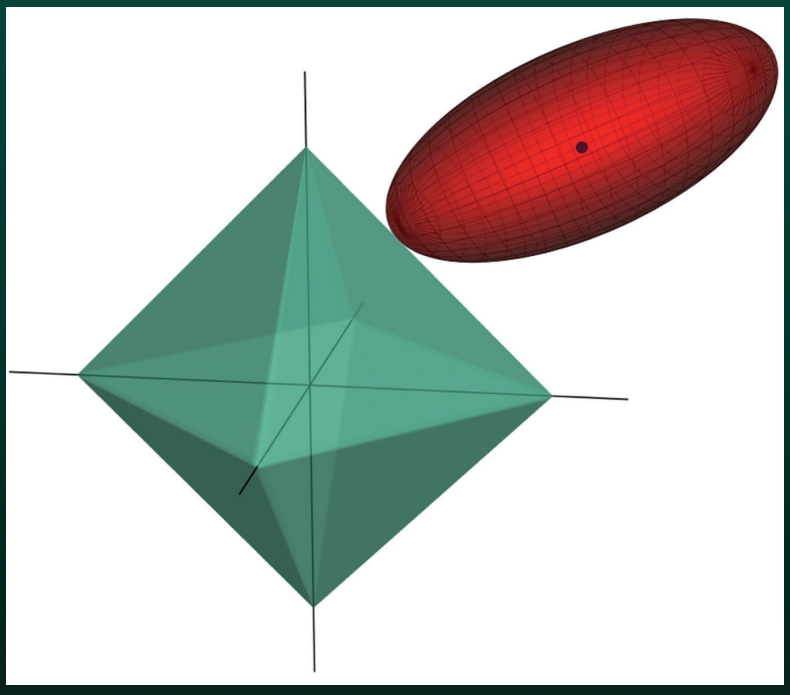
143 Monographs on Statistics and Applied Probability 143

Statistical Learning with Sparsity

The Lasso and Generalizations



Trevor Hastie
Robert Tibshirani
Martin Wainwright

CRC Press
Taylor & Francis Group
A CHAPMAN & HALL BOOK

Statistical Learning with Sparsity
Hastie • Tibshirani • Wainwright

Corrected December 19. 2016

To our parents:

Valerie and Patrick Hastie

Vera and Sami Tibshirani

Patricia and John Wainwright

and to our families:

Samantha, Timothy, and Lynda

Charlie, Ryan, Jess, Julie, and Cheryl

Haruko and Hana

Contents

Preface

In this monograph, we have attempted to summarize the actively developing field of statistical learning with sparsity. A sparse statistical model is one having only a small number of nonzero parameters or weights. It represents a classic case of "*less is more*": a sparse model can be much easier to estimate and interpret than a dense model. In this age of big data, the number of features measured on a person or object can be large, and might be larger than the number of observations. The sparsity assumption allows us to tackle such problems and extract useful and reproducible patterns from big datasets.

The ideas described here represent the work of an entire community of researchers in statistics and machine learning, and we thank everyone for their continuing contributions to this exciting area. We particularly thank our colleagues at Stanford, Berkeley and elsewhere; our collaborators, and our past and current students working in this area. These include Alekh Agarwal, Arash Amini, Francis Bach, Jacob Bien, Stephen Boyd, Andreas Buja, Emmanuel Candes, Alexandra Chouldechova, David Donoho, John Duchi, Brad Efron, Will Fithian, Jerome Friedman, Max G'Sell, Iain Johnstone, Michael Jordan, Ping Li, Po-Ling Loh, Michael Lim, Jason Lee, Richard Lockhart, Rahul Mazumder, Balasubramanian Narashimhan, Sahand Negahban, Guillaume Obozinski, Mee-Young Park, Junyang Qian, Garvesh Raskutti, Pradeep Ravikumar, Saharon Rosset, Prasad Santhanam, Noah Simon, Dennis Sun, Yukai Sun, Jonathan Taylor, Ryan Tibshirani,[1] Stefan Wager, Daniela Witten, Bin Yu, Yuchen Zhang, Ji Zhou, and Hui Zou. We also thank our editor John Kimmel for his advice and support.

Stanford University
and
University of California, Berkeley

Trevor Hastie
Robert Tibshirani
Martin Wainwright

[1]Some of the bibliographic references, for example in Chapters 4 and 6, are to Tibshirani$_2$, R.J., rather than Tibshirani, R.; the former is Ryan Tibshirani, the latter is Robert (son and father).

Chapter 1

Introduction

"I never keep a scorecard or the batting averages. I hate statistics. What I got to know, I keep in my head."

This is a quote from baseball pitcher Dizzy Dean, who played in the major leagues from 1930 to 1947.

How the world has changed in the 75 or so years since that time! Now large quantities of data are collected and mined in nearly every area of science, entertainment, business, and industry. Medical scientists study the genomes of patients to choose the best treatments, to learn the underlying causes of their disease. Online movie and book stores study customer ratings to recommend or sell them new movies or books. Social networks mine information about members and their friends to try to enhance their online experience. And yes, most major league baseball teams have statisticians who collect and analyze detailed information on batters and pitchers to help team managers and players make better decisions.

Thus the world is awash with data. But as Rutherford D. Roger (and others) has said:

"We are drowning in information and starving for knowledge."

There is a crucial need to sort through this mass of information, and pare it down to its bare essentials. For this process to be successful, we need to hope that the world is not as complex as it might be. For example, we hope that not all of the $30,000$ or so genes in the human body are directly involved in the process that leads to the development of cancer. Or that the ratings by a customer on perhaps 50 or 100 different movies are enough to give us a good idea of their tastes. Or that the success of a left-handed pitcher against left-handed batters will be fairly consistent for different batters.

This points to an underlying assumption of simplicity. One form of simplicity is *sparsity*, the central theme of this book. Loosely speaking, a sparse statistical model is one in which only a relatively small number of parameters (or predictors) play an important role. In this book we study methods that exploit sparsity to help recover the underlying signal in a set of data.

The leading example is linear regression, in which we observe N observations of an outcome variable y_i and p associated predictor variables (or features) $x_i = (x_{i1}, \ldots x_{ip})^T$. The goal is to predict the outcome from the

predictors, both for actual prediction with future data and also to discover which predictors play an important role. A linear regression model assumes that

$$y_i = \beta_0 + \sum_{j=1}^{p} x_{ij}\beta_j + e_i, \tag{1.1}$$

where β_0 and $\beta = (\beta_1, \beta_2, \ldots \beta_p)$ are unknown parameters and e_i is an error term. The method of least squares provides estimates of the parameters by minimization of the least-squares objective function

$$\underset{\beta_0,\beta}{\text{minimize}} \sum_{i=1}^{N}(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2. \tag{1.2}$$

Typically all of the least-squares estimates from (1.2) will be nonzero. This will make interpretation of the final model challenging if p is large. In fact, if $p > N$, the least-squares estimates are not unique. There is an infinite set of solutions that make the objective function equal to zero, and these solutions almost surely overfit the data as well.

Thus there is a need to constrain, or *regularize* the estimation process. In the *lasso* or ℓ_1*-regularized regression*, we estimate the parameters by solving the problem

$$\underset{\beta_0,\beta}{\text{minimize}} \sum_{i=1}^{N}(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 \text{ subject to } \|\beta\|_1 \leq t \tag{1.3}$$

where $\|\beta\|_1 = \sum_{j=1}^{p} |\beta_j|$ is the ℓ_1 norm of β, and t is a user-specified parameter. We can think of t as a budget on the total ℓ_1 norm of the parameter vector, and the lasso finds the best fit within this budget.

Why do we use the ℓ_1 norm? Why not use the ℓ_2 norm or any ℓ_q norm? It turns out that the ℓ_1 norm is special. If the budget t is small enough, the lasso yields sparse solution vectors, having only some coordinates that are nonzero. This does not occur for ℓ_q norms with $q > 1$; for $q < 1$, the solutions are sparse but the problem is not convex and this makes the minimization very challenging computationally. The value $q = 1$ is the smallest value that yields a convex problem. Convexity greatly simplifies the computation, as does the sparsity assumption itself. They allow for scalable algorithms that can handle problems with even millions of parameters.

Thus the advantages of sparsity are interpretation of the fitted model and computational convenience. But a third advantage has emerged in the last few years from some deep mathematical analyses of this area. This has been termed the "bet on sparsity" principle:

> *Use a procedure that does well in sparse problems, since no procedure does well in dense problems.*

We can think of this in terms of the amount of information N/p per parameter. If $p \gg N$ and the true model is not sparse, then the number of samples N is too small to allow for accurate estimation of the parameters. But if the true model is sparse, so that only $k < N$ parameters are actually nonzero in the true underlying model, then it turns out that we can estimate the parameters effectively, using the lasso and related methods that we discuss in this book. This may come as somewhat of a surprise, because we are able to do this even though we are not told *which* k of the p parameters are actually nonzero. Of course we cannot do as well as we could if we had that information, but it turns out that we can still do reasonably well.

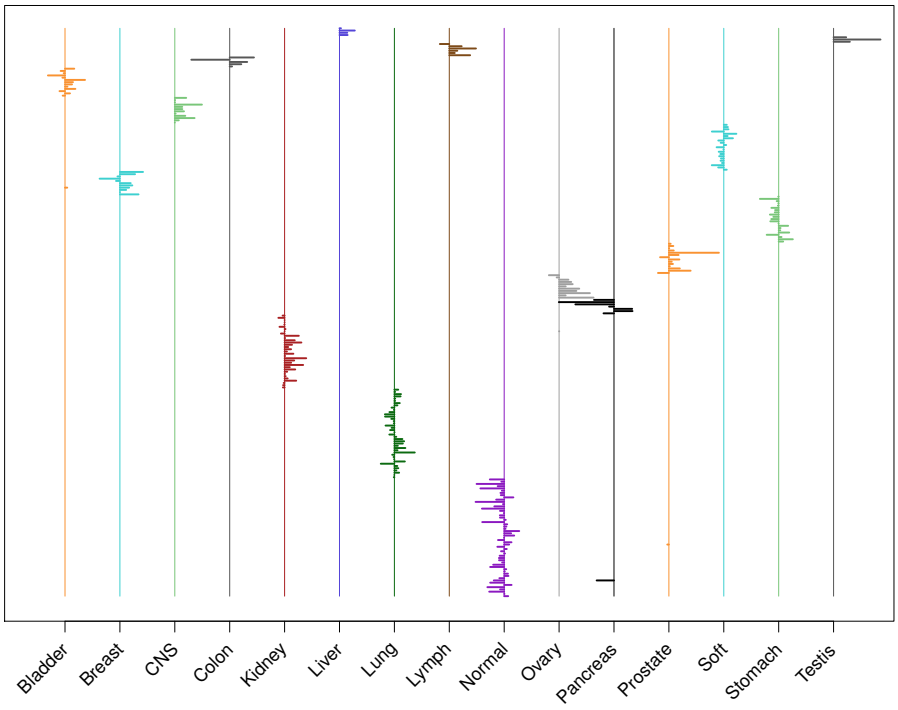


Figure 1.1 *15-class gene expression cancer data: estimated nonzero feature weights from a lasso-regularized multinomial classifier. Shown are the 254 genes (out of 4718) with at least one nonzero weight among the 15 classes. The genes (unlabelled) run from top to bottom. Line segments pointing to the right indicate positive weights, and to the left, negative weights. We see that only a handful of genes are needed to characterize each class.*

For all of these reasons, the area of sparse statistical modelling is exciting—for data analysts, computer scientists, and theorists—and practically useful. Figure 1.1 shows an example. The data consists of quantitative gene expression measurements of 4718 genes on samples from 349 cancer patients. The cancers have been categorized into 15 different types such as "Bladder," "Breast",

"CNS," etc. The goal is to build a classifier to predict cancer class based on some or all of the 4718 features. We want the classifier to have a low error rate on independent samples and would prefer that it depend only on a subset of the genes, to aid in our understanding of the underlying biology.

For this purpose we applied a lasso-regularized multinomial classifier to these data, as described in Chapter 3. This produces a set of 4718 weights or coefficients for each of the 15 classes, for discriminating each class from the rest. Because of the ℓ_1 penalty, only some of these weights may be nonzero (depending on the choice of the regularization parameter). We used cross-validation to estimate the optimal choice of regularization parameter, and display the resulting weights in Figure 1.1. Only 254 genes have at least one nonzero weight, and these are displayed in the figure. The cross-validated error rate for this classifier is about 10%, so the procedure correctly predicts the class of about 90% of the samples. By comparison, a standard support vector classifier had a slightly higher error rate (13%) using all of the features. Using sparsity, the lasso procedure has dramatically reduced the number of features without sacrificing accuracy. Sparsity has also brought computational efficiency: although there are potentially $4718 \times 15 \approx 70,000$ parameters to estimate, the entire calculation for Figure 1.1 was done on a standard laptop computer in less than a minute. For this computation we used the `glmnet` procedure described in Chapters 3 and 5.

Figure 1.2 shows another example taken from an article by Candès and Wakin (2008) in the field of *compressed sensing*. On the left is a megapixel image. In order to reduce the amount of space needed to store the image, we represent it in a wavelet basis, whose coefficients are shown in the middle panel. The largest $25,000$ coefficients are then retained and the rest zeroed out, yielding the excellent reconstruction in the right image. This all works because of sparsity: although the image seems complex, in the wavelet basis it is simple and hence only a relatively small number of coefficients are nonzero. The original image can be perfectly recovered from just $96,000$ incoherent measurements. Compressed sensing is a powerful tool for image analysis, and is described in Chapter 10.

In this book we have tried to summarize the hot and rapidly evolving field of sparse statistical modelling. In Chapter 2 we describe and illustrate the lasso for linear regression, and a simple coordinate descent algorithm for its computation. Chapter 3 covers the application of ℓ_1 penalties to generalized linear models such as multinomial and survival models, as well as support vector machines. Generalized penalties such as the elastic net and group lasso are discussed in Chapter 4. Chapter 5 reviews numerical methods for optimization, with an emphasis on first-order methods that are useful for the large-scale problems that are discussed in this book. In Chapter 6, we discuss methods for statistical inference for fitted (lasso) models, including the bootstrap, Bayesian methods and some more recently developed approaches. Sparse matrix decomposition is the topic of Chapter 7, and we apply these methods in the context of sparse multivariate analysis in Chapter 8. Graph-

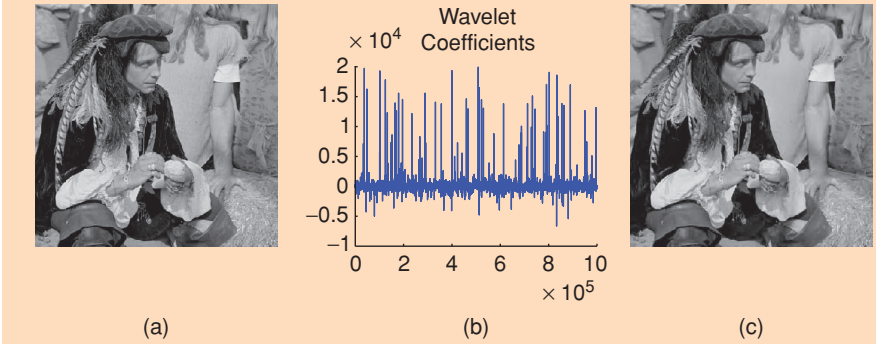


Figure 1.2 *(a) Original megapixel image with pixel values in the range* $[0, 255]$ *and (b) its wavelet transform coefficients (arranged in random order for enhanced visibility). Relatively few wavelet coefficients capture most of the signal energy; many such images are highly compressible. (c) The reconstruction obtained by zeroing out all the coefficients in the wavelet expansion but the* $25,000$ *largest (pixel values are thresholded to the range* $[0, 255]$*). The differences from the original picture are hardly noticeable.*

ical models and their selection are discussed in Chapter 9 while compressed sensing is the topic of Chapter 10. Finally, a survey of theoretical results for the lasso is given in Chapter 11.

We note that both *supervised* and *unsupervised* learning problems are discussed in this book, the former in Chapters 2, 3, 4, and 10, and the latter in Chapters 7 and 8.

Notation

We have adopted a notation to reduce mathematical clutter. Vectors are column vectors by default; hence $\beta \in \mathbb{R}^p$ is a column vector, and its transpose β^T is a row vector. All vectors are lower case and non-bold, except N-vectors which are bold, where N is the sample size. For example $\mathbf{x}_j$ might be the N-vector of observed values for the j^{th} variable, and $\mathbf{y}$ the response N-vector. All matrices are bold; hence $\mathbf{X}$ might represent the $N \times p$ matrix of observed predictors, and $\mathbf{\Theta}$ a $p \times p$ precision matrix. This allows us to use $x_i \in \mathbb{R}^p$ to represent the vector of p features for observation i (i.e., x_i^T is the i^{th} row of $\mathbf{X}$), while $\mathbf{x}_k$ is the k^{th} column of $\mathbf{X}$, without ambiguity.

Chapter 2

The Lasso for Linear Models

In this chapter, we introduce the lasso estimator for linear regression. We describe the basic lasso method, and outline a simple approach for its implementation. We relate the lasso to ridge regression, and also view it as a Bayesian estimator.

2.1 Introduction

In the linear regression setting, we are given N samples $\{(x_i, y_i)\}_{i=1}^N$, where each $x_i = (x_{i1}, \ldots, x_{ip})$ is a p-dimensional vector of features or predictors, and each $y_i \in \mathbb{R}$ is the associated response variable. Our goal is to approximate the response variable y_i using a linear combination of the predictors

$$\eta(x_i) = \beta_0 + \sum_{j=1}^{p} x_{ij}\beta_j. \tag{2.1}$$

The model is parametrized by the vector of regression weights $\beta = (\beta_1, \ldots, \beta_p) \in \mathbb{R}^p$ and an intercept (or "bias") term $\beta_0 \in \mathbb{R}$.

The usual "least-squares" estimator for the pair (β_0, β) is based on minimizing squared-error loss:

$$\underset{\beta_0,\beta}{\text{minimize}} \left\{ \frac{1}{2N} \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j\right)^2 \right\}. \tag{2.2}$$

There are two reasons why we might consider an alternative to the least-squares estimate. The first reason is *prediction accuracy*: the least-squares estimate often has low bias but large variance, and prediction accuracy can sometimes be improved by shrinking the values of the regression coefficients, or setting some coefficients to zero. By doing so, we introduce some bias but reduce the variance of the predicted values, and hence may improve the overall prediction accuracy (as measured in terms of the mean-squared error). The second reason is for the purposes of *interpretation*. With a large number of predictors, we often would like to identify a smaller subset of these predictors that exhibit the strongest effects.

This chapter is devoted to discussion of the *lasso*, a method that combines the least-squares loss (2.2) with an ℓ_1-constraint, or bound on the sum of the absolute values of the coefficients. Relative to the least-squares solution, this constraint has the effect of shrinking the coefficients, and even setting some to zero.[1] In this way it provides an automatic way for doing model selection in linear regression. Moreover, unlike some other criteria for model selection, the resulting optimization problem is convex, and can be solved efficiently for large problems.

2.2 The Lasso Estimator

Given a collection of N predictor-response pairs $\{(x_i, y_i)\}_{i=1}^N$, the lasso finds the solution $(\widehat{\beta}_0, \widehat{\beta})$ to the optimization problem

$$\begin{aligned} &\underset{\beta_0,\beta}{\text{minimize}} \left\{ \frac{1}{2N} \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 \right\} \\ &\text{subject to } \sum_{j=1}^{p} |\beta_j| \leq t. \end{aligned} \tag{2.3}$$

The constraint $\sum_{j=1}^p |\beta_j| \leq t$ can be written more compactly as the ℓ_1-norm constraint $\|\beta\|_1 \leq t$. Furthermore, (2.3) is often represented using matrix-vector notation. Let $\mathbf{y} = (y_1, \ldots, y_N)$ denote the N-vector of responses, and $\mathbf{X}$ be an $N \times p$ matrix with $x_i \in \mathbb{R}^p$ in its i^{th} row, then the optimization problem (2.3) can be re-expressed as

$$\begin{aligned} &\underset{\beta_0,\beta}{\text{minimize}} \left\{ \frac{1}{2N} \|\mathbf{y} - \beta_0 \mathbf{1} - \mathbf{X}\beta\|_2^2 \right\} \\ &\text{subject to } \|\beta\|_1 \leq t, \end{aligned} \tag{2.4}$$

where $\mathbf{1}$ is the vector of N ones, and $\|\cdot\|_2$ denotes the usual Euclidean norm on vectors. The bound t is a kind of "budget": it limits the sum of the absolute values of the parameter estimates. Since a shrunken parameter estimate corresponds to a more heavily-constrained model, this budget limits how well we can fit the data. It must be specified by an external procedure such as cross-validation, which we discuss later in the chapter.

Typically, we first standardize the predictors $\mathbf{X}$ so that each column is centered ($\frac{1}{N}\sum_{i=1}^N x_{ij} = 0$) and has unit variance ($\frac{1}{N}\sum_{i=1}^N x_{ij}^2 = 1$). Without

[1] A *lasso* is a long rope with a noose at one end, used to catch horses and cattle. In a figurative sense, the method "lassos" the coefficients of the model. In the original lasso paper (Tibshirani 1996), the name "lasso" was also introduced as an acronym for "Least Absolute Selection and Shrinkage Operator."
Pronunciation: in the US "lasso" tends to be pronounced "lass-oh" (oh as in goat), while in the UK "lass-oo." In the OED (2nd edition, 1965): "lasso is pronounced lăsoo by those who use it, and by most English people too."

standardization, the lasso solutions would depend on the units (e.g., feet versus meters) used to measure the predictors. On the other hand, we typically would not standardize if the features were measured in the same units. For convenience, we also assume that the outcome values y_i have been centered, meaning that $\frac{1}{N}\sum_{i=1}^{N} y_i = 0$. These centering conditions are convenient, since they mean that we can omit the intercept term β_0 in the lasso optimization. Given an optimal lasso solution $\widehat{\beta}$ on the centered data, we can recover the optimal solutions for the uncentered data: $\widehat{\beta}$ is the same, and the intercept $\widehat{\beta}_0$ is given by

$$\widehat{\beta}_0 = \bar{y} - \sum_{j=1}^{p} \bar{x}_j \widehat{\beta}_j,$$

where $\bar{y}$ and $\{\bar{x}_j\}_1^p$ are the original means.[2] For this reason, we omit the intercept β_0 from the lasso for the remainder of this chapter.

It is often convenient to rewrite the lasso problem in the so-called Lagrangian form

$$\underset{\beta \in \mathbb{R}^p}{\text{minimize}} \left\{ \frac{1}{2N} \|\mathbf{y} - \mathbf{X}\beta\|_2^2 + \lambda \|\beta\|_1 \right\}, \tag{2.5}$$

for some $\lambda \geq 0$. By Lagrangian duality, there is a one-to-one correspondence between the constrained problem (2.3) and the Lagrangian form (2.5): for each value of t in the range where the constraint $\|\beta\|_1 \leq t$ is active, there is a corresponding value of λ that yields the same solution from the Lagrangian form (2.5). Conversely, the solution $\widehat{\beta}_\lambda$ to problem (2.5) solves the bound problem with $t = \|\widehat{\beta}_\lambda\|_1$.

We note that in many descriptions of the lasso, the factor $1/2N$ appearing in (2.3) and (2.5) is replaced by $1/2$ or simply 1. Although this makes no difference in (2.3), and corresponds to a simple reparametrization of λ in (2.5), this kind of standardization makes λ values comparable for different sample sizes (useful for cross-validation).

The theory of convex analysis tells us that necessary and sufficient conditions for a solution to problem (2.5) take the form

$$-\frac{1}{N}\langle \mathbf{x}_j, \mathbf{y} - \mathbf{X}\beta \rangle + \lambda s_j = 0, \ j = 1, \ldots, p. \tag{2.6}$$

Here each s_j is an unknown quantity equal to $\text{sign}(\beta_j)$ if $\beta_j \neq 0$ and some value lying in $[-1, 1]$ otherwise—that is, it is a subgradient for the absolute value function (see Chapter 5 for details). In other words, the solutions $\hat{\beta}$ to problem (2.5) are the same as solutions $(\hat{\beta}, \hat{s})$ to (2.6). This system is a form of the so-called Karush–Kuhn–Tucker (KKT) conditions for problem (2.5). Expressing a problem in subgradient form can be useful for designing

[2] This is typically only true for linear regression with squared-error loss; it's not true, for example, for lasso logistic regression.

algorithms for finding its solutions. More details are given in Exercises (2.3) and (2.4).

As an example of the lasso, let us consider the data given in Table 2.1, taken from Thomas (1990). The outcome is the total overall reported crime rate per

Table 2.1 *Crime data: Crime rate and five predictors, for $N = 50$ U.S. cities.*

city	funding	hs	not-hs	college	college4	crime rate
1	40	74	11	31	20	478
2	32	72	11	43	18	494
3	57	70	18	16	16	643
4	31	71	11	25	19	341
5	67	72	9	29	24	773
⋮	⋮	⋮	⋮	⋮		
50	66	67	26	18	16	940

one million residents in 50 U.S cities. There are five predictors: annual police funding in dollars per resident, percent of people 25 years and older with four years of high school, percent of 16- to 19-year olds not in high school and not high school graduates, percent of 18- to 24-year olds in college, and percent of people 25 years and older with at least four years of college. This small example is for illustration only, but helps to demonstrate the nature of the lasso solutions. Typically the lasso is most useful for much larger problems, including "wide" data for which $p \gg N$.

The left panel of Figure 2.1 shows the result of applying the lasso with the bound t varying from zero on the left, all the way to a large value on the right, where it has no effect. The horizontal axis has been scaled so that the maximal bound, corresponding to the least-squares estimates $\tilde{\beta}$, is one. We see that for much of the range of the bound, many of the estimates are exactly zero and hence the corresponding predictor(s) would be excluded from the model. Why does the lasso have this model selection property? It is due to the geometry that underlies the ℓ_1 constraint $\|\beta\|_1 \leq t$. To understand this better, the right panel shows the estimates from *ridge regression*, a technique that predates the lasso. It solves a criterion very similar to (2.3):

$$\begin{aligned} &\underset{\beta_0,\beta}{\text{minimize}} \left\{ \frac{1}{2N} \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 \right\} \\ &\text{subject to } \sum_{j=1}^{p} \beta_j^2 \leq t^2. \end{aligned} \tag{2.7}$$

The ridge profiles in the right panel have roughly the same shape as the lasso profiles, but are not equal to zero except at the left end. Figure 2.2 contrasts the two constraints used in the lasso and ridge regression. The residual sum

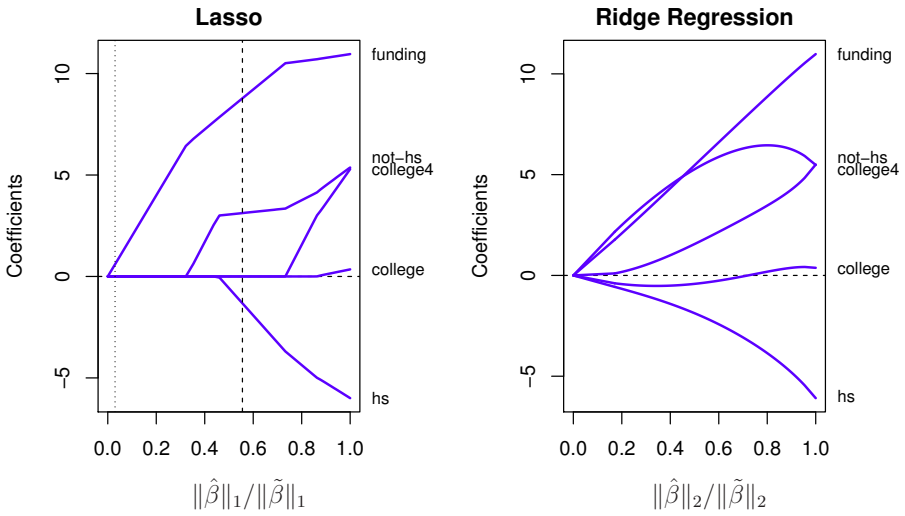


Figure 2.1 *Left: Coefficient path for the lasso, plotted versus the ℓ_1 norm of the coefficient vector, relative to the norm of the unrestricted least-squares estimate $\tilde{\beta}$. Right: Same for ridge regression, plotted against the relative ℓ_2 norm.*

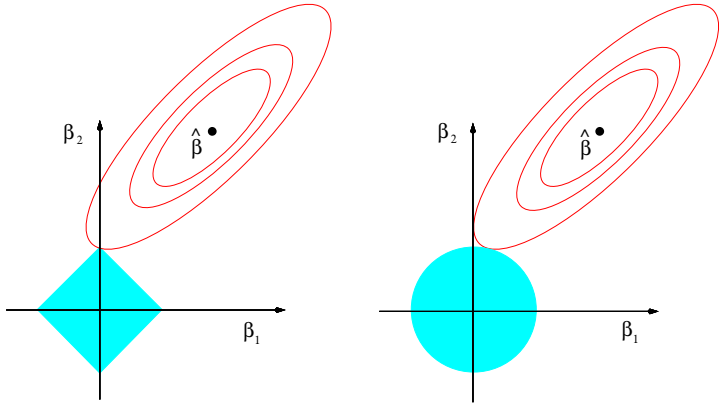


Figure 2.2 *Estimation picture for the lasso (left) and ridge regression (right). The solid blue areas are the constraint regions $|\beta_1|+|\beta_2| \leq t$ and $\beta_1^2+\beta_2^2 \leq t^2$, respectively, while the red ellipses are the contours of the residual-sum-of-squares function. The point $\widehat{\beta}$ depicts the usual (unconstrained) least-squares estimate.*

Table 2.2 *Results from analysis of the crime data. Left panel shows the least-squares estimates, standard errors, and their ratio (Z-score). Middle and right panels show the corresponding results for the lasso, and the least-squares estimates applied to the subset of predictors chosen by the lasso.*

	LS coef	SE	Z	Lasso	SE	Z	LS	SE	Z
`funding`	10.98	3.08	3.6	8.84	3.55	2.5	11.29	2.90	3.9
`hs`	-6.09	6.54	-0.9	-1.41	3.73	-0.4	-4.76	4.53	-1.1
`not-hs`	5.48	10.05	0.5	3.12	5.05	0.6	3.44	7.83	0.4
`college`	0.38	4.42	0.1	0.0	-	-	0.0	-	-
`college4`	5.50	13.75	0.4	0.0	-	-	0.0	-	-

of squares has elliptical contours, centered at the full least-squares estimates. The constraint region for ridge regression is the disk $\beta_1^2 + \beta_2^2 \leq t^2$, while that for lasso is the diamond $|\beta_1| + |\beta_2| \leq t$. Both methods find the first point where the elliptical contours hit the constraint region. Unlike the disk, the diamond has corners; if the solution occurs at a corner, then it has one parameter β_j equal to zero. When $p > 2$, the diamond becomes a rhomboid, and has many corners, flat edges, and faces; there are many more opportunities for the estimated parameters to be zero (see Figure 4.2 on page 58.)

We use the term *sparse* for a model with few nonzero coefficients. Hence a key property of the ℓ_1-constraint is its ability to yield sparse solutions. This idea can be applied in many different statistical models, and is the central theme of this book.

Table 2.2 shows the results of applying three fitting procedures to the crime data. The lasso bound t was chosen by cross-validation, as described in Section 2.3. The left panel corresponds to the full least-squares fit, while the middle panel shows the lasso fit. On the right, we have applied least-squares estimation to the subset of three predictors with nonzero coefficients in the lasso. The standard errors for the least-squares estimates come from the usual formulas. No such simple formula exists for the lasso, so we have used the bootstrap to obtain the estimate of standard errors in the middle panel (see Exercise 2.6; Chapter 6 discusses some promising new approaches for post-selection inference). Overall it appears that `funding` has a large effect, probably indicating that police resources have been focused on higher crime areas. The other predictors have small to moderate effects.

