Metabolomic Data Analysis with MetaboAnalyst 5.0

Name: guest16468220247862756278

December 6, 2023

1 Data Processing and Normalization

1.1 Reading and Processing the Raw Data

MetaboAnalyst accepts a variety of data types generated in metabolomic studies, including compound concentration data, binned NMR/MS spectra data, NMR/MS peak list data, as well as MS spectra (NetCDF, mzXML, mzDATA). Users need to specify the data types when uploading their data in order for MetaboAnalyst to select the correct algorithm to process them. Table 1 summarizes the result of the data processing steps.

1.1.1 Reading Peak Intensity Table

The peak intensity table should be uploaded in comma separated values (.csv) format. Samples can be in rows or columns, with class labels immediately following the sample IDs.

Samples are in rows and features in columns The uploaded file is in comma separated values (.csv) format. The uploaded data file contains 54 (samples) by 35 (peaks(mz/rt)) data matrix.

1.1.2 Data Integrity Check

Before data analysis, a data integrity check is performed to make sure that all the necessary information has been collected. The class labels must be present and contain only two classes. If samples are paired, the class label must be from -n/2 to -1 for one group, and 1 to n/2 for the other group (n is the sample number and must be an even number). Class labels with same absolute value are assumed to be pairs. Compound concentration or peak intensity values should all be non-negative numbers. By default, all missing values, zeros and negative values will be replaced by the half of the minimum positive value found within the data (see next section)

1.1.3 Missing value imputations

Too many zeroes or missing values will cause difficulties for downstream analysis. MetaboAnalyst offers several different methods for this purpose. The default method replaces all the missing and zero values with a small values (the half of the minimum positive values in the original data) assuming to be the detection limit. The assumption of this approach is that most missing values are caused by low abundance metabolites (i.e.below the detection limit). In addition, since zero values may cause problem for data normalization (i.e. log), they are also replaced with this small value. User can also specify other methods, such as replace by mean/median, or use K-Nearest Neighbours (KNN), Probabilistic PCA (PPCA), Bayesian PCA (BPCA) method, Singular Value Decomposition (SVD) method to impute the missing values [1]. Please choose the one that is the most appropriate for your data.

[1] Stacklies W, Redestig H, Scholz M, Walther D, Selbig J. *pcaMethods: a bioconductor package, providing PCA methods for incomplete data.*, Bioinformatics 2007 23(9):1164-1167

Zero or missing values were replaced by 1/5 of the min positive value for each variable.

1.1.4 Data Filtering

The purpose of the data filtering is to identify and remove variables that are unlikely to be of use when modeling the data. No phenotype information are used in the filtering process, so the result can be used with any downstream analysis. This step can usually improves the results. Data filter is strongly recommended for datasets with large number of variables ($>$ 250) datasets contain much noise (i.e.chemometrics data). Filtering can usually improve your results[2].

For data with number of variables < 250, this step will reduce 5% of variables; For variable number between 250 and 500, 10% of variables will be removed; For variable number bwteen 500 and 1000, 25% of variables will be removed; And 40% of variabled will be removed for data with over 1000 variables. The None option is only for less than 5000 features. Over that, if you choose None, the IQR filter will still be applied. In addition, the maximum allowed number of variables is ***10000***

No data filtering is performed.

[2]Hackstadt AJ, Hess AM.*Filtering for increased power for microarray data analysis*, BMC Bioinformatics. 2009; 10: 11.

Table 1: Summary of data processing results

	Features (positive)	Missing/Zero	Features (processed)
BLANK_2_Dup1	35	0	35
BLANK_2_Dup2	35	0	35
BLANK_2_Dup3	35	0	35
BLANK_2	35	0	35
C12.1	35	0	35
C12.3	35	0	35
C12.4	35	0	35
C15.1	35	0	35
C15.2	35	0	35
C15.3	35	0	35
C15.4	35	0	35
C9.1	35	0	35
C9.2	35	0	35
C9.3	35	0	35
C9.4	35	0	35
D12.1	35	0	35
D12.2	35	0	35
D12.3	35	0	35
D12.4	35	0	35
D15.1	35	0	35
D15.2	35	0	35
D15.3	35	0	35
D15.4	35	0	35
D9.1	35	0	35
D9.2	35	0	35
D9.3	35	0	35
D9.4	35	0	35
F12.1	35	0	35
F12.2	35	0	35
F12.3	35	0	35
F12.4	35	0	35
F15.1	35	0	35
F15.2	35	0	35
F15.3	35	0	35
F15.4	35	0	35
F9.1	35	0	35
F9.2	35	0	35
F9.3	35	0	35
F9.4	35	0	35
QC.1.Dup	35	0	35
QC.1	35	0	35
QC.2	35	0	35
QC.3	35	0	35
X12.1	35	0	35
X12.2	35	0	35
X12.3	35	0	35
X15.1	35	0	35
X15.2	35	0	35
X15.3	35	0	35
X15.4	35	0	35
X9.1	35	0	35
X9.2	35	0	35
X9.3	35	0	35
X9.4	35	0	35

1.2 Data Normalization

The data is stored as a table with one sample per row and one variable (bin/peak/metabolite) per column. The normalization procedures implemented below are grouped into four categories. Sample specific normalization allows users to manually adjust concentrations based on biological inputs (i.e. volume, mass); row-wise normalization allows general-purpose adjustment for differences among samples; data transformation and scaling are two different approaches to make features more comparable. You can use one or combine both to achieve better results.

The normalization consists of the following options:

1. Row-wise procedures:
   - Sample specific normalization (i.e. normalize by dry weight, volume)
   - Normalization by the sum
   - Normalization by the sample median
   - Normalization by a reference sample (probabilistic quotient normalization)[3]
   - Normalization by a pooled or average sample from a particular group
   - Normalization by a reference feature (i.e. creatinine, internal control)
   - Quantile normalization
2. Data transformation :
   - Log transformation (base 10)
   - Square root transformation
   - Cube root transformation
3. Data scaling:
   - Mean centering (mean-centered only)
   - Auto scaling (mean-centered and divided by standard deviation of each variable)
   - Pareto scaling (mean-centered and divided by the square root of standard deviation of each variable)
   - Range scaling (mean-centered and divided by the value range of each variable)

Figure 1 shows the effects before and after normalization.

