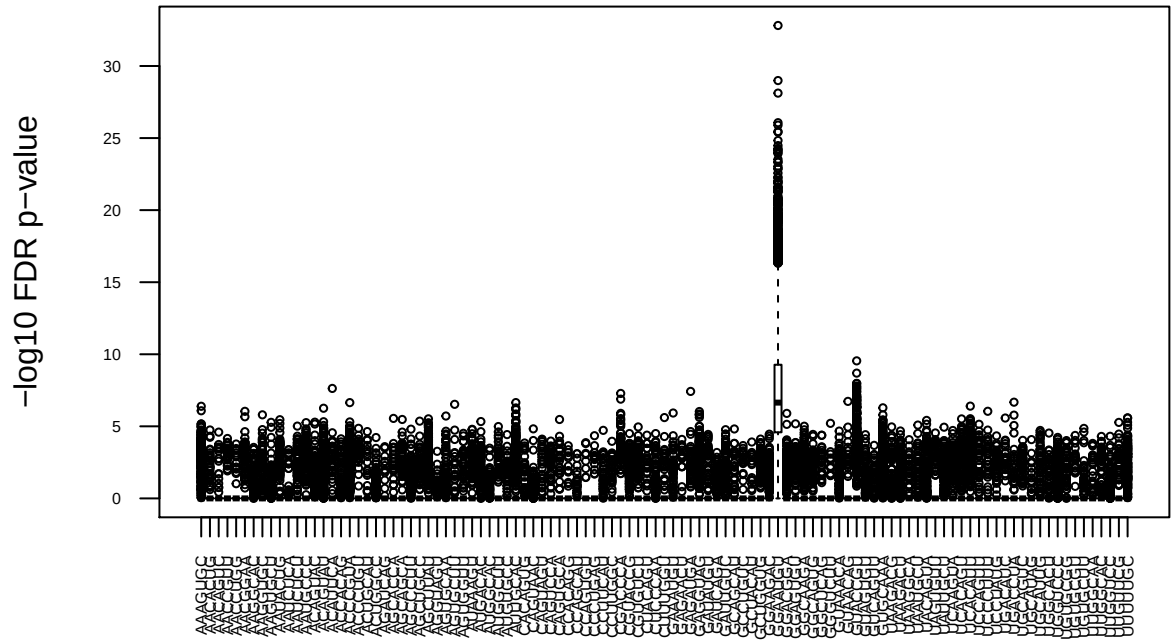
```
simAnalysisTable_miR1[1:5,1:5]
```

```
##         mlr.FDR.1    mlr.FDR.2    mlr.FDR.3    mlr.FDR.4    mlr.FDR.5
```

```
## AGCUGCC          1 4.314826e-02 1.000000e+00 1.000000e+00 1.000000e+00
## GGAAUGU          1 4.051250e-13 3.336818e-10 1.265647e-05 2.374072e-05
## UCACAUU          1 1.000000e+00 1.000000e+00 1.000000e+00 1.000000e+00
## AAUGCCC          1 1.000000e+00 1.000000e+00 1.000000e+00 1.000000e+00
## GUACCGU          1 1.000000e+00 1.000000e+00 1.000000e+00 1.000000e+00
```

We can better visualize the output of our simulated analysis with miRsift's `plotSimAnalysis` function:

```
plotSimAnalysis(simAnalysisTable_miR1)
```




We can then use another miRsift function, `analysisProb`, in order to determine the probability that a particular miRNA family would have as significant of a p-value when performing miRfist on only genes with an identifiable post-transcriptionally regulatory signature as compared to miRsift analysis on randomly selected genes.

```
analysisProb(simAnalysisTable, analysisTable, miRNAFamily)
```

This function requires as input both the output of the `miRsiftAnalysis` and `miRsiftSimAnalysis` functions, as well as the 7 nt "seed" sequence of the miRNA family we want to check.

