Computational methods for correcting the drift in LC/MS metabolomic data in R: intCor package vignette

Francesc Fernández-Albert
Polytechnic University of Catalonia
University of Barcelona

Rafael Llorach
University of Barcelona

Cristina Andrés-Lacueva
University of Barcelona

Alexandre Perera
Polytechnic University of Catalonia

Abstract

Liquid Chromatography coupled to mass Spectrometry (LC/MS) has become widely used in Metabolomics. Several artefacts have been identified during the acquisition step in large LC/MS metabolomics experiments, including ion suppression, carryover or changes in the sensitivity and intensity. Several sources have been pointed out as responsible for these effects. In this context, the drift effects of the peak intensity is one of the most frequent and may even constitute the main source of variance in the data, resulting in misleading statistical results when the samples are analysed. In this paper, we propose the introduction of a methodology based on a common variance analysis prior to the data normalisation to address this issue. This methodology was tested and compared with four other methods by calculating the Dunn and Silhouette indices of the Quality Control classes. The results showed that our proposed methodology performed better than any of the other four methods. As far as we know, this is the first time that this kind of approach has been applied in the metabolomics context.

Keywords: Metabolomics, Drift Correction, LC/MS.

1. Using intCor

intCor is an R package focused on drift removal and data normalisation for LC/MS metabolomic data. The package includes five different methods to correct drift effects in the data. It is mainly based on two functions, one for data importation and a second one for correcting the drift effects. Functions to perform graphical analyses (PCA, heat map plots) and to create output cdf files are also included in the package.

1.1. Importing data

The **intCor** package uses the function `importData` to read the samples or data sets that contain the data to be analysed. This function, accepts three different input formats: a set of external files (e.g. cdf or mzXML), a matrix or an `xcmsSet` object. The output of the package could be either a set of cdf files (if the input of the `importData` was a set of files) or a data matrix (regardless of the type of input).

Using an External Data Matrix

One possibility is to import the data through a data table and a class vector. The data matrix should have the samples as columns and variables (time or masses) as rows. The class vector should be a character with the class names. Each one of the components in the vector should have the same ordering than the columns of the data matrix (e.g. the first vector component should refer to the class of the first column of the data matrix). **intCor** contains the data used as cdf in the previous subsection as an RData:

```
R> data(intCorData)
```

The `normInt` object might be constructed as follows:

```
 R> intCor_table<-importData(data=dataMatrix,classes=classes)

The data matrix provided includes peak masses or not? (please answer yes or no)
1: no
Read 1 item
Starting the class-wise outlier detection...
Outlier detection for class Reference in progress...
Outlier removal in drift training data in progress...
Possible score Outliers: 1 2 3 5 6 7 9 10 36
Possible orthogonal Outliers: 1 6

Select sample ID number to be removed
1: 1
2: 6
3:
Read 2 items
Possible score Outliers: 1 2 3 4 5 6 7 8 34
Possible orthogonal Outliers: 2 34

Select extra sample ID number to be removed
1:
Read 0 items
Outlier removal finished
Outlier Removal for class Reference done
```

```
Outlier detection for class Water in progress...
Outlier removal in drift training data in progress...
Possible score Outliers: 1 2 3 4 5 6 7 8 12 15 16 22 23 56 86 89
Possible orthogonal Outliers: 15 23 68

Select sample ID number to be removed
1: 15
2: 23
3: 68
4:
Read 3 items
Possible score Outliers: 1 2 3 4 5 6 7 8 15 21 29 46 54 79 83 84 86
Possible orthogonal Outliers: 83

Select extra sample ID number to be removed
1:
Read 0 items
Outlier removal finished
Outlier Removal for class Water done

Outlier detection for class QC in progress...
Outlier removal in drift training data in progress...
Possible score Outliers: 1 2 3 8 46 47 48
Possible orthogonal Outliers: 47

Select sample ID number to be removed
1: 47
2:
Read 1 item
Possible score Outliers: 1 2 3 4 8 9 10 11 47
Possible orthogonal Outliers:

Select extra sample ID number to be removed
1:
Read 0 items
Outlier removal finished
Outlier Removal for class QC done

Outlier Removal stage QC finished!
```

