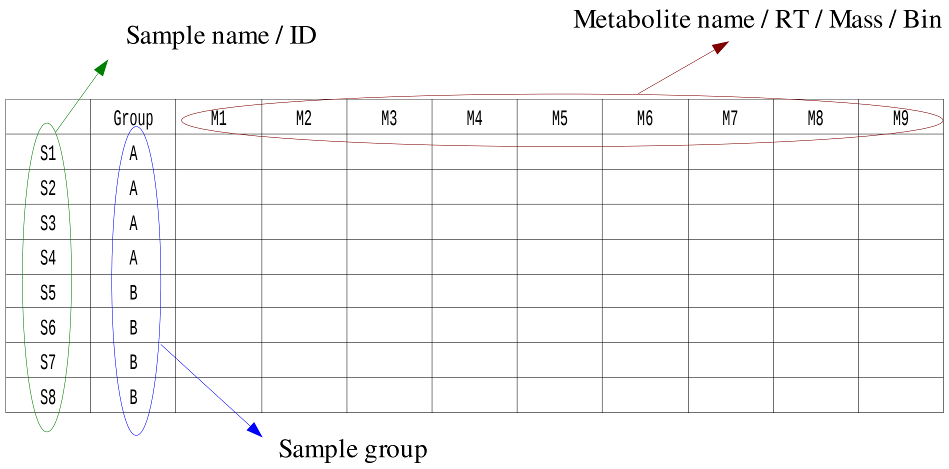
R Scripts

MA scripts

The bioinformatics group at Metabolomics Australia have created a number of scripts for use with R. These provide basic statistical functionality in a manner that allows a standard data input format to be transformed to a standard output format, allowing much more simple data analysis methods.

Input data format

If there is a consistent input format, scripts are much simpler to execute and troubleshoot if they do not work. In brief, the matrix should have the first column containing sample names, the second the groups and the remaining columns the variables to be processed by the script (Figure 1). These can be metabolites, retention times, masses, bins, or anything else that has been measured for the samples.



	Group	M1	M2	M3	M4	M5	M6	M7	M8	M9
S1	A									
S2	A									
S3	A									
S4	A									
S5	B									
S6	B									
S7	B									
S8	B									

Figure 1: The standard format of the data matrix for statistical analyses

Currently available scripts

All of the scripts that are in the ma-scripts repository (http://code.google.com/p/ma-bioinformatics/) use the input data format specified above, and will process that as necessary. These are still under development, but they will provide the necessary output as they currently stand. These scripts have comments in them to let you know what each step achieves.

R Scripts

Examples provided below are created using the file `input.csv`.

boxplot.r

This script will produce a boxplot for both samples (Figure 2) and variables (Figure 3).