[3]Dieterle F, Ross A, Schlotterbeck G, Senn H. *Probabilistic quotient normalization as robust method to account for dilution of complex biological mixtures. Application in 1H NMR metabonomics*, 2006, Anal Chem 78 (13);4281 - 4290

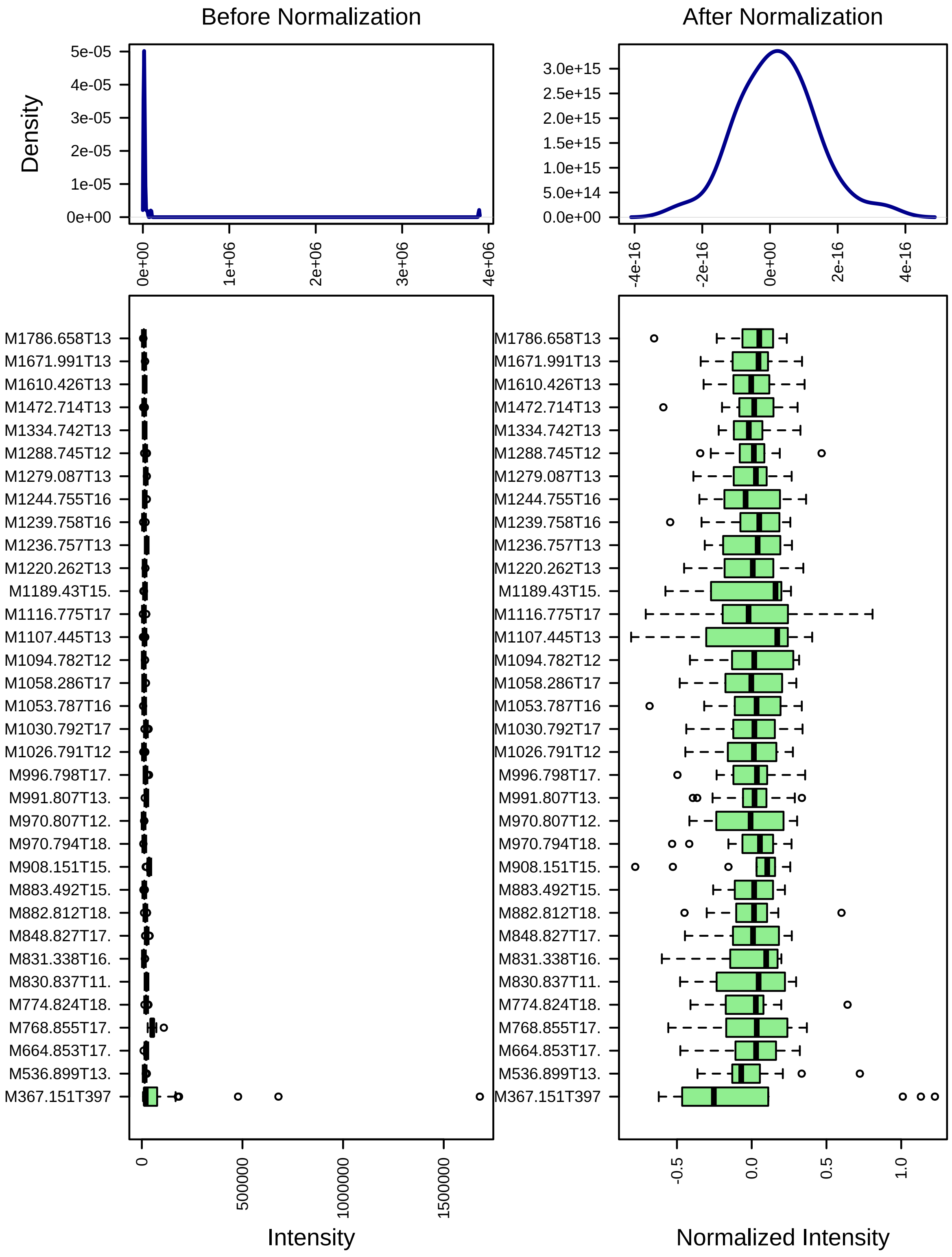

Figure 1: Box plots and kernel density plots before and after normalization. The boxplots show at most 50 features due to space limit. The density plots are based on all samples. Selected methods : Row-wise normalization: Normalization by a reference feature; Data transformation: Log10 Normalization; Data scaling: Pareto Scaling.

2 Statistical and Machine Learning Data Analysis

MetaboAnalyst offers a variety of methods commonly used in metabolomic data analyses. They include:

1. Univariate analysis methods:
   - Fold Change Analysis
   - T-tests
   - Volcano Plot
   - One-way ANOVA and post-hoc analysis
   - Correlation analysis
2. Multivariate analysis methods:
   - Principal Component Analysis (PCA)
   - Partial Least Squares - Discriminant Analysis (PLS-DA)
3. Robust Feature Selection Methods in microarray studies
   - Significance Analysis of Microarray (SAM)
   - Empirical Bayesian Analysis of Microarray (EBAM)
4. Clustering Analysis
   - Hierarchical Clustering
     - Dendrogram
     - Heatmap
   - Partitional Clustering
     - K-means Clustering
     - Self-Organizing Map (SOM)
5. Supervised Classification and Feature Selection methods
   - Random Forest
   - Support Vector Machine (SVM)

```
Please note:  some advanced methods are available only for two-group sample analyais.
```

2.1 One-way ANOVA

Univariate analysis methods are the most common methods used for exploratory data analysis. For multi-group analysis, MetaboAnalyst provides one-way Analysis of Variance (ANOVA). As ANOVA only tells whether the overall comparison is significant or not, it is usually followed by post-hoc analyses in order to identify which two levels are different. MetaboAnalyst provides two most commonly used methods for this purpose - Fisher's least significant difference method (Fisher's LSD) and Tukey's Honestly Significant Difference (Tukey's HSD). The univariate analyses provide a preliminary overview about features that are potentially significant in discriminating the conditions under study.

