Lecture Notes I
(still incompete, with "???" symbols)

Peter Bühlmann & Sara van de Geer

Contents

Chapter 1

Introduction

In many applications, datasets arise where the number of covariates is very large, e.g. in the thousands or ten-thousands, while the sample size is quite small, e.g. in the dozens or hundreds. More formally, such high-dimensional data is of the form

$$(X_1, Y_1), \ldots, (X_n, Y_n) \tag{1.1}$$

with p-dimensional covariates $X_i \in \mathcal{X} \subset \mathbb{R}^p$ and response variables $Y_i \in \mathcal{Y} \subset \mathbb{R}^q$. We say that a problem is high-dimensional if $p \gg n$, usually irrespective of the magnitude of the dimension q of the response. Quite often, the response is univariate with $q = 1$: the most prominent examples are regression where $\mathcal{Y} = \mathbb{R}$, classification with $\mathcal{Y}$ a categorical space (e.g. labeled as $\mathcal{Y} = \{0, 1, \ldots J-1\}$) but our book will also cover more general areas or models such as survival analysis with survival times $Y \in \mathcal{Y} = \mathbb{R}^+$ or Poisson regression with $\mathcal{Y} = \{0, 1, 2, \ldots\} = \mathbb{N}_0$. Typically, we assume that the pairs (X_i, Y_i) in (1.1) are independent, identically distributed (i.i.d.) but some generalization to stationary processes, covering areas of time series analysis or spatial statistics, is fairly straightforward.

Every reasonable model for data as in (1.1) involves at least one parameter per covariate: a linear model involves one parameter while say an additive model (in the covariates) would involve a few or many parameters in each basis expansion per covariate. Therefore, with high-dimensional data, we will be confronted with the situation where the number of parameters will be much larger than sample size. In general, it will be hopeless to estimate all unknown parameters from data unless the true underlying parameter vector is *sparse*. In case where the true parameter vector is sparse, e.g. many parameters are equal to zero or some norm of the parameter vector is small, it is possible to infer the true parameter vector and its structure, e.g. the zeroes of the parameter vector, from data. Our book will focus on methods and theory for estimating a sparse, high-dimensional parameter vector in various models. Furthermore, we will demonstrate some applications of these methods for problems and data-sets arising from molecular biology.

Chapter 2

Lasso for Linear Models

2.1 Introduction

The Lasso, proposed by Tibshirani [1996], is an acronym for Least Absolute Shrinkage and Selection Operator. Among the main reasons why it has become very popular for high-dimensional estimation problems are its statistical accuracy for prediction and variable selection coupled with its computational feasibility. Furthermore, since the Lasso is a penalized likelihood approach, the method is rather general and can be used in a broad variety of models. In the simple case of a linear model with orthonormal design, the Lasso equals the soft thresholding estimator (see e.g. Donoho and Johnstone [1994] and Donoho [1995]).

We consider here the setting where the observed data are realizations of

$$(X_1, Y_1), \ldots, (X_n, Y_n)$$

with p-dimensional covariates $X_i \in \mathcal{X} \subset \mathbb{R}^p$ and univariate response variables $Y_i \in \mathcal{Y} \subset \mathbb{R}$. The covariates are either deterministic fixed values or random variables: regarding the methodology, there is no difference between these two cases. Typically, we assume that the samples are independent but the generalization to stationary processes poses no essential methodological albeit some mathematical problems arise for proving properties.

Modeling high-dimensional data is challenging. For a continuous response variable $Y \in \mathbb{R}$, a simple yet very useful approach is given by a linear model

$$Y_i = \sum_{j=1}^{p} \beta_j X_i^{(j)} + \varepsilon_i \ (i = 1, \ldots, n), \tag{2.1}$$

where $\varepsilon_1, \ldots, \varepsilon_n$ i.i.d., independent of $\{X_i;\ i = 1, \ldots, n\}$ and with $\mathbb{E}[\varepsilon_i] = 0$. We often use the matrix- and vector-notation

$$\mathbf{Y} = \mathbf{X}\beta + \varepsilon$$

with response vector $\mathbf{Y}_{n\times 1}$, design matrix $\mathbf{X}_{n\times p}$, parameter vector $\beta_{p\times 1}$ and error vector $\varepsilon_{n\times 1}$.

For simplicity and without loss of generality, we assume that the intercept is zero and that all covariates are centered and measured on the same scale. Both of these assumptions

can be approximately achieved by mean centering and scaling with the standard deviation and thus, for practical purposes, we consider models of the form

$$Y_i - \overline{Y} = \sum_{j=1}^{p} \beta_j (X_i^{(j)} - \overline{X}^{(j)}) + \varepsilon_i,$$

with $\hat{\sigma}_j^2 = n^{-1} \sum_{i=1}^{n} (X_i^{(j)} - \overline{X}^{(j)})^2 = 1$ for all j. The only unusual aspect of the linear model in (2.1) is the fact that $p \gg n$.

The ordinary least squares estimator is not unique and will heavily overfit the data. Thus, a form of complexity regularization will be necessary. We focus here on regularization with the ℓ_1-penalty. The estimation of the parameters in model (2.1) can be done with the Lasso [Tibshirani, 1996]:

$$\hat{\beta}(\lambda) = \arg\min_{\beta} \Big(\|\mathbf{Y} - \mathbf{X}\beta\|_2^2/n + \lambda \|\beta\|_1 \Big), \tag{2.2}$$

where $\|\mathbf{Y} - \mathbf{X}\beta\|_2^2 = \sum_{i=1}^{n} (Y_i - (\mathbf{X}\beta)_i)^2$, $\|\beta\|_1 = \sum_{j=1}^{p} |\beta_j|$ and where $\lambda \geq 0$ is a penalty parameter. The estimator has the property that it does variable selection in the sense that $\hat{\beta}_j(\lambda) = 0$ for some j's (depending on the choice of λ) and $\hat{\beta}_j(\lambda)$ can be thought as a shrunken least squares estimator; hence, the name Least Absolute Shrinkage and Selection Operator. An intuitive explanation for the variable selection property is given below.

The optimization in (2.2) is convex, enabling efficient computation of the estimator, see Section 2.9. In addition, it is sometimes useful to know that the optimization problem in (2.2) is equivalent to

$$\hat{\beta}_{\text{primal}}(s) = \arg\min_{\beta; \|\beta\|_1 \leq s} \|\mathbf{Y} - \mathbf{X}\beta\|_2^2/n, \tag{2.3}$$

with a one-to-one correspondence between λ in (2.2) and s in (2.3) which depends on the data $(X_1, Y_1), \ldots, (X_n, Y_n)$. Such an equivalence holds since $\|\mathbf{Y} - \mathbf{X}\beta\|_2^2/n$ is convex in β with convex constraint $\|\beta\|_1 \leq s$. More details are described in e.g. Bertsekas [1995, Ch. 5.3].

Because of the ℓ_1-geometry, the Lasso is performing variable selection in the sense that an estimated component can be exactly zero. To see this, we consider the representation in (2.3) and Figure 2.1: the residual sum of squares reaches a minimal value (for certain constellations of the data) if its contour lines hit the ℓ_1-ball in its corner which corresponds to $\hat{\beta}_{\text{primal},1} = 0$. Figure 2.1 indicates that such a phenomenon does not occur with say Ridge regression,

$$\hat{\beta}_{\text{Ridge}}(\lambda) = \arg\min_{\beta} \Big(\|\mathbf{Y} - \mathbf{X}\beta\|_2^2/n + \lambda \|\beta\|_2^2 \Big),$$

with its equivalent primal equivalent solution

$$\hat{\beta}_{\text{Ridge;primal}}(s) = \arg\min_{\beta; \|\beta\|_2 \leq s} \|\mathbf{Y} - \mathbf{X}\beta\|_2^2/n, \tag{2.4}$$

with a one-to-one correspondence between λ and s which depends on the data.

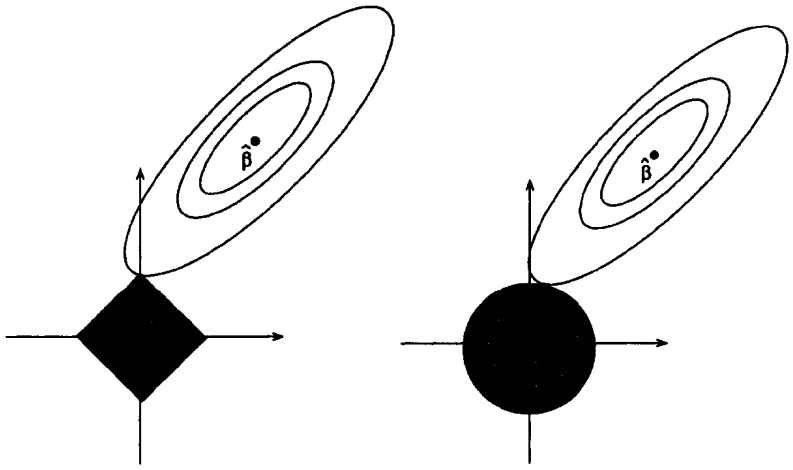

Figure 2.1: Left: Contour lines of residual sum of squares and ℓ_1-ball corresponding to the Lasso problem in (2.3). Right: Analogous to left panel but with ℓ_2-ball corresponding to Ridge regression in (2.4).

2.2 Orthonormal design

It is instructive to consider the orthonormal design where $p = n$ and the design matrix satisfies $n^{-1}\mathbf{X}^T\mathbf{X} = I_{p\times p}$. For this case, the Lasso estimator is the soft-threshold estimator

$$\hat{\beta}_j(\lambda) = \text{sign}(Z_j)(|Z_j| - \lambda/2)_+, \ Z_j = (\mathbf{X}^T\mathbf{Y})_j \ \ (j = 1, \ldots, p = n), \tag{2.5}$$

where $(x)_+ = \max(x, 0)$ denotes the positive part. This follows from the general characterization in Lemma 2.4.1 below and we leave the exact derivation as Exercise ???. This estimator can be written as

$$\hat{\beta}_j(\lambda) = g_{\text{soft}}(Z_j),$$

where $g_{\text{soft}}(\cdot)$ is the soft-threshold function depicted in Figure 2.2.

2.3 Prediction

We refer to prediction whenever the goal is the estimation of the regression function $\mathbb{E}[Y|X = x] = \sum_{j=1}^p \beta_j x^{(j)}$ in model (2.1). This is also the relevant quantity for predicting a new observation.

2.3.1 Practical aspects about the Lasso for prediction

From a practical perspective, prediction with the Lasso is straightforward and easy. Typically, we use some cross-validation (CV) scheme, e.g., 10-fold CV, to select a reasonable tuning parameter λ minimizing the cross-validated squared error risk. In addition, we can validate the accuracy of the performance by using again some cross-validation scheme.

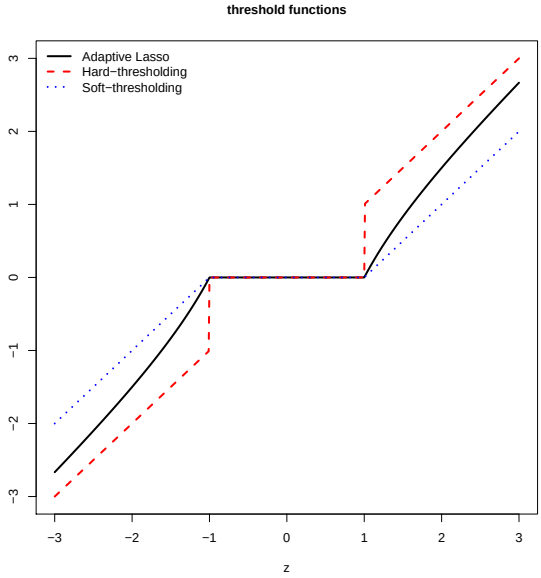

Figure 2.2: Various threshold functions $g(\cdot)$ for orthonormal design: soft-threshold (dashed line), hard-threshold (dotted line), Adaptive Lasso (solid line). The estimators are of the form $\hat{\beta}_j = g(Z_j)$ with Z_j as in (2.5).

Regarding the latter, we should cross-validate the whole procedure which includes the selection of the tuning parameter λ. In particular, by comparing the cross-validated risk, we can see whether the Lasso yields a performance which is better, equal or worse than another prediction algorithm. However, it is not straightforward to test rigorously whether the performances of two prediction algorithms are significantly different or not, see for example van de Wiel et al. [2009].

Binary classification of lymph node status using gene expressions

We consider a binary classification problem involving a binary response variable $Y \in \{0, 1\}$, describing the lymph node status of a cancer patient, and we have a covariate with $p = 7129$ gene expression measurements. There are $n = 49$ breast cancer tumor samples. The data is taken from West et al. [2001]. It is known that this is a difficult, high noise classification problem. The best methods achieve about a cross-validated misclassification error of about 20%.

Despite that this is a binary classification problem, we can use the Lasso as in (2.2) which yields an estimate of the conditional class probability $p(x) = \mathbb{P}[Y = 1|X = x] = \mathbb{E}[Y|X = x]$:

$$\hat{p}_\lambda(x) = x\hat{\beta}(\lambda).$$

Of course, we could use the Lasso also for logistic regression as described later in Chapter 3. In either case, we classify as follows:

$$\hat{\mathcal{C}}_\lambda(x) = \begin{cases} 1 & \hat{p}_\lambda(x) > 1/2 \\ 0 & \hat{p}_\lambda(x) \leq 1/2 \end{cases}$$

For comparison, we consider a forward variable selection method in penalized linear logistic regression with ℓ_2-norm (Ridge-type) penalty. The optimal regularization parameter, for Lasso and forward penalized logistic regression, is chosen by 10-fold cross-validation. For evaluating the performance of the tuned algorithms, we use a cross-validation scheme for estimating the test-set misclassification error. We randomly divide the sample into 2/3 training- and 1/3 test-data and we repeat this 100 times: the average test-set misclassification error is reported in Table 2.1. Note that we run a double cross-validation: one inner level for choosing the regularization parameter and one outer level for assessing the performance of the algorithm. Table 2.1 illustrates that the forward selection approach yields

Lasso	forw. penalized logist. regr.
21.1%	35.25%

Table 2.1: Misclassification test set error using cross-validation

much poorer performance than the Lasso. Forward selection methods tend to be unstable [Breiman, 1996]: they are of a very greedy nature striving for maximal improvement of the objective function (e.g. residual sum of squares) in every step.

Finally, we report that the Lasso selected on cross-validation average 13.12 out of $p = 7129$ variables (genes). Thus, the fitted linear model is very sparse with respect to the ℓ_0-norm, i.e. the selected number of variables is very small.

2.3.2 Some results from asymptotic theory

We describe here some results which are developed and described in detail in Chapter **??**. Greenshtein and Ritov [2004] have proved in the high-dimensional setting that the Lasso consistently estimates the regression function as sample size $n \to \infty$. To capture high-dimensional scenarios, the asymptotics is with respect to a triangular array of observations:

$$Y_{n;i} = \sum_{j=1}^{p_n} \beta_{n;j} X_{n;i}^{(j)} + \varepsilon_{n;i},\ i = 1, \ldots, n;\quad n = 1, 2, \ldots \tag{2.6}$$

Thereby, we allow that $p_n \gg n$. The assumptions about $\varepsilon_{n;i}$ are as in the linear model in (2.1). A consistency result requires some sparsity assumption of the form

$$\|\beta\|_1 = O\Big(\sqrt{\frac{n}{\log(p)}}\Big).$$

Assuming further mild regularity conditions, the following holds: for a suitable range of $\lambda = \lambda_n$ the Lasso is consistent for estimating the underlying regression function:

$$(\hat{\beta}(\lambda) - \beta_0)^T \Sigma_X (\hat{\beta}(\lambda) - \beta_0) = o_P(1)\ (n \to \infty), \tag{2.7}$$

where Σ_X is $n^{-1}\mathbf{X}^T\mathbf{X}$ in case of a fixed design or equals the covariance of the covariate X in case of a random design. Note that the left hand side in (2.7) can be written as the average squared error loss:

$$\begin{aligned} &\|\mathbf{X}(\hat{\beta} - \beta^0)\|_2^2/n \text{ for fixed design,} \\ &\mathbb{E}[(X_{new}(\hat{\beta}(\lambda) - \beta^0))^2] \text{ for random design,} \end{aligned}$$

where $\mathbb{E}$ is with respect to the new test observation X_{new} ($1 \times p$ vector) and $X(\hat{\beta}(\lambda) - \beta^0)$ is the difference between the estimated and true regression function $\hat{f}(X) - f^0(X)$. The asymptotics is according to the triangular array in (2.6). More details are presented in Chapter ??? in Corollary **??**.

In fact, under certain compatibility conditions on the design $\mathbf{X}$, one can show a so-called oracle inequality

$$\mathbb{E}[\|\mathbf{X}(\hat{\beta}(\lambda) - \beta^0)\|_2/n] = O\Big(\frac{s_0 \log(p)}{n}\Big), \tag{2.8}$$

where $s_0 = \text{card}(S_0) = \text{card}(\{j;\ \beta_j^0 \neq 0\})$, see Corollary **??**. This means that, up to the $\log(p)$-term, the mean-squared prediction error is of the order as if one would knew a-priori which of the s_0 covariates are relevant and using ordinary least squares estimation based on the true, relevant s_0 variables only. This rate is optimal, up to the factor $\log(p)$, in scenarios where the regression coefficients of the relevant s_0 variables are not very small.

