UNIVERSITÁ DEGLI STUDI DI MILANO
FACOLTÁ DI MEDICINA E CHIRURGIA

DIPARTIMENTO DI MEDICINA DEL LAVORO
SEZIONE DI STATISTICA MEDICA & BIOMETRIA

CORSO DI DOTTORATO DI RICERCA IN STATISTICA BIOMEDICA (D4R)
SETTORE SCIENTIFICO DISCIPLINARE MED/01 - XXIII CICLO

TESI DI DOTTORATO DI RICERCA

PENALIZED REGRESSION: BOOTSTRAP CONFIDENCE INTERVALS AND VARIABLE SELECTION FOR HIGH DIMENSIONAL DATA SETS.

DOTTORANDA:
Samantha SARTORI
R07738

RELATORE:
Prof. Silvano MILANI

COORDINATORE DEL DOTTORATO:
Prof. Silvano MILANI

ANNO ACCADEMICO 2009/2010

To Marco

Contents

Introduction

High dimensional and ultrahigh dimensional variable selection is a formidable challenge in biomedical statistics.
To face this problem, a number of promising approaches have recently been proposed. A very attractive method is penalized regression. This class of procedures shows many advantages but is still not very popular, mainly due to its computational cost. In this work, we focus our attention on these techniques and on their applications to genome-wide association studies (GWAS).

An overview of some of the most interesting penalization methods is given in the first chapter: Lasso, Least Angle Regression, Elastic Net, Adaptive Lasso, Scad, Combined Penalization and Relaxed Lasso. For each technique, we consider from a theoretical point of view the main ideas behind the method and we examine its pros and cons compared to other methods.

An important open problem in the field of ℓ_1-penalized regression is the construction of confidence intervals for the model coefficients. A popular approach to calculating confidence intervals is to use bootstrap simulation algorithms. In the second chapter, we investigate four bootstrap methods for regression models (parametric bootstrap, vector resampling, residual bootstrap, and two variants of one-step bootstrap - vector and residual resampling) and we consider their application to generalized linear model (GLMs). We review their functioning, we describe their implementation algorithms and we evaluate their performance by simulation studies.

In the third chapter, we start considering the residual bootstrap method for the lasso estimator of the regression parameters in a multiple linear regression model, recently proposed by [Chatterjee and Lahiri (2010)]. In the following section we extend this idea to penalized GLMs, using the notion of standardized Pearson's residuals. The results of the simulation studies show that this method has some serious drawbacks. As a result, in the sections that follows, we explore a completely different approach based on the fact that the coefficients of lasso for linear models can be approximated by ridge regression. After generalizing this result to ℓ_1-penalized GLMs, we develop a one-step (residual) resampling method for this class of models in the spirit of the one-step bootstrap for GLMs proposed by [Moulton and Zeger (1991)]. Then, applying the results of [Vinod (1995)], we build confidence intervals

(CIs) for the coefficients of the class of ℓ_1-penalized GLMs. The simulation studies suggest that by this method we are able to build CIs with good empirical coverage probabilities. In the final section, we consider the double bootstrap of Beran (1987) in order to further reduce the coverage errors of single bootstrap and to build confidence intervals with a higher order of accuracy.

Chapter four contains an overview of one of the most challenging and fascinating problems of modern biomedical statistics: ultrahigh dimensional variable selection in GWAS and gene environment-wide interaction (GEWI) studies, with particular attention on the evaluation of gene-gene and gene-environment interactions. This is a fundamental task in the investigation of complex patterns for complex disease. Sure Independence Screening (SIS), Iterative SIS (ISIS) and their variants are novel and effective methods for variable selection in ultrahigh dimensional settings. They are based on a prescreening step for dimension reduction followed by a selection/estimation step performed using ℓ_1-penalized regression. We test this method on a simulated dataset obtaining interesting results.

Finally, we briefly review some applications of penalized regression, random forests and bayesian networks in the field of GWAS and GEWI studies.

Acknowledgments:

This thesis would not have been possible without the help and support of Professor David Madigan, who welcomed me to Columbia University and provided the conceptual design underlying this project: I was fortunate to work on his proposal to build confidence intervals for penalized logistic regression models.

Additionally, I would like to pay special thanks to Dott. Marco Sandri, who helped me developing the software required to perform the simulations for this project.

I would like also to thank my advisor and program coordinator, Professor Silvano Milani, for supporting my desire to finish my program at Columbia University.

I am grateful to them also for reading, editing, and commenting on the numerous drafts of this thesis, along with Matthew Truesdale and Professor Micheal Hogan.

Finally, I would like to thank: all the staff, faculty, PhD students at Columbia University for welcoming me into the Department of Statistics and making me feel at home, and, last but not least, Anna, Elena, Maddalena, Cristian and all the staff and faculty at 'Clincia del Lavoro Luigi Devoto' for their precious support.

Chapter 1

Penalized regression

1.1 Introduction

The number of variables analyzed in recent clinical studies (especially in genome-wide association studies) is rapidly increasing. Typically, not all of these predictor variables may contribute equally well to the explanation of the outcome and some of them may not contribute at all to the modelization of the phenomenon under study. Thus, it is of fundamental importance to select from these variables only those that are informative in order to obtain the best model in terms of predictive accuracy and with the lowest number of variables.

Many variable selection procedures are based on the joint use of variable importance for ranking covariates and model estimation to generate, evaluate and compare a family of models. Following [Kohavi and John (1997)], [Kohavi and John (2001)] and [Guyon and Elisseeff (2003)], it is possible to distinguish three types of variable selection methods:

- 'filter' for which the score of variable importance does not depend on a given model design method;
- 'wrapper' which includes the prediction performance in the score calculation; and finally
- 'embedded' which combines more closely the variable selection and model estimation.

In addition, selection criteria are usually classified into two categories:

- consistent; a consistent criterion identifies the true model with a probability that approaches 1 in large samples when a set of candidate models contain the true model;
- efficient; an efficient criterion selects the model so that its average squared error is asymptotically equivalent to the minimum offered by

the candidate models when the true model is approximated by a family of candidate models.

Penalized regression, first introduced by [Tibshirani (1996)], is a novel and promising class of regression models for variable selection in classification and regression problems with large numbers of candidate predictor variables. This approach can be valuable in producing interpretable models, accurate predictions, and approximately unbiased inferences. Penalized regression is a typical 'embedded' variable selection method because variable selection and model estimation are performed together by minimizing the penalized objective function which yields a sparse vector of model coefficients.

The aim of the present chapter is to present a comprehensive view of the key ideas behind penalized regression and to review some of the most important techniques that use this approach for data modeling. In the following sections, we consider the Lasso, Least Angle Regression, Elastic Net, Adaptive Lasso, SCAD, CP and Relaxo.

An important problem related to penalized regression is the estimation of the optimal penalty parameter λ, that is the complexity/regularization parameter that controls simultaneously the amount of shrinkage on coefficients and the selected subset of variables included in the final model. In subsection 1.3.2, we consider some of the most important penalty estimation methods: cross-validation, generalized cross-validation, AIC, BIC and GIC.

1.2 Convex Optimization

Fitting a model to data that is estimates model parameters requires the (exact or approximate) solution of a problem of optimization: the maximization of the log-likelihood function or the minimization of the sum of squared model residuals. Penalized regression involves the solution of a problem of constrained optimization: the maximization/minimization of a function under some constraints.

The *constrained optimization problem* can be formulated as

$$\text{minimize } f_0(x) \qquad \text{subject to: } f_i(x) \leq b_i, \quad i = 1, \ldots, m, \tag{1.1}$$

where $x = (x_1, \ldots, x_k)$ is the *optimization variable* of the problem, the function $f_0 : \mathbb{R}^k \to \mathbb{R}$ is the *objective* function, the functions $f_i : \mathbb{R}^k \to \mathbb{R}$, $i = 1, \ldots, m$ are the (inequality) $constraint functions$, and the constants $b_1, \ldots, b_m$ are the limits, or bounds, for the constraints. A vector x^* is called *optimal*, or a *solution* of the problem (1.1), if it has the smallest objective value among all vectors that satisfy the constraints: for any x with $f_1(x) \leq b_1, \ldots, f_m(x) \leq b_m$, we have $f_0(x) \geq f_0(x^*)$.

In data fitting, the task is to find a model from a family of candidate models that best fits some observed data. The optimization variables are the parameters of the model. In the case of penalized regression, we will see that the constraint is the required limit on the l_1-norm of the vector of coefficients ($\sum_j |b_j| < t$).

In the context of penalized regression, *convex optimization* is often mentioned a special class of mathematical optimization problems, including least-squares and linear programming problems. While the mathematics of convex optimization has been studied for about a century, several related recent developments have stimulated new interest in the topic. Many applications have been discovered in areas such as automatic control systems, estimation and signal processing, communications and networks, electronic circuit design, data analysis and modeling, statistics, and finance (see [Boyd and Vandenberghe (2004)] and [Nocedal and Wright, (1999)]).

The set $S \in \mathbb{R}^k$ is a convex set if the straight line segment connecting any two points in S lies entirely inside S. Formally, for any two points $x \in S$ and $y \in S$, we have $ax + (1-a)y \in S$ for all $a \in [0,1]$.

A function f is convex if its domain is a convex set and if for any two points x and y in this domain, the graph of f lies below the straight line connecting $(x, f(x))$ to $(y, f(y))$ in the space $\mathbb{R}^{k+1}$. That is, we have

$$f(ax + (1-a)y) = af(x) + (1-a)f(y), \quad \forall a \in [0,1].$$

Optimization of a non-convex function in high-dimensional setting is generally a difficult task. Unfortunately many regularization procedures with otherwise attractive features involve, minimization of a non-convex function (e.g. [Fan and Li (2001)]). For high-dimensional problems, it is in general very costly to find a solution in this case due to the presence of local minima in the objective function.

Recognizing or formulating a problem as a convex optimization problem has many advantages.

- ◇ Global minimizer; if the objective function is convex and inequality constraint functions are concave, algorithms for optimization are usually guaranteed to converge to a global minimum.
- ◇ Robustness; algorithms for convex optimization perform well on a wide variety of problems in their class, for all reasonable choices of the initial variables.
- ◇ Efficiency; algorithms do not require too much computer time or storage.
- ◇ Accuracy; algorithms are able to identify a solution with precision, without being overly sensitive to errors in the data or to rounding errors.

Setting penalization in order to be a convex optimization problem is therefore an attractive choice (see [Meinshausen (2007b)]).

1.3 The LASSO

[Tibshirani (1996)] proposed a shrinkage method named LASSO (Least Absolute Shrinkage and Selection Operator), a constrained version of ordinary last squares (OLS). This technique is in some sense similar to ridge regression (see [Hastie et al. (2009)]) but it can shrink some coefficients to 0, and thus can implement variable selection. The LASSO method estimates the coefficients by maximizing the log-partial likelihood with the constraint that the sum of the absolute values of the model coefficients is bounded above by some positive number. The LASSO produces interpretable models like subset selection and exhibits the stability of ridge regression.

Let (x_i, y_i), $i = 1, 2 \ldots, N$, be a sample of N independent and identically distributed (i.i.d.) random vectors, where $x_i = (x_{i1}, x_{i2}, \ldots, x_{ip}) \in \mathbb{R}^p$ is the row vector of observations about p predictor variables for the ith sample unit and $y_i \in \mathbb{R}$ is the corresponding response vector.

The vector of $p + 1$ lasso estimates $(\hat{\beta}_0, \hat{\beta})$ of regression coefficients is defined by

$$\begin{aligned}
(\hat{\beta}_0, \hat{\beta})_{(\text{lasso})} &\equiv \underset{(b_0,\, b) \in \mathbb{R}^{p+1}}{\operatorname{argmin}} \ \text{MSE}(b) \quad \text{subject to} \sum_{j=1}^{p} |b_j| \le t \\
&= \underset{(b_0,\, b) \in \mathbb{R}^{p+1}}{\operatorname{argmin}} \left[\frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i(b_0, b))^2 \right] \ \text{s.t.} \sum_{j=1}^{p} |b_j| \le t \\
&= \underset{(b_0,\, b) \in \mathbb{R}^{p+1}}{\operatorname{argmin}} \left[\frac{1}{N} \sum_{i=1}^{N} (y_i - b_0 - x_i b)^2 \right] \ \text{s.t.} \sum_{j=1}^{p} |b_j| \le t, \quad (1.2)
\end{aligned}$$

where $b_0, \hat{\beta}_0 \in \mathbb{R}$, $b = (b_1, b_2 \ldots, b_p)^T$ is a column vector of p components and $\hat{\beta} = (\hat{\beta}_1, \cdots, \hat{\beta}_p)^T$.

Here $t \ge 0$ is a tuning parameter. For all t, the solution for b_0 is $\hat{\beta}_0 = \bar{y}$, where $\bar{y}$ is the mean of y. Without loss of generality, we can assume that the x_{ij} are standardized: $1/N \cdot \Sigma_{i=1}^{N} x_{ij} = 0$ and $1/N \cdot \Sigma_{i=1}^{N} x_{ij}^2 = 1$, for $j = 1, 2, \ldots, p$. Hence, we can omit b_0.

Equation (1.2) becomes

$$\hat{\beta}_{(\text{lasso})} = \underset{b \in \mathbb{R}^p}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^{N} (y_i - x_i b)^2 \quad \text{s.t.} \sum_{j=1}^{p} |b_j| \le t. \tag{1.3}$$

Using the Lagrangian form, this optimization problem is equivalent to

$$\hat{\beta}_{(\text{lasso})} = \underset{b \in \mathbb{R}^p}{\operatorname{argmin}} \left[\frac{1}{N} \sum_{i=1}^{N} (y_i - x_i b)^2 + \lambda \sum_{j=1}^{p} |b_j| \right], \tag{1.4}$$

where λ (the Lagrange multiplier) is a penalty parameter related to t.

In matrix form, equation (1.4) can be written as

$$\begin{aligned} \hat{\beta}_{(\text{lasso})} &= \underset{b \in \mathbb{R}^p}{\operatorname{argmin}}\, R(b, \lambda) \equiv \underset{b \in \mathbb{R}^p}{\operatorname{argmin}} \frac{1}{N} ||Y - Xb||_2^2 + \lambda ||b||_1 \\ &= \underset{b \in \mathbb{R}^p}{\operatorname{argmin}} \frac{1}{N} \left[(Y - Xb)^T (Y - Xb) \right] + \lambda \mathbf{1}_p b, \end{aligned} \tag{1.5}$$

where X is the $(n \times p)$ design matrix of observed covariates, Y is the $(n \times 1)$ column vector of the observed outcome, $||\cdot||_2$ is the L_2 vector norm, $||\cdot||_1$ is the L_1 vector norm, A^T is the transpose of matrix A and $\mathbf{1}_p$ is a row vector of p ones.

The first-order stationarity conditions ($\partial R / \partial b_j |_{b_j = \hat{\beta}_j} = 0$) for the optimization problem (1.5) are

$$\begin{aligned} \frac{2}{N} x_j^T (Y - X\hat{\beta}) &= \lambda \cdot \operatorname{sign}(\hat{\beta}_j) \quad \forall j : \hat{\beta}_j \neq 0 \\ \frac{2}{N} |x_j^T (Y - X\hat{\beta})| &\leq \lambda \quad \forall j : \hat{\beta}_j = 0 \end{aligned}$$

where sign is the function

$$\operatorname{sign}(x) = \begin{cases} -1 & \text{if } x < 0 \\ 0 & \text{if } x = 0 \\ 1 & \text{if } x > 0 \end{cases}$$

and $\hat{\beta} = \hat{\beta}_{(\text{lasso})}$. If we let $\lambda_n \equiv N/2 \cdot \lambda$, the above equations simplify to:

$$x_j^T (Y - X\hat{\beta}) = \lambda_n \cdot \operatorname{sign}(\hat{\beta}_j) \quad \forall j : \hat{\beta}_j \neq 0 \tag{1.6}$$

$$|x_j^T (Y - X\hat{\beta})| \leq \lambda_n \quad \forall j : \hat{\beta}_j = 0 \tag{1.7}$$

Computation of the solution to equation (1.5) is a quadratic programming problem with linear inequality constraints. Parameters t and λ_n control the amount of shrinkage that is applied to $\hat{\beta}$ estimates. Let $\hat{\beta}_j^{\text{OLS}}$ be the ordinary least squares estimates and let $t_0 = \Sigma |\hat{\beta}_j^{\text{OLS}}|$. Values of $t < t_0$ will cause shrinkage of the solution towards 0, and some coefficients may be exactly equal to 0. For example, if $t = t_0/2$, the effect will be roughly similar to finding the best subset of size $p/2$.

The most important advantage of lasso is its simultaneous execution of both variable selection and parameter estimation. Traditional methods work on the two problems separately, using different techniques: first selecting informative variable and, then, estimating model parameters.

The lasso penalty has some important limitations.

- ◇ When $p > n$, the lasso selects at most n variables before it saturates. This may not be a desirable feature for many practical problems, particularly microarray studies. In fact, it is unlikely that only n genes (usually a small number) are involved in the development of a complex disease.

- ◇ If there is a group of variables among which the pairwise correlations are very high, then the lasso tends to select only one variable or a few of them from the group and shrinks the rest to 0. Again, this may not be a desirable feature. For example, in microarray analysis, expression levels of genes that share one common biological pathway are usually highly correlated. These genes may all contribute to the biological process but lasso usually selects only one gene from the group.

- ◇ When $n > p$, if there are high correlations between predictors, it has been empirically observed that lasso is inferior to ridge regression in terms of prediction performance. Some theoretical studies (see for example [Zhao and Yu (2006)]) show that lasso, for variable selection, only works in a rather narrow range of problems (where the restrictive 'irrepresentable condition' of [Zhao and Yu (2006)] is satisfied), excluding cases where the design exhibits strong (empirical) correlations.

1.3.1 Estimated coefficients vs. penalization

In this section we investigate the relationships existing between regression coefficients $\hat{\beta}$ estimated by lasso and the regularization parameter λ_n, the fraction of deviance explained and the sum of absolute values of the regression coefficients (L1 norm of vector $\hat{\beta}$). Figure 1.1 shows an example of these behaviors for a sample generated using the DGP1 defined in section 1.1, with independent covariates and $n = 10000$. It is interesting to observe the linear relationship existing between the lasso coefficients and λ_n.

In the case of an orthonormal design matrix X, it is $X^T X = I$, where I is the identity matrix of order p and, of course, $x_j^T X = e_j$, where e_j is a row vector of p zeros with a 1 at the jth position. Hence, the vector of OLS estimates is given by

$$\hat{\beta}^{\text{OLS}} = (X^T X)^{-1} X^T Y = X^T Y$$

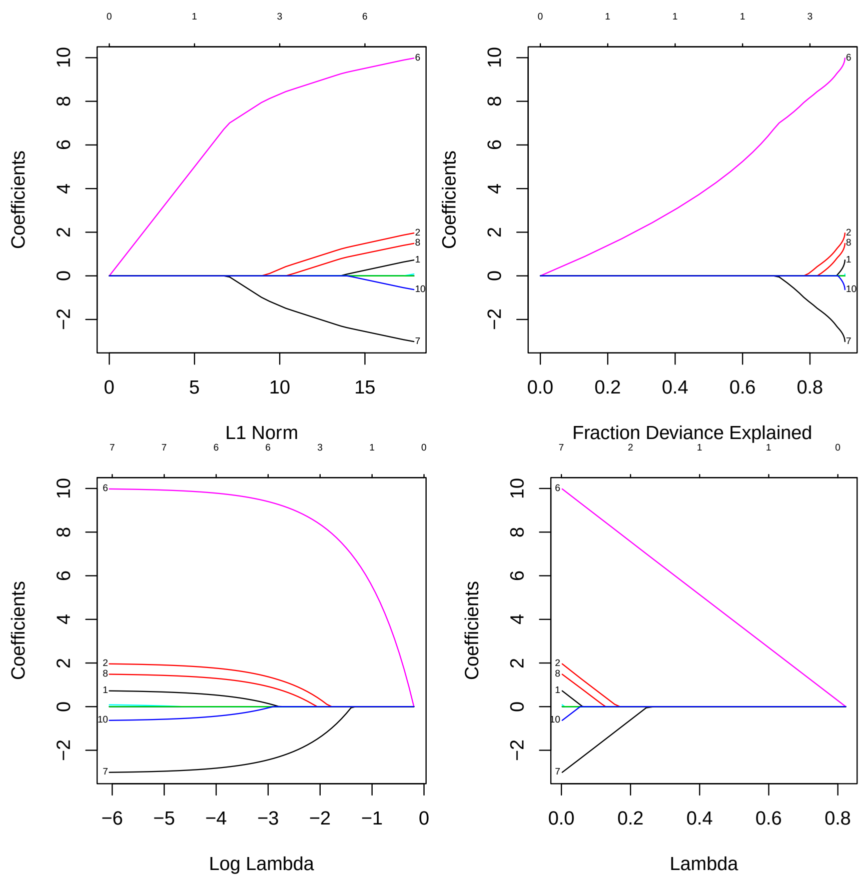

Figure 1.1: Plot of regression coefficients against L1-norm of $\hat{\beta}$, fraction of deviance explained, logarithm of λ_n and λ_n.

and, for $\hat{\beta}_j \neq 0$, (1.6) is

$$\begin{aligned} x_j^T Y - x_j^T X\hat{\beta} &= \lambda_n \cdot \mathrm{sign}(\hat{\beta}_j) \\ \hat{\beta}_j^{\mathrm{OLS}} - e_j\hat{\beta} &= \lambda_n \cdot \mathrm{sign}(\hat{\beta}_j) \\ \hat{\beta}_j^{\mathrm{OLS}} - \hat{\beta}_j &= \lambda_n \cdot \mathrm{sign}(\hat{\beta}_j) \end{aligned}$$

that is

$$\hat{\beta}_j = (|\beta_j^{\mathrm{OLS}}| - \lambda_n)_+ \,\mathrm{sign}(\hat{\beta}_j) \tag{1.8}$$

where

$$(x)_+ = \begin{cases} 0 & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases}$$

is the positive part of x.

When the columns of X are independent, the design X is asymptotically orthonormal. That is, $X^T X \to I$ as $n \to \infty$. In this case, for large n, equation (1.8) approximates the relationship existing between $\hat{\beta}$ and λ_n.

Figure 1.2 shows the plot of estimated lasso coefficients for different values of the regularization parameter in a sample generated using the DGP1 defined in section 1.1, with independent covariates and $n = 10000$. As expected, regression coefficients linearly increase (in absolute value) for decreasing values of λ_n. In addition, this linear behavior closely agree with equation (1.8). Dashed gray lines show the intercepts with the x-axis predicted by (1.8), $\lambda_n^0 = |\hat{\beta}_j^{\text{OLS}}|$.

Below we show the estimated and the theoretical values of slopes and x-axis intercepts of the plotted lines. As expected, slopes are all rather close to sign$(\hat{\beta})$ and x-axis intercepts to $|\hat{\beta}_j^{\text{OLS}}|$.

```
     slope.est slope.theo  x.intcp.est x.intcp.theo
X1   1.0078736          1  0.726862286  0.732585342
X2   1.0508485          1  1.911826492  2.009040066
X3  -0.9764716         -1  0.011817110  0.011539072
X4  -0.7578182         -1  0.007882202  0.005973276
X5  -1.3193448         -1  0.005917527  0.007807259
X6   0.9849023          1 10.171587262 10.018019926
X7  -1.0102826         -1  2.959106313  2.989533666
X8   1.0131297          1  1.476484698  1.495870534
X9   1.0579407          1  0.002695215  0.002851378
X10 -1.0401500         -1  0.630970223  0.656303658
```

To summarize, this simulation study shows that when the p covariates of X are independent and when n is large, a (quasi) linear relationship between $\hat{\beta}_n$ lasso coefficients and λ_n exists. Equation (1.8) can be used as an approximation of this linear function.

It is now worth to noting that when covariates X are correlated, this linear behavior ceases to exist. In figure 1.3, covariates X_7 is correlated with covariates X_6 (Pearson's correlation ≈ 0.5). The relationship between $\hat{\beta}_{n6}$ and λ_n is now approximately piece-wise linear and the x-axis intercepts calculated using equation (1.8) are clearly inadequate. The table below shows the estimated values of slopes and x-intercepts and the corresponding values given by equation(1.8):

```
     slope.est slope.theo x.intcp.est x.intcp.theo
X1   1.0138118          1 0.722604922  0.732585426
X2   1.3504147          1 1.487720334  2.009039418
X3  -0.9834043         -1 0.011733513  0.011538786
X4  -0.7596110         -1 0.007861377  0.005971588
X5  -1.3171489         -1 0.005927254  0.007807075
X6   1.1643135          1 8.598804628 10.011704553
```

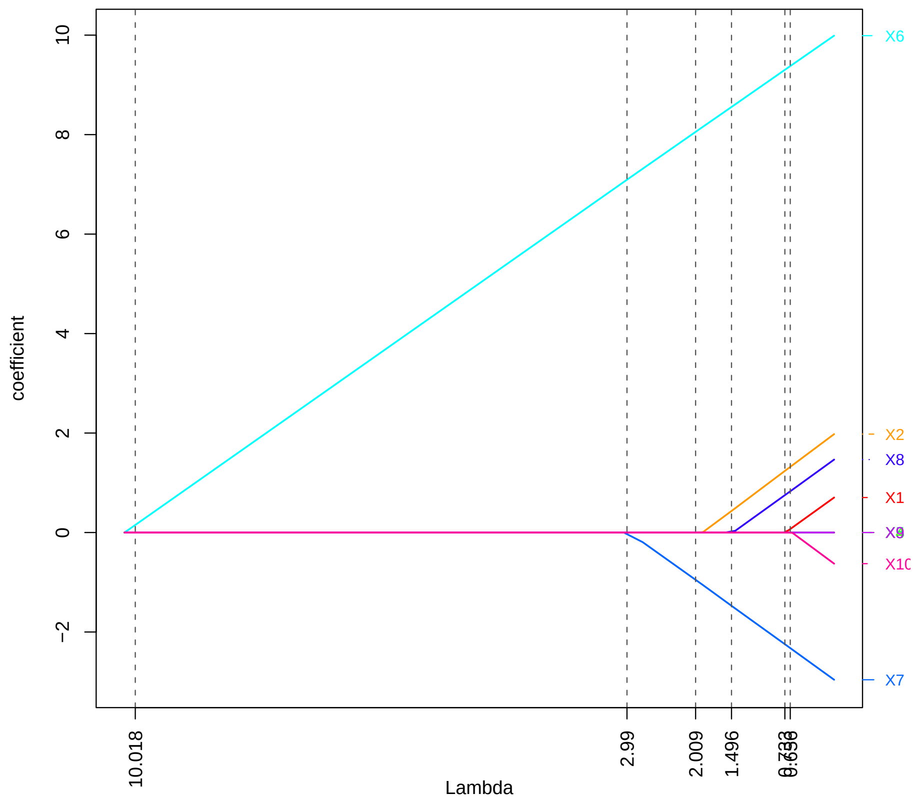

Figure 1.2: Case with uncorrelated variables X. Plot of $\hat{\beta}$ lasso coefficients vs. regularization parameter λ_n. The dotted lines show the theoretical values of x-axis intercepts ($x = |\hat{\beta}_j^{\text{OLS}}|$).

```
X7  -1.5737687         -1 1.898518934  2.987829746
X8   1.0353119          1 1.444850131  1.495870549
X9   1.0616849          1 0.002684975  0.002850598
X10 -1.0367392         -1 0.633046053  0.656303657
```

1.3.2 Finding optimal penalization

A fundamental step in lasso penalized regression is the choice of the regularization parameter λ_n. This is a complexity parameter that controls simultaneously the amount of shrinkage on coefficients and the selected subset of variables included in the final model. From one side, a too large λ_n leads to a situation where the amount of shrinkage is excessive and the subset of selected variables may not contain some informative variables. From the

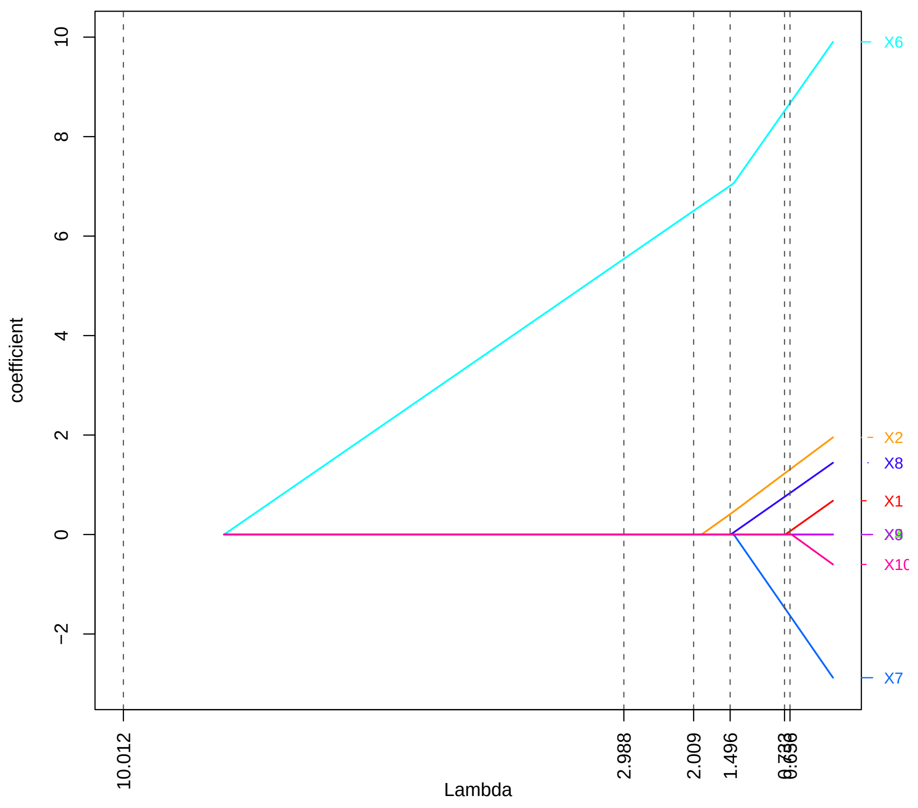

Figure 1.3: Case with correlation between X_6 and X_7. Plot of $\hat{\beta}$ lasso coefficients vs. regularization parameter λ_n. Dotted lines show the predicted values of x-axis intercepts.

other side, a too small λ_n leads to the opposite scenario where the amount of shrinkage is too small and the subset of selected variables may include some uninformative covariates. Consequently, this parameter should be 'optimal' in some sense.

In the original work [Tibshirani (1996)], three methods for the estimation of the lasso parameter t are proposed: cross-validation, generalized cross-validation and an analytical unbiased estimate of risk. The first two methods are applicable in the 'X-random' case, where it is assumed that the observations $(\mathbf{X}, \mathbf{Y})$ are drawn from some unknown distribution, and the third method applies to the X-fixed case. In real problems there is often no clear distinction between the two scenarios and we might simply choose the most convenient method. [Friedman et al. (2010)] developed the `glmnet` algorithm for computing the entire path of solutions for a grid of values of the regularization parameter. The authors suggest using prediction error to guide the choice of the optimal λ.

Cross-validation

Cross-validation is a popular data-driven technique and one of the simplest methods for estimating prediction error. It is frequently used in supervised learning to select a model from a family of candidate models.

This method is widely used for estimating the appropriate regularization parameter λ from the available data and is implemented in many R packages, for example `lars` (command `cv.lars`), `penalized` (command `cvl)`, `glmnet` (command `cv.glmnet`), `lqa` (command `cv.lqa`), `glmpath` (commands `cv.glmpath` and `cv.coxpath`).

When the sample size is high, users can set aside some fraction (say a third) of their data for prediction. They would then evaluate the prediction performance at each value of λ, and pick the model with the best performance via prediction performance.

When sample size is moderate, k-fold cross-validation is an interesting alternative. This method uses part of the data (the training set) to fit the model and use the remaining part (the validation set) to assess the performance of the model. The idea is to partition data into K ($K > 2$) sets. Only one of these sets is used for validation and the remaining $K - 1$ sets are pooled for training purposes. To reduce variability, this procedure is repeated K times and mean squared prediction error is calculated.

The CV algorithm for the calculation of the optimal λ can be summarized as follows:

(1) split the sample $z = (X, y)$ of size n into K equally-sized subsets z_k;

(2) using lasso, estimate the model parameters $\beta_{(k)}$ on the set

$$z_{(k)} = \{z_1, \cdots, z_{k-1}, z_{k+1}, \cdots, z_K\}$$

for a grid of J values of λ; call these estimates $\hat{\beta}_{(k)}(\lambda_j)$ and denote by $\hat{f}_{(k)}(x, \lambda_j)$ the fitted function, $j = 1, \cdots, J$;

(3) compute the prediction error $\mathrm{PE}_{(k)}$ of the estimated model on the test sample z_k using

$$\mathrm{PE}_{(k)}(\lambda_j) \equiv \frac{k}{n} \sum_{i \in z_k} (y_i - \hat{f}_{(k)}(x_i, \lambda_j))^2; \tag{1.9}$$

(4) repeat steps (2) and (3) for $k = 1, 2, \ldots, K$;

(5) compute the average of the K prediction errors as the overall estimate of the prediction error

$$CV(\hat{f}, \lambda_j) = \frac{1}{K} \sum_{k=1}^{K} \mathrm{PE}_{(k)}(\lambda_j) \tag{1.10}$$

Typical choices of k are 5 or 10. The case $k = n$ is known as *leave-one-out* cross-validation. In this case, the fit is computed using all the data except the ith unit.

The optimal value of the penalty parameter λ can be estimated using one of the following criteria:

(a) the value $\lambda_{\min}$ that gives minimum average cross-validated prediction error:

$$\lambda_{\min} \equiv \operatorname*{argmin}_{j=1,\cdots,J} CV(\hat{f}, \lambda_j);$$

(b) the value $\lambda_{1\mathrm{se}}$ such that error is within 1 standard error of the minimum average cross-validated prediction error.

Generalized cross-validation

Another method for estimating λ may be obtained using a linear approximation of the lasso estimate. The Lagrangian penalty $\lambda\Sigma|\beta_j|$ of equation (1.4) can be written as $\lambda\Sigma\beta_j^2/|\beta_j|$. Thus, we may write the constrained solution $\hat{\beta}$ as the ridge regression estimator

$$\tilde{\beta} = (X^T X + \lambda W^-)^{-1} X^T y$$

where W is a diagonal matrix with diagonal elements $|\tilde{\beta}_j|$ and W^- denotes the generalized inverse (the generalized inverse is necessary because some $\tilde{\beta}_j$ could be zero and W is not invertible). The number of effective parameters in the constrained fit $\tilde{\beta}$ may be approximated by

$$p(t) = tr\left\{X(X^T X + \lambda W^-)^{-1} X^T\right\}.$$

Let $\mathrm{RSS}(\lambda) = \|y - X\tilde{\beta}(\lambda)\|_2^2$ be the residual sum of squares for the constrained fit with penalty λ. [Tibshirani (1996)] constructs the generalized cross-validation style statistic

$$\mathrm{GCV}(\lambda) = \frac{1}{N} \frac{\mathrm{RSS}(\lambda)}{\{1 - p(\lambda)/N\}}. \tag{1.11}$$

The optimal λ is

$$\lambda_{\mathrm{GCV}} \equiv \operatorname*{argmin}_{\lambda} GCV(\lambda).$$

AIC, BIC and GIC

A family of widely-used selection criteria are those based on the likelihood or information measure. The methods described in this section AIC, BIC and GIC work in this way for a special class of estimates that are linear in their parameters.
Bayesian information criterion (BIC) with a large penalty performs well for 'small' models and poorly for 'large' models while Akaike's information criterion do just the opposite.

The Kullback-Leibler (K-L) information is a tool used as a means of discriminating between the true model and the candidate model. Let X be a continuous random vector and $f(x, \theta)$ be a probability density function of x, where $\theta \in \mathbb{R}^p$. Let θ^* be the true parameter of θ with density function $f(x, \theta^*)$. The K-L information measures the closeness of $f(x, \theta)$ to $f(x, \theta^*)$:

$$KL(\theta^*; \theta) = E[\log f(x, \theta^*) - \log f(x, \theta)].$$

The Akaike Information Criterion (AIC) was proposed by [Akaike (1973)] to estimate the expected Kullback-Leibler information between the model generating the data and a fitted candidate model. It is a criterion for selecting an optimum model in a class of nested and nonnested models or models fitted on different samples. The AIC criterion selects the model that minimizes

$$\text{AIC}(\lambda) = -2L_\lambda + 2p_\lambda \tag{1.12}$$

where L_λ is the maximum log-likelihood for the λth model and p_λ is the model complexity (in a linear model correspond to the number of predictors).

Equation (1.12) measures the loss of information when a given model is used to describe reality. It is an asymptotically unbiased estimate of the expected K-L information. There are a number of successful applications that support this method in spite of its 'nonconventional' style.

In the case of small sample size, [Hurvich and Tsai (1989)] proposes a corrected version of the AIC criterion:

$$\text{AIC}_\text{c} = \text{AIC} + \frac{2(m+1)(m+2)}{n-m-2} \tag{1.13}$$

where m is the number of covariates. For linear regression, AIC_c is unbiased, assuming that the candidate family of models includes the true model. AIC_c has better bias properties than does AIC.

The Bayesian information criterion (BIC) proposed by [Schwarz (1978)] has the same form of the AIC with the exception that the log-likelihood is penalized by $\log n$ instead of 2, where n is the sample size.

The BIC criterion selects the model that minimizes:

$$\text{BIC}(\lambda) = -2L_\lambda + \log(n)p_\lambda \tag{1.14}$$

The asymptotic properties of AIC and BIC were deeply studied and compared in literature.

[Zhang et al. (2010)] proposed to select the optimal regularization/penalty parameter λ minimizing a generalization of the information criterion (GIC) that contains a broad range of selection criteria.

GIC is defined as follows:

$$\mathrm{GIC}_{\kappa_n}(\lambda) = \frac{1}{N}\left[G(y, \hat{\beta}(\lambda)) + \kappa_n \mathrm{df}(\lambda)\right], \tag{1.15}$$

where $G(y, \hat{\beta}(\lambda))$ is a measure of fitting of the model, $y \in \mathbb{R}^N$, $\hat{\beta}(\lambda)$ is the penalized parameter estimator obtained by lasso (or other penalized regression method) and $\mathrm{df}(\lambda)$ is the number of degrees-of-freedom of the model. The optimal λ is defined by

$$\lambda_{\mathrm{GIC},\kappa_\mathrm{n}} = \underset{\lambda}{\mathrm{argmin}}\ \mathrm{GIC}_{\kappa_n}(\lambda).$$

It is possible to show (see [Zhang et al. (2010)]) that the difference between $\mathrm{df}(\lambda)$ and the number $d(\lambda)$ of nonzero parameters of the model is small. Therefore, $d(\lambda)$ can be used in place of $\mathrm{df}(\lambda)$ in equation (1.15).

The parameter κ_n is a positive number that controls the properties of variable selection. The larger κ_n, the higher the penalty for models with more variables. When $\kappa_n = 2$, GIC becomes AIC. If $\kappa_n \to 2$ GIC is called the AIC-type selector. When $\kappa_n = log(N)$, GIC becomes BIC. If $\kappa_n \to \infty$ and $\kappa_n/\sqrt{n} \to 0$, GIC is called the BIC-type selector.

[Zhang et al. (2010)] show two important asymptotic results for nonconcave penalized likelihood functions.