For miR-1,

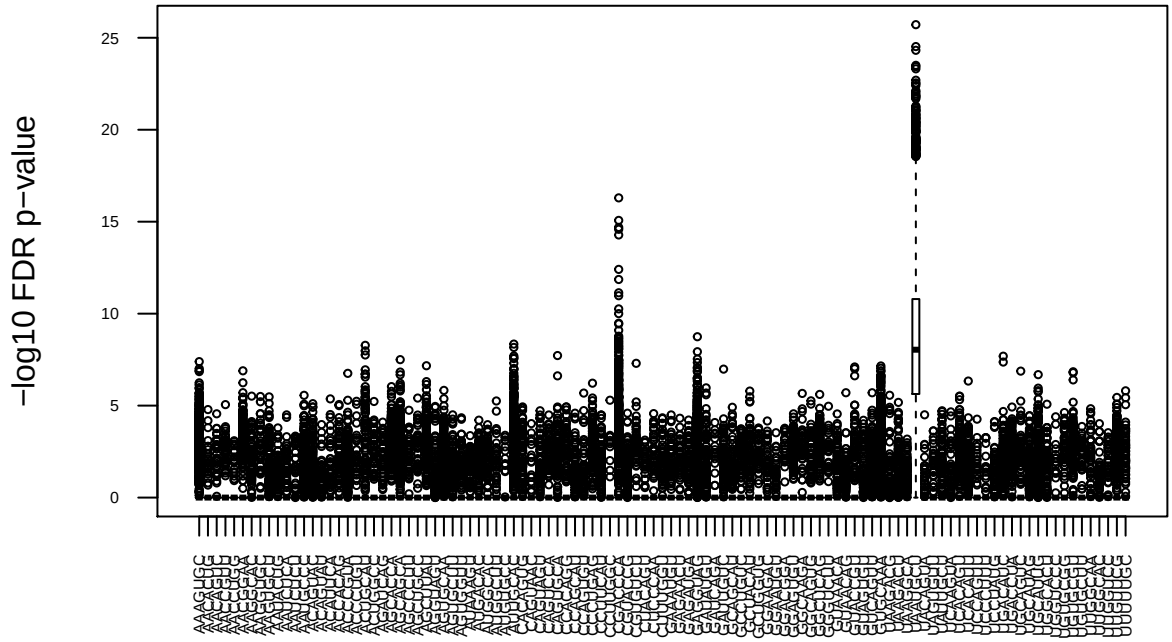
```
analysisProb(simAnalysisTable_miR1, analysisTable_miR1, "GGAAUGU")
```

```
## [1] 0.0513
```

We can then repeat the simulation and analysis for the miR-155 transfection data set:

```
simAnalysisTable_miR155 <- miRsiftSimAnalyze(contextTable, rnaseqTable_miR155, 281, 10000)
```

```
plotSimAnalysis(simAnalysisTable_miR155)
```




```
analysisProb(simAnalysisTable_miR155, analysisTable_miR155, "UAAUGCU")
```

```
## [1] 0.0899
```

*Section Command Summary from Start to Finish*

```
contextTable <- buildContextTable(SummaryCountsFile, SpeciesID) #build context table
rnaseqTable <- importSeq(seqFile, group, contrast, minDepth = 6) #import RNAseq data
analysisTable <- miRsiftAnalyze(contextTable, rnaseqTable, PTList=<PTList>) #perform analysis with PTLis
plotAnalysis(analysisTable) #visualize results
simAnalysisTable <- miRsiftSimAnalyze(contextTable, rnaseqTable, subsetSize, reps) #perform simulated an
plotSimAnalysis(simAnalysisTable) #visualize results of simulated analysis
analysisProb(simAnalysisTable, analysisTable, miRNAFamily) #calculate probabilites for miRNAs of interes
```

### 3.3 Example incorporating post-transcriptional regulatory analysis and small RNA sequencing data

*Incorporating small RNAseq data*

In this final section, we will show how small RNAseq data can be added in order to enhance the intepretation of a miRsift analysis.

For this demonstration, we will use a different data set from that used in 3.1 and 3.2. This data set compares changes in mRNA and miRNA family levels in five stages of male germ cell development in mice. We will specifically be looking at differences between the first and last stage, which occur before, and after, male germ cells enter meiosis.

To incorporate small RNAseq data, we first need to import it. This is done with the same function used to import RNAseq data, `importSeq`.