2.4 Variable screening and $\|\hat{\beta} - \beta\|_q$-norms

Instead of prediction $X_{new}\hat{\beta}$, we consider the estimation accuracy in terms of inference for the parameter β. Under some assumptions on the design $\mathbf{X}$ in a linear model, it can be shown that for some suitable range of λ,

$$\|\hat{\beta}(\lambda) - \beta^0\|_q \ \to 0 \text{ in probability } (n \to \infty), \tag{2.9}$$

where $q \in \{1, 2\}$ and $\|\beta\|_q = (\sum_j |\beta_j|^q)^{1/q}$. The asymptotic framework is again with respect to the triangular array described in (2.6) where $\beta^0 = \beta_n^0$ is allowed to depend on n. The derivation of such results is given in Chapter **??**, Section **??**.

The result in (2.9) has fairly direct and interesting implications in terms of variable screening. Consider the active set of variables

$$S_0 = \{j;\ 1 \leq j \leq p, \beta_j^0 \neq 0\}$$

which contains all covariates with non-zero corresponding regression coefficients. Note that in a setting as in (2.6), the active set $S_0 = S_{0;n}$ depends on n. Since the Lasso estimator in (2.2) is selecting some variables, in the sense that some of the coefficients are exactly zero ($\hat{\beta}_j(\lambda) = 0$ for some j's, depending on λ), we use it as screening set:

$$\widehat{S}(\lambda) = \{j;\ 1 \leq j \leq p, \hat{\beta}_j(\lambda) \neq 0\}. \tag{2.10}$$

It is worth pointing out that no significance testing is involved. We now argue that the variables with corresponding non-zero coefficients remain the same across different solutions $\hat{\beta}(\lambda)$ of the optimization in (2.2), see Lemma 2.4.1. Note that different solutions occur if the optimization is not strictly convex as in the case where $p > n$.

An important characterization of the solution $\hat{\beta}(\lambda)$ in (2.2) can be derived from the Karush-Kuhn-Tucker conditions (and some additional reasoning regarding uniqueness of zeroes).

Lemma 2.4.1 *Denote the gradient of $n^{-1}\|\mathbf{Y} - bx\beta\|^2$ by $G(\beta) = -2\mathbf{X}^T(\mathbf{Y} - \mathbf{X}\beta)/n$. Then: a necessary and sufficient condition for $\hat{\beta}$ to be a solution of (2.2) is:*

$$G_j(\hat{\beta}) = -\,sign(\hat{\beta}_j)\lambda \text{ if } \hat{\beta}_j \neq 0,$$
$$|G_j(\hat{\beta})| \leq \lambda \text{ if } \hat{\beta}_j = 0.$$

Moreover, if the solution of (2.2) is not unique (e.g. if $p > n$) and $G_j(\hat{\beta}) < \lambda$ for some solution $\hat{\beta}(\lambda)$, then $\hat{\beta}_j(\lambda) = 0$ for all solutions of (2.2).

Proof: For the first statements regarding a necessary and sufficient characterization of the solution, we invoke subdifferential calculus (BERTSEKAS ????). Denote the criterion function by

$$Q_\lambda(\beta) = \|\mathbf{Y} - \mathbf{X}\beta\|_2^2/n + \lambda\|\beta\|_1.$$

For a minimizer $\hat{\beta}(\lambda)$ of $Q_\lambda(\cdot)$ it is then necessary and sufficient that the subdifferential at $\hat{\beta}(\lambda)$ is zero. If the jth component $\hat{\beta}_j(\lambda) \neq 0$, this means that the ordinary first derivative at $\hat{\beta}(\lambda)$ has to be zero:

$$\frac{\partial Q_\lambda(\beta)}{\partial \beta_j} = -2\mathbf{X}_j^T(\mathbf{Y} - \mathbf{X}\beta) + \lambda \text{sign}(\beta_j)|_{\beta=\hat{\beta}(\lambda)} = 0.$$

Of course, this is equivalent to

$$G_j(\hat{\beta}(\lambda) = -2\mathbf{X}_j^T(\mathbf{Y} - \mathbf{X}\beta(\lambda) = -\lambda \text{sign}(\hat{\beta}_j(\lambda)) \text{ if } \hat{\beta}_j(\lambda) \neq 0.$$

On the other hand, if $\hat{\beta}_j(\lambda) = 0$, the subdifferential at $\hat{\beta}(\lambda)$ has to include the zero element (see BERTSEKAS ???). That is:

$$G_j(\hat{\beta}(\lambda)) + \lambda e = 0 \text{ for some } e \in [-1, 1], \text{ and if } \hat{\beta}_j(\lambda) = 0.$$

But this is equivalent to

$$|G_j(\hat{\beta}(\lambda))| \leq \lambda \text{ if } \hat{\beta}_j(\lambda) = 0.$$

And this is the second statement about the characterization of the solution of $\hat{\beta}(\lambda)$.

Regarding uniqueness of the zeroes among different solutions. Assume that there exist two solutions $\hat{\beta}^{(1)}$ and $\hat{\beta}^{(2)}$ such that for a component j we have $\hat{\beta}_j^{(1)} = 0$ with $G_j(\hat{\beta}^{(1)})| < \lambda$ but $\hat{\beta}_j^{(2)} \neq 0$. Because the set of all solutions is convex,

$$\hat{\beta}_\rho = (1 - \rho)\hat{\beta}^{(1)} + \rho\hat{\beta}^{(2)}$$

is also a minimizer for all $\rho \in [0, 1]$. By assumption $\hat{\beta}_{\rho,j} \neq 0$ and hence, by the first statement from the KKT conditions, $|G_j(\hat{\beta}_\rho)| = \lambda$ for all $\rho \in (0, 1)$. Hence, it holds for $g(\rho) = |G_j(\hat{\beta})_\rho)|$ that $g(0) < \lambda$ and $g(\rho) = \lambda$ for all $\rho \in (0, 1)$. But this is a contradiction to the fact that $g(\cdot)$ is continuous. Hence, a non-active (i.e. zero) component j with $|G_j(\hat{\beta}| < \lambda$ can not be active (i.e. non-zero) in any other solution. □

Ideally, we would like to infer the active set S_0 from data. We will see in Section 2.5 that the Lasso as used in (2.10) requires fairly strong conditions on the design matrix $\mathbf{X}$. A less ambitious but still relevant goal in practice is to find at least the covariates whose corresponding absolute values of the regressions coefficients $|\beta_j|$ are substantial (and other variables may be included as well). More formally, for some $C > 0$, define the substantial covariates as

$$S_{subst(C)} = \{j;\ 1 \leq j \leq p, |\beta_j| \geq C\}.$$

Using the result in (2.9), which holds under weaker assumptions than the restrictive neighborhood stability or irrepresentable condition:

$$\text{for any fixed } 0 < C < \infty : \ \mathbb{P}[\widehat{S}(\lambda) \supset S_{subst(C)}] \to 1 \ (n \to \infty). \tag{2.11}$$

We leave the proof of this fact as Exercise 5???. This result can be generalized as follows. Assume that

$$\|\hat{\beta}_n(\lambda_n) - \beta^0\|_1 \leq a_n \text{ with high probability.} \tag{2.12}$$

We note that typically, $a_n = const.s_0\sqrt{\log(p_n)/n}$ with $s_0 = |S_0|$. Then,

$$\text{for } C_n > a_n : \ \text{with high probability } \widehat{\mathcal{S}}_n(\lambda_n) \supset S_{subst(C_n)}. \tag{2.13}$$

The proof is elementary and we leave it as Exercise 6???. It may happen that $S_{subst(C_n)} = S_0$ and then, $\hat{S}(\lambda_n) \supset S_0$ with high probability.

We refer to the property in (2.11) or in (2.13) as *variable screening*: with high probability, the Lasso estimated model includes the substantial covariates. Variable screening with the Lasso has a great potential because of the following fact. Every Lasso estimated model has cardinality smaller or equal to $\min(n, p)$: this follows from the analysis of the LARS algorithm [Efron et al., 2004]. If $p \gg n$, $\min(n, p) = n$ is a small number and hence, we achieve a huge dimensionality reduction in terms of the original covariates. For example, in the lymph node status classification problem in Section 2.3.1, we reduce from $p = 7129$ to at most $n = 49$ covariates.

Tuning parameter selection for variable screening

Consider the prediction optimal parameter supplied by an oracle,

$$\lambda^* = \lambda_n^* = \ \text{argmin}_\lambda \mathbb{E}[(Y_{new} - \sum_{j=1}^{p} \hat{\beta}_j(\lambda) X_{new}^{(j)})^2], \tag{2.14}$$

where (X_{new}, Y_{new}) is an independent copy of (X_i, Y_i) $(i = 1, \ldots, n)$. Then, at least for some examples,

$$\mathbb{P}[\widehat{S}(\lambda_n^*) \supset S] \ \to 1 \ (n \to \infty), \tag{2.15}$$

see Meinshausen and Bühlmann [2006, Prop. 1]. The over-estimation behavior in (2.15) is quite typical for many finite-sample cases, that is, the prediction optimal estimated Lasso model is too large, containing with high probability the true model.

This ties in nicely with the screening property in (2.11) or (2.13). We summarize that the Lasso screening procedure is very useful and easy to implement: we choose the regularization parameter by cross-validation with respect to the prediction squared error loss, denoted by $\hat{\lambda}_{CV}$, and the Lasso screening procedure then yields $\hat{S}(\hat{\lambda}_{CV})$ which is expected to contain S_0 and whose cardinality is bounded by $|\hat{S}(\hat{\lambda}_{CV})| \leq \min(n, p)$.

As an alternative, we may pursue a Lasso screening procedure by including $\min(n, p)$ variables (e.g. using the LARS algorithm until the end [Efron et al., 2004]) and hence, no tuning parameter needs to be chosen. If $p \gg n$, this tuning-free dimensionality reduction can be very worthwhile for a first stage.

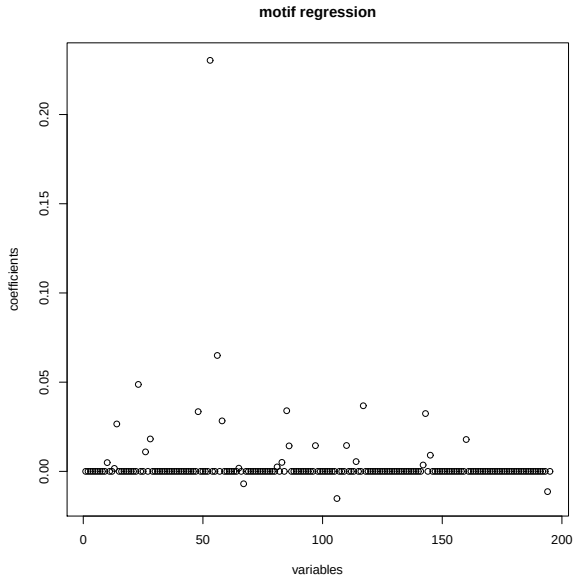

Figure 2.3: HIF1α motif regression with $n = 287,\ p = 195$.

2.5 Variable selection

The problem of variable selection for a high-dimensional linear model in (2.1) is important since in many areas of applications, the primary interest is about the relevance of covariates. As there are 2^p possible sub-models, computational feasibility is an important concern. The usual variable selection procedure is based on least squares and a penalty which involves the number of parameters in the candidate sub-model:

$$\hat{\beta}(\lambda) = \text{argmin}_\beta \Big\{ \frac{1}{n} \sum_{i=1}^{n} (Y_i - \sum_{j=1}^{p} \beta_j X_i^{(j)})^2 + \lambda \|\beta\|_0 \Big\}, \tag{2.16}$$

where the ℓ_0-norm penalty is $\|\beta\|_0 = \sum_{j=1}^{p} I(\beta_j \neq 0)$. Many well known model selection criteria such as the Akaike Information Criterion (AIC), the Bayesian Information Criterion (BIC) or the Minimum Description Length (MDL) fall into this framework. For example, when the error variance is known, AIC and BIC correspond to $\lambda = 4\sigma^2$ and $\lambda = \log(n)2\sigma^2$, respectively. The estimator in (2.16) is infeasible to compute when p is of medium or large size since the ℓ_0-norm penalty is a nonconvex function in β. Computational infeasibility remains even when using branch-and-bound techniques, cf. Hofmann et al. [2007], Gatu et al. [2007]. Forward selection strategies are computationally fast but they can be very instable [Breiman, 1996], as illustrated in Table 2.1 where forward selection produced a poor result. Other ad-hoc methods may be used to get approximations for the ℓ_0-norm penalized least squares estimator in (2.16). On the other hand, the requirement of computational feasibility and statistical accuracy can be met by the Lasso defined in (2.2): it can also be viewed as a convex relaxation of the optimization problem with the ℓ_0-norm in (2.16).

We will first build up the methodology and theory by using the Lasso in a single stage. We will describe later in Section 2.6 how to use the Lasso not just once but in two (or

more) stages. Consider the set of estimated variables using the Lasso as in (2.10):

$$\hat{S}(\lambda) = \{j;\ 1 \le j \le p, \hat{\beta}_j(\lambda) \neq 0\}.$$

In particular, we can compute all possible Lasso sub-models

$$\widehat{\mathcal{SUB}} = \{\widehat{S}(\lambda);\ \text{all } \lambda\} \tag{2.17}$$

with $O(np\min(n,p))$ operation counts, see Section 2.9. As pointed out above in Section 2.4, every sub-model in $\widehat{\mathcal{SUB}}$ has cardinality smaller or equal to $\min(n,p)$. Furthermore, the number of sub-models in $\widehat{\mathcal{SUB}}$ is typically of the order $O(\min(n,p))$ [Rosset and Zhu, 2007]. Thus, in summary, each Lasso estimated sub-model contains at most $\min(n,p)$ variables which is a small number if $p \gg n$, and the number of different Lasso estimated sub-models is $O(\min(n,p))$ which represents a huge reduction compared to all possible 2^p sub-models if $p \gg n$.

The question of interest is whether the true set of effective variables $S_0 = \{j;\ 1 \le j \le p, \beta_j^0 \neq 0\}$ is contained in $\widehat{\mathcal{SUB}}$ and if yes, which particular choice of λ will identify the true underlying set of active variables S_0.

An asymptotic result described below shows that with probability tending to 1, $S_0 \subseteq \widehat{\mathcal{SUB}}$ and that the Lasso is appropriate for addressing the problem of variable selection. As in Section 2.3, to capture high-dimensionality of the model (2.1) in an asymptotic sense, we consider the triangular array scheme in (2.6). The main and restrictive assumption for consistent variable selection concerns the (fixed or random) design matrix X. The condition, called neighborhood stability or irrepresentable condition, is described with some more rigour in Section 2.5.1. Under such a neighborhood stability condition, Meinshausen and Bühlmann [2006, Theorems 1 and 2] show the following: for a suitable $\lambda = \lambda_n$,

$$\mathbb{P}[\widehat{S}(\lambda) = S_0] \to 1\ (n \to \infty). \tag{2.18}$$

This can be seen as an asymptotic justification to replace (or relax) the computationally hard ℓ_0-penalty problem in (2.16) by the ℓ_1-penalty problem of the Lasso in (2.2).

It is worth mentioning here, that the neighborhood stability condition on the design is sufficient and necessary and hence, we have a sharp result saying when the Lasso is consistent for variable selection and when not. It should represent a warning that the restrictive assumptions on the design have some relevant implications on the statistical practice for high-dimensional model selection: with strongly correlated design, the Lasso can perform very poorly for variable selection. A further difficulty comes with the choice of the regularization parameter. An extension of formula (2.15) is: for some examples, it holds

$$\begin{aligned} &\mathbb{P}[\hat{S}(\lambda_n^*) \supset S] \ \to 1\ (n \to \infty), \\ &\limsup_{n\to\infty} \mathbb{P}[\hat{S}(\lambda_n^*) = \mathcal{S}] < 1\ (n \to \infty), \end{aligned} \tag{2.19}$$

where λ^* is the prediction optimal (theoretical) tuning parameter in (2.14). More detailed mathematical formulations and statements are provided in Chapter **??**. Furthermore, we will describe in Chapter **??** the relation between Gaussian Graphical Modeling and variable selection in a linear model.

2.5.1 Neighborhood stability and irrepresentable condition

There is certainly an interesting potential to use the Lasso for variable selection in high-dimensional models, as described in (2.18). However, the so-called neighborhood stability condition is crucial for consistent variable selection with the Lasso. In fact, this neighborhood stability condition is sufficient and essentially necessary for consistent model selection with the Lasso in the sense of (2.18), see Theorems 1, 2 and Proposition 3 in Meinshausen and Bühlmann [2006]. The word "essentially" refers to the fact that the necessary condition requires a quantity to be ≤ 1 while the sufficient condition requires strict < 1.