(1) If the true model is contained in a set of candidate linear and generalized linear models (GLIM), the BIC-type selector identifies the true model with probability tending to 1. In other words, the BIC-type selector has the oracle property.

(2) If the true model is approximated by a family of candidate GLM models, the AIC-type is asymptotically less efficient. In other words, the AIC-type selector identifies the model so that its average squared error is asymptotically equivalent to the minimum offered by the family of candidate models.

1.4 Least Angle Regression

The original lasso algorithm was proved to be relatively inefficient and complex. [Efron et al. (2004)] proposed an alternative model selection algorithm, *Least Angle Regression* (LARS). LARS is a less greedy version of traditional forward selection methods, such as All Subsets, Forward Selection and Backward Elimination.

An interesting characteristic of LARS is that it implements the lasso by a simple modification. The LARS modification calculates all possible lasso estimats for a given problem in an order of magnitude which requires a much smaller amount of computational time then previous methods.

Least angle regression is a stylized version of the Stagewise procedure [Efron et al. (2004)]. LARS is intimately connected with LASSO, and in fact provides an extremely efficient algorithm for computing the entire LASSO path. LARS uses a similar strategy as forward stepwise regression, but only enters "as much" of the predictor as it deserves. At the first step, it identifies the variable most correlated with the response; fits the variable completely, LARS moves the coefficient of this variable continuously toward its least-square value, as soon as another variable "catches up" in terms of correlation with the residual, the process is paused. The second variable joins the active set, and their coefficients are moved together in a way that keeps their correlations tied and decreasing. This process is continued until all the variables are in the model and ends at the full least-squares fit.

The LARS algorithm can be summarized as follows:

1. Standardize the predictors to have mean zero and unit norm. Start with residual $\mathbf{r} = \mathbf{y} - \hat{\mathbf{y}}$, $\beta_1, \beta_2, \cdots, \beta_p = 0$.

2. Find the predictor $\mathbf{x}_j$ most correlated with $\mathbf{r}$.

3. Move β_j from 0 towards its least-squares coefficient $\langle \mathbf{x}, \mathbf{r} \rangle$, until some other competitor $\mathbf{x}_k$ has as much correlation with the current residual as does $\mathbf{x}_j$.

4. Move β_j and β_k in the direction defined by their joint least squares coefficient of the current residual on $(\mathbf{x}_j, \mathbf{x}_k)$, until some other competitor $\mathbf{x}_l$ has as much correlation with the current residual.

   lasso modification. If a non-zero coefficient hits zero, drop its variable from the active set of variable and recompute the current joint least squares direction.

5. Continue in this way until all p predictors have been entered. After $\min(N-1, p)$ steps, we arrive at the full least-squares solution.

If $p > N - 1$, the LARS algorithm reaches a zero residual solution after $N - 1$ steps (the -1 is because we have centered the data).
The *lasso modification* in the fourth step is an efficient way of computing the solution to any lasso problem, especially when $p \gg N$.

1.5 The Elastic Net

There are many applications where strong correlations among variables are observed as in genomic where genes tend to operate in molecular pathways. Therefore, researchers developed new methods of penalized regression that go beyond the limits of lasso, maintaining its good statistical properties.

The elastic net proposed by [Zou and Hastie (2005)] is an improved version of the lasso. The authors proposed a penalty function that is a combination between the ridge and the lasso penalty. It has the form

$$J(\beta, \lambda_1, \lambda_2) = \sum_{j=1}^{p} \left[\lambda_1 |\beta_j| + \lambda_2 \beta_j^2\right]$$

The second term (ridge penalty) encourages highly correlated features to be averaged, while the first term (the lasso penalty) encourages a sparse solution in the coefficients of these average features. In other words, the elastic-net selects variables like the lasso and shrinks together the coefficients of correlated predictors like the ridge regression.

The elastic net penalty can be used with any linear model, in particular for regression or classification. Simulation studies and real data examples show that the elastic net often outperforms the lasso in terms of prediction accuracy. It is like a stretchable fishing net that retains 'all the big fish'. Hence, when the number of predictors p is much bigger than the number of the observations N, as in the case of non-orthogonal designs, the elastic net is preferable to lasso.

Consider the sample (x_i, y_i), $i = 1, 2, \ldots, N$, of i.i.d. random vectors, where $x_i = (x_{i1}, x_{i2}, \ldots, x_{ip})$ is the row vector of p observations (of p explanatory variables) for the ith sample unit and y_i is the response vector for the same unit. Without loss of generality, we assume that response is centered and predictors are standardized. That is, $\sum_{i=1}^{N} y_i = 0$, $\sum_{j=1}^{N} x_{ij} = 0$ and $\sum_{j=1}^{N} x_{ij}^2 = 1$, for $j = 1, 2 \ldots, p$.

For any fixed non-negative λ_1 and λ_2, the elastic-net estimator is defined as follows:

$$\hat{\beta}_{(enet)} = (1 + \lambda_2) \underset{b \in \mathbb{R}^p}{\operatorname{argmin}} \left[\frac{1}{N} \sum_{i=1}^{N} (y_i - x_i b)^2 + \lambda_1 \sum_{j=1}^{p} |b_j| + \lambda_2 \sum_{j=1}^{p} b_j^2\right]. \tag{1.16}$$

In matrix form equation (1.16) can be written as

$$\begin{aligned} \hat{\beta}_{(enet)} &= (1 + \lambda_2) \underset{b \in \mathbb{R}^p}{\operatorname{argmin}} \left[\frac{1}{N} ||y - Xb||_2^2 + \lambda_1 ||b||_1 + \lambda_2 ||b||_2^2\right] \qquad (1.17) \\ &= (1 + \lambda_2) \underset{b \in \mathbb{R}^p}{\operatorname{argmin}} \left[\frac{1}{N} (Y - Xb)^T (Y - Xb) + \lambda_1 \mathbf{1}_p b + \lambda_2 b^T b\right]. \end{aligned}$$

If the predictors are not standardized, then $1+\lambda_2$ changes to $1+\lambda_2/N$, as discussed in [Zou and Hastie (2005)].

The two terms of the penalty function play two complementary rôles. The ℓ_1 part of the elastic-net performs automatic variable selection, while the ℓ_2 part stabilizes the solution paths and, hence, improves the prediction. So when the correlations among the predictors become high, the elastic-net can significantly improve the prediction accuracy of lasso. [Zou and Hastie (2005)] proposed the LARS-EN method, an algorithm for computing the entire elastic net regularization path efficiently.

There are two tuning parameters in the elastic net, so one need to cross-validate on a two-dimensional surface. Two dimensional CV is computationally thrifty in the usual $n > p$ setting. In the $p \gg n$ case, the cost grows linearly with p and is still menageable.

1.6 The Adaptive Lasso

Recent work suggests that the traditional lasso estimator may not be fully efficient [Fan and Li (2001)], and its model selection result could be inconsistent [Zou (2006)]. The lasso asymptotic setup is somewhat unfair, because it forces the coefficients to be equally penalized in the ℓ_1 penalty.

To fix this problem, [Zou (2006)] proposed a new version of the lasso, the *adaptive lasso*, in which adaptive weights are used for penalizing different coefficients in the ℓ_1 penalty. This method replaces the ℓ_1-norm penalty by a re-weighted version. The fundamental idea behind adaptive lasso is that, by allowing a relatively higher penalty for zero coefficients and lower penalty for nonzero coefficients, it is possible to reduce the estimation bias and improve variable selection accuracy, compared with the standard lasso. Adaptive lasso is an effective way to address some bias problems of the (one-stage) lasso which may employ strong shrinkage of coefficients corresponding to important variables. In addition, the adaptive lasso is much more insensitive to many additional noise covariates.

Consider a sample (x_i, y_i), $i = 1, 2, \dots, N$ of i.i.d. random vectors, where $x_i = (x_{i1}, x_{i2}, \dots, x_{ip})$ is the row vector of p explanatory variables for the ith sample unit and y_i is the response vector for the same unit. Without loss of generality, we assume that the response is centered so the intercept is not included in the regression function. The adaptive lasso estimator is defined by

$$\hat{\beta}_{(\text{adapt})} = \underset{b\in\mathbb{R}^p}{\operatorname{argmin}} \left[\frac{1}{N}\sum_{i=1}^{N}(y_i - x_i b)^2 + \lambda_{\text{init}}\lambda_{\text{adapt}} \sum_{j=1}^{p} \frac{|b_j|}{|\hat{\beta}_{j,\text{init}}|} \right]. \qquad (1.18)$$

In matrix form equation (1.18) can be written as

$$\hat{\beta}_{(\text{adapt})} = \underset{b\in\mathbb{R}^p}{\text{argmin}} \left[\frac{1}{N}||y - Xb||_2^2 + \lambda_{\text{init}}\lambda_{\text{adapt}} \left|\left| \frac{b}{\hat{\beta}_{\text{init}}} \right|\right|_1 \right]. \tag{1.19}$$

Here, $\hat{\beta}_{\text{init}}$ is the one-stage lasso estimate defined in (1.2), with initial tuning parameter $\lambda = \lambda_{\text{init}}$, $\lambda_{\text{adapt}} > 0$ is the tuning parameter for the second stage, and $b/\hat{\beta}_{\text{init}}$, with $b, \hat{\beta}_{\text{init}} \in R^p$, is the element by element (element-wise) division of b by $\hat{\beta}_{\text{init}}$.

The adaptive-lasso is therefore a two-stage procedure.

1. In the first step, a standard lasso regression is estimated on sample data using the regularization parameter λ_{init}. A vector $\hat{\beta}_{\text{init}}$ of regression coefficients is obtained.

2. In the second step, the weighted lasso defined by

$$\hat{\beta}_{(\text{weight})} = \underset{b\in\mathbb{R}^p}{\text{argmin}} \left[\frac{1}{N}\sum_{i=1}^{N}(y_i - x_i b)^2 + \lambda_{\text{init}}\lambda_{\text{weight}} \sum_{j=1}^{p} w_j|b_j| \right] \tag{1.20}$$

is applied on the same sample data using the following weights

$$w_j = \frac{1}{|\hat{\beta}_{j,\text{init}}|}, \quad j = 1, 2, \dots, p. \tag{1.21}$$

When $|\hat{\beta}_{j,\text{init}}| = 0$ the jth covariate is excluded in the second stage.

From a practical point of view, the estimation of optimal λ_{init} and λ_{adapt} is often based on cross-validation (see for example [Fan and Lv (2010)] and [Huang et al. (2008a)]).

In their numerical studies, [Huang et al. (2008b)] split the simulated data into a training set and a testing set. For both the lasso and Adaptive lasso, tuning parameters are selected based on k-fold cross-validation using the training set only. After tuning parameter selection, the lasso and adaptive lasso estimates are computed using the training set again. Then they compute the prediction mean square error on the test set, using the training set estimates.

1.7 Smoothly Clipped Absolute Penalty

According to [Fan and Li (2001)], a 'good' penalty function should result in an estimator with three properties.

1. *Unbiasedness*: The resulting estimator is nearly unbiased when the true unknown parameter is large to avoid unnecessary modeling bias.

2. *Sparsity*: The resulting estimator has a thresholding rule that sets small estimated coefficients to zero to reduce model complexity.

3. *Continuity*: The resulting estimator is a continuous function of data to avoid instability in model prediction.

[Fan and Li (2001)] discuss various penalty functions in term of the above three properties and establish the conditions for a penalty that meets the requirements. They propose an appealing penalization technique able to produce sparse solutions (i.e. many estimated coefficients are zero), continuous models and variable selection procedures (hence more stable than the subset selection, which is a discrete and non-continuous process), and nearly unbiased estimates for large coefficients. They call this new penalty function SCAD (Smoothly Clipped Absolute Deviation), it is symmetric, nonconcave on $(0, \infty)$, and has singularities at the origin producing sparse solutions (see [Fan (1997)]).

Consider a sample (x_i, y_i), $i = 1, 2, \ldots, N$ of i.i.d. random vectors, where $x_i = (x_{i1}, x_{i2}, \ldots, x_{ip})$ is the row vector of p explanatory variables for the ith sample unit and y_i is the response vector for the same unit. Without loss of generality, we assume that response is centered so the intercept is not included in the regression function.

The SCAD estimator is defined by

$$\hat{\beta}_{(\text{scad})} = \underset{b \in \mathbb{R}^p}{\operatorname{argmin}} \left[||y - Xb||_2^2 + N \sum_{j=1}^{p} p_\lambda(b_j, a) \right] \tag{1.22}$$

where $p_\lambda(b_j, a)$ is the SCAD penalty given by

$$p_\lambda(b_j, a) = \begin{cases} \lambda |b_j|, & \text{if } |b_j| \leq \lambda, \\ -(b_j^2 - 2a\lambda|b_j| + \lambda^2)/[2(a-1)], & \text{if } \lambda < |b_j| \leq a\lambda, \\ (a+1)\lambda^2/2, & \text{if } |b_j| > a\lambda \end{cases} \tag{1.23}$$

with $a > 2$ and $\lambda > 0$.

Figure 1.4 shows an example of SCAD penalty function, compared to lasso and ridge penalties.

The first derivative of the SCAD penalty is

$$p'_\lambda(b_j, a) = \begin{cases} \operatorname{sign}(b_j)\lambda, & \text{if } |b_j| \leq \lambda, \\ \operatorname{sign}(b_j)(a\lambda - |b_j|)/(a-1), & \text{if } \lambda < |b_j| \leq a\lambda, \\ 0, & \text{if } |b_j| > a\lambda. \end{cases} \tag{1.24}$$

This expression can be written in a more compact form as follows

$$p'_\lambda(b_j, a) = \lambda \left[I(|b_j| \leq \lambda) + \frac{(a\lambda - b_j)_+}{(a-1)\lambda} I(|b_j| > \lambda) \right]. \tag{1.25}$$

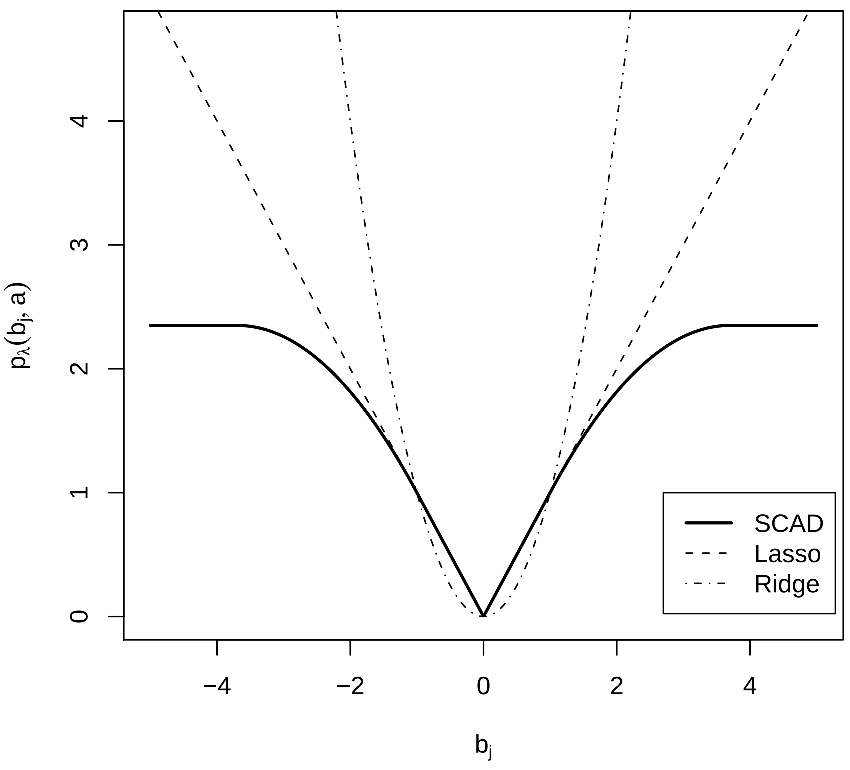

Figure 1.4: SCAD penalty with $a = 3.7$ and $\lambda = 1$, compared to lasso and ridge penalties.

The resulting thresholding rule is

$$\hat{\beta}_{j(\text{scad})} = \begin{cases} \text{sign}(\beta_j)(|\beta_j| - \lambda)_+, & \text{when } |\beta_j| \leq 2\lambda, \\ \left[(a-1)\beta_j - \text{sign}(\beta_j)a\lambda\right]/(a-2), & \text{when } 2\lambda < |\beta_j| \leq a\lambda, \\ \beta_j, & \text{when } |\beta_j| > a\lambda \end{cases} \tag{1.26}$$

An example of SCAD thresholding rule is given in figure 1.5 and is compared to the lasso and the hard thresholding rules.

The above thresholding rule involves two unknown parameters: λ and a. [Huang and Xie (2007)] suggested using generalized cross validation to estimate these parameters. In practice, one could search the best pair over the two-dimensional (λ, a) grid, that is the pair that minimizes the GCV score. Unfortunately, such an implementation can be computationally expensive. Using numerical simulations, the authors showed that the GCV score does not change much with a given λ. So, to improve computing efficiency, they proposed fixing $a = 3.7$, as suggested by [Fan and Li (2001)].

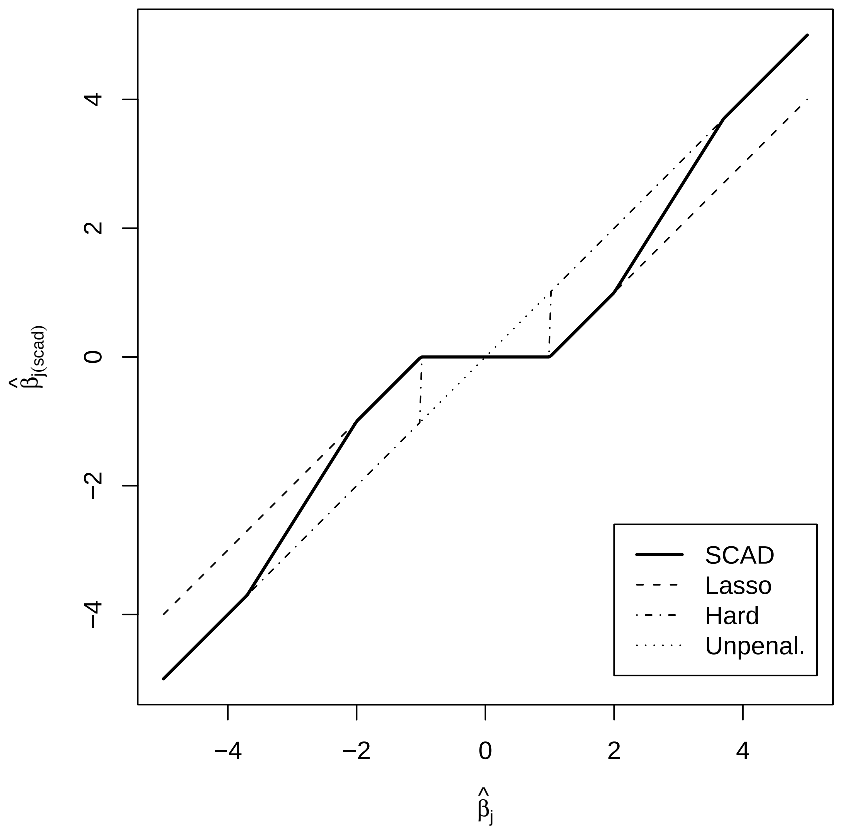

Figure 1.5: SCAD thresholding rule with $a = 3.7$ and $\lambda = 1$, compared to the lasso thresholding rule and unpenalized rule.

[Kim et al. (2008)] proved some interesting properties of the SCAD estimator. First, asymptotically, this estimator has the oracle property when the number of predictive variables are fixed or diverge more slowly than the sample size. In addition, the SCAD can achieve model selection consistency and optimal prediction simultaneously in high dimensional cases, which is impossible for lasso. In terms of prediction accuracy, numerical simulations show that the SCAD estimator is inferior to the oracle estimator. The authors claim that this might be in part due to the suboptimality of the selected regularization parameter.

[Kim et al. (2008)] also developed an efficient optimization algorithm for SCAD in high dimensions, showing that this method is computationally feasible for high-dimensional data.

Although the SCAD penalty and its related non-concave penalization techniques have produced some remarkable results, there are still concerns about applying SCAD to ultrahigh-dimensional problems. For example, a typical microarray data set often has many thousands of predictors (up to 500,000 genes) and small samples. The number of predictors in this case is much larger than the number of observations ($p \gg n$), and correlations among the predictors could be very high.

1.8 Combined Penalization

[Wang et al (2010)] proposed a new penalized least squares approach that outperforms SCAD when the number of predictors p is much larger then the number of observations n and/or the correlation among predictors is high. This approach can be used in high dimensional statistical problems.

The new regularization technique modifies the SCAD penalty by adding a quadratic penalty item. The combined penalty (CP) can be written as

$$J_{\lambda,v}(\beta) = \frac{v}{2}\beta^2 + P_\lambda(\beta), \quad \beta > 0 \tag{1.27}$$

where $P_\lambda(\beta)$ is the SCAD penalty of equation (1.23). Figure 1.6 depicts scatter plots of the combined penalty function with $\lambda = 1$ and $v = 0, 0.01, 0.1$, and 0.2, respectively.

The proposed penalty function is a mixture of the ridge and SCAD penalization techniques. The authors show that this mixture will combine the advantages of the two techniques, thereby minimizing the limitations of both. The ridge and SCAD techniques are special cases of the combined penalty, with $\lambda = 0$ for ridge and $v = 0$ for SCAD.

The resulting thresholding rule is:

$$\hat{\beta} = \begin{cases} \operatorname{sgn}(b_j)(|b_j| - \lambda)_+/(1+v), & \text{when } |b_j| \le (2+v)\lambda, \\ \{(a-1)b_j - \operatorname{sgn}(b_j)a\lambda\}/\{(a-1)(1+v)-1\}, & \text{when } (2+v)\lambda < |b_j| \le a(1+v)\lambda, \\ b_j/(1+v), & \text{when } |b_j| > a(1+v)\lambda. \end{cases} \tag{1.28}$$

for $a > 2$ and $\beta > 0$. In the original SCAD, [Fan and Li (2001)] suggest setting $a = 3.7$.

SCAD and CP functions have sparsity and continuity (see Figure 1.7). The CP function does not produce a *nearly unbiased* estimator for any $v > 0$, but rather an *asymptotically unbiased* estimator as $v \to 0$. The authors prove that this *asymptotic unbiasedness* also leads to the *oracle* property.

The CP function can be applied to high-dimensional data analysis with $p \gg n$, and also to situations where there are high correlations among predictors. Theoretical results, simulations and empirical studies show that the CP technique has superior performance compared to ridge, lasso, SCAD and elastic net. [Wang et al (2010)] also shows that combined penalization technique can be extended to general models, including logistic regression.

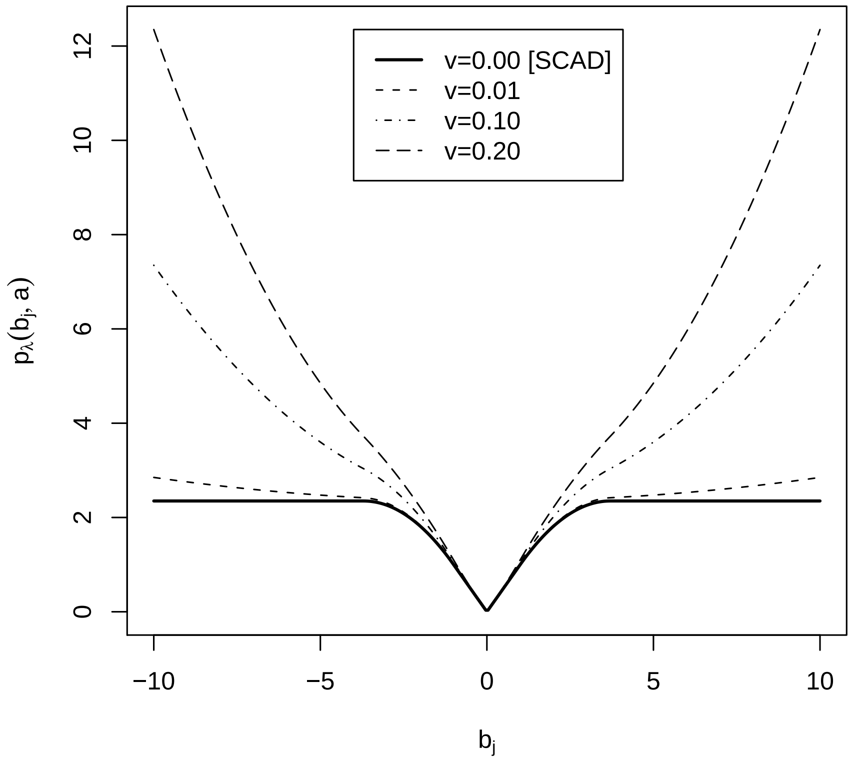

Figure 1.6: Scatter plots of combined penalty function with $\lambda = 1$ and $v = 0, 0.01, 0.1$, and 0.2, respectively.

1.9 The Relaxed Lasso

In section 1.3 we have seen that the ordinary Lasso estimator has two effects, model selection and shrinkage estimation. On the one hand, a certain set of coefficients is set to zero and hence excluded from the selected model. On the other hand, for all variables in the selected model (i.e. with non-zero coefficients), the coefficients are shrunken towards zero compared to the least-squares solution.

An interesting question is whether it is indeed optimal to control these two effects, model selection on the one hand and shrinkage estimation on the other hand, by a single parameter only. In fact, in some situations it might be desirable to estimate the coefficients of all selected variables without shrinkage (the so-called hard-thresholding).

[Meinshausen (2007b)] proposed a generalization of both soft- and hard-

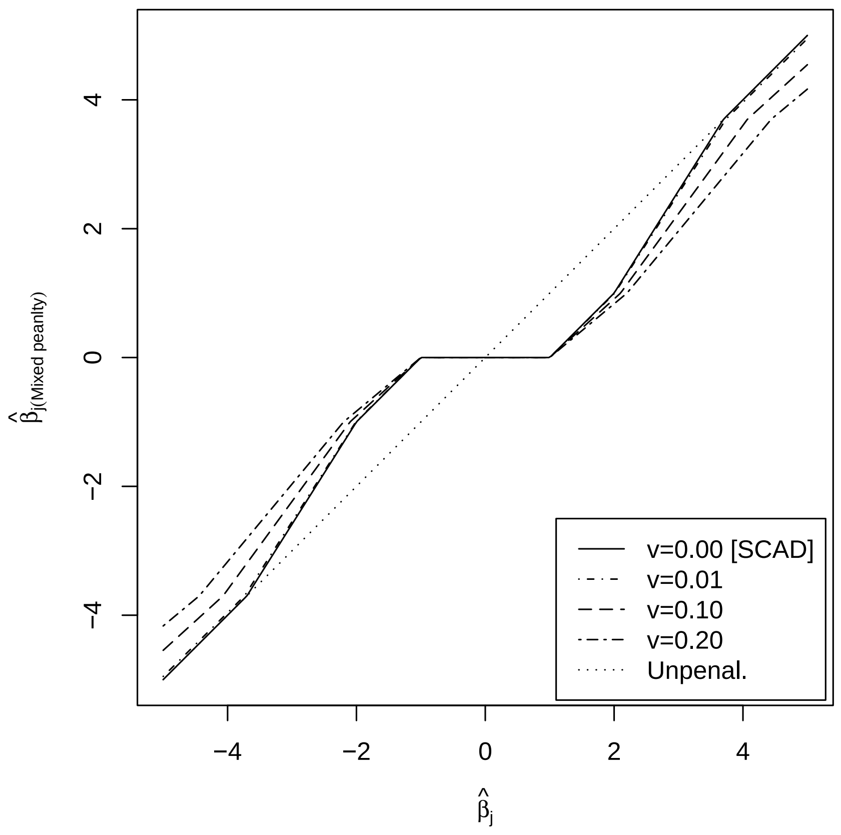

Figure 1.7: Plot of the Mixed threshold function.

thresholding, introducing a two-stage procedure, named the Relaxed Lasso. The main characteristic of this method is that it controls model selection and shrinkage estimation using two separate parameters, λ and ϕ.

Before describing the algorithm, we give some definitions. Let $\mathcal{M}_\lambda$ be the set of predictor variables selected by the lasso estimator $\hat{\beta}^\lambda$

$$\mathcal{M}_\lambda = \left\{ 1 \leqslant k \leqslant p | \hat{\beta}_k^\lambda \neq 0 \right\}, \tag{1.29}$$

where $\hat{\beta}_k^\lambda$ is the kth lasso solution corresponding to the λ penalty.

The relaxed lasso estimator defined for $\lambda \in [0, \infty)$ and $\phi \in (0, 1]$ is given by

$$\hat{\beta}^{\lambda,\phi} = \underset{\mathrm{b}}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} [y_i - x_i^{\mathrm{T}}(\mathrm{b} \cdot \mathbb{I}_{\mathcal{M}_\lambda})]^2 + \phi\lambda ||\mathrm{b}||_1, \tag{1.30}$$

where $\mathbb{I}_{\mathcal{M}_\lambda}$ is the indicator function on the set of variables $\mathcal{M}_\lambda \subseteq \{1, \ldots, p\}$, defined by

$$\{\beta \cdot \mathbb{I}_{\mathcal{M}_\lambda}\}_k = \begin{cases} 0, & k \notin \mathcal{M}_\lambda, \\ \beta_k, & k \in \mathcal{M}_\lambda. \end{cases} \tag{1.31}$$

This means that the relaxed lasso applies the lasso estimation twice: the first lasso selects variables using the λ penalty (thus it identifies the set $\mathcal{M}_\lambda$) and the second lasso estimates the model coefficients of the selected variables controlling the shrinkage by the relaxation parameter ϕ.

It is interesting to note that:

1. for $\phi = 0$, the relaxed lasso overlaps lasso estimation;

2. for $\phi < 1$, the shrinkage of the coefficients in the selected model is less than the shrinkage of the lasso estimation.

[Meinshausen (2007b)] developed an algorithm to compute the exact solutions of the relaxed lasso estimator. The algorithm can be summarized as follows.

1. Compute the entire path of ordinary lasso solutions. Let $\mathcal{M}_1, \ldots, \mathcal{M}_m$ be the resulting set of m models. Let $\lambda_1, \ldots, \lambda_m$ (where $\lambda_m = 0$) be a sequence of penalty values such that $\mathcal{M}_\lambda = \mathcal{M}_k$ if and only if $\lambda \in (\lambda_k, \lambda_{k-1}]$ (where $\lambda_0 = \infty$).

2. For each $k = 1, \ldots, m$, compute the entire path of lasso solutions on the set $\mathcal{M}_k$ of variables, varying the penalty parameter between 0 and λ_k. This set of solutions is the set of relaxed lasso solutions $\hat{\beta}^{\lambda,\phi}$, with $\lambda \in (\lambda_k, \lambda_{k-1}]$.

The relaxed Lasso solutions for all penalty parameters are given by the union of these sets. The parameters λ and ϕ can be chosen by cross-validation.

[Meinshausen (2007b)] shows that this algorithm can be refined because the ordinary Lasso solutions contain information that can be exploited in the second stage. The description of this algorithm goes beyond the scope of the present work.

The most common way for choosing the penalty parameter for the lasso is by cross-validation. This technique often erroneously selects many noise variables when the number of variables is large and the accuracy of its prediction is negatively affected by the presence of many noise variables, particularly for high signal-to-noise ratios.

There is almost no differences between lasso and relaxed lasso results when the number of informative variables is large. On the contrary, when there is a very large number of noise variables, the relaxed lasso estimator selects a much smaller number of covariates. This property produces models that are more amenable to interpret and with a much smaller mean square error. In other words, the relaxed lasso is performing better then lasso when just a few variables are carrying signal. The sparseness and the signal-to-noise ratio is, in general, unknown and the relaxed lasso is adaptive to both.

1.10 R packages

A large number of R packages that implements different forms of penalized regression are currently available. There are packages that fits generalized regression problems while imposing a constraint on parameters, for the estimation of the entire ridge, lasso, adaptive lasso or elastic-net regularization path for linear, logistic, multinomial, gamma, inverse gaussian, poisson regression models and the Cox model. Many cross-validation routines allow optimization of the tuning parameters.

In this section we present a brief overview of the main R packages for penalized regression.

Package name	Description	Characteristics	Commands	Tuning parameters
`biglars`	Least-Angle and Lasso regression for datasets with $n << p$ [Fraley and Hesterberg (2007)]	Least-Angle, Lasso and Stepwise Regression for linear regression	`biglars.fit`	-
`elasticnet`	Functions for fitting the entire solution path of the Elastic-Net [Zou and Hastie (2005)]	Elastic Net regression models and sparse Principal Components; k-fold cross-validated error curve for elastic net	`enet, cv.enet, predict.enet, spca`	`lambda`
`frailtypack`	Routines for fitting several classes of frailty models using penalized likelihood on the hazard function. [Rondeau, Commenges and Joly (2003)]	Frailty model using penalized likelihood estimation: shared gamma, joint, nested, additive; cross-validation for optimal penalty parameters `kappa1, kappa2`	`frailtyPenal, additivePenal`	`kappa1, kappa2`
`glmnet`	Efficient procedures for fitting the entire lasso or elastic-net regularization path for some GLM [Friedman et al. (2010)]	Elastic net model paths for linear, logistic, multinomial, poisson and Cox models; k-fold cross-validation for optimal penalty	`frailtyPenal`	`lambda, alpha`
`glmpath`	A path-following algorithm for some L1 regularized GLM [Park M.Y. and Hastie (2007)]	Regularization path for linear, logistic, poisson and Cox models with L1 penalty; k-fold cross-validation for optimal penalty; a regularization parameter for the L2 norm of the coefficients is also available	`glmpath, coxpath, cv.glmpath, cv.coxpath`	`s` (in `predict.glmpath` and `predict.coxpath` when `mode = lambda`, `s` is λ), `lambda2`

`grplasso`	Algorithms for fitting the regularization path of GLMs with group lasso penalty [Meier, van de Geer, Bühlmann (2008)]	Paths of a group lasso problem for linear, logistic and poisson model	`grplasso, predict.grplasso`	`lambda`
`grpreg`	Efficient algorithms for fitting the regularization path for penalized linear or logistic regression models [Breheny and Huang (2009)]	Paths for group lasso, group bridge and group MCP at a grid of values of the regularization parameter `lambda`; selection of `lambda` using AIC, BIC and GCV criteria	`grpreg, select, cv.glmpath, cv.coxpath`	`lambda, lambda2, delta, gamma, a`
`lars`	Efficient procedures for fitting an entire lasso sequence with the cost of a single least squares fit [Efron, Hastie, Johnstone and Tibshirani (2003)]	Least Angle Regression, Lasso and Infinitesimal Forward Stagewise regression models; computes k-fold cross-validated error curve for lars	`lars, cv.lars, predict.lars`	`s` in `predict.lars`
`lasso2`	Routines for solving regression problems while imposing an L1 constraint on the estimates [Osborne, Presnell, Turlach (2000)]	Fits a generalized regression problem while imposing an L1 constraint on parameters; supported families: gaussian, binomial, poisson, Gamma, inverse.gaussian and quasi; GCV score for selecting optimal `lambda`	`l1ce, gl1ce, gcv`	`bound`
`lqa`	Routines for fitting GLMs via penalized likelihood inference. Estimating procedures are the LQA algorithm, P-IRLS, RidgeBoost, GBlockBoost and ForwardBoost [Ulbricht (2010)]	Penalized GLMs with the following penalty functions: lasso, adaptive lasso, approximated octagon, bridge, elastic net, fused lasso, correlation based, OSCAR, ridge, SCAD, weighted fusion; estimation of optimal tuning parameters via CV or validation data	`lqa, lqa.cv, ForwardBoost, GBlockBoost, cv.nng, nnls, nnls2`	`lambda`
`ncvreg`	Efficient algorithms for fitting regularization paths for linear or logistic regression models penalized by MCP or SCAD	Paths for MCP- or SCAD-penalized (linear, logistic) regression models over a grid of values for the regularization parameter `lambda`; additional L2 penalty; k-fold cross-validation for optimal `lambda`	`ncvreg, cv.ncvreg`	`lambda, gamma, alpha`

`parcor`	Algorithms for estimating the matrix of partial correlations based on a matrix of observations [Kraemer, Schaefer and Boulesteix (2009)]	Four regularized regression techniques for the estimation of partial correlations: lasso, adaptive lasso, ridge regression, and Partial Least Squares. Provides CV model selection for lasso, adaptive lasso and Ridge regression	`mylars, adalasso, adalasso.net, rigde.cv, ridge.net, pls.net`	-
`penalized`	Efficient procedures for fitting the entire lasso or elastic-net regularization path for some GLM [Goeman (2010)]	Elastic net model paths for linear, logistic, poisson and Cox models; k-fold cross-validation for optimal `lambda1` and `lambda2`; positivity constraint on regression coefficients	`penalized, optL1, optL2, profL1, profL2, plotpath`	`lambda1, lambda2`
`penalizedSVM`	Algorithms for feature selection with support vector machines (SVMs) using penalty functions [Zhang *et al.* (2006)]	Smoothly clipped absolute deviation (SCAD) and L1-norm penalties; approximated GCV for SCAD SVM model	`svm.fs, lpsvm, scadsvc, findgacv.scad`	`lambda1.set` (in `svm.fs`), `lambda` (in `scadsvc`)
`plus`	Efficient procedures for fitting the entire piecewise linear regularization path for some penalized linear models [Zhang C.-H. (2007)]	Penalized Linear Unbiased Selection; fits linear regression with a quadratic spline penalty, including the Lasso, MC+ and SCAD	`plus, predict.plus`	`lam` and `m` (in `plus`)
`quantreg`	Quantile regression and related methods [Koenker (2005)]	Lasso penalized quantile regression	`rq.fit.lasso`	`lambda`
`relaxo`	Relaxed Lasso: a generalisation of the Lasso shrinkage technique for linear regression [Meinshausen (2007b)]	In relaxed lasso variable selection and parameter estimation is achieved by regular Lasso; the two steps use different penalty parameters: `lambda` and `phi`	`relaxo, cvrelaxo, predict.relaxo`	`lambda` (in `predict.relaxo`) and `phi` (in `relaxo`, `cvrelaxo` and `predict.relaxo`)
`SIS`	(Iterative) Sure Independence Screening (SIS) for GLMs and Cox's model [Fan, Samworth and Wu (2009)]	Different variants of (I)SIS: vanilla (I)SIS and two variants	`SIS, scadglm, scadcox, fullscadglm, fullscadcox, wtlassoglm, wtlassocox`	`nsis`

Chapter 2

Bootstrap for GLM

2.1 Introduction

In this chapter, we consider the application of bootstrap methods to regression models. Our interest is focused on the construction of bootstrap confidence intervals for the coefficients of the class of generalized linear models (GLMs) and penalized GLMs.

We start considering some basic notions about bootstrap and GLMs and introducing mathematical notation that will be useful in the sections that follow. Successively, we consider four bootstrap methods that approximate the distribution of model coefficients:

- parametric bootstrap;
- vector resampling;
- residual bootstrap;
- one-step bootstrap.

All these methods are tested by numerical simulations on a linear and a logistic data generating process (DGP). Their performances are quantified, calculating their approximation error about bias and variance of model coefficients and considering the distance between the distribution of model coefficients and the distribution of bootstrap replications.

The R codes used to perform these simulation studies are also considered.

