TECHNISCHE UNIVERSITÄT WIEN
Vienna University of Technology

DIPLOMARBEIT

Evaluation of Penalized Regression Methods in Chemometrics

Ausgeführt am Institut für

Statistik und Wahrscheinlichkeitstheorie
der Technischen Universitat Wien

unter der Anleitung von

Ao. Univ.-Prof. Dipl.-Ing. Dr.techn. Peter Filzmoser

durch

Gerald Hlavin, BSc
Traungasse 1/14
1030 Wien

Wien, September 2013

Abstract

In many cases chemometric data can be described by linear models. Particularly challenging in model-building is the high number of variables in many chemometric data-sets. Satisfactory results are often achieved, if the complexity of the model is specifically adapted to the data. One option to control this model complexity is the use of penalties in the model estimation process. This work aims to clarify the underlying ideas of these model estimation techniques, as well as to evaluate their applicability in chemometrics.

Zusammenfassung

In vielen Fällen können chemometrische Daten durch lineare Modelle beschrieben werden. Eine besondere Herausforderung bei der Modellierung ist die hohe Anzahl an Variablen in vielen chemometrischen Datensätzen. Befriedigende Ergebnisse lassen sich oftmals nur dann erzielen, wenn die Komplexität der Modelle genau an die Daten angepasst werden kann. Eine Möglichkeit zur Kontrolle dieser Modellkomplexität ist die Verwendung von Bestrafungstermen (Penalties) bei der Modellschätzung. Die vorliegende Arbeit soll die zugrundeliegenden Ideen dieser Schätzmethoden klären, sowie ihre Anwendbarkeit in der Chemometrie evaluieren.

Contents

1. Introduction **7**

2. Least squares regression **9**

2.1. Definitions and assumptions 9

2.2. High dimensional OLS regression 10

3. Penalized least squares regression **13**

3.1. Bias-variance decomposition 13

3.2. Sparsity and unbiasedness 14

4. Penalties **21**

4.1. Ridge regression 21

4.2. The Lasso 23

4.3. Best-subset-selection 23

4.4. Fused Lasso 24

4.5. Bayesian interpretation 24

5. Bridge penalties **27**

5.1. Power family 27

5.2. Elastic net 28

5.3. Generalized elastic net 28

5.4. Minimax concave penalty 30

6. Model calibration **33**

6.1. Generalized degrees of freedom 33

6.2. Estimation of the prediction error 35

6.3. Cross-validation 35

6.4. Generalized cross-validation 37

7. Optimization methods **39**

7.1. Pathwise coordinate descent 39

7.2. Generalized path seeking 40
7.3. R packages 41

8. Method comparison **43**
8.1. Datasets 43
8.2. Model estimation procedures 43
8.3. Results 44

9. Discussion **53**

A. R Code **55**

Bibliography **61**

List of Figures

3.1. Example penalty 18
3.2. Stationary points 19

4.1. Prior weights 26

5.1. Elastic net penalties for different values of α 29
5.2. Power family penalties and generalized elastic net penalties 30
5.3. Minimax concave penalties 32

8.1. Performance on the PAC data 49
8.2. Performance on the NIR data (glucose) 50
8.3. Performance on the NIR data (ethanol) 51

1. Introduction

To describe it in the words of Wold [1995], chemometrics deals with the question "How to get chemically relevant information out of measured chemical data, how to represent and display this information, and how to get such information into data". Chemical data can be very complex. They "tend to be characterized by many measured variables on each of a few observations" [Frank and Friedman, 1993].

In this work, we focus on two important types of chemometrical data. The first set of data is collected to model quantitative structure–property relationships or qantitative structure–activity relationships (QSPR/QSAR). Furthermore we have near-infrared (NIR) spectroscopy data, where the variables have a specific correlation structure. These data sets were already analyzed in Varmuza et al. [2013] and Liebmann et al. [2009], respectively. In both works, model estimation was performed by partial least squares (PLS) regression, which is a popular linear regression method in chemometrics (further information on PLS can be found e.g. in Varmuza and Filzmoser, 2009).

PLS regression has a nice geometrical interpretation and can perform well in the high dimensional case. Penalized regression methods, although popular in other fields of applied statistics, seem to be less popular in chemometrics, compared to PLS.

In this work we evaluate penalized model estimation procedures, which are available due to the recent development in the field of penalized regression. These developments involve both, the computation algorithms and the penalties itself.

To use PLS or penalized regression, at least one predefined parameter is required, which controls the complexity of the final model. In PLS this parameter is the number of the latent variables. In penalized regression the predefined parameter controls penalization. The question naturally arises, how this parameter should be set. Besides the popular answer to perform a cross-validation and take a model which produces small error, a relatively new concept arised with the generalized degrees of freedom. They generalize the effective degrees of freedom of an ordinary least squares (OLS) regression model. We use generalized degrees

of freedom beside cross-validation in the calibration process, whereever they are implemented in the optimization procedures.

Chapter 2 states definitions and assumptions, which will be used in the present work and summarizes results and interpretations of least squares regression in the high-dimensional case. Chapter 3 motivates the use of penalized regression methods by bias-variance decomposition. In this chapter, the origin of sparsity of particular penalties is explained in a simplified framework. In Chapter 4 different penalties are presented and important interpretations are discussed, whereas in Chapter 5 these penalties are generalized by bridge penalties. In Chapter 6 the problem of model calibration is discussed. Generalized degrees of freedom are defined and motivated, as well as the prediction error estimators C_p, cross-validation and generalized cross-validation. Chapter 7 shortly describes the used optimization methods. In Chapter 8, several penalized regression procedures are compared and evaluated. Results of PLS regression are also presented for comparison. The discussion of the results can be found in Chapter 9 and the– R code is presented in the Appendix.

2. Least squares regression

2.1. Definitions and assumptions

Assume that we have given a model

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}. \tag{2.1}$$

Here $\boldsymbol{X}$ is a known $n \times p$ matrix, and $\boldsymbol{\beta}$ is a $p-$dimensional column vector of unknown coefficients. $\boldsymbol{\epsilon}$ is an $n-$dimensional vector of uncorrelated normally distributed random variables with mean 0 and standard deviation σ. Thus $\boldsymbol{y}$ is a random vector of a multivariate normal distribution with mean vector $\boldsymbol{\mu} := \boldsymbol{X}\boldsymbol{\beta}$ and covariance matrix $\sigma^2\boldsymbol{I}$. As we focus on high-dimensional problems, in what follows we mostly assume that we have (much) more variables than observations, say $p \gg n$. The variables (the columns of $\boldsymbol{X}$) are assumed to be standardized to mean 0 and variance 1, and the response vector $\boldsymbol{y}$ to be mean-centered. This assumption makes some derivations more convenient, and establishes equivariance in some of the below described estimation methods.

For fixed $\boldsymbol{X}$ the main target is now to find an estimator $\hat{\boldsymbol{\beta}}(\boldsymbol{y})$ of $\boldsymbol{\beta}$ in a way, that $\hat{\boldsymbol{\mu}}(\boldsymbol{y}) := \boldsymbol{X}\hat{\boldsymbol{\beta}}(\boldsymbol{y})$ is close to $\boldsymbol{\mu}$. It should be noted that $\hat{\boldsymbol{\beta}}(\boldsymbol{y})$ does not necessarily depend linearly on $\boldsymbol{y}$. Let $\|\cdot\|_2$ be the Euclidean norm. In the terminology of Hastie et al. [2009], if $\boldsymbol{y}_0$ is an independent replication of $\boldsymbol{y}$, the in-sample prediction error is defined as

$$\mathrm{Err}_{in}(\boldsymbol{y}) := \mathbb{E}_{\boldsymbol{y}_0}\left[\|\boldsymbol{y}_0 - \hat{\boldsymbol{\mu}}(\boldsymbol{y})\|_2^2 \mid \boldsymbol{y}\right], \tag{2.2}$$

and the residual sum of squares for a parameter vector $\boldsymbol{b}$ is

$$\mathrm{RSS}(\boldsymbol{b}) := \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{b}\|_2^2. \tag{2.3}$$

Before we discuss the properties of regularized least squares regression, basic properties of OLS estimation in the high dimensional case will be revised.

2.2. High dimensional OLS regression

The problem is stated as

$$\begin{aligned}\hat{\boldsymbol{\beta}}(\boldsymbol{y}) &= \underset{\boldsymbol{b}}{\operatorname{argmin}} \left\{ \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{b}\|_2^2 \right\} \\ &= \underset{\boldsymbol{b}}{\operatorname{argmin}} \left\{ \mathrm{RSS}(\boldsymbol{b}) \right\}\end{aligned} \tag{2.4}$$

and it is well known, that a solution of this problem can be derived algebraically by solving the normal equations

$$\boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{b} = \boldsymbol{X}^\top \boldsymbol{y}. \tag{2.5}$$

In the case of $\boldsymbol{X}$ having full rank, the solution would be derived easily by multiplying both sides of (2.5) with $\left(\boldsymbol{X}^\top \boldsymbol{X}\right)^{-1}$ from the left. In the $p \gg n$ case, $\boldsymbol{X}$ has not full rank and the inverse of $\boldsymbol{X}^\top \boldsymbol{X}$ does not exist.

Solution in the high dimensional case

Let $\boldsymbol{U}\boldsymbol{D}\boldsymbol{T}^\top$ be the singular value decomposition (SVD) of $\boldsymbol{X}$, where $\boldsymbol{U}$ and $\boldsymbol{T}$ are orthogonal matrices with dimension $n \times n$, and $p \times p$, respectively. The matrix $\boldsymbol{D}$ is of the form $[\boldsymbol{D}_1; \boldsymbol{D}_2]$, where $\boldsymbol{D}_2$ is an $n \times (p-n)$ matrix of zeros and $\boldsymbol{D}_1$ is a positive semidefinite $n \times n$ diagonal matrix with the first $r \leq n$ diagonal elements d_i being strictly positive. The SVD exists for every $\boldsymbol{X}$ (see e.g. Havlicek [2006]).

Remark 2.1. Given an $m \times n$ matrix $\boldsymbol{A}$, an $n \times m$ matrix $\boldsymbol{B}$ is called a pseudoinverse of $\boldsymbol{A}$, if it suffices the conditions $\boldsymbol{A}\boldsymbol{B}\boldsymbol{A} = \boldsymbol{A}$ and $\boldsymbol{B}\boldsymbol{A}\boldsymbol{B} = \boldsymbol{B}$. The Matrix $\boldsymbol{B}$ is called the Moore-Penrose pseudoinverse of $\boldsymbol{A}$, if additionally $\boldsymbol{A}\boldsymbol{B}$ and $\boldsymbol{B}\boldsymbol{A}$ are symmetric. The Moore-Penrose pseudoinverse of a matrix is unique.

It is easy to see, that the $n \times n$ matrix

$$\boldsymbol{D}_1^\sharp := \begin{bmatrix} d_1^{-1} & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & \ddots & \ddots & & & \vdots \\ \vdots & \ddots & d_r^{-1} & \ddots & & \vdots \\ \vdots & & \ddots & 0 & \ddots & \vdots \\ \vdots & & & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & \cdots & 0 & 0 \end{bmatrix}$$

is the Moore-Penrose pseudoinverse of $\boldsymbol{D}_1$, as is $\left[\boldsymbol{D}_1^\sharp; \boldsymbol{D}_2\right]^\top$ of $[\boldsymbol{D}_1; \boldsymbol{D}_2]$.

Now $\boldsymbol{X}^\top \boldsymbol{X}$ can be rewritten as $\boldsymbol{T}\boldsymbol{D}^2\boldsymbol{T}^\top$ with $\boldsymbol{D}^2 := \boldsymbol{D}^\top \boldsymbol{D}$. By defining the $p \times p$ matrix

$$\boldsymbol{D}^{2\sharp} := \begin{bmatrix} d_1^{-2} & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & \ddots & \ddots & & & \vdots \\ \vdots & \ddots & d_r^{-2} & \ddots & & \vdots \\ \vdots & & \ddots & 0 & \ddots & \vdots \\ \vdots & & & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & \cdots & 0 & 0 \end{bmatrix} = \left[\boldsymbol{D}_1^\sharp; \boldsymbol{D}_2\right]^\top \left[\boldsymbol{D}_1^\sharp; \boldsymbol{D}_2\right],$$

it is straightforward to verify, that $\boldsymbol{D}^{2\sharp}$ is the Moore-Penrose pseudoinverse of $\boldsymbol{D}^2$, and $\left(\boldsymbol{X}^\top \boldsymbol{X}\right)^\sharp := \boldsymbol{T}\boldsymbol{D}^{2\sharp}\boldsymbol{T}^\top$ is the Moore-Penrose pseudoinverse of $\boldsymbol{X}^\top \boldsymbol{X}$.

The next statement is taken from Koecher [2003], where the proof can be found.

Lemma 2.2. *Let $\boldsymbol{B} \in \mathbf{R}^{n\times m}$ be the pseudoinverse of a matrix $\boldsymbol{A} \in \mathbf{R}^{m\times n}$ and $\boldsymbol{c} \in \mathbf{R}^{m\times 1}$. A solution of the system of linear equations $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{c}$ exists, if and only if*

$$\boldsymbol{A}\boldsymbol{B}\boldsymbol{c} = \boldsymbol{c} \tag{2.6}$$

Solutions have the form $\boldsymbol{x}_0 = \boldsymbol{y} - \boldsymbol{B}\boldsymbol{A}\boldsymbol{y} + \boldsymbol{B}\boldsymbol{c}$ with $\boldsymbol{y} \in \mathbf{R}^{n\times 1}$. If $\boldsymbol{B}$ is even the Moore-Penrose pseudoinverse of $\boldsymbol{A}$, then of all possible solutions of $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{c}$, $\boldsymbol{x}_0 := \boldsymbol{B}\boldsymbol{c}$ is the one with the least Euclidean length.

With this result and

$$\begin{aligned} \boldsymbol{D}^{2\sharp}\left[\boldsymbol{D}_1; \boldsymbol{D}_2\right]^\top &= \left[\boldsymbol{D}_1^\sharp; \boldsymbol{D}_2\right]^\top \left[\boldsymbol{D}_1^\sharp; \boldsymbol{D}_2\right]\left[\boldsymbol{D}_1; \boldsymbol{D}_2\right]^\top, \\ &= \left[\boldsymbol{D}_1^\sharp; \boldsymbol{D}_2\right]^\top \end{aligned} \tag{2.7}$$

the next corolary immediately follows:

Corollary 2.3. *The solution of* (2.5) *with minimum Euclidean length is*

$$\hat{\boldsymbol{\beta}}(\boldsymbol{y}) = \boldsymbol{T}\left[\boldsymbol{D}_1^\sharp; \boldsymbol{D}_2\right]^\top \boldsymbol{U}^\top \boldsymbol{y}. \tag{2.8}$$

Furthermore we have

$$\hat{\boldsymbol{\mu}}(\boldsymbol{y}) = \boldsymbol{X}\hat{\boldsymbol{\beta}} = \boldsymbol{U}\boldsymbol{I}_{r,n}\boldsymbol{U}^\top \boldsymbol{y}, \tag{2.9}$$

where $\boldsymbol{I}_{r,n}$ is the $n \times n$ diagonal matrix with the first r entries of the diagonal being ones and the last $n - r$ entries being zeros.

