Nonsingular subsampling for S-estimators with categorical predictors

Manuel Koller
Seminar for Statistics, ETH Zurich, Zurich, Switzerland

August 29, 2012

Abstract

An integral part of many algorithms for S-estimators of linear regression is random subsampling. For problems with only continuous predictors simple random subsampling is a reliable method to generate initial coefficient estimates that can then be further refined. For data with categorical predictors, however, random subsampling often does not work, thus limiting the use of an otherwise fine estimator. This also makes the choice of estimator for robust linear regression dependent on the type of predictors, which is an unnecessary nuisance in practice. For data with categorical predictors random subsampling often generates singular subsamples. Since these subsamples cannot be used to calculate coefficient estimates, they have to be discarded. This makes random subsampling slow, especially if some levels of categorical predictors have low frequency, and renders the algorithms infeasible for such problems. This paper introduces an improved subsampling algorithm that only generates nonsingular subsamples. We call it *nonsingular subsampling*. For data with continuous variables it is as fast as simple random subsampling but much faster for data with categorical predictors. This is achieved by using a modified LU decomposition algorithm that combines the generation of a sample and the solving of the least squares problem.

1 Introduction

In a nutshell, a random subsampling based algorithm for S-estimates of linear regression does the following. It takes a random sample of the observations of size equal to the number of predictors p, resulting in a square design matrix, solves a "least squares" problem on this reduced data set (which gives 0 residuals in this case) and refines the resulting parameter estimate using a redescending M-estimate of regression with a simultaneous scale on the whole data set. This is repeated for a pre-specified number of times. This final S-estimate is then taken to be the one that resulted in the smallest scale estimate. Definitions of these methods are given in Section 2.

The random subsampling algorithms described above work well for continuous data. For categorical data, however, they are often slow or fail completely. The problem lies in the generation of *good* subsamples, i.e., square subsamples that contain no collinearities. Otherwise the subsample has to be discarded, since it does not generate a well defined starting value. We illustrate the problem by means of a simple example. Imagine a simple one-way ANOVA with 3 groups and 3 repetitions each. Using treatment contrasts to encode the grouping structure, we get the following least squares problem.

$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \\ y_8 \\ y_9 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \end{pmatrix} + \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \\ \varepsilon_4 \\ \varepsilon_5 \\ \varepsilon_6 \\ \varepsilon_7 \\ \varepsilon_8 \\ \varepsilon_9 \end{pmatrix}$$

In this example, the size of the subsample is 3. It is clear that the coefficients $\underline{\beta}$ can only be estimated if at least one observation of each group is part of the subsample. From the total number of possible subsamples, 84, only 27 correspond to a nonsingular subsample. Therefore, even in this very simple example, the probability of discarding a subsample because of collinearities is about two thirds. This probability is much higher when some levels of factors have low frequencies. Then an excessively large number of subsamples is required, rendering the simple random subsampling algorithms unfeasible for such data sets.

Instead of discarding the whole sample, we propose to solve this problem by dropping the observation causing the singularity and continue sampling. We call this refined subsampling strategy *nonsingular subsampling*.

Maronna and Yohai (2000) have proposed another approach on this problem. They solve it using a combination of M and S-estimates, called *M-S-estimates*. The categorical part is estimated separately with M-estimates. The continuous part is estimated using S-estimates. Their approach works well for data that contains only purely categorical and continuous variables. For interactions of categorical and continuous variables, however, it is not clear how to split the data. Using the M-estimate for the interaction will result in a loss of robustness, while using the S-estimate will again produce singular subsamples.

In the next section we introduce the notation and provide definitions for all methods used. Then we develop the basic algorithms for M-estimates and S-estimates. In Section 4 we explain the nonsingular subsampling algorithm. Finally, we conclude with Section 5.

2 Notation and definitions

Consider the standard multiple linear regression model,

$$y_i = \underline{x}_i^\intercal \underline{\beta} + \varepsilon_i, \quad \varepsilon \sim \mathcal{N}\left(0, \sigma^2\right), \quad i = 1, \ldots, n,$$

with e_i i.i.d., independent of $\underline{x}_i$ and $\underline{\beta}$ of length p. Throughout this text we assume that the design matrix $\boldsymbol{X}$, combining all $\underline{x}_i$ into one large matrix, is of full rank p. We denote the residuals as $r_i(\underline{\hat{\beta}}) = y_i - \underline{x}_i^\intercal \underline{\hat{\beta}}$.

