Submitted to the Annals of Applied Statistics

STRUCTURED, SPARSE REGRESSION WITH APPLICATION TO HIV DRUG RESISTANCE

By Daniel Percival [*], Kathryn Roeder[*], Roni Rosenfeld[†], Larry Wasserman[*]

Carnegie Mellon University, Department of Statistics [*] and School of Computer Science [†]

We introduce a new version of forward stepwise regression. Our modification finds solutions to regression problems where the selected predictors appear in a structured pattern, with respect to a predefined distance measure over the candidate predictors. Our method is motivated by the problem of predicting HIV-1 drug resistance from protein sequences. We find that our methods improve the interpretability of drug resistance while producing comparable predictive accuracy to standard methods. We also demonstrate our method in a simulation study and present some theoretical results and connection.

1. Introduction. About twenty antiretroviral drugs are currently available for the treatment of human immunodeficiency virus type 1 (HIV-1). The great majority of these work by inhibiting the activity of various proteins produced by the HIV-1 virus. This effectively impairs the virus' ability to reproduce. Resistance to these drugs develops when a mutation changes the structure of the target protein enough to frustrate the drug while still maintaining the function of the protein. HIV-1 is capable of rapid mutation, and is thus often able to adapt to antiretroviral therapy. Understanding the genetic basis for this developed resistance would allow both for the more effective development of new drugs, as well as more informed prescription of the currently available drugs.

Sequencing HIV-1 proteins can be done reliably, and well-designed in-vitro experiments are available for testing the resistance of a particular strain of HIV-1 to drugs, see Petropoulos *et al.* (2000) and Zhang *et al.* (2005). We approach this problem using regression. This problem setting leads us to build models to predict drug resistance using mutations in the amino acid sequence of the target proteins. We desire models that are easy to interpret and take into account properties of proteins and amino acids. In particular, it is well known that proteins generally function using areas called active sites, that are, simply, areas of the sequence where the protein binds or otherwise interacts with other molecules. This fact leads us to believe that important mutations will tend to be clustered around such sites. When the

protein structure is known, three-dimensional distance can be calculated for any two amino acid positions. But even when the structure of the protein is unknown, because of the continuity of the primary sequence, clustering in three dimensional space generally corresponds to clustering in the protein primary sequence.

We therefore build models for predicting resistance from mutations that have the following two properties. (1) *Sparsity* - a model that uses only a few mutations is easier to interpret and apply. (2) *Structure* - following the concept of active sites, we wish to use mutations that are clustered in the protein primary sequence. Note that this second property is desirable in other applications. For example, Liu, Lin and Ghosh (2007) use genetic pathways to model the genetic influences on prostate cancer. These pathways can be modeled as a structure on individual genes. In this paper, we introduce a variable selection method that builds regression models that satisfy these two properties.

Forward stepwise regression and the lasso are two popular automatic variable selection techniques which are effective at finding sparse regression models. Given data $(X_1, Y_1), \ldots, (X_n, Y_n)$ where $Y_i \in \mathbb{R}$ and $X_i \in \mathbb{R}^p$, the lasso estimator due to Tibshirani (1994) $\widehat{\beta}$ minimizes

$$\sum_{i=1}^{n} (Y_i - \beta^T X_i)^2 + \lambda ||\beta||_1 \tag{1}$$

where $||\beta||_1 = \sum_j |\beta_j|$. Forward stepwise regression is a greedy method that adds one predictor, that is, one element X_i, at a time. Both produce sparse solutions meaning that $\widehat{\beta}_j = 0$ for most j. Sparse solutons are attractive both computationally and for interpretation.

Recent results show that both methods yield estimators with good properties. See, Bunea, Tsybakov and Wegkamp (2007), Greenshtein and Ritov (2004), Wainwright (2007) for results on the lasso and Barron *et al.* (2008) for results on forward stepwise regression. These papers show that, under weak conditions, both approaches yield predictors that are $O(n^{-1/4})$ close to the optimal sparse linear predictor. Moreover, this rate cannot be improved. In our application, extra information is available — we expect nonzero β_j's to cluster together. In this case, we would like to add an additional constraint to the regression.

In this paper we introduce a modification of forward stepwise regression that encourages the selection of new predictors that are "close" — with respect to a distance measure over the predictors — to those already included in the model. We show that our method, Clustered and Sparse Regression

(CaSpaR), is uesful in regression problems where we desire both a sparse and structured solution.

2. Data. The Stanford HIV drug resistance database described in Rhee *et al.* (2003) is a large dataset of HIV-1 protease sequences, along with resistance phenotypes for up to seven different protease inhibitor (PI) drugs for each sequence. This database is a combination of smaller datasets collected in different clinical trials. Since both the genotyping and phenotyping experiments are well standardized, such a joining of data will not give rise to significant heterogeneity-in-sample concerns. Each protease protein sequence is 99 amino acids long. The phenotypes are obtained from in vitro experiments, and are measured in terms of number of multiples of standard dose of drug needed to suppress virus reproduction.