Proof. We want to solve $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{c}$ with $\boldsymbol{A} := \boldsymbol{X}^\top \boldsymbol{X} = \boldsymbol{T}\boldsymbol{D}^2\boldsymbol{T}^\top$ and $\boldsymbol{c} := \boldsymbol{X}^\top \boldsymbol{y} =$. The Moore-Penrose pseudoinverse of $\boldsymbol{X}^\top \boldsymbol{X}$ is $\boldsymbol{T}\boldsymbol{D}^{2\sharp}\boldsymbol{T}^\top =: \boldsymbol{B}$. Now we have

$$\begin{aligned}
\boldsymbol{A}\boldsymbol{B}\boldsymbol{c} &= \boldsymbol{T}\boldsymbol{D}^2\boldsymbol{T}^\top\boldsymbol{T}\boldsymbol{D}^{2\sharp}\boldsymbol{T}^\top\boldsymbol{T}\boldsymbol{D}^\top\boldsymbol{U}^\top\boldsymbol{y} \\
&= \boldsymbol{T}\boldsymbol{D}^2\boldsymbol{D}^{2\sharp}\boldsymbol{D}^\top\boldsymbol{U}^\top\boldsymbol{y} \\
&= \underbrace{\boldsymbol{T}\underbrace{\boldsymbol{I}_{r,p}\boldsymbol{D}^\top}_{=\boldsymbol{D}^\top}\boldsymbol{U}^\top}_{=\boldsymbol{X}^\top}\boldsymbol{y} \\
&= \boldsymbol{c}
\end{aligned}$$

and according to Lemma 2.2,

$$\begin{aligned}
\boldsymbol{x}_0 &:= \boldsymbol{B}\boldsymbol{c} \\
&= \left(\boldsymbol{X}^\top\boldsymbol{X}\right)^\sharp\boldsymbol{X}^\top\boldsymbol{y} \\
&= \boldsymbol{T}\boldsymbol{D}^{2\sharp}\boldsymbol{T}^\top\boldsymbol{T}\boldsymbol{D}^\top\boldsymbol{U}^\top\boldsymbol{y} \\
&\overset{(2.7)}{=} \boldsymbol{T}\left[\boldsymbol{D}_1^\sharp; \boldsymbol{D}_2\right]^\top\boldsymbol{U}^\top\boldsymbol{y}
\end{aligned}$$

is the solution with the minimum Euclidean length.

The second statement follows immediately by using $\left[\boldsymbol{D}_1^\sharp; \boldsymbol{D}_2\right]\left[\boldsymbol{D}_1^\sharp; \boldsymbol{D}_2\right]^\top = I_{r,n}$. □

3. Penalized least squares regression

In applied science it is often necessary for a "good" model to have the following properties:

1. *It should be useful for making predictions, thus the prediction error should be small (prediction property), and*
2. *it should be interpretable. Often interpretation is much easier, if the number of variables describing the model is low (interpretability property).*

It is well known, that in many situations the OLS estimator fulfills neather of these two properties. Thus one approach of statisticians to solve this problem is to alter the regression problem by shrinking the coefficients of the OLS estimator towards zero. This can be achieved by constraining the set of possible parameter vectors $\boldsymbol{b}$, when minimizing the residual sum of squares:

$$\begin{aligned}\hat{\boldsymbol{\beta}}(\boldsymbol{y}) &= \arg\min_{\boldsymbol{b}} \left\{ \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{b}\|_2^2 \right\} \ s.t.\ p(\boldsymbol{b}) \leq t \\ &= \arg\min_{\boldsymbol{b}} \{\mathrm{RSS}(\boldsymbol{b})\} \ s.t.\ p(\boldsymbol{b}) \leq t,\end{aligned} \tag{3.1}$$

where $p(\cdot)$ is the penalization term and $t \geq 0$. In what follows, we often consider the Lagrangian form of minimization problem (3.1):

$$\hat{\boldsymbol{\beta}}(\boldsymbol{y}) = \arg\min_{\boldsymbol{b}} \{\mathrm{RSS}(\boldsymbol{b}) + \lambda p(\boldsymbol{b})\}. \tag{3.2}$$

3.1. Bias-variance decomposition

Another way to motivate penalized regression in general, is through the bias-variance decomposition. In consideration of the prediction property, estimators can be compared by their expected in-sample prediction error

$$\mathbb{E}_{\boldsymbol{y}}\left[\mathrm{Err}_{in}(\boldsymbol{y})\right] = \mathbb{E}_{\boldsymbol{y}}\left\{\mathbb{E}_{\boldsymbol{y}_0}\left[\|\boldsymbol{y}_0 - \hat{\boldsymbol{\mu}}(\boldsymbol{y})\|_2^2 \mid \boldsymbol{y}\right]\right\}.$$

Trivially the in-sample prediction error can be written as:

$$\begin{aligned}\mathrm{Err}_{in}(\boldsymbol{y}) &= \mathbb{E}_{\boldsymbol{y}_0}\left[\|(\boldsymbol{y}_0-\boldsymbol{\mu})+(\boldsymbol{\mu}-\mathbb{E}_{\boldsymbol{y}}\hat{\boldsymbol{\mu}}(\boldsymbol{y}))+(\mathbb{E}_{\boldsymbol{y}}\hat{\boldsymbol{\mu}}(\boldsymbol{y})-\hat{\boldsymbol{\mu}}(\boldsymbol{y}))\|_2^2 \mid \boldsymbol{y}\right] \\ &= \sigma^2+\|(\boldsymbol{\mu}-\mathbb{E}_{\boldsymbol{y}}\hat{\boldsymbol{\mu}}(\boldsymbol{y}))\|_2^2+\|(\mathbb{E}_{\boldsymbol{y}}\hat{\boldsymbol{\mu}}(\boldsymbol{y})-\hat{\boldsymbol{\mu}}(\boldsymbol{y}))\|_2^2+2C,\end{aligned}$$

with $C := (\boldsymbol{\mu}-\mathbb{E}_{\boldsymbol{y}}\hat{\boldsymbol{\mu}}(\boldsymbol{y}))^\top(\mathbb{E}_{\boldsymbol{y}}\hat{\boldsymbol{\mu}}(\boldsymbol{y})-\hat{\boldsymbol{\mu}}(\boldsymbol{y}))$.
Because $\mathbb{E}_{\boldsymbol{y}}[C] = \mathbb{E}_{\boldsymbol{y}}\left[(\boldsymbol{\mu}-\mathbb{E}_{\boldsymbol{y}}\hat{\boldsymbol{\mu}}(\boldsymbol{y}))^\top(\mathbb{E}_{\boldsymbol{y}}\hat{\boldsymbol{\mu}}(\boldsymbol{y})-\hat{\boldsymbol{\mu}}(\boldsymbol{y}))\right] = 0$, the expected in-sample prediction error is

$$\mathbb{E}_{\boldsymbol{y}}[\mathrm{Err}_{in}(\boldsymbol{y})] = \sigma^2+\underbrace{\|(\boldsymbol{\mu}-\mathbb{E}_{\boldsymbol{y}}\hat{\boldsymbol{\mu}}(\boldsymbol{y}))\|_2^2}_{Bias}+\underbrace{\mathbb{E}_{\boldsymbol{y}}\left[\|(\mathbb{E}_{\boldsymbol{y}}\hat{\boldsymbol{\mu}}(\boldsymbol{y})-\hat{\boldsymbol{\mu}}(\boldsymbol{y}))\|_2^2\right]}_{Variance}.$$

The second term on the right-hand side is the bias of the estimator, the third term on the right-hand side is the variance (to be more precise it is the trace of the variance matrix of $\hat{\boldsymbol{\mu}}(\boldsymbol{y})$). Using the results in Section 2.2, it is easy to see, that the OLS estimator is unbiased (note that $\boldsymbol{I}_{r,n}\boldsymbol{D} = \boldsymbol{D}$):

$$\begin{aligned}\mathbb{E}_{\boldsymbol{y}}[\hat{\boldsymbol{\mu}}_{ols}(\boldsymbol{y})] &= \boldsymbol{U}\boldsymbol{I}_{r,n}\boldsymbol{U}^\top\mathbb{E}_{\boldsymbol{y}}\boldsymbol{y} \\ &= \boldsymbol{U}\boldsymbol{I}_{r,n}\boldsymbol{U}^\top\boldsymbol{X}\boldsymbol{\beta} \\ &= \boldsymbol{U}\boldsymbol{I}_{r,n}\boldsymbol{U}^\top\boldsymbol{U}\boldsymbol{D}\boldsymbol{T}^\top\boldsymbol{\beta} \\ &= \boldsymbol{U}\boldsymbol{D}\boldsymbol{T}^\top\boldsymbol{\beta} \\ &= \boldsymbol{X}\boldsymbol{\beta}.\end{aligned}$$

The idea in constructing penalized estimators is now to accept some bias, if the variance of the estimator decreases enough to result in a smaller expected in-sample prediction error compared to OLS estimation. The decrease of the variance can be understood intuitively, because the set of the possible realizations of $\hat{\boldsymbol{\beta}}$ is $\{\boldsymbol{b}\in\mathbb{R}^p : p(\boldsymbol{b})\in[0,t]\}$, which is a proper subset of $\mathbb{R}^p$ for properly chosen $p(\cdot)$.

3.2. Sparsity and unbiasedness

In this section, sparsity and unbiasedness in problem (3.2) will be discussed, following the ideas of Fan and Li [2001]. The goal is to lay the foundation to build well-behaving and interpretable penalties.

The next assumption is only presented to make the following results geometri-

cally easier to interpret.

Assumption 1. *The columns of* $\boldsymbol{X}$ *are orthonormal and* $p \leq n$ *(*$\Leftrightarrow \boldsymbol{X}^\top \boldsymbol{X} = \boldsymbol{I}_p$*).*

Assumption 2. *The penalty* $p(\boldsymbol{b})$ *is the sum of equal penalties on every coordinate (also denoted by* p *for simplicity):*

$$p(\boldsymbol{b}) = \sum_{i=1}^{p} p(b_i) \tag{3.3}$$

According to Fan and Li [2001], assumptions 1 and 2 allow us to focus on the single components in our minimization problem:

$$\begin{aligned}
\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{b}\|_2^2 + \lambda \sum_{i=1}^{p} p(b_i) &= (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{b})^\top (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{b}) + \lambda \sum_{i=1}^{p} p(b_i) \\
&= \underbrace{\left\|\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\beta}}\right\|_2^2}_{=:c} + \left\|\boldsymbol{X}\hat{\boldsymbol{\beta}} - \boldsymbol{X}\boldsymbol{b}\right\|_2^2 + \lambda \sum_{i=1}^{p} p(b_i) \\
&= c + \left(\hat{\boldsymbol{\beta}} - \boldsymbol{b}\right)^\top \underbrace{\boldsymbol{X}^\top \boldsymbol{X}}_{\boldsymbol{I}_p} \left(\hat{\boldsymbol{\beta}} - \boldsymbol{b}\right) + \lambda \sum_{i=1}^{p} p(b_i) \\
&= \sum_{i=1}^{p} \left[\left(\boldsymbol{x}^i \boldsymbol{y} - b_i\right)^2 + \lambda p(b_i)\right],
\end{aligned} \tag{3.4}$$

where $\boldsymbol{x}^i$ denotes the ith column of $\boldsymbol{X}$, written as a row vector, and $\hat{\boldsymbol{\beta}} = \boldsymbol{X}^\top \boldsymbol{y}$ is the OLS estimate of the parameter vector $\boldsymbol{\beta}$.

The next assumption says, positive and negative coefficients are equally penalized and the zero coefficient vector will not be penalized. This assumption is fulfilled by the commonly used penalties in scientific practise.

Assumption 3. *The penalty* $p(b)$ *is symmetric around zero and* $p(0) = 0$*.*

The following assumption will provide the necessary mathematical structure to solve the given minimization problems.

Assumption 4. *The penalty* $p(b)$ *is continous on* $\mathbb{R}$ *and continuously differentiable on* $\mathbb{R} \setminus \{0\}$ *in* b*, and* $p(|b|)$ *is non decreasing.*

If the following assumption is fulfilled we can achieve sparsity. The motivation is shortly discussed in Zhang [2010] and is recapitulated below.

Assumption 5. $\dot{p}(0+) = c$ *with* $c \in (0, \infty)$

Sparsity

With Assumption 4, each term in (3.4) is continously differentiable on $\mathbb{R} \setminus \{0\}$. Minimizing every single addend will minimize the whole equation. Our goal is now to line out the stationary points of these terms, which are candidates for a global minimum.

The first-order derivation of a single term is

$$\lambda \dot{p}(b_i) + 2b_i - 2\boldsymbol{x}^i\boldsymbol{y}.$$

With $\dot{p}(0+) = 2$ according to Assumption 5 (if $\dot{p}(0+) = c$, we rescale $p(\cdot)$ by $\frac{2}{c}$ and λ by $\frac{c}{2}$) and Assumption 3 (symmetry; implies $\dot{p}(0-) = -2$), we get

$$\lim_{b_i \to 0+} [\lambda \dot{p}(b_i) + 2b_i] = 2\lambda \tag{3.5}$$

and

$$\lim_{b_i \to 0-} [\lambda \dot{p}(b_i) + 2b_i] = -2\lambda. \tag{3.6}$$

Now assume $|\boldsymbol{x}^i\boldsymbol{y}| < \lambda$. With (3.5) and (3.6) it follows, that there exist $\delta_+ > 0$ and $\delta_- < 0$, with $\lambda\dot{p}(b_i) + 2b_i > 2\boldsymbol{x}^i\boldsymbol{y}$ for $b_i \in (0, \delta_+)$, and $\lambda\dot{p}(b_i) + 2b_i < 2\boldsymbol{x}^i\boldsymbol{y}$ for $b_i \in (\delta_-, 0)$, respectively. Therefore the first-order derivation is positive on $(0, \delta_+)$ and negative on $(\delta_-, 0)$. With the assumed continuity of $p(\cdot)$, zero is a potential minimum, if $|\boldsymbol{x}^i\boldsymbol{y}| < \lambda$.

If $\dot{p}(0+) = 0$ contrary to Assumption 5, then also $\dot{p}(0-) = 0$ and $p(\cdot)$ is continously differentiable on $\mathbb{R}$. Substitution in the first order equation shows, that zero can only be a minimum, if $\boldsymbol{x}^i\boldsymbol{y} = 0$.

To get a clearer view into what this means, we assume normality as stated in Section 2.1. In this setting $\boldsymbol{x}^i\boldsymbol{y} \sim N(\beta_i, \sigma^2)$ (the variance is $\sigma^2\boldsymbol{x}^i\boldsymbol{x}^{i^\top} = \sigma^2$ and $\boldsymbol{x}^i\boldsymbol{\mu} = \boldsymbol{x}^i\boldsymbol{X}\boldsymbol{\beta} = \beta_i$ because of the orthogonality assumption). Therefore $\dot{p}(0+) = c \in \mathbb{R}^+$ implies

$$\Pr(b_i = 0 \text{ is a candidate }) \geq \Pr\left(\left|\boldsymbol{x}^i\boldsymbol{y}\right| \leq \lambda\right) = \Phi\left(\frac{\lambda - \beta_i}{\sigma}\right) - \Phi\left(\frac{-\lambda - \beta_i}{\sigma}\right) > 0,$$

and $\dot{p}(0+) = 0$ implies

$$\Pr(b_i = 0 \text{ is a candidate }) = \Pr\left(\boldsymbol{x}^i\boldsymbol{y} = 0\right) = 0.$$

The first result is the reason sparsity in penalized problems with $\dot{p}(0+) = c >$

0! The second result shows, that penalties which fulfill Assumption 4 but not Assumption 5, do not have this property.

