Lasso2*: An S-PLUS library to solve regression problems while imposing an L^1 constraint on the parameters

J Lokhorst† B A Turlach‡ W N Venables§

December 2, 1999

Contents

†Department of Statistics, University of Adelaide, Adelaide SA 5005, Australia
‡Department of Statistics, University of Adelaide, Adelaide SA 5005, Australia
§CSIRO Marine Laboratories, PO Box 120, Cleveland QLD 4163, Australia

1 Introduction

Consider the usual regression setting where we have data $(x_{i1}, \ldots, x_{im}, y_i)$, $i = 1, 2, \ldots, n$, and the x_{ij}s are the regressor variables and y_i the response for the ith observation. In this situation, ordinary least squares tries to find the linear combination of the x_{ij}s that minimises the residual sum of squares. However, if m is large or there are high correlations among the regressor variables then the least-squares estimators often have high variance. Traditional methods of addressing this problem include ridge regression and subset selection (see, among others, Miller, 1990; Hocking, 1996; Draper and Smith, 1998).

Tibshirani (1996) proposes the "least absolute shrinkage and selection operator" (lasso) as an alternative method for handling this problem. This approach amounts to minimising the residual sum of squares under a constraint on the sum of the absolute values of regression coefficient estimates. That is, the following optimisation problem has to be solved:

$$\underset{\beta_1,\ldots,\beta_m}{\text{minimise}} \qquad \frac{1}{2}\sum_{i=1}^{n}\left(y_i - \sum_{j=1}^{m} x_{ij}\beta_j\right)^2 \tag{1a}$$

$$\text{subject to} \qquad \sum_{j=1}^{m}|\beta_j| \leq t, \tag{1b}$$

for some $t > 0$. More specifically, Tibshirani (1996) proposes standardising each regressor so that it has (sample) mean zero and (sample) variance one and standardising the dependent variable to have mean zero. This standardisation amounts to incorporating an intercept term that is orthogonal to all other regressors, not part of the penalty and estimated by the mean of the dependent variable.

Tibshirani (1996) showed that this procedure has some interesting properties. Essentially, it shrinks the ordinary least squares estimates towards zero, typically setting some of them to be equal to zero. Thus, it seems to behave as a compromise between subset selection and ridge regression and may therefore be a useful tool for variable selection. Code to fit models of the form (1) is provided by Tibshirani (1995).

This manual describes an alternative S-PLUS library for fitting models of type (1). It is based on the algorithm developed by (Osborne *et al.*, 1999) and the actual implementation in C is described in Turlach (1998). A homotopy approach to calculate all solutions of (1) sequentially is described in Osborne *et al.* (1998a). Osborne (1998) discusses the relationship between variable selection and trust regions for (nonlinear) optimisation routines.

We claim the algorithm discussed by Osborne *et al.* (1999) has several advantages over the algorithm used by Tibshirani (1995, 1996). For instance it can still be applied if there are more regressors than observations (Osborne *et al.*, 1998b).

The underlying engine of this library is described in Turlach (1998). However, the S-PLUS interface adds much more functionality. Our library uses the modelling features of S-PLUS. Also, the S-PLUS interface is designed for the more general case where some other variables in addition to the intercept term are accepted *à priori* as important regressors and are not included in the constraint. Some details are given in Section 2. Instead of discussing all routines of the library in detail, we shall give two examples in Section 3 and refer to the printout

of the help files in Section 6. Due to some design decisions there are a few incompatibilities with the code provided by Tibshirani (1995), these are discussed in Section 5.

2 Some Details

Assume that the regressor variables are split up into $(x_{i1}, \ldots, x_{ip_1})$ and $(z_{i1}, \ldots, z_{ip_2})$ and we use the 'lasso' technique to select among the z-regressors only. If β denotes the vector of parameters for the x-variables and γ the vector of parameters for the z-variables, then we may use (1) by placing a constraint on γ only.

The S-PLUS interface allows one further extension of this approach. Specifically, let $\mathbf{X}$ and $\mathbf{Z}$ denote the corresponding design matrices, $\mathbf{Y}$ the vector with the dependent variable and $\mathbf{W}$ $(n \times n)$ a non-negative, diagonal weight matrix (possibly the identity). Then the central fitting routine, `l1ce`, (L^1 **c**onstrained **e**stimation) of this library proceeds as follows.

