An Algorithm for Nonlinear, Nonparametric Model Choice and Prediction

Frédéric Ferraty, Peter Hall

An Algorithm for Nonlinear, Nonparametric Model Choice and Prediction

Frédéric FERRATY Peter HALL

We introduce an algorithm which, in the context of nonlinear regression on vector-valued explanatory variables, aims to choose those combinations of vector components that provide best prediction. The algorithm is constructed specifically so that it devotes attention to components that might be of relatively little predictive value by themselves, and so might be ignored by more conventional methodology for model choice, but which, in combination with other difficult-to-find components, can be particularly beneficial for prediction. The design of the algorithm is also motivated by a desire to choose vector components that become redundant once appropriate combinations of other, more relevant components are selected. Our theoretical arguments show these goals are met in the sense that, with probability converging to 1 as sample size increases, the algorithm correctly determines a small, fixed number of variables on which the regression mean, g say, depends, even if dimension diverges to infinity much faster than n. Moreover, the estimated regression mean based on those variables approximates g with an error that, to first order, equals the error which would arise if we were told in advance the correct variables. In this sense the estimator achieves oracle performance. Our numerical work indicates that the algorithm is suitable for very high dimensional problems, where it keeps computational labour in check by using a novel sequential argument, and also for more conventional prediction problems, where dimension is relatively low.

Key Words: Feature and variable selection; combinations of variables; nonparametric regression; sequential algorithm

Frédéric Ferraty is Professor, Toulouse Mathematics Insitute, University of Toulouse, 31062 Toulouse, France (E-mail: frederic.ferraty@math.univ-toulouse.fr). Peter Hall is Professor, Department of Mathematics and Statistics, University of Melbourne, Parkville 3010, Australia (E-mail: halpstat@ms.unimelb.edu.au)

1. INTRODUCTION

For more than 30 years statisticians have sought to identify the relevant vector components in relatively high-dimensional prediction problems. Today, in the case of data from fields such as genomics, astronomy and consumer preference modeling, the challenges are greater than in the past, with the ratio of dimension to sample size often being higher than every before. In the present paper we suggest a new, highly adaptive algorithm that can be used to build predictive models in both contemporary and classical settings. Our approach is designed specifically for cases where the response is a nonlinear function of the predictors, and where we wish to be economical in our choice of variables.

Particularly in cases where dimension is greater than sample size, a great deal of attention has been devoted in the last 15 years to model choice in the framework of linear models. In this setting, Tibshirani's (1996) lasso was the starting point for the development of many techniques: coordinate descent methods (Fu, 1998, Friedman *et al.*, 2007), smoothly clipped absolute deviation (Fan and Li, 2001), least angle regression (Efron *et al.*, 2004), the elastic net (Zou and Hastie, 2005), the adaptive lasso (Zou, 2006), the Dantzig selector (Candès and Tao, 2007), the relaxed lasso (Meinshausen, 2007), the group lasso (Yuan and Lin, 2008), and the multi-step adaptative lasso (Bühlmann and Meier, 2008). Overviews of this work have been provided by Hastie *et al.* (2009), Fan and Lv (2010) and Bülhmann and van de Geer (2011).

These variable selection tools have been applied successfully to various high-dimensional datasets, but their effectiveness can be hindered by the assumption of a linear relationship between response and covariates. One problem is that the high-dimensional setting makes it difficult to validate the existence of the linear relationship. Moreover, it is common to encounter nonlinear structure even in standard, relatively low-dimensional multivariate regression models, and there is no a priori reason why such structure should not occur in high-dimensional cases.

However, it can be very challenging to investigate nonlinear relationships when there are many variables. There exists a literature on additive modeling, which often is treated as an extension of the lasso by combining the group lasso with basis expansion of each one-dimensional additive component. See, for example, the work of Meier *et al.* (2009),

Ravikumar *et al.* (2009) and Huang *et al.* (2010).

Ferraty *et al.* (2010) endeavoured to go beyond these techniques by developing methodology that captures interactions, using a stepwise forward search algorithm founded on minimizing a cross-validation criterion. However, although this approach enjoys good performance in many cases, it fails in a worrying number of settings, where small submodels are not detected.

The new algorithm suggested in this paper is based on enlarging the class of possible combinations of covariates retained at each step, while keeping the run time within reasonable bounds. This is an important issue from a practical viewpoint. Our methodology is given in Section 2, where our approach to building and selecting submodels is discussed first in overview and then described in detail. The technique is illustrated in Section 3 by application to a real genomics dataset, and in Section 4 in a simulation study. Theoretical issues are treated in Section 5.

2. METHODOLOGY

2.1 Measuring mean squared variation.

Given independent and identically distributed data pairs (X_i, Y_i) for $i \in \mathcal{S} = \{1, \ldots, n\}$, where $X_i = (X_{i1}, \ldots, X_{ip})$ is a p-vector and Y_i is a scalar, we wish to choose a small number of vector components, or variables or features, of X_i on which to regress Y_i, with the aim of predicting a future Y for a given $x = (x_1, \ldots, x_p)$.

Our methodology is built around an algorithm, discussed in Section 2.2 and defined concisely in Section 2.3, for determining the extent to which a given subset, $X_{ij_1}, \ldots, X_{ij_\ell}$ say, of the components of X_i successfully predict Y_i. Each step of the algorithm involves using our favorite nonparametric function estimator, for example a local linear approach or a spline, to construct a predictor $\hat{\gamma}_{j_1,\ldots,j_\ell}(x_{j_1}, \ldots, x_{j_\ell})$ of Y from the dataset $\{(X_{ij_1}, \ldots, X_{ij_\ell}, Y_i);\ i \in \mathcal{S}\}$, where $(x_{j_1}, \ldots, x_{j_\ell})$ is a subvector of x. Then compute the standard cross-validation criterion,

$$S(j_1, \ldots, j_\ell) = \sum_{i=1}^{n} \{Y_i - \hat{\gamma}^{-i}_{j_1,\ldots,j_\ell}(X_{ij_1}, \ldots, X_{ij_\ell})\}^2 \, w_\ell(X_{ij_1}, \ldots, X_{ij_\ell}) \,, \tag{2.1}$$

which measures the success of $\hat{\gamma}^{-i}_{j_1,\ldots,j_\ell}(X_{ij_1},\ldots,X_{ij_\ell})$ in predicting Y_i where $\hat{\gamma}^{-i}_{j_1,\ldots,j_\ell}$ is the leave-one-out estimator derived from $\mathcal{S}\backslash\{i\}$. In our theoretical study, the function w_ℓ in (2.1) is taken to be nonnegative, whereas in our implementation, w_l is set to 1.

In order to simplify notation, let $\mathcal{J} = \{j_1,\ldots,j_\ell\}$ be a subset of $\{1,\ldots,p\}$ so that, for any p-dimensional vector $u = (u_1,\ldots,u_p)$ of $\mathbb{R}^p$, $u^{\mathcal{J}}$ stands for the subvector $(u_{j_1},\ldots,u_{j_\ell})$. Then, (2.1) may be written in an equivalent way as

$$S(\mathcal{J}) = \sum_{i\in\mathcal{T}} \{Y_i - \hat{\gamma}^{-i}_{\mathcal{J}}(X_i^{\mathcal{J}})\}^2\, w_{|\mathcal{J}|}(X_i^{\mathcal{J}})\,, \tag{2.2}$$

where $|\mathcal{J}|$ is the size of $\mathcal{J}$.

If $\mathcal{J}_1,\ldots,\mathcal{J}_k$ are distinct subsets of indices then the permutation of $\mathcal{J}_1,\ldots,\mathcal{J}_k$ that is used in each of the steps in Section 2.3, for different values of k, is that which places the values of $S(\mathcal{J}_1),\ldots,S(\mathcal{J}_k)$ in increasing order. In the subsequent step of the algorithm we merge $\mathcal{J}_1,\ldots,\mathcal{J}_k$ in a pairwise manner, creating new subsets of indices $\mathcal{J}_1\cup\mathcal{J}_2$, ..., $\mathcal{J}_1\cup\mathcal{J}_k$, ..., $\mathcal{J}_{k-1}\cup\mathcal{J}_k$ that are rearranged again to rank the corresponding predictive values; and we repeat this process until we obtain a subset $\mathcal{J}$ with a sufficiently small value of $S(\mathcal{J})$.

2.2 Overview of algorithm.

The first step of the algorithm involves searching over all single subsets $\{1\},\ldots,\{p\}$, the next over all combinations $\{j,j'\}$, the third over all combinations of the previous ones (i.e. $\{j_1,j_1'\}\cup\{j_2,j_2'\}$), and so on. Normally this would be prohibitively expensive from a computational viewpoint. Indeed, in many problems doing even the $O(p^2)$ search over pairs of indices would be out of the question. However, we use the following "trick" to reduce labour. Having searched over single subsets and ranked the variables there, we look only at the top $\sqrt{p}$ variables when constructing the sets $\{j,j'\}$ over which we search in the next step. There are only $O(\sqrt{p}^2) = O(p)$ subsets of indices constructed in this way, and so the search over sets $\{j,j'\}$ is not much more onerous than it was in the case of the single subsets.

