lmdme: Linear Model on Designed Multivariate Experiments in R

Cristobal Fresno[1,2] and Elmer A. Fernández[1,2]

[1]Bio-science Data Mining Group, Universidad Católica de Córdoba
[2]CONICET, Argentina

April 30, 2013

Abstract

The lmdme package implements analysis of variance (ANOVA) decomposition through linear models on designed multivariate experiments in R (R Development Core Team, 2012), allowing ANOVA-principal component analysis (APCA) and ANOVA-simultaneous component analysis (ASCA). It also extends both methods with the application of partial least squares (PLS) through the specification of a desired output matrix. The package is freely available on the Bioconductor website (Gentleman et al., 2004), licenced under GNU general public license.

ANOVA decomposition methods for multivariate designed experiments are becoming popular in "omics" experiments (transcriptomics, metabolomics, etc.) where measurements are performed according to a predefined experimental design (Smilde et al., 2005), with several experimental factors or including subject specific clinical covariates, such as those present in current clinical genomic studies. ANOVA-PCA and ASCA are well-suited methods to study interaction patterns on multidimensional datasets. However, current R implementation of APCA is only available for *Spectra* data (ChemoSpec), meanwhile ASCA (Nueda et al., 2007) is based on average calculations over the indexes of up to three design matrices. Thus, no statistical inference over estimated effects is provided. Moreover, ASCA is not available in R package format.

Here, R implementation on ANOVA decomposition with PCA/PLS analysis is provided. It allows a flexible `formula` interface the specification on almost any linear model with appropriate inference over the estimated effects and display functions for both PCA and PLS.

We will present the model, implementation and a high-throughput *microarray* example one applied on interaction pattern analysis.

1 Introduction

Current "omics" experiments (proteomics, transcriptomics, metabolomics or genomics) are multivariate by nature. Modern technology allows the exploration of the whole genome or a big subset of the proteome, where each gene/protein is in essence a variable explored to elucidate its relationship with some outcome.

In addition, these experiments are including more and more experimental factors (time, dose, etc.) from design or subject specific information such as age, gender, linage and so on, and are available for analysis. Hence, in order to discover or evaluate experimental design or subject specific patterns, some multivariate approaches should be applied. In this context, principal component analysis (PCA) and partial least squares regression (PLS) are the most common. However, it is known that working with raw data could mask information of interest. Therefore, analysis of variance (ANOVA) based decomposition, is becoming popular in order to split variability sources, before the application of such multivariate approaches. Seminal works on genomics were De Haan et al. (2007) on ANOVA-PCA (APCA) and Smilde et al. (2005) on ANOVA-SCA (ASCA) models. However, as far as the authors know, R implementation of APCA is only available for *Spectra* data, ChemoSpec R package by Hanson (2012). Unfortunately, there is no R package for ASCA but, it is only accessible by uploading script-function files resulting from a MATLAB code translation (Nueda et al., 2007). As in the former, it only accepts up to three design matrices, limiting and making its use difficult. Moreover, coefficient estimations are based on average calculations using binary design matrices, without any statistical inference over them.
Here, a flexible linear model based decomposition framework is provided. Almost any model can be specified, according to the experimental design, by means of a flexible `formula` interface. Since the estimation is carried out by means of *maximum likelihood*, statistical significance on coefficient estimates is naturally given. It also provides both PCA or PLS analysis capabilities over appropriate ANOVA decomposition results, as well as graphical representations. The implementation is well-suited to directly analyze gene expression matrices (variables on rows) from high-throughput data such as *microarray* or *RNA-seq* experiments. One example will introduce the user to the package usage, through the exploration of interaction patterns on a microarray experiment.

