Users' Guide
MORE

Sonia Tarazona Campos
Maider Aguerralde Martin
Blanca Tomás Riquelme

July 14, 2025

Contents

1 Introduction

One of the most common questions to be addressed when performing a multi-omics experiment is how the levels of given biological entities are being regulated by other biological entities under certain conditions. An example of this type of study would be understanding the regulatory mechanisms behind the changes in gene expression.

Potential regulators of a given gene such as miRNAs, transcription factors (TF), methylation sites, etc., can be either retrieved or predicted from public databases or obtained by a combination of experimental and computational procedures. However, a methodology for selecting the specific regulators of a particular biological system studied under certain conditions is required. This is the goal of the MORE (Multi-Omics REgulation) method: modeling a target omic data expression (such as gene, protein, metabolite, ... expression) as a function of experimental variables, such as diseases or treatments, and the potential regulators of it. The idea is to obtain more specific candidate regulators for the biological system under study by applying regression models, specifically Multiple Linear Regression models (MLR) or Partial Least Squares (PLS). MORE facilitates the application of MLR or PLS to multi-omic data.

MORE requires several data inputs: a target omic expression data, regulatory omic data, conditions of considered samples, and potential associations between target omic features and regulators. With this input data, MORE generates the initial model equation, which is different for each target omic feature because each one of them has different potential regulators. MORE admits numerical omic data (continuous or discrete) or binary data.

It is strongly recommended to fit MORE models only to target omic features that present significant changes in any of the conditions studied, for example, when considering Gene Expression data as target omic to differentially expressed genes (DEGs). DEGs can be selected with the standard procedures depending on the conditions, but DEGs selection is not included in the MORE algorithm and must be done by the user.

This idea can be extended to potential regulators since regulators that do not change across conditions are not good candidates to regulate the target omic feature expression. Removing non-DE regulators will also help to reduce the number of predictors in the model since an excess of them would prevent the estimation of regression coefficients. Even so, MORE has several functionalities to filter regulators with missing values or low variation, highly correlated regulators, and perform variable selection.

MORE package also includes a function to retrieve the significant regulations and the magnitude of the regulatory effect under each condition considered and an additional function to graphically investigate the relationship between target omic features and regulators.

2 Getting started

The MORE method is available as an R package from `https://github.com/BiostatOmics/MORE.git`. As for other packages in GitHub, it can be installed from R with the following instructions:

```
> install.packages("devtools")
> devtools::install_github("BiostatOmics/MORE")
```

3 Input data

This section describes the main data files required by MORE to generate the regression models.

Target data Expression values for features of a target comic (e.g. genes of gene expression data), in rows, under each condition, in columns. MORE accepts either a **matrix** or a **data frame**. In MORE it should be provided to **targetData** argument. See an example below:

```
> head(TestData$GeneExpressionDE)
                   Sample_1 Sample_2 Sample_3 Sample_4 Sample_5 Sample_6 Sample_7 Sample_8 Sample_9
ENSMUSG00000000078 16.35068 16.43845 16.08969 16.18012 15.98894 16.24338 16.14683 16.20514 15.53671
ENSMUSG00000056999 14.04936 15.00476 15.39196 15.38862 15.48909 15.74698 16.15151 15.97799 13.66564
ENSMUSG00000024873 11.70604 11.92110 12.88689 12.96273 12.75602 12.67522 12.99590 13.11888 10.93298
ENSMUSG00000015461 16.08382 16.20221 16.27489 16.29576 16.03688 15.71148 15.45131 14.66903 14.17544
ENSMUSG00000058135 13.02257 15.52288 16.35661 16.42303 16.38726 16.13189 15.76151 15.01974 13.93261
ENSMUSG00000038208 13.52974 13.64620 13.55908 13.39410 13.36335 13.34064 13.23755 13.32766 13.15812
                   Sample_10 Sample_11 Sample_12 Sample_13 Sample_14 Sample_15 Sample_16
ENSMUSG00000000078  15.27647 15.355672 15.005449  15.36413  15.16874  15.65070  15.38943
ENSMUSG00000056999  14.99625 15.436961 15.854549  15.76431  15.81027  15.65091  15.53761
ENSMUSG00000024873  11.92766 11.730738 11.649180  11.99320  12.10198  12.74034  12.69285
ENSMUSG00000015461  12.50220  9.709291  3.501127  11.07505  13.25271  14.35546  14.98628
ENSMUSG00000058135  15.07768 15.749751 15.709056  15.87783  15.57376  15.74364  14.80781
ENSMUSG00000038208  13.23678 12.969414 13.338755  13.46027  13.00960  12.95872  12.75443
```

Condition Matrix or data frame containing the conditions of the samples, such as treatments, diseases, strains, dose of a drug, etc. The rows of the object must be the same as the columns in **targetData** and in the same order, as shown below. In MORE it should be provided to **condition** argument

```
> TestData$edesign
           Group
Sample_1  "Control"
Sample_2  "Control"
Sample_3  "Control"
Sample_4  "Control"
Sample_5  "Control"
Sample_6  "Control"
Sample_7  "Control"
Sample_8  "Control"
Sample_9  "Disease"
Sample_10 "Disease"
Sample_11 "Disease"
Sample_12 "Disease"
Sample_13 "Disease"
Sample_14 "Disease"
Sample_15 "Disease"
Sample_16 "Disease"
```

Regulatory omic data This object must be a list where each element is a matrix or data frame containing the data for each “regulatory” omic (miRNA expression, transcription factor expression, etc.), with a structure similar to target data: regulators in rows and conditions in columns (the columns must be the same as in target data and in the same order). In MORE it should be provided to **regulatoryData** argument. See the example below (TestData$data.omics$‘miRNA-seq‘).

```
> head(TestData$data.omics$`miRNA-seq`)
               Sample_1 Sample_2 Sample_3 Sample_4 Sample_5 Sample_6 Sample_7 Sample_8 Sample_9
mmu-miR-125a-5p     7100     7229    10066     8868    12649    16321    14760    15164     1655
mmu-miR-141-5p        19       49       40      107      176      183      168      343       58
mmu-miR-145a-3p   117329    85793    73670    57547    43926    21426    11769        2    11593
mmu-miR-148b-3p   125403   136517   137890   190417   150718   217291   203966   203880   108489
mmu-miR-150-5p      1256     1098      707     1025      599      253      190        0     1289
mmu-miR-152-3p       344      445      322      529      655     1082     1645     1441       63
               Sample_10 Sample_11 Sample_12 Sample_13 Sample_14 Sample_15 Sample_16
mmu-miR-125a-5p       166         8       562      2818      4997      7528      9159
mmu-miR-141-5p          8         2         9        13       117       121       283
mmu-miR-145a-3p      1539       125      7075     19253     32778     61536    106987
mmu-miR-148b-3p    116911    102194     94160    126998    150972    171944    205767
mmu-miR-150-5p        844       617       517       374       370       164         0
mmu-miR-152-3p         17         0       121        94       270       665       687
```

Associations For each regulatory omic, associations between regulators and target features which indicate which are the potential regulators of each feature that will be consequently incorporated into the initial equation of the regression model. The association objects must be data frames and stored in a single **list** (attached below the example of miRNA-seq, TestData$associations$‘miRNA-seq‘). The names of the elements of this list must be the names of the list of regulatory omic data and must be in the same order. In MORE it should be provided to **associations** argument.

If the user wants to consider all regulators of an omic as potential regulators they must set to NULL the object of this omic in the **associations** list. Moreover, if the user does not provide the list of **associations**, all regulators of all omics in **regulatoryData** will be considered potential regulators for all target features. However, this option is very time-consuming. By default, NULL.

```
> head(TestData$associations$`miRNA-seq`)
                  Gene              ID
1 ENSMUSG00000024873  mmu-miR-335-3p
2 ENSMUSG00000024873 mmu-miR-1912-3p
3 ENSMUSG00000024873  mmu-miR-615-5p
4 ENSMUSG00000024873  mmu-miR-322-5p
5 ENSMUSG00000024873 mmu-miR-1894-3p
6 ENSMUSG00000024873 mmu-miR-7082-3p
```