We can cast the problem of connecting genotype to phenotype as a regression problem by treating each mutation as a predictor. Previous work by Rhee *et al.* (2006), and Beerenwinkel *et al.* (2003) have used most modern sparse regression and classification techniques to attack this problem. We seek a model that will take into account protein active sites.

3. CaSpaR. We first introduce the usual regression setting. We have an $n \times p$ data matrix $\mathbf{X}$ and $1 \times n$ response vector $\mathbf{Y}$. We use the usual linear model

$$\mathbf{Y} = \mathbf{X}\beta + \epsilon. \tag{2}$$

Define the support of β by

$$\text{supp}(\beta) = \{i : \beta_i \neq 0, i = 1, \ldots, p\}. \tag{3}$$

We assume that β is sparse (most β_j's are 0) and also that supp(β) has structure. Specifically, we assume that the non-zero elements of β are spatially clustered with respect to a distance measure $d(i, j)$ over the predictors. In other words, the nonzero entries of β appear in some number of groups in which the members are "close" to each other — as defined by the distance measure. Our goal is to accurately recover β, with particular emphasis on this sparsity structure.

We want to modify a sparse regression technique to produce solutions with clusters of nonzero coefficients. Penalized techniques such as the LASSO are difficult to modify for this purpose. Recall that the LASSO finds $\widehat{\beta}$ that minimizes

$$Q(\beta) = \sum_{i=1}^{n} (Y_i - \beta^T \mathbf{X}_i)^2 + \lambda \sum_j |\beta_j|. \tag{4}$$

1. Input: $A = \emptyset$, $\mathbf{X}$, $\mathbf{y}$, $\epsilon > 0$.
2. Fit an OLS model: $\widehat{\beta} = \arg\min_\beta ||\mathbf{X}\beta - \mathbf{y}||_2^2$, s.t. $\text{supp}(\beta) \subseteq A$.
3. Set $i^* = \arg\max_{\{i \notin A\}} |(\mathbf{X}\beta - \mathbf{y})^T \mathbf{x}_i|$.
4. If $|x_{i^*}^T(\mathbf{X}\beta - \mathbf{y})| < \epsilon$ then stop, else set $A = A \cup i^*$ and go to step 2.

TABLE 1
Forward stepwise regression.

The LASSO is computationally efficient because $Q(\beta)$ is convex. It is difficult to add a penalty to $Q(\beta)$ that encourages clustered solutions while maintaining convexity. Note that the *fused lasso* due to Tibshirani and Saunders (2005) adds a penalty of the form $\sum_j |\beta_j - \beta_{j-1}|$. This forces nearby coefficients to be close together in sign and magnitude. We want the support points to be close together but we do not want to force the values of the coefficients to be close together. Instead, we only are concerned with the inclusion or exclusion of predictors.

Stepwise procedures are more flexible and easier to modify, since we do not need to worry about maintaining the convexity of an objective function. We therefore propose a modification to forward stepwise regression (see Table 1 for a description of forward stepwise regression).

We call our algorithm *CaSpaR* (Clustered and Sparse Regression); see Table 2. In each iteration of forward stepwise regression, the following quantities are used to select the next predictor to be added to the model:

$$C_j = |(\mathbf{X}\beta - \mathbf{y})^T \mathbf{x}_j|. \tag{5}$$

Here, $\mathbf{x}_j$ denotes the jth column of $\mathbf{X}$. Note that these are proportional to the correlations between each candidate predictor and the current residuals if the columns $\mathbf{X}$ are scaled to empirical mean zero, variance one. We wish to encourage the selection of predictors that are close, with respect to $d(i,j)$, to those already in the model. To do this, we multiply these C_j by a kernel, which we construct based on the current active set A. This kernel will weight the C_j so that predictors that are close to those already in the model receive more weight than those that are not.

Formally, suppose we have a kernel K_h that is centered at 0, where h denotes the bandwidth parameter. Then, for all $i \notin A$, we compute:

$$W_i = \frac{1}{|A|} \sum_{\{j \in A\}} K_h\left(d(i,j)\right). \tag{6}$$

1. Input: $A = \emptyset$, $\mathbf{X}$, $\mathbf{y}$, $h > 0$, $\alpha \in (0, 1)$, $\epsilon > 0$.
2. Fit an OLS model: $\widehat{\beta} = \arg\min_\beta ||\mathbf{X}\beta - \mathbf{y}||_2^2$, s.t. $\text{supp}(\beta) \subset A$.
3. $\forall i \notin A$, calculate: $W_i = \frac{1}{|A|} \sum_{\{j \in A\}} K_h\left(d(i,j)\right)$. If this is the first iteration of the algorithm, set $W_i = 1$, $\forall i$.
4. Set $i^* = \arg\max_{\{i \notin A\}} W_i |(\mathbf{X}\beta - \mathbf{y})^T \mathbf{x}_i|$.
5. If $|x_{i^*}^T(\mathbf{X}\beta - \mathbf{y})| < \epsilon$ then stop, else set $A = A \cup i^*$ and go to step 2.