2 The model

A detailed explanation of ANOVA decomposition and multivariate analysis can be found in Smilde et al. (2005) and Zwanenburg et al. (2011). Briefly and without the loss of generality, let's assume a *microarray* experiment where the expression of $(G_1, G_2, \ldots, G_g)$ genes are arrayed in a chip. In this context, let's consider an experimental design with two main factors: A with a levels $(A_1, A_2, \ldots, A_i, \ldots, A_a)$ and B with b levels $(B_1, B_2, \ldots, B_j, \ldots, B_b)$, with replicates $R_1, R_2, \ldots, R_k, \ldots, R_r$ for each $A \times B$ combination levels. After preprocessing steps as described elsewhere (Smyth, 2004), each chip is represented by a column vector of gene expression measurements of $g \times 1$. Then, the whole experimental data is arranged into a $g \times n$ expression matrix (X) where $n = a \times b \times r$. In this context, single gene measurements across the different treatment combinations $(A_i \times B_j)$ are presented in a row on the X matrix, as depicted in Figure 1. An equivalent X matrix structure needs to be obtained for *2D-DIGE* or *RNA-seq* experiments and so forth.

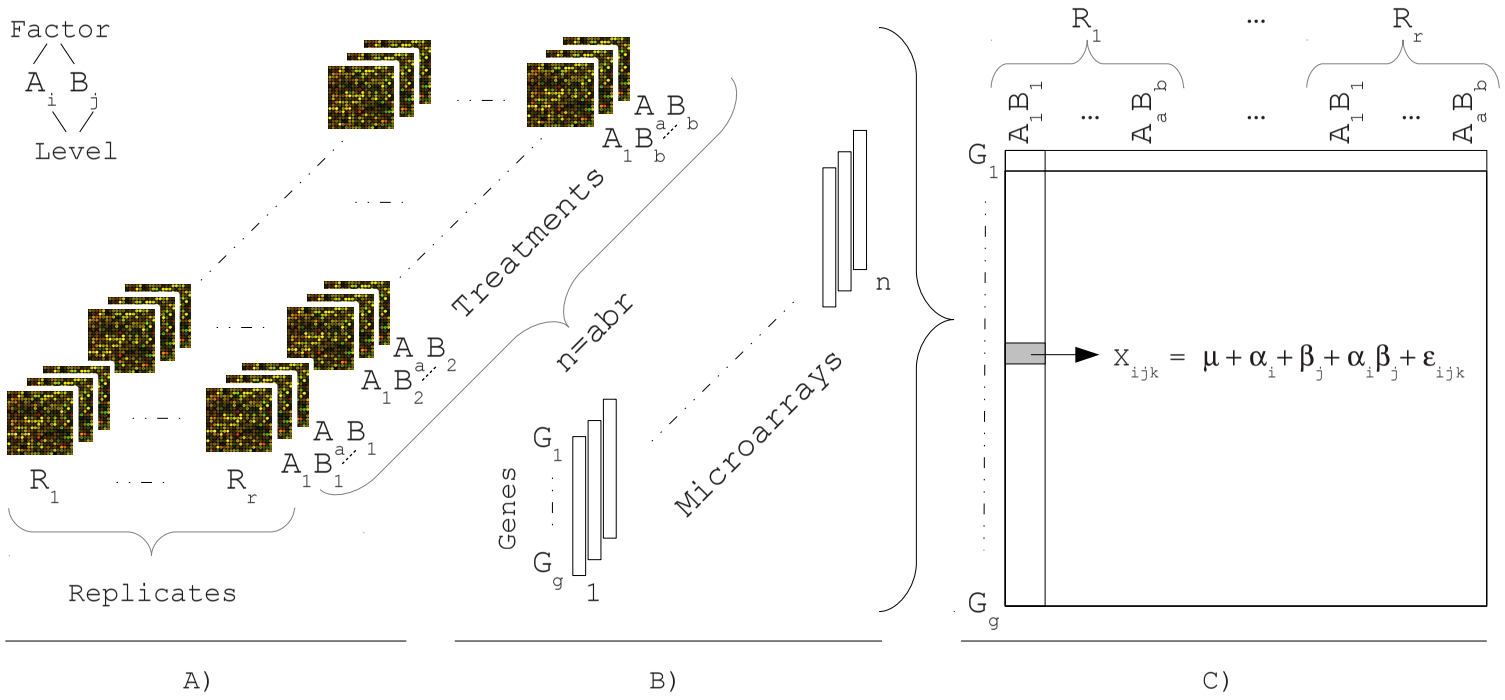


Figure 1: Microarray gene expression data representation. A) Genes are spotted over the chip. Then expression levels for each combination of treatment factor levels A_iB_j and their replicates R_k can be measured over the chips, yielding a total of $n = a \times b \times r$ microarrays. B) Gene expression of each chip (microarray) is then interpreted as a column vector of expression levels. C) Then, these column vectors will be bound producing the experiment gene expression matrix X. Expression measurements under all treatment combinations for a gene are represented through the X matrix rows. In this way, measurements on a row are subjected to the ANOVA model of Equation (1).

Regardless of data generation, the ANOVA model for each gene (row) in X can be written into the Equation (1):

$$x_{ijk} = \mu + \alpha_i + \beta_j + \alpha_i \times \beta_j + \varepsilon_{ijk} \tag{1}$$

where x_{ijk} is the measured expression for "some" gene, at combination "ij" of factors A and B for the k replicate; μ is the overall mean; α, β and $\alpha \times \beta$ are the main and interaction effects respectively; and the error term $\varepsilon_{ijk} \sim N(0, \sigma^2)$. Equation (1) can also be expressed in matrix form for all genes (2):

$$X = X_\mu + X_\alpha + X_\beta + X_{\alpha\beta} + E = \sum_{l \in \{\mu,\alpha,\beta,\alpha\beta\}} X_l + E \tag{2}$$