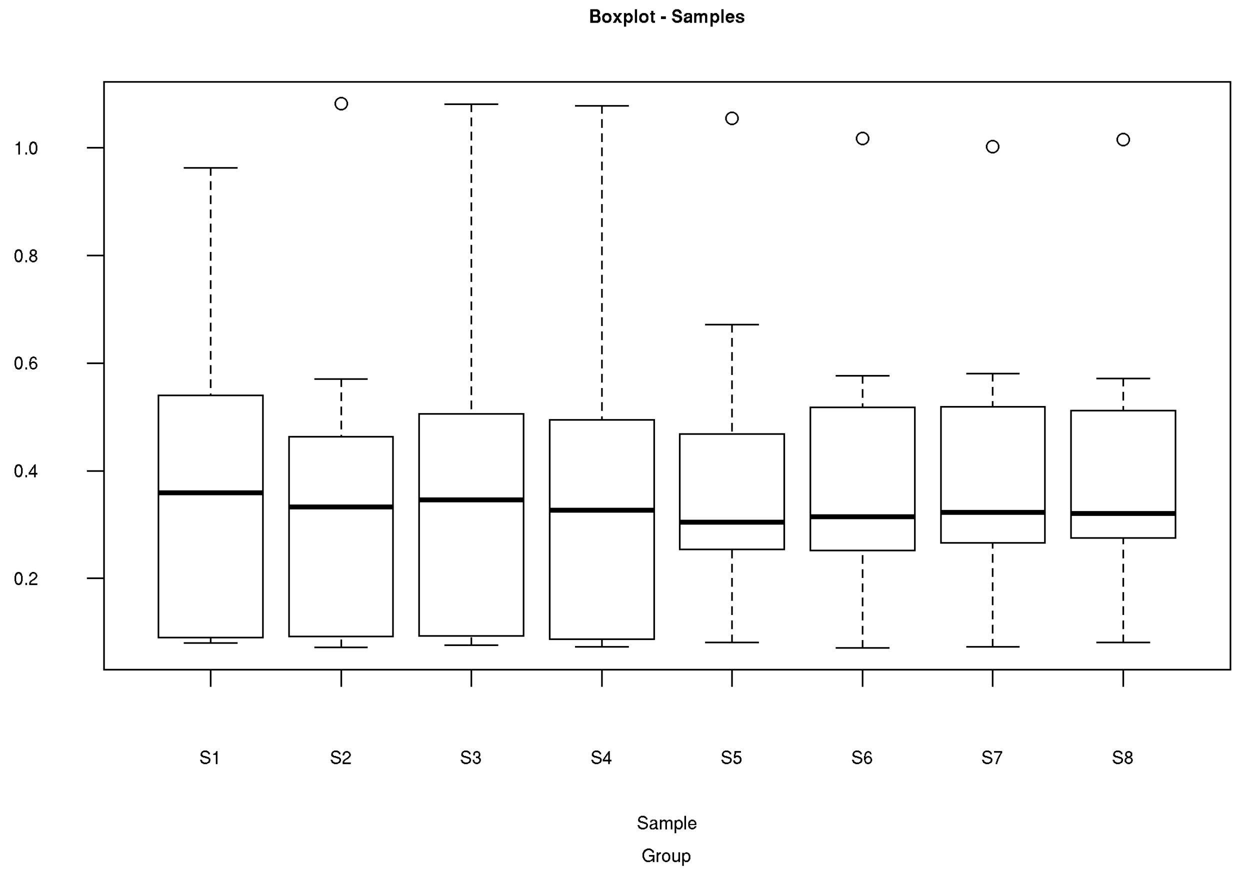


Figure 2: Boxplot for samples using data from input.csv

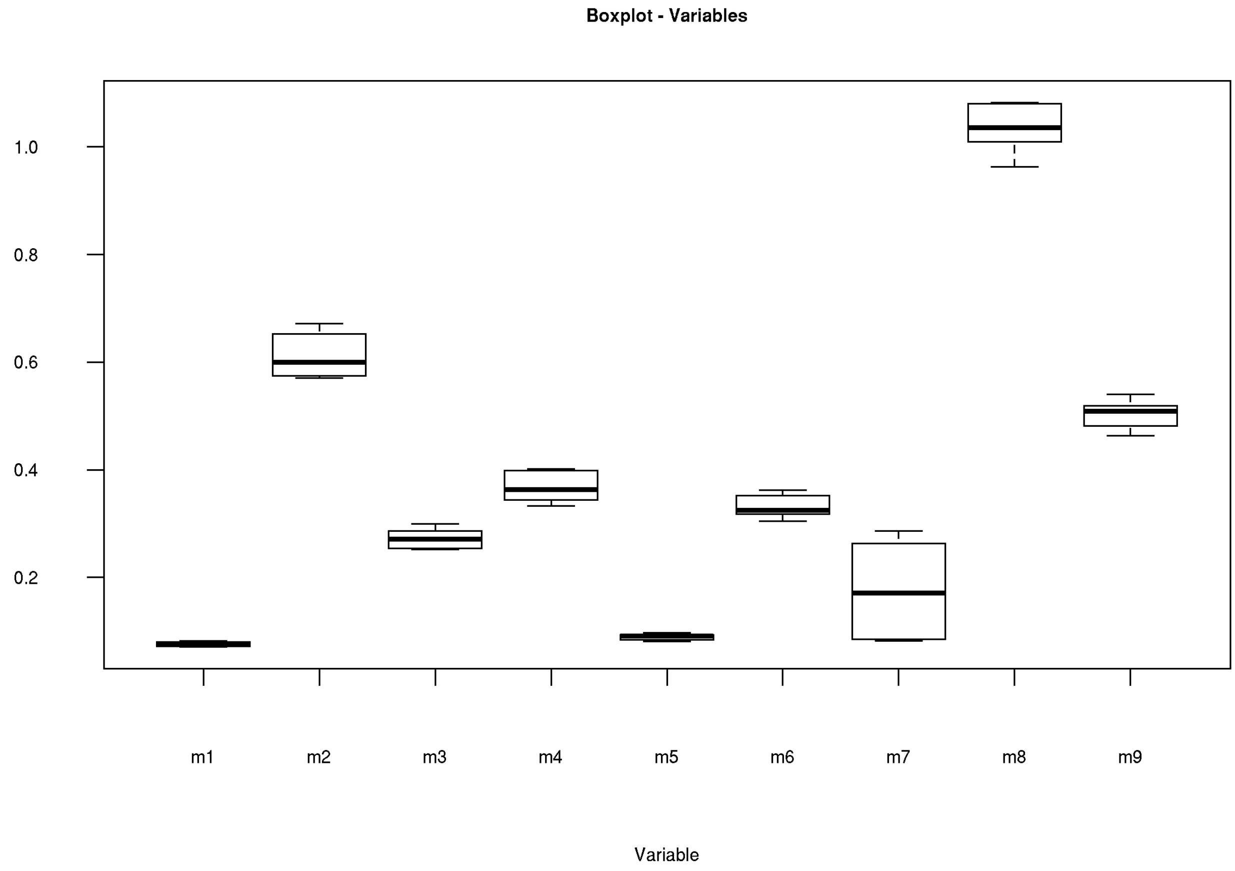


Figure 3: Boxplot of variables from input.csv

R Scripts

mms_plot.r

Produces a plot of the mean, median and standard deviation across all samples (Figure 4). Enables a graphical overview of similarity of samples, and allows for rapid identification of outliers.

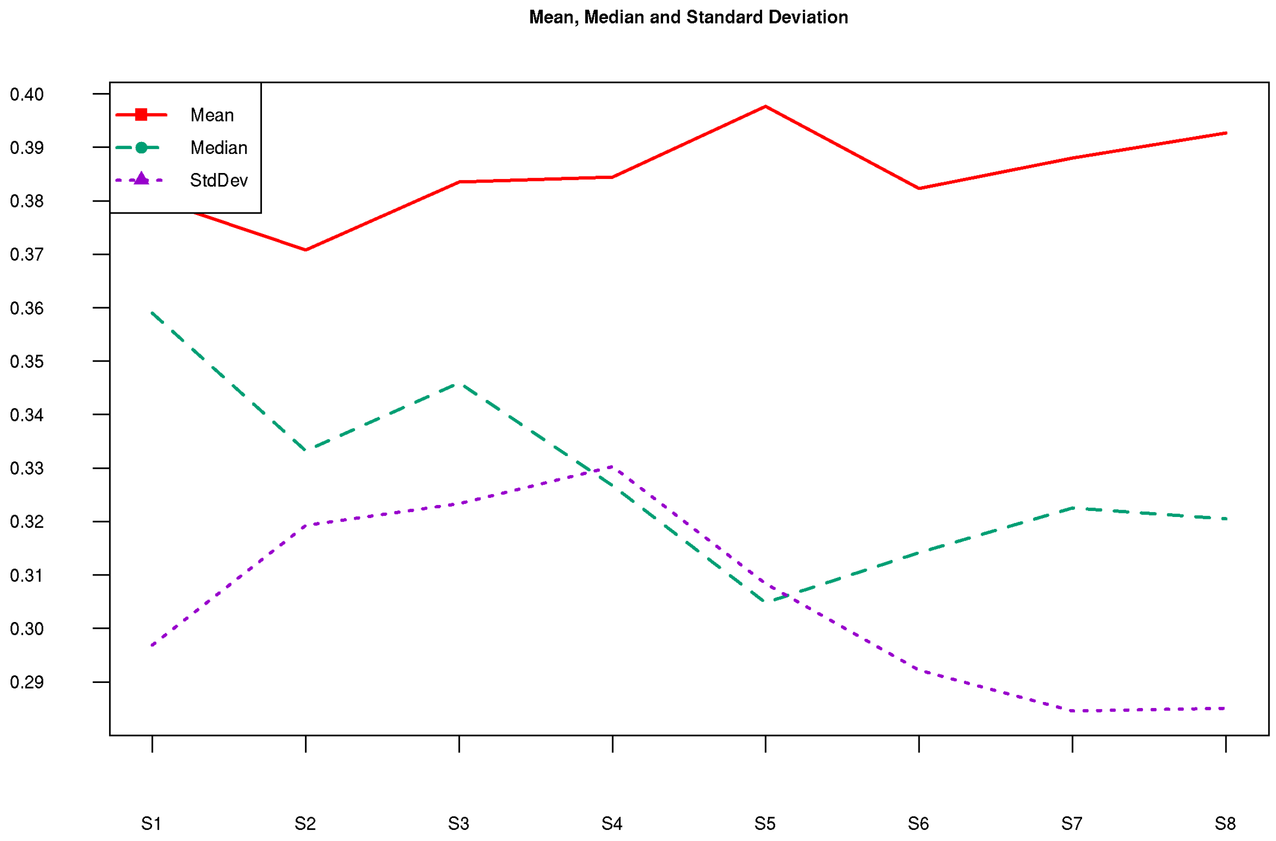


Figure 4: Line plot showing the mean, median and standard deviation .

mean_sd.r

It is important to explore a data set prior to choosing any normalisation or transformation method. Heteroscedastic noise (where the standard deviation of each metabolite in replicate samples changes with the mean of the metabolite) may change the results from the normalised data profoundly.