Using an xcmsSet object

intCor also supports using a `xcmsSet` object (from the **XCMS** package Smith, Want, O'Maille, Abagyan, and Siuzdak (2006)) as an input for the function `importData`. In this case, if the class information is not provided (argument `classes` of the function `importData`), the class

assignment would be retrieved from the `xcmsSet` object. **intCor** also includes an xcmsSet object of the provided sample files:

```
R> data(intCorXCMS)
R> xcg
An "xcmsSet" object with 180 samples

Time range: 17.7-451.2 seconds (0.3-7.5 minutes)
Mass range: 70.0626-683.9618 m/z
Peaks: 99333 (about 552 per sample)
Peak Groups: 526
Sample classes: QC, Reference, Water

Profile settings: method = bin
                  step = 0.1

Memory usage: 8.79 MB
```

The data importation in this case can be computed as:

```
R> normInt_xcms<-importData(data=xcg)
Starting the class-wise outlier detection...
Outlier detection for class QC in progress...
Outlier removal in drift training data in progress...
Possible score Outliers: 1 2 3 4 5 6 7 8 9 10 11 12 19
Possible orthogonal Outliers: 2

Select sample ID number to be removed
1: 2
2:
Read 1 item
Possible score Outliers: 1 2 4 5 6 7
Possible orthogonal Outliers: 1 2 3 4 7 18

Select extra sample ID number to be removed
1:
Read 0 items
Outlier removal finished
Outlier Removal for class QC done

Outlier detection for class Reference in progress...
Outlier removal in drift training data in progress...
Possible score Outliers: 1 2 3 4 5 6 9 10 16 27 36
Possible orthogonal Outliers: 1 5 6 10 16

Select sample ID number to be removed
1: 1
```

```
2: 5
3: 6
4: 10
5: 16
6:
Read 5 items
Possible score Outliers: 1 2 3 4 5 6 7 22 31
Possible orthogonal Outliers: 1 3 22 31

Select extra sample ID number to be removed
1:
Read 0 items
Outlier removal finished
Outlier Removal for class Reference done

Outlier detection for class Water in progress...
Outlier removal in drift training data in progress...
Possible score Outliers: 3 4 8 17 18 19 20 23 24 27 28 29 30 33 42 57 58 85 86 91 92 93 94
Possible orthogonal Outliers: 4

Select sample ID number to be removed
1: 4
2:
Read 1 item
Possible score Outliers: 3 6 7 22 23 26 27 28 29 35 40 41 84 85
Possible orthogonal Outliers:

Select extra sample ID number to be removed
1:
Read 0 items
Outlier removal finished
Outlier Removal for class Water done

Outlier Removal stage Water finished!
```

Using Sample files

When cdf or mzXML files are provided, the **intCor** package performs the drift correction on the chromatograms of the samples. To run the following example, download the data available in our website (http://b2slab.upc.edu/software-and-downloads/intensity-drift-correction/). The zip file contains the data of three different classes (QC, Reference and Water) in cdf format along with a csv file containing the metadata of the samples. If we set the R working directory to where the file `sampTable.csv` is, we can take a look on the provided metadata:

```
R> tab <- read.csv("sampTable.csv")
R> head(tab)
```

```
                                      FileName     Date  Time     Class     Column Batch
1 DM_20121211_POS_pl4_1-M00000000_0_X_X_POS_pl4 12/11/12 14:41 Reference LC-014-MET     4
2         DM_20121211_POS_pl4_1-MQCx1_1_POS_pl4 12/11/12 19:47     Water LC-014-MET     4
3         DM_20121211_POS_pl4_1-MQCx2_1_POS_pl4 12/11/12 20:02        QC LC-014-MET     4
4         DM_20121211_POS_pl4_1-MQC_1_1_POS_pl4 12/11/12 20:16     Water LC-014-MET     4
5         DM_20121211_POS_pl4_2-MQCx1_2_POS_pl4 12/12/12 01:23     Water LC-014-MET     4
6         DM_20121211_POS_pl4_2-MQCx2_2_POS_pl4 12/12/12 01:37        QC LC-014-MET     4
```

we can see that the data frame relates the sample IDs (column FileName) with information regarding the sample recording date and time (columns Date and Time), the class of the sample, the column ID of the chromatograph and the batch number of the sample. Only the columns relating the sample IDs and the classes are mandatory (columns FileName and Class) for the **intCor** package.