where X_l, E matrices are of dimension $g \times n$ and contain the level means of the corresponding $l-th$ term and the random error respectively. However, in the context of linear models X_l can also be written as a linear combination of two matrix multiplications (3):

$$X = \sum_{l \in \{\mu,\alpha,\beta,\alpha\beta\}} X_l + E = \sum_{l \in \{\mu,\alpha,\beta,\alpha\beta\}} B_l Z_l^T + E = B_\mu Z_\mu^T + \ldots + B_{\alpha\beta} Z_{\alpha\beta}^T + E = \mu 1^\top + B_\alpha Z_\alpha^T + \ldots + B_{\alpha\beta} Z_{\alpha\beta}^T + E \tag{3}$$

where B_l and Z_l are referenced in the literature as *coefficient* and *model* matrices, of dimensions $g \times m_{(l)}$ and $n \times m_{(l)}$ respectively, where $m_{(l)}$ is the number of levels of factor l. The first term is usually called *intercept*, being $B_\mu = \mu$ and $Z_\mu = \mathbf{1}$ of dimension $g \times 1$ and $n \times 1$ respectively. In this example, all Z_l are binary matrices, identifying whether a measurement belongs ("1") or not ("0"), to the corresponding factor.
In Smilde et al. (2005) and Nueda et al. (2007) implementations, the estimation of the coefficient matrices is based on *average* calculations using the design matrix (up to three design matrices $Z_{\alpha,\beta,\alpha\beta}$) to identify the average samples. Theoretically, they fully decompose the original matrix as shown in Equation (1). On the contrary, in this package the model coefficients are estimated, iteratively, by the *maximum likelihood* approach, using the `lmFit` function provided by limma package (Smyth et al., 2011). Consequently, three desirable features are also incorporated:

1. *Flexible* `formula` *interface* to specify any potential model. The user only needs to provide: i) the gene expression `matrix` (X), ii) the experimental `data.frame` (`design`) with treatment structure on it, and iii) the model in a `formula` style, just like in an ordinary `lm` R function. Internal `model.matrix` call, will automatically build the appropriate Z matrices, overcoming the constraint on factorial design size, and tedious model matrix definitions.

2. *Hypothesis tests* on coefficient B_l matrices. A T test is automatically carried out for the $s-th$ gene model, to test whether the $o-th$ coefficient is equal to zero or not, i.e., $H_0 : b_{so} = 0$ vs $H_1 : b_{so} \neq 0$. In addition, an F test is performed to simultaneously see if all b_{so} are equal to zero or not.