- Calculate $\hat{\beta} = (\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}\mathbf{Y}$ and project the (weighted) other variables (both dependent and independent) orthogonal to the column space of $\mathbf{W}^{1/2}\mathbf{X}$. That is

$$\mathbf{Z}^\star = \left\{\mathbf{I} - \mathbf{W}^{1/2}\mathbf{X}(\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}^{1/2}\right\}\mathbf{W}^{1/2}\mathbf{Z} \quad \text{and}$$
$$\mathbf{Y}^\star = \left\{\mathbf{I} - \mathbf{W}^{1/2}\mathbf{X}(\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}^{1/2}\right\}\mathbf{W}^{1/2}\mathbf{Y}.$$

- Standardise the columns of $\mathbf{Z}^\star$ to have (sample) variance 1. (This standardisation may be suppressed.)

- Solve

$$\underset{\gamma}{\text{minimise}} \qquad (\mathbf{Y}^\star - \mathbf{Z}^\star\gamma)^T(\mathbf{Y}^\star - \mathbf{Z}^\star\gamma) \tag{2a}$$
$$\text{subject to} \qquad \|\gamma\|_1 \leq t. \tag{2b}$$

- Adjust $\hat{\gamma}$ to take into account any standardisation of $\mathbf{Z}^\star$ and adjust $\hat{\beta}$ to take the projection of $\mathbf{Z}$ orthogonal to $\mathbf{X}$ into account.

By default $\mathbf{W}$ is the identity matrix, $p_1 = 1$ and $x_{i1} = 1$ for $i = 1, \ldots, n$.

There are two possibilities for specifying t in (2b). If the user provides a *relative* bound, s, (which must be between zero (exclusive) and one (inclusive)), then the maximum effective value for t, namely $t_0 = \|(\mathbf{Z}^{\star T}\mathbf{Z}^\star)^{-1}\mathbf{Z}^{\star T}\mathbf{Y}^\star\|_1$ is calculated and $t = st_0$ is used as constraint. (Since t_0 is calculated using the S-PLUS commands `qr` and `qr.fitted` this option can also be used if there are more regressors than observations. It is however unclear in this case whether t_0 has any desirable properties.) The second possibility is for the user to provide an *absolute* bound t which is then used directly. In this latter case the argument `absolute.t = T` must be specified.

It should be noted that the approach leading to (2) yields in general a different result from that found by solving the problem

$$\underset{\beta,\gamma}{\text{minimise}} \qquad (\mathbf{Y} - \mathbf{X}\beta - \mathbf{Z}\gamma)^T(\mathbf{Y} - \mathbf{X}\beta - \mathbf{Z}\gamma) \tag{3a}$$

$$\text{subject to} \qquad \|\gamma\|_1 \leq t. \tag{3b}$$

It seems natural, however, first to project orthogonal (in the metric defined by weight matrix $\mathbf{W}$) to the space spanned by the columns of $\mathbf{X}$ and then to select those columns (variables) in $\mathbf{Z}^\star$ that best explain the remaining variance. (Of course this approach is a mild and natural generalisation of the way the intercept term alone is handled in Tibshirani (1996).)

3 Examples

In this section we demonstrate some of the routines of this library on two examples. The first uses the prostate data (Stamey *et al.*, 1989) which is also used in Tibshirani (1996) and Osborne *et al.* (1999); the second uses the Iowa wheat yield data from a multiple regression problem in Draper and Smith (1998) and shows that this methodology can have some surprising effects.

Prostate Data

The prostate data (Stamey *et al.*, 1989) is provided as a data frame called `Prostate` by this library. We shall not discuss in detail how the results reported in Tibshirani (1996) and Osborne *et al.* (1999) can be reproduced with this library. Due to some incompatibilities between our code and that of Tibshirani (1995) exact reproduction of these published results requires a major effort.