In Section 2.3 we note that $O(p)$ may not, in general, be a good description of the upper bound to the capability of our computational resources. Instead we take $O(q)$ to

be that bound, where q might be larger than p if our resources are relatively extensive, or less than p if the inherent multiplier of a power of n, which for simplicity we omitted from the arguments above, is problematic. In this case our algorithm "sniffs out" the trace of potentially significant variables among the first $\sqrt{q}$ variables when building bivariate predictors, and subsequently also when constructing predictors of higher order. For now, however, we assume that $q = p$.

It should be stressed that the steps in our algorithm rely on the variables that are "useful" for prediction making themselves known, to at least some extent, when we are experimenting with prediction based on a single variable. Experimentation is described in Step 1 in Section 2.3. Variables that are useful for building higher-order predictors do not have to be present in the top few of the p variables, but some of them should be apparent with sufficient strength to lie among the top $\sqrt{p}$ variables. It is difficult to see how this constraint can be removed without using a relatively a crude, model-based approach to variable selection. The advantage of our alternative approach is that, if a variable shows itself to be just slightly useful for prediction in isolation, in particular if it lies among the top $\sqrt{p}$ variables, then we have an opportunity to detect its importance even if its main contributions are felt only when it operates in conjunction with one or more other variables. In contrast, conventional approaches to feature selection, based on linear models, can completely overlook variables that have a major impact only through interaction with one or more other variables.

2.3 Details of algorithm.

Step 1: Prediction based on a single variable. Consider the p singletons $\mathcal{J}_1 = \{1\}, \ldots, \mathcal{J}_p = \{p\}$, and compute the permutation $\hat{\jmath}_1(1), \ldots, \hat{\jmath}_1(p)$ of the indices $1, \ldots, p$ that represents the ranking $S\{\mathcal{J}^1(1)\} \leq \ldots \leq S\{\mathcal{J}^1(p)\}$, with $\mathcal{J}^1(k) = \mathcal{J}_{\hat{\jmath}_1(k)}$ for $1 \leq k \leq p$ and where S is defined as at (2.1). If $\hat{\jmath}_1(k_1) < \hat{\jmath}_1(k_2)$ then $X_{i\hat{\jmath}_1(k_1)}$ better explains Y_i, in a particular sense, than does $X_{i\hat{\jmath}_1(k_2)}$. In this sense, a regression of Y on the $\hat{\jmath}_1(1)$th component of X produces the "best" predictor based on a single variable.

Step 2: Prediction based on two variables. Assume that our computing resources are limited to $O(q)$ calculations, multiplied by a low power of n, and put $p_1 = \sqrt{q}$. From the top p_1 subsets $\mathcal{J}^1(1), \ldots, \mathcal{J}^1(p_1)$, build the set of all $p_2^* = \frac{1}{2}\, p_1\, (p_1 - 1) = O(q)$

pairs $\mathcal{J}_1 = \mathcal{J}^1(1) \cup \mathcal{J}^1(2), \ldots, \mathcal{J}_{p_1-1} = \mathcal{J}^1(1) \cup \mathcal{J}^1(p_1), \mathcal{J}_{p_1} = \mathcal{J}^1(2) \cup \mathcal{J}^1(3), \ldots, \mathcal{J}_{p_2^*} = \mathcal{J}^1(p_1 - 1) \cup \mathcal{J}^1(p_1)$. Then, compute the permutation $\hat{j}_2(1), \ldots, \hat{j}_2(p_2^*)$ of the indices $1, \ldots, p_2^*$ that places the values $S(\mathcal{J}_{\hat{j}_2(k)})$, for $1 \le k \le p_2^*$, in increasing order, and retain for the next step only the $p_2 = p_1 = \sqrt{q}$ top subsets $\mathcal{J}^2(1) = \mathcal{J}_{\hat{j}_2(1)}, \ldots, \mathcal{J}^2(p_2) = \mathcal{J}_{\hat{j}_2(p_2)}$. A regression of Y on $X^{\mathcal{J}^2(1)}$ provides the "best" predictor based on just two variables.

Steps 3,4,...: Prediction based on $\ell \ge 3$ variables. In step 1, or respectively step 2, the procedure builds only singletons, or respectively pairs. However, in step $\ell \ge 3$ the algorithm may generate subsets $\mathcal{J}$ of indices such that $\ell \le |\mathcal{J}| \le 2^{\ell-1}$. For instance, if we consider the sets $\mathcal{J}^2(1) = \{j_1, j_2\}$, $\mathcal{J}^2(2) = \{j_1, j_3\}$, $\mathcal{J}^2(3) = \{j_2, j_4\}$, ..., the third step of our algorithm will build a new family of subsets containing $\mathcal{J}^2(1) \cup \mathcal{J}^2(2) = \{j_1, j_2, j_3\}, \mathcal{J}^2(1) \cup \mathcal{J}^2(3) = \{j_1, j_2, j_4\}, \ldots, \mathcal{J}^2(2) \cup \mathcal{J}^2(3) = \{j_1, j_2, j_3, j_4\}, \ldots$, which produces subsets of size 3 or 4. Assume we have constructed, in the previous step, an ordered sequence of subsets $\mathcal{J}^{\ell-1}(1), \ldots, \mathcal{J}^{\ell-1}(p_{\ell-1})$ where all indices of each subset are listed in increasing numerical order and the subsets are ordered so that the corresponding values of $S\{\mathcal{J}^{\ell-1}(j)\}$ are increasing. The new family of subsets $\mathcal{J}_1 = \mathcal{J}^{\ell-1}(1) \cup \mathcal{J}^{\ell-1}(2), \ldots, \mathcal{J}_{p_{\ell-1}-1} = \mathcal{J}^{\ell-1}(1) \cup \mathcal{J}^{\ell-1}(p_{\ell-1})$, $\mathcal{J}_{p_{\ell-1}} = \mathcal{J}^{\ell-1}(2) \cup \mathcal{J}^{\ell-1}(3), \ldots$ is filtered in order to retain only p_ℓ^* distinct subsets, where $p_\ell^* \le \frac{1}{2} p_{\ell-1} (p_{\ell-1} - 1)$, and the indices in each subset form a strictly increasing sequence. Then, the permutation $\hat{j}_\ell(1), \ldots, \hat{j}_\ell(p_\ell^*)$ of $1, \ldots, p_\ell^*$ is carried out so that $S(\mathcal{J}_{\hat{j}_\ell(1)}) \le \ldots \le S(\mathcal{J}_{\hat{j}_\ell(p_\ell^*)})$, and we retain for the next step only the $p_\ell = \min(p_1, p_\ell^*)$ top subsets $\mathcal{J}^\ell(1) = \mathcal{J}_{\hat{j}_\ell(1)}, \ldots, \mathcal{J}^\ell(p_\ell) = \mathcal{J}_{\hat{j}_\ell(p_\ell)}$.

The algorithm can be terminated when a predetermined percentage of the mean squared variation among the Y_is is explained by the regressions, or when the difference between two successive measures of that variation falls below a given level, or there is a marked "kink" in a graph of the minimum value of $S\{\mathcal{J}^\ell(1)\}$ against ℓ. The second of these three rules can be interpreted as stopping as soon as, for some $\ell \ge 1$,

$$\frac{S\{\mathcal{J}^\ell(1)\} - S\{\mathcal{J}^{\ell+1}(1)\}}{S\{\mathcal{J}^\ell(1)\}} \le t\,, \tag{2.3}$$

where $t = t(n)$ is a user-choosable threshold expressing a necessary minimum gain in going to the next step. The estimator $\hat{g}$ is then computed in a standard way, using the "favorite nonparametric function estimator" referred to in Section 2.1, from the data pairs

$(X_i^{\mathcal{J}^\ell(1)}, Y_i)$ for $1 \leq i \leq n$, where $X_i^{\mathcal{J}^\ell(1)} = (X_{ij})_{j \in \mathcal{J}^\ell(1)}$. The "kink" approach is commonly used to determine a stopping point for clustering algorithms, where the value of S at (2.1) is replaced by a measure of the tightness of a cluster.

From now on, this nonparametric variable selection method will be referred to as NOVAS.

2.4 Practical issues.

Our method is computationally intensive; launching it with a very large dataset may be time consuming. One way to speed up computation is to parallelize the algorithm. Indeed, as soon as a computer is equipped with a multicore processor, which is the case for most current computers, parallelization allows us to process independent tasks simultaneously. The running time is then divided by the number of independent tasks that the multicore processor is able to manage. The programming language R (R Development Core Team, 2011) offers packages that make such a parallelization easy; see for instance the R package "doSNOW" of Revolution Analytic (2011). In addition, since R is freeware and used intensively by academic researchers, this programming language is one of the most popular in the statistical community.