Again looking on the first result, it can also be seen, that the sparsity can be controlled by λ. The higher λ, the higher the probability of getting estimates being exactly zero

Unbiasedness

Another property which can be useful in some cases, was pointed out by Fan and Li [2001]:

Assumption 6. $\dot{p}(b) = 0$ *for* $|b| \geq \gamma > 0$

From the first order equation $\lambda \dot{p}(b_i) + 2b_i - 2\boldsymbol{x}^i\boldsymbol{y} = 0$ we see, that $b_i = \boldsymbol{x}^i\boldsymbol{y}$ is a stationary point, if $|\boldsymbol{x}^i\boldsymbol{y}| \geq \gamma$.

Example

Here we are going to visualize the former discussed assumptions and properties. Let us assume that we have only one variable $\boldsymbol{x}$ with $\boldsymbol{x}^\top\boldsymbol{x} = 1$ Furthermore the following penalty is given:

$$p(b) = \begin{cases} 2|b| & |b| \leq \gamma \\ 2\gamma & |b| > \gamma \end{cases}, \tag{3.7}$$

where $\gamma = 2$ (Figure 3.1). It is clear that $\dot{p}(0+) = 2$ and $\dot{p}(b) = 0$ for $|b| \geq \gamma$, thus, Assumptions 5 and 6 are fulfilled. Assumption 4 is fulfilled, except for $b \in \{-2, 2\}$. These values have to be treated with particular attention.

Now let $\lambda = 1$. The first order equation leads to $\hat{\beta} = \boldsymbol{x}^\top\boldsymbol{y} - \frac{1}{2}\dot{p}\left(\hat{\beta}\right)$, which can have the possible solutions :

$$\hat{\beta} = \begin{cases} \boldsymbol{x}^\top\boldsymbol{y} & |\boldsymbol{x}^i\boldsymbol{y}| > 2 \\ \boldsymbol{x}^\top\boldsymbol{y} - 1 & \boldsymbol{x}^\top\boldsymbol{y} \in (1, 3) \\ \boldsymbol{x}^\top\boldsymbol{y} + 1 & \boldsymbol{x}^\top\boldsymbol{y} \in (-3, -1) \end{cases} \tag{3.8}$$

From (3.8) we get, that there are two stationary points in the interval $(2, 3)$ and $(-3, -2)$. This is visualized in Figure 3.2. The plots in the top row and the bottom left plot of the figure show the loss functions $\text{loss}(\beta) := \left(\boldsymbol{x}^\top\boldsymbol{y} - b\right)^2 + \lambda p(b)$ for different values of $\boldsymbol{x}^\top\boldsymbol{y}$. Let $\boldsymbol{x}^\top\boldsymbol{y} \in [0, 1]$ (top left). There exists only one

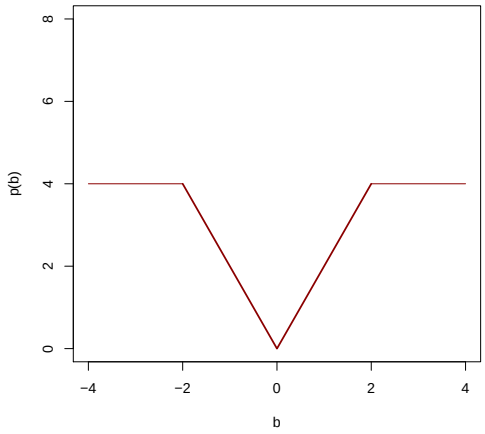


Figure 3.1.: Example penalty
$p(b)$ has the values $2|b|$ for $|b| \leq \gamma$ and 2γ for $|b| > \gamma$.

stationary point, which is 0. This follows from the above discussed fact, that $\left|\boldsymbol{x}^\top \boldsymbol{y}\right| \leq \lambda$. For $\boldsymbol{x}^\top \boldsymbol{y} \in (2,3)$ (bottom left) we have two stationary points ($\hat{\beta} = \boldsymbol{x}^\top \boldsymbol{y} - 1$ and $\hat{\beta} = \boldsymbol{x}^\top \boldsymbol{y}$). If $\boldsymbol{x}^\top \boldsymbol{y} \in (1,3)$ (top right) there is only one stationary point and the same holds true for $\boldsymbol{x}^\top \boldsymbol{y} \in [3,\infty)$. In the latter case we finaly reach unbiasedness. The bottom right plot relates $\boldsymbol{x}^\top \boldsymbol{y}$ to the solution $\hat{\beta}$. Again, sparsity, biasedness and unbiasedness is shown for different values of $\boldsymbol{x}^\top \boldsymbol{y}$.

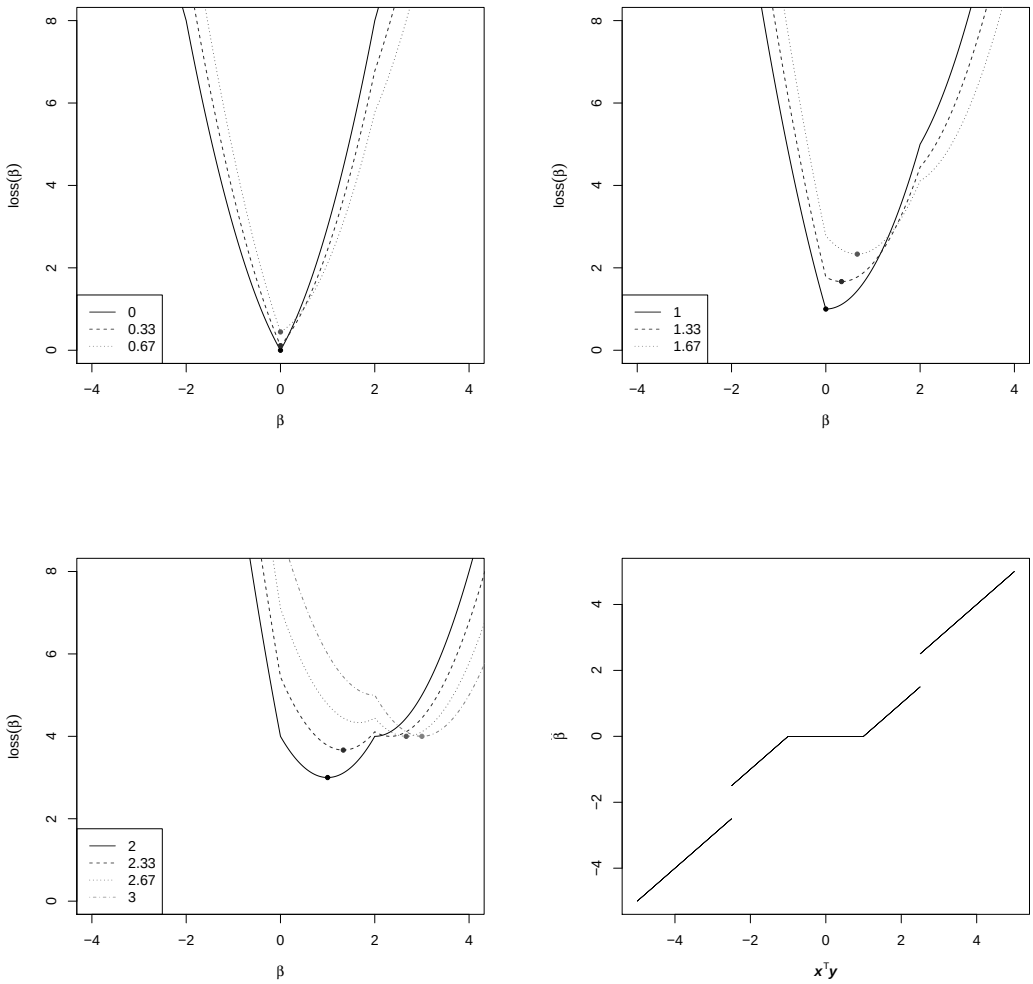


Figure 3.2.: Stationary points
The first three plots show loss functions for the example penalty. Different curves stand for different values of $\boldsymbol{x}^\top \boldsymbol{y}$. The solid circles show global minima of these curves. The last plot (bottom right) relates $\boldsymbol{x}^\top \boldsymbol{y}$ to the solution $\hat{\beta}$.

4. Penalties

In this section, we present the basic penalties for our penalized model estimation problem.

4.1. Ridge regression

We introduce Ridge regression Hoerl and Kennard [1970] by defining $p(\boldsymbol{b}) := \|\boldsymbol{b}\|_2^2$, which results in the minimization problem

$$\hat{\boldsymbol{\beta}}(\boldsymbol{y}) = \arg\min_{\boldsymbol{b}} \left\{ \mathrm{RSS}(\boldsymbol{b}) + \lambda \|\boldsymbol{b}\|_2^2 \right\} \tag{4.1}$$

with $\lambda > 0$. The first-order conditions of problem (4.1) are

$$-2\boldsymbol{X}^\top \boldsymbol{y} + 2\boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{b} + 2\lambda \boldsymbol{b} = \boldsymbol{0}$$

which lead to the solution

$$\hat{\boldsymbol{\beta}} = \left(\boldsymbol{X}^\top \boldsymbol{X} + \lambda \boldsymbol{I}_p \right)^{-1} \boldsymbol{X}^\top \boldsymbol{y}. \tag{4.2}$$

It is clear that Assumptions 2 - 4 in Section 3.2 are fulfilled, but $\dot{p}\,(0+) = 0$ contrary to Assumption 5. Therefore the Ridge solution is not sparse.

Existence and uniqueness of the solution

The symmetric matrix $\boldsymbol{X}^\top \boldsymbol{X} + \lambda \boldsymbol{I}_p$ can be written as $\boldsymbol{T}\boldsymbol{D}^2\boldsymbol{T}^\top + \lambda \boldsymbol{T}\boldsymbol{T}^\top$ by using the SVD in Section 2.2. It follows that $\boldsymbol{X}^\top \boldsymbol{X} + \lambda \boldsymbol{I}_p = \boldsymbol{T}\left(\boldsymbol{D}^2 + \lambda \boldsymbol{I}_p \right) \boldsymbol{T}^\top$ is a regular matrix when λ is strictly positive and the matrix inverse in (4.2) exists. Thus problem (4.1) has a unique solution.

Interpretation

By using above results, we have

$$\begin{aligned}\hat{\boldsymbol{\beta}} &= \left(\boldsymbol{X}^\top \boldsymbol{X} + \lambda \boldsymbol{I}_p\right)^{-1} \boldsymbol{X}^\top \boldsymbol{y} \\ &= \boldsymbol{T}\left(\boldsymbol{D}^2 + \lambda \boldsymbol{I}_p\right)^{-1} \boldsymbol{T}^\top \boldsymbol{T} \boldsymbol{D}^\top \boldsymbol{U}^\top \boldsymbol{y} \\ &= \boldsymbol{T}\left(\boldsymbol{D}^2 + \lambda \boldsymbol{I}_p\right)^{-1} \boldsymbol{D}^\top \boldsymbol{U}^\top \boldsymbol{y}\end{aligned}$$

and therefore

$$\boldsymbol{X}\hat{\boldsymbol{\beta}} = \boldsymbol{U} \underbrace{\boldsymbol{D}\left(\boldsymbol{D}^2 + \lambda \boldsymbol{I}_p\right)^{-1} \boldsymbol{D}^\top}_{=:\boldsymbol{M}} \boldsymbol{U}^\top \boldsymbol{y}. \tag{4.3}$$

The diagonal elements of the $(n \times n)$ −matrix $\boldsymbol{M}$ are $d_i^2/(d_i^2+\lambda) \leq 1$, where d_i is the (i,i) −element of $\boldsymbol{D}$. Following Hastie et al. [2009], (4.3) can be written as

$$\boldsymbol{X}\hat{\boldsymbol{\beta}} = \sum_{i=1}^{r} \boldsymbol{u}^{i\top} \left(\frac{d_i^2}{d_i^2 + \lambda}\right) \boldsymbol{u}^i \boldsymbol{y}. \tag{4.4}$$

This is the solution of the high dimensional OLS estimation in Section 2.2, except that orthonormal columns in $\boldsymbol{U}$ are shrunken towards the zero vector. From the factors $d_i^2/(d_i^2+\lambda)$ it can be seen, that for fixed λ, the shrinkage is high, if the d_i^2 are small.

Grouping effect

The grouping effect of particular regression methods works in favour of the interpretability property in Chapter 3. Equal variables result in equal coefficients, similar variables get similar coefficients. The next Lemma is the first part of Lemma 2 of Zou and Hastie [2005], where its proof can be found:

Lemma 4.1. *Assume the minimization problem* (3.2) *with* $\lambda > 0$ *and a strictly convex penalty* $p(\cdot)$*. If the columns* i *and* j *of* $\boldsymbol{X}$*, then* $\hat{\beta}_i = \hat{\beta}_j$

Because of the strict convexity of the l_2−penalty, Ridge regression estimates have the grouping effect property for equal variables.

4.2. The Lasso

Using the l_1-norm as a penalty in (3.1) results in the Lasso [Tibshirani, 1996]:

$$\hat{\boldsymbol{\beta}}(\boldsymbol{y}) = \arg\min_{\boldsymbol{\beta}} \left\{ \mathrm{RSS}(\boldsymbol{y}) + \lambda \left\| \boldsymbol{\beta} \right\|_1 \right\}. \tag{4.5}$$

It is easy to see, that in an orthonormal setting, the Lasso fulfills Assumptions 2 - 5 in Section 3.2. Therefore Lasso regression leads to sparse solutions. This is the reason, why the Lasso is now very popular in applied statistics: the variable selection and the model estimation are done simultaneously in only one step of calculation.

The results of the grouping effect in Ridge Regression do not hold true in the Lasso. Actually if $j_1, \ldots, j_k$ are the indices of equal columns of $\boldsymbol{X}$ and $\hat{\boldsymbol{\beta}}$ is a solution of (4.5), then $\boldsymbol{K}\hat{\boldsymbol{\beta}}$ is also a solution of this minimization problem. Here $\boldsymbol{K}$ is a $p \times p$-matrix defined as

$$\boldsymbol{K} := \begin{bmatrix} \alpha_1 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \alpha_p \end{bmatrix}$$

with $\sum_{i=1}^{p} \alpha_i = 1$, $\alpha_i \geq 0$ for $i \in \{j_1, \ldots, j_k\}$ and $\alpha_i = 0$ for $i \notin \{j_1, \ldots, j_k\}$. Therefore Lasso estimates are not unique in general.

Another drawback of Lasso regression in comparison to Ridge regression is, that the Lasso penalty is not differentiable at zero. Due to this fact optimization gets analytically harder with a Lasso penalty.

4.3. Best-subset-selection

On the problem of variable reduction, best-subset selection is very popular, however its formulation as a penalized regression problem is less common:

$$\hat{\boldsymbol{\beta}}(\boldsymbol{y}) = \arg\min_{\boldsymbol{b}} \left\{ \mathrm{RSS}(\boldsymbol{b}) + \lambda \left\| \boldsymbol{b} \right\|_0 \right\}, \tag{4.6}$$

Here the penalty is the l_0-"norm":

$$\left\| \boldsymbol{b} \right\|_0 := \sum_{i=1}^{p} \mathrm{I}\left(|b_i| > 0 \right),$$

where $\mathrm{I}(\cdot)$ is the indicator. Strictly speaking this does not define a norm.

As Shen and Ye [2002] pointed out, popular examples of problem (4.6) are the AIC and C_p ($\lambda = 2$ for both) and the BIC ($\lambda = \ln(n)$). It should be noted, that the usual stepwise methods do not necessarily lead to a global solution of (4.6). The derivation of global solutions for large p is computationally expensive and for very large p practically impossible (the brute-force search would require the calculation of 2^p models).