3. *Empirical Bayes correction* can also be achieved through the `eBayes limma` function. It uses an empirical Bayes method to shrink the row/gene-wise sample variances towards a common value and, to augment the degrees of freedom for the individual variances (Smyth, 2004).

In contrast, De Haan et al. (2007) main and interaction effects are estimated by overall mean subtraction. Hence, genes need to be treated as an additional factor. Meanwhile, in Smilde et al. (2005) and Nueda et al. (2007) implementations, the estimations are obtained in a gene by gene basis as in Equation (1). Therefore, in a two-way factor experiment e.g. $time \times oxygen$, De Haan's model includes two additional double interactions and a triple interaction, due to treating genes as a factor, in contrast to Smilde's and Nueda's.

2.1 The decomposition algorithm

The ANOVA model (2) is decomposed iteratively using Equation (3), where in each step the $l-th$ coefficients $\hat{B}_l$, $\hat{E}_l$ matrices and $\hat{\sigma}_l^2$ are estimated. Then, the particular term contribution matrix $\hat{X}_l = \hat{B}_l Z_l^\top$ is subtracted from the preceding residuals to feed the next model, as depicted in Equation (4):

$$
\begin{aligned}
X &= X_\mu + X_\alpha + X_\beta + X_{\alpha\beta} + E = \sum_{l \in \{\mu,\alpha,\beta,\alpha\beta\}} X_l + E \\
step\ \mu: \quad X &= X_\mu + E_\mu \Rightarrow X = \hat{B}_\mu Z_\mu^\top + \hat{E}_\mu \Rightarrow \hat{E}_\mu = X - \hat{B}_\mu Z_\mu^\top \\
step\ \alpha: \quad E_\mu &= X_\alpha + E_\alpha \Rightarrow \hat{E}_\mu = \hat{B}_\alpha Z_\alpha^\top + \hat{E}_\alpha \Rightarrow \hat{E}_\alpha = \hat{E}_\mu - \hat{B}_\alpha Z_\alpha^\top \\
&\vdots \quad \vdots \\
step\ l: E_{l-1} &= X_l + E_l \Rightarrow \hat{E}_{l-1} = \hat{B}_l Z_l^\top + \hat{E}_l \Rightarrow \hat{E}_l = \hat{E}_{l-1} - \hat{B}_l Z_l^\top \quad (4) \\
&\vdots \quad \vdots \\
step\ \alpha\beta: \quad E_\beta &= X_{\alpha\beta} + E \Rightarrow \hat{E}_\beta = \hat{B}_{\alpha\beta} Z_{\alpha\beta}^\top + \hat{E} \Rightarrow \hat{E} = \hat{E}_\beta - \hat{B}_{\alpha\beta} Z_{\alpha\beta}^\top
\end{aligned}
$$

where the hat ("^") denotes estimated coefficients. In this implementation, the first step always estimates the *intercept* term, i.e. `formula=~1` in R style, with $\hat{B}_\mu = \hat{\mu}$ and $Z_\mu = 1$. The following models, will only include the $l-th$ factor without the intercept, i.e. `formula=~lth_term-1` where lth_term stands for α, β or $\alpha\beta$ in this example. This procedure is quite similar to the one proposed by Harrington et al. (2005).

2.2 PCA and PLS analysis

These methods are concerned with explaining the variance/covariance structure of a set of observations (e.g. genes) through a few linear combinations of variables (e.g. experimental conditions). Both methods can be applied on the $l-th$ ANOVA decomposed step of Equation (4) tackling different aspects:

- PCA concerns with the *variance* of a single matrix, usually following two main objectives: i) data reduction and ii) interpretation. In this context, depending on the matrix which it is applied to, two different methods arise. When it is applied on *coefficient* matrix, $\hat{B}_l$, it is known as ASCA (Smilde et al., 2005). When it is calculated on the *residual*, $\hat{E}_{l-1}$, the procedure

follows the idea of APCA; in which it is usually applied on, $X_l + E$, i.e, the mean factor matrix X_l, plus the error of the fully decomposed model E of Equation (1) as in De Haan et al. (2007).

