Journal of Statistical Software
May 2015, Volume 65, Issue 1. *http://www.jstatsoft.org/*

The R Package groc for Generalized Regression on Orthogonal Components

Martin Bilodeau
Université de Montréal

Pierre Lafaye de Micheaux
Université de Montréal

Smail Mahdi
University of West Indies

Abstract

The R package **groc** for generalized regression on orthogonal components contains functions for the prediction of q responses using a set of p predictors. The primary building block is the grid algorithm used to search for components (projections of the data) which are most dependent on the response. The package offers flexibility in the choice of the dependence measure which can be user-defined. The components are found sequentially. A first component is obtained and a smooth fit produces residuals. Then, a second component orthogonal to the first is found which is most dependent on the residuals, and so on. The package can handle models with more than one response. A panoply of models can be achieved through package **groc**: robust multiple or multivariate linear regression, nonparametric regression on orthogonal components, and classical or robust partial least squares models. Functions for predictions and cross-validation are available and helpful in model selection. The merit of a fit through cross-validation can be assessed with the predicted residual error sum of squares or the predicted residual error median absolute deviation which is more appropriate in the presence of outliers.

Keywords: C++, correlation, dependence, nonparametric, partial least squares, R, regression.

1. Introduction

The R (R Core Team 2015) package **groc** (Bilodeau and Lafaye de Micheaux 2015) is for generalized regression on orthogonal components. It bears some similarities with the package **pls** (Mevik, Wehrens, and Liland 2013) used to fit partial least squares (PLS) models. In the multivariate case, using the SIMPLS algorithm of de Jong (1993), package **pls** finds two linear combinations, one of responses and another of predictors, for which their covariance is maximum. The linear combination of predictors is called the first PLS component. The second PLS component is obtained by repeating this optimization of covariance on the deflated responses and predictors. The deflated responses are obtained by computing the residuals of the classical linear regression of the responses on the first PLS component, and similarly for the deflated predictors. Package **groc** generalizes this scheme. It finds two linear combinations, one of responses and another of predictors, for which a general measure of dependence is maximum. The deflated responses in package **groc** may take several forms depending on the application one has in mind. It can be the residuals of a linear regression, a robust linear regression, or a nonparametric regression of the responses on the first component. The deflated predictors are computed to cover the linear span of predictors orthogonal to the first component. The deflated predictors considered in package **groc** are the residuals of the classical or robust linear regression of the predictors on the first component. Package **groc** is available from the Comprehensive R Archive Network (CRAN) at `http://CRAN.R-project.org/package=groc`.

A brief description of related regression methods follows to situate package **groc** in the existing literature. The multiple linear regression model has a mean function of the form

$$\mathsf{E}(\mathbf{y}) = c_1\tilde{\mathbf{x}}_1 + \cdots + c_p\tilde{\mathbf{x}}_p.$$

A non-parametric generalization is the generalized additive model (GAM) given by

$$\mathsf{E}(\mathbf{y}) = g_1(\tilde{\mathbf{x}}_1) + \cdots + g_p(\tilde{\mathbf{x}}_p),$$

with unknown functions g_j estimated by the backfitting algorithm of Buja, Hastie, and Tibshirani (1989). A GAM fit can be obtained with the function `gam` of the package **mgcv** (Wood 2006, 2015). These two models are defined in terms of the original predictors $\tilde{\mathbf{x}}_1,\ldots,\tilde{\mathbf{x}}_p$. When the predictors are multicollinear, a better approach to a multiple linear regression model is the PLS model

$$\mathsf{E}(\mathbf{y}) = c_1\mathbf{t}_1 + \cdots + c_h\mathbf{t}_h.$$

The number h of components is determined by cross-validation. With the original predictors written as the matrix $\mathbf{X} = (\tilde{\mathbf{x}}_1, \ldots, \tilde{\mathbf{x}}_p)$, each PLS component is a linear combination $\mathbf{t}_{i+1} = \mathbf{X}\mathbf{r}_{i+1}$ obtained sequentially as the solution of the optimization

$$\mathbf{r}_{i+1} = \arg \max_{\substack{\|\mathbf{r}\|=1 \\ \mathbf{X}\mathbf{r}\perp\mathbf{t}_1,\ldots,\mathbf{t}_i}} \mathsf{COV}\left(\mathbf{X}\mathbf{r}, \mathbf{y} - \sum_{j=1}^{i} \hat{c}_j\mathbf{t}_j\right),$$

where the second argument to COV is the residual of the multiple linear regression of $\mathbf{y}$ on the previous components $\mathbf{t}_1, \ldots, \mathbf{t}_i$. This formulation was used to make the comparison of package **pls** to package **groc** and projection pursuit regression (PPR) clearer. In fact, because of the linearity and orthogonality properties of PLS, $\mathbf{y}$ could be used equivalently to the residuals in the second argument. The model in package **groc** can be viewed as a generalization to PLS since it is a generalized additive model

$$\mathsf{E}(\mathbf{y}) = g_1(\mathbf{t}_1) + \cdots + g_h(\mathbf{t}_h) \tag{1}$$

defined in terms of orthogonal components. The orthogonal components are obtained sequentially by solving the optimization

$$\mathbf{r}_{i+1} = \arg \max_{\substack{\|\mathbf{r}\|=1 \\ \mathbf{X}\mathbf{r}\perp\mathbf{t}_1,\ldots,\mathbf{t}_i}} D\left(\mathbf{X}\mathbf{r}, \mathbf{y} - \sum_{j=1}^{i} g_j(\mathbf{t}_j)\right),$$

where D is a general dependence measure, and letting $\mathbf{t}_{i+1} = \mathbf{X}\mathbf{r}_{i+1}$. This algorithm requires an evaluation of the dependence measure for each candidate direction $\mathbf{r}$. Once $\mathbf{t}_{i+1}$ is found, the next function g_{i+1} is found by the backfitting algorithm of GAM which simultaneously updates the previous functions $g_1, \ldots, g_i$. In the special parametric case $g_i(\mathbf{t}_i) = c_i\mathbf{t}_i$ and $D = \mathsf{COV}$, the function `groc` from package **groc** reduces to the function `plsr` from package **pls**. The **groc** proposal is more flexible than **pls** and is more restrictive than projection pursuit regression (function `ppr` in package **stats**) which fits a model as in Equation 1 without imposing the constraint of orthogonality of components. The sequential algorithm proposed in `ppr` solves the optimization

$$\mathbf{r}_{i+1} = \arg\min_{\|\mathbf{r}\|=1} \left\| \mathbf{y} - \sum_{j=1}^{i} g_j(\mathbf{t}_j) - S_{\mathbf{r}}(\mathbf{X}\mathbf{r}) \right\|^2 .$$

Here, for each direction $\mathbf{r}$ one must find the smooth fit $S_{\mathbf{r}}$ which minimizes the residual sum of squares. The optimal value of $\mathbf{r}$ denoted $\mathbf{r}_{i+1}$ yields the next component $\mathbf{t}_{i+1} = \mathbf{X}\mathbf{r}_{i+1}$ and the estimated function $g_{i+1} = S_{\mathbf{r}_{i+1}}$. Friedman and Stuetzle (1981) also suggest to update the functions $g_1, \ldots, g_{i+1}$ by backfitting the GAM model based on the components $\mathbf{t}_1, \ldots, \mathbf{t}_{i+1}$.

Regardless of the fitting method used, prediction mean squared error is small when a good compromise between squared bias and variance is reached. Increasing flexibility of the models is generally accompanied by increasing variances of predictions and decreasing squared biases. It is expected that `groc` yields higher variances than `plsr` but smaller than `ppr`, with reverse ordering when considering squared biases. Depending on the particular data analysis task at hand, one may choose the best fitting method by cross-validation in which case `groc` becomes one among several competitors.

By an appropriate choice of the dependence measure D and of the smoother used in the GAM fit, the flexibility of `groc` can be used to fit robust multiple or multivariate linear regression, nonparametric regression on orthogonal components, and classical or robust partial least squares models. The **groc** model situated between **pls** and `ppr` has not been considered as such in the literature. The numerical grid algorithm originally proposed by Croux, Filzmoser, and Oliveira (2007) and implemented in the package **pcaPP** (Filzmoser, Fritz, and Kalcher 2014) for projection-pursuit robust principal component analysis is also used for the first time in the regression context to find optimal components. The grid algorithm could also be used to search for optimal components of `ppr` models in replacement to the algorithm of Rosenbrock (1960) used by Friedman and Stuetzle (1981). However, it is not clear which numerical algorithm does the most extensive search over the unit sphere since the algorithm of Rosenbrock (1960) in the context of `ppr` is not described in detail in Friedman and Stuetzle (1981).

The framework is oriented towards prediction and not towards interpretation of components or inference about model parameters. Prediction accuracy is measured by cross-validation, a practice that is common in machine learning. The usefulness of package **groc** is illustrated on several examples using simulated and mostly real data. Examples of real data analysis are: the influence of five anatomical factors on wood specific gravity, volume of timber as a function of height and girth of trees, densities of NIR spectra of PET yarns, a particle physics experiment to predict the combined energy of a particle using six predictors, and the prediction of six attributes from a sensory panel using five physico-chemical quality parameters of olive oil.

The paper is organized as follows. Section 2 treats models with a single response. The first component is defined in Section 2 and the grid algorithm for its computation is described in Section 2.1. The grid algorithm in package **groc** is similar to the one described in Croux *et al.* (2007) and integrated in the package **pcaPP** for robust principal component analysis. Models with only one component comprise the robust multiple linear regression model in Section 2.2 and nonparametric regression on the first component in Section 2.3. The algorithm to sequentially obtain the first few components is described in Section 3. The numerical accuracy of the package **groc** is shown on a real data example where the grid algorithm reproduces the results of the package **pls** for which the optimization problem has an explicit solution. Section 4 treats model selection through cross-validation. The function `grocCrossval` based on the `predict` method for '`groc`' objects computes the PRESS (predicted residual errors sum of squares) and the PREMAD (predicted residual errors median absolute deviation) which may be more appropriate for data with vertical outliers and bad leverage points. As a dependence measure, the distance covariance of Székely, Rizzo, and Bakirov (2007) is also presented in Section 4 and its usefulness is shown through an analysis of simulated data with quadratic functions of two components, borrowed from Friedman and Stuetzle (1981), where package **groc** is compared to the packages **pls** and `ppr`. The orthogonal invariance of package **groc** in Section 5 is used to speed up computations through a preprocessing step in Section 5.1 which replaces the predictors by their principal components whenever the number of predictors surpasses the number of observations. Section 6 generalizes the algorithms to multivariate models with several responses. In Section 7, the package **groc** is used to fit robust multivariate partial least squares models and, as a particular case, robust multivariate linear models. The robustness of package **groc** allows the identification of vertical outliers as well as bad leverage points. The function `plot` method for '`groc`' objects graphs robust Mahalanobis distances of residuals versus robust Mahalanobis distances of components for this identification. An example shows that package **groc** is not affected by vertical outliers contrary to the MATLAB (The MathWorks Inc. 2014) package **prm** (Daszykowskia, Serneels, Walczak, Espen, and Croux 2013; Daszykowskia, Serneels, Kaczmarek, Espen, Croux, and Walczak 2007) for partial robust M-regression in Serneels, Croux, Filzmoser, and Espen (2005). The R code to reproduce all examples is available in the supplementary material.