One of the incompatibilities is that Tibshirani (1995) returns parameters for the estimate on the "standardised" and the "unstandardised" scale of the regressors. Our implementation only returns the estimates on the scale on which the variables were specified. Hence, to get the parameter estimates mentioned in Tibshirani (1996) and Osborne *et al.* (1999) we would have to create a data frame in which the regressor variables are already centred. This can be done by the following commands

```
> p.mean <- apply(Prostate, 2, mean)
> pros <- sweep(Prostate, 2, p.mean, "-")
> p.std <- apply(pros, 2, var)
> pros <- sweep(pros, 2, sqrt(p.std), "/")
> pros[, "lpsa"] <- Prostate[, "lpsa"]
```

After loading the library `Lasso2` we can fit the model discussed by Tibshirani (1996) and Osborne *et al.* (1999) as follows; (to get intermediate results on the progress of the fitting, we may specify `trace = T`):

```
> l1ce(lpsa ~ ., pros, bound = 0.44)
Call:
l1ce(formula = lpsa ~ ., data = pros, bound = 0.44)

Coefficients:
 (Intercept)    lcavol    lweight age lbph       svi lcp gleason pgg45
    2.478387 0.5587661 0.09699974   0    0 0.1555875   0       0     0
```

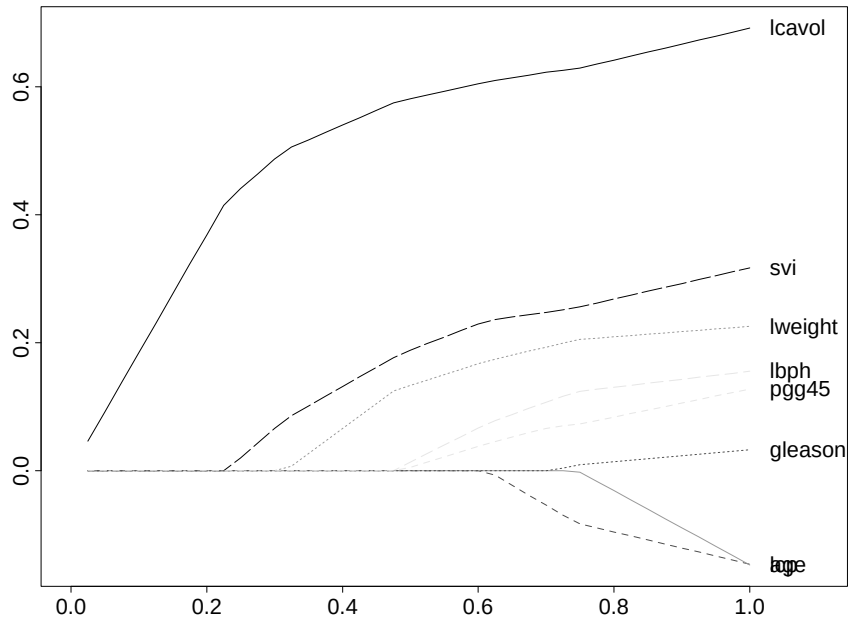

Figure 1: (Standardised) coefficients for prostate data for varying (relative) bounds

```
The relative L1 bound was      : 0.44
The absolute L1 bound was      : 0.8113534
The Lagrangian for the bound is:  17.89198
```

Note that you will get a warning message because `pros` is not a data frame. This can be avoided by issuing the command

```
> pros <- as.data.frame(pros)
```

In the above example the argument `bound` was a single constraint and hence a single model was fitted and `l1ce` returned an object of class `"l1ce"`. We may also specify a vector of constraints:

```
> res <- l1ce(lpsa ~ ., pros, bound = (1:40)/40)
```

In this case, an object of class `"l1celist"` is returned which contains all the fitted model objects corresponding to the specified bounds. The bounds may be specified in any order, and so, for example, `bound = (40:1)/40` is also permissible. The routine `l1ce` ensures that the underlying C engine will solve the problems sequentially from smallest to largest bound. This is the most efficient way of organising the calculations since each fit can utilise information from the previous one. You may verify this by specifying `trace = T`. In the returned result, however, the fitted models are ordered in the same way that `bound` was originally specified.