4.4. Fused Lasso

The last penalized regression technique which will be discussed in this chapter, is the Fused Lasso [Tibshirani et al., 2004]. Essentially it is the Lasso with an additional restriction, which penalizes the absolute differences of neighboring parameters. Therefore it is necessary to sort the columns of $\boldsymbol{X}$ in a logical order, where neighboring variables are highly correlated. In chemometrics this order can be the increasing wavelengths of a NIR spectroscopy for example. After sorting the problem can be stated as:

$$\hat{\boldsymbol{\beta}}(\boldsymbol{y}) = \arg\min_{\boldsymbol{b}} \left\{ \mathrm{RSS}(\boldsymbol{b}) + \lambda_1 \|\boldsymbol{b}\|_1 + \lambda_2 \|\boldsymbol{L}\boldsymbol{b}\|_1 \right\},$$

with $\lambda_1, \lambda_2 > 0$ and

$$\boldsymbol{L} := \begin{bmatrix} -1 & 1 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & -1 & 1 \end{bmatrix}.$$

4.5. Bayesian interpretation

To reach better interpretability of the penalization, we follow Frank and Friedman [1993] and present a useful Bayesian point of view.

For fixed $\boldsymbol{X}$ we consider the relation

$$\pi(\boldsymbol{b}|\boldsymbol{y}) \propto L(\boldsymbol{y}|\boldsymbol{b})\, \pi(\boldsymbol{b}),$$

where L is the likelihood (in our normal distribution framework), $\pi(\boldsymbol{b})$ is the (proper or improper) prior density of $\boldsymbol{\beta}$ and $\pi(\boldsymbol{b}|\boldsymbol{y})$ is its posterior density. One could consider the estimator $\hat{\boldsymbol{\beta}}$ as the maximizing $\boldsymbol{b}$ of $\pi(\boldsymbol{b}|\boldsymbol{y})$. The maximiza-

tion of $\pi(\boldsymbol{b}|\boldsymbol{y})$ is equivalent to the minimization of the negative logarithm of $L(\boldsymbol{y}|\boldsymbol{b})\,\pi(\boldsymbol{b})$. Now the equivalent minimization problem has the form

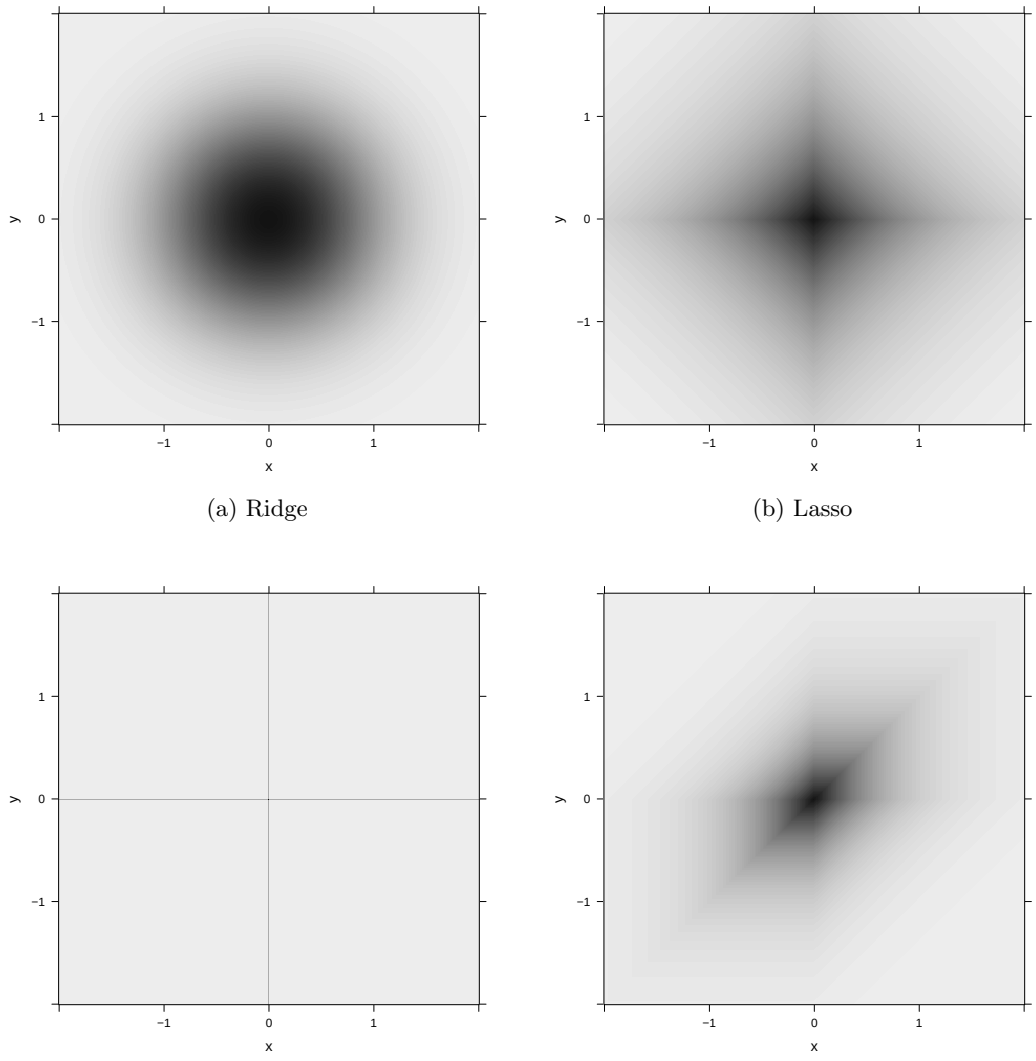
$$\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{b}\|_2^2 - 2\sigma^2 \ln(\pi(\boldsymbol{b})), \tag{4.7}$$

because multiplying by the factor $2\sigma^2$ does not change the solution. The question is now how to choose the prior $\pi(\boldsymbol{b})$ in (4.7) to get the same parameter vector, as in minimization problem (3.1). In other words, $-2\sigma^2 \ln(\pi(\boldsymbol{b}))$ has to be equal to $\lambda p(\boldsymbol{b})$, which leads to

$$\pi(\boldsymbol{b}) = \exp\left(-\left(2\sigma^2\right)^{-1} \lambda p(\boldsymbol{b})\right). \tag{4.8}$$

Substitution of the Ridge-, Lasso-, and Best-subset-penalty in (4.8) gets the (improper) prior densities, which define the equivalent minimization problems to (4.1), (4.5), and (4.6). Figure (4.1)(a)-(c) show the two-parameter case with $\lambda = (2\sigma^2)$. It can be seen that the prior induced by Ridge regression sets equal weights to parameters with equal Euclidean length. The weights are increasing near zero. In the Lasso, parameter combinations with equal Euclidean length can have different weights, depending on how near they are to a coordinate axis. In best-subset regression, only parameter vectors on a coordinate axis would be treated differently (with higher weights). In contrast, OLS regression would induce an improper prior with equal weights for each combination in $\mathbb{R}^p$.

Repeating the derivation of a Bayesian prior for fused Lasso with $\lambda_1 = \lambda_2 = (2\sigma^2)$ in Figure 4.1(d) reveals a gain of weight towards the $45°-$diagonal of two neighboring parameters. Thus the prior probability is increased, that two neighboring parameters have the same value.



(a) Ridge

(b) Lasso

(c) Best-subset selection

(d) Fused Lasso

Figure 4.1.: Prior weights

5. Bridge penalties

Each of the discussed penalties have their own advantages and disadvantages. Bridge penalties are generalizations of these penalties in the sense, that they have an additional meta parameter and that some of our former discussed penalties can be retrieved at particular values of this meta parameter. Bridge penalties can often be understood as a mixture of at least two of the penalties, discussed in Chapter 4. The additional parameter then controls the mixture ratio.

5.1. Power family

Frank and Friedman [1993] suggested a generalization of best-subset selection, Lasso- and Ridge regression. They called this generalized Ridge regression, which is also known as bridge regression [Fu, 1998]. We use the name bridge regression to denote a generalization of at least two of our basic frameworks (BSS, Lasso regression, Ridge regression).

For each $\alpha \in [0, \infty)$ we define the penalty

$$p(\boldsymbol{b}) := \|\boldsymbol{b}\|_\alpha^\alpha = \sum_{i=1}^{p} |b_i|^\alpha. \tag{5.1}$$

With this penalty and λ, $t \geq 0$ the minimization problem can again be written in two equivalent forms:

$$\hat{\boldsymbol{\beta}}(\boldsymbol{y}) = \arg\min_{\boldsymbol{b}} \{\mathrm{RSS}(\boldsymbol{b})\} \;\; s.t. \;\; \|\boldsymbol{b}\|_\alpha^\alpha \leq t, \tag{5.2}$$

and

$$\hat{\boldsymbol{\beta}}(\boldsymbol{y}) = \arg\min_{\boldsymbol{b}} \left\{ \underbrace{\mathrm{RSS}(\boldsymbol{b}) + \lambda \|\boldsymbol{b}\|_\alpha^\alpha}_{=:f} \right\}. \tag{5.3}$$

For $\alpha = 0$ and $\alpha = 1$ we have the best-subset case and the Lasso case respectively, for $\alpha = 2$, we have Ridge regression. Therefore the bridge regression parameter α

is considered to be in the interval $[0, 2]$ in practice.

It should be noted, that for $\alpha > 1$ the penalty does not fulfil Assumption 5 in Section 3.2, therefore we cannot expect sparse solutions at this values of α.

5.2. Elastic net

The next bridge penalty we discuss, is the elastic net [Zou and Hastie, 2005]. The basic (naive) version of this regression method is:

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{b}} \left\{ \mathrm{RSS}(\boldsymbol{b}) + \lambda_1 \|\boldsymbol{b}\|_1 + \lambda_2 \|\boldsymbol{b}\|_2^2 \right\}, \tag{5.4}$$

with $\lambda_1, \lambda_2 \geq 0$. Equation (5.4) is often written in an equivalent form:

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{b}} \left\{ \mathrm{RSS}(\boldsymbol{b}) + \lambda \left[(1-\alpha) \|\boldsymbol{b}\|_1 + \alpha \|\boldsymbol{b}\|_2^2 \right] \right\}, \tag{5.5}$$

where $\lambda \geq 0$ and $\alpha \in [0, 1]$. The term in the square brackets is the elastic net penalty. It is clear, that at $\alpha = 0$ and $\alpha = 1$, problem (5.5) degenerates to the Lasso and to Ridge regression, respectively.

Zou and Hastie [2005] provided a result on the grouping effect as well:

Theorem 5.1. *In addition to the global assumptions given in Section 2.1, let $\hat{\boldsymbol{\beta}}$ be the estimator in (5.4) with $\lambda_2 > 0$. If two components $\hat{\beta}_i$ and $\hat{\beta}_j$ of $\hat{\boldsymbol{\beta}}$ fulfil $\hat{\beta}_i \hat{\beta}_j > 0$ then:*

$$\frac{1}{\|\boldsymbol{y}\|_1} \left| \hat{\beta}_i - \hat{\beta}_j \right| \leq \frac{1}{\lambda_2} \sqrt{2\left(1 - (\boldsymbol{x}^i)^\top (\boldsymbol{x}^j)\right)} \tag{5.6}$$

Zou and Hastie [2005] also proposed a rescaled version of the above defined $\boldsymbol{\beta}^* = (1 + \lambda_2)\, \hat{\boldsymbol{\beta}}$ with $\hat{\boldsymbol{\beta}}$ from (5.4), which may improve results in practice.

Because of the Lasso penalty in minimization problem (5.5), except for $\alpha = 1$, Assumption 5 in Section 3.2 is fulfilled. Thus we can expect sparse solutions in regression problems using the elastic net.

The form of the elastic net penalty for different values of α can be seen in Figure 5.1.

5.3. Generalized elastic net

Friedman [2008] proposed an alternative to the penalties of the power family with

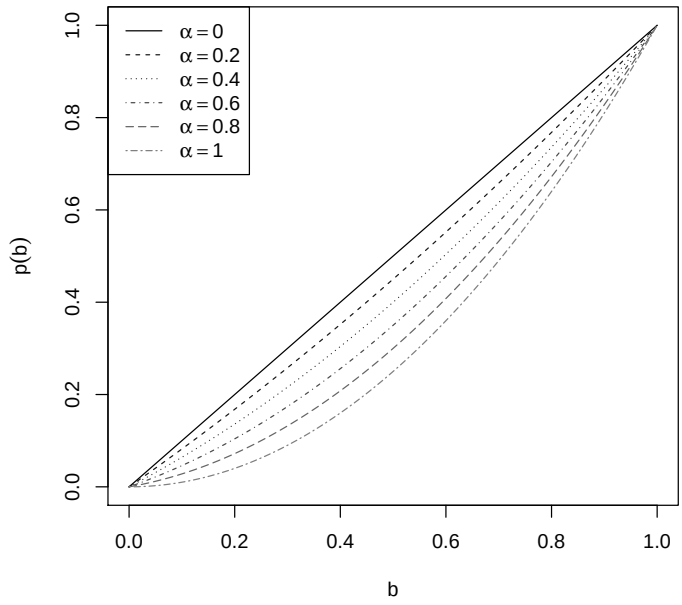


Figure 5.1.: Elastic net penalties for different values of α

parameter $\alpha \in [0, 1]$. The generalized elastic net penalty is defined as

$$p_\alpha(\boldsymbol{b}) := \sum_{j=1}^{p} \ln\left((1-\alpha)\,|b_j| + \alpha\right) \tag{5.7}$$

with $\alpha \in (0, 1)$.

Rescaling and centering (5.7) to

$$p_\alpha(\boldsymbol{b}) := \sum_{j=1}^{p} \left[\left(-\frac{1}{\ln(\alpha)}\right) \ln\left((1-\alpha)\,|b_j| + \alpha\right) + 1\right] \tag{5.8}$$

yields an equivalent minimization problem. By using l'Hôpital's rule we can see, that for $\alpha \longrightarrow 0+$ and $b = 0$ the square brackets in (5.8) converge to 0. On the other hand, if $\alpha \longrightarrow 0+$ and $b \neq 0$, they converge to 1. Therefore $p_\alpha(\boldsymbol{b})$ converges pointwisely to $\|\boldsymbol{b}\|_0$ for $\alpha \longrightarrow 0+$. It can be shown that for $\alpha \longrightarrow 1-$, $p_\alpha(\boldsymbol{b})$ converges pointwisely to $\|\boldsymbol{b}\|_1$. We have shown, that the generalized elastic net serves as a bridge between the best-subset selection and the Lasso framework.

The penalties of the generalized elastic net have a similar shape compared to

penalties of the power family as can be seen in Figure 5.2.

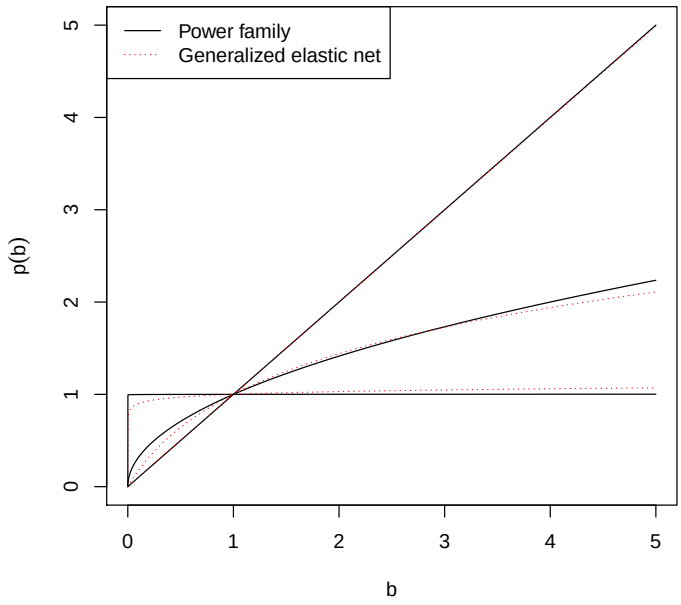


Figure 5.2.: Power family penalties and generalized elastic net penalties
The black solid lines show the penalties of the power family with $\alpha = 1$, $\alpha = 0.5$, and $\alpha = 0.001$. The red dotted lines show the (rescaled and centered) generalized elastic net penalities with $\alpha = 0.999$, $\alpha = \frac{1}{10^{10}}$, and $\alpha = 0.5$. It can be seen, that power family penalties can be approximated by generalized elastic net penalties.