Like the RNAseq data, it is important that the small RNAseq data be in a specific format. It should be in tab-delim format and contain a header line. The first column should be the 7 nucleotide "seed" sequence

for each miRNA family. All other columns should be raw, unnormalized counts in integer format. This file can be easily produced by first aligning small RNAseq reads to hairpins, then collapsing the hairpin-aligning reads into each family, summing up the counts for all members of a family to give the counts for that family. Alternatively, the tool mirdeep2 can be used to obtain raw counts for individual miRNAs, and these can then be collapsed by miRfamily.

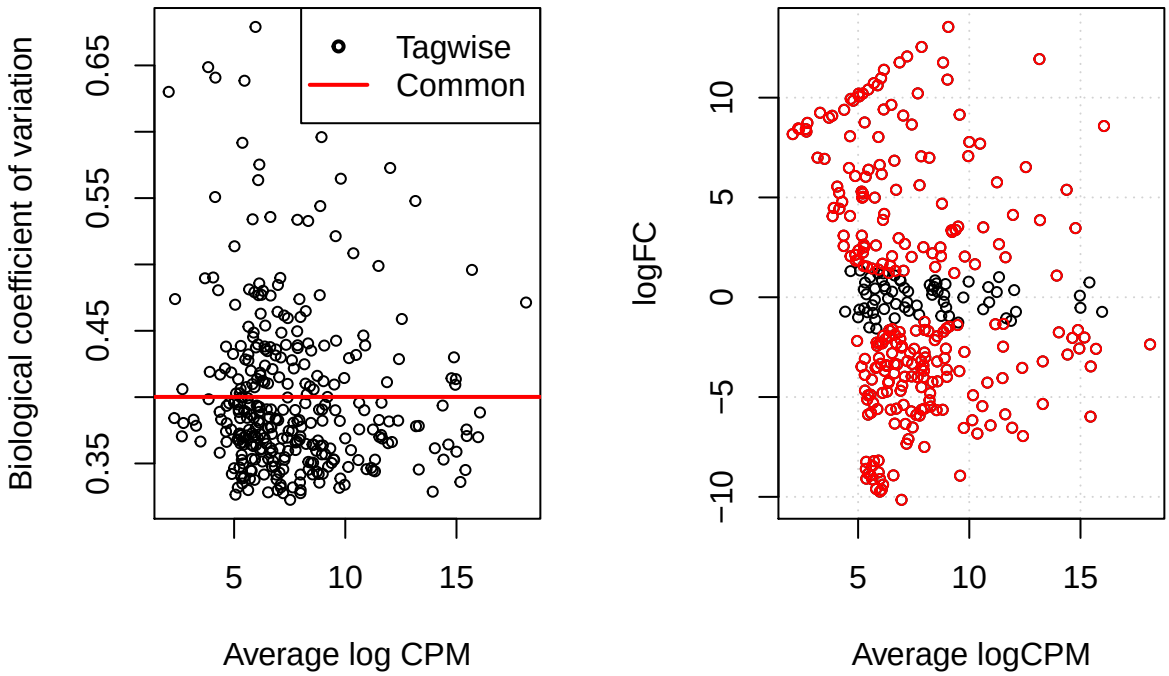
Here is an example of an appropriately formatted input file from the example dataset:

| seed | D1n1 | D1n2 | D3n1 | D3n2 | D7n1 | D7n2 | LZn1 | LZn2 | Pachn1 | Pachn2 |
|---|---|---|---|---|---|---|---|---|---|---|
| ACAGUAG | 3479 | 4448 | 20662 | 17328 | 6956 | 15230 | 94 | 14 | 23 | 12 |
| UAUAAAG | 3036 | 2266 | 10660 | 13676 | 7229 | 20333 | 1715 | 377 | 228 | 408 |
| CCUGUAC | 2947 | 6672 | 27335 | 13252 | 4862 | 5951 | 2222 | 1412 | 109 | 23 |
| AUCAUCG | 2891 | 2757 | 19080 | 20101 | 5009 | 14643 | 55 | 4 | 0 | 7 |
| CCACCGA | 2673 | 3553 | 9337 | 7323 | 3186 | 6549 | 10 | 5 | 11 | 3 |