The neighborhood stability condition is equivalent to the so-called irrepresentable condition (at least for the case where $p < n$ is fixed) which has been introduced by Zou [2006] and Zhao and Yu [2006] and which is easier to describe. We assume that the design matrix $\mathbf{X}$ satisfies

$$n^{-1}\mathbf{X}^T\mathbf{X} \to \Sigma,$$

where Σ is a positive definite matrix. For example, for a random and centered design Σ equals the covariance matrix of the p-dimensional covariate X. Without loss of generality, we assume that the active set $S_0 = \{j;\ \beta_j \neq 0\} = \{1, \ldots, s_0\}$ consists of the first s_0 variables. Let

$$\Sigma = \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix},$$

where Σ_{11} is a $s_0 \times s_0$ matrix corresponding to the active variables, $\Sigma_{12} = \Sigma_{21}^T$ is a $s_0 \times (p - s_0)$ matrix and Σ_{22} a $(p - s_0) \times (p - s_0)$ matrix. The irrepresentable condition then reads:

$$|\Sigma_{21}\Sigma_{11}^{-1}\text{sign}(\beta_1, \ldots, \beta_{s_0})| < 1 \tag{2.20}$$

where the inequality is to be understood componentwise, for all $(p - s_0)$ components, and $\text{sign}(\beta_1, \ldots, \beta_p) = (\text{sign}(\beta_1), \ldots, \text{sign}(\beta_p))^T$. As with the neighborhood stability condition, the irrepresentable condition in (2.20) is sufficient and essentially necessary for consistent model selection with the Lasso ("essentially" refers to the fact that the necessary condition requires the relation "≤ 1", while the sufficient condition requires strict < 1). For the high-dimensional setting and in terms of the triangular array as in (2.6), it is understood that the right-hand side of (2.20) is bounded away from 1 for all $n \in \mathbb{N}$.

Roughly speaking, the neighborhood stability or irrepresentable condition fails to hold if the design matrix $\mathbf{X}$ is too much "ill-posed" and exhibits a too strong degree of linear dependence within "smaller" sub-matrices of $\mathbf{X}$. Examples where the irrepresentable condition holds include the following, cf. Zhao and Yu [2006]:

2.6 The adaptive Lasso: a two-stage procedure

An interesting approach to correct Lasso's overestimation behavior, see formulae (2.15), (2.11) and (2.13), is given by the adaptive Lasso (Zou, 2006) which replaces the ℓ_1-norm

penalty by a re-weighted version. For a linear model as in (2.1), it is defined as a two-stage procedure:

$$\hat{\beta}_{adapt} = \text{argmin}_\beta \|\mathbf{Y} - \mathbf{X}\beta\|_2^2/n + \lambda \sum_{j=1}^{p} \frac{|\beta_j|}{|\hat{\beta}_{init,j}|}), \tag{2.21}$$

where $\hat{\beta}_{init}$ is an initial estimator.

In the high-dimensional context, we propose to use the Lasso from a first stage as the initial estimator, tuned in a prediction optimal way. Typically, we use cross-validation to select the tuning parameter, denoted here by $\lambda_{init,CV}$. Thus, the initial estimator is $\hat{\beta}_{init} = \hat{\beta}(\lambda_{init,CV})$ as in (2.2). For the second stage, we use again cross-validation to select the parameter λ in the adaptive Lasso (2.21). Proceeding this way, we select the regularization parameters in a sequential way: this is computationally much cheaper since we optimize twice over a single parameter instead of simultaneous optimization over two tuning parameters. The procedure is also described in Section 2.6.4 when using $k = 2$.

The adaptive Lasso has the following obvious property:

$$\hat{\beta}_{init,j} = 0 \;\Rightarrow\; \hat{\beta}_{adapt,j} = 0. \tag{2.22}$$

Furthermore, if $|\hat{\beta}_{init,j}|$ is large, the adaptive Lasso employs a small penalty (i.e. little shrinkage) for the jth coefficient β_j which implies less bias. Thus, the adaptive Lasso yields a sparse solution and it can be used to reduce the number of false positives (selected variables which are not relevant) from the first stage. This is a desirable property since the Lasso from the first stage has the screening property that $\hat{S} \supseteq S_0$ with high probability. Further details about variable selection with the adaptive Lasso are described below in Section 2.6.2 and Section **??**. In the latter, we treat the case where the penalty is of the form $\lambda \sum_{j=1}^{p} w_j|\beta_j|$ with $0 \le w_j < \infty$. That is, the weight $w_j = \infty$, arising in (2.21) with $\hat{\beta}_{init,j} = 0$, is not allowed. This can be seen as a less stringent way of preserving a zero estimate from the initial estimate.

An illustration

We illustrate the Lasso and adaptive Lasso on some simulated example from a linear model as in (2.1) with $p = 1000$ and $n = 50$. We choose $\beta_1 = 2,\ \beta_2 = 1,\ \beta_3 = 0.5$ and $\beta_4 = \ldots \beta_{1000} = 0$, $\varepsilon \sim \mathcal{N}(0,1)$ and $X^{(1)}, \ldots, X^{(1000)}$ i.i.d. $\sim \mathcal{N}(0,1)$. This amount to a "medium-size" signal to noise ratio

$$\frac{Var(f(X))}{\sigma^2} = 5.5,$$

where $f(x) = x\beta$. Figure 2.4 shows the coefficient estimates for the Lasso and the adaptive Lasso, with initial estimator from the Lasso, respectively. The tuning parameters are selected as follows. For the Lasso, we use the optimal λ from 10-fold cross-validation. This Lasso fit is used as initial estimator and we then choose λ for the second stage in adaptive Lasso by optimizing 10-fold cross-validation again. We exploit the fact that Lasso is a powerful screening method: all three relevant variables are selected, i.e., $\hat{S} \supseteq S_0$, but it also selects 41 noise covariates. The adaptive Lasso yields a substantially sparser fit: it selects all of the 3 relevant variables and 10 noise covariates in addition.

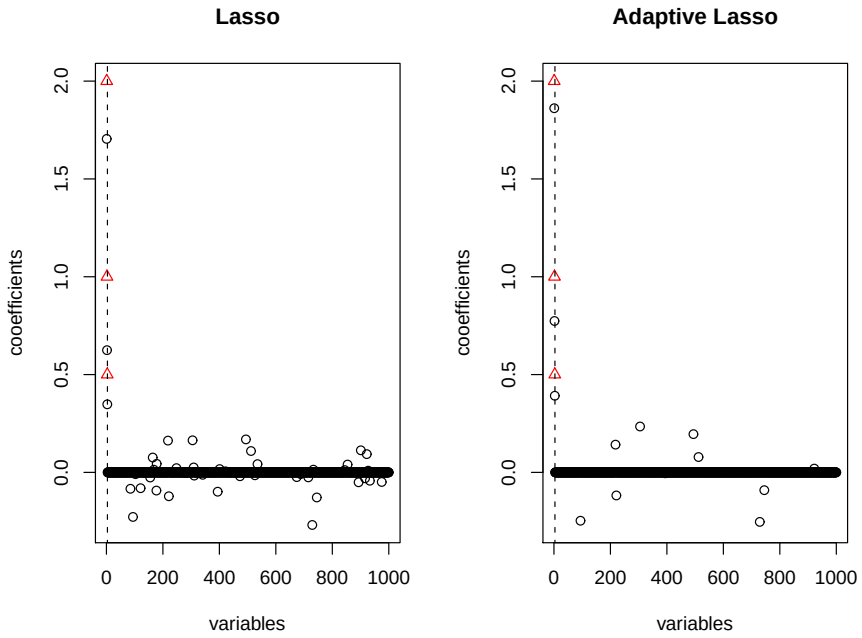

Figure 2.4: Estimated regression coefficients in linear model with $p = 1000$ and $n = 50$. Left: Lasso. Right: Adaptive Lasso with Lasso as initial estimator. Both methods used with tuning parameters selected from 10-fold cross-validation.

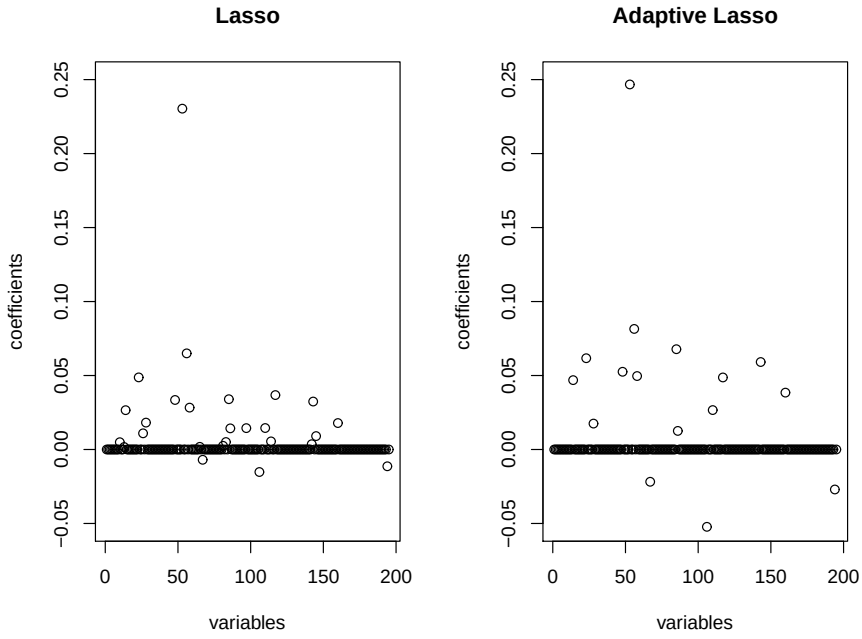

Figure 2.5: ???

2.6.1 Orthonormal design

In the special case of an orthonormal design with $p = n$ and $\hat{\Sigma} = n^{-1}\mathbf{X}^T\mathbf{X} = I_{p\times p}$, the adaptive Lasso has an explicit solution. We consider the case with the ordinary least

squares initial estimator $\hat{\beta}_{init,j} = n^{-1}(\mathbf{X}^T\mathbf{Y})_j = Z_j$ $(j = 1, \ldots, p = n)$. Then, the adaptive Lasso equals

$$\hat{\beta}_{adapt,j} = \text{sign}(Z_j)(|Z_j| - \frac{\lambda}{2|Z_j|})_+, \ Z_j = (\mathbf{X}^T\mathbf{Y})_j \ (j = 1, \ldots, p = n),$$

where $(x)_+ = \max(x, 0)$ denotes the positive part. This is again a thresholding-type estimator $\hat{\beta}_{adapt,j} = g(Z_j)$, where the thresholding function $g(\cdot)$ is depicted in Figure 2.2. The derivation is left as Exercise 2???.

Figure 2.2 has the following interpretation. Hard-thresholding yields a truncated least-squares estimator and hence, its bias is only due to the truncation (thresholding). Soft-thresholding, corresponding to Lasso, involves shrinkage, either to zero or to a value which is in absolute value smaller than the least squares estimate by λ. Hence, even if the least squares estimate is large in absolute value, soft-thresholding shrinks by the additive term λ. Finally, the adaptive Lasso "adapts" to the least squares estimate whenever the latter is large in absolute value and thus, the adaptive Lasso is less biased than the Lasso.

There is an interesting connection to the nonnegative garrote estimator [Breiman, 1995] which is defined as

$$\begin{aligned}
&\hat{\beta}_{nn-gar} = \hat{c}_j \hat{\beta}_{init,j}, \\
&\hat{c} = \ \text{argmin}_c(n^{-1}\sum_{i=1}^{n}(Y_i - \sum_{j=1}^{p} c_j \hat{\beta}_{init,j} X_i^{(j)})^2 \\
&\qquad\qquad \text{subject to } c_j \geq 0 \ (j = 1, \ldots, p) \text{ and } \sum_{j=1}^{p} c_j \leq \lambda.
\end{aligned}$$

In the special case of an orthonormal design and using ordinary least squares as initial estimator, the nonnegative garrote estimator is equal to the adaptive Lasso.

2.6.2 The adaptive Lasso: variable selection under weak conditions

For (consistent) variable selection in a linear model, the Lasso needs, as a necessary and sufficient condition, that the design matrix satisfies the neighborhood stability or irrepresentable condition described in Section 2.5.1. On the other hand, we have argued in Section 2.4 and formula (2.9) that the Lasso is reasonable for estimating the true underlying β_0 in terms of the $\|.\|_q$-norm with $q \in \{1, 2\}$. As an implication, the Lasso has the screening property where $\hat{S} \supseteq S_0$ with high probability. Thereby, we need to assume that the non-zero regression coefficients are not too small, i.e.

$$\min\{|\beta_j|; \ \beta_j \neq 0, \ j = 1, \ldots, p\} \geq C s_0 \sqrt{\log(p)/n}$$

for some constant $C > 0$, see also formula (2.13).

With the adaptive Lasso, the hope is that the two-stage process would be sufficient to correct Lasso's overestimation behavior. This can be mathematically proved, assuming compatibility conditions on the design $\mathbf{X}$ which are weaker than the neighborhood stability or irrepresentable condition. These compatibility conditions are sufficient to achieve variable selection consistency:

$$\mathbb{P}[\hat{S}_{\text{adapt.Lasso},n} = S_0] \to 1 \ (n \to \infty),$$

even if $p \gg n$. The fact that we can achieve consistent variable selection with the adaptive Lasso for cases where the Lasso is inconsistent for estimating the set S_0 is related to the issue that the adaptive Lasso exhibits less bias than the Lasso, as mentioned in Section 2.6.1. A detailed mathematical treatment for the adaptive Lasso is given in Section **??**.

2.6.3 Computation

The optimization for the adaptive Lasso in (2.21) can be re-formulated as a Lasso problem. We re-parameterize by re-scaling the covariates as follows:

$$\tilde{X}^{(j)} = |\hat{\beta}_{init,j}|X^{(j)}, \quad \tilde{\beta}_j = \frac{\beta_j}{|\hat{\beta}_{init,j}|}.$$

Then, the objective function in (2.21) becomes

$$\|\mathbf{Y} - \tilde{\mathbf{X}}\tilde{\beta}\|_2^2/n + \lambda\|\tilde{\beta}\|_1.$$

This is a Lasso-problem. Denote a solution by $\hat{\tilde{\beta}}$ and by back-transformation, we obtain a solution for the adaptive Lasso in (2.21) by

$$\hat{\beta}_{adapt} = |\hat{\beta}_{init,j}|\hat{\tilde{\beta}}_j.$$

In particular, any algorithm for solving the Lasso can be used for computation of the adaptive Lasso. We refer to Section 2.9 for Lasso algorithms.

2.6.4 Multi-step adaptive Lasso

For regularization in high-dimensional problems, we may want to use more than one or two tuning parameters. This can be achieved by pursuing more adaptive (or weighted) Lasso iterations where every iteration involves a separate tuning parameter (and as described below, these parameters are "algorithmically" constrained). The multi-step adaptive Lasso [Bühlmann and Meier, 2008] works as follows.

Multi-Step Adaptive Lasso (MSA-LASSO)

1. Initialize the weights $w_j^{(0)} \equiv 1$ $(j = 1 \ldots, p)$.
2. For $k = 1, 2 \ldots, M$:
   Use the adaptive Lasso with penalty function

   $$\lambda^{*(k)} \sum_{j=1}^{p} w_j^{(k-1)}|\beta_j|.$$

   where $\lambda^{*(k)}$ is the regularization parameter leading to prediction optimality. Denote the estimator by $\hat{\beta}^{(k)} = \hat{\beta}^{(k)}(\lambda^{*(k)})$. In practice, the value $\lambda^{*(k)}$ can be chosen with some cross-validation scheme.
   Up-date the weights

   $$w_j^{(k)} = \frac{1}{|\hat{\beta}^{(k-1)}(\lambda^{*(k-1)})_j|}, \; j = 1, \ldots, p.$$

For $k = 1$ (one-stage), we do an ordinary Lasso fit and $k = 2$ (two-stage) corresponds to the adaptive Lasso.

We will illustrate below the MSA-LASSO on a small simulated model and a real data set from molecular biology. Before doing so, we describe some properties of the method which are straightforward to derive.

First, MSA-LASSO increases the sparsity in every step in terms of the ℓ_0-norm, that is the number of selected variables decreases although there isn't necessarily a strict decrease. This follows immediately from (2.22). Second, MSA-LASSO can be computed using an algorithm for the Lasso problem in every step, see also Section 2.6.3. The computational complexity of computing all Lasso solutions over the whole range for the tuning parameter λ is of the order $O(np\min(n,p))$, see formula (2.29) below. Thus, MSA-LASSO has computational complexity $O(Mnp\min(n,p))$ since we fix the selected regularization parameter $\lambda^{*(k)}$ from the kth iteration. Due to the increase of sparsity in later, a later step is faster to compute than an early one. The computational load is in sharp contrast to computing all solutions over the whole range of all M tuning parameters: this would require $O(np(\min(n,p))^M)$ essential operations.

MSA-LASSO is related to approximating a non-convex optimization with the ℓ_r-norm penalty for r close to 0:

$$\hat{\beta}(\lambda) = \mathrm{argmin}_\beta(\|\mathbf{Y} - \mathbf{X}\beta\|_2^2/n + \lambda\|\beta\|_r^r). \tag{2.23}$$

The relation is based on an algorithmic descent property of a local linear approximation for the above ℓ_r-penalized least-squares criterion (2.23), as discussed in detail in Zou and Li [2008] (their Theorem 2 and the formula appearing just before their Proposition 2). In their framework though and in (2.23), there is a single regularization parameter λ while MSA-LASSO uses a few regularization parameters, each of them constrained to be prediction optimal. It is shown empirically in Bühlmann and Meier [2008] that the algorithmic restriction of choosing the regularization parameters in a sequentially optimal fashion is often very reasonable. We discuss in Chapter **??**, Section **??** properties of the ℓ_r-norm penalized least squares method with $0 < r < 1$.

Motif regression in computational biology

Reducing the number of false positives is often very desirable in biological applications since follow-up experiments can be costly and laborious. In fact, it can be appropriate to do conservative estimation with a low number of selected variables since we still see more selections than what may be validated in a laboratory.

