Normalization of Mass Spectrometry Data (NOMAD)

Carl Murie, Ola Larsson

Contents

1 Introduction

Identifying and quantifying protein abundances in biological samples is an important step when identifying differential protein expression from proteome wide studies. Tandem mass spectrometry (MS/MS)-based quantification using the iTRAQ isobaric tagging reagent is a commonly used technology for this purpose. iTRAQ MS allows for simultaneous quantitative comparison of protein abundance across multiple samples within a single MS run. Consequently this technology offers challenges in normalization and analysis due to potential sources of variation from sample preparation, protein concentrations, and the experimental process. In particular, samples that are run across different MS runs require normalization that enables across MS run comparison.

A common method of normalization is to use one of the iTRAQ channels as a pooled reference sample to provide relative quantitation between the other channels. For each protein a ratio of each sample over the experiments reference is used as the abundance score for that protein. There are at least two problems with the use of pooled reference samples. One is that at least one of the iTRAQ channels is used for a sample that is not of biological interest, which reduces the throughput of the assay. Another is that the variability of a protein abundance is doubled due to the abundance ratio being a product of two samples (Kerr et al. 2000).

Alternatively, an ANOVA approach, which is a simplified form of linear regression, has been proposed and is particularly suited for mass spectrometry assays that require multiple MS runs (Hill et al., 2008; Oberg et al., 2008). Sources of bias, such as MS run, iTRAQ channel, protein identifiers, and peptide identifiers can be used as regression factors in the ANOVA model. The residuals of the ANOVA model are the peptide abundance scores after removing the effects of the regression factors. They can then be used as unbiased peptide measurements to calculate the summary protein abundances. A significant issue for multiple MS run data sets is the computational complexity of solving the ANOVA linear model. Oberg et al, 2008 discuss this issue at length and provide possible solutions such as iterative or stagewise regression. Both of these potential solutions suffer from technical and computational limitations (Oberg et al., 2008). To date, there is no software designed for MS iTRAQ assays that implements an ANOVA approach in an accessible and efficient manner. Here we provide the NOMAD (NOrmalization for MAss spectrometry Data) R package that implements an ANOVA normalization method designed for iTRAQ mass spectrometry data in a computationally efficient manner.

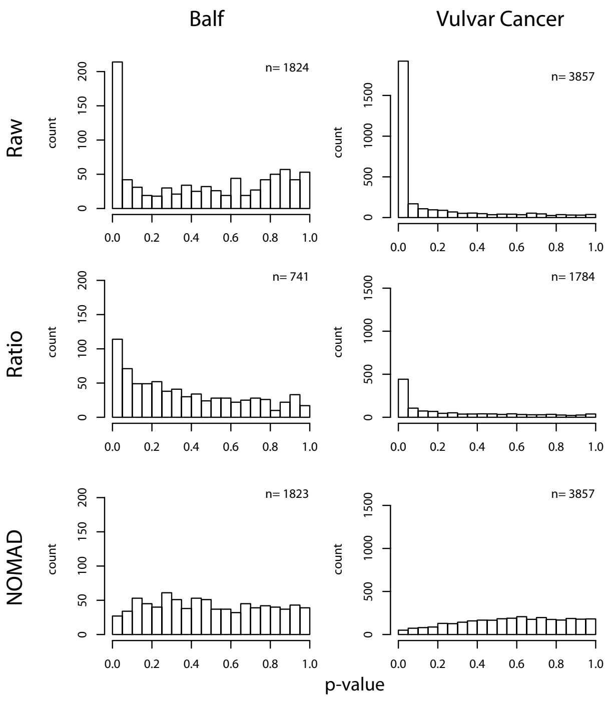

Figure 1: *Output of nomadCheckBias applied to the raw, reference normalized, and NOMAD normalized data from two MS runs. It shows the histograms of p-values produced from testing for a MS run effect using "Run" as the factor of interest. The raw data (first row) show a strong MS run effect which reduces the efficacy of protein abundance comparisons across MS runs. The ratio normalization method reduces the MS run bias but the NOMAD normalization removes the MS run bias entirely.*

2 Normalization

First peptide abundances are normalized using the *nomadNormalization* function.

2.1 Parameters

There are two required parameters.

1. *y*: Vector of peptide abundances.
2. *x*: A data.frame defining the experimental design. Each row of *x* will define the experimental information for the same row of *y*. The data.frame *x* must contain the following columns labelled Protein, Peptide, Run, iTRAQ, and, optionally, iTRAQCorrectionIndex.