2. The first component

We begin with regression models with multiple predictors and a single response. Höskuldsson (1998) gives the following interpretation of partial least squares regression: the first component is the linear combination of predictors with maximal covariance with the response. Partial least squares regression is available in the package **pls** through the function `plsr`. This proposal replaces the covariance by a general dependence measure. For n observations, the data consists of the response vector $\mathbf{y}$. The observations form the $n \times p$ design matrix $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)^\top$, where p is the number of predictors. It should be emphasized that $\mathbf{x}_i$ denotes case i, whereas $\tilde{\mathbf{x}}_j$ denoted variable j in Section 1. This means that $\mathbf{x}_i^\top$ is the ith row of $\mathbf{X}$ and $\tilde{\mathbf{x}}_j$ is its jth column. All variables are centered at their means. The first direction of partial least squares is obtained from the optimization

$$\mathbf{r}_1 = \arg \sup_{\|\mathbf{r}\|=1} \mathsf{COV}(\mathbf{X}\mathbf{r}, \mathbf{y}),$$

where $\mathsf{COV}(\mathbf{t}, \mathbf{u}) = \mathbf{t}^\top \mathbf{u}$ is the dot product. The first component is $\mathbf{t}_1 = \mathbf{X}\mathbf{r}_1$. Now, let $D(\mathbf{t}, \mathbf{u})$ be a measure of mutual dependence (thus more general than the covariance) between

Algorithm 1 Grid algorithm for optimal direction

```
Require: X, y
 1: $\mathbf{r} = \mathbf{e}_1$                                                     ▷ Initialization
 2: for $i = 1, \dots, N_c$ do
 3:     for $k = 1, \dots, N_g$ do
 4:         $\theta_k = -\frac{\pi}{2^{i-1}} + \frac{k-1}{N_g}\frac{\pi}{2^{i-2}}$        ▷ Discretization of $[-\pi/2^{i-1}, +\pi/2^{i-1})$
 5:     end for
 6:     for $j = 1, \dots, p$ do
 7:         $k^* = \arg\sup_{1 \le k \le N_g} D\left(\mathbf{X}\frac{\cos(\theta_k)\mathbf{r}+\sin(\theta_k)\mathbf{e}_j}{\|\cos(\theta_k)\mathbf{r}+\sin(\theta_k)\mathbf{e}_j\|}, \mathbf{y}\right)$    ▷ Optimization
 8:         $\mathbf{r} \leftarrow \frac{\cos(\theta_{k^*})\mathbf{r}+\sin(\theta_{k^*})\mathbf{e}_j}{\|\cos(\theta_{k^*})\mathbf{r}+\sin(\theta_{k^*})\mathbf{e}_j\|}$           ▷ Update r
 9:     end for
10: end for
11: return r                                                       ▷ Optimal direction
```

the vectors **t** and **u**, such as for example the distance covariance of Székely *et al.* (2007) or the statistic of Deheuvels described in Genest, Quessy, and Rémillard (2006). In our context of generalized regression on orthogonal components (`groc`), the optimization problem becomes

$$\mathbf{r}_1 = \arg \sup_{\|\mathbf{r}\|=1} D(\mathbf{X}\mathbf{r}, \mathbf{y}). \tag{2}$$

The first component is again $\mathbf{t}_1 = \mathbf{X}\mathbf{r}_1$. In this approach, the component is a linear combination of the original variables, which is not the case for kernel methods as in Rosipal and Trejo (2001). All dependent statistics considered in this paper are either nonnegative, or else, their sign can be inverted by a sign inversion of **r**. Hence, it is assumed without loss of generality that $D \geq 0$.

Once the first component $\mathbf{t}_1$ is found, a smooth fit of the response **y** on $\mathbf{t}_1$ is computed to obtain a predictive model

$$\hat{\mathbf{y}} = g_1(\mathbf{t}_1).$$

2.1. Grid algorithm for the first component

The algorithm for determining the first direction $\mathbf{r}_1$ is inspired from the grid algorithm in Croux *et al.* (2007) for robust principal components. In our context, the grid algorithm is as follows. Let $\mathbf{e}_j$ be the p-dimensional unit vector with a one in position j and zeros elsewhere. As described in Croux *et al.* (2007), during one of N_c cycles, a sequence of p grid searches over planes is carried out; the jth grid search updates the jth coordinate of **r**. When the second cycle starts, it is assumed that **r** is already pointing in the right direction but still needs local improvement. Hence, the grid search will not look over the whole plane, but only over the half-plane determined by the angles $[-\pi/2, +\pi/2)$. After every cycle, the search is limited to a more narrow interval of angles, but keeping the number N_g of grid points constant for every search. The grid Algorithm 1 searches for the global optimum for dependence functions D which need not be differentiable or even continuous as is the case when ranks are used.

The original grid algorithm in Croux *et al.* (2007) was proposed to find orthogonal principal components based on data **X**. For each candidate direction **r**, it computes a robust measure of scale of the component, say $\sigma(\mathbf{X}\mathbf{r})$. Algorithm 1 is the original grid algorithm with the exception that the dependence measure $D(\mathbf{Xr}, \mathbf{y})$ in step 7 is used in replacement to the measure of scale $\sigma(\mathbf{Xr})$.

2.2. Robust multiple linear regression

The least squares fit of a multiple linear regression finds the linear combination of predictors whose squared Pearson correlation with the response is the largest. A robust fit is obtained by using a robust correlation. A robust covariance matrix is the orthogonalized Gnanadesikan-Kettenring pairwise estimator (Maronna and Zamar 2002). The R package **robust** (Wang *et al.* 2014) contains a function `covRob` which computes this estimator with the option `estim = "pairwiseGK"`. It also has the option `corr = TRUE` for the correlation matrix. Our package **groc** has its own function `corrob` to compute this robust correlation. Our `corrob` function uses non-default values for the arguments of the original `covRob` function. Defining our own function `corrob` prevents to allow passing extra argument values to the D function. Note that our preference is to let the user define their own function D with only two arguments.

A robust multiple linear regression can be fitted in two steps:

1. Find the optimal direction
$$\mathbf{r}_1 = \arg \sup_{\|\mathbf{r}\|=1} \mathit{corrob}(\mathbf{Xr}, \mathbf{y})$$
and obtain the first component $\mathbf{t}_1 = \mathbf{Xr}_1$.

2. Fit a robust simple linear regression (with a single predictor) through the origin
$$\hat{\mathbf{y}} = c_1\mathbf{t}_1.$$

The fitted multiple linear regression becomes
$$\hat{\mathbf{y}} = c_1\mathbf{t}_1 = \mathbf{X}c_1\mathbf{r}_1 = \mathbf{X}\beta_1,$$
where $\beta_1 = c_1\mathbf{r}_1$. This robust multiple linear regression requires the repeated evaluation of robust bivariate correlations and a single robust simple linear regression.

Example 1. The data `wood` in the package **robustbase** (Todorov and Filzmoser 2009; Rousseeuw *et al.* 2015) is used to illustrate the proposed robust fitting method. The original data are from Draper and Smith (1966) and were used to determine the influence of five anatomical wood factors (`x1` to `x5`) on wood specific gravity (`y`). These data were contaminated by replacing a few observations with outliers in Rousseeuw and Leroy (1987). The robust fit was computed with the function `groc`. The option `method = "lts"` for least trimmed squares specifies the robust simple linear regression of step 2. Two common choices of the breakdown point are 0.5 for the highest possible breakdown and 0.25, the default in `groc`, which gives a better compromise between efficiency and breakdown. In Figure 1, which contains a plot of robust Mahalanobis distances for residuals versus robust Mahalanobis distances for components, four bad leverage points were found as in the analysis of Rousseeuw and Leroy (1987, p. 245). Identification of outliers was made by robustly scaling residuals with the τ scale of Yohai and Zamar (1988). The function `scaleTau2` of the package **robustbase** was used for that purpose. Scaled residuals in absolute value greater than the square root of the 0.975 quantile of a chi-square distribution with 1 degree of freedom (shown with a dashed line in Figure 1) were flagged as outliers. Identification of leverage points was done similarly using the component.

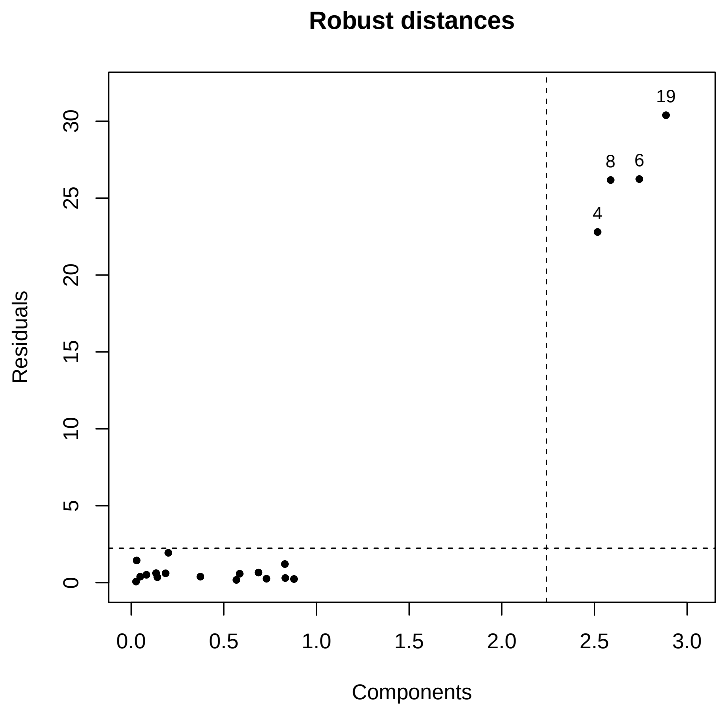

Figure 1: Diagnostic plot for the data `wood` in Example 1. Robust Mahalanobis distances for residuals versus robust Mahalanobis distances for components.

```
R> data("wood", package = "robustbase")
R> out <- groc(y ~ x1 + x2 + x3 + x4 + x5, ncomp = 1, data = wood,
+    D = corrob, method = "lts")
R> plot(out)
```

The squared robust correlation between the response $\mathbf{y}$ and the fitted value $\hat{\mathbf{y}}$ is obtained as follows:

```
R> corrob(wood$y, fitted(out))^2

[1] 0.9518
```

2.3. Nonparametric regression on the first component

A dependence measure suited to monotonic relations is Spearman's rank correlation which is the correlation between two vectors of ranks

$$\rho(\mathbf{t}, \mathbf{u}) = \frac{12}{n(n^2-1)} \sum_{i=1}^{n} \mathcal{R}(t_i)\mathcal{R}(u_i) - \frac{3(n+1)}{(n-1)},$$

where $\mathcal{R}(t_i)$ denotes the rank of t_i among $t_1, \ldots, t_n$.