Simultaneous M-estimates of regression and scale are defined as the solutions $\underline{\hat{\beta}}$ and $\hat{\sigma}$ to,

$$\sum_{i=1}^{n} \psi\left(\frac{r_i\left(\underline{\hat{\beta}}\right)}{\hat{\sigma}}\right) \underline{x}_i = \underline{0}, \tag{1}$$

$$\frac{1}{n}\sum_{i=1}^{n} \chi\left(\frac{r_i\left(\underline{\hat{\beta}}\right)}{\hat{\sigma}}\right) = \kappa, \tag{2}$$

where κ is a tuning constant, χ is a so-called ρ-function, and ψ is a ψ-function. A ρ-function, as defined in Maronna et al (2006), is assumed to be a nondecreasing function of $|r|$, with $\rho(0) = 0$ and strictly increasing for $r > 0$ where $\rho(r) < \rho(\infty)$. If a ρ-function is bounded, we assume $\rho(\infty) = 1$ and call it a *redescending* ρ-function. A ψ-function is the derivative of a ρ-function and is usually standardized so that $\psi'(0) = 1$. A solution $\hat{\sigma}$ to (2) for a given vector $\underline{r}$, $\hat{\sigma}(\underline{r})$, is called *M-estimate of scale*. The solutions to (1) for redescending ρ-functions are local minima of the corresponding optimization problem and are called *redescending M-estimates of regression*.

S-estimates of regression are the parameter value $\underline{\hat{\beta}}$ that minimizes the M-estimate of scale $\hat{\sigma} = \hat{\sigma}(\underline{r}(\underline{\hat{\beta}}))$ of the associated residuals,

$$\underline{\hat{\beta}} = \underset{\beta}{\operatorname{argmin}}\, \hat{\sigma}\left(\underline{r}\left(\underline{\beta}\right)\right). \tag{3}$$

The maximal breakdown point $(1-p/n)/2$ of the S-estimate is attained when using $\kappa = (1-p/n)/2$. Note that solutions of (3) are always also the solution to a simultaneous M-estimation of regression and scale problem. For an introduction to M-estimation and details about S-estimates, we refer to Maronna et al (2006).

A *LU decomposition* of a matrix $\boldsymbol{A}$ is defined as the product of a unit lower-triangular matrix $\boldsymbol{L}$ and an upper-triangular matrix $\boldsymbol{U}$. Such a decomposition does not always exist, therefore in practice one usually uses an *LU decomposition with partial pivoting*. This is an LU decomposition of the matrix $\boldsymbol{A}$ where the rows have been reordered by a permutation matrix $\boldsymbol{P}^\intercal$. The ordering is chosen in a way that minimizes numerical errors. The decomposed matrix can then expressed as,

$$\boldsymbol{A} = \boldsymbol{PLU}.$$

This form is useful to solve systems of linear equations $\boldsymbol{A}\underline{\beta} = \underline{y}$. Using the LU decomposition this problem can be solved directly using a forward and a backward elimination.

3 Basic algorithms

First we describe algorithms to compute M-estimates of regression and M-estimates of scale. The equations (1) and (2) are preferably solved using iterative reweighting algorithms. As outlined in Maronna et al (2006), one can rewrite (1) and (2) to take the form of a weighted least squares problem and a weighted sample variance. The weights then correspond to the respective robustness weights. Starting from some suitable initial estimate, the iterative reweighting algorithm then simply repeats the following two steps until convergence. The first step consists of the computation of the weights using the results from the previous iteration. In the second step the weighted problem is solved using the weights computed in the first step. Convergence of this algorithm is usually not a problem. For redescending M-estimates the algorithm converges to a local minimum. One can easily get an algorithm solving the simultaneous M-estimation of regression and scale problem by combining the two iterations.

We now come to the computation of S-estimates. Combining the above mentioned properties of S-estimates and M-estimates, we can derive a simple algorithm that we will later use as the basis for further improvements. See Algorithm 1 for a summary of the rest of the paragraph. Firstly, note that since any S-estimate is also a solution to (1), we can restrict the minimization to parameter vectors $\underline{\beta}$ that solve (1) and (2) simultaneously. Secondly, we use the iterative reweighting algorithm described above to generate such solutions from a set of initial estimates. Such a set of initial estimates may be found by solving all subproblems involving only p observations. Finally, we get the desired S-estimate as the solution resulting in the smallest scale estimate.