    - Peptide: Defines the peptide sequence. This can be a unique identifier or the actual peptide sequence.
    - Run: Defines the MS run that generated the peptide abundance.
    - iTRAQ: Defines which iTRAQ channel this peptide abundance was assigned to.
    - iTRAQCorrectionIndex: This column is necessary if iTRAQ isotope correction is desired. Each peptide abundance is generated from a particular spectrum for each run and each unique spectrum should contain an abundance score for each iTRAQ used in the assay. For example, if an assay uses 8 iTRAQs for a run then each spectrum should generate 8 abundances. The iTRAQCorrectionIndex should uniquely define each set of iTRAQs generated for a single spectrum. It may be possible to use the spectrum identifier for this purpose. However there have been cases where the same protein and peptide, although different isoforms, are present in different spectrums in the same run. In this case the spectrum cannot be used as it is not a unique identifier for the iTRAQs.

Other optional parameters are as follows:

1. *factors*: Optional list of experimental factors which will be used to normalize the data. The default value is NULL which will result in the use of the following factors for normalization: Peptide, Run, iTRAQ, Peptide & Run interaction, and iTRAQ & Run interaction. The user may add to these factors with those of their own choosing as long as these factors are defined in *x*.

2. *doRobust*: If TRUE then a robust approach to normalization is used (medians rather than means of levels are used). The default is TRUE.

3. *doLog*: If TRUE then the data is log2 transformed. The default is TRUE as it is generally advised to log transform the data. The checkLogTransform function can be used to decide whether to log transform the data.

4. *doiTRAQCorrection*: If TRUE then a correction for iTRAQ isotope signal overlap will be performed. If 4 or 8 iTRAQs were used then an iTRAQCorrectionTable is available by default. If the number of iTRAQs is not 4 or 8 then the user must supply the iTRAQCorrectionTable.

5. *iTRAQCorrectionTable*: An n by n matrix where n is the number of iTRAQs used. Each row corresponds to the ordered iTRAQs and shows the proportion of each iTRAQ signal that contributes to that iTRAQs abundance score. An example of an iTRAQCorrectionTable is found in Appendix A

The default and recommended call of *nomadNormalization* uses three main effects (Peptide, Run, iTRAQ) and two interaction effects(Peptide & Run, iTRAQ & Run). Notice that this requires either a balanced or randomized experimental design. A robust measure (median) is used when calculating normalization values and a log2 transformation of the data is applied before normalization. iTRAQCorrection is not done by default although if 4 or 8 iTRAQs are used then setting the doiTRAQCorrection parameter to TRUE will apply the correction. Generally the consensus is that iTRAQ-Correction should be done, but we have found that such correction will often introduce negative values (resulting in NAs) which questions the validity of this "background correction".

2.2 nomadNormalization function call

```
## read in data
data(BalfPeptides)

## convert to NOMAD inputs
y <- BalfPeptides$Abundance
```

```
x <- BalfPeptides[,c(1,2,3,4,6) ]
```

An example of the format of y and x from the BalfPeptides data follows:

```
y
[1]    1869.56
[2]    4245.72
[3]    21305.83
...
[5375] 2062.97

  x$Protein x$Peptide    x$Run      x$iTRAQ x$iTRAQCorrectionIndex
[1]    ALB  DDNPNLPR      1           1        2.1.1.2396.1
[2]    ALB  HPDYSVVLLLR   1           1        4.1.1.7634.1
[3]    ALB  LDELRDEGK     1           1        5.1.1.4815.1
...
[5375] CPA4  GLASIPR      3           8        8.1.1.4111.1
```

The normalization function is called as follows:

```
## default call
results <- nomadNormalization(y, x)

## default call but with iTRAQCorrection (assuming 4 or 8 iTRAQs)
results <- nomadNormalization(y, x, doiTRAQCorrection=TRUE,
                              iTRAQCorrectionTable=NULL)
```

The *nomadNormalization* function returns a list with four elements.

1. y: Vector of normalised peptide abundances. Missing data must be represented by NAs.
2. x: data.frame containing the same columns as the input x parameter except that the iTRAQCorrectionIndex will have been removed if present. Missing data must be represented by NAs.
3. *removedData:* Vector of line indices that were removed from the original y and x data. All lines that contain an NA or have missing data for y or x will be removed.