The first component $\mathbf{t}_1$ is determined to maximize Spearman's correlation with the response $\mathbf{y}$. A smoothing spline is then fitted to the scatterplot of $\mathbf{y}$ versus $\mathbf{t}_1$.

Nguyen and Rojo (2009) considered the optimization problem

$$\mathbf{r}_1 = \arg\sup_{\|\mathbf{r}\|=1} [\mathcal{R}(\mathbf{y})]^\top \mathcal{R}(\mathbf{X})\mathbf{r}, \tag{3}$$

where $\mathcal{R}(\mathbf{y})$ is the vector of ranks corresponding to $\mathbf{y}$ and $\mathcal{R}(\mathbf{X})$ is the matrix of ranks corresponding to predictors computed columnwise, i.e., separately for each predictor. The transformation to ranks is done in an attempt to robustify the partial least squares regression. It must be noted, however, that $\mathcal{R}(\mathbf{X})\mathbf{r}$ is different from $\mathcal{R}(\mathbf{X}\mathbf{r})$, hence the solution to problem (3) does not give the component $\mathbf{X}\mathbf{r}$ with the largest Spearman's correlation with the response.

Example 2. Consider the data `trees` of the package **MASS** (Venables and Ripley 2002) which provides measurements of the girth, height and volume of timber in 31 felled black cherry trees. The relation of volume to height and girth is nonlinear and most likely monotone. The squared Pearson correlation between actual volume and fitted volume by `groc` with the option `D = spearman` is obtained below:

```
R> data("trees", package = "datasets")
R> out <- groc(Volume ~ Height + Girth, ncomp = 1, D = spearman,
+    method = "s", data = trees)
R> cor(trees$Volume, fitted(out))^2

[1] 0.9769
```

Using the option `method = "s"`, the smoothing spline fitted to the scatterplot of volume versus the first component is shown in Figure 2.

```
R> plot(out$T, trees$Volume, xlab = "First component", ylab = "Volume",
+    pch = 20)
R> lines(sort(out$T), fitted(out)[order(out$T)])
```

But since volume is strictly positive, a Box-Cox transformation is also possible.

```
R> library("MASS")
R> out <- boxcox(Volume ~ Height + Girth, data = trees,
+    lambda = seq(-0.5, 0.5, length = 100), plotit = FALSE)
R> lambda <- out$x[which.max(out$y)]
R> out <- lm(Volume^lambda ~ Height + Girth, data = trees)
```

It achieved approximately the same fit with a squared Pearson correlation of

```
R> cor(trees$Volume, fitted(out)^(1/lambda))^2

[1] 0.9768
```

In applications where the response may take negative values, the Box-Cox transformation requires adding a somewhat arbitrary constant to achieve positivity. In this case, the `groc` fit can be applied directly which may be considered an advantage.

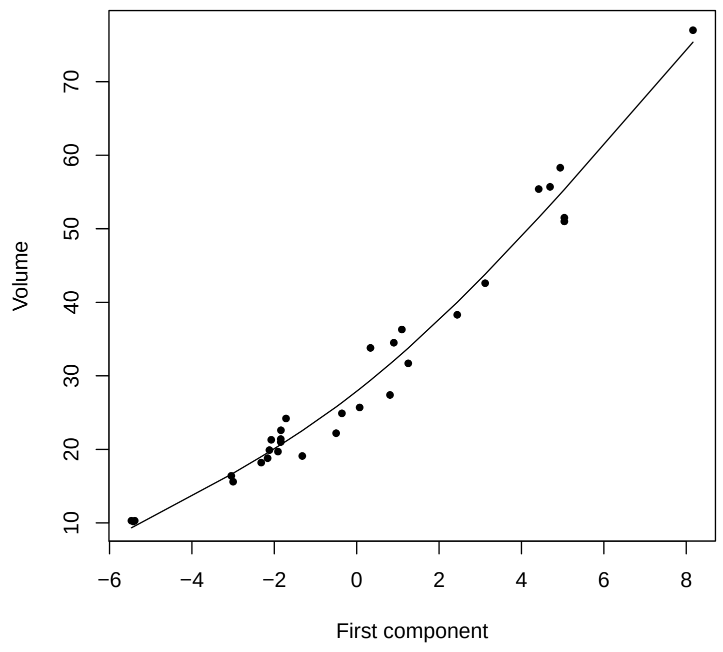

Figure 2: Smoothing spline of volume on first component determined by Spearman correlation.

3. Subsequent components

Initially, let $\mathbf{X}_0 = \mathbf{X}$ and $\mathbf{y}_0 = \mathbf{y}$. Having computed the first component, $\mathbf{t}_1 = \mathbf{X}_0\mathbf{r}_1$, each predictor in $\mathbf{X}_0$ is regressed on $\mathbf{t}_1$ to obtain the residuals

$$\mathbf{X}_1 = \left[\mathbf{I} - \frac{\mathbf{t}_1\mathbf{t}_1^\top}{\mathbf{t}_1^\top\mathbf{t}_1}\right]\mathbf{X}_0 = \mathbf{X}_0 - \mathbf{t}_1(\mathbf{t}_1^\top\mathbf{X}_0)/(\mathbf{t}_1^\top\mathbf{t}_1).$$

Since residuals are orthogonal to predictors, then $\mathbf{X}_1\mathbf{r}$ is orthogonal to $\mathbf{t}_1$, for all $\mathbf{r}$. The residuals are then computed

$$\mathbf{y}_1 = \mathbf{y}_0 - g_1(\mathbf{t}_1),$$

where g_1 is the result of some smooth fit applied to the scatterplot of $\mathbf{y}_0$ and $\mathbf{t}_1$. In other words, after the first component $\mathbf{t}_1$ is found, a fit is made of $\mathbf{y}_0$ using $\mathbf{t}_1$ as the predictor in a smooth fit. The residual of this fit is $\mathbf{y}_1$. If the user is satisfied with this residual, then only one component is used. Otherwise, a search is done to find some structure in the residual in the orthogonal complement of the space spanned by $\mathbf{t}_1$.

For the second component, optimize

$$\mathbf{r}_2 = \arg \sup_{\|\mathbf{r}\|=1} D(\mathbf{X}_1\mathbf{r}, \mathbf{y}_1)$$

as before using the grid Algorithm 1. Then, $\mathbf{t}_2 = \mathbf{X}_1\mathbf{r}_2$ is the second component.

For partial least squares regression, unlike for the proposed generalized regression on orthogonal components, Tenenhaus (1998) remarks that there is no need to modify $\mathbf{y}_0$. Indeed,

$$\mathbf{y}_1 = \left[\mathbf{I} - \frac{\mathbf{t}_1\mathbf{t}_1^\top}{\mathbf{t}_1^\top\mathbf{t}_1}\right]\mathbf{y}_0 = \mathbf{y}_0 - c_1\mathbf{t}_1$$

Algorithm 2 Algorithm for the first h orthogonal components

```
Require: $\mathbf{X}$, $\mathbf{y}$
 1: $\mathbf{X}_0 = \mathbf{X}$ and $\mathbf{y}_0 = \mathbf{y}$
 2: for $i = 1, \ldots, h$ do
 3:     $\mathbf{r}_i = \arg\sup_{\mathbf{r}} \{D(\mathbf{X}\mathbf{r}, \mathbf{y}), \|\mathbf{r}\| = 1\}$    ▷ Optimal direction with Algorithm 1
 4:     $\mathbf{t}_i = \mathbf{X}\mathbf{r}_i$    ▷ Orthogonal components
 5:     $\hat{\mathbf{y}}_0 = g_1(\mathbf{t}_1) + \cdots + g_i(\mathbf{t}_i)$    ▷ GAM fit
 6:     $\mathbf{b}_i = \mathbf{X}^\top \mathbf{t}_i / (\mathbf{t}_i^\top \mathbf{t}_i)$    ▷ Regression of $\mathbf{X}$ on $\mathbf{t}_i$
 7:     $\mathbf{X} \leftarrow \mathbf{X} - \mathbf{t}_i \mathbf{b}_i^\top$    ▷ Deflation of $\mathbf{X}$
 8:     $\mathbf{y} \leftarrow \mathbf{y}_0 - \hat{\mathbf{y}}_0$    ▷ Deflation of $\mathbf{y}$
 9: end for
10: return $\mathbf{t}_1, \ldots, \mathbf{t}_h$, $g_1, \ldots, g_h$.
```

is the residual of the linear regression through the origin of $\mathbf{y}_0$ on $\mathbf{t}_1$, where $c_1 = \mathbf{t}_1^\top \mathbf{y}_0 / (\mathbf{t}_1^\top \mathbf{t}_1)$. In this case, for the second component, since $\mathbf{X}_1^\top \mathbf{t}_1 = 0$,

$$\mathsf{COV}(\mathbf{X}_1\mathbf{r}, \mathbf{y}_1) = \mathsf{COV}(\mathbf{X}_1\mathbf{r}, \mathbf{y}_0).$$

Returning to `groc`, all the components can be computed sequentially. Having computed the first few components $\mathbf{t}_1, \ldots, \mathbf{t}_h$, compute

$$\mathbf{X}_h = \left[\mathbf{I} - \frac{\mathbf{t}_1\mathbf{t}_1^\top}{\mathbf{t}_1^\top\mathbf{t}_1} - \cdots - \frac{\mathbf{t}_h\mathbf{t}_h^\top}{\mathbf{t}_h^\top\mathbf{t}_h}\right]\mathbf{X} = \mathbf{X}_{h-1} - \mathbf{t}_h(\mathbf{t}_h^\top\mathbf{X}_{h-1})/(\mathbf{t}_h^\top\mathbf{t}_h).$$

Then, the residual of a generalized additive model (GAM) fit of $\mathbf{y}_0$ on the first h components is

$$\mathbf{y}_h = \mathbf{y}_0 - \sum_{i=1}^{h} g_i(\mathbf{t}_i). \tag{4}$$

For all $\mathbf{r}$, $\mathbf{X}_h\mathbf{r}$ is orthogonal to $\mathbf{t}_1, \ldots, \mathbf{t}_h$. Therefore, the optimization problem for the next component is

$$\mathbf{r}_{h+1} = \arg\sup_{\|\mathbf{r}\|=1} D(\mathbf{X}_h\mathbf{r}, \mathbf{y}_h).$$

The next orthogonal component is $\mathbf{t}_{h+1} = \mathbf{X}_h\mathbf{r}_{h+1}$.