Since for subproblems with p observations the design matrix is square, the least squares problem reduces to a set of linear equations. It has only a defined solution if the design matrix is invertible, i.e., there are no collinearities. We may simply discard all singular subproblems. Note that the check for singularity does not require an additional step. If the system of linear equations of the subproblem can be solved we may continue, otherwise we discard the subsample.

Data: $\underline{y}$, $\boldsymbol{X}$.
Result: $\underline{\hat{\beta}}$, $\hat{\sigma}$.

```
σ̂ ← ∞
for all subsets J ⊂ {1, ..., n} of size p do
    if design matrix X_J contains no collinearities then
        β̲̂_0 ← X_J^{-1} y̲_J
        β̲̂_J, σ̂_J ← Solve (2) and (1) using iterative reweighting algorithm starting from β̲̂_0.
        if σ̂_J < σ̂ then
            σ̂ ← σ̂_J
            β̲̂ ← β̲̂_J
```

Algorithm 1: Basic algorithm for the computation of S-estimates.

This algorithm using exhaustive resampling, i.e., running over all possible subproblems, as shown in Algorithm 1, is of course only suitable for small problems. For large problems, it is neither feasible nor sensible to look at all the subsamples. Instead, it is enough to consider only set of random subsamples. Depending on the expected proportion of outliers, we can use simple

combinatorics to determine how many random subsamples are required to select at least one outlier-free subsample with a probability of, say, 0.999. The number of subsamples grows exponentially with p. Taking 1000 random subsamples has proven to work well in practice. Exact numbers are given in Table 5.3 of Maronna et al (2006). The same reference also summarizes many more optimizations. Worth mentioning is the paper by Salibian-Barrera and Yohai (2006) where they develop a complete strategy for computing S-estimates, dealing also with very large data sets.

4 Nonsingular subsampling

Simple random subsampling algorithms have the drawback that they cannot guarantee the generation of nonsingular subsamples, i.e., without collinearities. They work on a simple trial-and-error basis. In the following, we propose an algorithm that produces only nonsingular subsamples. The algorithm we propose has the advantage that it is much faster than simple random subsampling algorithms for hard problems, without sacrificing any time for easy problems.

The nonsingular subsampling algorithm merges the two steps of generating the random subsample and solving the system of linear equations. The latter consists of computing a LU decomposition (for a definition, see Section 2) and then solving two triangular linear systems of equations. Instead of generating the whole subsample at once, we propose to select observation by observation. A new observation is only added to the subsample if it is not collinear to the observations already selected. Proceeding in this way, we will always end up with a nonsingular subsample. The speed up is achieved by using a modified LU decomposition algorithm. It computes the LU decomposition observation by observation, without having to recompute any results of previously selected observations if one observation needs to be dropped. So if the random subsample is nonsingular in the first place, e.g., for continuous predictors, the algorithm does the same as the simple random subsampling algorithms. But for singular subsamples, we can avoid restarting from scratch, simply dropping the observation and continue with the next candidate is enough.

The modified LU decomposition algorithm is based on the so-called Gaxpy variant of the LU decomposition algorithm as found in Golub and Van Loan (1996). It is of the same complexity as other LU decomposition algorithms. The Gaxpy variant delays the computation of columns of $\boldsymbol{U}$ until they are actually needed. To compute the ith column of $\boldsymbol{L}$, we need only columns 1 to i of $\boldsymbol{U}$. In case a singularity is detected, it is therefore enough to only repeat the last step using a new observation / column. Results obtained prior to this step do not need to be recomputed.

The nonsingular subsampling algorithm is shown in Algorithm 2.

The numerical stability of the algorithm can be improved by using a matrix preconditioning technique that reduces the condition number of the design matrix. We propose to use a method called *equilibration*, described, e.g., in Demmel (1997). Tests have shown that it is enough to apply this method to the complete design matrix, even if only submatrices are used later on.

5 Conclusions

Current algorithms for S-estimates of linear regression have trouble coping with categorical predictors, especially if interactions of continuous and categorical predictors are involved. These issues can be avoided by using nonsingular subsampling instead of simple random subsampling. By merging the steps of generating the subsample and fitting the least squares problem, the new subsampling algorithm can generate nonsingular subsamples much more efficiently than simple random subsampling for data sets with categorical predictors. Comparing the runtimes of S-estimates using nonsingular subsampling and M-S-estimates showed only a modest increase of computing time (around 10%) even for quite large designs ($n = 8088, p = 340$, 2 of them continuous predictors). The nonsingular subsampling algorithm is implemented in the `lmrob` function of the R package `robustbase` from version `0.9-3`.