- PLS not only generalizes but, also combines features from PCA and regression, to explore the *covariance* structure between input and some output matrices, Abdi and Williams (2010) and Shawe-Taylor and Cristianini (2004). It is particularly useful when one or several dependent variables (outputs - O) must be predicted from a large and potentially highly correlated set of independent variables (inputs). In our implementation, the input could be the *coefficient* matrix $\hat{B}_l$, or the *residual* $\hat{E}_{l-1}$ and the output matrix a diagonal `O=diag(nrow(`$\hat{B}_l$`))` or design matrix $O = Z_l$, when using the coefficient or residual respectively. In addition the user can specify their own output matrix, O, to verify some particular hypothesis. For instance, in functional genomics it could be the Gene Ontology class matrix as used in Gene Set Enrichment Analysis (Subramanian et al., 2005).

When working with the *coefficient* matrix, the user can directly use the reduced factor dimension of X, instead of worrying about the expected number of components (rank of the matrix), given the number of replicates per treatment level as suggested by Smilde et al. (2005). They are directly summarized in the $\hat{B}_l$ matrix. In addition, for both PCA/PLS, the lmdme package also offers different methods for visualization results e.g. `biplot`, `loadingplot` and `screeplot` or `leverage` calculation, in order to filter out rows/genes as in Tarazona et al. (2012).

3 Examples

In this section we will give an overview of lmdme package by Fresno and Fernández (2012a). The example goes through a gene expression interaction pattern analysis application, where we address: how to define the model, undertake ANOVA decomposition, perform PCA/PLS analysis and visualize the results. From here onwards, some outputs were suppressed for reasons of clarity and the examples were carried out with `options(digits=4)`.

3.1 Package overview

The original data files for the first example are available at Gene Expression Omnibus, (Edgar et al., 2002), with accession GSE37761 and stemHypoxia package on the Bioconductor website (Fresno and Fernández, 2012b). In this dataset, Prado-Lopez et al. (2010) studied differentiation of human embryonic stem cells under hypoxia conditions. They measured gene expression at different time points under controlled oxygen levels. This experiment has a typical two-way ANOVA structure, where factor A stands for "*time*" with $a = 3$ levels $\{0.5, 1, 5\ days\}$, factor B for "*oxygen*" with $b = 3$ levels $\{1, 5, 21\%\}$ and $r = 2$ replicates, yielding a total of 18 samples. The rest of the dataset was excluded in order to account for a balanced design, as suggested by Smilde et al. (2005) to fulfill orthogonality assumptions in ANOVA decomposition.

First, we need to load stemHypoxia package to access R objects calling the function `data(stemHypoxia)`, which will then load the experimental `design` and gene expression intensities `M`.

```
R> library("stemHypoxia")
R> data(stemHypoxia)
```

Now we manipulate `design` object to maintain only those treatment levels which create a balanced dataset. Then, change `rownames(M)` of each gene in `M`, with their corresponding `M$Gene_ID`.

```
R> timeIndex<-design$time %in% c(0.5, 1, 5)
R> oxygenIndex<-design$oxygen %in% c(1, 5, 21)
R> design<-design[timeIndex & oxygenIndex, ]
R> design$time<-as.factor(design$time)
R> design$oxygen<-as.factor(design$oxygen)
R> rownames(M)<-M$Gene_ID
R> M<-M[, colnames(M) %in% design$samplename]
```