A `plot` method for objects of class `"l1celist"` can be used to plot how the fitted parameters

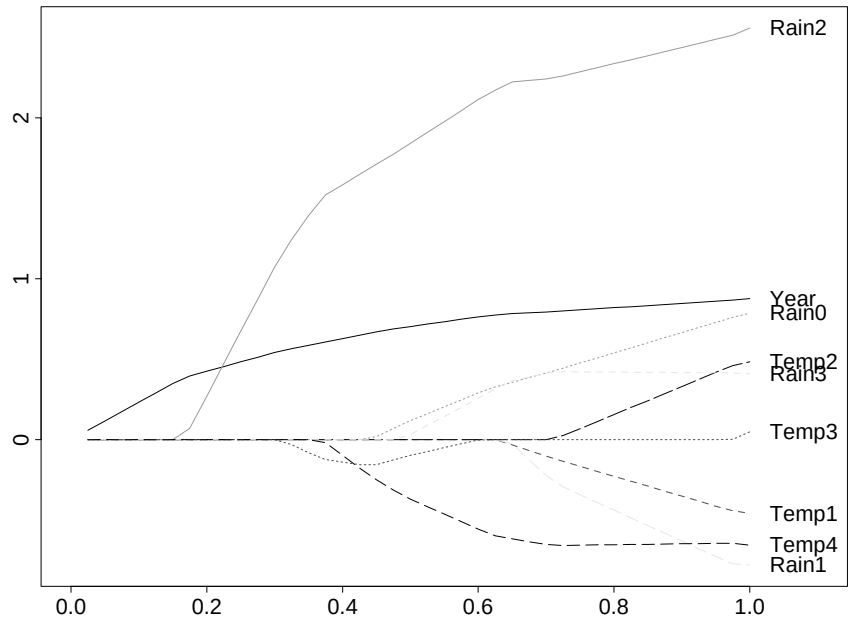

Figure 2: Coefficients for Iowa wheat data for varying (relative) bounds

change with a changing constraint. The plot produced is typically not very helpful, though, as it constructs a plot of all the coefficients, including those that were not under the constraint (here only the intercept). The coefficients that are not under the constraint may be on quite a different scale. Hence it is better to save the object returned by `plot` and tailor it so that the final plot is useful given the context. This can be done using a technique demonstrated by the following commands (the result is shown in Figure 1):

```
> plres <- plot(res)
> matplot(plres$bound[,"rel"], plres$mat[,-1], type = "l", xlim = c(0, 1.1))
> text(cbind(1.03, coef(res[40])[-1]), labels(res), adj = 0)
```

Iowa Wheat Data

Tibshirani (1996) has a figure similar to Figure 1 and remarks that "in this example, the curves decrease in a monotone fashion to 0, but this does not always happen in general". The Iowa wheat yield data of Draper and Smith (1998) (supplied in this library as data frame `Iowa`) provides an example where this monotone behaviour does not obtain.

The following commands, similar to those used in the last example, produced Figure 2:

```
> res <- l1ce(Yield~., Iowa, bound=(1:40)/40)
> plres <- plot(res, plot=F)
> matplot(plres$bound[,"rel"], plres$mat[,-1], type = "l", xlim = c(0, 1.1))
```