For these reasons we decided to use the R programming language to implement our variable selection method. All results presented with respect to the real data application (see Section **??**) were obtained using a laptop with a 4-core, 2 GHz processor with 4 GB RAM. For both real and simulated data, the default threshold parameter $t = 0.05$ is used throughout, except when one addressing the influence of t in Section 4.3.. To give an impression of the run time, the R routine NOVAS was repeated for an artificial dataset containing $p = 100$, 500, 1,000, 5,000, 10,000 and 50,000 covariates, in such a way that $\ell = 4$ steps were run systematically for each p with $n = 100$. Seven parallel jobs were launched, this being an efficient number for the laptop we were using. The corresponding run times, in seconds, are 8, 45, 86, 481, 1041 and 5648. It is worth noting there is an almost perfect linear relationship between the logarithm of the number of variables, i.e. $\log p$, and the corresponding log run time. Consequently, considering only $p = 100$ and $p = 500$ is adequate for obtaining a good approximation to the run time for much higher dimensional cases. The simulation study, which requires substantial computation,

was made possible by access to a supercomputer; see the acknowledgements.

The nonparametric regression estimator $\hat{\gamma}_{\mathcal{J}}$, introduced in (2.1), is the usual local linear one (see e.g. Fan and Gijbels, 1996). In order to speed up computation, the covariates were standardized and, for each subset $\mathcal{J}$, we chose a common smoothing parameter, among a given set of candidates, to minimize the cross-validation criterion $S(\mathcal{J})$, defined at (2.2) with $w_{|\mathcal{J}|} \equiv 1$. The kernel estimator is adaptive in the sense that the smoothing parameter is defined in terms of nearest neighbors.

Material for implementing NOVAS using real or simulated data is available online at http://www.math.univ-toulouse.fr/~ferraty. Click on the link "Softwares & Materials Online" in the left menu, and online resources will appear.

3. GENOMICS DATASET

This dataset was discussed by Bushel *et al.* (2007). It is also addressed in the R package mixOmics, which is designed to explore and integrate omics data and was developed by Dejean *et al.* (2011). The dataset treats liver toxicity and contains the expression levels of 3116 genes, or covariates, and nine clinical measurements, or scalar responses, for 64 rats. The original dataset included supplementary clinical responses, but these were not used since only three distinct values were available.

Our aim was to select, for each scalar response, the genes leading to the best predictor in terms of the cross-validation criterion at (2.1). Table **??** details stages of NOVAS when the aim is to predict the level of urea nitrogen. As pointed out earlier, the final model may involve variables not necessarily identified as the most predictive ones in the previous stages. Table **??** gives, for each clinical measurement, the gene numbers, i.e. the subset $\widehat{\mathcal{J}}$,

Table 1: The table gives, at each stage, the best predictive subset of variable number(s) and corresponding leave-one-out cross-validation criterion.

Stage number	Selected gene numbers	cv
1	1165	6.83
2	1866 2050	5.22
3	1000 1167 1837 1957	3.76
4	1000 1167 1837 1899 1957	3.27

selected by NOVAS, together with values of the corresponding cross-validation criterion,

i.e. $S(\widehat{\mathcal{J}})$, where the clinical response abbreviations were defined as follows: BUN, urea

Table 2: NOVAS selected models for each clinical measurement.

Clinical measurement	Selected gene numbers
BUN	1000 1167 1837 1899 1957
TP	1159 1970 2020 2173 2923 2927 2971
ALB	1038 1165 1992 2020 2105 2669 2867 2921
ALT	1846 1871 1883 1909 1910 1911 1915 1921 2042
SDH	764 1145 1624 1866 1940 1992 1996 2894
AST	977 1116 1161 1335 1826 1891 1909 1961 2197 2201
ALP	1064 1484 1817 1823 2007 2385 2819
TBA	1891 1913 1916 1917 1954 2200 2205
CHOL	1836 1875 2044

nitrogen; TP, total protein; ALB, albumin; ALT, alanine aminotransferase; SDH, sorbitol dehydrogenase; AST, aspartate aminotransferase; ALP, alkaline phosphatase; TBA, total bile acids; and CHOL, cholesterol. Two clinical responses, BUN and CHOL, require three or five genes. Other clinical measurements, including AST, involve many more genes. Figure **??** displays, for each clinical variable, the observed values plotted against the leave-one-out predictions. As can be seen, clinical responses are well explained by the selected genes.

An important question arises: is the quality of the leave-one-out predictions high? To answer this question we compared the leave-one-out cross-validation criterion, i.e. $S(\widehat{\mathcal{J}})$, obtained by NOVAS, with various alternative predictive methods:

- Partial least squares regression, PLS, which is a non-selective iterative linear method and derives successive linear combinations, or loadings, of covariates maximizing its correlation with the response. It was originally developed by Wold (1966) for applications in economics and became a popular tool in the chemometrics community; see, for instance, Geladi and Kowalski (1986) or Martens and Naes (1989).

- A sparse version of PLS, sPLS, including a lasso step leading to sparse loadings, developed by Lê Cao *et al.* (2008), who experimented with this genomics dataset.

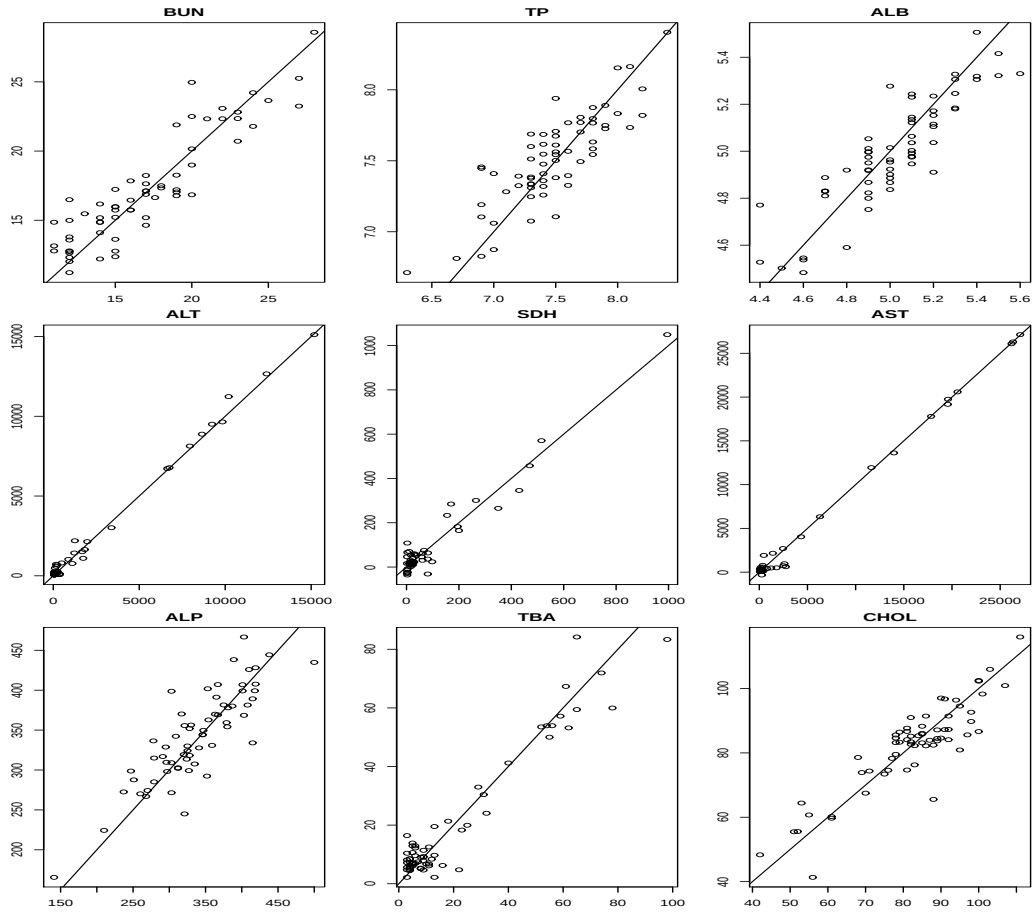

Figure 1: Observations (horizontal axis) plotted against leave-one-out predictions for each clinical measurement.

- Least angle regression, LAR, introduced by Efron *et al.* (2004), which is one of the most popular selective linear regression methods.

- Most predictive design points, MPDP, which is also an existing nonparametric alternative method and which we shall discuss in Section 4.5.

The PLS method requires the choice of only one parameter, the total number of loadings, whereas sPLS needs several variables, specifically the total number of loadings and the sparsity expressed as the number of zeros for each loading. The LAR procedure requires choice of the optimal fraction of nonzero values in the vector of parameters. For all these competing methods, the parameters were optimized so as to minimize the predictive leave-one-out criterion, and Table 3 gives the smallest leave-one-out cross-validation values obtained for each procedure. It can be seen that NOVAS outperforms alternative linear methods, since it is able to take nonlinearities into account.