Now we can explore microarray gene expression data present on the `M` matrix, with $g = 40736$ rows (individuals/genes) and $n = 18$ columns (samples/microarrays). In addition, the experimental `design` data.frame contains main effect columns (e.g. *time* and *oxygen*) and the `samplename`. A brief summary of these objects is shown using `head` function:

```
R> head(design)
```

```
  time oxygen samplename
3  0.5      1    12h_1_1
4  0.5      1    12h_1_2
5  0.5      5    12h_5_1
6  0.5      5    12h_5_2
7  0.5     21   12h_21_1
8  0.5     21   12h_21_2
```

```
R> head(M)[, 1:3]
```

```
             12h_1_1 12h_1_2 12h_5_1
A_24_P66027    7.182   7.512   8.225
A_32_P77178    6.385   6.035   6.440
A_23_P212522   9.562   9.390   9.211
A_24_P934473   6.288   6.397   6.265
A_24_P9671    12.007  11.995  12.282
A_32_P29551   10.176   9.273   9.360
```

Once finished the preprocessing of the experiment data, `library("lmdme")` should be loaded. This instruction will automatically load the required packages: limma (Smyth et al., 2011) and pls (Mevik et al., 2011). Once they are loaded, the ANOVA decomposition of section 2.1 can be carried out using Equation (4) by `lmdme` function with the `model` formula, actual `data` and experimental `design`.

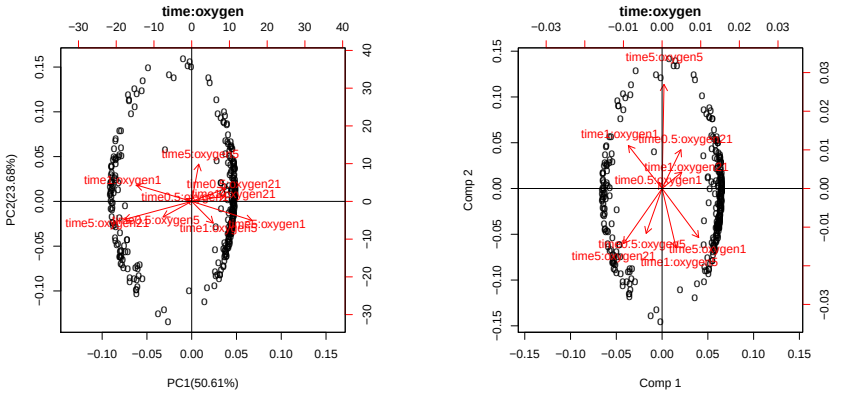
```
R> library("lmdme")
R> fit<-lmdme(model=~time*oxygen, data=M, design=design)
R> fit

lmdme object:
Data dimension:  40736  x  18
Design (head):
  time oxygen samplename
3  0.5      1    12h_1_1
4  0.5      1    12h_1_2
5  0.5      5    12h_5_1
6  0.5      5    12h_5_2
7  0.5     21   12h_21_1
8  0.5     21   12h_21_2

Model:~time * oxygen
Model decomposition:
  Step       Names             Formula CoefCols
1    1 (Intercept)                 ~ 1        1
2    2        time         ~ -1 + time        3
3    3      oxygen       ~ -1 + oxygen        3
4    4 time:oxygen ~ -1 + time:oxygen        9
```

The results of `lmdme` will be stored inside the `fit` object, which is an S4 R class. By invoking the `fit` object, a brief description of the used *data* and *design* are shown. In addition, the applied `Model` and a decomposition summary are shown. This `data.frame` describes for each `Step`, the applied `Formula` and `Names`, as well as the amount of estimated coefficients for each gene (`CoefCols`). At this point, we can choose those subjects/genes in which at least one interaction coefficient is statistically different from zero (F test on the coefficients) with a threshold p-value of 0.001 and perform ASCA on the interaction *coefficient* `term`, and PLS against the identity matrix (default option).

```
R> id<-F.p.values(fit, term="time:oxygen")<0.001
R> decomposition(fit, decomposition="pca", type="coefficient",
+    term="time:oxygen", subset=id, scale="row")
R> fit.plsr<-fit
R> decomposition(fit.plsr, decomposition="plsr", type="coefficient",
+    term="time:oxygen", subset=id, scale="row")
```