We illustrate the MSA-LASSO method on a problem of motif regression [Conlon et al., 2003] for finding transcription factor binding sites in DNA sequences. Such transcription factor binding sites, also called motifs, are short "words" of DNA base pairs denoted by $\{A, C, G, T\}$, typically 6-15 base pairs long. Beer and Tavazoie [2004] contains a collection of microarray data and a collection of motif candidates for yeast. The latter is typically extracted from computational algorithms based on DNA sequence data only: for every of the n genes we have a score for each of the p candidate motifs which describes the abundance of occurrences of the candidate motif up-stream of every gene. This yields a $n \times p$ design matrix $\mathbf{X}$ with motif scores for every gene (i.e. rows of $\mathbf{X}$) and every candidate motif (i.e. columns of $\mathbf{X}$). The idea is to predict the gene expression value of a gene based on motif scores.

The dataset which we consider consists of $n = 2587$ gene expression values of a heat-shock experiment and $p = 666$ motif scores. We use a training set of size 1300 and a validation set of size 650. The remaining data is used as a test-set. We use a linear model and the MSA-LASSO for fitting the model which is fairly high-dimensional with $n_{train} \approx 2p$.

The squared prediction error on the test-set, approximating $\mathbb{E}[(\hat{Y}_{new} - Y_{new})] = (\hat{\beta} - \beta)^T \Sigma (\hat{\beta} - \beta) + \mathrm{Var}(\varepsilon)$ with $\Sigma = Cov(X)$, remains essentially constant for all estimators (probably due to high noise, i.e. large value of $\mathrm{Var}(\varepsilon)$). But the number of selected variables decreases substantially:

	Lasso ($k = 1$)	1-Step ($k = 2$)	2-Step ($k = 3$)
test set squared prediction error	0.6193	0.6230	0.6226
number of selected variables	91	42	28

The list of top-ranked candidate motifs (i.e. the selected covariates ranked according to $|\hat{\beta}_j|$) gets slightly rearranged between the different estimators. The hope (and in part a verified fact) is that the 1- or 2-step estimator yields more stable lists with fewer false positives.

2.7 The relaxed Lasso

The relaxed Lasso Meinshausen [2007] is similar to the adaptive Lasso in the sense that it addresses the bias problems of the Lasso. The method works as follows. In a first stage, all possible Lasso sub-models in $\widehat{\mathcal{SUB}}$ defined in (2.17) are computed. Then, in a second stage, every sub-model $\hat{S}$ is considered and the Lasso with smaller penalty parameter is used on such sub-models. That is, we consider the estimator

$$\begin{aligned} \hat{\beta}_{\hat{S}}(\lambda, \phi) &= \operatorname*{arg\,min}_{\beta \in \hat{S}(\lambda)} \left\{ \|Y - X\beta\|_2^2 + \phi \cdot \lambda \|\beta\|_1 \right\} \ (0 \le \phi \le 1), \\ \hat{\beta}_{\hat{S}^c}(\lambda, \phi) &= 0, \end{aligned} \tag{2.24}$$

where $\hat{S}(\lambda)$ is the estimated sub-model from the first stage (see (2.17)) and $\beta_S = \{\beta_j;\ j \in S\}$ for some subset $S \subseteq \{1, \ldots, p\}$. It is worth pointing out that once we have computed the Lasso with parameter λ in the first stage, it is often very fast to compute the relaxed estimator in (2.24). The tuning parameters λ and ϕ can be selected by some cross-validation scheme. However, unlike as for the adaptive Lasso, we should select them simultaneously. A special case occurs with $\phi = 0$ which is known as the Lasso-OLS hybrid Efron et al. [2004], using an OLS estimator in the second stage.

The relaxed and the adaptive Lasso seem to perform similarly in practice. Both procedures can be generalized to other penalties and models.

2.8 Degrees of freedom of the Lasso

Degrees of freedom are often used to quantify the complexity of a model fit and we can use them for choosing the amount of regularization. So far, we have always mentioned cross-validation for choosing reasonable tuning parameters of the Lasso or some multi-stage Lasso method. Another possibility is to use information criteria, such as the Akaike

Information Criterion (AIC) or the Bayesian Information Criterion (BIC), which penalize the likelihood by the degrees of freedom of the fitted model. For example, for a Gaussian linear model as in (2.1), the estimated model with fitted values $\hat{Y}_i$ $(i = 1, \ldots, n)$ has BIC-score:

$$\begin{aligned} \text{BIC} &= n \log(\hat{\sigma}^2) + \log(n) \cdot \text{df}(\hat{\mathbf{Y}}), \\ \hat{\sigma}^2 &= n^{-1} \sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2, \end{aligned}$$

where $\text{df}(\hat{\mathbf{Y}})$ denotes the degrees of freedom of the fitted model.

Degrees of freedom can be defined in various ways, particularly when using different estimators than maximum likelihood. Stein's theory about unbiased risk estimation leads to a rigorous definition of degrees of freedom in a Gaussian linear model as in (2.1) with fixed design and errors $\varepsilon_i \sim \mathcal{N}(0, \sigma^2)$. We denote by $\mathcal{H}\mathbf{Y} = \hat{\mathbf{Y}}$ the hat-operator which maps the response vector $\mathbf{Y} = (Y_1, \ldots, Y_n)^T$ to its fitted values $\hat{\mathbf{Y}} = (\hat{Y}_1, \ldots, \hat{Y}_n)^T$. The degrees of freedom for a possibly non-linear hat-operator $\mathcal{H}$ are then defined as

$$\text{df}(\mathcal{H}) = \sum_{i=1}^{n} \text{Cov}(\hat{Y}_i, Y_i)/\sigma^2, \tag{2.25}$$

where $\hat{Y}_i$ arise from any model fitting method, see Efron [2004].

When using maximum likelihood estimation in parametric models, the degrees of freedom equal the number of estimated parameters. Or for linear hat-operators where $\hat{\mathbf{Y}} = \mathcal{H}\mathbf{Y}$ with a hat-matrix $\mathcal{H}$, the degrees of freedom in (2.25) equal

$$\text{df}(\mathcal{H}) = \text{trace}(\mathcal{H}) \tag{2.26}$$

which is a standard formula for degrees of freedom of linear hat-operators, see Hastie and Tibshirani [1990]. The derivation of (2.26)is left as Exercise 4.

At first sight, it seems difficult to assign degrees of freedom of the Lasso. First, it is a nonlinear fitting method, e.g. soft-thresholding in the special case of an orthonormal design, and hence, formula (2.26) cannot be used. Secondly, counting the number of parameters seems completely wrong. A bit surprisingly, it is this second view which leads to a very useful formula.

We can easily count the number of non-zero estimated parameters, i.e. $|\hat{S}|$. It is plausible that shrinkage estimators involve less degrees of freedom than non-shrunken maximum likelihood estimates. On the other hand, the Lasso is estimating the sub-model with the active set $\hat{S}$, i.e. $\hat{S}$ is random, which adds variability and degrees of freedom in comparison to the situation where the model would be fixed. Surprisingly, the cost of search for estimating the model and the fact that shrinkage instead of maximum likelihood estimators are used compensate each other. The following result holds: for the Lasso with penalty parameter λ and associated hat-operator $\mathcal{H} = \mathcal{H}(\lambda)$, the degrees of freedom are,

$$\text{df}(\mathcal{H}) = \mathbb{E}[|\hat{S}|],$$

see Zou et al. [2007]. In words, the expected number of selected variables from a Lasso(λ) estimate equals the degree of freedom. A simple unbiased estimator for the degrees of freedom of the Lasso is then:

$$\widehat{\text{df}}(\mathcal{H}) = |\hat{S}|.$$

Needless to say that this formula is extremely easy to use. We can now choose the regularization parameter λ according to e.g. the BIC criterion

$$\hat{\lambda}_{\mathrm{BIC}} = \ \mathrm{argmin}_{\lambda}(n \log(n^{-1}\|\mathbf{Y} - \mathcal{H}(\lambda)\mathbf{Y}\|^2) + \log(n) \cdot |\hat{S}(\lambda)|). \tag{2.27}$$

As we will see in Section 2.9, the regularization path of $\hat{\beta}(\lambda)$ is piecewise linear as a function of λ. Hence, the minimizer of (2.27) can be evaluated exactly.

2.9 Path-following algorithms

Usually, we want to compute the estimator $\hat{\beta}(\lambda)$ in (2.2) for many values of λ. For example, selection of a good value of λ, e.g. by using cross-validation, typically requires the computation over many different candidate values.

For the estimator in (2.2), it is possible to compute the whole regularized solution path over all values of λ in the following sense. The regularized solution path $\{\hat{\beta}(\lambda);\ \lambda \in \mathbb{R}^+\}$ is piecewise linear with respect to λ. That is:

$$\begin{aligned} &\text{there exist } \lambda_0 = 0 < \lambda_1 < \lambda_{m-1} < \lambda_m = \infty,\ \gamma_0, \gamma_1, \ldots, \gamma_{m-1} \in \mathbb{R}^p \text{ such that} \\ &\hat{\beta}(\lambda) = \hat{\beta}(\lambda_k) + (\lambda - \lambda_k)\gamma_k \text{ for } \lambda_k \le \lambda \le \lambda_{k+1}\ (0 \le k \le m). \end{aligned} \tag{2.28}$$

The implication of the definition and additional fact is that the non-zero coefficients of $\hat{\beta}(\lambda)$ only change at the points λ_k, and there is a maximal value $\lambda_{\max} = \lambda_{m-1}$ where $\hat{\beta}(\lambda) = 0$ for all $\lambda \ge \lambda_{\max}$ and $\hat{\beta}_j(\lambda) \neq 0$ for $\lambda < \lambda_{\max}$ and some j. The value $\lambda_{\max}$ is characterized by

$$\lambda_{\max} = \max_{1 \le j \le p} |2\mathbf{X}_j^T \mathbf{Y}|/n.$$

This follows from the characterization of the Lasso solutions in Lemma 2.4.1. Furthermore, at every λ_k only a single component of $\hat{\beta}(\lambda_k)$ changes. The number of different λ_k-values is typically of the order $m = O(n)$, see Rosset and Zhu [2007].

The fact that the estimator in (2.2) has a piecewise linear solution path as in (2.28) has computational consequences. All what we need to compute are the values (λ_k, γ_k) $(k = 0, \ldots, m-1)$. Having these, we can easily reconstruct the whole regularized solution path. The (modified) LARS algorithm from [Efron et al., 2004] can be used for this task. Its computational complexity, for computing the whole regularization path is:

$$O(np \min(n, p)) \text{ essential operation counts.} \tag{2.29}$$

Hence, if $p \gg n$, $O(np \min(n, p)) = O(p)$ and we have a computational complexity which is linear in the dimensionality p.

Despite the fact that the LARS algorithm is exact for the whole piecewise regularization path, other algorithms described in Section 2.9.1 can be considerably faster for computing the Lasso over a large grid of λ-values [Friedman et al., 2007]. In addition, for other models and penalties, there is often no piecewise regularization path anymore and other algorithms are needed.

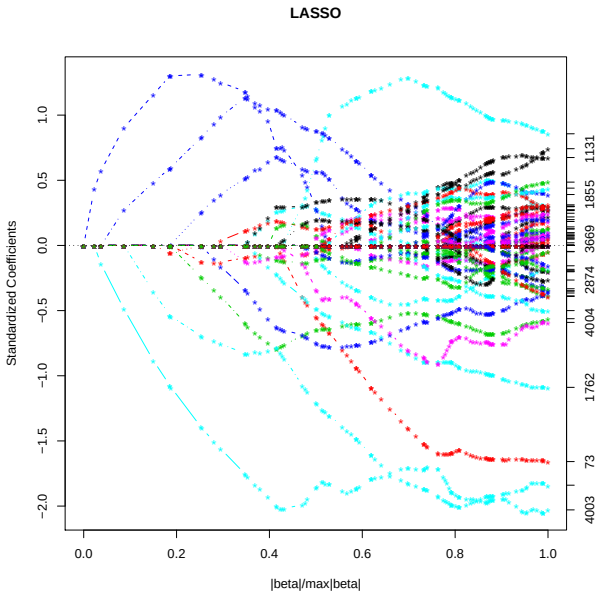

Figure 2.6: Plot of regularization path for riboflavin production data with $n = 71,\ p = 4088$. x-axis: $\|\hat{\beta}(\lambda)\|_1 / \max\{\|\hat{\beta}(\lambda)\|_1;\ \lambda\}$; y-axis: $\hat{\beta}_j \sqrt{\widehat{Var}(X^{(j)})(n-1)}$.

2.9.1 Coordinatewise optimization and shooting algorithms

For very high-dimensional but sparse problems, coordinate descent algorithms are often much faster than exact path-following methods such as the LARS-algorithm [Efron et al., 2004]. In addition, when using other loss functions than squared error or when having a group-structure in the penalty function, exact path-following algorithms are not available and other optimization algorithms are needed. These two facts are the main motivation to focus on coordinatewise methods. We refer to Efron et al. [2004] for a description of the LARS algorithm for solving the Lasso optimization in (2.2).

Despite the fact that the regularized solution path for $\hat{\beta}(\lambda)$ in (2.2) is piecewise linear, see (2.28), it is often sufficient (or even better) for practical purposes to compute $\hat{\beta}(\lambda)$ on a grid of values $\Lambda = \{0 \le \lambda_{\text{grid},1} < \lambda_{\text{grid},2} < \lambda_{\text{grid},g}\}$. In particular, the value λ_k in (2.28) are data-dependent and hence, they change for say every cross-validation run. Therefore, when determining the best regularization parameter λ with cross-validation, we have to use fixed (data-independent) candidate values for λ anyway (or work with a fixed parameter on another scale).

We recommend to choose the grid to be equi-distant on the log-scale as follows. Choose

$$\lambda_{\text{grid},g} = \lambda_{\max} = \max_{1 \le j \le p} |2\mathbf{X}_j^T \mathbf{Y}|/n,$$
$$\lambda_{\text{grid},k-1} = \lambda_{\text{grid},k} \exp(-C),$$

where $C > 0$ is a constant. Typically, we would choose C as a function of $\lambda_{\text{grid},1}$: for the latter, we recommend

$$\lambda_{\text{grid},1} \approx n^{-1},$$

and hence

$$C = \frac{\log(\lambda_{\max}) - \log(\lambda_{\text{grid},1})}{g-1}.$$

The general idea is to compute a solution $\hat{\beta}(\lambda_{\text{grid},g})$ and use it as a starting value for the computation of $\hat{\beta}(\lambda_{\text{grid},g-1})$ and so on: the value $\hat{\beta}(\lambda_{\text{grid},k})$ is used as a warm-start for the computation of $\hat{\beta}(\lambda_{\text{grid},k-1})$. Hence, we will focus in the sequel on the computation for a single regularization parameter λ.

The simplest algorithm which exploits the characterization from Lemma 2.4.1 pursues coordinate descent minimization. Denote by

$$Q_\lambda(\beta) = \|\mathbf{Y} - \mathbf{X}\beta\|^2/n + \lambda\|\beta\|_1$$

the criterion function in (2.2). Denote by

$$G_j(\beta) = 2X_j^T(\mathbf{Y} - \mathbf{X}\beta)/n$$

the gradient of $\|\mathbf{Y} - \mathbf{X}\beta\|_2^2/n$.

Consider the following algorithm.

Algorithm 1 Coordinate descent minimization

1: Let $\beta^{(0)} \in \mathbb{R}^p$ be an initial parameter vector. Set $m = 0$.
2: **repeat**
3: Increase m by one: $m = m + 1$.
4: For $j = 1, \ldots, p$:

$$\text{if } |G_j(\beta_{-j}^{(m-1)})| \le \lambda : \text{ set } \beta_j^{(m)} = 0,$$
$$\text{otherwise: } \beta_j^{(m)} = \arg\min_{\beta_j} Q_\lambda(\beta_{+j}^{(m-1)}),$$

where β_{-j} is the parameter vector setting the jth component to zero and $\beta_{+j}^{(m-1)}$ is the parameter vector which equals $\beta^{(m-1)}$ except for the jth component where it is equal to β_j.
5: **until** numerical convergence

In case of the squared error loss, we obtain an explicit up-dating formula. The gradient equals

$$G_j(\beta) = -2\mathbf{X}_j^T(y - \mathbf{X}\beta)/n$$

and the up-date is

$$\beta_j^{(m)} = \frac{\text{sign}(Z_j)(|Z_j| - \lambda/2)_+}{\hat{\Sigma}_{jj}},$$
$$Z_j = \mathbf{X}_j^T(\mathbf{Y} - \mathbf{X}\beta_{-j})/n, \quad \hat{\Sigma} = n^{-1}\mathbf{X}^T\mathbf{X}. \tag{2.30}$$

Thus, we are doing componentwise soft-thresholding. The derivation is left as Exercise 7. For more details about such an algorithm and variations for other Lasso-related problems,

we refer to Friedman et al. [2007]. Fu [1998]'s shooting algorithm for the Lasso is a special case of a coordinate descent approach.

Numerical convergence of the coordinate descent minimization algorithm is guaranteed as follows. Denote by $\hat{\beta}^{(m)}$ the parameter vector after m iterations. Then every limit point of the sequence $\{\beta^{(m)}; m = 1, 2, \ldots\}$ is a minimum point of $Q_\lambda(\cdot)$. This fact follows from the more general result in Proposition 4.6.1 in Chapter 4.