Note that the lasso sets two of the five coefficients to zero, and tends to shrink the coefficients of the others toward zero relative to the full least-squares estimate. In turn, the least-squares fit on the subset of the three predictors tends to expand the lasso estimates away from zero. The nonzero estimates from the lasso tend to be biased toward zero, so the debiasing in the right panel can often improve the prediction error of the model. This two-stage process is also known as the *relaxed lasso* (Meinshausen 2007).

2.3 Cross-Validation and Inference

The bound t in the lasso criterion (2.3) controls the complexity of the model; larger values of t free up more parameters and allow the model to adapt more closely to the training data. Conversely, smaller values of t restrict the parameters more, leading to sparser, more interpretable models that fit the data less closely. Forgetting about interpretability, we can ask for the value of t that gives the most accurate model for predicting independent test data from the same population. Such accuracy is called the *generalization* ability of the model. A value of t that is too small can prevent the lasso from capturing the main signal in the data, while too large a value can lead to overfitting. In this latter case, the model adapts to the noise as well as the signal that is present in the training data. In both cases, the prediction error on a test set will be inflated. There is usually an intermediate value of t that strikes a good balance between these two extremes, and in the process, produces a model with some coefficients equal to zero.

In order to estimate this best value for t, we can create artificial training and test sets by splitting up the given dataset at random, and estimating performance on the test data, using a procedure known as *cross-validation*. In more detail, we first randomly divide the full dataset into some number of groups $K > 1$. Typical choices of K might be 5 or 10, and sometimes N. We fix one group as the test set, and designate the remaining $K - 1$ groups as the training set. We then apply the lasso to the training data for a range of different t values, and we use each fitted model to predict the responses in the test set, recording the mean-squared prediction errors for each value of t. This process is repeated a total of K times, with each of the K groups getting the chance to play the role of the test data, with the remaining $K-1$ groups used as training data. In this way, we obtain K different estimates of the prediction error over a range of values of t. These K estimates of prediction error are averaged for each value of t, thereby producing a *cross-validation error curve*.

Figure 2.3 shows the cross-validation error curve for the crime-data example, obtained using $K = 10$ splits. We plot the estimated mean-squared prediction error versus the relative bound $\tilde{t} = \|\widehat{\beta}(t)\|_1/\|\tilde{\beta}\|_1$, where the estimate $\widehat{\beta}(t)$ correspond to the lasso solution for bound t and $\tilde{\beta}$ is the ordinary least-squares solution. The error bars in Figure 2.3 indicate plus and minus one standard error in the cross-validated estimates of the prediction error. A vertical dashed line is drawn at the position of the minimum ($\tilde{t} = 0.56$) while a dotted line is drawn at the "one-standard-error rule" choice ($\tilde{t} = 0.03$). This is the smallest value of t yielding a CV error no more than one standard error above its minimum value. The number of nonzero coefficients in each model is shown along the top. Hence the model that minimizes the CV error has three predictors, while the one-standard-error-rule model has just one.

We note that the cross-validation process above focused on the bound parameter t. One can just as well carry out cross-validation in the Lagrangian

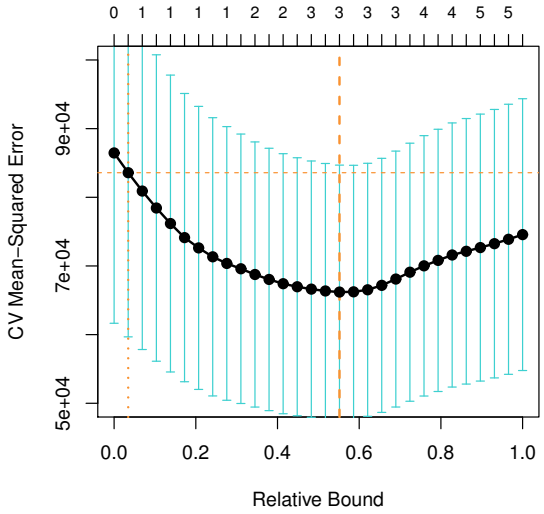


Figure 2.3 *Cross-validated estimate of mean-squared prediction error, as a function of the relative ℓ_1 bound $\tilde{t} = \|\widehat{\beta}(t)\|_1/\|\tilde{\beta}\|_1$. Here $\widehat{\beta}(t)$ is the lasso estimate corresponding to the ℓ_1 bound t and $\tilde{\beta}$ is the ordinary least-squares solution. Included are the location of the minimum, pointwise standard-error bands, and the "one-standard-error" location. The standard errors are large since the sample size N is only 50.*

form (2.5), focusing on the parameter λ. The two methods will give similar but not identical results, since the mapping between t and λ is data-dependent.

2.4 Computation of the Lasso Solution

The lasso problem is a convex program, specifically a quadratic program (QP) with a convex constraint. As such, there are many sophisticated QP methods for solving the lasso. However there is a particularly simple and effective computational algorithm, that gives insight into how the lasso works. For convenience, we rewrite the criterion in Lagrangian form:

$$\underset{\beta\in\mathbb{R}^p}{\text{minimize}} \left\{ \frac{1}{2N}\sum_{i=1}^{N}(y_i - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda\sum_{j=1}^{p}|\beta_j| \right\}. \tag{2.8}$$

As before, we will assume that both y_i and the features x_{ij} have been standardized so that $\frac{1}{N}\sum_i y_i = 0$, $\frac{1}{N}\sum_i x_{ij} = 0$, and $\frac{1}{N}\sum_i x_{ij}^2 = 1$. In this case, the intercept term β_0 can be omitted. The Lagrangian form is especially convenient for numerical computation of the solution by a simple procedure known as *coordinate descent*.

2.4.1 Single Predictor: Soft Thresholding

Let's first consider a single predictor setting, based on samples $\{(z_i, y_i)\}_{i=1}^N$ (for convenience we have given the name z_i to this single x_{i1}). The problem then is to solve

$$\underset{\beta}{\text{minimize}} \left\{ \frac{1}{2N} \sum_{i=1}^{N} (y_i - z_i\beta)^2 + \lambda|\beta| \right\}. \tag{2.9}$$

The standard approach to this univariate minimization problem would be to take the gradient (first derivative) with respect to β, and set it to zero. There is a complication, however, because the absolute value function $|\beta|$ does not have a derivative at $\beta = 0$. However we can proceed by direct inspection of the function (2.9), and find that

$$\widehat{\beta} = \begin{cases} \frac{1}{N}\langle \mathbf{z}, \mathbf{y} \rangle - \lambda & \text{if } \frac{1}{N}\langle \mathbf{z}, \mathbf{y} \rangle > \lambda, \\ 0 & \text{if } \frac{1}{N}|\langle \mathbf{z}, \mathbf{y} \rangle| \leq \lambda, \\ \frac{1}{N}\langle \mathbf{z}, \mathbf{y} \rangle + \lambda & \text{if } \frac{1}{N}\langle \mathbf{z}, \mathbf{y} \rangle < -\lambda. \end{cases} \tag{2.10}$$

(Exercise 2.2), which we can write succinctly as

$$\widehat{\beta} = \mathcal{S}_\lambda\big(\tfrac{1}{N}\langle \mathbf{z}, \mathbf{y} \rangle\big). \tag{2.11}$$

Here the *soft-thresholding operator*

$$\mathcal{S}_\lambda(x) = \text{sign}(x)\left(|x| - \lambda\right)_+ \tag{2.12}$$

translates its argument x toward zero by the amount λ, and sets it to zero if $|x| \leq \lambda$.[3] See Figure 2.4 for an illustration. Notice that for standardized data with $\frac{1}{N}\sum_i z_i^2 = 1$, (2.11) is just a soft-thresholded version of the usual least-squares estimate $\tilde{\beta} = \frac{1}{N}\langle \mathbf{z}, \mathbf{y} \rangle$. One can also derive these results using the notion of subgradients (Exercise 2.3).

2.4.2 Multiple Predictors: Cyclic Coordinate Descent

Using this intuition from the univariate case, we can now develop a simple coordinatewise scheme for solving the full lasso problem (2.5). More precisely, we repeatedly cycle through the predictors in some fixed (but arbitrary) order (say $j = 1, 2, \ldots, p$), where at the j^{th} step, we update the coefficient β_j by minimizing the objective function in this coordinate while holding fixed all other coefficients $\{\widehat{\beta}_k, k \neq j\}$ at their current values.

Writing the objective in (2.5) as

$$\frac{1}{2N} \sum_{i=1}^{N} (y_i - \sum_{k \neq j} x_{ik}\beta_k - x_{ij}\beta_j)^2 + \lambda \sum_{k \neq j} |\beta_k| + \lambda|\beta_j|, \tag{2.13}$$

[3] t_+ denotes the positive part of $t \in \mathbb{R}$, equal to t if $t > 0$ and 0 otherwise.

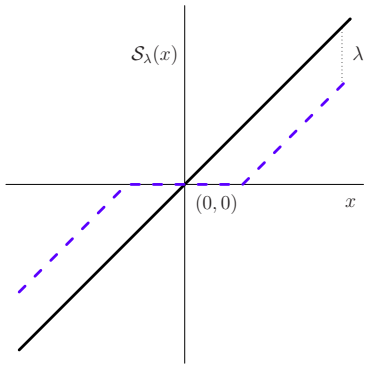


Figure 2.4 *Soft thresholding function* $\mathcal{S}_\lambda(x) = \text{sign}(x)\,(|x| - \lambda)_+$ *is shown in blue (broken lines), along with the* 45° *line in black.*

we see that solution for each β_j can be expressed succinctly in terms of the *partial residual* $r_i^{(j)} = y_i - \sum_{k \neq j} x_{ik}\widehat{\beta}_k$, which removes from the outcome the current fit from all but the j^{th} predictor. In terms of this partial residual, the j^{th} coefficient is updated as

$$\widehat{\beta}_j = \mathcal{S}_\lambda\Big(\tfrac{1}{N}\langle \mathbf{x}_j, \boldsymbol{r}^{(j)}\rangle\Big). \tag{2.14}$$

Equivalently, the update can be written as

$$\widehat{\beta}_j \leftarrow \mathcal{S}_\lambda\Big(\widehat{\beta}_j + \tfrac{1}{N}\langle \mathbf{x}_j, \boldsymbol{r}\rangle\Big), \tag{2.15}$$

where $r_i = y_i - \sum_{j=1}^p x_{ij}\widehat{\beta}_j$ are the full residuals (Exercise 2.4). The overall algorithm operates by applying this soft-thresholding update (2.14) repeatedly in a cyclical manner, updating the coordinates of $\widehat{\beta}$ (and hence the residual vectors) along the way.

Why does this algorithm work? The criterion (2.5) is a convex function of β and so has no local minima. The algorithm just described corresponds to the method of *cyclical coordinate descent*, which minimizes this convex objective along each coordinate at a time. Under relatively mild conditions (which apply here), such coordinate-wise minimization schemes applied to a convex function converge to a global optimum. It is important to note that some conditions are required, because there are instances, involving nonseparable penalty functions, in which coordinate descent schemes can become "jammed." Further details are in given in Chapter 5.

Note that the choice $\lambda = 0$ in (2.5) delivers the solution to the ordinary least-squares problem. From the update (2.14), we see that the algorithm does a univariate regression of the partial residual onto each predictor, cycling through the predictors until convergence. When the data matrix $\mathbf{X}$ is of full

rank, this point of convergence is the least-squares solution. However, it is not a particularly efficient method for computing it.

In practice, one is often interested in finding the lasso solution not just for a single fixed value of λ, but rather the entire path of solutions over a range of possible λ values (as in Figure 2.1). A reasonable method for doing so is to begin with a value of λ just large enough so that the only optimal solution is the all-zeroes vector. As shown in Exercise 2.1, this value is equal to $\lambda_{max} = \max_j |\frac{1}{N}\langle \mathbf{x}_j, \mathbf{y}\rangle|$. Then we decrease λ by a small amount and run coordinate descent until convergence. Decreasing λ again and using the previous solution as a "warm start," we then run coordinate descent until convergence. In this way we can efficiently compute the solutions over a grid of λ values. We refer to this method as *pathwise coordinate descent*.

Coordinate descent is especially fast for the lasso because the coordinate-wise minimizers are explicitly available (Equation (2.14)), and thus an iterative search along each coordinate is not needed. Secondly, it exploits the sparsity of the problem: for large enough values of λ most coefficients will be zero and will not be moved from zero. In Section 5.4, we discuss computational hedges for guessing the active set, which speed up the algorithm dramatically.

Homotopy methods are another class of techniques for solving the lasso. They produce the entire path of solutions in a sequential fashion, starting at zero. This path is actually piecewise linear, as can be seen in Figure 2.1 (as a function of t or λ). The *least angle regression* (LARS) algorithm is a homotopy method that efficiently constructs the piecewise linear path, and is described in Chapter 5.

2.4.3 Soft-Thresholding and Orthogonal Bases

The soft-thresholding operator plays a central role in the lasso and also in signal denoising. To see this, notice that the coordinate minimization scheme above takes an especially simple form if the predictors are orthogonal, meaning that $\frac{1}{N}\langle \mathbf{x}_j, \mathbf{x}_k\rangle = 0$ for each $j \neq k$. In this case, the update (2.14) simplifies dramatically, since $\frac{1}{N}\langle \mathbf{x}_j, \boldsymbol{r}^{(j)}\rangle = \frac{1}{N}\langle \mathbf{x}_j, \mathbf{y}\rangle$ so that $\widehat{\beta}_j$ is simply the soft-thresholded version of the univariate least-squares estimate of $\mathbf{y}$ regressed against $\mathbf{x}_j$. Thus, in the special case of an orthogonal design, the lasso has an explicit closed-form solution, and no iterations are required.

Wavelets are a popular form of orthogonal bases, used for smoothing and compression of signals and images. In wavelet smoothing one represents the data in a wavelet basis, and then denoises by soft-thresholding the wavelet coefficients. We discuss this further in Section 2.10 and in Chapter 10.

2.5 Degrees of Freedom

Suppose we have p predictors, and fit a linear regression model using only a subset of k of these predictors. Then if these k predictors were chosen without

regard to the response variable, the fitting procedure "spends" k degrees of freedom. This is a loose way of saying that the standard test statistic for testing the hypothesis that all k coefficients are zero has a Chi-squared distribution with k degrees of freedom (with the error variance σ^2 assumed to be known)

However if the k predictors were chosen using knowledge of the response variable, for example to yield the smallest training error among all subsets of size k, then we would expect that the fitting procedure spends more than k degrees of freedom. We call such a fitting procedure *adaptive*, and clearly the lasso is an example of one.

Similarly, a forward-stepwise procedure in which we sequentially add the predictor that most decreases the training error is adaptive, and we would expect that the resulting model uses more than k degrees of freedom after k steps. For these reasons and in general, one cannot simply count as degrees of freedom the number of nonzero coefficients in the fitted model. However, it turns out that for the lasso, one *can* count degrees of freedom by the number of nonzero coefficients, as we now describe.

First we need to define precisely what we mean by the degrees of freedom of an adaptively fitted model. Suppose we have an additive-error model, with

$$y_i = f(x_i) + \epsilon_i,\ i = 1, \ldots, N, \tag{2.16}$$

for some unknown f and with the errors ϵ_i iid $(0, \sigma^2)$. If the N sample predictions are denoted by $\widehat{\mathbf{y}}$, then we define

$$\mathrm{df}(\widehat{\mathbf{y}}) := \frac{1}{\sigma^2} \sum_{i=1}^{N} \mathrm{Cov}\left(\widehat{y}_i, y_i\right). \tag{2.17}$$

The covariance here is taken over the randomness in the response variables $\{y_i\}_{i=1}^N$ with the predictors held fixed. Thus, the degrees of freedom corresponds to the total amount of *self-influence* that each response measurement has on its prediction. The more the model fits—that is, adapts—to the data, the larger the degrees of freedom. In the case of a fixed linear model, using k predictors chosen independently of the response variable, it is easy to show that $\mathrm{df}(\widehat{\mathbf{y}}) = k$ (Exercise 2.7). However, under adaptive fitting, it is typically the case that the degrees of freedom is larger than k.

Somewhat miraculously, one can show that for the lasso, with a fixed penalty parameter λ, the number of nonzero coefficients k_λ is an unbiased estimate of the degrees of freedom[4] (Zou, Hastie and Tibshirani 2007, Tibshirani$_2$ and Taylor 2012). As discussed earlier, a variable-selection method like forward-stepwise regression uses more than k degrees of freedom after k steps. Given the apparent similarity between forward-stepwise regression and the lasso, how can the lasso have this simple degrees of freedom property? The

[4] An even stronger statement holds for the LAR path, where the degrees of freedom after k steps is exactly k, under some conditions on $\mathbf{X}$. The LAR path relates closely to the lasso, and is described in Section 5.6.

reason is that the lasso not only selects predictors (which inflates the degrees of freedom), but also shrinks their coefficients toward zero, relative to the usual least-squares estimates. This shrinkage turns out to be just the right amount to bring the degrees of freedom down to k. This result is useful because it gives us a qualitative measure of the amount of fitting that we have done at any point along the lasso path.

In the general setting, a proof of this result is quite difficult. In the special case of an orthogonal design, it is relatively easy to prove, using the fact that the lasso estimates are simply soft-thresholded versions of the univariate regression coefficients for the orthogonal design. We explore the details of this argument in Exercise 2.8. This idea is taken one step further in Section 6.3.1 where we describe the *covariance test* for testing the significance of predictors in the context of the lasso.

2.6 Uniqueness of the Lasso Solutions

We first note that the theory of convex duality can be used to show that when the columns of $\mathbf{X}$ are in general position, then for $\lambda > 0$ the solution to the lasso problem (2.5) is unique. This holds even when $p \geq N$, although then the number of nonzero coefficients in any lasso solution is at most N (Rosset, Zhu and Hastie 2004, Tibshirani$_2$ 2013). Now when the predictor matrix $\mathbf{X}$ is not of full column rank, the least squares fitted values are unique, but the parameter estimates themselves are not. The non-full-rank case can occur when $p \leq N$ due to collinearity, and always occurs when $p > N$. In the latter scenario, there are an infinite number of solutions $\widehat{\beta}$ that yield a perfect fit with zero training error. Now consider the lasso problem in Lagrange form (2.5) for $\lambda > 0$. As shown in Exercise 2.5, the fitted values $\mathbf{X}\widehat{\beta}$ are unique. But it turns out that the solution $\widehat{\beta}$ may not be unique. Consider a simple example with two predictors $\mathbf{x}_1$ and $\mathbf{x}_2$ and response $\mathbf{y}$, and suppose the lasso solution coefficients $\widehat{\beta}$ at λ are $(\widehat{\beta}_1,\ \widehat{\beta}_2)$. If we now include a third predictor $\mathbf{x}_3 = \mathbf{x}_2$ into the mix, an identical copy of the second, then for any $\alpha \in [0, 1]$, the vector $\tilde{\beta}(\alpha) = (\widehat{\beta}_1,\ \alpha \cdot \widehat{\beta}_2,\ (1-\alpha) \cdot \widehat{\beta}_2)$ produces an identical fit, and has ℓ_1 norm $\|\tilde{\beta}(\alpha)\|_1 = \|\widehat{\beta}\|_1$. Consequently, for this model (in which we might have either $p \leq N$ or $p > N$), there is an infinite family of solutions.

In general, when $\lambda > 0$, one can show that if the columns of the model matrix $\mathbf{X}$ are in *general position*, then the lasso solutions are unique. To be precise, we say the columns $\{\mathbf{x}_j\}_{j=1}^p$ are in general position if any affine subspace $\mathbb{L} \subset \mathbb{R}^N$ of dimension $k < N$ contains at most $k + 1$ elements of the set $\{\pm\mathbf{x}_1, \pm\mathbf{x}_2, \ldots \pm \mathbf{x}_p\}$, excluding antipodal pairs of points (that is, points differing only by a sign flip). We note that the data in the example in the previous paragraph are not in general position. If the X data are drawn from a continuous probability distribution, then with probability one the data are in general position and hence the lasso solutions will be unique. As a result, non-uniqueness of the lasso solutions can only occur with discrete-valued data, such as those arising from dummy-value coding of categorical predic-

tors. These results have appeared in various forms in the literature, with a summary given by Tibshirani$_2$ (2013).

We note that numerical algorithms for computing solutions to the lasso will typically yield valid solutions in the non-unique case. However, the particular solution that they deliver can depend on the specifics of the algorithm. For example with coordinate descent, the choice of starting values can affect the final solution.

2.7 A Glimpse at the Theory

There is a large body of theoretical work on the behavior of the lasso. It is largely focused on the mean-squared-error consistency of the lasso, and recovery of the nonzero support set of the true regression parameters, sometimes called *sparsistency*. For MSE consistency, if β^* and $\hat{\beta}$ are the true and lasso-estimated parameters, it can be shown that as $p, n \to \infty$

$$\|\mathbf{X}(\hat{\beta} - \beta^*)\|_2^2/N \leq C \cdot \|\beta^*\|_1 \sqrt{\log(p)/N} \tag{2.18}$$

with high probability (Greenshtein and Ritov 2004, Bühlmann and van de Geer 2011, Chapter 6). Thus if $\|\beta^*\|_1 = o(\sqrt{N/\log(p)})$ then the lasso is consistent for prediction. This means that the true parameter vector must be sparse relative to the ratio $N/\log(p)$. The result only assumes that the design $\mathbf{X}$ is fixed and has no other conditions on $\mathbf{X}$. Consistent recovery of the nonzero support set requires more stringent assumptions on the level of cross-correlation between the predictors inside and outside of the support set. Details are given in Chapter 11.

2.8 The Nonnegative Garrote

The *nonnegative garrote* (Breiman 1995)[5] is a two-stage procedure, with a close relationship to the lasso.[6] Given an initial estimate of the regression coefficients $\tilde{\beta} \in \mathbb{R}^p$, we then solve the optimization problem

$$\begin{aligned} &\underset{c \in \mathbb{R}^p}{\text{minimize}} \left\{ \sum_{i=1}^{N} \Big(y_i - \sum_{j=1}^{p} c_j x_{ij} \tilde{\beta}_j\Big)^2 \right\} \\ &\text{subject to } c \succeq 0 \text{ and } \|c\|_1 \leq t, \end{aligned} \tag{2.19}$$

where $c \succeq 0$ means the vector has nonnegative coordinates. Finally, we set $\widehat{\beta}_j = \hat{c}_j \cdot \tilde{\beta}_j,\ j = 1, \ldots, p$. There is an equivalent Lagrangian form for this procedure, using a penalty $\lambda\|c\|_1$ for some regularization weight $\lambda \geq 0$, plus the nonnegativity constraints.

[5] A garrote is a device used for execution by strangulation or by breaking the neck. It is a Spanish word, and is alternately spelled *garrotte* or *garotte*. We are using the spelling in the original paper of Breiman (1995).

[6] Breiman's paper was the inspiration for Tibshirani's 1996 lasso paper.

In the original paper (Breiman 1995), the initial $\tilde{\beta}$ was chosen to be the ordinary-least-squares (OLS) estimate. Of course, when $p > N$, these estimates are not unique; since that time, other authors (Yuan and Lin 2007*b*, Zou 2006) have shown that the nonnegative garrote has attractive properties when we use other initial estimators such as the lasso, ridge regression or the elastic net.

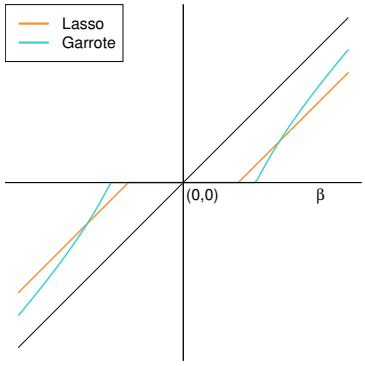


Figure 2.5 *Comparison of the shrinkage behavior of the lasso and the nonnegative garrote for a single variable. Since their λs are on different scales, we used 2 for the lasso and 7 for the garrote to make them somewhat comparable. The garrote shrinks smaller values of β more severely than lasso, and the opposite for larger values.*

The nature of the nonnegative garrote solutions can be seen when the columns of $\mathbf{X}$ are orthogonal. Assuming that t is in the range where the equality constraint $\|c\|_1 = t$ can be satisfied, the solutions have the explicit form

$$\widehat{c}_j = \left(1 - \frac{\lambda}{\tilde{\beta}_j^2}\right)_+, \; j = 1, \ldots, p, \tag{2.20}$$

where λ is chosen so that $\|\hat{c}\|_1 = t$. Hence if the coefficient $\tilde{\beta}_j$ is large, the shrinkage factor will be close to 1 (no shrinkage), but if it is small the estimate will be shrunk toward zero. Figure 2.5 compares the shrinkage behavior of the lasso and nonnegative garrote. The latter exhibits the shrinkage behavior of the nonconvex penalties (next section and Section 4.6). There is also a close relationship between the nonnegative garrote and the *adaptive lasso*, discussed in Section 4.6; see Exercise 4.26.

Following this, Yuan and Lin (2007*b*) and Zou (2006) have shown that the nonnegative garrote is *path-consistent* under less stringent conditions than the lasso. This holds if the initial estimates are $\sqrt{N}$-consistent, for example those based on least squares (when $p < N$), the lasso, or the elastic net. "Path-consistent" means that the solution path contains the true model somewhere in its path indexed by t or λ. On the other hand, the convergence of the parameter estimates from the nonnegative garrote tends to be slower than that of the initial estimate.

Table 2.3 *Estimators of β_j from (2.21) in the case of an orthonormal model matrix* $\mathbf{X}$.

q	Estimator	Formula
0	Best subset	$\tilde{\beta}_j \cdot \mathbb{I}[\lvert\tilde{\beta}_j\rvert > \sqrt{2\lambda}]$
1	Lasso	$\text{sign}(\tilde{\beta}_j)(\lvert\tilde{\beta}_j\rvert - \lambda)_+$
2	Ridge	$\tilde{\beta}_j/(1+\lambda)$

2.9 ℓ_q Penalties and Bayes Estimates

For a fixed real number $q \geq 0$, consider the criterion

$$\underset{\beta \in \mathbb{R}^p}{\text{minimize}} \left\{ \frac{1}{2N} \sum_{i=1}^{N} (y_i - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} |\beta_j|^q \right\}. \tag{2.21}$$

This is the lasso for $q = 1$ and ridge regression for $q = 2$. For $q = 0$, the term $\sum_{j=1}^{p} |\beta_j|^q$ counts the number of nonzero elements in β, and so solving (2.21) amounts to best-subset selection. Figure 2.6 displays the constraint regions corresponding to these penalties for the case of two predictors ($p = 2$). Both

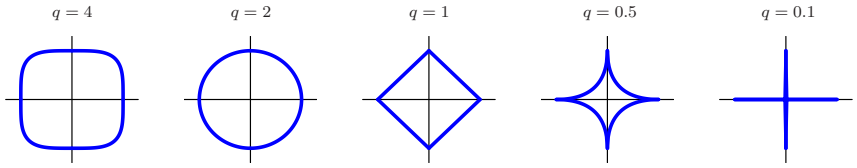


Figure 2.6 *Constraint regions $\sum_{j=1}^{p} |\beta_j|^q \leq 1$ for different values of q. For $q < 1$, the constraint region is nonconvex.*

the lasso and ridge regression versions of (2.21) amount to solving convex programs, and so scale well to large problems. Best subset selection leads to a nonconvex and combinatorial optimization problem, and is typically not feasible with more than say $p = 50$ predictors.

In the special case of an orthonormal model matrix $\mathbf{X}$, all three procedures have explicit solutions. Each method applies a simple coordinate-wise transformation to the least-squares estimate $\tilde{\beta}$, as detailed in Table 2.9. Ridge regression does a proportional shrinkage. The lasso translates each coefficient by a constant factor λ and truncates at zero, otherwise known as soft thresholding. Best-subset selection applies the hard thresholding operator: it leaves the coefficient alone if it is bigger than $\sqrt{2\lambda}$, and otherwise sets it to zero.

The lasso is special in that the choice $q = 1$ is the smallest value of q (closest to best-subset) that leads to a convex constraint region and hence a

convex optimization problem. In this sense, it is the closest convex relaxation of the best-subset selection problem.

There is also a Bayesian view of these estimators. Thinking of $|\beta_j|^q$ as proportional to the negative log-prior density for β_j, the constraint contours represented in Figure 2.6 have the same shape as the equi-contours of the prior distribution of the parameters. Notice that for $q \leq 1$, the prior concentrates more mass in the coordinate directions. The prior corresponding to the $q = 1$ case is an independent double exponential (or Laplace) distribution for each parameter, with joint density $(1/2\tau)\exp(-\|\beta\|_1)/\tau)$ and $\tau = 1/\lambda$. This means that the lasso estimate is the Bayesian MAP (maximum *aposteriori*) estimator using a Laplacian prior, as opposed to the mean of the posterior distribution, which is not sparse. Similarly, if we sample from the posterior distribution corresponding to the Laplace prior, we do not obtain sparse vectors. In order to obtain sparse vectors via posterior sampling, one needs to start with a prior distribution that puts a point mass at zero. Bayesian approaches to the lasso are explored in Section 6.1.

2.10 Some Perspective

The lasso uses an ℓ_1-penalty, and such penalties are now widely used in statistics, machine learning, engineering, finance, and other fields. The lasso was proposed by Tibshirani (1996), and was directly inspired by the nonnegative garrote of Breiman (1995). Soft thresholding was popularized earlier in the context of wavelet filtering by Donoho and Johnstone (1994); this is a popular alternative to Fourier filtering in signal processing, being both "local in time and frequency." Since wavelet bases are orthonormal, wavelet filtering corresponds to the lasso in the orthogonal $\mathbf{X}$ case (Section 2.4.1). Around the same time as the advent of the lasso, Chen, Donoho and Saunders (1998) proposed the closely related *basis pursuit* method, which extends the ideas of wavelet filtering to search for a sparse representation of a signal in over-complete bases using an ℓ_1-penalty. These are unions of orthonormal frames and hence no longer completely mutually orthonormal.

Taking a broader perspective, ℓ_1-regularization has a pretty lengthy history. For example Donoho and Stark (1989) discussed ℓ_1-based recovery in detail, and provided some guarantees for incoherent bases. Even earlier (and mentioned in Donoho and Stark (1989)) there are related works from the 1980s in the geosciences community, for example Oldenburg, Scheuer and Levy (1983) and Santosa and Symes (1986). In the signal processing world, Alliney and Ruzinsky (1994) investigated some algorithmic issues associated with ℓ_1 regularization. And there surely are many other authors who have proposed similar ideas, such as Fuchs (2000). Rish and Grabarnik (2014) provide a modern introduction to sparse methods for machine learning and signal processing.

In the last 10–15 years, it has become clear that the ℓ_1-penalty has a number of good properties, which can be summarized as follows:

Interpretation of the final model: The ℓ_1-penalty provides a natural way to encourage or enforce sparsity and simplicity in the solution.

Statistical efficiency: In the book *The Elements of Statistical Learning* (Hastie, Tibshirani and Friedman 2009), the authors discuss an informal "*bet-on-sparsity principle*." Assume that the underlying true signal is sparse and we use an ℓ_1 penalty to try to recover it. If our assumption is correct, we can do a good job in recovering the true signal. Note that sparsity can hold in the given bases (set of features) or a transformation of the features (e.g., a wavelet bases). But if we are wrong—the underlying truth is not sparse in the chosen bases—then the ℓ_1 penalty will not work well. However in that instance, no method can do well, relative to the Bayes error. There is now a large body of theoretical support for these loose statements: see Chapter 11 for some results.

Computational efficiency: ℓ_1-based penalties are convex and this fact and the assumed sparsity can lead to significant computational advantages. If we have 100 observations and one million features, and we have to estimate one million nonzero parameters, then the computation is very challenging. However, if we apply the lasso, then at most 100 parameters can be nonzero in the solution, and this makes the computation much easier. More details are given in Chapter 5.[7]

In the remainder of this book, we describe many of the exciting developments in this field.

Exercises

Ex. 2.1 Show that the smallest value of λ such that the regression coefficients estimated by the lasso are all equal to zero is given by

$$\lambda_{\max} = \max_j |\frac{1}{N}\langle \mathbf{x}_j, \mathbf{y}\rangle|.$$

Ex. 2.2 Show that the soft-threshold estimator (2.12) yields the solution to the single predictor lasso problem (2.9). (Do not make use of subgradients, and note that $\mathbf{z}$ is standardized).