4. *call*: A listing of the parameters that were used in the normalization call.

y and x may have a lesser number of values after normalization as negative infinite values may be generated and removed due to log transformation. All rows containing an NA or missing data will be removed. Additionally iTRAQCorrection may also produce negative peptide abundances that will be discarded. The return parameter "removedData" show the line numbers that were removed.

2.3 nomadNormalization function call with user added factors

A user may wish to change or add to the factors used in NOMAD normalization. For example a researcher may believe that the iTRAQ factor is not necessary and wishes to remove the iTRAQ and iTRAQ & Run interaction factors. Additionally, their lab has a "Sloppy_Joe" factor where they are concerned that a particular technician may affect iTRAQ channels 1 to 4 differently than 5 to 8. An additional column named "Sloppy_Joe" must be added to x where rows associated with iTRAQ channels 1-4 and 5-8 are assigned 0 and 1, respectively. A list defining the factors to be used in the normalization must be passed to *nomadNormalization*.

```
## define new list of factors
myFactors <- list("Peptide", "Run", "Sloppy_Joe",
                           c("Run", "Peptide"))

## add Sloppy_Joe column to "x"
len <- dim(x)/2
Sloppy_Joe <- rep(0, dim(x)[[1]])
Sloppy_Joe[x$iTRAQ >= 5] <- 1
x$Sloppy_Joe <- Sloppy_Joe

results <- nomadNormalization(y, x, factors=myFactors)
```

3 Protein Assembly

The second step is to assemble the protein scores from the normalized peptide abundances with the *nomadAssembleProteins* function.

3.1 Parameters

As with *nomadNormalization* there are two required parameters.

1. *y*: Vector of peptide abundances.

2. *x*: A data.frame defining the experimental design. Each row of *x* will define the experimental information for the same row of *y*. The data.frame *x* must contain the following columns labelled Peptide, Protein, Run, and iTRAQ. These columns are the same format as the *x* parameter in *nomadNormalization*. Missing data must be represented by NAs.

Other optional parameters are as follows:

1. *method*: Defines the method that is used to summarize the peptide abundances for each protein. The default is the robust measure Tukey biweight ("TukeyBW"). Alternatively, "mean" or "median" can also be used.

2. *format*: Defines the format of the output. The default is "bySample" where each row is a unique protein and the columns are the abundances (or NA if not present for that MS run) for each sample (one column for each sample). The alternative is "byProtein" where the columns are abundance, MS run and iTRAQ and each row is a protein.

3. *combineDupPeptides*: It is possible that there are duplicated peptide measurements. If this parameter is TRUE the average (based on method parameter) of the duplicated peptides is used as a single measurement of that peptide. If FALSE then each duplicated peptide measurement is used as an independent measure. The default is FALSE as it is likely that the duplicated peptides are indeed independent.

4. *standardizeAbundance*: If TRUE then transform each sample (each unique combination of Run and iTRAQ) with a robust Z adjustment. The default is FALSE.

5. *mySep*: A character string used as string separator that must not exist in any factor in *x*. We advise not to change this unless an error message is generated telling the user that the separator is found in the data.

3.2 nomadAssembleProteins function call

The output of *nomadNormalization* should be used as the inputs to *nomadAssembleProteins*. From the return list (*results*) generated from the *nomadNormalization* in the previous example 2.2 one can use:

```
proteinScores <- nomadAssembleProteins(results$y, results$x)
```

The *nomadAssembleProteins* function returns a list with two elements.

1. *scores*: Data.frame containing the output whose format is defined by the "format" input parameter.
2. *call*: A listing of the parameters that were used in the normalization call.

4 Diagnostics

Two plotting functions are included.

4.1 nomadCheckLogTransform

The ANOVA approach assumes a constant (homogeneous) variance across peptide signal for the normalization to to be valid. Our experience with mass spectrometry data is that, in its raw form, the variance is heterogenous and after log transformation the variance is closer to homogenous. This function plots the effect of a log2 transformation of the peptide abundances. Mean versus standard deviation plots for both peptide and protein abundances are shown to determine homogeneity of variance. The protein abundances are calculated post-log based on the *nomadAssembleProteins* parameters. The homogeneity assumption is only necessary for the peptide abundances thus the protein abundance plots are shown only for interest.

4.1.1 Parameters

Note that y and x are identical to the inputs in *nomadNormalization*.

