ropls: PCA, PLS(-DA) and OPLS(-DA) for multivariate analysis and feature selection of omics data

Etienne A. Thevenot

24 April 2017

Package version: ropls 1.8.0

Contents

1 The *ropls* package

The ropls R package implements the **PCA**, **PLS(-DA)** and **OPLS(-DA)** approaches with the original, **NIPALS**-based, versions of the algorithms (Wold, Sjostrom, and Eriksson 2001, J. Trygg and Wold (2002)). It includes the **R2** and **Q2** quality metrics (Eriksson et al. 2001, Tenenhaus (1998)), the permutation **diagnostics** (Szymanska et al. 2012), the computation of the **VIP** values (Wold, Sjostrom, and Eriksson 2001), the score and orthogonal distances to detect **outliers** (Hubert, Rousseeuw, and Vanden Branden 2005), as well as many **graphics** (scores, loadings, predictions, diagnostics, outliers, etc).

The functionalities from ropls can also be accessed via a graphical user interface in the **Multivariate** module from the Workflow4Metabolomics.org online resource for computational metabolomics, which provides a user-friendly, **Galaxy**-based environment for data pre-processing, statistical analysis, and annotation (Giacomoni et al. 2015).

2 Context

2.1 Orthogonal Partial Least-Squares

Partial Least-Squares (PLS), which is a latent variable regression method based on covariance between the predictors and the response, has been shown to efficiently handle datasets with multi-collinear predictors, as in the case of spectrometry measurements (Wold, Sjostrom, and Eriksson 2001). More recently, J. Trygg and Wold (2002) introduced the **Orthogonal Partial Least-Squares (OPLS)** algorithm to model separately the variations of the predictors correlated and orthogonal to the response.

OPLS has a similar predictive capacity compared to PLS and improves the **interpretation** of the predictive components and of the systematic variation (Pinto, Trygg, and Gottfries 2012). In particular, OPLS modeling of single responses only requires one predictive component.

Diagnostics such as the **Q2Y** metrics and permutation testing are of high importance to **avoid overfitting** and assess the statistical significance of the model. The **Variable Importance in Projection (VIP)**, which reflects both the loading weights for each component and the variability of the response explained by this component (Pinto, Trygg, and Gottfries 2012; Mehmood et al. 2012), can be used for feature selection (J. Trygg and Wold 2002; Pinto, Trygg, and Gottfries 2012).

2.2 OPLS software

OPLS is available in the **SIMCA-P** commercial software (Umetrics, Umea, Sweden; Eriksson et al. (2001)). In addition, the kernel-based version of OPLS (M. Bylesjo et al. 2008) is available in the open-source R statistical environment (R Development Core Team 2008), and a single implementation of the linear algorithm in R has been described recently (Gaude et al. 2013).

3 The *sacurine* metabolomics dataset

3.1 Study objective

The objective was to study the influence of **age**, **body mass index (bmi)**, and **gender** on metabolite concentrations in **urine**, by analysing 183 samples from a **cohort** of adults with liquid chromatography coupled to high-resolution mass spectrometry (**LC-HRMS**; Thevenot et al. (2015)).

3.2 Pre-processing and annotation

Urine samples were analyzed by using an LTQ Orbitrap in the negative ionization mode. A total of **109 metabolites** were identified or annotated at the MSI level 1 or 2. After retention time alignment with XCMS, peaks were integrated with Quan Browser. Signal drift and batch effect were corrected, and each urine profile was normalized to the osmolality of the sample. Finally, the data were log10 transformed (Thevenot et al. 2015).

3.3 Covariates

The volunteers' **age**, **body mass index (bmi)**, and **gender** were recorded.

4 Hands-on

4.1 Loading

We first load the ropls package:

```
library(ropls)
```

We then load the **sacurine** dataset which contains:

1. The **dataMatrix** matrix of numeric type containing the intensity profiles (log10 transformed),
2. The **sampleMetadata** data frame containg sample metadata,
3. The **variableMetadata** data frame containg variable metadata

```
data(sacurine)
names(sacurine)
## [1] "dataMatrix"       "sampleMetadata"   "variableMetadata"
```

We attach **sacurine** to the search path and display a summary of the content of the **dataMatrix**, **sampleMetadata** and **variableMetadata** with the `strF` *F*unction of the ropls package (see also `str`):

```
attach(sacurine)
```

```
strF(dataMatrix)
##        dim  class    mode typeof   size NAs  min mean median max
##  183 x 109 matrix numeric double 0.2 Mb   0 -0.3  4.2    4.3   6
##        (2-methoxyethoxy)propanoic acid isomer (gamma)Glu-Leu/Ile ...
## HU_011                             3.019766011        3.888479324 ...
## HU_014                              3.81433889        4.277148905 ...
## ...                                        ...                ... ...
## HU_208                             3.748127215        4.523763202 ...
## HU_209                             4.208859398        4.675880567 ...
##        Valerylglycine isomer 2  Xanthosine
## HU_011             3.889078716 4.075879575
## HU_014             4.181765852 4.195761901
## ...                        ...         ...
## HU_208             4.634338821 4.487781609
## HU_209              4.47194762 4.222953354
strF(sampleMetadata)
##      age     bmi gender
##  numeric numeric factor
##  nRow nCol size NAs
##   183    3 0 Mb   0
##         age   bmi gender
## HU_011   29 19.75      M
## HU_014   59 22.64      F
## ...     ...   ...    ...
## HU_208   27 18.61      F
## HU_209 17.5 21.48      F
strF(variableMetadata)
##  msiLevel      hmdb chemicalClass
##   numeric character     character
##  nRow nCol size NAs
##   109    3 0 Mb   0
##                                       msiLevel      hmdb chemicalClass
## (2-methoxyethoxy)propanoic acid isomer        2                  Organi
```

```
## (gamma)Glu-Leu/Ile                              2                  AA-pep
## ...                                           ...       ...           ...
## Valerylglycine isomer 2                         2           AA-pep:AcyGly
## Xanthosine                                      1 HMDB00299        Nucleo
```

4.2 Principal Component Analysis (PCA)

We perform a **PCA** on the **dataMatrix** matrix (samples as rows, variables as columns), with the `opls` method:

```
sacurine.pca <- opls(dataMatrix)
```

A **summary** of the model (8 components were selected) is printed:

```
## PCA
## 183 samples x 109 variables
## standard scaling of predictors
##       R2X(cum) pre ort
## Total    0.501   8   0
```

In addition the default **summary** figure is displayed:

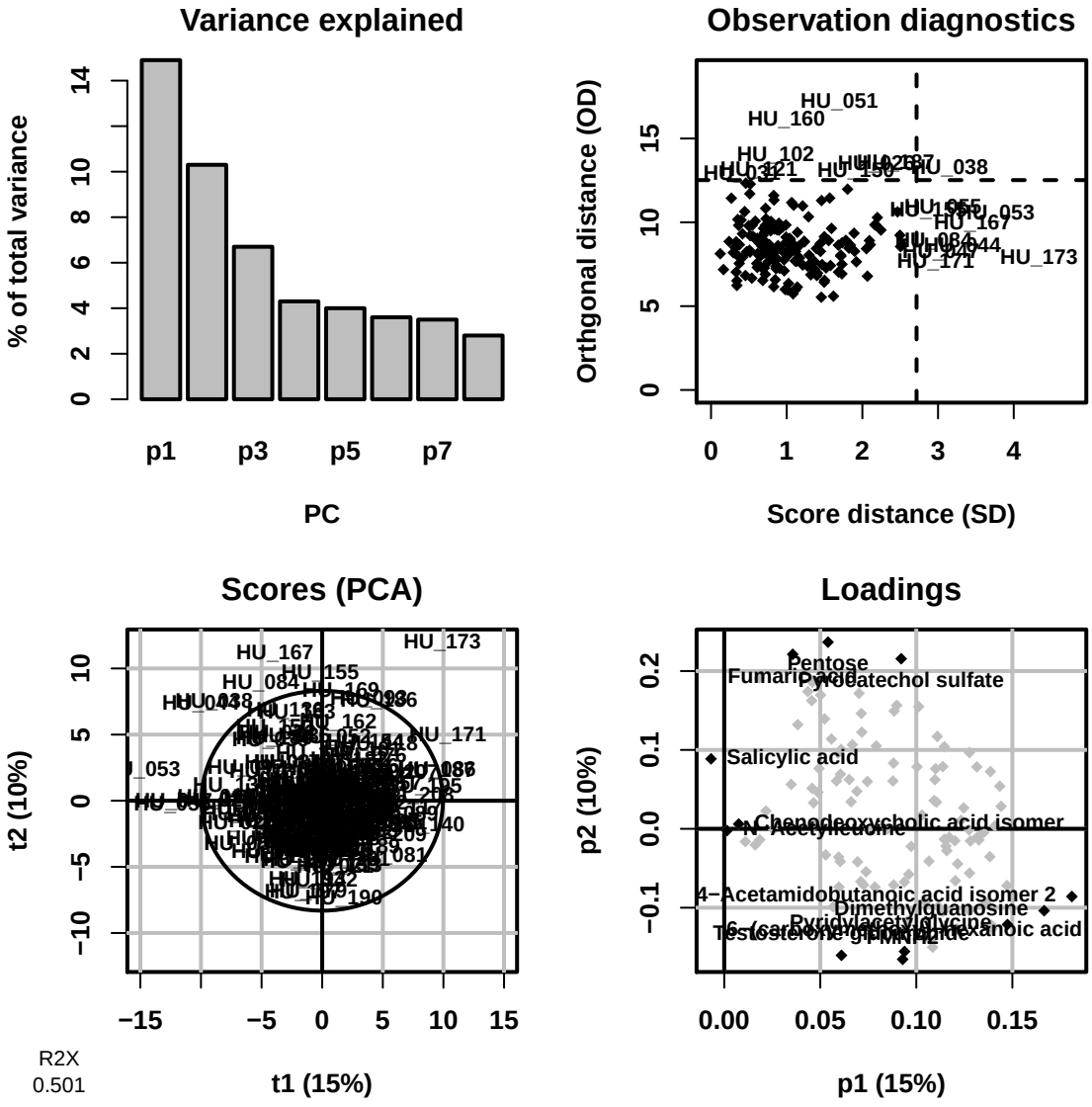


Figure 1: PCA summary plot. Top left `overview`: the **scree** plot (i.e., inertia barplot) suggests that 3 components may be sufficient to capture most of the inertia; **Top right** `outlier`: this graphics shows the distances within and orthogonal to the projection plane (Hubert, Rousseeuw, and Vanden Branden 2005): the name of the samples with a high value for at least one of the distances are indicated; **Bottom left** `x-score`: the variance along each axis equals the variance captured by each component: it therefore depends on the total variance of the dataMatrix X and of the percentage of this variance captured by the component (indicated in the labels); it decreases when going from one component to a component with higher indice; **Bottom right** `x-loading`: the 3 variables with most extreme values (positive and negative) for each loading are black colored and labeled.

Note:

1. Since **dataMatrix** does not contain missing value, the singular value decomposition was used by default; NIPALS can be selected with the `algoC` argument specifying the *algo*rithm (*C*haracter),
2. The `predI = NA` default number of *pred*ictive components (*I*nteger) for PCA means that components (up to 10) will be computed until the cumulative variance exceeds 50%. If the 50% have not been reached at the 10th component, a warning message will be issued (you can still compute the following components by specifying the `predI` value).

Let us see if we notice any partition according to gender, by labeling/coloring the samples according to *gender*

(parAsColFcVn) and drawing the Mahalanobis ellipses for the male and female subgroups (parEllipseL).

```
genderFc <- sampleMetadata[, "gender"]
plot(sacurine.pca, typeVc = "x-score",
parAsColFcVn = genderFc, parEllipsesL = TRUE)
```

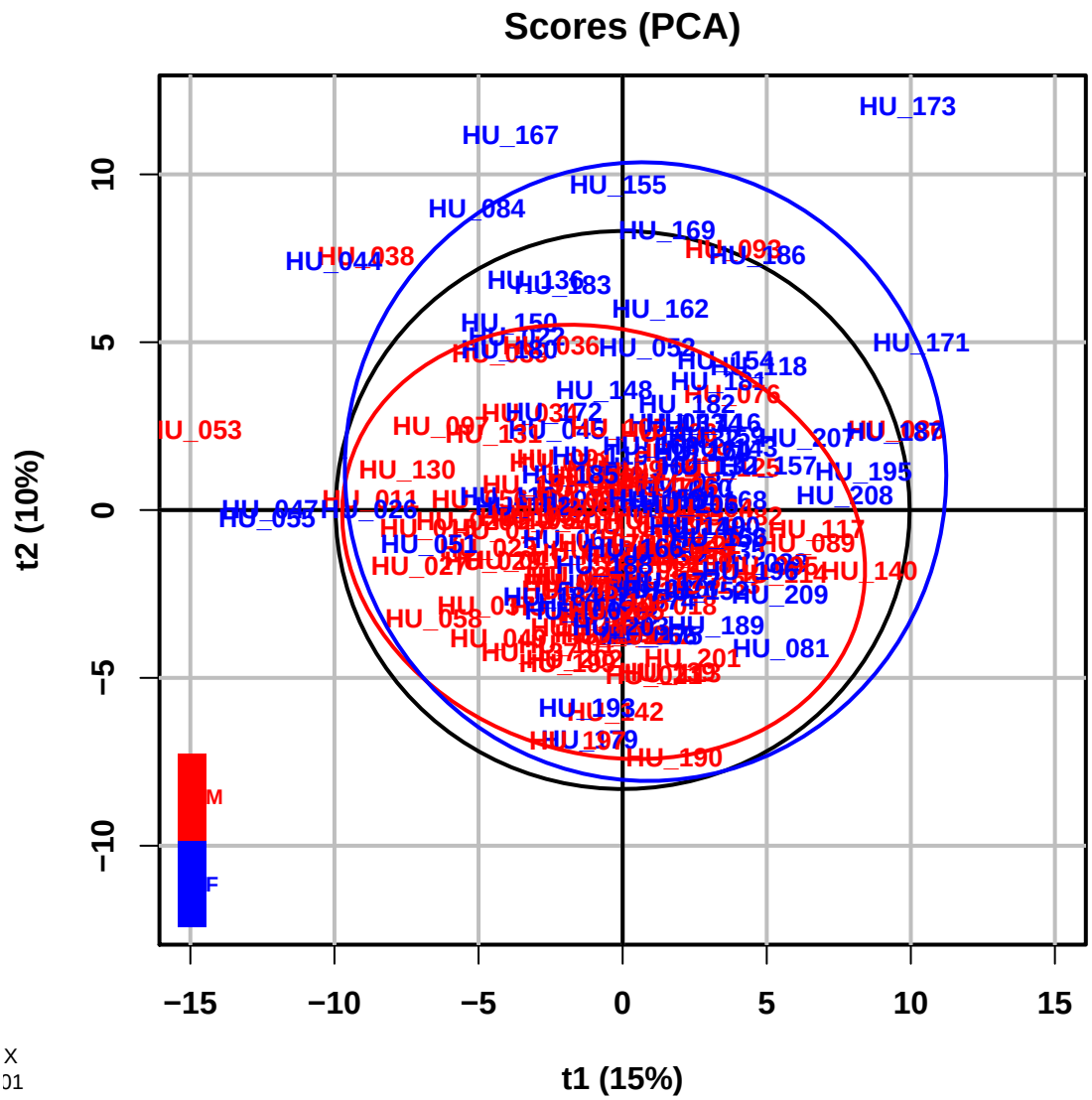


Figure 2: PCA score plot colored according to gender.

Note:

1. The plotting *par*ameter to be used *As Col*ors (*F*actor of *c*haracter type or *V*ector of *n*umeric type) has a length equal to the number of rows of the **dataMatrix** (ie of samples) and that this qualitative or quantitative variable is converted into colors (by using an internal palette or color scale, respectively). We could have visualized the *age* of the individuals by specifying `parAsColFcVn = sampleMetadata[, "age"]`.
2. The displayed components can be specified with `parCompVi` (plotting *par*ameter specifying the *Comp*onents: *V*ector of 2 *i*ntegers)

4.3 Partial least-squares: PLS and PLS-DA

For **PLS (and OPLS)**, the **Y** response(s) must be provided to the `opls` method. **Y** can be either a numeric vector (respectively matrix) for single (respectively multiple) **(O)PLS regression**, or a character factor for **(O)PLS-DA classification** as in the following example with the *gender* qualitative response:

```
sacurine.plsda <- opls(dataMatrix, genderFc)
```

```
## PLS-DA
## 183 samples x 109 variables and 1 response
## standard scaling of predictors and response(s)
##       R2X(cum) R2Y(cum) Q2(cum) RMSEE pre ort pR2Y  pQ2
## Total    0.275     0.73   0.584 0.262   3   0 0.05 0.05
```