A `normInt` object is created by giving the name of the data frame (sampTable.csv) and the extension of the sample files (cdf) to the `importData` function. The function performs a class-wise user-supervised outlier removal stage by default (argument `removeOutliers`) that can be set as automatic (argument `automaticOutlierRemoval`). The function uses the Hotelling method to detect the outliers. For each class, the function gives an estimation of the score and orthogonal outliers and ask to the user for the samples to be removed. Once the sample removal is performed, the Hotelling distances are computed again to look for more outliers of that class. The loop ends when no outliers (blank input) is given. For example in the following, we are going to import the data through the cdf files and remove the samples 1 and 6 of the Reference class, the samples 15, 23 and 68 of the Water class and the sample 47 of the QC class.

```
R> library(intCor)

R> normInt<-importData(dataDir="data", fileType="cdf", tabName="sampTable.csv")

Reading input table...Done
Extraction of the Total Ion Chromatograms initiated...
 % done: 10  20  30  40  50  60  70  80  90  100  Extraction of the Total Ion Chromatogram
Starting the class-wise outlier detection...
Outlier detection for class Reference in progress...
Outlier removal in drift training data in progress...
Possible score Outliers: 1 2 3 5 6 7 9 10 36
Possible orthogonal Outliers: 1 6

Select sample ID number to be removed
1: 1
2: 6
3:
```

```
Read 2 items
Possible score Outliers: 1 2 3 4 5 6 7 8 34
Possible orthogonal Outliers: 2 34

Select extra sample ID number to be removed
1:
Read 0 items
Outlier removal finished
Outlier Removal for class Reference done

Outlier detection for class Water in progress...
Outlier removal in drift training data in progress...
Possible score Outliers: 1 2 3 4 5 6 7 8 12 15 16 22 23 56 86 89
Possible orthogonal Outliers: 15 23 68

Select sample ID number to be removed
1: 15
2: 23
3: 68
4:
Read 3 items
Possible score Outliers: 1 2 3 4 5 6 7 8 15 21 29 46 54 79 83 84 86
Possible orthogonal Outliers: 83

Select extra sample ID number to be removed
1:
Read 0 items
Outlier removal finished
Outlier Removal for class Water done

Outlier detection for class QC in progress...
Outlier removal in drift training data in progress...
Possible score Outliers: 1 2 3 8 46 47 48
Possible orthogonal Outliers: 47

Select sample ID number to be removed
1: 47
2:
Read 1 item
Possible score Outliers: 1 2 3 4 8 9 10 11 47
Possible orthogonal Outliers:

Select extra sample ID number to be removed
1:
Read 0 items
Outlier removal finished
Outlier Removal for class QC done
```

```
Outlier Removal stage QC finished!
```

1.2. Correcting the drift in the data

Once the run of the `importData` function is finished, the data have been imported into a `normInt` object along with the metadata stored in the table:

```
R> normInt
normInt object with a retention time range of [0.333,7.502] seconds containing:
93 samples of the class Water
47 samples of the class QC
34 samples of the class Reference
The data has not been normalised
```

We can also retrieve the information regarding the outlier removal stage by running a summary of the object:

```
R> summary(normInt)
normInt object with retention a time range of [0.333,7.502] seconds containing:
93 samples of the class Water
47 samples of the class QC
34 samples of the class Reference

There were removed the following outliers:
2 samples of class Water
3 samples of class QC
1 samples of class Reference
The data has not been normalised
```