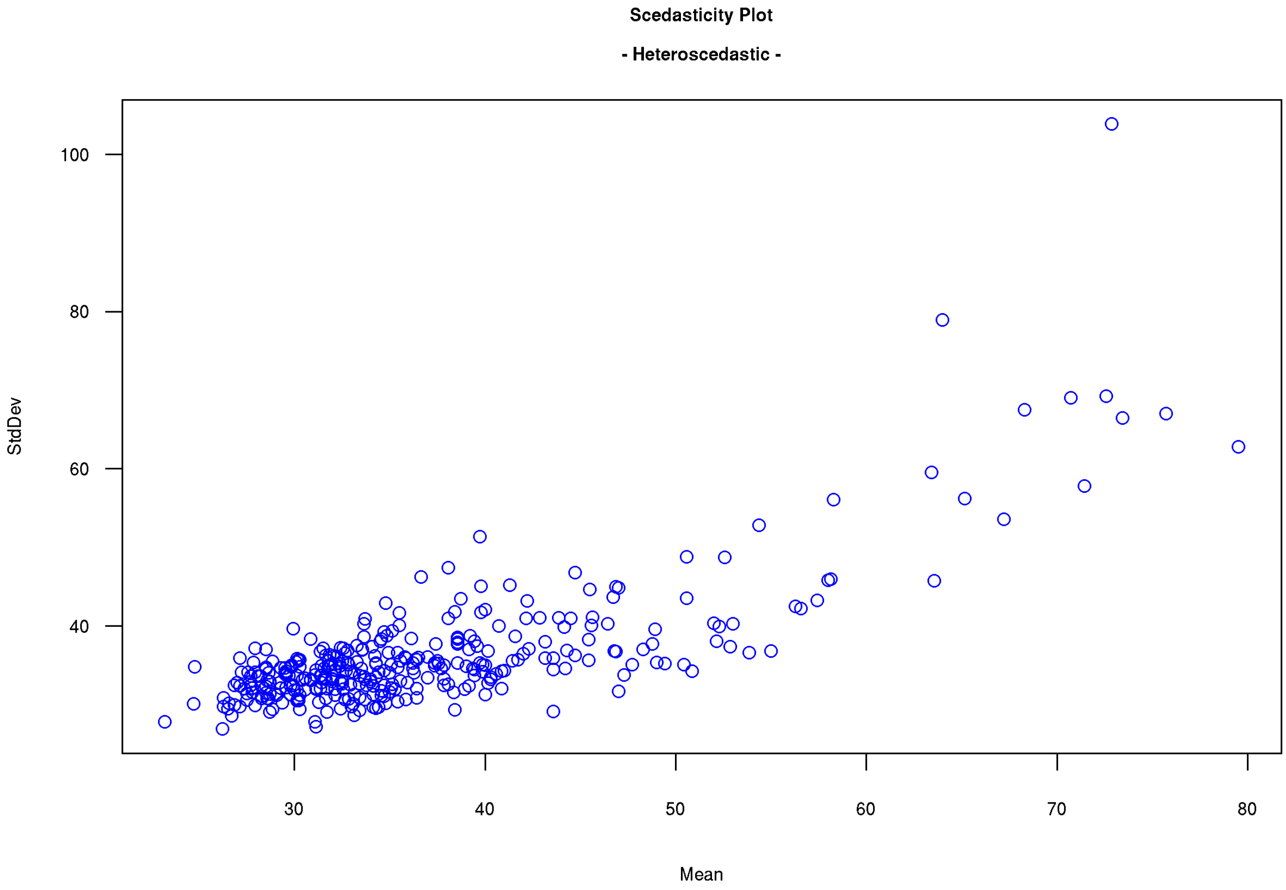


Figure 5: Heteroscedastic data. The overall shape of the data is a fanned line, as the standard deviation increases with the size of the mean.

The optimal transformation to change heteroscedastic noise into homoscedastic noise depends on the structure of the heteroscedasticity in the signals. If the standard deviation is proportional to the mean of the signal (as in Figure 5), then a log transform is optimal to treat the data. If the standard deviation is proportional to the root of the mean then square root transformation provides a homoscedastic noise pattern (as in Figure 6).