Ex. 2.3 *Soft thresholding and subgradients.* Since (2.9) is a convex function, it is guaranteed to have a subgradient (see Chapter 5 for more details), and any optimal solution must satisfy the *subgradient* equation

$$-\frac{1}{N}\sum_{i=1}^{N}(y_i - z_i\beta)z_i + \lambda s = 0, \qquad \text{where } s \text{ is a subgradient of } |\beta|. \quad (2.22)$$

For the absolute value function, subgradients take the form $s \in \text{sign}(\beta)$, meaning that $s = \text{sign}(\beta)$ when $\beta \neq 0$ and $s \in [-1, +1]$ when $\beta = 0$. The general

[7] Ridge regression also enjoys a similar efficiency in the $p \gg N$ case.

theory of convex optimization, as discussed in Chapter 5, guarantees that any pair $(\widehat{\beta}, \widehat{s})$ that is a solution to the zero subgradient Equation (2.22) with $\widehat{s} \in \text{sign}(\widehat{\beta})$ defines an optimal solution to the original minimization problem (2.9).

Solve Equation (2.22) and hence arrive at solutions (2.10) and (2.11).

Ex. 2.4 Show that the subgradient equations for Problem (2.5) take the form given in (2.6). Hence derive expressions for coordinate descent steps (2.14) and (2.15).

Ex. 2.5 *Uniqueness of fitted values from the lasso.* For some $\lambda \geq 0$, suppose that we have two lasso solutions $\widehat{\beta}, \widehat{\gamma}$ with common optimal value c^*.

(a) Show that it must be the case that $\mathbf{X}\widehat{\beta} = \mathbf{X}\widehat{\gamma}$, meaning that the two solutions must yield the same predicted values. (*Hint:* If not, then use the strict convexity of the function $f(\mathbf{u}) = \|\mathbf{y} - \mathbf{u}\|_2^2$ and convexity of the ℓ_1-norm to establish a contradiction.)

(b) If $\lambda > 0$, show that we must have $\|\widehat{\beta}\|_1 = \|\widehat{\gamma}\|_1$.

(Tibshirani$_2$ 2013).

Ex. 2.6 Here we use the bootstrap as the basis for inference with the lasso.

(a) For the crime data, apply the bootstrap to estimate the standard errors of the estimated lasso coefficients, as in the middle section of Table 2.2. Use the nonparametric bootstrap, sampling features and outcome values (x_i, y_i) with replacement from the observed data. Keep the bound t fixed at its estimated value from the original lasso fit. Estimate as well the probability that an estimated coefficient is zero.

(b) Repeat part (a), but now re-estimate $\hat{\lambda}$ for each bootstrap replication. Compare the results to those in part (a).

Ex. 2.7 Consider a fixed linear model based on k predictors and fit by least squares. Show that its degrees of freedom (2.17) is equal to k.

Ex. 2.8 *Degrees of freedom for lasso in the orthogonal case.* Suppose that $y_i = \beta_0 + \sum_j x_{ij}\beta_j + \epsilon_i$ where $\epsilon_i \sim N(0, \sigma^2)$, with the x_{ij} fixed (non-random). Assume that the features are centered and also assume they are uncorrelated, so that $\sum_i x_{ij}x_{ik} = 0$ for all j, k. Stein's lemma (Stein 1981) states that for $Y \sim N(\mu, \sigma^2)$ and all absolutely continuous functions g such that $\mathbb{E}|g'(Y)| < \infty$,

$$\mathbb{E}(g(Y)(Y - \mu)) = \sigma^2 \mathbb{E}(g'(Y)). \tag{2.23}$$

Use this to show that the degrees of freedom (2.17) for the lasso in the orthogonal case is equal to k, the number of nonzero estimated coefficients in the solution.

Ex. 2.9 Derive the solutions (2.20) to the nonnegative garrote criterion (2.19).

Ex. 2.10 *Robust regression view of lasso.* Consider a robust version of the standard linear regression problem, in which we wish to protect ourselves against perturbations of the features. In order to do so, we consider the min-max criterion

$$\underset{\beta}{\text{minimize}} \max_{\mathbf{\Delta} \in \mathcal{U}} \left\{ \frac{1}{2N} \|\mathbf{y} - (\mathbf{X} + \mathbf{\Delta})\beta\|_2^2 \right\}, \tag{2.24}$$

where the allowable perturbations $\mathbf{\Delta} := (\boldsymbol{\delta}_1, \ldots, \boldsymbol{\delta}_p)$ belong to the subset of $\mathbb{R}^{N \times p}$

$$\mathcal{U} := \Big\{ (\boldsymbol{\delta}_1, \boldsymbol{\delta}_2, \ldots \boldsymbol{\delta}_p) \;\mid\; \|\boldsymbol{\delta}_j\|_2 \le c_j \text{ for all } j = 1, 2, \ldots, p \Big\}. \tag{2.25}$$

Hence each feature value x_{ij} can be perturbed by a maximum amount c_j, with the ℓ_2-norm of the overall perturbation vector for that feature bounded by c_j. The perturbations for different features also act independently of one another. We seek the coefficients that minimize squared error under the "worst" allowable perturbation of the features. We assume that both $\mathbf{y}$ and the columns of $\mathbf{X}$ have been standardized, and have not included an intercept.

Show that the solution to this problem is equivalent to

$$\min_{\beta \in \mathbb{R}^p} \left\{ \frac{1}{2N} \|\mathbf{y} - \mathbf{X}\beta\|_2^2 + \sum_{j=1}^{p} c_j |\beta_j| \right\}. \tag{2.26}$$

In the special case $c_j = \lambda$ for all $j = 1, 2, \ldots, p$, we thus obtain the lasso, so that it can be viewed as a method for guarding against uncertainty in the measured predictor values, with more uncertainty leading to a greater amount of shrinkage. (See Xu, Caramanis and Mannor (2010) for further details.)

Ex. 2.11 *Robust regression and constrained optimization.* This exercise doesn't involve the lasso itself, but rather a related use of the ℓ_1-norm in regression. We consider the model

$$y_i = \sum_{j=1}^{p} x_{ij}\beta_j + \gamma_i + \epsilon_i$$

with $\epsilon_i \sim N(0, \sigma^2)$ and $\gamma_i,\ i = 1, 2, \ldots, N$ are unknown constants.

Let $\boldsymbol{\gamma} = (\gamma_1, \gamma_2, \ldots, \gamma_N)$ and consider minimization of

$$\underset{\beta \in \mathbb{R}^p, \boldsymbol{\gamma} \in \mathbb{R}^N}{\text{minimize}} \frac{1}{2} \sum_{i=1}^{N} (y_i - \sum_{j=1}^{p} x_{ij}\beta_j - \gamma_i)^2 + \lambda \sum_{1}^{N} |\gamma_i|. \tag{2.27}$$

The idea is that for each i, γ_i allows y_i to be an outlier; setting $\gamma_i = 0$ means that the observation is not deemed an outlier. The penalty term effectively limits the number of outliers.

(a) Show this problem is jointly convex in β and $\boldsymbol{\gamma}$.

(b) Consider Huber's loss function

$$\rho(t;\lambda) = \begin{cases} \lambda|t| - \lambda^2/2 & \text{if } |t| > \lambda \\ t^2/2 & \text{if } |t| \le \lambda. \end{cases} \tag{2.28}$$

This is a tapered squared-error loss; it is quadratic for $|t| \le \lambda$ but linear outside of that range, to reduce the effect of outliers on the estimation of β. With the scale parameter σ fixed at one, Huber's robust regression method solves

$$\underset{\beta \in \mathbb{R}^p}{\text{minimize}} \sum_{i=1}^{N} \rho(y_i - \sum_{j=1}^{p} x_{ij}\beta_j; \lambda). \tag{2.29}$$

Show that problems (2.27) and (2.29) have the same solutions $\widehat{\beta}$. (Antoniadis 2007, Gannaz 2007, She and Owen 2011).

Chapter 3

Generalized Linear Models

In Chapter 2, we focused exclusively on linear regression models fit by least squares. Such linear models are suitable when the response variable is quantitative, and ideally when the error distribution is Gaussian. However, other types of response arise in practice. For instance, binary variables can be used to indicate the presence or absence of some attribute (e.g., "cancerous" versus "normal" cells in a biological assay, or "clicked" versus "not clicked" in web browsing analysis); here the binomial distribution is more appropriate. Sometimes the response occurs as counts (e.g., number of arrivals in a queue, or number of photons detected); here the Poisson distribution might be called for. In this chapter, we discuss generalizations of simple linear models and the lasso that are suitable for such applications.

3.1 Introduction

With a binary response coded in the form $Y \in \{0, 1\}$, the linear logistic model is often used: it models the log-likelihood ratio as the linear combination

$$\log \frac{\Pr(Y = 1 \mid X = x)}{\Pr(Y = 0 \mid X = x)} = \beta_0 + \beta^T x, \tag{3.1}$$

where $X = (X_1, X_2, \ldots X_p)$ is a vector of predictors, $\beta_0 \in \mathbb{R}$ is an intercept term, and $\beta \in \mathbb{R}^p$ is a vector of regression coefficients. Inverting this transformation yields an expression for the conditional probability

$$\Pr(Y = 1 \mid X = x) = \frac{e^{\beta_0 + \beta^T x}}{1 + e^{\beta_0 + \beta^T x}}. \tag{3.2}$$

By inspection, without any restriction on the parameters (β_0, β), the model specifies probabilities lying in $(0, 1)$. We typically fit logistic models by maximizing the binomial log-likelihood of the data.

The logit transformation (3.1) of the conditional probabilities is an example of a *link function*. In general, a link function is a transformation of the conditional mean $\mathbb{E}[Y \mid X = x]$—in this case, the conditional probability that $Y = 1$—to a more natural scale on which the parameters can be fit without constraints. As another example, if the response Y represents counts, taking

values in $\{0, 1, 2, \ldots\}$, then we need to ensure that the conditional mean is positive. A natural choice is the log-linear model

$$\log \mathbb{E}[Y \mid X = x] = \beta_0 + \beta^T x, \tag{3.3}$$

with its log link function. Here we fit the parameters by maximizing the Poisson log-likelihood of the data.

The models (3.1) and (3.3) are both special cases of *generalized linear models* (McCullagh and Nelder 1989). These models describe the response variable using a member of the *exponential family*, which includes the Bernoulli, Poisson, and Gaussian as particular cases. A transformed version of the response mean $\mathbb{E}[Y \mid X = x]$ is then approximated by a linear model. In detail, if we use $\mu(x) = \mathbb{E}[Y \mid X = x]$ to denote the conditional mean of Y given $X = x$, then a GLM is based on a model of the form

$$g\big[\mu(x)\big] = \underbrace{\beta_0 + \beta^T x}_{\eta(x)}, \tag{3.4}$$

where $g : \mathbb{R} \to \mathbb{R}$ is a strictly monotonic link function. For example, for a binary response $Y \in \{0, 1\}$, the logistic regression model is based on the choices $\mu(x) = \Pr[Y = 1 \mid X = x]$ and $g(\mu) = \text{logit}(\mu) = \log\big(\mu/(1-\mu)\big)$. When the response variable is modeled as a Gaussian, the choices $\mu(x) = \beta_0 + \beta^T x$ and $g(\mu) = \mu$ recover the standard linear model, as discussed in the previous chapter.

Generalized linear models can also be used to model the multicategory responses that occur in many problems, including handwritten digit classification, speech-recognition, document classification, and cancer classification. The multinomial replaces the binomial distribution here, and we use a symmetric log-linear representation:

$$\Pr[Y = k \mid X = x] = \frac{e^{\beta_{0k} + \beta_k^T x}}{\sum_{\ell=1}^{K} e^{\beta_{0\ell} + \beta_\ell^T x}}. \tag{3.5}$$

Here there are K coefficients for each variable (one per class).

In this chapter, we discuss approaches to fitting generalized linear models that are based on maximizing the likelihood, or equivalently minimizing the negative log-likelihood along with an ℓ_1-penalty

$$\underset{\beta_0, \beta}{\text{minimize}} \left\{ -\frac{1}{N} \mathcal{L}(\beta_0, \beta; \mathbf{y}, \mathbf{X}) + \lambda \|\beta\|_1 \right\}. \tag{3.6}$$

Here $\mathbf{y}$ is the N-vector of outcomes and $\mathbf{X}$ is the $N \times p$ matrix of predictors, and the specific form of the log-likelihood $\mathcal{L}$ varies according to the GLM. In the special case of Gaussian responses and the standard linear model, we have $\mathcal{L}(\beta_0, \beta; \mathbf{y}, \mathbf{X}) = \frac{1}{2\sigma^2} \|\mathbf{y} - \beta_0 \mathbf{1} - \mathbf{X}\beta\|_2^2 + c$, where c is a constant independent of (β_0, β), so that the optimization problem (3.6) corresponds to the ordinary linear least-squares lasso.

Similar forms of ℓ_1-regularization are also useful for related models. With survival models, the response is the time to failure (death), with possible censoring if subjects are lost to followup. In this context, a popular choice is the Cox proportional hazards model, which takes the form

$$h(t \mid x) = h_0(t)e^{\beta^T x}. \tag{3.7}$$

Here $t \mapsto h(t \mid x)$ is the *hazard function* for an individual with covariates x: the value $h(t \mid x)$ corresponds to the instantaneous probability of failure at time $Y = t$, given survival up to time t. The function h_0 specifies the baseline hazard, corresponding to $x = 0$.

As another example, the support-vector machine (SVM) is a popular classifier in the machine-learning community. Here the goal is to predict a two-class response $y \in \{-1, +1\}$,[1] in the simplest case using a linear classification boundary of the form $f(x) = \beta_0 + \beta^T x$, with the predicted class given by $\text{sign}(f(x))$. Thus, the correctness of a given decision can be determined by checking whether or not the margin $yf(x)$ is positive. The traditional *soft-margin* linear SVM is fit by solving the optimization problem[2]

$$\underset{\beta_0,\beta}{\text{minimize}} \left\{ \frac{1}{N} \sum_{i=1}^{N} \underbrace{[1 - y_i f(x_i)]_+}_{\phi(y_i f(x_i))} + \lambda \|\beta\|_2^2 \right\}. \tag{3.8}$$

The first term, known as *hinge loss*, is designed to penalize the negative margins that represent incorrect classifications. In general, an optimal solution vector $\beta \in \mathbb{R}^p$ to the standard linear SVM (3.8) is not sparse, since the quadratic penalty has no sparsity-enforcing properties. However, replacing the quadratic penalty by the ℓ_1-norm $\|\beta\|_1$ leads to an ℓ_1 linear SVM, which does produce sparse solutions.

In the following sections, we discuss each of these models in more detail. In each case, we provide examples of their applications, discuss some of the issues that arise, as well as computational approaches for fitting the models.

3.2 Logistic Regression

Logistic regression has been popular in biomedical research for half a century, and has recently gained popularity for modeling a wider range of data. In the high-dimensional setting, in which the number of features p is larger than the sample size, it cannot be used without modification. When $p > N$, any linear model is over-parametrized, and regularization is needed to achieve a stable fit. Such high-dimensional models arise in various applications. For example, document classification problems can involve binary features (presence versus

[1] For SVMs, it is convenient to code the binary response via the sign function.

[2] This is not the most standard way to introduce the support vector machine. We discuss this topic in more detail in Section 3.6.

absence) over a predefined dictionary of $p = 20,000$ or more words and tokens. Another example is genome-wide association studies (GWAS), where we have genotype measurements at $p = 500,000$ or more "SNPs," and the response is typically the presence/absence of a disease. A SNP (pronounced "snip") is a single-nucleotide polymorphism, and is typically represented as a three-level factor with possible values $\{\mathrm{AA}, \mathrm{Aa}, \mathrm{aa}\}$, where "A" refers to the wild-type, and "a" the mutation.

When the response is binary, it is typically coded as 0/1. Attention then focuses on estimating the conditional probability $\Pr(Y = 1 \mid X = x) = \mathbb{E}[Y \mid X = x]$. Given the logistic model (3.1), the negative log likelihood with ℓ_1-regularization takes the form

$$
\begin{aligned}
&-\frac{1}{N}\sum_{i=1}^{N}\{y_i \log \Pr(Y=1 \mid x_i) + (1-y_i)\log \Pr(Y=0 \mid x_i)\} + \lambda\|\beta\|_1 \\
&\qquad = -\frac{1}{N}\sum_{i=1}^{N}\left\{y_i(\beta_0 + \beta^T x_i) - \log(1 + e^{\beta_0+\beta^T x_i})\right\} + \lambda\|\beta\|_1. \quad (3.9)
\end{aligned}
$$

In the machine-learning community, it is more common to code the response Y in terms of sign variables $\{-1, +1\}$ rather than $\{0, 1\}$ values; when using sign variables, the penalized (negative) log-likelihood has the form

$$
\frac{1}{N}\sum_{i=1}^{N}\log(1 + e^{-y_i f(x_i;\beta_0,\beta)}) + \lambda\|\beta\|_1, \quad (3.10)
$$

where $f(x_i;\beta_0,\beta) := \beta_0 + \beta^T x_i$. For a given covariate-response pair (x, y), the product $yf(x)$ is referred to as the *margin*: a positive margin means a correct classification, whereas a negative margin means an incorrect classification. From the form of the log-likelihood (3.10), we see that maximizing the likelihood amounts to minimizing a loss function monotone decreasing in the margins. We discuss the interplay of the margin and the penalty in Section 3.6.1.

3.2.1 Example: Document Classification

We illustrate ℓ_1-regularized logistic regression in a domain where it has gained popularity, namely document classification using the 20-Newsgroups corpus (Lang 1995). We use the particular feature set and class definition defined by Koh, Kim and Boyd (2007).[3] There are $N = 11,314$ documents and $p = 777,811$ features, with 52% in the positive class. Only 0.05% of the features are nonzero for any given document.

[3]The positive class consists of the 10 groups with names of the form `sci.*`, `comp.*` and `misc.forsale`, and the rest are the negative class. The feature set consists of trigrams, with message headers skipped, no stoplist, and features with less than two documents omitted.

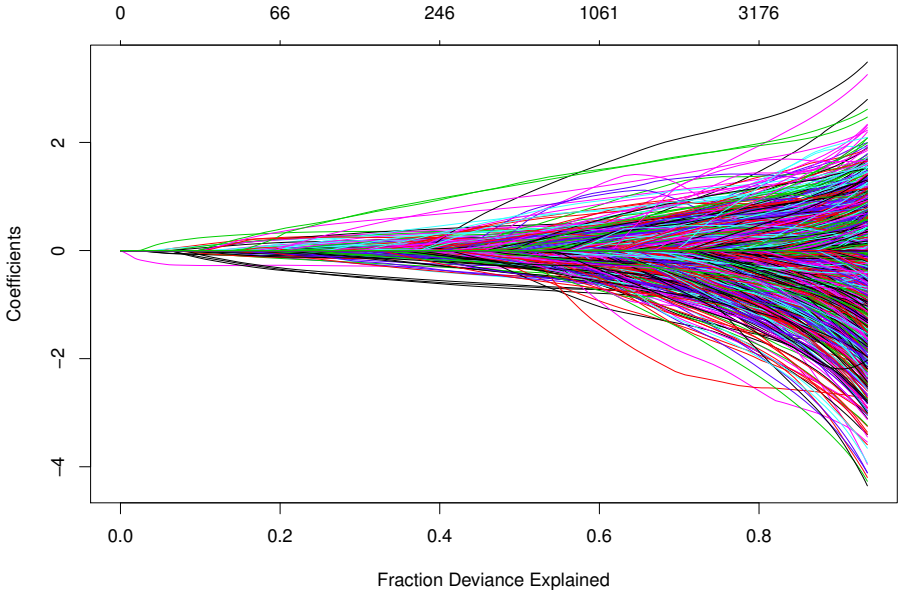


Figure 3.1 *Coefficient paths for an ℓ_1-regularized logistic regression for a document-classification task—the "NewsGroup" data. There are 11K documents roughly divided into two classes, and 0.78M features. Only 0.05% of the features are nonzero. The coefficients are plotted as a function of the fraction of null deviance explained.*

Figure 3.1 shows the coefficient profile, computed using the R package `glmnet`. Although the solutions were computed at 100 values of λ, uniformly spaced on the log scale, we have indexed the solutions by the *fraction of deviance explained*[4] on the training data:

$$\mathrm{D}^2{}_\lambda = \frac{\mathrm{Dev}_{\mathrm{null}} - \mathrm{Dev}_\lambda}{\mathrm{Dev}_{\mathrm{null}}}. \tag{3.11}$$

Here the deviance Dev_λ is defined as minus twice the difference in the log-likelihood for a model fit with parameter λ and the "saturated" model (having $\widehat{y} = y_i$). $\mathrm{Dev}_{\mathrm{null}}$ is the null deviance computed at the constant (mean) model. Since for these data the classes are separable, the range of λ is chosen so as not to get too close to the saturated fit (where the coefficients would be undefined; see the next section).

The maximum number of nonzero coefficients in any of these models can be shown to be $\min(N, p)$, which is equal $11,314$ in this case. In Figure 3.1, the largest model actually had only $5,277$ coefficients since `glmnet` did not go to the very end of the solution path. Although it might seem more natural to plot against the $\log(\lambda)$ sequence, or perhaps $\|\hat{\beta}(\lambda)\|_1$, there are problems with both in the $p \gg N$ setting. The former quantity is data and problem dependent,

[4]the name D^2 is by analogy with R^2, the fraction of variance explained in regression.

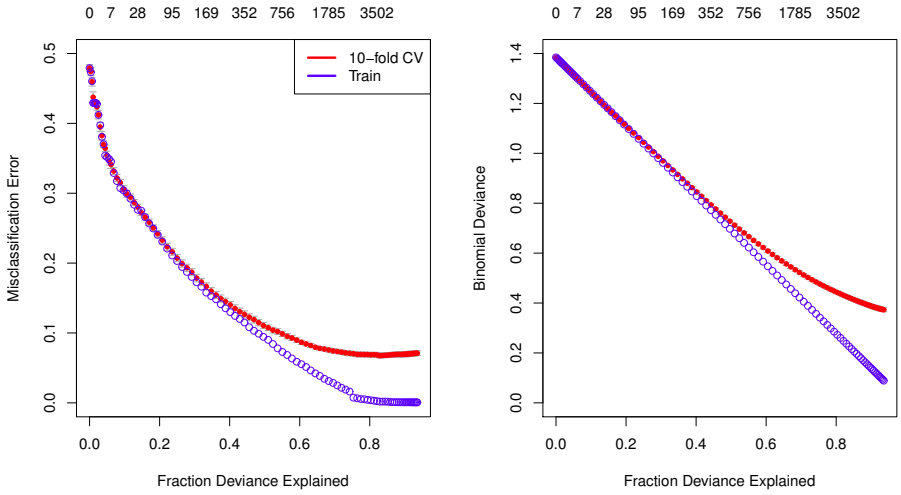


Figure 3.2 *Lasso (ℓ_1)-penalized logistic regression. Tenfold cross-validation curves for the Newsgroup data are shown in red, along with pointwise standard-error bands (not visible). The left plot shows misclassification error; the right plot shows deviance. Also shown in blue is the training error for each of these measures. The number of nonzero coefficients in each model is shown along the top of each plot.*

and gives no indication of the amount of overfitting, whereas for the latter measure, the graph would be dominated by the less interesting right-hand side, in which the coefficients and hence their norm explode.

Figure 3.2 shows the results of tenfold cross-validation for these data, as well as training error. These are also indexed by the fraction of deviance explained on the training data. Figure 3.3 shows the analogous results to those in Figure 3.2, for ridge regression. The cross-validated error rates are about the same as for the lasso. The number of nonzero coefficients in every model is $p = 777,811$ compared to a maximum of $5,277$ in Figure 3.2. However the rank of the ridge regression fitted values is actually $\min(N, p)$ which equals $11,314$ in this case, not much different from that of the lasso fit. Nonetheless, ridge regression might be more costly from a computational viewpoint. We produced the cross-validation results in Figure 3.3 using the `glmnet` package; for ridge the tenfold cross-validation took 8.3 minutes, while for lasso under one minute. A different approach would be to use the kernel trick (Hastie and Tibshirani 2004, for example), but this requires a singular value or similar decomposition of an $11,314 \times 11,314$ matrix.

For this example, using the package `glmnet`, we fit the regularization path in Figure 3.1 at 100 values of λ in 5 secs on a 2.6 GHz Macbook Pro. In examples like this with so many features, dramatic speedups can be achieved by screening the features. For example, the first feature to enter the regularization path achieves $\lambda_{\max} = \max_j |\langle x_j, \mathbf{y} - \bar{\mathbf{p}} \rangle|$, where $\mathbf{y}$ is the vector of binary

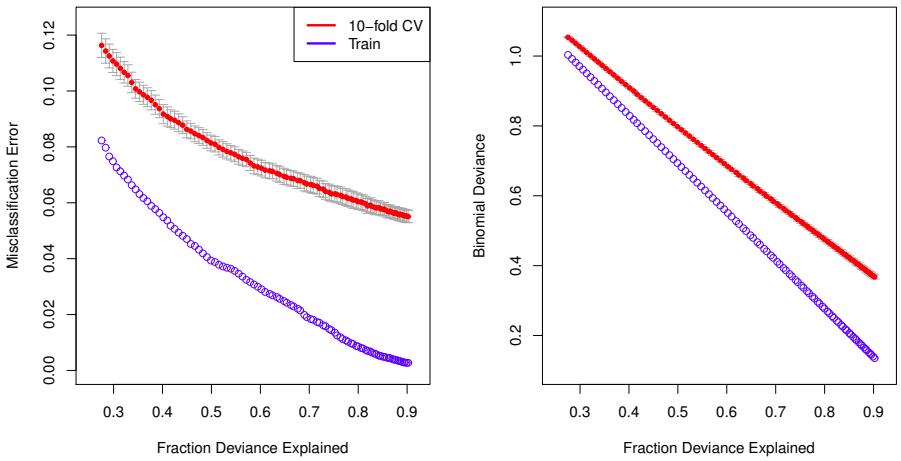


Figure 3.3 *Ridge (ℓ_2)-penalized logistic regression: tenfold cross validation curves for the Newsgroup data are shown in red, along with pointwise standard-error bands. The left plot shows misclassification error; the right plot shows deviance. Also shown in blue is the training error for each of these measures.*

outcomes, and $\bar{\mathbf{p}} = 0.52\,\mathbf{1}$ is a vector of the overall mean. This is the entry value for λ; that is the smallest value for which all coefficients are zero. When computing the solution path from $\lambda_{\max}$ down to a slightly lower value λ_1, we can screen out the vast majority of variables for which this inner-product is substantially lower than λ_1. Once we have computed the solution with the much smaller subset, we can check if any those screened were omitted in error. This can be repeated as we move down the path, using inner-products with the current residuals. This "strong-rule" screening is implemented in the `glmnet` package that we used for the computations in the above example. We discuss strong rules and other computational speedups in more detail in Section 5.10.

3.2.2 Algorithms

Two-class logistic regression is a popular generalization of linear regression, and as a consequence much effort has gone into fitting lasso-penalized logistic models. The objective (3.9) is convex and the likelihood part is differentiable, so in principle finding a solution is a standard task in convex optimization (Koh et al. 2007).

Coordinate descent is both attractive and efficient for this problem, and in the bibliographic notes we give a partial account of the large volume of research on this approach; see also Sections 2.4.2 and 5.4. The `glmnet` package uses a proximal-Newton iterative approach, which repeatedly approximates the negative log-likelihood by a quadratic function (Lee, Sun and Saunders 2014).

In detail, with the current estimate $(\tilde{\beta}_0, \tilde{\beta})$, we form the quadratic function

$$Q(\beta_0, \beta) = \frac{1}{2N} \sum_{i=1}^{N} w_i (z_i - \beta_0 - \beta^T x_i)^2 + C(\tilde{\beta}_0, \tilde{\beta}), \tag{3.12}$$

where C denotes a constant independent of (β_0, β), and

$$z_i = \tilde{\beta}_0 + \tilde{\beta}^T x_i + \frac{y_i - \tilde{p}(x_i)}{\tilde{p}(x_i)(1 - \tilde{p}(x_i))}, \quad \text{and} \quad w_i = \tilde{p}(x_i)(1 - \tilde{p}(x_i)), \tag{3.13}$$

with $\tilde{p}(x_i)$ being the current estimate for $\Pr(Y = 1 \mid X = x_i)$. Each outer loop then amounts to a weighted lasso regression. By using warm starts on a fine grid of values for λ, typically only a few outer-loop iterations are required, because locally the quadratic approximation is very good. We discuss some of the features of `glmnet` in Sections 3.7 and 5.4.2.

3.3 Multiclass Logistic Regression

Some classification and discrimination problems have $K > 2$ output classes. In machine learning a popular approach is to build all $\binom{K}{2}$ classifiers ("one versus one" or OvO), and then classify to the class that wins the most competitions. Another approach is "one versus all" (OvA) which treats all but one class as the negative examples. Both of these methods can be put on firm theoretical grounds, but also have limitations. OvO can be computationally wasteful, and OvA can suffer from certain masking effects (Hastie et al. 2009, Chapter 4). With multiclass logistic regression, a more natural approach is available. We use the multinomial likelihood and represent the probabilities using the log-linear representation

$$\Pr(Y = k \mid X = x) = \frac{e^{\beta_{0k} + \beta_k^T x}}{\sum_{\ell=1}^{K} e^{\beta_{0\ell} + \beta_\ell^T x}}. \tag{3.14}$$

This model is over specified, since we can add the linear term $\gamma_0 + \boldsymbol{\gamma}^T x$ to the linear model for each class, and the probabilities are unchanged. For this reason, it is customary to set one of the class models to zero—often the last class—leading to a model with $K - 1$ linear functions to estimate (each a contrast with the last class). The model fit by maximum-likelihood is invariant to the choice of this base class, and the parameter estimates are equivariant (the solution for one base can be obtained from the solution for another).

Here we prefer the redundant but symmetric approach (3.14), because

- we regularize the coefficients, and the regularized solutions are not equivariant under base changes, and
- the regularization automatically eliminates the redundancy (details below).

For observations $\{(x_i, y_i\}_{i=1}^N$, we can write the regularized form of the negative

log-likelihood as

$$-\frac{1}{N}\sum_{i=1}^{N}\log \Pr(Y=y_i \mid x_i;\{\beta_{0k},\beta_k\}_{k=1}^{K})+\lambda\sum_{k=1}^{K}\|\beta_k\|_1. \tag{3.15}$$

Denote by $\mathbf{R}$ the $N \times K$ *indicator response* matrix with elements $r_{ik} = \mathbb{I}(y_i = k)$. Then we can write the log-likelihood part of the objective (3.15) in the more explicit form

$$\frac{1}{N}\sum_{i=1}^{N} w_i\left[\sum_{k=1}^{K} r_{ik}(\beta_{0k}+\beta_k^T x_i)-\log\left\{\sum_{k=1}^{K} e^{\beta_{0k}+\beta_k^T x_i}\right\}\right]. \tag{3.16}$$

We have included a weight w_i per observation, where the setting $w_i = 1/N$ is the default. This form allows for *grouped* response data: at each value x_i we have a collection of n_i multicategory responses, with r_{ik} in category k. Alternatively, the rows of $\mathbf{R}$ can be a vector of class proportions, and we can provide $w_i = n_i$ as the observation weights.

As mentioned, the model probabilities and hence the log-likelihood are invariant under a constant shift in the K coefficients for each variable x_j—in other words $\{\beta_{kj} + c_j\}_{k=1}^{K}$ and $\{\beta_{kj}\}_{k=1}^{K}$ produce exactly the same probabilities. It is therefore up to the penalty in the criterion (3.15) to resolve the choice of c_j. Clearly, for any candidate set $\{\tilde{\beta}_{kj}\}_{k=1}^{K}$, the optimal c_j should satisfy

$$c_j = \arg\min_{c\in\mathbb{R}}\left\{\sum_{k=1}^{K}|\tilde{\beta}_{kj}-c|\right\}. \tag{3.17}$$

Consequently, as shown in Exercise 3.3, for each $j = 1,\ldots,p$, the maximizer of the objective (3.17) is given by the median of $\{\tilde{\beta}_{1j},\ldots,\tilde{\beta}_{Kj}\}$. Since the intercepts $\{\beta_{0k}\}_{k=1}^{K}$ are not penalized, we do need to resolve their indeterminacy; in the `glmnet` package, they are constrained to sum to zero.