The coordinatewise optimization above can easily incorporate the more general case where some parameters are unpenalized, i.e.

$$\hat{\beta} = \arg\min_\beta Q_\lambda(\beta),$$

$$Q_\lambda(\beta) = \|\mathbf{Y} - \mathbf{X}\beta\|^2/n + \lambda \sum_{j=q+1}^{p} |\beta_j|,$$

and thus, $\beta_1, \ldots, \beta_q$ are unpenalized. The up-dating step in the optimization algorithm then looks as follows:

$$\begin{aligned}
&\text{if } j \in \{1, \ldots, q\}: \ \beta_j^{(m)} = \arg\min_{\beta_j} Q_\lambda(\beta_{+j}^{(m-1)}), \\
&\text{if } j \in \{q+1, \ldots, p\}: \\
&\quad \text{if } |G_j(\beta_{-j}^{(m-1)})| \le \lambda: \ \text{set } \beta_j^{(m)} = 0, \\
&\quad \text{otherwise: } \beta_j^{(m)} = \arg\min_{\beta_j} Q_\lambda(\beta_{+j}^{(m-1)}).
\end{aligned}$$

2.10 Functions in R

DESCRIPTION AND REFERENCE TO R-PACKAGES

2.11 Exercises

Exercise 1.
(i) Show that in the orthonormal case, the Lasso equals the soft-threshold estimator which is shown in Figure 2.2.
(ii) Show that the ℓ_0-penalty estimator in (2.16) equals the hard-threshold estimator which is shown in Figure 2.2.

Exercise 2.
(i) For the orthonormal case, derive the threshold function for the adaptive Lasso with ordinary least squares initial estimator. This threshold function is shown in Figure 2.2. (Hint: Consider every component and the parameter $\lambda_j = \lambda/|Z_j|$) .
(ii) For the orthonormal case, show that the nonnegative garrote estimator with ordinary least squares initial estimate equals the adaptive Lasso.

Exercise 3. Derive formula (2.13) from (2.9).

Exercise 4.
Prove that formula (2.26) holds for linear hat-operators $\hat{\mathbf{Y}} = \mathcal{H}\mathbf{Y}$ where $\mathcal{H}$ is linear (i.e. $\mathcal{S}$ is a $n \times n$ matrix).

Exercise 5.
Assume that (2.9) holds. For fixed $0 < C < \infty$, prove formula (2.11).

Exercise 6. (Similar to Exercise 5).
Assume that (2.12) holds. Prove formula (2.13).

Exercise 7. Prove formula (2.30).

Chapter 3

Generalized Linear Models and the Lasso

Generalized linear models (GLMs) [McCullagh and Nelder, 1989] are very useful to treat many extensions of a linear model in a unified way. We consider a model with univariate response Y and p-dimensional covariates X:

$$\begin{aligned} &Y_1, \ldots, Y_n \text{ independent} \\ &g(\mathbb{E}[Y_i|X_i = x]) = \mu + \sum_{j=1}^{p} \beta_j x^{(j)}, \end{aligned} \tag{3.1}$$

where $g(\cdot)$ is a real-valued, know link function, μ denotes the intercept term and the covariates X_i are either fixed or random. We use the notation

$$f(x) = f_{\mu,\beta}(x) = \mu + \sum j = 1^p \beta_j x^{(j)}$$

to denote the linear predictor. An implicit assumption of the model in (3.1) is that the conditional distribution of Y_i given X_i is depending on X_i only through the function $g(\mathbb{E}[Y_i|X_i]) = f_{\mu,\beta}(X_i) = \mu + \sum_{j=1}^{p} \beta_j x^{(j)}$. That is, the conditional probability (density) of $Y|X = x$ is of the form

$$p(y|x) = p_{f(x)}(y|x) = p_{\mu,\beta}(y|x), \tag{3.2}$$

where the last equality follows since the link function $g(\cdot)$ is known and hence, the unknowns are only the parameters μ and β, see (3.1).

Obviously, a linear model is a special case of a generalized linear model with the identity link function $g(x) = x$. Other well-known examples are described below.

3.1 The Lasso estimator: penalizing the negative log-likelihood

For generalized linear models, the Lasso estimator is defined by penalizing the negative log-likelihood with the ℓ_1-norm.

The negative log-likelihood equals

$$-\sum_{i=1}^{n} \log(p_{\mu,\beta}(Y_i|X_i)),$$

where $p_{\mu,\beta}(y|x)$ is as in (3.2). This expression can be re-written (and scaled by the factor n^{-1})) as an empirical risk with a loss function $\rho(.,.)$:

$$n^{-1}\sum_{i=1}^{n}\rho_{\mu,\beta}(X_i,Y_i),$$
$$\rho_{\mu,\beta}(x,y) = -\log(p_{\mu,\beta}(Y_i|X_i)).$$

For many examples and models, the loss function $\rho_{\mu,\beta}(x,y)$ is convex for all values x, y. In particular, if the conditional distribution of Y given $X = x$ is from an sub-class of the exponential family model (see McCullagh and Nelder [1989, Section 2.2]), we obtain convexity of $\rho_{\mu,\beta}(x,y) = \rho_{f(\mu,\beta)}(x,y)$ which depends on μ, β only through some linear function $f(\mu,\beta)$. Rather than striving for the most general set-up, we will present important examples below.

The ℓ_1-norm penalized Lasso estimator is then defined as:

$$\hat{\mu},\hat{\beta} = \arg\min_{\mu,\beta}(n^{-1}\sum_{i=1}^{n}\rho_{\mu,\beta}(X_i,Y_i) + \lambda\|\beta\|_1).$$

Note that we typically do not penalize the intercept term.

The properties for the Lasso in generalized linear models are very similar as for the linear models case. We have again some high-dimensional consistency, some oracle inequalities (and hence optimality) and some variable screening (and selection) properties. The theory can be derived in a similar fashion as for the Lasso in linear models, see ???.

3.1.1 Binary response variable and logistic regression

Consider the case of logistic regression where $Y_i|X_i = x \sim$ Bernoulli($\pi(x)$) (i.e. Binomial(1,$\pi(x)$)) with

$$\log\left(\frac{\pi(x)}{1-\pi(x)}\right) = \mu + \sum_{j=1}^{p}\beta_j x^{(j)}.$$

This is a GLM with link function $g(\pi) = \log(\frac{\pi}{1-\pi})$ ($\pi \in (0,1)$).

The negative log-likelihood equals

$$-\sum_{i=1}^{n}\log(f_{\mu,\beta}(Y_i|X_i)) = \sum_{i=1}^{n}\left(-Y_i f_{\mu,\beta}(X_i) + \log(1+\exp(f_{\mu,\beta}(X_i)))\right),$$

and the corresponding loss function is

$$\rho_{\mu,\beta}(x,y) = -y(\mu + \sum_{j=1}^{p}\beta_j x^{(j)}) + \log(1+\exp(\mu + \sum_{j=1}^{p}\beta_j x^{(j)})).$$

In terms of the linear predictor, this loss function equals

$$\rho(x,y) = \rho(f(x),y) = -yf + \log(1+\exp(f)),$$

where we abbreviate $f(x) = f$ on the right hand side. This is a convex function in f since the first term is linear, the second term has positive second derivative and the sum

of convex functions is convex. Furthermore, $f = f_{\mu,\beta}(x) = \mu + \sum_{j=1}^{p} \beta_j x^{(j)}$ is linear and hence

$$\rho_{\mu,\beta}(x, y) = h_y(f_{\mu,\beta}(x))$$

is convex in μ, β as a composition of a convex function $h_y(\cdot)$ (convex for all y) and a linear function.

The loss function can be written as

$$\begin{aligned} &\rho(f, y) = \log(1 + \exp(-(2y - 1)f)) = \log(1 + \exp(-\tilde{y}f)), \\ &\tilde{y} = 2y - 1 \in \{-1, 1\}. \end{aligned} \tag{3.3}$$

We see from this formulation that the loss function is a function of $\rho(\tilde{y}f)$ of a single argument, the so-called margin in binary classification. We leave the derivation of (3.3) as Exercise ???. By scaling, the equivalent loss function is often used:

$$\rho(f, y) = \log_2(1 + \exp(-\tilde{y}f)), \tag{3.4}$$

which equals one at the value zero and hence, it becomes an upper bound of the misclassification error, see Figure ???.

3.1.2 Poisson regression

For response variable Y taking values in $0, 1, 2, \ldots$, i.e. count data, we consider Poisson regression where the conditional distribution $Y_i|X_i = x \ \sim Poisson(\lambda(x))$. Using the link function

$$\log(\lambda(x)) = \mu + \sum_{j=1}^{p} \beta_j x^{(j)}.$$

we have a GLM as in (3.1).

The negative log-likelihood equals

$$-\sum_{i=1}^{n} \log(f_{\mu,\beta}(Y_i|X_i)) = \sum_{i=1}^{n} \{-Y_i f_{\mu,\beta}(X_i) + \exp(f_{\mu,\beta}(X_i)),$$

and the corresponding loss function is

$$\rho_{\mu,\beta}(x, y) = -y(\mu + \sum_{j=1}^{p} \beta_j x^{(j)}) + \exp(\mu + \sum_{j=1}^{p} \beta_j x^{(j)}).$$

The first term is linear and hence convex in μ, β, the second term is a composition of a convex and a linear function and hence convex in μ, β, the sum of convex functions is convex μ, β, and hence the loss function is convex in μ, β.

3.1.3 Multi-category response and Multinomial distribution

The Multinomial distribution is an example with a vector-valued link function. Consider a response $Y \in \{0, 1, \ldots, k-1\}$ which appears in multi-category classification problems. We assume that the conditional distribution of Y given $X = x$ is $Y|X = x \sim Multinom(\pi(x))$, where $\pi(x) = \pi_0(x), \ldots, \pi_{k-1}(x)$ with $\sum_{r=0}^{k-1} \pi_r(x) = 1$ for all x. The link function

$$g : [0,1]^k \to \mathbb{R}^k,\ \pi = (\pi_0, \ldots, \pi_{k-1}) \mapsto f = (f_0, \ldots, f_{k-1})$$

is easier to describe by its inverse

$$g_r^{-1}(f) = \pi_r = \frac{\exp(f_r)}{\sum_{s=0}^{k-1} \exp(f_s)},\ r = 0, \ldots, k-1.$$

This automatically ensures that $\sum_{r=0}^{k-1} \pi_r = 1$. Thus,

$$\log(\pi_r) = f_r - \log(\sum_{s=0}^{k-1} \exp(f_s)).$$

The linear predictors are parameterized as

$$f_r(x) = \mu_r + \sum_{j=1}^{p} \beta_{r;j} x^{(j)},\ r = 0, \ldots, k-1.$$

Note that this is over-parameterized since it would suffice to determine say $f_1, \ldots, f_{k-1}$ (without say f_0), but the constraint $\sum_{r=0}^{k-1} \pi_r(x) = 1$ for all x is automatically enforced.

The negative log-likelihood is

$$-\sum_{i=1}^{n} \sum_{r=0}^{k-1} \log(\pi_r(X_i)) I(Y_i = r) = \sum_{i=1}^{n} \log(\sum_{s=0}^{k-1} \exp(f_s(X_i))) - \sum_{r=0}^{k-1} I(Y_i = r) f_r(X_i),$$

$$f_r(X_i) = \mu_r + \sum_{j=1}^{p} \beta_{r;j} X_i^{(j)}.$$

The corresponding loss function is

$$\rho_{\mu,\beta}(x, y) = \log(\sum_{s=0}^{k-1} \exp(\mu_s + \sum_{j=1}^{p} \beta_{s;j} x^{(j)})) - \sum_{r=0}^{k-1} I(y = r)(\mu_r + \sum_{j=1}^{p} \beta_{r;j} x^{(j)}).$$

This is again a convex function in $\{\mu_r,\ \beta_{r;j};\ r = 0, \ldots, k-1,\ j = 1, \ldots, p\}$. The reasoning is as follows. The second term includes linear functions only and hence convexity follows since the sum of convex functions is convex. The first term is of the form

$$\log(\sum_{s=0}^{k-1} \exp(\mu_s + \sum_{j=1}^{p} \beta_{s;j} x^{(j)})) = \log(\sum_{s} \exp(f_s(\mu_s, \beta_s))),\ f_s = \mu_s + \sum_{j=1}^{p} \beta_{s;j} x^{(j)}.$$

The function

$$\log(\sum_{s} \exp(f_s)) \tag{3.5}$$

is convex in $f_0, \ldots, f_{k-1}$: it is the so-called "log-sum-exp" function, see Boyd & Vandenberghe, Section 3.1.5. Hence, the composition of linear functions $f_s(\mu_s, \beta_s)$ $(s = 0, \ldots, k-1)$ with the convexity of the "log-sum-exp" function shows that the first term is convex in the parameters $\{\mu_r,\ \beta_{r;j};\ r = 0, \ldots, k-1,\ j = 1, \ldots, p\}$ as well and hence we have convexity of the loss function (since sums of convex functions are convex). The convexity of the "log-sum-exp" function is left as an Exercise ???.

Contingency tables

The multinomial distribution arises also when modelling contingency tables. Consider q categorical factor variables $Z^{(1)}, \ldots, Z^{(q)}$ where each factor $Z^{(j)} \in \mathcal{I}^{(j)}$, $\mathcal{I}^{(j)}$ denoting a categorical space of $d^{(j)}$ levels (labels). Thus, the q factors take values in the categorical space

$$\mathcal{I} = \mathcal{I}^{(1)} \times \ldots \times \mathcal{I}^{(q)},$$

and we can enumerate $\mathcal{I} = \{i;\ i = 0, 1, \ldots, k-1\}$ where $k = \sum_{j=1}^{q} |\mathcal{I}^{(j)}|$. We then denote by

$$Y = (Z^{(1)}, \ldots, Z^{(q)}) \in \mathcal{I}.$$

The observations are $Y_1, \ldots, Y_n$ i.i.d. with $Y_i \in \mathcal{I}$ and $Y_i \sim Multinom(\pi)$ with $k = |\mathcal{I}|$-dimensional π satisfying $\sum_{r=0}^{k-1} \pi_r = 1$. Very often, a log-linear model is used:

$$\log(\pi) = \mu + X\beta,$$

with $k \times p$ $(k = |\mathcal{I}|)$ design matrix X which encodes the full saturated model (with $p = k$) or some sub-model including only intersection terms up to a certain order (with $p < k$). Typically, an intercept term μ is used to ensure that $\sum_{r=0}^{k-1} \pi_r = 1$. This can be enforced in the same way as for Multinomial regression. We use

$$\pi_r = \frac{\exp(\mu + (X\beta)_r)}{\sum_{t \in \mathcal{I}} \exp(\mu + (X\beta)_t)},\ r \in \mathcal{I} \tag{3.6}$$

which implies

$$\log(\pi_r) = \mu + (X\beta)_r - \log(\sum_{t \in \mathcal{I}} \exp(\mu + (X\beta)_t)),\ r \in \mathcal{I}.$$

With the parameterization in (3.6), the negative log-likelihood equals

$$-\sum_{i=1}^{n} \log(f_{\mu,\beta}(Y_i) = -\sum_{i=1}^{n} \sum_{r \in \mathcal{I}} I(Y_i = r)\{\mu + (X\beta)_r - \log(\sum_{t \in \mathcal{I}} \exp(\mu + (X\beta)_t))\}.$$

and the corresponding loss function, involving y only, is

$$\rho_{\mu,\beta}(y) = \log(\sum_{s \in \mathcal{I}} \exp(\mu + (X\beta)_s)) - \sum_{r \in \mathcal{I}} I(y = r)(\mu + (X\beta)_r).$$

The loss function is convex in μ, β by using the same argument as for the corresponding loss for Multinomial regression.

The Lasso estimator is then

$$\hat{\mu}, \hat{\beta} = \arg\min_{\mu,\beta} n^{-1} \sum_{i=1}^{n} \rho_{\mu,\beta}(Y_i) + \lambda \|\beta\|_1,$$

This Lasso estimator has the interesting property that it can be used for problems where many cells have zero counts, i.e. $\sum_{i=1}^{n} I(Y_i = r) = 0$ for many $r \in \mathcal{I}$, which arises when having a moderate number of factors q implying that $k = |\mathcal{I}|$ is very large. From a conceptual point of view, one would often aim for an estimator where whole main or interactions terms (with respect to the structure of the factors $Z^{(1)}, \ldots, Z^{(q)}$) are zero or not: this can be naturally achieved with the Group Lasso described in Chapter 4, see [Dahinden et al., 2007].

A major drawback of the Lasso (also without penalty; and also of the Group Lasso) estimator as defined above is its computational cost. Even when restricting the model to lower-order interactions (with $p < k$), the row-dimension of X remains to be $k = |\mathcal{I}|$ and the computation of the estimator is at least linear in k. Thus, this naive Lasso strategy can only work for say k up to say 10^6. For example, if every factors has 2 levels only, this would require approximately $2^q \leq 10^6$ and hence $q \leq \log_2(10^6) \approx 20$: that is, we cannot handle more than 20 factors with such an approach. For special cases with binary factor variables, fast componentwise ℓ_1-penalization is possible (Wainwright et al???). More generally, decomposition approaches based on graphical models can be used (Dahinden and PB ???) but they are not well understood from a theoretical perspective.

3.2 Exercises

Exercise 1.
Derive formula (3.3), i.e. the margin point of view of logistic regression.

Exercise 2.
Prove that the log-sum-exp function in (3.5) is a convex function in its k arguments $f_0, \ldots, f_{k-1}$. Hint: Prove this by directly verifying the definition of a convex function

$$f(ax + (1-a)y) \leq af(x) + (1-a)f(y)$$

for all $x, y, 0 \leq a \leq 1$.