4 Generating the regression models with MORE

The **more** function in MORE adjusts a regression model for each target feature (gene, protein, metabolite, etc.) in the **targetData** object to determine which regulators and conditions (treated vs. non-treated, control vs disease, etc.) have a significant effect on the response variable inferred by advanced variable selection methods. MORE can apply more conventional regression models such as Multiple Linear Regression models (MLR) in combination with Elastic Net (EN) or Iterative Sparse Group Lasso (ISGL) regularization regression methods if the user selects 'MLR' as the method to apply. It can also use a dimensionality reduction technique such as Partial Least Squares (PLS) if the selected method is 'PLS1' or 'PLS2'. These are the arguments the function accepts, described in detail in Section 4.1.

```
more(targetData, regulatoryData, associations, omicType = NULL,
     condition = NULL, clinic = NULL, clinicType = NULL,
     minVariation = 0, percNA = 0.2, scaleType = 'auto',
     epsilon = 0.00001, interactions = TRUE, varSel = 'EN',
     alfaEN = NULL, correlation = 0.7, groupingISGL = 'MF_0.7',
     alfa = 0.05, vip = 0.8, method  ='MLR', parallel = FALSE,
     seed = 123)
```

4.1 Arguments for more() function

targetData Matrix or data frame containing data from a target omic with its features in rows and samples in columns. The row names must be the target omic features IDs; e.g. when Gene Expression is considered as targetData gene IDs are to be considered.

regulatoryData List where each element corresponds to a different omic data type (miRNAs, transcription factors, methylation, etc.). The names of this list will be the omics, and each element of the list is a matrix or data frame with omic regulators in rows and samples in columns. *Attention!* We clarify that the user can not consider as an omic the data considered as **targetData**.

associations List where each element corresponds to a different omic data type (miRNAs, transcription factors, methylation, etc.). The names of the elements of the list will be the omics (in the same order as in **regulatoryData**). Each element is a data frame with two columns (optionally three) describing the potential interactions between target omic features and regulators for that omic. The first column must contain the regulators, the second the target features IDs, and an additional column can be added to describe the type of interaction (for example, in methylation data, if a CpG site is located in the promoter region of the gene, in the first exon, etc.). Optionally, the user can set the **associations** data frame of an omic equal to NULL if they want to consider all the regulators of that omic as potential regulators for all the target features. They can even set **associations** to NULL if they want to consider all regulators of all omics in **regulatoryData** as potential regulators to all target features. Even if it can be done, we do not recommend the user to do it as it can be very time-consuming.

omicType Vector with as many elements as the number of omics, indicating whether the omic values are numeric (0) or binary (1). When NULL is indicated, **MORE** will estimate which type of omics are provided and display them on the screen. If a single value is provided, the type for all the omics is set to that value. By default, NULL. If the estimated type of omics are incorrect, the user must halt the process and manually specify the omic type.

condition Data frame or matrix describing the condition to which samples belong. Rows must be the samples, that is, the columns in the **targetData**, and columns must be the conditions to be included in the model, such as disease, treatment, etc.

clinic Data frame or matrix containing clinical variables values where rows must represent samples and columns variables.

clinicType Vector with as many elements as the number of clinical variables, indicating whether the variables values are numeric (0) or categorical/binary (1). When NULL is indicated, **MORE** will estimate which type of variables are provided and display them on the screen. If a single value is provided, the type for all the variables is set to that value. By default, NULL. If the estimated type of variables are incorrect, the user must halt the process and manually specify the clinic type.

minVariation Vector with as many elements as the number of omics (names of this vector will be the omics), indicating the minimum change in the standard deviation that a regulator must show across conditions in order not to be considered as having low variation and be removed from the regression models, for numerical regulators. Or the minimum change in the proportion a regulator must show across conditions for binary regulators. When a single value is given, the minimum change will be considered the same for all omics. The user has the option to set this value to NA if they do not want to provide a value but are sure that they want to filter more than constant regulators across conditions. In this case, the value will be calculated as the 10% of the maximum observed variability across conditions for continuous regulators and as the 10% of the maximum observed proportion difference across conditions for binary regulators. Additionally, the user can combine both functionalities; indeed, the user has the option to provide a vector containing the minimum change in the standard deviation for some omics and NA for those omics for which they do not want to provide a value. By default, its value is 0. If the user has been very restrictive an error message will be provided.

percNA Maximum percentage of missing values present in regulatoryData regulators and observations to use to construct the models. Only used in PLS models since MLR models do not allow any missing values in the data. By default, 0.2.

scaleType Type of scaling to be applied when adjusting a model if scaling is requested. It can be: 'auto', 'softBlock', or 'hardBlock'. The first applies the autoscaling method; so that scales each variable independently. The second applies the soft block-scaling to the omics. The third applies the hard block-scaling considering as block each of the omics in **regulatoryData** and the interactions of experimental design variables with them if they were.

epsilon A threshold for the positive convergence tolerance in the MLR model. By default, 0.00001.

interactions If TRUE (default), **MORE** allows for interactions between each regulator and the conditions.

varSel Type of variable selection method to apply, different options depending on MLR or PLS method selection. Four options:

- **EN** Applies a Multiple Linear Regression (MLR) with ElasticNet (EN) regularization.
- **ISGL** Applies a Multiple Linear Regression (MLR) with Iterative Sparse Group Lasso (ISGL) regularization.
- **Jack** Applies Jack-Knife resampling technique for the calculation of the significance of the coefficients in Partial Least Squares (PLS) models.
- **Perm** Applies a resampling technique for the calculation of the significance of the coefficients in Partial Least Squares (PLS) models in which the response variable is permuted 100 times to obtain the distribution of the coefficients and compute then their associated p-value.

alfaEN ElasticNet mixing parameter (α). By default, NULL. These are the values that can be passed to this argument:

- **NULL** α parameter will be automatically optimized for *cvup*, which is the mean cross-validated error plus the estimate of the standard error. For computational efficiency, it will only be tested with values ranging from 0 to 1 in increments of 0.1.
- **Value between 0 and 1** ElasticNet is applied with this α being the combination between ridge and lasso penalization.

Vector of α's ElasticNet will be applied for each of the α values provided in the vector, and the one that optimizes the *cvup* will be selected.

Value 0 The ridge penalty.

Value 1 The lasso penalty.

We make clear that the shrinkage parameter (λ) will in all cases be optimized by cross-validation.

correlation Correlation threshold (in absolute value) to decide which regulators are correlated, in which case, a/some representative of the group of correlated regulators is chosen to enter the model. By default, 0.7.

groupingISGL Type of approach to take for creating the groups for the Iterative Sparse Group Lasso regularization. By default, 'MF_0.7'. There are two options:

MF_X Takes the same approach as in the multicollinearity filter. Grouping the variables which correlate more than X. E.g: MF_0.7 groups variables in the same groups when they are correlated more than 0.7. By default, MF_0.7.

PCA_X Extracts as many PCA components to explain X percent of the variability of the data and assigns the variables to the component in which they had the highest loadings.

alfa Significance level to consider a regulator as significant in PLS models. By default, 0.05.

vip The Variable Importance in Projection, VIP, score threshold to apply together with the **alfa** threshold to take a variable as significant; both requirements should be met to take a variable as significant. By default, 0.8.

method This parameter indicates whether a MLR model will be applied, 'MLR', or if a PLS model will be applied instead. In this case, the user can ask for a PLS1 model using 'PLS1' in which for each of the target features in **targetData** a different PLS model will be applied or for a PLS2 model using 'PLS2' in which a single PLS model will be computed which creates a single model for all target features. The user must be aware that in the 'PLS2' model, the **association** list will not be considered and must be set to NULL.

parallel If FALSE, **MORE** will be run sequentially. If TRUE, **MORE** will be run using parallelization with as many cores as the available ones minus one so the system is not overcharged. If the user wants to specify how many cores they want to use, they can also provide the number of cores to use in this parameter. Parallelization is only implemented for MLR with ElasticNet regularization and PLS1 methods. By default, FALSE.

seed Sets the seed to guaranty reproducibility of the results for the methods that require random sampling of the observations (i.e., all MLR approaches and PLS with Perm variable selection). By default, 123.

4.2 User recommendations

1. When running the PLS2 methodology in **MORE**, we recommend running it in a cluster rather than on the desktop due to its memory consumption.
2. When selecting PLS2 methodology and considering high-dimensional datasets, rather for **targetData** or **regulatoryData** datasets we recommend using '*Perm*' methodology for p-value computation (**varSel**='Perm') because of memory consumption efficiency.
3. If the user still faces problems due to memory when running PLS2 models, we recommend running separate models of **MORE**. One for each conditions setting interactions = FALSE. To create the differential network, the user would only have to merge the columns of the considered conditions in **RegulationPerCondition** matrices.

4.3 more output

The object returned by the **more** function varies depending on the selected method.