3.3.1 Example: Handwritten Digits

As an illustration, we consider the US post-office handwritten digits data (Le Cun, Boser, Denker, Henderson, Howard, Hubbard and Jackel 1990). There are $N = 7291$ training images of the digits $\{0, 1, \ldots, 9\}$, each digitized to a 16×16 gray-scale image. Using the $p = 256$ pixels as features, we fit a 10-class lasso multinomial model. Figure 3.4 shows the training and test misclassification error as a function of the sequence of λ values used. In Figure 3.5 we display the coefficients as images (on average about 25% are nonzero). Some of these can be identified as appropriate contrast functionals for highlighting each digit.

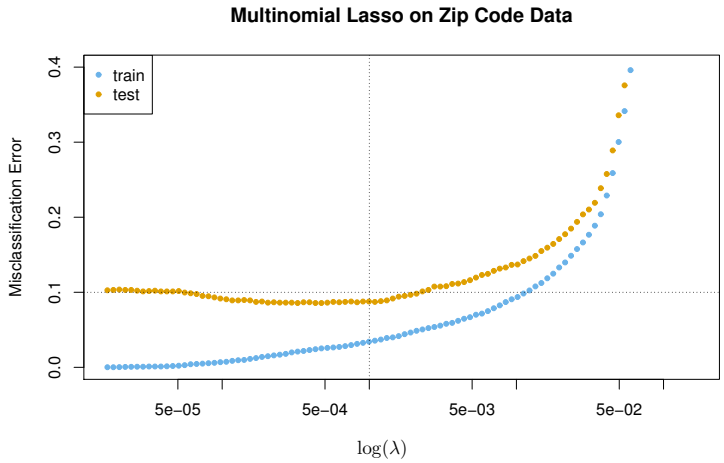


Figure 3.4 *Training and test misclassification errors of a multinomial lasso model fit to the zip code data, plotted as a function of* $\log(\lambda)$*. The minimum test error here is around* 0.086*, while the minimum training error is* 0*. We highlight the value* $\lambda = 0.001$*, where we examine the individual class coefficients in Figure 3.5.*

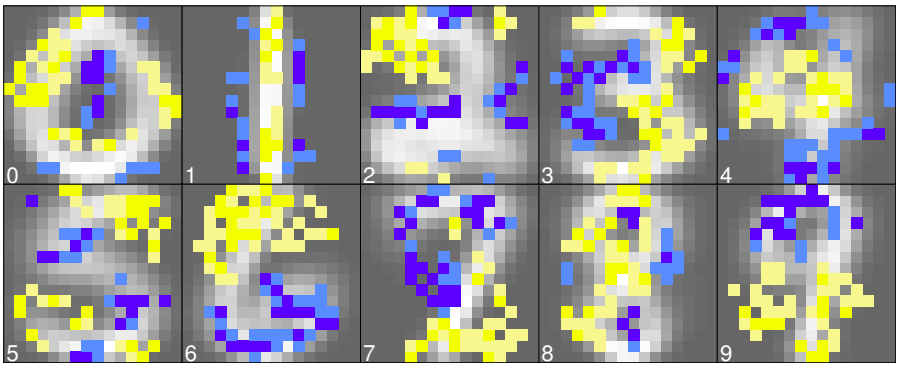


Figure 3.5 *Coefficients of the multinomial lasso, displayed as images for each digit class. The gray background image is the average training example for that class. Superimposed in two colors (yellow for positive, blue for negative) are the nonzero coefficients for each class. We notice that they are nonzero in different places, and create discriminant scores for each class. Not all of these are interpretable.*

3.3.2 Algorithms

Although one could tackle this problem with standard convex-optimization software, we have found coordinate-descent to be particularly effective (Friedman, Hastie, Simon and Tibshirani 2015). In the two-class case, there is an outer Newton loop and an inner weighted least-squares step. The outer loop can be seen as making a quadratic approximation to the log-likelihood, centered at the current estimates $(\tilde{\beta}_{0k}, \tilde{\beta}_k\}_{k=1}^K$. Here we do the same, except we hold all but one class's parameters fixed when making this approximation. In detail, when updating the parameters $(\beta_{0\ell}, \beta_\ell)$, we form the quadratic function

$$Q_\ell(\beta_{0\ell}, \beta_\ell) = -\frac{1}{2N}\sum_{i=1}^N w_{i\ell}\,(z_{i\ell} - \beta_{0\ell} - \beta_\ell^T x_i)^2 + C(\{\tilde{\beta}_{0k}, \tilde{\beta}_k\}_{k=1}^K), \quad (3.18)$$

where C denotes a constant independent of $(\beta_{0\ell}, \beta_\ell)$, and

$$z_{i\ell} = \tilde{\beta}_{0\ell} + \tilde{\beta}_\ell^T x_i + \frac{r_{i\ell} - \tilde{p}_\ell(x_i)}{\tilde{p}_\ell(x_i)(1-\tilde{p}_\ell(x_i))}, \quad \text{and} \quad w_{i\ell} = \tilde{p}_\ell(x_i)(1-\tilde{p}_\ell(x_i))$$

where $\tilde{p}_\ell(x_i)$ is the current estimate for the conditional probability $\Pr(Y = \ell \mid x_i)$. Our approach is similar to the two-class case, except now we have to cycle over the classes as well in the outer loop. For each value of λ, we create an outer loop which cycles over $\ell \in \{1, \ldots, K\}$ and computes the partial quadratic approximation Q_ℓ about the current parameters $(\tilde{\beta}_0, \tilde{\beta})$. Then we use coordinate descent to solve the weighted lasso problem problem

$$\underset{(\beta_{0\ell}, \beta_\ell) \in \mathbb{R}^{p+1}}{\text{minimize}} \left\{ Q(\beta_{0\ell}, \beta_\ell) + \lambda \|\beta_\ell\|_1 \right\}. \quad (3.19)$$

3.3.3 Grouped-Lasso Multinomial

As can be seen in Figure 3.5, the lasso penalty will select different variables for different classes. This can mean that although individual coefficient vectors are sparse, the overall model may not be. In this example, on average there are 25% of the coefficients nonzero per class, while overall 81% of the variables are used.

An alternative approach is to use a grouped-lasso penalty (see Section 4.3) for the set of coefficients $\beta_j = (\beta_{1j}, \beta_{2j}, \ldots, \beta_{Kj})$, and hence replace the criterion (3.15) with the regularized objective

$$-\frac{1}{N}\sum_{i=1}^N \log \Pr(Y = y_i \mid X = x_i; \{\beta_j\}_{j=1}^p) + \lambda \sum_{j=1}^p \|\beta_j\|_2. \quad (3.20)$$

It is important that this criterion involves the sum of the ordinary ℓ_2-norms $\|\cdot\|_2$, as opposed to the squared ℓ_2-norms. In this way, it amounts to imposing a block ℓ_1/ℓ_2 constraint on the overall collection of coefficients: the

sum of the ℓ_2-norms over the groups. The effect of this group penalty is to select all the coefficients for a particular variable to be in or out of the model. When included, they are all nonzero in general, and as shown in Exercise 3.6, they will automatically satisfy the constraint $\sum_{k=1}^{K} \beta_{kj} = 0$. Criterion (3.20) is convex, so standard methods can be used to find the optimum. As before, coordinate descent techniques are one reasonable choice, in this case block coordinate descent on each vector β_j, holding all the others fixed; see Exercise 3.7 for the details. The group lasso and variants are discussed in more detail in Chapter 4.3.

3.4 Log-Linear Models and the Poisson GLM

When the response variable Y is nonnegative and represents a count, its mean will be positive and the Poisson likelihood is often used for inference. In this case we typically use the log-linear model (3.3) to enforce the positivity. We assume that for each $X = x$, the response Y follows a Poisson distribution with mean μ satisfying

$$\log \mu(x) = \beta_0 + \beta^T x. \tag{3.21}$$

The ℓ_1-penalized negative log-likelihood is given by

$$-\frac{1}{N}\sum_{i=1}^{N}\left\{y_i(\beta_0 + \beta^T x_i) - e^{\beta_0+\beta^T x_i}\right\} + \lambda\|\beta\|_1. \tag{3.22}$$

As with other GLMs, we can fit this model by iteratively reweighted least squares, which amounts to fitting a weighted lasso regression at each outer iteration. Typically, we do not penalize the intercept β_0. It is easy to see that this enforces the constraint that the average fitted value is equal to the mean response—namely, that $\frac{1}{N}\sum_{i=1}^{N}\hat{\mu}_i = \bar{y}$, where $\hat{\mu}_i := e^{\hat{\eta}(x_i)} = e^{\widehat{\beta}_0+\widehat{\beta}^T x_i}$.

Poisson models are often used to model rates, such as death rates. If the length T_i of the observation window is different for each observation, then the mean count is $\mathbb{E}(y_i \mid X_i = x_i) = T_i\mu(x_i)$ where $\mu(x_i)$ is the rate per unit time interval. In this case, our model takes the form

$$\log(\mathbb{E}(Y \mid X = x, T)) = \log(T) + \beta_0 + \beta^T x. \tag{3.23}$$

The terms $\log(T_i)$ for each observation require no fitting, and are called an *offset*. Offsets play a role in the following example as well.

3.4.1 Example: Distribution Smoothing

The Poisson model is a useful tool for estimating distributions. The following example was brought to our attention by Yoram Singer (Singer and Dubiner 2011). Suppose that we have a sample of N counts $\{y_k\}_{k=1}^{N}$ from an

N-cell multinomial distribution, and let $r_k = y_k / \sum_{\ell=1}^N y_\ell$ be the corresponding vector of proportions. For example, in large-scale web applications, these counts might represent the number of people in each county in the USA that visited a particular website during a given week. This vector could be sparse, depending on the specifics, so there is a desire to regularize toward a broader, more stable distribution $\mathbf{u} = \{u_k\}_{k=1}^N$ (for example, the same demographic, except measured over a year). Singer and Dubiner (2011) posed the following problem

$$\underset{\mathbf{q}\in\mathbb{R}^N,\ q_k\geq 0}{\text{minimize}} \sum_{k=1}^N q_k \log\left(\frac{q_k}{u_k}\right) \quad \text{such that } \|\mathbf{q}-\boldsymbol{r}\|_\infty \leq \delta \text{ and } \textstyle\sum_{k=1}^N q_k = 1. \tag{3.24}$$

In words, we find the distribution, within a δ tolerance in the ℓ_∞-norm from the observed distribution, that is as close as possible to the nominal distribution $\mathbf{u}$ in terms of Kullback–Leibler (KL) divergence. It can be shown (see

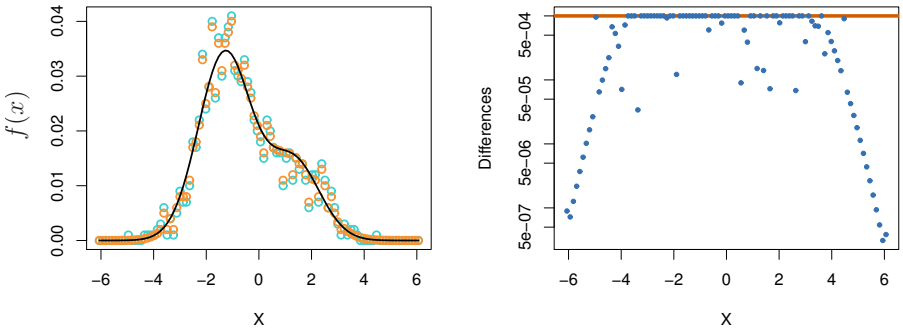


Figure 3.6 *Estimating distributions via the Poisson. In the left panel, the solid black curve is the parent distribution* $\mathbf{u}$*, here represented as a discretization of a one-dimensional distribution* $f(x)$ *into 100 cells. The blue points represent the observed distribution, and the orange points represent the distribution recovered by the model. While the observed distribution may have many zero counts, the modeled distribution has the same support as* $\mathbf{u}$*. The right plot shows the* $N = 100$ *differences* $|\hat{q}_k - r_k|$*, which are constrained to be less than* $\delta = 0.001$*, which is the horizontal orange line.*

Exercise 3.4) that the Lagrange-dual to the optimization problem (3.24) has the form

$$\underset{\beta_0,\boldsymbol{\alpha}}{\text{maximize}} \left\{ \sum_{k=1}^N \left[r_k \log q_k(\beta_0,\alpha_k) - q_k(\beta_0,\alpha_k) \right] - \delta\|\boldsymbol{\alpha}\|_1 \right\}, \tag{3.25}$$

where $q_k(\beta_0,\alpha_k) := u_k e^{\beta_0+\alpha_k}$. This is equivalent to fitting a Poisson GLM with offset $\log(u_k)$, individual parameter α_k per observation, and the extremely sparse design matrix $\mathbf{X} = \mathbf{I}_{N\times N}$. Consequently, it can be fit very efficiently using sparse-matrix methods (see Section 3.7 below). Figure 3.6

shows a simulation example, where the distribution u_k is a discretized continuous distribution (mixture of Gaussians). There are $N = 100$ cells, and a total of $\sum_{k=1}^{N} y_k = 1000$ observations distributed to these cells. As discussed above, the presence of the unpenalized β_0 ensures that $\sum_{k=1}^{N} \hat{q}_k = \sum_{k=1}^{N} r_k = 1$ (see also Exercise 3.5). Although we only show one solution in Figure 3.6, the path gives solutions $\hat{q}_k(\delta)$ that vary smoothly between the background distribution u_k and the observed distribution r_k.

3.5 Cox Proportional Hazards Models

In medical studies, the outcome of interest after treatment is often time to death or time to recurrence of the disease. Patients are followed after their treatment, and some drop out because they move away, or perhaps die from an independent cause. Such outcomes are called *right censored*. Denoting by T the underlying survival time, for each patient we observe the quantity $Y = \min(T, C)$ where C is a *censoring time*. Interest tends to focus on the survivor function $S(t) := \Pr(T > t)$, the probability of surviving beyond a certain time t.

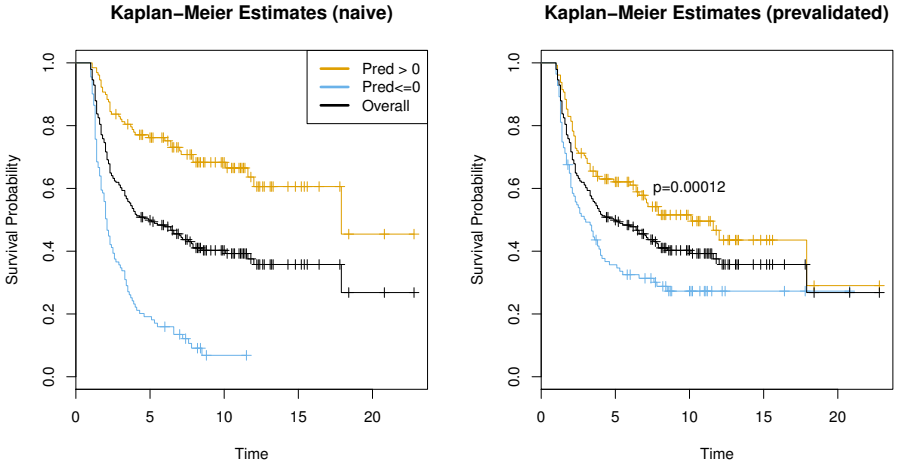


Figure 3.7 *The black curves are the Kaplan–Meier estimates of $S(t)$ for the Lymphoma data. In the left plot, we segment the data based on the predictions from the Cox proportional hazards lasso model, selected by cross-validation. Although the tuning parameter is chosen by cross-validation, the predictions are based on the full training set, and are overly optimistic. The right panel uses* prevalidation *to build a prediction on the entire dataset, with this training-set bias removed. Although the separation is not as strong, it is still significant. The spikes indicate censoring times. The p-value in the right panel comes from the log-rank test.*

The black curves in Figure 3.7 show estimates of $S(t)$ for a population of $N = 240$ Lymphoma patients (Alizadeh et al. 2000). Each of the spikes in the

plot indicates a censoring point, meaning a time at which a patient was lost for follow-ups. Although survival curves are useful summaries of such data, when incorporating covariates it is more common to model the *hazard function*, a monotone transformation of S. More specifically, the hazard at time t is given by

$$h(t) = \lim_{\delta \to 0} \frac{\Pr(Y \in (t, t+\delta) \mid Y \geq t)}{\delta} = \frac{f(t)}{S(t)}, \tag{3.26}$$

and corresponds to the instantaneous probability of death at time t, given survival up till t.

We now discuss Cox's proportional hazards model that was used to produce the blue and orange survival curves in Figure 3.7. The proportional hazards model (CPH) is based on the hazard function

$$h(t; x) = h_0(t) e^{\beta^T x}, \tag{3.27}$$

where $h_0(t)$ is a baseline hazard (the hazard for an individual with $x = 0$).

We have data of the form (x_i, y_i, δ_i), where δ_i is binary-valued indicator of whether y_i is a death time or censoring time. For the lymphoma data, there are $p = 7399$ variables, each a measure of gene expression. Of the $N = 240$ samples, a total of 102 samples are right censored. Here we fit an ℓ_1-penalized CPH by solving

$$\underset{\beta}{\text{minimize}} \left\{ - \sum_{\{i \,|\, \delta_i = 1\}} \log \left[\frac{e^{\beta^T x_i}}{\sum_{j \in R_i} e^{\beta^T x_j}} \right] + \lambda \|\beta\|_1 \right\}, \tag{3.28}$$

where for each $i = 1, \ldots, N$, R_i is the *risk set* of individuals who are alive and in the study at time y_i. The first term is the log of the *partial likelihood*, corresponding to the conditional probability in the risk set of the observed death. Note that the baseline hazard does not play a role, an attractive feature of this approach. Here we have assumed that there are no ties, that is, the survival times are all unique. Modification of the partial likelihood is needed in the event of ties.

Figure 3.8 shows the coefficients obtained in fitting the model (3.28) to the Lymphoma data. Since $p \gg N$, the model would "saturate" as $\lambda \downarrow 0$, meaning that some parameters would diverge to $\pm\infty$, and the log partial likelihood would approach 0. We see evidence of this undesirable behavior as λ gets small.

The computations for the Cox model are similar to those for the multinomial model but slightly more complex. Simon, Friedman, Hastie and Tibshirani (2011) give details for an algorithm based on coordinate-descent.

3.5.1 Cross-Validation

All the models in this chapter require a choice of λ, and we typically use $K-$fold cross-validation where K equal to 5 or 10, as in Figure 3.2. For the

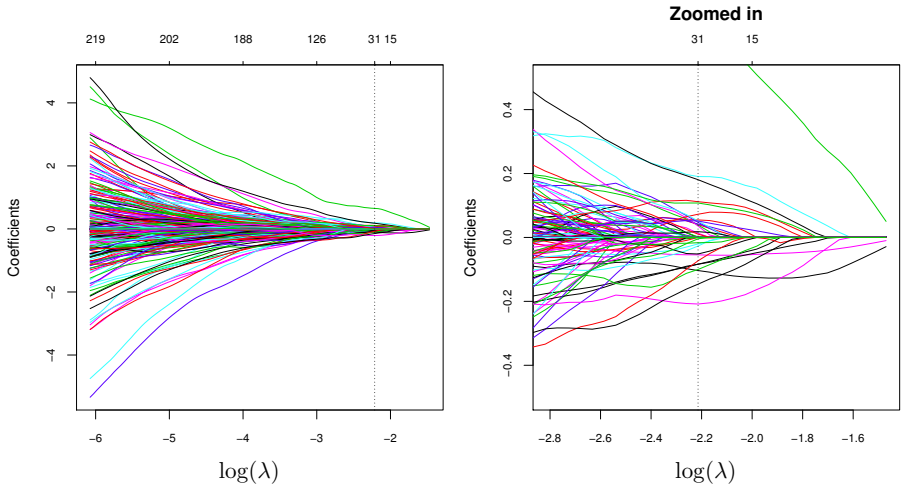


Figure 3.8 *The ℓ_1-regularized coefficient path for the Cox model fit to the Lymphoma data. Since $p \gg N$, the plot has a trumpet shape near the end, corresponding to a saturated model with partial likelihood equal to one. The right-hand plot zooms in on the area of interest, a fairly heavily regularized solution with* 31 *nonzero coefficients.*

Cox model, we compute the cross-validated deviance, which is minus twice the log partial likelihood. An issue arises in computing the deviance, since if N/K is small, there will not be sufficient observations to compute the risk sets. Here we use a trick due to van Houwelingen et al. (2006). When fold k is left out, we compute the coefficients $\widehat{\beta}^{-k}(\lambda)$, and then compute

$$\widehat{\text{Dev}}_\lambda^k := \text{Dev}[\widehat{\beta}^{-k}(\lambda)] - \text{Dev}^{-k}[\widehat{\beta}^{-k}(\lambda)]. \tag{3.29}$$

The first term on the right uses all N samples in computing the deviance, while the second term omits the fold-k samples. Finally $\text{Dev}_\lambda^{CV} = \sum_{k=1}^K \widehat{\text{Dev}}_\lambda^k$ is obtained by subtraction. The point is that each of these terms has sufficient data to compute the deviance, and in the standard cases (that is, any of the other generalized linear models), the estimate would be precisely the deviance on the left-out set.

The deviance in Figure 3.9 was computed in this fashion; we zoom in on the right-hand section. We see that the minimum is achieved at 31 nonzero coefficients. Figure 3.7 shows the effect of the chosen model. We compute $\hat{\eta}(x_i) = x_i^T \hat{\beta}(\lambda_{\min})$ for each observation, and then create two groups by thresholding these scores at zero. The two colored survival curves in the left-hand plot show the difference in survival for the two groups thus formed. They are well separated, which suggests we have derived a powerful signature. However, these scores are biased: we are evaluating their performance on the same data for which they were computed.

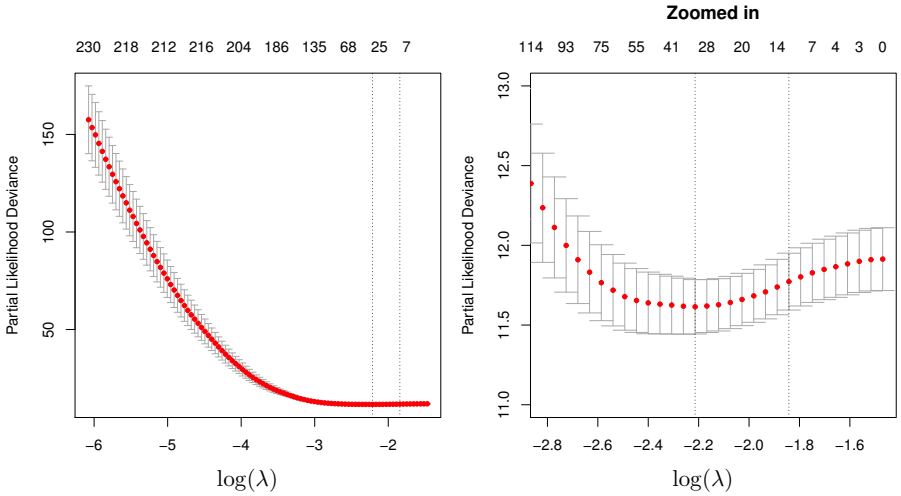


Figure 3.9 *Cross-validated deviance for the lymphoma data, computed by subtractions, as described in the text. The right-hand plot zooms in on the area of interest. The dotted vertical line on the left corresponds to the minimum, and the model we chose in this case; the one on the right corresponds to the rightmost point on the curve (simplest model) within one standard error of the minimum. This is a basis for a more conservative approach to selection. The number of nonzero coefficients is shown along the top of each plot.*

3.5.2 Pre-Validation

In Figure 3.7, we used a variant of cross-validation, known as *pre-validation* (Tibshirani and Efron 2002), in order to obtain a fair evaluation of the model. Cross-validation leaves out data in order to obtain a reasonably unbiased estimate of the error rate of a model. But the error rate is not a very interpretable measure in some settings such as survival modelling. The method of *pre-validation* is similar to cross-validation, but instead produces a new set of "unbiased data" that mimics the performance of the model applied to independent data. The pre-validated data can then be analyzed and displayed. In computing the score $\hat{\eta}(x_i)^{(k)}$ for the observations in fold k, we use the coefficient vector $\widehat{\beta}^{(-k)}$ computed with those observations omitted.[5] Doing this for all K folds, we obtain the "pre-validated" dataset $\{(\hat{\eta}(x_i)^{(k)}, y_i, \delta_i)\}_{i=1}^N$. The key aspect of this pre-validated data is that each score $\hat{\eta}(x_i)^{(k)}$ is derived independently of its response value (y_i, δ_i). Hence we can essentially treat these scores as if they were derived from a dataset completely separate from the "test data" $\{(x_i, y_i, \delta_i)\}_{i=1}^N$. In the right-hand panel of Figure 3.7, we have split the pre-validated scores into two groups and plotted the corresponding

[5]Strictly speaking λ should be chosen each time as well, but we did not do that here.

survival curves. Although the curves are not as spread out as in the left-hand plot, they are still significantly different.

3.6 Support Vector Machines

We now turn to a method for binary classification known as the support vector machine (SVM). The idea is shown in Figure 3.10. The decision boundary is the solid line in the middle of the yellow slab. The *margin* is the half-width of the yellow slab. Ideally, all of the blue data points should lie above the slab on the right, and the red points should lie below it on the left. However in the picture, three red points and two blue points lie on the wrong side of their margin. These correspond to the "errors" ξ_i. The SVM decision boundary is chosen to maximize the margin, subject to a fixed budget on the total error $\sum_{i=1}^{N} \xi_i$. The idea is that a decision boundary achieving the largest margin has more space between the classes and will generalize better to test data. This leads to the optimization problem

$$\underset{\beta_0,\, \beta,\, \{\xi_i\}_1^N}{\text{maximize}} \; M \quad \text{subject to } y_i \underbrace{(\beta_0 + \beta^T x_i)}_{f(x_i;\, \beta_0,\, \beta)} \geq M(1 - \xi_i) \; \forall i, \tag{3.30}$$

$$\text{and } \xi_i \geq 0 \; \forall i, \; \sum_{i=1}^{N} \xi_i \leq C, \text{ and } \|\beta\|_2 = 1. \tag{3.31}$$

(See Section 3.6.1 for an explanation of this particular form.)

This problem involves a linear cost function subject to convex constraints, and many efficient algorithms have been designed for its solution. It can be shown to be equivalent to the penalized form (3.8) previously specified on page 31, which we restate here:

$$\underset{\beta_0,\beta}{\text{minimize}} \left\{ \frac{1}{N} \sum_{i=1}^{N} [1 - y_i f(x\,; \beta_0, \beta)]_+ + \lambda \|\beta\|_2^2 \right\}. \tag{3.32}$$

Decreasing λ has a similar effect to decreasing C.[6] The linear SVM can be generalized using a *kernel* to create nonlinear boundaries; it involves replacing the squared ℓ_2-norm in the objective (3.32) by the squared Hilbert norm defined by a symmetric bivariate kernel. Details on this extension can be found elsewhere—for instance, see Hastie et al. (2009), Section 5.8.

Since the criterion (3.32) involves a quadratic penalty, the estimated coefficient vector will not be sparse. However, because the hinge loss function is piecewise linear, it introduces a different kind of sparsity. It can be shown via the dual formulation of the SVM that the solution $\hat{\beta}$ has the form

$$\hat{\beta} = \sum_{i=1}^{N} \hat{\alpha}_i y_i x_i; \tag{3.33}$$

[6] Solutions to (3.32) do not have $\|\hat{\beta}\|_2 = 1$, but since a linear classifier is scale invariant, the solution coefficients can be rescaled.

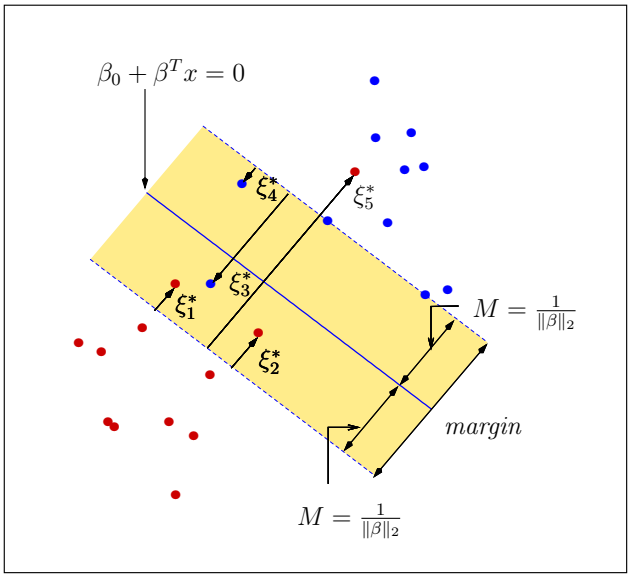


Figure 3.10 *Support vector classifier: The decision boundary is the solid line, while broken lines bound the shaded maximal margin of width* $2M = 2/\|\beta\|_2$. *The points labelled* ξ_j^* *are on the wrong side of their margin by an amount* $\xi_j^* = M\xi_j$; *points on the correct side have* $\xi_j^* = 0$. *The margin is maximized subject to a total budget* $\sum_{i=1}^N \xi_i \leq C$. *Hence* $\sum_{i=1}^N \xi_j^*$ *is the total distance of points on the wrong side of their margin.*

each observation $i \in \{1, \ldots, N\}$ is associated with a nonnegative weight $\hat{\alpha}_i$, and only a subset $\mathcal{V}_\lambda$, referred to as the *support set*, will be associated with nonzero weights.

SVMs are popular in high-dimensional classification problems with $p \gg N$, since the computations are $\mathcal{O}(pN^2)$ for both linear and nonlinear kernels. Additional efficiencies can be realized for linear SVMs, using stochastic subgradient methods (Shalev-Shwartz, Singer and Srebro 2007). They are not, however, sparse in the features. Replacing the ℓ_2 penalty in the objective (3.32) with an ℓ_1 penalty promotes such sparsity, and yields the *ℓ_1-regularized linear SVM*:

$$\underset{\beta_0,\beta}{\text{minimize}} \left\{ \frac{1}{N} \sum_{i=1}^N [1 - y_i f(x_i; \beta_0, \beta)]_+ + \lambda \|\beta\|_1 \right\}. \tag{3.34}$$

The optimization problem (3.34) is a linear program with many constraints (Zhu, Rosset, Hastie and Tibshirani 2004, Wang, Zhu and Zou 2006), and efficient algorithms can be complex (Exercise 3.9). The solution paths (in fine detail) can have many jumps, and show many discontinuities. For this reason, some authors prefer to replace the usual hinge loss $\phi_{\text{hin}} = (1 - t)_+$

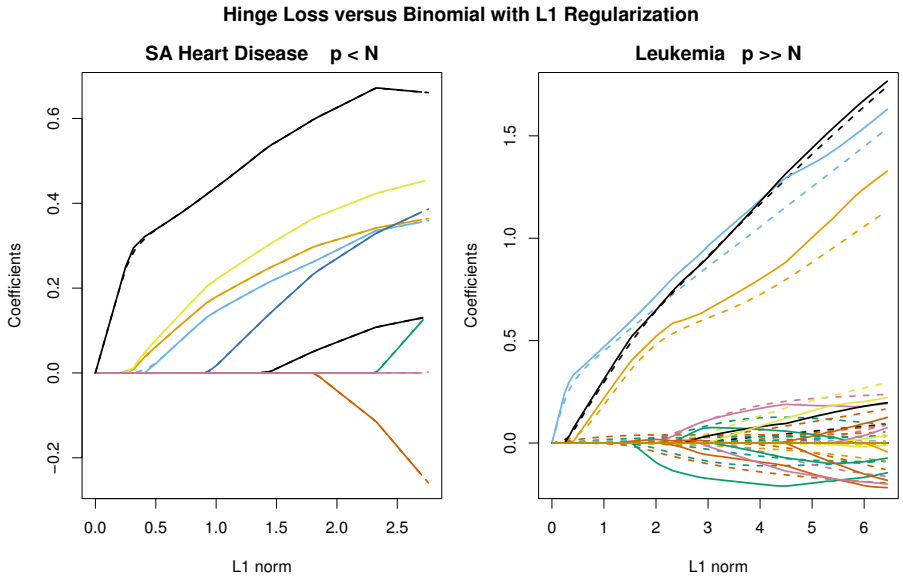


Figure 3.11 *A comparison of the coefficient paths for the ℓ_1-regularized SVM versus logistic regression on two examples. In the left we have the South African heart disease data ($N = 462$ and $p = 9$), and on the right the Leukemia data ($N = 38$ and $p = 6087$). The dashed lines are the SVM coefficients, the solid lines logistic regression. The similarity is striking in the left example, and strong in the right.*

with *squared hinge loss* $\phi_{\text{sqh}}(t) = (1-t)_+^2$, which is differentiable everywhere (see Exercise 3.8).