The simulation studies presented in this and the next chapter are based on two data generating processes. The first, DGP, is the linear model used in [Chatterjee and Lahiri (2010)]:

$$y = X\beta + \varepsilon, \qquad \varepsilon \sim N(0,1) \quad \text{(i.i.d.)} \tag{2.1}$$

where:

- y is the response variable and X is an orthogonal $(n \times m)$ design matrix with $n = 250$ and $m = 10$;
- the m columns of X are generated independently from the uniform distribution in the interval $[-\frac{1}{2}, \frac{1}{2}]$;
- ε is a vector of n independent and identically distributed (i.i.d.) standard gaussian errors $N(0,1)$;
- β is the vector of m regression coefficients with values:

  $$\beta = (0.75, 2, 0, 0, 0, 10, -3, 1.5, 0, -0.65)^{\mathrm{T}},$$

  that is, there are $m_I = 6$ informative variables.

The second DGP used in our numerical simulations is the following logistic model:

$$y \sim Bern(p), \qquad p = \frac{1}{1 + \exp^{-\eta}}, \qquad \eta = X\beta, \tag{2.2}$$

where :

- y is the response variable and X is an orthogonal $(n \times m)$ design matrix with $n = 250$ and $m = 10$;
- the m columns of X are generated independently from the uniform distribution in the interval $[-\frac{1}{2}, \frac{1}{2}]$;
- p is the vector of success probabilities of the Bernoulli probability distribution;
- β is the vector of m regression coefficients with values:

  $$\beta = (0.75, 2, 0, 0, 0, 10, -3, 1.5, 0, -0.65)^{\mathrm{T}},$$

  that is, there are $p_I = 6$ informative variables.

2.2 Basic notions on bootstrap

The bootstrap is a data-based simulation method for statistical inference that was first introduced by Bradley Efron in the seminal paper [Efron (1979)]. It is a general approach to statistical inference intended to help avoid tedious calculations based on questionable assumptions. Bootstrap methods are computer-intensive techniques that can simplify calculations, are straightforward to apply, and can yield reliable standard errors, confidence intervals, and other measures of uncertainty for a wide range of problems, including complex estimators of complex parameters. The goal of bootstrap is a computer-based implementation of basic statistical concepts.

To understand the basic idea behind the bootstrap, it is useful to start considering vector resampling (a simple and popular bootstrap method also known as case resampling) for the assessment of the uncertainty about the coefficients of a regression model.

The basic idea behind the bootstrap is to generate a large number of bootstrap samples by sampling with replacement from the original data set (see figure 2.1). We denote the original training set by $\mathbf{Z} = (z_1, z_2, \dots, z_n)$ where $z_i = (x_i, y_i)$. A bootstrap sample $\mathbf{Z}^{*b}$ is generated by sampling with replacement n times from the original datasets Z, with probability $1/n$. Drawing B times, we produce B independent bootstrap data sets $\mathbf{Z}^{*1}, \dots, \mathbf{Z}^{*B}$.

Let $S(Z)$ be the quantity of interest computed from the data set Z, that is the vector of coefficients of the GLM estimated using Z. The next step of the bootstrap algorithm is the calculation of the statistic S on each bootstrap training set. We obtain B bootstrap replicates $S(\mathbf{Z}^{*1}), \dots, S(\mathbf{Z}^{*B})$ that can be used to assess many aspects of the distribution of S. For example, using the bootstrap replicates we can estimate its variance:

$$\widehat{Var}_{\text{boot}}(S) = \frac{1}{B-1} \sum_{b=1}^{B} (S(\mathbf{Z}^{*b}) - \bar{S}^*)^2,$$

where $\bar{S}^* = \sum_b S(\mathbf{Z}^{*b})/B$. It is possible to prove that under mild conditions, vector resampling yields a ‘good’ approximation of the distribution of the vector of model coefficients.

It is worth to note that a particular bootstrap method may give good approximations of some quantities but bad approximations for other quantities. For example, suppose we want to estimate the prediction error of a model using vector resampling. We fit the model on each bootstrap samples and keep track of how well it predicts the original training set. From the B bootstrap data sets, we can estimate

$$\widehat{Err}_{\text{boot}} = \frac{1}{n \cdot B} \sum_{b=1}^{B} \sum_{i=1}^{n} L(y_i, \hat{f}^{*b}(x_i)),$$

where $\hat{f}^{*b}(x_i)$ is the predicted value at x_i.

It is easy to see that $\widehat{Err}_{\text{boot}}$, in general, is not a good estimate. The reason lies in the fact that the bootstrap data sets switch the rule of training set with the original database. So, the original training set is playing the role of the test sample. This overlapping over observations makes the predictions look unrealistically good. The disadvantage of the bootstrap to be overly optimistic is considered in [Hastie et al. (2009)] and [Davison and Hinkley (1997)].

Bootstrap methods for GLMs can be categorized into three classes.

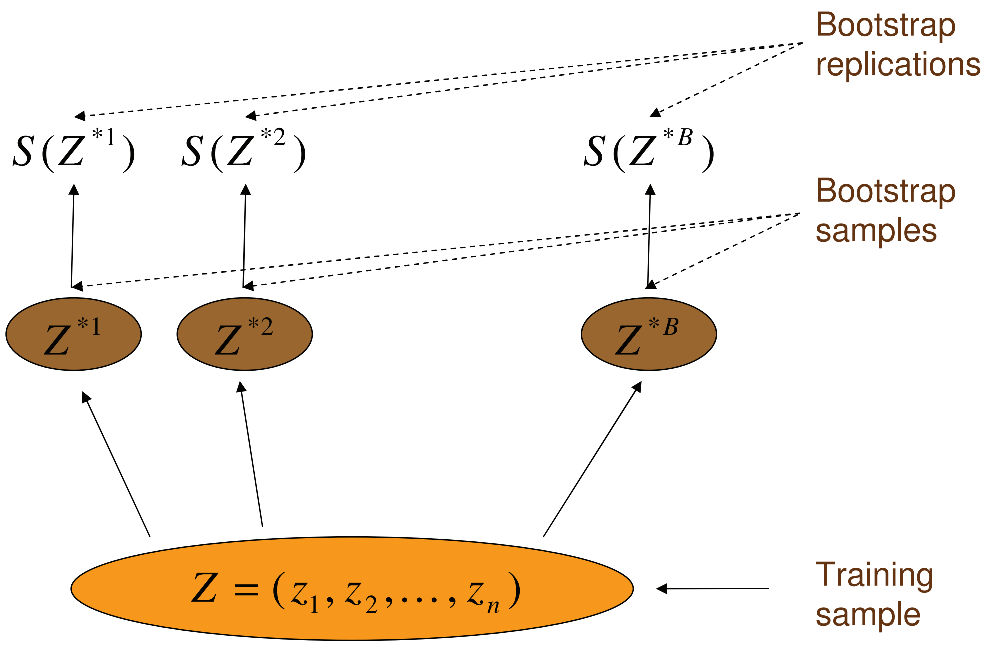

Figure 2.1: Schematic representation of the bootstrap process.

- *Parametric.* The parametric bootstrap for a GLM involves simulating new sets of data from the fitted parametric model. The performances of this method strongly depends on the goodness of fit of the model. If the model poorly fits, datasets generated using parametric bootstrap may not have the statistical properties of the original data.

- *Nonparametric.* The nonparametric approach generates artificial data without assuming that the original data have some particular parametric distribution. A simple and completely nonparametric bootstrap method is based on resampling the observed cases (vector resampling).

- *Semiparametric.* The semiparametric approach involves the resampling of model errors. It is a popular and appreciated class of bootstrap methods for GLMs. It relies on the assumption of exchangeability of the error terms.

2.3 Basic notions on GLM

The building blocks of a GLM are:

- the *stochastic component*; the distribution of the responses y_i is assumed to be a member of the exponential family. The density function of this family has the following form:

  $$f(\theta, \phi, y) = \exp\left\{ \frac{y_i \theta_i - b(\theta_i)}{a(\phi)} + c(y_i, \phi) \right\} \tag{2.3}$$

  The exponential family covers a large number of distributions. For example, discrete distributions as the bernoulli, binomial and poisson, which can handle binary and count data or continuous distributions as the normal, gamma or inverse gaussian distribution.

  The functions $a(\cdot)$, $b(\cdot)$ and $c(\cdot)$ will vary for different Y distributions. The parameter of interest is θ, which is called the canonical parameter. The additional parameter ϕ is only relevant for some of the distributions and is considered as a nuisance parameter.

  For example, the probability density function of the normal distribution is
  $$f(\mu_i, \sigma, y) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{ \frac{-(y_i - \mu_i)^2}{2\sigma^2} \right\}.$$
  This function can be written as in (2.3) by setting:

  $$\begin{aligned} &\theta_i = \mu_i, \quad \phi = \sigma, \quad a(\phi) = \phi^2, \quad b(\theta_i) = \frac{\theta_i^2}{2} \\ &\text{and} \quad c(y_i, \phi) = -\frac{y_i^2}{2\sigma^2} - \log(\sqrt{2\pi}\sigma). \end{aligned}$$

  Another example is the probability mass function of the Bernoulli distribution
  $$P(Y = y_i) = p_i^{y_i}(1 - p_i)^{1 - y_i}$$
  that can be rewritten as:

  $$f(p_i, y_i) = \exp\left\{ y_i \ln\left(\frac{p_i}{1 - p_i}\right) + \ln(1 - p_i) \right\}.$$

  This function is equivalent to (2.3) by setting:

  $$\begin{aligned} &\theta_i = \ln\left(\frac{p_i}{1 - p_i}\right) \quad a(\phi) = 1, \\ &b(\theta_i) = -\ln(1 - p_i) = \ln(1 + e^{\theta_i}), \quad c(y_i, \phi) = 0. \end{aligned}$$

The mean μ_i and the variance v_i of Y_i can be determined from the distribution (2.3) and can be expressed in terms of ϕ, θ_i and the functions $a(\cdot)$, $b(\cdot)$ and $c(\cdot, \cdot)$:

$$\begin{aligned} \mu_i &= E(Y_i, \theta, \phi) = b'(\theta_i) && (2.4) \\ v_i &= V(Y_i, \theta, \phi) = b''(\theta_i)a(\phi) = V(\mu_i)a(\phi), && (2.5) \end{aligned}$$

where b' and b'' are the first and the second derivativs of $b(\cdot)$, respectively, and $V(\mu_i) = \partial\mu/\partial\theta$.

For the normal distribution, $\mu_i = \theta_i$, $V(\mu_i) = 1$, $v_i = \sigma^2$ and for the bernoulli distribution, $\mu_i = e^{\theta_i}/(1+e^{\theta_i})$, $v_i = V(\mu_i) = e^{\theta_i}/(1+e^{\theta_i})^2 = \mu_i(1-\mu_i)$.

- the *systematic component*; the m covariates x combine linearly with the coefficients β to form the linear predictor, $\eta = X\beta$.
- the *link* between the random and the systematic components; the linear predictor $\eta = X\beta$ is a function of the mean parameter μ via a link function, $\eta = g(\mu)$.

  For the normal linear model, g is the identity function and $\mu_i = \eta_i = x_i\beta$; for the logistic model, g is the logistic function, $\eta_i = \ln\{\mu_i/(1-\mu_i)\}$ and $\mu_i = e^{x_i\beta}/(1+e^{x_i\beta}) = 1/(1+e^{-x_i\beta})$.

The log-likelihood for the exponential family is:

$$\mathcal{L}(\theta, \phi, y, X) = \sum_{i=1}^{n} \left\{ \frac{y_i\theta_i - b(\theta_i)}{a(\phi)} + c(y_i, \phi) \right\} \tag{2.6}$$

By assuming that the distribution of Y belongs to the exponential family, it is possible to derive maximum likelihood estimates for the coefficients of a GLM. Maximum likelihood estimates (MLEs) of β are generally not available in closed form, but a numerical approximation can be obtained via an algorithm known as iteratively weighted least squares (IWLS) or iteratively reweighted least squares (IRLS). IWLS is one of the key practical ways in which GLMs are, in fact, 'general'. This algorithm provides MLEs for a wide class of commonly used models.

In summary, at each iteration the IWLS algorithm performs a weighted least squares regression of the adjusted response variables, z, on the explanatory variables, X. In this regression the dependent variable is not y but z, a linearized form of the link function applied to y, and the weights, W, are functions of the fitted values, $\hat{\mu}$. The process is iterative because both the adjusted dependent variable, z, and the weight, W, depend on the fitted values, for which only current estimates are available.
The kth iteration of the IWLS algorithm consists of the following steps.

1. Calculate the adjusted responses
$$\begin{aligned} z_i^{(k)} &= \eta_i^{(k)} + (y_i - \hat{\mu}_i^{(k)})\dot{g}(\hat{\mu}_i^{(k)}) \qquad (2.7)\\ &= x_i\hat{\beta}^{(k)} + (y_i - \hat{\mu}_i^{(k)})\dot{g}(\hat{\mu}_i^{(k)}). \end{aligned}$$

2. Build the weight diagonal matrix
$$W^{(k)} = \text{diag}\left(\frac{1}{a(\phi)[\dot{g}(\hat{\mu}_1^{(k)})]^2 V(\hat{\mu}_1^{(k)})}, \dots, \frac{1}{a(\phi)[\dot{g}(\hat{\mu}_n^{(k)})]^2 V(\hat{\mu}_n^{(k)})}\right). \qquad (2.8)$$

3. Run the weighted regression of the $z_i^{(k)}$ on the covariates x_i with weights given by equation (2.8) and calculate the coefficients $\hat{\beta}^{(k)}$ given by:
$$\begin{aligned} \hat{\beta}^{(k+1)} &= (X^{\mathrm{T}}W^{(k)}X)^{-1}X^{\mathrm{T}}W^{(k)}z^{(k)} \qquad (2.9)\\ &= \beta^{(k)} + (X^{\mathrm{T}}W^{(k)}X)^{-1}X^{\mathrm{T}}W^{(k)}\Gamma^{(k)}(y_i - \hat{\mu}_i^{(k)}) \end{aligned}$$
where $\Gamma^{(k)}$ is the diagonal matrix defined by $\Gamma^{(k)} = \text{diag}(g(\hat{\mu}_1^{(k)}), \dots, g(\hat{\mu}_n^{(k)}))$.

4. Proceed to the next iteration.

This algorithm can be repeated until convergence in $\hat{\beta}$ or log-likelihood or deviance.

A reasonable approach for the initialization of the IWLS algorithm is to set the initial fitted values $\hat{\mu}_i$ to the mean of the response variable. Alternatively, another approach is to set $\hat{\mu}_i$ to $(y_i + \bar{y})/2$ for a nonbinomial model and $k_i(y_i + 0.5)/(k_i + 1)$ for a binomial model (i.e. $(y_i + 0.5)/2$ for a logistic model).

The asymptotic variance of $\hat{\beta}$ is given by
$$V(\hat{\beta}) = \phi(X^{\mathrm{T}}WX)^{-1} \qquad (2.10)$$
where W is the matrix (2.8) computed at the final iteration. Asymptotically-valid standard errors for the coefficients are obtained by taking the square root of the leading diagonal of $V(\hat{\beta})$.

For the linear regression model, the adjusted responses agree exactly with y, $z_i^{(k)} = y_i$, the weight matrix W is the identity matrix, the estimator (2.9) is the ordinary least squares estimator
$$\hat{\beta}^{(k)} = \hat{\beta} = (X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}y$$
and no iteration is necessary in the IWLS algorithm.

For the logistic regression model, we have
$$\dot{g}(\mu_i) = \frac{1}{\mu_i(1-\mu_i)} = \frac{1}{V(\mu_i)} \quad \text{and} \quad W_i = \mu_i(1-\mu_i) = V(\mu_i) = 1/\dot{g}(\mu_i).$$

We now consider a simple R code that implements the IWLS algorithm. We start generating data using a logistic DGP.

```
n <- 250
beta0 <- 0
betas <- c(0.75,2,0,0,0,10,-3,1.5,0,-0.65)
set.seed(-517072325)
m <- length(betas)
X  <- matrix(runif(n*m), nrow=n, ncol=m)-0.5
Xb <- beta0+ X %*% betas
pr <- 1/(1 + exp(-Xb))
Y <- rbinom(n = n, prob = pr, size = 1)
dset.orig <- data.frame(y=Y, X=X)
```

Then, we estimate β coefficients by IWLS:

```
X1 <- cbind(rep(1,n), X)
beta.iwls <- rep(0,m+1)
mu.hat <- rep(mean(Y),n)
nstep <- 25
k <- 1
while (k < nstep) {
  V.mu <- mu.hat*(1-mu.hat)
  W <- diag(as.numeric(V.mu))
  z <- X1%*%beta.irls + (Y-mu.hat)/V.mu
  beta.iwls <- solve(t(X1) %*% W %*% X1) %*% t(X1) %*% W %*% z
  eta <- X1 %*% beta.iwls
  mu.hat <- 1/(1+exp(-eta))
  k <- k+1
}
```

We compare these estimates to the results given by the `glm` command:

```
fit.glm <- glm(y ~ ., data = dset.orig,
             family = binomial(link = "logit"))
beta.glm <- coef(fit.glm)
res.unpen <- cbind(beta.iwls,beta.glm)
dimnames(res.unpen)[[2]] <- c("IWLS","glm")
print(res.unpen)
```

The two algorithms yield estimates that agree exactly:

```
                  IWLS        glm
(Intercept) -0.1721319 -0.1721319
X.1          1.0094538  1.0094538
X.2          1.7772169  1.7772169
X.3          0.6605817  0.6605817
X.4         -0.1508727 -0.1508727
X.5          0.1766377  0.1766377
```

```
X.6           8.5455813  8.5455813
X.7          -3.5701801 -3.5701801
X.8           1.9911067  1.9911067
X.9          -0.1011964 -0.1011964
X.10         -0.9803812 -0.9803812
```

The IWLS algorithm is the basis of the one-step bootstrap for GLM proposed by [Moulton and Zeger (1991)] and considered in section 2.7.

2.4 The parametric bootstrap for GLM

This section starts investigating parametric bootstrap for linear regression using the following simulation study:

1. $S = 500$ samples are generated using the linear DGP defined in (2.1). The R code used for this step is:

   ```
   n <- 250
   beta0 <- 0
   betas <- c(0.75,2,0,0,0,10,-3,1.5,0,-0.65)
   m <- length(betas)
   X <- matrix(runif(n*m), nrow=n, ncol=m)-0.5
   Y <- beta0 + X%*%betas + rnorm(n)
   dset.orig <- data.frame(y=Y, X=X)
   ```

2. For each sample, a linear model is fitted to data, $\hat{\beta} = (X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}Y$ coefficients and standard deviation $\hat{\sigma}$ of the component error ε are estimated, predictions $\hat{y} = X\hat{\beta}$ of the outcome y are calculated.

   ```
   glm.fit <- glm(y ~ ., data = dset.orig,
             family = gaussian(link = "identity"))
   beta.hat <- t(coefficients(glm.fit))
   y.hat <- predict(glm.fit,type="response")
   sigma.hat <- sd(glm.fit$res)
   ```

3. For each sample, $B = 500$ bootstrap samples are generated using the X matrix and the outcome $y^* \sim N(X\hat{\beta}, \hat{\sigma})$, where $\hat{\sigma}$ is the estimated standard deviation of the component error ε.

   ```
   Y.boot <-  rnorm(n, mean=y.hat, sd=sigma.hat)
   ```

4. For each bootstrap sample, a linear model is fitted to data and $\hat{\beta}^*$ bootstrap coefficients are estimated.

```
B <- 500
beta.boot <- matrix(0, B, m+1)
cnt2 <- 1
 while (cnt2 <= B) {
   Y.boot <-  rnorm(n, mean=y.hat, sd=sigma.hat)
   dset.boot <- data.frame(y = Y.boot, X = X)
   glm.fit.boot <- glm(y ~ ., data = dset.boot,
            family = gaussian(link = "identity"))
   if (glm.fit.boot$converged) {
   beta.boot.b <- coefficients(glm.fit.boot)
   beta.boot[cnt2,] <- beta.boot.b
        cnt2 <- cnt2 +1
     }
}
```

Globally, this simulation yields 500 values of the $\hat{\beta}$ estimated coefficients and $250,000$ $\hat{\beta}^*$ bootstrap estimates.

Figure 2.3 and 2.4 show the estimated density functions of the $\hat{\beta}$ and $\hat{\beta}^*$ coefficients of model (2.1).

Figure 2.2 is a Q-Q plot that shows the degree of 'closeness' between the distributions of $T_j = \sqrt{n}(\hat{\beta}_j - \beta_j)$ and $T_j^* = \sqrt{n}(\hat{\beta}_j^* - \hat{\beta}_j)$. That is, for each model coefficient, this figure compares the distribution $\hat{G}$ of the differences between bootstrap estimates $\hat{\beta}_j^*$ and $\hat{\beta}_j$ estimates to the distribution G of the differences between the original estimates $\hat{\beta}_j$ and the true values β_j. If the bootstrap method works correctly, the two resulting distributions are close and the points of the Q-Q plot lies on the dashed line.

The results of figure 2.2 evidences that the distance between the distributions G and $\hat{G}$ is small and that parametric bootstrap is able to reproduce correctly the randomness of the $\hat{\beta}$ estimated coefficients.

These results are confirmed by the empirical coverage probabilities of table 2.3. They are all close to the desired nominal coverage rate of 90%.

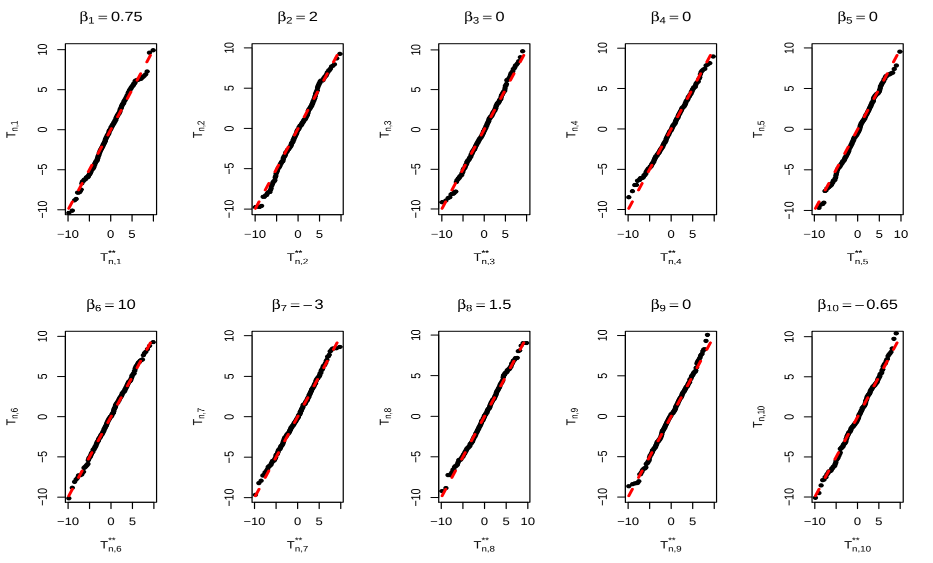

Figure 2.2: Parametric bootstrap for the linear model (2.1). Comparison of G and $\hat{G}$ distributions of the coefficients using Q-Q plots.

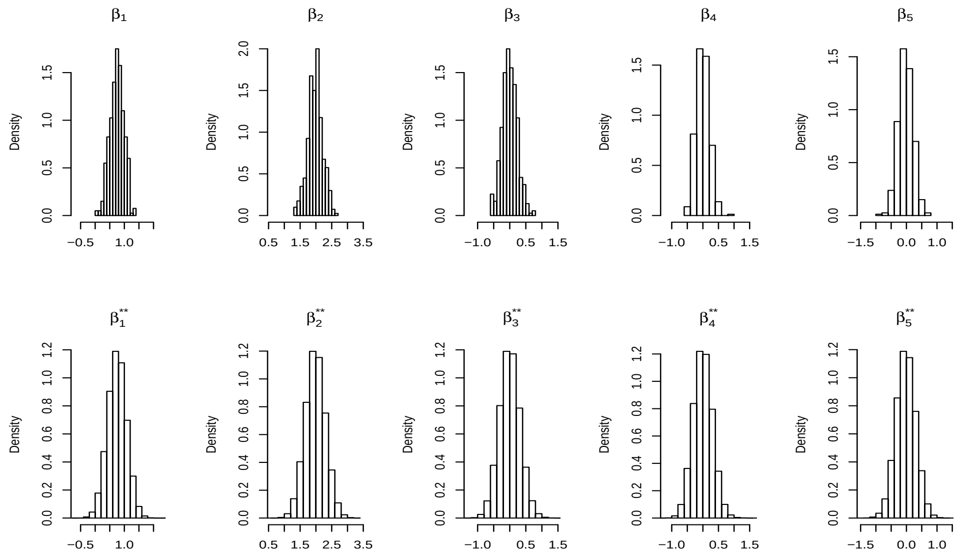

Figure 2.3: Parametric bootstrap for the first five coefficients of the linear model (2.1). Density functions of the estimated coefficients (above) and of the bootstrap estimates (below).

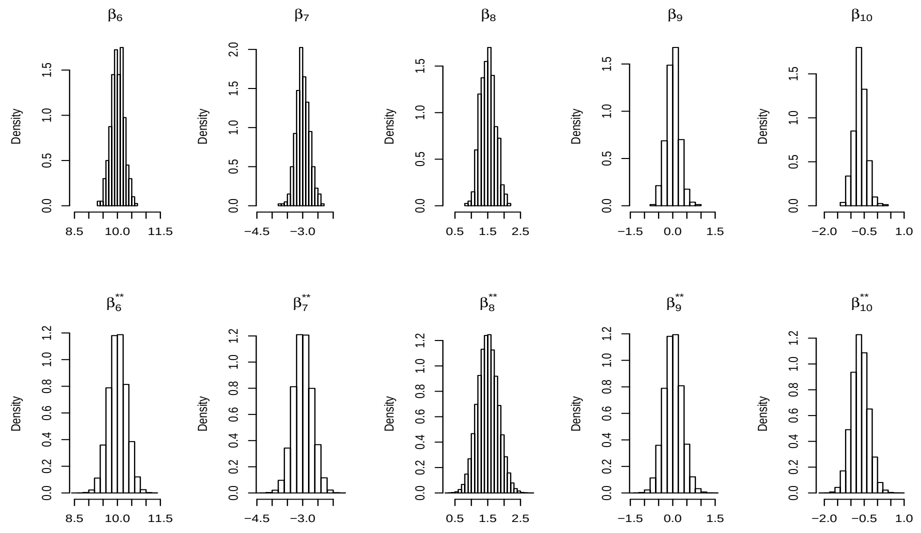

Figure 2.4: Parametric bootstrap for the last five coefficients of the linear model (2.1). Density functions of the estimated coefficients (above) and of the bootstrap estimates (below).

For comparison purposes, here we consider a numerical simulation for studying the properties of parametric bootstrap when applied to logistic regression. The simulation study of this section is organized as follows.

1. $S = 500$ samples are generated using the logistic DGP defined in equation (2.2).

```
n <- 250
beta0 <- 0
betas <- c(0.75,2,0,0,0,10,-3,1.5,0,-0.65)
m <- length(betas)
X <- matrix(runif(n*m), nrow=n, ncol=m)-0.5
Xb <- beta0 + X%*%betas
pr <- 1/(1+exp(-Xb))
Y <- rbinom(n = n, prob = pr, size = 1)
dset.orig <- data.frame(y=Y, X=X)
```

2. For each sample, a logistic model is fitted to data. Parameters $\hat{\beta}$ are estimated and the success probabilities $\hat{p}_i$ are estimated, $i = 1, 2, \dots, n$.

```
glm.fit.orig <- glm(y ~ ., data = dset.orig,
             family = binomial(link = "logit"))
beta.hat <- t(coefficients(glm.fit))
pr.hat <- predict(glm.fit.orig,type="response")
```

3. For each sample, probabilities $\hat{p}_i$ are generated using $B = 500$ bootstrap samples.

4. For each bootstrap sample, a logistic model is fitted to data and parameters $\hat{\beta}^*$ are estimated.

```
B <- 500
beta.boot <- matrix(0, B, m+1)
cnt2 <- 1
while (cnt2 <= B) {
   Y.boot <- rbinom(n = n, prob = pr.hat, size = 1)
   dset.boot <- data.frame(y = Y.boot, X = X)
   glm.fit.boot <- glm(y ~ ., data = dset.boot,
              family = binomial(link = "logit"))
   if (glm.fit.boot$converged) {
   beta.boot.b <- coefficients(glm.fit.boot)
   beta.boot[cnt2,] <- beta.boot.b
   cnt2 <- cnt2 +1
   }
}
```

The Q-Q plot of figure 2.5 shows that the distribution of the T_j^* differences have some very extreme values. These outliers are not attributable to the non-convergence of the IWLS algorithm because when the estimation process does not converge, we discard the corresponding sample (or bootstrap sample) and generate a new one.

Unfortunately, given the wide range of the axis, it is not possible to draw conclusions about the performance of parametric bootstrap using figure 2.5.

Consequently, the Q-Q plots are then re-plotted in figure 2.6 after the elimination of these outliers. Now, it is evident that the distance between G and $\hat{G}$ distributions is small only for non informative variables (i.e. covariates with $\beta = 0$) and it grows for growing values of the model coefficients β. Simulations show that β_6 and β_7 have the most marked differences.

This result is confirmed by the low values of the empirical coverage probabilities estimated for these coefficients (see Table 2.3). Coefficients β_6 and β_7 have the lowest coverage probabilities.

The bootstrap confidence intervals were estimated using the `boot` R package [Canty and Ripley (2010)]:

```
require(boot)
boot.out <- list(t0 = t(beta.orig),
            t = as.matrix(beta.boot),
            R = B)
ci <- boot.ci(boot.out, type="perc",
              conf=alpha, index=k)$percent[4:5]
```

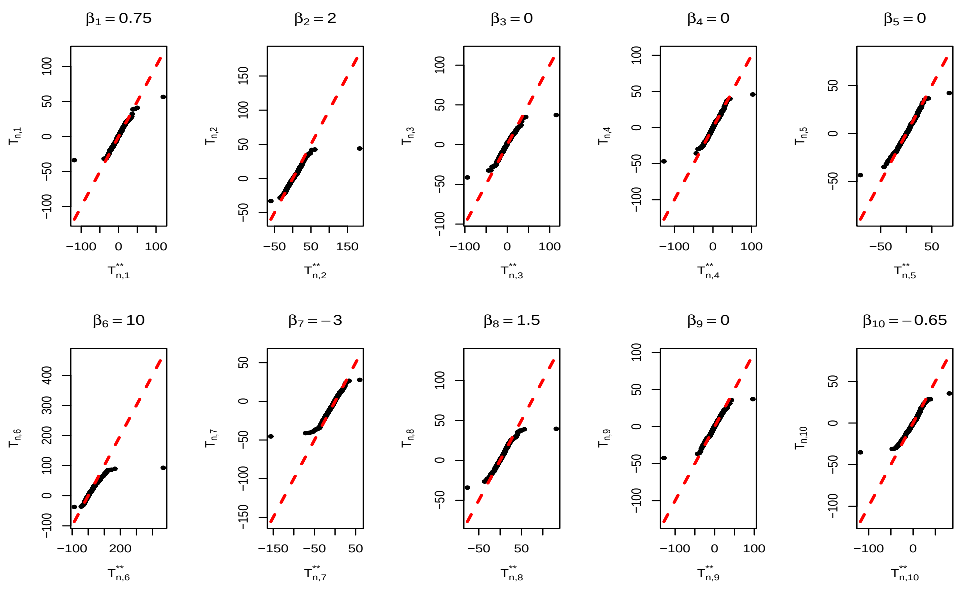

Figure 2.5: Parametric bootstrap for the logistic model (2.2). Comparison of G and $\hat{G}$ distributions using Q-Q plots.

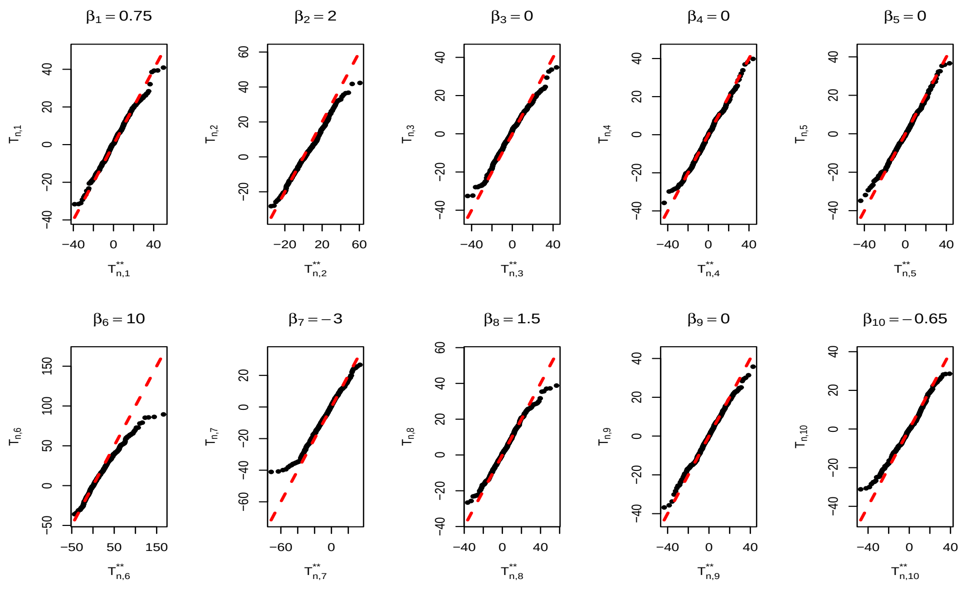

Figure 2.6: Parametric bootstrap for the logistic model (2.2). Comparison of G and $\hat{G}$ distributions using Q-Q plots, after deleting extreme values

The results of this section are interesting because they show that, even under non-critical conditions, parametric bootstrap for logistic regression can be problematic. The comparison of bootstrap estimates to original estimates shows that the distribution of bootstrap estimates have a (left or right, it depends on the sign of the coefficient) 'heavy' tail, that is an anomalous abundance of values at one of the extreme of the range. This problem yields poor bootstrap approximations of the distribution, G.

2.5 Vector resampling

Vector (or case) resampling is a type of nonparametric boostrap that imagines the data as a sample from some bivariate distribution, F, of (X, Y). For GLMs, it is based on the following algorithm:

1. Resample with replacement from the set of pairs $(x_1, y_1), \dots, (x_n, y_n)$, ($x_i$ is the ith row of X) and generate the bootstrapped pairs (x_1^*, y_1^*), …, (x_n^*, y_n^*). Call (X^*, Y^*) the corresponding bootstrap dataset.

2. Fit the generalized linear model to (X^*, Y^*) data and get the $\hat{\beta}^*$ estimate.

3. Repeat steps 1 and 2 B times.

Now we test vector resampling on the two DGPs used in the previous section: the linear model (2.1) and the logistic model (2.2). The R code used for resampling with replacement the pairs $(x_1, y_1), \dots, (x_n, y_n)$ (that is the rows of the X matrix and the elements of the y vector) is:

```
idx.boot <- sample(1:n, replace = T)
X.boot <- X[idx.boot,]
Y.boot <- Y[idx.boot]
dset.boot <- data.frame(y = Y.boot, X = X.boot)
```

The QQ-plots of figure 2.5 and the empirical coverage probabilities of table 2.3 evidence that, for data generated using the linear model, the nonparametric bootstrap works correctly, giving good approximations of the G distributions for all the (informative and non informative) covariates.

On the other side and similarly to the above section on parametric bootstrap, we find a different scenario when we apply nonparametric bootstrap to the logistic DGP. Again, the results show some very extreme values of the $\hat{\beta}^*$ bootstrap estimates and show that this bootstrap technique is able to approximate the distribution of $\hat{\beta}$ only partially and moderately (see figures 2.8 and 2.9 and table 2.3).

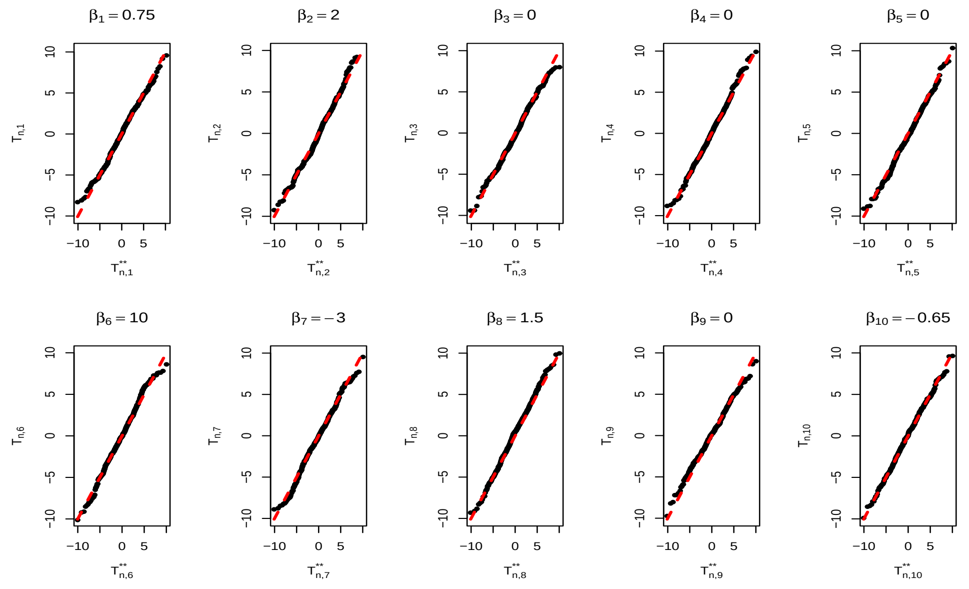

Figure 2.7: Nonparametric bootstrap for the linear model (2.1). Comparison of G and $\hat{G}$ distributions using Q-Q plots.

2.6 The residual bootstrap

The residual bootstrap for linear regression models is based on the following algorithm:

1. Calculate the OLS estimate $\hat{\beta} = (X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}Y$ of the model coefficients and the modified residuals:

$$r_i = \frac{y_i - \hat{y}_i}{\sqrt{1-h_i}} \quad i = 1, 2, \ldots, n, \tag{2.11}$$

   where $\hat{y}_i = X\hat{\beta}$ is the predicted value of the outcome y_i and h_i is the leverage of the ith observation.

2. Calculate the centered residuals $e_i = r_i - \overline{r}$, where $\overline{r}$ is the average of the r_i values.

3. Randomly resample with replacement e_i^* from e_i.

4. Set $y^* = X\hat{\beta} + e^*$.

5. Fit OLS regression to (X, y^*) data and get $\hat{\beta}^*$ estimates.

6. Calculate the confidence interval $[2\hat{\beta} - \hat{\beta}^*_{(1-\alpha/2)};\ 2\hat{\beta} - \hat{\beta}^*_{(\alpha/2)}]$, where $\hat{\beta}^*_{\alpha}$ is the αth percentile of the $\hat{\beta}^*$ boostrap distribution.