4.3.1 more output for MLR

The object returned by the **more** when fitting a MLR model (with elasticnet or ISGL penalization indifferently) is a list that contains the following elements:

ResultsPerTargetF is a list with as many elements as target features in the **targetData** object. For each target feature, there is a list containing the following information:

- **Y** Data frame with the response variable values for that target feature (y), the values fitted by the model (fitted.y), and the residuals of the model (residuals).
- **X** Data frame with all the predictors included in the final model.
- **coefficients** Matrix with the estimated coefficients for the regulators selected as relevant by the EN or the ISGL regularization methods.
- **allRegulators** Data frame with all the initial potential regulators in rows and the following information in columns: target feature, regulator, omic, area (the third optional column in associations), filter (if the regulator has been filtered out of the model, this column indicates the reason), and Rel (1 if the regulator is considered relevant and 0 if not). Regarding the filter column, several values are possible:

- *MissingValue*: If the regulator has been filtered out of the study because it has missing values.
- *LowVariation*: If the regulator has been filtered out of the study because it has lower variability than the threshold set by the user in **minVariation** parameter.
- *Model*: When the regulator is included in the initial equation model.
- *omic_mcX_X_X*: For example, *TF_mc1_1_R*. This notation is related to highly correlated regulators and how they are treated to avoid the multicollinearity problem. Only present for MLR models when EN regularization is selected. Following the *TF_mc1_1_R* example, two or more regulators, which potentially regulate the target feature, are highly correlated (in absolute value). In such cases, one is chosen as the representative and indicated with _R. The rest of the regulators considered if they are directly highly correlated to it are labeled with _P, which means that they are positively correlated with the representative and with _N if negatively correlated. Once a representative is taken for them, if there are still highly correlated regulators, the process is repeated and indicated in the _1 of the example. An additional row is then added to this table, with the regulator *TF_mc1_1_R* and the filter label being *Model*, since only this representative is considered in the model. When there are several groups of correlated regulators for the same omic, it is indicated with _mc1_, _mc2_, etc.

relevantRegulators A character vector containing the relevant regulators.

GlobalSummary List that contains the following elements:

GoodnessOfFit Matrix that collects the R-squared value (which for MLRs is defined as the percentage of deviance explained by the model),the adjusted R-squared value, the Root Mean Square Error (RMSE), the Normalized Root Mean Square Error (NRMSE) and the number of relevant regulators for all the target features that had at least a relevant regulator.

ReguPerTargetF Matrix containing, for each omic and target feature, the number of initial regulators, the number of regulators included in the initial model, and the number of relevant regulators.

TargetFNOmodel List of target features for which the final MLR model with the EN or ISGL regularization could not be obtained. There are four possible reasons for that, and they are indicated: “Too many missing values”, “-Inf/Inf values”, ”No regulators left after NA/LowVar filtering” and ”No regulators selected after variable selection”.

TargetFNOregu List of target features for which there were no initial regulators, only generated in case any target feature was under this condition.

GlobalRegulators Vector that contains regulators that relevantly regulate more target features. It comprises the third quartile of regulators with the highest number of regulations. If the count is below 10, the top 10 regulators are selected.

HubTargetF Vector that contains target features that are relevantly regulated by more regulators. It comprises the third quartile of target features with the highest number of regulators. If the count is below 10, the top 10 target features are selected.

Arguments List with the arguments used to generate the model: conditions, minimum degrees of freedom in the residuals, significance level, etc.

4.3.2 more output for PLS

The object returned by the **more** when fitting a PLS (PLS1 or PLS2 indifferently) model is a list that contains the following elements:

ResultsPerTargetF is a list with as many elements as target features in the **targetData** object. For each target feature, there is a list containing the following information:

Y Data frame with the response variable values for that target feature (y), the values fitted by the model (fitted.y), and the residuals of the model (residuals).

X Data frame with all the predictors included in the final model.

coefficients Matrix with the estimated coefficients for the regulators selected as significant by the selected p-value computation method (**varSel**) and whose Variable Importance in Projection (**vip**) has been higher than the threshold.

allRegulators Data frame with all the initial potential regulators in rows and the following information in columns: target feature, regulator, omic, area (the third optional column in associations), filter (if the regulator has been filtered out of the model, this column indicates the reason), and Sig (1 if the regulator is considered significant and 0 if not). Regarding the filter column, several values are possible:

- *MissingValue*: If the regulator has been filtered out of the study because it has missing values.
- *LowVariation*: If the regulator has been filtered out of the study because it has lower variability than the threshold set by the user in **minVariation** parameter.
- *Model*: When the regulator is included in the initial equation model.

significantRegulators A character vector containing the significant regulators.

GlobalSummary List that contains the following elements:

GoodnessOfFit Matrix that collects the R-squaredY value (the R squared of the response variable), the Q-squared (the goodness of prediction), the square root of the mean error between the actual and the predicted responses (RMSE), the Normalized Root Mean Square Error (NRMSE) and the number of significant regulators for all the target features that had at least a significant regulator.

ReguPerTargetF Matrix containing, for each omic and target feature, the number of initial regulators, the number of regulators included in the initial model, and the number of significant regulators.

TargetFNOmodel List of target features for which the final PLS model could not be obtained. There are three possible reasons for that, and they are indicated: "Too many missing values", "-Inf/Inf values", and "No regulators left after NA/LowVar filtering".

TargetFNOregu List of target features for which there were no initial regulators, only generated in case any target feature was under this condition.

GlobalRegulators Vector that contains regulators that significantly regulate more target features. It comprises the third quartile of regulators with the highest number of regulations. If the count is below 10, the top 10 regulators are selected.

HubTargetF Vector that contains target features that are significantly regulated by more regulators. It comprises the third quartile of target features with the highest number of regulators. If the count is below 10, the top 10 target features are selected.

Arguments List with the arguments used to generate the model: conditions, significance level, etc.

4.3.3 more summary

Making use of the output object returned by **more**, in both cases in MLR and PLS models, the user can ask for a summary of the results obtained by:

```
summary(object, plot.more=FALSE)
```

This summary takes two arguments as input:

object MORE object obtained from applying **more** function, indifferent to the method that has been used ('MLR', 'PLS1' or 'PLS2').

plot.more If TRUE, the top 10 global regulators will be plotted against the target features they regulate. By default, FALSE. It could be very time-consuming if the global regulators regulate a huge number of target features, so it is not recommended unless the user knows that there are only a few of them. Instead, it is recommended to use the **plotMORE** function to plot the specific regulations.

Once the function is used the following information will be printed on the screen:

1. Number of genes for which a model was computed.
2. Number of genes that did not have initial regulators.
3. Number of genes for which the final model could not be obtained.
4. The mean of relevant/significant regulators of the genes.
5. Top 10 hub genes and the number of relevant/significant regulators for each.
6. Top 10 global regulators and the number of genes they regulate. They will have to regulate at least 10 genes to be considered global regulators.
7. If required with plot.more = TRUE, the plots of the global regulators against the genes they regulate.

4.4 Running an example

An example of the execution of **more** function for the 'MLR' option is shown next by using simulated data. Even if the data file `TestData.RData` is available in the package; the results shown below are related to the STATegra database available here.

In this file, the targetData matrix corresponds to the omic RNA-seq (**GeneExpressionDE**), and there is a list with four matrices of regulators in the **data.omics** object (all values are normalized):

miRNA-seq miRNA expression data.

DNase-seq measures the chromatin accessibility expression.

Methyl-seq Methylation per CpG site (M values).

TF TF expression data.

The experimental design matrix (**edesign**) consists of 6 time points in two conditions, which results in a total of 12 experimental samples, but we are not going to consider it as we are only interested in comparing temporal profiles for the two experimental groups.

We can run the following **more** code to obtain the regression models for our genes:

```
> set.seed(123)
> SimMLR = more(targetData = TestData$GeneExpressionDE,
                regulatoryData = TestData$data.omics,
                associations = TestData$associations,
                omicType = c(1,0,0), condition = TestData$edesign,
                minVariation = 0, interactions = TRUE,
                alfaEN = NULL, varSel = 'EN', correlation = 0.7,
                method = 'MLR')
```

Some of the estimated coefficients of the relevant regulators in the final MLR model computed by **more** for the gene ENSMUSG00000036932 are

```
> head(SimGLM$ResultsPerGene$ENSMUSG00000024873$coefficients)
                  coefficient
(Intercept)      12.299482593
mmu-miR-16-1-3p  -0.001460894
mmu-miR-1905      0.035645115
mmu-miR-301a-5p   0.008315882
mmu-miR-7084-3p   0.047222937
mmu-miR-92a-2-5p  0.044996137
```