The SVM loss function shares many similarities with the binomial loss (Hastie et al. 2009, Section 12.3), and their solutions are not too different. Figure 3.11 compares their ℓ_1 regularization paths on two examples, and supports this claim. In the left-hand plot, they are virtually identical. In the right-hand plot, for more than half of the path, the training data are separated by the solution. As we proceed to the end of the path, the logistic coefficients become less stable than those of the SVM, and can account for the bigger discrepancies.

The support vector machine, on the other hand, is designed for finding maximal-margin solutions for separable data, and its coefficients do not blow up at the least-regularized end of the path. However, in terms of the ℓ_1 penalty, this is at the nonsparse end of the path. In light of this, we do not recommend the ℓ_1 regularized linear SVM as a variable selector, because the corresponding logistic regression problem (3.6) gives very similar solutions when the penalty is active, and the algorithms are more stable.

3.6.1 Logistic Regression with Separable Data

It is a well-known fact that without a penalty on the coefficients, the linear logistic regression model fails when the two classes are linearly separable (Exercise 3.1); the maximum-likelihood estimates for the coefficients are infinite. The problem in this case is that the likelihood is trying to make the probabilities all 1s and 0s, and inspection of (3.2) shows that this cannot be achieved with finite parameters. Once we penalize the criterion as in (3.6) the problem goes away, for as long as $\lambda > 0$, very large coefficients will not be tolerated.

With wide data ($N \ll p$), the classes are almost always separable, unless there are exact ties in covariate space for the two classes. Figure 3.1 shows the logistic-regression coefficient path for a wide-data situation; notice how the coefficients start to fan out near the end of the path. One has to take care at this end of the path, and not allow λ to get too small. In many situations, this end represents the overfit situation, which is not of primary interest. It appears not to be the case in this example, as can be seen in the cross-validation plots in Figure 3.2.

The ends of the path have special meaning in the machine-learning community, since we will see they amount to maximal-margin classifiers. Before giving the details, we review some geometry associated with linear classification. Consider the boundary $\mathcal{B} := \{x \in \mathbb{R}^p \mid f(x) = 0\}$ associated with a linear classifier $f(x) \equiv f(x\,;\beta_0,\beta) = \beta_0 + \beta^T x$. The Euclidean distance from a point x_0 to the boundary is given by

$$\text{dist}_2(x_0,\mathcal{B}) := \inf_{z\in\mathcal{B}} \|z - x_0\|_2 \;=\; \frac{|f(x_0)|}{\|\beta\|_2} \tag{3.35}$$

(Exercise 3.2). Consequently, for a given predictor-response pair (x, y), the quantity $\frac{y\,f(x)}{\|\beta\|_2}$ is the signed Euclidean distance to the boundary: it will be negative if the sign of y disagrees with that of $f(x)$. For separable data, the optimal separating hyperplane $f^*(x) = 0$ solves the optimization problem

$$M_2^* = \max_{\beta_0,\beta} \left\{ \min_{i\in\{1,\ldots,N\}} \frac{y_i f(x_i\,;\beta_0,\beta)}{\|\beta\|_2} \right\}. \tag{3.36}$$

In words, it maximizes the Euclidean distance of the closest sample to the boundary.

Rosset et al. (2004) establish an interesting connection between this optimal separating hyperplane and a certain limiting case of ridge-regularized logistic regression. In particular, suppose that we replace the ℓ_1-penalty in the objective (3.10) with a squared ℓ_2-penalty, and solve the problem

$$\underset{\beta_0,\beta}{\text{minimize}} \left\{ \frac{1}{N} \sum_{i=1}^{N} \log(1 + e^{-y_i f(x_i\,;\beta_0,\beta)}) + \lambda\|\beta\|_2^2 \right\}; \tag{3.37}$$

let $(\tilde\beta_0(\lambda),\, \tilde\beta(\lambda))$ be the optimal solution, specifying a particular linear classifier. We then consider the behavior of this linear classifier as the regularization

weight λ vanishes: in particular, it can be shown (Rosset et al. 2004) that

$$\lim_{\lambda \to 0} \left\{ \min_{i \in \{1,\ldots,N\}} \frac{y_i f(x_i; \tilde{\beta}_0(\lambda), \tilde{\beta}(\lambda))}{\|\tilde{\beta}(\lambda)\|_2} \right\} = M_2^*. \tag{3.38}$$

Thus, the end of the ℓ_2-regularized logistic regression path corresponds to the SVM solution. In particular, if $(\breve{\beta}_0, \breve{\beta})$ solves the SVM objective (3.30) with $C = 0$, then

$$\lim_{\lambda \to 0} \frac{\tilde{\beta}(\lambda)}{\|\tilde{\beta}(\lambda)\|_2} = \breve{\beta}. \tag{3.39}$$

How does this translate to the setting of ℓ_1-regularized models? Matters get a little more complicated, since we move into the territory of general projections and dual norms (Mangasarian 1999). The analog of the ℓ_2-distance (3.35) is the quantity

$$\text{dist}_\infty(x_0, \mathcal{B}) := \inf_{z \in \mathcal{B}} \|z - x_0\|_\infty = \frac{|f(x_0)|}{\|\beta\|_1}, \tag{3.40}$$

For a given $\lambda \geq 0$, let $(\widehat{\beta}_0(\lambda), \widehat{\beta}(\lambda))$ denote an optimal solution to the ℓ_1-regularized logistic regression objective (3.10). Then as λ decreases toward zero, we have

$$\lim_{\lambda \to 0} \left[\min_{i \in \{1,\ldots,N\}} \frac{y_i f(x_i; \widehat{\beta}_0(\lambda), \widehat{\beta}(\lambda))}{\|\widehat{\beta}(\lambda)\|_1} \right] = M_\infty^*, \tag{3.41}$$

so that the worst-case margin of the ℓ_1-regularized logistic regression converges to the ℓ_1-regularized version of the support vector machine, which maximizes the ℓ_∞ margin (3.40).

In summary, then, we can make the following observations:

- At the end of the path, where the solution is most dense, the logistic regression solution coincides with the SVM solution.
- The SVM approach leads to a more stable numerical method for computing the solution in this region.
- In contrast, logistic regression is most useful in the sparser part of the solution path.

3.7 Computational Details and `glmnet`

Most of the examples in this chapter were fit using the R package `glmnet` (Friedman et al. 2015). Here we detail some of the options and features in `glmnet`. Although these are specific to this package, they also would be natural requirements in any other similar software.

Family: The family option allows one to pick the loss-function and the associated model. As of version 1.7, these are `gaussian`, `binomial`, `multinomial` (grouped or not), `poisson`, and `cox`. The `gaussian` family allows for multiple responses (multitask learning), in which case a group lasso is used to select coefficients for each variable, as in the grouped multinomial. Associated with each family is a *deviance* measure, the analog of the residual sum-of-squares for Gaussian errors. Denote by $\hat{\boldsymbol{\mu}}_\lambda$ the N-vector of fitted mean values when the parameter is λ, and $\tilde{\boldsymbol{\mu}}$ the unrestricted or *saturated* fit. Then

$$\text{Dev}_\lambda \doteq 2[\ell(\mathbf{y}, \tilde{\boldsymbol{\mu}}) - \ell(\mathbf{y}, \hat{\boldsymbol{\mu}}_\lambda)]. \tag{3.42}$$

Here $\ell(\mathbf{y}, \boldsymbol{\mu})$ is the log-likelihood of the model $\boldsymbol{\mu}$, a sum of N terms. The *null deviance* is $\text{Dev}_{\text{null}} = \text{Dev}_\infty$; typically this means $\hat{\boldsymbol{\mu}}_\infty = \bar{y}\mathbf{1}$, or in the case of the `cox` family $\hat{\boldsymbol{\mu}}_\infty = \mathbf{0}$. `Glmnet` reports D^2, the fraction of deviance explained, as defined in (3.11) on page 33.

Penalties: For all models, the `glmnet` algorithm admits a range of elastic-net penalties ranging from ℓ_2 to ℓ_1. The general form of the penalized optimization problem is

$$\underset{\beta_0,\beta}{\text{minimize}} \left\{ -\frac{1}{N}\ell(\mathbf{y}; \beta_0, \beta) + \lambda \sum_{j=1}^{p} \gamma_j \left\{ (1-\alpha)\beta_j^2 + \alpha|\beta_j| \right\} \right\}. \tag{3.43}$$

This family of penalties is specified by three sets of real-valued parameters:

- The parameter λ determines the overall complexity of the model. By default, the `glmnet` algorithm generates a sequence of 100 values for λ that cover the whole path (on the log scale), with care taken at the lower end for saturated fits.
- The elastic-net parameter $\alpha \in [0, 1]$ provides a mix between ridge regression and the lasso. Although one can select α via cross-validation, we typically try a course grid of around three to five values of α.
- For each $j = 1, 2, \ldots, p$, the quantity $\gamma_j \geq 0$ is a penalty modifier. When $\gamma_j = 0$, the j^{th} variable is always included; when $\gamma_j =$ `inf` it is always excluded. Typically $\gamma_j = 1$ (the default), and all variables are treated as equals.

Coefficient bounds: With coordinate descent, it is very easy to allow for upper and lower bounds on each coefficient in the model. For example, we might ask for a nonnegative lasso. In this case, if a coefficient exceeds an upper or lower bound during the coordinate-descent cycle, it is simply set to the bound.

Offset: All the models allow for an *offset* term. This is a real valued number o_i for each observation, that gets added to the linear predictor, and is not associated with any parameter:

$$\eta(x_i) = o_i + \beta_0 + \beta^T x_i. \tag{3.44}$$

The offset has many uses. Sometimes we have a previously-fit model $h(z)$ (where z might include or coincide with x), and we wish to see if augmenting it with a linear model offers improvement. We would supply $o_i = h(z_i)$ for each observation.

For Poisson models the offset allows us to model rates rather than mean counts, if the observation period differs for each observation. Suppose we observe a count Y over period t, then $\mathbb{E}[Y \mid T = t, X = x] = t\mu(x)$, where $\mu(x)$ is the rate per unit time. Using the log link, we would supply $o_i = \log(t_i)$ for each observation. See Section 3.4.1 for an example.

Matrix input and weights: Binomial and multinomial responses are typically supplied as a 2 or K-level factor. As an alternative `glmnet` allows the response to be supplied in matrix form. This allows for *grouped* data, where at each x_i we see a multinomial sample. In this case the rows of the $N \times K$ response matrix represent counts in each category. Alternatively the rows can be proportions summing to one. For the latter case, supplying an observation weight equal to the total count for each observation is equivalent to the first form. Trivially an indicator response matrix is equivalent to supplying the data as a factor, in *ungrouped* form.

Sparse model matrices $\mathbf{X}$*:* Often when $p \gg N$ is very large, there are many zeros in the input matrix $\mathbf{X}$. For example, in document models, each feature vector $x_i \in \mathbb{R}^p$ might count the number of times each word in a very large dictionary occurs in a document. Such vectors and matrices can be stored efficiently by only storing the nonzero values, and then row and column indices of where they occur. Coordinate descent is ideally suited to capitalize on such sparsity, since it handles the variables one-at-a-time, and the principal operation is an inner-product. For example, in Section 3.4.1, the model-matrix $\mathbf{X} = \mathbf{I}$ is the extremely sparse $N \times N$ identity matrix. Even with $N = 10^6$, the program can compute the relaxation path at 100 values of δ in only 27 seconds.

Bibliographic Notes

Generalized linear models were proposed as a comprehensive class of models by Nelder and Wedderburn (1972); see the book by McCullagh and Nelder (1989) for a thorough account. Application of the lasso to logistic regression was proposed in Tibshirani (1996); coordinate descent methods for logistic, multinomial, and Poisson regression were developed in Friedman, Hastie, Hoefling and Tibshirani (2007), Friedman, Hastie and Tibshirani (2010*b*), Wu and Lange (2008), and Wu, Chen, Hastie, Sobel and Lange (2009). Pre-validation was proposed by Tibshirani and Efron (2002). Boser, Guyon and Vapnik (1992) described the support vector machine, with a thorough treatment in Vapnik (1996).

Exercises

Ex. 3.1 Consider a linear logistic regression model with separable data, meaning that the data can be correctly separated into two classes by a hyperplane. Show that the likelihood estimates are unbounded, and that the log-likelihood objective reaches its maximal value of zero. Are the fitted probabilities well-defined?

Ex. 3.2 For a response variable $y \in \{-1, +1\}$ and a linear classification function $f(x) = \beta_0 + \beta^T x$, suppose that we classify according to $\text{sign}(f(x))$. Show that the signed Euclidean distance of the point x with label y to the decision boundary is given by

$$\frac{1}{\|\beta\|_2}\, y\, f(x). \tag{3.45}$$

Ex. 3.3 Here we show that for the multinomial model, the penalty used automatically imposes a normalization on the parameter estimates. We solve this problem for a general elastic-net penalty (Section 4.2). For some parameter $\alpha \in [0, 1]$ consider the problem

$$c_j(\alpha) = \arg\min_{t \in \mathbb{R}} \left\{ \sum_{\ell=1}^{K} \left[\frac{1}{2}(1-\alpha)(\beta_{j\ell} - t)^2 + \alpha|\beta_{j\ell} - t| \right] \right\}. \tag{3.46}$$

Let $\bar{\beta}_j = \frac{1}{K}\sum_{\ell=1}^{K} \beta_{j\ell}$ be the sample mean, and let $\tilde{\beta}_j$ be a sample median. (For simplicity, assume that $\bar{\beta}_j \le \tilde{\beta}_j$). Show that

$$\bar{\beta}_j \;\le\; c_j(\alpha) \;\le\; \tilde{\beta}_j \quad \text{for all } \alpha \in [0, 1] \tag{3.47}$$

with the lower inequality achieved if $\alpha = 0$, and the upper inequality achieved if $\alpha = 1$.

Ex. 3.4 Derive the Lagrange dual (3.25) of the *maximum-entropy* problem (3.24). Note that positivity is automatically enforced, since the log function in the objective (3.24) serves as a barrier. (*Hint:* It may help to introduce additional variables $w_i = p_i - r_i$, and now minimize the criterion (3.24) with respect to both $\{p_i, w_i\}_{i=1}^N$, subject to the additional constraints that $w_i = p_i - r_i$.)

Ex. 3.5 Recall the dual (3.25) of the maximum entropy problem, and the associated example motivating it. Suppose that for each cell, we also measure the value x_k corresponding to the mid-cell ordinate on the continuous domain x. Consider the model

$$q_k = u_k e^{\beta_0 + \sum_{m=1}^{M} \beta_m x_k^m + \alpha_k}, \tag{3.48}$$

and suppose that we fit it using the penalized log-likelihood (3.25) without penalizing any of the coefficients. Show that for the estimated distribution $\hat{\mathbf{q}} = \{\hat{q}_k\}_{k=1}^N$, the moments of X up to order M match those of the empirical distribution $\boldsymbol{r} = \{r_k\}_{k=1}^N$.

Ex. 3.6 Consider the group-lasso-regularized version of multinomial regression (3.20). Suppose that for a particular value of λ, the coefficient $\widehat{\beta}_{kj}$ is *not equal* to 0. Show that $\widehat{\beta}_{\ell j} \neq 0$ for all $\ell \in (1, \ldots, K)$, and moreover that $\sum_{\ell=1}^{K} \widehat{\beta}_{\ell j} = 0$.

Ex. 3.7 This problem also applies to the group-lasso-regularized form of multinomial regression (3.20). Suppose that for a particular value of λ, and the fitted probabilities are $\hat{\pi}_i = (\hat{\pi}_{i1}, \ldots, \hat{\pi}_{iK})^T$. Similarly let $r_i = (r_{i1}, \ldots, r_{iK})^T$ be the observed proportions. Suppose we consider including an additional variable (vector) Z with observed values z_i, and wish to update the fit. Let $g = \sum_{i=1}^{N} z_i(r_i - \hat{\pi}_i)$. Show that if $\|g\|_2 < \lambda$, then the coefficients of Z are zero, and the model remains unchanged.

Ex. 3.8 The *squared hinge loss function* $\phi_{\text{sqh}}(t) := (1-t)_+^2$ can be used as a margin-based loss function $\phi(y\, f(x))$ for binary classification problems.

(a) Show that ϕ_{sqh} is differentiable everywhere.

(b) Suppose $Y \in \{-1, +1\}$ with $\Pr(Y = +1) = \pi \in (0, 1)$. Find the function $f : \mathbb{R}^p \to \mathbb{R}$ that minimizes (for each $x \in \mathbb{R}^p$) the criterion

$$\underset{f}{\text{minimize}}\, \mathbb{E}_Y \left[\phi_{\text{sqh}}\big(Y f(x)\big)\right] \tag{3.49}$$

(c) Repeat part (b) using the usual hinge loss $\phi_{\text{hin}}(t) = (1-t)_+$.

Ex. 3.9 Given binary responses $y_i \in \{-1, +1\}$, consider the ℓ_1-regularized SVM problem

$$(\widehat{\beta}_0,\ \widehat{\beta}) = \underset{\beta_0,\, \beta}{\arg\min} \left\{ \sum_{i=1}^{N} \{1 - y_i f(x_i\,; \beta_0, \beta)\}_+ + \lambda \sum_{j=1}^{p} |\beta_j| \right\}, \tag{3.50}$$

where $f(x\,; \beta_0, \beta) := \beta_0 + \beta^T x$. In this exercise, we compare solutions of this problem to those of weighted ℓ_2-regularized SVM problem: given nonnegative weights $\{w_j\}_{j=1}^p$, we solve

$$(\tilde{\beta}_0,\ \tilde{\beta}) = \underset{\beta_0,\, \beta}{\arg\min} \left\{ \sum_{i=1}^{N} \{1 - y_i f(x_i\,; \beta_0, \beta)\}_+ + \frac{\lambda}{2} \sum_{j=1}^{p} w_j \beta_j^2 \right\}. \tag{3.51}$$

(a) Show that if we solve the problem (3.51) with $w_j = 1/|\widehat{\beta}_j|$, then $(\tilde{\beta}_0, \tilde{\beta}) = (\widehat{\beta}_0, \widehat{\beta})$.

(b) For a given weight sequence $\{w_j\}_{j=1}^p$ with $w_j \in (0, \infty)$ for all $j = 1, \ldots, p$, show how to solve the criterion (3.51) using a regular unweighted SVM solver. What do you do if $w_j = \infty$ for some subset of indices?

(c) In light of the preceding parts, suggest an iterative algorithm for the problem (3.50) using a regular SVM solver.

Chapter 4

Generalizations of the Lasso Penalty

4.1 Introduction

In the previous chapter, we considered some generalizations of the lasso obtained by varying the loss function. In this chapter, we turn to some useful variations of the basic lasso ℓ_1-penalty itself, which expand the scope of the basic model. They all inherit the two essential features of the standard lasso, namely the shrinkage and selection of variables, or groups of variables.

Such generalized penalties arise in a wide variety of settings. For instance, in microarray studies, we often find groups of correlated features, such as genes that operate in the same biological pathway. Empirically, the lasso sometimes does not perform well with highly correlated variables. By combining a squared ℓ_2-penalty with the ℓ_1-penalty, we obtain the *elastic net*, another penalized method that deals better with such correlated groups, and tends to select the correlated features (or not) together. In other applications, features may be structurally grouped. Examples include the dummy variables that are used to code a multilevel categorical predictor, or sets of coefficients in a multiple regression problem. In such settings, it is natural to select or omit all the coefficients within a group together. The *group lasso* and the *overlap group lasso* achieve these effects by using sums of (un-squared) ℓ_2 penalties. Another kind of structural grouping arises from an underlying index set such as time; our parameters might each have an associated time stamp. We might then ask for time-neighboring coefficients to be the same or similar. The *fused lasso* is a method naturally tailored to such situations.

Finally, a variety of nonparametric smoothing methods operate implicitly with large groups of variables. For example, each term in an additive smoothing-spline model has an associated cubic-spline basis. The grouped lasso extends naturally to these situations as well; the COSSO and the SPAM families are examples of such nonparametric models. In summary, all these variants deal with different kinds of groupings of the features in natural ways, and it is the goal of this chapter to explore them in some more detail.

4.2 The Elastic Net

The lasso does not handle highly correlated variables very well; the coefficient paths tend to be erratic and can sometimes show wild behavior. Consider a simple but extreme example, where the coefficient for a variable X_j with a particular value for λ is $\widehat{\beta}_j > 0$. If we augment our data with an *identical* copy $X_{j'} = X_j$, then they can share this coefficient in infinitely many ways—any $\tilde{\beta}_j + \tilde{\beta}_{j'} = \hat{\beta}_j$ with both pieces positive—and the loss and ℓ_1 penalty are indifferent. So the coefficients for this pair are not defined. A quadratic penalty, on the other hand, will divide $\hat{\beta}_j$ exactly equally between these two twins (see Exercise 4.1). In practice, we are unlikely to have an identical

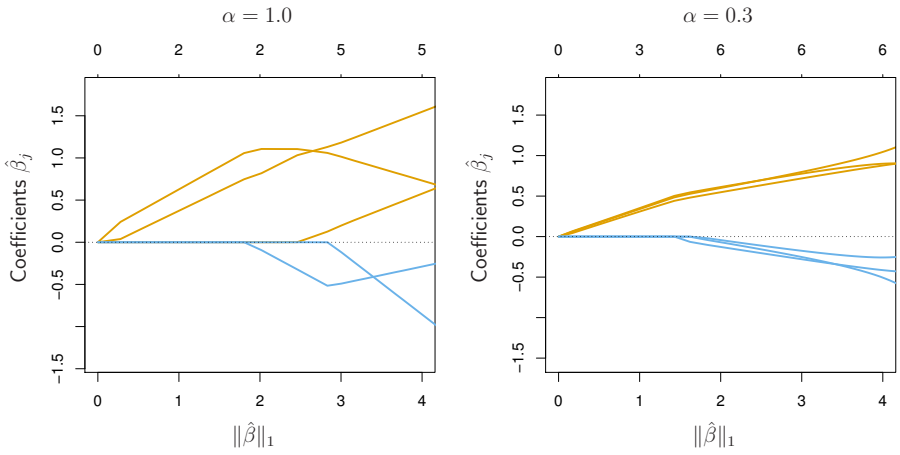


Figure 4.1 *Six variables, highly correlated in groups of three. The lasso estimates ($\alpha = 1$), as shown in the left panel, exhibit somewhat erratic behavior as the regularization parameter λ is varied. In the right panel, the elastic net with ($\alpha = 0.3$) includes all the variables, and the correlated groups are pulled together.*

pair of variables, but often we do have groups of very correlated variables. In microarray studies, groups of genes in the same biological pathway tend to be expressed (or not) together, and hence measures of their expression tend to be strongly correlated. The left panel of Figure 4.1 shows the lasso coefficient path for such a situation. There are two sets of three variables, with pairwise correlations around 0.97 in each group. With a sample size of $N = 100$, the data were simulated as follows:

$$\begin{aligned} &Z_1,\ Z_2 \sim N(0,1) \text{ independent,} \\ &\quad Y = 3 \cdot Z_1 - 1.5Z_2 + 2\varepsilon,\ \text{with } \varepsilon \sim N(0,1), \\ &\quad X_j = Z_1 + \xi_j/5,\ \text{with } \xi_j \sim N(0,1) \text{ for } j = 1,2,3, \text{ and} \\ &\quad X_j = Z_2 + \xi_j/5,\ \text{with } \xi_j \sim N(0,1) \text{ for } j = 4,5,6. \end{aligned} \tag{4.1}$$

As shown in the left panel of Figure 4.1, the lasso coefficients do not reflect the relative importance of the individual variables.

The *elastic net* makes a compromise between the ridge and the lasso penalties (Zou and Hastie 2005); it solves the convex program

$$\underset{(\beta_0,\beta)\in\mathbb{R}\times\mathbb{R}^p}{\text{minimize}} \quad \left\{ \frac{1}{2}\sum_{i=1}^{N}(y_i-\beta_0-x_i^T\beta)^2+\lambda\left[\tfrac{1}{2}(1-\alpha)\|\beta\|_2^2+\alpha\|\beta\|_1\right]\right\}, \quad (4.2)$$

where $\alpha \in [0,1]$ is a parameter that can be varied. By construction, the penalty applied to an individual coefficient (disregarding the regularization weight $\lambda > 0$) is given by

$$\frac{1}{2}(1-\alpha)\beta_j^2+\alpha|\beta_j|. \quad (4.3)$$

When $\alpha = 1$, it reduces to the ℓ_1-norm or lasso penalty, and with $\alpha = 0$, it reduces to the squared ℓ_2-norm, corresponding to the ridge penalty.[1]

Returning to Figure 4.1, the right-hand panel shows the elastic-net coefficient path with $\alpha = 0.3$. We see that in contrast to the lasso paths in the left panel, the coefficients are selected approximately together in their groups, and also approximately share their values equally. Of course, this example is idealized, and in practice the group structure will not be so cleanly evident. But by adding some component of the ridge penalty to the ℓ_1-penalty, the elastic net automatically controls for strong within-group correlations. Moreover, for any $\alpha < 1$ and $\lambda > 0$, the elastic-net problem (4.2) is *strictly convex:* a unique solution exists irrespective of the correlations or duplications in the X_j.

Figure 4.2 compares the constraint region for the elastic net (left image) to that of the lasso (right image) when there are three variables. We see that the elastic-net ball shares attributes of the ℓ_2 ball and the ℓ_1 ball: the sharp corners and edges encourage selection, and the curved contours encourage sharing of coefficients. See Exercise 4.2 for further exploration of these properties.

The elastic net has an additional tuning parameter α that has to be determined. In practice, it can be viewed as a higher-level parameter, and can be set on subjective grounds. Alternatively, one can include a (coarse) grid of values of α in a cross-validation scheme.

The elastic-net problem (4.2) is convex in the pair $(\beta_0, \beta) \in \mathbb{R} \times \mathbb{R}^p$, and a variety of different algorithms can be used to solve it. Coordinate descent is particularly effective, and the updates are a simple extension of those for the lasso in Chapter 2. We have included an unpenalized intercept in the model, which can be dispensed with at the onset; we simply center the covariates x_{ij}, and then the optimal intercept is $\widehat{\beta}_0 = \bar{y} = \frac{1}{N}\sum_{j=1}^{N} y_j$. Having solved for the optimal $\widehat{\beta}_0$, it remains to compute the optimal vector $\widehat{\beta} = (\widehat{\beta}_1, \ldots, \widehat{\beta}_p)$. It can be verified (Exercise 4.3) that the coordinate descent update for the j^{th}

[1]The $\frac{1}{2}$ in the quadratic part of the elastic-net penalty (4.3) leads to a more intuitive soft-thresholding operator in the optimization.

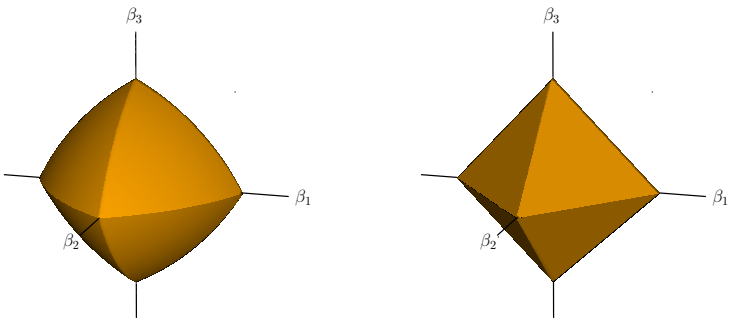


Figure 4.2 *The elastic-net ball with $\alpha = 0.7$ (left panel) in $\mathbb{R}^3$, compared to the ℓ_1 ball (right panel). The curved contours encourage strongly correlated variables to share coefficients (see Exercise 4.2 for details).*

coefficient takes the form

$$\widehat{\beta}_j = \frac{\mathcal{S}_{\lambda\alpha}\big(\sum_{i=1}^N r_{ij}x_{ij}\big)}{\sum_{i=1}^N x_{ij}^2 + \lambda(1-\alpha)}, \tag{4.4}$$

where $\mathcal{S}_\mu\big(z\big) := \text{sign}(z)\,(z-\mu)_+$ is the soft-thresholding operator, and $r_{ij} := y_i - \widehat{\beta}_0 - \sum_{k\neq j} x_{ik}\widehat{\beta}_k$ is the partial residual. We cycle over the updates (4.4) until convergence. Friedman et al. (2015) give more details, and provide an efficient implementation of the elastic net penalty for a variety of loss functions.

4.3 The Group Lasso

There are many regression problems in which the covariates have a natural group structure, and it is desirable to have all coefficients within a group become nonzero (or zero) simultaneously. The various forms of group lasso penalty are designed for such situations. A leading example is when we have qualitative factors among our predictors. We typically code their levels using a set of dummy variables or contrasts, and would want to include or exclude this group of variables together. We first define the group lasso and then develop this and other motivating examples.

Consider a linear regression model involving J groups of covariates, where for $j = 1, \ldots, J$, the vector $Z_j \in \mathbb{R}^{p_j}$ represents the covariates in group j. Our goal is to predict a real-valued response $Y \in \mathbb{R}$ based on the collection of covariates $(Z_1, \ldots, Z_J)$. A linear model for the regression function $\mathbb{E}(Y \mid Z)$

takes the form $\theta_0 + \sum_{j=1}^{J} Z_j^T \theta_j$, where $\theta_j \in \mathbb{R}^{p_j}$ represents a group of p_j regression coefficients.[2]

Given a collection of N samples $\{(y_i, z_{i1}, z_{i,2}, \ldots, z_{i,J})\}_{i=1}^{N}$, the group lasso solves the convex problem

$$\underset{\theta_0 \in \mathbb{R}, \theta_j \in \mathbb{R}^{p_j}}{\text{minimize}} \left\{ \frac{1}{2} \sum_{i=1}^{N} \left(y_i - \theta_0 - \sum_{j=1}^{J} z_{ij}^T \theta_j \right)^2 + \lambda \sum_{j=1}^{J} \|\theta_j\|_2 \right\}, \tag{4.5}$$

where $\|\theta_j\|_2$ is the Euclidean norm of the vector θ_j.

This is a group generalization of the lasso, with the properties:

- depending on $\lambda \geq 0$, either the entire vector $\hat{\theta}_j$ will be zero, or all its elements will be nonzero;[3]
- when $p_j = 1$, then we have $\|\theta_j\|_2 = |\theta_j|$, so if all the groups are singletons, the optimization problem (4.5) reduces to the ordinary lasso.

Figure 4.3 compares the constraint region for the group lasso (left image) to that of the lasso (right image) when there are three variables. We see that the group lasso ball shares attributes of both the ℓ_2 and ℓ_1 balls.