TABLE 2
CaSpaR: Clustered and Sparse Regression

We then select the next predictor i^* using a weighted criteria: $W_j(\mathbf{X}\beta - \mathbf{y})^T\mathbf{x}_j$. For most familiar kernels, such as a gaussian kernel or an Epanechnikov kernel, this has the effect of boosting the criteria for predictors "near" those already included in the model, and diminishing the criteria for those "far away". For practical application, we recommend a mixture of a familiar kernel, such as a boxcar or Epanechnikov, and a uniform distribution. This mixture, which we call the Stetson kernel, introduces an additional mixing parameter α:

$$K_{h,\alpha}(x) = \alpha + (1 - \alpha)K_h\left(d(x)\right) \tag{7}$$

where K_h is a kernel such as a boxcar, Epanechnikov, or Gaussian. An example of this kernel appears in Figure 1. We particularly recommend the Epanechnikov or the boxcar kernel, since these kernels have no impact at all on predictors outside of their bandwidth, and so $W_i = \alpha$ for these predictors. While this usually makes no difference in predictor selection, it simplifies precise computation and interpretation.

The advantage of the Stetson kernel is that this mixture allows multiple groups of predictors to appear in the sparsity structure. If we were to instead only use a familiar kernel, then we would have that $W_j = 0$ (or extremely small) for those j far enough away from predictors already included in the model. This has the effect of allowing only a single group in the sparsity structure, built around the first selected predictor. This is often not desirable; in most applications we wish to allow multiple groups. The Stetson kernel avoids this problem. The uniform part of the Stetson kernel allows for new predictor groups to enter the model. The kernel part of the mixture encourages clustering around predictors already included in the model.

3.1. *Tuning Parameters.* CaSpaR has three tuning parameters: ϵ, h and α. The parameter ϵ controls the sparsity of the fitted model. The parameters

h and α control the amount of structure in the estimated support. For the Stetson kernel, as the bandwidth h decreases, the groups of predictors become more tightly grouped. As α increases, new clusters are allowed to form more easily. In the special case where $\alpha = 1$, the method reduces to the usual forward stepwise regression method. Let $CV(\epsilon, h, \alpha)$ denote the cross-validation score. We choose the parameters by minimizing $CV(\epsilon, h, \alpha)$. Note that since small changes in h or α do not effect the order of predictor selection, this tuning can be accomplished using a simple grid search.

4. Results. We now return to our application to HIV drug resistance. Our data set consists of many amino acid sequences, all 99 amino acids in length. There are twenty known amino acids, so each position in these sequences has twenty possible values. In order to define predictors, we look across all of the sequences, and record the amino acids that appear in each position. We refer to these predictors as mutations.

Since each mutation has an associated position, we can then define a distance between predictors as the absolute difference of their positions. Note this means that the mutations that occur at the same position are distance 0 from each other.

We compare CsSpaR to forward stepwise regression and LASSO models. For the former, we use the Stetson kernel and cross validation to choose all the tuning parameters. We perform a grid search over $\alpha = \{0, 0.1, 0.2, ..., 1\}$, and over $h = \{1, 2, 3, 4\}$ to find the optimal tuning parameters.

We present a summary of our results in Table 3. Compared to stepwise regression, CaSpaR has comparable mean-squared-error (MSE) and number of mutations selected. In most cases, CaSpaR selects a few more mutations and has a slightly lower MSE. The LASSO generally does better in terms of MSE, but includes many more mutations. These results are complicated and cumbersome to interpret as a model of resistance. Overall, CaSpaR gives relatively sparse models, as desired.

See Figure 2 for comparisons of the sparsity structure in four of the drugs. If we compare the sparsity patterns of the stepwise and CaSpaR solutions, we see that CaSpaR gives more clustered solutions, as expected. As mentioned before, CaSpaR and stepwise regression select about the same number of mutations. The clustered CaSpaR solutions, however, select mutations from fewer positions than stepwise regression. The CaSpaR models therefore give a comparable level of prediction accuracy and sparsity, while also having a better biological interpretation. As previously stated, we believe that these clusters correspond to a functional area of the protein. The CaSpaR models therefore have a concise interpretation.

TABLE 3

Summary of results across all models and drugs. For each model, we give the mean-square-error, as well as the number of mutations (predictors) selected in parentheses. We see that CaSpaR is comparable to forward stepwise regression in terms of MSE, with about the same number of predictors included in the model. The LASSO does better in MSE, but includes many more mutations than either stepwise method. As we previously noted, neither forward stepwise regression nor the LASSO allows for a structured sparse solution.

Drug Name	Stepwise	CaSpaR	LASSO
APV	0.514 (7)	0.477 (14)	0.422 (51)
ATV	0.588 (6)	0.494 (11)	0.477 (39)
IDV	0.541 (13)	0.58 (10)	0.449 (77)
LPV	0.614 (5)	0.507 (15)	0.518 (35)
NFV	0.65 (19)	0.637 (22)	0.661 (40)
RTV	0.659 (8)	0.714 (5)	0.57 (58)
SQV	0.426 (31)	0.508 (21)	0.447 (63)

5. Simulation Study. We next report the results of a simulation study. We show that CaSpaR recovers a structured sparsity pattern more effectively than forward stepwise regression and LASSO. In this experiment we use the R function glmnet to obtain the LASSO solution. For CaSpaR, we use a Stetson kernel, and tune the parameters with a grid of $h = \{1, 2, 3, 4\}$, and $\alpha = \{.1, .2, \ldots, 1\}$. To measure the performance of each method we use recovery error

$$\text{Recovery Error} = \frac{||\widehat{\beta} - \overline{\beta}||_2^2}{||\overline{\beta}||_2^2} \tag{8}$$

Where $\widehat{\beta}$ is coefficient estimated by the method, and $\overline{\beta}$ is the true coefficient vector. For each method, we use 10-fold cross validation to choose all tuning parameters and stopping times.

