Penalised Regression Methods for Genomic Applications

Jeremy Cai

October 2023

Preface

Plagiarism declaration

This piece of work is a result of my own work and I have complied with the Department's guidance on multiple submission and on the use of AI tools. Material from the work of others not involved in the project has been acknowledged, quotations and paraphrases suitably indicated, and all uses of AI tools have been declared.

Contents

Chapter 1

Introduction

In the past century, the study of genes has been a thriving area of research, giving major contributions to a wide range of fields in medicine and biotechnology [1]. During the 1990s, key inventions were made in microarray technology, such as, complementary DNA microarrays [2], nylon microarrays [3] and oligonucleotide chips [4]. These quantified the expression levels of certain genes through measurements of varying fluorescence intensities and allowed researchers to study multiple genes at once. A gene expression matrix could then be formulated through the compilation of several microarrays from samples under different conditions [5].

We consider two genomic applications using microarray data. Firstly, we wish to identify candidate genetic biomarkers for specific types of cancers, based on whether the sample is from a tumour or non-tumour tissue. Secondly, we aim to investigate the relation between any known/candidate genetic biomarkers with other genes in the dataset. Our simulation data is based on the colon dataset from the R package bigLasso [6]. Our real life data is taken from the Gene Expression Omnibus (GEO) database [7] and queried via the R package GEOquery [8]. The queries are: GDS4102 [9], GDS4336 [10], GDS4103 [11], and all concern pancreatic cancer data.

Mathematically, we approach these applications by fitting a penalised regression model with our response being respectively either Logistic or Gaussian. Recent literature for genomic biomarker identification commonly includes methods such as the Lasso [12] [13] [14] [15] and its variants such as the Adaptive Lasso [16]. However, despite its popular use, the Lasso has many theoretical shortcomings. Firstly, as Zou noted, it does not satisfy the oracle properties as it over-penalises large coefficients [17]. Therefore, Zou proposed the Adaptive Lasso but this only satisfied the properties, given some regularity conditions, which most high dimensional estimators violate. Similarly, Fan and Li proposed SCAD to reduce this excessive bias problem [18]. Secondly, as Zhao and Yu proved, the Lasso is model selection inconsistent unless the irrepresentable conditions are satisfied [19]. Since genes contributing to the same biological process depend on each other, our data will exhibit strong multicollinearity, so the conditions will be violated. Addressing this, Zhang developed the Minimax Concave Penalty (MCP) which can be model selection consistent even if the irrepresentable condition is not satisfied [20]. Furthermore, Zou and Hastie showed that the Lasso selects collinear predictors randomly, hence proposing the Elastic Net [17].

The rest of this report is organised as follows. In chapter 2 we review linear regression and explore its limitations to motivate penalised linear regression. In chapter 3 we present the theory for 6 penalised regression methods: Ridge, Lasso, Adaptive Lasso, Elastic Net, SCAD, and MCP. Then, in chapter 4, we explore the computational aspects of the model fitting process. After that, in chapter 5, we compare our methods empirically with simulations. Finally, in chapter 6, we apply our methods on real life data and conclude with a discussion in chapter 7.

Chapter 2

Basics of Regression Modelling

In this chapter we provide an overview of classical linear regression, generalised linear models and common variable selection methods. We then explore their limitations in order to motivate penalised regression.

2.1 The Linear Model

Definition 2.1.1 (The Linear Model [21])**.** We denote n as our total sample size and p as our total number of parameters. Recall the linear model in matrix form.

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

where $y \in \mathbb{R}^n$ represents our response variable, $X \in \mathbb{R}^{n \times p}$ is our data matrix, $\beta \in \mathbb{R}^p$ is a list of regression coefficients and $\epsilon \in \mathbb{R}^n$ are our errors.

We have the following assumptions:

- A1. Linearity $E(\epsilon_i) = 0$
- A2. Homoscedasticity $Var(\epsilon_i) = \sigma^2$
- A3. Independence $Cov(\epsilon_i, \epsilon_j) = 0$
- A4. Large Sample size $n > p$
- A5. $\epsilon_i \sim N(0, \sigma^2)$

From these assumptions, we can calculate the distribution of our response. Since ϵ is the only random variable,

$$E(y) = X\beta + E(\epsilon) = X\beta$$
$$Var(y) = 0 + Var(\epsilon) = \sigma^2 I$$

Hence, $y \sim \mathcal{N}(X\beta, \sigma^2 I)$. Often, if the response y is continuous but not normal, we can perform a Box-Cox transformation. Note that this transformation is monotonic.

Definition 2.1.2 (Box-Cox Transformation, pg 214 [22])**.** For $i = 1, ..., n$,

$$y_i = \begin{cases} \frac{y_i^\lambda - 1}{\lambda} & \text{if } \lambda \neq 0 \\ \log(y_i) & \text{if } \lambda = 0 \end{cases} \tag{2.2}$$

λ is typically estimated using profile likelihood methods.

Definition 2.1.3 (Ordinary Least Squares (OLS) [21])**.** Recall that one can estimate β by minimizing the residual sum of squares. We obtain:

$$\hat{\beta} = \arg\min_{\beta \in \mathbb{R}^p} \|y - X\beta\|_2^2 \tag{2.3}$$

$$= (X^T X)^{-1} X^T y \tag{2.4}$$

Note that since $E(\hat{\beta}) = \beta$, $\hat{\beta}$ is an unbiased estimator.

2.2 Generalised Linear Models

In many gene expression data sets for cancer the response variable is often binary. Namely, it is made up of 1's and 0's to represent whether a patient has a tumour or not respectively. Hence, the continuous response $y \sim \mathcal{N}(X\beta, \sigma^2 I)$ is no longer suitable. Therefore, we present an extension of the linear model.

Definition 2.2.1 (Generalised linear model (GLM) [23])**.** Let $x_i \in \mathbb{R}^p$ represent an arbitrary row (observation) of X. The generalised linear model is specified through the following components.
1. A linear predictor.

$$\eta = \beta^T x_i \tag{2.5}$$

2. An injective response function h.

$$\mu = E(y|x_i, \beta) = h(\eta) \tag{2.6}$$

Or equivalently,

$$g(\mu) = \beta^T x_i \tag{2.7}$$

where $g = h^{-1}$ is the link function.
3. A distributional assumption which is described by an exponential dispersion family (EDF) with parameters θ and ϕ depending on x_i and β.

$$P(y|x_i, \beta) = P(y|\theta(x_i, \beta), \phi(x_i, \beta)) = \exp\left(\frac{y\theta - b(\theta)}{\phi} + c(y, \phi)\right) \tag{2.8}$$

Intuitively, we can think of the link function as a map between the responses generated from the linear model and a GLM, which is under a different distribution, as specified by the distributional assumption. Note that the linear model itself is a GLM.

Example 2.2.1 (Gaussian GLM with identity link)**.** Let $\eta = \beta^T x_i$ be our linear predictor. We use the identity link, which maps the linear predictor to itself $h(\eta) = \eta$. Let $y \sim \mathcal{N}(\mu, \sigma^2 I)$. We first write the Gaussian probability density function as an EDF:

$$P(y_i|\mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}(y_i - \mu)^2\right) \tag{2.9}$$

$$= \exp\left(-\frac{1}{2\sigma^2}(y_i^2 - 2y_i\mu + \mu^2) - \frac{1}{2}\log(2\pi\sigma^2)\right) \tag{2.10}$$

Hence, $\theta = \mu$, $\phi = \sigma^2$, and $c(y_i, \phi) = -\frac{y_i^2}{2\sigma^2} - \frac{1}{2}\log(2\pi\sigma^2)$. Now, since we use the identity link, $\eta = \theta = \mu$, so $y \sim \mathcal{N}(X\beta, \sigma^2 I)$ and we obtain the linear model.

Example 2.2.2 (Logistic GLM with logit link pg 35 [23])**.** Following the above framework, let $x_i \in \mathbb{R}^p$ be a single observation. Our linear predictor is $\eta = \beta^T x_i$. We use the logit link function

$$g(x_i) = \log\left(\frac{\pi(x_i)}{1-\pi(x_i)}\right) \tag{2.11}$$

And a Bernoulli distributional assumption

$$P(y|\pi(x_i)) = \exp\left(y \log\left(\frac{\pi(x_i)}{1-\pi(x_i)}\right) + \log(1-\pi(x_i))\right) \tag{2.12}$$

We see that by the properties of an EDF, $E(Y|\pi(x_i)) = \pi(x_i)$, and hence we can write the logit link as $g(\mu) = \log(\frac{\mu}{1-\mu})$. Its log likelihood is:

$$l(\beta) = \sum_i y_i \beta^T x_i - \log(1 + e^{\beta^T x_i}) \tag{2.13}$$

Using maximum likelihood estimation, we see that the logistic score function is

$$S(\beta) = \sum_{i=1}^{n} \left(y_i x_i - \frac{e^{\beta^T x_i}}{1 + e^{\beta^T x_i}} x_i\right) \tag{2.14}$$

This system of equations is nonlinear and a common method to solve it is through Iterated Re-weighted Least Squares (IRLS) [24].

2.3 Variable Selection Methods

Once we have estimated the coefficients under the Gaussian or Logistic frameworks, it is often desirable to use some variable selection method to find a simpler model, especially if the number of parameters is large. This suggests to us which parameters are significant in affecting the response and improves our model interpretation. Typically, Gaussian OLS estimates have low bias but high variance [25]. By setting some coefficients to zero, we may achieve a lower variance and better predictive accuracy at the cost of slightly more bias. This is called the bias variance trade-off.

Two of the most common model selection methods are best subset selection and stepwise selection. Both differentiate models based on some criterion such as Mallows C_p [26], Bayesian Information Criterion (BIC) [27] and Akaike Information Criterion (AIC) [28].

Definition 2.3.1 (Best Subset Selection [25])**.** This method fits models to every possible combination of predictors and selects the best one according to the criterion given above.

Definition 2.3.2 (Forward/Backward Stepwise Selection [21])**.** Forward stepwise selection starts with the null model and adds parameters, whereas backward stepwise selection starts with the saturated model and discards parameters. The processes stop according to the criterion given above.

Whilst both methods are easy to implement, they have significant shortcomings. Firstly, for large p best subset selection has infeasible computational costs, as we would have to consider all 2^p possible subset models. Secondly, both processes are discrete, which leads to high variance in prediction [17]. Thirdly, stepwise processes may be caught up into a cycle, thereby giving a local optimal model instead of the global optimal model [17]. In addition, stepwise selection is unstable [29]. For example, if one carries out a backwards stepwise selection on a dataset and then on the same dataset with one observation removed, the two selected models have different numbers of parameters. Hence, "small perturbations" in the data give "drastic changes" to the selected models. We discuss this further in Section 5.4.