It is also straightforward to run a PCA Scoreplot of the data (see Figure 1 for the PCA score plots regarding the raw data and their class and time labels) by running the `pcaPlot` function on the `normInt` object:

```
R> pcaPlot(normInt)
```

At this point, we can correct the drift effects in the data though the `corrModel` function. The function supports five different methods: Component Correction (function argument `method="cc"`), Common Principal Components analysis (function argument `method="cpca"`), Common Principal Components Analysis + Median Normalisation (function argument `method="cpcaMed"`), Median Normalisation (function argument `method="medians"`) and Batch compensation through

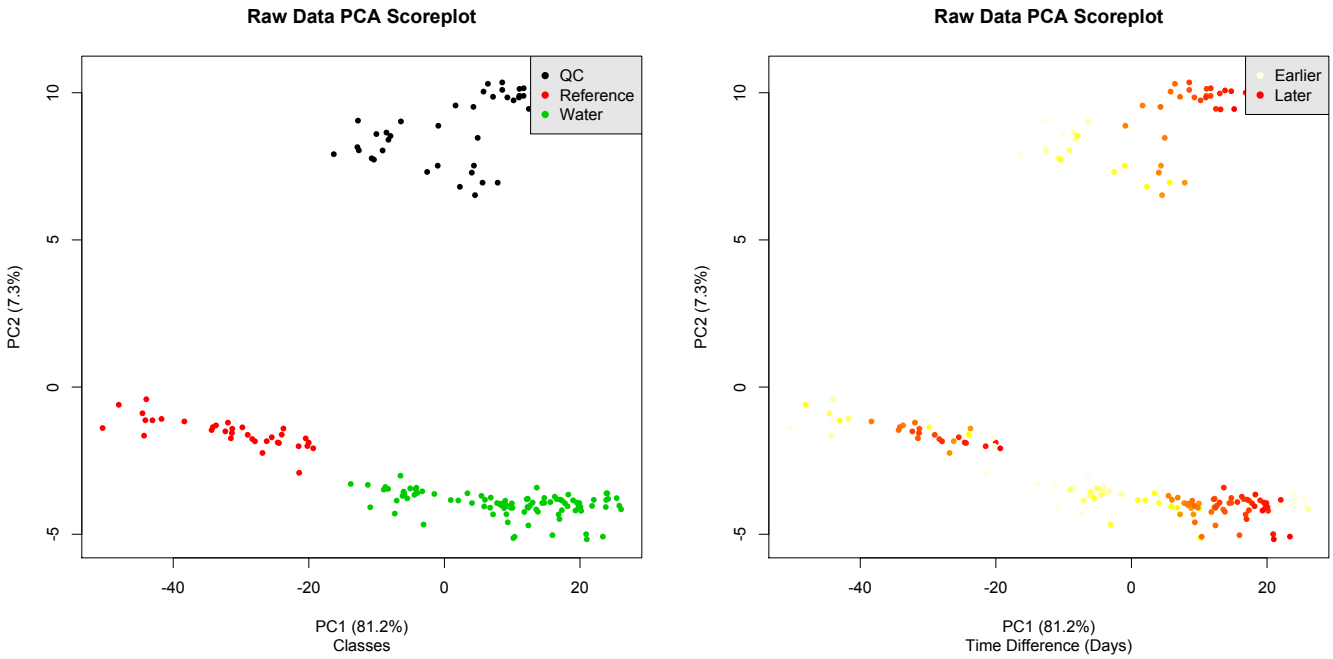

Figure 1: Raw Data score plots. The left plot depicts the samples into the plane PC1-PC2 using the class labels whereas the right plot shows the same PCA score plot but using the time labels.

the ComBat function (function argument `method="batch"`) Leek, Johnson, Parker, Jaffe, and Storey (2012).