Figure 2 shows the important features identified by ANOVA analysis.

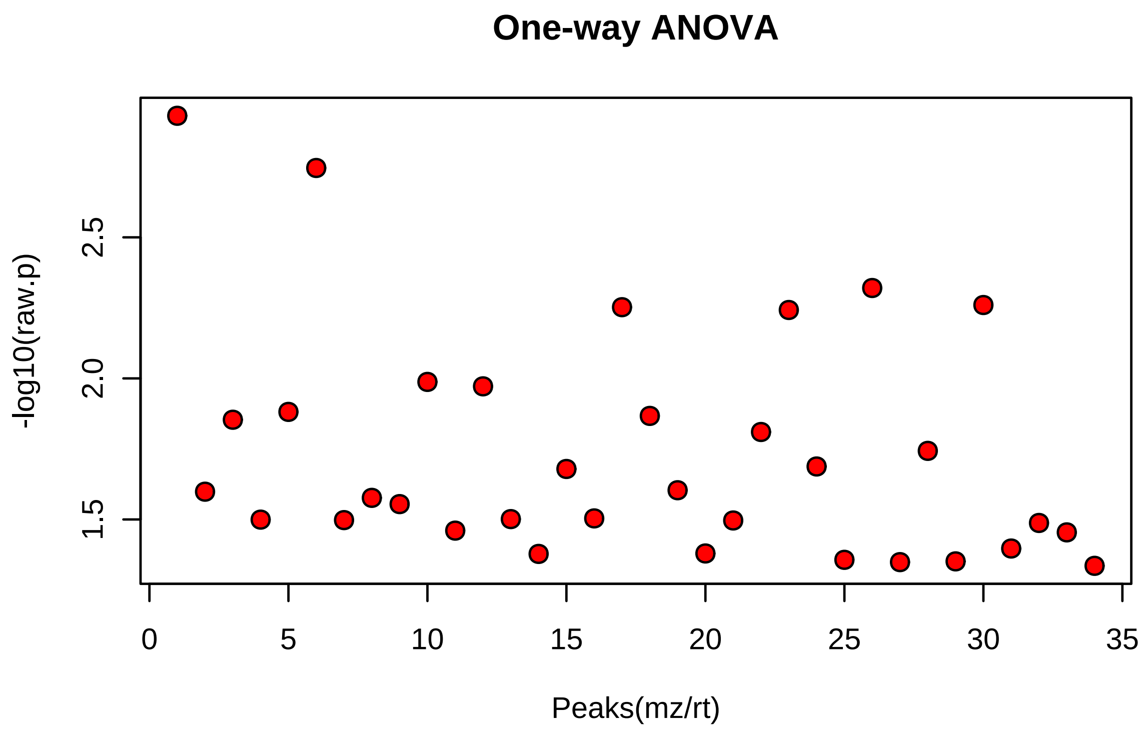

Figure 2: Important features selected by ANOVA plot with p value threshold 0.05.

[1] "No significant features were found using the given threshold for One-way ANOVA and post-hoc analysis"

2.2 Principal Component Analysis (PCA)

PCA is an unsupervised method aiming to find the directions that best explain the variance in a data set (X) without referring to class labels (Y). The data are summarized into much fewer variables called *scores* which are weighted average of the original variables. The weighting profiles are called *loadings*. The PCA analysis is performed using the `prcomp` package. The calculation is based on singular value decomposition.

The Rscript `chemometrics.R` is required. Figure 3 is pairwise score plots providing an overview of the various seperation patterns among the most significant PCs; Figure 4 is the scree plot showing the variances explained by the selected PCs; Figure 5 shows the 2-D scores plot between selected PCs; Figure 6 shows the biplot between the selected PCs. Interactive 3-D scores plots are not included here and can be directly downloaded from website.