2.4 Limitations of Classical Regression Modelling

With microarray data, we will typically encounter two statistical issues. Firstly, we have a problem of high dimensionality, with $p > n$ as the number of genes far exceeds the patient sample size. For example, the colon dataset [6] we use for simulations has 2000 genes but only 62 samples. Secondly, we have strong multicollinearity in the data, as genes contributing to the same biological process will affect each other.

As a result, OLS cannot be calculated, since the columns of $X^T X$ will not be linearly independent so, $X^T X$ will not be invertible. Similarly, the system of equations used to estimate β for the logistic case will be under-determined, as we will have $p - n$ free variables. Therefore, we will need to introduce regularization.

We also desire some properties for our modelling methods. Firstly, the computation must be efficient as we analyse data sets with thousands of parameters at a time Hence, selection methods similar to best subset selection are infeasible. Secondly, we prefer no over fitting, namely, we do not want our model to fit perfectly to the data, as it is only a small proportion of the population due to its small sample size. Thirdly, we would like stable model selection with respect to observation removal. Since, our sample size is small with only 50-70 samples, compared to a parameter size of order 10^5, there may be missing observations. Finally, since our applications are focused on identifying significant genes according to the response, we would like model selection consistent methods. We also analyse the parameter estimation consistency of our methods. These consistency properties mean that the method selects the true model and estimates the true parameter values with probability 1 as $n \to \infty$ respectively. They are also commonly rephrased mathematically as the oracle properties.

2.5 Penalised Regression

Definition 2.5.1 (Penalised Regression)**.** For Gaussian and Logistic models respectively, a penalised regression method takes the form:

$$\hat{\beta} = \underset{\beta \in \mathbb{R}^p}{\arg\min} \|y - X\beta\|_2^2 + P(\beta; \theta) \tag{2.15}$$

$$\hat{\beta} = \underset{\beta \in \mathbb{R}^p}{\arg\min} \sum_{i=1}^{n} \left(-y_i \beta^T x_i + \log(1 + e^{\beta^T x_i}) \right) + P(\beta; \theta) \tag{2.16}$$

where the set of positive tuning parameters which control the shrinkage of the estimated coefficients is θ. For $\beta \neq 0$, $P > 0$ and when $\beta = 0$, $P = 0$. Essentially, we add a penalty term to the classical minimisation problems to estimate β. In the Gaussian case, the penalty is added to the residual sum of squares. For the logistic case, we add the penalty to the negative logistic log likelihood as we maximise the positive log likelihood to estimate β.

Remark 2.5.1. Note, that X must be standardised, (with column mean 0 and variance 1) before the penalty is applied to ensure all the covariates are penalised equally. In many R packages, such as, glmnet [30] and ncvreg [31], this is done automatically.

In this report, we will explore penalised regression for linear methods. A broader overview of penalised regression for group, additive, partial linear and non-parametric models can be found in [32].

Chapter 3

Penalised Methods

In this chapter, we present the theoretical ideas behind six penalised methods: Ridge, Lasso, Elastic Net, Adaptive Lasso, SCAD, and MCP. We explore their properties concerning model selection, parameter estimation and stability.

3.1 Ridge

Definition 3.1.1 (Ridge, pg 63 [25])**.** First introduced in 1962, the Ridge penalty uses the L^2 norm [33]. Let $\lambda > 0$. For the Gaussian and Logistic models respectively, the Ridge estimates are:

$$\hat{\beta}^{Ridge} = \underset{\beta \in \mathbb{R}^p}{\arg\min} \|y - X\beta\|_2^2 + \lambda \|\beta\|_2^2 \tag{3.1}$$

$$\hat{\beta}^{Ridge} = \underset{\beta \in \mathbb{R}^p}{\arg\min} \sum_{i=1}^{n} \left(-y_i \beta^T x_i + \log(1 + e^{\beta^T x_i}) \right) + \lambda \|\beta\|_2^2 \tag{3.2}$$

Alternatively, we can write our problem in terms of the classical minimisation problem under a constraint. For some $t > 0$,

$$\hat{\beta}^{Ridge} = \underset{\beta \in \mathbb{R}^p}{\arg\min} \|y - X\beta\|_2^2 \text{ subject to } \|\beta\|_2^2 \leq t \tag{3.3}$$

$$\hat{\beta}^{Ridge} = \underset{\beta \in \mathbb{R}^p}{\arg\min} \sum_{i=1}^{n} \left(-y_i \beta^T x_i + \log(1 + e^{\beta^T x_i}) \right) \text{ subject to } \|\beta\|_2^2 \leq t \tag{3.4}$$

The tuning parameter λ has a one-to-one correspondence with t. Both act as a constraint on the magnitude of the β_j. We do not penalize the intercept, so the dimension p refers to the total number of parameters excluding the intercept. In Figure 3.1 below, we see an example of the quadratic shrinkage effect.

Lemma 3.1.1 (Closed form solution to Gaussian Ridge)**.**

$$\hat{\beta}^{Ridge} = (X^T X + \lambda I)^{-1} X^T y \tag{3.5}$$

Proof. The proof follows by differentiation and rearrangement.

$$\text{Let } f(\beta) \coloneqq \underset{\beta \in \mathbb{R}^p}{\arg\min} \|y - X\beta\|_2^2 + \lambda \|\beta\|_2^2$$
$$f(\beta) = (y - X\beta)^T (y - X\beta) + \lambda \beta^T \beta$$

$$
\begin{aligned}
&= y^T y - 2y^T X\beta + \beta^T X^T X\beta + \lambda\beta^T\beta \\
\frac{\partial f}{\partial \beta} &= -2X^T y + 2X^T X + 2\lambda\beta I \\
\frac{\partial^2 f}{\partial \beta^T \partial \beta} &= 2X^T X + 2\lambda I
\end{aligned}
$$

Now, note that for any vector $v \in \mathbb{R}^p$, $v^T X^T X v = (Xv)^T Xv = \|Xv\|_2^2 \geq 0$, so $X^T X$ is positive semi-definite. Hence, the second derivative is positive definite, which implies we have a minimum. Setting the first derivative to 0, we obtain,

$$(2X^T X + 2\lambda I)\hat{\beta} = 2X^T y$$

So, $\hat{\beta}^{Ridge} = (X^T X + \lambda I)^{-1} X^T y$ □

The Ridge estimates add a positive constant λ to the diagonal of $X^T X$, which allows it to be invertible even with instances of multicollinearity and when $p > n$.

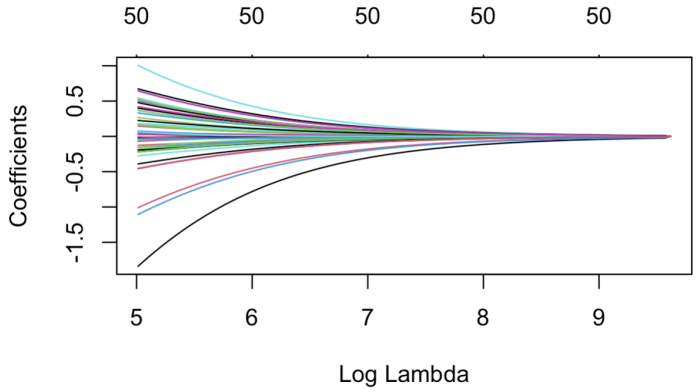

Figure 3.1: A plot of the Ridge estimation path using glmnet. As $\log(\lambda)$ increases, the $|\beta_j|$ become smaller. The numbers at the top indicate how many parameters were selected per λ value. The data was: $X \sim \mathcal{N}(0, I), Y \sim \mathcal{N}(X\beta, I)$ There were 5 non-zero $\beta_j \sim \mathcal{U}(-20, 20)$ and 45 zero β_j.

3.2 Lasso

Definition 3.2.1 (Lasso [34])**.** Let $\lambda > 0$. Similarly, to Ridge, the Gaussian and Logistic Lasso estimates are defined as:

$$\hat{\beta}^{Lasso} = \underset{\beta \in \mathbb{R}^p}{\arg\min} \|y - X\beta\|_2^2 + \lambda\|\beta\|_1 \tag{3.6}$$

$$\hat{\beta}^{Lasso} = \underset{\beta \in \mathbb{R}^p}{\arg\min} \sum_{i=1}^{n} (-y_i \beta^T x_i + \log(1 + e^{\beta^T x_i})) + \lambda\|\beta\|_1 \tag{3.7}$$

Or Alternatively, for some $t > 0$,

$$\hat{\beta}^{Lasso} = \underset{\beta \in \mathbb{R}^p}{\arg\min} \|y - X\beta\|_2^2 \text{ subject to } \|\beta\|_1 \leq t \tag{3.8}$$

$$\hat{\beta}^{Lasso} = \underset{\beta \in \mathbb{R}^p}{\arg\min} \sum_{i=1}^{n} (-y_i \beta^T x_i + \log(1 + e^{\beta^T x_i})) \text{ subject to } \|\beta\|_1 \leq t \tag{3.9}$$

Since the Lasso makes use of the L^1 norm, β is not differentiable at 0 so we have no general closed form solution. However, the idea is still similar: by adding/subtracting a constant to X^TX, the Lasso is able to give a solution in the presence of high dimensional and multicollinear data. Additionally, an important property of the Lasso (unlike Ridge) is that it sets small coefficients to 0, so we can perform simultaneous model selection and parameter estimation. Figure 3.2 below shows an example of the Lasso shrinkage effect for different λ.

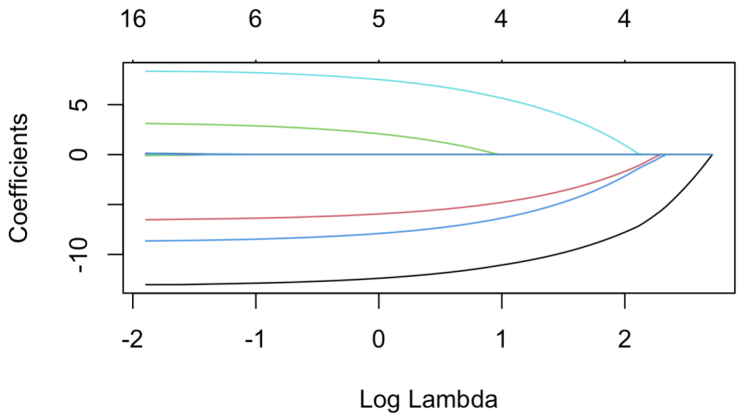

Figure 3.2: A plot of Lasso paths computed with the same data as in Figure 3.1.

Comparing Figures 3.1 and 3.2, we see that for the Lasso there exists values of λ for which some coefficients are non-zero and some are zero. This is an example of the Lasso's simultaneous estimation and selection property. By contrast, the Ridge coefficients are non-zero for all λ.