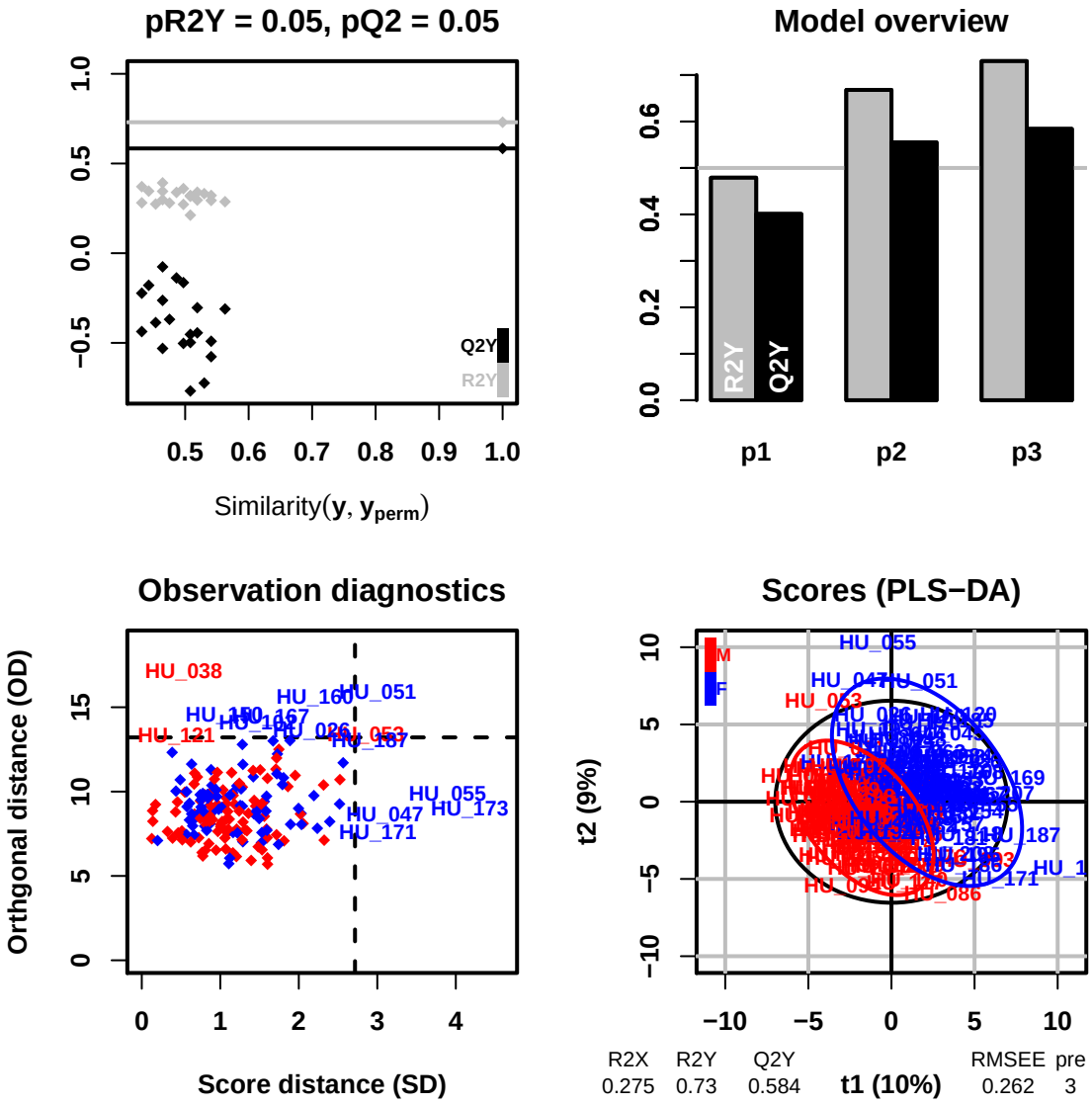


Figure 3: PLS-DA model of the gender response. Top left: `significance` diagnostic: the **R2Y** and **Q2Y** of the model are compared with the corresponding values obtained after random permutation of the *y* response; **Top right**: `inertia` barplot: the graphic here suggests that 3 orthogonal components may be sufficient to capture most of the inertia; **Bottom left**: `outlier` diagnostics; **Bottom right**: `x-score` plot: the number of components and the cumulative **R2X**, **R2Y** and **Q2Y** are indicated below the plot.

Note:

1. When set to NA (as in the default), **the number of components** `predI` is determined automatically as follows (Eriksson et al. 2001): A new component *h* is added to the model if:

- $R2Y_h \geq 0.01$, i.e., the percentage of **Y** dispersion (i.e., sum of squares) explained by component *h* is more than 1 percent, and
- $Q2Y_h = 1 - PRESS_h / RSS_{h-1} \geq 0$ for PLS (or 5% when the number of samples is less than 100) or 1% for OPLS: $Q2Y_h \geq 0$ means that the predicted residual sum of squares ($PRESS_h$) of the model including the new component *h* estimated by 7-fold cross-validation is less than the residual sum of squares (RSS_{h-1}) of the model with the previous components only (with RSS_0 being the sum of squared **Y** values).

2. The **predictive performance** of the full model is assessed by the cumulative **Q2Y** metric: $Q2Y = 1 - \prod_{h=1}^{r}(1 - Q2Y_h)$. We have $Q2Y \in [0, 1]$, and the higher the **Q2Y**, the better the performance. Models trained on datasets with a larger number of features compared with the number of samples can be prone to **overfitting**: in that case, high **Q2Y** values are obtained by chance only. To estimate the significance of **Q2Y** (and **R2Y**) for single response models, permutation testing (Szymanska et al. 2012) can be used: models are built after random permutation of the **Y** values, and $Q2Y_{perm}$ are computed. The *p*-value is equal to the proportion of $Q2Y_{perm}$ above $Q2Y$ (the **default number of permutations is 20** as a compromise between quality control and computation speed; it can be increased with the `permI` parameter, e.g. to 1,000, to assess if the model is significant at the 0.05 level),
3. The **NIPALS** algorithm is used for PLS (and OPLS); *dataMatrix* matrices with (a moderate amount of) missing values can thus be analysed.

We see that our model with 3 predictive (*pre*) components has significant and quite high R2Y and Q2Y values.

4.4 Orthogonal partial least squares: OPLS and OPLS-DA

To perform **OPLS(-DA)**, we set `orthoI` (number of components which are *ortho*gonal; *I*nteger) to either a specific number of orthogonal components, or to NA. Let us build an OPLS-DA model of the *gender* response.

```
sacurine.oplsda <- opls(dataMatrix, genderFc,
predI = 1, orthoI = NA)
```

```
## OPLS-DA
## 183 samples x 109 variables and 1 response
## standard scaling of predictors and response(s)
##       R2X(cum) R2Y(cum) Q2(cum) RMSEE pre ort pR2Y  pQ2
## Total    0.275     0.73   0.602 0.262   1   2 0.05 0.05
```

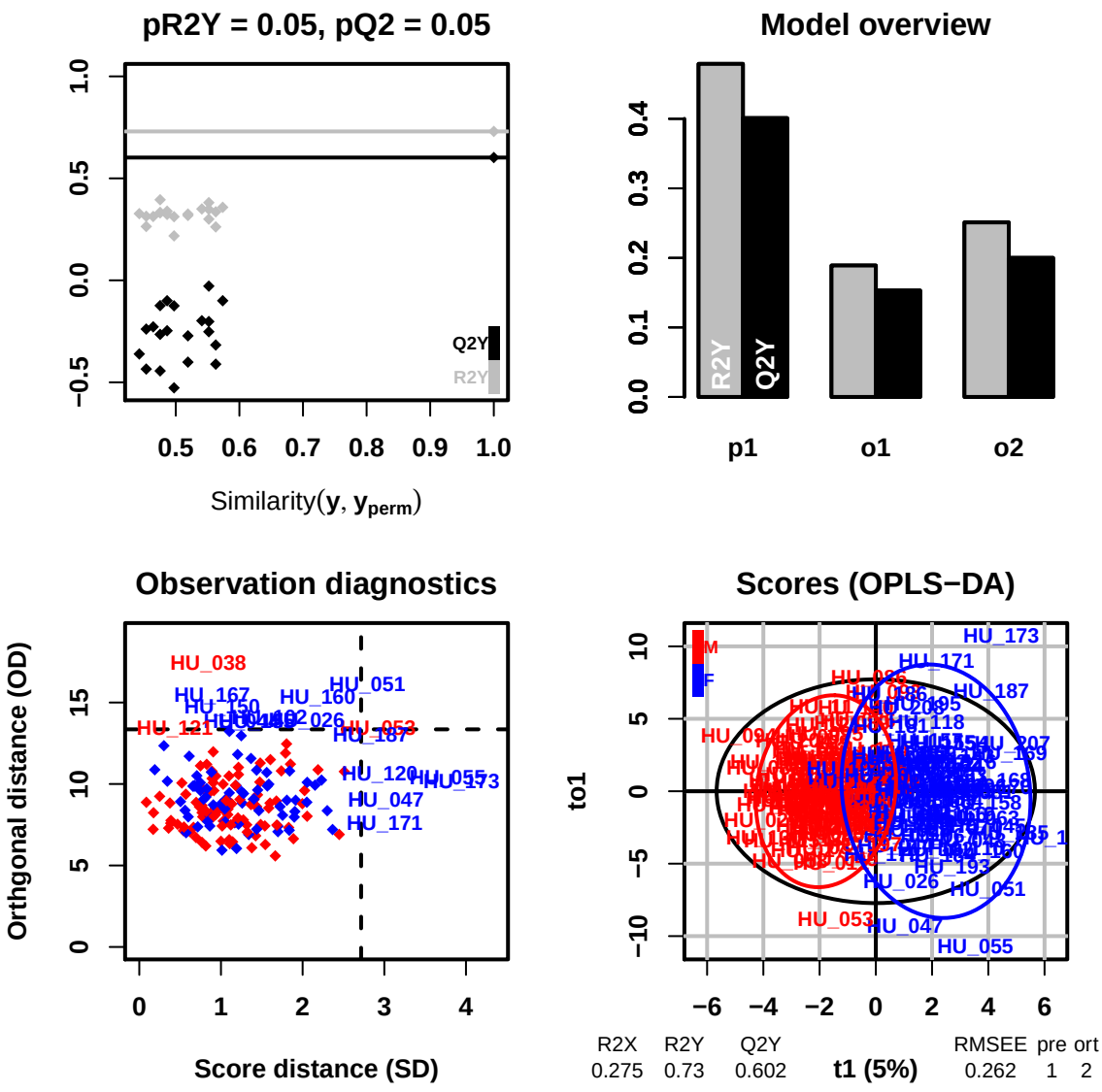


Figure 4: OPLS-DA model of the gender response.

Note:

1. For OPLS modeling of a single response, the number of predictive component is 1,
2. In the (`x-score` plot), the predictive component is displayed as abscissa and the (selected; default = 1) orthogonal component as ordinate.