We simulate an $n \times p$ data matrix $\mathbf{X}$ with $p = 250$ columns. Each entry of $\mathbf{X}$ is an i.i.d. draw from a standard normal distribution. We generate true coefficient vector $\overline{\beta}$ so that it has 7 groups of 5 nonzero coefficients, randomly placed. Thus, there are 35 nonzero entries in $\overline{\beta}$. Within each nonzero group, we set one entry of β equal to 6, and the rest equal to 3 (see the top panel of Figure 4 for a display of a sample coefficient vector). We then randomize the signs of the nonzero entries. We add independent gaussian noise with variance 1 to the simulated response.

To compare the three methods, we increase n from 50 to 150 and compare the recovery errors of the three methods. See Figure 3 for a graphical comparison of the recovery errors. CaSpaR is able to give near optimal per-

formance with fewer data points than the other methods. An example of the differences in performance between the three method on a single simulated data set ($n = 100$) is given in Figure 4. CaSpaR recovers the signal well, while the other two methods do not. We therefore conclude that for CaSpaR can reconstruct sparse signals more effectively than stepwise regression or the LASSO.

6. Theoretical Properties. In this section, we discuss the theoretical properties of CaSpaR. We begin by explaining how CaSpaR relates to other methods.

6.1. *Related Work.* There are several existing regression methods that take into account structure as well as sparsity. Yuan and Lin (2006) introduced the *grouped lasso* which allows only groups of predictors to be selected at once. This is desirable when these groups represent closely linked predictors — such as a set of predictors that code the levels of a multi-level factor predictor. Since this method modifies the LASSO, it can be cast as a convex minimization problem. However, these groups have to be predefined, and the method does not allow for overlap between groups, making this method somewhat inflexible.

Huang, Zhang and Metaxas (2009) introduced an algorithm called StructOMP that modifies forward stepwise regression (also known as orthogonal matching pursuit or OMP). Here, the desired sparsity structure is encoded as a set of blocks, each of which is assigned a cost. The algorithm proceeds by greedily adding blocks one at a time to reduce the loss, scaled by the cost of the added block. StructOMP allows for very flexible sparsity structures. In particular, it can approximate a general class of sparsity structures the authors term as graph sparsity, which we discuss in section 6.2.

Recent work by Jacob, Obozinski and Vert (2009) relating to the grouped lasso extend the possible group structures to include overlapping groups. Like StructOMP, the overlapping group penalty can produce models that approximately follow graph sparsity. This approach has the advantage of being a convex minimization problem. As we discuss in the next section, for graph sparsity, this method, like StructOMP, gives only an approximation to graph sparsity due to computational considerations.

6.2. *Graph Sparsity.* Graph sparsity is a specific type of structured sparsity introduced by Huang, Zhang and Metaxas (2009). Consider a graph G, whose nodes include the set $\mathcal{I} = \{1, 2, \ldots, p\}$. Thus, each predictor is a node of G, but for generality we allow other nodes to be in the graph as well. We then define the neighborhood of a node v as the set of nodes with an edge

connecting it to v. More generally, we could allow for k-neighborhoods — the set of all nodes with a path of at most k edges connecting it to v. We then consider a sparsity structure where the important predictors appear within neighborhoods, or a series of connected neighborhoods.

For example, consider a grid graph, such as in the case of a pixelated image. Each pixel is connected to four neighbors, one to each cardinal direction. The sparsity structure for this graph connects visually related components in the image.

CaSpaR can approximate graph sparsity by choosing the appropriate distance function and bandwidth. Given a graph G , the distance function can be defined in terms of the graph:

$$d(i,j) = \min\{\text{Length of paths from } i \text{ to } j, \text{ as defined by } G\} \tag{9}$$

More generally, each edge can be weighted, and $d(i,j)$ can be the minimal weighted path length. We then can define neighborhood size via the bandwidth h. For the Stetson kernel, the mixing parameter α controls the number of connected neighborhoods, where $\alpha = 0$ allows only one. In the image example, we can define $d(i,j)$ as above. Then, with $h \in (1,2)$, CaSpaR would find a sparsity structure of connected pixels.

CaSpaR is a very flexible way to approximate graph sparsity. First, it allows for neighborhoods to be locally defined through the bandwidth while still allowing neighborhoods to grow arbitrarily large as the method proceeds. Second, when used with the Stetson kernel, CaSpaR allows the user to control the degree to which graph sparsity is adhered via the mixing parameter α.