1. y: Vector of peptide abundances.
2. x: A data.frame defining the experimental design. Each row of x will define the experimental information for the same row of y. The data.frame x must contain the following columns labelled Peptide, Protein, Run, and iTRAQ. These columns are the same format as x in *nomadNormalization*.

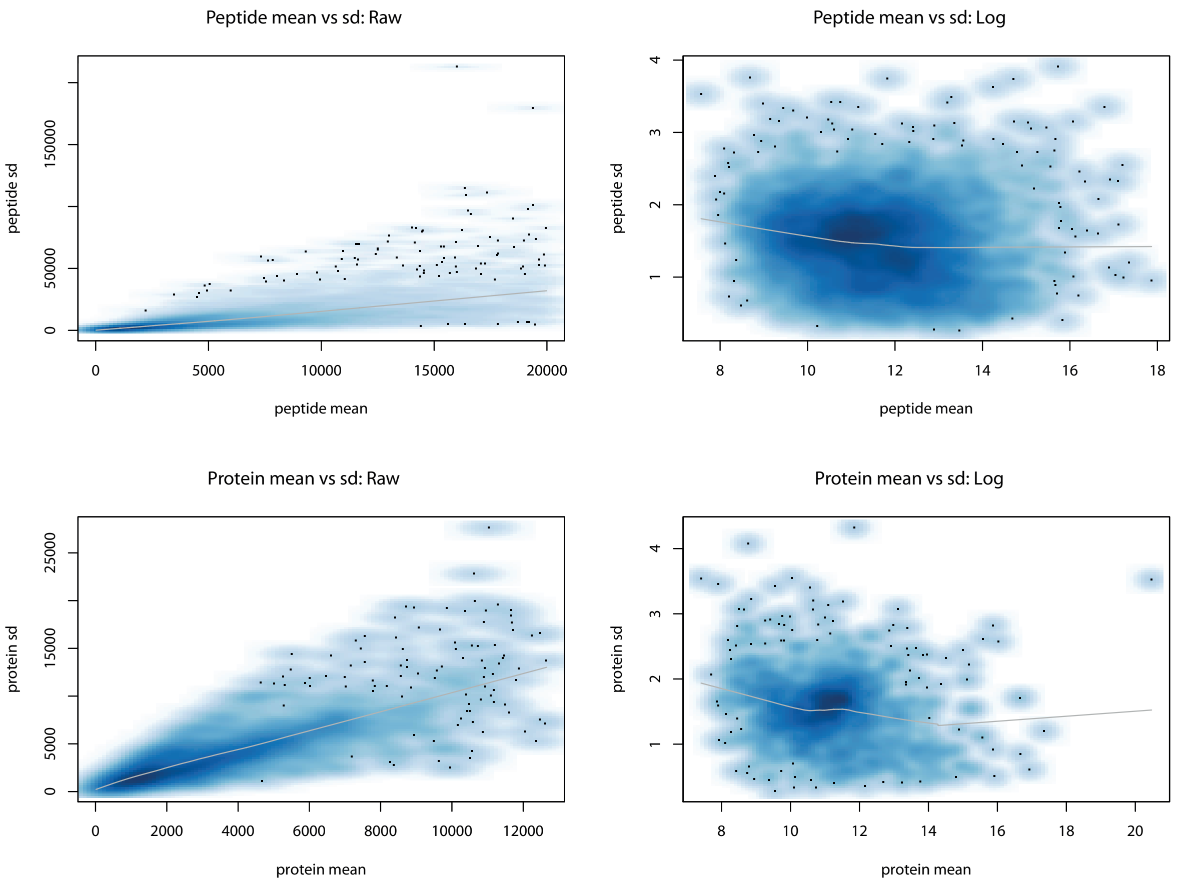

Figure 2: *Output of checkLogTransform. Note that the variance is much more homogenous across peptide abundance means after the log transformation.*

3. *method*: Defines the method that is used to summarize the peptide abundances for each protein. The default is the robust measure Tukey biweight ("TukeyBW"). Alternatively, "mean" or "median" can also be used.

4. *rawTrim*: Quantile value (between 0 and 1) used to reduce the scale for plotting raw data. Will plot raw means up to the specified quantile (default=1). Set *rawTrim* to lower values (say 0.9 or 0.8) to reduce the scale and better view data.