Table 3: Leave-one-out cross-validation values for each clinical measurement, or response, and each method. Minimum values in each row are given in bold.

Responses \ Methods	PLS	sPLS	LAR	MPDP	NOVAS
BUN	6.62	8.106	8.67	**2.62**	3.27
TP	0.104	0.115	0.117	0.072	**0.045**
ALB	0.0351	0.0378	0.044	0.02	**0.015**
ALT	1236163	1260917	1709814	**46834**	60621
SDH	19736.38	23512.95	21314.9	2669.6	**1404.7**
AST	5232362	6131264	92534432	2580580	**318682**
ALP	3097.81	3194.96	3203.1	1225.4	**1043.7**
TBA	138.26	118.63	153.12	67.50	**39.73**
CHOL	76.91	72.87	94.96	**26.77**	40.68

To enable predictive performances to be visualized, Figure **??** compares, for each method, the results of leave-one-out estimation applied to a sample of four clinical measurements: TP, SDH, AST and CHOL. Clearly, the two nonparametric selection procedures, NOVAS and MPDP, have significantly greater predictive performance than the linear procedures, and NOVAS outperforms MPDP six times out of nine. For example, NOVAS leads to more accurate predictions in most cases, and enjoys spectacular performance when applied to predicting the clinical measurement AST. However, MPDP is superior for three clinical measurements out of nine. An appropriate methodology consists of boosting NOVAS by comparing it systematically with MPDP, so as to be sure to select the best predictive model.

4. ASSESSING PERFORMANCE

4.1 Simulated regression models.

We consider five models, indexed by a superscript m in square brackets and having the form

$$Y_i \;=\; \gamma^{[m]}_{1,2,3}(X_{i1}, X_{i2}, X_{i3}) \,+\, \varepsilon^{[m]}_i\,, \quad i = 1, \ldots, n_m\,,$$

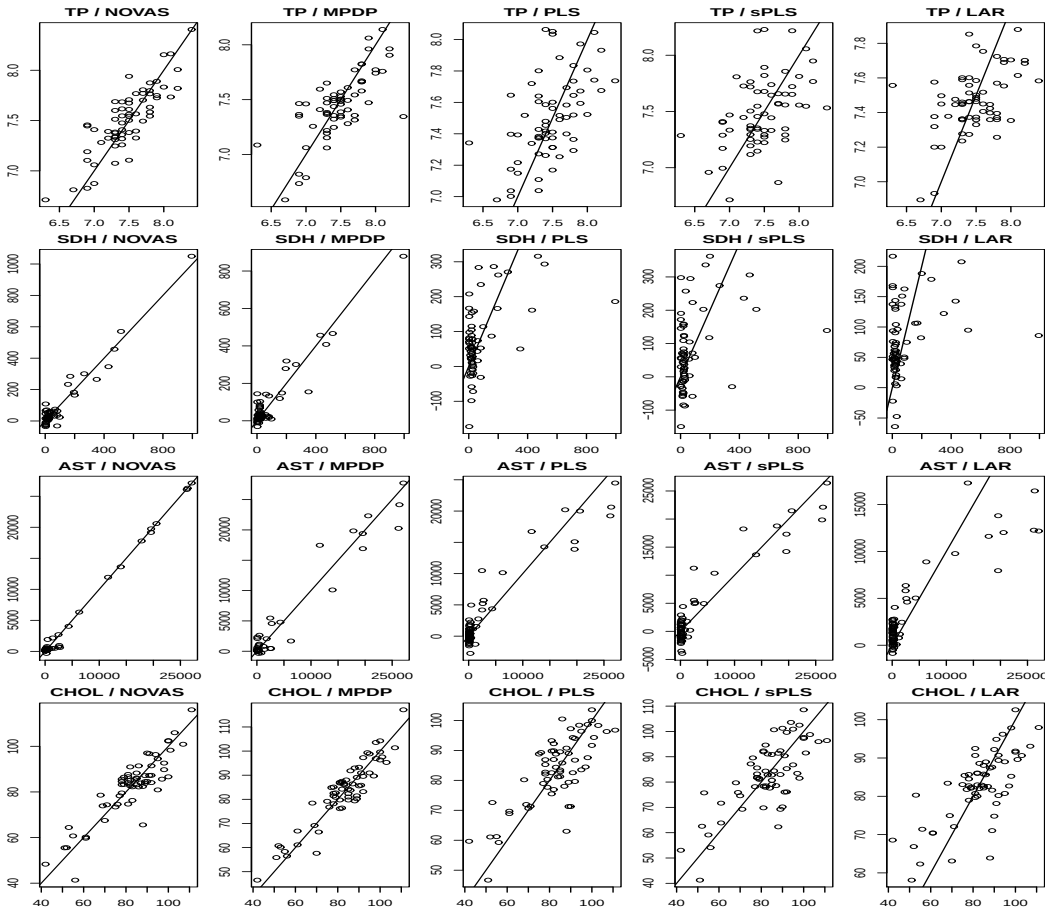

Figure 2: Four observed responses, TP, SDH, AST and CHOL, against leave-one-out estimations.

where

$$\gamma^{[1]}_{1,2,3}(X_{i1}, X_{i2}, X_{i3}) = X^2_{i1} + X^2_{i2} + X^2_{i3}\,,$$

$$\gamma^{[2]}_{1,2,3}(X_{i1}, X_{i2}, X_{i3}) = |X_{i1}X_{i2}| + |X_{i1}X_{i3}| + |X_{i2}X_{i3}|\,,$$

$$\gamma^{[3]}_{1,2,3}(X_{i1}, X_{i2}, X_{i3}) = |X_{i1}X_{i2}X_{i3}|\,,$$

$$\gamma^{[4]}_{1,2,3}(X_{i1}, X_{i2}, X_{i3}) = \frac{|X_{i1}X_{i2}| + X^2_{i3}}{2 + X_{i1}X_{i2}X_{i3}}\,,$$

$$\gamma^{[5]}_{1,2,3}(X_{i1}, X_{i2}, X_{i3}) = \frac{|X_{i1}X_{i2}| + |X_{i1}X_{i3}|}{2 + |X_{i2}X_{i3}|}\,.$$

The vector components X_{ij} are taken to be independent and identically distributed as uniform $[-1, 1]$, and the errors $\varepsilon_i^{[m]}$ are independent and identically distributed as nor-

mal $\mathrm{N}(0, \sigma_m^2)$, where $\sigma_m^2 = 0.05\,\mathrm{var}\{\gamma_{1,2,3}^{[m]}(X_{i1}, X_{i2}, X_{i3})\}$. Different sample sizes will be considered in the simulation study, to take into account the varying complexities and dimensionalities of these models.

4.2 Influence of the sample size, n, and the number, p, of covariates.

Is NOVAS able to recognize the correct subset, $\mathcal{J} = \{1, 2, 3\}$, even for large values of p? To answer this question our selected procedure was launched for each of the five models, with four sample sizes, n = 50, 100, 150, 200; three different sets of covariates, of sizes p = 100, 1,000, 10,000; and with the threshold parameter t, defined at (2.3), set equal to 0.05 for each run. We then considered $5 \times 4 \times 3$ situations, and the simulation scheme was repeated 100 times, producing 100 datasets in each situation. Table **??** presents the results in order of sample size. It can be seen that the higher the complexity of the model, the larger should be the the sample size if the model is to be identified correctly. So, the role played by the sample size reflects what happens usually in statistics: for higher dimensional models, larger sample sizes are needed to obtain good results. For too small a sample size, i.e. $n = 50$, NOVAS is unable to recognize models with a good degree of accuracy, except in the case of Model 1 when $p = 100$. At the opposite end of the spectrum, when considering a relatively large sample size, specifically $n = 200$, we obtain good results for all models, even for large sets of covariates. The influence of p on NOVAS is clear: the higher the number of covariates, the lower the frequency with which NOVAS selects the correct model. However, for a sufficiently large sample size, the influence of p is more modest; for any p = 100, 1,000, 10,000, one obtains good and stable results for models 1 and 2 when $n = 100$, for model 3 when $n = 150$, and for models 4 and 5 when $n = 200$.

4.3 Influence of the threshold t.

As noted in the previous section, considering different sample sizes allows us to reduce the effect on NOVAS of the dimension of the simulated models; see Table **??**. A high value of t results in the procedure being stopped too early, in which case NOVAS retains too small

Table 4: Number of times, out of 100, that NOVAS selected the correct model.