In comparison, the algorithms for the StructOMP of Huang, Zhang and Metaxas (2009) and graph lasso of Jacob, Obozinski and Vert (2009) approximate graph sparsity by constructing a set of node neighborhoods, based on the graph structure. These then generate a set of blocks or groups, which are then used in the OMP or group LASSO framework, respectively. However, to control the computational cost, they limit the neighborhood size used to make these blocks or groups. With CaSpaR, which grows neighborhoods instead of seeking to add them all at once as a group or block, this is not necessary. While these algorithms can handle large groups or blocks, they can only do so at significant computational cost.

Further, in StructOMP, there is no clear way to control the degree to which graph sparsity is followed in the solution. The blocks are each assigned a cost, but this cost is relatively restrictive. In graph LASSO, the group penalty is controlled by a parameter λ, just as with the ℓ_1 penality

in the LASSO. However, the group penalty controls sparsity as well as the structure, so as λ decreases, the model becomes less sparse as well as less structured. A separate ℓ_1 penalty could allow the model to be controlled for sparsity and structure separately.

6.3. *Relation to Stepwise Regression.* CaSpaR is closely related to forward stepwise regression. Indeed, with $\alpha = 1$ CaSpaR reduces to forward stepwise regression. Therefore, as long we consider $\alpha = 1$ when picking parameters, we always consider the forward stepwise regression solution. Consequently we have a loose guarantee that CaSpaR does no worse than forward stepwise regression. Moreover, we expect some theoretical results relating to forward stepwise regression can be adapted to CaSpaR.

6.4. *Consistency.* We now explain how a result in Zhang (2009) on stepwise regression can be adapted to CaSpaR. Under assumptions about the data matrix and the response, it can be shown that, with high probability the forward stepwise procedure stops with all correctly selected predictors — i.e. all the nonzero entries of the final $\widehat{\beta}$ — are also nonzero in the true target β. Note that there may be additional "false negatives". Moreover, if all of the target coefficients are above a threshold set by the noise level, then the entire sparsity pattern is captured exactly.

We closely follow the proof in Zhang (2009). This result requires more conditions than the similar result for stepwise regression. However, since we assume that we have an oracle set of tuning parameters $\{\alpha, h\}$, the assumptions are not too harsh. For ease of reference, we use notation similar to Zhang (2009).

We have an $n \times p$ matrix X consisting of p length n-vectors $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_p\}$, and n-vector $\mathbf{y}$. We assume that there is a target $\overline{\beta} \in \mathbb{R}^p$, such that:

$$\mathbb{E}\mathbf{y} = X\overline{\beta}. \tag{10}$$

This assumption means that the linear model is correct. It also roughly means there is a target coefficient vector $\overline{\beta}$ that estimates $\mathbf{y}$ wells, relative to the noise level. For both stepwise and CaSpar methods, we define $\beta^{(k)}$ as the coefficient vector after the kth step. Recall the definition of the support of a vector:

$$\text{supp}(\beta) = \{j : \beta_j \neq 0\}. \tag{11}$$

We then define $F^{(k)} = \text{supp}(\beta^{(k)})$, $\overline{F} = \text{supp}(\overline{\beta})$. Let

$$\widehat{\beta}_{\mathbf{X}}(F, \mathbf{y}) = \arg\min_{\beta \in \mathbb{R}^{|} } ||\mathbf{X}\beta - \mathbf{y}||_2^2 \quad \text{subject to: } \text{supp}(\beta) \subseteq F. \tag{12}$$

Finally, we define two technical quantities:

$$\mu_X(\overline{F}) = \max_{j \notin \overline{F}} ||(X_{\overline{F}}^T X_{\overline{F}})^{-1} X_{\overline{F}}^T \mathbf{x}_j||_1, \tag{13}$$

and

$$\rho_X(\overline{F}) = \inf \left\{ \frac{1}{n} ||X\beta||_2^2 / ||\beta||_2^2 : \text{supp}(\beta) \subset \overline{F} \right\}. \tag{14}$$

For CaSpar, we define a distance measure on our predictor index $1, 2, \ldots, p$: $d(i, j)$. We assume that we are using a boxcar kernel, or a Stetson kernel with a boxcar kernel: $K_{h,i}(j) = I_{d(i,j)<h}$. We then define the following set, which represents the candidate predictors — predictors not already included in the model —"underneath" the kernel:

$$\mathbb{A}^{(k)} = \left\{ j : d(i, j) < h, j \notin F^{(k)} \right\}. \tag{15}$$

It follows that

$$W_j = \begin{cases} \alpha + (1 - \alpha)/k & : j \in \mathbb{A}^{(k)} \\ \alpha & : j \notin \mathbb{A}^{(k)}. \end{cases} \tag{16}$$

Finally, recall that we have ϵ as the stopping criteria for CaSpar. If at step k we select $\mathbf{x}_{i(k)}$ as the next predictor to be included in the model, then if

$$|\mathbf{x}_{i(k)}^T (X\beta^{(k-1)} - \mathbf{y})| \leq \epsilon, \tag{17}$$

the CaSpar stops at step $k - 1$.