In the next command, we correct the data using the Common Principal Components Analysis + Median Normalisation method. One common principal component was selected for removal and all the classes were taken for generating the model:

```
normInt_cpcaMed_1C <- corrModel(normInt=normInt,method="cpcaMed",
            modClasses=c("Water","QC","Reference"),nComps=1)

Detected available classes to select for correction:
       QC Reference     Water
       47        34        93
Selected classes to compute the drift model:
[1] "Water"     "QC"        "Reference"

Computing CPC model...Model done!

Computing CPC explained variance...
CPC1
 0.7
Correcting Intensity Drift...
Computing clustering indices ...
```

```
Raw Data
, , kmeans

                     3
Connectivity 14.5682540
Dunn          0.0242528
Silhouette    0.5038728

, , hierarchical

                     3
Connectivity 3.64246032
Dunn         0.07857024
Silhouette   0.49989042

Corrected Data
, , kmeans

                    3
Connectivity 0.0000000
Dunn         0.9661138
Silhouette   0.8929673

, , hierarchical

                    3
Connectivity 0.0000000
Dunn         0.9661138
Silhouette   0.8929673
```

Figure 2 depicts the PCA score plots for the corrected data set. The comparison between the clustering indices can be retrieved by running a summary of the `normInt` object:

```
R> summary(normInt_cpcaMed_1C)
normInt object with retention a time range of [0.333,7.502] seconds containing:
93 samples of the class Water
47 samples of the class QC
34 samples of the class Reference

There were removed the following outliers:
2 samples of class Water
3 samples of class QC
1 samples of class Reference

The data was normalised using the cpcaMed method
The variance captured for each component was
```