		$n = 50$	$n = 100$	$n = 150$	$n = 200$
Model 1	$p = 100$	84	99	100	100
	$p = 1000$	46	100	100	100
	$p = 10000$	12	100	100	100
Model 2	$p = 100$	49	100	100	100
	$p = 1000$	25	99	100	100
	$p = 10000$	1	97	100	100
Model 3	$p = 100$	4	89	100	100
	$p = 1000$	2	79	100	100
	$p = 10000$	0	56	99	100
Model 4	$p = 100$	18	78	97	100
	$p = 1000$	6	58	91	98
	$p = 10000$	0	34	74	95
Model 5	$p = 100$	2	46	76	97
	$p = 1000$	1	28	67	90
	$p = 10000$	0	1	54	76

Table 5: Number of times, out of 100, that NOVAS selected the correct model when $p = 1000$.

t	0.01	0.05	0.1	0.15	0.2	0.25	0.3	0.35	0.4	0.45	0.5
Model 1 ($n = 100$)	100	100	100	100	100	98	95	72	46	12	8
Model 2 ($n = 100$)	99	100	99	100	98	87	69	41	25	5	1
Model 3 ($n = 150$)	100	100	99	99	94	88	61	44	9	1	0
Model 4 ($n = 200$)	98	98	99	94	69	38	12	6	1	0	0
Model 5 ($n = 200$)	92	90	87	89	84	75	45	17	2	1	0

a set of variables. When the cross-validation criterion (2.2) decreases slowly around its minimum, a low value of t results in the selected model incorporating too large a number of variables. However, as we shall show, NOVAS is not particularly sensitive to the value of t. As indicated in Table **??**, there is a range of values for t, i.e. $t \leq 0.2$, where NOVAS provides stable results. This encouraged us to use the default value $t = 0.05$.

4.4 Influence of noise-to-signal ratio.

Noise-to-signal ratio is defined by $\text{nsr} = \sigma_m^2/\text{var}\{\gamma_{1,2,3}^{[m]}(X_{i1}, X_{i2}, X_{i3})\}$. Up to now, $\text{nsr} = 0.05$ has been used in our numerical experiments. Table **??** summarises the influence of noise-to-signal ratio on the behaviour of NOVAS, and it can be seen that performance decreases by 10 to 54% as noise-to-signal ratio increases by 100 to 700%. Nevertheless,

Table 6: Number of times, out of 100, that NOVAS selected the correct model when $p = 1000$.

nsr	0.05	0.1	0.2	0.4
Model 1 ($n = 100$)	100	100	100	78
Model 2 ($n = 100$)	100	98	91	58
Model 3 ($n = 150$)	100	100	96	84
Model 4 ($n = 200$)	98	96	99	89
Model 5 ($n = 200$)	90	96	75	41

the performance of NOVAS remains stable with respect to noise-to-signal ratio; when nsr = 0.1, the ability of NOVAS to recognise the true subset is very good for all models; when nsr = 0.2, NOVAS is still largely correct for models 1 to 4; and when nsr = 0.4, the results for models 1, 3 and 4 are reasonable. The performance of NOVAS, and of the competing methods that we shall discuss in the next section, degrades for higher values of nsr.

4.5 Comparison with other methods.

In this section we compare NOVAS with the competing methods introduced in Section 3: PLS, sPLS, LAR and MPDP, which is a nonparametric selection technique called "most predictive design points", introduced by Ferraty *et al.* (2010). Originally developed for functional data, the method remains valid in the more conventional high-dimensional setting of the present paper. The idea is to select, one by one, several variables among a large number of candidates, in order to predict nonparametrically a scalar response. The first step of MPDP chooses the most predictive variable by minimizing (2.1), and updates the subset of candidates by dropping it; the second step selects the most predictive variable among the new subset of candidates, with respect to (2.1), and again updates the subset of candidates; and so on. This procedure is repeated until the relative gain, measured by the cross-validation criterion, between two consecutive steps does not exceed some given threshold; see (2.3). The nonparametric regression estimator suggested for MPDP is the local linear one. The fundamental difference from NOVAS comes at the second step; for any $\ell > 1$, NOVAS may drop at step $\ell + 1$ some covariates selected at step ℓ, whereas that is not possible with the sequential feature of MPDP.

In order to implement a comparison study, a family of regression models,

$$Y_i \,=\, \gamma^{\alpha}_{1,2,3}(X_{i1}, X_{i2}, X_{i3}) \,+\, \varepsilon_i \,,$$

indexed by a scalar α, was simulated, with

$$\gamma^{\alpha}_{1,2,3}(X_{i1}, X_{i2}, X_{i3}) \,=\, 3 \,+\, \alpha\, (X_{i1} + X_{i2} + X_{i3}) \,+\, (1-\alpha)\left(X^2_{i1} + X^2_{i2} + X^2_{i3}\right),$$

where the X_{ij}s and the ε_is are defined according to the same scheme described in Section 4.1. We took $n = 50$, $p = 1{,}000$ and snr $= 0.1$. This family of models allows us to consider the purely nonlinear situation, $\alpha = 0$, as well as the purely linear setting, $\alpha = 1$. We also took into account intermediate semilinear models, by setting $\alpha = 0.35$; this value allowed us to balance the contributions to variability from linear and nonlinear parts. Only variables 1, 2, and 3 were active. For each value of α, 100 datasets were simulated. This particular simulation scheme tested the selective procedures rather severely, since most of the time they were not able to detect the exact set of active covariates; see Table **??**.

LAR clearly outperforms both MPDP and NOVAS in the purely linear case. However, in the purely nonlinear setting, the behavior of MPDP and NOVAS is much better than LAR (this was expected, since LAR is not designed for nonlinear relationships), with NOVAS having a significant advantage. In fact, most of the time, MPDP selected extra

Table 7: Number of times, out of 100, that the correct model is selected.

	$\alpha = 0$ (nonlinear)	$\alpha = 0.35$ (semilinear)	$\alpha = 1$ (linear)
LAR	0	2	50
MPDP	28	23	0
NOVAS	44	51	0

covariates outside the active ones. In order to better assess the selective performance of these methods, Table **??** details how many times, out of 100, each active covariate was selected, and does this for each α. Of course, when tabulating all selected models, in addition to the active variables 1, 2, and 3, a number of other covariates are retained, three times at most; see the column "others" in Table **??**.

Simulating as many as 1000 variables has, in a variety of settings, the effect of unduly

Table 8: Number of times, out of 100, that indicated covariates are selected; the column "others" gives the maximum number of times that an extra covariate is selected over the 100 runs.

	$\alpha = 0$ (nonlinear)				$\alpha = 0.35$ (semilinear)				$\alpha = 1$ (linear)			
Covariates	1	2	3	others	1	2	3	others	1	2	3	others
LAR	0	1	0	3	38	46	39	3	100	100	100	2
MPDP	48	51	52	2	69	74	72	2	79	76	77	4
NOVAS	70	69	70	2	82	84	89	2	93	90	93	3

activating some of them, essentially randomly, simply because there are so many of them. Hence, the algorithm may select some them without apparent reason. This phenomenon also occurs with the linear LAR procedure, but since MPDP and NOVAS are more flexible than LAR, both these nonlinear methods tend to suffer more than does LAR, which behaves more systematically, particularly when the sample size is small. Most importantly, the true model is often included in the selected set by NOVAS (and less frequently in the case of MPDP).

One way of overcoming this over-selection issue is to launch MPDP using a higher threshold (for instance, $t = 0.2$) on the preselected set using NOVAS, in order to retain only covariates that significantly improve the prediction criterion. The resulting run time is practically the same, since MPDP operates on a very small set of covariates, and the number of runs needed to select correctly the linear model is much larger (78 out of 100 instead of 0).

Table **??** compares the predictive performance of NOVAS with that of all competing methods. To obtain these results the mean of the cross-validation estimator of average prediction error, defined at (2.1) and (2.2), was computed over 100 simulated datasets for each value of α. It can be seen from Table **??** that PLS, sPLS and LAR perform similarly, although the predictive performance of PLS in the linear setting is poor. Naturally, in the unfavorable setting of the nonlinear model, linear methods fail; PLS, sPLS and LAR perform almost exactly the same as the simple leave-one-out empirical mean of the responses, where for any i, Y_i is predicted naively by $(n-1)^{-1} \sum_{j \neq i} Y_j$. When comparing NOVAS with MPDP, it seems that both methods have similar predictive behavior, although NOVAS enjoys smaller variances in all cases.

The superiority of NOVAS, relative to MPDP, includes an ability to correctly identify

Table 9: mean and variance, in parentheses, of cross-validation criterion, out of 100 simulated datasets.

	$\alpha = 0$ (nonlinear)	$\alpha = 0.5$ (semilinear)	$\alpha = 1$ (linear)
PLS	0.30 (0.004)	0.27 (0.004)	1.07 (0.037)
sPLS	0.38 (0.023)	0.25 (0.007)	0.18 (0.004)
LAR	0.28 (0.004)	0.23 (0.003)	0.21 (0.006)
MPDP	0.11 (0.001)	0.07 (0.001)	0.15 (0.043)
NOVAS	0.10 (0.0005)	0.06 (0.0002)	0.11 (0.004)

the variables on which the regression actually depends, but using a shorter running time. Indeed, it is easy to see that, for achieving k steps when dealing with a set of p covariates, MPDP requires estimation of $kp - 0.5k(k-1) + 1$ regression models, whereas NOVAS involves only $kp/2$ models, essentially half the previous number. Here, for this comparative study, MPDP needs on average one step more than NOVAS, which implies that the overall MPDP run time is at least twice as long as the NOVAS one.