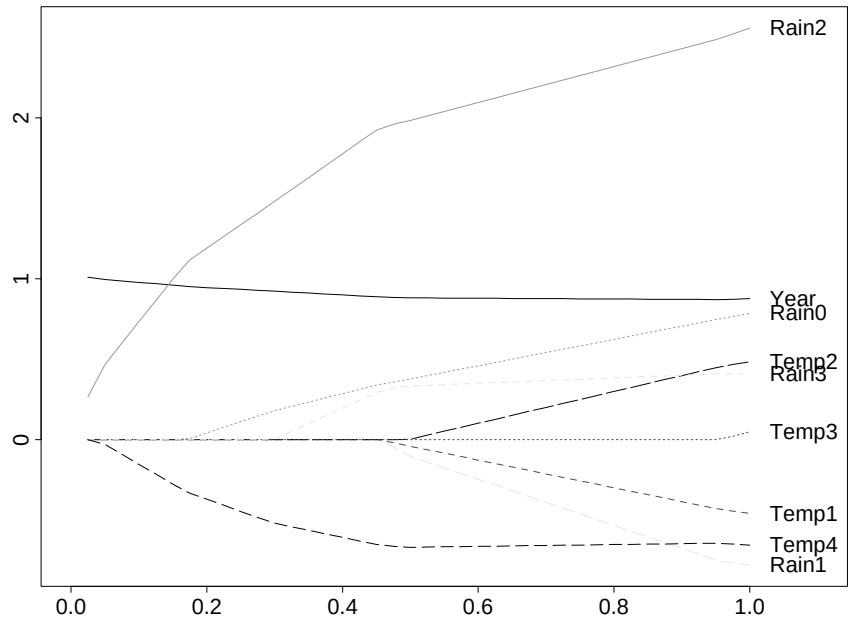

Figure 3: Coefficients for Iowa wheat data for varying (relative) bounds, if `Year` is not included in the constraint

```
> text(cbind(1.03, coef(res[40])[-1]), labels(res), adj = 0)
```

Note the behaviour of the coefficient labelled `Temp3`. It first enters the model with a negative coefficient, then drops out and later reappears with a positive coefficient when we are close to the unconstrained linear model.

In this example the variable `Year` enters the model first and seems to be the only unequivocally important variable for explaining `Yield`. It could be interesting, then, to see what happens if accept it *ab initio* and exclude it from the constraint. This can be done using either command

```
> res <- l1ce(Yield ~ ., Iowa, sweep.out = ~Year, bound = (1:40)/40)
```

or

```
> res <- update(res, sweep.out = ~Year)
```

That is, we 'sweep out' the variable `Year` as well as the intercept before imposing the constraint. If we now plot the result with the following commands, we obtain Figure 3.

```
> plres <- plot(res, plot = F)
> matplot(plres$bound[,"rel"], plres$mat[,-1], type = "l", xlim = c(0, 1.1))
> text(cbind(1.03, coef(res[40])[-1]), labels(res), adj = 0)
```

Now we do not have the changing sign effect any longer, but note that the coefficient labelled `Temp4` still shows slightly non-monotone behaviour.

4 Generalised Linear Models

`gl1ce` is an extension to `l1ce` that allows to apply the LASSO techniques to generalised linear models. The relationship between `gl1ce` and `l1ce` is similar to that of the (more standard) S-PLUS functions `glm` and `lm`. Especially, `gl1ce` uses the same way of specifying the generalised linear model (family, link function etc.) as `glm`. This extension was part of JL's honours project and more details on this function (and its performance) can be found in his honours report. You should find a (compressed) PostScript file of that report the same directory as this manual.

5 Incompatibilities

The code of Tibshirani (1995) also includes a routine for fitting Cox models. Such a routine is not (yet) implemented in our library. For normal regression models our library provides the function `l1ce` and below we describe some incompatibilities with Tibshirani's code. Generalised linear models can be handled via `gl1ce`. Details on this function can be found in JL's honours report. However, the approach taken by `gl1ce` is fundamentally different to Tibshirani's approach so that this routines are inherently incompatible.

Use of weights

If weights are used, a close inspection of the code of the routine `lasso` of Tibshirani (1995) shows that it first centres and scales the design matrix. Then the centred/rescaled design matrix and the y-vector are multiplied by (the square root of) the weights and finally the (now weighted) y-observations are centred. In our view this procedure is not an appropriate way of handling weights, and is incompatible with the way our routine `l1ce` handles them. Our procedure is described in Section 2.

The Lagrangian

In all calculations that involve the Lagrangian of the constraint our code uses the correct Lagrangian as derived in Osborne *et al.* (1999). Tibshirani's `findlam` routine searches for the Lagrangian on the discrete grid `seq(0, 2, length = 40)`. Hence any calculations that depend on the value of the Lagrangian will nearly certainly differ.

Covariance matrix of the estimators

For calculating the covariance matrix, our routine `vcov` uses by default the formula given in Osborne *et al.* (1999). If `type = "Tibshirani"` is used, then the formula of Tibshirani (1996) is used with the Moore-Penrose generalised inverse. To obtain the same result as with Tibshirani's routine `lasso.se` one would also have to specify `gen.inverse.diag = 1e11` and modify the Lagrangian (see comments above).