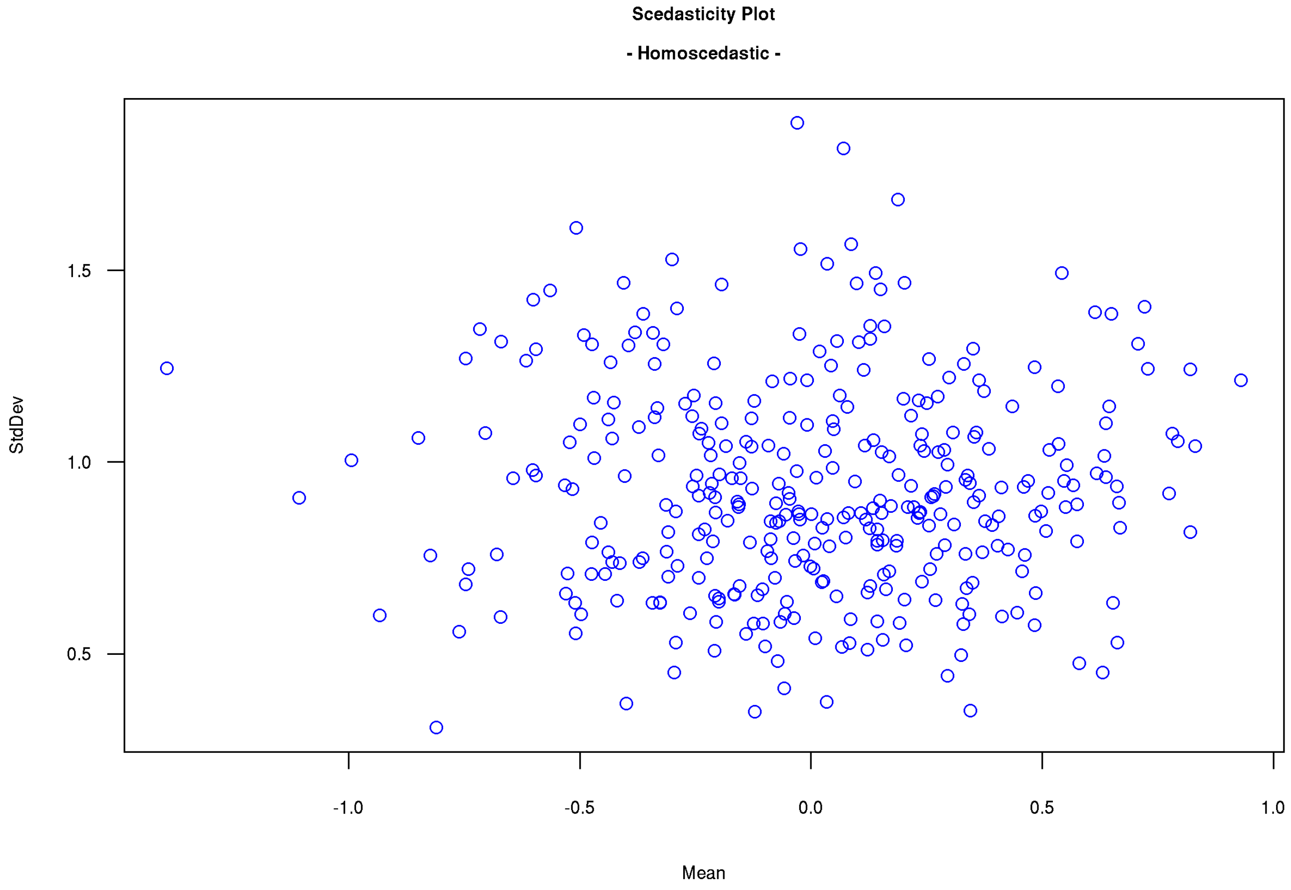


Figure 6: Homoscedastic data. Points are distributed randomly over the plot, and standard deviation does not vary with the size of the mean.

R Scripts

Transformation

This is an operation that is applied to each variable, and the type of transformation applied depends on the statistical test that is necessary to answer a biological question. For example, a *t*-test assumes that the data is normally distributed, hence it is important to transform a right- or left-skewed data set to a normal distribution.

log.r

Performs a $\log_{10}$ transform of the input data, and generates a new data matrix [`log_transform.csv`].

Normalisation

This process is applied to the entire data set, and enables samples to be directly compared to one another by reducing the influence that is non-biological in origin. As an example, running samples on different days can cause samples to *appear* different when in actual fact they are the same. Normalisation attempt to removes this influence so that samples from different groups can directly be compared. The output of these scripts are all .csv files.

norm_is.r

Uses an internal standard (in the column immediately following the "Group" column) to normalise the data set [`norm_data_IS.csv`].

norm_ext_vec.r

Uses an external vector (e.g. creatinine for urine samples) to normalise the data set [`norm_data_ext_vec.csv`].

norm_median.r

Takes the median value for each sample (i.e. row) and uses this value to normalise the data set [`norm_data_median.csv`].

norm_Zscore.r

Z score is a measure of how many standard deviations a sample is from the mean. This script calculates the Z score for each sample and uses that value to normalise the data set [`norm_data_zscore.csv`].

norm_sum.r

This script normalises the values on a per-sample basis by taking the sum of all values for each sample and using that value as the normalisation factor [`norm_data_sum.csv`].

R Scripts

Cluster analysis

hca.r

Hierarchical cluster analysis (HCA) is a technique that identifies groups of samples that show similar characteristics, and then quantifies the structural characteristics of the sample (or variable) data by constructing a hierarchy in a tree-like structure. There are two kinds of procedures that are commonly used to construct the tree, namely agglomerative and divisive. In an agglomerative procedure, each sample (or variable) starts in a cluster of its won, and then continually joins clusters until there is only one cluster containing all samples. The divisive method operates in the exact reverse.

This script uses complete linkage method but other different linkage measures can be used in HCA, including average, single and Ward's, which may result in different clusters. The main objective of HCA is to classify the data into groups by structuring it, which helps to identify relationships among observations (samples/variables).

This script uses Manhattan distance to calculate the similarity, but other methods can be used by changing `“manhattan”` to one of `"euclidean"`, `"maximum"`, `"canberra"`, `"binary"` or `"minkowski”.`

It will generate a plot of both the sample distances (see Figure 7) as well as the distance for the variables (Figure 8).

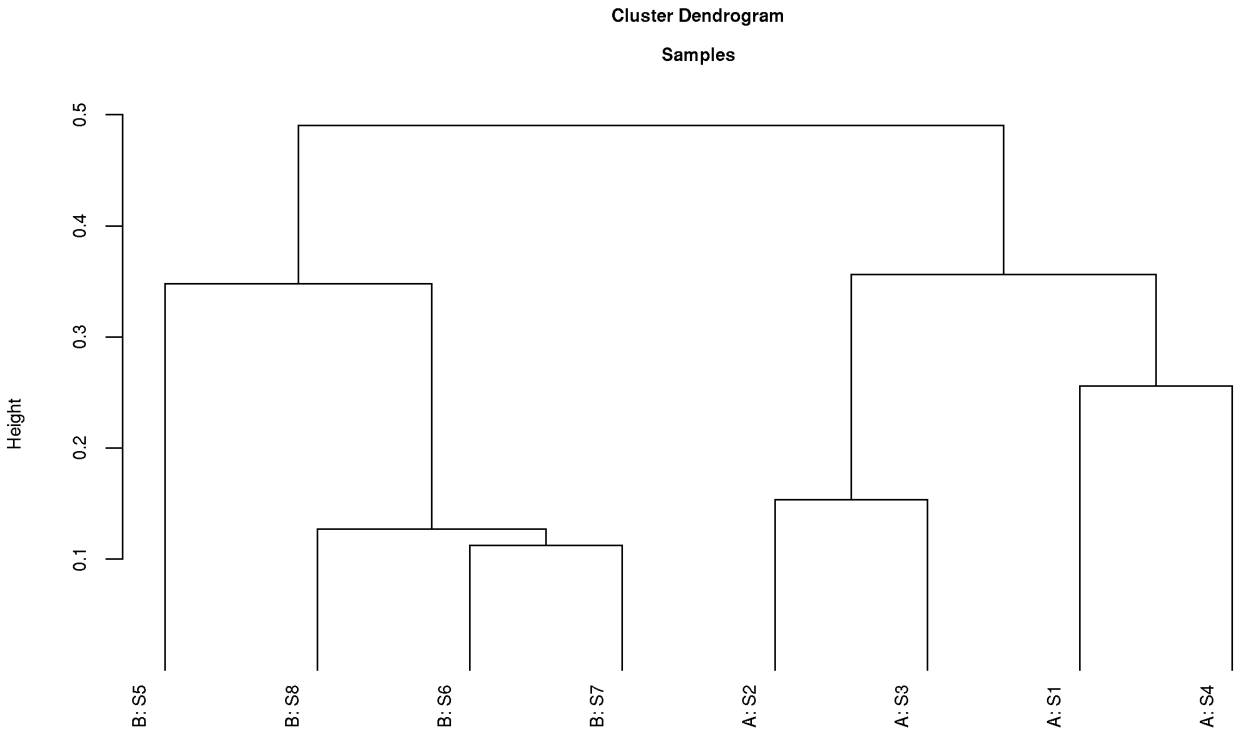


Figure 7: Hierarchical Cluster Analysis (HCA) for samples. Group labels are incorporated in the names on the leaves of the dendogram.