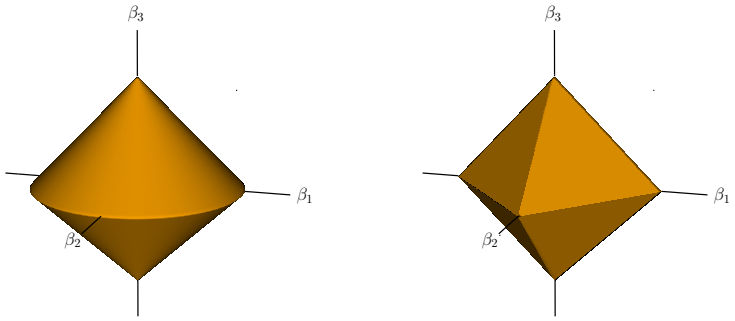


Figure 4.3 *The group lasso ball (left panel) in $\mathbb{R}^3$, compared to the ℓ_1 ball (right panel). In this case, there are two groups with coefficients $\theta_1 = (\beta_1, \beta_2) \in \mathbb{R}^2$ and $\theta_2 = \beta_3 \in \mathbb{R}^1$.*

In the formulation (4.5), all groups are equally penalized, a choice which leads larger groups to be more likely to be selected. In their original proposal, Yuan and Lin (2006) recommended weighting the penalties for each group according to their size, by a factor $\sqrt{p_j}$. In their case, the group matrices $\mathbf{Z}_j$ were orthonormal; for general matrices one can argue for a factor

[2]To avoid confusion, we use Z_j and θ_j to represent groups of variables and their coefficients, rather than the X_j and β_j we have used for scalars.

[3]Nonzero for generic problems, although special structure could result in some coefficients in a group being zero, just as they can for linear or ridge regression.

$\|\mathbf{Z}_j\|_F$ (Exercise 4.5). These choices are somewhat subjective, and are easily accommodated; for simplicity, we omit this modification in our presentation.

We now turn to some examples to illustrate applications of the group lasso (4.5).

Example 4.1. Regression with multilevel factors. When a predictor variable in a linear regression is a multilevel factor, we typically include a separate coefficient for each level of the factor. Take the simple case of one continuous predictor X and a three-level factor G with levels g_1, g_2, and g_3. Our linear model for the mean is

$$\mathbb{E}(Y \mid X, G) = X\beta + \sum_{k=1}^{3} \theta_k \, \mathbb{I}_k[G], \tag{4.6}$$

where $\mathbb{I}_k[G]$ is a 0-1 valued indicator function for the event $\{G = g_k\}$. The model (4.6) corresponds to a linear regression in X with different intercepts θ_k depending on the level of G.

By introducing a vector $Z = (Z_1, Z_2, Z_3)$ of three *dummy variables* with $Z_k = \mathbb{I}_k[G]$, we can write this model as a standard linear regression

$$\mathbb{E}(Y \mid X, G) = \mathbb{E}(Y \mid X, Z) = X\beta + Z^T\theta, \tag{4.7}$$

where $\theta = (\theta_1, \theta_2, \theta_3)$. In this case Z is a group variable that represents the single factor G. If the variable G—as coded by the vector Z—has no predictive power, then the full vector $\theta = (\theta_1, \theta_2, \theta_3)$ should be zero. On the other hand, when G is useful for prediction, then at least generically, we expect that all coefficients of θ are likely to be nonzero. More generally we can have a number of such single and group variables, and so have models of the form

$$\mathbb{E}(Y \mid X, G_1, \ldots, G_J) = \beta_0 + X^T\beta + \sum_{j=1}^{J} Z_j^T\theta_j. \tag{4.8}$$

When selecting variables for such a model we would typically want to include or exclude groups at a time, rather than individual coefficients, and the group lasso is designed to enforce such behavior.

With unpenalized linear regression with factors, one has to worry about aliasing; in the example here, the dummy variables in a set add to one, which is aliased with the intercept term. One would then use contrasts to code factors that enforce, for example, that the coefficients in a group sum to zero. With the group lasso this is not a concern, because of the ℓ_2 penalties. We use the symmetric full representation as above, because the penalty term ensures that the coefficients in a group sum to zero (see Exercise 4.4). ♢

Variables can be grouped for other reasons. For example, in gene-expression arrays, we might have a set of highly correlated genes from the same biological pathway. Selecting the group amounts to selecting a pathway. Figure 4.4 shows the coefficient path for a group-lasso fit to some genomic data for splice-site

detection (Meier, van de Geer and Bühlmann 2008, Section 5). The data arise from human DNA, and each observation consists of seven bases with values $\{A, G, C, T\}^7$. Some of the observations are at exon-intron boundaries (splice sites), and others not, coded in a binary response; see Burge and Karlin (1977) for further details about these data. The regression problem is to predict the binary response Y using the seven four-level factors G_j as predictors, and we use a training sample of 5610 observations in each class.

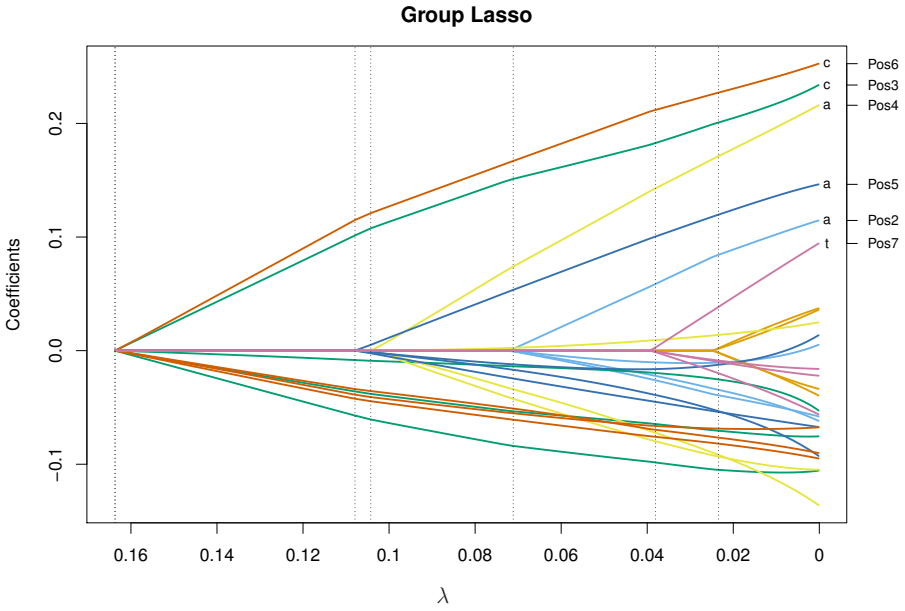


Figure 4.4 *Coefficient profiles from the group lasso, fit to splice-site detection data. The coefficients come in groups of four, corresponding to the nucleotides A, G, C, T. The vertical lines indicate when a group enters. On the right-hand side we label some of the variables; for example, "Pos6" and the level "c". The coefficients in a group have the same color, and they always average zero.*

Example 4.2. Multivariate regression. Sometimes we are interested in predicting a multivariate response $Y \in \mathbb{R}^K$ on the basis of a vector $X \in \mathbb{R}^p$ of predictors (also known as *multitask learning*). Given N observations $\{(y_i, x_i)\}_{i=1}^N$, we let $\mathbf{Y} \in \mathbb{R}^{N\times K}$ and $\mathbf{X} \in \mathbb{R}^{N\times p}$ be matrices with y_i and x_i, respectively, as their i^{th} row. If we assume a linear model for the full collection of data, then it can be written in the form

$$\mathbf{Y} = \mathbf{X}\mathbf{\Theta} + \mathbf{E} \tag{4.9}$$

where $\mathbf{\Theta} \in \mathbb{R}^{p\times K}$ is a matrix of coefficients, and $\mathbf{E} \in \mathbb{R}^{N\times K}$ a matrix of errors.

One way to understand the model (4.9) is as a coupled collection of K standard regression problems in $\mathbb{R}^p$, each sharing the same covariates, in which

the k^{th} column $\theta_k \in \mathbb{R}^p$ of $\mathbf{\Theta}$ is the coefficient vector for the k^{th} problem. Thus, in principle, we could fit a separate regression coefficient vector θ_k for each of the K different problems, using the lasso in the case of a sparse linear model. In many applications, the different components of the response vector $Y \in \mathbb{R}^K$ are strongly related, so that one would expect that the underlying regression vectors would also be related. For instance, in collaborative filtering applications, the different components of Y might represent a given user's preference scores for different categories of objects, such as books, movies, music, and so on, all of which are closely related. For this reason, it is natural—and often leads to better prediction performance—to solve the K regression problems jointly, imposing some type of group structure on the coefficients. In another example, each response might be the daily return of an equity in a particular market sector; hence we have multiple equities, and all being predicted by the same market signals.

As one example, in the setting of sparsity, we might posit that there is an unknown subset $S \subset \{1, 2, \ldots, p\}$ of the covariates that are relevant for prediction, and this same subset is *preserved* across all K components of the response variable. In this case, it would be natural to consider a group lasso penalty, in which the p groups are defined by the rows $\{\theta'_j \in \mathbb{R}^K,\ j = 1, \ldots, p\}$ of the full coefficient matrix $\mathbf{\Theta} \in \mathbb{R}^{p\times K}$. Using this penalty, we then solve the regularized least-squares problem

$$\underset{\mathbf{\Theta}\in\mathbb{R}^{p\times K}}{\text{minimize}} \left\{ \frac{1}{2}\|\mathbf{Y} - \mathbf{X}\mathbf{\Theta}\|_{\text{F}}^2 + \lambda \left(\sum_{j=1}^{p} \|\theta'_j\|_2 \right) \right\}, \tag{4.10}$$

where $\|\cdot\|_{\text{F}}$ denotes the Frobenius norm.[4] This problem is a special case of the general group lasso (4.5), in which $J = p$, and $p_j = K$ for all groups j. ♢

4.3.1 Computation for the Group Lasso

Turning to computational issues associated with the group lasso, let us rewrite the relevant optimization problem (4.5) in a more compact matrix-vector notation:

$$\underset{(\theta_1,\ldots,\theta_J)}{\text{minimize}} \left\{ \frac{1}{2}\|\mathbf{y} - \sum_{j=1}^{J} \mathbf{Z}_j\theta_j\|_2^2 + \lambda \sum_{j=1}^{J} \|\theta_j\|_2 \right\}. \tag{4.11}$$

For simplicity we ignore the intercept θ_0, since in practice we can center all the variables and the response, and it goes away. For this problem, the zero subgradient equations (see Section 5.2.2 take the form

$$-\mathbf{Z}_j^T(\mathbf{y} - \sum_{\ell=1}^{J} \mathbf{Z}_\ell\widehat{\theta}_\ell) + \lambda\widehat{s}_j = 0, \quad \text{for } j = 1, \cdots J, \tag{4.12}$$

[4]The Frobenius norm of a matrix is simply the ℓ_2-norm applied to its entries.

where $\widehat{s}_j \in \mathbb{R}^{p_j}$ is an element of the subdifferential of the norm $\|\cdot\|_2$ evaluated at $\widehat{\theta}_j$. As verified in Exercise 5.5 on page 135, whenever $\widehat{\theta}_j \neq 0$, then we necessarily have $\widehat{s}_j = \widehat{\theta}_j/\|\widehat{\theta}_j\|_2$, whereas when $\widehat{\theta}_j = 0$, then $\widehat{s}_j$ is any vector with $\|\widehat{s}_j\|_2 \leq 1$. One method for solving the zero subgradient equations is by holding fixed all block vectors $\{\widehat{\theta}_k,\ k \neq j\}$, and then solving for $\widehat{\theta}_j$. Doing so amounts to performing block coordinate descent on the group lasso objective function. Since the problem is convex, and the penalty is block separable, it is guaranteed to converge to an optimal solution (Tseng 1993). With all $\{\widehat{\theta}_k,\ k \neq j\}$ fixed, we write

$$-\mathbf{Z}_j^T(\boldsymbol{r}_j - \mathbf{Z}_j\widehat{\theta}_j) + \lambda\widehat{s}_j = 0, \tag{4.13}$$

where $\boldsymbol{r}_j = \mathbf{y} - \sum_{k\neq j}\mathbf{Z}_k\widehat{\theta}_k$ is the j^{th} *partial residual*. From the conditions satisfied by the subgradient $\widehat{s}_j$, we must have $\widehat{\theta}_j = 0$ if $\|\mathbf{Z}_j^T\boldsymbol{r}_j\|_2 < \lambda$, and otherwise the minimizer $\widehat{\theta}_j$ must satisfy

$$\widehat{\theta}_j = \left(\mathbf{Z}_j^T\mathbf{Z}_j + \frac{\lambda}{\|\widehat{\theta}_j\|_2}\mathbf{I}\right)^{-1}\mathbf{Z}_j^T\boldsymbol{r}_j. \tag{4.14}$$

This update is similar to the solution of a ridge regression problem, except that the underlying penalty parameter depends on $\|\widehat{\theta}_j\|_2$. Unfortunately, Equation (4.14) does not have a closed-form solution for $\widehat{\theta}_j$ unless $\mathbf{Z}_j$ is orthonormal. In this special case, we have the simple update

$$\widehat{\theta}_j = \left(1 - \frac{\lambda}{\|\mathbf{Z}_j^T\boldsymbol{r}_j\|_2}\right)_+\mathbf{Z}_j^T\boldsymbol{r}_j, \tag{4.15}$$

where $(t)_+ := \max\{0, t\}$ is the positive part function. See Exercise 4.6 for further details.

Although the original authors (Yuan and Lin 2006) and many others since have made the orthonormality assumption, it has implications that are not always reasonable (Simon and Tibshirani 2012). Exercise 4.8 explores the impact of this assumption on the dummy coding used here for factors. In the general (nonorthonormal case) one has to solve (4.14) using iterative methods, and it reduces to a very simple one-dimensional search (Exercise 4.7).

An alternative approach is to apply the composite gradient methods of Section 5.3.3 to this problem. Doing so leads to an algorithm that is also iterative within each block; at each iteration the block-optimization problem is approximated by an easier problem, for which an update such as (4.15) is possible. In detail, the algorithm would iterate until convergence the updates

$$\omega \leftarrow \widehat{\theta}_j + \nu \cdot \mathbf{Z}_j^T(\boldsymbol{r}_j - \mathbf{Z}_j\widehat{\theta}_j), \text{ and} \tag{4.16a}$$

$$\widehat{\theta}_j \leftarrow \left(1 - \frac{\nu\lambda}{\|\omega\|_2}\right)_+\omega, \tag{4.16b}$$

where ν is a step-size parameter. See Exercise 4.9 for details of this derivation.

4.3.2 Sparse Group Lasso

When a group is included in a group-lasso fit, all the coefficients in that group are nonzero. This is a consequence of the ℓ_2 norm. Sometimes we would like sparsity both with respect to which groups are selected, and which coefficients are nonzero within a group. For example, although a biological pathway may be implicated in the progression of a particular type of cancer, not all genes in the pathway need be active. The *sparse group lasso* is designed to achieve such within-group sparsity.

In order to achieve within-group sparsity, we augment the basic group lasso (4.11) with an additional ℓ_1-penalty, leading to the convex program

$$\underset{\{\theta_j \in \mathbb{R}^{p_j}\}_{j=1}^{J}}{\text{minimize}} \left\{ \frac{1}{2} \|\mathbf{y} - \sum_{j=1}^{J} \mathbf{Z}_j \theta_j\|_2^2 + \lambda \sum_{j=1}^{J} \left[(1-\alpha)\|\theta_j\|_2 + \alpha \|\theta_j\|_1 \right] \right\}, \quad (4.17)$$

with $\alpha \in [0,1]$. Much like the elastic net of Section 4.2, the parameter α creates a bridge between the group lasso ($\alpha = 0$) and the lasso ($\alpha = 1$). Figure 4.5 contrasts the group lasso constraint region with that of the sparse group lasso for the case of three variables. Note that in the two horizontal axes, the constraint region resembles that of the elastic net.

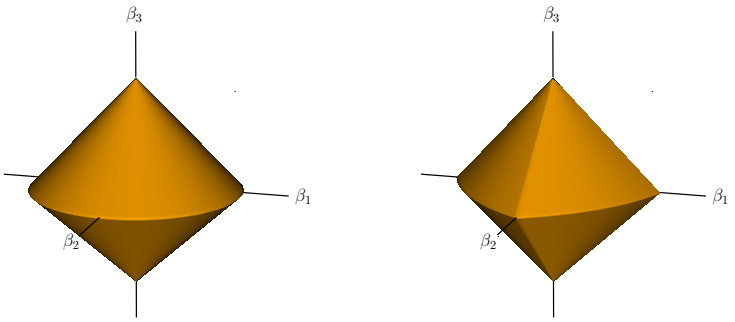


Figure 4.5 *The group lasso ball (left panel) in $\mathbb{R}^3$, compared to the sparse group-lasso ball with $\alpha = 0.5$ (right panel). Depicted are two groups with coefficients $\theta_1 = (\beta_1, \beta_2) \in \mathbb{R}^2$ and $\theta_2 = \beta_3 \in \mathbb{R}^1$.*

Since the optimization problem (4.17) is convex, its optima are specified by zero subgradient equations, similar to (4.13) for the group lasso. More precisely, any optimal solution must satisfy the condition

$$-\mathbf{Z}_j^T(\mathbf{y} - \sum_{\ell=1}^{J} \mathbf{Z}_\ell \widehat{\theta}_\ell) + \lambda(1-\alpha) \cdot \widehat{s}_j + \lambda\alpha \widehat{t}_j = 0, \text{ for } j = 1, \cdots, J, \quad (4.18)$$

where $\widehat{s}_j \in \mathbb{R}^{p_j}$ belongs to the subdifferential of the Euclidean norm at $\widehat{\theta}_j$,

and $\widehat{t}_j \in \mathbb{R}^{p_j}$ belongs to the subdifferential of the ℓ_1-norm at $\widehat{\theta}_j$; in particular, we have each $\widehat{t}_{jk} \in \text{sign}(\theta_{jk})$ as with the usual lasso.

We once again solve these equations via block-wise coordinate descent, although the solution is a bit more complex than before. As in Equation (4.13), with $\boldsymbol{r}_j$ the partial residual in the j^{th} coordinate, it can be seen that $\widehat{\theta}_j = 0$ if and only if the equation

$$\mathbf{Z}_j^T \boldsymbol{r}_j = \lambda(1-\alpha)\widehat{s}_j + \lambda\alpha\,\widehat{t}_j \tag{4.19}$$

has a solution with $\|\widehat{s}_j\|_2 \leq 1$ and $\widehat{t}_{jk} \in [-1, 1]$ for $k = 1, \ldots, p_j$. Fortunately, this condition is easily checked, and we find that (Exercise 4.12)

$$\widehat{\theta}_j = 0 \quad \text{if and only if} \quad \|\mathcal{S}_{\lambda\alpha}\big(\mathbf{Z}_j^T \boldsymbol{r}_j\big)\|_2 \leq \lambda(1-\alpha), \tag{4.20}$$

where $\mathcal{S}_{\lambda\alpha}\big(\,\cdot\,\big)$ is the soft-thresholding operator applied here componentwise to its vector argument $\mathbf{Z}_j^T \boldsymbol{r}_j$. Notice the similarity with the conditions for the group lasso (4.13), except here we use the soft-thresholded gradient $\mathcal{S}_{\lambda\alpha}\big(\mathbf{Z}_j^T \boldsymbol{r}_j\big)$. Likewise, if $\mathbf{Z}_j^T \mathbf{Z}_j = \mathbf{I}$, then as shown in Exercise 4.13, we have

$$\widehat{\theta}_j = \left(1 - \frac{\lambda(1-\alpha)}{\|\mathcal{S}_{\lambda\alpha}\big(\mathbf{Z}_j^T \boldsymbol{r}_j\big)\|_2}\right)_+ \mathcal{S}_{\lambda\alpha}\big(\mathbf{Z}_j^T \boldsymbol{r}_j\big). \tag{4.21}$$

In the general case when the $\mathbf{Z}_j$ are not orthonormal and we have checked that $\widehat{\theta}_j \neq 0$, finding $\widehat{\theta}_j$ amounts to solving the subproblem

$$\underset{\theta_j \in \mathbb{R}^{p_j}}{\text{minimize}} \left\{ \frac{1}{2}\|\boldsymbol{r}_j - \mathbf{Z}_j\theta_j\|_2^2 + \lambda(1-\alpha)\|\theta_j\|_2 + \lambda\alpha\|\theta_j\|_1 \right\}. \tag{4.22}$$

Here we can again use generalized gradient descent (Section (5.3.3)) to produce a simple iterative algorithm to solve each block, as in Equation (4.16a). The algorithm would iterate until convergence the sequence

$$\omega \leftarrow \widehat{\theta}_j + \nu \cdot \mathbf{Z}_j^T(\boldsymbol{r}_j - \mathbf{Z}_j\widehat{\theta}_j), \text{ and} \tag{4.23a}$$

$$\theta_j \leftarrow \left(1 - \frac{\nu\lambda(1-\alpha)}{\|\mathcal{S}_{\lambda\alpha}\big(\omega\big)\|_2}\right)_+ \mathcal{S}_{\lambda\alpha}\big(\omega\big), \tag{4.23b}$$

where ν is the step size. See Exercise 4.10 for the details.

4.3.3 The Overlap Group Lasso

Sometimes variables can belong to more than one group: for example, genes can belong to more than one biological pathway. The *overlap group lasso* is a modification that allows variables to contribute to more than one group.

To gain some intuition, consider the case of $p = 5$ variables partitioned into two groups, say of the form

$$Z_1 = (X_1, X_2, X_3), \text{ and } Z_2 = (X_3, X_4, X_5). \tag{4.24}$$

Here X_3 belongs to both groups. The overlap group lasso simply replicates a variable in whatever group it appears, and then fits the ordinary group lasso as before. In this particular example, the variable X_3 is replicated, and we fit coefficient vectors $\theta_1 = (\theta_{11}, \theta_{12}, \theta_{13})$ and $\theta_2 = (\theta_{21}, \theta_{22}, \theta_{23})$ using the group lasso (4.5), using a group penalty $\|\theta_1\|_2 + \|\theta_2\|_2$. In terms of the original variables, the coefficient $\widehat{\beta}_3$ of X_3 is given by the sum $\widehat{\beta}_3 = \widehat{\theta}_{13} + \widehat{\theta}_{21}$. As a consequence, the coefficient $\widehat{\beta}_3$ can be nonzero if either (or both) of the coefficients $\widehat{\theta}_{13}$ or $\widehat{\theta}_{21}$ are nonzero. Hence, all else being equal, the variable X_3 has a better chance of being included in the model than the other variables, by virtue of belonging to two groups.

Rather than replicate variables, it is tempting to simply replicate the coefficients in the group-lasso penalty. For instance, for the given grouping above, with $X = (X_1, \ldots, X_5)$, and $\beta = (\beta_1, \ldots, \beta_5)$, suppose that we define

$$\theta_1 = (\beta_1, \beta_2, \beta_3), \quad \text{and} \quad \theta_2 = (\beta_3, \beta_4, \beta_5), \tag{4.25}$$

and then apply the group-lasso penalty $\|\theta_1\|_2 + \|\theta_2\|_2$ as before. However, this approach has a major drawback. Whenever $\widehat{\theta}_1 = 0$ in any optimal solution, then we must necessarily have $\widehat{\beta}_3 = 0$ in *both groups*. Consequently, in this particular example, the only possible sets of nonzero coefficients are $\{1, 2\}$, $\{4, 5\}$, and $\{1, 2, 3, 4, 5\}$; the original groups $\{1, 2, 3\}$ and $\{3, 4, 5\}$ are not considered as possibilities, since if either group appears, then both groups must appear.[5] As a second practical point, the penalty in this approach is not separable, and hence coordinate-descent algorithms may fail to converge to an optimal solution (see Section 5.4 for more details).

Jacob, Obozinski and Vert (2009) recognized this problem, and hence proposed the replicated variable approach (4.24) or *overlap group lasso*. For our motivating example, the possible sets of nonzero coefficients for the overlap group lasso are $\{1, 2, 3\}$, $\{3, 4, 5\}$, and $\{1, 2, 3, 4, 5\}$. In general, the sets of possible nonzero coefficients always correspond to groups, or the unions of groups. They also defined an implicit penalty on the original variables that yields the replicated variable approach as its solution, which we now describe.

Denote by $\nu_j \in \mathbb{R}^p$ a vector which is zero everywhere except in those positions corresponding to the members of group j, and let $\mathcal{V}_j \subseteq \mathbb{R}^p$ be the subspace of such possible vectors. In terms of the original variables, $X = (X_1, \cdots, X_p)$, the coefficient vector is given by the sum $\beta = \sum_{j=1}^{J} \nu_j$, and hence the overlap group lasso solves the problem

$$\underset{\nu_j \in \mathcal{V}_j \ j=1,\ldots,J}{\text{minimize}} \left\{ \frac{1}{2} \big\| \mathbf{y} - \mathbf{X}\big(\sum_{j=1}^{J} \nu_j \big) \big\|_2^2 + \lambda \sum_{j=1}^{J} \|\nu_j\|_2 \right\}. \tag{4.26}$$

This optimization problem can be re-cast in the terms of the original β vari-

[5] More generally, the replicated-variable approach always yields solutions in which the sets of zero coefficients are unions of groups, so that the sets of nonzeros must be the intersections of complements of groups.

ables by defining a suitable penalty function. With

$$\Omega_{\mathcal{V}}(\beta) := \inf_{\substack{\nu_j \in \mathcal{V}_j \\ \beta = \sum_{j=1}^{J} \nu_j}} \sum_{j=1}^{J} \|\nu_j\|_2, \tag{4.27}$$

it can then be shown (Jacob et al. 2009) that solving problem (4.26) is equivalent to solving

$$\underset{\beta \in \mathbb{R}^p}{\text{minimize}} \left\{ \frac{1}{2} \|\mathbf{y} - \mathbf{X}\beta\|_2^2 + \lambda\, \Omega_{\mathcal{V}}(\beta) \right\}. \tag{4.28}$$

This equivalence is intuitively obvious, and underscores the mechanism underlying this penalty; the contributions to the coefficient for a variable are distributed among the groups to which it belongs in a norm-efficient manner.

Figure 4.6 contrasts the group lasso constraint region with that of the overlap group lasso when there are three variables. There are two rings corresponding to the two groups, with X_2 in both groups.

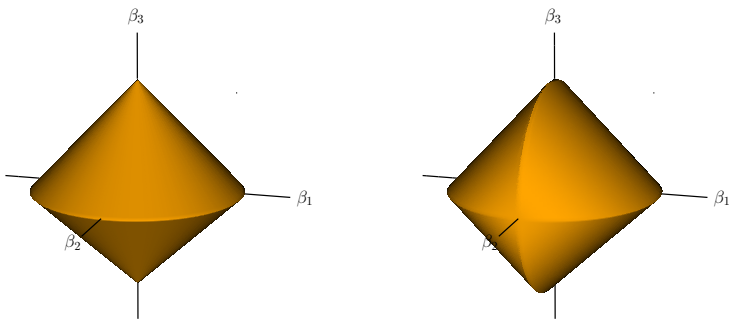


Figure 4.6 *The group-lasso ball (left panel) in $\mathbb{R}^3$, compared to the overlap-group-lasso ball (right panel). Depicted are two groups in both. In the left panel the groups are $\{X_1, X_2\}$ and X_3; in the right panel the groups are $\{X_1, X_2\}$ and $\{X_2, X_3\}$. There are two rings corresponding to the two groups in the right panel. When β_2 is close to zero, the penalty on the other two variables is much like the lasso. When β_2 is far from zero, the penalty on the other two variables "softens" and resembles the ℓ_2 penalty.*

Example 4.3. Interactions and hierarchy. The overlap-group lasso can also be used to enforce *hierarchy* when selecting interactions in linear models. What this means is that interactions are allowed in the model only in the presence of both of their main effects. Suppose Z_1 represents the p_1 dummy variables for the p_1 levels of factor G_1; likewise Z_2 the p_2 dummy variables for G_2. A linear model with Z_1 and Z_2 is a *main-effects* model. Now let $Z_{1:2} = Z_1 \star Z_2$,

a $p_1 \times p_2$ vector of dummy variables (the vector of all pairwise products). Lim and Hastie (2014) consider the following formulation for a pair of such categorical variables[6]

$$\underset{\mu,\alpha,\tilde{\alpha}}{\text{minimize}} \left\{ \frac{1}{2} \left\| \mathbf{y} - \mu\mathbf{1} - \mathbf{Z}_1\alpha_1 - \mathbf{Z}_2\alpha_2 - [\mathbf{Z}_1\ \mathbf{Z}_2\ \mathbf{Z}_{1:2}] \begin{bmatrix} \tilde{\alpha}_1 \\ \tilde{\alpha}_2 \\ \alpha_{1:2} \end{bmatrix} \right\|_2^2 + \lambda \left(\|\alpha_1\|_2 + \|\alpha_2\|_2 + \sqrt{p_2\|\tilde{\alpha}_1\|_2^2 + p_1\|\tilde{\alpha}_2\|_2^2 + \|\alpha_{1:2}\|_2^2} \right) \right\} \quad (4.29)$$

subject to the constraints

$$\sum_{i=1}^{p_1} \alpha_1^i = 0, \quad \sum_{j=1}^{p_2} \alpha_2^j = 0, \quad \sum_{i=1}^{p_1} \tilde{\alpha}_1^i = 0, \quad \sum_{j=1}^{p_2} \tilde{\alpha}_2^j = 0, \quad (4.30)$$

$$\sum_{i=1}^{p_1} \alpha_{1:2}^{ij} = 0 \text{ for fixed } j, \quad \sum_{j=1}^{p_2} \alpha_{1:2}^{ij} = 0 \text{ for fixed } i. \quad (4.31)$$

The summation constraints are standard in hierarchical ANOVA formulations. Notice that the main effect matrices $\mathbf{Z}_1$ and $\mathbf{Z}_2$ each have two different coefficient vectors α_j and $\tilde{\alpha}_j$, creating an overlap in the penalties, and their ultimate coefficient is the sum $\theta_j = \alpha_j + \tilde{\alpha_j}$. The $\sqrt{p_2\|\tilde{\alpha}_1\|_2^2 + p_1\|\tilde{\alpha}_2\|_2^2 + \|\alpha_{1:2}\|_2^2}$ term results in estimates that satisfy strong hierarchy, because either $\widehat{\tilde{\alpha}}_1 = \widehat{\tilde{\alpha}}_2 = \widehat{\alpha}_{1:2} = 0$ or *all* are nonzero, i.e., interactions are always present with both main effects. They show that the solution to the above constrained problem (4.29)–(4.31) is equivalent to the solution to the simpler unconstrained problem

$$\underset{\mu,\beta}{\text{minimize}} \left\{ \frac{1}{2} \|\mathbf{y} - \mu\mathbf{1} - \mathbf{Z}_1\beta_1 - \mathbf{Z}_2\beta_2 - \mathbf{Z}_{1:2}\beta_{1:2}\|_2^2 + \lambda \left(\|\beta_1\|_2 + \|\beta_2\|_2 + \|\beta_{1:2}\|_2 \right) \right\} \quad (4.32)$$

(Exercise 4.14). In other words, a linear model in $Z_{1:2}$ is the full interaction model (i.e., interactions with main effects implicitly included). A group lasso in Z_1, Z_2, and $Z_{1:2}$ will hence result in a hierarchical model; whenever $Z_{1:2}$ is in the model, the pair of main effects is implicitly included. In this case the variables do not strictly overlap, but their subspaces do. A different approach to the estimation of hierarchical interactions is the *hierNet* proposal of Bien, Taylor and Tibshirani (2013). ♢

[6]This extends naturally to more than two pairs, as well as other loss functions, e.g., logistic regression, as well as interactions between factors and quantitative variables.

4.4 Sparse Additive Models and the Group Lasso

Suppose we have a zero-mean response variable $Y \in \mathbb{R}$, and a vector of predictors $X \in \mathbb{R}^J$, and that we are interested in estimating the regression function $f(x) = \mathbb{E}(Y \mid X = x)$. It is well-known that nonparametric regression suffers from the curse of dimensionality, so that approximations are essential. Additive models are one such approximation, and effectively reduce the estimation problem to that of many one-dimensional problems. When J is very large, this may not be sufficient; the class of sparse additive models limits these approximations further, by encouraging many of the components to be zero. Methods for estimating sparse additive models are closely related to the group lasso.