The cross-validation function

Tibshirani's routine `lasso.gcv` only centres and rescales the design matrix and omits to centre the response variable. Thus, to obtain the same results with our `gcv` routine one would have to construct a data frame that contains the centred and rescaled design matrix and then call `l1ce` (with `bound = seq(0, 1, length = 10)`) with a formula that *does not* include an intercept and `sweep.out = NULL`. Furthermore, it would be necessary to specify `type = "Tibshirani"`, `gen.inverse.diag = 1e11` and to modify all the Lagrangians of the different models.

Standard errors

Standard errors are calculated and returned by the `summary` function. These standard errors are calculated by default using the formula given in Osborne *et al.* (1999). Thus, to use the alternative formula of Tibshirani it is necessary to specify `type = "Tibshirani"` and `gen.inverse.diag = 1e11`. Care must also be taken about the value of the Lagrangian parameter since these calculations depend on it. Finally, our routine uses as estimate for the residual standard error the (square root) of the residual sum of squares of the model divided by the degrees of freedom (depending on the covariance formula used). By way of contrast, Tibshirani's routine `lasso.se` uses the residual standard error obtained by fitting an unconstrained linear model. Thus, to duplicate his results this estimate for the residual standard error has to be provided to `summary` via the parameter `sigma`.

The distribution of the constrained parameter estimates will typically have a condensation of probability at zero and so will be far from normal. This suggests that summarising the uncertainty by standard errors is possibly not appropriate. We suggest that this point deserves further theoretical investigation.

References

Draper, N.R. and Smith, H. (1998). *Applied Regression Analysis*, Wiley Series in Probability and Statistics, 3 edn, John Wiley & Sons, New York.

Hocking, R.R. (1996). *Methods and Applications of Linear Models: Regression and the Analysis of Variance*, Wiley Series in Probability and Statistics, John Wiley & Sons, New York.

Miller, A.J. (1990). *Subset Selection in Regression*, Vol. 40 of *Monographs on Statistics and Applied Probability*, Chapman and Hall, London.

Osborne, M.R. (1998). Variable selection and control in least squares problems, *Mathematics Research Report MRR98–047*, Centre for Mathematics and its Applications, School of Mathematical Sciences, Australian National University, Canberra ACT 0200.

Osborne, M.R., Presnell, B. and Turlach, B.A. (1998a). Berwin's problem, *Mathematics Research Report MRR98–049*, Centre for Mathematics and its Applications, School of Mathematical Sciences, Australian National University, Canberra ACT 0200. Submitted.
URL: *http://www.stats.adelaide.edu.au/people/bturlach/psfiles/MRR98-049.ps.gz*

Osborne, M.R., Presnell, B. and Turlach, B.A. (1998b). Knot selection for regression splines via the lasso, *in* S. Weisberg (ed.), *Dimension Reduction, Computational Complexity, and Information*, Vol. 30 of *Computing Science and Statistics*, Interface Foundation of North America, Inc., Fairfax Station, VA 22039–7460, pp. 44–49.
URL: *http://www.stats.adelaide.edu.au/people/bturlach/psfiles/interface98.ps.gz*

Osborne, M.R., Presnell, B. and Turlach, B.A. (1999). On the LASSO and its dual, *Journal of Computational and Graphical Statistics*. To appear.

Stamey, T.A., Kabalin, J.N., McNeal, J.E., Johnstone, I.M., Freiha, F., Redwine, E.A. and Yang, N. (1989). Prostate specific antigen in the diagnosis and treatment of adenocarcinoma of the prostate: II. radical prostatectomy treated patients, *Journal of Urology* **141**(5): 1076–1083.

Tibshirani, R. (1995). Lasso, Available from the S archive on StatLib.
URL: *http://lib.stat.cmu.edu/S/lasso*

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso, *Journal of the Royal Statistical Society, Series B* **58**(1): 267–288.

Turlach, B.A. (1998). Algorithms using the "lasso", Noweb file.