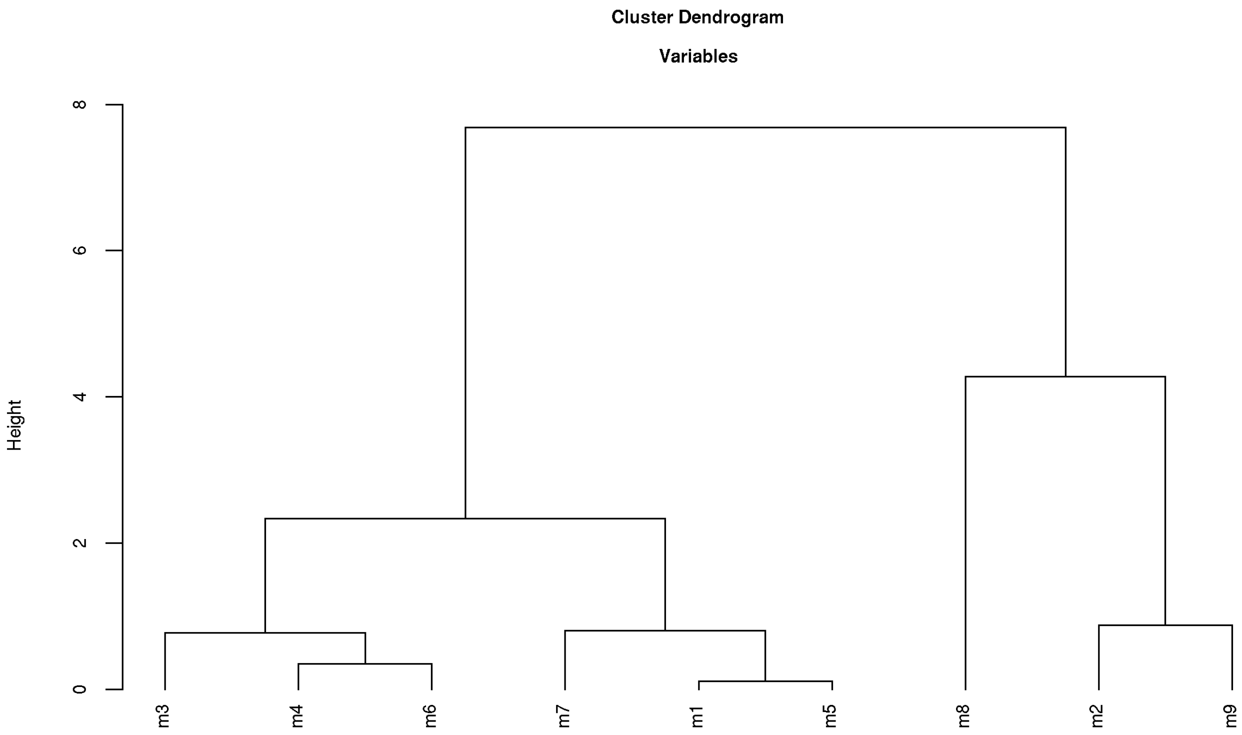


Figure 8: HCA for variables.

pca.r

Principal Component Analysis (PCA) is a data transformation technique which is used to reduce a multidimensional data set to a lower number of dimensions for further analysis. In PCA, a data set of interrelated variables is transformed to a new set of variables called principal components (PCs) in such a way that they are uncorrelated, and the first few of these PCs retain most of the variation present in the entire data set. The first PC is a linear combination of all the actual variables in such a way that it has the greatest amount of variation, and the second PC is a combination of the variables that have the next greatest variation in the remaining PCs.

This script produces multiple plots – residual variance (Figure 9), scores plot labelled with sample names (Figure 10), scores plot labelled with group names (Figure 11) and a loading plot (Figure 12).

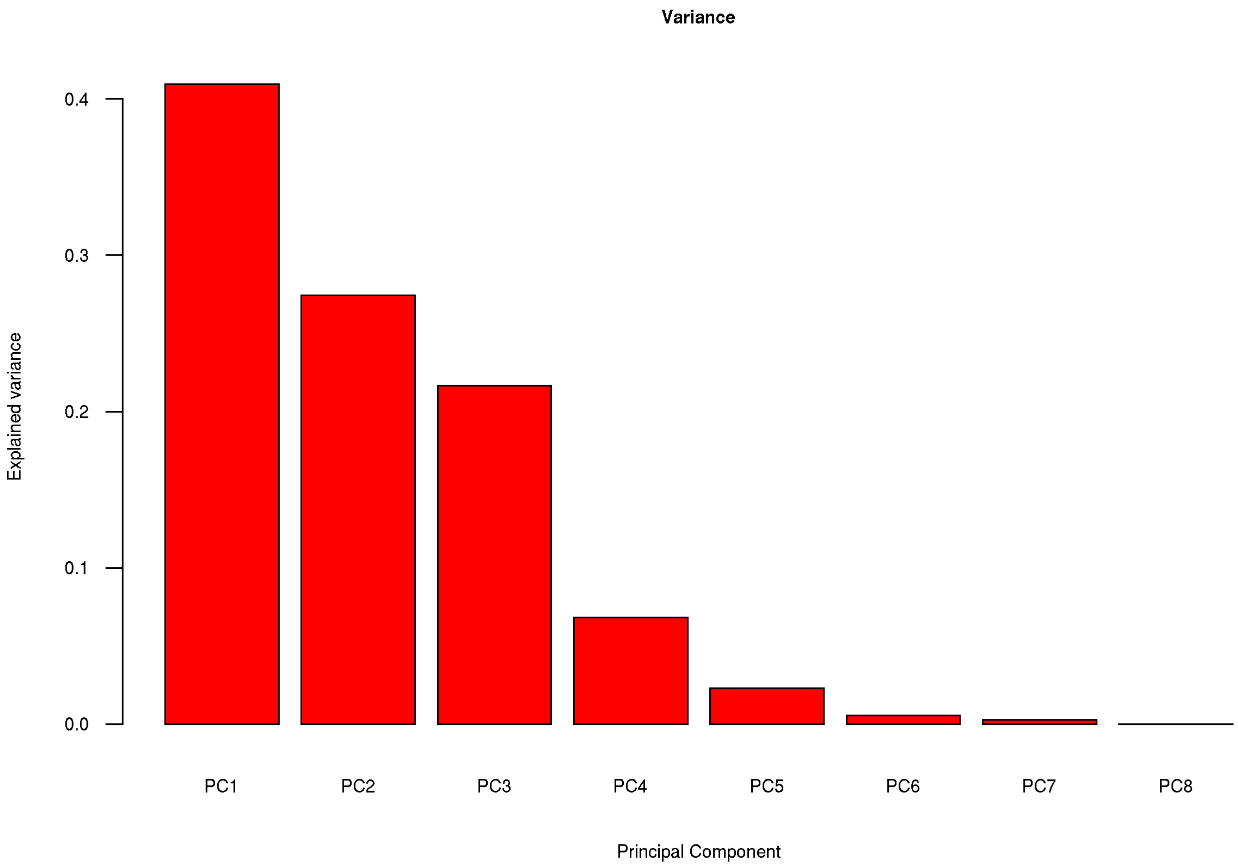


Figure 9: Residual variance plot.