Chapter 4

The Group Lasso

In some applications, a high-dimensional parameter vector β in a regression model is structured in groups $\mathcal{G}_1, \ldots, \mathcal{G}_q$ which build a partition of the index set $\{1, \ldots, p\}$. That is, $\cup_{j=1}^{q} \mathcal{G}_j = \{1, \ldots, p\}$ and $\mathcal{G}_j \cap \mathcal{G}_k = \emptyset$ $(j \neq k)$. The parameter vector β then carries the structure

$$\beta = (\beta_{\mathcal{G}_1}, \ldots, \beta_{\mathcal{G}_q}), \quad \beta_{\mathcal{G}_j} = \{\beta_r;\ r \in \mathcal{G}_r\}. \tag{4.1}$$

An important class of examples where some group structure occurs are in connection with factor variables. For example, consider a real-valued response variable Y and p categorical covariates $X^{(1)}, \ldots, X^{(p)}$ where each $X^{(j)} \in \mathcal{X}$ has 4 levels encoded with the labels from $\mathcal{X} = \{0, 1, 2, 3\}$. Then, for encoding a main effect, we need 3 parameters, encoding a first-order interaction requires 9 parameters and so on. Having chosen a parameterization with a parameter vector β, e.g. with sum contrasts, the group structure is as follows. The main effect of $X^{(1)}$ corresponds to $\beta_{\mathcal{G}_1}$ with $|\beta_{\mathcal{G}_1}| = 3$; and likewise, the main effect of all other variables $X^{(j)}$ corresponds to $\beta_{\mathcal{G}_j}$ with $|\beta_{\mathcal{G}_j}| = 3$ for all $j = 1, \ldots, p$. Furthermore, a first-order interaction of $X^{(1)}$ and $X^{(2)}$ corresponds to $\beta_{\mathcal{G}_{p+1}}$ with $|\beta_{\mathcal{G}_{p+1}}| = 9$, and so on.

Another example are nonparametric additive regression models where the groups $\mathcal{G}_j$ correspond to basis expansions for the jth additive function of the jth covariate $X^{(j)}$. A detailed treatment is given in Chapter **??**.

4.1 The Group Lasso penalty

When estimating models with a group structure for the parameter vector, we often want to encourage sparsity on the group-level. Either all entries of $\hat{\beta}_{\mathcal{G}_j}$ should be zero or all of them non-zero. This can be achieved with the Group Lasso penalty

$$\lambda \sum_{j=1}^{q} m_j \|\beta_{\mathcal{G}_j}\|_2, \tag{4.2}$$

where $\|\beta_{\mathcal{G}_j}\|_2$ denotes the standard Euclidean norm. The multiplier m_j serves for balancing the cases where groups are of very different sizes. Typically we would choose

$$m_j = \sqrt{T_j},$$

where T_j denotes the cardinality $|\beta_{\mathcal{G}_j}|$.

The Group Lasso estimator in a linear or generalized linear model as in (2.1) or (3.1) respectively is then defined as

$$\hat{\beta}(\lambda) = \arg\min_{\beta} Q_\lambda(\beta),$$
$$Q_\lambda(\beta) = n^{-1}\sum_{i=1}^{n} \rho_\beta(X_i, Y_i) + \lambda \sum_{j=1}^{q} m_j \|\beta_{\mathcal{G}_j}\|_2,$$

where $\rho_\beta(x, y)$ is a loss function which is convex in β. For example, $\rho_\beta(x, y) = |y - \beta^T x|^2$ or one of the loss function described in Chapter 3 or $\rho_\beta(x, y) = -\log_\beta(f(y|x))$ where $f(\cdot)$ is the density of Y given $X = x$. As in Chapter 3, we often include an unpenalized intercept term: the estimator is then

$$\hat{\mu}(\lambda), \hat{\beta}(\lambda) = \arg\min_{\mu,\beta} S_\lambda(\mu, \beta),$$
$$S_\lambda(\mu, \beta) = n^{-1}\sum_{i=1}^{n} \rho_{\mu,\beta}(X_i, Y_i) + \lambda \sum_{g=1}^{G} s(df_g)\|\beta_{\mathcal{G}_g}\|_2. \tag{4.3}$$

As examples of such loss functions we mention $\rho_{\mu,\beta}(x, y) = |y - \mu - \beta^T x|$ or loss functions described in Chapter 3. In the sequel, we often focus on the notationally simpler case without intercept; in practice the intercept term is often important but there is no conceptual difficulty in including it as described in (4.3).

Lemma 4.1.1 *Assume that $\rho_\beta(x, y) \geq C > -\infty$ for all x, y, β. Then, for $\lambda > 0$ and $m_j > 0$ for all j, the minimum in the optimization problem (4.9) is attained.*

Proof. Because $Q_\lambda(\beta) = n^{-1}\sum_{i=1}^{n} \rho_\beta(X_i, Y_i) + \lambda \sum_{j=1}^{q} m_j \|\beta_{\mathcal{G}_j}\|_2) \to \infty$ if $\|(\beta_{\mathcal{G}_1}, \ldots, \beta_{\mathcal{G}_q})\|_2 \to \infty$ the minimum is attained. □

The Group Lasso estimator has the following properties. Depending on the value of the regularization parameter λ, the estimated coefficients within a group $\mathcal{G}_j$ satisfy: $(\hat{\beta}_{\mathcal{G}_j})_r \equiv 0$ for all components $r = 1, \ldots, T_j$ or $(\hat{\beta}_{\mathcal{G}_j})_r \neq 0$ for all components $r = 1, \ldots, T_j$. This has to do with the non-differentiability of the $\sqrt{\cdot}$ function at zero. Furthermore, with trivial groups consisting of singletons $\mathcal{G}_j = j$ for all $j = 1, \ldots, p$, and using $m_j = T_j \equiv 1$, the penalty function in (4.2) equals the standard Lasso penalty. Finally, the Group Lasso penalty is invariant under orthonormal transformations.

The Group Lasso estimator has similar qualitative properties as the Lasso. It exhibits good accuracy for prediction and parameter estimation, and it has the groupwise variable screening property saying that all relevant groups whose corresponding parameter vector $\beta_{\mathcal{G}} \neq 0$ are also estimated as active groups with corresponding parameter vector $\hat{\beta}_{\mathcal{G}} \neq 0$. We give more details in Section 4.3 and present some rigorous mathematical theory in ???.

4.2 Factor variables as covariates

As mentioned earlier, grouping of the parameter vector occurs naturally with factor variables. We consider here the simple case with just two covariates $X^{(1)}, X^{(2)} \in \mathcal{X} =$

$\{0,1,2,3\}$ (where $\{0,1,2,3\}$ denotes a set of four categorical labels), i.e. two factors each having 4 levels. Consider a linear model with real-valued response Y and some dummy variables encoding the contribution of the two factors:

$$\begin{aligned} Y_i &= \mu + \sum_{k=0}^{3} \gamma_k I(X_i^{(1)} = k) + \sum_{k=0}^{3} \delta_k I(X_i^{(2)} = k) \\ &+ \sum_{k,\ell=0}^{3} \kappa_{k,\ell} I(X_i^{(1)} = k, X_i^{(2)} = \ell) + \varepsilon_i \quad (i = 1, \ldots, n), \end{aligned} \tag{4.4}$$

where we assume sum-constraints $\sum_k \gamma_k = \sum_k \delta_k = 0$, $\sum_k \kappa_{k,\ell} = \sum_\ell \kappa_{k,\ell} = 0$ for all k, ℓ, $I(\cdot)$ denotes the indicator function and $\varepsilon_1, \ldots, \varepsilon_n$ are i.i.d. variables with $\mathbb{E}[\varepsilon_i] = 0$. This model can be parameterized as

$$\mathbf{Y} = \mathbf{X}\beta + \varepsilon, \tag{4.5}$$

with $Y = (Y_1, \ldots, Y_n)$, $\varepsilon = (\varepsilon_1, \ldots, \varepsilon_n)$ and $n \times 16$ design matrix $\tilde{\mathbf{X}}$ which ensures the sum-constraints from above.

The parameterization in (4.5) can be achieved as follows. A first model matrix $\tilde{\mathbf{X}}$ can be constructed which ensures the sum-constraints mentioned above. In the R-software, the function `model.matrix` provides such a first design matrix $\tilde{X}$. Next, we center all columns of $\tilde{\mathbf{X}}$ to mean zero. This is typically more appropriate since we do not penalize the intercept term (and hence, we project onto the space of variables which are not penalized). Afterwards, we parameterize using orthonormal bases for the sub-spaces corresponding to the two main effects (parameterized in (4.4) with γ, δ) and to the interaction effect (parameterized in (4.4) with κ). As a result, we end up with a design matrix $\mathbf{X}$ as in (4.5) and we can apply the Group Lasso for estimation of β. It is worth pointing out that the sum-constraint plays no special role here: other constraints such as Helmert contrasts can be parameterized with orthonormal bases for the sub-spaces of the main effects and interactions. Since the Group Lasso penalty is invariant under orthonormal transformations of the parameter vector, the estimation results (for $\hat{\mathbf{Y}} = \mathbf{X}\hat{\beta}$) are not affected by the choice of the contrast. However, we point out that the estimation depends whether we choose orthonormal bases for the different sub-spaces or not. It is not true that orthonormal bases will necessarily yield the best results: in general, finding the best basis is a very difficult problem.

4.2.1 Prediction of splice sites in DNA sequences

The prediction of short DNA motifs plays an important role in many areas of computational biology. Gene finding algorithms such as GENIE [Burge and Karlin, 1997] often rely on the prediction of splice sites. Splice sites are the regions between coding (exons) and non-coding (introns) DNA segments. The 5' end of an intron is called a donor splice site and the 3' end an acceptor splice site. A donor site whose first two intron positions are the letters "GT" is called canonical, whereas an acceptor site is called canonical if the corresponding intron ends with "AG". An overview of the splicing process and of some models that are used for detecting splice sites can be found in Burge [1998].

We analyze here the MEMset Donor dataset. It consists of a training set of 8'415 true (encoded as $Y = 1$) and 179'438 false (encoded as $Y = 0$) human donor sites. An additional

test set contains 4'208 true and 89'717 false donor sites. A sequence of a real splice site consists of the last 3 bases of the exon and the first 6 bases of the intron. False splice sites are sequences on the DNA which match the consensus sequence at position four and five. Removing the consensus "GT" results in a sequence length of 7 with values in $\{A, C, G, T\}^7$: thus, the predictor variables are 7 factors, each having 4 levels. The data are available at `http://genes.mit.edu/burgelab/maxent/ssdata/`. A more detailed description can be found in Yeo and Burge [2004].

We fit a logistic regression model using the Group Lasso penalty for the main effects and higher-order interactions among the 7 factors $X^{(1)}, \ldots, X^{(p)}$. For $p(x) = \mathbb{P}[Y = 1|X = x]$, we model $\text{logit}(p(x))$ analogously as in (4.4), but now in the logistic setting and with 7 factors. We use the sum-constraint as encoding scheme for the dummy variables, i.e. the coefficients have to add up to zero. The entire predictor space has dimension $4^9 = 262'144$ but we restrict ourselves to interactions of at most order 2. After re-parameterizing with orthonormal bases for all groups $\mathcal{G}_j$ corresponding to the sub-spaces from main effects or interaction terms, we end up with a model

$$\text{logit}(\pi) = \beta_0 + \mathbf{X}\beta$$

with $n \times 1155$ design matrix $\mathbf{X}$. We then use the Group Lasso estimator

$$\hat{\beta}(\lambda) = \arg\min_{\beta} -\ell(\beta; Y_1, \ldots, Y_n) + \lambda \sum_{j=1}^{q} \sqrt{T_j} \|\beta_{\mathcal{G}_j}\|_2, \tag{4.6}$$

where the intercept β_0 is unpenalized and $T_j = |\mathcal{G}_j|$.

The original training dataset is used to build a smaller balanced training dataset (5'610 true, 5'610 false donor sites) and an unbalanced validation set (2'805 true, 59'804 false donor sites). All sites are chosen randomly without replacement such that the two sets are disjoint. The additional test set (4'208 true and 89'717 false donor sites) remains unchanged. Note that the ratio of true to false sites are equal for the validation and the test set.

All models are fitted on the balanced training dataset. As the ratio of true to false splice sites strongly differs from the training to the validation and the test set, the intercept is corrected as follows [King and Zeng, 2001]:

$$\widehat{\beta}_0^{corr} = \widehat{\beta}_0 - \log\left(\frac{\bar{y}}{1-\bar{y}}\right) + \log\left(\frac{\pi}{1-\pi}\right),$$

where π is the proportion of true sites in the validation set.

The penalty parameter λ is chosen according to the (unpenalized) log-likelihood score on the validation set using the corrected intercept estimate.

For a threshold $\tau \in (0, 1)$ we assign observation i to class 1 if $p_{\hat{\beta}}(x_i) > \tau$ and to class 0 otherwise. Note that the class assignment can also be constructed without intercept correction by using a different threshold.

The correlation coefficient ρ_τ corresponding to a threshold τ is defined as the Pearson correlation between the binary random variable of the true class membership and the binary random variable of the predicted class membership. In Yeo and Burge [2004] the maximal correlation coefficient

$$\rho_{max} = \max\{\rho_\tau \,|\, \tau \in (0, 1)\}$$

is used as a goodness of fit statistics on the test set.

The candidate model that was used for the Logistic Group Lasso consists of all 3-way and lower order interactions involving 64 terms or $p = 1156$ parameters. Such a Group Lasso fitted model achieves $\rho_{max} = 0.6593$ on the test set which is very competitive with published results from Yeo and Burge [2004] whose best ρ_{max} equals 0.6589.

In the spirit of the adaptive Lasso in Section 2.6 or the relaxed Lasso in 2.7, we consider here also some two-stage procedures. Instead of an adaptive group ℓ_1-penalization which could be used here, we consider the following. The first stage is Group Lasso yielding a parameter vector $\hat{\beta}(\lambda)$. Denote by $\hat{S}(\lambda) = \{j;\ \hat{\beta}_j(\lambda) \neq 0\}$. In the second stage, we either use maximum likelihood estimation (Group Lasso/MLE hybrid) or ℓ_2-penalization (Group Lasso/Ridge hybrid) on the reduced space given by the selected variables in $\hat{S}(\lambda)$. The latter amounts to the following: when splitting the parameter vector into the components $(\beta_{\hat{S}(\lambda)}, \beta_{\hat{S}^c(\lambda)})$ where the estimator $\hat{\beta}(\lambda)$ is non-zero and zero, respectively, we define:

$$\hat{\beta}_{\hat{S}(\lambda)}(\lambda, \kappa) = \underset{\beta_{\hat{S}(\lambda)}}{\arg\min} -\ell((\beta_{\hat{S}(\lambda)}, 0_{\hat{S}^c(\lambda)}); Y_1, \ldots, Y_n) + \kappa \|\beta_{\hat{S}(\lambda)}\|_2^2,$$

and for $\kappa = 0$, we have the Group Lasso/MLE hybrid. The penalty parameters λ and κ are again chosen according to the (unpenalized) log-likelihood score on the validation set using the corrected intercept estimate.

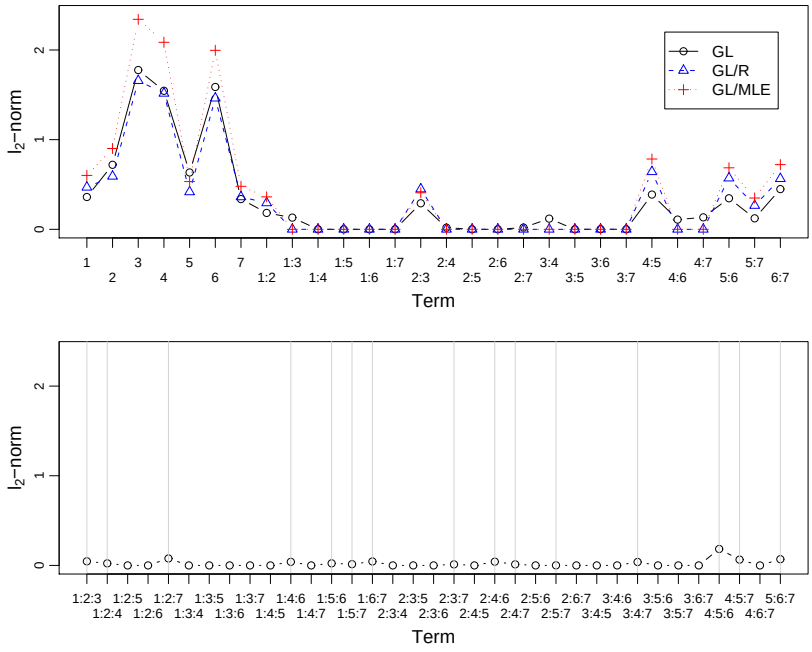

Figure 4.1: ℓ_2-norms $\|\hat{\beta}_j\|_2$, $j \in \{1, \ldots, q\}$ of the parameter groups with respect to the blockwise orthonormalized design matrix when using a candidate model with all 3-way interactions. $i : j : k$ denotes the 3-way interaction between the ith, jth and kth sequence position. The same scheme applies to the 2-way interactions and the main effects. Active 3-way interactions are additionally marked with vertical lines.

In terms of predictive accuracy, there is no benefit when using such two-stage procedures. On the other hand, while the Group Lasso solution has some active 3-way interactions, the

Group Lasso/Ridge hybrid and the Group Lasso/MLE hybrid only contain 2-way interactions. Figure 4.1 shows the ℓ_2-norm of each parameter group for the three estimators. The 3-way interactions of the Group Lasso solution seem to be very weak. Considering also the non-hierarchical models for the two-stage procedures yields the same selected terms. Decreasing the candidate model size to only contain 2-way interactions gives similar results.