4.4.1 Additive Models and Backfitting

We begin by introducing some background on the class of additive models, which are based on approximating the regression function by sums of the form

$$f(x) = f(x_1, \ldots, x_J) \approx \sum_{j=1}^{J} f_j(x_j), \qquad f_j \in \mathcal{F}_j,\ j = 1, \ldots, J, \tag{4.33}$$

where the $\mathcal{F}_j$ are a fixed set of univariate function classes. Typically, each $\mathcal{F}_j$ is assumed to be a subset of $L^2(\mathbb{P}_j)$ where $\mathbb{P}_j$ is the distribution of covariate X_j, and equipped with the usual squared $L^2(\mathbb{P}_j)$ norm $\|f_j\|_2^2 := \mathbb{E}[f_j^2(X_j)]$.

In the population setting, the best additive approximation to the regression function $\mathbb{E}(Y|X = x)$, as measured in the $L^2(\mathbb{P})$ sense, solves the problem

$$\underset{f_j \in \mathcal{F}_j,\ j=1,\ldots,J}{\text{minimize}} \ \mathbb{E}\big[\big(Y - \sum_{j=1}^{J} f_j(X_j)\big)^2\big]. \tag{4.34}$$

The optimal solution $(\tilde{f}_1, \ldots, \tilde{f}_J)$ is characterized by the *backfitting equations*, namely

$$\tilde{f}_j(x_j) = \mathbb{E}\big[Y - \sum_{k \neq j} \tilde{f}_k(X_k) \mid X_j = x_j\big], \text{ for } j = 1, \ldots, J. \tag{4.35}$$

More compactly, this update can be written in the form $\tilde{f}_j = \mathcal{P}_j(R_j)$, where $\mathcal{P}_j$ is the conditional-expectation operator in the j^{th} coordinate, and the quantity $R_j := Y - \sum_{k \neq j} \tilde{f}_k(X_k)$ is the j^{th} partial residual.

Given data $\{(x_i, y_i)\}_{i=1}^N$, a natural approach is to replace the population operator $\mathcal{P}_j$ with empirical versions, such as scatterplot smoothers $\mathcal{S}_j$, and then solve a data-based version version of the updates (4.35) by coordinate descent or *backfitting* (Hastie and Tibshirani 1990). Hence we repeatedly cycle over the coordinates $j = 1, \ldots, J$, and update each function estimate $\widehat{f}_j$ using

the smooth of the partial residuals

$$\widehat{f}_j \leftarrow \mathcal{S}_j(\mathbf{y} - \sum_{k \neq j} \widehat{\mathbf{f}}_k),\ j = 1, \ldots, J, \tag{4.36}$$

until the fitted functions $\widehat{f}_j$ stabilize. In (4.36) $\widehat{\mathbf{f}}_k$ is the fitted function $\hat{f}_k$ evaluated at the N sample values $(x_{1k}, \ldots, x_{Nk})$. The operator $\mathcal{S}_j$ represents an algorithm that takes a response vector $\boldsymbol{r}$, *smooths* it against the vector $\mathbf{x}_j$, and returns the function $\widehat{f}_j$. Although $\mathcal{S}_j$ will have its own tuning parameters and *bells and whistles*, for the moment we regard it as a black-box that estimates a conditional expectation using data.

4.4.2 Sparse Additive Models and Backfitting

An extension of the basic additive model is the notion of a *sparse additive model*, in which we assume that there is a subset $S \subset \{1, 2, \ldots, J\}$ such that the regression function $f(x) = \mathbb{E}(Y \mid X = x)$ satisfies an approximation of the form $f(x) \approx \sum_{j \in S} f_j(x_j)$. Ravikumar, Liu, Lafferty and Wasserman (2009) proposed a natural extension of the backfitting equations, motivated by a sparse analog of the population level problem (4.34). For a given sparsity level $k \in \{1, \ldots, J\}$, the best k-sparse approximation to the regression function is given by

$$\underset{\substack{|S|=k \\ f_j \in \mathcal{F}_j, j=1,\ldots,J}}{\text{minimize}} \ \mathbb{E}\Big(Y - \sum_{j \in S} f_j(X_j)\Big)^2. \tag{4.37}$$

Unfortunately, this criterion is nonconvex and computationally intractable, due to combinatorial number—namely $\binom{J}{k}$—of possible subsets of size k. Suppose that instead we measure the sparsity of an additive approximation $f = \sum_{j=1}^J f_j$ via the sum $\sum_{j=1}^J \|f_j\|_2$, where we recall that $\|f_j\|_2 = \sqrt{\mathbb{E}[f_j^2(X_j)]}$ is the $L^2(\mathbb{P}_j)$ norm applied to component j. For a given regularization parameter $\lambda \geq 0$, this relaxed notion defines an alternative type of best sparse approximation, namely one that minimizes the penalized criterion

$$\underset{f_j \in \mathcal{F}_j,\ j=1,\ldots,J}{\text{minimize}} \left\{ \mathbb{E}\Big(Y - \sum_{j=1}^J f_j(X_j)\Big)^2 + \lambda \sum_{j=1}^J \|f_j\|_2 \right\}. \tag{4.38}$$

Since this objective is a convex functional of $(f_1, \ldots, f_J)$, Lagrangian duality ensures that it has an equivalent representation involving an explicit constraint on the norm $\sum_{j=1}^J \|f_j\|_2$. See Exercise 4.15.

Ravikumar et al. (2009) show that any optimal solution $(\tilde{f}_1, \ldots, \tilde{f}_J)$ to the penalized problem (4.38) is characterized by the *sparse backfitting equations*

$$\tilde{f}_j = \left(1 - \frac{\lambda}{\|\mathcal{P}_j(R_j)\|_2}\right)_+ \mathcal{P}_j(R_j), \tag{4.39}$$

where the residual R_j and the conditional expectation operator $\mathcal{P}_j$ were defined in the text after the ordinary backfitting equations (4.35).

In parallel with our earlier development, given data $\{(x_i, y_i)\}_1^N$, these population-level updates suggest the natural data-driven analog, in which we replace the population operator $\mathcal{P}_j$ with the scatterplot smoother $\mathcal{S}_j$, and then perform the updates

$$\widetilde{f}_j = \mathcal{S}_j\big(\mathbf{y} - \sum_{k\neq j} \widehat{\mathbf{f}}_k\big), \quad \text{and} \quad \widehat{f}_j = \left(1 - \frac{\lambda}{\|\widetilde{\mathbf{f}}\|_2}\right)_+ \widetilde{f}_j, \tag{4.40}$$

for $j = 1, \ldots, J$, again iterating until convergence. Figure 4.7 illustrates the performance of the SPAM updates (4.40) on some air-pollution data. We use smoothing-splines, with a fixed degree of freedom $df = 5$ for each coordinate (Hastie and Tibshirani 1990).

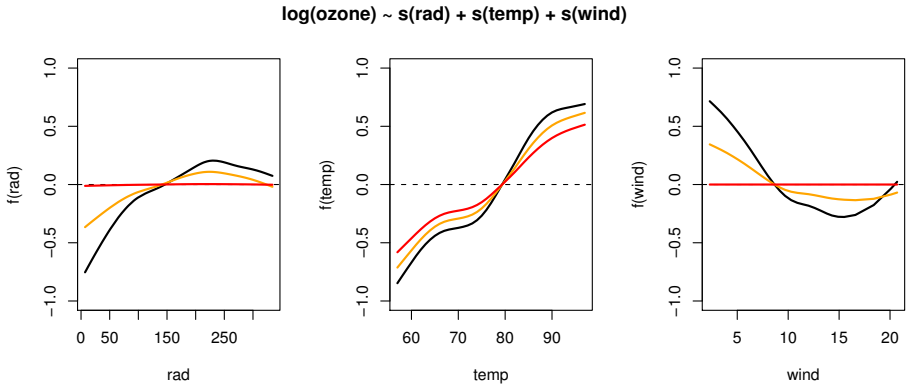


Figure 4.7 *A sequence of three SPAM models fit to some air-pollution data. The response is the log of ozone concentration, and there are three predictors: radiation, temperature, and wind speed. Smoothing splines were used in the additive model fits, each with df* $= 5$ *The three curves in each plot correspond to* $\lambda = 0$ *(black curves),* $\lambda = 2$ *(orange curves), and* $\lambda = 4$ *(red curves). We see that while the shrinkage leaves the functions of* `temp` *relatively untouched, it has a more dramatic effect on* `rad` *and* `wind`*.*

We can make a more direct connection with the grouped lasso if the smoothing method for variable X_j is a projection on to a set of basis functions. Consider

$$f_j(\cdot) = \sum_{\ell=1}^{p_j} \psi_{j\ell}(\cdot)\beta_{j\ell}, \tag{4.41}$$

where the $\{\psi_{j\ell}\}_1^{p_j}$ are a family of basis functions in X_j, such as cubic splines with a collection of knots along the range of X_j. Let $\mathbf{\Psi}_j$ be the $N \times p_j$ matrix of evaluations of the $\psi_{j\ell}$, and assume that $\mathbf{\Psi}_j^T\mathbf{\Psi}_j = \mathbf{I}_{p_j}$. Then for

any coefficient vector $\theta_j = (\beta_{j1}, \ldots, \beta_{jp_j})^T$ and corresponding fitted vector $\mathbf{f}_j = \mathbf{\Psi}_j \theta_j$, we have $\|\mathbf{f}_j\|_2 = \|\theta_j\|_2$. In this case it is easy to show that the updates (4.40) are equivalent to those for a group lasso with predictor matrix $\mathbf{\Psi} := \begin{bmatrix} \mathbf{\Psi}_1 & \mathbf{\Psi}_2 & \cdots & \mathbf{\Psi}_J \end{bmatrix}$ and a corresponding block vector of coefficients $\theta := \begin{bmatrix} \theta_1 & \theta_2 & \cdots \theta_J \end{bmatrix}$ (see Exercise 4.16 for more details).

4.4.3 Approaches Using Optimization and the Group Lasso

Although the population-level sparse backfitting equations (4.39) do solve an optimization problem, in general, the empirical versions (4.40) do not, but rather are motivated by analogy to the population version. We now discuss the *Component Selection and Smoothing Operator* or COSSO for short, which does solve a data-defined optimization problem. The COSSO method (Lin and Zhang 2003) is a predecessor to the SPAM method, and operates in the world of reproducing kernel Hilbert spaces, with a special case being the smoothing spline model.

We begin by recalling the traditional form of an additive smoothing-spline model, obtained from the optimization of a penalized objective function:

$$\underset{f_j \in \mathcal{H}_j,\ j=1,\ldots,J}{\text{minimize}} \left\{ \frac{1}{N} \sum_{i=1}^{N} (y_i - \sum_{j=1}^{J} f_j(x_{ij}))^2 + \lambda \sum_{j=1}^{J} \frac{1}{\gamma_j} \|f_j\|_{\mathcal{H}_j}^2 \right\}. \tag{4.42}$$

Here $\|f_j\|_{\mathcal{H}_j}$ is an appropriate Hilbert-space norm for the j^{th} coordinate. Typically, the Hilbert space $\mathcal{H}_j$ is chosen to enforce some type of smoothness, in which context the parameter $\lambda \geq 0$ corresponds to overall smoothness, and the parameters $\gamma_j \geq 0$ are coordinate specific modifiers. For example, a roughness norm for a cubic smoothing spline on $[0, 1]$ is

$$\|g\|_{\mathcal{H}}^2 := \left(\int_0^1 g(t)dt \right)^2 + \left(\int_0^1 g'(t)dt \right)^2 + \int_0^1 g''(t)^2 dt. \tag{4.43}$$

When this particular Hilbert norm is used in the objective function (4.42), each component $\widehat{f}_j$ of the optimal solution is a cubic spline with knots at the unique sample values of X_j. The solution can be computed by the backfitting updates (4.36), where each $\mathcal{S}_j$ is a type of cubic spline smoother with penalty λ/γ_j.

Instead of the classical formulation (4.42), the COSSO method is based on the objective function

$$\underset{f_j \in \mathcal{H}_j,\ j=1,\ldots,J}{\text{minimize}} \left\{ \frac{1}{N} \sum_{i=1}^{N} (y_i - \sum_{j=1}^{J} f_j(x_{ij}))^2 + \tau \sum_{j=1}^{J} \|f_j\|_{\mathcal{H}_j} \right\}. \tag{4.44}$$

As before, the penalties are norms rather than squared norms, and as such result in coordinate selection for sufficiently large τ. Note that, unlike the

usual penalty for a cubic smoothing spline, the norm in (4.43) includes a linear component; this ensures that the entire function is zero when the term is selected out of the model, rather than just its nonlinear component. Despite the similarity with the additive spline problem (4.38), the structure of the penalty $\|f_j\|_{\mathcal{H}_j}$ means that the solution is not quite as simple as the sparse backfitting equations (4.40).

Equipped with the norm (4.43), the space $\mathcal{H}_j$ of cubic splines is a particular instance of a reproducing-kernel Hilbert space (RKHS) on the unit interval $[0,1]$. Any such space is characterized by a symmetric positive definite kernel function $\mathcal{R}_j : [0,1] \times [0,1] \to \mathbb{R}$ with the so-called reproducing property. In particular, we are guaranteed for each $x \in [0,1]$, the function $\mathcal{R}_j(\cdot, x)$ is a member of $\mathcal{H}_j$, and moreover that $\langle \mathcal{R}(\cdot, x),\, f\rangle_{\mathcal{H}_j} = f(x)$ for all $f \in \mathcal{H}_j$. Here $\langle \cdot,\, \cdot \rangle_{\mathcal{H}_j}$ denotes the inner product on the Hilbert space $\mathcal{H}_j$.

Using the reproducing property, it can be shown (Exercise 4.17) that the j^{th} coordinate function $\widehat{f}_j$ in any optimal COSSO solution can be written in the form $\widehat{f}_j(\cdot) = \sum_{i=1}^N \widehat{\theta}_{ij} \mathcal{R}_j(\cdot,\, x_{ij})$, for a suitably chosen weight vector $\widehat{\boldsymbol{\theta}}_j \in \mathbb{R}^N$. Moreover, it can be shown that $\widehat{f}_j$ has Hilbert norm $\|\widehat{f}_j\|^2_{\mathcal{H}_j} = \widehat{\boldsymbol{\theta}}_j^T \mathbf{R}_j \widehat{\boldsymbol{\theta}}_j$, where $\mathbf{R}_j \in \mathbb{R}^{N \times N}$ is a *Gram matrix* defined by the kernel—in particular, with entries $(\mathbf{R}_j)_{ii'} = \mathcal{R}_j(x_{ij}, x_{i'j})$. Consequently, the COSSO problem (4.44) can be rewritten as a more general version of the group lasso: in particular, it is equivalent to the optimization problem

$$\underset{\boldsymbol{\theta}_j \in \mathbb{R}^N,\, j=1,\ldots,J}{\text{minimize}} \left\{ \frac{1}{N} \|\mathbf{y} - \sum_{j=1}^J \mathbf{R}_j \boldsymbol{\theta}_j\|_2^2 + \tau \sum_{j=1}^J \sqrt{\boldsymbol{\theta}_j^T \mathbf{R}_j \boldsymbol{\theta}_j} \right\}, \tag{4.45}$$

as verified in Exercise 4.17.

We are now back in a parametric setting, and the solution is a more general version of the group lasso (4.14). It can be shown that any optimal solution $(\widehat{\boldsymbol{\theta}}_1, \ldots, \widehat{\boldsymbol{\theta}}_J)$ is specified by the fixed point equations

$$\widehat{\boldsymbol{\theta}}_j = \begin{cases} \mathbf{0} & \text{if } \sqrt{\boldsymbol{r}_j^T \mathbf{R}_j \boldsymbol{r}_j} < \tau \\ \left(\mathbf{R}_j + \frac{\tau}{\sqrt{\widehat{\boldsymbol{\theta}}_j^T \mathbf{R}_j \widehat{\boldsymbol{\theta}}_j}} \mathbf{I} \right)^{-1} \boldsymbol{r}_j & \text{otherwise,} \end{cases} \tag{4.46}$$

where $\boldsymbol{r}_j := \mathbf{y} - \sum_{k \neq j} \mathbf{R}_k \widehat{\boldsymbol{\theta}}_k$ corresponds to the j^{th} partial residual. Although $\widehat{\boldsymbol{\theta}}_j$ appears in both sides of the Equation (4.46), it can be solved with a one-time SVD of $\mathbf{R}_j$ and a simple one-dimensional search; see Exercise 4.7 for the details.

Lin and Zhang (2003) propose an alternative approach, based on introducing a vector $\gamma \in \mathbb{R}^J$ of auxiliary variables, and then considering the joint

optimization problem

$$\underset{\substack{\gamma \geq 0 \\ f_j \in \mathcal{H}_j,\, j=1,\ldots J}}{\text{minimize}} \left\{ \frac{1}{N} \sum_{i=1}^{N} (y_i - \sum_{j=1}^{J} f_j(x_{ij}))^2 + \sum_{j=1}^{J} \frac{1}{\gamma_j} \|f_j\|_{\mathcal{H}_j}^2 + \lambda \sum_{j=1}^{J} \gamma_j \right\}. \tag{4.47}$$

As shown in Exercise 4.18, if we set $\lambda = \tau^4/4$ in the lifted problem (4.47), then the $\widehat{\boldsymbol{\theta}}$ component of any optimal solution coincides with an optimal solution of the original COSSO (4.44).

The reformulation (4.47) is useful, because it naturally leads to a convenient algorithm that alternates between two steps:

- For γ_j fixed, the problem is a version of our earlier objective (4.42), and results in an additive-spline fit.
- With the fitted additive spline fixed, updating the vector of coefficients $\gamma = (\gamma_1, \ldots, \gamma_J)$ amounts to a nonnegative lasso problem. More precisely, for each $j = 1, \ldots, J$, define the vector $\mathbf{g}_j = \mathbf{R}_j \boldsymbol{\theta}_j / \gamma_j \in \mathbb{R}^N$, where $\mathbf{f}_j = \mathbf{R}_j \boldsymbol{\theta}_j$ is the fitted vector for the j^{th} function using the current value of γ_j. Then we update the vector $\gamma = (\gamma_1, \ldots, \gamma_J)$ by solving

$$\min_{\gamma \geq 0} \left\{ \frac{1}{N} \|\mathbf{y} - \mathbf{G}\gamma\|_2^2 + \lambda \|\gamma\|_1 \right\}, \tag{4.48}$$

  where $\mathbf{G}$ is the $N \times J$ matrix with columns $\{\mathbf{g}_j, j = 1, \ldots, J\}$. These updates are a slightly different form than that given in Lin and Zhang (2003); full details are mapped out in Exercise 4.19.

When applied with the cubic smoothing-spline norm (4.43), the COSSO is aimed at setting component functions f_j to zero. There are many extensions to this basic idea. For instance, given a univariate function g, we might instead represent each univariate function in the form $g(t) = \alpha_0 + \alpha_1 t + h(t)$, and focus the penalty on departures from linearity using the norm

$$\|h\|_{\mathcal{H}}^2 := \int_0^1 h''(t)^2 dt, \tag{4.49}$$

In this setting, a variant of COSSO can select between nonlinear and linear forms for each component function.

We discuss penalties for additive models further in Section 4.4.4, in particular the benefits of using more than one penalty in this context.

4.4.4 Multiple Penalization for Sparse Additive Models

As we have seen thus far, there are multiple ways of enforcing sparsity for a nonparametric problem. Some methods, such as the SPAM back-fitting pro-

cedure, are based on a combination of the ℓ_1-norm with the empirical L^2-norm—namely, the quantity

$$\|f\|_{N,1} := \sum_{j=1}^{J} \|f_j\|_N, \tag{4.50}$$

where $\|f_j\|_N^2 := \frac{1}{N}\sum_{i=1}^{N} f_j^2(x_{ij})$ is the squared empirical L^2-norm for the univariate function f_j.[7] Other methods, such as the COSSO method, enforce sparsity using a combination of the ℓ_1-norm with the Hilbert norm

$$\|f\|_{\mathcal{H},1} := \sum_{j=1}^{J} \|f_j\|_{\mathcal{H}_j}. \tag{4.51}$$

Which of these two different regularizers is to be preferred for enforcing sparsity in the nonparametric setting?

Instead of focusing on only one regularizer, one might consider the more general family of estimators

$$\min_{\substack{f_j \in \mathcal{H}_j \\ j=1,\ldots,J}} \left\{ \frac{1}{N}\sum_{i=1}^{N}(y_i - \sum_{j=1}^{J} f_j(x_{ij}))^2 + \lambda_{\mathcal{H}} \sum_{j=1}^{J} \|f_j\|_{\mathcal{H}_j} + \lambda_N \sum_{j=1}^{J} \|f_j\|_N \right\}, \tag{4.52}$$

parametrized by the pair of nonnegative regularization weights $(\lambda_{\mathcal{H}}, \lambda_N)$. If we set $\lambda_N = 0$, then the optimization problem (4.52) reduces to the COSSO estimator, whereas for $\lambda_{\mathcal{H}} = 0$, we obtain a method closely related to the SPAM estimator. For any nonnegative $(\lambda_{\mathcal{H}}, \lambda_N)$, the optimization problem (4.52) is convex. When the underlying univariate Hilbert spaces $\mathcal{H}_j$ are described by a reproducing kernel, then the problem (4.52) can be re-formulated as a second-order cone program, and is closely related to the group lasso. Whenever the Hilbert space $\mathcal{H}_j$ is defined by a reproducing kernel $\mathcal{R}_j$, then the j^{th} coordinate function $\widehat{f}_j$ in any optimal solution again takes the form $\widehat{f}_j(\cdot) = \sum_{i=1}^{N} \widehat{\theta}_{ij}\mathcal{R}_j(\cdot, x_{ij})$ for a vector of weights $\widehat{\theta}_j \in \mathbb{R}^N$. This fact allows us to reduce the solution of the infinite-dimensional problem (4.52) to the simpler problem

$$\min_{\substack{\boldsymbol{\theta}_j \in \mathbb{R}^N \\ j=1,\ldots,J}} \left\{ \frac{1}{N}\|y - \sum_{j=1}^{J} \mathbf{R}_j\boldsymbol{\theta}_j\|_2^2 + \lambda_{\mathcal{H}} \sum_{j=1}^{J} \sqrt{\boldsymbol{\theta}_j^T \mathbf{R}_j \boldsymbol{\theta}_j} + \lambda_N \sum_{j=1}^{J} \sqrt{\boldsymbol{\theta}_j^T \mathbf{R}_j^2 \boldsymbol{\theta}_j} \right\}. \tag{4.53}$$

As before, for each coordinate $j \in \{1, \ldots, J\}$, the matrix $\mathbf{R}_j \in \mathbb{R}^{N \times N}$ is the kernel Gram matrix, with entries $[\mathbf{R}_j]_{ii'} = \mathcal{R}_j(x_{ij}, x_{i'j})$. See Exercise 4.20 for more details on this reduction.

[7] $\|f_j\|_{N,1}$ is the same as the $\|\mathbf{f}_j\|_2$ used in Section 4.4.2; here we are using a more generalizable notation.

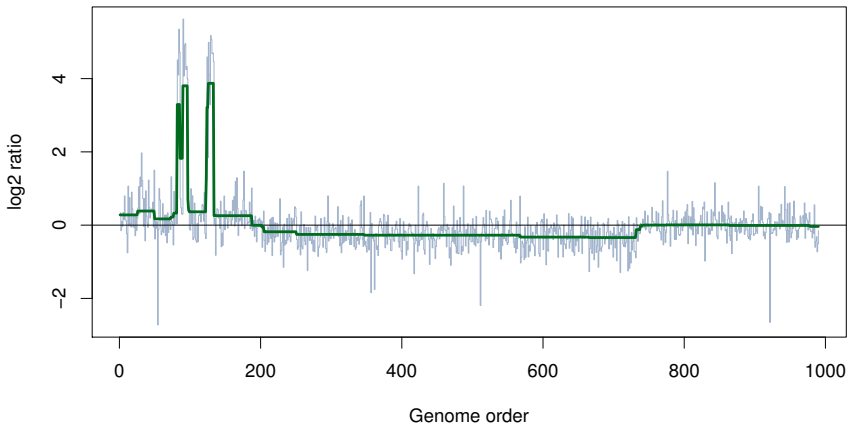


Figure 4.8 *Fused lasso applied to CGH data. Each spike represents the copy number of a gene in a tumor sample, relative to that of a control (on the log base-2 scale). The piecewise-constant green curve is the fused lasso estimate.*

The optimization problem (4.53) is an instance of a second-order cone program, and can be solved efficiently by a variant of the methods previously described. But why it is useful to impose two forms of regularization? As shown by Raskutti, Wainwright and Yu (2012), the combination of these two regularizers yields an estimator that is minimax-optimal, in that its convergence rate—as a function of sample size, problem dimension, and sparsity—is the fastest possible.

4.5 The Fused Lasso

Consider the gray spikes in Figure 4.8, the results of a *comparative genomic hybridization (CGH)* experiment. Each of these represents the (log base 2) relative copy number of a gene in a cancer sample relative to a control sample; these copy numbers are plotted against the chromosome order of the gene. These data are very noisy, so that some kind of smoothing is essential. Biological considerations dictate that it is typically segments of a chromosome—rather than individual genes—that are replicated, Consequently, we might expect that the underlying vector of true copy numbers to be piecewise-constant over contiguous regions of a chromosome. The *fused lasso signal approximator* exploits such structure within a signal, and is the solution of the following optimization problem

$$\underset{\boldsymbol{\theta}\in\mathbb{R}^N}{\text{minimize}} \left\{ \frac{1}{2}\sum_{i=1}^{N}(y_i-\theta_i)^2 + \lambda_1\sum_{i=1}^{N}|\theta_i| + \lambda_2\sum_{i=2}^{N}|\theta_i-\theta_{i-1}| \right\}. \tag{4.54}$$

The first penalty is the familiar ℓ_1-norm, and serves to shrink the θ_i toward zero. Since the observation index i orders the data (in this case along the chromosome), the second penalty encourages neighboring coefficients θ_i to be similar, and will cause some to be identical (also known as *total-variation denoising*). Notice that (4.54) does not include a constant term θ_0; the coefficient θ_i represents the response y_i directly, and for these kinds of problems zero is a natural origin. (See Exercise 4.21 for further exploration of this intercept issue.) The green curve in Figure 4.8 is fit to these data using the fused lasso.

There are more general forms of the fused lasso; we mention two here.

- We can generalize the notion of neighbors from a linear ordering to more general neighborhoods, for examples adjacent pixels in an image. This leads to a penalty of the form
$$\lambda_2 \sum_{i \sim i'} |\theta_i - \theta_{i'}|, \tag{4.55}$$
where we sum over all neighboring pairs $i \sim i'$.
- In (4.54) every observation is associated with a coefficient. More generally we can solve
$$\underset{(\beta_0,\beta)\in\mathbb{R}\times\mathbb{R}^p}{\text{minimize}} \left\{ \frac{1}{2}\sum_{i=1}^{N}(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda_1 \sum_{j=1}^{p} |\beta_j| + \lambda_2 \sum_{j=2}^{p} |\beta_j - \beta_{j-1}| \right\}. \tag{4.56}$$
Here the covariates x_{ij} and their coefficients β_j are indexed along some sequence j for which neighborhood clumping makes sense; (4.54) is clearly a special case.

4.5.1 Fitting the Fused Lasso

Problem (4.54) and its relatives are all convex optimization problems, and so all have well-defined solutions. As in other problems of this kind, here we seek efficient *path algorithms* for finding solutions for a range of values for the tuning parameters. Although coordinate descent is one of our favorite algorithms for lasso-like problems, it need not work for the fused lasso (4.54), because the difference penalty is not a separable function of the coordinates. Consequently, coordinate descent can become "stuck" at a nonoptimal point as illustrated in Figure 5.8 on page 111. This separability condition is discussed in more detail in Section 5.4.

We begin by considering the structure of the optimal solution $\widehat{\boldsymbol{\theta}}(\lambda_1, \lambda_2)$ of the fused lasso problem (4.54) as a function of the two regularization parameters λ_1 and λ_2. The following result due to Friedman et al. (2007) provides some useful insight into the behavior of this optimum:

Lemma 4.1. For any $\lambda_1' > \lambda_1$, we have

$$\widehat{\theta}_i(\lambda_1', \lambda_2) = \mathcal{S}_{\lambda_1' - \lambda_1}\big(\widehat{\theta}_i(\lambda_1, \lambda_2)\big) \text{ for each } i = 1, \ldots, N, \tag{4.57}$$

where $\mathcal{S}$ is the soft-thresholding operator $\mathcal{S}_\lambda\big(z\big) := \text{sign}(z)(|z| - \lambda)_+$.
One important special case of Lemma 4.1 is the equality

$$\widehat{\theta}_i(\lambda_1, \lambda_2) = \mathcal{S}_{\lambda_1}\big(\widehat{\theta}_i(0, \lambda_2)\big) \text{ for each } i = 1, \ldots, N. \tag{4.58}$$

Consequently, if we solve the fused lasso with $\lambda_1 = 0$, all other solutions can be obtained immediately by soft thresholding. This useful reduction also applies to the more general versions of the fused lasso (4.55). On the basis of Lemma 4.1, it suffices to focus our attention on solving the problem[8]

$$\underset{\boldsymbol{\theta} \in \mathbb{R}^N}{\text{minimize}} \left\{ \frac{1}{2} \sum_{i=1}^{N} (y_i - \theta_i)^2 + \lambda \sum_{i=2}^{N} |\theta_i - \theta_{i-1}| \right\}. \tag{4.59}$$

We consider several approaches to solving (4.59).

4.5.1.1 Reparametrization

One simple approach is to reparametrize problem (4.59) so that the penalty is additive. In detail, suppose that we consider a linear transformation of the form $\boldsymbol{\gamma} = \mathbf{M}\boldsymbol{\theta}$ for an invertible matrix $\mathbf{M} \in \mathbb{R}^{N \times N}$ such that

$$\gamma_1 = \theta_1, \text{ and } \gamma_i = \theta_i - \theta_{i-1} \text{ for } i = 2, \ldots, N. \tag{4.60}$$

In these transformed coordinates, the problem (4.59) is equivalent to the ordinary lasso problem

$$\underset{\boldsymbol{\gamma} \in \mathbb{R}^N}{\text{minimize}} \left\{ \frac{1}{2} \|\mathbf{y} - \mathbf{X}\boldsymbol{\gamma}\|^2 + \lambda \|\boldsymbol{\gamma}\|_1 \right\}, \text{ with } \mathbf{X} = \mathbf{M}^{-1}. \tag{4.61}$$

In principle, the reparametrize problem (4.61) can be solved using any efficient algorithm for the lasso, including coordinate descent, projected gradient descent or the LARS procedure. However, $\mathbf{X}$ is a lower-triangular matrix with all nonzero entries equal to 1, and hence has large correlations among the "variables." Neither coordinate-descent nor LARS performs well under these circumstances (see Exercise 4.22). So despite the fact that reparametrization appears to solve the problem, it is not recommended, and more efficient algorithms exist, as we now discuss.

[8] Here we have adopted the notation λ (as opposed to λ_2) for the regularization parameter, since we now have only one penalty.

4.5.1.2 A Path Algorithm

The one-dimensional fused lasso (4.59) has an interesting property, namely that as the regularization parameter λ increases, pieces of the optimal solution can only be joined together, not split apart. More precisely, letting $\widehat{\boldsymbol{\theta}}(\lambda)$ denote the optimal solution to the convex program (4.59) as a function of λ, we have:

Lemma 4.2. Monotone fusion. Suppose that for some value of λ and some index $i \in \{1, \ldots, N-1\}$, the optimal solution satisfies $\widehat{\theta}_i(\lambda) = \widehat{\theta}_{i+1}(\lambda)$. Then for all $\lambda' > \lambda$, we also have $\widehat{\theta}_i(\lambda') = \widehat{\theta}_{i+1}(\lambda')$.