In the GAM fit (4), although the components are orthogonal, the last function g_h is fitted and the previous functions $g_1, \ldots, g_{h-1}$ are updated by backfitting to improve the fit. Backfitting could be ignored to speed up computations. This is unlike `plsr`, in which case the last coefficient c_h in the linear function $g_h(\mathbf{t}_h) = c_h\mathbf{t}_h$ is obtained by ordinary least squares and the previous coefficients $c_1, \ldots, c_{h-1}$ remain unchanged.

The method in `groc` for computing the fit on the first h components is summarized in Algorithm 2.

Example 3. Consider again the data `wood` with one response and five predictors of Example 1. This example is used to show the numerical accuracy of Algorithms 1 and 2. Recall that `groc` with `covariance` as the dependence statistic and `lm` (ordinary least squares) as the smoother is equivalent to `plsr` in the sense that they are two solutions to the same optimization problem. However, unlike `plsr` which relies on an explicit solution of the optimization, `groc` does an extensive search of directions on the unit sphere in five dimensional Euclidean space using the grid algorithm. The relative differences, in percentages, between fitted values

```
R> plsr.out <- plsr(y ~ x1 + x2 + x3 + x4 + x5, data = wood)
R> groc.out <- groc(y ~ x1 + x2 + x3 + x4 + x5, data = wood)
R> apply(abs((fitted(plsr.out) - fitted(groc.out)) /
+    fitted(plsr.out)), 3, max) * 100

  1 comps   2 comps   3 comps   4 comps   5 comps
1.768e-03 2.012e-03 1.160e-03 2.299e-03 2.119e-14
```

are all very small regardless of the numbers of components.

The much smaller relative difference for $h = p$ (five here) components is expected. Indeed, `plsr` and `groc` with the option `method = "lm"` produce a least squares regression on the p extracted components which are linearly independent and even orthogonal. In both cases, the matrices of components which may differ have the form $\mathbf{T} = \mathbf{XA}$, where the $p \times p$ matrices $\mathbf{A}$ are non singular. Since fitted values of least squares regression are invariant to non singular linear transformations of the predictors, the two methods numerically coincide in this case. This holds regardless of the dependence function D used in `groc`. Due again to the invariance property, when $h = p$, a `groc` fit with the option `method = "lts"` produces the same result as a least trimmed squares fit of $\mathbf{y}$ on the original predictors $\mathbf{X}$.

4. Model selection

The merits of several models are compared by cross-validation. A cross-validation criterion is defined in terms of the absolute error of predictions $|y - \hat{y}|$. The package **pls** has the function `crossval` to compute the predicted residual errors sum of squares (PRESS) by cross-validation which is the sum of squares of $|y - \hat{y}|$ over all predictions made. The corresponding function for package **groc** is `grocCrossval`. In addition to PRESS, `grocCrossval` also computes the predicted residual errors median absolute deviation (PREMAD) which is the median of $|y - \hat{y}|$ over all predictions. It is more relevant for model selection in the presence of outliers.

For the next example, we introduce a dependence measure which may not be known to most users. For accuracy of the optimization by the grid algorithm, dependence measures which are smooth functions of the data should be preferred. A smooth dependence measure is the squared distance covariance of Székely *et al.* (2007)

$$dcov^2(\mathbf{t}, \mathbf{u}) = \frac{1}{n^2} \sum_{i,j=1}^{n} A_{ij} B_{ij}, \tag{5}$$

where

$$\begin{array}{lll} a_{ij} = |t_i - t_j|, & \bar{a}_{i.} = \frac{1}{n} \sum_{j=1}^{n} a_{ij}, & \bar{a}_{.j} = \frac{1}{n} \sum_{i=1}^{n} a_{ij}, \\ \bar{a}_{..} = \frac{1}{n^2} \sum_{i,j=1}^{n} a_{ij}, & A_{ij} = a_{ij} - \bar{a}_{i.} - \bar{a}_{.j} + \bar{a}_{..}, & \end{array}$$

and, similarly, $b_{ij} = |u_i - u_j|$, $B_{ij} = b_{ij} - \bar{b}_{i.} - \bar{b}_{.j} + \bar{b}_{..}$. It consistently estimates

$$\frac{1}{\pi^2} \int \frac{|\varphi_{T,U}(t,u) - \varphi_T(t)\varphi_U(u)|^2}{t^2 u^2} dt du,$$

where $\varphi_{T,U}$ is the joint characteristic function, and φ_T, φ_U are the marginal versions. It is thus universal in the sense that it can detect any kind of dependence. A similar dependence measure is the statistic of Deheuvels described in Genest *et al.* (2006). It is a discontinuous function of the data since it is computed from ranks. It is also universal as it consistently estimates

$$\int [F_{T,U}(t,u) - F_T(t)F_U(u)]^2 \, dF_T(t)F_U(u),$$

in terms of distribution functions. Our preference, however, is for the distance covariance of Székely *et al.* (2007) which leads to greater numerical accuracy of the optimization in the grid algorithm.

Example 4. A synthetic example is the nonlinear regression model with conditional mean $\mathsf{E}(y \mid x_1, x_2) = x_1 x_2$ found in Friedman and Stuetzle (1981) to illustrate the method of projection pursuit regression. This model can be rewritten as a sum of two quadratic functions of linear functions of x_1 and x_2,

$$y = \frac{1}{4}(x_1 + x_2)^2 - \frac{1}{4}(x_1 - x_2)^2 + \epsilon. \tag{6}$$

The predictors x_1 and x_2 are independently and uniformly distributed over the interval $(-1, 1)$ and the errors ϵ are normally distributed with a variance of 0.04. The sample size is $n = 200$. Two models were fitted to these simulated data: a `plsr` model and a `groc` model using the distance covariance `dcov` and smoothing splines. Prediction errors as measured by PRESS values are obtained by ten-fold cross-validation using a `groc` model with two components:

```
R> n <- 200
R> x1 <- runif(n, -1, 1)
R> x2 <- runif(n, -1, 1)
R> y <- x1 * x2 + rnorm(n, 0, sqrt(.04))
R> data <- data.frame(x1 = x1, x2 = x2, y = y)
R> plsr.out <- plsr(y ~ x1 + x2, data = data)
R> groc.out <- groc(y ~ x1 + x2, D = dcov, method = "s", data = data)
R> plsr.v <- crossval(plsr.out, segment.type = "consecutive")
R> groc.v <- grocCrossval(groc.out, segment.type = "consecutive")
```

The PRESS values obtained using our function:

```
R> groc.v$validation$PRESS

  1 comps 2 comps
y   12.53   8.697
```

are considerably smaller than those obtained for `plsr`:

```
R> plsr.v$validation$PRESS

  1 comps 2 comps
y   25.88   25.85
```

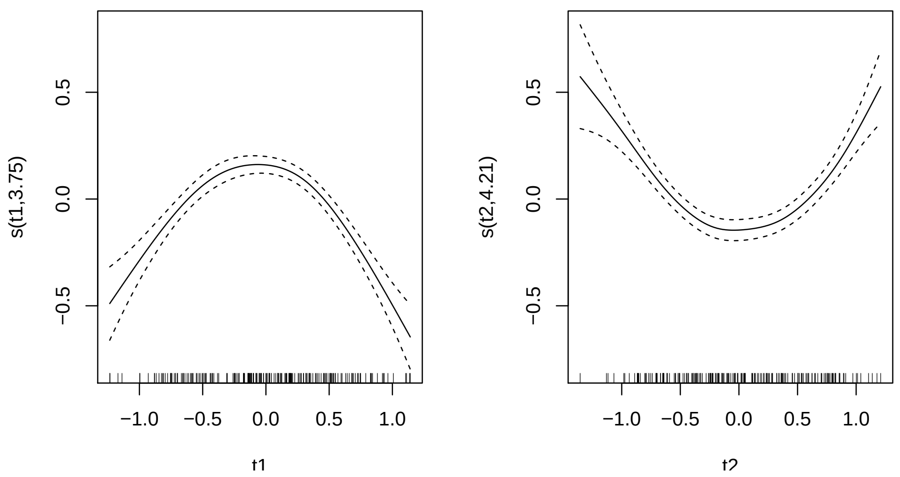

Figure 3: Plot of fitted functions by a `groc` model with components found by the distance covariance measure and smoothing splines.

The functions found by smoothing splines of components in Figure 3 resemble two quadratic functions of opposite signs as in Equation 6. Since the true model is not additive, a GAM fit using smoothing splines of the original variables would be unable to find this structure in the data where our method was successful. This example is concluded with a comparison of `groc` with projection pursuit regression (`ppr`). Since there is no cross-validation function for `ppr`, the value of PRESS for ten-fold cross-validation is evaluated directly by deleting subsets of 20 consecutive observations. It ensures that predictions for `groc` and `ppr` are made exactly at the same data points. Evaluation of PRESS for a `ppr` fit is very easy since the function `predict` may be applied to an R object of class '`ppr`'. In terms of PRESS, predictions with `groc` were better than with `ppr` with a PRESS value of 10.21 for the `ppr` model with two components. The `ppr` fit was computed with the option `optlevel = 3`, the highest level of thoroughness of the optimization routine. It may be argued, however, that this example favors `groc` because the two components $x_1 + x_2$ and $x_1 - x_2$ are uncorrelated. Components in `ppr` are generally not orthogonal.

5. Orthogonal invariance

Let $\mathbf{r}_1$ be the solution to the generalized optimization problem (2) for the first component. Consider the orthogonal transformation $\mathbf{X} \mapsto \mathbf{XG}$ for an orthogonal $p \times p$ matrix $\mathbf{G}$. The optimization problem for the transformed data is

$$\sup_{\|\mathbf{r}\|=1} D(\mathbf{XGr}, \mathbf{y}).$$

Since $\mathbf{Gr}$ runs over the unit sphere when $\mathbf{r}$ does, the solution is $\mathbf{G}^\top \mathbf{r}_1$. Hence, the first component is invariant to this orthogonal transformation since

$$\mathbf{Xr}_1 \mapsto (\mathbf{XG})(\mathbf{G}^\top \mathbf{r}_1) = \mathbf{Xr}_1.$$

The orthogonal invariance of the other components follows by induction as in Denham (1995) for partial least squares.

5.1. Preprocessing of the data

Denham (1995) wrote: "By the use of orthogonal transformations it is therefore possible to reduce the original problem to a much smaller equivalent problem in which p is effectively reduced to the rank of the centered matrix, $\mathbf{X}$, which for typical infrared problems will be $n-1$." It is most important to reduce computational time. Remember that the grid Algorithm 1 for each direction searches for a unit vector in a p dimensional space. This can be a burden especially when p is much larger than n which is often the case for partial least squares applications. The following preprocessing can be done as soon as $p > n$.