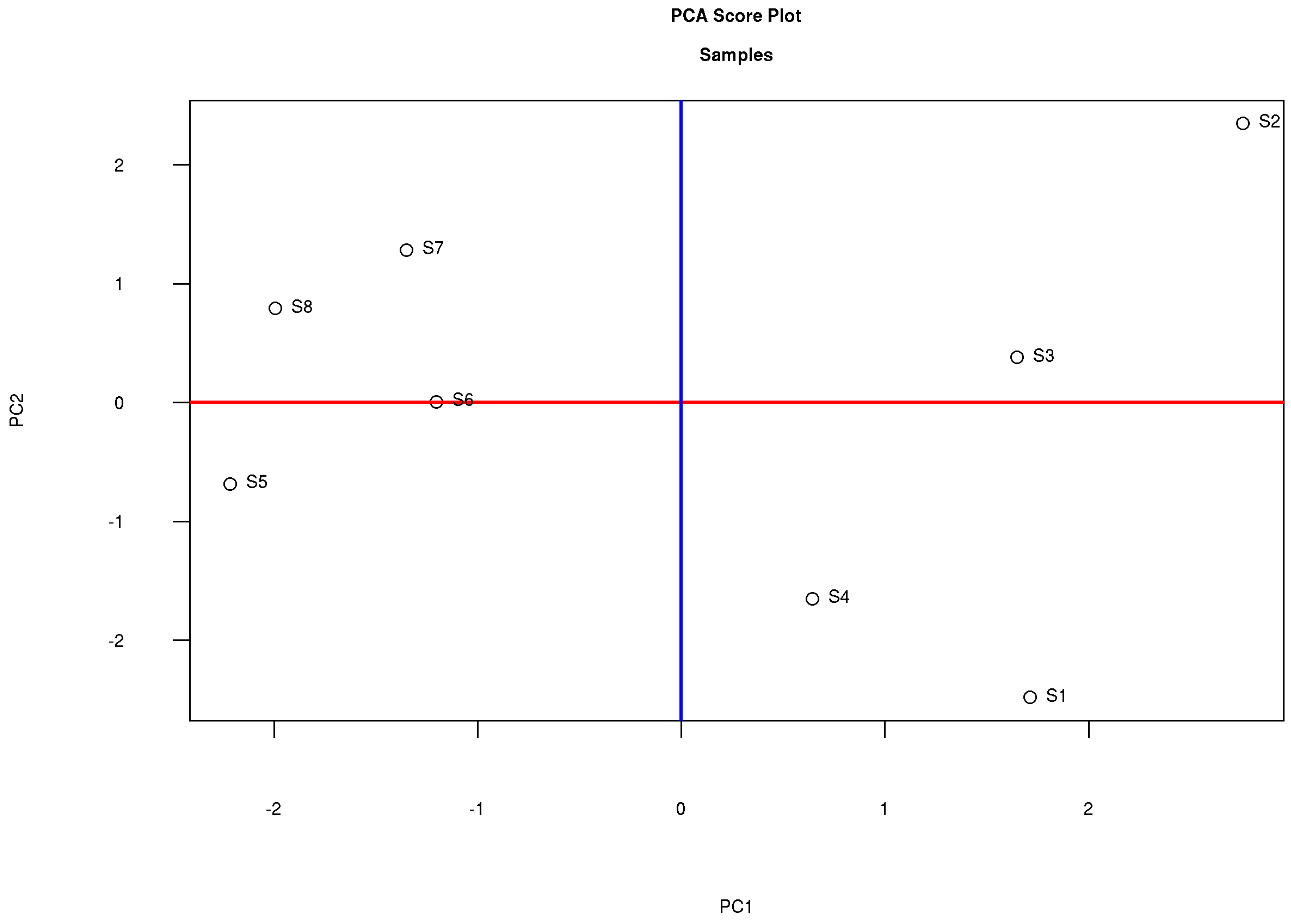


Figure 10: Scores plot with sample names as labels

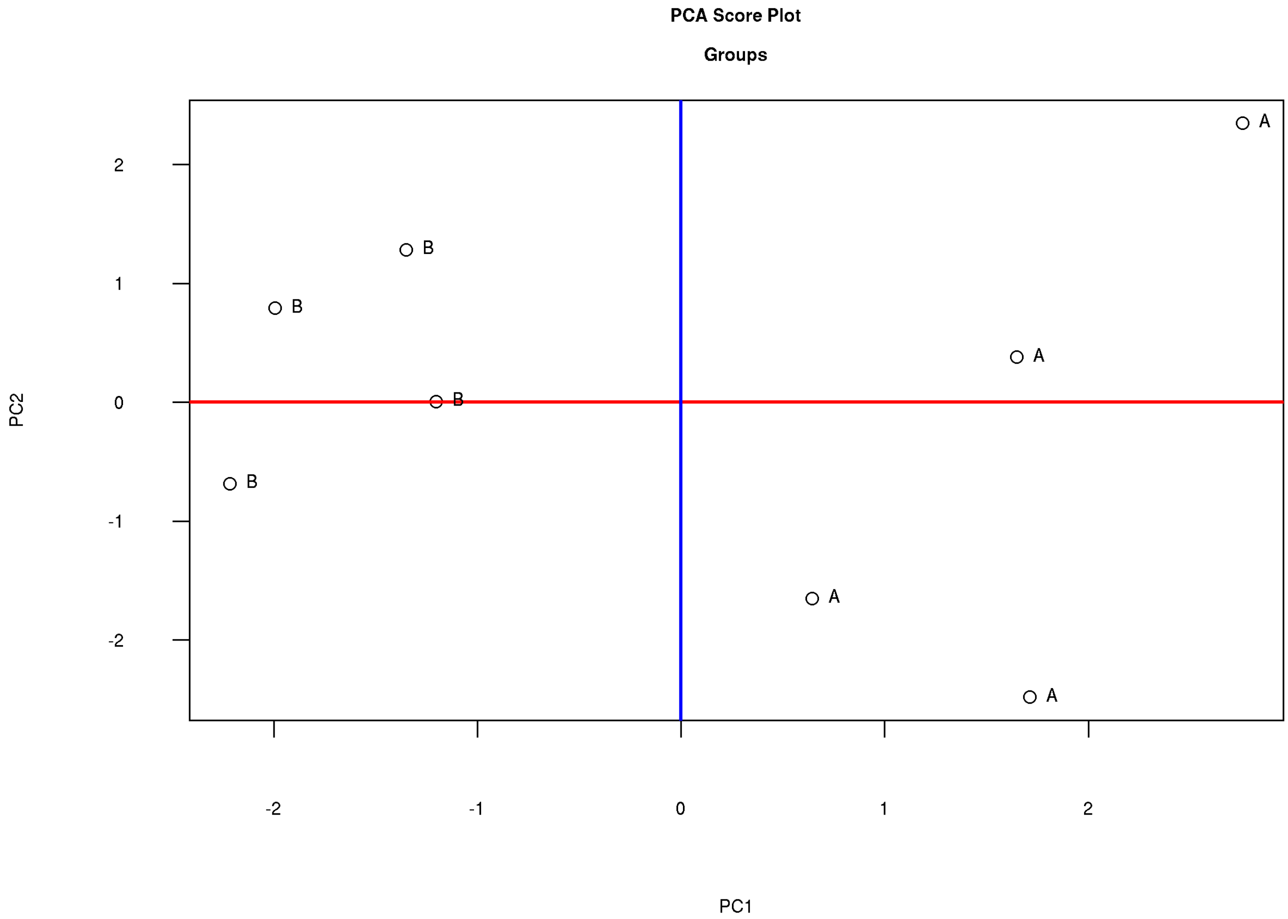


Figure 11: Scores plot with group names as labels

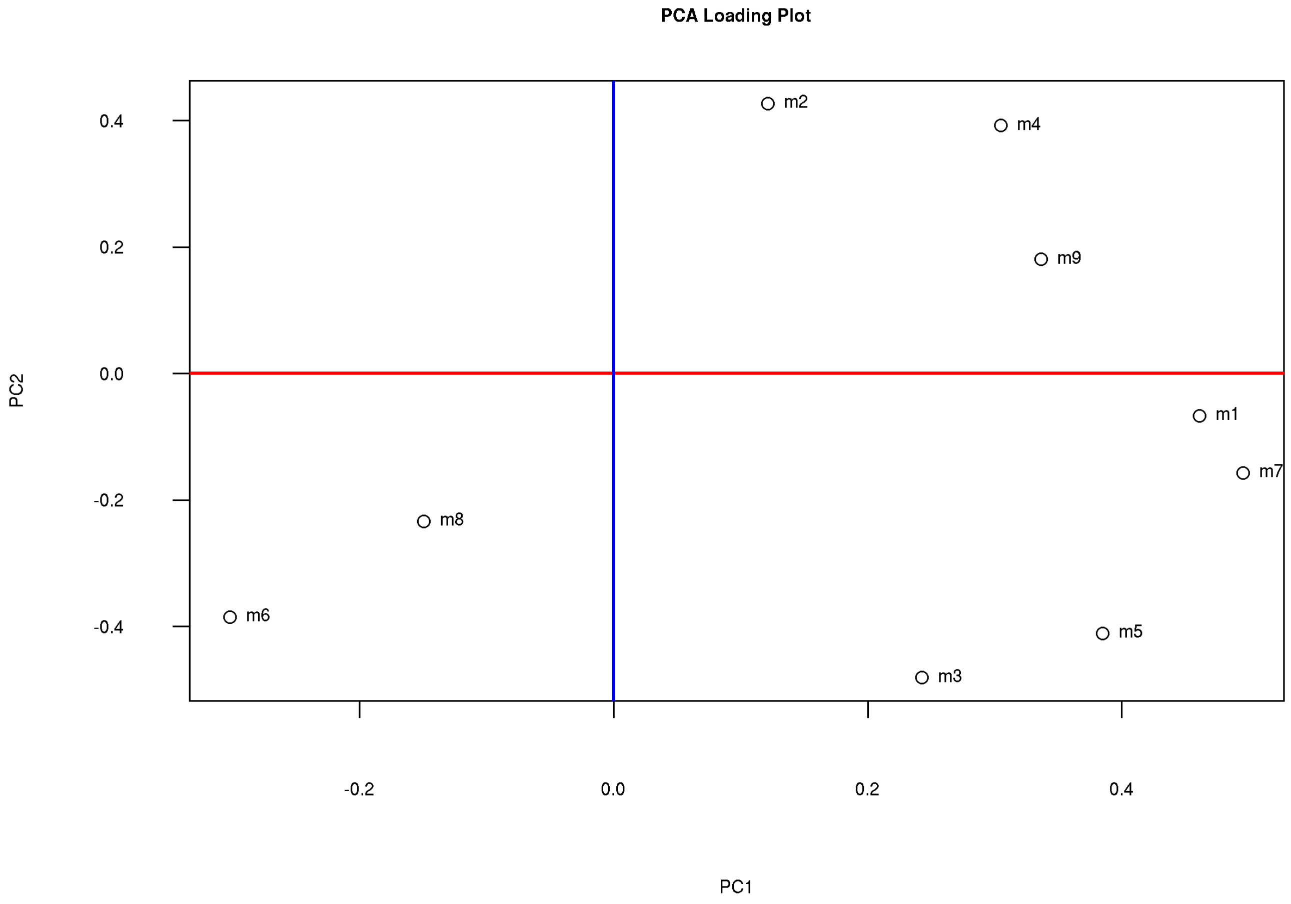


Figure 12: Loadings plot. Indicates which variables contributed to the separation seen in the scores plot.

lda.r

Linear Discriminant Analysis (LDA) is a classical technique to predict groups of samples. This is a supervised method that requires prior knowledge of the groups. LDA is therefore well suited for non-targeted metabolic profiling data which is almost always “grouped”. LDA is very similar to PCA, except that the technique maximises the ratio of between-class variance to the within-class variance in a set of data, and thus gives maximal separation between the classes, as shown in Figure 13.