THEOREM 1. *Suppose that:*

1. $\frac{1}{n}||\boldsymbol{x}_j||_2^2 = 1 \ \forall j \in 1, 2, \ldots, p$
2. $\exists \overline{\beta} \in \mathbb{R}^p$, *with* $\overline{F} = supp(\overline{\beta})$ *s.t.* $\boldsymbol{y} = X\overline{\beta}$
3. $\mu_X(\overline{F}) < 1$
4. $\rho_X(\overline{F}) > 0$
5. *The elements of* $\boldsymbol{y}$; $[y_i]_{i=1,2,\ldots,n}$ *are independent sub-guassian random variables:* $\exists \sigma > 0$ *s.t.* $\forall i, \forall t \in \mathbb{R}, \mathbb{E}e^{t(y_i - \mathbb{E}y_i)} \leq e^{\sigma^2 t^2/2}$.
6. *Given* $\eta \in (0, 1)$, *let the stopping criteria satisfy:*

$$\epsilon > \frac{1}{1 - \mu_X(\overline{F})} \sigma \sqrt{2 \log(2p/\eta)}.$$

7. *There are* $\{\alpha, h\}$ *such that for each* k*, at least one of the following conditions hold:*
   (a) $\frac{\max_{j \notin \overline{F}} |\boldsymbol{x}_j^T (X\beta^{(k-1)} - \boldsymbol{y})|}{\max_{i \in \overline{F}} |\boldsymbol{x}_i^T (X\beta^{(k-1)} - \boldsymbol{y})|} < \alpha$
   (b) $\mathbb{A}^{(k-1)} \subseteq \overline{F}$
   (c) $\mathbb{A}^{(k-1)} \supseteq \overline{F}$

Then, when the procedure stops at step $k-1$*, with probability greater than* $1 - 2\eta$*, the following hold:*

1. $F^{(k-1)} \subset \overline{F}$
2. $|\overline{F} - F^{(k-1)}| \leq 2\left|\{j \in \overline{F} : |\overline{\beta}_j| < 3\epsilon\rho_X(\overline{F})^{-1}/\sqrt{n}\}\right|$
3. $||\beta^{(k-1)} - \widehat{\beta}_{\boldsymbol{X}}(\overline{F}, \boldsymbol{y})||_2 \leq \epsilon\rho_X(\overline{F})^{-1}\sqrt{|\overline{F} - F^{(k-1)}|/n}$
4. $||\widehat{\beta}_{\boldsymbol{X}}(\overline{F}, \boldsymbol{y}) - \overline{\beta}||_\infty \leq \sigma\sqrt{2\log(2|\overline{F}/\eta)/(n\rho_X(\overline{F}))}$

We omit the proof since it is very similar to the proof in Zhang (2009).

6.4.1. *Discussion of the Result.* The theorem states that when the procedure stops: (1) the selected predictors have truly nonzero $\overline{\beta}_i$; (2) the number of false negatives is bounded by the number of small truly nonzero $\overline{\beta}_j$ — relative to the noise level; (3) the estimator is close to the best possible β, which is estimated in the presence of noise using all the truly nonzero predictors; and (4) the difference between the best estimate in the presence of noise and that of the true $\overline{\beta}$ is bounded.

The proof of this result is based on induction at each step of the procedure. The extra conditions are motivated by the following analysis. We denote any predictor for which $\overline{\beta}_j = 0$ as a *noise* predictor and any predictor for which $\overline{\beta}_j \neq 0$ as a *signal* predictor. When we consider adding a predictor in a step of forward stepwise regression, we consider two quantities:

$$\max_{j \notin \overline{F}} |\mathbf{x}_j^T (X\beta^{(k-1)} - \mathbf{y})|, \tag{18}$$

$$\max_{i \in \overline{F}} |\mathbf{x}_i^T (X\beta^{(k-1)} - \mathbf{y})|. \tag{19}$$

These are, respectively, proportional to the maximum correlation between the current residuals and a noise predictor and the maximum correlation between the current residuals and a signal predictor. We refer to these two predictors as the "best" signal predictor and "best" noise predictor.

For CaSpaR, we must consider how the weights applied to these quantities affect the analysis. We therefore consider the cases where: (a) the best signal

predictor and the best noise predictor are in $\mathbb{A}^{(k)}$, (b) neither the best signal predictor nor the best noise predictor are in $\mathbb{A}^{(k)}$, or (c) the best signal predictor is in $\mathbb{A}^{(k)}$ but the best noise predictor is not, or (d) the best noise predictor is in $\mathbb{A}^{(k)}$ but the best signal predictor is not. Except for scenario (d), the original result for stepwise regression holds. We therefore make additional assumptions to ensure that case (d) does not occur. Those conditions are:

1. The ratio of the criteria for the best noise predictor to the best signal predictor is less than α.
2. All of the predictors under the kernel are signal predictors.
3. All of the signal predictors are under the kernel.

The first ensures that in case (d) the correlation between the signal predictor is large enough to still be selected even in this case. Because the weights W_j only depend on membership in $\mathbb{A}^{(k)}$, the second and third conditions ensure that case (d) never occurs: the second means there are only signal predictors in $\mathbb{A}^{(k)}$, and the third means that there are no signal predictors not in $\mathbb{A}^{(k)}$.