Lemma 3.2.1 (Gaussian Lasso and Ridge solutions under orthogonal design. [25])**.** *Assume that the columns of X are orthonormal. Let $\hat{\beta}$ represent the OLS estimates.*

$$\textit{(i)}\ \hat{\beta}_j^{\,Lasso} = sign(\hat{\beta}_j)(|\hat{\beta}_j| - \lambda)_+ \tag{3.10}$$

$$\textit{(ii)}\ \hat{\beta}_j^{\,Ridge} = \frac{\hat{\beta}_j}{1+\lambda} \tag{3.11}$$

Where $(x)_+$ returns the positive part of (x), which is x if $x > 0$ and 0 otherwise, and $sign(\cdot)$ gives the sign of $(\cdot)$.

Proof. We base the proof of (i) on [35]. For algebraic clarity, we multiply the RSS by a constant $\frac{1}{2}$. This is a monotonic transformation of the objective function, so the β that minimises it will be unaffected. We write the Gaussian Lasso as

$$\begin{aligned}
\min_\beta \frac{1}{2}\|y - X\beta\|_2^2 + \lambda\|\beta\|_1 &= \min_\beta \frac{1}{2}(y^Ty - 2y^TX\beta + \beta^TX^TX\beta) + \lambda\|\beta\|_1 \\
&= \min_\beta -y^TX\beta + \frac{1}{2}\beta^TX^TX\beta + \lambda\|\beta\|_1 \\
&= \min_\beta \sum_{j=1}^{p}\left(-\hat{\beta}_j\beta_j + \frac{1}{2}\beta_j^2 + \lambda|\beta_j|\right)
\end{aligned}$$

since the columns of X are orthonormal, $X^TX = I$ and $\hat{\beta} = X^Ty$. We have discarded y^Ty since it does not depend on β. Now our problem is the sum of p independent equations, so we minimise

each one individually. Let

$$l_j = -\hat{\beta}_j \beta_j + \frac{1}{2}\beta_j^2 + \lambda|\beta_j|$$

Consider $\hat{\beta}_j \leq 0$, then we must have $\beta_j \leq 0$, since we want $-\hat{\beta}_j \beta_j \leq 0$ to minimise l_i. Taking the derivative,

$$\frac{\partial l_j}{\partial \beta_j} = -\hat{\beta}_j + \beta_j - \lambda = 0$$

Hence, $\hat{\beta}_j^{\,Lasso} = \hat{\beta}_j + \lambda = sign(\hat{\beta}_j)(|\hat{\beta}_j| - \lambda)_+$. Now, consider, $\hat{\beta}_j > 0$, similarly $\beta_j \geq 0$.

$$\frac{\partial l_j}{\partial \beta_j} = -\hat{\beta}_j + \beta_j + \lambda = 0$$

Hence, $\hat{\beta}_j^{\,Lasso} = (\hat{\beta}_j - \lambda)_+ = sign(\hat{\beta}_j)(|\hat{\beta}_j| - \lambda)_+$. Since we get the same answer in both cases, we are done. For (ii) recall from Lemma 3.1.1

$$\begin{aligned}
\hat{\beta}_j^{\,Ridge} &= (X^T X + \lambda I)^{-1}(X^T y) \\
&= (I + \lambda I)^{-1}(X^T y) \\
&= \frac{\hat{\beta}_j}{1 + \lambda}
\end{aligned}$$

□

We see that Ridge estimates are shrunk linearly in proportion to the size of β_j so the coefficients will not become 0. By contrast, each Lasso estimate is truncated by a constant factor λ. Note when $\beta_j = 0$, there is no shrinkage, so on a Lasso path, a positive coefficient will not be penalised into a negative one and vice versa. In general, however, the Lasso estimates may not retain the same signs (positive or negative) as the least squares estimates [34]. In Figure 3.3, we see a comparison of the Lasso and Ridge estimates against Least Squares.

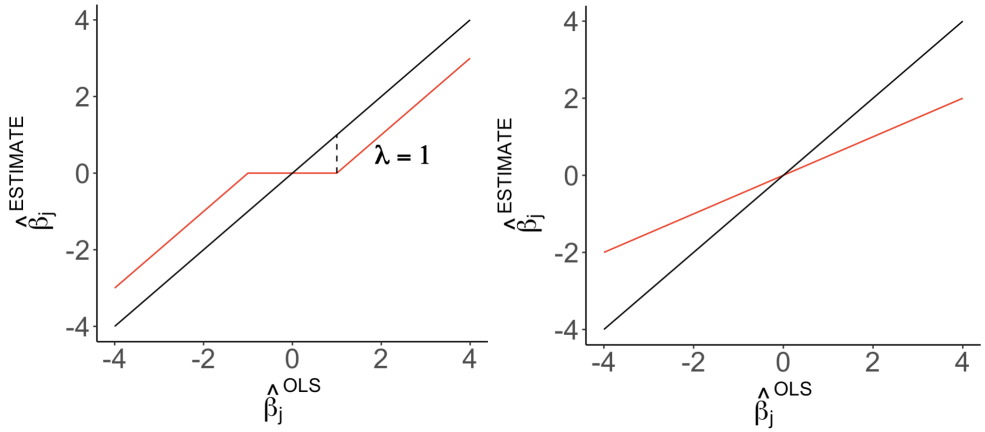

Figure 3.3: Shrinkage comparisons of Lasso (left) against Ridge (right) given an orthogonal design. The black line is the Least Squares estimate for reference. The red lines represent the Lasso and Ridge estimates as a linear transformation of the OLS estimates. In both figures $\lambda = 1$. In this report, all of the shrinkage graphs are plotted using ggplot2. [36]

Example 3.2.1. Let our data be $X \sim \mathcal{N}(0, I)$, with $n = 30$ observations and $p = 5$ parameters. We specify below 3 non-zero β_j and 2 zero β_j. Let $Y = X\beta + \epsilon$ with $\epsilon \sim \mathcal{N}(0, 1)$. We compare OLS, Ridge and Lasso estimates. The models also contain the intercept, which we omit for our comparison. We round our estimated values to 3 decimal places.

Parameters	True β	OLS	Ridge	Lasso
β_1	-2.78	-2.514	-2.350	-2.334
β_2	2.99	2.985	2.564	2.587
β_3	7.27	7.464	6.707	7.149
β_4	0	-0.245	-0.049	0
β_5	0	0.227	0.187	0.046

As an example of model selection, the Lasso estimate sets β_4 to 0.

3.2.1 The Irrepresentable Conditions

We now examine model selection consistency of the Lasso from a theoretical viewpoint. Since consistency is a limit property, we introduce new notation to describe how our variables change dependent on the sample size n.

Definition 3.2.2 (Limit Notation, pg 2544 [19])**.**

- Let $\beta^n \in \mathbb{R}^p$ and $\hat{\beta^n} \in \mathbb{R}^p$ be our true and estimated regression coefficients at some sample size n, respectively.
- Let $\beta^n_{(1)} = \{\beta^n_1, ..., \beta^n_q\}^T$ be the vector of q true non-zero regression coefficients.
- Let $\beta^n_{(2)} = \{\beta^n_{q+1}, ..., \beta^n_p\}^T$ be the vector of $p-q$ true zero regression coefficients.
- Let X_n denote a data matrix with sample size n. Let $X_n(1)$ and $X_n(2)$ denote the first q and last $p-q$ columns of, X respectively.
- Let
$$C^n = \frac{X_n^T X_n}{n} = \begin{pmatrix} C^n_{(11)} & C^n_{(12)} \\ C^n_{(21)} & C^n_{(22)} \end{pmatrix}$$
where, $C^n_{(11)} = \frac{1}{n} X_n(1)^T X_n(1)$, $C^n_{(22)} = \frac{1}{n} X_n(2)^T X_n(2)$, $C^n_{(12)} = \frac{1}{n} X_n(1)^T X_n(2)$, and $C^n_{(21)} = \frac{1}{n} X_n(2)^T X_n(1)$.

The matrix $X^T X$ is commonly referred to as the Gram Matrix. We can center the columns of X by subtracting the column mean from each column value, namely, for some fixed j, $x^*_{ij} = x_{ij} - \frac{1}{n}\sum_{i=1}^{n} x_{ij}$. After centering, $\frac{1}{n-1} X^T X$ is the covariance matrix, so we can think of $\frac{1}{n} X_n^T X_n$ as a matrix of scaled covariances Cov_{sc}. Hence,

$$C^n_{(11)ij} = \text{Cov}_{sc}(X_n(1)_i, X_n(1)_j)$$
$$C^n_{(21)ij} = \text{Cov}_{sc}(X_n(2)_i, X_n(1)_j)$$

where $X_n(1)_i$ represents the i^{th} column of $X_n(1)$. We now introduce two different definitions of statistical consistency.

Definition 3.2.3 (Parameter Estimation Consistency pg 2543 [19])**.**

$$\hat{\beta^n} - \beta \xrightarrow{p} 0, \text{ as } n \to \infty$$

where, $\xrightarrow{p}$ means convergence in probability, more specifically, $\lim_{n\to\infty} \mathbb{P}(|\hat{\beta^n} - \beta| \geq \epsilon) = 0, \forall \epsilon > 0$. An estimator possessing this property is normally referred to as consistent.

Definition 3.2.4 (Model Selection Consistency pg 2543 [19])**.**

$$\mathbb{P}(\{i : \hat{\beta^n_i} \neq 0\} = \{i : \beta^n_i \neq 0\}) \to 1, \text{ as } n \to \infty$$

The above two definitions state that, as our sample size increases to infinity, we expect, with certainty, our parameter estimates to converge to our true parameters values and to select the parameters in the true model. Since our application is about finding relevant predictors, we are more concerned with model selection consistency.

We now define a stronger notion of model selection consistency through sign consistency. This means that the signs (positive, negative, zero) of the $\hat{\beta}_j$ must match the true β_j eventually. In contrast, model selection consistency only requires zero and non-zero β_j to match.

Definition 3.2.5 (Equal in sign, pg 2543 [19])**.** An estimate $\hat{\beta}^n$ which is equal in sign with the true model β^n can be written as,

$$\hat{\beta}^n =_s \beta^n$$

Definition 3.2.6 (Strong Sign Consistency pg 2544 [19])**.** An estimate is strong sign consistent, if $\exists \lambda_n = f(n)$, that is, a function of n and independent of the response and data such that

$$\lim_{n\to\infty} \mathbb{P}(\hat{\beta}^n(\lambda_n) =_s \beta^n) = 1 \tag{3.12}$$

Definition 3.2.7 (General Sign Consistency pg 2544 [19])**.** An estimate is general sign consistent if

$$\lim_{n\to\infty} \mathbb{P}(\exists \lambda \geq 0, \hat{\beta}^n(\lambda) =_s \beta^n) = 1 \tag{3.13}$$

Strong sign consistency means that we can use a predefined λ as a function of n to achieve model selection consistency, whereas general sign consistency means that during some random realization, there exists a λ that achieves model selection consistency. Note, both definitions imply model selection consistency.