The **allRegulators** table shows, for each gene, their regulators, omic, area, the kind of filter applied, and if the regulator is considered relevant or not. In this case (see **filter** column), in miRNA-seq, the regulator mmu-miR-335-3p has been chosen as representative regulator (R) for being correlated to other regulators. In the filter column, the correlations are indicated using (R) for the representative, (P), and (N) for positive and negative correlation with the representative, respectively. In the same column, `Model` means that the regulator was included in the model by itself. On the other hand, the **Rel** column returns 1 if the regulator was considered relevant in the final model and 0 if not.

```
> head(SimGLM$ResultsPerGene$ENSMUSG00000024873$allRegulators)
                              gene       regulator      omic area          filter Rel
mmu-miR-335-3p  ENSMUSG00000024873  mmu-miR-335-3p miRNA-seq      miRNA-seq_mc1_R   1
mmu-miR-1912-3p ENSMUSG00000024873 mmu-miR-1912-3p miRNA-seq                Model   0
mmu-miR-615-5p  ENSMUSG00000024873  mmu-miR-615-5p miRNA-seq                Model   0
mmu-miR-322-5p  ENSMUSG00000024873  mmu-miR-322-5p miRNA-seq                Model   1
mmu-miR-1894-3p ENSMUSG00000024873 mmu-miR-1894-3p miRNA-seq                Model   0
mmu-miR-7082-3p ENSMUSG00000024873 mmu-miR-7082-3p miRNA-seq                Model   0
> tail(SimGLM$ResultsPerGene$ENSMUSG00000024873$allRegulators)
                                gene         regulator      omic area filter Rel
miRNA-seq_mc3_R   ENSMUSG00000024873   miRNA-seq_mc3_R miRNA-seq       Model   0
miRNA-seq_mc4_1_R ENSMUSG00000024873 miRNA-seq_mc4_1_R miRNA-seq       Model   1
miRNA-seq_mc5_1_R ENSMUSG00000024873 miRNA-seq_mc5_1_R miRNA-seq       Model   1
miRNA-seq_mc6_R   ENSMUSG00000024873   miRNA-seq_mc6_R miRNA-seq       Model   0
miRNA-seq_mc7_R   ENSMUSG00000024873   miRNA-seq_mc7_R miRNA-seq       Model   0
miRNA-seq_mc8_R   ENSMUSG00000024873   miRNA-seq_mc8_R miRNA-seq       Model   0
```

Finally, we can ask for a litlle summary of the created model calling summary function:

```
> summary(SimGLM)
A model was computed for 20 genes.
0 genes had no intial regulators.
For 0 genes, the final GLM model could not be obtained.
Genes presented a mean of  32.36842 relevant regulators.
These are the top 10 hub genes and the number of relevant regulators for each:
ENSMUSG00000049624 ENSMUSG00000038208 ENSMUSG00000012535 ENSMUSG00000021583 ENSMUSG00000016018 ENSMUSG00000041995 ENSMUSG00000050439 ENSMUSG00000033985
                91                 68                 66                 57                 55                 42                 34                 28
ENSMUSG00000024873 ENSMUSG00000026563
                28                 24
There were not global regulators (regulators that regulate more than 10 genes).
```

5 Retrieving significant regulations from MORE results

The function **RegulationPerCondition** is applied to the **more** output. It returns a summary table containing all the relevant/significant regulations, that is, all the pairs target feature-regulator considered relevant/significant in MORE models (depending if a MLR or a PLS model was applied). Moreover, it provides the regression coefficient that relates the target feature and the regulator for each experimental condition after testing if this coefficient is relevant/significant or not.

```
RegulationPerCondition(output, filterR2 = 0)
```

5.1 RegulationPerCondition input parameters

output Object containing the output of **more** function.

filterR2 Filters out target features with less R2 than the specified. By default, 0.

5.2 Interpreting RegulationPerCondition output with an example

Following the previous example, we can run the **RegulationPerCondition** function.

```
> myresults = RegulationPerCondition(SimMLR)
```

The output is the following table, where some pairs target feature-regulator are selected to have a complete vision of the output of this function:

```
> head(myresults)
             targetF       regulator      omic area representative Group_Control Group_Disease
1 ENSMUSG00000000078  mmu-miR-139-3p miRNA-seq                          -0.002225     -0.002225
2 ENSMUSG00000000078           Mef2d        TF                           0.000000     -0.771200
3 ENSMUSG00000000078            Rxra        TF                           0.000000     -0.004460
4 ENSMUSG00000024873 mmu-miR-16-1-3p miRNA-seq                          -0.001215     -0.001291
5 ENSMUSG00000024873    mmu-miR-1905 miRNA-seq                           0.056730      0.056730
6 ENSMUSG00000024873 mmu-miR-301a-5p miRNA-seq                           0.015220      0.015220
> tail(myresults)
               targetF regulator omic area representative Group_Control Group_Disease
526 ENSMUSG00000036932    Nfe2l2   TF              Nfe2l2         0.000       -0.2076
527 ENSMUSG00000036932      Rara   TF              Nfe2l2         0.000        0.2076
528 ENSMUSG00000036932     Runx1   TF              Nfe2l2         0.000        0.2076
529 ENSMUSG00000036932    Srebf2   TF              Zfp787        -0.405       -0.5161
530 ENSMUSG00000036932      Tcf3   TF              Nfe2l2         0.000        0.2076
531 ENSMUSG00000036932    Zfp787   TF              Zfp787        -0.405       -0.5161
```

This table shows the relevant regulators for each target feature. The **representative** column indicates if the regulator was chosen as the representative of a correlated group of regulators or, otherwise, which regulator was taken as the representative of the group. When no information is provided in this column, it means that the regulator was not part of a correlated group of regulators. Regulators correlated positively with the representative will have the same coefficients (same sign) as the representative, while negatively correlated regulators will have the same coefficients as the representative but with the opposite sign.

The final columns correspond to the regression coefficients of each regulator for each experimental group. In this case, the experimental design matrix (**edesign**) contained two conditions, so the column `Group_Control` corresponds to the first condition, and `Group_Disease` corresponds to the second one. These are the conclusions we can draw from the coefficients:

- If two experimental groups have the same coefficients, it means that the regulator has the same effect on the target feature in both groups.
- If one of the coefficients is 0, it means that the regulator does not have an effect on the target feature under this experimental condition.
- Experimental groups with different non-zero coefficients indicate that the regulator affects the target feature in all these experimental groups but the magnitude of the effect is not the same for all these groups.

NOTE 1: When MLR models are used, a statistical test should be applied to determine whether the beta coefficients significantly differ between conditions, thereby validating the significance of interaction terms. This test should be applied as the variable selection methods we used in the MLR models do not guarantee the significance of the selected variables. MORE includes this statistical test under the function **BetaTest**.

5.3 BetaTest

This auxiliary function conducts a hypothesis test on the sum of the regression coefficients of each condition to determine whether there are statistical differences between conditions. As previously explained, this test should be applied when MLR models are used. **See that the statistical differences will be analyzed against a group of reference, so that when more than two groups are considered and the user wants to test statistical differences between all groups, it would be necessary to rerun the more() main function, changing the group of reference and applying this test every time.**

```
BetaTest(output, outputRegpcond, alfa = 0.05)
```

This function takes three arguments as input:

output Output object of running **more** function. Should be an output object from running the function for a MLR model.

outputRegpcond Output object of running **RegulationPerCondition** function.

alfa Significance level (α) that the user wants to consider. By default, 0.05. If the *p-value* is greater than α, the null hypothesis is accepted, and the corresponding regression coefficient is set to zero. Otherwise the coefficient is set to the actual sum of the relevant coefficients.

Following the previous example, we can run the BetaTest function by:

```
myresutls = BetaTest(SimMLR, myresults, alfa = 0.05)
```

The output of the function is the modified table of **RegulationPerCondition**:

```
> myresults = BetaTest(SimMLR, myresults)
  |====================================================================================================================================================| 100%
> head(myresults)
            targetF       regulator      omic area representative Group_Control Group_Disease
1 ENSMUSG00000000078  mmu-miR-139-3p miRNA-seq                        -0.002225     -0.002225
2 ENSMUSG00000000078           Mef2d        TF                         0.000000     -0.771200
4 ENSMUSG00000024873 mmu-miR-16-1-3p miRNA-seq                        -0.001215     -0.001291
5 ENSMUSG00000024873    mmu-miR-1905 miRNA-seq                         0.056730      0.056730
6 ENSMUSG00000024873 mmu-miR-301a-5p miRNA-seq                         0.015220      0.015220
7 ENSMUSG00000024873 mmu-miR-1249-5p miRNA-seq                         0.002520      0.002520
> tail(myresults)
              targetF regulator omic area representative Group_Control Group_Disease
526 ENSMUSG00000036932    Nfe2l2   TF              Nfe2l2         0.000       -0.2076
527 ENSMUSG00000036932      Rara   TF              Nfe2l2         0.000        0.2076
528 ENSMUSG00000036932     Runx1   TF              Nfe2l2         0.000        0.2076
529 ENSMUSG00000036932    Srebf2   TF              Zfp787        -0.405       -0.5161
530 ENSMUSG00000036932      Tcf3   TF              Nfe2l2         0.000        0.2076
531 ENSMUSG00000036932    Zfp787   TF              Zfp787        -0.405       -0.5161
```

As shown in this example, although *Rxra* was initially selected as a relevant regulator of the *ENSMUSG0000000078* gene under the Disease condition, the application of the **BetaTest** function revealed that the relationship was not statistically significant. Consequently, the coefficient was set to zero, ans since the regulator had no effect in either condition, it was removed from the table.