We can then import this data with the `importSeq` function, once again specifying the group information and comparison to be made:

```
smallrnaseqTable <- importSeq('meiotic_smallrnaseq.txt', c(1,1,2,2,3,3,4,4,5,5), c(1,5))
```

```
## [1] "Square root of common dispersion:  0.4"
```



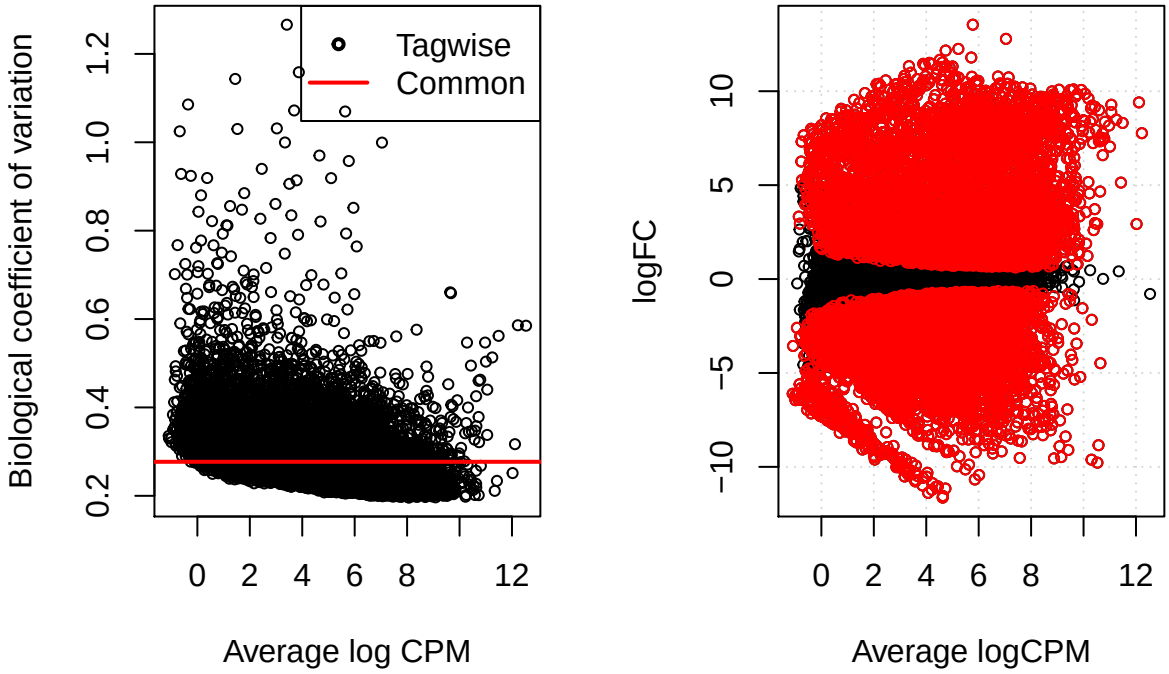

Before we then incorporate this data into the results, we first need to quickly create our `contextTable` and `rnaseqTable` inputs for this new dataset. As this is a mouse data set, we first need to create a new `contextTable` specific for mouse:

```
contextTable <- buildContextTable('Summary_Counts.all_predictions.txt', 10090)
```

We then import our RNAseq data, which is in the same format as the small RNAseq data we imported:

```
rnaseqTable <- importSeq('meiotic_rnaseq.txt', c(1,1,2,2,3,3,4,4,5,5), c(1,5))
```

```
## [1] "Square root of common dispersion:  0.277"
```




And finally, we import our list of post-transcriptionally regulated genes:

```
PTList <- importPT("eisa.txt")
```