Definition 3.2.8 (The Irrepresentable Conditions pg 2544 [19])**.** Assuming C_{11}^n is invertible, the (weak) irrepresentable condition is:

$$|C_{21}^n (C_{11}^n)^{-1} \text{sign}(\beta_{(1)}^n)| < \mathbf{1} \tag{3.14}$$

where $\mathbf{1} \in \mathbb{R}^{p-q}$ is a vector of 1's and the inequality holds element wise. Note, when the signs of the true β are unknown, the irrepresentable conditions (weak and strong) become

$$|C_{21}^n (C_{11}^n)^{-1}| < \mathbf{1} - \eta \tag{3.15}$$

for some $\eta > 0$.

In this report, we will refer to condition 3.15 as the 'irrepresentable condition', since, we do not know the signs of the true β in application. Symbolically, this means that the modulus row sums of the left side matrix must all be strictly less than 1. We can verify this by computing $\|C_{21}^n (C_{11}^n)^{-1}\|_\infty$ and checking it is < 1.

Example 3.2.2. Let

$$C^n = \begin{pmatrix} 1 & 2 & 3 & 0.5 \\ 2 & 1 & 0 & 0 \\ 3 & 0 & 1 & 0.1 \\ 0.5 & 0 & 0.1 & 1 \end{pmatrix}, \quad C_{11}^n = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 1 & 0 \\ 3 & 0 & 1 \end{pmatrix}, \quad C_{21}^n = \begin{pmatrix} 0.5 & 0 & 0.1 \end{pmatrix}$$

Here, we have 3 true, non-zero $\beta_j, j = \{1, 2, 3\}$, and 1 zero β_4 weakly associated to the true β_j. Assume, we do not know the signs of β.

$$\|C_{21}^n (C_{11}^n)^{-1}\|_\infty = \left\| \tfrac{-1}{60} \quad \tfrac{1}{30} \quad \tfrac{3}{20} \right\|_\infty = \frac{1}{6} < 1$$

So the irrepresentable condition is satisfied.

Example 3.2.3. Let

$$C^n = \begin{pmatrix} 1 & 0 & 0 & 4 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 1 & 2 \\ 4 & 3 & 2 & 5 \end{pmatrix}, \quad C_{11}^n = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad C_{21}^n = \begin{pmatrix} 4 & 3 & 2 \end{pmatrix}$$

We use the same β formulation as above, but now β_4 strongly associated to the true β_j. Assume, we do not know the signs of β.

$$\|C_{21}^n (C_{11}^n)^{-1}\|_\infty = \begin{Vmatrix} 4 & 3 & 2 \end{Vmatrix}_\infty = 9 > 1$$

Hence, the irrepresentable condition is not satisfied.

These two examples give some intuitive insight on when the irrepresentable condition holds. It seems that if the irrelevant predictors are too strongly associated with the relevant ones, the condition will not hold. In fact, "the total amount of an irrelevant covariates represented by the covariates in the model is not to reach 1, (therefore the name 'irrepresentable') " [19]. We now state the main selection consistency result without proof.

Theorem 3.2.2 (Lasso Selection Consistency [19])**.** *Let x_i represent an observed row of X. Let p, q and $\beta^n = \beta$ be fixed. The latter means that the true β is fixed regardless of the sample size. Under the regularity conditions:*

1. *$C^n \to C$, where C is positive definite.*
2. *$\frac{1}{n} max_{1 \le i \le n}((x_i^n)^T x_i^n) \to 0$ as $n \to \infty$, where (x_i^n) represents the i^{th} sample, given a sample size of n.*

We have that, the Lasso is general sign consistent only if $\exists N$ such that the weak irrepresentable condition holds for all $n > N$.

Intuitively, the first regularity condition says that the population has a fixed covariance and that as we increase sample size our sample covariance will tend to the population covariance. The second regularity condition states that $\text{Cov}_{sc}(X_i, X_i) = \text{Var}_{sc}(X_i) \to 0$ as $n \to \infty$. This means that, having more data is beneficial since the uncertainty of our parameter observations tends to 0, as n is increased. For our application, we only require general sign consistency. Note, this theorem does not guarantee that we will select the correct λ when we fit the model, even if it exists.

Definition 3.2.9 (Principal Components Decomposition of $X^T X$)**.** Let $X \in \mathbb{R}^{n \times p}$. Consider the singular value decomposition of X.

$$X = UZP^T$$

where, $U \in \mathbb{R}^{p \times p}, P \in \mathbb{R}^{n \times n}$ and both matrices are orthogonal. $Z \in \mathbb{R}^{n \times p}$ a rectangular diagonal matrix. Note all matrices can be decomposed in this way [37]. We write the Gram Matrix as:

$$\begin{aligned} X^T X &= PZ^T U^T UZP^T \\ &= PDP^T \end{aligned}$$

where $D \in \mathbb{R}^{p \times p}$. The values $\sigma_i \ge 0$ on the diagonal of D are called the principal components. We specify them and generate P randomly. This produces a positive semi definite matrix which we can scale and center to get C^n and the covariance Σ. We can also set some σ_i to be 0 to introduce linear dependence, hence generating multicollinearity.

We now present a simulation based on the algorithm below to investigate the relationship between the irrepresentable condition and selection consistency empirically.

Algorithm 1 Irrepresentable Condition Simulation

1. Generate 50 different designs (covariance matrices) with 200 parameters using the principal components decomposition.
2. Calculate $\eta = 1 - \|C_{21}^n (C_{11}^n)^{-1}\|_\infty$ for each design and note the value. If $\eta < 0$ then the irrepresentable condition is not satisfied.
3. For each design run 50 iterations of the following:
   - Generate data based on the design. $X \sim N(0, \Sigma)$
   - Set the true β_j as $\{1, 4, -6, -2, 2\}$ for $j = 1, ..., 5$ and let the rest be 0.
   - $\epsilon \sim \mathcal{N}(0, 1)$
   - $Y = X\beta + \epsilon$
4. Record how many times the Lasso is general sign consistent by identifying if there exists a λ in the Lasso path which gives coefficients of the right sign.

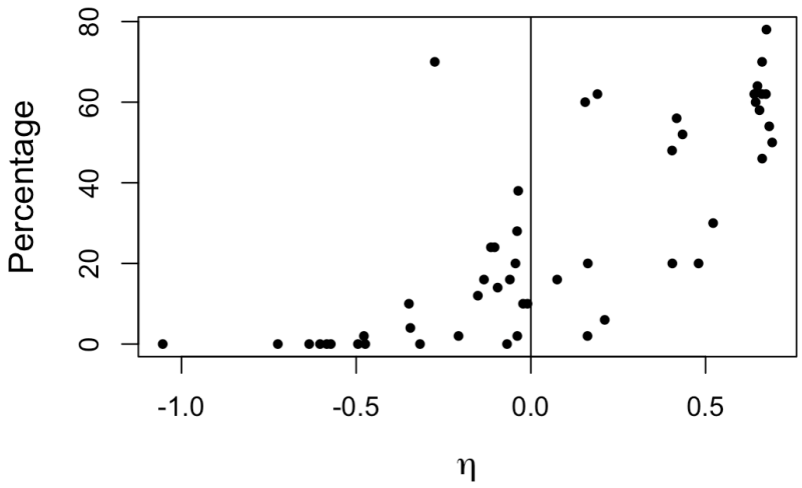

Figure 3.4: A scatter plot comparing the percentage general sign consistency on the y-axis, with how strongly the irrepresentable condition is satisfied. If $\eta > 0$, the condition is satisfied.

Figure 3.4 shows an increasing trend of sign consistency as η increases, suggesting that the irrepresentable condition is necessary for model selection consistency in practice as well. With microarray data, the irrepresentable condition is often violated as the genes are strongly dependent on each other. Hence, the Lasso may not be an ideal method for our application.

3.2.2 Identical Predictors

Below we explore a further property of the Lasso estimates under multicollinearity. In this section, we will use a few common results from convex analysis. More details on these can be found in Section 4.1.

Definition 3.2.10 (Grouping Effect pg 306 [38])**.** A regression method exhibits a grouping effect if a group of strongly correlated variables have almost equal coefficients. In the case where we have identical predictors, the coefficients should be (theoretically) equal.

In practice, estimated coefficients will rarely be equal due to other factors in the computation process such as floating point errors, so we only look for predictors with similar coefficients.

Consider the scenario where two variables are strongly pairwise correlated. We present a result in the case that these two variables are identical and expand on the proof given in [38].

Theorem 3.2.3 (Lasso and Identical Variables [38])**.** *Assume we have two identical columns in the data. I.e, let* $\mathbf{x}^i = \mathbf{x}^j \in \mathbb{R}^n, i, j, \in \{1, ..., p\}$*. Let* $\hat{\beta}$ *represent the minimiser of our penalised regression problem.*

(i) Let $P(\cdot)$ *be an arbitrary penalty function which is positive for* $\beta \neq 0$*. If* P *is a strictly convex function, then,* $\hat{\beta}_i = \hat{\beta}_j, \forall \lambda > 0$*.*

(ii) Now, if $P(\cdot)$ *is* $\|\beta\|_1$*, then* $\hat{\beta}_i \hat{\beta}_j \geq 0$ *and* $\hat{\beta}^*$ *is another minimiser of the Lasso problem where*

$$\hat{\beta}_k^* = \begin{cases} \hat{\beta}_k & \text{if } k \neq i \text{ and } k \neq j \\ (\hat{\beta}_i + \hat{\beta}_j) \cdot (s) & \text{if } k = i \\ (\hat{\beta}_i + \hat{\beta}_j) \cdot (1 - s) & \text{if } k = j \end{cases}$$

for any $s \in [0, 1]$*.*

Proof. For part (i), fix $\lambda > 0$ and assume $\hat{\beta}_i \neq \hat{\beta}_j$. Now consider, $\tilde{\beta}$ where

$$\tilde{\beta}_k = \begin{cases} \hat{\beta}_k & \text{if } k \neq i, k \neq j \\ \frac{1}{2}(\hat{\beta}_i + \hat{\beta}_j) & \text{if } k = i \text{ or } k = j \end{cases}$$

$$X\tilde{\beta} = \begin{bmatrix} \sum_{k=1, k\neq i, k\neq j}^{p} \left(x_{1k}\hat{\beta}_k\right) + x_{1i}\frac{1}{2}(\hat{\beta}_i + \hat{\beta}_j) + x_{1j}\frac{1}{2}(\hat{\beta}_i + \hat{\beta}_j) \\ \vdots \\ \sum_{k=1, k\neq i, k\neq j}^{p} \left(x_{nk}\hat{\beta}_k\right) + x_{ni}\frac{1}{2}(\hat{\beta}_i + \hat{\beta}_j) + x_{nj}\frac{1}{2}(\hat{\beta}_i + \hat{\beta}_j) \end{bmatrix} = X\hat{\beta}$$

since $\mathbf{x}^i = \mathbf{x}^j$. Hence, $\|y - X\tilde{\beta}\|_2^2 = \|y - X\hat{\beta}\|_2^2$. Now, since P is strictly convex,

$$\begin{aligned} P(\tilde{\beta}) &= P\left(\frac{1}{2}(\hat{\beta}_1, ..., \hat{\beta}_i, \hat{\beta}_j, ..., \hat{\beta}_p) + \frac{1}{2}(\hat{\beta}_1, ..., \hat{\beta}_j, \hat{\beta}_i, ..., \hat{\beta}_p)\right) \\ &< \frac{1}{2}P(\hat{\beta}_1, ..., \hat{\beta}_i, \hat{\beta}_j, ..., \hat{\beta}_p) + \frac{1}{2}P(\hat{\beta}_1, ..., \hat{\beta}_j, \hat{\beta}_i, ..., \hat{\beta}_p) \\ &= P(\hat{\beta}) \end{aligned}$$