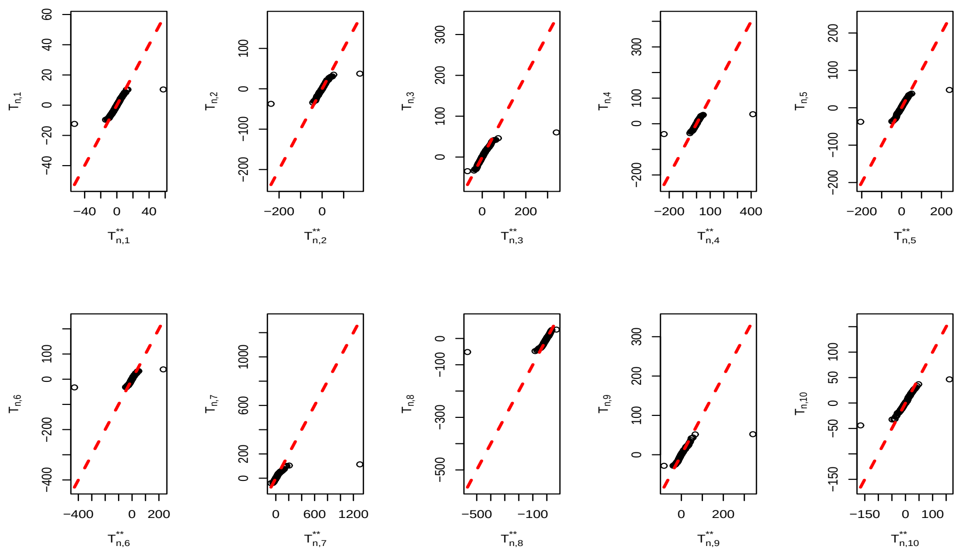

Figure 2.8: Nonparametric bootstrap for the logistic model (2.2). Comparison of G and $\hat{G}$ distributions using Q-Q plots.

7. Repeat steps 3 to 6 B times.

Influence points are those observations that cause large changes in the parameter estimates when they are deleted. The *leverage* value h_i is typically defined as the ith diagonal element of the hat matrix $H = X(X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}$. The h_i values are always inside the interval $[0, 1]$. An influence point will typically have a high leverage value h_i. The converse is not always true: a point with a high leverage is not necessarily an influential point.

[Moulton and Zeger (1991)] extend this type of residual bootstrap to the class of GLMs, using a generalization of the notion of residuals. The central idea is to consider Pearson residuals, obtained from row residuals $y_i - \hat{\mu}_i$, scaled by the estimated standard deviation, $\sqrt{v_i}$ of y_i,

$$r_i = \frac{y_i - \hat{\mu}_i}{\sqrt{v_i}}$$

or to consider the standardized Pearson residuals, that is the more nearly exchangeable quantities:

$$r_i = \frac{y_i - \hat{\mu}_i}{\sqrt{v_i(1 - h_i)}}, \tag{2.12}$$

where y_i is the observed outcome, $\hat{\mu}_i$ the prediction of the model, h_i is the leverage of the ith observation that is element of the hat matrix $H =$

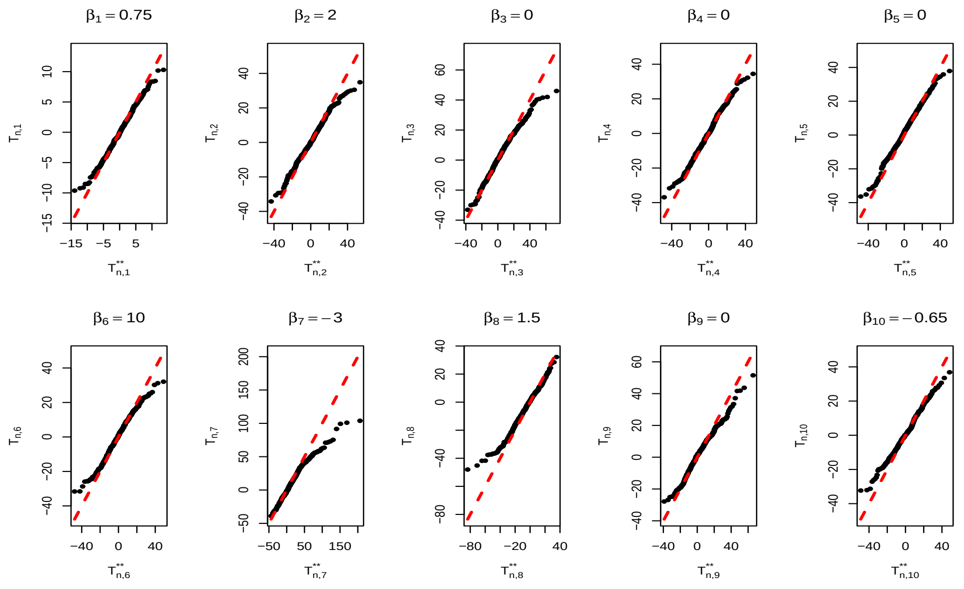

Figure 2.9: Nonparametric bootstrap for the logistic model (2.2). Comparison of G and $\hat{G}$ distributions using Q-Q plots, after deleting extreme values.

$W^{1/2}X(X^{\mathrm{T}}WX)^{-1}X^{\mathrm{T}}W^1/2 = G(G^{\mathrm{T}}G)^{-1}G$, where $G = W^{1/2}X$ and W is defined in (2.8).

For the logistic regression $y_i \in \{0,1\}$, $v_i = \hat{\mu}_i(1-\hat{\mu}_i)$, where $\hat{\mu}_i = \hat{p}_i$ is the success probability for the ith observation estimated by the logistic model

$$\hat{\mu} = \frac{1}{1+\exp(-\hat{\eta})}, \qquad \hat{\eta} = X\hat{\beta}.$$

Pearson residuals and standardized Pearson residuals are quite simple to estimate in R. Below we show an example of the R code for estimating these residuals. We start generating data from the logistic model (2.2). Then we estimate the logistic model using the `glm` command and calculate probabilities $\hat{\mu}_i$ (`probs.hat`) and variances v_i (`V.probs.hat`).

```
glm.fit <- glm(y ~ X, family = binomial(link = "logit"))
probs.hat <- predict(glm.fit, newx=X, type="response")
V.probs.hat <- glm.fit$family$variance(probs.hat)
pears.res <- (y - probs.hat)/sqrt(V.probs.hat)
```

The standardized Pearson residuals are calculated using the following code:

```
hi <- hatvalues(glm.fit)
std.pears.res <- pears.res/sqrt(1 - hi)
```

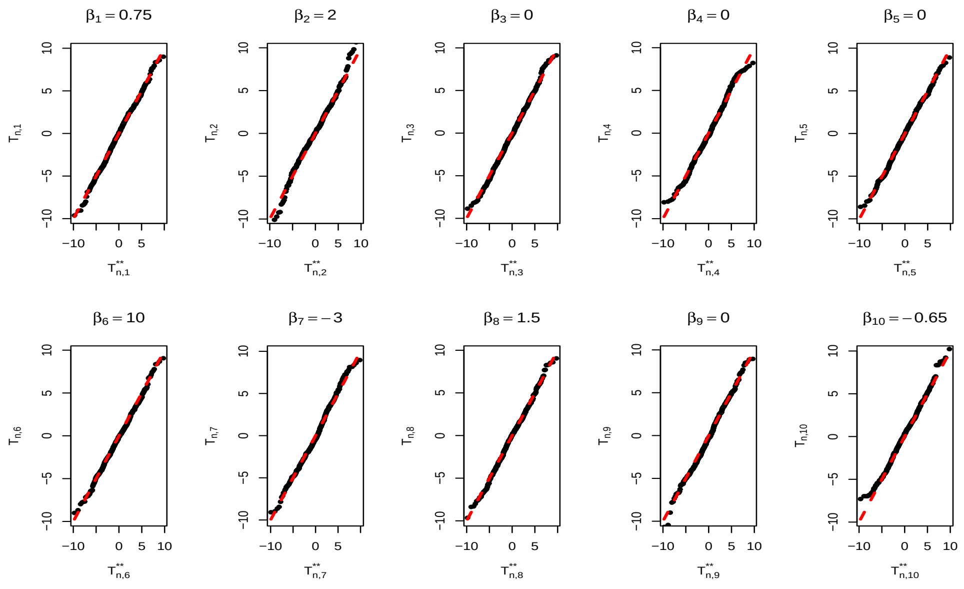

Figure 2.10: Residual bootstrap for the linear model (2.1). Comparison of G and $\hat{G}$ distributions using Q-Q plots, after deleting extreme values.

The distributions of Pearson and standardized Pearson residuals for the above logistic model are shown in figure (2.11).

The simplest form of residual bootstrap for GLM can be described by the following algorithm:

1. Fit GLM, estimate model coefficients, calculate standardized Pearson residuals r_i defined by equation (2.12) and mean-adjusted residuals $e_i = r_i - \bar{r}$, where $\bar{r} = 1/n \sum_i r_i$.

2. Resample with replacement the mean-adjusted residuals e and generate the bootstrapped residuals $e_1^*, e_2^*, \ldots, e_n^*$.

3. Calculate the bootstrapped response variable by

$$y_i^* = \hat{\mu}_i + \sqrt{v_i} e_i^*. \tag{2.13}$$

4. Fit the generalized linear model to the (X, y^*) data set and calculate the $\hat{\beta}^*$ estimate.

5. Calculate the confidence interval $[2\hat{\beta} - \hat{\beta}^*_{(1-\alpha/2)};\ 2\hat{\beta} - \hat{\beta}^*_{(\alpha/2)}]$, where $\hat{\beta}^*_\alpha$ is the αth percentile of the $\hat{\beta}^*$ boostrap distribution.

6. Repeat steps 2 to 5 B times .

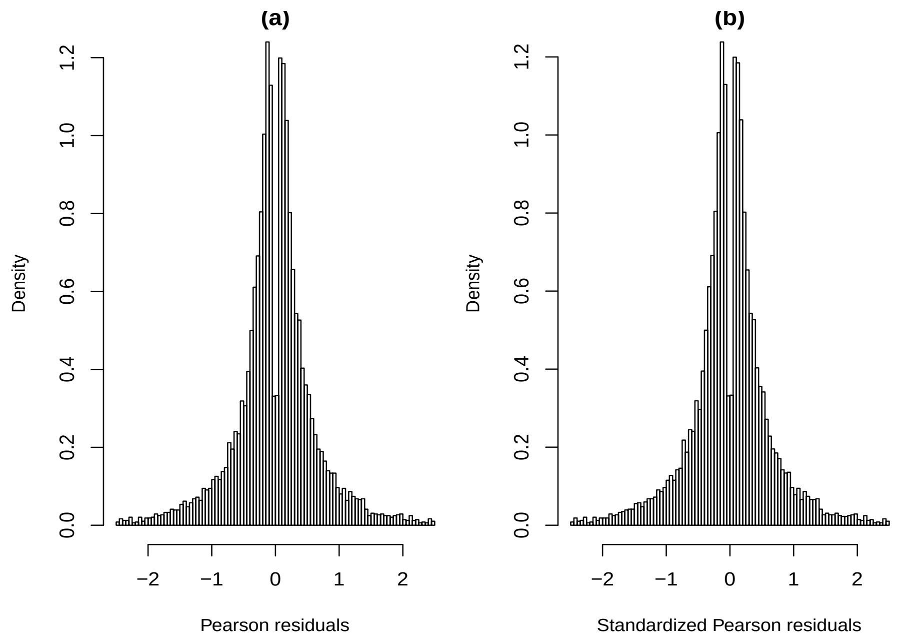

Figure 2.11: Pearson residuals and standardized Pearson residuals for a logistic model

The R code for resampling mean-adjusted residuals and for generating the bootstrapped response variable is:

```
ma.std.pears.res <- std.pears.res - mean(std.pears.res)
res.boot <- sample(ma.std.pears.res, replace = T)
probs.boot <- probs.hat + sqrt(V.probs.hat)*res.boot
```

Figure 2.12 (a) shows an example of the distribution of bootstrapped standardized Pearson residuals and (b) of the distribution of the bootstrapped response variable y_i^*.

One obvious drawback of the residual bootstrap method for GLM is that it does not yield discrete 0/1 y_i^* values, but it generates non-integer values, negative values and values above 1.

We consider two methods for the discretization of y_i^*.

- A first, simple fix is to round values of y_i^* to the nearest value 0 or 1. That is
$$\tilde{y}_i^* = \begin{cases} 0 & \text{if } y_i^* \le 0.5 \\ 1 & \text{if } y_i^* > 0.5 \end{cases}$$
where $\tilde{y}_i^*$ is the discretized version of y_i^*.

  This method can be implemented in R using

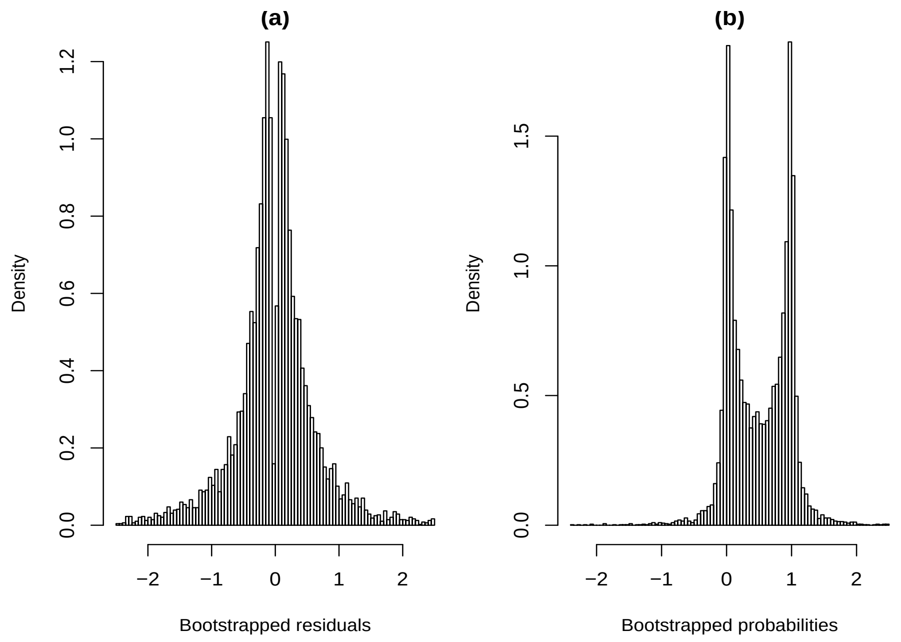

Figure 2.12: Bootstrapped residuals and probabilities of a logistic model

```
Y.boot <- array(0,n)
Y.boot[probs.boot>0.5] <- 1
```

- A second way for generating bootstrapped 0/1 values of the outcome y consists in modifying y_i^* using the rule

$$\tilde{y}_i^* = \begin{cases} 0 & \text{if } y_i^* < 0 \\ y_i^* & \text{if } 0 \leq y_i^* \leq 1 \\ 1 & \text{if } y_i^* > 1 \end{cases} \tag{2.14}$$

and then generating the 0/1 outcome using these $\tilde{y}_i^*$ values ($0 \leq \tilde{y}_i^* \leq 1$) as probabilities in a Bernoulli random number generator.

The R code for implementing this method is

```
probs.boot[probs.boot<0] <- 0
probs.boot[probs.boot>1] <- 1
Y.boot <- rbinom(n = n, prob = probs.boot, size = 1)
```

In figure 2.13 we show the distributions of (a) the probabilities predicted by the logistic model and (b) the bootstrapped values $\tilde{y}_i^*$ given by equation (2.13) and modified according to rule (2.14).

It is evident that these distributions are rather different. Bootstrapped $[0,1]$-resized probabilities shows a predominance of extreme (0 and 1) values (see [Davison and Hinkley (1997)], p. 334).

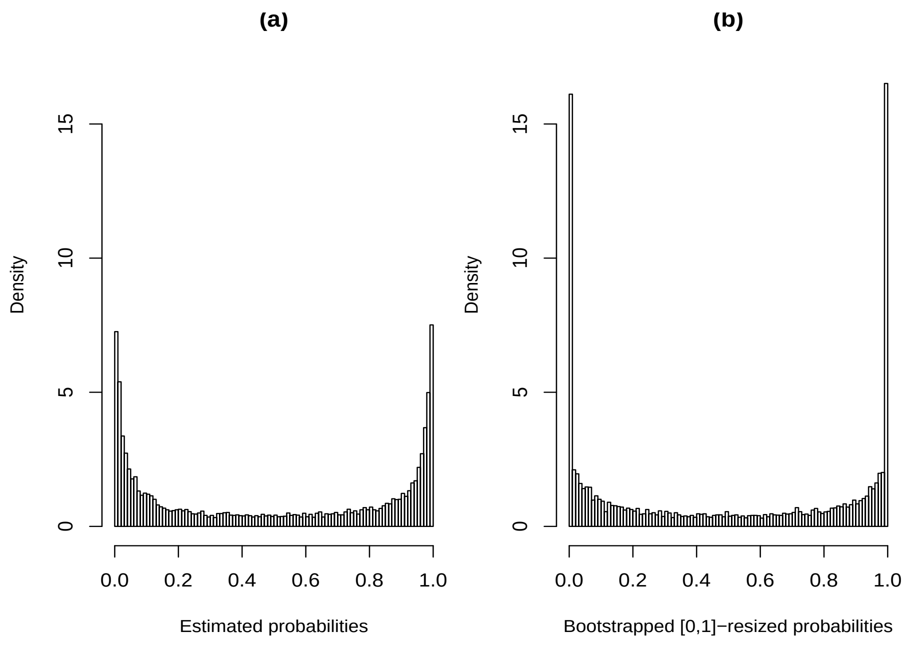

Figure 2.13: Histograms of (a) predicted probabilities $\hat{\mu}$ and (b) bootstrapped probabilities obtained by rule (2.14).

[Moulton and Zeger (1991)] recognized that the approach of residual bootstrap has some drawbacks. The more serious problem is the risk of obtaining 'extreme' data replications for which either the likelihood fail to converge (rare) or the parameter estimates fails to converge (much more likely). Discarding the divergent bootstrap replicates can introduce bias.

2.6.1 A simulation study for λ_{min} and λ_{1se}

In this subsection we investigate the statistical properties of λ_{min} and λ_{1se}, two cross-validation (CV) estimates of the optimal penalty parameter, by means of a simple cimulation study.

Figure (2.15) shows an example of estimation of λ_{min} and λ_{1se} for a sample generated by the linear DGP described in (2.1) and using the `cv.glmnet` command of the `glmnet` R package developed by [Friedman et al. (2010)]. The estimated values are $\lambda_{\text{min}} = 0.00224$ and $\lambda_{\text{1se}} = 0.015$. The number of non zero $\hat{\beta}$ coefficients estimated by lasso are 7 and 6, respectively.

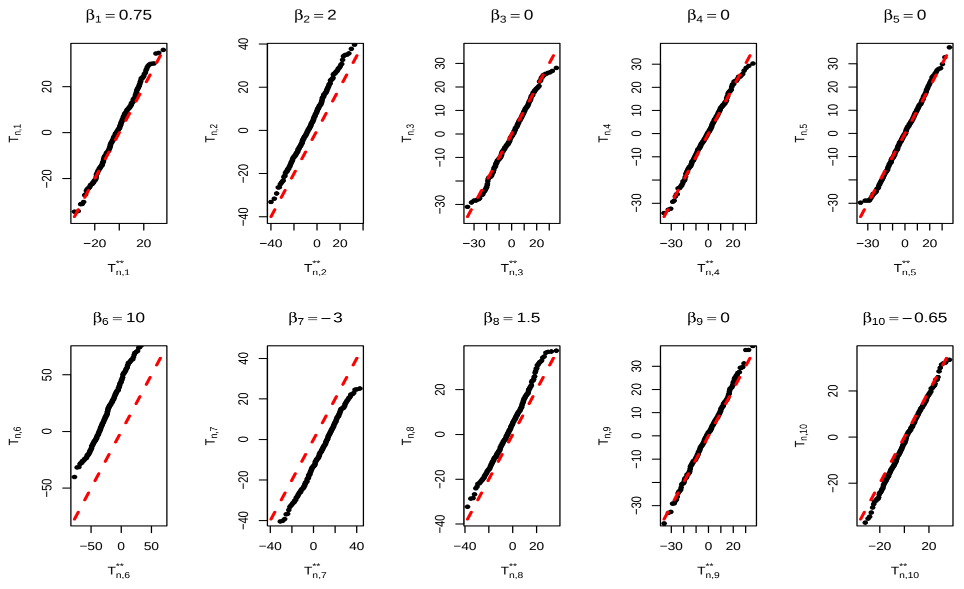

Figure 2.14: Residual bootstrap for the logistic model (2.2). Comparison of G and $\hat{G}$ distributions of the coefficients using Q-Q plots.

In general, $\lambda_{\text{min}} \leq \lambda_{\text{1se}}$ and the use of λ_{min} gives a greater number of selected covariates. This means that models estimated by lasso using λ_{min} are less regularized than models estimated with λ_{1se}.

In figure (2.16) we study the distribution of λ_{min} in samples generated by (**??**) (λ_{CV}) and in data resampled by residual bootstrap (λ^*_{CV}). In figure (2.17) we performed the same study for λ_{1se}.

It is worth to notice some interesting fact:

- ◇ distributions are bimodal and one of the modes is close to 0 (absence of regularization);
- ◇ the main differences between the two distributions of λ_{min} are located in the left tails; bootstrapped data show more frequently larger values of λ_{min} than original data.

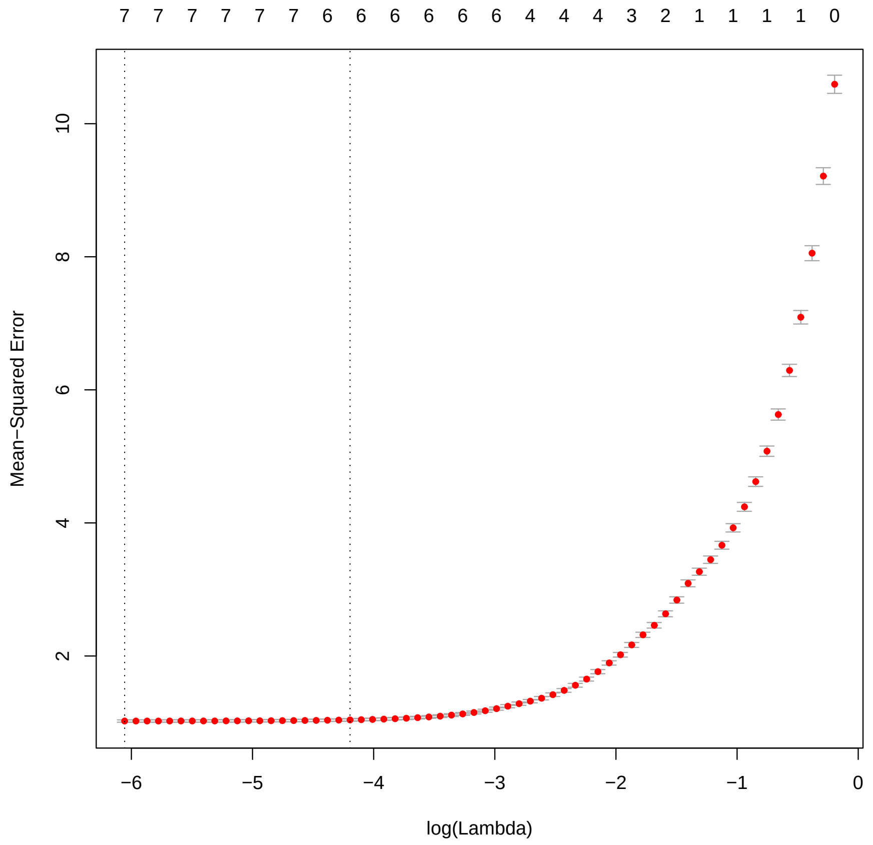

Figure 2.15: Plot of the mean cross-validation squared prediction error as a function of the penalty parameter λ for the linear DGP (2.1). The dotted line on the left side corresponds to λ_{min}. The second line is λ_{1se}.

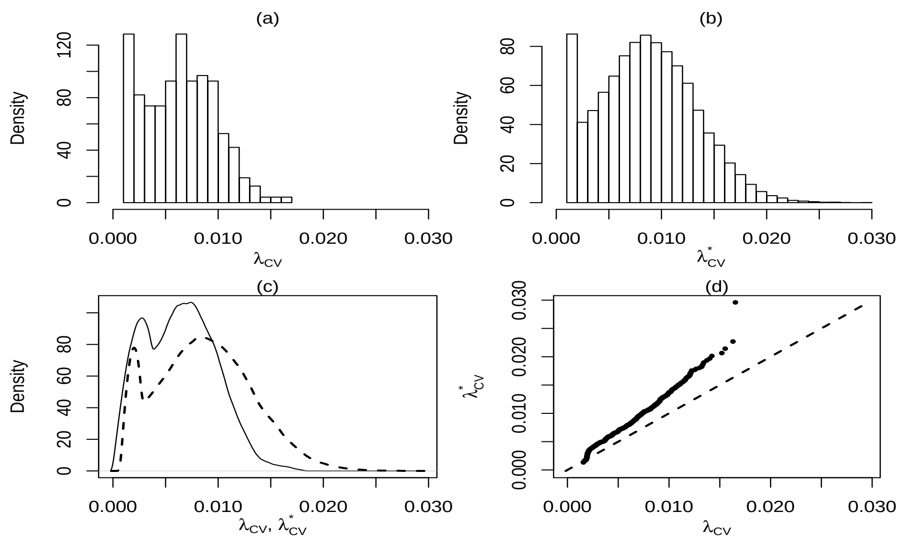

Figure 2.16: Distribution of λ_{min} in (a) original and (b) residual bootstrapped data. In (c) the kernel density estimations of the two distributions are reported (dashed line is for bootstrapped data) and in (d) the two distributions are compared by a Q-Q plot.

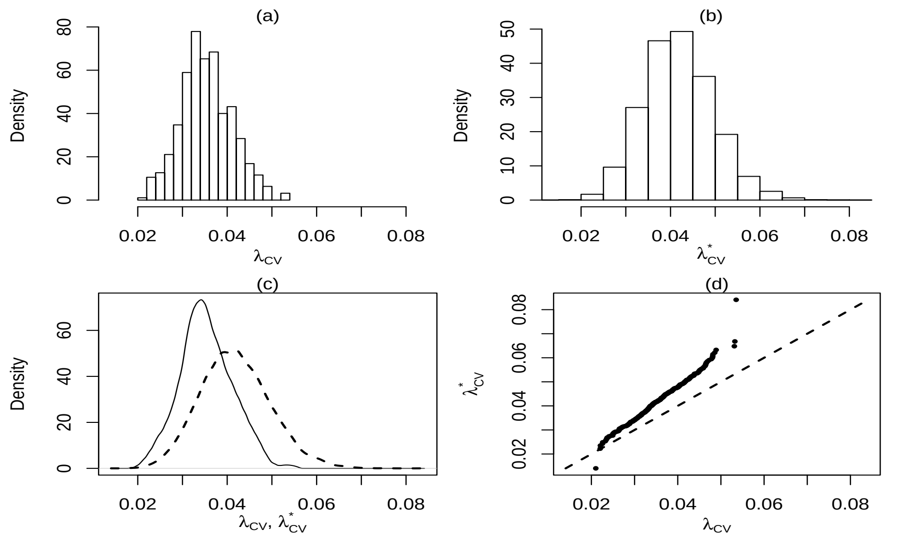

Figure 2.17: Distribution of λ_{1se} in (a) original and (b) residual bootstrapped data. In (c) the kernel density estimations of the two distributions are reported (dashed line is for bootstrapped data) and in (d) the two distributions are compared by a Q-Q plot.

2.7 One-step bootstrap for GLM

Another bootstrapping method that is worth considering here is the one-step bootstrap for GLMs proposed in [Moulton and Zeger (1991)] and extended in [Claeskensa *et al.* (2003)]. This method is appealing because it is easy to implement, it is fast and does not generate extreme values.

[Moulton and Zeger (1991)] proposed two bootstrap procedures based on a one-step approximation of the distribution of the $\hat{\beta}$ coefficients. The first procedure bootstraps a GLM in a manner analogous to residuals resampling and the second one is in some sense similar to vector resampling.

The idea behind this method is to estimate the $\hat{\beta}^*$ bootstrap coefficients using the first step of the IWLS estimation algorithm. The algorithm is not allowed to run until it reaches convergence, but is stopped after the first iteration. This is why it is called "one-step" bootstrap.

One-step residual resampling

1. Fit the GLM, estimate model coefficients $\hat{\beta}$ and outcome predictions $\hat{\mu}$, calculate standardized Pearson residuals r_i defined by equation (2.12) and mean-adjusted residuals $e_i = r_i - \bar{r}$, where $\bar{r} = 1/n \sum_i r_i$.

2. Resample with replacement the mean-adjusted residuals e and generate the bootstrapped residuals $e^* = (e_1^*, e_2^*, \dots, e_n^*)$.

3. Calculate the one-step bootstrapped β^* coefficients by:

$$\begin{aligned} \beta^* &= (X^{\mathrm{T}}WX)^{-1}X^{\mathrm{T}}Wz^* \\ &= (X^{\mathrm{T}}WX)^{-1}X^{\mathrm{T}}W(X\hat{\beta} + \Gamma V^{1/2}e^*) \\ &= \hat{\beta} + (G^{\mathrm{T}}G)^{-1}G^{\mathrm{T}}e^*, \end{aligned} \tag{2.15}$$

   where $V = \mathrm{diag}(v_1, \dots, v_n)$, $\Gamma = \mathrm{diag}(\dot{g}(\hat{\mu}_1), \dots, \dot{g}(\hat{\mu}_n))$, $\Gamma V^{1/2} = W^{-1/2}$, $G = W^{1/2}X$ and W is defined in (2.8).

4. Repeat steps 2 and 3 B times .

For a linear model $W = I_n$, where I_n is the identity matrix of order n, $G = X$ and

$$\begin{aligned} \beta^* &= \hat{\beta} + (X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}e^* = (X^{\mathrm{T}}X)^{-1}(X^{\mathrm{T}}X)\hat{\beta} + (X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}e^* \\ &= (X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}(X\hat{\beta} + e^*) = (X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}y^* \end{aligned}$$

where $y^* = X\hat{\beta}+e^*$. This means that for the linear regression model the one-step residual resampling corresponds to the residual resampling of section 2.6.

For a logistic model $W = V$ and $G = V^{1/2}X$.

The results of the simulation study about the linear model (2.1) and the logistic model (2.2) are shown in figure 2.18, figure 2.19 and table 2.3.

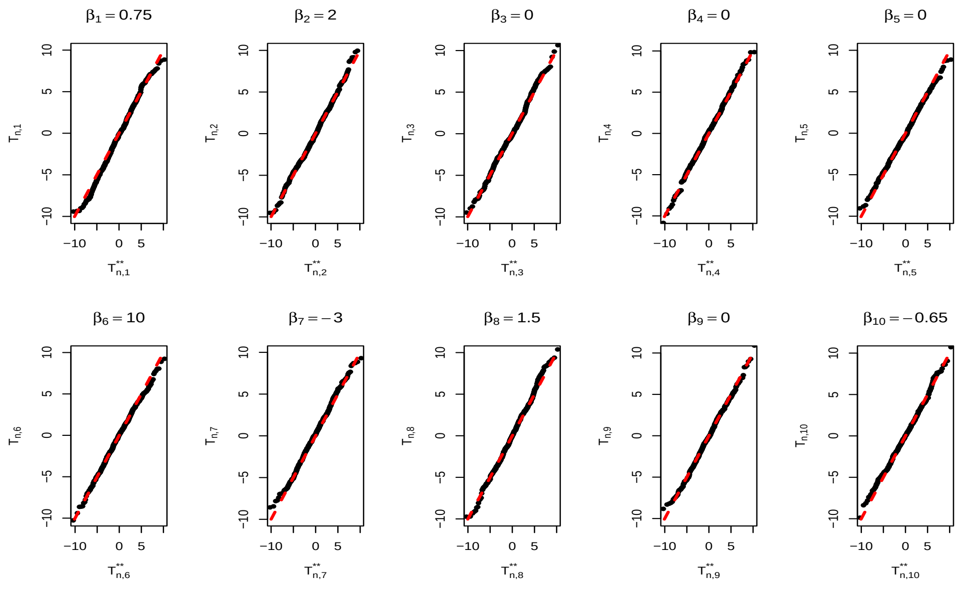

Figure 2.18: One-step bootstrap (based on residual resampling) for the linear model (2.1). Comparison of G and $\hat{G}$ distributions of the model coefficients using Q-Q plots.

One-step vector resampling

1. Fit the GLM, estimate model coefficients $\hat{\beta}$, calculate standardized Pearson residuals r_i defined by equation (2.12) and mean-adjusted residuals $e_i = r_i - \bar{r}$, where $\bar{r} = 1/n \sum_i r_i$.

2. Generate a resampling (diagonal) matrix $D = \text{diag}(d)$, where $d = (d_1, d_2, \ldots, d_n)$ is distributed as a multinomial random vector of parameters n (number of independent trials) and $\mathbf{p} = (1/n, \ldots, 1/n)$ (vector of probabilities of each outcome).

3. Calculate the one-step bootstrapped β^* coefficients by:
$$\beta^* = \hat{\beta} + (G^{\mathrm{T}} D G)^{-1} G^{\mathrm{T}} D e, \tag{2.16}$$
where $G = W^{1/2} X$ as defined above.

4. Repeat steps 2 and 3 B times .

For a linear model $G = X$ and

$$\begin{aligned}\beta^* &= \hat{\beta} + (X^{\mathrm{T}} D X)^{-1} X^{\mathrm{T}} D e^* = (X^{\mathrm{T}} D X)^{-1} (X^{\mathrm{T}} D X) \hat{\beta} + (X^{\mathrm{T}} D X)^{-1} X^{\mathrm{T}} D e^* = \\ &= (X^{\mathrm{T}} D X)^{-1} X^{\mathrm{T}} D (X \hat{\beta} + e^*) = (X^{\mathrm{T}} D X)^{-1} X^{\mathrm{T}} D y^*\end{aligned}$$

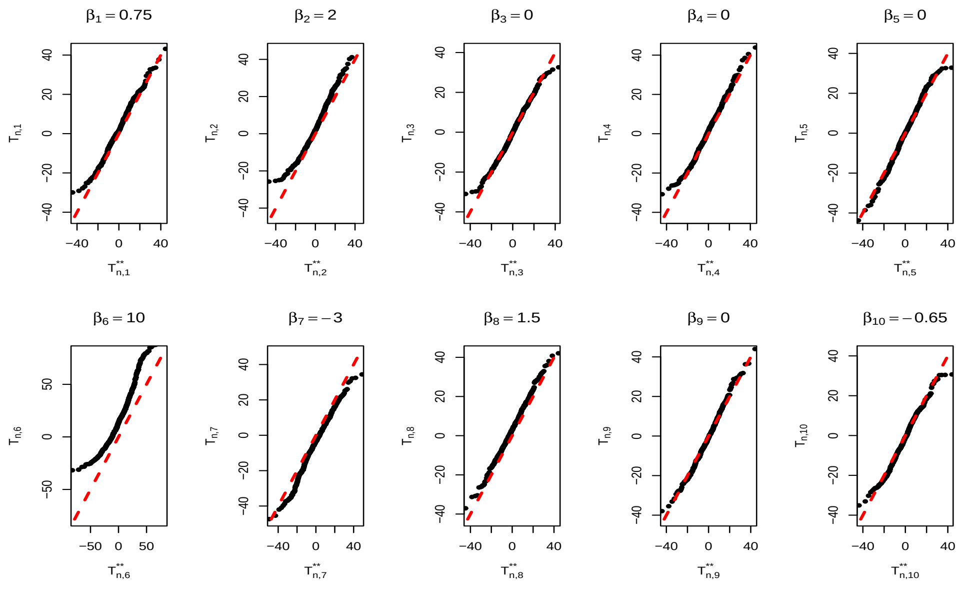

Figure 2.19: One-step bootstrap (based on residual resampling) for the logistic model (2.2). Comparison of G and $\hat{G}$ distributions of the model coefficients using Q-Q plots.

where $y^* = X\hat{\beta} + e^*$ and (DX, Dy) is the dataset obtained sampling pairs with replacement from (x_1, y_1), …, (x_n, y_n). This means that for the linear regression model, the one-step vector resampling corresponds to the vector resampling procedure of section 2.5.

The results of the simulation study for the linear model (2.1) and the logistic model (2.2) are shown in figure 2.20, figure 2.21 and table 2.3.

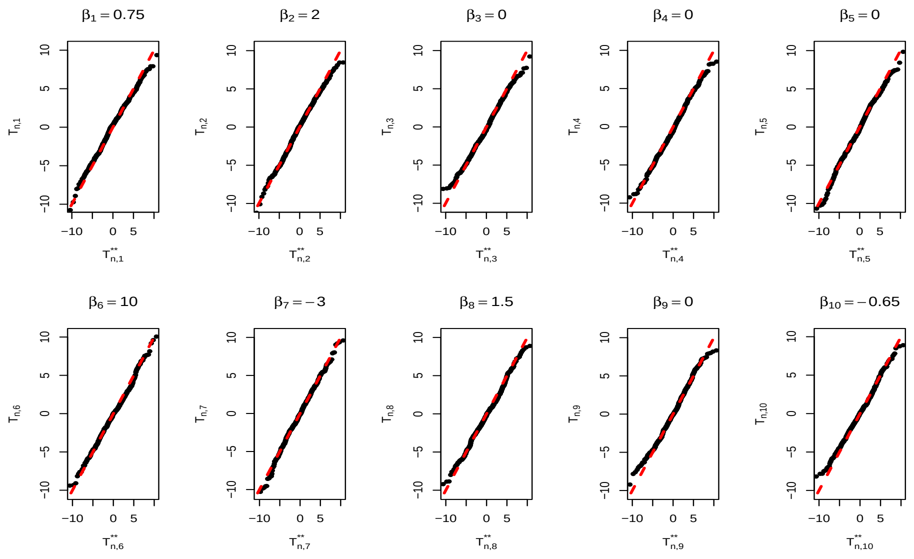

Figure 2.20: One-step bootstrap (based on vector resampling) for the linear model (2.1). Comparison of G and $\hat{G}$ distributions of the coefficients using Q-Q plots.

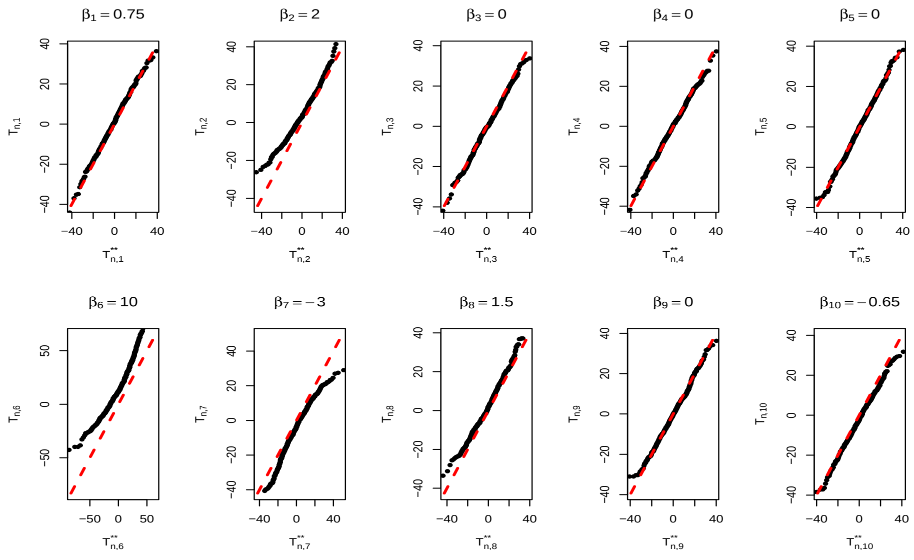

Figure 2.21: One-step bootstrap (based on vector resampling) for the logistic model (2.2). Comparison of G and $\hat{G}$ distributions of the coefficients using Q-Q plots.