It consists of entering Algorithm 2 for the first h orthogonal components not with $\mathbf{X}$ but with principal components derived from $\mathbf{X}$. For mean centered $\mathbf{X}$, a principal component analysis gives $\mathbf{XH} = \mathbf{U}$, where $\mathbf{X}$ is $n \times p$ of rank a (say), $\mathbf{H}$ is $p \times a$ with orthonormal columns (loadings) and $\mathbf{U}$ is the matrix $n \times a$ of principal components (scores). In fact, if all principal components were calculated we would have

$$\mathbf{X}(\mathbf{H}, \mathbf{F}_{p\times(p-a)}) = (\mathbf{U}, \mathbf{0}_{n\times(p-a)}),$$

where $(\mathbf{H}, \mathbf{F}_{p\times(p-a)})$ is now $p \times p$ orthogonal. The last $p-a$ principal components are degenerate at 0; all the variability is explained with the first a components. The data actually belongs to an a dimensional space and after the appropriate rotation the last $p-a$ components of the data are all 0. So $\mathbf{X}$ and $\mathbf{U}$ are the same data apart from the rotation. As a consequence, the components from $\mathbf{X}$ are the same as the one from $\mathbf{U}$. Indeed, because of the orthogonal invariance, the components from $\mathbf{X}$ are the same as those from $(\mathbf{U}, \mathbf{0}_{n\times(p-a)})$. The following condition on the dependence statistic is assumed.

Condition 1. Either $D(c\mathbf{t}, \mathbf{u}) = cD(\mathbf{t}, \mathbf{u})$, or $D(c\mathbf{t}, \mathbf{u}) = D(\mathbf{t}, \mathbf{u})$, for all scalar $c > 0$.

The dependence statistic, after the orthogonal transformation, is

$$D(\mathbf{X}(\mathbf{H}, \mathbf{F})\mathbf{r}, \mathbf{y}) = D((\mathbf{U}, \mathbf{0})\mathbf{r}, \mathbf{y}) = D(\mathbf{Uf}, \mathbf{y}),$$

where $\mathbf{f}$ is the vector of the first a components of $\mathbf{r}$, which satisfies $\|\mathbf{f}\| \leq \|\mathbf{r}\| = 1$. Then, the first direction is the solution to

$$\sup_{\|\mathbf{f}\|\leq 1} D(\mathbf{Uf}, \mathbf{y}).$$

However, since D can always increase, or remain constant, when $\mathbf{f}$ is multiplied by a scalar $c \geq 1$, it follows that the solution is always on the boundary, $\|\mathbf{f}\| = 1$.

Example 5. Orthogonal invariance is shown empirically by way of the chemometric data `yarn` of the **pls** package. The response is density and the predictors are the NIR spectra. There are 28 observations and 268 predictors. The function `plsr` does not preprocess to principal components, whereas, our function `groc` preprocesses the data whenever $n < p$. The maximal number of components in this case is 27; it is 26 when the merit of the fit is evaluated by leave-one-out cross-validation. The relative differences between fitted values with and without preprocessing were for all observations, response variables and number of components less than 0.03% even in high dimensions. Since optimal partial least squares directions have an explicit solution, it cannot be expected that `groc` which did an extensive search by the grid Algorithm 1 in about 1.6 sec performs as fast as `plsr` which required only 0.02 sec. The PRESS values produced respectively by `plsr` and `groc` were also very close and suggested a model with 12 components. On the grounds of the PREMAD values, a model with only 8 components may be suitable.

```
R> data("yarn", package = "pls")
R> n <- nrow(yarn)
R> plsr.out <- plsr(density ~ NIR, ncomp = n - 2, data = yarn)
R> groc.out <- groc(density ~ NIR, Nc = 20, ncomp = n - 2, data = yarn)
R> plsr.v <- crossval(plsr.out, segments = n, trace = FALSE)
R> groc.v <- grocCrossval(groc.out, segments = n, trace = FALSE)
```

Note that the PRESS values for `plsr` (not shown) can be obtained using subsetting with `plsr.v$validation$PRESS`.

```
R> summary(groc.v)

Data:         X dimension: 268
       Y dimension: 1
Fit method: lm
Number of components considered: 26

Cross-validated using 28 leave-one-out segments.
VALIDATION: PRESS
         1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
density    592.5    425.8    122.3    16.54     7.01    5.482
         7 comps  8 comps  9 comps  10 comps  11 comps  12 comps
density    2.463    1.955    1.769     1.484     1.333     1.199
         13 comps  14 comps  15 comps  16 comps  17 comps  18 comps
density     1.283     1.332     1.387     1.441     1.448     1.452
         19 comps  20 comps  21 comps  22 comps  23 comps  24 comps
density      1.44     1.441     1.437     1.436     1.437     1.437
         25 comps  26 comps
density     1.436     1.436

VALIDATION: PREMAD
         1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
density     2.16      1.9    1.073   0.3793    0.216   0.2446
         7 comps  8 comps  9 comps  10 comps  11 comps  12 comps
density   0.1224    0.103    0.127    0.1355     0.116    0.1108
         13 comps  14 comps  15 comps  16 comps  17 comps  18 comps
density    0.1113     0.111    0.1048    0.1054    0.1061    0.1076
         19 comps  20 comps  21 comps  22 comps  23 comps  24 comps
density    0.1072    0.1048    0.1034    0.1032    0.1033    0.1031
         25 comps  26 comps
density    0.1032    0.1032
```

```
VALIDATION: RMSEP
         1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
density      4.6      3.9     2.09   0.7686   0.5004   0.4425
         7 comps  8 comps  9 comps  10 comps  11 comps  12 comps
density   0.2966   0.2643   0.2514    0.2303    0.2182     0.207
         13 comps  14 comps  15 comps  16 comps  17 comps  18 comps
density    0.2141    0.2181    0.2226    0.2269    0.2274    0.2277
         19 comps  20 comps  21 comps  22 comps  23 comps  24 comps
density    0.2268    0.2269    0.2266    0.2265    0.2265    0.2265
         25 comps  26 comps
density    0.2265    0.2265

Data:   X dimension: 268
        Y dimension: 1
Fit method: lm
Number of components considered: 26

Cross-validated using 28 leave-one-out segments.
VALIDATION: PRESS
        1 comps 2 comps  3 comps  4 comps  5 comps  6 comps  7 comps 8 comps
density 592.5   425.8    122.3    16.54    7.01     5.482    2.463   1.955
        9 comps 10 comps 11 comps 12 comps 13 comps 14 comps 15 comps
density 1.769   1.484    1.333    1.199    1.283    1.332    1.387
       16 comps 17 comps 18 comps 19 comps 20 comps 21 comps 22 comps
density 1.441   1.448    1.452     1.44    1.441    1.437    1.436
       23 comps 24 comps 25 comps 26 comps
density 1.437   1.437    1.436    1.436

VALIDATION: PREMAD
        1 comps 2 comps 3 comps 4 comps 5 comps 6 comps 7 comps 8 comps
density 2.16    1.9     1.073   0.3793  0.216   0.2446  0.1224  0.103
        9 comps 10 comps 11 comps 12 comps 13 comps 14 comps 15 comps
density 0.127    0.1355   0.116    0.1108   0.1113   0.111    0.1048
       16 comps 17 comps 18 comps 19 comps 20 comps 21 comps 22 comps
density 0.1054   0.1061   0.1076   0.1072   0.1048   0.1034   0.1032
       23 comps 24 comps 25 comps 26 comps
density 0.1033   0.1031   0.1032   0.1032
```

A nonlinear example is now analyzed.

Example 6. The data `prim7` of the **groc** package is a particle physics experiment analyzed by projection pursuit regression in Friedman and Stuetzle (1981). It has 7 variables on 500 observations. In our approach, a nonlinear model to predict the combined energy of the three π mesons X_1 using the other variables $X_2, \ldots, X_7$ as predictors was fitted using the distance covariance measure and smoothing splines. The orthogonal invariance is numerically verified in this example with $n \geq p$. Preprocessing must be done separately because the function groc does preprocessing only when $n < p$. The first three directions $\mathbf{r}_i$, $i = 1, 2, 3$, without preprocessing

```
R> data("prim7", package = "groc")
R> prim7.out <- groc(X1 ~ ., ncomp = 3, D = dcov, method = "s", data = prim7)
R> prim7.out$R

    Comp 1  Comp 2   Comp 3
X2 -0.3093  0.2677 -0.62536
X3  0.1544  0.4163  0.13388
X4 -0.3750 -0.6825  0.31972
X5  0.1490  0.4040  0.59173
X6  0.6012 -0.2787 -0.36735
X7  0.5968 -0.2199  0.06076
```

and with preprocessing

```
R> pca <- princomp(~ ., data = as.data.frame(prim7[, -1]))
R> prim7.pca <- data.frame(X1 = prim7$X1, scores = pca$scores)
R> prim7.pca.out <- groc(X1 ~ ., ncomp = 3, D = dcov, method = "s",
+    data = prim7.pca)
R> pca$loadings %*% prim7.pca.out$R

    Comp 1  Comp 2  Comp 3
X2 -0.3097  0.2657 -0.6237
X3  0.1546  0.4183  0.1347
X4 -0.3748 -0.6821  0.3218
X5  0.1487  0.4050  0.5982
X6  0.6013 -0.2759 -0.3583
X7  0.5966 -0.2218  0.0554
```

are very close. This nonlinear fit is now compared to plsr by means of ten-fold cross-validation. The PRESS values for the nonlinear groc model

```
R> groc.v <- grocCrossval(prim7.out, segment.type = "consecutive")

Segment: 1 2 3 4 5 6 7 8 9 10

R> groc.v$validation$PRESS

   1 comps 2 comps 3 comps
X1    1239   765.4   471.4
```

are markedly smaller than those for the linear plsr model.

```
R> plsr.out <- plsr(X1 ~ ., ncomp = 3, data = prim7)
R> plsr.v <- crossval(plsr.out, segment.type = "consecutive")
R> plsr.v$validation$PRESS
```

```
   1 comps 2 comps 3 comps
X1    2428    1949    1846
```

The groc fit was also compared to ppr with three components. Based on the PRESS value of ppr (i.e., 640), the groc fit provided a reduction of roughly 20% in prediction errors over ppr.

The following example illustrates the usefulness of package **groc** in the context where $n < p$ and the simulated model is nonlinear with two components.