Input permutations of the β_j components have no effect on the penalty as every component of the penalty $P_j(\cdot)$ is the same function but applied on a different β_j. This implies that $\hat{\beta}$ is not the minimiser of our problem, so we have a contradiction. Hence $\hat{\beta}_i = \hat{\beta}_j$. For part (ii) assume $\hat{\beta}_i \hat{\beta}_j < 0$. Using the same $\tilde{\beta}$ as above, consider,

$$\|\tilde{\beta}\|_1 = \sum_{k\neq i, k\neq j} \left(|\hat{\beta}_k|\right) + \frac{1}{2}|\hat{\beta}_i + \hat{\beta}_j| + \frac{1}{2}|\hat{\beta}_i + \hat{\beta}_j| < \sum_{k=1}^{p} |\hat{\beta}_k| = \|\hat{\beta}\|_1$$

since, $\hat{\beta}_i$ and $\hat{\beta}_j$ are of different signs. But this contradicts our assumption that $\hat{\beta}$ is the minimiser of our Lasso problem. So, $\hat{\beta}_i \hat{\beta}_j \geq 0$. Through a similar reasoning as part (i) we see that $\|y - X\hat{\beta}^*\|_2^2 = \|y - X\hat{\beta}\|_2^2$. Hence,

$$\|\hat{\beta}^*\|_1 = \sum_{k\neq i, k\neq j} \left(|\hat{\beta}_k|\right) + s|\hat{\beta}_i + \hat{\beta}_j| + (1 - s)|\hat{\beta}_i + \hat{\beta}_j| = \sum_{k=1}^{p} |\hat{\beta}_k| = \|\hat{\beta}\|_1$$

So, the Lasso is not theoretically guaranteed to give equal estimates for identical predictors. □

Lemma 3.2.4. *The Ridge penalty is strictly convex and hence guaranteed to have the grouping effect when there are identical predictors.*

Proof. Let $P(\beta) = \lambda\|\beta\|_2^2$. Consider the Hessian of P

$$\frac{\partial^2 P}{\partial \beta^T \partial \beta} = 2\lambda I > 0$$

since $\lambda > 0$. By Lemma 4.1.2, we have strict convexity and the grouping effect follows from the above theorem. □

Example 3.2.4 (Comparison of Ridge and Lasso on Almost Equal and Equal Variables)**.** In this example, we compare Ridge and Lasso Estimates on 2 identical predictors and 2 heavily correlated predictors. We generate our data as follows:

1. $X \sim \mathcal{N}(0, I)$ with $n = 30$, $p = 3$
2. Let $\mathbf{x}^4 = \mathbf{x}^3$, and let $\mathbf{x}^{4\text{near}} = \mathbf{x}^3 + 0.5$
3. Combine X and $\mathbf{x}^4$ to get X_{equal}. Combine X and $\mathbf{x}^{4\text{near}}$ to get X_{near}.

Then, we let $\beta = \{1, -3, 4, 4\}$, $\epsilon \sim \mathcal{N}(0, 1)$, $Y = X\beta + \epsilon$ and apply the Lasso and Ridge on X_{near} and X_{equal}. We round our estimates to 3 decimal places.

β	Lasso β_{Equal}	Ridge β_{Equal}	Lasso β_{Near}	Ridge β_{Near}
1	0.932	1.140	0.905	1.152
-3	-2.715	-2.600	-2.689	-2.659
4	7.354	3.722	7.318	3.768
4	0.246	3.709	0.243	3.752

We observe that the Ridge estimates exhibit the grouping effect, as its estimates are very close. In our application, we want to identify which genes have an effect on the response. If a group of highly correlated genes contribute to an effect, the entire group should be selected. However, although the Ridge method has this good property, it does not give a sparse solution. Hence, we present the Elastic Net, which is a compromise between the Ridge and the Lasso methods and retains the desirable properties discussed so far from both methods.

3.3 Elastic Net

We first present the Naive Elastic Net below. In some texts, this is called the 'Elastic Net' [25] [39]. In this report we follow the definitions suggested by [38] and make clear the distinction between the Naive Elastic Net and the Elastic Net.

Definition 3.3.1 (Gaussian Naive Elastic Net pg 303 [38])**.**

$$\arg\min_{\beta \in \mathbb{R}^p} \|y - X\beta\|_2^2 + \lambda_2|\beta\|_2^2 + \lambda_1\|\beta\|_1 \tag{3.16}$$

where $\lambda_2, \lambda_1 > 0$. The Logistic case is similar, except that the RSS is switched out for the negative log likelihood. Let

$$\alpha = \frac{\lambda_1}{\lambda_1 + 2\lambda_2}, \lambda = \lambda_1 + 2\lambda_2$$

An alternative definition is

$$\arg\min_{\beta \in \mathbb{R}^p} \|y - X\beta\|_2^2 + \lambda\left(\frac{1-\alpha}{2}|\beta\|_2^2 + \alpha\|\beta\|_1\right) \tag{3.17}$$

with $\alpha \in [0,1]$ and $\lambda > 0$. This is precisely the equation that glmnet uses for getting a Naive Elastic Net estimate [39].

When $\alpha = 0$, we obtain the Ridge method and when $\alpha = 1$, we obtain the Lasso. From here onwards, we use the glmnet formulation. Note that the penalty is strictly convex as it is the sum of a strictly convex penalty and a convex penalty. Hence, it enjoys the grouping effect in the case of identical predictors.

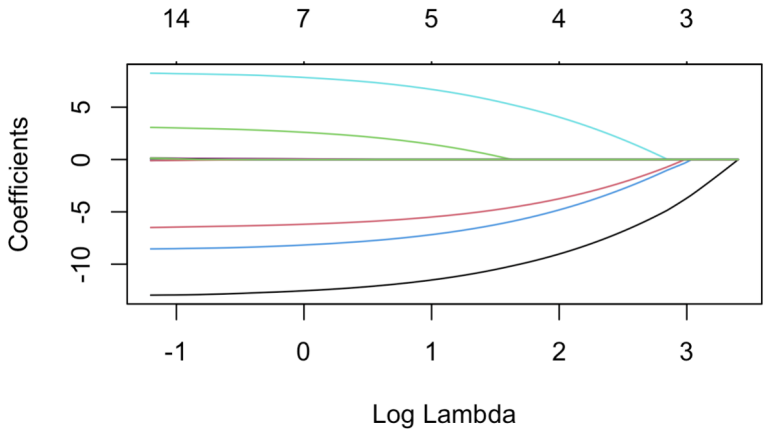

Figure 3.5: A plot of a Naive Elastic Net path computed with the same data as 3.1.

Lemma 3.3.1 (Gaussian Solution for Naive Elastic Net under Orthogonal design)**.**

$$\hat{\beta_j}^{elnet} = \frac{sign(\hat{\beta}_j)(|\hat{\beta}_j| - \alpha\lambda)_+}{1 + \lambda(1-\alpha)} \tag{3.18}$$

where, $\hat{\beta}_j$ is the OLS estimate.

Proof. We start with,

$$l_j = -\hat{\beta}_j\beta_j + \frac{1}{2}\beta_j^2 + \frac{\lambda(1-\alpha)}{2}\beta_j^2 + \alpha\lambda|\beta_j|$$

This can be found in a similar way to Lemma 3.2.1. For $\hat{\beta}_j \geq 0$ which implies $\beta_j \geq 0$,

$$\frac{\partial l_j}{\partial \beta_j} = -\hat{\beta}_j + \beta_j + \lambda(1-\alpha)\beta_j + \alpha\lambda$$

Setting the derivative to 0,

$$\hat{\beta_j}^{\text{elnet}} = \frac{(\hat{\beta}_j - \alpha\lambda)_+}{1 + \lambda(1-\alpha)} = \frac{sign(\hat{\beta}_j)(|\hat{\beta}_j| - \alpha\lambda)_+}{1 + \lambda(1-\alpha)}$$

Similarly, for $\hat{\beta}_j \leq 0$ which implies $\beta_j \leq 0$,

$$\frac{\partial l_j}{\partial \beta_j} = -\hat{\beta}_j + \beta_j + \lambda(1-\alpha)\beta_j - \alpha\lambda$$

Setting the derivative to 0,

$$\hat{\beta_j}^{\text{elnet}} = \frac{sign(\hat{\beta}_j)(|\hat{\beta}_j| - \alpha\lambda)_+}{1 + \lambda(1-\alpha)}$$

□

Definition 3.3.2 (Elastic Net pg 307 [38])**.**

$$\hat{\beta}(\text{Elastic Net}) = (1 + \lambda_2)\hat{\beta}(\text{Naive Elastic Net}) \tag{3.19}$$

Or alternatively,

$$\hat{\beta}(\text{Elastic Net}) = (1 + \lambda(1 - \alpha))\,\hat{\beta}(\text{Naive Elastic Net}) \tag{3.20}$$

The Elastic Net estimate is a re-scaling of the Naive estimate. The Naive method first shrinks the coefficients via Ridge regression and then once more through the Lasso. This produces excessive bias without much reduction in prediction variance [38]. The re-scaling undoes some of this "double shrinkage". Figure 3.6 shows the difference between the two versions of the Elastic Net. The scaling does not affect the parameters selected but the estimates instead.