5.4. Minimax concave penalty

In this section the minimax concave penalty (MCP) according to Zhang [2010] will be derived.

Definition 5.2. A given penalty $p(b)$ is assumed to be differentiable on the interval $(0, \infty)$. The maximum concavity of $p(\cdot)$ for a given λ is defined as

$$\kappa(p) := \sup_{0<b_1<b_2} \left[-\frac{\dot{p}(b_2) - \dot{p}(b_1)}{b_2 - b_1} \right] \tag{5.9}$$

Non-convexity of a penalty can lead to computational problems and unstable results in our minimization framework. As an example one can look at our example penalty in Section 3.2 and its unstable behavior (bottom right plot in Figure

3.2). The goal is now to find a penalty, which minimizes the maximum concavity $\kappa(p)$, and leads to sparse and nearly unbiased solutions with a threshold $\lambda\gamma$. In other words, $\kappa(p) = \sup_{0<b_1<b_2}\left[-\frac{\dot{p}(b_2)-\dot{p}(b_1)}{b_2-b_1}\right]$ has to be minimized subject to $\dot{p}\,(0+) = 1$ and $\dot{p}\,(b) = 0$ for $b \geq \lambda\gamma$ (according to Assumption 5 and Assumption 6 in Section 3.2). By setting $b_1 \longrightarrow 0$ and $b_2 = \lambda\gamma$, it can easily be seen that for every penalty fulfilling these restrictions, we have

$$\kappa(p) \geq -\frac{\dot{p}\,(\lambda\gamma) - \dot{p}\,(0+)}{\lambda\gamma - (0+)} = \frac{\dot{p}\,(0+)}{\lambda\gamma} = \frac{1}{\lambda\gamma}. \tag{5.10}$$

It is clear, that $\dot{p}\,(b) := \left(1 - \frac{b}{\lambda\gamma}\right)_+$ fulfils both of the above constraints. Three cases should be considered:

1. $b_1, b_2 \geq \lambda\gamma$
   It follows that, $-\frac{\dot{p}(b_2)-\dot{p}(b_1)}{b_2-b_1} = 0$.

2. $b_1 \in (0, \lambda\gamma), b_2 \geq \lambda\gamma$
   This implies that

$$\begin{aligned} -\frac{\dot{p}\,(b_2) - \dot{p}\,(b_1)}{b_2 - b_1} &= -\frac{\left(1 - \frac{b_2}{\lambda\gamma}\right)_+ - \left(1 - \frac{b_1}{\lambda\gamma}\right)_+}{b_2 - b_1} \\ &= \frac{1 - \frac{b_1}{\lambda\gamma}}{b_2 - b_1} \\ &= \frac{1}{\lambda\gamma}\frac{\lambda\gamma - b_1}{b_2 - b_1}. \end{aligned}$$

   Because $\frac{\lambda\gamma-b_1}{b_2-b_1} \leq 1$, we have $\frac{\dot{p}(b_2)-\dot{p}(b_1)}{b_2-b_1} \leq \frac{1}{\lambda\gamma}$.

3. $b_1, b_2 \in (0, \lambda\gamma)$
   We immediately have $-\frac{\dot{p}(b_2)-\dot{p}(b_1)}{b_2-b_1} = \frac{1}{\lambda\gamma}\frac{b_2-b_1}{b_2-b_1} = \frac{1}{\lambda\gamma}$.

With (5.10) it follows, that

$$p\,(b) := \int_0^b \left(1 - \frac{x}{\lambda\gamma}\right)_+ dx \tag{5.11}$$

minimizes the maximum concavity in the set of all admissible penalties. Hence (5.11) is called the minimax concave penalty. By definition, this penality leads to sparse solutions and is nearly unbiased.

Penalties for different values of α are plotted in Figure 5.3.

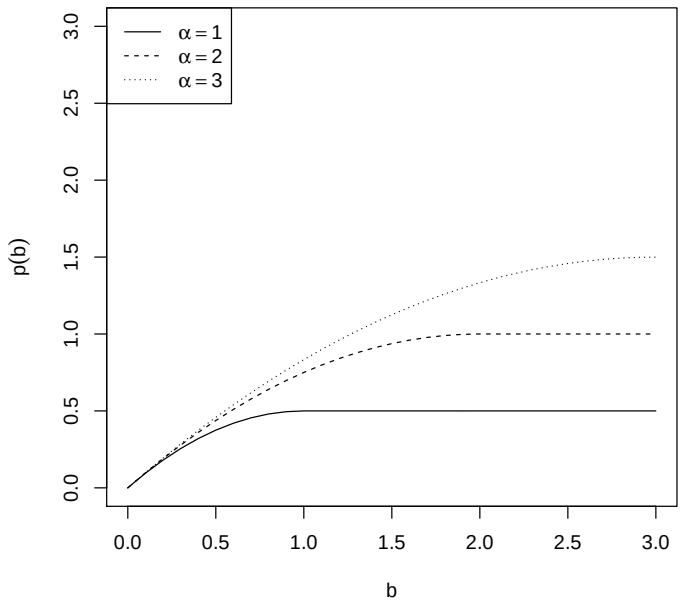


Figure 5.3.: Minimax concave penalties

6. Model calibration

6.1. Generalized degrees of freedom

Degrees of freedom of linear estimators

We consider the fitting procedure $\hat{\boldsymbol{\mu}} : \mathbb{R}^n \longrightarrow \mathbb{R}^n : \boldsymbol{y} \mapsto \boldsymbol{X}\hat{\boldsymbol{\beta}}(\boldsymbol{y})$ with components $\hat{\mu}_i(\boldsymbol{y})$. At first we assume that $\hat{\boldsymbol{\mu}}(\cdot)$ is the OLS estimator: $\hat{\boldsymbol{\mu}}(\boldsymbol{y}) = \boldsymbol{H}\boldsymbol{y}$ with $\boldsymbol{H} := \boldsymbol{X}\left(\boldsymbol{X}^\top\boldsymbol{X}\right)^{-1}\boldsymbol{X}^\top$. The degrees of freedom in the full rank OLS case are commonly defined as

$$\mathrm{df} := \mathrm{tr}[\boldsymbol{H}], \tag{6.1}$$

which is equal to the rank of $\boldsymbol{X}$. This holds true for the high-dimensional OLS case, where $\boldsymbol{H} := \boldsymbol{U}\boldsymbol{I}_{r,n}\boldsymbol{U}^\top$ according to the results in Section 2.2. Again the trace of $\boldsymbol{H}$ is the rank of $\boldsymbol{X}$.

We can generalize the definition of the degrees of freedom to all linear estimators of $\boldsymbol{\mu}$. In Ridge regression, with $\boldsymbol{H} := \boldsymbol{U}\boldsymbol{D}\left(\boldsymbol{D}^2 + \lambda\boldsymbol{I}_p\right)^{-1}\boldsymbol{D}^T\boldsymbol{U}^T$, the degrees of freedom are

$$\begin{aligned}\mathrm{df} &= \mathrm{tr}[\boldsymbol{H}] \\ &= \sum_{i=1}^{r} \frac{d_i^2}{d_i^2 + \lambda}.\end{aligned} \tag{6.2}$$

Degrees of freedom of almost differentiable estimators

For linear estimators, it is clear that $\boldsymbol{H}$ is the Jacobi matrix $\left[\frac{\partial\hat{\boldsymbol{\mu}}(\boldsymbol{y})}{\partial\boldsymbol{y}}\right]$. Thus the trace of $\boldsymbol{H}$ is

$$\mathrm{df} = \sum_{i=1}^{n} \frac{\partial\hat{\mu}_i(\boldsymbol{y})}{\partial y_i}, \tag{6.3}$$

which is the divergence of the estimator $\mathrm{div}\,\hat{\boldsymbol{\mu}}(\boldsymbol{y})$.

The next definition of almost differentiability and the following lemma are taken from Stein [1981] and play an important role in the development of a generalized definition of degrees of freedom.

Definition 6.1. $h : \mathbb{R}^n \longrightarrow \mathbb{R}$ is called almost differentiable, if there exists a function $\boldsymbol{\nabla} h : \mathbb{R}^n \longrightarrow \mathbb{R}^n$, such that for all $\boldsymbol{s} \in \mathbb{R}^n$,

$$h(\boldsymbol{z}+\boldsymbol{s}) - h(\boldsymbol{z}) = \int_0^1 \boldsymbol{s}^\top \boldsymbol{\nabla} h(\boldsymbol{z}+t\boldsymbol{s})\, dt \tag{6.4}$$

The function $\boldsymbol{\nabla} h$ can be identified with the vector of the partial derivations of h .

For the next lemma, with $\boldsymbol{y} = \boldsymbol{\mu} + \boldsymbol{\epsilon}$ defined in accordance with Section 2.1, we define $\boldsymbol{z} = \frac{1}{\sigma}\boldsymbol{y}$. Then $\boldsymbol{z} \sim N\left(\frac{1}{\sigma}\boldsymbol{\mu}, \boldsymbol{I}\right)$.

Lemma 6.2 (Stein's Lemma)**.** *If $h : \mathbb{R}^n \longrightarrow \mathbb{R}$ is an almost differentiable function, $\boldsymbol{z} \sim N\left(\frac{1}{\sigma}\boldsymbol{\mu}, \boldsymbol{I}\right)$, and $\mathbb{E}_{\boldsymbol{\mu}} \|\boldsymbol{\nabla} h(\boldsymbol{z})\| < \infty$, then*

$$\mathbb{E}_{\boldsymbol{\mu}} [\boldsymbol{\nabla} h(\boldsymbol{z})] = \mathbb{E}_{\boldsymbol{\mu}} \left[\left(\boldsymbol{z} - \frac{1}{\sigma}\boldsymbol{\mu} \right) h(\boldsymbol{z}) \right] \tag{6.5}$$

Now assume that $\hat{\boldsymbol{\mu}}$ is almost differentiable (by that we mean that every $\hat{\mu}_i$ is almost differentiable). It follows from Stein's Lemma, that

$$\begin{aligned}
\mathbb{E}_{\boldsymbol{\mu}} [\boldsymbol{\nabla}_{\boldsymbol{y}} \hat{\mu}_i(\boldsymbol{y})] &= \frac{1}{\sigma} \mathbb{E}_{\boldsymbol{\mu}} [\boldsymbol{\nabla}_{\boldsymbol{z}} \hat{\mu}_i(\sigma\boldsymbol{z})] \\
&= \frac{1}{\sigma} \mathbb{E}_{\boldsymbol{\mu}} [\boldsymbol{\nabla}_{\boldsymbol{z}} \tilde{\mu}_i(\boldsymbol{z})] \\
&= \frac{1}{\sigma} \mathbb{E}_{\boldsymbol{\mu}} \left[\left(\boldsymbol{z} - \frac{1}{\sigma}\boldsymbol{\mu} \right) \tilde{\mu}_i(\boldsymbol{z}) \right] \\
&= \frac{1}{\sigma^2} \mathbb{E}_{\boldsymbol{\mu}} [(\boldsymbol{y} - \boldsymbol{\mu})\, \hat{\mu}_i(\boldsymbol{y})],
\end{aligned} \tag{6.6}$$

for every $i \in 1, \ldots, n$, where $\tilde{\mu}_i(\boldsymbol{z}) := \hat{\mu}_i(\sigma\boldsymbol{z})$.

From (6.6) it is clear that $\mathbb{E}_{\boldsymbol{\mu}} [\boldsymbol{\nabla}_{\boldsymbol{y}} \hat{\mu}_i(\boldsymbol{y})]$ is the $i-$th column of the covariance matrix of $\boldsymbol{y}$ and $\hat{\boldsymbol{\mu}}(\boldsymbol{y})$. Therefore we have the following result:

$$\begin{aligned}
\sum_{i=1}^n \frac{\partial \hat{\mu}_i(\boldsymbol{y})}{\partial y_i} &= \sum_{i=1}^n \frac{1}{\sigma^2} \mathbb{E}_{\boldsymbol{\mu}} [(y_i - \mu_i)\, \hat{\mu}_i(\boldsymbol{y})] \\
&= \frac{1}{\sigma^2} \operatorname{tr} Cov(\boldsymbol{y}, \hat{\boldsymbol{\mu}}(\boldsymbol{y})) \\
&= \sum_{i=1}^n \frac{Cov(y_i, \hat{\mu}_i(\boldsymbol{y}))}{\sigma^2}.
\end{aligned} \tag{6.7}$$

Equation (6.7) leads to the definition of the generalized degrees of freedom (see Ye, 1998) as

$$\mathrm{df} :== \sum_{i=1}^{n} \frac{\partial \hat{\mu}_i(\boldsymbol{y})}{\partial y_i} = \sum_{i=1}^{n} \frac{Cov(y_i, \hat{\mu}_i(\boldsymbol{y}))}{\sigma^2}. \tag{6.8}$$

6.2. Estimation of the prediction error

To get a good calibrated model in the sense of prediction property 1 in Chapter 3, the idea of minimizing the expected in-sample prediction error

$$\mathbb{E}_{\boldsymbol{y}} \mathrm{Err}_{in}(\boldsymbol{y}) = \mathbb{E}_{\boldsymbol{y}} \mathbb{E}_{\boldsymbol{y}_0} \left[\|\boldsymbol{y}_0 - \hat{\boldsymbol{\mu}}(\boldsymbol{y})\|_2^2 \mid \boldsymbol{y} \right] \tag{6.9}$$

seems natural.

The expected in-sample error (6.9) can be written as

$$\mathbb{E}_{\boldsymbol{y}} \mathrm{Err}_{in}(\boldsymbol{y}) \quad = \quad \mathbb{E}_{\boldsymbol{y}} \left[\|\boldsymbol{y} - \hat{\boldsymbol{\mu}}(\boldsymbol{y})\| + 2\sigma^2 \,\mathrm{df} \right]. \tag{6.10}$$

Following Hirose et al. [2011], where also the proof of equation (6.10) can be found, this motivates a C_p-type estimator for (6.9):

$$C_p := \|\boldsymbol{y} - \hat{\boldsymbol{\mu}}(\boldsymbol{y})\| + 2\sigma^2 \,\mathrm{df}\,. \tag{6.11}$$

This estimator written as (6.11) assumes, that the term $2\sigma^2 \,\mathrm{df}$ is known. This is not the case in practice, where we either need an estimation for $Cov(y_i, \hat{\mu}_i(\boldsymbol{y}))$, or estimations for $\frac{\partial \hat{\mu}_i(\boldsymbol{y})}{\partial y_i}$, and σ^2, respectively.

6.3. Cross-validation

One of the most important concepts in the area of model selection is cross-validation (CV). It combines intuitive plausibility with wide applicability. Its aim is to give a reasonable estimate of

$$\frac{1}{n} \mathbb{E}_{\boldsymbol{y}} \mathrm{Err}_{in}(\boldsymbol{y}). \tag{6.12}$$

Validation

In statistical practice, model validation techniques often are applied to the same data, on which the model depends. Selecting models on these techniques would likely cause overfitting. To overcome this problem, it is recommended to split the data in two independent parts. The first one is called the training set, on which

the model will be estimated. The resulting model will be tested on the second part of the data that is the test set.