In summary, the prediction performance of the Group Lasso estimate in a simple logistic regression factor model is competitive with Maximum Entropy models that were used in Yeo and Burge [2004] and which have been viewed as (among) the best for short motif modeling and splice site prediction. Advantages of the Group Lasso include selection of terms. In addition, other (possibly continuous) predictor variables as for example global sequence information could be naturally included in the Group Lasso approach to improve the rather low correlation coefficients [Yeo and Burge, 2004].

4.3 Properties of the Group Lasso for generalized linear models

Denote by $f(x) = \beta^T x$ and $\hat{f}_\lambda(x) = \hat{\beta}^T(\lambda)x$ the linear predictor and its estimate in a generalized linear model as in (3.1). For prediction, when choosing an appropriate regularization parameter λ, the Group Lasso estimator is consistent in high-dimensional settings where $p = p_n$ is of much larger order than sample size n:

$$(\hat{\beta}(\lambda) - \beta_0)^T \Sigma_X (\hat{\beta}(\lambda) - \beta_0) = o_P(1) \ (n \to \infty),$$

where Σ_X is $n^{-1}\mathbf{X}^T\mathbf{X}$ in case of a fixed design or equals the covariance of the covariate X in case of a random design. Note that the quantity on the left-hand side can be expressed as

$$\begin{aligned} &n^{-1}\sum_{i=1}^{n}\{(\hat{\beta}(\lambda) - \beta^0)^T X_i\} \text{ for fixed design,} \\ &\mathbb{E}[\{(\hat{\beta}(\lambda) - \beta^0)^T X_{new}\}^2] \text{ for random design,} \end{aligned}$$

where $\mathbb{E}$ is with respect to the new test observation X_{new} Under additional assumptions regarding the "compatibility" of the design matrix $\mathbf{X}$, we obtain the convergence rate

$$(\hat{\beta}(\lambda) - \beta_0)^T \Sigma_X (\hat{\beta}(\lambda) - \beta_0) = O_P(\frac{(1 + \log(q)/\sqrt{m})ms_0}{n\phi^2}), \tag{4.7}$$

where we assume, for simplicity, equal group-size $m \equiv \mathcal{G}_j$ for all $j = 1, \ldots, q$, q is the number of groups and ϕ^2 is a number which depends on the compatibility of the design at best is bounded below by a positive constant. More mathematical details are given in ???. When comparing this rate of convergence with (2.8) for the Lasso (which also involves a number ϕ^2 which we omitted there), we see that we do not realize any essential gain in terms of prediction power by using the Group Lasso; nor is there an essential loss assuming that m isn't very large. We also see from (4.7) that is the group-sizes are large, say in the order of sample size n, the Group Lasso is not consistent for prediction. For such cases, we need additional assumptions such as smoothness to be able to achieve consistency of predictions. This is treated in greater detail in Chapter **??**.

Furthermore, under "compatibility" assumptions on the design matrix X, the Group Lasso estimator achieves oracle optimality in terms of $\|\hat{\beta}(\lambda) - \beta\|_r$ for $r \in \{1, 2\}$. Detailed mathematical arguments are given in ???.

The variable screening property on the groupwise level, analogous to the description in Section 2.4, also holds for the Group Lasso. Denote by $S_{\text{group}} = \{j;\ \beta_{\mathcal{G}_j} \neq 0\}$ the set of groups whose corresponding coefficient vector is not equal to the 0-vector (i.e. at least one component is different from zero). Analogously we denote by $\hat{S}_{\text{group}}(\lambda)$ its estimated version. Then, for suitable $\lambda = \lambda_n$:

$$\mathbb{P}[\hat{S}_{\text{group}}(\lambda) \supseteq \mathcal{S}_{\text{group}}] \to 1 \ (n \to \infty).$$

Such a result follows from a convergence rate of $\|\hat{\beta}(\lambda) - \beta\|_1$ and assuming that the smallest non-zero coefficient is larger than a certain detection limit. More details are given in ???. The variable screening property is very useful to do effective dimensionality reduction while keeping the relevant groups in the model. Typically, the number of groups in $\widehat{S}_{\text{group}}$ is much smaller than the total number q of groups. Furthermore, if the group-sizes are relatively small, the number of parameters in $\widehat{\mathcal{S}}_{\text{group}}$ is often smaller than sample size n. As pointed out above, if the group-sizes are large, additional smoothness assumptions still yield statistically meaningful (or even optimal) results. This topic is treated in greater detail in Chapter **??**. We emphasize that even if there is no prediction gain (nor loss) with the Group Lasso i comparison to the Lasso, it may still be very worthwhile to use it since it has sparsity for whole groups and corresponding group selection may be very desirable in practical applications, for example when using dealing with factor variables.

4.4 The generalized Group Lasso penalty

The Group Lasso penalty in (4.2) is

$$\lambda \sum_{j=1}^{q} m_j \|\beta_{\mathcal{G}_j}\|_2 = \lambda \sum_{j=1}^{q} m_j \sqrt{\beta_{\mathcal{G}_j}^T \beta_{\mathcal{G}_j}}.$$

In some applications, we need a more flexible penalty of the form:

$$\lambda \sum_{j=1}^{q} m_j \sqrt{\beta_{\mathcal{G}_j}^T A_j \beta_{\mathcal{G}_j}}, \tag{4.8}$$

where A_j are positive definite $df_j \times df_j$ matrices. A concrete example are additive models treated in more detail in Section **??**.

Due to the fact that A_j is positive definite, we can re-parameterize:

$$\tilde{\beta}_{\mathcal{G}_j} = A_j^{1/2} \beta_{\mathcal{G}_j},$$

and hence, an ordinary Group Lasso penalty arises of the form

$$\lambda \sum_{j=1}^{q} s(df_j) \|\tilde{\beta}_{\mathcal{G}_j}\|_2.$$

The matrix $A_j^{1/2}$ can be derived as follows. Using e.g. the Choleski decomposition $A_j = R_j^T R_j$ for some quadratic matrix R_j which we denote by $A_j^{1/2} = R_j$.

Of course, we also need to re-parameterize the (generalized) linear model part:

$$X\beta = \sum_{j=1}^{q} X_{\mathcal{G}_j}\beta_{\mathcal{G}_j}.$$

The re-parameterization is then for every sub-design matrix $X_{\mathcal{G}_j}$:

$$\tilde{X}_{\mathcal{G}_j} = X_{\mathcal{G}_j}R_j^{-1} = X_{\mathcal{G}_j}A_j^{-1/2},\ j = 1,\ldots,q$$

such that $X\beta = \sum_{j=1}^{q}\tilde{X}_{\mathcal{G}_j}\tilde{\beta}_{\mathcal{G}_j}$.
The generalized Group Lasso estimator in a linear model is defined by:

$$\hat{\beta} = \arg\min_{\beta}\|Y - X\beta\|_n^2 + \lambda\sum_{j=1}^{q} s(df_j)\sqrt{\beta_{\mathcal{G}_j}^T A_j \beta_{\mathcal{G}_j}}.$$

Equivalently, we have:

$$\hat{\beta}_{\mathcal{G}_j} = A_j^{-1/2}\hat{\tilde{\beta}}_{\mathcal{G}_j},$$
$$(\hat{\tilde{\beta}}_{\mathcal{G}_1},\ldots,\hat{\tilde{\beta}}_{\mathcal{G}_1})^T = \arg\min_{\tilde{\beta}_{\mathcal{G}_1},\ldots,\tilde{\beta}_{\mathcal{G}_q}}\|Y - \sum_{j=1}^{q}\tilde{X}_{\mathcal{G}_j}\tilde{\beta}_{\mathcal{G}_j}\|_n^2 + \lambda\sum_{j=1}^{q} s(df_j)\|\tilde{\beta}_{\mathcal{G}_j}\|_2.$$

4.5 The adaptive Group Lasso

The idea of the adaptive Lasso in Section 2.6 can also be applied to the generalized Group Lasso. As a starting point, we assume to have an initial estimator $\hat{\beta_{init}}$. Ideally, it is tailored for the structure with groups $\mathcal{G}_1,\ldots,\mathcal{G}_q$ as in (4.1) so that we have sparsity of $\hat{\beta}_{init,\mathcal{G}_r}$ in the sense that the whole sub-vector estimate $\hat{\beta}_{init,\mathcal{G}_r}$ is zero or all components thereof are non-zero. A natural candidate for an initial estimator is the Group Lasso estimate in (4.6) or the generalized Group Lasso estimate with the penalty in (4.8). From a practical perspective, we would tune the regularization parameter for the initial estimator according to prediction optimality using some cross-validation scheme. Thereby, we would measure prediction accuracy with the squared error or negative log-likelihood loss.

The adaptive Group Lasso is then defined with the following re-weighted penalty. Instead of (4.2), we take

$$\lambda\sum_{j=1}^{q} s(df_j)\frac{\|\beta_{\mathcal{G}_j}\|_2}{\|\hat{\beta}_{init,\mathcal{G}_j}\|_2}.$$

In terms of computation, we can re-scale the covariates in a linear or generalized linear model:

$$\tilde{X}^{(j)} = X^{(j)}\|\hat{\beta}_{init,\mathcal{G}_r}\|_2 \text{ if } j \in \mathcal{G}_r.$$

Then, $\sum_{j=1}^{p}\beta_j X^{(j)} = \sum_{j=1}^{p}\tilde{\beta}_j\tilde{X}^{(j)}$ with

$$\tilde{\beta}_j = \frac{\beta_j}{\|\hat{\beta}_{init,\mathcal{G}_r}\|_2} \text{ if } j \in \mathcal{G}_r.$$

Hence, we can use the same program to compute the adaptive Group Lasso as for the plain (non-adaptive) case.

Obviously, we can also use an adaptive generalized Group Lasso. Instead of (4.8) we use

$$\lambda \sum_{j=1}^{q} s(df_j) \frac{\sqrt{\beta_{\mathcal{G}_j}^T A_j \beta_{\mathcal{G}_j}}}{\sqrt{\hat{\beta}_{init,\mathcal{G}_j}^T A_j \hat{\beta}_{init,\mathcal{G}_j}}}.$$

As above, for computation we can make a reduction to a plain generalized Group Lasso penalty by the rescaling,

$$\tilde{X}^{(j)} = X^{(j)} w_j \text{ if } j \in \mathcal{G}_r,$$
$$w_j = \sqrt{\hat{\beta}_{init,\mathcal{G}_j}^T A_j \hat{\beta}_{init,\mathcal{G}_j}}.$$

The adaptive Group Lasso is primarily recommended to be used for better selection of groups of variables. The heuristics and motivation is the same as for the adaptive Lasso described in Section 2.6. Moreover, when using the Group Lasso as initial estimator, the adaptive Group Lasso is always at least as sparse in terms of number of non-zero coefficients (and number of groups with non-zero coefficients). This can be desirable in practice and if the underlying true structure is indeed very sparse, we get better prediction results as well.

4.6 Algorithms for the Group Lasso

The Group Lasso estimator $\hat{\beta}_\lambda$ is given by a minimizer of the convex function

$$Q_\lambda(\beta) = n^{-1} \sum_{i=1}^{n} \rho_\beta(x_i, Y_i) + \lambda \sum_{g=1}^{G} s(df_g) \|\beta_g\|_2, \tag{4.9}$$

where $\rho_\beta(x_i, Y_i)$ is a loss function which is convex in β. For the squared error loss, we consider

$$\rho_\beta(x, Y) = |Y - x^T\beta|^2, \quad (Y \in \mathbb{R},\ x \in \mathbb{R}^p),$$

and for the logistic loss we have

$$\rho_\beta(x, Y) = -Y f_\beta(x) + \log(1 + \exp(f_\beta(x))), \quad (Y \in \{0, 1\},\ x \in \mathbb{R}^p),$$
$$f_\beta(x) = x^T\beta.$$

We denote in the sequel the empirical risk by

$$\rho(\beta) = n^{-1} \sum_{i=1}^{n} \rho_\beta(x_i, Y_i)$$

and the penalized version by

$$Q_\lambda(\beta) = \rho(\beta) + \lambda \sum_{g=1}^{q} s(df_g) \|\beta_{\mathcal{G}_\}}\|_2.$$

As a consequence of the Karush-Kuhn-Tucker (KKT) conditions (REFERENCE BERTSEKAS???), and assuming that $\rho(\beta)$ is convex, a necessary and sufficient condition for $\hat{\beta}$ to be a solution of (4.9) is

$$\begin{aligned}
&\nabla\rho(\hat{\beta})_g + \lambda s(df_g)\frac{\hat{\beta}_{\mathcal{G}_g}}{\|\hat{\beta}_{\mathcal{G}_g}\|_2} = 0 \text{ if } \hat{\beta}_{\mathcal{G}_g} \neq 0 \text{ (i.e. not equal to the 0-vector)},\\
&\|\nabla\rho(\hat{\beta})_g\|_2 \leq \lambda s(df_g) \text{ if } \hat{\beta}_{\mathcal{G}_g} \equiv 0. \qquad (4.10)
\end{aligned}$$

Proof of formula (4.10): If $\hat{\beta}_{\mathcal{G}_g} \neq 0$, the criterion function $Q_\lambda(\cdot)$ is partially differentiable with respect t $\beta_{\mathcal{G}_g}$ and it is necessary and sufficient (there are no local minima due to convexity) that these partial derivatives are zero: that is, the first equation in (4.10). If $\hat{\beta}_{\mathcal{G}_g} \equiv 0$, the criterion function $Q_\lambda(\cdot)$ is not differentiable but we can invoke subdifferential calculus (BERTSEKAS???). The subdifferential of $Q_\lambda(\cdot)$ with respect to β_g is the set

$$\begin{aligned}
&\partial Q_\lambda(\beta_g) = \{\nabla\rho(\beta_g) + \lambda\mathbf{e},\ \mathbf{e} \in E(\beta_g)\},\\
&E(\beta_g) = \{\mathbf{e} \in \mathbb{R}^{df_g} : \mathbf{e} = s(df_g)\frac{\beta_g}{\|\beta_g\|_2} \text{ if } \beta_g \neq 0 \text{ and } \|\mathbf{e}\|_2 \leq s(df_g) \text{ if } \beta_g = 0\}.
\end{aligned}$$

Note that the latter case with $\beta_g \neq 0$ is of interest: then, the vector is $\mathbf{e}$ is any vector within the ball having Euclidean radius $s(df_g)$ In addition, the parameter vector β_g minimizes $Q_\lambda(\beta_g)$ if and only if $0 \in \partial Q_\lambda(\beta_g)$ which is equivalent to the (first and) second statement in (4.10). □

Note that (4.10) is a generalization of the first statements in Lemma 2.4.1 from Section ???.

4.6.1 Block Coordinate Descent

For the squared error loss, we can proceed in a simple way using some block coordinate descent algorithm, as proposed by Yuan and Lin [2006]. The idea of block coordinate descent is more general, however, and we can use it also for other loss functions $\rho_\beta(\cdot,\cdot)$, as in formula (4.9) which is differentiable with respect to β.

We cycle through the parameter groups and minimize the objective function $Q_\lambda(\cdot)$, keeping all but the current parameter group fixed. This leads us to the algorithm presented in Table 4.1, where we denote by β_{-g} the parameter vector β when setting β_g to 0 while all other components remain unchanged. Similarly, X_g denotes the $n \times df_g$ matrix consisting of the columns of the design matrix corresponding to the predictor from group g.

In step (3), the ℓ_2-norm of the negative gradient looks as follows for the squared and logistic loss, respectively:

$$\begin{aligned}
&\|2n^{-1}X_g^T(y - X\beta_{-g})\|_2 \leq \lambda s(df_g) \ \text{ for the squared error loss},\\
&\|n^{-1}X_g^T(y - p_{\beta_{-g}})\|_2 \leq \lambda s(df_g) \ \text{ for the logistic loss},
\end{aligned}$$

where for the latter $p_{\beta_i} = \mathbb{P}_\beta[Y_i = 1|X_i]$. In step (3), we first check whether the minimum is at the non-differentiable point $\beta_g = 0$. If not, we can use a standard numerical minimizer, e.g. a Newton type algorithm, to find the optimal solution with respect to β_g.

In case of the squared error loss, the block-update is explicit if $n^{-1}X_g^TX_g = I_{df_g}$. Note that this assumption is quite "natural" since the penalty term is invariant under orthonormal

Block Coordinate Descent Algorithm

(1) Let $\beta \in \mathbb{R}^{p+1}$ be an initial parameter vector.
(2) $\beta_0 \leftarrow \arg\min_{\beta_0} Q_\lambda(\beta)$
(3) **for** $g = 1, \ldots, G$
  if $\|(-\nabla \rho_{\beta_{-g}}(Y, X))_g\|_2 \leq \lambda s(df_g)$
    $\beta_g \leftarrow 0$
  else
    $\beta_g \leftarrow \arg\min_{\beta_g} Q_\lambda(\beta)$
  end
end
(4) Repeat steps (2)–(3) until some convergence criterion is met.