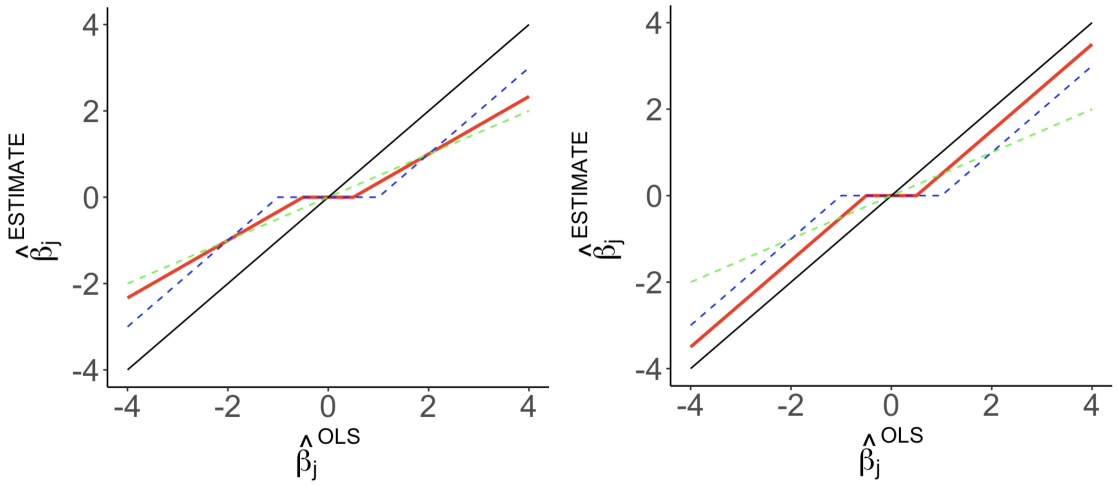

Figure 3.6: Shrinkage graphs for the Naive Elastic Net (left) and the Elastic Net (right) with the red line representing each respective estimate. The black line is the OLS estimate, the blue line is the Lasso estimate and the green line is the Ridge estimate. We take $\lambda = 1$ for all of the methods and $\alpha = 0.5$.

3.4 Adaptive Lasso

Definition 3.4.1 (Adaptive Lasso [17])**.** Let $\hat{\mathbf{w}} := \frac{1}{|\hat{\beta}|^\gamma}$, where $\hat{\beta}$ is an initial estimator.

$$P(\beta_j; \lambda, \gamma) = \lambda \sum_{j=1}^{p} \hat{w}_j |\beta_j| \tag{3.21}$$

where $\lambda > 0, \gamma > 0$.

Unlike the previous methods we have presented, the Adaptive Lasso is a two-step method. The first step consists of using an initial estimator $\hat{\beta}$ to gain a rough estimate of the parameters. We then use this estimator in the second stage to assign weights to the Lasso penalty. If $\hat{\beta}_j \to 0$, then $w_j \to \infty$. Therefore, if a parameter is found to be insignificant in the first step, it will be penalised more harshly in the second step. Likewise, if $\hat{\beta}_j \to \infty$ then $w_j \to 0$. γ controls how strongly our second step penalisation depends on our initial estimator. When $\gamma \to 0$ we get the Lasso estimates. If, $\gamma \to \infty$ then, all coefficients with small initial estimates between 0 and 1 will be shrunk to 0. Hence, we can calibrate the amount of shrinkage rather than uniformly penalising the parameters as in the Lasso.

There are a few choices for $\hat{\beta}$. Zou recommends that if $p \leq n$, we use OLS, otherwise if $p > n$ and the parameters are strongly correlated, we use Ridge [17] . For genomic applications, Algamal

and Lee proposed a custom initial estimator, CBPLR, dependent on pairwise correlations between different genes [16]. In this report, we will use Ridge coefficients as our initial estimator.

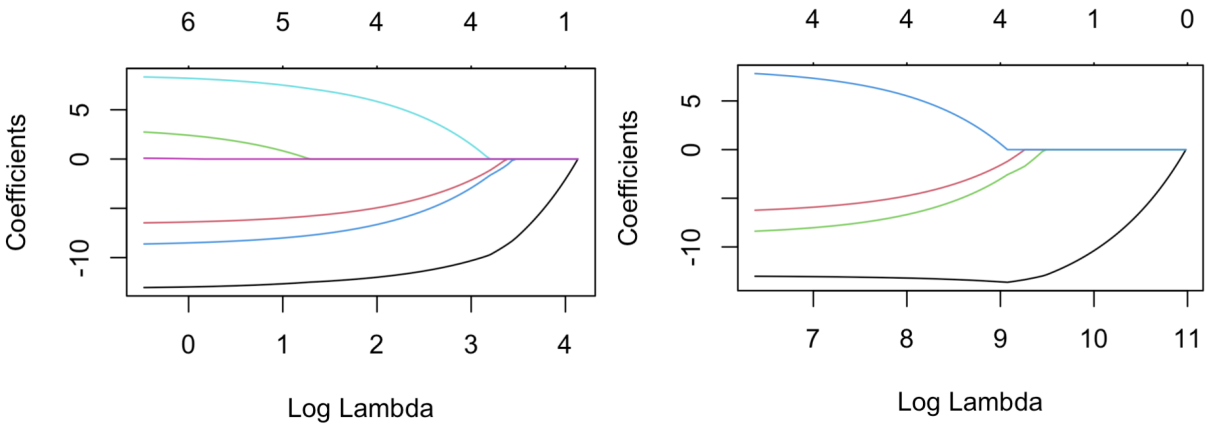

Figure 3.7: Adaptive Lasso paths using Ridge coefficients as the initial estimator with different γ values. On the left $\gamma = 0.5$ and on the right $\gamma = 2$. We use the same data as in Figure 3.2

Lemma 3.4.1 (Gaussian Adaptive Lasso solution under Orthogonal Design)**.**

$$\hat{\beta}_j^{\,Adap} = sign(\hat{\beta}_j)(|\hat{\beta}_j| - \hat{w}_j\lambda)_+ \tag{3.22}$$

Under a Ridge initial estimator, this is

$$\hat{\beta}_j^{\,Adap} = sign(\hat{\beta}_j)\left(|\hat{\beta}_j| - \frac{\lambda(1+\tilde{\lambda})^\gamma}{|\hat{\beta}_j|^\gamma}\right)_+ \tag{3.23}$$

where $\hat{\beta}_j$ is the OLS estimate, and $\tilde{\lambda}$ is the shrinkage parameter used to find the Ridge estimate.

We omit the proof, as it follows similarly to Lemma 3.2.1. In Figure 3.8 below, we see the shrinkage effects of the adaptive lasso using the same ridge initial estimators with varying γ.

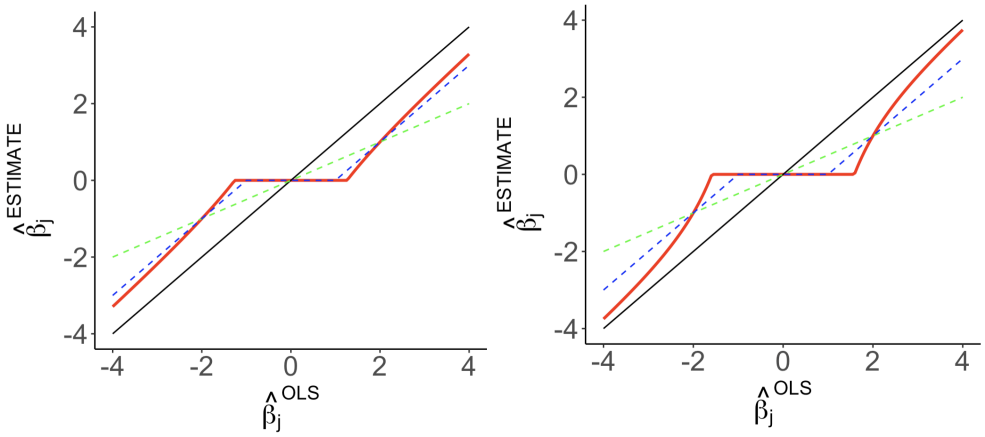

Figure 3.8: Two shrinkage graphs for the Adaptive Lasso, represented by the red lines. The black line represents OLS estimates, the green line represents Ridge estimates, the blue line represents Lasso estimates. We take $\lambda = 1$ for all the methods. On the left $\gamma = 0.5$ and on the right $\gamma = 2$.

3.4.1 Oracle Properties

Definition 3.4.2 (Convergence in distribution pg 352 [40])**.** We say that X_n converges to X in distribution, $X_n \xrightarrow{d} X$ if the respective cumulative functions converge to each other

$$\lim_{n\to\infty} F_n(x) = F(x) \tag{3.24}$$

for all $x \in \mathbb{R}^n$ such that the functions are continuous.

Definition 3.4.3 (Root-n/a_n Consistency [17])**.** We say an estimator is root-n consistent if

$$\sqrt{n}(\hat{\beta} - \beta) = O_p(1) \tag{3.25}$$

and a_n consistent if

$$a_n(\hat{\beta} - \beta) = O_p(1) \tag{3.26}$$

where a_n is a divergent sequence and $O_p(1)$ means bounded in probability.

Definition 3.4.4 (Oracle properties [17])**.** A procedure/method δ is an oracle procedure if the estimates $\hat{\beta}(\delta)$ satisfy:

1. Model Selection Consistency - same as Definition 3.2.4.
2. Optimal Estimation Rate.
$$\sqrt{n}\left(\hat{\beta}(\delta) - \beta(\delta)\right) \xrightarrow{d} \mathcal{N}(0, \Sigma)$$
where Σ is the covariance matrix of the true model.

The second property is a weaker version of the parameter estimation consistency presented in definition 3.2.3. As $n \to \infty, \sqrt{n}$ diverges. Hence, the property is saying that the differences between the estimates and the true values must be small enough such that they form a bell curve around 0.

Theorem 3.4.2 (Gaussian Adaptive Lasso satisfies the Oracle Properties. pg 1420 [17])**.** *Let $Y \sim \mathcal{N}(X\beta, \sigma^2 I)$ and assume $\frac{1}{n}X^T X \to C$. Also assume that, $\lambda_n \backslash \sqrt{n} \to 0$ and $\lambda_n n^{(\gamma-1)\backslash 2} \to \infty$. Further, assume that the initial estimate is root-n consistent. Then, the Adaptive Lasso satisfies the oracle properties with the second property being,*

$$\sqrt{n}\left(\hat{\beta}(\delta) - \beta(\delta)\right) \xrightarrow{d} \mathcal{N}(0, \sigma^2 C_{11}^{-1})$$

where C_{11} is the part of the gram matrix relating to the true model parameters.

We omit the proof as it is highly technical. The result does not require the irrepresentable condition. The main reason why the Adaptive Lasso is an oracle procedure is due to the weights. Since the initial estimator is assumed to be parameter estimation consistent, the weights will become more accurate as our sample size increases. In particular, as $n \to \infty$, $\hat{w}_j \to \infty$ for true $\beta_j = 0$ and $\hat{w}_j$ goes to some finite constant for true $\beta_j \neq 0$. Therefore, our second estimate will also be gradually more accurate given more samples.