Let $I \subset \{1, \ldots, n\}$ and $I^- := \{1, \ldots, n\} \setminus I$ be the training set and the test set, respectively. The vector $\boldsymbol{y}^-$ is of length $|I^-|$, and contains the y_i with $i \in I^-$. The vector $\boldsymbol{\mu}^-$ contains the entries of $\boldsymbol{\mu}$ with indices $i \in I^-$. $\hat{\boldsymbol{\mu}}_{\boldsymbol{\lambda}}^-$ is an estimator of $\boldsymbol{\mu}^-$, based on the training set and the model parameter λ. We define the estimator of (6.12) by

$$MSE_{TEST} := \frac{1}{|I^-|} \left\| \boldsymbol{y}^- - \hat{\boldsymbol{\mu}}_{\boldsymbol{\lambda}}^- \right\|_2^2,$$

which is called the mean squared error.

k-fold cross-validation

In general collecting data is often expensive in time and money. Splitting the data can be seen as a drawback of the validation method. Only $n - |I^-|$ are going to be used for estimation. Also training data and test data have to be representative for the total population. If by chance the data splitting leads to unrepresentative test data and/or training data, this can result in bad model estimates.

These drawbacks can be overcome by repeating this validation procedure on the same data several times. More precisely, the data set is split in k parts of (almost) equal size. Then every of these k parts is a test set for a model, estimated by the remaining $k - 1$ parts. In obvious generalization of the notation from above (where "$-(i)$" means, that the ith part is the respective test set), we define an estimator

$$\begin{aligned} MSE_{CV} &:= \frac{1}{|n|} \sum_{i=1}^{k} MSE_{TEST}^{-(i)} \cdot \left| I^{-(i)} \right| \\ &= \frac{1}{|n|} \sum_{i=1}^{k} \left\| \boldsymbol{y}^{-(\boldsymbol{i})} - \hat{\boldsymbol{\mu}}_{\boldsymbol{\lambda}}^{-(\boldsymbol{i})} \right\|_2^2. \end{aligned} \tag{6.13}$$

A special case of the k-fold cross-validation is the leave-one-out cross-validation, where $k = n$.

Double cross-validation

CV can be used for both, for finding the optimal parameters for our penalized regression models (calibration), and for estimating the prediction error of the final model. One may want to use CV for both. Like above it would be necessary

to split the data successively into test sets and training sets for the CV, which estimates the prediction error of the final model. The training sets itself are split in the same manner into new test sets and training sets to perform a CV for model calibration. The result is a procedure with two nested CV loops. We call that double cross-validation. In accordance to Varmuza and Filzmoser [2009], we call the training set of the outer CV loop the calibration set, and the test set of the inner CV loop the validation set.

If estimator (6.13) results from the inner CV loop (applied on the calibration set), we denote it by MSE_{CAL}, instead of MSE_{CV}.

Varmuza and Filzmoser [2009] suggest to repeat the outer CV loop several times. They called the procedure repeated double CV (rdCV).

6.4. Generalized cross-validation

Generalized cross-validation (GCV) is a model selection method developed by Golub et al. [1979]. GCV was initially applied in Ridge regression. A generalization for possibly non-linear estimators with model parameter γ is suggested by Ye [1998] as

$$\mathrm{GCV}\left(\hat{\boldsymbol{\mu}}_{\boldsymbol{\gamma}}\right) = \frac{\|\boldsymbol{y} - \hat{\boldsymbol{\mu}}_{\boldsymbol{\gamma}}(\boldsymbol{y})\|_2^2}{\left(n - \mathrm{df}\left(\hat{\boldsymbol{\mu}}_{\boldsymbol{\gamma}}\right)\right)^2}. \tag{6.14}$$

The minimizing γ in (6.14) would then be chosen as the model parameter.

This approach has the advantage, that if an estimator for $\sum_{i=1}^{n} \frac{\partial \hat{\mu}_i(\boldsymbol{y})}{\partial y_i}$ exists, it is not necessary to estimate the unknown σ^2 in order to select a reasonable model.

7. Optimization methods

In this chapter, we describe common optimization methods for the above defined minimization problems.

7.1. Pathwise coordinate descent

Recently, Friedman et al. [2007] introduced coordinate descent methods in the field of penalized regression with convex penalties. These methods have in common, that the parameters are estimated one at a time. Usually parameters are updated cyclically. This can be described as follows:

1. Begin with a starting vector $\boldsymbol{b}_0$ (which commonly is set equal to $\mathbf{0}$)

2. k runs through $1, \ldots, p, p+1, \ldots, 2p, 2p+1, \ldots$

   a) set $\boldsymbol{b}$ equal to $\boldsymbol{b}_{k-1}$, the solution from the previous iteration

   b) set $i = k \bmod p$, define $\boldsymbol{X}^{-(i)}$ as the matrix $\boldsymbol{X}$ without the ith column $\boldsymbol{x}_i$, and $\boldsymbol{b}^{-(i)}$ as the vector $\boldsymbol{b}$ without the ith entry. Now solve the minimization problem $\left\| \boldsymbol{y} - \boldsymbol{X}^{-(i)}\boldsymbol{b}^{-(i)} - \boldsymbol{x}_i b_i^* \right\|_2^2 + \lambda p(b_i^*; \boldsymbol{b}^{-(i)})$ for variable b_i^*, where $\boldsymbol{b}^{-(i)}$ is considered to be fixed

   c) set $\boldsymbol{b}_k$ equal to $\boldsymbol{b}_{k-1}$ exept for the j-th component, which is set equal to b_i^*

This algorithm works fast when the one-variable-minimization in 2.(b) can be solved easily.

The algorithm can be improved by predefining a monotonic sequence of the regularization parameter $\lambda_1, \ldots, \lambda_m$. Then the solution of the penalized regression problem with regularization parameter λ_j can be used as the starting vector for solving the problem with the parameter λ_{j+1}.

7.2. Generalized path seeking

The generalized path seeking algorithm was introduced by Friedman [2008] and is a fast method to solve penalized regression problems, if the penalty fulfills $\frac{\partial p(\boldsymbol{b})}{\partial |b_i|} > 0$.

For $t \geq 0$ we consider the penalized regression problem:

$$\hat{\boldsymbol{\beta}}(t) := \arg\min_{\boldsymbol{b}} \{\mathrm{RSS}(\boldsymbol{b})\} \ \ s.t.\ p(\boldsymbol{b}) \leq t.$$

We set $\boldsymbol{b}(0) = 0$ and $0 < \Delta t \ll 1$, and calculate $\boldsymbol{b}(t + \Delta t)$ from $\boldsymbol{b}(t)$ by altering only one coordiate i:

$$b_j(t + \Delta t) = \begin{cases} b_j(t) & j = i \\ b_j(t) + s \cdot \Delta t & j \neq i \end{cases},$$

where s is either -1 or 1. For every t, $\boldsymbol{b}(t)$ can be seen as an approximation of $\hat{\boldsymbol{\beta}}(t)$.

Two questions arise in each iteration:

1. Which coordinate should be altered?

2. Is s either -1 or 1?

By a relaxation of the constraint t to $t + \Delta t$ we want $RSS(\boldsymbol{b})$ to decrease as much as possible, while simultaneously $p(\boldsymbol{b})$ should increase only a little. In other words, the absolute fraction of "ultility" and "costs"

$$\left| \frac{-\frac{\partial RSS(b_1(t),\ldots,b_p(t))}{\partial b_i}}{\frac{\partial p(|b_1(t)|,\ldots,|b_p(t)|)}{\partial b_i}} \right| = \left| \underbrace{\frac{-\frac{\partial RSS(b_1(t),\ldots,b_p(t))}{\partial b_i}}{\frac{\partial p(|b_1(t)|,\ldots,|b_p(t)|)}{\partial |b_i|}}}_{=:k_i(t)} \cdot \mathrm{sign}(b_i) \right|$$

should be as big as possible.

The algorithm searches for indices i with $k_i(t)\,\mathrm{sign}(b_i(t)) < 0$ and collects them in the set S. Then with $j = \arg\max_{i \in S} |k_i(t)|$, the variable b_j is altered towards zero, say

$$b_j(t + \Delta t) = \begin{cases} b_j(t) + \Delta t & b_i(t) < 0 \\ b_j(t) - \Delta t & b_i(t) > 0 \end{cases}.$$

If S is empty, we choose the coordinate $j = \arg\max_i |k_i(t)|$ and

$$b_j(t + \Delta t) = b_j(t) + \text{sign}\,(b_i(t))\,\Delta t.$$

This procedure will be repeated until all $k_j(t)$ are equal to zero.

Friedman described the algorithm as follows:

```
1  Set  t = 0  and  b_i(0) = 0, ∀i ∈ {1,...,p}
2  do {
3      calculate  k(t)
4      S := {i : k_i(t) sign(b_i(t)) < 0}
5      if  (S = ∅)  {
6        j = arg max_i |k_i(t)|
7      }  else  {
8        j = arg max_{i∈S} |k_i(t)|
9      }
10     b_j(t + Δt) = b_j(t) + sign(k_i(t)) · Δt
11     b_i(t + Δt) = b_i(t) ,  i ∈ {1,...,j − 1, j + 1,...,p}
12     t ← t + Δt
13 }  while  (k(t) ≠ 0)
```

7.3. R packages

sparsenet

SparseNet [Mazumder et al., 2011] is a fast variant of the pathwise coordinate descent optimization in Section 7.1, which works also for nonconvex penalties. Mazumder et al. [2011] focused on regression problems with rescaled version of the minimax concave penalty in Section 5.4. An R implementation of this particular penalty is provided in the sparsenet package version 1.1 [Mazumder et al., 2013].

glmnet

Another implementation of pathwise coordinate descent is the R package glmnet version 1.9-3 [Friedman et al., 2010]. Its same-named function uses the elastic net penalty and calculates a path with varying λ and fixed α.

msgps

An R implementation of the generalized path seeking algorithm is the package msgps version 1.3 [Hirose, 2012]. Simultaneously to the coefficient path, corresponding generalized degrees of freedom estimates are calculated.

penalized

For fused Lasso regression, coordinate descent methods can fail and R implementations with the generalized path seeking algorithm do not exist. Therefore we use the R package penalized version 0.9-42 [Goeman, 2012]. This package optimizes with a combination of gradient ascent and the Newton-Raphson algorithm.

8. Method comparison

8.1. Datasets

Polycyclic Aromatic Compound (PAC)

The first dataset comes from 209 polycyclic aromatic compounds. The goal is to describe quantitative structure-property relationships (QSPR) by a linear model. Given are 2688 descriptors (after exclusion of almost constant variables), that are structural features of molecular 3D structures ($\boldsymbol{X}$ matrix). The dependend variable that should be predicted, consists of the gas chromatographic retention index ($\boldsymbol{y}$ vector). This dataset was already used by Varmuza et al. [2013] to describe and evaluate variable selection methods.

Near Infrared Spectroscopy (NIR)

The second dataset contains 235 variables (transformed NIR absorbance values) of 166 alcoholic fermentation mashes of rye, wheat and corn. These variables are ordered by increasing wavelength. Two dependend variables are available to be taken into the model separately. This dataset was further described by Liebmann et al. [2009].

8.2. Model estimation procedures

We examine two model estimation procedures, based on sparsenet. The calibration is done by cross-validation. Looking at the MSE_{CAL} values for different values of α and λ, we either choose the pair (λ^*, α^*) with

$$(\lambda^*, \alpha^*) := \underset{(\lambda,\alpha)}{\arg\min}\, MSE_{CAL}\,(\lambda, \alpha) \tag{8.1}$$

("snet min"), or the model with the least generalized degrees of freedom of the set of all models (λ, α) with

$$MSE_{CAL}(\lambda, \alpha) < MSE_{CAL}(\lambda^*, \alpha^*) + \sigma_{CAL},$$

where (λ^*, α^*) is the solution of (8.1), and σ_{CAL} is the standard deviation of the squared errors, estimated on the validation set ("snet 1se").

With the functions, provided in package glmnet we evaluate Lasso regression, Ridge regression, and regression with the elastic net penalty. For Lasso regression and Ridge regression, we consider both model calibration procedures, which we use in the sparsenet based model estimation ("lasso min" or "lasso 1se", and "ridge min" and "ridge 1se"). For regression with the elastic net penalty, the cross-validation and model selection analogue to (8.1) is performed ("elastic net").

The elastic net penalty as well as the generalized elastic net penalty are included in the msgps package. Because for each parameter estimate the corresponding degrees of freedom are provided, we evaluate these penalized regression methods by using the model selection criteria (6.11) and (6.14) ("enet cp" or "enet gcv", and "genet cp" or "genet gcv").

For NIR spectroscopy data, results from the fused Lasso (which are implemented in the R package penalized) are presented ("fused").

We compare these penalized regression methods with PLS regression, where all variables are included ("pls"). The number of latent variables are obtained by cross-validation (minimum MSE_{CV}). The necessary computation are done, using the R package pls version 2.3-0.

8.3. Results

For the evaluation of each method, repeated cross-validation was done with $r = 10$ repetitions (except for fused Lasso with only one repetition) and $k = 10$ splits of the calibration set for cross-validation. If for an estimation method, calibration was done by cross-validation too, the training set is split in 10 parts as well.

PAC data

Quantiles of the absolute cross-validation errors are listed in Table 8.1.

The performance of "enet cp" and "enet gcv" on this data is bad. It seems to be, that the generalized path seeking algorithm converges at a suboptimal point

in this setting. The methods “elasticnet” and “lasso min” perform well, and the method “lasso 1se” performs best. All of these three methods have small standard deviation estimates, but “lasso 1se” stands out, because the largest absolute error is 28.8, which is less than a half compared to all other competing methods. The sparsenet methods perform at a level with “elasticnet”, but produce a few severe outliers. The “pls” method is beaten by “lasso 1se”, because of higher outliers, but performs equally well comparing to “elasticnet” and “lasso min”.

Boxplots for visualization can be found in Figure 8.1(a). The boxplots of “enet cp” and “enet gcv” are left out, because they would inflate the y-scale.

	0%	25%	50%	75%	90%	95%	99%	100%	RMSEP
elasticnet	0.00	1.62	3.59	6.81	10.98	14.41	21.48	97.33	7.73
enet cp	0.01	26.91	61.68	85.13	106.47	122.58	143.44	149.57	69.82
enet gcv	0.01	27.39	62.39	85.21	106.05	122.39	143.12	146.56	69.82
genet cp	0.00	4.23	8.52	15.88	27.49	32.79	45.68	83.07	15.77
genet gcv	0.00	2.06	5.03	10.42	17.96	22.66	37.20	371.62	13.79
lasso 1se	0.00	1.95	4.24	7.82	12.38	15.51	21.47	28.88	7.45
lasso min	0.00	1.66	3.71	7.03	11.23	14.88	22.53	121.73	8.43
pls	0.00	1.53	3.74	7.17	12.87	16.62	28.30	88.49	9.30
ridge 1se	0.01	2.66	6.15	12.16	20.01	28.02	50.05	318.72	17.18
ridge min	0.00	2.17	5.21	9.48	14.71	18.01	31.62	342.33	12.34
snet 1se	0.01	1.73	3.79	7.28	12.79	15.62	24.21	1186.71	27.86
snet min	0.00	1.73	3.85	7.00	11.89	15.41	24.54	238.01	12.58

Table 8.1.: Quantiles and RMSEP of absolute errors (PAC data)

NIR data (glucose)

In this case, the performance of our estimation methods is homogeneous, except for the ridge regression methods “fused”, “ridge 1se” and “ridge min” (see Table 8.2 and Figure 8.2(a)).