Let us assess the **predictive performance** of our model. We first train the model on a subset of the samples (here we use the odd subset value which splits the data set into two halves with similar proportions of samples for each class; alternatively, we could have used a specific subset of indices for training):

```
sacurine.oplsda <- opls(dataMatrix, genderFc, predI = 1, orthoI = NA,
subset = "odd")
```

```
## OPLS-DA
## 92 samples x 109 variables and 1 response
## standard scaling of predictors and response(s)
##       R2X(cum) R2Y(cum) Q2(cum) RMSEE RMSEP pre ort
## Total     0.26    0.825   0.608 0.213 0.341   1   2
```

We first check the predictions on the **training** subset:

```
trainVi <- getSubsetVi(sacurine.oplsda)
table(genderFc[trainVi], fitted(sacurine.oplsda))
##
##      M  F
##   M 50  0
##   F  0 42
```

We then compute the performances on the **test** subset:

```
table(genderFc[-trainVi],
      predict(sacurine.oplsda, dataMatrix[-trainVi, ]))
##
##      M  F
##   M 43  7
##   F  7 34
```

As expected, the predictions on the test subset are (slightly) lower. The classifier however still achieves 91% of correct predictions.

4.5 Comments

4.5.1 Overfitting

Overfitting (i.e., building a model with good performances on the training set but poor performances on a new test set) is a major caveat of machine learning techniques applied to data sets with more variables than samples. A simple simulation of a random **X** data set and a **y** response shows that perfect PLS-DA classification can be achieved as soon as the number of variables exceeds the number of samples, as detailed in the example below, adapted from Wehrens (2011):

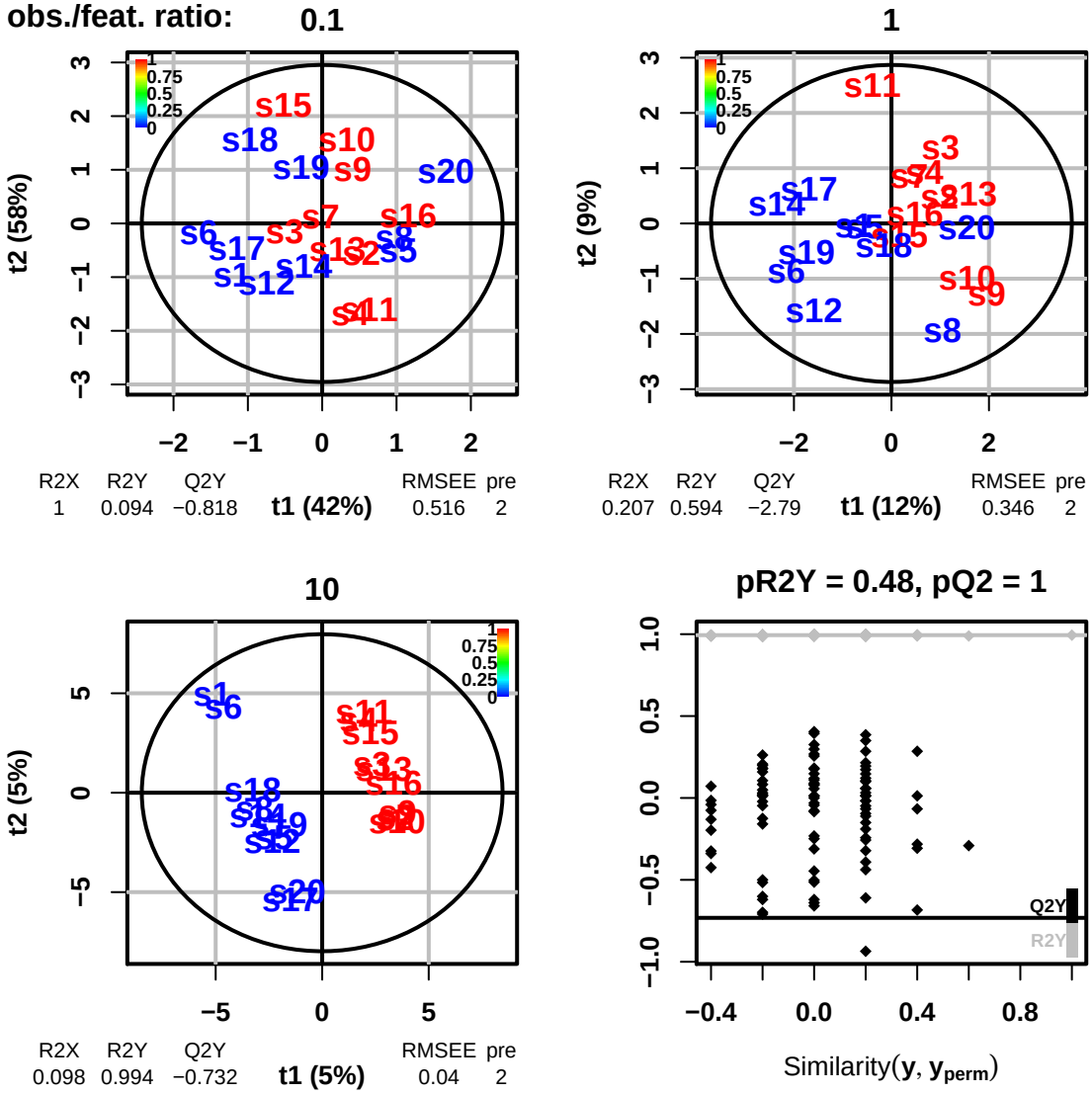


Figure 5: Risk of PLS overfitting. In the simulation above, a **random matrix X** of 20 observations x 200 features was generated by sampling from the uniform distribution $U(0,1)$. A **random y** response was obtained by sampling (without replacement) from a vector of 10 zeros and 10 ones. **Top left**, **top right**, and **bottom left**: the X-**score plots** of the PLS modeling of **y** by the (sub)matrix of **X** restricted to the first 2, 20, or 200 features, are displayed (i.e., the observation/feature ratios are 0.1, 1, and 10, respectively). Despite the good separation obtained on the bottom left score plot, we see that the **Q2Y** estimation of predictive performance is low (negative); **Bottom right**: a significant proportion of the models (in fact here all models) trained after random permutations of the labels have a higher **Q2Y** value than the model trained with the true labels, confirming that PLS cannot specifically model the **y** response with the **X** predictors, as expected.

This simple simulation illustrates that PLS overfit can occur, in particular when the number of features exceeds the number of observations. **It is therefore essential to check that the $Q2Y$ value of the model is significant by random permutation of the labels**.

4.5.2 VIP from OPLS models

The classical **VIP** metric is not useful for OPLS modeling of a single response since (Galindo-Prieto, Eriksson, and Trygg 2014, Thevenot et al. (2015)): 1. **VIP** values remain identical whatever the number of orthogonal components selected, 2. **VIP** values are univariate (i.e., they do not provide information about interactions between variables). In fact, when

features are standardized, we can demonstrate a mathematical relationship between VIP and *p*-values from a Pearson correlation test (Thevenot et al. 2015), as illustrated by the figure below:

```
## OPLS
## 183 samples x 109 variables and 1 response
## standard scaling of predictors and response(s)
##       R2X(cum) R2Y(cum) Q2(cum) RMSEE pre ort pR2Y  pQ2
## Total    0.212    0.476    0.31  7.53   1   1 0.05 0.05
```