2.8 Discussion

In this chapter we investigated and compared five bootstrap methods for GLMs. Our main objective was the construction of reliable confidence intervals for model coefficients.

In the case of a linear model, all five methods yield intervals with good empirical coverage and their performance is comparable. More problematic is the construction of confidence intervals for a GLM. In this case our simulative studies suggest that the one-step vector resampling algorithm of [Moulton and Zeger (1991)] is the better method, followed by the one-step residual bootstrap.

These two methods show a high capability of reproducing the uncertainty about coefficients of regression models and are also very fast because they do not use the complete IWLS estimation algorithm (like parametric bootstrap, case resampling and residual resampling) but only the first step.

In the next chapter we investigate the possibility to generalize the two one-step algorithms to the class of ℓ_1-penalized GLMs.

Table 2.1: Bootstrap estimates of bias for the coefficients of the linear model (2.1) and the logistic model (2.2).

	β_0	β_1	β_2	β_3	β_4	β_5	β_6	β_7	β_8	β_9	β_{10}
Coefficients	0	0.75	2	0	0	0	10	-3	1.5	0	-0.65
Linear model											
True value of bias	0	0	0	0	0	0	0	0	0	0	0
Parametric	0.0002	-0.0001	-0.0001	0.0004	-0.0005	0.0003	0.0005	0.0003	0.0001	-0.0003	-0.0011
Nonparametric	0.0001	-0.0003	-0.0001	-0.0004	0.0000	0.0000	0.0007	-0.0011	0.0004	0.0000	0.0003
Residual	-0.0002	-0.0009	-0.0003	-0.0011	-0.0001	0.0006	0.0005	0.0005	0.0000	-0.0008	-0.0005
One-step (residual resamp.)	-0.0002	-0.0004	0.0003	-0.0003	-0.0002	-0.0002	-0.0008	-0.0005	0.0003	-0.0002	-0.0002
One-step (vector resamp.)	0.0003	-0.0001	0.0000	-0.0004	0.0001	-0.0010	0.0008	-0.0011	0.0012	0.0003	-0.0005
Logistic model											
True value of bias	0	0	0	0	0	0	0	0	0	0	0
Parametric bootstrap	-0.0010	0.1057	0.2482	0.0005	-0.0012	-0.0044	1.2924	-0.3841	0.2023	-0.0030	-0.0814
Nonparametric bootstrap	-0.0021	0.1035	0.2781	0.0030	0.0111	0.0012	1.3731	-0.4083	0.2114	0.0046	-0.0852
Residual bootstrap	0.0075	-0.1110	-0.3260	0.0100	-0.0057	-0.0020	-1.5126	0.4906	-0.2309	0.0076	0.0991
One-step (residual resamp.)	-0.0002	0.0026	0.0001	-0.0015	-0.0003	0.0018	0.0005	0.0020	-0.0022	-0.0003	0.0002
One-step (vector resamp.)	0.0024	-0.0015	0.0014	-0.0006	-0.0016	0.0006	0.0029	0.0004	-0.0032	-0.0004	-0.0018

Table 2.2: Bootstrap estimates of the standard deviation for the coefficients of the linear model (2.1) and the logistic model (2.2). (Percentage error in round brackets)

	β_0	β_1	β_2	β_3	β_4	β_5	β_6	β_7	β_8	β_9	β_{10}
Coefficients	0	0.75	2	0	0	0	10	-3	1.5	0	-0.65
Linear model											
True value of std	0.0632	0.2191	0.2191	0.2191	0.2191	0.2191	0.2191	0.2191	0.2191	0.2191	0.2191
Parametric	0.0631	0.2194	0.2193	0.2199	0.2193	0.2190	0.2191	0.2188	0.2188	0.2190	0.2204
	(0.20)	(-0.16)	(-0.09)	(-0.35)	(-0.10)	(0.04)	(-0.00)	(0.14)	(0.12)	(0.06)	(-0.58)
Nonparametric	0.0643	0.2236	0.2226	0.2232	0.2229	0.2228	0.2230	0.2229	0.2229	0.2241	0.2235
	(-1.73)	(-2.07)	(-1.59)	(-1.89)	(-1.75)	(-1.70)	(-1.76)	(-1.72)	(-1.72)	(-2.31)	(-2.03)
Residual	0.0630	0.2180	0.2176	0.2191	0.2179	0.2177	0.2176	0.2186	0.2184	0.2178	0.2186
	(0.46)	(0.50)	(0.68)	(0.00)	(0.53)	(0.64)	(0.70)	(0.20)	(0.30)	(0.60)	(0.24)
One-step (residual resamp.)	0.0645	0.2240	0.2235	0.2245	0.2241	0.2238	0.2240	0.2244	0.2242	0.2244	0.2244
	(-1.98)	(-2.26)	(-2.02)	(-2.46)	(-2.30)	(-2.15)	(-2.26)	(-2.41)	(-2.33)	(-2.44)	(-2.45)
One-step (vector resamp.)	0.0662	0.2288	0.2297	0.2290	0.2290	0.2292	0.2296	0.2291	0.2293	0.2284	0.2289
	(-4.62)	(-4.41)	(-4.86)	(-4.51)	(-4.51)	(-4.60)	(-4.78)	(-4.56)	(-4.64)	(-4.24)	(-4.46)
Logistic model											
True value of std	0.2222	0.7781	0.8165	0.7716	0.7716	0.7717	1.4376	0.8699	0.7969	0.7716	0.7764
Parametric bootstrap	0.2517	0.8850	0.9411	0.8743	0.8775	0.8787	1.9191	1.0291	0.9242	0.8719	0.8789
	(-13.24)	(-13.74)	(-15.26)	(-13.32)	(-13.73)	(-13.86)	(-33.49)	(-18.31)	(-15.97)	(-12.99)	(-13.20)
Nonparametric bootstrap	0.2671	0.9366	1.0145	0.9259	0.9339	0.9294	2.0945	1.1029	0.9779	0.9286	0.9270
	(-20.17)	(-20.37)	(-24.25)	(-20.01)	(-21.03)	(-20.43)	(-45.69)	(-26.79)	(-22.71)	(-20.35)	(-19.40)
Residual bootstrap	0.2180	0.7622	0.7954	0.7518	0.7548	0.7498	1.3799	0.8439	0.7761	0.7529	0.7620
	(1.88)	(2.05)	(2.59)	(2.56)	(2.18)	(2.84)	(4.02)	(2.98)	(2.61)	(2.42)	(1.85)
One-step (residual resamp.)	0.2227	0.7801	0.8147	0.7741	0.7745	0.7730	1.4378	0.8678	0.7960	0.7741	0.7738
	(-0.19)	(-0.26)	(0.22)	(-0.33)	(-.038)	(-0.16)	(-0.01)	(0.24)	(0.12)	(-0.32)	(0.33)
One-step (vector resamp.)	0.2423	0.8442	0.8837	0.8411	0.8450	0.8399	1.4912	0.9403	0.8638	0.8416	0.8475
	(-9.02)	(-8.50)	(-8.23)	(-9.01)	(-9.51)	(-8.84)	(-3.73)	(-8.10)	(-8.39)	(-9.06)	(-9.16)

Table 2.3: Empirical coverage probabilities calculated using parametric, nonparametric, residual and one-step bootstrap for the linear model (2.1) and the logistic model (2.2). (Coverage errors of confidence intervals in round brackets)

	β_0	β_1	β_2	β_3	β_4	β_5	β_6	β_7	β_8	β_9	β_{10}
Coefficients	0	0.75	2	0	0	0	10	-3	1.5	0	-0.65
Linear model											
Desired nominal coverage	90	90	90	90	90	90	90	90	90	90	90
Parametric	88.3	87.8	86.5	88.0	92.5	87.3	87.8	90.5	89.3	88.0	88.5
	(-1.7)	(-2.2)	(-3.5)	(-2.0)	(2.5)	(-2.7)	(-2.2)	(0.5)	(-0.7)	(-2.0)	(-1.5)
Nonparametric	89.5	91.8	89.0	88.8	88.3	88.8	86.5	87.8	88.0	93.3	89.8
	(-0.5)	(1.8)	(1.0)	(-1.2)	(-1.7)	(-1.2)	(-3.5)	(-2.2)	(-2.0)	(3.3)	(-0.2)
Residual	91.8	90.3	89.5	87.8	88.0	91.3	91.3	89.3	88.5	91.3	91.0
	(1.8)	(0.3)	(-0.5)	(-2.2)	(-2.0)	(1.3)	(1.3)	(-0.7)	(-1.5)	(1.3)	(1.0)
One-step (residual resamp.)	89.4	85.8	90.4	88.4	89.0	91.0	91.2	88.6	88.6	90.6	92.0
	(-0.6)	(-4.2)	(0.4)	(-1.6)	(-1.0)	(1.0)	(1.2)	(-1.4)	(-1.4)	(0.6)	(2.0)
One-step (vector resamp.)	90.8	92.8	91.2	92.2	91.2	90.0	90.6	90.6	91.4	90.6	91.2
	(0.8)	(2.8)	(1.2)	(2.2)	(1.2)	(0.0)	(0.6)	(0.6)	(1.4)	(0.6)	(1.2)
Logistic model											
Desired nominal coverage	90	90	90	90	90	90	90	90	90	90	90
Parametric bootstrap	85.0	86.3	88.5	90.5	87.5	88.3	65.8	83.8	85.0	90.0	88.8
	(-5.0)	(-3.7)	(-1.5)	(0.5)	(-2.5)	(-1.7)	(-24.2)	(-6.2)	(-5.0)	(0.0)	(-1.2)
Nonparametric bootstrap	89.8	88.8	85.8	88.3	88.3	92.5	64.5	82.5	86.8	92.2	88.5
	(-0.2)	(-1.2)	(-4.2)	(-1.7)	(-1.7)	(2.5)	(-25.5)	(-7.5)	(-3.2)	(2.2)	(-1.5)
Residual bootstrap	89.0	89.8	90.3	93.5	93.8	92.8	76.3	88.8	91.5	92.0	91.8
	(-1.0)	(-0.2)	(0.3)	(3.5)	(3.8)	(2.8)	(-13.7)	(-1.2)	(1.5)	(2.0)	(1.8)
One-step (residual resamp.)	86.2	89.2	89.0	90.4	88.4	86.0	83.0	87.4	90.4	86.4	88.4
	(-3.8)	(-0.8)	(-1.0)	(0.4)	(-1.6)	(-4.0)	(-7.0)	(-2.6)	(0.4)	(-3.6)	(-1.6)
One-step (vector resamp.)	91.4	91.2	93.4	91.0	93.2	90.6	89.2	89.2	90.6	90.0	92.4
	(1.4)	(1.2)	(3.4)	(1.0)	(3.2)	(0.6)	(-0.8)	(-0.8)	(0.6)	(0.0)	(2.4)

Chapter 3

Bootstrap for penalized GLMs

3.1 Introduction

An important open problem in the field of ℓ_1-penalized regression is the construction of confidence intervals for the model coefficients. A popular approach to calculating confidence intervals is to use bootstrap simulation algorithms.

Compared to GLMs, the construction of confidence intervals for the coefficients of a penalized GLM must face an additional obstacle. In fact, the introduction of a penalization in the objective function generates a shrinkage effect, which is a bias on the estimated coefficients.

This means that, for example, the 'naive percentile' interval with $(1-\alpha)$ coverage

$$[\hat{\beta}^*_{\text{LO}}(\alpha);\ \hat{\beta}^*_{\text{UP}}(\alpha)] = [\hat{\beta}^*_{(\alpha/2)};\ \hat{\beta}^*_{(1-\alpha/2)}]$$

or the bootstrap confidence interval

$$[2\hat{\beta} - \hat{\beta}^*_{(1-\alpha/2)};\ 2\hat{\beta} - \hat{\beta}^*_{(\alpha/2)}]$$

may be unreliable if the estimator $\hat{\beta}$ is biased. The first interval is based on the hypothesis that $\hat{\beta}^* \approx \beta$, which is not true for a biased estimator. The second interval is based on the idea that the distribution of $(\hat{\beta} - \beta)$ can be approximated through the bootstrap distribution of $(\hat{\beta}^* - \hat{\beta})$. Again, in general, for a biased estimator this is not realistic because $E(\hat{\beta} - \beta) \neq E(\hat{\beta}^* - \hat{\beta})$. In the next sections we will see that this condition is valid for the residual bootstrap proposed by [Chatterjee and Lahiri (2010)], but only when applied to the linear model and not for the whole class of GLMs.

In sections 3.2 and 3.3 we start considering two bootstrap methods for penalized linear models based on the resampling of model residuals. We first review their functioning and their statistical properties and then we evaluate their performance by a simulation study.

Section 3.4 describes case and vector resampling for penalized GLMs and, using a simulation study with a logistic DGP, shows that these methods do not allow building at good confidence intervals.

In the subsequent sections we develop some ideas for generalizing the method of [Chatterjee and Lahiri (2010)] to ℓ_1-penalized GLMs. The approximation of the lasso solutions by ridge regression is the core of section 3.5. This approximation proves to be useful for the development of a one-step bootstrap method for penalized GLMs. This method is described and tested in section 3.6.

In section 3.7 we consider double bootstrap, a resampling technique that reduces the error of single bootstrap and builds confidence intervals with a higher order of accuracy. We apply double bootstrap to our one-step residual resampling algorithm, showing that it can offer sensibly reduced coverage error (see sectionone.step.boot.glm.double).

3.2 The residual bootstrap for penalized LMs

Consider the linear regression model

$$y_i = x_i^{\mathrm{T}}\beta + \varepsilon_i, \quad i = 1, \dots, n, \tag{3.1}$$

where y_i is the response, $x_i = (x_{i1}, \dots, x_{ip})^{\mathrm{T}}$ is a p-dimensional covariate vector, $\beta = (\beta_1, \dots, \beta_p)^{\mathrm{T}}$ is the regression parameter and $\{\varepsilon_i : i = 1, \dots, n\}$ are independent and identically distributed errors.

For the penalized linear regression, we know that the lasso estimator of β is defined as the minimizer of the ℓ_1-norm penalized least square criterion function,

$$\hat{\beta}^{\text{lasso}} = \arg\min_{u \in \mathbb{R}^p} \sum_{i=1}^{n} (y_i - x_i^{\mathrm{T}} u)^2 + \lambda \sum_{j=1}^{p} |u_j|, \tag{3.2}$$

where λ is the regularization parameter.

In this section we start considering a simple resampling algorithm for the ℓ_1-penalized linear model. In the preceding chapter we have shown that there are two major approaches to bootstrap for regression models, depending on whether the x_i's are assumed to be random or not. In the case where x_i is random, it is of interest to study the joint distribution of the covariates and the response, hence vector resampling (i.e. pairwise bootstrap) is a relevant choice. In contrast, one can assume that the x_i's are non-random. In this situation, the standard approach to bootstrapping is the residual bootstrap [Efron (1979)].

A straightforward implementation of the residual bootstrap algorithm for the ℓ_1-penalized linear model can be summarized as the following steps.

1. Calculate, by cross-validation, the optimal penalization parameter $\hat{\lambda}$ and estimate $\hat{\beta}$ using the lasso estimator of β defined in (3.2)

2. Let e_i be the associated residuals:
$$r_i = y_i - x_i^{\mathrm{T}} \hat{\beta}, \quad i = 1, \dots, n.$$

3. Consider the set of centered residuals $\{e_i = r_i - \bar{r} : i = 1, \dots, n\}$, where $\bar{r} = n^{-1} \sum_{k=1}^{n} r_i$.

4. Select with replacement a sample of size n, $\{e_i^* : i = 1, \dots, n\}$ from the set of centered residuals e_i.

5. Calculate the bootstrapped version of the outcome variable (3.1) as
$$y_i^* = x_i^{\mathrm{T}} \hat{\beta} + e_i^*, \quad i = 1, \dots, n.$$

6. Using the bootstrap dataset $\{(y_i^*, x_i) : i = 1, \dots, n\}$ and the penalization parameter $\hat{\lambda}$, calculate the bootstrap version of the lasso estimator defined as:
$$\hat{\beta}^* = \arg\min_{u \in \mathbb{R}^p} \sum_{i=1}^{n} (y_i^* - x_i^{\mathrm{T}} u)^2 + \hat{\lambda} \sum_{j=1}^{p} |u_j|, \tag{3.3}$$

7. Calculate the confidence interval $[2\hat{\beta} - \hat{\beta}^*_{(1-\alpha/2)};\ 2\hat{\beta} - \hat{\beta}^*_{(\alpha/2)}]$, with coverage $(1 - \alpha)$, where $\hat{\beta}^*_j$ is the jth percentile of the $\hat{\beta}^*$ bootstrap distribution.

8. Repeat steps 2 to 7 B times.

The bootstrap version of $T \equiv n^{1/2}(\hat{\beta} - \beta)$ is $T^* = n^{1/2}(\hat{\beta}^* - \hat{\beta})$. The residual bootstrap estimator of the unknown distribution G of T is the (conditional) distribution $\widehat{G}(\cdot)$ of T^* given the observations $\{(y_i, x_i) : i = 1, \dots, n\}$:
$$\widehat{G}(B) = pr_*(T^* \in B), \quad B \in \mathcal{B}(\mathbb{R}^p), \tag{3.4}$$
where pr_* denotes the conditional probability given the errors $\{\varepsilon_i : i = 1, \dots, n\}$ and $\mathcal{B}(\mathbb{R}^p)$ denotes the Borel σ-field on $\mathbb{R}^p$.

For the bootstrap approximation to be useful, one would expect $\widehat{G}(\cdot)$ to be close to $G(\cdot)$. However, for the above algorithm, this is not the case. In a recent work, [Chatterjee and Lahiri (2010)] show that the residual bootstrap estimator $\widehat{G}(\cdot)$, instead of converging to the deterministic limit of the G distribution given by [Knight and Fu (2000)], converges weakly to a random probability measure and therefore, it fails to provide a valid approximation of $G(\cdot)$.

The inconsistency of the standard residual bootstrap arises when some components of β are zero. The residual bootstrap approximation has a random limit and is inconsistent. The lasso estimators of the non-zero components of β estimate their signs correctly with high probability, but the estimators of the zero-components take both positive and negative values with positive probabilities. The residual bootstrap mimics the main features of the regression model closely but it fails to reproduce the sign of the zero-components of β with sufficient accuracy in the formulation of the bootstrap lasso estimation criterion, leading to the random limit [Chatterjee and Lahiri (2010)].

The results of our simulation study with a linear DGP show that the residual bootstrap works acceptably well.

The comparisons between the distributions of $\hat{\beta}_j^* - \hat{\beta}_j$ and $\hat{\beta}_j - \beta_j$, $j = 1, \ldots, p$ using Q-Q plots, are shown in figure 3.2. Figure 3.1 shows the empirical probability distributions of $\hat{\beta}_6^*$ and $\hat{\beta}_6$.

The $\hat{\beta}_j^* - \hat{\beta}_j$, $j = 1, \ldots, p$ differences are in mean rather close to the bias $\hat{\beta}_j - \beta_j$:

	β_0	β_1	β_2	β_3	β_4	β_5	β_6	β_7	β_8	β_9	β_{10}
$(\hat{\beta} - \beta)$	0	-0.0744	-0.0691	0	0	0	-0.0679	0.0749	-0.0655	0	0.0765
$(\hat{\beta}^* - \hat{\beta})$	0.0010	-0.0674	-0.0707	0.0035	0.0009	0.0104	-0.0792	0.0755	-0.0759	-0.0029	0.0537

The bootstrap approximation of the variability of the coefficients is accurate for all βs , with a 5% maximum error for β_{10} (see Table 3.2). These approximation errors are all positive for the null coefficients ($\beta_3, \beta_4, \beta_5$ and β_9) and all negative for the non zero coefficients. This fact may be a sign of some systematic distortion.

The empirical coverage is close to the desired $(1-\alpha)$ coverage (see Table 3.3). Not surprisingly, the maximum coverage errors correspond to β_{10} and β_1, the two smaller non-zero coefficients. For a limited sample size, these coefficients can be difficult to estimate correctly. In fact, for some samples, they can be erroneously shrinked to zero. This problem is well depicted in figure 3.2.

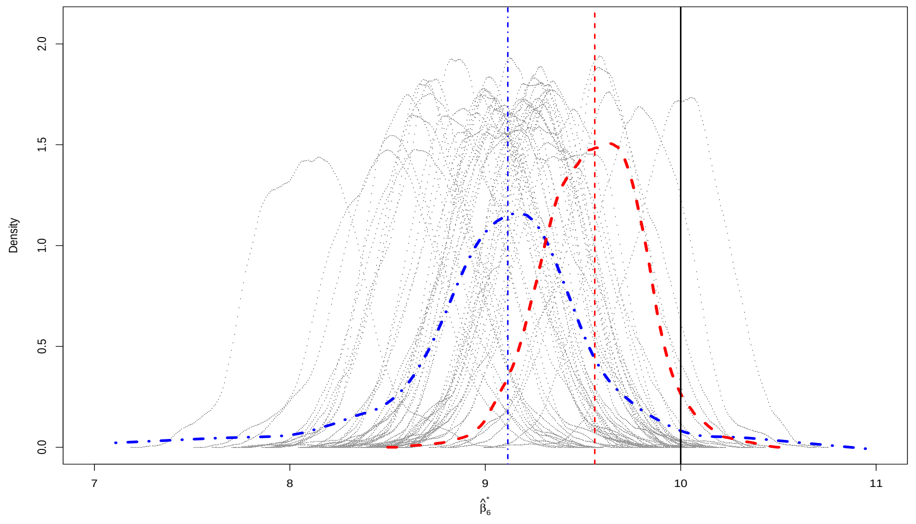

Figure 3.1: Estimated pdf of the ℓ_1-penalized $\hat{\beta}_6$ model coefficient (dashed line) and estimated pdf of the residual bootstrap replications (dot-dashed line). The vertical lines show the true value (solid), the mean value of the penalized coefficient (dashed) and the mean value of the bootstrap replications (dot-dashed).

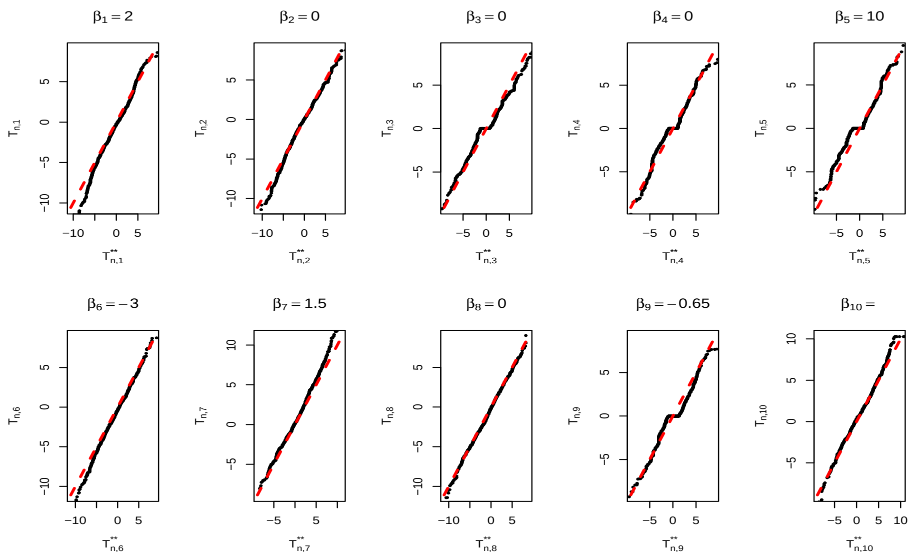

Figure 3.2: Residual resampling for the ℓ_1-penalized logistic model (2.2). Comparison of G and $\hat{G}$ distributions of the model coefficients using Q-Q plots.

3.3 A modified bootstrap method

[Chatterjee and Lahiri (2010)] propose a modified version of the bootstrap lasso estimator that more closely reproduces the sign-vector corresponding to the unknown parameter β. The idea is to force components of the lasso estimator $\hat{\beta}$ to be exactly zero whenever they are close to zero. The original lasso estimator is $\sqrt{n}$-consistent and its fluctuations are of the order $n^{-1/2}$ around the true value. This suggests that a neighborhood of order larger than $n^{-1/2}$ around the true value will contain the values of the lasso estimator with high probability.

Let $\{a_n\}$ be a sequence of real numbers such that

$$a_n + (n^{-1/2}\log n)a_n^{-1} \to 0 \quad as \quad n \to \infty. \tag{3.5}$$

For example, $a_n = cn^{-\delta}$ satisfies (3.5) for all $c \in (0,\infty)$ and $\delta \in (0, 2^{-1})$.

[Chatterjee and Lahiri (2010)] threshold the components of the lasso estimator $\hat{\beta}$ at a_n and define the modified lasso estimator as

$$\tilde{\beta} = \left(\tilde{\beta}_1, \dots, \tilde{\beta}_p\right)^{\mathrm{T}} \quad \text{with} \quad \tilde{\beta}_j = \hat{\beta}_j \cdot \mathbb{I}\left(|\hat{\beta}_j| \leq a_n\right), \tag{3.6}$$

where $\hat{\beta}$ is the usual lasso estimate defined in (3.2), $\mathbb{I}(\cdot)$ denotes the indicator function and $j = 1, \dots, p$.

Note that for a nonzero component β_j,

$$|\hat{\beta}_j| = |\beta_j| + O_p\left(n^{-1/2}\right) > \frac{|\beta_j|}{2} \geq a_n$$

for large n, with high probability. Hence, for n large, $\tilde{\beta}_j = \hat{\beta}_j$ with probability tending to 1. Thus, this shrinkage does not have any significant effect on the non-zero components.

However, for a zero component, $\beta_j = 0$,

$$|\hat{\beta}_j| = |\beta_j| + O_p\left(n^{-1/2}\right) = O_p\left(n^{-1/2}\right) \in [-a_n, a_n],$$

with probability tending to 1 as $n \to \infty$. Thus, for large n:

$$\tilde{\beta}_j = \hat{\beta}_j \cdot \mathbb{I}\left(|\hat{\beta}_j| \leq a_n\right) = 0$$

with probability tending to 1.

The shrinkage by a_n accomplishes our main objective of capturing the signs of the zero components precisely with probability tending to 1, as the simple size n goes to infinity.

Here is the modified algorithm proposed by [Chatterjee and Lahiri (2010)].

1. Calculate by cross-validation the optimal penalization parameter $\hat{\lambda}$ and estimate the lasso estimator $\hat{\beta}$ defined in (3.2) using the data set (X, y).

2. Calculate the thresholded coefficients $\tilde{\beta}$ defined in (3.6).

3. Calculate the modified residuals

$$r_i = y_i - x_i^{\mathrm{T}}\tilde{\beta}, \quad i = 1, \ldots, n.$$

   and consider the set of centered residuals $\{e_i = r_i - \bar{r} : i = 1, \ldots, n\}$, where $\bar{r} = n^{-1}\sum_{k=1}^{n} r_i$.

4. Select with replacement a random sample $\{e_1^{**}, \ldots, e_n^{**}\}$ of size n from the centered residuals e_i and set

$$y_i^{**} = x_i^{\mathrm{T}}\tilde{\beta} + e_i^{**}, \quad i = 1, \ldots, n.$$

5. Estimate the modified bootstrap lasso estimator using the penalization $\hat{\lambda}$ and

$$\hat{\beta}^{**} = \arg\min_{u \in \mathbb{R}^p} \sum_{i=1}^{n} (y_i^{**} - x_i^{\mathrm{T}}u)^2 + \hat{\lambda}\sum_{j=1}^{p} |u_j|. \tag{3.7}$$

6. Calculate the confidence interval $[\hat{\beta} + \tilde{\beta} - \hat{\beta}^{**}_{(1-\alpha/2)};\ \hat{\beta} + \tilde{\beta} - \hat{\beta}^{**}_{(\alpha/2)}]$.

7. Repeat steps 2 to 6 B times.

Let $\tilde{G}(\cdot)$ denote the conditional distribution of T^{**} given the observations, i.e. $\tilde{G}(B) = pr_*(T^{**} \in B), B \in \mathbb{R}^p)$. Thus, $\tilde{G}(\cdot)$ is the modified bootstrap approximation of the unknown distribution $G(\cdot)$ of T.

[Chatterjee and Lahiri (2010)] show that the modified bootstrap gives a valid approximation to the distribution, that $\tilde{G}(\cdot)$ is a consistent estimate of $G(\cdot)$:

$$\rho\left(\tilde{G}(\cdot), G(\cdot)\right) \to 0, \quad \text{as} \quad n \to \infty, \quad \text{with probability 1,}$$

where $\rho(\cdot,\cdot)$ is the Prohorov metric defined on the set of all the probability measures on $(\mathbb{R}^p, \mathcal{B}(\mathbb{R}^p))$.

In addition, they prove that the modified bootstrap also produces strongly consistent estimators of the asymptotic bias and variance of T.

[Chatterjee and Lahiri (2010)] also propose a computational-intensive algorithm for the choice of the optimal penalty parameter λ_0, where $\lambda_n = \sqrt{n}\lambda_0$. This algorithm is based on the calculation of $T_n^{**}(\lambda_0, a) = \sqrt{n}(\beta_n^{**} - \tilde{\beta}_n)$ on a grid of values of the penalization λ_0 and of the thresholding value

a, where β_n^{**} is the vector of lasso coefficients estimated on boostrap samples and $\tilde{\beta}_n$ is the vector of modified lasso estimates.

The optimal value λ_0^* is defined as:

$$\lambda_0^* \equiv \underset{\lambda_0, a}{\operatorname{argmin}}\, E_* ||T_n^{**}(\lambda_0, a)||^2 = \underset{\lambda_0, a}{\operatorname{argmin}}\, \sqrt{n} E_* ||\beta_n^{**} - \tilde{\beta}_n||^2 \tag{3.8}$$

In addition, the authors suggest to use the jackknife-after-bootstrap for selecting an appropriate threshold a in finite samples.

The computational burden of this algorithm is rather high even for moderate size datasets. This point cannot be ignored when making extensive simulation studies. Therefore, in all our numerical simulations we prefer to use the faster method based on cross-validation.

Figure 3.3 shows that the modified residual bootstrap gives a good approximation of distribution G for the coefficients β_2 to β_9. For β_1 and β_{10} the method yields approximations that are worse than the residual bootstrap approximations. This is not surprising because the true values of these coefficients are near zero (0.75 and -0.65, respectively), penalized regression often shrinks estimates toward zero and the modified residual bootstrap introduces a second shrinkage for the near-zero components of the $\hat{\beta}$ lasso estimator. This is evidenced in figures 3.4 and 3.5 where the distributions of $\tilde{\beta}_1$ and $\tilde{\beta}_{10}$ are clearly bimodal; the first modal value is near 0 and the second one is close to the true value (0.75 and -0.65, respectively).

Globally, Tables 3.1, 3.2 and 3.3 show that the modified residual bootstrap do not offer any evident improvement compared to residual bootstrap. On the contrary, the empirical coverage for β_1 and β_{10} are markedly lower than the corresponding coverage of the residual bootstrap.

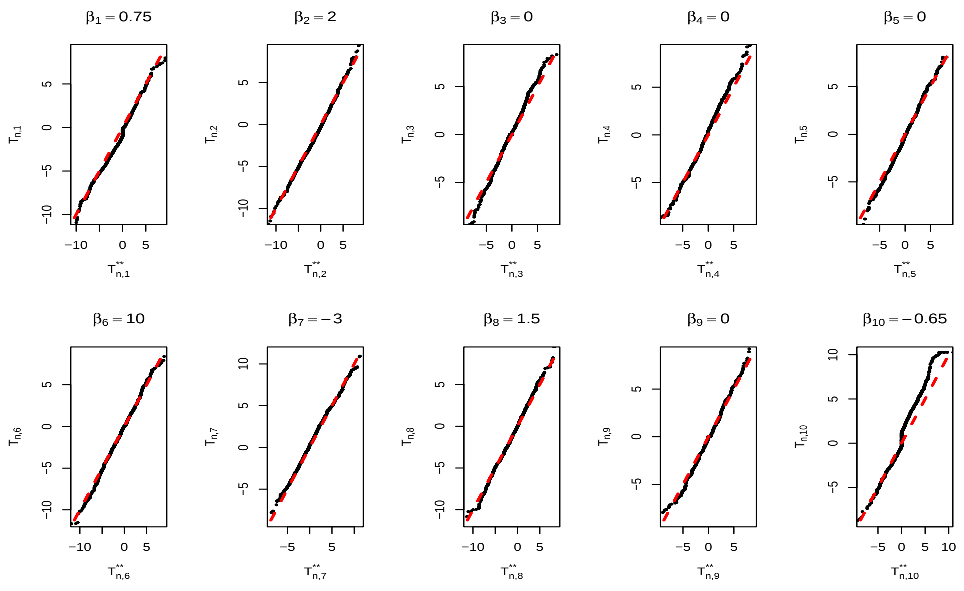

Figure 3.3: Modified residual bootstrap for the penalized coefficients (3.2) of the linear model (3.1). Comparison of the empirical G and $\hat{G}$ distributions of the coefficients using Q-Q plots.

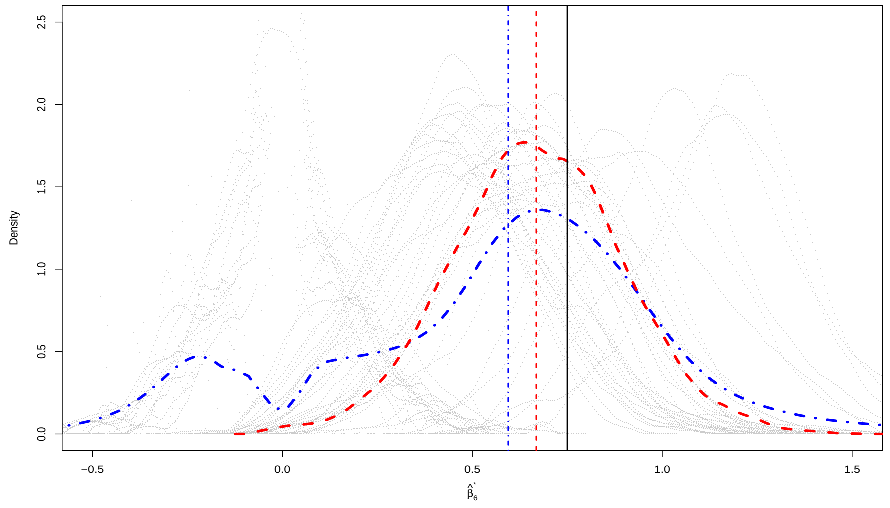

Figure 3.4: Estimated pdf of the ℓ_1-penalized $\hat{\beta}_2$ model coefficient (dashed line) and estimated pdf of the modified residual bootstrap replications (dot-dashed line). The vertical lines show the true value (solid, $\beta_2 = 0.75$), the mean value of the penalized coefficient (dashed) and the mean value of the bootstrap replications (dot-dashed).

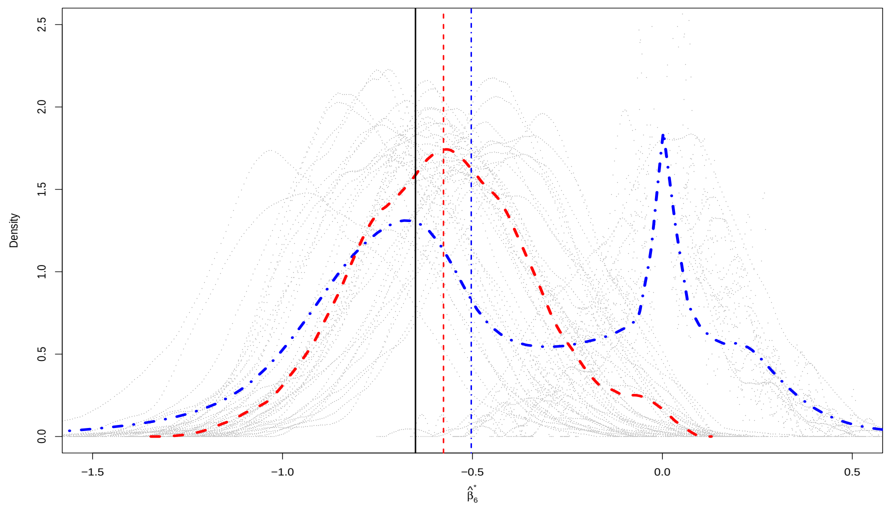

Figure 3.5: Estimated pdf of the ℓ_1-penalized $\hat{\beta}_{10}$ model coefficient (dashed line) and estimated pdf of the modified residual bootstrap replications (dot-dashed line). The vertical lines show the true value (solid, $\beta_2 = -0.65$), the mean value of the penalized coefficient (dashed) and the mean value of the bootstrap replications (dot-dashed).

3.4 Vector and residual resampling for penalized GLMs

In this section we start adapting the residual bootstrap algorithm to the class of penalized GLMs using the notion of standardized residual bootstrap.

This algorithm is very similar to the residual bootstrap for GLMs and can be summarized as follows.

1. Estimate the penalty $\hat{\lambda}$ by cross-validation, fit the penalized GLM, estimate model coefficients, calculate standardized Pearson residuals defined by $r_i = (y_i - \hat{\mu}_i)/\sqrt{v_i(1-h_i)}$ and mean-adjusted residuals $e_i = r_i - \bar{r}$, where $\bar{r} = 1/n \sum_i r_i$.

2. Resample with replacement the mean-adjusted residuals e_i and generate the bootstrapped residuals $e_1^*, e_2^*, \dots, e_n^*$.

3. Calculate the bootstrapped response variable by
$$y_i^* = \hat{\mu}_i + \sqrt{v_i} e_i^*. \tag{3.9}$$

4. Fit the penalized GLM to the (X, y^*) data set using the penalty $\hat{\lambda}$ and calculate the $\hat{\beta}^*$ estimate.

5. Calculate the confidence interval $[2\hat{\beta} - \hat{\beta}^*_{(1-\alpha/2)};\ 2\hat{\beta} - \hat{\beta}^*_{\alpha/2}]$.

6. Repeat steps 2 to 5 B times .

Figure 3.6 compares the empirical pdf G_j of the differences $T_j = \sqrt{n}(\hat{\beta}_j - \beta)$ to the bootstrap approximations $\hat{G}_j$ of $T_j^* = \sqrt{n}(\hat{\beta}_j^* - \hat{\beta})$ obtained applying the above algorithm to the logistic DGP (2.2). The distributions of zero coefficients appear to be adequately reproduced by bootstrap replications while for non zero coefficients the true and the approximated distributions show differences that increase for increasing (absolute) values of coefficients. The main difference between G_j and $\hat{G}_j$ is attributable to the inability of residual bootstrap to reproduce the bias of $\hat{\beta}$, that is $E(\hat{\beta} - \beta) \neq E(\hat{\beta}^* - \hat{\beta})$. This is clearly depicted in figure 3.7 and by data reported in tables 3.1 and 3.2. Of course, the resulting empirical coverage is inadequate (see Table 3.3).