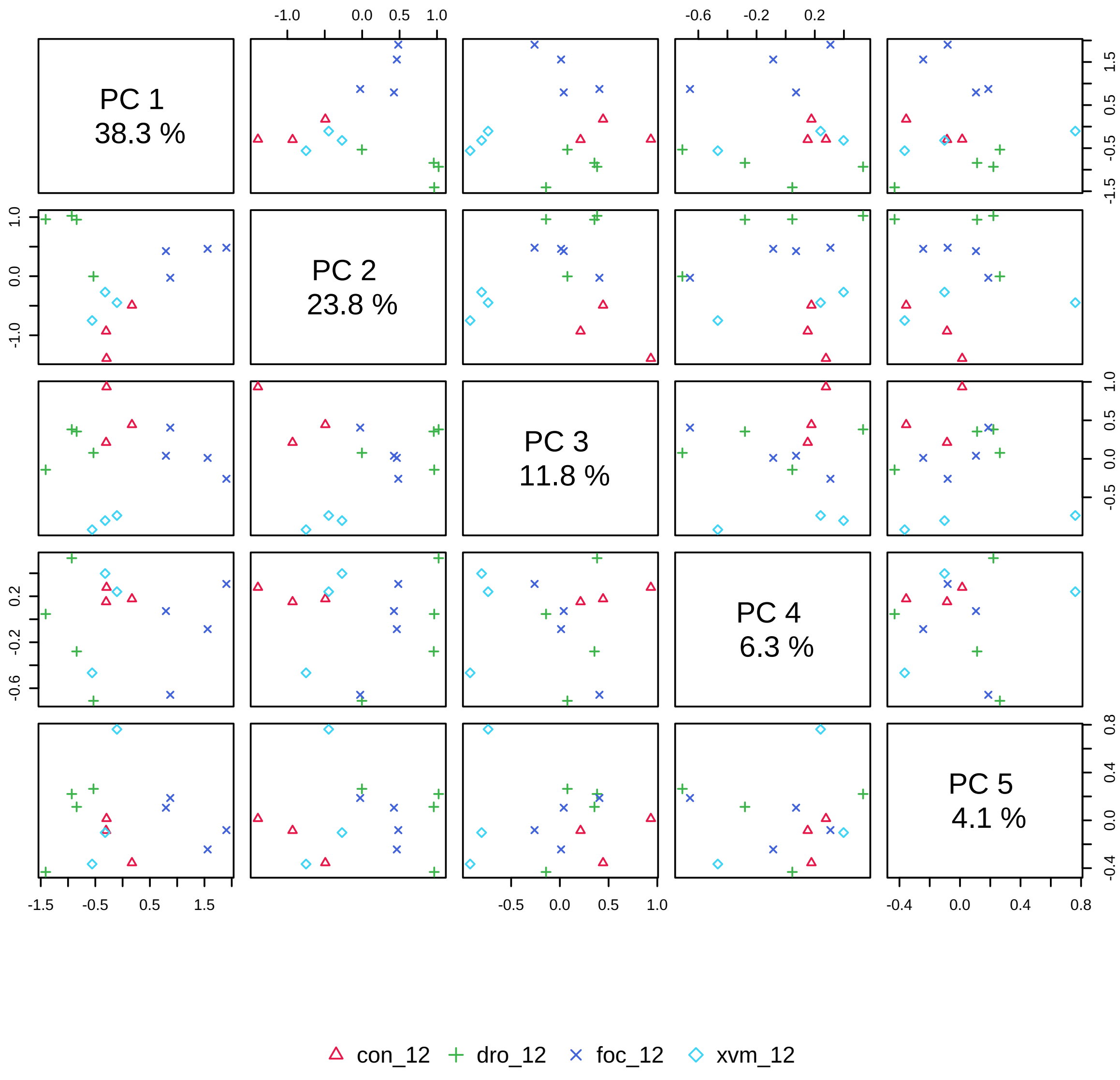

Figure 3: Pairwise score plots between the selected PCs. The explained variance of each PC is shown in the corresponding diagonal cell.

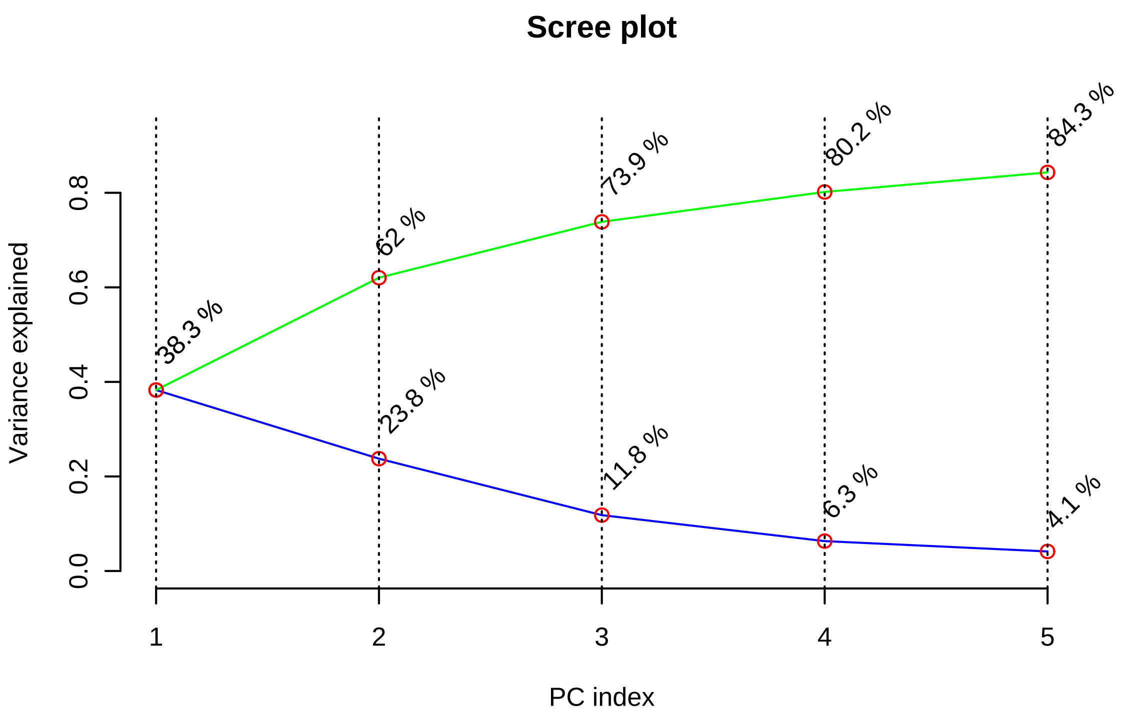

Figure 4: Scree plot shows the variance explained by PCs. The green line on top shows the accumulated variance explained; the blue line underneath shows the variance explained by individual PC.