These assumptions are fairly mild, especially if we have a strong belief that $\text{supp}(\overline{\beta})$ is truly structured. We propose that the first condition holds for early steps of CaSpar. We can reasonably assume that it is possible for an oracle α to be such that the signal is sufficiently dominant over noise. The last two conditions should hold for later steps of the algorithm: enough points within each cluster have already been discovered so that it only remains to fill in the clusters.

7. Conclusion. We introduced a new method, CaSpaR, that allows us to build sparse regression models where we have some additional information about the structure of the sparsity pattern. We presented an application as well as a simulation study that show the method performs differently than the most popular sparse regression techniques. We discussed the general concept of graph sparsity, and showed that our method provides a flexible way to approximate graph sparsity.

Our simulation study suggests that under structured sparsity conditions, CaSpaR can recover the true target with less data than standard techniques. This motivates future work to show that this property has a theoretical basis. Other topics of interest include adding backward steps to the CaSpaR algorithm as well as an extension to a convex minimization procedure, which may have some computational advantages over the stepwise procedure.

References.

Barron, A. R., Cohen, A., Dahmen, W. and DeVore, R. A. (2008). Approximation and learning by greedy algorithms. *Ann. Statist.* **36** 64-94.

Beerenwinkel, N., Daumer, M., Oette, M., Korn, K., Hoffmann, D., Kaiser, R., Lengauer, T., Selbig, J. and Walter, H. (2003). Geno2pheno: estimating phenotypic drug resistance from HIV-1 genotypes. *Nucleic Acids Research* **31** 3850-3855.

Bunea, F., Tsybakov, A. and Wegkamp, M. (2007). Sparsity oracle inequalities for the Lasso. *Electronic Journal of Statistics* **1** 169-194.

Greenshtein, E. and Ritov, Y. (2004). Persistency in high dimensional linear predictor-selection and the virtue of over-parametrization. *Bernoulli* **10** 971-988.

Huang, J., Zhang, T. and Metaxas, D. (2009). Learning with structured sparsity. In *ICML '09: Proceedings of the 26th Annual International Conference on Machine Learning* 417–424. ACM, New York, NY, USA.

Jacob, L., Obozinski, G. and Vert, J.-P. (2009). Group lasso with overlap and graph lasso. In *ICML '09: Proceedings of the 26th Annual International Conference on Machine Learning* 433–440. ACM, New York, NY, USA.

Liu, D., Lin, X. and Ghosh, D. (2007). Semiparametric regression for multi-dimensional genomic pathway data: Least square kernel machines and linear mixed models. *Biometrics* **63** 1079–1088.

Petropoulos, C. J., Parkin, N. T., Limoli, K. L., Lie, Y. S., Wrin, T., Huang, W., Tian, H., Smith, D., Winslow, G. A., Capon, D. J. and Whitcomb, J. M. (2000). A novel phenotypic drug susceptibility assay for human immunodeficiency virus type 1. *Antimicrobial Agents and Chemotherapy* **44** 920-928.

Rhee, S.-Y., Gonzales, M. J., Kantor, R., Betts, B. J., Ravela, J. and Shafer, R. W. (2003). Human immunodeficiency virus reverse transcriptase and protease sequence database. *Nucleic Acids Research* **31** 298-303.

Rhee, S.-Y., Taylor, J., Wadhera, G., Ben-hur, A., Brutlag, D. L. and Shafer, R. W. (2006). Genotypic predictors of human immunodeficiency virus type 1 drug resistance. In *Proceedings of the National Academy of Sciences of the United States of America*.

Tibshirani, R. (1994). Regression Shrinkage and Selection Via the Lasso. *Journal of the Royal Statistical Society, Series B* **58** 267–288.

Tibshirani, R. and Saunders, M. (2005). Sparsity and smoothness via the fused lasso. *J. R. Statist. Soc. B* **67** 91108.

Wainwright, M. J. (2007). Information-theoretic limits on sparsity recovery in the high-dimensional and noisy setting. `arXiv:math/0702301v2 [math.ST].`

Yuan, M. and Lin, Y. (2006). Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society, Series B* **68** 49–67.

Zhang, T. (2009). On the Consistency of Feature Selection using Greedy Least Squares Regression. *J. Mach. Learn. Res.* **10** 555–568.

Zhang, J., Rhee, S.-Y., Taylor, J. and Shafer, R. W. (2005). Comparison of the precision and sensitivity of the antivirogram and PhenoSense HIV drug susceptibility assays. *Journal of Aquired Immune Deficiency Syndrones* **38** 439-444.