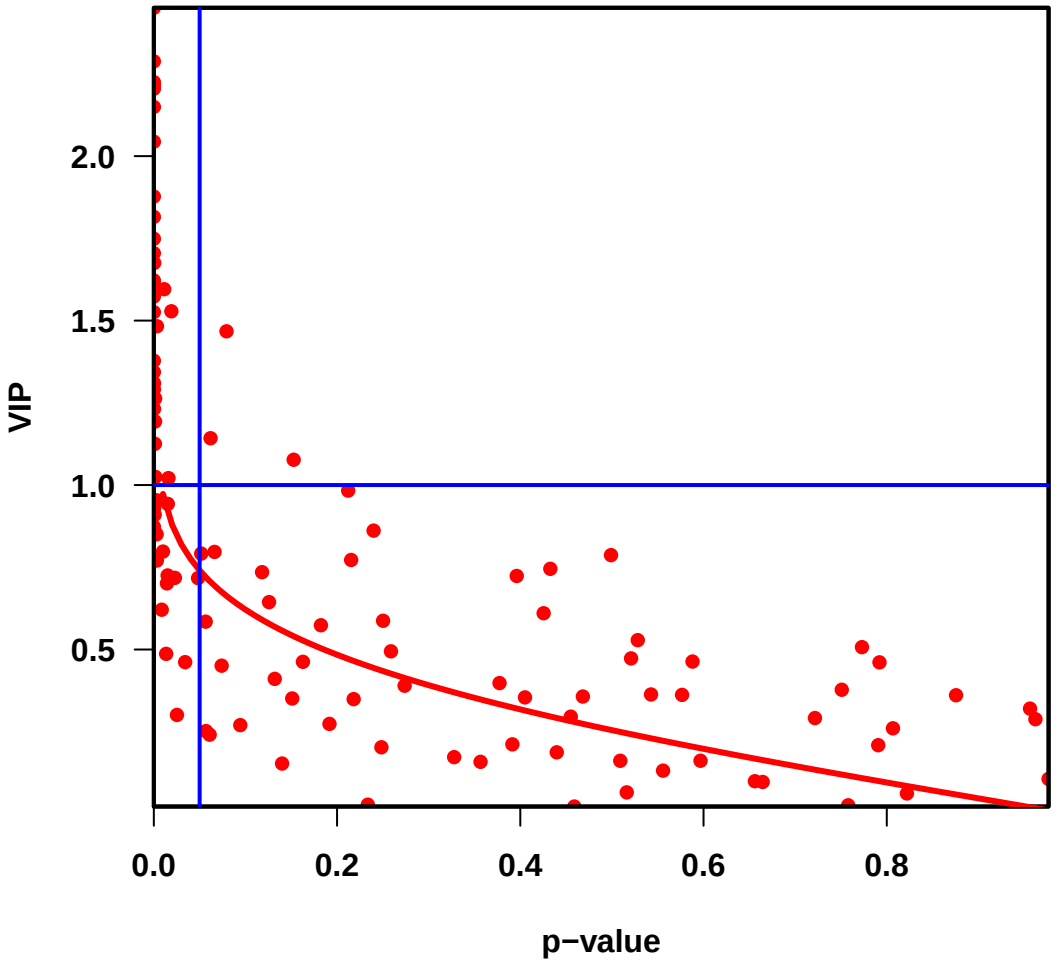


Figure 6: Relationship between VIP from one-predictive PLS or OPLS models with standardized variables, and p-values from Pearson correlation test. The (p_j, VIP_j) pairs corresponding respectively to the VIP values from OPLS modelling of the *age* response with the *sacurine* dataset, and the *p*-values from the Pearson correlation test are shown as red dots. The $y = \Phi^{-1}(1 - x/2)/z_{rms}$ curve is shown in red (where Φ^{-1} is the inverse of the probability density function of the standard normal distribution, and z_{rms} is the quadratic mean of the z_j quantiles from the standard normal distribution; $z_{rms} = 2.6$ for the *sacurine* dataset and the *age* response). The vertical (resp. horizontal) blue line corresponds to univariate (resp. multivariate) thresholds of $p = 0.05$ and $VIP = 1$, respectively (Thevenot et al. 2015).

The **VIP** properties above result from:

1. OPLS models of a single response have a single predictive component,

2. in the case of one-predictive component (O)PLS models, the general formula for VIPs can be simplified to $VIP_j = \sqrt{m} \times |w_j|$ for each feature j, were m is the total number of features and **w** is the vector of loading weights,
3. in OPLS, **w** remains identical whatever the number of extracted orthogonal components,
4. for a single-response model, **w** is proportional to **X'y** (where ' denotes the matrix transposition),
5. if **X** and **y** are standardized, **X'y** is the vector of the correlations between the features and the response.

Galindo-Prieto, Eriksson, and Trygg (2014) have recently suggested new VIP metrics for OPLS, **VIP_pred** and **VIP_ortho**, to separately measure the influence of the features in the modeling of the dispersion correlated to, and orthogonal to the response, respectively (Galindo-Prieto, Eriksson, and Trygg 2014).

For OPLS(-DA) models, you can therefore get from the model generated with `opls`:

1. the **predictive VIP vector** (which corresponds to the $VIP_{4,pred}$ metric measuring the variable importance in prediction) with `getVipVn(model)`,
2. the orthogonal VIP vector which is the $VIP_{4,ortho}$ metric measuring the variable importance in orthogonal modeling with `getVipVn(model, orthoL = TRUE)`. As for the classical **VIP**, we still have the mean of VIP^2_{pred} (and of VIP^2_{ortho}) which, each, equals 1.

4.5.3 (Orthogonal) Partial Least Squares Discriminant Analysis: (O)PLS-DA

4.5.3.1 Two classes When the **y** response is a factor of 2 levels (character vectors are also allowed), it is internally transformed into a vector of values $\in \{0, 1\}$ encoding the classes. The vector is centered and unit-variance scaled, and the (O)PLS analysis is performed.

Brereton and Lloyd (2014) have demonstrated that when the sizes of the 2 classes are **unbalanced**, a **bias** is introduced in the computation of the decision rule, which penalizes the class with the highest size (Brereton and Lloyd 2014). In this case, an external procedure using **resampling** (to balance the classes) and taking into account the class sizes should be used for optimal results.

4.5.3.2 Multiclass In the case of **more than 2 levels**, the **y** response is internally transformed into a matrix (each class is encoded by one column of values $\in \{0, 1\}$). The matrix is centered and unit-variance scaled, and the PLS analysis is performed.

In this so-called **PLS2** implementation, the proportions of 0 and 1 in the columns is usually unbalanced (even in the case of balanced size of the classes) and the bias described previously occurs (Brereton and Lloyd 2014). The multiclass PLS-DA results from ropls are therefore indicative only, and we recommend to set an external procedure where each column of the matrix is modeled separately (as described above) and the resulting probabilities are aggregated (see for instance M Bylesjo et al. (2006)).

4.6 Working on *ExpressionSet* omics objects from bioconductor

The **ExpressionSet** class from the Biobase bioconductor package has been developed to conveniently handle preprocessed omics objects, including the variables x samples matrix of intensities, and data frames containing the sample and variable metadata (Huber et al. 2015). The matrix and the two data frames can be accessed by the `exprs`, `pData` and `fData` respectively (note that the data matrix is stored in the object with samples in columns).

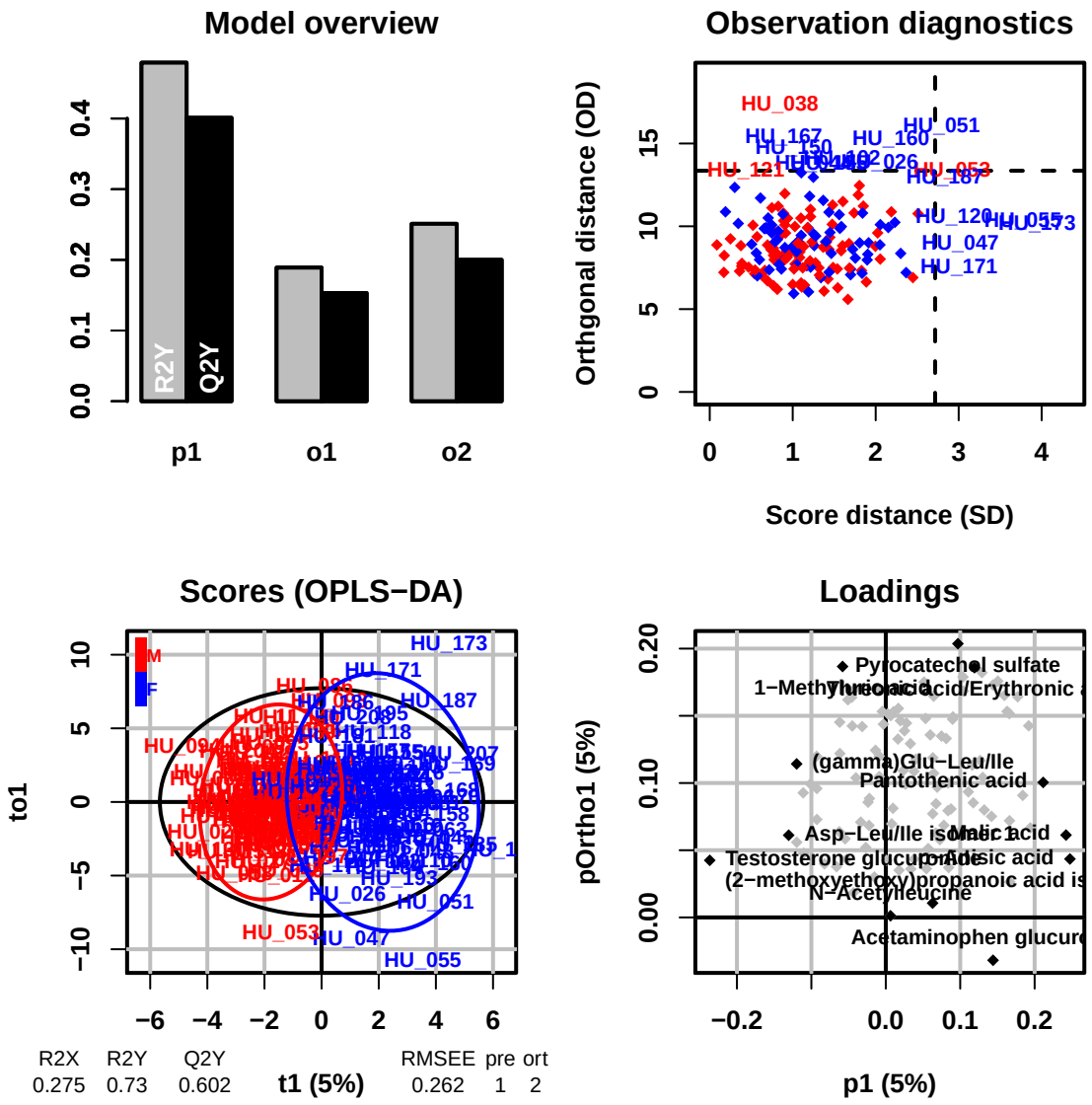
The `opls` method can be applied to an **ExpressionSet** object, by using the object as the `x` argument, and, for (O)PLS(-DA), by indicating as the `y` argument the name of the sampleMetadata to be used as the response.