Example 7. The simulated model is

$$y = (x_1 + x_2)^2 + (x_1 + 5x_2)^2 + \epsilon,$$

where $x_1,\ldots,$ x_5 are independently and uniformly distributed over the interval $(-1, 1)$ and the errors ϵ are normally distributed with a variance of 1. Note that $x_3, \ldots, x_5$ are part of the data but not involved in the data generating process. The sample size is $n = 50$. The data also contain 50 more predictors perfectly collinear with the first 5 predictors resulting in a situation where $n = 50$ and $p = 55$. It should be noted that the two variables $x_1 + x_2$ and $x_1 + 5x_2$ are not uncorrelated, and therefore, the two components are not approximately orthogonal. A model is fitted with plsr, ppr and groc always using two components. For each of the three fits, the correlation between the actual response and the fitted value is computed and the user time is recorded. This model fitting scheme is repeated 30 times. Average correlations and user times over these 30 fits are finally reported in the following order: plsr, ppr, and groc. The average correlations

```
[1] 0.3193 0.9228 0.9589
```

achieved by ppr and groc are both very high as compared to plsr. In terms of average user times,

```
[1] 0.004933 3.511100 0.903033
```

plsr is the clear winner with groc being about four times as fast as ppr.

6. The multivariate case

The response matrix $\mathbf{Y}$ is now of dimension $n \times q$. Initially, let $\mathbf{X} = \mathbf{X}_0$ and $\mathbf{Y} = \mathbf{Y}_0$. The first direction $\mathbf{r}_1$ is the solution for $\mathbf{r}$ in the optimization

$$\sup_{\|\mathbf{r}\|=\|\mathbf{s}\|=1} D(\mathbf{X}_0\mathbf{r}, \mathbf{Y}_0\mathbf{s}). \tag{7}$$

The first component is $\mathbf{t}_1 = \mathbf{X}_0\mathbf{r}_1$. Having computed the components $\mathbf{t}_1, \ldots, \mathbf{t}_h$, compute the deflated design matrix

$$\mathbf{X}_h = \mathbf{X}_{h-1} - \mathbf{t}_h(\mathbf{t}_h^\top \mathbf{X}_{h-1})/(\mathbf{t}_h^\top \mathbf{t}_h).$$

Then, for each response variable, compute the residuals from a GAM fit

$$\mathbf{Y}_h = \mathbf{Y}_0 - \sum_{i=1}^{h} \left(g_{i1}(\mathbf{t}_i), \dots, g_{iq}(\mathbf{t}_i)\right),$$

where $g_{1j}, \dots, g_{hj}$ is the result of a GAM fit of the jth variable (column) in $\mathbf{Y}_0$ on $\mathbf{t}_1, \dots, \mathbf{t}_h$. The next orthogonal component is $\mathbf{t}_{h+1} = \mathbf{X}_h \mathbf{r}_{h+1}$, where $\mathbf{r}_{h+1}$ is the solution for $\mathbf{r}$ in the optimization

$$\sup_{\|\mathbf{r}\|=\|\mathbf{s}\|=1} D(\mathbf{X}_h\mathbf{r}, \mathbf{Y}_h\mathbf{s}). \tag{8}$$

When D is the covariance and ordinary least squares is the smoother in the GAM fit, `groc` reproduces the `plsr` model estimated by the SIMPLS algorithm of de Jong (1993). A justification in terms of the optimization of the objective function (8) is given in Boulesteix and Strimmer (2007). The optimization (8) is done using the grid Algorithm 1, in which at every cycle, a search over $\mathbf{r}$, with $\mathbf{s}$ fixed, is followed by a search over $\mathbf{s}$, with $\mathbf{r}$ fixed, alternating this way until convergence in a similar way as in the nonlinear iterative partial least squares (NIPALS) algorithm of Wold (1975).

Example 8. The data `oliveoil` of the **pls** package contains scores on 6 attributes from a sensory panel and measurements of 5 physico-chemical quality parameters on 16 olive oil samples. The first five oils are Greek, the next five are Italian and the last six are Spanish. A comparison of `plsr` with the option `method = "simpls"` and `groc` with the options `D = covariance` and `method = "lm"` gives relative differences less than 0.009% between predicted values for all observations, response variables and number of components. The PRESS values from leave-one-out cross-validation for each response and any number of components are given next.

```
R> data("oliveoil", package = "pls")
R> n <- nrow(oliveoil)
R> plsr.out <- plsr(sensory ~ chemical, data = oliveoil, method = "simpls")
R> groc.out <- groc(sensory ~ chemical, data = oliveoil)
R> groc.v <- grocCrossval(groc.out, segments = n)
R> groc.v$validation$PRESS

       1 comps 2 comps 3 comps 4 comps 5 comps
yellow  5755.1 4145.28 4469.99  5245.2  7551.9
green   9122.9 6691.10 7294.80  9185.4 12889.7
brown    258.4  254.29  254.33   269.9   328.8
glossy   417.6  426.14  496.51   664.9   780.9
transp   842.9  819.90  940.01  1237.5  1404.2
syrup     72.9   86.49   98.23   138.2   150.4
```

Averaging the PRESS values over responses,

```
R> colMeans(groc.v$validation$PRESS)

1 comps 2 comps 3 comps 4 comps 5 comps
   2745    2071    2259    2790    3851
```

a model with two components is suggested. The correlations between response and predicted value by the model with two components

```
R> Y <- oliveoil$sensory
R> for (j in 1:ncol(Y)) print(cor(Y[, j], fitted(groc.out)[, j, 2]))

[1] 0.6855
[1] 0.6618
[1] 0.7952
[1] 0.7174
[1] 0.6757
[1] 0.7668
```

vary approximately between 0.7 and 0.8.

7. Robust multivariate partial least squares regression

Multivariate partial least squares regression by the SIMPLS algorithm can be computed by `groc` with the options `D = covariance` and `method = "lm"`. Although, not as fast as `plsr` with the option `method = "simpls"`, `groc` is flexible and can be adapted to produce a robust fit able to withstand vertical outliers and bad leverage points. For this purpose, a robust covariance using the robust τ scale estimate of Yohai and Zamar (1988) is defined as suggested by Gnanadesikan and Kettenring (1972)

$$covrob(\mathbf{t},\mathbf{u}) = \frac{1}{4}\left[\tau(\mathbf{t}+\mathbf{u})^2 - \tau(\mathbf{t}-\mathbf{u})^2\right].$$

In Algorithm 2, the GAM fit

$$\hat{\mathbf{y}}_0 = g_1(\mathbf{t}_1) + \cdots + g_i(\mathbf{t}_i)$$

is replaced by a least trimmed squares regression. The deflation of $\mathbf{X}$ by ordinary least squares,

$$\begin{aligned} \mathbf{b}_i &= \mathbf{X}^\top\mathbf{t}_i/(\mathbf{t}_i^\top\mathbf{t}_i) \\ \mathbf{X} &\leftarrow \mathbf{X} - \mathbf{t}_i\mathbf{b}_i^\top, \end{aligned}$$

is also robustified by least trimmed squares (`lts`). By default, all the least trimmed squares regressions use a breakdown point of 0.25 for a good compromise between robustness and efficiency. No multivariate implementation of least trimmed squares is available. Hence, the coefficients $\mathbf{b}_i$ are replaced by the coefficients of `lts` regressions computed for each predictor separately. The deflation of $\mathbf{X}$ finally computes the residuals of these `lts` regressions. The `groc` function can be used with the option `plsrob = TRUE`.

The flexibility of the package **groc** allows to fit robust partial least squares regression by robustifying each step of Algorithm 2.

Example 9. The data `cookie` described in Osborne, Fearn, Miller, and Douglas (1984) contains measurements from quantitative NIR spectroscopy. It is part of the **ppls** package (Krämer, Boulesteix, and Tutz 2008; Kraemer and Boulesteix 2014) which fits only models

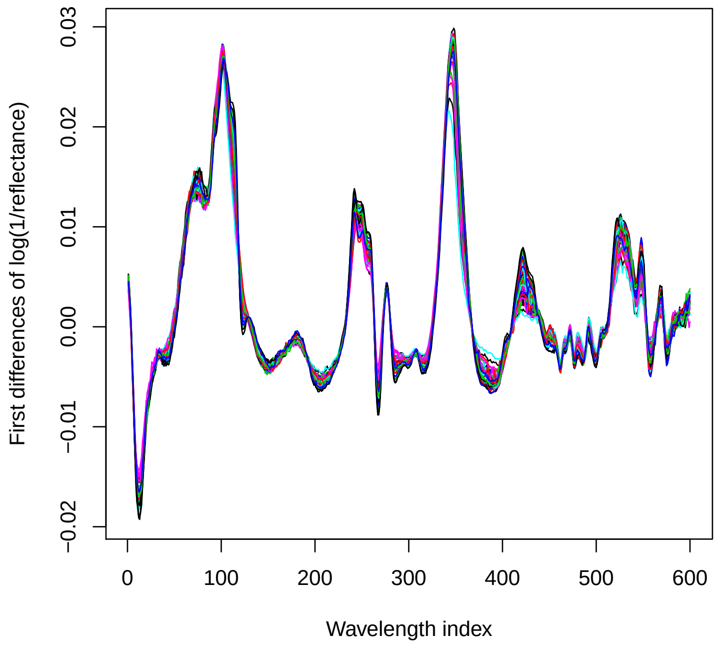

Figure 4: The biscuit NIR preprocessed reflectance spectrum.

with a single response. The data arise from an experiment done to test the feasibility of NIR spectroscopy to measure the composition of biscuit dough pieces (formed but unbaked biscuits). A sample set of size 40 was made up, with the standard recipe varied to provide a large range for each of the four constituents under investigation: fat, sucrose, dry flour, and water. The calculated percentages of these four ingredients represent the 4 responses. Observation 23 was identified as an outlier by Osborne *et al.* (1984). An NIR reflectance spectrum is available for each dough piece. The spectral data consist of 700 points measured from 1100 to 2498 nanometers (nm) in steps of 2 nm. The first and last 50 points of the spectrum were discarded because of lower instrumental reliability. Then, the data were transformed as in Hubert, Rousseeuw, and Verboven (2002). Firstly, a logarithmic transformation of NIR reflectance spectrum was applied to eliminate drift and background scatter. Secondly, we used first differences to remove constants and sudden shifts. After this preprocessing, we ended up with a design matrix with 40 observations and 600 predictors. Figure 4 shows the preprocessed reflectance spectrum. The number of components using `plsr` was found with the leave-one-out cross-validation PRESS criterion averaged over all 4 responses. The plot in Figure 5 suggests a model with 3 components. The correlations between observed response and fitted value are very large for the 4 responses: 0.9787, 0.9335, 0.9185 and 0.9426. Outlier detection was done using 3 components with `plsr` and `groc` with the `plsrob = TRUE` option. For `groc`, robust Mahalanobis distances were computed in the space of components

$$\left[\sum_{i=1}^{h}\left(\frac{t_{ji}-\hat{\mu}_i}{\hat{\tau}_i}\right)^2\right]^{1/2},\ j=1,\ldots,n,$$

where $\hat{\mu}_i$ and $\hat{\tau}_i$ are the location and scale τ estimates of Yohai and Zamar (1988) from the component $\mathbf{t}_i$ with components $t_{1i},\ldots,t_{ni}$. Since components are orthogonal, Mahalanobis