Another difference between MPDP and NOVAS is noticed when one has to deal with redundant variables that are correlated with non-redundant ones. To illustrate this aspect, we simulated data from model 4 with different sample sizes, as in Section **??** ($n = 50, 100, 150, 200$), with nsr $= 0.05$, and with 1,000 covariates. We replaced the 1,000th explanatory variable by a combination of both the first two: For $i = 1, \ldots, n$ we took $X_{i\,1000} = X_{i1}^2 \,|\, X_{i\,2}|^{1/3}$. This 1,000th variable contains redundant information which can mask the main role played by variables 1 and 2 in the simulated regression model. Table **??** details the selected variables, over 100 runs for each sample size.

Table 10: Number of times, out of 100, that the indicated subsets were selected; $\mathcal{J}_{no\,intruder}$ is one of $\{1, 3, 1000\}$, $\{2, 3, 1000\}$ and $\{3, 1000\}$, and $\mathcal{J}_{intruder}$ represents any subset containing intruder(s) (i.e. j with $j \notin \{1, 2, 3, 1000\}$).

	$\{1,2,3\}$		$\{1,2,3,1000\}$		$\mathcal{J}_{no\,intruder}$		$\mathcal{J}_{intruder}$	
n	MPDP	NOVAS	MPDP	NOVAS	MPDP	NOVAS	MPDP	NOVAS
50	1	1	0	0	6	19	93	80
100	7	38	2	1	17	33	74	28
150	2	79	18	2	19	15	71	4
200	0	96	43	4	23	0	34	0

Note that the ability of MPDP to identify variables 1, 2, and 3, but most of the time combined with variable 1000, increases with n; for instance, $\{1, 2, 3, 1000\}$ is selected 43 times, out of 100, when $n = 200$. The artificially redundant 1,000th covariate acts as a

"trap" for MPDP, which looks for the most predictive variable at each step. Moreover, this mechanism leads systematically to selecting variable 3 at the first step, and variable 1,000 at the second step, for the largest sample size. Consequently, MPDP selects variables 1, 2 and 3 only rarely, and never chooses them for large n, whereas the performance of NOVAS increases significantly with n.

Another weakness of MPDP is its propensity to retain, essentially arbitrarily, extra covariates which have no connection with the variables involved in the model. Even if this trend is reduced for large sample sizes, it still occurred 34 times out of 100 for the largest sample size in our simulation study. The algorithm NOVAS is much less sensitive to this trap, because it allows quite new models, built from submodels which are not necessarily the most predictive, to be selected.

5. THEORETICAL PROPERTIES

5.1. Reminder of notation. Recall from section 2 that our algorithm reorders the components of the vectors X_i, and that the reordering alters from step to step until we terminate the algorithm. If $\mathcal{J} = \{j_1, \ldots, j_\ell\} \subseteq \{1, \ldots, p\}$ then, at any given step (where $\mathcal{J}$ depends not only on that step but on the steps that preceded it), the reordered X_i is denoted by $X_i^{\mathcal{J}}$, where for a general p-dimensional vector $u = (u_1, \ldots, u_p)$ of $\mathbb{R}^p$, $u^{\mathcal{J}}$ denotes a particular subvector of length ℓ: $(u_{j_1}, \ldots, u_{j_\ell})$. The reader is referred to the discussion on either side of (2.2) for details of the argument used to define $u^{\mathcal{J}}$. In particular, as noted there, in successive steps of the algorithm we merge sets $\mathcal{J}_1, \ldots, \mathcal{J}_k$ "in a pairwise manner, creating new subsets of indices $\mathcal{J}_1 \cup \mathcal{J}_2$, ..., $\mathcal{J}_1 \cup \mathcal{J}_k$, ..., $\mathcal{J}_{k-1} \cup \mathcal{J}_k$ that are rearranged again to rank the corresponding predictive values; and we repeat this process until we obtain a subset $\mathcal{J}$ with a sufficiently small value of $S(\mathcal{J})$," where $S(\mathcal{J})$ is defined at (2.2), and practical definitions of "sufficiently small" are given in the paragraph containing (2.3).

In consequence, in step ℓ the algorithm generates subsets $\mathcal{J}$ of indices such that $\ell \le |\mathcal{J}| \le 2^{\ell-1}$. In the development in section 2.3 these subsets were written as $\mathcal{J}_{\hat{\jmath}_\ell(1)}, \ldots, \mathcal{J}_{\hat{\jmath}_\ell(p_\ell^*)}$, where $\hat{\jmath}_\ell(1), \ldots, \hat{\jmath}_\ell(p_\ell^*)$ was a permutation of a subset of $1, \ldots, p$ determined by working through the first ℓ steps.

5.2. Main result. In the discussion in this section, and in the proof of Theorem 1,

we simplify notation by writing $\hat{\jmath}_\ell(1), \ldots, \hat{\jmath}_\ell(p^*_\ell)$ as simply $j_1, \ldots, j_\ell$, and dropping the superscript notation involving $\mathcal{J}$ on the X_is. In this notation we take $\hat{\gamma}_{j_1,\ldots,j_\ell}$, in (2.1), to be a conventional local linear estimator in a regression on ℓ variables, i.e.

$$\hat{\gamma}_{j_1,\ldots,j_\ell}(x) = \bar{Y}(x) + \{\bar{X}(x) - x\}^{\mathrm{T}}\, \widehat{\Sigma}(x)^{-1}\, T(x)\,, \tag{5.1}$$

where

$$\bar{X}(x) = \frac{\sum_i K\{(x - X_i)/h\}\, X_i}{\sum_i K\{(x - X_i)/h\}}\,, \quad \bar{Y}(x) = \frac{\sum_i K\{(x - X_i)/h\}\, Y_i}{\sum_i K\{(x - X_i)/h\}}\,,$$
$$\widehat{\Sigma}(x) = \frac{\sum_i \{X_i - \bar{X}(s)\}\, \{X_i - \bar{X}(s)\}^{\mathrm{T}}\, K\{(x - X_i)/h\}\, X_i}{\sum_i K\{(x - X_i)/h\}}\,,$$

$K(u_1, \ldots, u_\ell) = K_1(u_1) \ldots K_1(u_\ell)$, K_1 is a univariate, uniformly bounded, compactly supported, symmetric probability density, and h is a bandwidth.

This relieves us of the very complex notational task of recording in detail all the changing permutations mentioned in section 5.1, as we progress through the steps of the algorithm. Thus, when in Theorem 1 below we refer to "the first r components of X_i alone" we do not mean the first r components of the original p-vector X_i in the model introduced in the first paragraph of section 2.1, but the first r components of the p-vector after its components have been permuted repeatedly by steps $1, \ldots, \ell$ of the algorithm, where ℓ is taken to increase until, as the theorem asserts, the algorithm terminates.

For simplicity we use the same bandwidth for each component, although of course we could be more ambitious. Our assumptions, (5.4), (5.5) and (5.6) are stated and discussed in Sections 5.3–5.5. The theorem is proved in Appendix A.

Theorem 1. *If* (5.4), (5.5) *and* (5.6) *hold then, with probability converging to 1 as* $n \to \infty$*, the algorithm correctly concludes that* $g(X_i) = E(Y_i \,|\, X_i)$ *is a function of the first* r *components of* X_i *alone, and in particular the algorithm terminates at Step* r*.*

It is straightforward to prove from the theorem that, if the assumptions there hold, then the regression estimator based on the components to which the algorithm leads has, to first order, the same asymptotic properties as an oracle procedure based on being told in advanced that $E(Y_i \,|\, X_i)$ is a function of the first r components of X_i alone. In particular, the asymptotic bias and variance of estimators of g that are founded on the

conclusion of the algorithm are first-order equivalent to their counterparts for an oracle estimator.