These instructions will perform ASCA and PLS `decomposition` over the `scaled="row"` version of the 305 selected subjects/genes (`subset=id`) on `fit` and `fit.plsr` object respectively. The results will be stored inside these objects. In addition, we have explicitly indicated `type="coefficient"` in order to apply the decomposition on the `coefficient` matrix, on interaction `term` `"time:oxygen"` ($\hat{B}_{\alpha\beta}$).
Now, we can visualize the associated `biplots` (see Figures 2 (a) and (b)).

```
R> biplot(fit, xlabs="o", expand=0.7)
R> biplot(fit.plsr, which="loadings", xlabs="o",
+    ylabs=colnames(coefficients(fit.plsr, term="time:oxygen")),
+    var.axes=TRUE)
```



(a) ANOVA Simultaneous Component Analysis

(b) ANOVA Partial Least Squares

Figure 2: Biplot on the decomposed interaction coefficients ($time \times oxygen$) on genes satisfying the F test with p-value < 0.001. It is worth noticing that the interaction matrix in the ASCA model is of rank 9-1, thus a score plot with 9 points is expected.

For visual clarity, `xlabs` are changed with the `"o"` symbol, instead of using the `rownames(M)` with manufacturer ids, and second axis with the `expand=0.7` option to avoid cutting off loading labels. In addition PLS `biplot`, is modified from the default `pls` behavior to obtain a graphic similar to ASCA output (`which="loadings"`). In this context,`ylabs` is changed to match the corresponding interaction `coefficients term` and `var.axes` is set to `TRUE`.
The ASCA biplot of the first two components (see Figure 2(a)), explains over 70% of the coefficient variance. The genes are arranged in an elliptical shape. In this context, it is possible to observe that some genes tend to interact with different combinations of time and oxygen. Similar behavior is present in PLS biplot of Figure 2(b).

The interaction effect on the `fit` object, can also be displayed by the use of the `loadingplot` function (see Figure 3). The figure shows for every combination of two consecutive levels of factors (time and oxygen), an interaction effect on the first component, which explains 50.61% of the total variance of the "time:oxygen" term.

```
R> loadingplot(fit, term.x="time", term.y="oxygen")
```

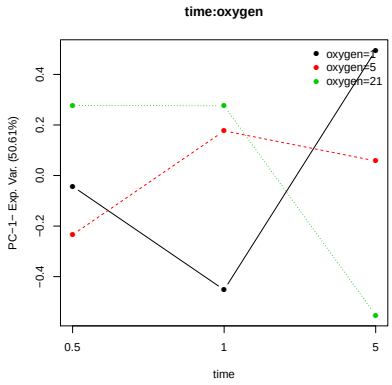


Figure 3: ANOVA Simultaneous Component Analysis `loadingplot` on genes satisfying the F test with p-value < 0.001 on the interaction coefficients ($time \times oxygen$).

In the case of an ANOVA-PCA/PLS analysis, the user only needs to change the `type = "residuals"` parameter in the `decomposition` function and perform a similar exploration.

Acknowledgements

Funding: This work was supported by the National Agency for Promoting Science and Technology, Argentina (PICT00667/07 to E.A.F. and PICT 2008-0807 BID to E.A.F.), Córdoba Ministry of Science and Technology, Argentina (PID2008 to E.A.F and PIP2009 to M.G.B.), Catholic University of Córdoba, Argentina and National Council of Scientific and Technical Research (CONICET), Argentina.

References

Abdi, H. and Williams, L. (2010). Principal component analysis. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(4):433–459.

De Haan, J., Wehrens, R., Bauerschmidt, S., Piek, E., Van Schaik, R., and Buydens, L. (2007). Interpretation of anova models for microarray data using pca. *Bioinformatics*, 23(2):184–190.

Edgar, R., Domrachev, M., and Lash, A. E. (2002). Gene expression omnibus: Ncbi gene expression and hybridization array data repository. *Nucleic Acids Research*, 30(1):207–210.