Table 4.1: Group Lasso Algorithm using Block Coordinate Descent Minimization.

transformations: that is, it does not matter how we proceed to orthonormalize the design sub-matrices corresponding to the different groups. It then holds that the minimizer in Step (3) is given by:

$$\text{if } \|(-\nabla \rho_{\beta_{-g}}(Y, X))_g\|_2 = \|X_g^T(Y - X\beta_{-g})\|_2 > \lambda s(df_g):$$
$$\beta_g = \arg\min_{\beta_g} Q_\lambda(\beta) = (1 - \frac{\lambda s(df_g)/2}{\|U_g\|_2})_+ U_g, \quad U_g = n^{-1} X_g^T (y - X\beta_{-g}) = n^{-1} X_g^T y,$$

where $(x)_+ = max(x, 0)$.

In case of non-squared error loss, we need to do numerical optimization for a block-update. Then, the values of the last iteration can be used as starting values to save computing time. If the group was not in the model in the last iteration, we first go a small step in the opposite direction of the gradient of the negative log-likelihood function to ensure that we start at a differentiable point.

Proposition 4.6.1 *For the quantities in formula (4.9), assume that $Q_\lambda(\beta)$ is convex and $\rho_\beta(\cdot, \cdot)$ is differentiable with respect to β. Then, Steps (2) and (3) of the block coordinate descent algorithm perform groupwise minimizations of $Q_\lambda(\cdot)$ and are well defined in the sense that the corresponding minima are attained. Furthermore, if we denote by $\widehat{\beta}^{(t)}$ the parameter vector after t block updates, then every limit point of the sequence $\{\widehat{\beta}^{(t)}\}_{t \geq 0}$ is a minimum point of $Q_\lambda(\cdot)$.*

Proof. The fact that the groupwise minima are attained follows from the same arguments as in the proof of Lemma 4.1.1. We now show that step (3) minimizes the convex function $Q_\lambda(\beta_g)$ for $g \geq 1$. Since $Q_\lambda(\beta_g)$ is not differentiable everywhere, we invoke subdifferential calculus [Bertsekas, 1995]. The subdifferential of $Q_\lambda(\cdot)$ with respect to β_g is the set $\partial Q_\lambda(\beta_g) = \{-X_g^T(y - p_\beta) + \lambda e, e \in E(\beta_g)\}$, $E(\beta_g) = \{e \in \mathbb{R}^{df_g} : e = s(df_g)\frac{\beta_g}{\|\beta_g\|_2}$ if $\beta_g \neq 0$ and $\|e\|_2 \leq s(df_g)$ if $\beta_g = 0\}$. The parameter vector β_g minimizes $Q_\lambda(\beta_g)$ if and only if $0 \in \partial Q_\lambda(\beta_g)$ which is equivalent to the formulation of step 3. Furthermore conditions (A1), (B1) - (B3) and (C2) in Tseng [2001] hold. By Lemma 3.1 and Proposition 5.1 in Tseng [2001] every limit point of the sequence $\{\widehat{\beta}^{(t)}\}_{t \geq 0}$ is a stationary point of the convex function $Q_\lambda(\cdot)$, hence a minimum point. □

Because the iterates can be shown to stay in a compact set, the existence of a limit point is guaranteed.

The main drawback of such an algorithm is for the case of non squared error loss where the blockwise minimizations of the active groups have to be performed numerically. However, for small and moderate sized problems in the dimension p and the group sizes df_g this turns out to be sufficiently fast. For large-scale applications it would be attractive to have a closed form solution for a block update as for the case of squared error loss (but for a different problem than in Step (3) of the Block Coordinate Descent Algorithm). This will be discussed in the next subsection.

4.6.2 Block Coordinate Gradient Descent

As discussed above, the blockwise up-dates are available in closed form for squared error loss. For other loss functions, the idea is to use a quadratic approximation which then allows for some rather explicit blockwise up-dates. More technical, the key idea is to combine a quadratic approximation of the empirical loss with an additional line search which in fact is the block coordinate gradient descent method from Tseng and Yun [2008]. The description here follows closely Meier et al. [2008].

Using a second order Taylor expansion at $\widehat{\beta}^{(t)}$ and replacing the Hessian of the empirical risk $\rho(\beta)$ by a suitable matrix $H^{(t)}$ we define

$$\begin{aligned} M_\lambda^{(t)}(d) &= \rho(\widehat{\beta}^{(t)}) + d^T \nabla\rho(\widehat{\beta}^{(t)}) + \frac{1}{2} d^T H^{(t)} d \\ &\quad + \lambda \sum_{g=1}^{G} s(df_g) \|\widehat{\beta}_g^{(t)} + d_g\|_2 \\ &\approx Q_\lambda(\widehat{\beta}^{(t)} + d), \end{aligned} \tag{4.11}$$

where $d \in \mathbb{R}^{p+1}$.

Now we consider the minimization of $M_\lambda^{(t)}(\cdot)$ with respect to the gth penalized parameter group. This means that we restrict ourselves to vectors d with $d_k = 0$ for $k \neq g$. Moreover, we assume that the corresponding $df_g \times df_g$ submatrix $H_{gg}^{(t)}$ is of the form $H_{gg}^{(t)} = h_g^{(t)} \cdot I_{df_g}$ for some scalar $h_g^{(t)} \in \mathbb{R}$.

If $\|\nabla\rho(\widehat{\beta}^{(t)})_g - h_g^{(t)} \widehat{\beta}_g^{(t)}\|_2 \leq \lambda s(df_g)$, the minimizer of (4.11) is

$$d_g^{(t)} = -\widehat{\beta}_g^{(t)}.$$

Note that this is similar to ???, due to the KKT conditions ???, where we also examine the absolute value of the gradient. Otherwise,

$$d_g^{(t)} = -\frac{1}{h_g^{(t)}} \left\{ \nabla\rho(\widehat{\beta}^{(t)})_g - \lambda s(df_g) \frac{\nabla\rho(\widehat{\beta}^{(t)})_g - h_g^{(t)} \widehat{\beta}_g^{(t)}}{\|\nabla\rho(\widehat{\beta}^{(t)})_g - h_g^{(t)} \widehat{\beta}_g^{(t)}\|_2} \right\}.$$

If $d^{(t)} \neq 0$, an inexact line search using the Armijo rule has to be performed: Let $\alpha^{(t)}$ be the largest value in $\{\alpha_0 \delta^l\}_{l \geq 0}$ such that

$$Q_\lambda(\widehat{\beta}^{(t)} + \alpha^{(t)} d^{(t)}) - Q_\lambda(\widehat{\beta}^{(t)}) \leq \alpha^{(t)} \sigma \Delta^{(t)},$$

Block Coordinate Gradient Descent Algorithm

(1) Let $\beta \in \mathbb{R}^{p+1}$ be an initial parameter vector.
(2) **for** $g = 0, \ldots, G$
  $H_{gg} \leftarrow h_g(\beta) \cdot I_{df_g}$
  $d \leftarrow \underset{d \,|\, d_k = 0,\, k \neq g}{\arg\min} \; M_\lambda(d)$
  if $d \neq 0$
    $\alpha \leftarrow$ Line search
    $\beta \leftarrow \beta + \alpha \cdot d$
  end
end
(3) Repeat step (2) until some convergence criterion is met.

Table 4.2: Group Lasso Algorithm for non squared error loss using Block Coordinate Gradient Descent Minimization. An unpenalized intercept term can be easily incorporated as outlined in the text.

where $0 < \delta < 1$, $0 < \sigma < 1$, $\alpha_0 > 0$, and $\Delta^{(t)}$ is the improvement in the objective function $Q_\lambda(\cdot)$ when using a linear approximation for the log-likelihood, i.e.

$$\Delta^{(t)} = (d^{(t)})^T \nabla \rho(\widehat{\beta}^{(t)}) + \lambda s(df_g) \|\widehat{\beta}_g^{(t)} + d_g^{(t)}\|_2 - \lambda s(df_g) \|\widehat{\beta}_g^{(t)}\|_2.$$

Finally, we define

$$\widehat{\beta}^{(t+1)} = \widehat{\beta}^{(t)} + \alpha^{(t)} d^{(t)}.$$

The algorithm is outlined in Table 4.2. It is worth pointing out that the block updates are fairly explicit, similarly to the Block coordinate descent algorithm in Table 4.1 for the squared error loss.

When minimizing $M_\lambda^{(t)}(\cdot)$ with respect to a penalized group, we first have to check whether the minimum is at a non-differentiable point as outlined above. For an (unpenalized) intercept β_0, this is not necessary and the solution can be directly computed

$$d_0^{(t)} = -\frac{1}{h_0^{(t)}} \nabla \rho(\widehat{\beta}^{(t)})_0.$$

For a general matrix $H^{(t)}$ the minimization with respect to the gth parameter group depends on $H^{(t)}$ only through the corresponding submatrix $H_{gg}^{(t)}$. To ensure a reasonable quadratic approximation in (4.11), $H_{gg}^{(t)}$ is ideally chosen to be close to the corresponding submatrix of the Hessian of the empirical risk function. Restricting ourselves to matrices of the form $H_{gg}^{(t)} = h_g^{(t)} \cdot I_{df_g}$, a possible choice is [Tseng and Yun, 2008]

$$h_g^{(t)} = \max \left[\operatorname{diag} \left\{ \nabla^2 \rho(\widehat{\beta}^{(t)})_{gg} \right\}, \, c_* \right], \tag{4.12}$$

where $c_* > 0$ is a lower bound to ensure convergence (see Proposition 4.6.2). The matrix $H^{(t)}$ does not necessarily have to be recomputed in each iteration. Under some mild conditions on $H^{(t)}$ convergence of the algorithm is assured as can be seen from Tseng and Yun [2008] and from the proof of Proposition 4.6.2.

Standard choices for the tuning parameters are for example $\alpha_0 = 1$, $\delta = 0.5$, $\sigma = 0.1$ [Bertsekas, 1995, Tseng and Yun, 2008]. Other definitions of $\Delta^{(t)}$ as for example to include

the quadratic part of the improvement are also possible. We refer the reader to Tseng and Yun [2008] for more details and proofs that $\Delta^{(t)} < 0$ for $d^{(t)} \neq 0$ and that the line search can always be performed.

Proposition 4.6.2 *Assume that $\rho(\beta)$ is convex. If $H_{gg}^{(t)}$ is chosen according to (4.12), then every limit point of the sequence $\{\widehat{\beta}^{(t)}\}_{t\geq 0}$ is a minimum point of $Q_\lambda(\cdot)$.*

This result is a consequence of a more general theory on the coordinate gradient descent method, see Tseng and Yun [2008, Theorem 4.1].

Remark 4.6.1 *When cycling through the coordinate blocks, we could restrict ourselves to the current active set and visit the remaining blocks e.g. every 10th iteration to update the active set. This is especially useful for very high-dimensional settings and it easily allows for $p \approx 10^4 - 10^5$. Moreover, it is also possible to update the coordinate blocks in a non-cyclic manner or all at the same time which would allow for a parallelizable approach with the convergence result still holding.*

Remark 4.6.2 *The block coordinate gradient descent algorithm can be applied to the Group Lasso in any generalized linear model where the response y has a distribution from the exponential family. This is available in the R-package* `grplasso`.

To calculate the solutions $\widehat{\beta}_\lambda$ on a grid of the penalty parameter $0 \leq \lambda_K < \ldots < \lambda_1 \leq \lambda_{max}$ we can for example start at

$$\lambda_{max} = \max_{g\in\{1,\ldots,G\}} \frac{1}{s(df_g)} \|\nabla\rho(\beta)_g|_{\beta\equiv 0}\|_2$$

where all parameters in all the groups are equal to zero. We then use $\widehat{\beta}_{\lambda_k}$ as a starting value for $\widehat{\beta}_{\lambda_{k+1}}$ and proceed iteratively until $\widehat{\beta}_{\lambda_K}$ with λ_K equal or close to zero. Instead of updating the approximation of the Hessian $H^{(t)}$ in each iteration, we can use a constant matrix based on the previous parameter estimates $\widehat{\beta}_{\lambda_k}$ to save computing time, i.e.

$$H_{gg}^{(t)} = h_g(\widehat{\beta}_{\lambda_k}) I_{df_g},$$

for the estimation of $\widehat{\beta}_{\lambda_{k+1}}$. Some cross-validation can then be used for choosing the parameter λ.

Bibliography

Michael A. Beer and Saeed Tavazoie. Predicting gene expression from sequence. *Cell*, 117: 185–198, 2004.

D.P. Bertsekas. *Nonlinear programming*. Athena Scientific, Belmont, MA, 1995.

L. Breiman. Better subset regression using the nonnegative garrote. *Technometrics*, 37: 373–384, 1995.

L. Breiman. Bagging predictors. *Machine Learning*, 24:123–140, 1996.

Peter Bühlmann and Lukas Meier. Discussion of "One-step sparse estimates in nonconcave penalized likelihood models" (auths Zou, H. and Li, R.). *Annals of Statistics*, 36(4): 1534–1541, 2008.

Christopher Burge. Modeling dependencies in pre-mRNA splicing signals. In S.L. Salzberg, D.B. Searls, and S. Kasif, editors, *Computatational Methods in Molecular Biology*, chapter 8, pages 129–164. Elsevier Science, 1998.

Christopher Burge and Samuel Karlin. Prediction of complete gene structures in human genomic DNA. *Journal of Molecular Biology*, 268(1):78–94, 1997.

Erin M. Conlon, X. Shirley Liu, Jason D. Lieb, and Jun S. Liu. Integrating regulatory motif discovery and genome-wide expression analysis. *Proceedings of the National Academy of Science*, 100:3339 – 3344, 2003.

C. Dahinden, G. Parmigiani, M.C. Emerick, and P. Bühlmann. Penalized likelihood for sparse contingency tables with an application to full-length cDNA libraries. *BMC Bioinformatics*, 8(476):1–11, 2007.

D.L. Donoho. Denoising via soft-thresholding. *IEEE Trans. Info. Theory*, 41:613–627, 1995.

D.L. Donoho and J.M. Johnstone. Ideal spatial adaptation by wavelet shrinkage. *Biometrika*, 81(3):425–455, 1994.

B. Efron. The estimation of prediction error: covariance penalties and cross-validation. *Journal of the American Statistical Association*, 99:619–632, 2004.

B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression (with discussion). *The Annals of Statistics*, 32:407–451, 2004.

Jerome Friedman, Trevor Hastie, Holger Höfling, and Robert Tibshirani. Pathwise coordinate optimization. *Annals of Applied Statistics*, 1(2):302–332, 2007.

W.J. Fu. Penalized regressions: The Bridge versus the Lasso. *Journal of Computational and Graphical Statistics*, 7:397–418, 1998.

C. Gatu, P.I. Yanev, and E.J. Kontoghiorghes. A graph approach to generate all possible regression submodels. *Comp. Statist. & Data Anal.*, 52:799–815, 2007.

E. Greenshtein and Y. Ritov. Persistence in high-dimensional predictor selection and the virtue of over-parametrization. *Bernoulli*, 10(6):971–988, 2004.

T. Hastie and R. Tibshirani. *Generalized Additive Models*. Chapman & Hall, London, 1990.

M. Hofmann, C. Gatu, and E.J. Kontoghiorghes. Efficient algorithms for computing the best subset regression models for large-scale problems. *Comp. Statist. & Data Anal.*, 52:16–29, 2007.

Gary King and Langche Zeng. Logistic regression in rare events data. *Political Analysis*, 9(2):137–163, 2001.

P. McCullagh and J.A. Nelder. *Generalized linear models*. Chapman & Hall, London, second edition, 1989.

Lukas Meier, Sara van de Geer, and Peter Bühlmann. The Group Lasso for logistic regression. *Journal of the Royal Statistical Society Series B*, 70(1):53–71, 2008.

N. Meinshausen. Relaxed Lasso. *Computational Statistics & Data Analysis*, 52:374–393, 2007.

N. Meinshausen and P. Bühlmann. High-dimensional graphs and variable selection with the Lasso. *The Annals of Statistics*, 34:1436–1462, 2006.

S. Rosset and J. Zhu. Piecewise linear regularized solution paths. *The Annals of Statistics*, 35:1012–1030, 2007.

R. Tibshirani. Regression analysis and selection via the Lasso. *Journal Royal Statist. Soc. B*, 58:267–288, 1996.

Paul Tseng. Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of Optimization Theory and Applications*, 109(3):475–494, 2001.

Paul Tseng and Sangwoon Yun. A coordinate gradient descent method for nonsmooth separable minimization. *Mathematical Programming, Series B*, 117(1):387–423, 2008.

M.A. van de Wiel, Berkhof J., and van Wieringen W.N. Testing the prediction error difference between two predictors. *Biostatistics*, 10:550–560, 2009.

M. West, C. Blanchette, H. Dressman, E. Huang, S. Ishida, R. Spang, H. Zuzan, J. Olson, J. Marks, and J. Nevins. Predicting the clinical status of human breast cancer by using gene expression profiles. *Proceedings of the National Academy of Sciences (USA)*, 98: 11462–11467, 2001.

Gene W. Yeo and Christopher B. Burge. Maximum entropy modeling of short sequence motifs with applications to RNA splicing signals. *Journal of Computational Biology*, 11 (2/3):475–494, 2004.

Ming Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society Series B*, 68(1):49–67, 2006.

P. Zhao and B. Yu. On model selection consistency of Lasso. *Journal of Machine Learning Research*, 7:2541–2563, 2006.

H. Zou. The adaptive Lasso and its oracle properties. *Journal of the American Statistical Association*, 101:1418–1429, 2006.

H. Zou and R. Li. One-step sparse estimates in nonconcave penalized likelihood models (with discussion). *The Annals of Statistics*, 36:1509–1566, 2008.

H. Zou, T. Hastie, and R. Tibshirani. On the "degrees of freedom" of the Lasso. *The Annals of Statistics*, 35:2173–2192, 2007.