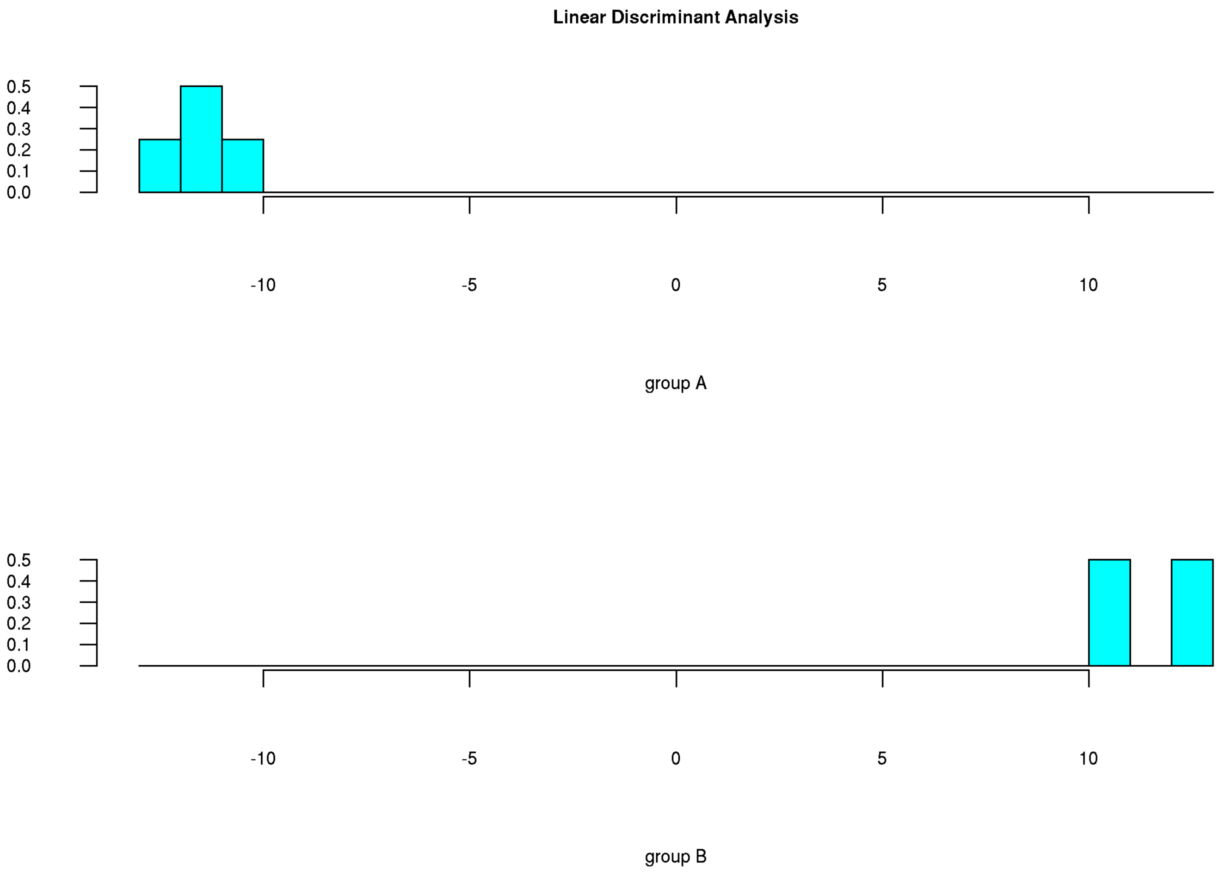


Figure 13: Linear Discriminant Analysis for two groups. This example shows a very clear separation between the groups.

heatmap.r

Produces a hierarchical cluster dendogram for both samples (*x*-axis) and variables (*y*-axis), and uses colours to represent the values. This results in a readily interpretable figure as shown in Figure 14.

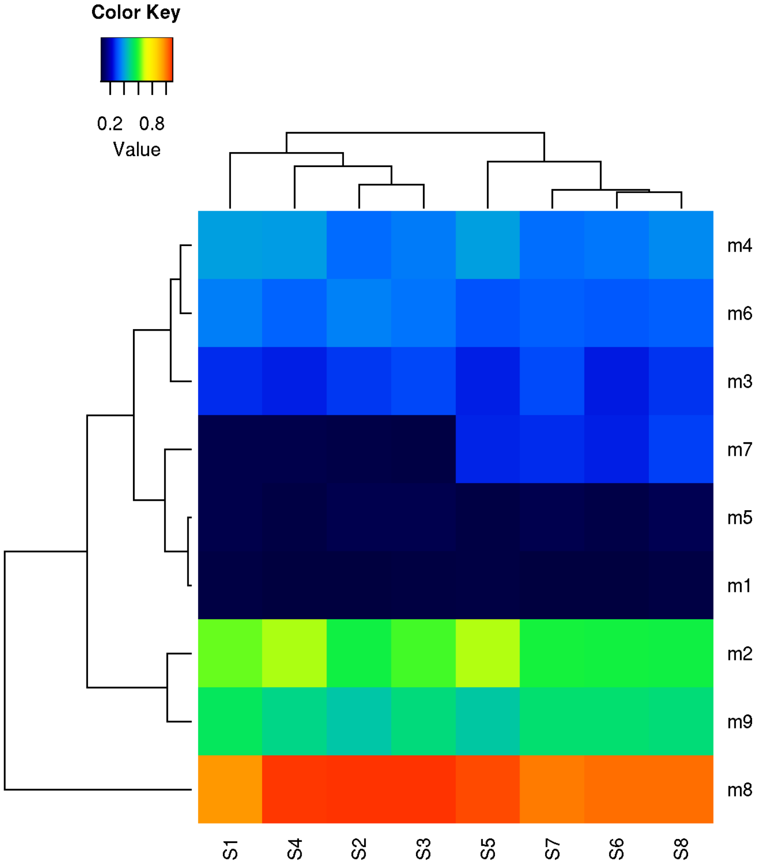


Figure 14: Heatmap showing sample clustering along the x*-axis and metabolites along the* y*-axis. The Color Key* (sic) *shows the value represented by the colours in the body of the plot.*

R Scripts

t_test.r

Performs a Student's *t*-test, and outputs a .csv file with the *P*-values for each variable. This test assumes a normal distribution [`p_t_test.csv`].

wilcoxn.r

Performs Wilcoxon Rank Sum Test, a test that does not assume a normal distribution. Output is a .csv file containing the *P*-values for each varaible [`p_wilcoxon_test.csv`].

fold_chg.r

Calculates the fold change from the (alphanumerically) first group in the data matrix (i.e. for our example input data, this would be fold change from group A). Fold change is calculated by determining the mean for each variable in each sample, and then dividing each mean by the mean for the first group. If the value is less than 1, the returned value is the negative inverse (i.e. if $\mathrm{mean}_{(\mathrm{expt})}/\mathrm{mean}_{(\mathrm{ctrl})} > 1$, no change; if $\mathrm{mean}_{(\mathrm{expt})/}\mathrm{mean}_{(\mathrm{ctrl})} < 1$, fold change = $(-1/\mathrm{mean}_{(\mathrm{expt})}/\mathrm{mean}_{(\mathrm{ctrl}))}$ It returns a table of the means as well as the table of fold changes.

volcano_plot.r

A volcano plot is a type of scatter plot which shows fold change on the *x*-axis and the statistical significance (*P*-values from a *t*-test) on the *y*-axis. It readily shows which metabolites have large magnitude changes that are statistically significant between two conditions.