Friedman et al. (2007) observed that this fact greatly simplifies the construction of the fused lasso solution path. One starts with $\lambda = 0$, for which there are no fused groups, and then computes the smallest value of λ that causes a fused group to form. The parameter estimates for this group are then fused together (i.e., constrained to be equal) for the remainder of the path. Along the way, a simple formula is available for the estimate within each fused group, so that the resulting procedure is quite fast, requiring $\mathcal{O}(N)$ operations. However, we note that the monotone-fusion property in Lemma 4.2 is special to the one-dimensional fused lasso (4.59). For example, it does not hold for the general fused lasso (4.56) with a model matrix $\mathbf{X}$, nor for the two-dimensional fused lasso (4.55). See Friedman et al. (2007) and Hoefling (2010) for more details on this approach.

4.5.1.3 A Dual Path Algorithm

Tibshirani$_2$ and Taylor (2011) take a different approach, and develop path algorithms for the *convex duals* of fused lasso problems. Here we illustrate their approach on the problem (4.59), but note that their methodology applies to the general problem (4.56) as well.

We begin by observing that problem (4.59) can be written in an equivalent *lifted* form

$$\underset{(\boldsymbol{\theta},\mathbf{z})\in\mathbb{R}^N\times\mathbb{R}^{N-1}}{\text{minimize}} \left\{ \frac{1}{2}\|\mathbf{y}-\boldsymbol{\theta}\|_2^2 + \lambda\|\mathbf{z}\|_2 \right\} \text{ subject to } \mathbf{D}\boldsymbol{\theta} = \mathbf{z}, \tag{4.62}$$

where we have introduced a vector $\mathbf{z} \in \mathbb{R}^{N-1}$ of auxiliary variables, and $\mathbf{D}$ is a $(N-1) \times N$ matrix of first differences. Now consider the Lagrangian associated with the lifted problem, namely

$$L(\boldsymbol{\theta}, \mathbf{z}; \mathbf{u}) := \frac{1}{2}\|\mathbf{y}-\boldsymbol{\theta}\|_2^2 + \lambda\|\mathbf{z}\| + \mathbf{u}^T(\mathbf{D}\boldsymbol{\theta} - \mathbf{z}), \tag{4.63}$$

where $\mathbf{u} \in \mathbb{R}^{N-1}$ is a vector of Lagrange multipliers. A straightforward computation shows that the Lagrangian dual function $\mathcal{Q}$ takes form

$$\mathcal{Q}(\mathbf{u}) := \inf_{(\boldsymbol{\theta},\mathbf{z})\in\mathbb{R}^N\times\mathbb{R}^{N-1}} L(\boldsymbol{\theta}, \mathbf{z}; \mathbf{u}) = \begin{cases} -\frac{1}{2}\|\mathbf{y} - \mathbf{D}^T\mathbf{u}\|^2 & \text{if } \|\mathbf{u}\|_\infty \le \lambda, \\ -\infty & \text{otherwise.} \end{cases} \tag{4.64}$$

The Lagrangian dual problem is to maximize $\mathcal{Q}(\mathbf{u})$, and given an optimal solution $\widehat{\mathbf{u}} = \widehat{\mathbf{u}}(\lambda)$, we can recover an optimal solution $\widehat{\boldsymbol{\theta}} = \widehat{\boldsymbol{\theta}}(\lambda)$ to the original problem by setting $\widehat{\boldsymbol{\theta}} = \mathbf{y} - \mathbf{D}^T\widehat{\mathbf{u}}$. See Exercise 4.23 for the details of these duality calculations.

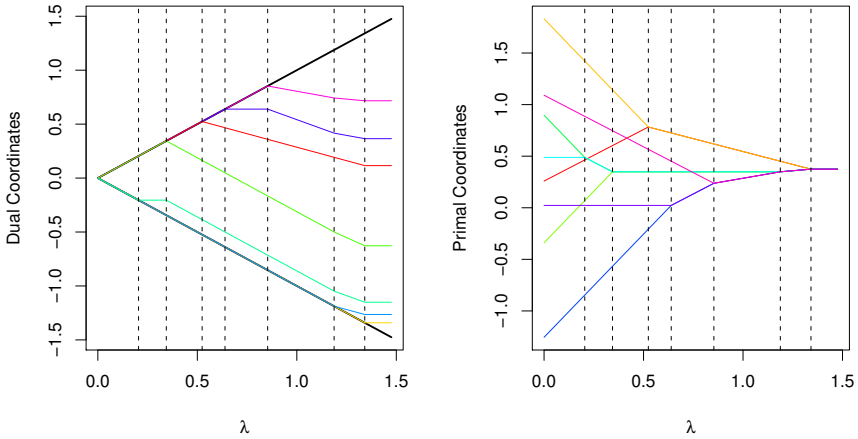


Figure 4.9 *The dual path algorithm in action on a small example. The left panel shows the progress of* $\widehat{\mathbf{u}}(\lambda)$*, while the right panel shows* $\widehat{\boldsymbol{\theta}}(\lambda)$*. We see that in the dual coordinates, as a parameter hits the boundary, an* unfusing *occurs in the primal coordinates.*

When the regularization parameter λ is sufficiently large, the dual maximization, or equivalently the problem of minimizing $-\mathcal{Q}(\mathbf{u})$, reduces to an unrestricted linear regression problem, with optimal solution

$$\mathbf{u}^* := (\mathbf{D}\mathbf{D}^T)^{-1}\mathbf{D}\mathbf{y}. \tag{4.65}$$

The restrictions kick in when λ decreases to the critical level $\|\mathbf{u}^*\|_\infty$. Tibshirani$_2$ and Taylor (2011) show that as we decrease λ, once elements $\widehat{u}_j(\lambda)$ of the optimal solution hit the bound λ, then they are guaranteed to never leave the bound. This property leads to a very straightforward path algorithm, similar in spirit to LARS in Section 5.6; see Figure 4.9 for an illustration of the dual path algorithm in action. Exercise 4.23 explores some of the details.

4.5.1.4 Dynamic Programming for the Fused Lasso

Dynamic programming is a computational method for solving difficult problems by breaking them down into simpler subproblems. In the case of the one-dimensional fused lasso, the linear ordering of the variables means that

fixing any variable breaks down the problem into two separate subproblems to the left and right of the fixed variable. In the "forward pass," we move from left to right, fixing one variable and solving for the variable to its left, as a function of this fixed variable. When we reach the right end, a backward pass then gives the complete solution.

Johnson (2013) proposed this dynamic programming approach to the fused lasso. In more detail, we begin by separating off terms in (4.59) that depend on θ_1, and rewrite the objective function (4.59) in the form

$$f(\boldsymbol{\theta}) = \underbrace{\frac{1}{2}(y_1-\theta_1)^2+\lambda|\theta_2-\theta_1|}_{g(\theta_1,\theta_2)} + \left\{\frac{1}{2}\sum_{i=2}^{N}(y_i-\theta_i)^2+\lambda\sum_{i=3}^{N}|\theta_i-\theta_{i-1}|\right\}. \tag{4.66}$$

This decomposition shows the subproblem to be solved in the first step of the forward pass: we compute $\widehat{\theta}_1(\theta_2) := \arg\min_{\theta_1\in\mathbb{R}} g(\theta_1,\theta_2)$, We have thus eliminated the first variable, and can now focus on the reduced objective function $f_2 : \mathbb{R}^{N-1} \to \mathbb{R}$ given by

$$f_2(\theta_2,\ldots\theta_N) = f(\widehat{\theta}_1(\theta_2),\theta_2,\ldots\theta_N). \tag{4.67}$$

We can then iterate the procedure, maximizing over θ_2 to obtain $\widehat{\theta}_2(\theta_3)$, and so on until we obtain $\widehat{\theta}_N$. Then we back-substitute to obtain $\widehat{\theta}_{N-1} = \widehat{\theta}_{N-1}(\widehat{\theta}_N)$, and so on for the sequences $\widehat{\theta}_{N-2},\ldots,\widehat{\theta}_2,\widehat{\theta}_1$.

If each parameter θ_i can take only one of K distinct values, then each of the minimizers $\widehat{\theta}_j(\theta_{j+1})$ can be easily computed and stored as a $K\times K$ matrix. In the continuous case, the functions to be minimized are piecewise linear and quadratic, and care must be taken to compute and store the relevant information in an efficient manner; see Johnson (2013) for the details. The resulting algorithm is the fastest that we are aware of, requiring just $\mathcal{O}(N)$ operations, and considerably faster than the path algorithm described above. Interestingly, if we change the ℓ_1 difference penalty to an ℓ_0, this approach can still be applied, despite the fact that the problem is no longer convex. Exercise 4.24 asks the reader to implement the discrete case.

4.5.2 Trend Filtering

The first-order absolute difference penalty in the fused lasso can be generalized to use a higher-order difference, leading to the problem

$$\underset{\boldsymbol{\theta}\in\mathbb{R}^N}{\text{minimize}}\left\{\frac{1}{2}\sum_{i=1}^{N}(y_i-\theta_i)^2+\lambda\cdot\|\mathbf{D}^{(k+1)}\boldsymbol{\theta}\|_1\right\}. \tag{4.68}$$

This is known as *trend filtering*. Here $\mathbf{D}^{(k+1)}$ is a matrix of dimension $(N-k-1)\times N$ that computes discrete differences of order $k+1$. The fused

lasso uses first-order differences ($k = 0$), while higher-order differences encourage higher-order smoothness. In general, trend filtering of order k results in solutions that are piecewise polynomials of degree k. Linear trend filtering ($k = 1$) is especially attractive, leading to piecewise-linear solutions. The

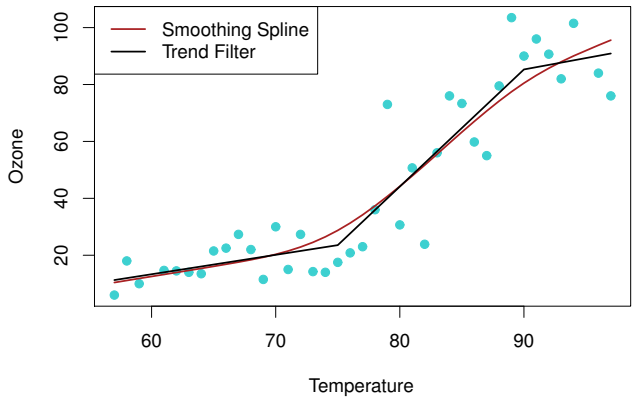


Figure 4.10 *A piecewise-linear function fit to some air-pollution data using trend filtering. For comparison, a smoothing spline with the same degrees of freedom is included.*

knots in the solution need not be specified but fall out of the convex optimization procedure. Kim, Koh, Boyd and Gorinevsky (2009) propose an efficient interior point algorithm for this problem. Tibshirani$_2$ (2014) proves that the trend filtering estimate adapts to the local level of smoothness much better than smoothing splines, and displays a surprising similarity to locally-adaptive regression splines. Further, he shows that the estimate converges to the true underlying function at the minimax rate for functions whose k^{th} derivative is of bounded variation (a property not shared by linear estimators such as smoothing splines). Furthermore, Tibshirani$_2$ and Taylor (2011) show that a solution with m knots has estimated degrees of freedom given by $\mathrm{df} = m + k + 1$.[9]

Figure 4.10 shows a piecewise-linear function fit by trend filtering to some air-pollution data. As a comparison, we include the fit of a smoothing spline, with the same effective df = 4. While the fits are similar, it appears that trend filtering has found natural changepoints in the data.

In (4.68) it is assumed that the observations occur at evenly spaced positions. The penalty can be modified (Tibshirani$_2$ 2014) to accommodate arbitrary (ordered) positions x_i as follows:

$$\underset{\boldsymbol{\theta}\in\mathbb{R}^N}{\text{minimize}} \left\{ \frac{1}{2}\sum_{i=1}^{N}(y_i-\theta_i)^2 + \lambda \cdot \sum_{i=1}^{N-2} \left| \frac{\theta_{i+2}-\theta_{i+1}}{x_{i+2}-x_{i+1}} - \frac{\theta_{i+1}-\theta_i}{x_{i+1}-x_i} \right| \right\} \tag{4.69}$$

[9] This is an unbiased estimate of the degrees of freedom; see Section 2.5.

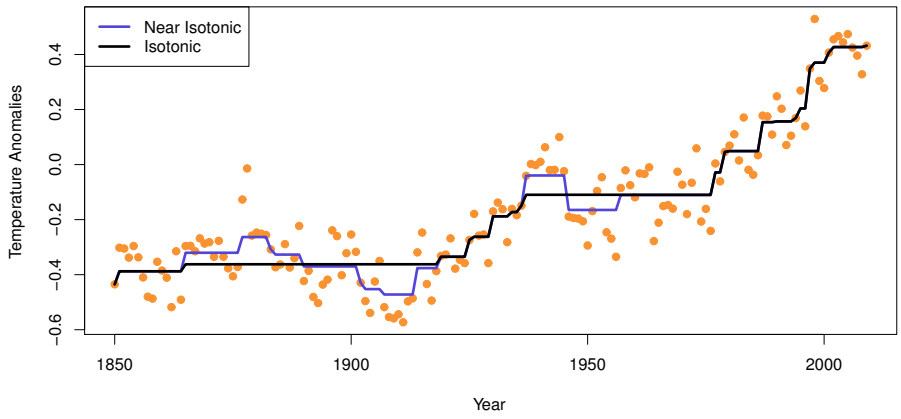


Figure 4.11 *Near isotonic fit to global-warming data, showing annual temperature anomalies. The value of λ was chosen by cross-validation, and the fit appears to support the evidence of nonmonotone behavior seen in the data.*

It compares the empirical slopes for adjacent pairs, and encourages them to be the same. This is the penalty that was used in Figure 4.10, since the `Temperature` values are not uniformly spaced.

4.5.3 Nearly Isotonic Regression

Tibshirani$_2$, Hoefling and Tibshirani (2011) suggest a simple modification of the one-dimensional fused lasso that encourages the solution to be monotone. It is based on a relaxation of isotonic regression. In the classical form of isotonic regression, we estimate $\boldsymbol{\theta} \in \mathbb{R}^N$ by solving the constrained minimization problem

$$\underset{\boldsymbol{\theta}\in\mathbb{R}^N}{\text{minimize}} \left\{ \sum_{i=1}^{N} (y_i - \theta_i)^2 \right\} \text{ subject to } \theta_1 \le \theta_2 \le \ldots \le \theta_N. \tag{4.70}$$

The resulting solution gives the best monotone (nondecreasing) fit to the data. Monotone nonincreasing solutions can be obtaining by first flipping the signs of the data. There is a unique solution to problem (4.70), and it can be obtained using the *pool adjacent violators algorithm* (Barlow, Bartholomew, Bremner and Brunk 1972), or PAVA for short.

Nearly isotonic regression is a natural relaxation, in which we introduce a regularization parameter $\lambda \ge 0$, and instead solve the penalized problem

$$\underset{\boldsymbol{\theta}\in\mathbb{R}^N}{\text{minimize}} \left\{ \frac{1}{2} \sum_{i=1}^{N} (y_i - \theta_i)^2 + \lambda \sum_{i=1}^{N-1} (\theta_i - \theta_{i+1})_+ \right\}. \tag{4.71}$$

The penalty term penalizes adjacent pairs that violate the monotonicity property, that is, having $\theta_i > \theta_{i+1}$. When $\lambda = 0$, the solution interpolates the data,

and letting $\lambda \to \infty$, we recover the solution to the classical isotonic regression problem (4.70). Intermediate value of λ yield nonmonotone solutions that trade off monotonicity with goodness-of-fit; this trade-off allows one to assess the validity of the monotonicity assumption for the given data sequence. Figure 4.11 illustrates the method on data on annual temperature anomalies from 1856 to 1999, relative to the 1961–1990 mean. The solution to the nearly isotonic problem (4.71) can be obtained from a simple modification of the path algorithm discussed previously, a procedure that is analogous to the PAVA algorithm for problem (4.70); see Tibshirani$_2$ et al. (2011) for details.

4.6 Nonconvex Penalties

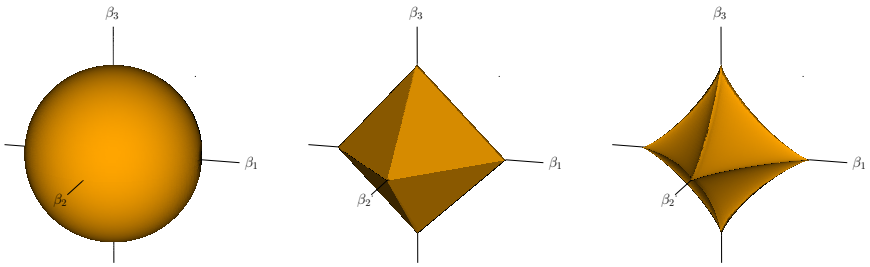


Figure 4.12 *The ℓ_q unit balls in $\mathbb{R}^3$ for $q = 2$ (left), $q = 1$ (middle), and $q = 0.8$ (right). For $q < 1$ the constraint regions are nonconvex. Smaller q will correspond to fewer nonzero coefficients, and less shrinkage. The nonconvexity leads to combinatorially hard optimization problems.*

By moving from an ℓ_2 penalty to ℓ_1, we have seen that for the same *effective df* the lasso selects a subset of variables to have nonzero coefficients, and shrinks their coefficients less. When p is large and the number of relevant variables is small, this may not be enough; in order to reduce the set of chosen variables sufficiently, lasso may end up over-shrinking the retained variables. For this reason there has been interest in nonconvex penalties.

The natural choice might be the ℓ_q penalty, for $0 \leq q \leq 1$, with the limiting ℓ_0 corresponding to best-subset selection. Figure 4.12 compares the ℓ_q unit balls for $q \in \{2, 1, 0.8\}$. The spiky nonconvex nature of the ball on the right implies edges and coordinate axes will be favored in selection under such constraints. Unfortunately, along with nonconvexity comes combinatorial computational complexity; even the simplest case of ℓ_0 can be solved exactly only for $p \approx 40$ or less. For this and related statistical reasons alternative nonconvex penalties have been proposed. These include the SCAD (Fan and Li 2001, *smoothly clipped absolute deviation*) and MC+ (Zhang 2010, *minimax concave*) penalties. Figure 4.13 shows four members of the MC+ penalty family in $\mathbb{R}^1$, indexed by the nonconvexity parameter $\gamma \in (1, \infty)$; this family

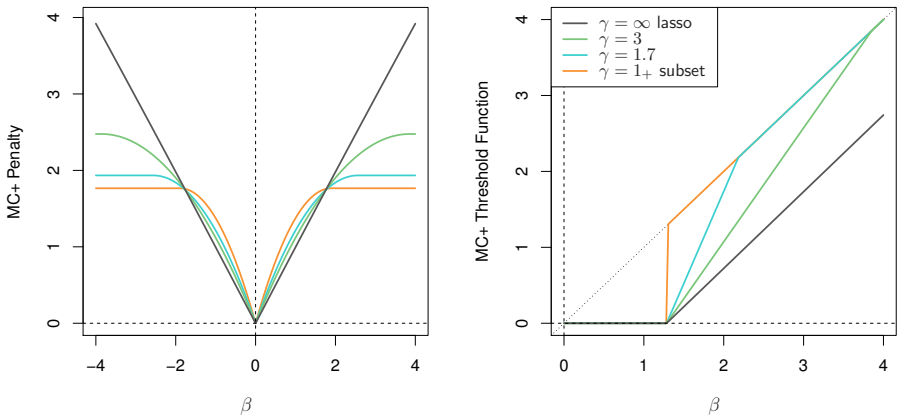


Figure 4.13 *Left: The MC+ family of nonconvex sparsity penalties, indexed by a sparsity parameter $\gamma \in (1, \infty)$. Right: piecewise-linear and continuous threshold functions associated with MC+ (only the north-east quadrant is shown), making this penalty family suitable for coordinate descent algorithms.*

bridges the gap between lasso ($\gamma = \infty$) and best-subset ($\gamma = 1_+$). The penalty functions are piecewise quadratic (see Exercise 4.25), and importantly the corresponding threshold functions are piecewise linear and continuous. In detail, for squared-error loss we pose the (nonconvex) optimization problem

$$\underset{\beta \in \mathbb{R}^p}{\text{minimize}} \left\{ \frac{1}{2}\|\mathbf{y} - \mathbf{X}\beta\|_2^2 + \sum_{j=1}^{p} P_{\lambda,\gamma}(\beta_j) \right\}, \tag{4.72}$$

with the MC+ penalty on each coordinate defined by

$$P_{\lambda,\gamma}(\theta) := \int_0^{|\theta|} \left(1 - \frac{x}{\lambda\gamma}\right)_+ dx. \tag{4.73}$$

With coordinate descent in mind, we consider solving a one-dimensional version of (4.72) (in standardized form)

$$\underset{\beta \in \mathbb{R}^1}{\text{minimize}} \left\{ \frac{1}{2}(\beta - \tilde{\beta})^2 + \lambda \int_0^{|\beta|} \left(1 - \frac{x}{\lambda\gamma}\right)_+ dx \right\}. \tag{4.74}$$

The solution is unique[10] for $\gamma > 1$ and is given by

$$\mathcal{S}_{\lambda,\gamma}(\tilde{\beta}) = \begin{cases} 0 & \text{if } |\tilde{\beta}| \le \lambda \\ \text{sign}(\tilde{\beta})\left(\frac{|\tilde{\beta}| - \lambda}{1 - \frac{1}{\gamma}}\right) & \text{if } \lambda < |\tilde{\beta}| \le \lambda\gamma \\ \tilde{\beta} & \text{if } |\tilde{\beta}| > \lambda\gamma \end{cases} \tag{4.75}$$

[10]Despite the nonconvexity, there is a unique solution in $\mathbb{R}^1$; this is not necessarily the case for the p-dimensional problem (4.72).

The right panel in Figure 4.13 shows examples of (4.75). Large values of $\tilde{\beta}$ are left alone, small values are set to zero, and intermediate values are shrunk. As γ gets smaller, the intermediate zone gets narrower, until eventually it becomes the hard-thresholding function of best subset (orange curve in figure). By contrast, the threshold functions for the ℓ_q family ($q < 1$) are discontinuous in $\tilde{\beta}$.

Mazumder, Friedman and Hastie (2011) exploit the continuity of $\mathcal{S}_{\lambda,\gamma}$ (in both λ and γ) in a coordinate-descent scheme for fitting solution paths for the entire MC+ family. Starting with the lasso solution, their `R` package `sparsenet` (Mazumder, Hastie and Friedman 2012) moves down a sequence in γ toward sparser models, and for each fits a regularization path in λ. Although it cannot claim to solve the nonconvex problem (4.72), this approach is both very fast and appears to find good solutions.

Zou (2006) proposed the *adaptive lasso* as a means for fitting models sparser than lasso. Using a pilot estimate $\tilde{\beta}$, the adaptive lasso solves

$$\underset{\beta\in\mathbb{R}^p}{\text{minimize}} \left\{ \frac{1}{2}\|\mathbf{y} - \mathbf{X}\beta\|_2^2 + \lambda \sum_{j=1}^{p} w_j|\beta_j| \right\}, \tag{4.76}$$

where $w_j = 1/|\tilde{\beta}_j|^\nu$. The adaptive lasso penalty can be seen as an approximation to the ℓ_q penalties with $q = 1 - \nu$. One advantage of the adaptive lasso is that given the pilot estimates, the criterion (4.76) is convex in β. Furthermore, if the pilot estimates are $\sqrt{N}$ consistent, Zou (2006) showed that the method recovers the true model under more general conditions than does the lasso. If $p < N$ one can use the least-squares solutions as the pilot estimates. When $p \geq N$, the least-squares estimates are not defined, but the univariate regression coefficients can be used for the pilot estimates and result in good recovery properties under certain conditions (Huang, Ma and Zhang 2008). Exercise 4.26 explores the close connections between the adaptive lasso and the nonnegative garrote of Section 2.8.

We end this section by mentioning a practical alternative to nonconvex optimization for sparse model-path building. Forward-stepwise methods (Hastie et al. 2009, Chapter 3) are very efficient, and are hard to beat in terms of finding good, sparse subsets of variables. Forward stepwise is a greedy algorithm—at each step fixing the identity of the terms already in the model, and finding the best variable to include among those remaining. The theoretical properties of forward-stepwise model paths are less well understood, partly because of the algorithmic definition of the procedure, as opposed to being a solution to an optimization problem.

Bibliographic Notes

The elastic net was proposed by Zou and Hastie (2005), who also distinguished between the naive version, similar to the one presented here, and a debiased

version that attempts to undo the biasing effect of the ridge shrinkage. Friedman et al. (2015) build a system of coordinate-descent algorithms for fitting elastic-net penalized generalized linear models, implemented in the `R` package `glmnet`. Yuan and Lin (2006) introduced the group lasso, and their paper has stimulated much research. Meier et al. (2008) extended the group lasso to logistic regression problems, whereas Zhao, Rocha and Yu (2009) describe a more general family of structured penalties, including the group lasso as a special case. A line of theoretical work has sought to understand when the group lasso estimator has lower statistical error than the ordinary lasso. Huang and Zhang (2010) and Lounici, Pontil, Tsybakov and van de Geer (2009) establish error bounds for the group lasso, which show how it outperforms the ordinary lasso in certain settings. Negahban, Ravikumar, Wainwright and Yu (2012) provide a general framework for analysis of M-estimators, including the group lasso as a particular case as well as more general structured penalties. Obozinski, Wainwright and Jordan (2011) characterize multivariate regression problems for which the group lasso does (or does not) yield better variable selection performance than the ordinary lasso.

The overlap group lasso was introduced by Jacob et al. (2009), and the sparse group lasso by Puig, Wiesel and Hero (2009) and Simon, Friedman, Hastie and Tibshirani (2013). Various algorithms have been developed for solving the group and overlap group lassos, as well as a variety of structured generalizations; see Bach, Jenatton, Mairal and Obozinski (2012) for a good review.

Additive models were proposed by Stone (1985) as a means of side-stepping the curse of dimensionality in nonparametric regression; see Hastie and Tibshirani (1990) for further background on (generalized) additive models. The COSSO model was developed by Lin and Zhang (2003) in the context of reproducing kernel Hilbert spaces, and ANOVA spline decompositions. The books by Wahba (1990) and Gu (2002) provide further background on splines and RKHSs. Ravikumar et al. (2009) followed up with the SPAM model, which is somewhat simpler and more general, and established certain forms of high-dimensional consistency for their estimator. Meier, van de Geer and Bühlmann (2009) studied a related family of estimators, based on explicit penalization with the empirical L^2-norm, corresponding to the doubly penalized estimator with $\lambda_{\mathcal{H}} = 0$. Koltchinski and Yuan (2008, 2010) analyzed the COSSO estimator, as well as the doubly penalized estimator (4.52). Raskutti et al. (2009, 2012) derived minimax bounds for sparse additive models, and also show that the doubly penalized estimator (4.52) can achieve these bounds for various RKHS families, including splines as a special case.

The fused lasso was introduced by Tibshirani, Saunders, Rosset, Zhu and Knight (2005). Various algorithms have been proposed for versions of the fused lasso, including the methods of Hoefling (2010), Johnson (2013), and Tibshirani$_2$ and Taylor (2011).

The MC+ threshold function was first described in Gao and Bruce (1997) in the context of wavelet shrinkage. There has been a lot of activity in non-

convex penalties for sparse modeling. Zou and Li (2008) develop local linear approximation algorithms for tackling the nonconvex optimization problems. These and other approaches are discussed in Mazumder et al. (2011).

Exercises

Ex. 4.1 Suppose we have two identical variables $X_1 = X_2$, and a response Y, and we perform a ridge regression (see (2.7) in Section 2.2) with penalty $\lambda > 0$. Characterize the coefficient estimates $\widehat{\beta}_j(\lambda)$.

Ex. 4.2 Consider a slightly noisy version of the identical twins example in the beginning of Section 4.2, where the two variables are strongly positively correlated. Draw a schematic of the contours of the loss function and the penalty function, and demonstrate why the elastic net encourages coefficient sharing more than does the lasso.

Ex. 4.3 Consider the elastic-net problem (4.2).

(a) Show how to simplify the calculation of $\widehat{\beta}_0$ by centering each of the predictors, leading to $\widehat{\beta}_0 = \bar{y}$ (for all values of λ). How does one convert back to the estimate of $\widehat{\beta}_0$ for uncentered predictors?

(b) Verify the soft-thesholding expression (4.4) for the update of $\widehat{\beta}_j$ by coordinate descent.

Ex. 4.4 Consider the solution to the group lasso problem (4.5) when some of the variables are factors. Show that when there is an intercept in the model, the optimal coefficients for each factor sum to zero.

Ex. 4.5 This exercise investigates the penalty modifier for the group lasso. Consider the *entry* criterion $\|\mathbf{Z}_j^T \boldsymbol{r}_j\|_2 < \lambda$ for the group lasso (Section 4.3.1). Suppose $\boldsymbol{r}_j$ is i.i.d noise with mean $\mathbf{0}$ and covariance $\sigma^2\mathbf{I}$—a null situation. Show that

$$\mathbb{E}\|\mathbf{Z}_j^T \boldsymbol{r}_j\|_2^2 = \sigma^2 \|\mathbf{Z}\|_F^2. \tag{4.77}$$

Hence argue that to make comparisons fair among the penalty terms in the group lasso, one should replace $\lambda \sum_{j=1}^{J} \|\theta_j\|_2$ in Equation (4.5) with

$$\lambda \sum_{j=1}^{J} \tau_j \|\theta_j\|_2, \tag{4.78}$$

where $\tau_j = \|\mathbf{Z}_j\|_F$. Show that when $\mathbf{Z}_j$ is orthonormal, this results in $\tau_j = \sqrt{p_j}$.

Ex. 4.6 Show that under the orthonormality condition $\mathbf{Z}_j^T \mathbf{Z}_j = \mathbf{I}$, the update (4.15) solves the fixed point Equation (4.13).

Ex. 4.7 Consider the block-wise solution vector (4.14) for the group lasso. If $\|\hat{\theta}_j\|$ is known, we can write the solution in closed form. Let $\mathbf{Z}_j = \mathbf{UDV}^T$ be the singular value decomposition of $\mathbf{Z}_j$. Let $r^* = \mathbf{U}^T \boldsymbol{r}_j \in \mathbb{R}^{p_j}$. Show that $\phi = \|\hat{\theta}_j\|$ solves

$$\sum_{\ell=1}^{p_j} \frac{r_\ell^{*2} d_\ell^2}{(d_\ell^2 \phi + \lambda)^2} = 1, \tag{4.79}$$

where d_ℓ is the ℓ^{th} diagonal element of $\mathbf{D}$. Show how to use a golden search strategy to solve for ϕ. Write an R function to implement this algorithm, along with the golden search.

Ex. 4.8 Discuss the impact of the normalization $\mathbf{Z}_j^T \mathbf{Z}_j = \mathbf{I}$ in the context of a matrix of dummy variables representing a factor with p_j levels. Does the use of contrasts rather than dummy variables alleviate the situation?

Ex. 4.9 Using the approach outlined in Section 5.3.3, derive the generalized gradient update (4.16a) for the group lasso. Write a R function to implement this algorithm (for a single group). Include an option to implement Nesterov acceleration.

Ex. 4.10 Using the approach outlined in Section 5.3.3, derive the generalized gradient update (4.23) for the sparse group lasso.

Ex. 4.11 Run a series of examples of increasing dimension to compare the performance of your algorithms in Exercises 4.7 and 4.9. Make sure they are producing the same solutions. Compare their computational speed—for instance, the command `system.time()` can be used in R.

Ex. 4.12 Consider the condition (4.19) for $\hat{\theta}_j$ to be zero for the sparse group lasso. Define

$$\begin{aligned} J(t) &= \frac{1}{\lambda(1-\alpha)} \|\mathbf{Z}_j^T \boldsymbol{r}_j - \lambda\alpha \cdot t\|_2 \\ &= \|s\|_2. \end{aligned} \tag{4.80}$$

Now solve

$$\min_{t: t_k \in [-1,1]} J(t), \tag{4.81}$$

and show that this leads to the condition $\hat{\theta}_j = 0$ if and only if $\|\hat{g}_j\|_2 \leq \lambda(1-\alpha)$ with $\hat{g}_j = \mathcal{S}_{\lambda\alpha}(\mathbf{Z}_j^T \boldsymbol{r}_j)$.

Ex. 4.13 Show that if $\mathbf{Z}_j^T \mathbf{Z}_j = \mathbf{I}$, then (4.21) solves (4.12).