4.1.2 nomadCheckLogTransform function call

```
## read in data
data(BalfPeptides)

## convert to NOMAD inputs
y <- BalfPeptides$Abundance
x <- BalfPeptides[,c(1:4,6)]

## check log transform function
nomadCheckLogTransform(y, x, rawTrim=0.9)
```

Figure 2 shows that the raw peptide abundances have an increasing variance as the peptide abundance increases while the logged abundances have a relatively flat variance across the abundance range.

4.2 nomadCheckBias

Apply statistical testing for a particular normalization factor to check whether that factor is a source of bias. A two-way test based on the supplied factor is applied to each protein across all samples and a histogram of the distribution of p-values is plotted. For example, one can check whether a MS run effect exists and how well NOMAD addresses it. Figure 1 shows the output of *nomadCheckBias* applied to raw, reference normalized, and NOMAD normalized MS data with "Run" as the factor of interest to two MS data sets. The figure shows a strong batch effect for both data sets that is partially addressed by reference normalization and completely removed with NOMAD normalization.

4.2.1 Parameters

1. *dat*: A matrix where rows are proteins and columns are samples (each iTRAQ sample per day). This is the same format as the "*scores*" output of *nomadAssembleProteins* with format=bySample. Missing data must be represented by NAs.

2. *fact*: Vector of labels for defining the levels of the factor of interest. For example, the protein abundances file generated with BalfPeptides from *nomadAssembleProteins* (protaov in the example below) consists of 3 MS runs with each MS run containing 8 iTRAQ channels. The columns are ordered so that the first 8 columns are the 8 iTRAQ channels for the first MS run, the next 8 columns are the 8 iTRAQ channels for the second MS run, and so on. To examine the MS run factor a vector of labels must be constructed so that each MS run (8 columns) has a unique identifier. In this case a vector such as c(1,1,1,1,1,1,1,1, 2,2,2,2,2,2,2,2,3,3,3,3,3,3,3,3) can be used. To examine the iTRAQ factor the vector of labels must uniquely identify each iTRAQ. A vector such as c(1,2,3,4,5,6,7,8,1,2,3,4,5,6,7,8,1,2,3,4,5,6,7,8) can be used.

3. *numNAs*: Number of missing data points allowed. Defaults to sample size of dat.

4. *label*: Text to describe factor. It is used in plot title.

5. *yMax*: Upper range for plotting histograms. Useful for putting multiple *checkBias* plots on same scale.

6. *showSampleSize*: If TRUE, then add text box showing the sample size. default is TRUE.

4.2.2 nomadCheckBias function call

```
## read in data
data(BalfPeptides)

## convert to NOMAD inputs
y <- BalfPeptides$Abundance
x <- BalfPeptides[,c(1:4,6)]

## apply NOMAD normalization
res <- nomadNormalization(y, x, doRobust=TRUE, doLog=TRUE)
```

```
## apply protein assembly
protaov  <- nomadAssembleProteins(y=res$y, x=res$x, method="TukeyBW",
                         format="BySample", combineDupPeptides=FALSE,
                         standardizeAbundance=FALSE)

## apply protein assembly to non-normalized data
protaovRaw  <- nomadAssembleProteins(y=log2(y), x=x, method="TukeyBW",
                    format="BySample", combineDupPeptides=FALSE,
                    standardizeAbundance=FALSE)

## protaov and protaovRaw files are organized such that each column is
## one iTRAQ channel (8 iTRAQ channels for each of the 3
## runs, or days).
runInd <- rep(1:3, each=8)    ## look at bias due to run effect
itraqInd <- rep(1:8, 3)       ## look at bias due to iTRAQ channel

## plot run and itraq biases for normalized and non-normalized data.
par(mfrow=c(2,2))
nomadCheckBias(dat=protaovRaw$scores, fact=runInd, numNAs=12,
               label="MS run: Raw")
nomadCheckBias(dat=protaov$scores, fact=runInd, numNAs=12,
               label="MS run: NOMAD")
nomadCheckBias(dat=protaovRaw$scores, fact=itraqInd, numNAs=12,
               label="iTRAQ: Raw")
nomadCheckBias(dat=protaov$scores, fact=itraqInd, numNAs=12,
               label="iTRAQ: NOMAD")
```