In the example below, we will first build a minimal **ExpressionSet** object from the *sacurine* data set, and we subsequently perform an OPLS-DA.

```
library(Biobase)
sacSet <- ExpressionSet(assayData = t(dataMatrix),
```

```
phenoData = new("AnnotatedDataFrame", data = sampleMetadata))
opls(sacSet, "gender", orthoI = NA)
```

```
## OPLS-DA
## 183 samples x 109 variables and 1 response
## standard scaling of predictors and response(s)
##       R2X(cum) R2Y(cum) Q2(cum) RMSEE pre ort pR2Y  pQ2
## Total    0.275     0.73   0.602 0.262   1   2 0.05 0.05
```



4.7 Importing/exporting data from/to the Workflow4metabolomics infrastructure

Galaxy is a web-based environment providing powerful graphical user interface and workflow management functionalities for omics data analysis (Goecks et al. (2010); Boekel et al. (2015)). Wrapping an R code into a Galaxy module is quite straight-forward: examples can be found on the toolshed central repository and in the RGalaxy bioconductor package.

Workflow4metabolomics (W4M) is the online infrastructure for computational metabolomics based on the Galaxy environment (Giacomoni et al. 2015). W4M enables to build, run, save and share workflows efficiently. In addition,

workflows and input/output data (called **histories**) can be referenced, thus enabling fully reproducible research. More than 30 modules are currently available for LC-MS, GC-MS and NMR data preprocessing, statistical analysis, and annotation, including wrappers of `xcms`, `CAMERA`, `metaMS`, `ropls`, and `biosigner`, and is open to new contributions.

In order to facilitate data import from/to W4M, the `fromW4M` function (respectively the `toW4M` method) enables import from (respectively export to) the **W4M 3 tabular file format** (**dataMatrix.tsv**, **sampleMetadata.tsv**, **variableMetadata.tsv**) into (respectively from) an `ExpressionSet` object, as shown in the following example which uses the 3 .tsv files stored in the **extdata** repository of the package to create a *sacSet* ExpressionSet object:

```
sacSet <- fromW4M(file.path(path.package("ropls"), "extdata"))
sacSet
## ExpressionSet (storageMode: lockedEnvironment)
## assayData: 109 features, 183 samples
##   element names: exprs
## protocolData: none
## phenoData
##   sampleNames: HU_011 HU_014 ... HU_209 (183 total)
##   varLabels: age bmi gender
##   varMetadata: labelDescription
## featureData
##   featureNames: X.2.methoxyethoxy.propanoic.acid.isomer
##     X.gamma.Glu.Leu.Ile ... Xanthosine (109 total)
##   fvarLabels: msiLevel hmdb chemicalClass
##   fvarMetadata: labelDescription
## experimentData: use 'experimentData(object)'
## Annotation:
```

The generated *sacSet* `ExpressionSet` object can be used with the `opls` method as described in the previous section.

Conversely, an `ExpressionSet` (with filled **assayData**, **phenoData** and **featureData** slots) can be exported to the 3 table W4M format:

```
toW4M(sacSet, paste0(getwd(), "/out_"))
```

Before moving to the next session whith another example dataset, we detach *sacurine* from the search path:

```
detach(sacurine)
```

5 Pre-processing and annotation of mass spectrometry data

To illustrate how *dataMatrix*, *sampleMetadata* and *variableMetadata* can be obtained from raw mass spectra file, we use the LC-MS data from the faahKO package (Saghatelian et al. 2004). We will pre-process the raw files with the xcms package (Smith et al. 2006) and annotate isotopes and adducts with the CAMERA package (Kuhl et al. 2012), as described in the corresponding vignettes (all these packages are from **bioconductor**).

Let us start by getting the paths to the 12 raw files (6 KO and 6 WT mice) in the **.cdf** open format. The files are grouped in two sub-directories (**KO** and **WT**) since xcms can use sample class information when grouping the peaks and correcting retention times.

```
library(faahKO)
cdfpath <- system.file("cdf", package = "faahKO")
cdffiles <- list.files(cdfpath, recursive = TRUE, full.names = TRUE)
basename(cdffiles)
##  [1] "ko15.CDF" "ko16.CDF" "ko18.CDF" "ko19.CDF" "ko21.CDF" "ko22.CDF"
##  [7] "wt15.CDF" "wt16.CDF" "wt18.CDF" "wt19.CDF" "wt21.CDF" "wt22.CDF"
```

Next, xcms is used to pre-process the individual raw files, as described in the vignette.

```
library(xcms)
```

```
xset <- xcmsSet(cdffiles)
```

```
xset
## An "xcmsSet" object with 12 samples
##
## Time range: 2506.1-4147.7 seconds (41.8-69.1 minutes)
## Mass range: 200.1-599.3338 m/z
## Peaks: 4721 (about 393 per sample)
## Peak Groups: 0
## Sample classes: KO, WT
##
## Feature detection:
##  o Peak picking performed on MS1.
## Profile settings: method = bin
##                   step = 0.1
##
## Memory usage: 0.741 MB
xset <- group(xset)
## Processing 3195 mz slices ...
## OK
```

```
xset2 <- retcor(xset, family = "symmetric", plottype = "mdevden")
```

```
## Performing retention time correction using 133 peak groups.
```

```
xset2 <- group(xset2, bw = 10)
## Processing 3195 mz slices ...
## OK
```

```
xset3 <- fillPeaks(xset2)
```

Finally, the `annotateDiffreport` from CAMERA annotates isotopes and adducts and builds a peak table containing the peak intensities and the variable metadata.

```
library(CAMERA)
diffreport <- annotateDiffreport(xset3, quick=TRUE)
## Start grouping after retention time.
## Created 128 pseudospectra.
## Generating peak matrix!
## Run isotope peak annotation
##  % finished: 10  20  30  40  50  60  70  80  90  100
## Found isotopes: 81
diffreport[1:4, ]
##                 name      fold     tstat       pvalue    mzmed    mzmin
## 300.2/3390 M300T3390  5.693594 -14.44368 5.026336e-08 300.1898 300.1706
## 301.2/3390 M301T3390  5.876588 -15.57570 6.705719e-08 301.1879 301.1659
## 298.2/3187 M298T3187  3.870918 -11.93891 3.310025e-07 298.1508 298.1054
## 491.2/3397 M491T3397 24.975703 -16.83986 4.463361e-06 491.2000 491.1877
##               mzmax    rtmed    rtmin    rtmax npeaks KO WT      ko15
## 300.2/3390 300.2000 3390.324 3386.765 3396.335     12  6  6 4534353.6
## 301.2/3390 301.1949 3389.627 3386.765 3392.101      7  6  1  962353.4
## 298.2/3187 298.1592 3186.803 3184.124 3191.312      4  4  0  180780.8
## 491.2/3397 491.2063 3397.160 3367.123 3424.681      6  6  0  432037.0
##                 ko16      ko18      ko19      ko21      ko22       wt15
```

```
## 300.2/3390 4980914.5 5290739.1 4564262.9 4733236.1 3931592.6 349660.885
## 301.2/3390 1047934.1 1109303.0  946943.4  984787.2  806171.5  86450.412
## 298.2/3187  203927.0  191015.9  190626.8  156869.1  220288.6  16269.096
## 491.2/3397  332159.1  386966.8  334951.5  294816.2  373577.6   7643.138
##                 wt16      wt18      wt19        wt21       wt22  isotopes
## 300.2/3390 491793.18 645526.70 634108.85 1438254.446 1364627.84   [9][M]+
## 301.2/3390 120096.52 143007.95 137319.69  218483.143  291392.97 [9][M+1]+
## 298.2/3187  43677.78  54739.13  76318.01   54726.115   49679.94
## 491.2/3397  10519.94  26472.29  33598.32    8030.467       0.00
##            adduct pcgroup
## 300.2/3390             20
## 301.2/3390             20
## 298.2/3187            102
## 491.2/3397             28
```