5.3. Assumptions (5.4) *and* (5.5). These are the main conditions for the theorem. To state (5.4), let f denote the p-variate density of X, and, given $j_1, \ldots, j_\ell$, let $\phi_{j_1,\ldots,j_\ell}(x_1, \ldots, x_p)$ be the p-variate density proportional to $f(x_1, \ldots, x_p)\, w_\ell(x_{j_1}, \ldots, x_{j_\ell})$, where w_ℓ is as in (2.1). Define $\psi_{j_1,\ldots,j_\ell}(x_{j_1}, \ldots, x_{j_\ell})$ to be the integral of $\phi_{j_1,\ldots,j_\ell}(x_1, \ldots, x_p)$ over x_i for each $i \notin \{j_1, \ldots, j_\ell\}$, and let $\gamma_{j_1,\ldots,j_\ell}(x_{j_1}, \ldots, x_{j_\ell})$ be the value that $E\{g(X_i) \,|\, X_{ij_1} = x_{j_1}, \ldots, X_{ij_\ell} = x_{j_\ell}\}$ would take if X had density $\phi_{j_1,\ldots,j_\ell}$ rather than f:

$$\gamma_{j_1,\ldots,j_\ell}(x_{j_1}, \ldots, x_{j_\ell}) = \frac{\int g(x_1, \ldots, x_r)\, \phi_{j_1,\ldots,j_\ell}(x_1, \ldots, x_p)\, dx''}{\psi_{j_1,\ldots,j_\ell}(x_{j_1}, \ldots, x_{j_\ell})}\,, \tag{5.2}$$

where x'' is the $(p-\ell)$-vector that remains after $x_{j_1}, \ldots, x_{j_\ell}$ have been removed from $(x_1, \ldots, x_p)$. Define

$$\begin{aligned} u_\ell(j_1, \ldots, j_\ell) &= \int \{g(x_1, \ldots, x_r) - \gamma_{j_1,\ldots,j_\ell}(x_{j_1}, \ldots, x_{j_\ell})\}^2 \\ &\qquad \times \phi_{j_1,\ldots,j_\ell}(x_1, \ldots, x_p)\, dx_1 \ldots dx_p\,, \qquad (5.3) \\ u_0 &= E\{g(X) - Eg(X)\}^2\,. \end{aligned}$$

We assume that, for a subset $\mathcal{S}_\ell$ of $\mathbb{R}^\ell$ that we take to be a finite union of nondegenerate compact spheres,

(a) Among all ℓ-vectors $(j_1, \ldots, j_\ell)$ satisfying $1 \le j_1 < \ldots < j_\ell \le p$ and $1 \le \ell \le r$, the choice $(1, \ldots, r)$ uniquely minimises $u_\ell(j_1, \ldots, j_\ell)$, in the sense that the minimum over all choices exceeds $u_r(1, \ldots, r)$ by at least a fixed constant $B_3 > 0$, uniformly in n; (b) for some $\eta > 0$, and for $1 \le \ell \le r$, the number of distinct ℓ-vectors $j_1, \ldots, j_\ell$, with $1 \le j_1 < \ldots < j_\ell \le p$, for which $u_\ell(j_1, \ldots, j_\ell) > n^{\eta - \{4/(\ell+4)\}}$, is of strictly smaller order than $\sqrt{q}$, and, for $1 \le \ell \le r$, this includes all ℓ-vectors of distinct integers chosen from $1, \ldots, r$; (c) for each $\ell = 1, 2, \ldots, r+1$ there exists a constant $C_\ell > 1$ such that, for all distinct $j_1, \ldots, j_\ell \in \{1, \ldots, p\}$, the joint density of $f_{j_1,\ldots,j_\ell}$ is bounded below C_ℓ and above C_ℓ^{-1} on $\mathcal{S}_\ell$. (5.4)

Finally we impose basic conditions on the univariate kernel K_1, bandwidth h and weight function w_ℓ in (2.1), and on the manner in which the algorithm is terminated.

Recall that $\mathcal{S}_\ell$ was introduced prior to (5.4).

(a) K_1 is a symmetric, compactly supported, Hölder continuous probability density; (b) the bandwidth $h = h(n)$, when used to construct the ℓ-variate regression estimator $\hat{\gamma}_{j_1,\ldots,j_\ell}$ at (5.1), equals a constant multiple of $n^{-1/(\ell+4)}$; (c) the support of the weight function w_ℓ equals $\mathcal{S}_\ell$, and w_ℓ is bounded and twice differentiable there; (d) we terminate the algorithm using the rule at (2.3), where $t = t(n)$ satisfies $n^{\eta+\{\ell/(\ell+4)\}} \le t \le B_5\, n$, $0 < B_5 < B_3$, B_3 is as in (5.4)(a) and η is as in (5.4)(b). (5.5)

5.4. Discussion of assumptions (5.4) and (5.5). An example where (5.4)(a) fails, and our algorithm consequently has difficulty, arises when $g(x) = x_1 \ldots x_r$, the first r components of X are independent of one another and distributed symmetrically about zero, $w_\ell(t_1, \ldots, t_\ell) = v(t_1) \ldots v(t_\ell)$ where the nonnegative function v is symmetric, and $\mathcal{S}_\ell$ is a sphere centered at the origin. Then the fitted function $\gamma_{j_1,\ldots,j_\ell}$, whenever $1 \le j_1 < \ldots < j_\ell \le r$ and $1 \le \ell \le r-1$, equals 0, and so approximating g by its expected value, conditional on one or more of the first $r-1$ variables, is ineffective. In particular, no one variable has a visible advantage over any other, and so there is no clear opportunity for choosing the correct variables. However, this difficulty evaporates if we take the functions $w_1, \ldots, w_r$ to be sufficiently asymmetric. This example points to the potential influence of w_ℓ in (2.1); for example, it can be used to counteract the negative effects that symmetry has on the algorithm.

Property (5.4)(a) implies that the choice $\ell = r$ and $j_k = k$ for $1 \le k \le r$ uniquely minimises asymptotic mean square prediction error, but by itself (5.4)(a) does not ensure that the algorithm in Section 2.3 takes us to that particular combination of variables. However, the latter property is guaranteed by (5.4)(b), which implies that, with high probability, the singletons (j), for $1 \le j \le r$; the doublets (j_1, j_2), for $1 \le j_1 < j_2 \le r$; and so on up to the r-tuple $(1, \ldots, r)$; are, with probability converging to 1, present among the vectors of indices treated in Steps $1, 2, \ldots, r$, respectively, in the algorithm. Additionally, property (5.6)(d) in section 5.4, which tells us that Y equals a function of the first r components of X alone, plus an independent error, implies that passing

from these r components to $r+1$ components produces, with probability converging to 1, at most a negligibly small decrease in prediction error. In consequence, the rule (2.3) for terminating the algorithm will, with probability converging to 1, lead to a halt immediately after we have concluded, in Step r, that the r-vector $(1, \ldots, r)$ is appropriate. Assumption (5.5)(d), below, also helps in this regard.

More generally, the implication from (5.6)(d) and (5.4)(a) that the choice of variable indices $(j_1, \ldots, j_\ell) = (1, \ldots, r)$ uniquely minimises $u_\ell(j_1, \ldots, j_\ell)$ allows us to investigate an oracle property of conventional type; see the theorem below. That is, there exists a unique choice of variables that leads to best prediction of Y. Without (5.6)(d) and (5.4)(a) there could exist many different choices that produced the same asymptotic minimum mean squared prediction error. For example, without the uniqueness part of (5.4)(a) there could exist components of the p-vector X that were simply copies of the first r components but were positioned quite differently in that vector. On the other hand, if we were not interested in establishing such a result then we could relax (5.4) and (5.6).

Assumption (5.4)(b) guarantees that, although the number of variables that can be used effectively to at least partially explain Y may diverge with increasing n, the number is not so large that extraneous variables fatally confuse the algorithm given in Section 2.3, resulting in the algorithm not correctly identifying the variable indices $(j_1, \ldots, j_\ell) = (1, \ldots, r)$ that best predict a future value of Y.

Condition (5.4)(c) asks merely that the features have the sorts of joint distributions that enable reasonable nonparametric estimation of conditional means such as $E\{g(X_i) \,|\, X_{ij_1}, \ldots, X_{ij_\ell}\}$. To appreciate the reason for the bandwidth choices made in (5.5)(b) we note that, when computing a point estimator $\hat\gamma_{j_1,\ldots,j_\ell}$ of $\gamma_{j_1,\ldots,j_\ell}$, the bias is of order h^2 (since we assumed, in (5.4)(b) and (5.4)(e), that $f_{j_1,\ldots,j_\ell}$ and g have two bounded derivatives), and the error about the mean is of size $(nh^\ell)^{-1/2}$; here we used (5.4)(c). Therefore the optimal bandwidth for point estimation of $\gamma_{j_1,\ldots,j_\ell}$ is of size $n^{-1/(\ell+4)}$, and that is the size assumed in (5.5)(b). The resulting estimator of u_ℓ, $n^{-1}\, S_\ell(j_1, \ldots, j_\ell)$, is in error by $O_p\{(nh^\ell)^{-1/2} + h^2\}$. Actually the term $(nh^\ell)^{-1/2}$ here can be replaced by a quantity of smaller order, and the overall accuracy improved by using a smaller bandwidth, but in practice it will often be the case that the bandwidth is chosen to optimise performance for estimating $\gamma_{j_1,\ldots,j_\ell}$ rather than estimating u_ℓ, and so it is appropriate to

proceed as suggested above.

5.5. Assumption (5.6). Condition (5.6) is standard, except perhaps for the assertion in (5.6)(d) that $g(x_1, \ldots, x_p)$ depends only on the first r components $x_1, \ldots, x_r$. However, since our algorithm is invariant under reorderings of vector components then this assumption is made without loss of generality. It allows us to take g to not depend on n.