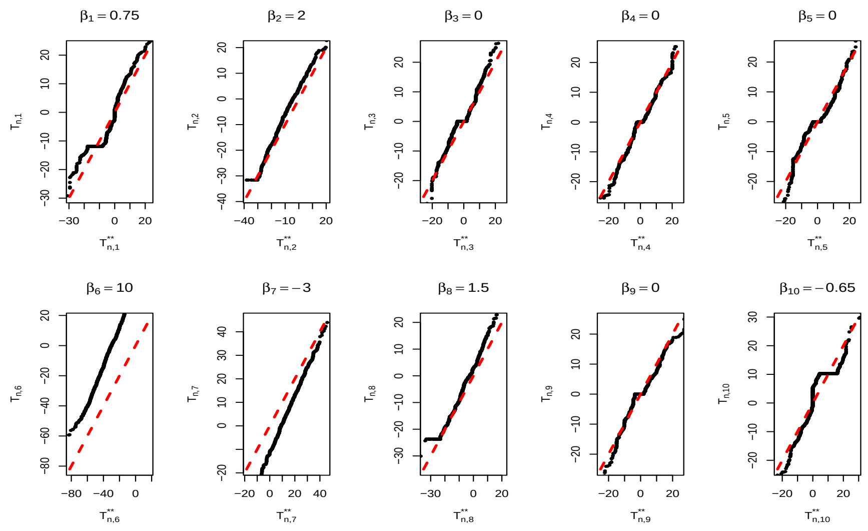

Figure 3.6: Residual resampling for the ℓ_1-penalized logistic model (2.2). Comparison of G and $\hat{G}$ distributions of the model coefficients using Q-Q plots.

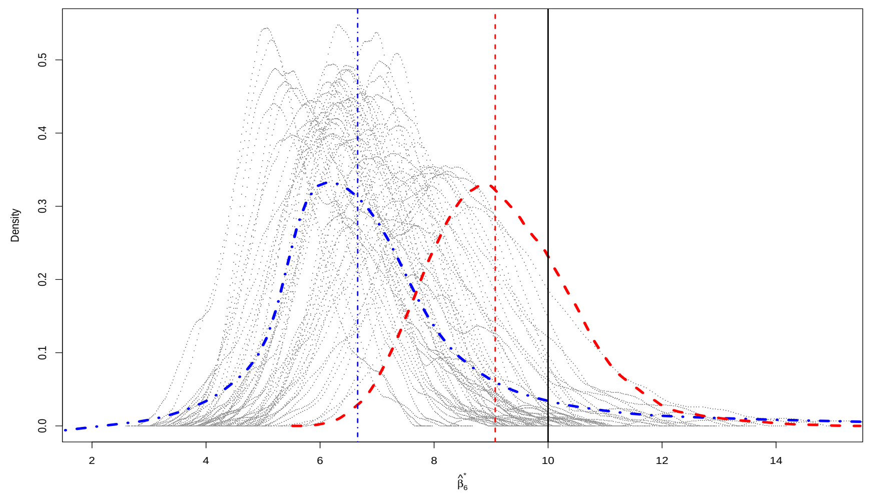

Figure 3.7: Estimated pdf of the ℓ_1-penalized $\hat{\beta}_6$ model coefficient (dashed line) and estimated pdf of the residual bootstrap replications (dot-dashed line). The vertical lines show the true value (solid), the mean value of the penalized coefficient (dashed) and the mean value of the bootstrap replications (dot-dashed).

The algorithm for implementing vector resampling in penalized GLMs is the same described for unpenalized GLMs.

1. Find the optimal penalty parameter $\hat{\lambda}$ using cross-validation on the $(x_1, y_1), \ldots, (x_n, y_n)$ data set.

2. Calculate the vector of parameters $\hat{\beta}$ of the penalized model using the penalty $\hat{\lambda}$.

3. Resample with replacement from the set of pairs $(x_1, y_1), \ldots, (x_n, y_n)$, ($x_i$ is the ith row of X) and generate the bootstrapped pairs (x_1^*, y_1^*), …, (x_n^*, y_n^*). Call (X^*, Y^*) the corresponding bootstrap data set.

4. Fit the penalized GLM to (X^*, Y^*) data using $\hat{\lambda}$ and get the $\hat{\beta}^*$ estimate.

5. Repeat steps 3 and 4 B times.

As expected, this algorithm, when applied to a logistic DGC, is not able to reproduce correctly the intrinsic bias of $\hat{\beta}$ due to shrinkage (see figures 3.8, 3.9 and table 3.1).

In addition, table 3.2 shows that vector resampling is not able to approximate adequately the standard deviation of the estimated $\hat{\beta}$ coefficients. The approximation errors are high and unacceptable for many zero and non-zero coefficients $(25 - 30\%)$.

Here we built confidence intervals using the 'naive percentile' interval $[\hat{\beta}^*_{(\alpha/2)};\ \hat{\beta}^*_{(1-\alpha/2)}]$. This choice was not motivated by any deductive reasoning but only by the fact that bootstrap estimates seem (very) approximately centered around the true β values . Unexpectedly, the resulting empirical coverages are fairly good (maximum error 6.3%).

In conclusion, vector resampling for penalized GLMs deserves further investigations.

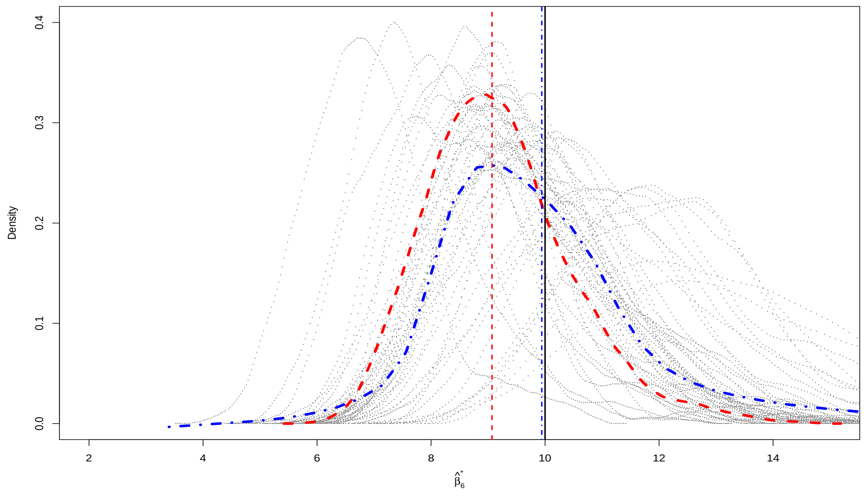

Figure 3.8: Estimated pdf of the ℓ_1-penalized $\hat{\beta}_6$ model coefficient (dashed line) and estimated pdf of vector resampling replications (dot-dashed line). The vertical lines show the true value (solid), the mean value of the penalized coefficient (dashed) and the mean value of the bootstrap replications (dot-dashed).

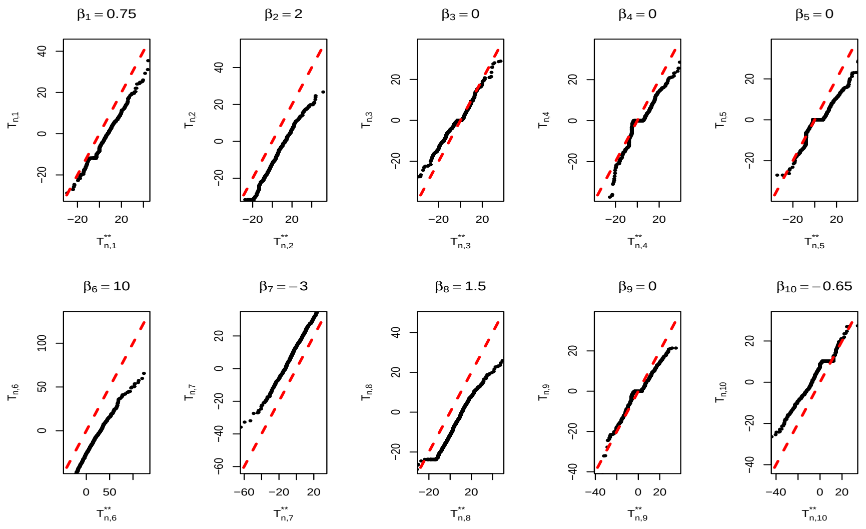

Figure 3.9: Vector resampling for the ℓ_1-penalized logistic model (2.2). Comparison of G and $\hat{G}$ distributions of the model coefficients using Q-Q plots.

3.5 Approximating lasso solutions

In a linear regression model, ridge regression shrinks the regression coefficients by imposing a quadratic penalty $\sum \beta_j^2$, that is by minimizing the following penalized residual sum of square errors

$$\begin{aligned}\hat{\beta}^{\text{ridge}} &= \underset{u\in\mathbb{R}^p}{\arg\min} \sum_{i=1}^{n}(y_i - x_i^{\mathrm{T}}u)^2 + \lambda\sum_{j=1}^{p}u_j^2 \qquad (3.10)\\ &= \underset{u\in\mathbb{R}^p}{\arg\min}\ (y - Xu)^{\mathrm{T}}(y - Xu) + \lambda u^{\mathrm{T}}u.\end{aligned}$$

The solution of the above minimization problem is:

$$\hat{\beta}^{\text{ridge}} = (X^{\mathrm{T}}X + \lambda I_p)^{-1}X^{\mathrm{T}}y, \qquad (3.11)$$

where X^{T} is the transpose of the X matrix, X^{-1} is the inverse of X and I_p is the identity matrix of order p.

The variance of $\hat{\beta}^{\text{ridge}}$ is

$$\mathrm{Var}(\hat{\beta}^{\text{ridge}}) = \sigma^2[(X^{\mathrm{T}}X) + \lambda I_p]^{-1}(X^{\mathrm{T}}X)[(X^{\mathrm{T}}X) + \lambda I_p]^{-1}$$

If we set different shrinkage parameters λ_j, $j = 1, 2, \ldots, p$, for the p covariates, the minimization problem of the ridge regression becomes

$$\begin{aligned}\hat{\beta}^{\text{ridge}} &= \underset{u\in\mathbb{R}^p}{\arg\min} \sum_{i=1}^{n}(y_i - x_i^{\mathrm{T}}u)^2 + \sum_{j=1}^{p}\lambda_j u_j^2 \qquad (3.12)\\ &= \underset{u\in\mathbb{R}^p}{\arg\min}\ (y - Xu)^{\mathrm{T}}(y - Xu) + u^{\mathrm{T}}\Lambda u,\end{aligned}$$

where Λ is the diagonal matrix defined by $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_p)$.

In this case, the solution of the minimization problem is given by

$$\hat{\beta}^{\text{ridge}} = (X^{\mathrm{T}}X + \Lambda)^{-1}X^{\mathrm{T}}y. \qquad (3.13)$$

[Tibshirani (1996)] noted that the lasso penalty $\sum|\beta_j|$ in (3.2) can be rewritten as $\sum\beta_j^2/|\beta_j|$. Therefore, the minimization problem of lasso can be rewritten as the ridge regression minimization problem of equation (3.12), where $\lambda_j = 1/|u_j|$:

$$\begin{aligned}\hat{\beta}^{\text{lasso}} &= \underset{u\in\mathbb{R}^p}{\arg\min} \sum_{i=1}^{n}(y_i - x_i^{\mathrm{T}}u)^2 + \lambda\sum_{j=1}^{p}\frac{1}{|u_j|}u_j^2 \qquad (3.14)\\ &= \underset{u\in\mathbb{R}^p}{\arg\min}\ (y - Xu)^{\mathrm{T}}(y - Xu) + \lambda u^{\mathrm{T}}\Lambda_u u, ,\end{aligned}$$

where $\Lambda_u \equiv \mathrm{diag}(1/|u_1|, \ldots, 1/|u_p|)$.

Of course, a division by zero arose when one or more u_j are zero. To obviate this problem, it is preferable to redefine Λ_u as

$$\Lambda_u = \mathrm{diag}(|u_1|, \ldots, |u_p|)$$

and to use the notion of generalized inverse Λ_u^- of a matrix [1] This is equivalent to put on the diagonal of Λ_u, the reciprocal of $|u_j|$ when $|u_j| > 0$ and 0 when $u_j = 0$. Equation (3.14) becomes:

$$\hat{\beta}^{\text{lasso}} = \underset{u \in \mathbb{R}^p}{\arg\min}\ (y - Xu)^{\mathrm{T}}(y - Xu) + \lambda u^{\mathrm{T}} \Lambda_u^- u.$$

Therefore, the $\hat{\beta}^{\text{lasso}}$ solution of the above optimization problem with quadratic constraints can be approximated by ridge regression using equation (3.13):

$$\hat{\beta}^{\text{lasso}} \approx (X^{\mathrm{T}}X + \lambda \Lambda_{\beta}^-)^{-1} X^{\mathrm{T}} y, \tag{3.15}$$

where $\Lambda_\beta = \mathrm{diag}(|\hat{\beta}_1|, \ldots, |\hat{\beta}_p|)$ and $\hat{\beta}^{\text{lasso}} = (\hat{\beta}_1, \ldots, \hat{\beta}_p)$.

Remembering the description of the IWLS algorithm given in the previous chapter and equation (2.9), this result can be directly extended to GLM giving

$$\hat{\beta}^{\text{lasso}} \approx (X^{\mathrm{T}}WX + \lambda \Lambda_{\beta}^-)^{-1} X^{\mathrm{T}} W z, \tag{3.16}$$

where W is the weight matrix (2.8) calculated at the final step and z is the vector of adjusted responses $z_i = x_i\hat{\beta}^{\text{lasso}} + (y_i - \hat{\mu}_i)\dot{g}(\hat{\mu}_i)$ calculated at the final step.

We can show this result by a numerical example using the logistic DGP considered in section 2.3.

```
set.seed(9876789)
n <- 1000
beta0 <- 0
betas <- c(0.75,2,0,0,0,10,-3,1.5,0,-0.65)
m <- length(betas)
X <- matrix(runif(n*m), nrow=n, ncol=m)-0.5
Xb <- beta0 + X %*% betas
```

[1] Let A be a diagonal matrix of order q. Suppose that the first $r < q$ elements on the diagonal of A are non-zero elements and the remaining $q - r$ are null values. Matrix A can be written

$$A = \begin{bmatrix} D_{r\times r} & 0 \\ 0 & 0 \end{bmatrix},$$

where $D_{r\times r}$ is a diagonal matrix of order r with all non-zero values on the diagonal. The generalized inverse (or pseudoinverse, or Moore-Penrose inverse) of A is given by:

$$A^- = \begin{bmatrix} D_{r\times r}^{-1} & 0 \\ 0 & 0 \end{bmatrix},$$

where $D_{r\times r}^{-1}$ is the inverse of the diagonal matrix $D_{r\times r}$.

```
pr <- 1/(1+exp(-Xb))
Y <- rbinom(n = n, prob = pr, size = 1)
dset.orig <- data.frame(y=Y, X=X)
```

We calculate the penalization parameter λ by cross-validation (command `optL1`) and estimate the lasso solution by the `penalized` command of the `penalized` R package [Goeman (2010)].

```
require(penalized)
opt0 <- optL1(y, penalized=~.-y, trace=F,
         model = "logistic", standardize = F,
         fold = 10, data = dset.orig)
lambda0.opt <- opt0$lambda
fit.lasso <- penalized(y, penalized=~.-y, trace=F,
              data = dset.orig, lambda1=lambda0.opt, lambda2=0)
beta.lasso <- c(fit.lasso@unpenalized,fit.lasso@penalized)
```

Now calculate the approximated solution given in (3.16).

```
mu.hat <- fit.lasso@fitted
V.mu <- mu.hat * (1 - mu.hat)
W  <- diag(V.mu)
Lambda.beta <- diag( abs(beta.lasso) )
Lambda.beta[1,1] <- 0          # Do not penalize intercept
X1 <- cbind(rep(1,n), X)
z <-  X1 %*% beta.lasso + (Y - mu.hat)/V.mu
beta.lasso.approx <- solve(t(X1) %*% W %*% X1 +
  lambda0.opt * ginv(Lambda.beta)) %*% t(X1) %*% W %*% z
```

The results below show that the approximation is rather good except for the zeros coefficients. This can be explained considering that one of the main differences between ridge and lasso regression is the ability of lasso to give sparse solutions.

```
            beta.true  beta.lasso      rigde.approx
(Intercept)      0.00  0.08171339        0.08252286
X.1              0.75  0.95987231        0.96108777
X.2              2.00  1.99905095        1.99397832
X.3              0.00  0.00000000        0.04139084
X.4              0.00 -0.15876658       -0.15981665
X.5              0.00  0.00000000       -0.01106971
X.6             10.00  8.36865274        8.36876934
X.7             -3.00 -2.63379702       -2.63343705
X.8              1.50  0.97299500        0.97087887
X.9              0.00  0.00000000       -0.03008823
X.10            -0.65 -0.42360931       -0.42227225
```

Another way to approximate the $\hat{\beta}^{\text{lasso}}$ estimates is to use the IWLS algorithm with ridge penalization.
The kth iteration of this algorithm consists of:

1. Calculate the adjusted responses

$$\begin{aligned} z_i^{(k)} &= \eta_i^{(k)} + (y_i - \hat{\mu}_i^{(k)})\dot{g}(\hat{\mu}_i^{(k)}) \\ &= x_i\hat{\beta}^{(k)} + (y_i - \hat{\mu}_i^{(k)})\dot{g}(\hat{\mu}_i^{(k)}) \end{aligned} \tag{3.17}$$

2. Build the weight diagonal matrix

$$W^{(k)} = \text{diag}\left(\frac{1}{a(\phi)[\dot{g}(\hat{\mu}_1^{(k)})]^2V(\hat{\mu}_1^{(k)})}, \ldots, \frac{1}{a(\phi)[\dot{g}(\hat{\mu}_n^{(k)})]^2V(\hat{\mu}_n^{(k)})}\right) \tag{3.18}$$

3. Build the diagonal matrix of penalties

$$\Lambda_\beta^{(k)} = \text{diag}(|\hat{\beta}_1^{(k)}|, \ldots, |\hat{\beta}_p^{(k)}|)$$

4. Run the weighted regression of the $z^{(k)}$ adjusted response on the covariates X with weights $W^{(k)}$ and calculate the coefficients $\hat{\beta}^{(k+1)}$ given by:

$$\hat{\beta}^{(k+1)} = \left(X^{\mathrm{T}}W^{(k)}X + \lambda\Lambda_\beta^{(k)-}\right)^{-1} X^{\mathrm{T}}W^{(k)}z^{(k)} \tag{3.19}$$

   and proceed to the next iteration.

This algorithm can be repeated until convergence in $\hat{\beta}$ or log-likelihood or deviance. The variance of the $\hat{\beta}$ coefficients can be estimated from the final iteration:

$$V(\hat{\beta}) = (X^{\mathrm{T}}WX + \lambda\Lambda_\beta^{-})^{-1}Var[X^{\mathrm{T}}(y - \hat{\mu})](X^{\mathrm{T}}WX + \lambda\Lambda_\beta^{-})^{-1}, \tag{3.20}$$

where W and Λ_β are calculated at the final step.

Below we implement the above algorithm in R testing it on the logistic DGP.

```
beta.iwls <- rep(0,p+1)
beta.iwls[1] <- log(mean(Y)/(1-mean(Y)))
mu.hat <- rep(mean(Y),n)
nstep <- 25
Lambda.beta <- diag(1,p+1)
Lambda.beta[1,1] <- 0      # Do not penalize intercept
for (k in 1:nstep) {
  W <- diag(as.numeric(mu.hat*(1-mu.hat)))
  z <- X1 %*% beta.iwls + (Y - mu.hat)/V.mu
  beta.iwls <- solve(t(X1) %*% W %*% X1 +
      lambda0.opt * Lambda.beta) %*% t(X1) %*% W %*% z
  eta <- X1 %*% beta.iwls
  mu.hat <- 1/(1+exp(-eta))
  Lambda.beta <- ginv(diag(as.numeric(abs(beta.iwls))))
  Lambda.beta[1,1] <- 0
}
```

The command `Lambda.beta[1,1] <- 0` means that we do not penalize the intercept of the linear model.

We compare the ridge-IWLS approximation with the lasso estimate and the ridge approximation given above.

```
            beta.true  beta.lasso      ridge.approx     beta.iwls
(Intercept)      0.00  0.08171339        0.08252286  0.0818521772
X.1              0.75  0.95987231        0.96108777  0.9605606545
X.2              2.00  1.99905095        1.99397832  1.9971903510
X.3              0.00  0.00000000        0.04139084  0.0000002508
X.4              0.00 -0.15876658       -0.15981665 -0.1591810940
X.5              0.00  0.00000000       -0.01106971 -0.0000003419
X.6             10.00  8.36865274        8.36876934  8.3686161031
X.7             -3.00 -2.63379702       -2.63343705 -2.6342924739
X.8              1.50  0.97299500        0.97087887  0.9728298768
X.9              0.00  0.00000000       -0.03008823 -0.0323964359
X.10            -0.65 -0.42360931       -0.42227225 -0.4236083564
```

Below, we give the approximation errors for the two methods. The ridge-IWLS approximation is clearly better that the ridge approximation, especially for the null coefficients.

```
            beta.true  ridge.approx     beta.iwls
(Intercept)      0.00  0.0008094701  0.0001387867
X.1              0.75  0.0012154524  0.0006883408
X.2              2.00 -0.0050726223 -0.0018605952
X.3              0.00  0.0413908357  0.0000002508
X.4              0.00 -0.0010500672 -0.0004145099
X.5              0.00 -0.0110697068 -0.0000003419
X.6             10.00  0.0001165994 -0.0000366366
X.7             -3.00  0.0003599733 -0.0004954533
X.8              1.50 -0.0021161271 -0.0001651209
X.9              0.00 -0.0300882277 -0.0323964359
X.10            -0.65  0.0013370590  0.0000009561
```

3.6 One-step bootstrap for penalized GLMs

The considerations of the previous section suggest a first approach for the development of a one-step bootstrap for ℓ_1-penalized GLMs which is similar to the one-step procedure proposed by [Moulton and Zeger (1991)].

Let $\hat{\beta}$ be the estimated coefficients of the penalized model and $\hat{\mu}$ be the vector of the outcome estimates. We propose to generate the bootstrap replications of the penalized coefficients $\hat{\beta}^{\text{lasso}}$ using the approximating equation (3.16)

$$\hat{\beta}^* = (X^{\mathrm{T}}WX + \lambda\Lambda_{\beta}^{-})^{-1}X^{\mathrm{T}}Wz^*, \tag{3.21}$$

where z^* is the bootstrapped version of the adjusted response vector

$$z^* = X\hat{\beta} + \Gamma V^{1/2}e^*. \tag{3.22}$$

In this equation, $\Gamma = \text{diag}(\dot{g}(\hat{\mu}_1), \ldots, \dot{g}(\hat{\mu}_n))$, $V = \text{diag}(v_1, \ldots, v_n)$, $v_i = V(\hat{\mu}_i)a(\phi)$, and the vector $e^* = \{e_1^*, \ldots, e_n^*\}$ is generated by resampling with replacement the mean-adjusted standardized Pearson residuals given in equation (2.12).

Defining $\Psi \equiv (X^{\mathrm{T}}WX + \lambda\Lambda_{\beta}^{-})^{-1}X^{\mathrm{T}}W$ and combining the two above equations, we can write

$$\hat{\beta}^* = \Psi z^* = \Psi X\hat{\beta} + \Psi\Gamma V^{1/2}\, e^*. \tag{3.23}$$

For unpenalized GLMs (i.e. $\lambda = 0$), we have $\Psi = (X^{\mathrm{T}}WX)^{-1}X^{\mathrm{T}}W$ and

$$\begin{aligned}
\hat{\beta}^* &= \Psi z^* = \Psi X\hat{\beta} + \Psi\Gamma V^{1/2}\, e^* \\
&= (X^{\mathrm{T}}WX)^{-1}(X^{\mathrm{T}}WX)\hat{\beta} + (X^{\mathrm{T}}WX)^{-1}X^{\mathrm{T}}W\Gamma V^{1/2}\, e^* \\
&= \hat{\beta} + (G^{\mathrm{T}}G)^{-1}G^{\mathrm{T}}e^*,
\end{aligned}$$

where $G = W^{1/2}X$ and $\Gamma V^{1/2} = W^{-1/2}$. This means that when $\lambda = 0$, the proposed method of equation (3.23) corresponds to the one-step bootstrap of [Moulton and Zeger (1991)].

Another important point to consider here is the construction of bootstrap confidence intervals. The usual bootstrap confidence interval with coverage $(1-\alpha)$:

$$[2\hat{\beta} - \hat{\beta}^*_{(1-\alpha/2)};\ 2\hat{\beta} - \hat{\beta}^*_{\alpha/2}]$$

is based on the idea that the distribution of $(\hat{\beta} - \beta)$ can be approximated through the bootstrap distribution of $(\hat{\beta}^* - \hat{\beta})$. For biased estimators similar to ridge or lasso regression this is not in general true because these estimators may lack a pivot. The quantity $(\hat{\beta} - \beta)$ is a pivot if its sampling distribution does not depend on β. For a shrinkage estimator $\hat{\beta}$, it is possible to show that $(\hat{\beta} - \beta) \approx (\hat{\beta}^* - \hat{\beta} + A - \hat{A})$ where A depends on β. [Vinod (1995)] proposes to solve the lack of a pivot problem by the following confidence interval:

$$\left[(\Psi X)^{-1}\hat{\beta} + \hat{\beta} - (\Psi X)^{-1}\hat{\beta}^*_{(1-\alpha/2)};\ (\Psi X)^{-1}\hat{\beta} + \hat{\beta} - (\Psi X)^{-1}\hat{\beta}^*_{\alpha/2}\right].$$

For unpenalized linear models, $\Psi = (X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}$ and the above equation reduces to:

$$[2\hat{\beta} - \hat{\beta}_{(1-\alpha/2)};\ 2\hat{\beta} - \hat{\beta}_{(\alpha/2)}].$$

The proposed one-step bootstrap algorithm for penalized GLM can be summarized as follows.

1. Find the penalty $\hat{\lambda}$ by cross-validation, fit the penalized GLM, estimate model coefficients $\hat{\beta}$ and the vector $\hat{\mu}$ of estimated outcomes.

2. Calculate the weight matrix

$$W = \operatorname{diag}\left(\frac{1}{a(\phi)[\dot{g}(\hat{\mu}_1)]^2 V(\hat{\mu}_1)}, \ldots, \frac{1}{a(\phi)[\dot{g}(\hat{\mu}_n)]^2 V(\hat{\mu}_n)}\right),$$

   the diagonal matrix of penalties

$$\Lambda_\beta = \operatorname{diag}(|\hat{\beta}_1|, \ldots, |\hat{\beta}_p|)$$

   and the matrix

$$\Psi = (X^{\mathrm{T}}WX + \lambda\Lambda_\beta^-)^{-1}X^{\mathrm{T}}W.$$

3. Calculate the standardized Pearson residuals $r_i = (y_i - \hat{\mu}_i)/\sqrt{v_i(1-h_i)}$ and the mean-adjusted residuals $e_i = r_i - \bar{r}$, where $\bar{r} = 1/n\sum_i r_i$.

4. Resample with replacement the mean-adjusted residuals e and generate the bootstrapped residuals $e^* = (e_1^*, \ldots, e_n^*)$.

5. Calculate the one-step bootstrapped β^* coefficients by:

$$\hat{\beta}^* = \Psi \cdot \left(X\hat{\beta} + \Gamma V^{1/2} e^*\right). \tag{3.24}$$

6. Repeat steps 4 and 5 B times and save the B bootstrap estimates $\hat{\beta}^*$.

7. Using the set $\{\hat{\beta}_1^*, \ldots \hat{\beta}_B^*\}$, estimate the quantiles $\hat{\beta}_{\alpha/2}^*$ and $\hat{\beta}_{1-\alpha/2}^*$

8. Calculate the confidence interval with coverage probability α:

$$\left[\hat{\beta} + (\Psi X)^{-1}(\hat{\beta} - \hat{\beta}_{(1-\alpha/2)}^*);\ \hat{\beta} + (\Psi X)^{-1}(\hat{\beta} - \hat{\beta}_{(\alpha/2)}^*)\right].$$

For a linear model ,$W = I_n$ and $\Psi = (X^{\mathrm{T}}X + \lambda\Lambda_\beta^-)^{-1}X^{\mathrm{T}}$, where I_n is the identity matrix of order n. Therefore,

$$\hat{\beta}^* \quad = (X^{\mathrm{T}}X + \lambda\Lambda_\beta^-)^{-1}X^{\mathrm{T}} \cdot (X\hat{\beta} + e^*)$$

where $X\hat{\beta} + \epsilon^* = y^*$.

The algorithm was applied to the linear (2.1) and the logistic (2.2) DGPs. Two simulation studies were performed with $S = 2500$ samples and $B = 250$ bootstrap replicates for each sample. The results given in Table 3.3 confirm the validity of the proposed method. The empirical coverages for the $\hat{\beta}_j$ coefficients, $j = 1, \dots, 10$, are rather close to the desired 90% coverage, particularly in the linear case where the coverage errors are similar to the errors of the residual bootstrap (largest error is -3.8%). For the logistic model, the coverage errors are all approximately below 3%, except for the the largest coefficients $\hat{\beta}_6$, whose error is 7.9%. This coverage error is probably due to the fact that the method roughly approximates the standard deviation of $\hat{\beta}_6$ (1.65 vs 1.25, +33%, see Table 3.2).

In the next sections we show how it is possible to further reduce these errors using the double boostrap algorithm.

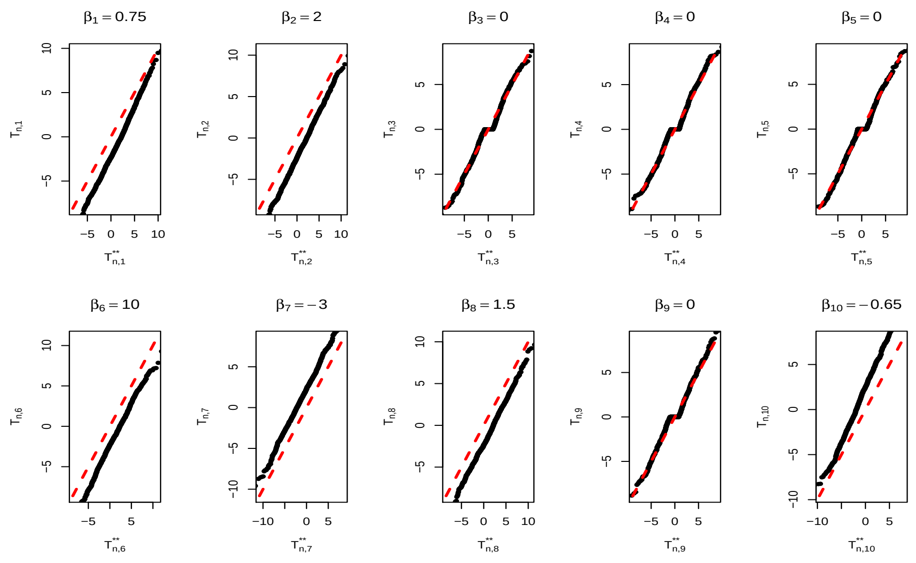

Figure 3.10: One-step bootstrap for the ℓ_1-penalized linear model (2.1). Comparison of G and $\hat{G}$ distributions of the model coefficients using Q-Q plots.

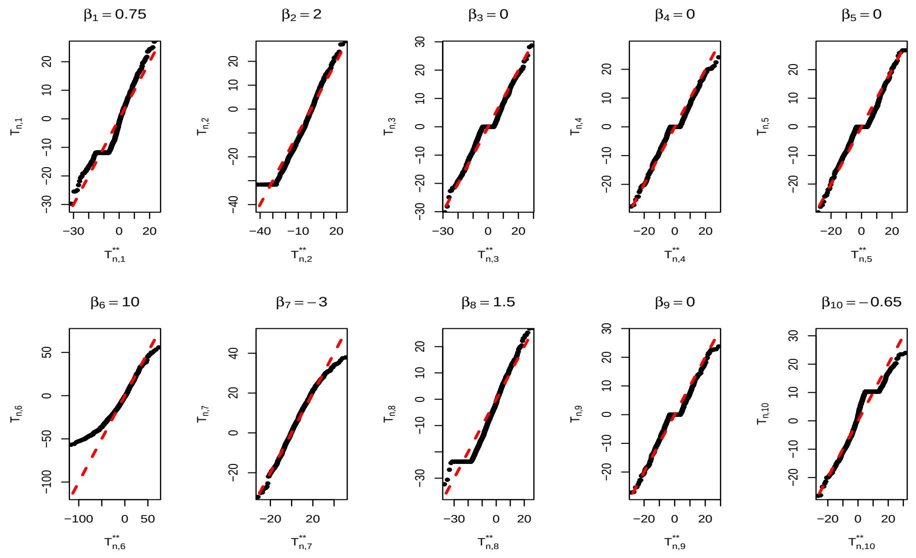

Figure 3.11: One-step bootstrap for the ℓ_1-penalized logistic model (2.2). Comparison of G and $\hat{G}$ distributions of the model coefficients using Q-Q plots.

3.7 The double bootstrap

The double bootstrap was first proposed by [Beran (1987)]. Its main advantage over the single bootstrap is that the double bootstrap confidence intervals typically have a higher order of accuracy. The double bootstrap improves the accuracy of a single bootstrap by estimating the coverage error of a confidence interval and then using this estimate to adjust the single bootstrap.

We start describing the double bootstrap algorithm for a linear model following the suggestions given in [McCullagh and Vinod (1998)].

1. Based on the original sample $\mathbf{z}_n = (z_1, \ldots, z_n)$, with $z_i = (\mathbf{x}_i, y_i)$ and $\mathbf{x}_i = (x_{i1}, \ldots, x_{ip})$, calculate the estimate $\hat{\theta}$ for the parameter of interest θ and a vector of fitted values $\hat{y}$.

2. Form the residuals $r = y - \hat{y}$ and the recentered residuals $e = r - \bar{r}$, where $\bar{r} = 1/n \sum_i r_i$.

3. A large number of time $j = 1, \ldots J$, randomly resample with replacement from e to from J vectors of resampled residuals e^*_j.

4. Form the vector $y^*_j = \hat{y} + e^*_j$.

5. For each j, calculate the bootstrap estimates of the parameter of interest $\hat{\theta}^*_j$ and store the resampled residuals e^*_j.

6. Calculate the bootstrap standard deviation $\hat{\sigma}^* = [\frac{1}{J-1} \sum_{j=1}^{J} (\hat{\theta}^*_j - \bar{\theta}^*)^2]^{\frac{1}{2}}$ where $\bar{\theta}^* = \frac{1}{J} \sum_{j=1}^{J} \hat{\theta}^*_j$ and form the $j = 1, \ldots, J$ roots
$$\hat{R}^*_j = \frac{\hat{\theta}^*_j - \hat{\theta}}{\hat{\sigma}^*}.$$

7. For each first-stage bootstrap resample j, a large number of times $k = 1, \ldots, K$, shuffle e^*_j to form e^{**}_{jk}, form the output vector $y^{**}_{jk} = \hat{y}^*_j + e^{**}_{jk}$ and estimate $\hat{\theta}^{**}_{jk}$.

8. Form the root
$$\hat{R}^{**}_{jk} = \frac{\hat{\theta}^{**}_{jk} - \hat{\theta}^*_j}{\hat{\sigma}^{**}_j},$$
where $\hat{\sigma}^{**}_j = [\frac{1}{K-1} \sum_{k=1}^{K} (\hat{\theta}^{**}_{jk} - \bar{\theta}^{**}_j)^2]^{1/2}$ and $\bar{\theta}^{**}_j = \frac{1}{K} \sum_{k=1}^{K} \hat{\theta}^{**}_{jk}$.

9. Calculate the proportion of times Z_j that $\hat{R}^{**}_{jk} \leq \hat{R}^*_j$, i.e.
$$Z_j = \frac{\#\{\hat{R}^{**}_{jk} \leq \hat{R}^*_j\}}{K}.$$

10. After all bootstrapping operations are complete, we have J estimates $\hat{\theta}_j^*$, $\hat{R}_j^*$ and Z_j. Variable $Z_j \in [0, 1]$ is now used to adjust the first-stage intervals. Suppose we need to determine the upper $(1-\alpha)$ limit for θ. First, we choose the $(1-\alpha)$-th quantile $q_{(1-\alpha)}$ of Z_j. Then, we choose the $q_{(1-\alpha)}$-th quantile of $\hat{R}_j^*$. This is the adjusted upper $1-\alpha$ limit for the true root R^*. Similar operations determine a lower limit.

The functioning of the double bootstrap is based on some basic facts, which we briefly summarize here. Let $\mathbf{z}_n = (z_1, \ldots, z_n)$, with $z_i = (\mathbf{x}_i, y_i)$ and $\mathbf{x}_i = (x_{i1}, \ldots, x_{ip})$ be a sample from a distribution with true cdf F, which has a parameter $\theta(F)$ with range Θ. Let θ_n be an estimator based on the sample $\mathbf{z}_n$ with standard deviation σ_n. Consider the studentized statistics:

$$R(\mathbf{z}_n, \theta) = \frac{\hat{\theta} - \theta}{\hat{\sigma}}, \tag{3.25}$$

and let $H(\cdot)$ be its distribution.

If $H(\cdot)$ does not depend on θ, then R is said to be a pivot. Otherwise it is a root. It is known that even if the transformation (3.25) does not completely eliminate the dependence of R on θ, it can decrease the dependence and yield an improved confidence interval.

[Beran (1987)] proves that R^{**} is closer to being pivotal than R^*, i.e. the distribution H^{**} of R^{**} is less dependent on F than is the distribution H^* of R^*. Therefore, if pivoting once can improve the order of accuracy of the bootstrap estimates, pivoting twice can improve it more. In the same way that the single bootstrap uses $\hat{R}^* - \hat{R}$ to approximate $\hat{R} - R$, the double bootstrap uses $\hat{R}^{**} - \hat{R}^*$ to approximate $\hat{R}^* - \hat{R}$.

[Beran (1987)] also shows that the second stage sampling distribution of Z_j is uniform over [0, 1] only in the ideal situation when the root R^* is exactly pivotal. [Vinod (1995)] proposes to plot the distribution of Z_j for revealing its shape and its deviations from the uniform distribution.

3.8 Improving the one-step bootstrap for penalized GLMs

In this section we apply double bootstrap to the one-step bootstrap for ℓ_1-penalized GLMs proposed in section 3.6. The resulting algorithm is a bit long and can be summarized as follows.

1. Based on the original sample $\mathbf{z}_n = (z_1, \dots, z_n)$, with $z_i = (\mathbf{x}_i, y_i)$ and $\mathbf{x}_i = (x_{i1}, \dots, x_{ip})$, estimate, by cross-validation, the penalization parameter $\hat{\lambda}$ and then calculate the vector $\hat{\beta}$ of penalized model coefficients for the parameter of interest β and a vector of fitted values $\hat{\mu}$.

2. Form the vector of adjusted responses $z_j^* = x_j\hat{\beta} + \dot{g}(\hat{\mu}_j)\sqrt{v_j}e_j^*$, the weight matrix

$$W = \operatorname{diag}\left(\frac{1}{a(\phi)[\dot{g}(\hat{\mu}_1)]^2 V(\hat{\mu}_1)}, \dots, \frac{1}{a(\phi)[\dot{g}(\hat{\mu}_n)]^2 V(\hat{\mu}_n)}\right),$$

   the penalty matrix

$$\Lambda_\beta = \operatorname{diag}(|\hat{\beta}_1|, \dots, |\hat{\beta}_p|)$$

   and the matrix

$$\Psi = (X^{\mathrm{T}}WX + \hat{\lambda}\Lambda_\beta^-)^{-1}X^{\mathrm{T}}W.$$

3. Form the standardized Pearson's residuals $r_i = (y_i - \hat{\mu}_i)/\sqrt{v_i(1-h_i)}$ and the recentered residuals $e_i = r_i - \bar{r}$, where $\bar{r} = 1/n\sum_i r_i$.