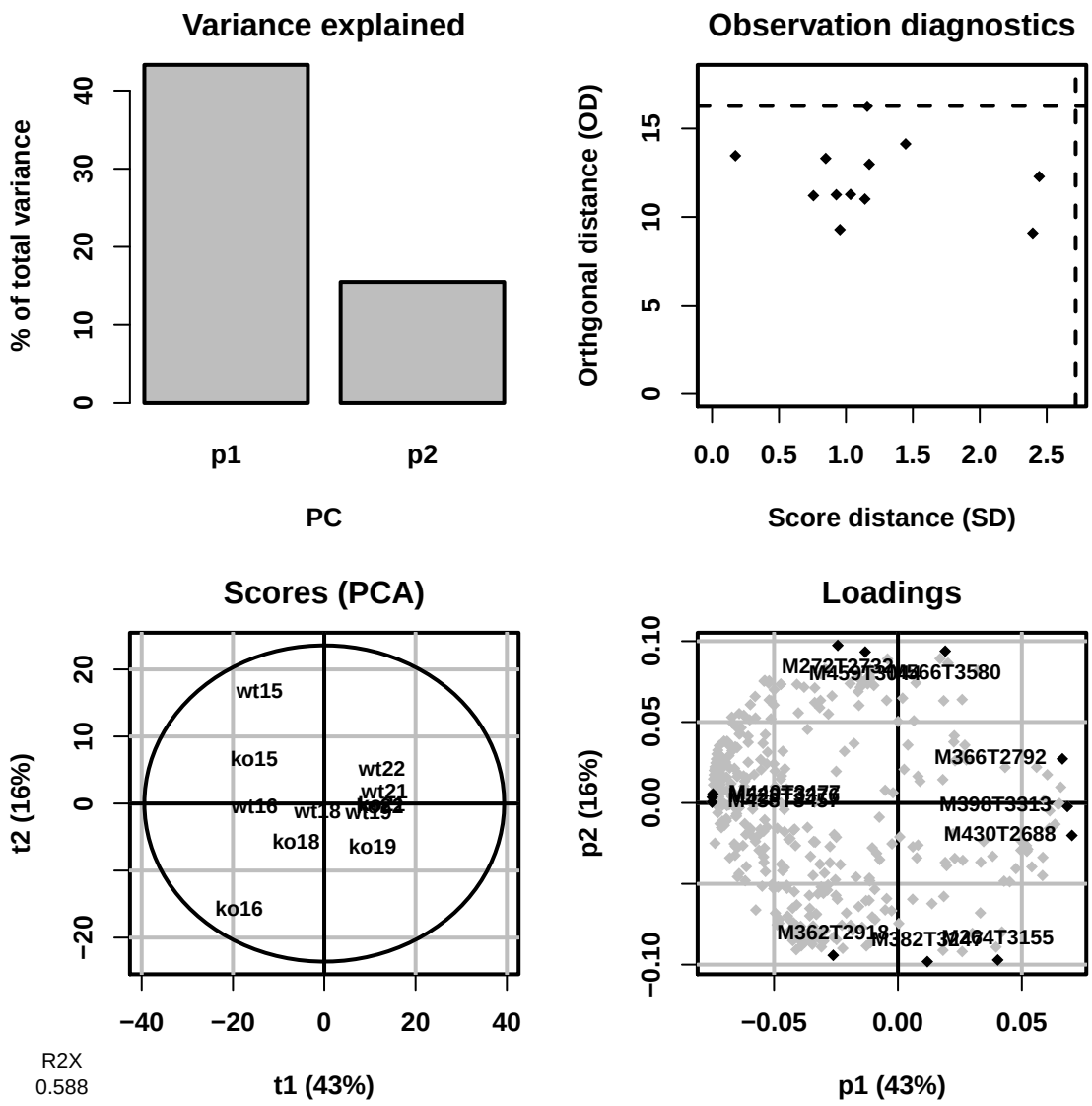
We then build the dataMatrix, sampleMetadata and variableMetadata matrix and dataframes as follows:

```
sampleVc <- grep("^ko|^wt", colnames(diffreport), value = TRUE)
dataMatrix <- t(as.matrix(diffreport[, sampleVc]))
dimnames(dataMatrix) <- list(sampleVc, diffreport[, "name"])
sampleMetadata <- data.frame(row.names = sampleVc,
genotypeFc = substr(sampleVc, 1, 2))
variableMetadata <- diffreport[, !(colnames(diffreport) %in% c("name", sampleVc))]
rownames(variableMetadata) <- diffreport[, "name"]
```

The data can now be analysed with the ropls package as described in the previous section (i.e. by performing a PCA and an OPLS-DA):

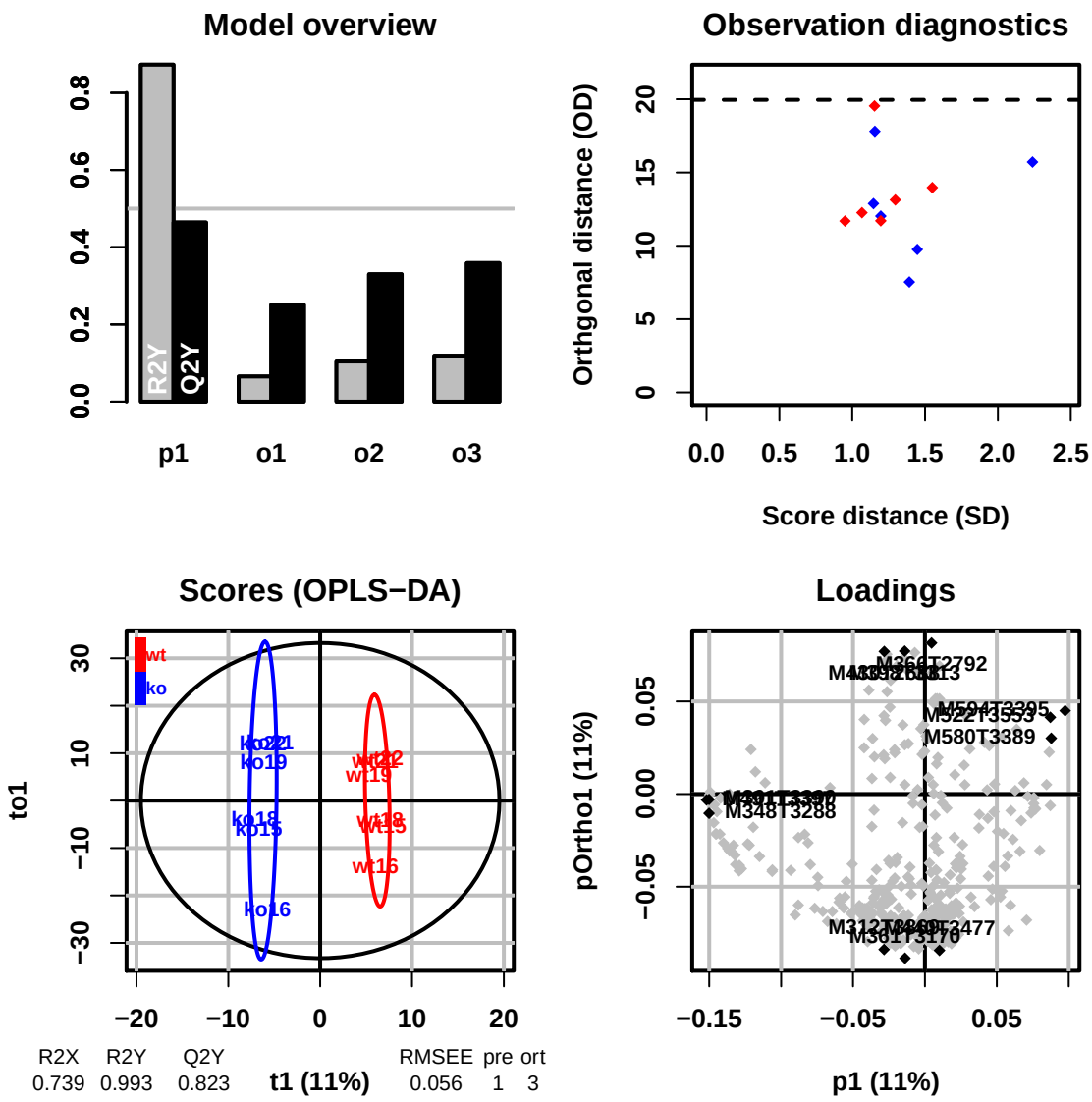
```
library(ropls)
opls(dataMatrix)
```

```
## PCA
## 12 samples x 400 variables
## standard scaling of predictors
##       R2X(cum) pre ort
## Total    0.588   2   0
```




```
opls(dataMatrix, sampleMetadata[, "genotypeFc"], orthoI = NA)
```

```
## Warning: OPLS: number of predictive components ('predI' argument) set to 1
## OPLS-DA
## 12 samples x 400 variables and 1 response
## standard scaling of predictors and response(s)
##       R2X(cum) R2Y(cum) Q2(cum)  RMSEE pre ort pR2Y  pQ2
## Total    0.739    0.993   0.823 0.0555   1   3 0.05 0.05
```



Note that the warning message is just a reminder that OPLS(-DA) models of a single response have only 1 predictive component, and could have been avoided by specifying `predI = 1` in the `opls` call.

6 Other datasets

In addition to the *sacurine* dataset presented above, the package contains the following datasets to illustrate the functionalities of PCA, PLS and OPLS (see the examples in the documentation of the opls function):

- **aminoacids** Amino-Acids Dataset. Quantitative structure property relationship (QSPR) (Wold, Sjostrom, and Eriksson 2001).
- **cellulose** NIR-Viscosity example data set to illustrate multivariate calibration using PLS, spectral filtering and OPLS (Multivariate calibration using spectral data. Simca tutorial. Umetrics, Sweden).
- **cornell** Octane of various blends of gasoline: Twelve mixture component proportions of the blend are analysed (Tenenhaus 1998).
- **foods** Food consumption patterns accross European countries (FOODS). The relative consumption of 20 food items was compiled for 16 countries. The values range between 0 and 100 percent and a high value corresponds to

a high consumption. The dataset contains 3 missing data (Eriksson et al. 2001).
- **linnerud** Three physiological and three exercise variables are measured on twenty middle-aged men in a fitness club (Tenenhaus 1998).
- **lowarp** A multi response optimization data set (LOWARP) (Eriksson et al. 2001).
- **mark** Marks obtained by french students in mathematics, physics, french and english. Toy example to illustrate the potentialities of PCA (Baccini 2010).