5.4 RegulationInCondition

In case the user is interested in a particular biological condition, to summarize the information, MORE includes the function **RegulationInCondition** which takes as input the output of **RegulationPerCondition** and the condition in which they are interested. It returns a list containing the hub genes, global regulators, and regulators with their coefficients specific to that condition.

```
RegulationInCondition(output_regpcond, cond)
```

Following the previous example, we can run the **RegulationInCondition** function for example for the *Disease* condition.

```
> myres_dis = RegulationInCondition(myresults, 'Disease')
```

The results contain a compact version of **RegulationPerCondition** only with those regulators that present a relevant/significant regulation in the specified condition, the global regulators and the hub genes in that condition.

```
> myres_dis$GlobalRegulators
character(0)
> myres_dis$HubTargetF
 [1] "ENSMUSG00000012535" "ENSMUSG00000038208" "ENSMUSG00000021583" "ENSMUSG00000016018" "ENSMUSG00000041995" "ENSMUSG00000050439" "ENSMUSG00000033985"
 [8] "ENSMUSG00000024873" "ENSMUSG00000036932" "ENSMUSG00000058135"
> head(myres_dis$RegulationInCondition)
             targetF       regulator      omic Group_Disease
1 ENSMUSG00000000078  mmu-miR-139-3p miRNA-seq     -0.002225
2 ENSMUSG00000000078           Mef2d        TF     -0.771200
4 ENSMUSG00000024873 mmu-miR-16-1-3p miRNA-seq     -0.001291
5 ENSMUSG00000024873    mmu-miR-1905 miRNA-seq      0.056730
6 ENSMUSG00000024873 mmu-miR-301a-5p miRNA-seq      0.015220
7 ENSMUSG00000024873 mmu-miR-1249-5p miRNA-seq      0.002520
```

6 Plotting MORE results

MORE package includes several plots for the interpretation of the results.

6.1 summaryPlot function

MORE package includes the function **summaryPlot** to graphically represent the summary of the relationship between target features and regulators found when creating the models. It creates two types of summary plots depending on user specifications.

```
summaryPlot(output, output_regpcond, filterR2 = 0, byTargetF =TRUE)
```

6.1.1 summaryPlot input parameters

output Object generated by the function **more**.

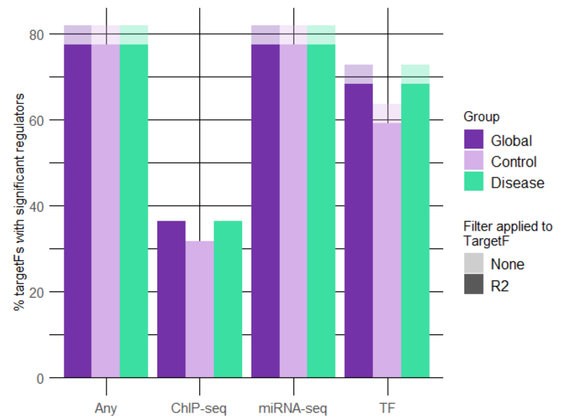
outputRegpcond Object generated by the function **RegulationPerCondition** when applied to a **more** object.

filterR2 Highlights the results for the genes that showed a R2 above the one indicated in this parameter. By default, 0.

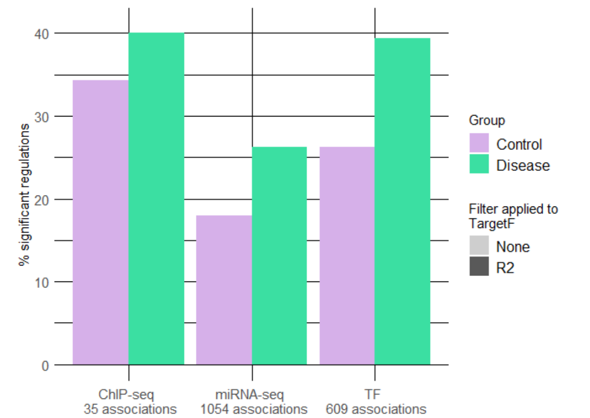
byTargetF If TRUE (default), the function plots the percentage of target features with significant regulators globally and per omic. If FALSE, it plots the percentage of significant regulations per omic.

As an example we introduce the summary plots generated for the model previously created with **more**:

```
summaryPlot(SimMLR, myresults,
            filterR2 = 0.5)
```




```
summaryPlot(SimMLR, myresults,
            byTargetF = FALSE)
```



6.2 FilterRegulationPerCondition

MORE package includes the function **FilterRegulationPerCondition** to filter out the genes with low R^2 from the **RegulationPerCondition** matrix. The function filters out genes with R^2 bellow the one specified by the user. Thanks to **summaryPlot** the user can be aware of the percentage of genes that will be filtered out when setting a specific threshold.

```
FilterRegulationPerCondition(output, outputRegpcond, filterR2=0)
```

This function takes two arguments as input:

output Output object of running **more** function.

outputRegpcond Output object of running **RegulationPerCondition** function.

filterR2 By default, 0. Filters out the results for the genes that showed a R2 bellow the one indicated.

6.3 globaregPlot

MORE package includes the function **globalregPlot** to graphically visualize the associations between the global regulators and the genes in a specific experimental condition. The function plots a type of corrplot in which the significant regulations are colored by the effect on the target feature (repressor/activator), and the potential regulations that not resulted significant are also shown.

```
globalregPlot(outputRegincond, byNetwork=FALSE)
```

This function takes two arguments as input:

outputRegincond Output object of running **RegulationInCondition** function.

byNetwork By default, FALSE. If TRUE, information would be plotted on a network instead of a corrplot.

Due to the scarcity of the data used as an example, there were no global regulators, so we can not show the results in this case.

6.4 differentialRegPlot function

MORE package includes the function **differentialRegPlot** to graphically visualize the differential regulators across experimental conditions and analyzed omics. The function creates an upset plot in which the proportions of the barplots show the proportion of the identified regulators across omic types for each combination of the experimental conditions. Numbers within bars indicate the absolute counts of regulators for each omic in each specific combination.

```
differentialRegPlot(output, outputRegpcond)
```

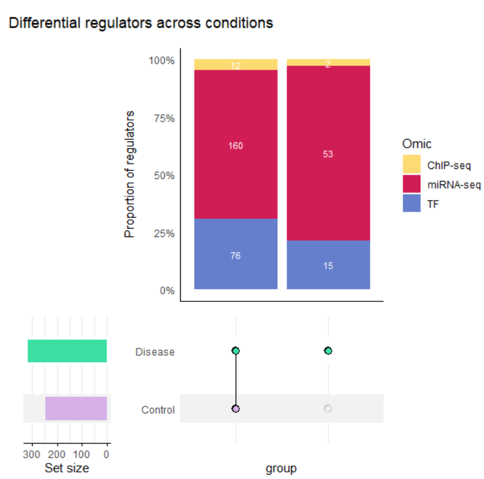
This function takes two arguments as input:

output Output object of running **more** function.

outputRegpcond Output object of running **RegulationPerCondition** function.