The condition for root-n consistency can be relaxed to a_n consistency [17]. For low dimensional data $p \leq n$, this condition is easily satisfied, as we can take the initial estimator to be OLS. However, it is difficult to obtain an a_n consistent initial estimator for high dimensional data $p > n$. We use Ridge for our initial estimator, but it remains to be shown that Ridge estimates are a_n consistent [17]. So, we do not know if the oracle properties are satisfied for Ridge Adaptive estimates.

3.5 Folded Concave Penalties

In this section, we look at two penalties, SCAD and MCP. Unlike normed penalties, the penalty functions are not convex. Instead, they are symmetrically concave for $\beta_j > 0$ and $\beta_j < 0$. Hence, the name 'folded concave' penalties. We also define the penalty functions via a single component β_j, as $P(\beta; \lambda, \gamma) = \sum_{j=1}^{p} P(\beta_j; \lambda, \gamma)$.

Definition 3.5.1 (Smoothly Clipped Absolute Deviation (SCAD) Penalty [18])**.**
Let $j \in \{1, ..., p\}$.

$$P(\beta_j; \lambda, \gamma) = \begin{cases} \lambda|\beta_j| & \text{if } |\beta_j| \leq \lambda \\ \frac{2\gamma\lambda|\beta_j| - \beta_j^2 - \lambda^2}{2(\gamma-1)} & \text{if } \lambda < |\beta_j| < \gamma\lambda \\ \frac{\lambda^2(\gamma+1)}{2} & \text{if } |\beta_j| \geq \gamma\lambda \end{cases} \tag{3.27}$$

where $\lambda > 0, \gamma > 2$.

For small regression coefficients, $|\beta_j| \leq \lambda$, the penalty is equivalent to the Lasso. Afterwards, the penalty follows a quadratic curve, before tailing off to a constant. γ alters the concavity of the quadratic section. Since, the Lasso penalises all the coefficients by λ equally, it often biases large, relevant coefficients too excessively. The penalisation rate tails off for SCAD to minimise this problem. In terms of consistency, SCAD satisfies the oracle properties [18].

Definition 3.5.2 (Minimax Concave Penalty (MCP) [20])**.**

$$P(\beta_j; \lambda, \gamma) = \begin{cases} \lambda|\beta_j| - \frac{\beta_j^2}{2\gamma} & \text{if } |\beta_j| \leq \gamma\lambda \\ \frac{1}{2}\gamma\lambda^2 & \text{if } |\beta_j| > \gamma\lambda \end{cases} \tag{3.28}$$

where $\lambda > 0, \gamma > 1$.

MCP follows similarly to the SCAD penalty, but immediately begins with a quadratic penalisation curve before tailing off to a constant. It is still non-differentiable at β_j so a sparse solution is guaranteed. In Section 3.2.1, the Lasso was analysed to be selection inconsistent unless the irrepresentable condition was satisfied. MCP has been proved to be selection consistent without the need for the irrepresentable condition [20]. Furthermore, SCAD and Lasso may also eliminate relevant predictors with small coefficients. MCP alleviates this issue by relaxing the penalisation rate immediately.

Below, in Figures 3.9 and 3.10, we present of example of the regularisation paths of SCAD and MCP respectively, and a plot of the penalties in comparison to the Lasso.

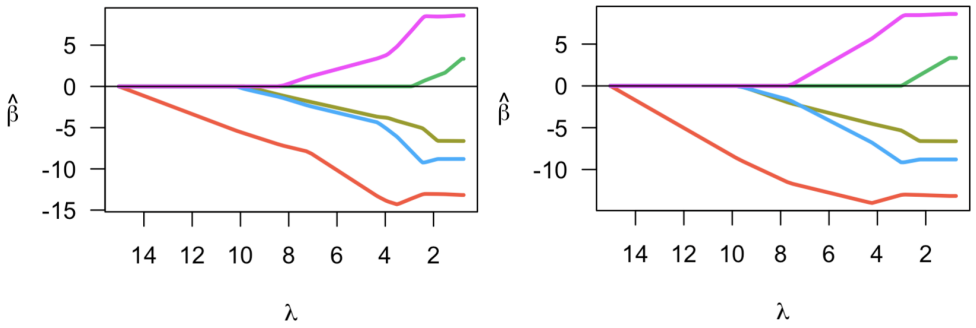

Figure 3.9: Regularisation paths for SCAD (left) and MCP (right). Computed with same data as in Figure 3.2. We use the default $\gamma = 3.7, \gamma = 3$ for SCAD and MCP, respectively.

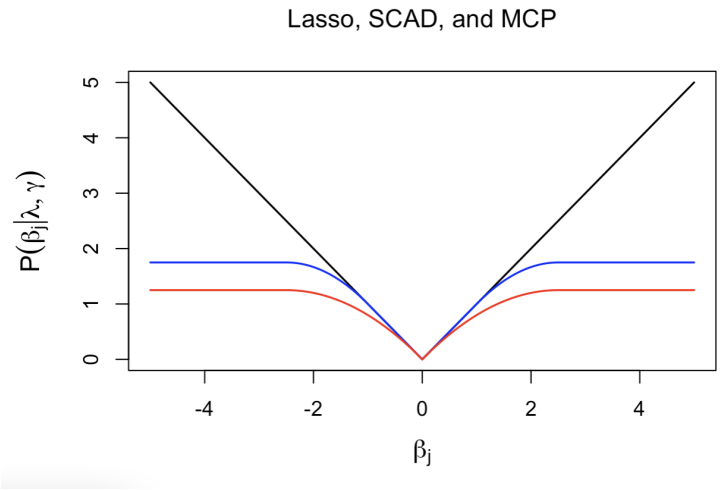

Figure 3.10: Plot of folded concave penalties against the Lasso penalty.
Black = Lasso, blue = SCAD, red = MCP. $\lambda = 1, \gamma = 2.5$.

Lemma 3.5.1 (Gaussian SCAD solution under orthogonal design. pg 1351 [18])**.** *Let our data X have orthonormal columns and let $\hat{\beta}_j$ represent the OLS estimate. Then we can write*

$$\hat{\beta}_j^{SCAD} = \begin{cases} sign(\hat{\beta}_j)(|\hat{\beta}_j| - \lambda)_+ & \text{if } |\hat{\beta}_j| \leq 2\lambda \\ \frac{(\gamma-1)\hat{\beta}_j - sign(\hat{\beta}_j)\gamma\lambda}{\gamma-2} & \text{if } 2\lambda < |\hat{\beta}_j| \leq \gamma\lambda \\ \hat{\beta}_j & \text{if } |\hat{\beta}_j| > \gamma\lambda \end{cases}$$

Proof. We prove this in a similar style to Lemma 3.2.1 but consider cases for $|\beta_j|$ as the penalty is piecewise.
Case 1: If $|\beta_j| \leq \lambda$, then we have the Lasso problem, and our SCAD estimate is naturally the same as the Lasso estimate. So,

$$\hat{\beta}_j^{\text{SCAD}} = sign(\hat{\beta}_j)(|\hat{\beta}_j| - \lambda)_+$$

Case 2: If $\lambda < |\beta_j| < \gamma\lambda$, then we wish to minimise

$$l_j = -\hat{\beta}_j\beta_j + \frac{1}{2}\beta_j^2 + \frac{2\gamma\lambda|\beta_j| - \beta_j^2 - \lambda^2}{2(\gamma-1)}$$

Consider $\hat{\beta}_j \geq 0$, which implies $\beta_j \geq 0$. Taking the derivative we get,

$$\begin{aligned} \frac{\partial l_j}{\partial \beta_j} &= -\hat{\beta}_j + \beta_j + \frac{\gamma\lambda - \beta_j}{\gamma - 1} \\ &= -\hat{\beta}_j + \frac{\gamma\beta_j + \gamma\lambda - 2\beta_j}{\gamma - 1} \end{aligned}$$

Setting the derivative to 0,

$$\hat{\beta}_j^{\text{SCAD}} = \frac{(\gamma-1)\hat{\beta}_j - \gamma\lambda}{\gamma - 2} = \frac{(\gamma-1)\hat{\beta}_j - sign(\hat{\beta}_j)\gamma\lambda}{\gamma - 2}$$

Now consider $\hat{\beta}_j \leq 0$, which implies $\beta_j \leq 0$. Similarly, we get,

$$\frac{\partial l_j}{\partial \beta_j} = -\hat{\beta}_j + \frac{\gamma\beta_j - \gamma\lambda - 2\beta_j}{\gamma - 1} = 0$$

$$\hat{\beta_j}^{\text{SCAD}} = \frac{(\gamma-1)\hat{\beta}_j + \gamma\lambda}{\gamma-2}$$
$$= \frac{(\gamma-1)\hat{\beta}_j - sign(\hat{\beta}_j)\gamma\lambda}{\gamma-2}$$

which proves the second case. We note $\gamma > 2$ for the estimate to be feasible. For case 3, $|\hat{\beta}_j| > \gamma\lambda$,

$$l_j = -\hat{\beta}_j + \beta_j + \frac{\lambda^2(\gamma+1)}{2}$$

After taking derivatives and rearranging, we find that,

$$\hat{\beta_j}^{\text{SCAD}} = \hat{\beta}_j$$

Now, notice we must match up the inequalities for these piece wise estimates to maintain continuity. Case 1 and Case 2 intersect at $|\hat{\beta}_j| = 2\lambda$. Case 2 and Case 3 intersect at $|\hat{\beta}_j| = \gamma\lambda$. This can be seen if one substitutes $\hat{\beta}_j = 2\lambda, \hat{\beta}_j = \gamma\lambda$ into Case 2. □

Lemma 3.5.2 (Gaussian MCP solution under orthogonal design)**.** *Let X have orthonormal columns and $\hat{\beta}_j$ represent OLS estimates. Then,*

$$\hat{\beta_j}^{MCP} = \begin{cases} \frac{\gamma sign(\hat{\beta}_j)(|\hat{\beta}_j|-\lambda)_+}{\gamma-1} & \text{if } |\beta_j| \leq \gamma\lambda \\ \hat{\beta}_j & \text{if } |\beta_j| > \gamma\lambda \end{cases} \tag{3.29}$$

We omit the proof for this result, but it can be derived via similar means to the Lasso or SCAD case. We see that, unless $\gamma > 1$, the first estimate won't be feasible. In Figure 3.11 we see shrinkage comparisons of SCAD and MCP against OLS with the same λ and γ. Note for large $|\beta_j|$ both estimates become the same as OLS.