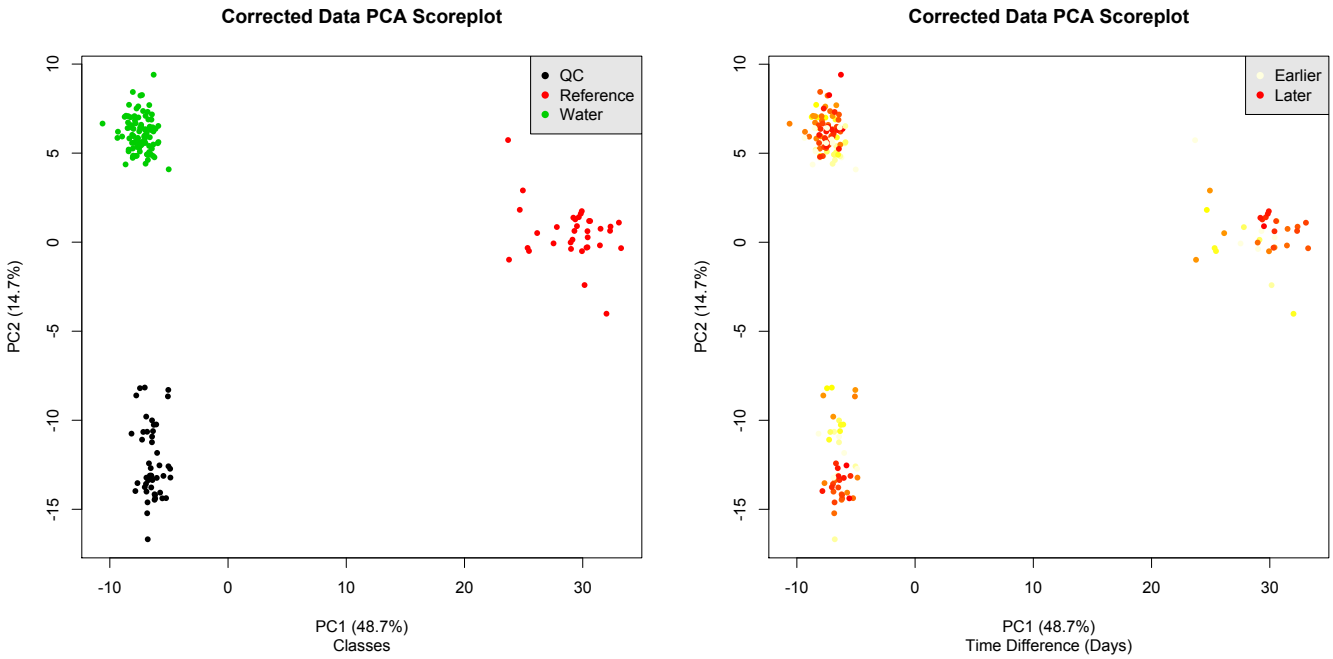

Figure 2: PCA score plots of the corrected data using the Common Principal Components + Median Normalisation method. The class assignment (left) and the time elapsed since the first sample injection (right) were used as labelling.

```
0.7 for component 1

The computed Dunn indices before and after the normalisation:
Raw Dunn index =  0.0242528
Corrected Dunn index =  0.9661138

The computed Silhouette indices before and after the normalisation:
Raw Silhouette index =  0.5038728
Corrected Silhouette index =  0.8929673
```

The clustering indices show an important improvement after the data correction. The other methods are launched in a similar way, for example the method regarding the ComBat function:

```
normInt_comBat<-corrModel(normInt=normInt,method="batch")
Defining Model Matrix
Correcting Batch effects...Found 12 batches
Found 2  categorical covariate(s)
Standardizing Data across genes
Fitting L/S model and finding priors
Finding parametric adjustments
Adjusting the Data
Done
```

```
Computing explained variance...
[1] 0.881

Computing clustering indices ...
Raw Data
, , kmeans

                     3
Connectivity 14.5682540
Dunn          0.0242528
Silhouette    0.5038728

, , hierarchical

                     3
Connectivity 3.64246032
Dunn         0.07857024
Silhouette   0.49989042

Corrected Data
, , kmeans

                    3
Connectivity 0.0000000
Dunn         0.6363887
Silhouette   0.8027747

, , hierarchical

                    3
Connectivity 0.0000000
Dunn         0.6363887
Silhouette   0.8027747
```

gave the PCA score plots in the Figure 3, or the medians method:

```
R> normInt_medians<-corrModel(normInt=normInt,method="medians")

Computing clustering indices ...
Raw Data
, , kmeans

                     3
Connectivity 14.5682540
Dunn          0.0242528
Silhouette    0.5038728
```