Fresno, C. and Fernández, E. A. (2012a). *lmdme: Linear Model Decomposition for Designed Multivariate Experiments*. R package version 1.1.5.

Fresno, C. and Fernández, E. A. (2012b). *stemHypoxia: Differentiation of Human Embryonic Stem Cells Under Hypoxia Gene Expression Dataset by Prado-Lopez et al. (2010)*. R package version 0.99.2.

Gentleman, R. C., Carey, V. J., Bates, D. M., and others (2004). Bioconductor: Open software development for computational biology and bioinformatics. *Genome Biology*, 5:R80.

Hanson, B. A. (2012). *ChemoSpec: Exploratory Chemometrics for Spectroscopy*. R package version 1.51-2.

Harrington, P. d. B., Vieira, N., Espinoza, J., Nien, J., Romero, R., and Yergey, A. (2005). Analysis of variance–principal component analysis: A soft tool for proteomic discovery. *Analytica Chimica Acta*, 544(1):118–127.

Mevik, B.-H., Wehrens, R., and Liland, K. H. (2011). *pls: Partial Least Squares and Principal Component regression*. R package version 2.3-0.

Nueda, M., Conesa, A., Westerhuis, J., Hoefsloot, H., Smilde, A., Talón, M., and Ferrer, A. (2007). Discovering gene expression patterns in time course microarray experiments by anova–sca. *Bioinformatics*, 23(14):1792–1800.

Prado-Lopez, S., Conesa, A., Armiñán, A., Martínez-Losa, M., Escobedo-Lucea, C., Gandia, C., Tarazona, S., Melguizo, D., Blesa, D., Montaner, D., et al. (2010). Hypoxia promotes efficient differentiation of human embryonic stem cells to functional endothelium. *Stem Cells*, 28(3):407–418.

R Development Core Team (2012). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0.

Shawe-Taylor, J. and Cristianini, N. (2004). *Kernel Methods for Pattern Analysis*. Cambridge University Press.

Smilde, A., Jansen, J., Hoefsloot, H., Lamers, R., Van Der Greef, J., and Timmerman, M. (2005). Anova-simultaneous component analysis (asca): A new tool for analyzing aesigned metabolomics data. *Bioinformatics*, 21(13):3043–3048.

Smyth, G. K. (2004). Linear models and empirical bayes methods for assessing differential expression in microarray experiments. *Statistical Applications in Genetics and Molecular Biology*, 3(1):Article 3.

Smyth, G. K., Ritchie, M., Silver, J., Wettenhall, J., Thorne, N., Langaas, M., Ferkingstad, E., Davy, M., Pepin, F., Choi, D., McCarthy, D., Wu, D., Oshlack, A., de Graaf, C., Hu, Y., Shi, W., and Phipson., B. (2011). *limma: Linear Models for Microarray Data*. R package version 3.12.1.

Subramanian, A., Tamayo, P., Mootha, V., Mukherjee, S., Ebert, B., Gillette, M., Paulovich, A., Pomeroy, S., Golub, T., Lander, E., et al. (2005). Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Sciences of the United States of America*, 102(43):15545–15550.

Tarazona, S., Prado-López, S., Dopazo, J., Ferrer, A., and Conesa, A. (2012). Variable selection for multifactorial genomic data. *Chemometrics and Intelligent Laboratory Systems*, 110(1):113–122.

Zwanenburg, G., Hoefsloot, H., Westerhuis, J., Jansen, J., and Smilde, A. (2011). Anova-principal component analysis and anova-simultaneous component analysis: A comparison. *Journal of Chemometrics*, 25(10):561–567.

Session Info

```
R> sessionInfo()

R version 3.0.0 (2013-04-03)
Platform: x86_64-apple-darwin10.8.0 (64-bit)

locale:
[1] C

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods
[7] base

other attached packages:
[1] lmdme_1.2.1        pls_2.3-0          limma_3.16.2
[4] stemHypoxia_0.99.3

loaded via a namespace (and not attached):
[1] tools_3.0.0
```