Data: $n \times p$ matrix $\boldsymbol{X}$, response vector $\underline{y}$, singularity treshold ε.
Result: Return code (0 for success, otherwise number of failing step), initial estimate $\underline{\hat{\beta}}$.
Initialize variables, pivoting table $\underline{p}$, selected subsample index vector $\underline{s}$
$\boldsymbol{U} \leftarrow \boldsymbol{0}$; $\boldsymbol{L} \leftarrow \boldsymbol{I}$; $\underline{p} \leftarrow 1:p$; $\underline{s} \leftarrow 1:p$; $k \leftarrow 1$
Permutate observations randomly
$\underline{t} \leftarrow \texttt{perm}(1:n)$
$\boldsymbol{A} \leftarrow \boldsymbol{X}^{\intercal}_{\underline{t},1:p}$
$\underline{y} \leftarrow \underline{y}_{\underline{t}}$
for j **in** 1 **to** p **do** ## Find non-singular subsample and calculate LU decomposition
  if $j == 1$ **then** $\underline{v}_{1:p} \leftarrow \boldsymbol{A}_{1:p,k}$
  else
    ## (Forward)solve to get required column of $\boldsymbol{U}$
    $\boldsymbol{U}_{1:j-1,j} \leftarrow \boldsymbol{L}^{-1}_{1:j-1,1:j-1}\boldsymbol{A}_{1:j-1,k}$
    $\underline{v}_{j:p} \leftarrow \boldsymbol{A}_{j:p,k} - \boldsymbol{L}_{j:p,1:j-1}\boldsymbol{U}_{1:j-1,j}$
  if $j < p$ **then**
    ## Find pivot
    $\mu \leftarrow \text{argmax}_{l=j}^{p} |\underline{v}_l|$
    if $|\underline{v}_\mu| \geq \varepsilon$ **then**
      ## Subsample is still non-singular
      $\underline{p}_j \leftarrow \mu$
      $\underline{s}_j \leftarrow k$
      ## Swap elements of $\underline{v}$ and rows of $\boldsymbol{A}$
      $\underline{v}_j \leftrightarrow \underline{v}_\mu$
      $\boldsymbol{A}_{j,k+1:n} \leftrightarrow \boldsymbol{A}_{\mu,k+1:n}$
      ## Update $\boldsymbol{L}$
      $\boldsymbol{L}_{j+1:p,j} \leftarrow \underline{v}_{j+1:p}/\underline{v}_j$
      ## Swap rows of $\boldsymbol{L}$
      $\boldsymbol{L}_{j,1:j-1} \leftrightarrow \boldsymbol{L}_{\mu,1:j-1}$
  if $|\underline{v}_j| < \varepsilon$ **then**
    ## Singularity detected: skip this column and try again if possible
    if $k < n$ **then**
      $k \leftarrow k + 1$
      Goto *6*
    else ## Return with an error
      return j
  $\boldsymbol{U}_{j,j} \leftarrow \underline{v}_j$
  $k \leftarrow k + 1$
Solve $X^{-1}_{\underline{s},1:p}\underline{y}_{\underline{s}}$ and undo pivoting
$\underline{\hat{\beta}} \leftarrow \boldsymbol{L}^{-\intercal}\boldsymbol{U}^{-\intercal}\underline{y}_{\underline{s}}$
for j **in** $p-2$ **to** 0 **do** $\underline{\hat{\beta}}_j \leftrightarrow \underline{\hat{\beta}}_{\underline{p}_j}$
return 0, $\underline{\hat{\beta}}$

Algorithm 2: Constrained subsampling using modified Gaxpy variant of LU decomposition. We use vector index notation to select subvectors and submatrices. The index vector $1:p-1$ in $\underline{v}_{1:p-1}$ indicates that an operation only acts on the elements 1 to $p-1$ of the vector.

References

Demmel J (1997) Applied Numerical Linear Algebra. Society for Industrial and Applied Mathematics

Golub GH, Van Loan CF (1996) Matrix computations, 3rd edn. The Johns Hopkins University Press, Baltimore

Maronna RA, Yohai VJ (2000) Robust regression with both continuous and categorical predictors. Journal of Statistical Planning and Inference 89(1–2):197 – 214, DOI 10.1016/S0378-3758(99)00208-6

Maronna RA, Martin RD, Yohai VJ (2006) Robust Statistics, Theory and Methods. Wiley, N.Y.

Salibian-Barrera M, Yohai V (2006) A fast algorithm for S-regression estimates. Journal of Computational and Graphical Statistics 15(2):414–427