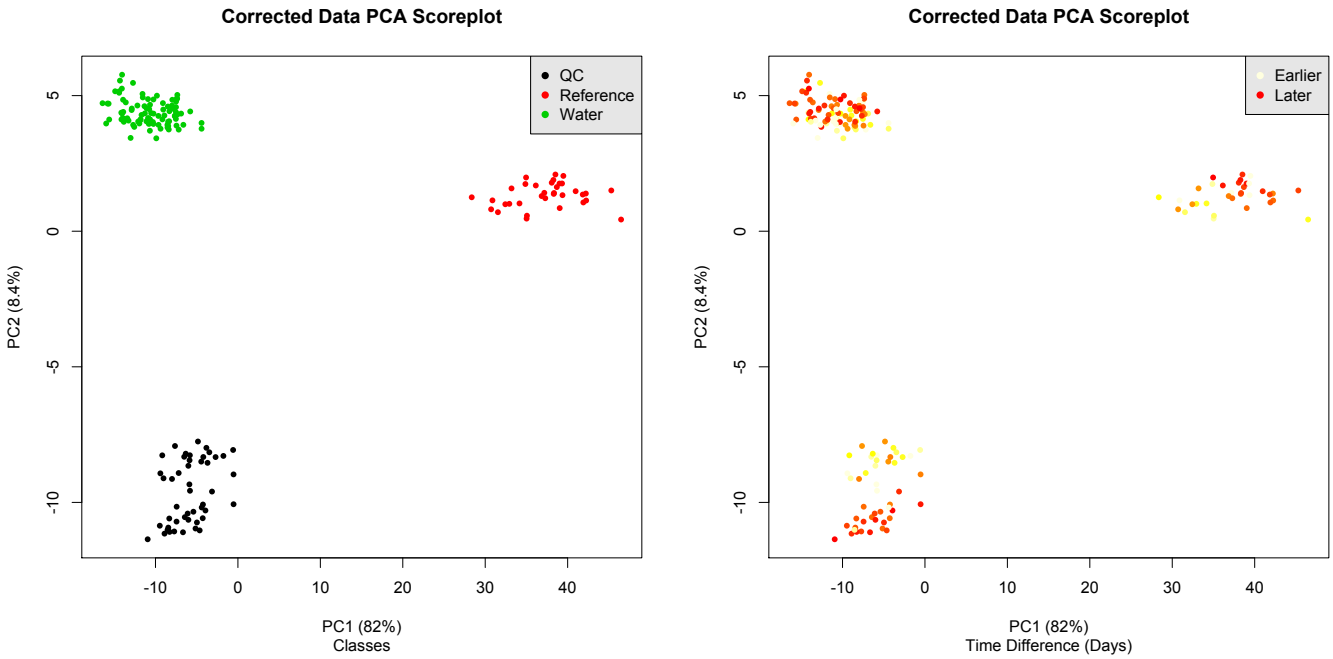

Figure 3: PCA score plots of the corrected data using the Batch compensation (ComBat function) method. The class assignment (left) and the time elapsed since the first sample injection (right) were used as labelling.

```
, , hierarchical

                    3
Connectivity 3.64246032
Dunn         0.07857024
Silhouette   0.49989042

Corrected Data
, , kmeans

                    3
Connectivity 0.0000000
Dunn         0.7873576
Silhouette   0.8622102

, , hierarchical

                    3
Connectivity 0.0000000
Dunn         0.7873576
Silhouette   0.8622102
```