7 Session info

Here is the output of `sessionInfo()` on the system on which this document was compiled:

```
## R version 3.4.0 (2017-04-21)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 16.04.2 LTS
##
## Matrix products: default
## BLAS: /home/biocbuild/bbs-3.5-bioc/R/lib/libRblas.so
## LAPACK: /home/biocbuild/bbs-3.5-bioc/R/lib/libRlapack.so
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=en_US.UTF-8        LC_COLLATE=C
##  [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] parallel  stats     graphics  grDevices utils     datasets  methods
## [8] base
##
## other attached packages:
##  [1] CAMERA_1.32.0       faahKO_1.15.0       xcms_1.52.0
##  [4] MSnbase_2.2.0       ProtGenerics_1.8.0  mzR_2.10.0
##  [7] Rcpp_0.12.10        BiocParallel_1.10.0 Biobase_2.36.0
## [10] BiocGenerics_0.22.0 ropls_1.8.0         BiocStyle_2.4.0
##
## loaded via a namespace (and not attached):
##  [1] splines_3.4.0          lattice_0.20-35        colorspace_1.3-2
##  [4] htmltools_0.3.5        stats4_3.4.0           base64enc_0.1-3
##  [7] yaml_2.1.14            vsn_3.44.0             XML_3.98-1.6
## [10] survival_2.41-3        RBGL_1.52.0            foreign_0.8-67
## [13] affy_1.54.0            RColorBrewer_1.1-2     affyio_1.46.0
## [16] foreach_1.4.3          plyr_1.8.4             mzID_1.14.0
## [19] stringr_1.2.0          zlibbioc_1.22.0        munsell_0.4.3
## [22] pcaMethods_1.68.0      gtable_0.2.0           htmlwidgets_0.8
## [25] codetools_0.2-15       evaluate_0.10          latticeExtra_0.6-28
## [28] knitr_1.15.1           IRanges_2.10.0         doParallel_1.0.10
## [31] BiocInstaller_1.26.0   MassSpecWavelet_1.42.0 htmlTable_1.9
## [34] preprocessCore_1.38.0  acepack_1.4.1          checkmate_1.8.2
## [37] scales_0.4.1           backports_1.0.5        limma_3.32.0
## [40] S4Vectors_0.14.0       Hmisc_4.0-2            graph_1.54.0
## [43] gridExtra_2.2.1        RANN_2.5               impute_1.50.0
```

```
## [46] ggplot2_2.2.1          digest_0.6.12          stringi_1.1.5
## [49] grid_3.4.0             rprojroot_1.2          tools_3.4.0
## [52] magrittr_1.5           lazyeval_0.2.0         tibble_1.3.0
## [55] cluster_2.0.6          Formula_1.2-1          MASS_7.3-47
## [58] Matrix_1.2-9           data.table_1.10.4      rmarkdown_1.4
## [61] iterators_1.0.8        rpart_4.1-11           MALDIquant_1.16.2
## [64] igraph_1.0.1           nnet_7.3-12            multtest_2.32.0
## [67] compiler_3.4.0
```

References

Baccini, A. 2010. "Statistique Descriptive Multidimensionnelle (Pour Les Nuls)."

Boekel, J., JM. Chilton, IR. Cooke, PL. Horvatovich, PD. Jagtap, L. Kall, J. Lehtio, P. Lukasse, PD. Moerland, and TJ. Griffin. 2015. "Multi-Omic Data Analysis Using Galaxy." *Nature Biotechnology* 33 (2): 137–39. doi:10.1038/nbt.3134.

Brereton, Richard G., and Gavin R. Lloyd. 2014. "Partial Least Squares Discriminant Analysis: Taking the Magic Away." *Journal of Chemometrics* 28 (4): 213–25. http://dx.doi.org/10.1002/cem.2609.

Bylesjo, M, M Rantalainen, O Cloarec, J Nicholson, E Holmes, and J Trygg. 2006. "OPLS Discriminant Analysis: Combining the Strengths of PLS-DA and SIMCA Classification." *Journal of Chemometrics* 20: 341–51. http://dx.doi.org/10.1002/cem.1006.

Bylesjo, M., M. Rantalainen, J. Nicholson, E. Holmes, and J. Trygg. 2008. "K-OPLS Package: Kernel-Based Orthogonal Projections to Latent Structures for Prediction and Interpretation in Feature Space." *BMC Bioinformatics* 9 (1): 106. http://dx.doi.org/10.1186/1471-2105-9-106.

Eriksson, L., E. Johansson, N. Kettaneh-Wold, and S. Wold. 2001. *Multi- and Megavariate Data Analysis. Principles and Applications*. Umetrics Academy.

Galindo-Prieto, B., L. Eriksson, and J. Trygg. 2014. "Variable Influence on Projection (VIP) for Orthogonal Projections to Latent Structures (OPLS)." *Journal of Chemometrics* 28 (8): 623–32. http://dx.doi.org/10.1002/cem.2627.

Gaude, R., F. Chignola, D. Spiliotopoulos, A. Spitaleri, M. Ghitti, JM. Garcia-Manteiga, S. Mari, and G. Musco. 2013. "Muma, an R Package for Metabolomics Univariate and Multivariate Statistical Analysis." *Current Metabolomics* 1: 180–89. http://dx.doi.org/10.2174/2213235X11301020005.

Giacomoni, F., G. Le Corguille, M. Monsoor, M. Landi, P. Pericard, M. Petera, C. Duperier, et al. 2015. "Workflow4Metabolomics: A Collaborative Research Infrastructure for Computational Metabolomics." *Bioinformatics* 31 (9): 1493–5. http://dx.doi.org/10.1093/bioinformatics/btu813.

Goecks, J., A. Nekrutenko, J. Taylor, and The Galaxy Team. 2010. "Galaxy: A Comprehensive Approach for Supporting Accessible, Reproducible, and Transparent Computational Research in the Life Sciences." *Genome Biology* 11 (8): R86. doi:10.1186/gb-2010-11-8-r86.

Huber, W., VJ. Carey, R. Gentleman, S. Anders, M. Carlson, BS. Carvalho, HC. Bravo, et al. 2015. "Orchestrating High-Throughput Genomic Analysis with Bioconductor." *Nature Methods* 12 (2): 115–21. doi:10.1038/nmeth.3252.

Hubert, M., PJ. Rousseeuw, and K. Vanden Branden. 2005. "ROBPCA: A New Approach to Robust Principal Component Analysis." *Technometrics* 47: 64–79. http://dx.doi.org/10.1198/004017004000000563.

Kuhl, C., R. Tautenhahn, C Bottcher, TR. Larson, and S. Neumann. 2012. "CAMERA: An Integrated Strategy for Compound Spectra Extraction and Annotation of Liquid Chromatography/Mass Spectrometry Data Sets." *Analytical Chemistry* 84 (1): 283–89. http://dx.doi.org/10.1021/ac202450g.

Mehmood, T., KH. Liland, L. Snipen, and S. Saebo. 2012. "A Review of Variable Selection Methods in Partial Least Squares Regression." *Chemometrics and Intelligent Laboratory Systems* 118 (0): 62–69. http://dx.doi.org/10.1016/j.

chemolab.2012.07.010.

Pinto, RC., J. Trygg, and J. Gottfries. 2012. "Advantages of Orthogonal Inspection in Chemometrics." *Journal of Chemometrics* 26 (6): 231–35. http://dx.doi.org/10.1002/cem.2441.

R Development Core Team. 2008. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. http://www.R-project.org.

Saghatelian, A., SA. Trauger, EJ. Want, EG. Hawkins, G. Siuzdak, and BF. Cravatt. 2004. "Assignment of Endogenous Substrates to Enzymes by Global Metabolite Profiling." *Biochemistry* 43 (45): 14332–9. http://dx.doi.org/10.1021/bi0480335.

Smith, CA., EJ. Want, G. O'Maille, R. Abagyan, and G. Siuzdak. 2006. "XCMS: Processing Mass Spectrometry Data for Metabolite Profiling Using Nonlinear Peak Alignment, Matching, and Identification." *Analytical Chemistry* 78 (3): 779–87. http://dx.doi.org/10.1021/ac051437y.

Szymanska, E., E. Saccenti, AK. Smilde, and JA. Westerhuis. 2012. "Double-Check: Validation of Diagnostic Statistics for PLS-DA Models in Metabolomics Studies." *Metabolomics* 8 (1, 1): 3–16. http://dx.doi.org/10.1007/s11306-011-0330-3.

Tenenhaus, M. 1998. *La Regression PLS : Theorie et Pratique*. Editions Technip.

Thevenot, EA., A. Roux, X. Ying, E. Ezan, and C. Junot. 2015. "Analysis of the Human Adult Urinary Metabolome Variations with Age, Body Mass Index and Gender by Implementing a Comprehensive Workflow for Univariate and OPLS Statistical Analyses." *Journal of Proteome Research* 14 (8): 3322–35. http://dx.doi.org/10.1021/acs.jproteome.5b00354.

Trygg, J., and S. Wold. 2002. "Orthogonal Projection to Latent Structures (O-PLS)." *Journal of Chemometrics* 16: 119–28. http://dx.doi.org/10.1002/cem.695.

Wehrens, R. 2011. *Chemometrics with R: Multivariate Data Analysis in the Natural Sciences and Life Sciences*. Springer.

Wold, S., M. Sjostrom, and L. Eriksson. 2001. "PLS-Regression: A Basic Tool of Chemometrics." *Chemometrics and Intelligent Laboratory Systems* 58: 109–30. http://dx.doi.org/10.1016/S0169-7439(01)00155-1.