4. A large number of time $j = 1, \dots J$, randomly resample with replacement from e to J vectors of resampled residuals e_j^*.

5. For each j, store the resampled residuals e_j^* and estimate the one-step bootstrap β^* coefficients by:

$$\hat{\beta}^* = \Psi \cdot \left(X\hat{\beta} + \Gamma V^{1/2}\, e^*\right), \qquad (3.26)$$

   where $\Gamma = \operatorname{diag}(\dot{g}(\hat{\mu}_1), \dots, \dot{g}(\hat{\mu}_n))$ and $V = \operatorname{diag}(v_1, \dots, v_n)$, $v_i = V(\hat{\mu}_i)a(\phi)$.

6. Calculate the bootstrap standard deviation $\hat{\sigma}^* = [\frac{1}{J-1}\sum_{j=1}^J(\hat{\beta}_j^* - \bar{\beta}^*)^2]^{\frac{1}{2}}$ where $\bar{\beta}^* = \frac{1}{J}\sum_{j=1}^J \hat{\beta}_j^*$ and form the $j = 1, \dots, J$ roots

$$\hat{R}_j^* = \frac{\hat{\beta}_j^* - \hat{\beta}}{\hat{\sigma}^*}.$$

7. For each bootstrap resample j, shuffle e_j^* K times to form e_{jk}^{**}, generate the output vector $y_{jk}^{**} = \hat{y}_j^* + e_{jk}^{**}$ and estimate

$$\hat{\beta}_{jk}^{**} = \Psi \cdot \left(X\hat{\beta} + \Gamma V^{1/2}\, e_{jk}^{**} \right).$$

8. Form the second-stage root

$$\hat{R}_{jk}^{**} = \frac{\hat{\beta}_{jk}^{**} - \hat{\beta}_j^*}{\hat{\sigma}_j^{**}},$$

where $\hat{\sigma}_j^{**} = [\frac{1}{K-1} \sum_{k=1}^{K} (\hat{\beta}_{jk}^{**} - \bar{\beta}_j^{**})^2]^{1/2}$ and $\bar{\beta}_j^{**} = \frac{1}{K} \sum_{k=1}^{K} \hat{\beta}_{jk}^{**}$.

9. Calculate the proportion of times Z_j that $\hat{R}_{jk}^{**} \leq \hat{R}_j^*$, i.e.

$$Z_j = \frac{\#\{\hat{R}_{jk}^{**} \leq \hat{R}_j^*\}}{K}.$$

10. After all bootstrapping operations are complete, for each of the p model coefficients estimate the $(\alpha/2)$-th and the $(\alpha/2)$-th quantiles of Z_j: $q_{(\alpha/2)}$ and $q_{(\alpha/2)}$. Then, estimate the $q_{(\alpha/2)}$-th and the $q_{(\alpha/2)}$-th adjusted quantile of $\hat{\beta}_j^*$: $\hat{\beta}_{\mathrm{LO}}^*$ and $\hat{\beta}_{\mathrm{UP}}^*$, respectively. Finally, calculate the confidence interval:

$$\left[\hat{\beta} + (\Psi X)^{-1} (\hat{\beta} - \hat{\beta}_{\mathrm{UP}}^*);\ \hat{\beta} + (\Psi X)^{-1} (\hat{\beta} - \hat{\beta}_{\mathrm{LO}}^*) \right].$$

We have tested the above algorithm on the linear and logistic DGPs of equations (2.1) and (2.2). The results are encouraging. We observe a sensible reduction of the coverage error for β_6 (from 7.9% to −0.4%). The coverage error on the other coefficients remains substantially unchanged. It is worth to note only a slight increase of the coverage errors for zero coefficients (approximately from 3% to 4%).

In our opinion, these results show that the addition of double bootstrap to our one-step algorithm for ℓ_1-penalized GLMs can be beneficial and therefore deserves further detailed investigation.

Table 3.1: Bootstrap estimates of bias for the ℓ_1-penalized coefficients of the linear (2.1) and the logistic (2.2) DGPs.

	β_0	β_1	β_2	β_3	β_4	β_5	β_6	β_7	β_8	β_9	β_{10}
Coefficients	0	0.75	2	0	0	0	10	-3	1.5	0	-0.65
Linear model											
Empirical value of bias	0	-0.0744	-0.0691	0	0	0	-0.0679	0.0749	-0.0655	0	0.0765
Vector resampling	0.0021	-0.0525	-0.0599	-0.0010	0.0014	0.0062	-0.0652	0.0638	-0.0693	0.0014	0.0589
Residual resampling	0.0010	-0.1418	-0.1398	0.035	0.0009	0.0104	-0.1471	0.1504	-0.1414	-0.0029	0.1302
Modified residual	0.0005	-0.1420	-0.1448	-0.0051	0.0016	-0.0042	-0.1563	0.1426	-0.1447	0.0041	0.1424
One-step residual resam.	0.0001	-0.0642	-0.0738	0.0009	0.0026	0.0008	-0.0757	0.0736	-0.0710	0.0008	0.0618
Logistic model											
Empirical value of bias	0	-0.2700	-0.4176	0	0	0	-0.8124	0.5124	-0.3890	0	0.2200
Vector resampling	-0.0081	-0.1306	-0.2686	0.0032	0.0032	-0.0013	0.1584	0.2335	-0.2286	-0.0205	0.1289
Residual resampling	-0.0023	-0.4814	-1.0977	-0.0027	-0.0091	0.0074	-3.2424	1.4568	-0.8790	-0.0040	0.4176
One-step residual resam.	-0.0009	-0.1602	-0.3715	0.0012	0.0060	0.0052	-1.0987	0.5064	-0.2865	0.0053	0.1391

Table 3.2: Bootstrap estimates of the standard deviation for the ℓ_1-penalized coefficients of the linear (2.1) and the logistic (2.2) DGPs. (Percentage errors in round brackets)

	β_0	β_1	β_2	β_3	β_4	β_5	β_6	β_7	β_8	β_9	β_{10}
Coefficients	0	0.75	2	0	0	0	10	-3	1.5	0	-0.65
Linear model											
Empirical value of std	0.0645	0.2218	0.2291	0.1815	0.1815	0.1815	0.2273	0.2296	0.2226	0.1815	0.2288
Vector resampling	0.0644	0.2265	0.2267	0.1998	0.1999	0.1994	0.2269	0.2276	0.2266	0.1994	0.2242
	(-0.15)	(2.12)	(-1.05)	(0.10)	(0.10)	(9.86)	(-0.18)	(-0.18)	(1.80)	(9.86)	(-2.01)
Residual resampling	0.0630	0.2201	0.2225	0.1879	0.1879	0.1872	0.2223	0.2224	0.2222	0.1875	0.2173
	(-2.33)	(-0.77)	(-2.88)	(3.53)	(3.53)	(3.14)	(-2.20)	(-3.14)	(-0.18)	(3.31)	(-5.03)
Modified residual	0.0622	0.2139	0.2278	0.1720	0.1722	0.1723	0.2186	0.2181	0.2184	0.1720	0.2096
	(-3.57)	(-3.56)	(-0.57)	(-5.23)	(-5.12)	(-5.07)	(-3.83)	(-5.01)	(-1.89)	(-5.23)	(-8.40)
One-step residual	0.0629	0.1933	0.2106	0.1668	0.1662	0.1717	0.2172	0.2129	0.2075	0.1718	0.1881
resampling	(-2.48)	(-12.85)	(-8.08)	(-8.10)	(-8.43)	(-5.40)	(-4.44)	(-7.27)	(-6.78)	(-5.34)	(-17.79)
Logistic model											
Empirical value of std	0.1932	0.5751	0.7567	0.4833	0.4833	0.4833	1.2387	0.8064	0.7116	0.4833	0.5687
Vector resampling	0.2150	0.6669	0.7982	0.6264	0.6272	0.6222	1.5276	0.8787	0.7446	0.6274	0.6575
	(11.28)	(15.96)	(5.48)	(29.61)	(29.77)	(28.74)	(23.32)	(8.97)	(4.64)	(29.82)	(15.61)
Residual resampling	0.1713	0.4574	0.5798	0.4266	0.4266	0.4283	0.9437	0.6498	0.5278	0.4241	0.4487
	(-11.33)	(-20.47)	(-23.38)	(-11.73)	(-11.73)	(-11.38)	(-23.82)	(-19.42)	(-25.83)	(-12.25)	(-21.10)
One-step residual	0.1928	0.4706	0.6096	0.4817	0.4675	0.4695	1.6523	0.7489	0.5403	0.4736	0.4808
resampling	(-0.21)	(-18.17)	(-19.44)	(-0.33)	(-3.27)	(-2.86)	(33.39)	(-7.13)	(-24.07)	(-2.01)	(-15.46)

Table 3.3: Empirical coverage probabilities calculated using parametric, nonparametric, residual and one-step bootstrap for the ℓ_1-penalized coefficients of the linear (2.1) and the logistic (2.2)DGPs. (Coverage errors of confidence sets in round brackets)

	β_0	β_1	β_2	β_3	β_4	β_5	β_6	β_7	β_8	β_9	β_{10}
Coefficients	0	0.75	2	0	0	0	10	-3	1.5	0	-0.65
Linear model											
Desired nominal coverage	90	90	90	90	90	90	90	90	90	90	90
Vector resampling	88.7	87.0	85.3	93.3	93.5	93.8	84.9	84.4	85.5	93.1	84.0
	(-1.3)	(-3.0)	(-4.7)	(3.3)	(3.5)	(3.8)	(-5.1)	(-5.6)	(-4.5)	(3.1)	(-6.0)
Residual	88.3	84.4	88.5	89.6	89.1	90.8	88.9	88.8	88.5	90.0	81.5
	(-1.7)	(-5.6)	(-1.5)	(0.4)	(-0.9)	(0.8)	(-1.1)	(-1.2)	(-1.5)	(0.0)	(-8.5)
Modified residual	88.2	76.0	90.0	87.3	89.3	88.3	88.7	88.7	88.0	88.4	75.3
	(-1.8)	(-14.0)	(0.0)	(-2.7)	(-0.7)	(-1.7)	(-1.3)	(-1.3)	(-2.0)	(-1.6)	(-14.7)
One-step residual resam.	90.4	88.4	88.6	88.0	86.2	89.4	90.4	89.8	88.4	88.6	86.8
	(0.4)	(-1.6)	(-1.4)	(-2.0)	(-3.8)	(-0.6)	(0.4)	(-0.2)	(-1.6)	(-1.4)	(-3.2)
One-step residual resam.	88.2	90.0	87.2	88.8	89.0	88.6	86.7	90.0	90.2	88.4	88.0
with double bootstrap	(-1.8)	(0.0)	(-2.8)	(-1.2)	(-1.0)	(-1.4)	(-3.3)	(0.0)	(0.2)	(-1.6)	(-2.0)
Logistic model											
Desired nominal coverage	90	90	90	90	90	90	90	90	90	90	90
Vector resampling	88.3	87.6	86.9	95.4	94.7	96.3	91.7	86.9	85.8	95.6	85.3
	(-1.7)	(-2.4)	(-3.1)	(5.4)	(4.7)	(6.3)	(1.7)	(-3.1)	(-4.1)	(5.6)	(-4.7)
Residual resampling	73.8	67.8	68.2	81.5	81.1	80.8	46.0	61.9	71.3	81.9	69.5
	(-16.2)	(-22.2)	(-21.8)	(-9.5)	(-9.9)	(-9.2)	(-44.0)	(-28.1)	(-19.7)	(-9.1)	(-21.5)
One-step residual resam.	86.3	87.7	88.4	87.1	87.1	87.0	97.9	87.5	87.2	85.9	87.6
	(-3.7)	(-2.3)	(-1.6)	(-2.9)	(-2.9)	(-3.0)	(7.9)	(-2.5)	(-2.8)	(-4.1)	(-2.4)
One-step residual resam.	86.5	87.6	87.8	85.4	86.1	86.9	89.6	87.0	85.9	86.7	87.2
with double bootstrap	(-3.5)	(-2.4)	(-2.2)	(-4.6)	(-3.9)	(-3.1)	(-0.4)	(-3.0)	(-4.1)	(-3.3)	(-2.8)

Chapter 4

GWAS

4.1 Introduction

One of the most challenging breakthroughs of the last decade involving high dimensional variable selection, has to deal with humans, genomes, and genetics. The question is what a particular genome can tell us about our backgrounds and the quality of our futures [Khoury and Wacholder (2008)]. Among these studies the most ambitious are genomewide association studies (GWAS) that can be defined as any study of genetic variation accords the entire human genome that is designed to identify genetic associations with observed traits, or presence or absence of a disease or condition. In genome wide association studies, hundreds of thousands of single-nucleotide polymorphisms (SNPs) are tested for association with a disease.
The success of genome-wide association studies in finding susceptibility genes for many common diseases presents large opportunities for epidemiological studies of environmental risk factors.
Advances in GWAS will have great impact in the future to shed light on the mechanism of common diseases. In the past, GWAS identified SNPs implicating hundreds of robustly replicated loci for common traits, despite the identified variants which explained only a small proportion of the heritability of most complex diseases.
Explanations concerning complex diseases could reside in gene-environment (G-E) interactions or more complex pathways involving multiple genes and exposures. Therefore, the evaluation of gene-environment interaction in GWAS is necessary for complex diseases (e.g., diabetes, asthma, cancer), caused by the interaction of multiple genes and environmental factors.
Many methodological challenges are involved in the design and analysis of gene-environment-wide association studies [Thomas (2010)]. The goal of this chapter is to provide a review of fresh findings concerning GWAS, outlying methodological issues in gene-environment-wide interaction studies (GEWI).

4.2 Single nucleotide polymorphisms SNPs

DNA is the well know large spiral-shaped molecule in the nucleus of almost every cell in the human body. DNA is composed of 3 billion pairs of nucleotides A (adenine), C (cytosine), G (guanine), and T (thymine). Taken together, these letters carry the traits inherited form our mothers and fathers.

The human genome contains millions of DNA polymorphism scattered across the different chromosomes. DNA polymorphism - which stems from Greek and means having many shapes - can be defined as a change in the DNA sequence - a polymorphous part of DNA among the individuals of the same population may cause a change in the function of the gene.
This change may cause disease, and may affect how a person reacts to bacteria, viruses, drugs, and other substances. The most common type of DNA polymorphism employed for large-scale linkage mapping is *single nucleotide polymorphism (SNPs)* in which, for instance, the single nucleotide T in one sequence is replaced by a G in the corresponding sequence.

The genome sequences of any two people are 99.9% identical. About one in 1000 nucleotides of human DNA can vary in the form of SNP. SNPs are not considered abnormal; they are simply part of the natural genetic variation within a population that creates diversity, whether the SNPs influence eye color of susceptibility to heart disease. Not all single-base changes in DNA are SNPs. To be classified as a SNP, at least one percent of the general population must have that change. SNPs may lie within a gene, where they can cause alterations in the resulting gene function ranging from profound to no effect [Hartwell et al. (2004)]. Therefore, the result is a change in the amino acid sequence of a protein that may affect the function of the protein. As can be seen in the figure (4.1) the double stranded DNA sequence in the region is identical between these two samples save for one base pair. At that point the pair A/T is C/G in the other sequence. This variation is a SNP.

Theoretically, a SNP could have four possible forms, or alleles, since there are four types of bases in DNA. In reality, most SNPs have only two alleles. Therefore, if some people have a T at a certain place in their genome while everyone else has G, that place in the genome is a SNP with a T allele and a G allele.

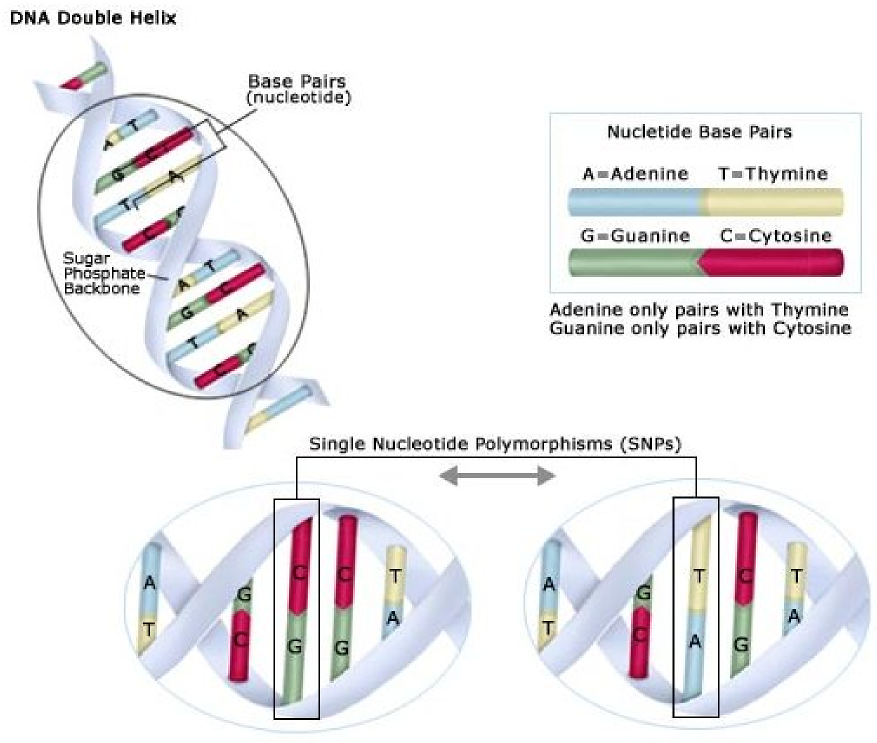

Figure 4.1: Single nucleotide polymorphisms SNPs

4.2.1 Genetic Model

For practical purposes we describe the main characteristics of the variable SNP. The genetic models can be viewed in terms of the mode of inheritance: additive model, recessive model or dominant models.

1. **Additive Model**:

   One SNP carries three genotypes AA, AB, and BB. Assuming that B is the risk allele and A is the common allele, an additive model will imply a proportional increase of risk of disease per copy of allele B. That is, the risk of disease for a subject carrying BB is double compared to an AB carrier. In cases like this, the SNP is often treated as a quantitative variable indicating the count of B alleles and association between the SNP and the phenotype is carried out by logistic regression model with the estimation of a single coefficient (Odds Radio) for the SNP.

2. **Recessive Model**:

Under the recessive model, the risk category is identified by the BB genotype (ie: homozygousity of the risk allele) and subjects with genotypes AA or AB are not at risk of disease. The relationship between the number of copies of the allele B and the risk of disease is not linear as the OR of AB vs. AA is 1 and the OR of BB vs. AA is > 1.

3. **Dominant Model**:

   Under the dominant model, the genetic risk category includes all carries of at least one copy of the allele B. Also, in this case there is a non-linear relationship: in fact, the OR of BB vs AA is the same as the OR of AB vs. AA.

Other models are possible but less common. In a codominant model, homozygous individuals (AA or BB) have a higher risk of disease compared to heterozygous individuals (AB). The excess of heterozygosity model implies that heterozygous individuals have a higher risk of disease compared to homozygous.

Given that the additive model was proven to be very robust in terms of detecting true associations even when the underlying genetic model is additive or recessive, GWAS are usually performed assuming an additive effect. This choice reduces the computational burden and simplifies the estimation of association statistics as well as the control for multiple testing. A better assessment of the correct genetic model is done a posteriori on a reduced set of SNPs that have been detected to be associated with the trait.

There is another very rare combination: the subject at risk carries the AB genotypes and the subjects carrying AA and BB are not at risk or vice versa.

For the sake of simplicity, performing GWA studies the additive model is often used, there is evidence that this model is a good approximation of all these different scenarios.

Moreover, studying 300,000 snips is not even doable assuming a different genetic model for each SNP. Otherwise, when a previous screen is done, and the candidate SNPs are few, it is reasonable to perform the models that are able to explain the data and provide more precise answers.

4.3 Genome-wide Association Studies

We are finally in the era where the genomewide association studies are becoming possible and we are dealing with the new frontiers in our understanding and treatment of diseases. The execution and analysis of such studies will require great care. In GWAS, hundreds of thousands of single-nucleotide polymorphisms (SNPs) are tested for association with a disease. To be more precise a genetic variant is genotyped in a population for which phenotypic information is available (such as disease occurrence, or a range of different trait values). If a correlation is observed between genotype and phenotype, there is said to be an association between the variant and the disease or trait.

GWAS is a step ahead of ‘candidate gene’ studies, in which sample size is generally smaller and the variants assayed are limited to a selected few, often on the basis of an imperfect understanding of the biological pathways and often yielding associations that are difficult to replicate [Manolio (2010)]. Nevertheless GWAS are a step beyond family-based linkage studies, in which inheritance patterns are related to several hundreds to thousands of genomic markers.

The success of genome-wide association studies in finding susceptibility genes for many common diseases presents large opportunities for epidemiological studies of environmental risk factors.

Advances in GWAS will have great impact in the future to shedding light on the mechanism of common diseases. In the past, GWAS have identified SNPs implicating hundreds of robustly replicated loci for common traits, though these identified variants explain only a small proportion of the heritability of most complex diseases.

A genomewide association study is typically based on a case-control design in which SNPs across the human genome are genotyped. In addition to case-control, scans of a person’s DNA can also be used in other study designs, as cohort studies and clinical trials. The most important challenge in GWAS is the sheer scale of tests of association required, at least one per SNP. Thus far, statistical analyses of GWAS data have focused mainly on the simplest tests of one SNP at a time, leaving open the possibility that more sophisticated analyses may reveal more important results.
At this point, advances in statistical methodology are crucial. A GWAS special issue published in Statistical Science 2009 provides an interesting overview of the state of the art and helps to foster these new methodological challenges.

The most important methodological issues in GWAS concern:

- the roles of different study designs [Laird and Lange 2009];
- multiple testing, significance level and weighted hypothesis testing [Roeder and Wasserman 2009];
- methodological issues in multistage designs [Thomas et al. 2010];
- bayesian methods for detecting and characterizing allelic heterogeneity and boosting signals [Su et al. 2009];
- confounding problems [Astle and Balding (2009)];
- strategies to harness gene-gene (G-G) and gene-environment (G-E) interactions [Kooperberg at al. 2009];

- new methods for case-control data involving the use of "retrospective" likelihood to improve the power by adding model assumptions [Chatterjee at al. (2009)];
- robust test scans under incomplete linkage disequilibrium, robust test under complete linkage disequilibrium often have greater power than Pearson's chi-square test [Zheng at al. (2009)];
- new approaches to fit effect estimation and prediction from genome-wide marker data [Goddard et al. 2009];
- using the data from GWAS not only to identify SNPs but also to study other types of heritable variations as copy number variants, using other types of heritable variation may contribute to the understanding of common, complex disease [Zöllner and Teslovich 2009];
- the detection of association of disease with SNPs can be difficult using the data from a single study, combining data from several case-control GWAS procedures to combine data from several case-control GWAS [Pfeiffer at al. 2009]
- replication in GWAS is a very important issue in genotype-phenotype association studies in terms of credibility[Kraft et al. 2009].

4.3.1 Findings

Up to the present (October 6th, 2010), 904 SNP-trait associations are reported as significant ($P < 5 * 10^{-8}$) for 165 traits [Hindorff et al. (2009)] , nevertheless GWAS have provided valuable insights into genetic architecture.
An interactive catalog of published GWAS was developed for use in investigating genomic characteristics of trait/disease-associated SNPs (TASs) and is available at http://www.genome.gov. Among the treat-associated SNPs located in coding regions, there are 40% located in noncoding introns (defined as the portions of a gene that are removed (spliced out) before translation of a protein, introns may contain regulatory information that is crucial for appropriate gene expression) and another 40% fall in intergenic regions (defined as segments of DNA that do not contain or overlap genes). Therefore, these findings have sharpened the focus on the potential roles of intronic and intergenic regions in regulating gene expression. Consequently, much more remains to be learned about how variations in intronic and intergenic regions (where the great majority of SNP-traits lie) influence gene expression, protein coding, and disease phenotypes.
The long-term investment in such novel and exciting research promises not only to advance human biology but also to revolutionize the practice of modern medicine.

[Manolio (2010)] reports a number of GWAS which have proved successful in identifying genetic associations with complex treats.

4.4 Gene-environment-wide association studies

Even if GWAS have proved successful in identifying genetic associations with complex treats, the identified variants are not enough to explain the heritability of most complex diseases. The gene-environment (G-E) interaction can be defined as a joint effect of one or more genes with one or more environmental factors that cannot be readily explained by their separate marginal effect. The small proportion of the explained heritability can be improved taking into account potential G-E interactions of more complex pathways involving multiple genes and exposures.
There are a number of different epidemiological designs that are available to the scientist to explore the potential interactions among SNPs and environmental exposures. In the next section, we report the most important epidemiological designs explored by [Thomas (2010)].

4.5 Epidemiological designs

4.5.1 Study designs for gene-environment interactions

[Thomas (2010)] reviews the emerging approaches in GEWI studies. The author focuses his review on gene-environment interaction defined as a joint effect of one or more genes with one or more environmental factors that cannot be readily expanded by their separate marginal effects. The challenges to GxE studies, in addition to the usual challenges for genetic association studies [Manolio (2010)], are exposure assessment, sample size and heterogeneity.

4.5.2 Basic epidemiologic designs

Considerable methodological challenges are involved in the design and analysis of gene-environment interaction studies. The standard epidemiological designs for studying the main effects of genes or environmental factors can be applied to the study of GxE interactions. Table 4.1 shows the basic epidemiological designs for cohort, case-control, case-only, randomized trial and crossover trials. Among these designs, case-only (or sometimes called 'case-case' design) can be used for testing interactions and not main effects due to the fact that the design relies on an assumption of gene-environment independence on the cases.

Scientists often use this design to study GxE interactions, because it is a more appropriate approach, in terms of controlling the trade-off be-

Table 4.1: Basic epidemiologic designs.

Design	Approach	Advantages	Disadvantages	Settings
Cohort	Comparison of incidence of new cases across groups defined by E and G	Freedom from most biases; clear temporal sequences of cause and effect	Large cohorts and/or long follow-up needed to obtain sufficient numbers of cases; possible biased losses to follow-up; changes in exposure may require recurring observation	Common Ds or multiple end points; commonly used in biobanks
Case-control	Comparison of prevalence of E and G between cases and controls	Modest sample sizes needed for rare Ds; can individually match on confounders	Recall bias for E; selection bias, particularly for control group	Rare Ds with common E and G risk factors
Case-only	Test of G-E association among cases, assuming G-E independence in the source population	Greater power than case-control or cohort	Bias if G-E assumption is incorrect	G-E studies in which G-E independence can be assumed
Randomized trial	Cohort study with random assignment of E across individuals	Experimental control of confounders	Prevention trials for D incidence can require very large sample sizes	Experimental confirmation for chronic effects
Crossover trial	Exposes each individual to the different Es in random order	Experimental control of confounders; within-individual comparisons	Small sample sizes; only low doses possible if E is potentially harmful	Experimental confirmation for acute effects

tween bias and efficiency. these have been developed as empirical Bayes ([Mukherjee et al. (2008)]) or Bayes model averaging ([Li and Conti (2009)]).

4.5.3 Hybrid designs

Among hybrid-designs, the novel and interesting ones are *two-phase case-control* and *counter-matching designs* (Table 4.2).
The main purpose is to improve the power for detecting either main effects or interactions using different ways of selecting controls. Two-phase case-control can be a good choice when a surrogate variable of the exposure is easily available. This design involves independent subsampling on the basis of disease status and the exposure surrogate variable from a first-phase case-control or cohort study. The data from the two phases are combined in the

analysis.

Counter-matching is the matched variant of the two-phase design.

Table 4.2: Hybrid designs.

Design	Approach	Advantages	Disadvantages	Settings
Nested case-control	Selection of matched controls for each case from cohort members who are still disease free	The freedom from bias of a cohort design combined with the efficiency of a case-control design; simple analysis	Each case group requires a separate control series	Studies within cohorts requiring additional data collection
Case-cohort	Unmatched comparison of cases from a cohort with a random sample of the cohort	Same advantages as nested case-control; the same control group can be used for multiple case series	Complex analysis	Studies within cohorts with stored baseline biospecimens
Two-phase case-control	Stratified sampling on D, E and G for additional measurements (for example, biomarkers)	High statistical efficiency for subsample measurements	Complex analysis	Substudies for which outcome and predictor data are already available
Counter matching	Matched selection of controls who are discordant for a surrogate for E	Permits individual matching; highly efficient for E main effect and G-E interactions	Complex control selection	Substudies in which a matched design is needed
Joint case-only and case-control	Bayesian compromise between case-only and case-control comparison	Power advantage of case-only combined with robustness of case-control	Some bias when G-E association is moderate	G-E studies for which G-E independence is uncertain

4.5.4 Family-based designs

The apparently homogeneous population can hide different subgroups of individuals with different ancestral origins and different allele frequencies at many loci.

The family-based designs (table 4.3) seem to be more powerful for testing GxE interactions if relatives' exposures are not highly correlated because they avoid bias from population stratification, even though these designs are generally less powerful in detecting main effects compared to case-control studies.

Table 4.3: Family-based designs and GWA designs.

Design	Approach	Advantages	Disadvantages	Settings
Family-based designs				
Case-sibling (or-cousin)	Case-control comparison of E and G using unaffected relatives as controls	More powerful than case-control for G-E; immune to population stratification bias	Discordant sibships difficult to enroll; overmatching for G main effects	Populations with potential substructure
Case-parent triad	Comparison of Gs for cases with Gs that could have been inherited from parents, stratified by case's E	More powerful than case-control for G-E; immune to population stratification bias for G main effects	Difficult to enroll complete triads; possible bias in G-E if G and E are associated within parental mating types	Substructured populations, particularly for Ds of childhood
Twin studies	Comparison of D concordance between monozygotic (MZ) and dizygotic (DZ) pairs in different environments	No genetic data required; can be extended to include half-siblings, twins reared together or apart, or to compare discordant pairs on measured G and E	Used mainly to identify interactions with unmeasured genes; assumption of similar E between MZ and DZ pairs	Exploratory studies of potential for G-E before specific genes have been identified
GWA designs				
Two-stage genotyping	Use of high-density panel on part of a case-control sample to select a subset of SNPs with suggestive Gs or G-E interaction for testing; the SNPs are tested using a custom panel in an independent sample, with joint analysis of both samples	Highly cost efficient	Only part of sample has GWA genotypes	GWA studies for which complete SNP data on all subjects is not needed
Two-step interaction analysis	Preliminary filtering of a GWA scan for G-E association in combined case-control sample, followed by G-E testing of a selected subset	Much more powerful for G-E or G-G interactions than a single-step analysis	Can miss some interactions	GWA studies with complete SNP data and focus on G-E
DNA pooling	Comparison of allelic density in pools of cases and controls stratified by E, followed by individual genotyping	Highly cost efficient	Technical difficulties in forming pools and assaying allelic density; limited possibilities for testing interactions	GWA studies for which an initial scan is severely limited by cost

4.6 The statistical analysis of GWAS, a formidable challenge

Genetic analysis of complex diseases needs novel statistical methods to investigate data collected on thousands of variables by genome-wide association studies. The complexity of such analyses increase dramatically when one has to consider interaction effects, either among the genetic variations (gene-gene interactions) or with environment risk factors (gene-environment interactions).

Current statistical methods for GWAS largely rely on marginal information from genes studied one at time and ignore potentially valuable information as the interaction of multiple loci. Each responsible gene may have a small marginal effect in causing the disease and so this weak effect is difficult to detect.

In recent years, new and promising approaches for variable selection in GWAS have been proposed. Penalized regression represents one of the most attractive methods and the number of applications of this technique on association mapping of disease genes is rapidly growing. Of particular interest, is the class of data mining techniques. Below, we briefly consider two of these approaches, Random Forests (RFs) and Bayesian Networks.

4.7 The SIS method

Sure Independence Screening (SIS) is a two-stage procedure for ultrahigh dimensional feature selection first introduced by [Fan and Lv (2008)] and then developed by [Fan at al. (2009b)].

This approach consists of a screening stage and a selection stage. In the screening stage, main-effects are crudely screened by using marginal utilities. In the selection stage, variable selection and parameter estimation are carried out simultaneously using a penalized regression with SCAD penalty.

More formally, the SIS algorithm has the following steps:

1. Let $\{(x_i, y_i), i = 1, \ldots, n\}$ be a data set with sample size n, with $x_i \in \mathbb{R}^p$. For each covariate X_j, $j = 1, \ldots, p$, calculate the marginal utility
$$L_j = \min_{b_0, b_j} \frac{1}{n} \sum L(y_i, b_0 + x_{ij} b_j), \tag{4.1}$$
where $L(\cdot,\cdot)$ is a generic loss function. In other words, fit p bivariate models (e.g. GLMs) and calculate the p marginal utilities.

2. Rank the variables according to these marginal utilities. The smaller L_j the more important the covariate.

3. Select the first d features. Typically $d = \lfloor n/\log n \rfloor$, where $\lfloor \cdot \rfloor$ is the floor function. Call $\widehat{A}$ this subset of prescreened covariates.

4. Estimate the model coefficients of the penalized regression by

$$(\hat{\beta}_0, \hat{\beta}) = \underset{(b_0, b) \in \mathbb{R}^{d+1}}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} L(y_i, b_0 + x_{i,\widehat{A}} b_{\widehat{A}}) + \sum_{j \in \widehat{A}} p_\lambda(|b_j|),$$

   where $x_{i,\widehat{A}} \in \mathbb{R}^d$ is the subvector obtained from $x_i \in \mathbb{R}^p$ using the $d \ll p$ prescreened variables of $\widehat{A}$. The penalty $p_\lambda(|b_j|)$ is the SCAD penalty.

For a linear model, we can use an L_2 loss function

$$L(y_i, \beta_0 + x_i\beta) = (y_i - \beta_0 - x_i\beta)^2$$

or an L_1 loss function (robust regression)

$$L(y_i, \beta_0 + x_i\beta) = |y_i - \beta_0 - x_i\beta|.$$

For a logistic regression, the loss function can be defined by

$$L(y_i, \beta_0 + x_i\beta) = \sum_{i=1}^{n} \left[\log(1 + e^{\beta_0 + x_i\beta}) - y_i(\beta_0 + x_i\beta) \right].$$

When d is large enough, [Fan and Lv (2008)] showed that the first screening stage of the above algorithm has a high probability of selecting all of the informative covariates. For this reason, the method is called Sure Independence Screening.

In the second stage, the SCAD penalized regression performs a further variable selection and estimates the main effects for the remaining variables.

An important drawback of SIS is that if a variable is discarded in the first stage, it is not possible to select it in the second stage. Therefore, the SIS methodology may break down if a covariate is marginally unrelated, but jointly related with response, or if a covariate is jointly uncorrelated with the response but has higher marginal correlation with the response than some other informative variables.

[Fan at al. (2009b)] proposed two interesting variants of SIS that have attractive theoretical properties in terms of reducing the false selection rate.

The steps of the first variant of the SIS algorithm are:

1. Split the sample $\{(x_i, y_i), i = 1, \ldots, n\}$ into two halves at random.

2. For each partition and for each covariate X_j, $j = 1, \ldots, p$, calculate the marginal utility

$$L_j = \min_{b_0, b_j} \frac{1}{n} \sum L(y_i, b_0 + x_{ij} b_j), \tag{4.2}$$

   rank the variables according to the L_j values and select the first $d = \lfloor n / \log n \rfloor$ features. Call $\widehat{A_1}$ and $\widehat{A_2}$ the subset of prescreened covariates for the two partitions ($\#\widehat{A_1} = \#\widehat{A_2} = d$).

3. Find the intersection $\widehat{A} = \widehat{A_1} \cap \widehat{A_2}$.

4. Estimate the model coefficients of the penalized regression using the prescreened variables $\widehat{A}$

$$(\hat{\beta}_0, \hat{\beta}) = \underset{(b_0,\, b) \in \mathbb{R}^{d+1}}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} L(y_i, b_0 + x_{i,\widehat{A}} b_{\widehat{A}}) + \sum_{j \in \widehat{A}} p_\lambda(|b_j|).$$

The second variant of the SIS algorithm can be summarized as follows.

1. Split the sample $\{(x_i, y_i), i = 1, \dots, n\}$ into two halves at random.

2. For each partition and for each covariate X_j, $j = 1, \dots, p$, calculate the marginal utility

$$L_j = \min_{b_0, b_j} \frac{1}{n} \sum L(y_i, b_0 + x_{ij} b_j), \tag{4.3}$$

and rank the variables according to the L_j values.

3. Select in the two partitions a number of covariates such that the intersection $\widehat{A} = \widehat{A_1} \cap \widehat{A_2}$ has $d = \lfloor n / \log n \rfloor$ elements (i.e. $\# \widehat{A} = d$). $\widehat{A_1}$ and $\widehat{A_2}$ are the subset of prescreened covariates for the two partitions.

4. Estimate the model coefficients of the penalized regression using the prescreened variables $\widehat{A}$

$$(\hat{\beta}_0, \hat{\beta}) = \underset{(b_0,\, b) \in \mathbb{R}^{d+1}}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} L(y_i, b_0 + x_{i,\widehat{A}} b_{\widehat{A}}) + \sum_{j \in \widehat{A}} p_\lambda(|b_j|).$$

4.8 The ISIS method

The Iterative SIS (ISIS) method proposed by [Fan at al. (2009b)] seeks to overcome the difficulties of SIS by iteratively adding and deleting covariates, but maintaining the efficiency and stability of the SIS method.

The steps of the ISIS algorithm are:

1. Apply SIS to the (x_i, y_i) data set. Let $\widehat{M_1}$ the subset of k_1 selected covariates.

2. For each variable, calculate the second-step marginal utilities

$$L_j^{(2)} = \min_{b_0,\, b_{\widehat{M}_1},\, b_j} \frac{1}{n} \sum_{i=1}^{n} L(y_i, b_0 + x_{i,\widehat{M}_1} b_{\widehat{M}_1} + x_{ij} b_j)$$

where $x_{i,\widehat{M}_1}$ is the subvector of x_i consisting of those elements in M_1.

3. Order the $(p-k_1)$ values $L_j^{(2)}$ and select the covariates corresponding to the smallest k_2 elements. Call $\widehat{A}_2$ this prescreening subset.

4. Use penalized regression to obtain the model coefficients:

$$\hat{\beta}_2 = \underset{(b_0, b_{\widehat{M}_1}, b_{\widehat{A}_2})}{\operatorname{argmin}} \frac{1}{n}\sum_{i=1}^{n} L(y_i, b_0 + x_{i,\widehat{M}_1} b_{\widehat{M}_1} + x_{i,\widehat{A}_2} b_{\widehat{A}_2}) + \sum_{j \in \widehat{M}_1 \cup \widehat{A}_2} p_\lambda(|b_j|).$$

   The values of $\hat{\beta}_2$ that are non-zero yield a new subset M_2 of selected variables.

5. Repeat steps 3 and 4 until the set $\widehat{M_k}$ has stabilized (i.e. $\widehat{M_k} = \widehat{M_{k-1}}$) or has reached the prescribed size d.

The choice of the k_r values is critical. Large values decrease the computation cost and the probability that ISIS will terminate prematurely but in the meantime they make ISIS more similar to SIS (with the drawbacks described in subsection 4.7). [Fan at al. (2009b)] suggest choosing $d = \lfloor n/\log n \rfloor$, $k_1 = \lfloor 2/3 \cdot d \rfloor$ and $k_r = d - \#\widehat{M_k}$, where $\#\widehat{M_k}$ is the cardinality of the set $\widehat{M_k}$.