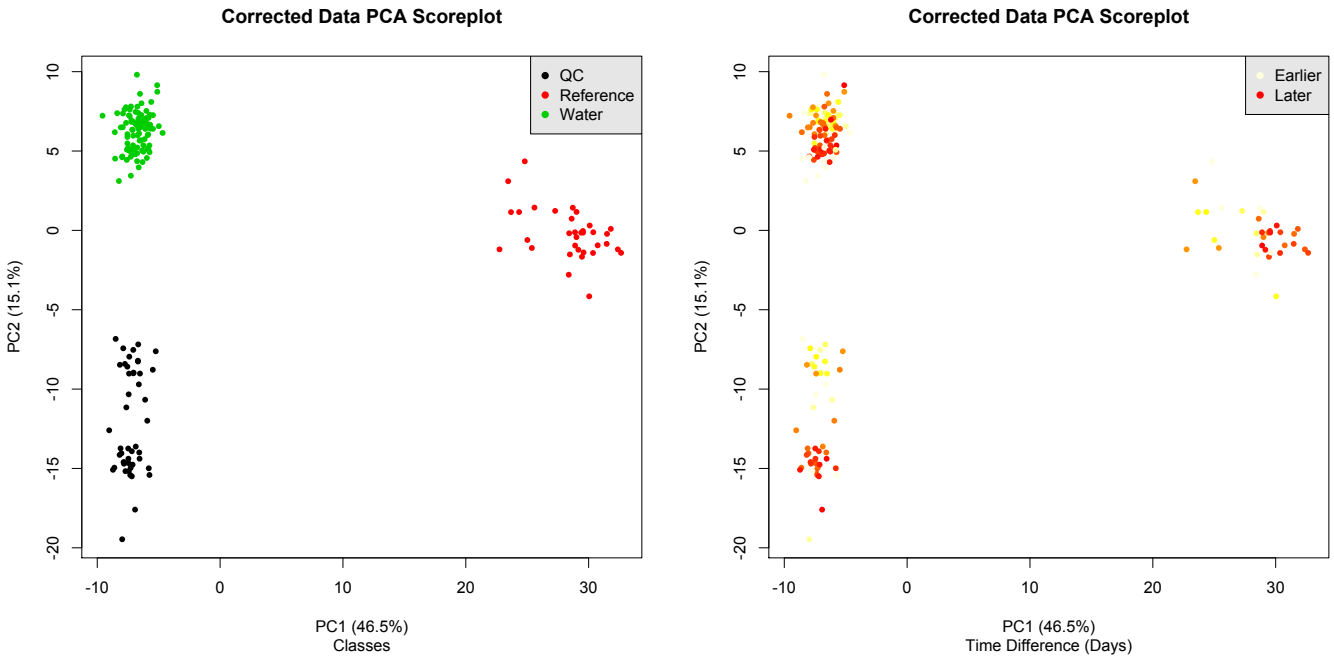

Figure 4: PCA score plots of the corrected data using the Median Normalisation method. The class assignment (left) and the time elapsed since the first sample injection (right) were used as labelling.

which gave the PCA score plots depicted in Figure 4

1.3. intCor Output

If the data was imported through external files, there is the possibility of printing cdf files back as an output. For example, to print the cdf files of the corrected data through the `cpcaMed` method, we can use the function `cdfFileCreator` as:

```
R> cdfFileCreator(dataMatrix=getCorrData(normInt_cpcaMed_1C),dir2print="correctedCDFs",
                  dataDir="data",fileType="cdf",scanRange=NULL)
```

Finally, another possibility that can be used regardless of the data importation method, is to retrieve the corrected table using the function `getCorrData` and (optionally) print a CSV file:

```
R> write.csv(file="corrData_cpcaMed_1C.csv",getCorrData(normInt_cpcaMed_1C))
```

2. Acknowledgements

This research was supported by Spanish national grants AGL2009-13906-C02-01/ALI, AGL2010-10084-E, the CONSOLIDER INGENIO 2010 Programme and FUN-C-FOOD (CSD2007-063)

under the MICINN, as well as Merck Serono 2010 Research Grants (Fundación Salud 2000). R. Llorach thanks the MICINN and The European Social Funds for their financial contribution to the R. L. Ramón y Cajal contract (Ramon y Cajal Programme, MICINN-RYC). This work has been partially supported by the Spanish Ministerio de Ciencia y Tecnología through the TEC2010-20886-C02-02 and TEC2010-20886-C02-01 grants, and the Ramon y Cajal programme. A. Perera is part of the 2009SGR-1395 consolidated research group of the Generalitat de Catalunya, Spain. CIBER-BBN is an initiative of the Spanish ISCIII. F. FernÃąndez-Albert thanks EVALXARTA-UB and Agència de Gestió d'Ajuts Universitaris I de Recerca, AGAUR (Generalitat de Catalunya) for their financial support.

References

Leek JT, Johnson WE, Parker HS, Jaffe AE, Storey JD (2012). "The sva package for removing batch effects and other unwanted variation in high-throughput experiments." *Bioinformatics*, **28**(6), 882–883. http://bioinformatics.oxfordjournals.org/content/28/6/882.full.pdf+html.

Smith CA, Want EJ, O'Maille G, Abagyan R, Siuzdak G (2006). "XCMS: processing mass spectrometry data for metabolite profiling using nonlinear peak alignment, matching, and identification." *Analytical Chemistry*, **78**(3), 779–787. ISSN 00032700. doi:10.1021/ac051437y. URL http://pubs3.acs.org/acs/journals/doilookup?in_doi=10.1021/ac051437y.

Affiliation:

Francesc Fernández-Albert and Alexandre Perera
Department d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial
Universitat Politècnica de Catalunya
Barcelona, Spain
E-mail: francesc.fernandez.albert@upc.edu
Alexandre.Perera@upc.edu

Francesc Fernández-Albert, Rafael Llorach and Cristina Andrés-Lacueva
Nutrition and Food Science Department, XaRTA INSA, INGENIO-CONSOLIDER Program, FUN-C-Food CSD2007-063
Avinguda Joan XXIII sn, 08028 Barcelona
Pharmacy School
University of Barcelona, Spain
Barcelona, Spain
E-mail: rafallorach@ub.edu
candres@ub.edu