(a) $p = p(n)$ is a function of n, diverging at a rate no faster than n^{B_1}, for some $B_1 > 0$, as n increases; (b) the data pairs (X_i, Y_i), for $1 \le i \le n$, are independent and identically distributed, with a common distribution that can depend on n; (c) for all choices of $j_1 < \ldots < j_\ell$ from $1, \ldots, p$, each subvector $(X_{ij_1}, \ldots, X_{ij_\ell})$ of X_i has a well-defined probability density $f_{j_1,\ldots,j_\ell}$ (which may depend on n), and, for each fixed ℓ, all second derivatives of $f_{j_1,\ldots,j_\ell}$ are bounded uniformly in distinct choices of $j_1, \ldots, j_\ell$ from $1, \ldots, p$; (d) $Y_i = g(X_{i1}, \ldots, X_{ir}) + \sigma(X_i)\,\epsilon_i$, where the fixed function g is uniformly bounded and has two uniformly bounded derivatives, the function σ (which may depend on n) is uniformly bounded, and, conditional on X_i, the errors ϵ_i have a distribution depending on neither X_i nor n, with zero mean and satisfying $E|\epsilon_i|^{B_2} < \infty$ for a sufficiently large constant $B_2 > 2$. (5.6)

APPENDIX A: PROOF OF THEOREM

To simplify notation we assume throughout that the function σ, in (5.6)(d), is identically 1. Let $\hat\gamma_{j_1,\ldots,j_\ell}$, $\gamma_{j_1,\ldots,j_\ell}$ and u_ℓ be as at (5.1), (5.2) and (5.3), respectively, define S_ℓ as at (2.1), and let the random variable ϵ have the common distribution of the errors ϵ_i in (5.6)(d). Take η_1 to satisfy $0 < \eta_1 < \eta$, where η is as in (5.4)(b). Then, in view of the assumption of uniform boundedness of second derivatives in (5.6)(c) and (5.6)(d), and the assumption about the support of the density $f_{j_1,\ldots,j_\ell}$ in (5.4)(c),

$$n^{-1}\, S_\ell(j_1, \ldots, j_\ell) = u_\ell(j_1, \ldots, j_\ell) + E\big(\epsilon^2\big) + O_p\big(n^{\eta_1 - \{4/(\ell+4)\}}\big)\,, \tag{A.1}$$

uniformly in $1 \le j_1, \ldots, j_\ell \le p$ and $1 \le \ell \le r+1$. To prove (A.1) we need the constant B_2 in (5.6)(d) to be chosen sufficiently large, depending on B_1 in (5.6)(a)

and η_1 in (A.1). The proof uses Markov's inequality to bound the probability that $|n^{-1}\, S_\ell(j_1, \ldots, j_\ell) - \{u_\ell(j_1, \ldots, j_\ell) + E(\epsilon^2)\}|$ exceeds $n^{\eta_2 - \{4/(\ell+4)\}}$, and observes that, since $p \le n^{B_1}$ (see (5.6)(a)), then, for $1 \le \ell \le r+1$, the number of vectors $(j_1, \ldots, j_\ell)$ being considered is no larger than $O(n^{(r+1)\, B_1})$.

In Step 1 of the algorithm we take $\ell = 1$ and rank values of $S_1(j)$ in order of size. It follows from (5.4)(b) and (A.1) that, with probability converging to 1 as $n \to \infty$, all the indices j for which $u_1(j) \ge n^{\eta - (1/5)}$ are listed among the $\sqrt{q}$ indices for which $S_1(j)$ achieves its $\sqrt{q}$ highest values. Similarly, in Step 2 of the algorithm we conclude with a list of ranked pairs of indices, containing all pairs chosen from among $1, \ldots, r$; and so on, until the rth step, when the list of selected r-vectors $(j_1, \ldots, j_r)$ includes $1, \ldots, r$. It now follows from (5.4)(a) that, with probability converging to 1 as $n \to \infty$, the algorithm will give $(1, \ldots, r)$ the highest rank in Step r, and, from (5.4)(a) and (A.1), that with probability converging to 1 as $n \to \infty$, the inequality in (2.3) holds for the first time when $\ell = r + 1$. Therefore, with probability converging to 1 Step r is the last step, and the r-tuple that is ranked most highly there, i.e. $(1, \ldots, r)$, is the vector of component indices with which the algorithm concludes.

ACKNOWLEDGEMENT

The simulation study of this work was granted access to the HPC resources of the scientific grouping CALMIP under the allocation 2013-P1309. This grouping aims to promote the use of the new technologies in scientific computation in the research community of Toulouse and the French province of Midi-Pyrénées.

REFERENCES

Bühlmann, P., van de Geer, S. (2011). *Statistics for High-Dimensional Data: Methods, Theory and Applications*. Springer.

Bühlmann, P., Meier, L. (2008). Discussion of "One-step sparse estimates in nonconcave penalized likelihood models" (authors H. Zou and R. Li). *Ann. Statist.*, **36**, 1534–1541.

Bushel, P., Wolfinger, R. D., Gibson, G. (2007). Simultaneous clustering of gene expression data with clinical chemistry and pathological evaluations reveals phenotypic prototypes. *BMC Systems Biology* **1**(15). doi:10.1186/1752-0509-1-15.

Candès, E., Tao, T. (2007). The Dantzig selector: statistical estimation when p is much

larger than n. *Ann. Statist.*, **35**, 2313–2351.

Dejean, S., Gonzalez, I., Le Cao, K.-A., Monget, P., Coquery, J. (2011). mixOmics: Omics Data Integration Project. R package version 3.0. http://CRAN.R-project.org/package=mixOmics.

Efron, B., Hastie, T., Johnstone, I., Tibshirani, R. (2004). Least angle regression. *Ann. Statist.*, **32**, 407–499.

Fan, J., Gijbels, I, (1996). *Local Polynomial Modeling and its Applications*. Chapman and Hall, London.

Fan, J., Li, R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *J. Am. Statist. Ass.*, **96**, 1348–1360.

Fan, J., Lv, J. (2010). A selective overview of variable selection in high dimensional feature space (invited review article). *Statistica Sinica*, **20**, 101–148.

Ferraty, F., Vieu, P. (2002). The functional nonparametric model and application to spectrometric data. *Computational Statistics*, **17**, 545–564.

Ferraty, F., Hall, P., Vieu, P. (2010). Most-predictive design points for functional data predictors. *Biometrika*, **97**, 807–824.

Friedman, J., Hastie, T., Höfling, H. and Tibshirani, R. (2007). Pathwise coordinate optimization. *Ann. Appl. Statist.*, **1**, 302–332.

Fu, W. (1998). Penalized regressions: the Bridge versus the lasso. *Journal of Computational and Graphical Statistics*, **7**, 397–416.

Geladi, P., Kowalski, B.R. (1986). Partial least squares regression: A Tutorial. *Analytica Chimica Acta* **185**, 1–17.

Hastie, T., Tibshirani, R., Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction* (2nd edition). Springer, New York.

Huang, J., Horowitz, J. L., Wei, F. (2010). Variable selection via nonparametric additive models. *Ann. Statist.*, **38**, 2282–2313.

Lê Cao, K. A., Rossouw D., Robert-Granié, C., Besse, P. (2008). A sparse PLS for variable selection when integrating Omics data. *Statist. Appl. Genet. Mol. Biol.* **7**, article 35.

Martens, H., Naes, T. (1989). *Multivariate calibration*. John Wiley & Sons Ltd.

Meier, L., van de Geer, S., Bühlmann, P. (2009). High-dimensional additive modeling. *Ann. Statist.*, **37**, 3779–3821.

Meinshausen, N. (2007). Relaxed lasso. *Computational Statistics and Data Analysis*, **52**, 374–393.

R Development Core Team (2011). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org/.

Ravikumar, P., Lafferty, J., Liu, H., Wasserman, L. (2009). Sparse additive models. *J. R. Statist. Soc.* B, **71**, 1009–1030.

Revolution Analytics (2011). doSNOW: Foreach parallel adaptor for the snow package. R package version 1.0.5. (Available from http://CRAN.R-project.org/package =doSNOW.)

Tibshirani, R. (1996). Regression analysis and selection via the lasso. *J. R. Statist. Soc.* B, **58**, 267–288.

Wold, H. (1966). Estimation of principal components and related models by iterative least squares. In: Krishnaiah, P.R. (editors). *Multivariate Analysis*. Academic Press, N.Y., 391–420.

Yuan, M., Lin, Y. (2006). Model selection and estimation in regression with grouped variables. *J. R. Statist. Soc.* B, **68**, 49–67.

Zou, H. (2006). The adaptive lasso and its oracle properties. *J. Am. Statist. Ass.*, **101**, 1418–1429.

Zou, H., Hastie, T. (2005). Regularization and variable selection via the Elastic Net. *J. R. Statist. Soc.* B, **67**, 301–320.