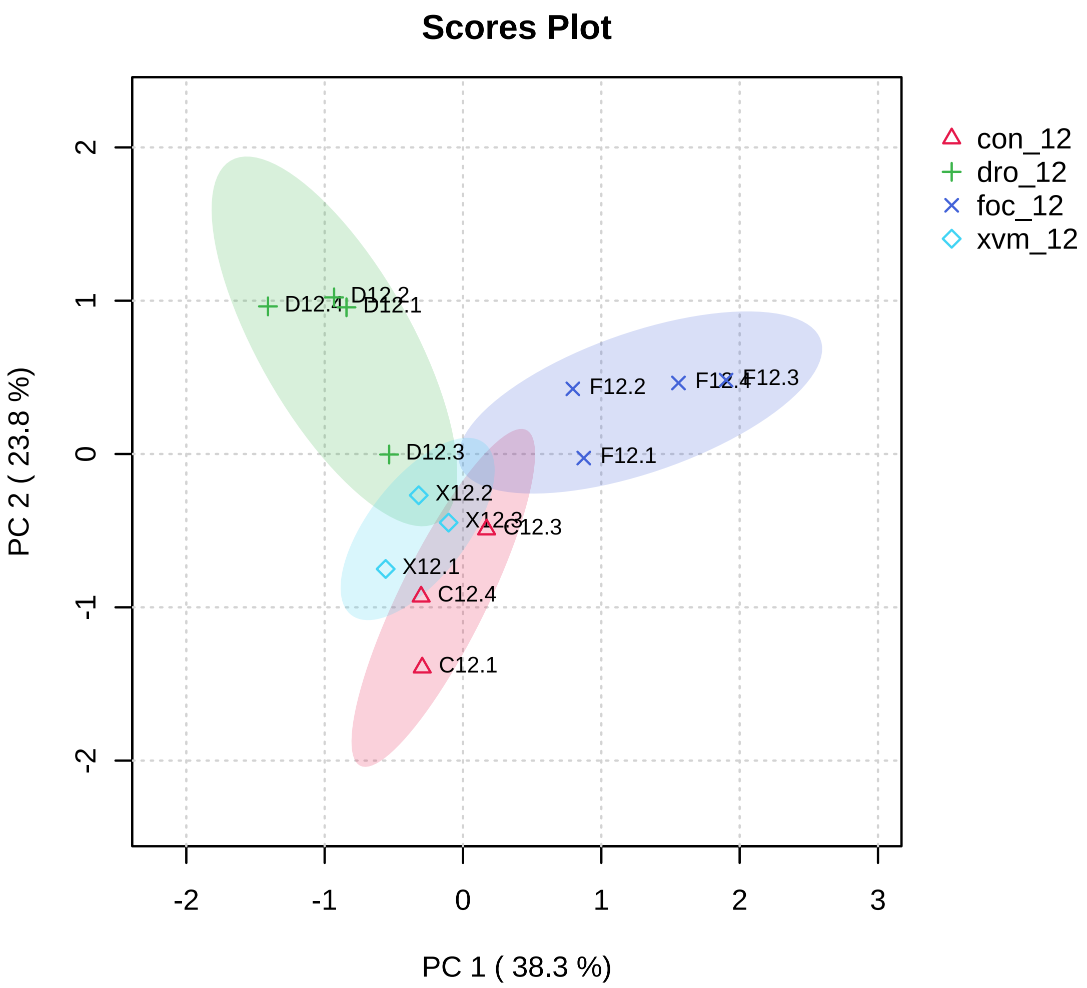

Figure 5: Scores plot between the selected PCs. The explained variances are shown in brackets.

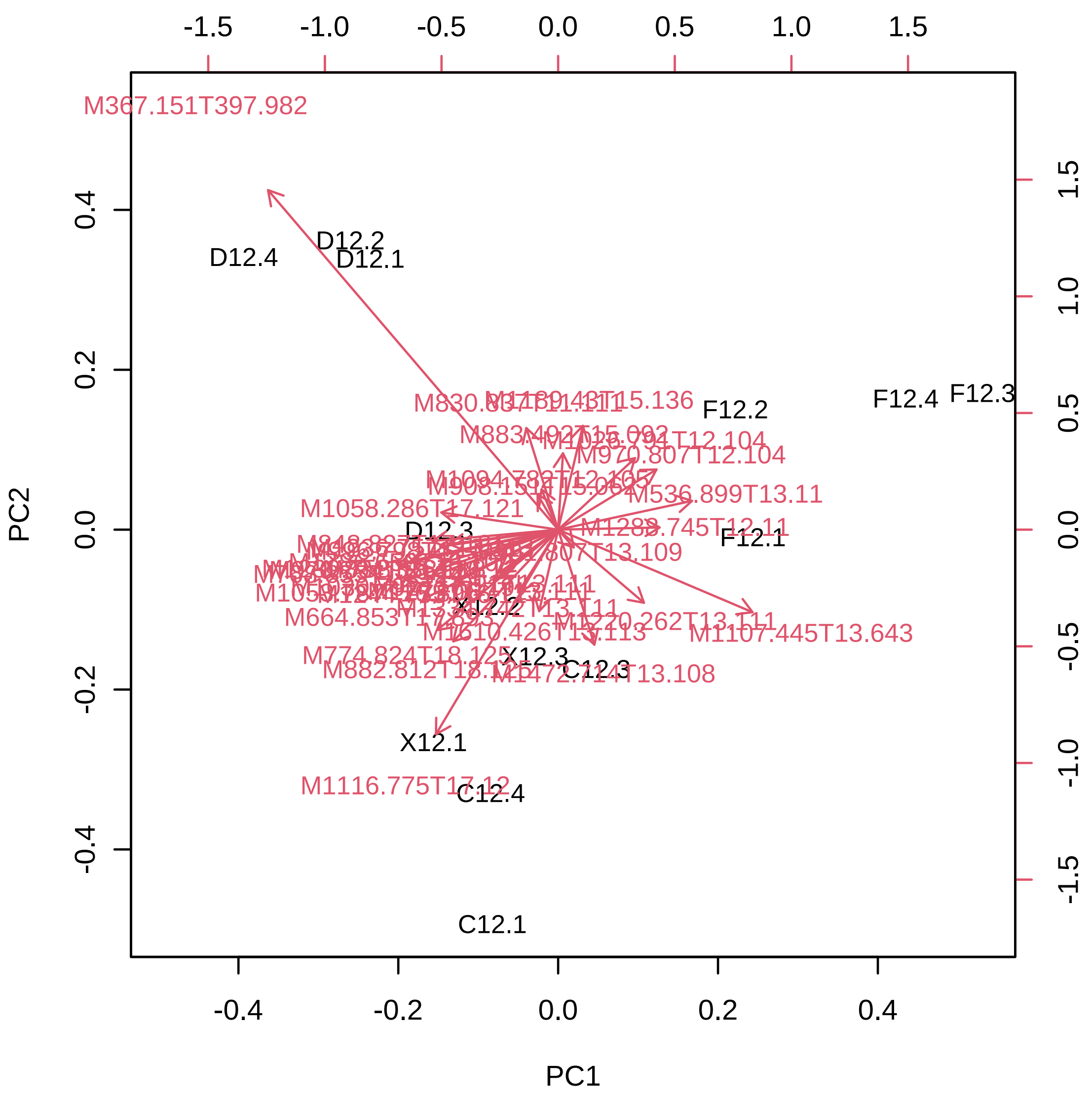

Figure 6: PCA biplot between the selected PCs. Note, you may want to test different centering and scaling normalization methods for the biplot to be displayed properly.