If we run the code for the example we were analyzing, it generates the following plot:

```
differentialRegPlot(SimMLR, myresults)
```



6.5 plotMORE function

MORE package includes the function **plotMORE** to represent the relationship between the target features and regulators graphically: for a given pair target feature-regulator, to explore the regulators of a given target feature, or to analyze which target features are regulated by a specific regulator.

```
plotMORE(output, targetF, regulator = NULL, simplify = FALSE,
         reguValues = NULL, plotPerOmic = FALSE, targetF.col = 1,
         regu.col = NULL, order = TRUE, xlab = "", cont.var = NULL,
         cond2plot = NULL)
```

6.5.1 plotMORE input parameters

output Object generated by the function **more**.

targetF ID of the target feature to be plotted.

regulator ID of the regulator to be plotted. If NULL (default value), all the regulators of the target feature are plotted.

simplify If TRUE, a boxplot (if the regulator is binary) or a Scatterplot (otherwise) is plotted to represent the relationship between the target feature and the regulator provided to the function. If FALSE (default), the target feature and the regulator profiles will be plotted. When many samples are provided to create the models, it is hard to differentiate something in this second option.

reguValues Vector containing the values of a regulator that the user can optionally provide. If NULL (default value), these values are taken from **output** as long as they are available.

plotPerOmic If TRUE, all the relevant regulators of the given target feature and the same omic are plotted in the same graph. If FALSE (default value), each regulator is plotted in a separate plot.

targetF.col Color to plot the target feature. By default, biostatomics colors.

regu.col Color to plot the regulator. If NULL (default), a color will be assigned by the function, that will be different for each regulatory omic.

order If TRUE (default), the values in X-axis are ordered according to target feature expression.

xlab Label for the X-axis.

cont.var Vector with length equal to the number of observations in data, which optionally may contain the values of the numerical variable (e.g. time) to be plotted on the X-axis. By default, NULL. It plots a range for each observation in which the observation could take values taking into account the numerical variable introduced.

cond2plot Vector or factor indicating the experimental group of each value to represent. If NULL (default), the labels are taken from the experimental design matrix.

smooth If TRUE (default), smoothing is applied via splines to the actual targetF and regulator values so that profiles can be more easily seen. The smoothed profiles would be displayed in bright colors while the real values are displayed with some degradation.

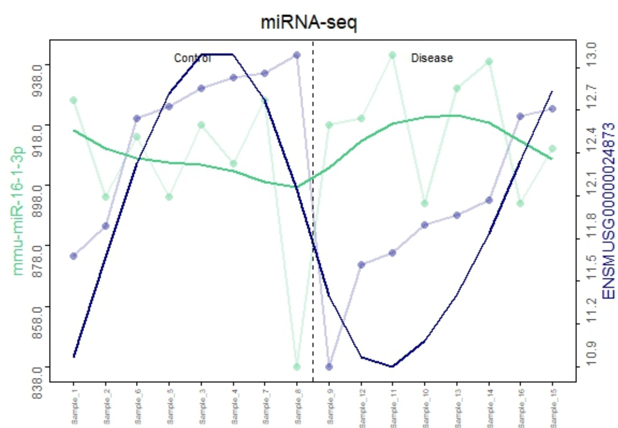
size Size of the X-axis labels. By default, 3.

breakby Range in which values should be displayed in Y-axis for regulators and target features, respectively. By default, c(0.3,0.3).

6.5.2 Interpretation of MORE plots

Following the previous example, the MORE graphic below represents the expression profile of a given gene (ENSMUSG00000024873) and the values for a relevant regulator of this gene (miRNA regulator, mmu-miR-16-1-3p). It can be generated with the following code:

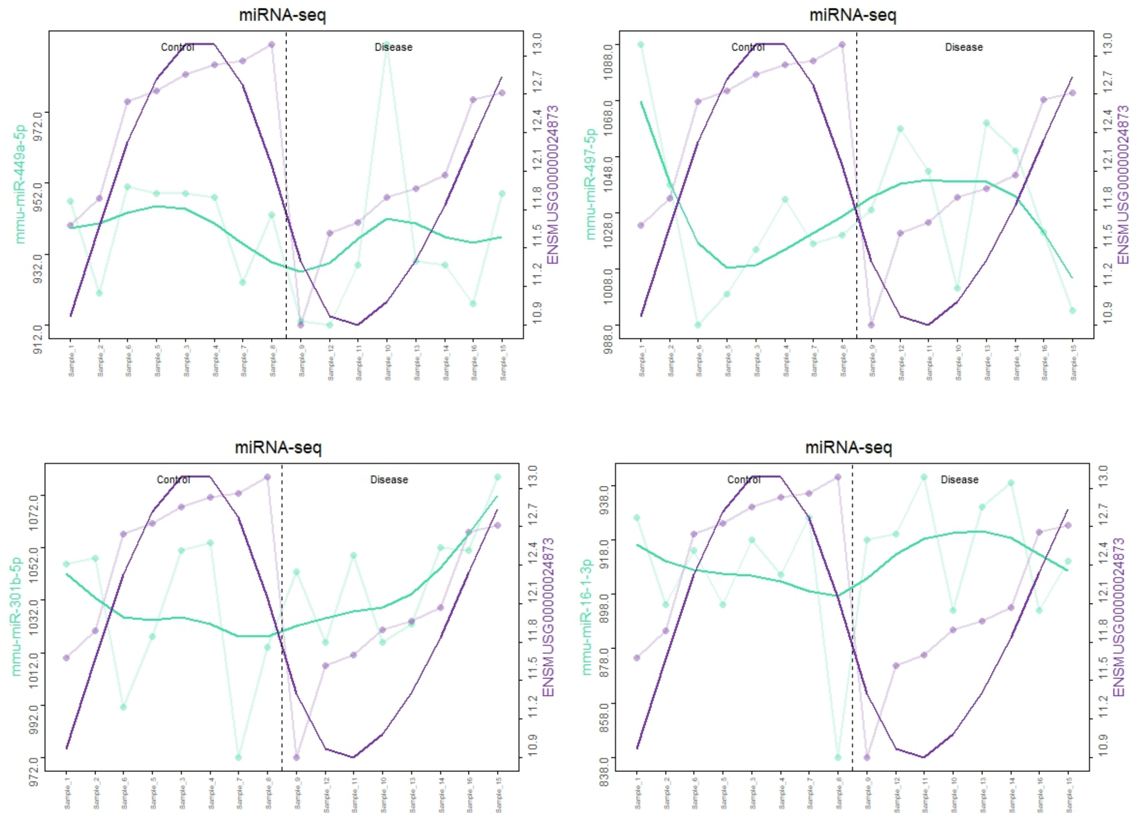
```
> plotMORE(output = SimMLR, targetF = "ENSMUSG00000024873",
           regulator = "mmu-miR-16-1-3p", plotPerOmic = FALSE,
           targetF.col = "blue4", order = TRUE,
           regu.col = "seagreen3", size = 5, breakby = c(20,0.3))
```



The X-axis is divided into two conditions (Control or Disease), and within each condition, the observations are displayed. The right Y-axis shows the expression values for the gene (plotted in blue), while the left Y-axis indicates the values for the regulator (plotted in green).

If we set the regulator argument to NULL, all the relevant regulators of gene ENSMUSG-00000024873 will be plotted (27 regulators). Only 4 will be presented:

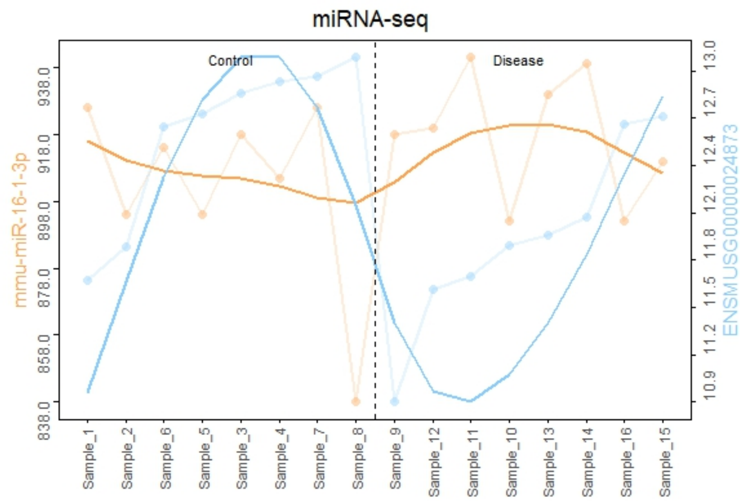
```
> plotMORE(output = SimMLR, targetF = "ENSMUSG00000024873",
          regulator = NULL, order =TRUE, plotPerOmic = FALSE,
          size = 5, breakby = c(20,0.3))
```



The title of each plot indicates the omic represented in that plot and the area. The values for relevant regulators are plotted in different colors according to the omic. The values for the gene are plotted in sea green, as indicated in the previous code.