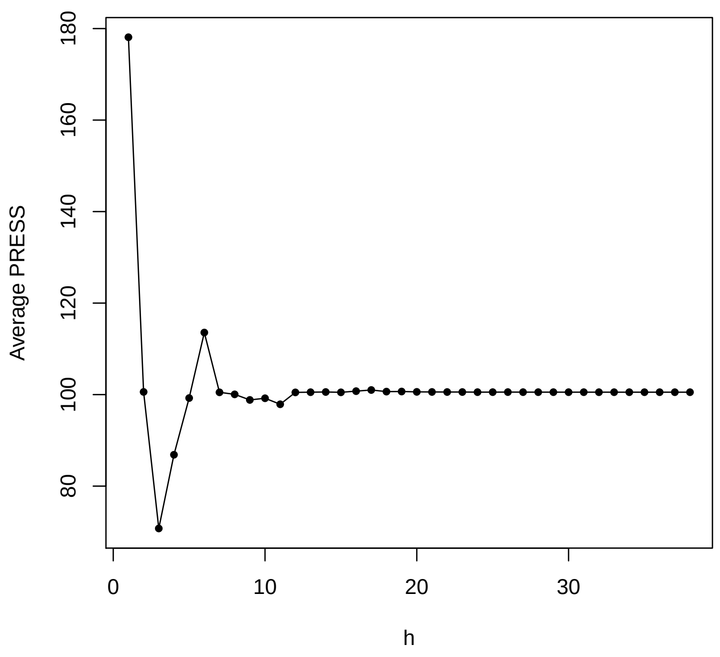

Figure 5: Average PRESS values over all responses as a function of the number of components.

distances reduce to simple sums. Robust Mahalanobis distances were also computed in the space of residuals

$$\left[(\hat{\mathbf{e}}_j - \hat{\mu})^\top \hat{\Sigma}^{-1} (\hat{\mathbf{e}}_j - \hat{\mu})\right]^{1/2}, \; j = 1, \ldots, n,$$

where $\hat{\mu}$ and $\hat{\Sigma}$ are the robust location vector and scatter matrix of the orthogonalized Gnanadesikan-Kettenring pairwise estimator (Maronna and Zamar 2002) computed from the residuals. For `plsr`, we used classical Mahalanobis distances with mean location, standard deviation scale, and sample covariance matrix.

Critical threshold for squared Mahalanobis distances were determined as the 0.975 quantile of a chi-squared distribution with degrees of freedom equal to the dimension of the corresponding space. The robust correlations evaluated by `corrob` between each response and its corresponding fitted value by `groc` (0.961, 0.9618, 0.9839 and 0.9888) are now even stronger. Figure 6 reveals a bad leverage point (observation 23) and two other influential points (observations 7 and 21). These 3 points are much more singled out by the robust fit which also identified other points with large residuals. The robust distances on the right panel of Figure 6 may differ slightly for different runs of the program since `lts` does an approximation by an exhaustive enumeration up to 5000 samples with the default option `nsamp = "best"`.

Another method called partial robust M-regression (`prm`) in Serneels *et al.* (2005) is faster but not as robust as `groc`. A MATLAB routine for models with a single response is implemented in (Daszykowskia *et al.* 2013). Two reasons for the lack of robustness of `prm` are:

1. Their weight function for residuals in models with a single response is

$$w_i^r = f\left(\frac{\hat{e}_i}{\hat{\sigma}}, c\right),$$

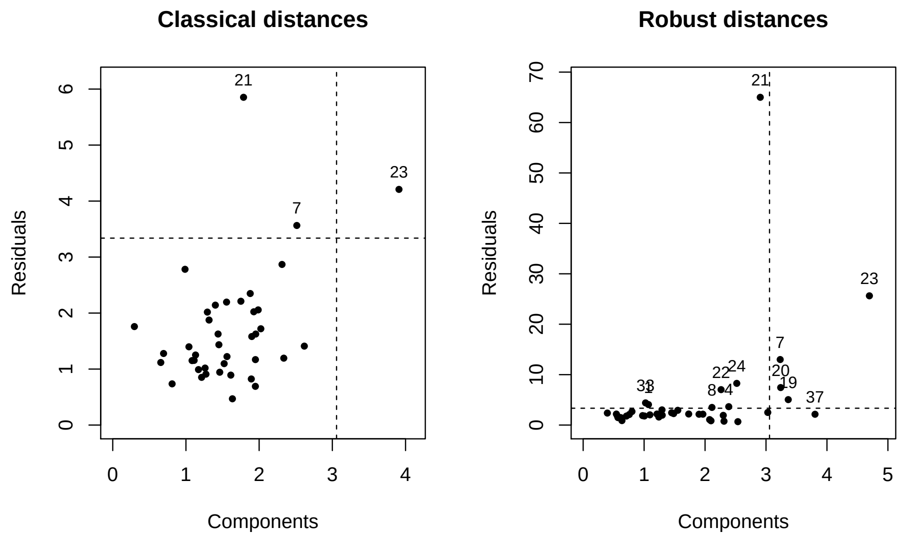

Figure 6: Diagnostic plots for the data `cookie` in Example 9.

where

$$f(z,c) = \left(1 + \left|\frac{z}{c}\right|\right)^{-2}$$

and c is a tuning constant, taken as $c = 4$. The estimate $\hat{\sigma}$ is the robust scale estimate `qn` of Rousseeuw and Croux (1993). Therefore, the weights are never zero even for points with very large standardized residuals. The same remark holds for the weights measuring the leverage in the space of components.

2. But most importantly, in the first step of their routine, they define residual weights from the residuals taken as $y_i - med_j y_j$ and leverage weights using

$$w_i^x = f\left(\frac{\|\mathbf{x}_i - med_{L_1}(\mathbf{X})\|}{med_i\|\mathbf{x}_i - med_{L_1}(\mathbf{X})\|}, c\right)$$

from the original cases $\mathbf{x}_i$. The notation med_{L_1} stands for the L_1 median also called the spatial median. Incorrect identification of outliers and leverage points in the first step of their routine may lead to a local minimum corresponding to a nonrobust solution. This happens for data with vertical outliers which are neither outlying in the space of y_i's, nor in the space of $\mathbf{x}_i$'s.

An artificial dataset best illustrates this caveat.

Example 10. Consider the data with $n = 30$ generated from the model $y = t_1 + \epsilon$, where t_1 is uniform on the interval $(-1, 1)$ and ϵ is normal with a standard deviation of 0.05. Vertical outliers are induced at observations 14, 15, and 16 by adding the constant 0.5 to the y values. Next, the `prm` and `groc` methods are applied to an augmented dataset containing the response $\mathbf{y}$, the predictor $\mathbf{t}_1$, and two perfectly collinear predictors $2\mathbf{t}_1$ and $-1.5\mathbf{t}_1$. Figure 7 exhibits the data and the fitted values. The lack of robustness of `prm` is evident from its fitted line which is attracted by the vertical outliers.

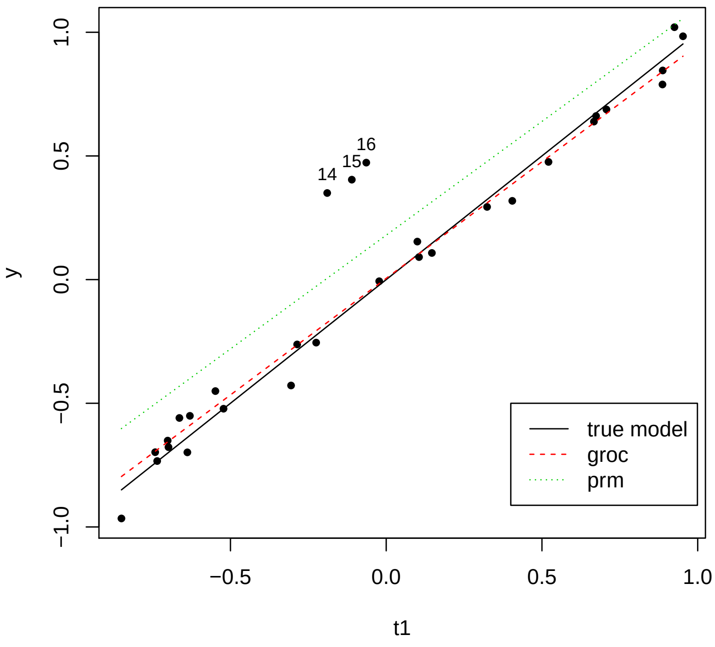

Figure 7: Plot of the artificial data of Example 10.

When the design matrix $\mathbf{X}$ is of full column rank p, a multivariate least squares regression is equivalent to a partial least squares regression by the SIMPLS algorithm with p components. Hence, `groc` can also be used for robust multivariate least squares regression. This is now illustrated with a last example on a real data related to pulp fiber.

Example 11. Cited from Rousseeuw, Aelst, Driessen, and Gulló (2004): "The dataset contains measurements of properties of pulp fibers and the paper made from them. The aim is to investigate relations between pulp fiber properties and the resulting paper properties. The dataset contains $n = 62$ measurements of the following four pulp fiber characteristics: arithmetic fiber length, long fiber fraction, fine fiber fraction, and zero span tensile. The four paper properties that have been measured are breaking length, elastic modulus, stress at failure, and burst strength." As in the previous example, a robust multivariate linear regression is fitted by `groc` with the option `plsrob = TRUE` with four components. Identification of outliers and leverage points is done again with robust Mahalanobis distances in both spaces: components and residuals. The cutoff point is the same in both spaces since here $p = q = 4$, and is taken as the square root of the 0.975 quantile of the chi-squared distribution with four degrees of freedom as in Rousseeuw *et al.* (2004). Our result is given in Figure 8 which is very close to their Figure 3. Citing again from Rousseeuw *et al.* (2004): "By exploring the origin of the collected data we found out that all but the last four pulp samples (observations 59–62) were produced from fir wood." and "For example, observation 62 is the only sample from a chemi-thermomechanical pulping process, observations 60 and 61 are the only samples from a solvent pulping process, and observations 51, 52 and 56 are obtained from a kraft pulping process." The bad leverage points, especially observations 60 and 61, exert an undue influence on the classical SIMPLS fit which gives small residuals at these points on the left panel of Figure 8.