2.3 Hierarchical Clustering

In (agglomerative) hierarchical cluster analysis, each sample begins as a separate cluster and the algorithm proceeds to combine them until all samples belong to one cluster. Two parameters need to be considered when performing hierarchical clustering. The first one is similarity measure - Euclidean distance, Pearson's correlation, Spearman's rank correlation. The other parameter is clustering algorithms, including average linkage (clustering uses the centroids of the observations), complete linkage (clustering uses the farthest pair of observations between the two groups), single linkage (clustering uses the closest pair of observations) and Ward's linkage (clustering to minimize the sum of squares of any two clusters). Heatmap is often presented as a visual aid in addition to the dendrogram.

Hierachical clustering is performed with the `hclust` function in package `stat`. Figure 7 shows the clustering result in the form of a dendrogram. Figure 8 shows the clustering result in the form of a heatmap.

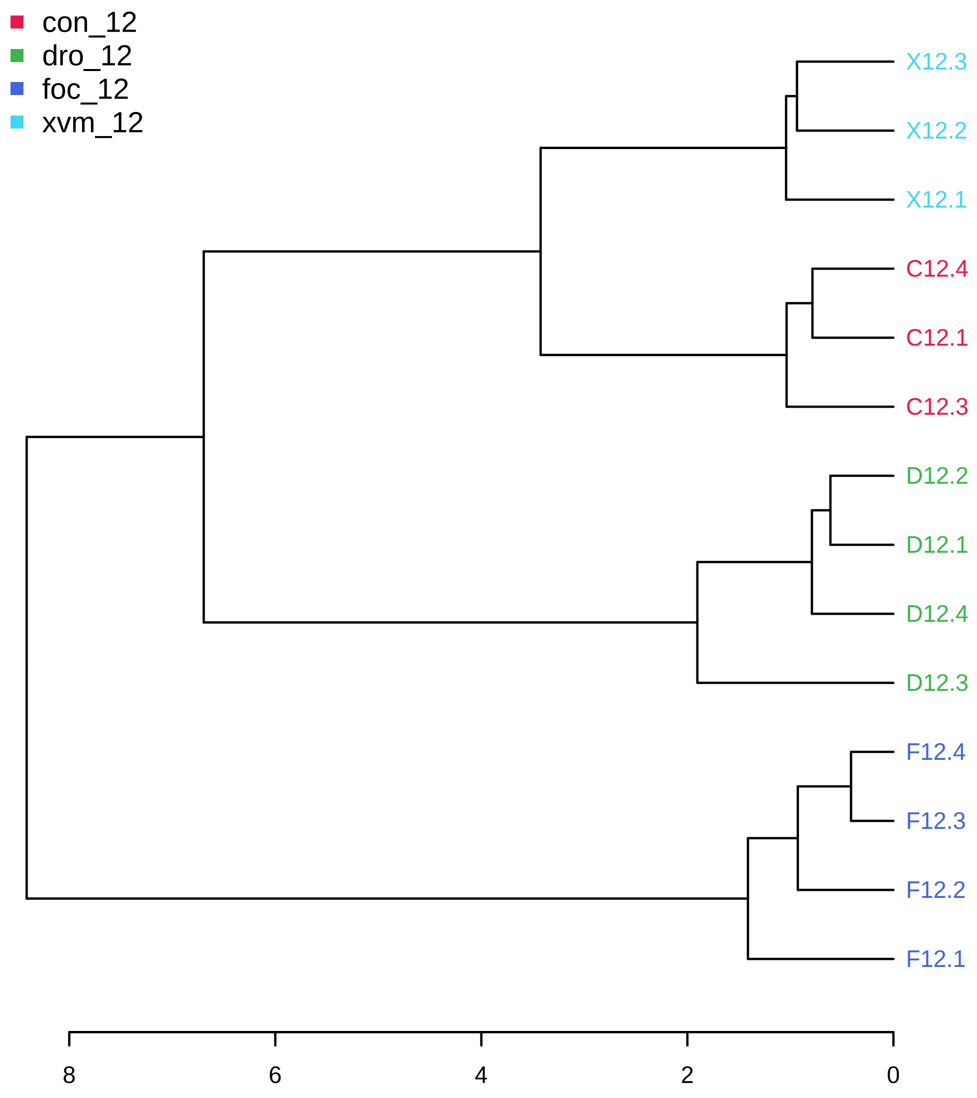

Figure 7: Clustering result shown as dendrogram (distance measure using `pearson`, and clustering algorithm using `ward.D`).