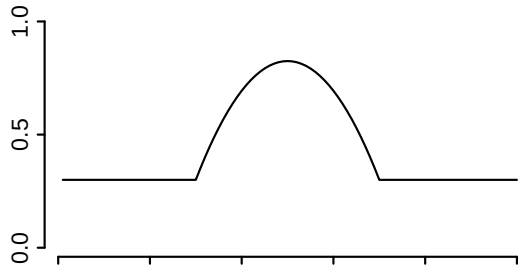

FIG 1. *The Stetson kernel, with an Epanechnikov kernel.*

CARNEGIE MELLON UNIVERSITY
DEPARTMENT OF STATISTICS
PITTSBURGH, PA 15213 USA
E-MAIL: dperciva@stat.cmu.edu
roeder@stat.cmu.edu
larry@stat.cmu.edu

CARNEGIE MELLON UNIVERSITY
SCHOOL OF COMPUTER SCIENCE
PITTSBURGH, PA 15213 USA
E-MAIL: Roni.Rosenfeld@cs.cmu.edu

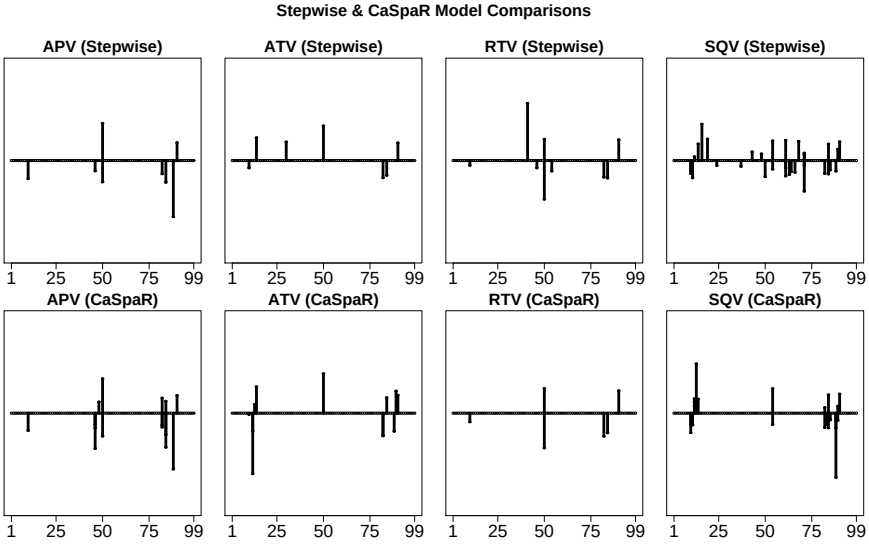

FIG 2. *Comparison of stepwise and CaSpaR models across four drugs: APV, ATV, RTV, and SQV. Each plot gives the coefficients for the selected mutation predictors, versus the locations of these mutations in the protein sequence. Each vertical line is a the coefficient for a mutation predictor - note some sequence locations can have multiple mutations.*

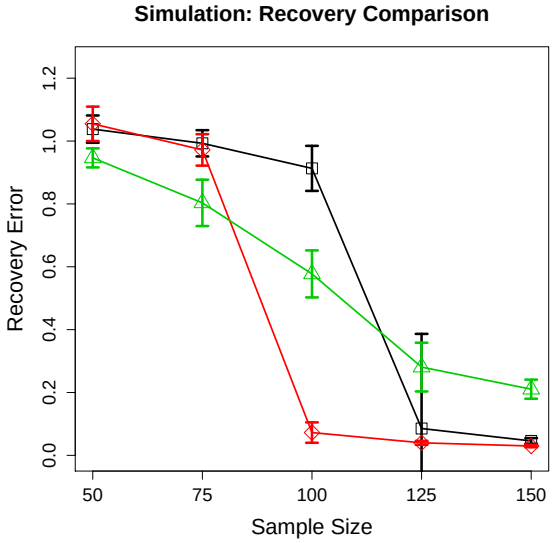

FIG 3. *Recovery error* $\left(||\widehat{\beta} - \beta||_2^2/||\beta||_2^2\right)$ *on simulated data with 1-dimensional structured sparsity. Black points: stepwise regression; Green points: LASSO; Red points: CaSpar. We can see that CaSpaR achieves a much better recovery rate than either of the other two methods with less data.*

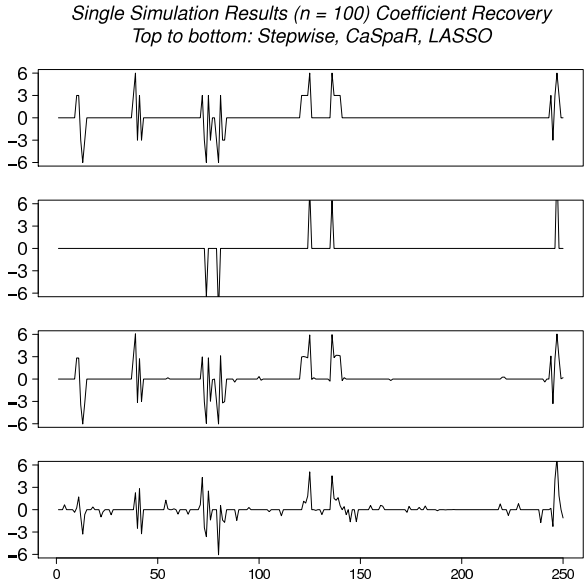

FIG 4. *Recovery of coefficients for a single simulated data set. The top panel displays the true coefficient vector. The recovery errors for the three methods are: Stepwise = 0.8483164, CaSpaR = 0.05876328, LASSO = 0.5416991.*