NIR data (ethanol)

Again “enet cp“ performs not as good as the competing methods. The RMSEP is 9.21 and its maximum absolute error is 20.52. These values differ highly from the rest (see Table 8.3 and Figure 8.3(a)). Except “enet cp”, “fused”, and the ridge regression methods, all of the methods perform exceptionally good with RMSEPs between 1.44 and 2.79.

	0%	25%	50%	75%	90%	95%	99%	100%	RMSEP
elasticnet	0.00	1.76	3.81	6.72	8.92	9.93	14.11	28.86	5.71
enet cp	0.00	2.18	4.81	8.56	11.68	13.68	17.68	25.10	7.17
enet gcv	0.00	1.69	4.27	6.66	9.24	10.58	14.88	29.19	5.93
fused	0.07	3.73	5.94	14.51	19.72	21.47	28.59	31.73	11.41
genet cp	0.00	1.95	4.24	7.56	10.54	12.22	16.78	33.61	6.47
genet gcv	0.00	1.94	3.71	6.16	8.56	10.38	14.64	33.95	5.44
lasso 1se	0.00	1.94	4.06	7.09	9.55	10.65	15.46	26.42	6.06
lasso min	0.01	1.82	3.64	6.62	8.83	9.76	14.30	25.35	5.59
pls	0.00	1.67	3.35	6.15	9.16	10.46	12.71	22.34	5.32
ridge 1se	0.00	2.64	5.91	12.63	16.88	19.20	23.79	27.29	10.04
ridge min	0.00	2.49	6.07	11.99	16.14	18.14	22.33	25.72	9.55
snet 1se	0.00	1.87	3.80	6.66	9.69	11.03	14.23	28.76	5.81
snet min	0.00	1.78	3.76	6.43	9.22	10.73	13.71	21.98	5.54

Table 8.2.: Quantiles and RMSEP of absolute errors (NIR data [glucose])

Summary

To interpret these results correctly, we define a measure analogously to the idea of the coefficient of determination R^2. Evaluation of a model estimation procedure by repeated cross-validation results in rn estimates $\hat{y}_1, \ldots, \hat{y}_{rn}$, where r is the number of repetitions. Let y_{j_i} be the (true) entry of the vector $\boldsymbol{y}$, corresponding to the estimate $\hat{y}_i$. Then the predicted residual error sum of squares is defined as

$$PRESS := \sum_{i=1}^{rn} (y_{j_i} - \hat{y}_i)^2 = \sum_{j=1}^{r} n \cdot MSE_{CV}^{(j)},$$

where $MSE_{CV}^{(j)}$ is estimator (6.13) applied on the jth repetition of the r CV runs. We further define the predicted sum of absolute residual errors

$$PSARE := \sum_{i=1}^{rn} |y_{j_i} - \hat{y}_i| ,$$

and the total sum of squares and the sum of absolute deviations of the original values

$$TSS := r \sum_{i=1}^{n} (y_i - \bar{y})^2$$

and

$$TSA := r \sum_{i=1}^{n} |y_i - \tilde{y}| ,$$

	0%	25%	50%	75%	90%	95%	99%	100%	RMSEP
elasticnet	0.00	0.44	0.98	1.65	2.25	3.18	5.08	5.56	1.56
enet cp	0.00	5.25	8.06	11.23	14.08	15.53	18.17	20.52	9.21
enet gcv	0.00	0.39	0.83	1.47	2.42	3.27	4.53	7.44	1.49
fused	0.01	2.25	3.96	7.58	12.95	15.62	21.18	22.80	7.47
genet cp	0.00	0.98	1.93	3.03	4.60	5.66	7.39	9.64	2.79
genet gcv	0.00	0.42	0.88	1.47	2.33	3.03	4.60	5.54	1.44
lasso 1se	0.00	0.60	1.23	1.91	2.64	3.51	5.53	6.55	1.77
lasso min	0.00	0.50	1.07	1.74	2.50	3.38	5.02	6.13	1.64
pls	0.00	0.38	0.83	1.51	2.28	2.93	4.14	5.50	1.40
ridge 1se	0.00	2.43	4.11	6.65	9.20	11.39	14.20	17.26	5.81
ridge min	0.00	2.18	3.93	6.15	8.75	11.06	13.93	16.31	5.55
snet 1se	0.00	0.49	1.11	1.90	2.99	3.85	5.24	7.70	1.81
snet min	0.00	0.52	1.11	1.75	2.70	3.39	4.80	5.64	1.66

Table 8.3.: Quantiles and RMSEP of absolute errors (NIR data [ethanol])

with $\bar{y}$ being the mean of the y_i and $\tilde{y}$ being the median. Then we define the quotient of squared errors as $PRESS/TSS$, and the quotient of absolute errors as $PSARE/TSA$.

The smaller these quotients get, the better the corresponding model predicts.

From Table 8.4 and Table 8.5 we see, that above quotients are relatively high for methods applied to the NIR (glucose) data. The methods “elasticnet”, “lasso 1se”, “lasso min”, “snet 1se”, and “snet min” perform well in the other situations, and “genet cp” and “genet gcv” perform better than their “enet” counterparts. For the NIR data (glucose and ethanol), the “ridge” methods and “fused” performed worse than the competing methods.

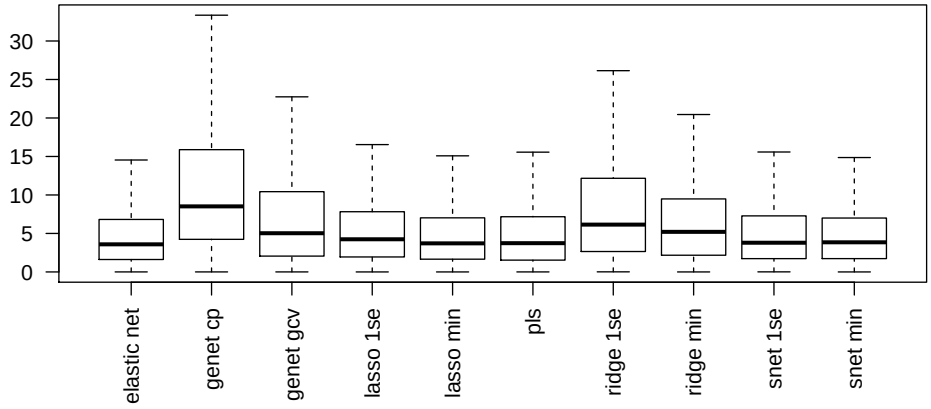
We also see, that “pls” is always among the best model estimation procedures, but it can always be replaced by a penalized regression procedure, leading to sparse solution. More exactly the methods “elasticnet”, “lasso 1se”, “lasso min” get almost equal residual errors with outliers of similar magnitude.

	PAC	NIR (glucose)	NIR (ethanol)
elasticnet	0.074	0.375	0.062
enet cp	0.867	0.472	0.423
enet gcv	0.867	0.384	0.057
fused		0.742	0.292
genet cp	0.169	0.422	0.116
genet gcv	0.113	0.360	0.056
lasso 1se	0.081	0.402	0.072
lasso min	0.077	0.368	0.066
pls	0.082	0.348	0.055
ridge 1se	0.139	0.654	0.248
ridge min	0.102	0.622	0.235
snet 1se	0.090	0.383	0.071
snet min	0.084	0.367	0.067

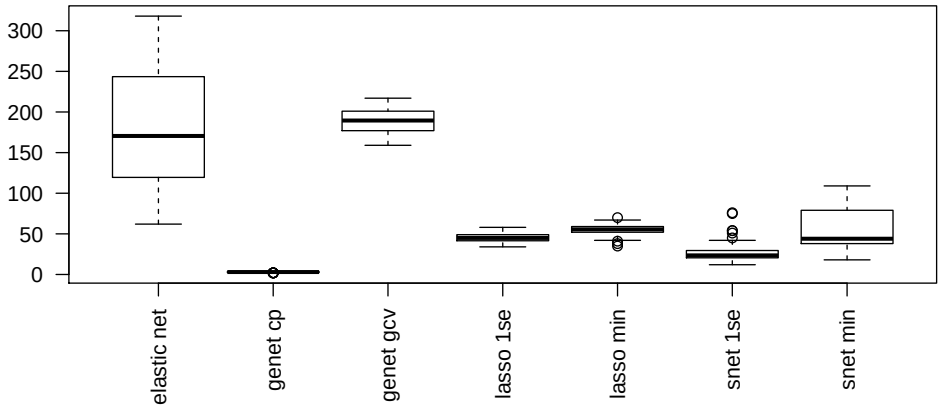
Table 8.4.: Quotients of absolute errors

	PAC	NIR (glucose)	NIR (ethanol)
elasticnet	0.009	0.163	0.005
enet cp	0.751	0.257	0.171
enet gcv	0.751	0.175	0.004
fused		0.650	0.112
genet cp	0.038	0.209	0.016
genet gcv	0.029	0.148	0.004
lasso 1se	0.009	0.183	0.006
lasso min	0.011	0.156	0.005
pls	0.013	0.141	0.004
ridge 1se	0.045	0.503	0.068
ridge min	0.023	0.455	0.062
snet 1se	0.120	0.169	0.007
snet min	0.024	0.153	0.006

Table 8.5.: Quotients of squared errors

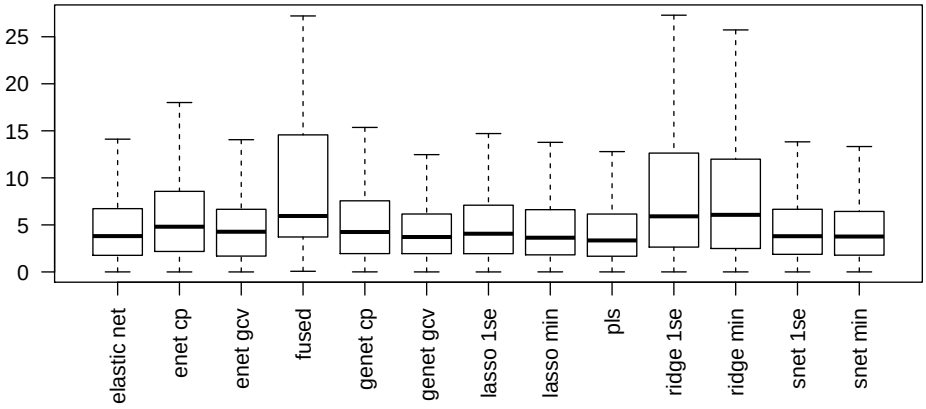


(a) Absolute errors (without outliers)

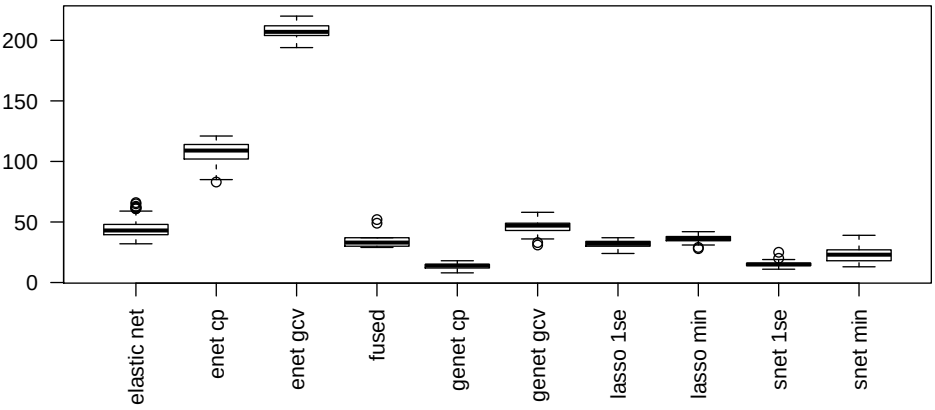


(b) Number of non-zero parameters

Figure 8.1.: Performance on the PAC data

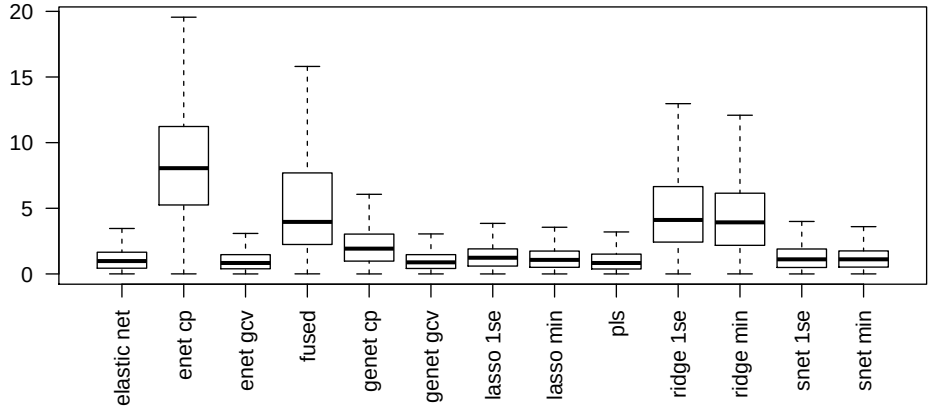


(a) Absolute errors (without outliers)

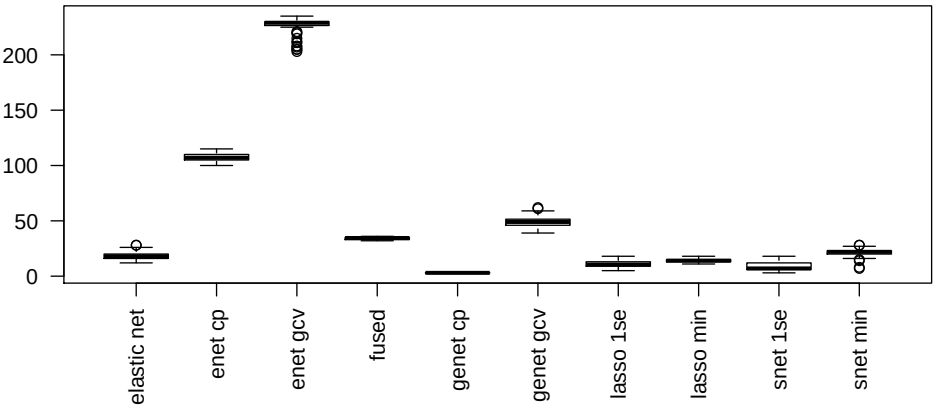


(b) Number of non-zero parameters

Figure 8.2.: Performance on the NIR data (glucose)



(a) Absolute errors (without outliers)



(b) Number of non-zero parameters

Figure 8.3.: Performance on the NIR data (ethanol)

9. Discussion

In this work we focused on three concepts: sparsity, (bridge) penalties, and model selection based on the in-sample prediction error. We tried to summarize and to motivate them with a little mathematics.

In chemometrics, penalized regression seems to be far less popular in comparison to the methods PLS or its close relative principal component regression (PCR). After viewing our results, this fact may not be easy to explain. With the recently introduced bridge penalties, more possibilities are available to adapt the model complexity to the data. But the results also show, that even the more simple and more intuitive penalized regression methods can serve the needs well, with the Lasso leading the way. Another striking fact is, that in all situations a well performing model with many non-zero variables, can be replaced by an equally well performing model with much less non-zero variables. Not necessarily a compromise between the prediction property and the interpretability of model estimation methods has to be found.

One reason for these good results may be the optimization methods, which were introduced recently in the field of penalized regression. Especially coordinate descent methods converge very fast to a suitable stationary point, even in the high-dimensional case.