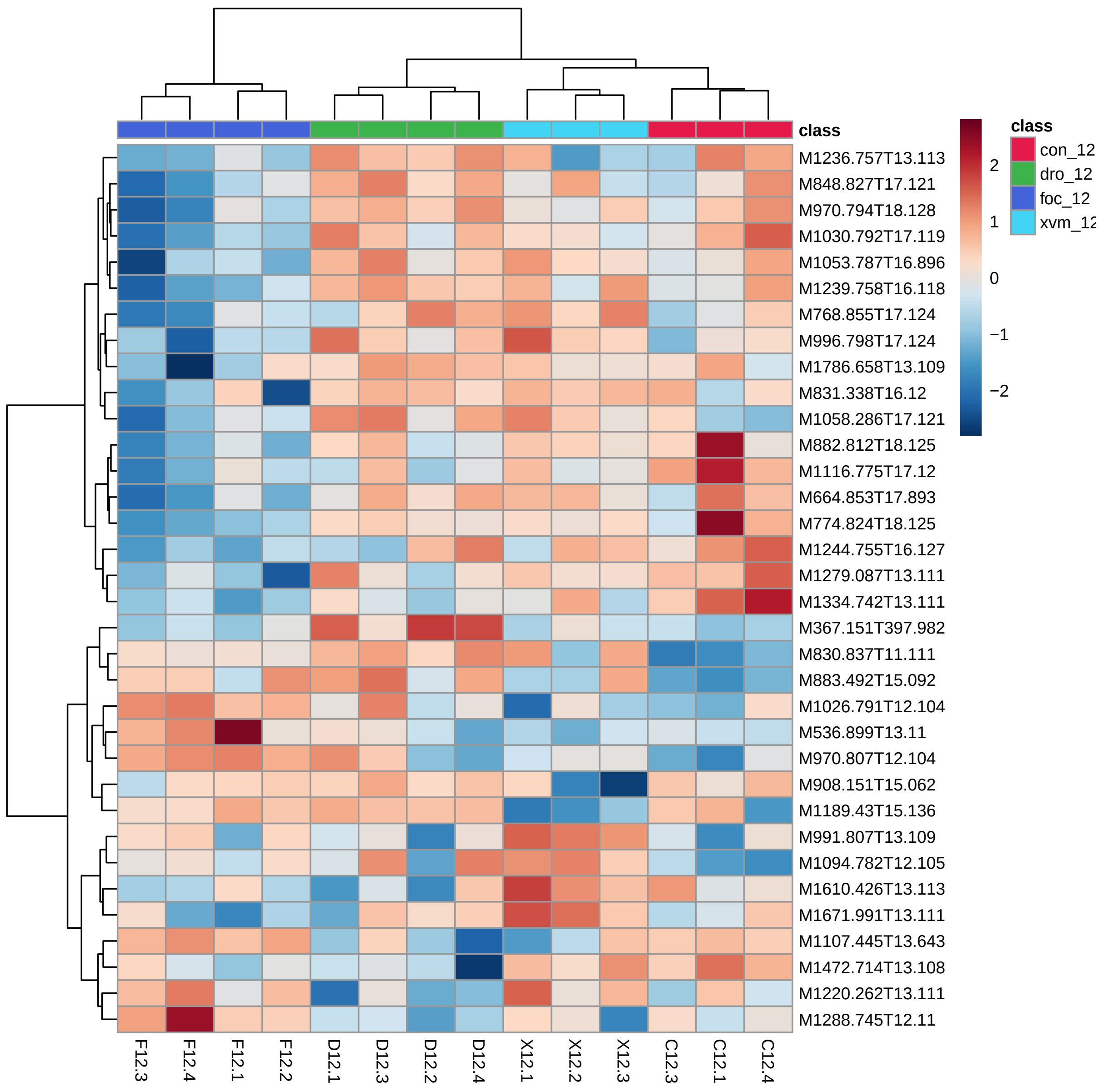

Figure 8: Clustering result shown as heatmap (distance measure using `euclidean`, and clustering algorithm using `ward.D`).