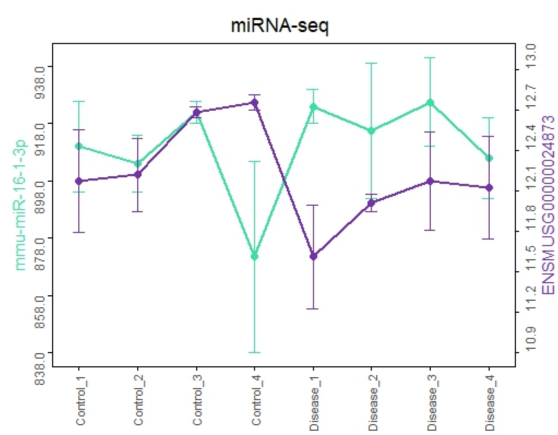
If we want to plot all the genes that are considered to be relevantly regulated by a given regulator (e.g. mmu-miR-16-1-3p), we must set the gene argument to NULL as follows. In this case, the miRNA regulates one gene: ENSMUSG00000024873.

```
> plotMORE(output = SimMLR, targetF = NULL,
           regulator = "mmu-miR-16-1-3p", plotPerOmic = FALSE,
           order = FALSE, targetF.col = "skyblue1",
           regu.col = "tan1", size = 8, breakby=c(20,0.3))
```



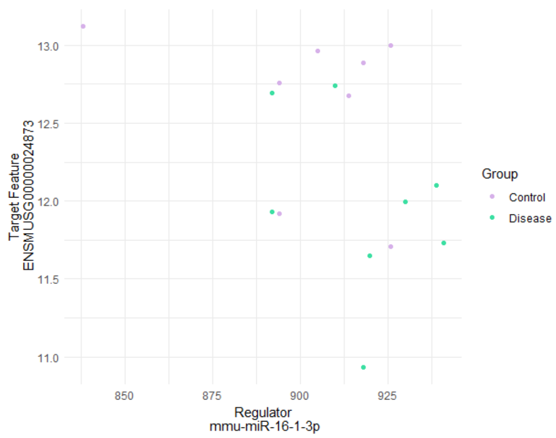
Additionally, if the user knows that the samples belong to different time points, they can provide this information in **cont.var** vector. A 'confidence interval' will be plotted for each of the time points to which the samples belong to. The code and the resulting graph, where the gene is plotted in light orange, and the regulator is plotted in blue, can be found below.

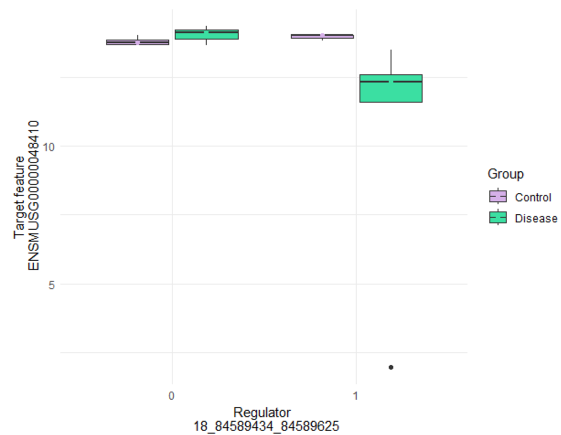
```
> plotMORE(output = SimMLR, targetF = "ENSMUSG00000024873",
          regulator = "mmu-miR-16-1-3p", plotPerOmic = FALSE,
          cont.var = c(1,2,3,4,1,2,3,4), size = 5, breakby=c(20,0.3))
```



In this case, as only 16 samples are plotted the profiles can clearly be seen. However, in case there were difficulties, we encourage the user to apply simplify parameter to see the relationships between a gene and a regulator. An example for binary regulator and continuous regulator are shown:

```
> plotMORE(output = SimMLR, targetF = "ENSMUSG00000024873",
           regulator = "mmu-miR-16-1-3p", simplify =TRUE)
> plotMORE(output = SimMLR, targetF = "ENSMUSG00000048410",
           regulator = "18_84589434_84589625", simplify =TRUE)
```





6.6 Plots for PLS models

Models created by MLR regressions in MORE apply a previous multi-collinearity filter, and users can see in the table generated by **RegulationPerCondition** the regulators that co-act jointly in the target feature expression. This multi-collinearity filter is not applied in PLS models as these models benefit from the multi-collinearity. In order to analyze the regulators that co-act jointly and the groups to which they are related, in MORE, we introduce two functions:

6.6.1 plotWeight function

The **plotWeight** function in MORE plots the weighting star of the regulators identified as significant in PLS models.

```
plotWeight(output, targetF, axe1 = 1, axe2 =2)
```

output Object generated by the function **more**. It must be generated either by PLS1 or PLS2 models.

targetF ID of the target feature to be plotted.

axe1 Component to plot in the X-axis. Default, 1.

axe2 Component to plot in the X-axis. Default, 2.

6.6.2 plotScores function

The **plotScores** function in MORE plots the scores of the samples under the PLS model generated by the regulators identified as significant.

```
plotScores(output, targetF, axe1 = 1, axe2 =2)
```

output Object generated by the function **more**. It must be generated either by PLS1 or PLS2 models.

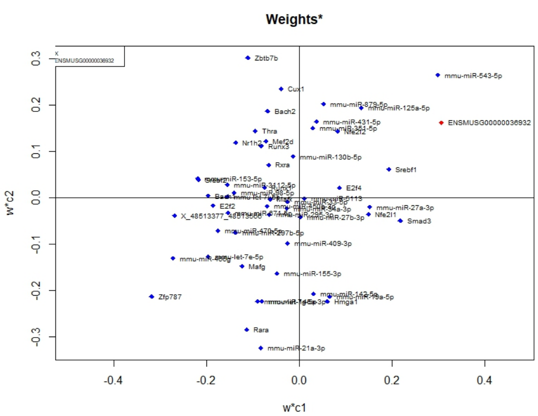
targetF ID of the target feature to be plotted.

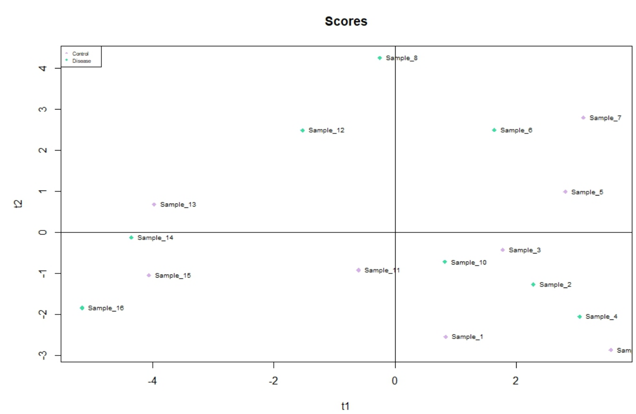
axe1 Component to plot in the X-axis. Default, 1.

axe2 Component to plot in the X-axis. Default, 2.

```
plotWeight(SimPLS,
targetF= "ENSMUSG00000036932")
```

```
plotScores(SimPLS,
targetF= "ENSMUSG00000036932")
```





To generate the examples below, we run a PLS model with the PLS1 method and Jack-Knife variable selection using the rest of the default parameters of MORE for PLS models and the same input data we used to create the SimMLR model.

6.7 networkMORE function

MORE package includes the function **networkMORE** to graphically represent the networks concluded by applying **more** function.

```
networkMORE(outputRegpcond, cytoscape = TRUE, group1 = NULL,
            group2 = NULL, pc = 0, pathway = NULL,
            annotation = NULL, save = FALSE)
```

6.7.1 networkMORE input parameters

outputRegpcond Object generated by the application of **RegulationPerCondition** function to a **more** object.

cytoscape If TRUE (default), the function plots the network induced from **more** in *Cytoscape*. For that, it is necessary to install *RCy3* package and to maintain *Cytoscape* software opened while running the function. If FALSE, it plots this network in R using *igraph* package and saves it so that the user can introduce it later to *Cytoscape*. This option is not recommended for plotting huge networks, as the visualization is complex.

group1 Name of the group to take as a reference in the creation of differential networks. If it is not provided, the networks of all groups will be plotted. If it is provided without providing *group2* only the specific network for this group would be created. By default, NULL.

group2 Name of the group to compare to the reference in the differential network creation. If it is not provided, the networks of all groups will be plotted. By default, NULL.

pc Percentile to plot regulations of most pc% affecting regulators of target omics. This value must be selected in the range 0 to 1. By default, 0.

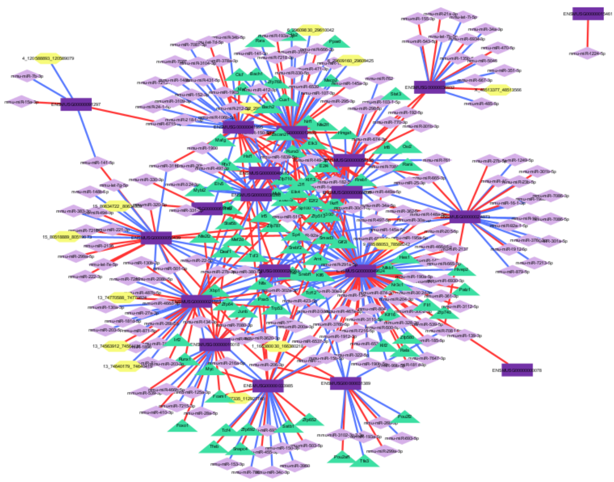
pathway If provided, the function will print the regulatory network involved in the specified pathway instead of the entire regulatory network. The provided pathway name must have the same terminology as in the annotation matrix. By default, NULL.

annotation Annotation matrix with target features in the first column, GO term accession in the second and GO term name in the third. Only necessary when a specific pathway has to be plotted. By default, NULL.