5 Data Format

The first step of NOMAD analysis is to reformat the peptide level output of the MS quantification software (e.ge. Protein Pilot) so it can be used as input to NOMAD. A small subset of the Balf data, which was obtained from three MS runs, is included in the package as an example. Shown below is R code that reformats the Balf data. Each MS run is stored in a separate file (Balf1.txt, Balf2.txt, Balf3.txt). This section of code reads in the data files and writes a single file that is of the proper format to be used with the NOMAD R package.

```
 ## load the three peptide files for parsing
```

```
data(Balf1, Balf2, Balf3)

## columns to keep in file: index, protein annotation, peptide
## sequence, spectrum, and abundance scores.
colInd <- c(1,4,9,12,13,15,17,19,21,23,25, 27)

## keep track of each peptides 8 iTRAQ values (one row) for possible
## use in correcting iTRAQ isotope overlap.
iTRAQCount <- 0

Balfs <- list(Balf1, Balf2, Balf3)   ## put final data set in this variable
allDat <- NULL

## loop through the three files
for(i in 1:3) {

    ## get a single Balf file
    tmp <- Balfs[[i]]

    ## filter peptides according to whatever rules one has. In
    ## this case we require a confidence greater than 0.95 and
    ## a "used" value of 1
    rowInd <- as.numeric(tmp[,8]) >= 0.95 & as.numeric(tmp[,5]) == 1

    ## keep only the rows and columns we want
    filtDat <- tmp[rowInd, colInd]

    ## reformat so each row is a single peptide with columns
    ## defining protein name, peptide sequence, run, itraq, and
    ## peptide abundance.

    ## inefficient re-formatting to show steps. Loop through each
    ## row and reformat. Each row will be put into a 8 by 4 matrix
    ## (8 iTRAQS for a single peptide)
    formDat <- NULL
    for(row in 1:dim(tmp)[1]) {

        iTRAQCount <- iTRAQCount + 1   # unique id for this row

        protein <- rep(filtDat[row,2], 8)   ## protein label
        peptide <- rep(filtDat[row,3], 8)   ## peptide label
        exp <- rep(i, 8)                     ## run label
        itraq <- 1:8                         ## itraqs
        itraqCount <- rep(iTRAQCount, 8)     ## peptide itraq ID

        ## abundance for each itraq (1 to 8)
        abundances <- as.numeric(filtDat[row, 5:12])

        thisRow <- cbind(protein, peptide, exp, itraq, itraqCount,
                         abundances)

        formDat <- rbind(formDat, thisRow)

    } ## end for row

    ## append each file to final data variable
    allDat <- rbind(allDat, formDat)

} ## end for i

## create column names. The column names must be labeled with the
## following column names (except Abundance)
colnames(allDat) <- c("Protein", "Peptide", "Run", "iTRAQ",
                      "iTRAQCorrectionIndex", "Abundance")

inPath <- "/some/path/to/directory/"
write.table(allDat, paste(inPath, "BalfPeptidesSubset.txt", sep=""),
            sep="\t")
```

6 Appendix

A iTRAQCorrectionTable example

The iTRAQCorrectionTable is a matrix of dimension n by n where n is the number of iTRAQs used in the assay. Each row corresponds to the ordered iTRAQs and shows the proportion of each iTRAQ signal that contributes to that iTRAQs abundance score.

```
iTRAQTable <- rbind(
   c(.9287, 0.0689, .0024,  0,    0,     0,      0,     0),
   c(.0094,   0.93,   .059,  .0016, 0,      0,      0,     0),
   c(0,     .0188,  .9312, .049,  .001,  0,      0,     0),
   c(0,     0,      .0282, .9321, .039,  0.0007, 0,     0),
   c(0,     0,      .0006, .0377, .9329, .0288,  0,     0),
   c(0,     0,      0,     .0009, .0471, .9329, .0191, 0),
   c(0,     0,      0,     0,     .0014, .0566,  .9333, .0087),
   c(0,     0,      0,     0,     0,     .0027,  .0744, .9211)
 )
```

References

[1] Kerr K., Mitchell M., & Churchill G., *Analysis of variance for gene expression microarray data*, Journal of computational biology, Vol 7, 6, pgs. 819-837, (2000)

[2] Hill, E. et al., *A statistical model for iTRAQ data analysis*, Journal of proteome research, vol 7, 8, pgs. 3091-3101 (2008)

[3] Oberg A. et al., *Statistical analysis of relative labeled mass spectrometry data from complex samples using ANOVA*, Journal of proteome research, Vol 7, pgs 225-233, 1, (2008)

[4] Kerr, K., & Churchill, G., *Statistical design and the analysis of gene expression microarray data*, Genetical research, Vol 77, 2, pgs. 123-128, (2001)