The two variants of SIS proposed by [Fan at al. (2009b)] can be easily extended to ISIS and can improve its performance. In many simulation studies the second variants of ISIS shows superior performance (in terms of percentage of selected informative variables, false selection rate and prediction error) compared to the other algorithms.

The `SIS` package of R developed by [Fan et al. (2010)] implements the iterative sure independence screening with functions `GLMvanISISscad`, `GLMvanISISscad`, `COXvanISISscad`, `COXvarISISscad` for different variants of SIS and ISIS.

Now we test the ISIS approach on a DGP characterized by a binary outcome Y with a Bernoulli distribution whose probabilities $p_i = P(Y_i = 1|X_{ij} = x_{ij}, j = 1, 2, \dots, p)$ are defined by:

$$p_i = \mu + \sum_{j=1}^{p} \beta_j x_{ij} \tag{4.4}$$

where $i = 1, 2, \dots, n$, $x_{ij} \in \{0, 1\}$ is the observed value for the jth binary variable of the ith sample units, and μ, β_j are model coefficients $(j = 1, 2, \dots, p)$.

The parameters of the simulated dataset are:

- sample size $n = 1000$ and number of covariates $p = 400$;
- $\beta_j \neq 0$ for $j \in \mathcal{J}$, $\beta_j = 0$ for $j \notin \mathcal{J}$, where $\mathcal{J} = \{j_1, \dots, j_{10}\}$ is a set of randomly selected indexes with $1 \leq j_k \leq p$. In addition, $\sum_{j=1}^{p} |\beta_j| < 1$. Values of β_j for $j \in \mathcal{J}$ are

$$\beta = (-0.1, -0.1, -0.1, -0.1, -0.1, -0.1,\ 0.1,\ 0.1,\ 0.1,\ 0.09);$$

- $\mu = \frac{1}{2}\left(1 - \sum_{j=1}^{p} \beta_j\right) = 0.605$.

Binary variables x_j are generated as follows:

- generate n random binary realizations of the first covariate X_{i1} according to a Bernoulli distribution with probability $= 0.5$;
- for $j = 2, 3, \dots, p$:

$$X_{ij} = \begin{cases} X_{i,j-1} & \text{with probability } \rho \\ 0 & \text{with probability } (1-\rho)/2 \\ 1 & \text{with probability } (1-\rho)/2; \end{cases} \tag{4.5}$$

  with $\rho = 0.80$.

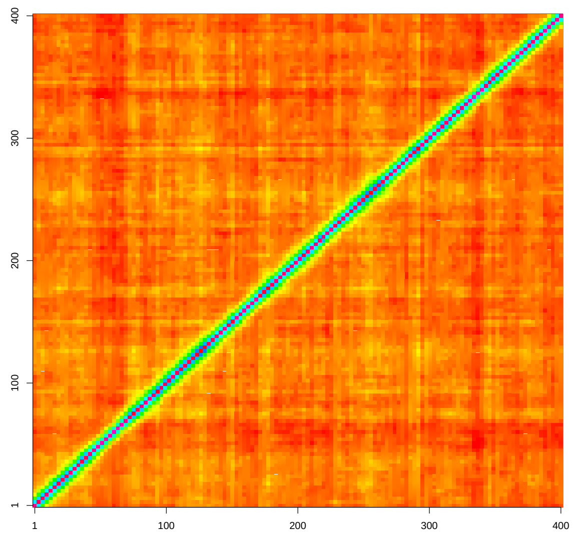

Figure 4.2: Correlation matrix for the DGP of equation (4.4).

This DGP is interesting because it is characterized by binary covariates that are strongly associated to their "neighbors". In fact:

$$\rho_V(X_i, X_{i+1}) \approx 0.80, \quad \rho_V(X_i, X_{i+2}) \approx 0.64, \quad \rho_V(X_i, X_{i+3}) \approx 0.51,$$
$$\rho_V(X_i, X_{i+4}) \approx 0.41, \quad \rho_V(X_i, X_{i+5}) \approx 0.33, \quad \rho_V(X_i, X_{i+6}) \approx 0.26,$$

where ρ_V is the estimated Cramer's V. The complete association structure of the $p = 400$ covariates is shown in figure 4.2.

In addition, it is worth to noting that this DGP is not a logistic model because probabilities p_i are linearly correlated to informative covariates.

Therefore, when using available R packages for penalized logistic regression, we will be able to test the performance of ISIS under model misspecification.

The R code for generating a sample from the above DGP is

```
# Set DGP parameters
set.seed(987654)
n <- 1000
p <- 400
rho <- 0.8
K <- 100
betas <- c(rep(-0.1,6), rep(0.1,3), 0.09)
mu <- (1-sum(betas))/2

# Generate associated covariates X
X <- matrix(0,n,p)
X[,1] <- rbinom(n, 1, 0.5)
for (j in 2:p) {
   for (i in 1:n) {
       rnd <- runif(1)
       if (rnd < rho) {
          X[i,j] <- X[i,j-1]
       }
       else if (rnd > (1+rho)/2) {
          X[i,j] <- 1
       }
   }
}

# Choose informative covariates
Inf.Vars <- t(order(runif(p))[1:10])
# Calculate probabilities
pi <- mu + X[,Inf.Vars]%*%betas
# Generate outcome Y
y <- rbinom(n,1,pi)
```

We use the `SIS` package and apply the second variant of the ISIS approach to our sample. The option `vartype` permits to specifying the variant of SIS (or ISIS) to use; `vartype=0` no variants, `vartype=1` first type, `vartype=2` second type.

```
require(SIS)
out <- SIS(data=list(x=X, y=y),
    model='glm', family=binomial(link="logit"),
    vartype=2, nsis=NULL, rank.method='obj', inittype='NoPen',
    tune.method='BIC', folds=NULL, post.tune.method='CV',
    post.tune.folds=NULL, DOISIS=TRUE, ISIStypeCumulative=FALSE,
    maxloop=5, xtune=NULL, ytune=NULL, detail=TRUE)
```

Now we calculate the confusion matrix and the prediction error, comparing the true outcome `y` and the predicted outcome `yhat`, obtained setting to 0.5 the threshold on the predicted probabilities `phat`.

```
########################
# Prediction errors
pred.beta <- out$ISIScoef$SCADcoef
Xb <- pred.beta[1]+X%*%pred.beta[-1]
phat <- 1/(1+exp(-Xb))
yhat <- phat>0.5
# Confusion matrix
table(y,yhat)
require(ROCR)
pred <- prediction(phat, y)
auroc <- performance(pred,"auc")@y.values[[1]]
```

```
   yhat
y   FALSE TRUE
  0   334  167
  1   183  316

auroc = 0.7048228
```

Below we compare the covariates selected by ISIS with the informative/uninformative variables.

```
#########################
# Selection performance
True.Inf.Uninf <- rep(0,p)
True.Inf.Uninf[Inf.Vars] <- 1

Pred.Inf <- out$SISresult$ISISind
modelsize <- length(Pred.Inf)
Pred.Inf.Uninf <- rep(0,p)
Pred.Inf.Uninf[Pred.Inf] <- 1

conmtx <- table(Pred.Inf.Uninf,True.Inf.Uninf)
if (all(Pred.Inf.Uninf==0)) {
conmtx <- rbind(conmtx,c(0,0))
}
conmtx
```

```
             True.Inf.Uninf
Pred.Inf.Uninf   0   1
             0 388   2
             1   2   8
```

Eight informative variables (80%) have been correctly selected and 388 uninformative variables (97%) have been correctly discarded.

```
####################
# Accuracy measures
kI <- length(betas)
Ne <- p-kI
Po <- kI
TNe <- conmtx[1,1]
TPo <- conmtx[2,2]
FNe <- conmtx[1,2]
FPo <- conmtx[2,1]

# Accuracy. P(Yhat = Y). Estimated as: (TP+TN)/(P+N)
acc <- (TPo+TNe)/(Po+Ne)
# Error rate. P(Yhat != Y). Estimated as: (FP+FN)/(P+N)
err <- (FPo+FNe)/(Po+Ne)
# True positive rate. P(Yhat = + | Y = +). Estimated as: TP/P
# Sensitivity. Same as tpr
sens <- TPo/Po
# True negative rate. P(Yhat = - | Y = -). Estimated as: TN/N
# Specificity. Same as tnr
spec <- TNe/Ne
# Positive predictive value. P(Y = + | Yhat = +). Estimated as: TP/(TP+FP).
ppv <- TPo/(TPo+FPo)
# Negative predictive value. P(Y = - | Yhat = -). Estimated as: TN/(TN+FN).
npv <- TNe/(TNe+FNe)

#####################
# Estimation errors
true.beta <- rep(0,p)
true.beta[Inf.Vars] <- c(rep(-0.1,6), rep(0.1,3), 0.09)
est.err.l1 <- sum(abs(true.beta - pred.beta[-1]))
est.err.l2 <- sqrt(sum((true.beta - pred.beta[-1])^2))
cbind(betas, pred.beta[-1][Inf.Vars])

\newpage
           performance measure
```

```
modelsize 10.0000000
accuracy   0.9900000
err        0.0100000
sens       0.8000000
spec       0.9948718
ppv        0.8000000
npv        0.9948718
auroc      0.7048228
esterrl1   4.1630736
esterrl2   1.3575159
```

The estimation error has been calculated using the sum of absolute errors `esterrl1`$=\sum_j |\hat{\beta}_j - \beta_j|$ and the square root of the sum of squared errors `esterrl2`$=\sqrt{\sum_j (\hat{\beta}_j - \beta_j)^2}$.

4.9 Application of penalized regression to GWAS

The LASSO-Patternsearch algorithm of [Shi et al. (2007)] is a multi-step algorithm with a LASSO-type penalized likelihood method at its core. It is specifically designed to detect and model interactions between predictor variables (SNPs and other covariates) in GWAS. The first stage of the algorithm is a prescreening step where the significance of SNPs is tested evaluating the main effects by a logistic regression model. When one SNP passes the test, the algorithm evaluates its interactions with all of the other SNPs and covariates. The SNPs and the interaction terms that survive the first screening step are introduced in a ℓ_1-penalized regression model that performs a variable selection. The penalty parameter is estimated by cross-validation. In the final step, the algorithm puts the selected terms into a parametric logistic regression and tests the significance of each term at level α. Simulation studies show that the LASSO-Patternsearch method has a good ability to identify important SNPs and covariates, and to separate cases from controls. Hence, it provides a useful tool for the analysis of genetic data.

The work of [Wu et al. (2009)] evaluates the performance of ℓ_1-penalized logistic regression in case-control disease gene mapping with a large number of SNPs. The procedure performs an initial prescreening by a score criterion and then performs a lasso identification and quantification of interactions among previously selected features. Their simulation studies demonstrate that the lasso penalized regression is easily capable of identifying informative predictors and the computational speed is remarkable. Another interesting finding concerns interaction effects that can be found readily in the case of orthogonal designs. The software for performing this technique is available at the UCLA Human Genetics web site.

[Zhou et al. (2010)] extended the work of [Wu et al. (2009)] on lasso penalized logistic regression in GWAS to the field of rare genetic variants. The proposed algorithm improves both false positive and false negative rates and the results show that mixed group and lasso penalties outperform lasso penalty alone, especially when common and rare variants are present simultaneously. This statistical tool is a part of the Mendel statistical genetics package.

4.10 Methods of statistical machine learning for GWAS

As pointed out by [Szymczak et al. 2009], machine learning approaches are promising complements to standard single-and multi-SNP analysis methods for understanding the overall genetic architecture of complex human diseases. However, at present, most of them are not optimized for genome-wide SNP data. Improved implementations and new variable selection procedures are therefore required.

In this section we briefly overview three variable selection methods originally developed in the field of machine learning and then generalized and adapted to ultrahigh dimensional selection problems.

4.10.1 Random Forest

Random Forests [Breiman (2001)] (RFs) for ultrahigh dimensional data sets has been recently proposed by [Schwarz, König and Ziegler (2010)]. RFs are ensemble learners based on Classification and Regression Trees (CARTs) which can cope with 'small n large p' problems, complex interactions and highly correlated predictor variables. CARTs fit data by creating recursive binary partitions of the sample space. Hence, they are naturally suited for investigating interactions. In addition, using RFs, it is possible to calculate the variable importance of each covariate, taking into account not only the direct effect of the covariate on outcome but also the contributions of its interactions with all the other covariates (see [Sohns et al. (2009)], [Sandri and Zuccolotto (2010)]). As a result, variable importances offer a valuable tool for ranking and selecting informative/predictive variables in complex problems.

The RF approach is becoming popular in the GWAS literature because it has several characteristics that make it ideal for these data sets. RFs are appealing because:

- ◇ they can be used when there are many more variables than observations;

- ◇ they have good predictive performance even when there are many uninformative variables;
- ◇ they do not overfit;
- ◇ they can handle a mixture of categorical and continuous predictors;
- ◇ they incorporate interactions among predictor variables;
- ◇ they return measures of variable (gene) importance.
- ◇ they there is little need to fine-tune parameters to achieve excellent performance; the most important parameter to choose is `mtry`, the number of input variables tried at each split. It has been reported that the default value is often a good choice; in addition, the user needs to decide how many trees to grow for each forest (`ntree`) as well as the minimum size of the terminal nodes (`nodesize`);
- ◇ there are high quality and free software implementations of RFs; a popular package for the statistical software R is `randomForest` developed by [Liaw and Winter (2002)]; the free open source software package called Random Jungle (RJ) proposed in [Schwarz et al. (2010)], is devoted to facilitate the rapid analysis of GWAS data. RJ have an impressive computational efficiency, and memory management of RJ allows to analyzing high-dimensional data in an acceptable amount of time. Concerning the problem of evaluation of potential interactions between genes and genes and environment, RFs may help to identify the interactions that cannot be found using traditional statistical approaches.

4.10.2 Bayesian Networks

Bayesian Networks (BNs) are probabilistic graphical models representing a joint probability distribution over a set of random variables $X_1, X_2, \ldots, X_n$ by a Directed Acyclic Graph (DAG). BNs are typically used to learn the structure of a set of random variables that reflects relationships of dependence and conditional independence among them. In a BNs, variables are represented as vertices (nodes) and dependencies as arcs (or edges) between the variable nodes. The directions of edges indicate the directions of dependencies.

The elements that compose a DAG G are two; a finite set V of nodes and a finite set E of directed edges (arrows) between the nodes. Hence, $G = (V, E)$. The DAG defines the structure of the Bayesian network. To each node $v \in V$ in the graph corresponds a random variable X_v. The set of variables associated with the graph G is then $X = (X_v)_{v \in V}$. To each node v with parent nodes $\mathrm{pa}(v)$ a local probability distribution, $p(x_v|x_{\mathrm{pa}(v)})$, is attached. The set of local probability distributions for all variables in the network is $\mathcal{P}$.

A Bayesian Network for a set of random variables X is the pair $(G, \mathcal{P})$. That is, a BN is a DAG with the set of local probability distributions $\mathcal{P}$ defined on it.

Given three random variables X, Y and Z, X and Y are said conditionally independent given Z if $p(x|y,z) = p(x|z)$, whenever $p(y,z) > 0$. In other words, learning the value of Y does not provide additional information about X, once we know Z.

The lack of directed edges in G encodes conditional independencies between the random variables X through the factorization of the joint probability distribution,

$$p(X_1, \ldots, X_n) = \prod_{i=1}^{N} p(x_v | x_{\mathrm{pa}(v)}) \tag{4.6}$$

Bayesian networks, as Random Forests, are a statistical tool with good capabilities in handling the complexity of GWAS in different scenarios and at different levels. They seem capable of capturing biologically meaningful interactions among a group of factors involved in a complex manner in common diseases.

[Jiang et al. (2010)] develop and evaluate a multi-locus method for detecting genetic interactions based on Bayesian Networks and the minimum description length (MDL) principle. The method is called Bayesian network minimum bit length (BNMBL). The experimental results of the authors indicate that BNMBL has significantly greater power and is substantially faster computationally than the multifactor dimensionality reduction (MDR) of [Ritchie et al. 2001].

[Yang and Gu (2009)] evaluate the ability of RFs and BNs to analyze GWAS data sets and show the crucial importance of prescreening. In fact, the inclusion of too many uninformative (noisy) SNPs in the analyses may seriously affect the performance of the two methods. In particular, the performance of BN analysis markedly deteriorates as more noisy SNPs were included in the analysis.

4.10.3 Gene-based analysis

[Lo et al. (2008)] shows the advantage to carry out a gene-based analysis by treating each gene as a basic unit while incorporating relevant information form all the SNPs within that gene. This technique provide a novel tool to evaluate pairwise and third-order interactions. The method proposed by [Lo et al. (2008)] is a 'gene-based approach' in which information on SNPs is combined into one unit for that gene. The effect of a certain region or a certain gene, is expressed as the average effect of all individual effects due to all of the SNPs within that gene. Analogously, a two-way interaction of two genes is calculated by the average of all pairwise interactions of SNP pairs formed from the two genes. To establish statistical significance, the authors generate a set of permutations of the (case/control) dependent variable and compare the measures of interaction from the real data with those from the permutations. In addition, there is the possibility to improve the selection of the initial set of candidate genes integrating a priori information on genes and pathways into GWAS. [Sohns et al. (2009)] suggest to apply and combine gene set enrichment analysis (GSEA) and hierarchical Bayes prioritization (HBP).

Bibliography

[Akaike (1973)] Akaike H. (1973). Information theory and an extension of the maximum likelihood principle. In *2nd International Symposium of Information Theory*, Eds B.N. Petrov & F. Csaki, 267-281, Budapest Akadémia Kiadò.

[Akaike (1978)] Akaike H. (1978). A Bayesian analysis of the minimum AIC procedure. *Annals of the Institute of Statistical Mathematics*, **30**1: 9-14.

[Astle and Balding (2009)] Astle W. and Balding D.J. (2009): Population structure and cryptic relatedness in genetic association studies. *Statistical Science*, **24**(4): 451-471.

[Beran (1987)] Beran R. (1987): Prepivoting to reduce level error of confidence sets. *Biometrika*, **74**(3): 457-468.

[Boyd and Vandenberghe (2004)] Boyd S., Vandenberghe L. (2004). *Convex Optimization*, Cambridge University Press, Cambridge.

[Breiman (1992)] Breiman L. (2004). The Little Bootstrap and Other Methods for Dimensionally Selection in Regression: X-Fixed Prediction Error. *Journal of the American Statistical Association*, **87**: 738-754.

[Breiman (2001)] Breiman L. (2001). Random Forests. *Machine Learning*, **45**: 5-32.

[Breheny and Huang (2009)] Breheny P., Huang J. (2009). Coordinate descent algorithms for nonconvex penalized regression methods. *Annals of Applied Statistics*, to appear.

[Canty and Ripley (2010)] Canty A., Ripley B. (2010). `boot`: Bootstrap R (S-Plus) Functions. R package version 1.2-42.

[Carpenter and Bithell (2000)] Carpenter J., Bithell J. (2000). Bootstrap confidence intervals: when, which, what? A practical guide for medical statisticians. *Statistics in Medicine*, Volume **19**: 1141-1164.

[Chatterjee and Lahiri (2010)] Chatterjee A., Lahiri S.N. (2010). Asymptotic properties of the residual bootstrap for Lasso estimators. *Proceedings of the American Mathematical Society*, Volume **138**: 4497-4509.

[Chatterjee and Lahiri (2010)] Chatterjee A., Lahiri S.N. (2010). Bootstrapping Lasso estimators. *submitted*, : .

[Chatterjee at al. (2009)] Chatterjee N., Chen Y.H., Luo S., and Carroll R. (2009). Analysis of case-control association studies: SNPs, imputation and Haplotypes. *Statistical Science*, **24**(4): 489-502.

[Claeskensa *et al.* (2003)] Claeskensa G., Aerts M., Molenberghs G. (2003). A quadratic bootstrap method and improved estimation in logistic regression. *Statistics & Probability Letters*, **61**: 383-394.

[Crivellia *et al.* (1995)] Crivellia A., Firinguettia L., Montañoa R., Muñóza M. (1995). Confidence intervals in ridge regression by bootstrapping the dependent variable: a simulation study. *Communications in Statistics - Simulation and Computation*, **24**(3): 631-652.

[Davison and Hinkley (1997)] Davison A.C., Hinkley D.V. (1997). *Bootstrap Methods and their Application*, Cambridge University Press, New York.

[Díaz-Uriarte and Alvarz de Andrés (2005)] Díaz-Uriarte R. and Alvarz de Andrés S. (2005). Gene selection and classification of microarray data using random forest. *BMC Bioinformatics*, **7**(3).

[Efron (1979)] Efron B. (1979). Bootstrap methods: another look at the jackknife. *The Annals of Statistics*, **7**(1): 1-26.

[Efron et al. (2004)] Efron B., Hastie T., Johnstone I, Tibshirani R. (2004). Least Angle Regression. *The Annals of Statistics*, **32**(2): 407-499.

[Fan (1997)] Fan J. (1997). "Comment on 'Wavelets in Statistics: A Review' by A. Antoniadis". *Journal of the Italian Statistical Association*, **6**: 131-138.

[Fan and Li (2001)] Fan J., Li R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, **96**: 1348-1360.

[Fan and Lv (2008)] Fan J., Lv R. (2008). Sure independence screening for ultrahigh dimensional feature space. *Journal of the Royal Statistical Society*, **70**(5): 849-911.

[Fan at al. (2009)] Fan J., Feng Y. and Wu Y. (2009). Network exploration via the adaptive lasso and SCAD penalties. *The Annals of Applied Statistics*, **3**(2): 521-541.

[Fan at al. (2009a)] Fan J., Feng Y. and Wu Y. (2009). Network exploration via the adaptive lasso and SCAD penalties. *The Annals of Applied Statistics*, **3**(2): 521-541.

[Fan at al. (2009b)] Fan J., Samworth R. and Wu Y. (2009). Ultrahigh dimensional feature selection: beyond the linear model. *Journal of Machine Learning Research*, **10**: 2013-2038.

[Fan and Lv (2010)] Fan J., Lv J. (2010). A selective overview of variable selection in high dimensional feature space. *Statistica Sinica*, **20**: 101-148.

[Fan et al. (2010)] Fan J., Feng Y., and Samworth Y.W. (2010). Sure Independence Screening. *R package version 0.5.* http://CRAN.R-project.org/package=SIS.

[Fraley and Hesterberg (2009)] Fraley C., Hesterberg T. (2009). Least-Angle Regression and LASSO for Large Datasets. *Statistical Analysis and Data Mining*, **1**: 251-259.

[Freund and Schapire (1997)] Freund I., Schapire R.E. (1997): A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, **55**: 119-139.

[Friedl and Tilg (1995)] Friedl H., Tilg N. (1995): Variance estimates in logistic regression using the bootstrap. *Communications in Statistics - Theory and Methods*, **24**(2): 473-486.

[Friedman et al. (2010)] Friedman J., Hastie T., Tibshirani R. (2010): Regularization Path for Generalized Linear Models via Coordinate Descent. *Journal of Statistical Software*, **33**(1).

[Goeman (2010)] Goeman J.J. (2010): L_1 Penalized Estimation in the Cox Proportional Hazards Model. *Biometrical Journal*, **52**(1): 70-84.

[Goddard et al. 2009] Goddard M.E., Wray N.R., Verbyla K and Wisscher P.M. (2009): Estimating effects and making predictions from genome-wide marker data. *Statistical Science*, **24**(4): 517-529.

[Guyon and Elisseeff (2003)] Guyon I., Elisseeff A. (2003): An Introduction to Variable and Feature Selection. *Journal of Machine Learning Research*, **3**: 1157-1182.

[Hastie et al. (2009)] Hastie T., Tibshirani R., Friedman J. (2009). *The Elements of Statistical Learning*, Springer, New York.

[Hartwell et al. (2004)] HHartwell L.H., Hood L., Goldberg M.L., Reynolds A.E., Silver L.M., Veres C.V. (2004). *Genetics: From Genes to Genomes*, McGraw-Hill, New York.

[Hesterberg et al. (2008)] Hesterberg T., Choi N.H., Maier L., and Fraily C. (2008). Least angle and l_1 penalized regression: A review. *Statistics Surveys*, **2**: 61-93.

[Hindorff et al. (2009)] Hindorff L.A., Sethuparthy P., Junkins H.A., Ramos E.M., Mehta J.P., Collins F.S., and Manolio T.A. (2009). Potential etiologic and functional implications of genome-wide association loci for human diseases and traits. *PANS*, **106**(23): 9362-7.

[Hurvich and Tsai (1989)] Hurvich C.M. and Tsai C.L. (1989). Regression and Time Series Model Selection in Small Samples. *Biometrika*, **76**: 297-307.

[Huang (2003)] Huang F. (2003). Prediction error property of the lasso estimator and its generalization. *Australian & New Zealand Journal of Statistics*, **45**(2): 217-228.

[Huang and Xie (2007)] Huang J. (2007). Asymptotic oracle properties of SCAD-penalized last squares estimators. *IMS Lecture Notes-Monograph Series. Asymptotics: Particles, Processes and Inverse Problems*, **55**: 149-166.

[Huang et al. (2008a)] Huang J., Ma S., and Zhang C.H. (2008). The Integrated Lasso for High-Dimensional Logistic Regression. *Technical Report*, **No. 392**. http://www.stat.uiowa.edu/techrep/tr392.pdf

[Huang et al. (2008b)] Huang J., Ma S., and Zhang C.H. (2008). Adaptive lasso for sparse high-dimensional regression models. *Statistica Sinica*, **18**: 1603-1618.

[Jiang et al. (2010)] Jiang X., Bermada M.M, and Visweswaran S. (2010). Identifying Genetic Interactions in Genome-Wide Data Using Bayesian Networks. *Genetic Epidemiology*, **34**: 575-581.

[Khoury and Wacholder (2008)] Khoury M.J., and Wacholder S. (2008). Invited Commentary: From Genome-Wide Association Studies to Gene-Environment-Wide Interaction Studies – Challenges and Opportunities. *American Journal of Epidemiology*, **169**(2): 227-230.

[Kim et al. (2008)] Kim Y., Choi H., and Oh H.S. (2008). Smoothly Clipped Absolute Deviation on High Dimensions. *Journal of the American Statistical Association*, **103**(484): 1665-1673.

[Kim et al. (2006)] Kim Y., Kim J., and Kim K. (2006). Blockwise Sparse Regression. *Statistica Sinica*, **16**(2): 375-390.

[Knight and Fu (2000)] Knight K., Fu W. (2000). Asymptotics for the lasso-type estimators. *The Annals of Statistics*, **28**(5): 1356-1378.

[Koenker (2005)] Koenker R.W. (2005). Quantile Regression. *Cambridge University Press.*

[Kohavi and John (1997)] Kohavi R., John G.H. (1997). Wrappers for feature subset selection. *Artificial Intelligence*, **97**(1-2): 273-324.

[Kohavi and John (2001)] Kohavi R., John G.H. (2001). The wrapper approach. in Liu H. and Motoda H., Sprihger, editors, *Feature extraction construction and selection: a data mining perspective*, Dordrecht, NL: Kluwer Academic Press. Chapter 3.

[Kooperberg at al. 2009] Kooperberg C., LeBlanc M., Dai J.Y. and Rajapakse I. (2009): Structures and assumptions: strategies to harness gene x gene and gene x environment interactions in GWAS. *Statistical Science*, **24**(4): 472-488.

[Kraft et al. 2009] Kraft P. Zeggini E. and Ioannidis J.P.A. (2009): Replication in genome-wide association studies. *Statistical Science*, **24**(4): 561-573.

[Krämer at al. (2009)] Krämer N., Schäfer J., Boulesteix A.L (2009). Regularized estimation of large-scale gene association networks using graphical Gaussian models. *BMC Bioinformatics*, **10**: 384.

[Laird and Lange 2009] Laird N.M., Lange C. (2009): The role of family-based designs in genome-wide association studies. *Statistical Science*, **24**(4): 388-397.

[Langley, 1994] Langley P. (1994): Selection of Relevant Features in Machine Learning. *Institute for the Study of Learning and Expertise*, Technical Report**94-3**.

[Lee (1990)] Lee K.W. (1990). Bootstrapping logistic regression models with random regressors. *Communications in Statistics - Theory and Methods*, **19**(7): 2527-2539.

[Li and Conti (2009)] Li D. and Conti D.V. (2009). Detecting gene-environment interactions using a combined case-only and case-control approach. *American Journal of Epidemiology*, **169**: 497-504.

[Liaw and Winter (2002)] Liaw A. and Wiener M. (2002). Classification and Regression by Random Forest. *R News*, **2**(3): 18-22.

[Lo et al. (2008)] Lo S.H., Chernoff H., Cong L., Ding Y. and Zheng T. (2008). Discovering interactions among BRCA1 and other candidate genes associated with sporadic breast cancer. *PANS*, **105**(34): 12387-92.

[Luo et al. (2006)] Luo X., Stefanski L.A., and Boss D.D. (2006). Tuning Variable Selection Procedures by Adding Noise. *Technometrics*, **48**: 165-175.

[Manolio (2010)] Manolio T.A. (2010). Genomewide Association Studies and Assessment of the Risk of Disease. *The New England Journal of Medicine*, **363**(2): 166-176.

[McCullagh and Nelder (1989)] McCullagh P., Nelder J.A. (1989). *Generalized Linear Models*, Chapman and Hall, London.

[McCullagh and Vinod (1998)] McCullagh P., Vinod H.D. (1998) Implementing the Double Bootstrap. *Computational Economics*, **12**, 79-95.

[Meier et al. (2008)] Meier L, van de Geer S. and Bühlman P. (2008). The group lasso for logistic regression. *Journal of the Royal Statistical Society: Series B*, **70**(1): 53-71.

[Meinshausen (2006)] Meinshausen N. and Bühlmann P. (2006). High-Dimensional Graphs and Variable Selection With the Lasso. *The Annals of Statistics*, **34**: 1012-1030.

[Meinshausen et al. (2007a)] Meinshausen N., Rocha G., and Yu B. (2007). Discussion: a tale of three cousins: LASSO, L2BOOSTING and DANTZIG. *The Annals of Statistics*, **35**(6): 2373-2384.

[Meinshausen (2007b)] Meinshausen N. (2007). Relaxed Lasso. *Computational Statistics and Data Analysis*, **51**(1): 374-393.

[Meinshausen et al. (June 2009)] Meinshausen N., Meier L., and Bühlmann P. (June 12, 2009). P-Values for High-Dimensional Regression. *Technometrics*, **48**: 165-175.

[Meinshausen and Yu (2009)] Meinshausen N., Yu B. (2009). LASSO-Type recovery of sparse representations for high-dimensional data. *The Annals of Statistics*, **37**(1): 246-270.

[Meinshausen (2009)] Meinshausen N. (2009). Node harvest: simple and interpretable regression and classification. *Submitted 12 October 2009*, under revision. http://arxiv4.library.cornell.edu/abs/0910.2145v1

[Moulton and Zeger (1991)] Moulton L.H., Zeger S.L.(1991). Bootstrapping generalized linear models. *Computational Statistics & Data Analysis*, **11**: 53-63.

[Mukherjee et al. (2008)] Mukherjee B., Ahn J., Gruber S.B., Rennert G., Moreno V. and Chatterjee N.(2008). Tests for gene-environment interaction from case-control data: a novel study of type I error, power and designs. *Genetic Epidemiology*, **32**: 615-626.

[Murcray et al. (2007)] Murcray C.E., Lewinger J.P., and Gauderman W.J.(2007). Gene-Environment Interaction in Genome-Wide Association Studies. *American Journal of Epidemiology*, **169**(2): 219-226.

[Nocedal and Wright, (1999)] Nocerdal J., Wright S.J.(1999). Numerical Optimization. *Springer*, New York.

[Osborne at al. (2000)] Osborne M.R., Persnell B., Turlach B.A. (2000). Journal of Computational and Graphical Statistics. **9**(2): 319-337.

[Park and Hastie (2008)] Park M.I. , Hastie T.(2008). Penalized logistic regression for detecting gene interactions. Biometrics. **9**(1): 30-50.

[Park T. and Casella (2008)] Park T., Casella G. (2008). The Bayesian Lasso. *Journal of the American Statistical Association*, **103**(482): 681-686.

[Park M.Y. and Hastie (2007)] Park M.Y., T. Hastie T. (2007). L1 regularization path algorithm for generalized linear models. *J. R. Statist. Soc. B*, **69**: 659-677.

[Pfeiffer at al. 2009] Pfeiffer R.M., Gail M.H. and Pee D. (2009): On combining data from genome-wide association studies to discover disease-associated SNPs. *Statistical Science*, **24**(4): 547-560.

[Pötscher and Leeb (2009)] B.M. Pötscher, H. Leeb (2009). On the distribution of penalized maximum likelihood estimators: The LASSO, SCAD, and thresholding. *Journal of Multivariate Analysis*, **100**: 2065-2082.

[Ritchie et al. 2001] Ritchie M.D., Hahn L.W., Roodi N., Bailey L.R., Dupoint W.D., Parl F.F., and Moore H.J. (2001): Multifactor-Dimensionality Reduction Reveals High-Order Interactions among Estrogen-Metabolism Genes in Sporadic Breast Cancer. *The American Journal of Human Genetics*, **69**: 138-147.

[Roeder and Wasserman 2009] Roeder K., Wasserman L. (2009): Genome-wide significance levels and weighted hypothesis testing. *Statistical Science*, **24**(4): 398-413.

[Rondeau et al. (2003)] Rondeau v., Commenges D., Joly P. (2003). Maximum penalized likelihood estimation in a gamma-frailty. *Lifetime Data Analysis*, **9**(2): 139-153.

[Sandri and Zuccolotto (2008)] Sandri M., Zuccolotto P. (2008). A bias correction algorithm for the Gini Variable importance measure in classification trees. *Journal of Computational and Graphical Statistics*, **17**(3): 611-628.

[Sandri and Zuccolotto (2010)] Sandri M., Zuccolotto P. (2010). Analysis and correction of bias in Total Decrease in Node Impurity measures for tree-based algorithms. *Statistics and Computing*, **20**: 393-407.

[Schwarz (1978)] Schwarz G. (1978). Estimating the Dimension of a Model. *Annals of Statistics*, **6**: 461-464.

[Schwarz et al. (2010)] Schwarz D.F. König I.R., and Ziegler A. (2010). On safari to Random Jungle: a fast implementation of Random Forests for high-dimensional data. *Bionformatics*, **26**(14): 1752-1758.

[Shi et al. (2007)] Shi W., Lee K.E., and Wahba G. (2007). Detecting disease-causing genes by LASSO-Patternsearch algorithm. *BMC Proceedings*, **1**(Suppl 1): S60.

[Sohns et al. (2009)] Sohns M., Rosenberger A. and Bickeböller H. (2009). Integration of a priori gene set information into genome-wide association studies. *BMC Proceedings*, **3**(Suppl 7): S95.

[Su et al. 2009] Su Z., Cardin N., the welcome trust case control consortium, Donnely P. and Manchini J. (2009): A bayesian method for detecting and characterizing allelic heterogeneity and boosting signals in genome-wide association studies. *Statistical Science*, **24**(4): 430-450.

[Szymczak et al. 2009] Szymczak S., Biernacka J.M., Cordell H.J., González-Recio O., König I.R., Zhang H., and Sun V.Y. (2009): Machine Learning in Genome-Wide Association Studies. *Genetic Epidemiology*, **33**(Supplement 1): S51-S57.

[Thomas et al. 2010] Thomas D.C., Casey G., Conti D.V., Haile R.W., Lewinger J.P. and Stram D.O.(2009): Methodological issues in multi-stage genome-wide association studies. *Statistical Science*, **24**(4): 414-429.

[Thomas (2010)] Thomas D. (2010). Gene-environment-wide association studies: emerging approaches. *Nature Reviews Genetics*, **11**: 259-272.

[Tian et al. (2008)] Tian G.L., Tang M.L., Fang H.B., and Tan M. (2008). Efficient methods for estimating constrained parameters with applications to regularized (lasso) logistic regression. *Computational Statistics & Data Analysis*, **52**: 3528-3542.

[Tibshirani (1996)] Tibshirani R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society*, Ser. B **58**: 267-288.

[Tutz G. and Ulbricht J. (2009)] Tutz G. and Ulbricht J. (2009). Penalized regression with correlation-based penalty. *Statistics and Computing*, **19**: 239-253.

[Ulbricht (2010)] Ulbricht J. (2010). Variable selection in generalized linear models. *Ph.D Thesis*, LMU Munich.

[van de Geer et al. (2010)] Van de Geer S., Bühlmannn P., Zhou S. (2010): Prediction and variable selection with the adaptive LASSO. *arXiv:1001.5176v2*, ().

[Vinod (1995)] Vinod H.D. (1995). Double bootstrap for shrinkage estimators. *Journal of Econometrics*, **68**(287-302).

[Wang et al (2010)] Wang X., Park T. and Carriere K.C. (2010). Variable selection via combined penalization for high-dimensional data analysis. *Computational Statistics & Data Analysis*, **54**(10): 2230-2243.

[Wang and Leng (2007)] Wang H., Leng C. (2007). Unified LASSO Estimation by Least Squares Approximation. *Journal of the American Statistical Association*, **102**(479): 1039-1048.

[Wu et al. (2009)] Wu T.T., Chen Y.F., Hastie T., Sobel E., and Lange K. (2009). Genome-wide association analysis by lasso penalized logistic regression. *Bionformatics*, **25**(6): 714-721.

[Wu et al. (2007)] Wu Y., Boss D.D., and Stefansky L.A. (2007). Controlling Variable Selection by the Addition of Pseudovariables. *Journal of the American Statistical Association*, **102**(477): 235-243.

[Yang and Gu (2009)] Yang W.W., and Gu C.C. (2009). Selection of important variables by statistical learning in genome-wide association analysis. *BMC Proceedings*, **3**(Suppl 7): S70.

[Zheng at al. (2009)] Zheng G., Joo J., Zaykin D., Wu C. and Geller N. (2009). Robust test in genome-wide scans under incomplete linkage disequilibrium. *Statistical Science*, **24**(4): 503-516.

[Zhang (2007)] Zhang G.H. (2007). Penalized linear unbiased selection. *Technical Report*, nr.**003**. Department of Statistics, Rutgers University.

[Zhang et al. (2006)] Zhang H.H., Ahn J., Lin X. and Park C. (2007). Gene selection using support vector machines with non-convex penalty. *BIOINFORMATICS*, **22**(1): 88-95.

[Zhang et al. (2010)] Zhang Y., Li R. and Tsai C.L. (2010). Regularization Parameter Selections via Generalized Information Criterion *Journal of the American Statistical Association*, **105**(489): 312-323.

[Zhao and Yu (2006)] Zhao P., Yu B. (2006). On Model Selection Consistency of Lasso. *Journal of Machine Learning Research*, **7**: 2541-2563.

[Zhou et al. (2010)] Zhou H., Shel M.E., Sinsheimer J.S., and Lange K. (2010). Association screening of common and rare genetic variants by penalized regression. *BIOINFRMATICS*, **26**(19): 2375-2382.

[Zöllner and Teslovich 2009] Zöllner S., Teslovich T.M. (2009): Using GWAS data to identify copy number variants contributing to common complex diseases. *Statistical Science*, **24**(4): 530-546.

[Zou and Hastie (2005)] Zou H., Hastie T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society*, Ser. B **67**: 301-320.

[Zou (2006)] Zou H.(2006). The adaptive lasso and its oracle properties. *Journal of the American Statistical Association*, **101**: 1418-1429.

[National Cancer Institute] http://www.cancer.gov/.