Further research has to be done in the problem of model selection. Especially on the elastic net, we saw that different model selection procedures may have a severe impact on the performance in model estimation. The most stable results where gained by using cross-validation.

In conclusion, we strongly recommend penalized regression methods as an alternative to PLS, since these methods show equal or better performance, and due to sparsity they lead to much simpler models, being more stable and better interpretable.

A. R Code

```
CrossValGroups <- function(n, g) {
  # forms random groups for cross validation
  #
  # Args:
  #   n: number of observations
  #   g: number of groups
  #
  # Returns:
  #   vector of groupselection (integer; ranges from 1 to g)
  groups <- rep(1:g, len = n)
  perm.groups <- sample(groups)
  return(perm.groups)
}

ParSelElasticNet <- function(x.inner, y.inner, alpha.split=10) {
  # selects the parameters alpha and lambda with the least cross validation
  # error. Alpha runs through alpha.split equidistant points between 0 and 1
  # This function uses package "glmnet"
  #
  # Args:
  #   x.inner: matrix of variables
  #   y.inner: vector of the dependend variable
  #   alpha.split: number of equidistant values of alpha between 0 and 1
  #
  # Returns:
  #   list of alpha and lambda
  cvm.matrix <- NULL
  lambda.matrix <- NULL
  n.inner <- length(y.inner)
  for (j in 0:(alpha.split-1)) {
    groups.for.cv <- CrossValGroups(n=n.inner, g=10)
    res.cv.glmnet <- cv.glmnet(x.inner, y.inner, alpha=j/alpha.split,
                               foldid=groups.for.cv)
    cvm.matrix <- cbind(cvm.matrix, res.cv.glmnet$cvm)
```

```
    lambda.matrix <- cbind(lambda.matrix, res.cv.glmnet$lambda)
  }
  idx.cvm.min <- which(cvm.matrix == min(cvm.matrix), arr.ind=TRUE)
  lambda <- lambda.matrix[idx.cvm.min]
  alpha <- (idx.cvm.min[2] - 1)/alpha.split
  return(list(alpha=alpha, lambda=lambda))
}

ParSelElasticNet2 <- function(x.inner, y.inner, alpha.split=10, bridge="enet",
                              method="cp") {
  # selects the parameters alpha and lambda with either mallow's Cp or
  # generalized cross validation. Alpha runs through alpha.split equidistant
  # points between 0 and 1. This function uses package "msgps"
  # Args:
  #   x.inner: matrix of variables
  #   y.inner: vector of the dependend variable
  #   alpha.split: number of equidistant values of alpha between 0 and 1
  #   method: "cp" or "gcv"
  #   bridge: either "enet" or "genet" for (generalized) elastic net
  #
  # Returns:
  #   list of alpha and lambda
  criterion <- NULL
  lambda.vector <- NULL
  if (bridge == "enet") {
    for (j in 0:(alpha.split-1)){
      res.msgps <- msgps(x.inner, y.inner, penalty=bridge, alpha=j/alpha.split)
      if (method == "cp") {
        criterion <- c(criterion, min(res.msgps$dfcp_result$result))
        w.crit <- which.min(res.msgps$dfcp_result$result)
        lambda.vector <- c(lambda.vector,
                           res.msgps$dfgps_result$tuning_stand[w.crit])
      }
      if (method == "gcv") {
        criterion <- c(criterion, min(res.msgps$dfgcv_result$result))
        w.crit <- which.min(res.msgps$dfgcv_result$result)
        lambda.vector <- c(lambda.vector,
                           res.msgps$dfgps_result$tuning_stand[w.crit])
      }
    }
  }
  if (bridge == "genet") {
```

```
    for (j in 1:(alpha.split-1)){
      res.msgps <- msgps(x.inner, y.inner, penalty=bridge, alpha=j/alpha.split)
      if (method == "cp") {
        criterion <- c(criterion, min(res.msgps$dfcp_result$result))
        w.crit <- which.min(res.msgps$dfcp_result$result)
        lambda.vector <- c(lambda.vector,
                           res.msgps$dfgps_result$tuning_stand[w.crit])
      }
      if (method == "gcv") {
        criterion <- c(criterion, min(res.msgps$dfgcv_result$result))
        w.crit <- which.min(res.msgps$dfgcv_result$result)
        lambda.vector <- c(lambda.vector,
                           res.msgps$dfgps_result$tuning_stand[w.crit])
      }
    }
  }
  idx.crit.min <- which.min(criterion)
  alpha <- idx.crit.min/alpha.split
  lambda <- lambda.vector[idx.crit.min]
  return(list(alpha=alpha, lambda=lambda))
}

ParSelFusedLasso <- function(x.inner, y.inner) {
  mse.matrix <- matrix(NA, nrow=5, ncol=5)
  n.inner <- length(y.inner)
  for (j in 1:5) {
    print("III")
    print(j)
    lambda1 <- exp(j/3) - 1
    for (k in 1:5) {
      print("IV")
      print(k)
      lambda2 <- exp(k/3 ) - 1
      groups.for.cv <- CrossValGroups(n.inner, 10)
      y.pred <- NULL
      y.test <- NULL
      for (i in 1:10) {
        print("V")
        print(i)
        x.cv <- x.inner[groups.for.cv != i, ]
        y.cv <- y.inner[groups.for.cv != i]
        x.test <- x.inner[groups.for.cv == i, ]
```

```
        y.test <- c(y.test, y.inner[groups.for.cv == i])
        print(y.test)
        fused.obj <- penalized(y.cv, x.cv, fusedl=TRUE, lambda1=lambda1,
                               lambda2=lambda2, model="linear", maxiter=100)
        y.pred <- c(y.pred, predict(fused.obj, penalized=x.test)[, 1])
        print(y.pred)
      }
      mse.matrix[j, k] <- sum((y.pred - y.test)^2)
    }
    print(mse.matrix)
  }
  idx.mse.min <- which(mse.matrix == min(mse.matrix), arr.ind=TRUE)
  print(idx.mse.min)
  lambda1 <- exp(idx.mse.min[1]/3) - 1
  lambda2 <- exp(idx.mse.min[2]/3) - 1
  return(list(lambda1=lambda1, lambda2=lambda2))
}

MainFunction <- function(x, y, k.val=10, repetitions.val=10,
                         method1="elasticnet", method2="cp", alpha.split=10) {
  # performs the repeated double cross validation and calls subfunctions for
  # calibration
  #
  # Args:
  #   x: matrix of variables
  #   y: vector of the dependent variable
  #   k.val: number of splits for cross validation
  #   repetitions.val: number of outer cross validation loops
  #   method1: one of the following: "sparsenet", "elasticnet", "enet",
  #                                  "genet", "ridge", "lasso", "fused", "pls"
  #   method2: if method1 == "sparsenet"; either "parms.min" or "parms.1se"
  #            if method1 == "enet" or "genet"; either "cp" or "gcv"
  #            if method1 == "ridge" or "lasso"; either "lambda.1se" or
  #                          "lambda.min"
  #   alpha.split: number of equidistant alpha values between 0 and 1
  #
  # Returns:
  #   a list containing the original and predicted y values and a vector
  #   of the numbers of nonzero estimates
  n <- length(y)
  y.val.save <- NULL
  y.pred.save <- NULL
```

```
nzero.save <- NULL
for (j in 1:repetitions.val) {
  groups <- CrossValGroups(n, k.val)
  for (i in 1:k.val) {
    x.inner <- x[groups != i, ]
    y.inner <- y[groups != i]
    y.val.save <- c(y.val.save, y[groups == i])
    if (method1 == "elasticnet") {
      parameters <- ParSelElasticNet(x.inner, y.inner, alpha.split)
      obj.glmnet <- glmnet(x.inner, y.inner, alpha=parameters$alpha)
      pred.glmnet <-predict(obj.glmnet, newx=x[groups == i, ],
                            s=parameters$lambda, type="response")
      nzero.new <- nrow(predict(obj.glmnet, newx=x[groups == i, ],
                        s=parameters$lambda, type="nonzero"))
      y.pred.save <- c(y.pred.save, pred.glmnet)
      nzero.save <- c(nzero.save, nzero.new)
    }
    if (method1 == "ridge"){
      cv.ridge <- cv.glmnet(x.inner, y.inner, alpha=0)
      pred.ridge <-predict(cv.ridge, newx=x[groups == i, ], s=method2,
                           type="response")
      nzero.new <- nrow(predict(cv.ridge, newx=x[groups == i, ], s=method2,
                                type="nonzero"))
      y.pred.save <- c(y.pred.save, pred.ridge)
      nzero.save <- c(nzero.save, nzero.new)
    }
    if (method1 == "lasso"){
      obj.lasso <- cv.glmnet(x.inner, y.inner, alpha=1)
      pred.lasso <-predict(obj.lasso, newx=x[groups == i, ], s=method2,
                           type="response")
      nzero.new <- nrow(predict(obj.lasso, newx=x[groups == i, ], s=method2,
                                type="nonzero"))
      y.pred.save <- c(y.pred.save, pred.lasso)
      nzero.save <- c(nzero.save, nzero.new)
    }
    if ((method1 == "enet") | (method1 == "genet")) {
      parameters <- ParSelElasticNet2(x.inner, y.inner, alpha.split,
                                      bridge=method1, method=method2)
      obj.msgps <- msgps(x.inner, y.inner, penalty=method1,
                         alpha=parameters$alpha)
      pred.msgps <- predict(obj.msgps, X=x[groups == i, ],
```

```
                          tuning=parameters$lambda)
  y.pred.save <- c(y.pred.save, pred.msgps)
  if (method2 == "cp") {
    nzero.new <- sum(coef(obj.msgps)[,"Cp"] != 0)
  }
  if (method2 == "gcv") {
    nzero.new <- sum(coef(obj.msgps)[,"GCV"] != 0)
  }
  nzero.save <- c(nzero.save, nzero.new)
}
if (method1 == "sparsenet") {
  obj.cv.sparsenet <- cv.sparsenet(x.inner, y.inner)
  pred.sparsenet <- predict(obj.cv.sparsenet, which=method2,
                            newx=x[groups == i,])
  y.pred.save <- c(y.pred.save, pred.sparsenet)
  if (method2 == "parms.1se") {
    nzero.new <- obj.cv.sparsenet$nzero[obj.cv.sparsenet$which.1se[1],
                                        obj.cv.sparsenet$which.1se[2]]
    nzero.save <- c(nzero.save, nzero.new)
  }
  if (method2 == "parms.min") {
    nzero.new <- obj.cv.sparsenet$nzero[obj.cv.sparsenet$which.min[1],
                                        obj.cv.sparsenet$which.min[2]]
    nzero.save <- c(nzero.save, nzero.new)
  }
}
if (method1 == "fused"){
  parameters <- ParSelFusedLasso(x.inner, y.inner)
  obj.fused <- penalized(y.inner, x.inner, fusedl=TRUE,
                         lambda1=parameters$lambda1,
                         lambda2=parameters$lambda2,
                         model="linear", maxiter=200)
  pred.fused <- predict(obj.fused, x[groups == i,])[, 1]
  nzero.new <- length(coef(obj.fused))-1
  nzero.save <- c(nzero.save, nzero.new)
  y.pred.save <- c(y.pred.save, pred.fused)
}
if (method1 == "pls") {
  x.test <- x[groups == i, ]
  obj.pls <- plsr(y.inner~x.inner, ncomp=30, validation="CV",
                  segments=10, segment.type="random", method="simpls")
```

```
        opt.ncomps <- which.min(RMSEP(obj.pls)$val[1,,-1])
        pred.pls <- predict(obj.pls, x.test, ncomp=opt.ncomps, type="response")
        y.pred.save <- c(y.pred.save, pred.pls)
      }
    }
  }
  return(list(y.val.save=y.val.save, y.pred.save=y.pred.save,
              nzero.save=nzero.save))
}
```

Bibliography

J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96(456): 1348–1360, 2001.

l. E. Frank and J. H. Friedman. A statistical view of some chemometrics regression tools. *Technometrics*, 35(2):109–135, 1993.

J. Friedman, T. Hastie, H. Höfling, and R. Tibshirani. Pathwise coordinate optimization. *The Annals of Applied Statistics*, 1(2):302–332, 2007.

J. Friedman, T. Hastie, and R. Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1, 2010.

J. H. Friedman. Fast sparse regression and classification. 2008. URL `http://www-stat.stanford.edu/~jhf/ftp/GPSpub.pdf`.

W. J. Fu. Penalized regressions: the bridge versus the lasso. *Journal of Computational and Graphical Statistics*, 7(3):397–416, 1998.

J. J. Goeman. *Penalized: L1 (lasso and fused lasso) and L2 (ridge) penalized estimation in GLMs and in the Cox model*, 2012. R package version 0.9-42.

G. H. Golub, M. Heath, and G. Wahba. Generalized cross-validation as a method for choosing a good ridge parameter. *Technometrics*, 21(2):215–223, 1979.

T. J. Hastie, R. J. Tibshirani, and J. H. Friedman. *The elements of statistical learning: Data mining, inference, and prediction*. Springer, New York, 2nd edition, 2009. corrected 10th printing.

H. Havlicek. *Lineare Algebra für Technische Mathematiker*. Heldermann, Lemgo, 2006.

K. Hirose. *msgps: Degrees of freedom of elastic net, adaptive lasso and generalized elastic net*, 2012. URL `http://CRAN.R-project.org/package=msgps`. R package version 1.3.

K. Hirose, S. Tateishi, and S. Konishi. Efficient algorithm to select tuning parameters in sparse regression modeling with regularization. *arXiv preprint arXiv:1109.2411*, 2011.

A. E. Hoerl and R. W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

M. Koecher. *Lineare Algebra und analytische Geometrie*. Springer, Berlin, 4th edition, 2003.

B. Liebmann, A. Friedl, and K. Varmuza. Determination of glucose and ethanol in bioethanol production by near infrared spectroscopy and chemometrics. *Analytica Chimica Acta*, 642(1):171–178, 2009.

R. Mazumder, J. H. Friedman, and T. Hastie. Sparsenet: Coordinate descent with nonconvex penalties. *Journal of the American Statistical Association*, 106 (495), 2011.

R. Mazumder, T. Hastie, and J. Friedman. *sparsenet: Fit sparse linear regression models via nonconvex optimization*, 2013. URL `http://CRAN.R-project.org/package=sparsenet`. R package version 1.1.

X. Shen and J. Ye. Adaptive model selection. *Journal of the American Statistical Association*, 97(457):210–221, 2002.

C. M. Stein. Estimation of the mean of a multivariate normal distribution. *The Annals of Statistics*, 9(6):1135–1151, 1981.

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

R. Tibshirani, M. Saunders, S. Rosset, J. Zhu, and K. Knight. Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society: Series B*, 67(1):91–108, 2004.

K. Varmuza and P. Filzmoser. *Introduction to multivariate statistical analysis in chemometrics*. CRC press, Boca Raton, FL, 2009.

K. Varmuza, P. Filzmoser, and M. Dehmer. Multivariate linear QSPR/QSAR models: Rigorous evaluation of variable selection for PLS. *Computational and Structural Biotechnology Journal*, 5, 2013.

S. Wold. Chemometrics; what do we mean with it, and what do we want from it? *Chemometrics and Intelligent Laboratory Systems*, 30(1):109–115, 1995.

J. Ye. On measuring and correcting the effects of data mining and model selection. *Journal of the American Statistical Association*, 93(441):120–131, 1998.

C.-H. Zhang. Nearly unbiased variable selection under minimax concave penalty. *The Annals of Statistics*, 38(2):894–942, 2010.

H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67 (2):301–320, 2005.