3 Appendix: R Command History

```
 [1] "mSet<-InitDataObjects(\"pktable\", \"stat\", FALSE)"
 [2] "mSet<-Read.TextData(mSet, \"Replacing_with_your_file_path\", \"rowu\", \"disc\");"
 [3] "mSet<-SanityCheckData(mSet)"
 [4] "mSet<-ReplaceMin(mSet);"
 [5] "mSet<-SanityCheckData(mSet)"
 [6] "mSet<-FilterVariable(mSet, \"F\", 25, \"iqr\", 0)"
 [7] "mSet<-PreparePrenormData(mSet)"
 [8] "mSet<-GetGroupNames(mSet, \"\")"
 [9] "feature.nm.vec <- c(\"\")"
[10] "smpl.nm.vec <- c(\"\")"
[11] "grp.nm.vec <- c(\"con_12\",\"dro_12\",\"foc_12\",\"xvm_12\")"
[12] "mSet<-UpdateData(mSet, T)"
[13] "mSet<-PreparePrenormData(mSet)"
[14] "mSet<-Normalization(mSet, \"CompNorm\", \"LogNorm\", \"ParetoNorm\", \"Sodium_Formate\", ratio
[15] "mSet<-PlotNormSummary(mSet, \"norm_0_\", \"png\", 72, width=NA)"
[16] "mSet<-PlotSampleNormSummary(mSet, \"snorm_0_\", \"png\", 72, width=NA)"
[17] "mSet<-ANOVA.Anal(mSet, F, 0.05, FALSE)"
[18] "mSet<-PlotANOVA(mSet, \"aov_0_\", \"png\", 72, width=NA)"
[19] "mSet<-Calculate.ANOVA.posthoc(mSet, \"tukey\", 0.05)"
[20] "mSet<-ANOVA.Anal(mSet, F, 0.05, FALSE)"
[21] "mSet<-PlotANOVA(mSet, \"aov_1_\", \"png\", 72, width=NA)"
[22] "mSet<-UpdateLoadingCmpd(mSet, \"M1094.782T12.105\")"
[23] "mSet<-PlotCmpdSummary(mSet, \"M1094.782T12.105\",\"NA\", 0, \"png\", 72, width=NA)"
[24] "mSet<-UpdateLoadingCmpd(mSet, \"M1334.742T13.111\")"
[25] "mSet<-PlotCmpdSummary(mSet, \"M1334.742T13.111\",\"NA\", 1, \"png\", 72, width=NA)"
[26] "mSet<-UpdateLoadingCmpd(mSet, \"M367.151T397.982\")"
[27] "mSet<-PlotCmpdSummary(mSet, \"M367.151T397.982\",\"NA\", 2, \"png\", 72, width=NA)"
[28] "mSet<-UpdateLoadingCmpd(mSet, \"M830.837T11.111\")"
[29] "mSet<-PlotCmpdSummary(mSet, \"M830.837T11.111\",\"NA\", 3, \"png\", 72, width=NA)"
[30] "mSet<-UpdateLoadingCmpd(mSet, \"M774.824T18.125\")"
[31] "mSet<-PlotCmpdSummary(mSet, \"M774.824T18.125\",\"NA\", 4, \"png\", 72, width=NA)"
[32] "mSet<-GetGroupNames(mSet, \"null\")"
[33] "mSet<-PlotCmpdSummary(mSet, \"M774.824T18.125\",\"NA\", 100, \"png\", 300, width=NA)"
[34] "mSet<-UpdateLoadingCmpd(mSet, \"M882.812T18.125\")"
[35] "mSet<-PlotCmpdSummary(mSet, \"M882.812T18.125\",\"NA\", 5, \"png\", 72, width=NA)"
[36] "mSet<-PlotCmpdSummary(mSet, \"M882.812T18.125\",\"NA\", 100, \"png\", 300, width=NA)"
[37] "mSet<-UpdateLoadingCmpd(mSet, \"M1786.658T13.109\")"
[38] "mSet<-PlotCmpdSummary(mSet, \"M1786.658T13.109\",\"NA\", 6, \"png\", 72, width=NA)"
[39] "mSet<-UpdateLoadingCmpd(mSet, \"M991.807T13.109\")"
[40] "mSet<-PlotCmpdSummary(mSet, \"M991.807T13.109\",\"NA\", 7, \"png\", 72, width=NA)"
[41] "mSet<-PlotCmpdSummary(mSet, \"M991.807T13.109\",\"NA\", 100, \"png\", 300, width=NA)"
[42] "mSet<-UpdateLoadingCmpd(mSet, \"M768.855T17.124\")"
[43] "mSet<-PlotCmpdSummary(mSet, \"M768.855T17.124\",\"NA\", 8, \"png\", 72, width=NA)"
[44] "mSet<-UpdateLoadingCmpd(mSet, \"M1189.43T15.136\")"
[45] "mSet<-PlotCmpdSummary(mSet, \"M1189.43T15.136\",\"NA\", 9, \"png\", 72, width=NA)"
[46] "mSet<-UpdateLoadingCmpd(mSet, \"M996.798T17.124\")"
[47] "mSet<-PlotCmpdSummary(mSet, \"M996.798T17.124\",\"NA\", 10, \"png\", 72, width=NA)"
[48] "mSet<-UpdateLoadingCmpd(mSet, \"M848.827T17.121\")"
[49] "mSet<-PlotCmpdSummary(mSet, \"M848.827T17.121\",\"NA\", 11, \"png\", 72, width=NA)"
[50] "mSet<-UpdateLoadingCmpd(mSet, \"M1786.658T13.109\")"
[51] "mSet<-PlotCmpdSummary(mSet, \"M1786.658T13.109\",\"NA\", 12, \"png\", 72, width=NA)"
[52] "mSet<-UpdateLoadingCmpd(mSet, \"M536.899T13.11\")"
[53] "mSet<-PlotCmpdSummary(mSet, \"M536.899T13.11\",\"NA\", 13, \"png\", 72, width=NA)"
[54] "mSet<-PlotCmpdSummary(mSet, \"M536.899T13.11\",\"NA\", 100, \"png\", 300, width=NA)"
[55] "mSet<-UpdateLoadingCmpd(mSet, \"M1030.792T17.119\")"
[56] "mSet<-PlotCmpdSummary(mSet, \"M1030.792T17.119\",\"NA\", 14, \"png\", 72, width=NA)"
```

```
[57] "mSet<-PlotCmpdSummary(mSet, \"M1030.792T17.119\",\"NA\", 100, \"png\", 300, width=NA)"
[58] "mSet<-PCA.Anal(mSet)"
[59] "mSet<-PlotPCAPairSummary(mSet, \"pca_pair_0_\", \"png\", 72, width=NA, 5)"
[60] "mSet<-PlotPCAScree(mSet, \"pca_scree_0_\", \"png\", 72, width=NA, 5)"
[61] "mSet<-PlotPCA2DScore(mSet, \"pca_score2d_0_\", \"png\", 72, width=NA, 1,2,0.95,0,0)"
[62] "mSet<-PlotPCALoading(mSet, \"pca_loading_0_\", \"png\", 72, width=NA, 1,2);"
[63] "mSet<-PlotPCABiplot(mSet, \"pca_biplot_0_\", \"png\", 72, width=NA, 1,2)"
[64] "mSet<-PlotPCA3DLoading(mSet, \"pca_loading3d_0_\", \"json\", 1,2,3)"
[65] "mSet<-PlotPCA2DScore(mSet, \"pca_score2d_1_\", \"png\", 72, width=NA, 1,2,0.95,1,0)"
[66] "mSet<-PlotPCA2DScore(mSet, \"pca_score2d_2_\", \"png\", 72, width=NA, 1,2,0.95,1,0)"
[67] "mSet<-PlotHCTree(mSet, \"tree_0_\", \"png\", 72, width=NA, \"euclidean\", \"ward.D\")"
[68] "mSet<-PlotHCTree(mSet, \"tree_1_\", \"png\", 72, width=NA, \"pearson\", \"ward.D\")"
[69] "mSet<-PlotHCTree(mSet, \"tree_1_\", \"png\", 300, width=NA, \"pearson\", \"ward.D\")"
[70] "mSet<-PlotHeatMap(mSet, \"heatmap_0_\", \"png\", 72, width=NA, \"norm\", \"row\", \"euclidean\
[71] "mSet<-SaveTransformedData(mSet)"
[72] "mSet<-PreparePDFReport(mSet, \"guest16468220247862756278\")\n"
```

The report was generated on Wed Dec 6 08:02:08 2023 with R version 4.2.2 (2022-10-31), OS system: Linux, version: -Ubuntu SMP Mon May 15 15:18:26 UTC 2023 .