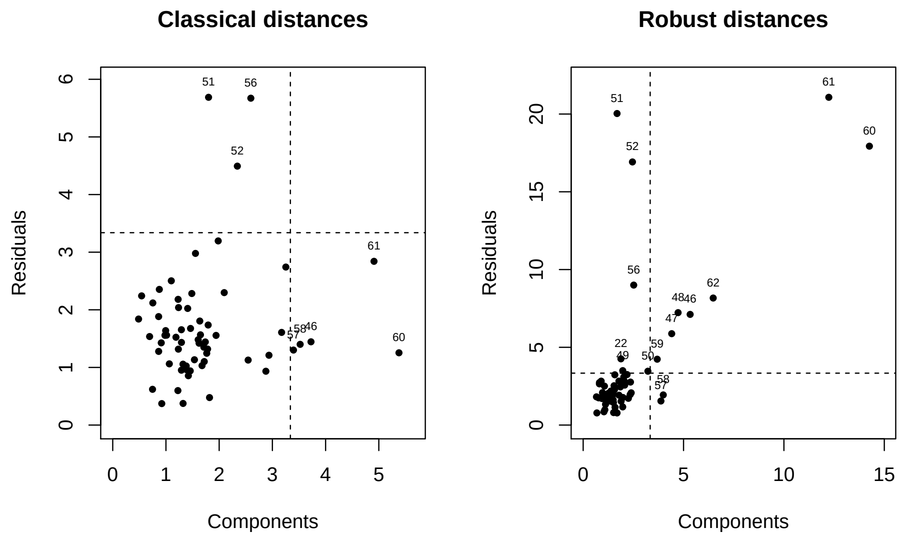

Figure 8: Diagnostic plots for the data `pulpfiber` in Example 11.

This robust fit is now compared to the classical SIMPLS fit with four components by ten-fold cross-validation. In presence of outliers, meaningful comparisons are made on the basis of their PREMAD values which are

```
Segment: 1 2 3 4 5 6 7 8 9 10

    Y1     Y2     Y3     Y4
0.7509 0.3037 0.3907 0.1759
```

for the robust fit, and

```
    Y1     Y2     Y3     Y4
0.9562 0.3188 0.4243 0.2289
```

for classical SIMPLS. Relative reductions in PREMAD values of the robust fit over the classical SIMPLS fit

```
     Y1       Y2       Y3      Y4
-21.470  -4.747  -7.911 -23.143
```

vary approximately between 5% and 20% for all four responses.

8. Conclusion

When the true model between responses and predictors is nonlinear, **groc** naturally outperforms **pls** (partial least squares). It can even perform better than `ppr` (projection pursuit regression) because the orthogonality constraints between components of package **groc** yields a reduction in variability of predictions, as compared to `ppr`, which can largely compensate for the increased squared biais. When the number of predictors is large, package **groc** can also outperform `ppr` in terms of computer user time.

When the true model between responses and predictors is linear, package **groc** is more robust than package **pls** and `prm` (partial robust M-regression). It is thus better suited for the identification of outliers. It could be used in conjunction with package **pls** by identifying outliers with package **groc**, and performing the final analysis on the good data with package **pls**.

Acknowledgments

The research of the first and second authors has been funded by the National Sciences and Engineering Reasearch Council of Canada.

References

Bilodeau M, Lafaye de Micheaux P (2015). ***groc**: Generalized Regression on Orthogonal Components*. R package version 1.0.5, URL `http://CRAN.R-project.org/package=groc`.

Boulesteix AL, Strimmer K (2007). "Partial Least Squares: A Versatile Tool for the Analysis of High-Dimensional Genomic Data." *Briefings in Bioinformatics*, **8**(1), 32–44.

Buja A, Hastie T, Tibshirani R (1989). "Linear Smoothers and Additive Models." *The Annals of Statistics*, **17**(2), 453–510.

Croux C, Filzmoser P, Oliveira M (2007). "Algorithms for Projection-Pursuit Robust Principal Component Analysis." *Chemometrics and Intelligent Laboratory Systems*, **87**(2), 218–225.

Daszykowskia M, Serneels S, Kaczmarek K, Espen PV, Croux C, Walczak B (2007). "**TOMCAT**: A MATLAB Toolbox for Multivariate Calibration Techniques." *Chemometrics and Intelligent Laboratory Systems*, **85**(2), 269–277.

Daszykowskia M, Serneels S, Walczak B, Espen PV, Croux C (2013). ***TOMCAT**: A MATLAB Toolbox for Multivariate Calibration Techniques.* URL `http://chemometria.us.edu.pl/RobustToolbox/`.

de Jong S (1993). "SIMPLS: An Alternative Approach to Partial Least Squares Regression." *Chemometrics and Intelligent Laboratory Systems*, **18**(3), 251–263.

Denham MC (1995). "Implementing Partial Least Squares." *Statistics and Computing*, **5**(3), 191–202.

Draper NR, Smith H (1966). *Applied Regression Analysis.* John Wiley & Sons, New York.

Filzmoser P, Fritz H, Kalcher K (2014). ***pcaPP**: Robust PCA by Projection Pursuit.* R package version 1.9-60, URL `http://CRAN.R-project.org/package=pcaPP`.

Friedman JH, Stuetzle W (1981). "Projection Pursuit Regression." *Journal of the American Statistical Association*, **76**(376), 817–823.

Genest C, Quessy JF, Rémillard B (2006). "Local Efficiency of a Cramér-von Mises Test of Independence." *Journal of Multivariate Analysis*, **97**(1), 274–294.

Gnanadesikan R, Kettenring JR (1972). "Robust Estimates, Residuals, and Outlier Detection with Multiresponse Data." *Biometrics*, **28**(1), 81–124.

Höskuldsson A (1998). "PLS Regression Methods." *Journal of Chemometrics*, **2**(3), 211–228.

Hubert M, Rousseeuw PJ, Verboven S (2002). "A Fast Method for Robust Principal Components with Applications to Chemometrics." *Chemometrics and Intelligent Laboratory Systems*, **60**(1–2), 101–111.

Kraemer N, Boulesteix AL (2014). ***ppls**: Penalized Partial Least Squares.* R package version 1.6-1, URL http://CRAN.R-project.org/package=ppls.

Krämer N, Boulesteix AL, Tutz G (2008). "Penalized Partial Least Squares with Applications to B-Spline Transformations and Functional Data." *Chemometrics and Intelligent Laboratory Systems*, **94**(1), 60–69.

Maronna RA, Zamar RH (2002). "Robust Estimates of Location and Dispersion for High-Dimensional Datasets." *Technometrics*, **44**(4), 307–317.

Mevik BH, Wehrens R, Liland KH (2013). ***pls**: Partial Least Squares and Principal Component Regression.* R package version 2.4-3, URL http://CRAN.R-project.org/package=pls.

Nguyen TS, Rojo J (2009). "Dimension Reduction of Microarray Gene Expression Data: The Accelerated Failure Time Model." *Journal of Bioinformatics and Computational Biology*, **7**(6), 939–954.

Osborne BG, Fearn T, Miller AR, Douglas S (1984). "Application of Near Infrared Reflectance Spectroscopy to the Compositional Analysis of Biscuits and Biscuit Doughs." *Journal of the Science of Food and Agriculture*, **35**(1), 99–105.

R Core Team (2015). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/.

Rosenbrock HH (1960). "An Automatic Method for Finding the Greatest or Least Value of a Function." *The Computer Journal*, **3**(3), 175–184.

Rosipal R, Trejo LJ (2001). "Kernel Partial Least Squares Regression in Reproducing Kernel Hilbert Space." *Journal of Machine Learning Research*, **2**(December), 97–123.

Rousseeuw P, Croux C, Todorov V, Ruckstuhl A, Salibian-Barrera M, Verbeke T, Koller M, Maechler M (2015). ***robustbase**: Basic Robust Statistics.* R package version 0.92-3, URL http://CRAN.R-project.org/package=robustbase.

Rousseeuw PJ, Aelst SV, Driessen KV, Gulló JA (2004). "Robust Multivariate Regression." *Technometrics*, **46**(3), 293–305.

Rousseeuw PJ, Croux C (1993). "Alternatives to the Median Absolute Deviation." *Journal of the American Statistical Association*, **88**(424), 1273–1283.

Rousseeuw PJ, Leroy AM (1987). *Robust Regression and Outlier Detection*. Wiley Series in Probability and Mathematical Statistics: Applied Probability and Statistics. John Wiley & Sons, New York.

Serneels S, Croux C, Filzmoser P, Espen PJV (2005). "Partial Robust M-Regression." *Chemometrics and Intelligent Laboratory Systems*, **79**(1–2), 55–64.

Székely GJ, Rizzo ML, Bakirov NK (2007). "Measuring and Testing Dependence by Correlation of Distances." *The Annals of Statistics*, **35**(6), 2769–2794.

Tenenhaus M (1998). *La Régression PLS: Théorie et Pratique*. Éditions Technip.

The MathWorks Inc (2014). *MATLAB – The Language of Technical Computing, Version R2014b*. Natick, Massachusetts. URL http://www.mathworks.com/products/matlab/.

Todorov V, Filzmoser P (2009). "An Object-Oriented Framework for Robust Multivariate Analysis." *Journal of Statistical Software*, **32**(3), 1–47. URL http://www.jstatsoft.org/v32/i03/.

Venables WN, Ripley BD (2002). *Modern Applied Statistics with S*. 4th edition. Springer-Verlag, New York. URL http://www.stats.ox.ac.uk/pub/MASS4/.

Wang J, Zamar R, Marazzi A, Yohai V, Salibian-Barrera M, Maronna R, Zivot E, Rocke D, Martin D, Maechler M, Konis K (2014). ***robust**: Robust Library*. R package version 0.4-16, URL http://CRAN.R-project.org/package=robust.

Wold H (1975). "Path Models with Latent Variables: The NIPALS Approach." In HM Blalock, A Aganbegian, FM Borodkin, R Boudon, V Capecchi (eds.), *Quantitative Sociology: International Perspectives on Mathematical and Statistical Modeling*, pp. 307–357. Academic Press, New York.

Wood SN (2006). *Generalized Additive Models: An Introduction with R*. Chapman & Hall/CRC.

Wood SN (2015). ***mgcv**: Mixed GAM Computation Vehicle with GCV/AIC/REML Smoothness Estimation*. R package version 1.8-5, URL http://CRAN.R-project.org/package=mgcv.

Yohai VJ, Zamar RH (1988). "High Breakdown-Point Estimates of Regression by Means of the Minimization of an Efficient Scale." *Journal of the American Statistical Association*, **83**(402), 406–413.

Affiliation:

Pierre Lafaye de Micheaux
Département de mathématiques et de statistique
Université de Montréal

C.P. 6128, succursale centre-ville
Montréal, Québec H3C 3J7, Canada
E-mail: lafaye@dms.umontreal.ca
URL: http://www.biostatisticien.eu/

Journal of Statistical Software http://www.jstatsoft.org/
published by the American Statistical Association http://www.amstat.org/

Volume 65, Issue 1 *Submitted:* 2013-07-13
May 2015 *Accepted:* 2014-10-13