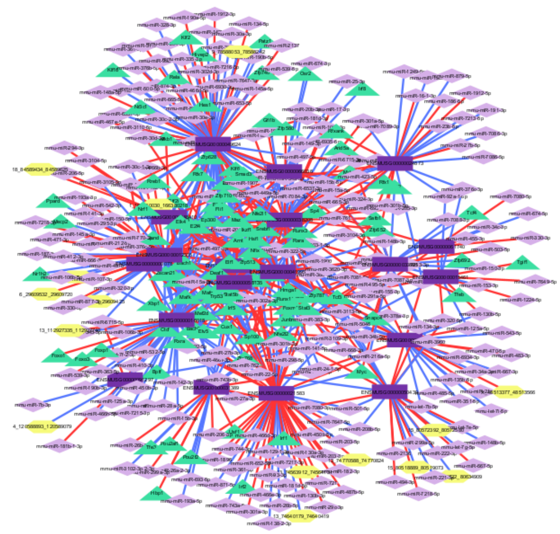
save If TRUE, only when *cytoscape* is set to FALSE, networks are saved in *gml* extension. By default, FALSE.

Following previous example, networks can be plotted by:

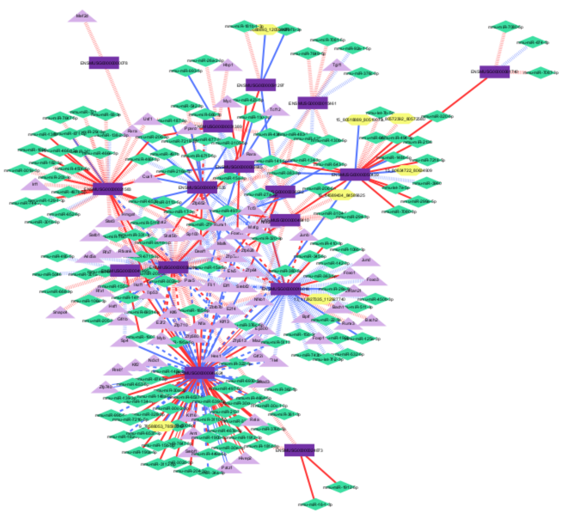
```
networkMORE(myresults, cytoscape = TRUE)
networkMORE(myresults, cytoscape = TRUE, group1 = 'Control',
            group2 = 'Disease')
```



Regulatory network in Control



Regulatory network in Disease



Differential regulatory network
Disease vs. Control

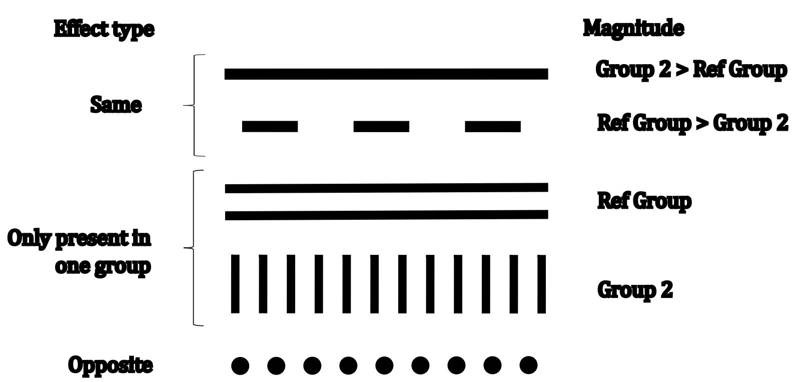
NOTE 1: By default, regulators repressing the target feature expression will be connected by red edges and regulators activating the target feature expression will be connected by blue edges.

NOTE 2: By default, response variable (Gene Expression, protein expression,..), transcription factors (TFs) and microRNAs (miRNAs) will be plotted with rectangle, triangle and diamond shapes, respectively. Considering there is still no consensus in the literature on the shapes of all omics, we applied the ones by default to the rest of the omics. It is up to the user if they want to change it.

NOTE 3: Considering that we want to compare two groups, group1 (reference) and group2, in the differential network creation the different line styles will represent different scenarios.

- When both groups present same type of effect (activator or repressor):
  - When effect of reference is bigger than compared group, dashed lines will be used.
  - When the compared group presents bigger effect than reference, solid lines will be used.
- When one of the groups does not present a regulation but the other does:
  - When reference group is the only one which shows a regulation it will be represented by double parallel lines.
  - When the compared group is the only one which shows a regulation it will be represented by vertical lines.
- When compared groups present opposite type of effect, dotted lines will be used to represent it.

We summarize in the figure below the line types used in MORE differential networks depending on the effect types of the groups and the magnitude of them.



NOTE 4: When really huge networks are to be created, it can be very time consuming the R Cytoscape connection, so we recommend to save the networks and load them in Cytoscape. The user would be able to set their own style for the networks. However, if they want to use the default style of MORE, we recommend to create a small network induced

by a small subset of MORE's **RegulationPerCondition** output and run **networMORE** for that little subset in one condition. MORE's style would be loaded and the following loaded networks will inherit MORE's default style.

NOTE 5: When MLR models are used and the user wants to compare networks of different conditions, they should use the **RegulationPerCondition** tables modified by the **BetaTest**. See that those comparisons should also be in contrast to the reference group. Suppose the comparison of groups not involving the reference group is to be done. In that case, the user should rerun **more**, changing the reference group to one of those, calculate **RegulationPerCondition** table, use the **BetaTest** and finally compare those groups by the differential networks.

7 Downstream analysis

Apart from creating networks, the MORE package also includes several other functionalities to do a complete downstream analysis and, in that way, enable researchers to understand the biological systems under study.

7.1 Over Representation Analysis

The Over Representation Analysis (ORA) methodology determines if a particular biological annotation is significantly enriched in a target set of genes compared to a reference set. In order to apply this methodology, in MORE, we include **oraMORE** function. In MORE, we can either use the target set of genes as the target features pointed as hub target features or the target features regulated by the regulators pointed as global regulators in a specific condition by MORE analysis.

```
oraMORE(output, outputRegincond, byHubs = TRUE, byOmic = NULL,
        annotation, alpha = 0.05, p.adjust.method = "fdr")
```

7.1.1 oraMORE input parameters

output Object generated by the function **more**.

outputRegincond Object generated by the function **RegulationInCondition**.

byHubs Indicates whether to perform the ORA for the Hub target features, TRUE, or for the target features regulated by the global regulators, FALSE. By default, TRUE.

byOmic If provided, it performs the ORA to the regulators of the specified omic (it must follow the same nomenclature that in regulatoryData). Incompatible with other methodologies specified in byHubs parameter. By default, NULL.

annotation Annotation matrix with genes in the first column, GO term accession in the second and GO term name in the third.

alpha p-value cutoff to consider. By default, 0.05.

p.adjust.method p-value adjustment method to consider. By default, 'fdr'.

7.2 Gene Set Enrichment Analysis

Gene Set Enrichment Analysis (GSEA) evaluates whether a set of genes shows statistically significant differences between two experimental conditions. In MORE we center the analysis of the differences in the number of regulators of the genes in the experimental conditions. **gseaMORE** function performs GSEA analysis to MORE output making use of *clusterProfiler* package, so that all plots included in that package could be applied to the output object of running that function.

```
gseaMORE(outputRegincond, outputRegincond2, annotation, alpha = 0.05,
         p.adjust.method = "fdr")
```

7.2.1 gseaMORE input parameters

outputRegincond Object generated by the function **RegulationInCondition**.

outputRegincond2 Object generated by the function **RegulationInCondition** for another group different from the previously considered. By default, NULL. If NULL, the analysis will be centered only in the study group, so the results will apply only to that group and will not mean the statistical differences between groups. If provided **MORE** computes and internal score comparing the number of significant regulators in both groups.

annotation Annotation matrix with genes in the first column, GO term accession in the second and GO term names in the third.

alpha p-value cutoff to consider. By default, 0.05.

p.adjust.method p-value adjustment method to consider. By default, 'fdr'.

NOTE 5: In the case where no condition is provided, if the user wants to use functions that require **RegulationInCondition** as input, instead, it must provide the output of **RegulationPerCondition**.

8 How to cite MORE package

Aguerralde-Martin, Maider; Tomás-Riquelme, Blanca; Clemente-Císcar, Mónica; Conesa, Ana; Tarazona, Sonia. (2023). MORE: Multi-Omics REgulation. R package version 1.0.