We then perform the analysis, this time including the optional `smallrnaseqTable` parameter:

```
analysisTable <- miRsiftAnalyze(contextTable, rnaseqTable, PTList=PTList, smallrnaseqTable=smallrnaseqTa
```

```
## [1] "89 % of expressed genes in your dataset are predicted to be miRNA targets and will be used for r
## [1] "Performing regression on matrix of dimensions: 3129(genes) x 108(miRNA families)"
```

```
head(analysisTable)
```

```
##         miRfamily.logCPM miRfamily.logFC   slr.pvalue      slr.FDR
## AGCACCA         8.770723       4.6868456 6.839045e-11 3.658889e-09
## GGCAGUG        16.057200       8.5783545 1.272278e-12 1.361337e-10
## CCUUGGC               NA              NA 2.099720e-06 5.616752e-05
## GAUUGUC               NA              NA 5.059826e-06 6.804116e-05
## AAUGCCC         6.173068      -0.5921950 2.886777e-06 6.177702e-05
## GCUGGUG         5.762437       0.4350121 5.087190e-06 6.804116e-05
##         mlr.coefficient   mlr.pvalue      mlr.FDR
## AGCACCA        2.204554 6.605716e-06 0.0001892173
## GGCAGUG        1.981255 8.347822e-06 0.0001892173
## CCUUGGC        1.876823 4.789281e-03 0.0651342151
## GAUUGUC        2.445527 4.262196e-03 0.0651342151
## AAUGCCC        2.156446 1.013813e-02 0.1148988271
## GCUGGUG        1.174245 1.741455e-02 0.1691699453
```

You can now see that the output includes not only information about targeting signatures for indivdual miRNA families, but also expression and differential expression information as well.

*Filtering by miRNA expression* This option is only recommended if you would like to focus your analysis on miRNA targeting signatures to only those miRNA which are highly expressed, based on small RNA sequencing data.

To use this option, in addition to including `smallrnaseqTable`, an optional variable `smallrnaseqMinExp` is also included in the `miRsiftAnalyze` call. This optional variable sets the minimum expression required for a miRNA family to be included in the analysis. For our example, we will set this optional variable equal to 100:

```
analysisTable <- miRsiftAnalyze(contextTable, rnaseqTable, PTList=PTList, smallrnaseqTable=smallrnaseqTa
```

```
## [1] "89 % of expressed genes in your dataset are predicted to be miRNA targets and will be used for r
## [1] "Performing regression on matrix of dimensions: 3129(genes) x 108(miRNA families)"
```

```
head(analysisTable)
```

```
##         miRfamily.logCPM miRfamily.logFC   slr.pvalue      slr.FDR
## AGCACCA         8.770723       4.6868456 6.839045e-11 3.658889e-09
## GGCAGUG        16.057200       8.5783545 1.272278e-12 1.361337e-10
## CCUUGGC               NA              NA 2.099720e-06 5.616752e-05
## GAUUGUC               NA              NA 5.059826e-06 6.804116e-05
## AAUGCCC         6.173068      -0.5921950 2.886777e-06 6.177702e-05
## GCUGGUG         5.762437       0.4350121 5.087190e-06 6.804116e-05
##         mlr.coefficient   mlr.pvalue      mlr.FDR
## AGCACCA        2.204554 6.605716e-06 0.0001892173
## GGCAGUG        1.981255 8.347822e-06 0.0001892173
## CCUUGGC        1.876823 4.789281e-03 0.0651342151
## GAUUGUC        2.445527 4.262196e-03 0.0651342151
## AAUGCCC        2.156446 1.013813e-02 0.1148988271
## GCUGGUG        1.174245 1.741455e-02 0.1691699453
```

*Section Command Summary from Start to Finish*

```
contextTable <- buildContextTable(SummaryCountsFile, SpeciesID) #build context table
rnaseqTable <- importSeq(rnaseqFile, group, contrast, minDepth = 6) #import RNAseq data
smallrnaseqTable <- importSeq(smallrnaseqFile, group, contrast, minDepth = 6) #import RNAseq data
analysisTable <- miRsiftAnalyze(contextTable, rnaseqTable, PTList=<PTList>, smallrnaseqTable=<smallrnase
plotAnalysis(analysisTable) #visualize results
simAnalysisTable <- miRsiftSimAnalyze(contextTable, rnaseqTable, subsetSize, reps) #perform simulated an
plotSimAnalysis(simAnalysisTable) #visualize results of simulated analysis
analysisProb(simAnalysisTable, analysisTable, miRNAFamily) #calculate probabilites for miRNAs of interes
```

# References

Will fill in later.