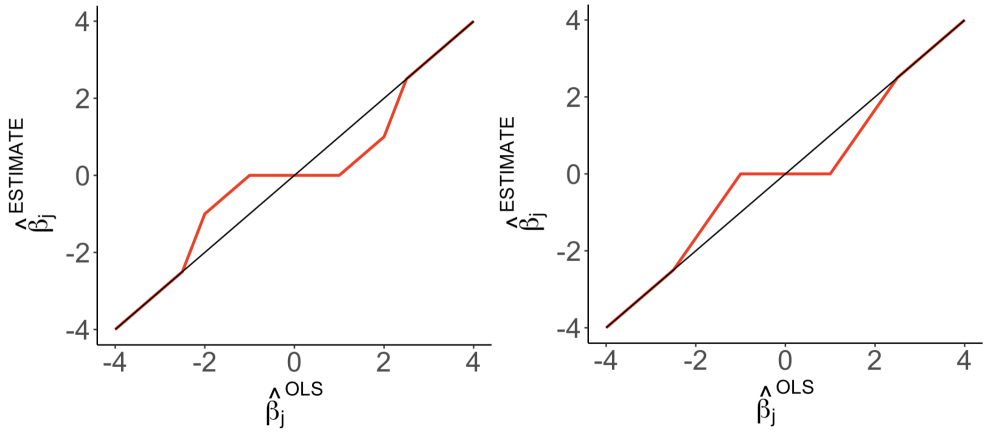

Figure 3.11: Shrinkage graphs for SCAD (left) and MCP (right). Similarly to Fig 3.3, the black line represents the OLS estimates and the red line represents the SCAD or MCP estimates. In both cases, we take $\lambda = 1, \gamma = 2.5$.

3.6 Stability

Above, we have presented 5 penalised regression methods which give sparse solutions. However, as we show is this section, these methods are not necessarily stable.

Definition 3.6.1 (Algorithm Output pg 2 [41])**.** We call the output of an algorithm fitted on the training data $\mathbb{L}_{(y,X)}$, where y is our response, and X is our data matrix.

In the case of regression, the solutions are functions of β. Now, we define what we mean by sparsity and stability.

Definition 3.6.2 (IRF pg 3 [41])**.** An estimator β^* identifies redundant features (IRF) of X if

$$\forall i \neq j, \mathbf{x}_i = \mathbf{x}_j \implies \beta_i^* \beta_j^* = 0. \tag{3.30}$$

In short, if two predictors are the same, the estimator must be 0 for one. An algorithm is IRF if there exists a $\beta^* \in \mathbb{L}_{(y,X)}$ which is IRF. In example 3.2.4, we see that the Lasso is IRF, whereas, Ridge is not. We say that an algorithm is sparse if it gives a model that has identified redundant features.

Definition 3.6.3 (Uniform Stability pg 2 [41])**.** Let $\mathcal{Z}$ be the space of responses and data points. Typically, $\mathcal{Z} \subset \mathbb{R}^{p+1}$. An algorithm has uniform stability ϵ_n with respect to a loss function l if $\forall y \in \mathbb{R}^n, \forall X \in \mathbb{R}^{n \times p}$, and $\forall i \in \{1, .., n\}$ the following holds:

$$\max_{z_k \in \mathcal{Z}} |l(\mathbb{L}_{(y,X)}, z_k) - l(\mathbb{L}_{(y,X)\setminus i}, z_k)| \leq \epsilon_n \tag{3.31}$$

where $z_k = (y_k, X_k)$, and X_k is the k^{th} row of X. Note, X and y refer to the training data whereas, z_k refers to both training data and test data. For Gaussian regression,

$$l(\mathbb{L}_{(y,X)}, z_k) = \|\mathbb{L}_{(y,X)}(X_k) - y_k\|_2^2 \tag{3.32}$$

The model fitted when the i^{th} observation is removed is $\mathbb{L}_{(y,X)\setminus i}$. In essence, the definition gives the maximum difference of loss functions on datasets with single differing observations. We say that an algorithm is stable if its uniform stability $\epsilon_n \to 0$ as $n \to \infty$.

We now make an assumption [41]. Given some j, data (y, X) and two estimators β^1 and β^2, suppose that $\beta^1 <_{(y,X)} \beta^2$ and $\beta_j^1 = \beta_j^2 = 0$. Namely, the algorithm outputs/prefers β^2 over β^1. Then, for any, new predictor $\hat{\mathbf{x}}_j$, we still have $\beta^1 <_{(y,\hat{X})} \beta^2$, where $\hat{X} = (\mathbf{x}_1, ..., \mathbf{x}_{j-1}, \hat{\mathbf{x}}_j, \mathbf{x}_{j+1}, ..., \mathbf{x}_p)$. This means that if a parameter is set to 0 between a pair of estimators, changing the data corresponding to that parameter does not change the preference ordering the algorithm has on the estimators.

Theorem 3.6.1 (Sparse Algorithms are not Stable pg 3 [41])**.** *If an algorithm satisfies the above assumption and is IRF, its uniform stability bound is bounded below by b_n and does not go to 0 as $n \to \infty$.*

We omit the proof since it is highly technical. This theorem states that all of our methods except Ridge and Naive Elastic Net are not stable. However, it does not quantify the extent of the instability. Therefore, we will explore the stability of our algorithms empirically in Section 5.4.

Chapter 4

Fitting the Model

In this chapter, we explore the theory behind the estimation process of β for the methods described in the previous chapter, and conclude with a summary on model diagnostics. We present a brief introduction on convex analysis before looking at coordinate descent and cross validation algorithms.

4.1 Convex Analysis

Definition 4.1.1 (Convex set pg 23 [42]). A set C is convex if for all elements $x_1, x_2 \in C$ and $t \in [0, 1]$,

$$tx_1 + (1-t)x_2 \in C$$

Clearly, the set $\mathbb{R}^p \ \forall \ p \in \mathbb{N}$ is a convex set.

Definition 4.1.2 (Convex function pg 67 [42]). Let X be some convex subset of a real vector space. A function $f : X \to \mathbb{R}$ is convex if for all elements $x_1, x_2 \in X$, and $t \in [0, 1]$,

$$f(tx_1 + (1-t)x_2) \leq tf(x_1) + (1-t)f(x_2)$$

If we have a strict inequality, we say that the function is strictly convex. Notice that the sum of two convex functions is convex.

Example 4.1.1 (All norms are convex). For all $u, v \in V$, where V is some vector space and for all $0 \leq t \leq 1$, we have

$$\|tu + (1-t)v\| \leq \|tu\| + \|(1-t)v\| \leq t\|u\| + (1-t)\|v\|$$

where we use the norm properties, triangle inequality and multiplication by a scalar constant for the first and second inequalities respectively.

Lemma 4.1.1 (First order convexity condition pg 69 [42]). *A differentiable function $f : \mathbb{R}^n \mapsto \mathbb{R}$ on some convex set is convex if and only if*

$$f(y) \geq f(x) + \nabla f(x)^T (y - x) \tag{4.1}$$

A proof can be found in [42]. Essentially, the condition says that all the points of the convex function must lie above the tangent at any given point. We provide a diagram below to illustrate this for one dimension.

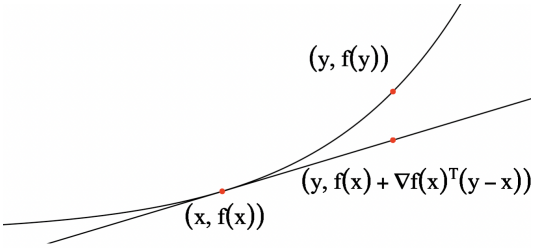

Figure 4.1: An illustration of the first order convexity condition for a univariate function.

Lemma 4.1.2 (Second-order Convexity Condition pg 71 [42]). *Assume f is twice differentiable. Then, f is convex if and only if its domain is a convex set and its Hessian is positive semi-definite.*

$$\nabla^2 f(x) \geq 0$$

If its Hessian is positive definite, then, we have strict convexity.

$$\nabla^2 f(x) > 0$$

The proof follows from the 1st order convexity condition.

Example 4.1.2 (Convexity of RSS). Let $f(\beta)$ be the RSS.

$$\begin{aligned} f(\beta) &= \|y - X\beta\|_2^2 \\ &= (y - X\beta)^T (y - X\beta) \\ &= y^T y - 2y^T X\beta + \beta^T X^T X\beta \end{aligned}$$

$$\frac{\partial^2 f}{\partial \beta^T \partial \beta} = 2X^T X$$

From the proof of Lemma 3.1.1, we see that the Hessian above is positive semi-definite. Hence, by the second-order convexity condition, the RSS is convex.

Example 4.1.3 (Convexity of Negative Logistic Log Likelihood). Let $f(\beta)$ be the negative logistic log likelihood.

$$\begin{aligned} f(\beta) &= \sum_{i=1}^{n} \left(-y_i \beta^T x_i + \log(1 + e^{\beta^T x_i}) \right) \\ \frac{\partial f}{\partial \beta} &= \sum_{i=1}^{n} \left(-y_i x_i + \frac{x_i e^{\beta^T x_i}}{1 + e^{\beta^T x_i}} \right) \\ \frac{\partial f}{\partial \beta^T \partial \beta} &= \sum_{i=1}^{n} \left(\frac{x_i x_i^T e^{\beta^T x_i x_i^T \beta}}{(1 + e^{\beta^T x_i})^2} \right) \\ &= XDX^T \end{aligned}$$

where $D_{ii} = \frac{\exp(\beta^T x_i x_i^T \beta)}{(1+\exp(\beta^T x_i))^2} \geq 0$ with $D \in \mathbb{R}^{n \times n}$ and $\sum_{i=1}^{n} x_i x_i^T = XX^T$ with $X \in \mathbb{R}^{p \times n}$. Now consider a vector $v \in \mathbb{R}^p$.

$$v^T XDX^T v = v^T XD^{\frac{1}{2}} D^{\frac{1}{2}} X^T v = (D^{\frac{1}{2}} X^T v)^T D^{\frac{1}{2}} X^T v = \|D^{\frac{1}{2}} X^T v\|_2^2 \geq 0$$

Therefore, XDX^T is positive semi-definite, so the negative logistic log likelihood is convex by the second order convexity condition.

For the Elastic Net, Ridge and Lasso, our optimisation problem is the sum of a convex loss function and a convex norm penalty. Hence, the minimisation function is convex. Next, we introduce the notion of subgradients which is a generalisation of derivatives and is useful for optimising convex functions which are not differentiable.

Definition 4.1.3 (Subgradients pg 4 [43])**.** A vector $g \in \mathbb{R}^n$ is a subgradient of $f : \mathbb{R}^n \mapsto \mathbb{R}$ at $x \in \text{dom } f$, that is the domain of f, if, $\forall y \in \text{dom } f$

$$f(y) \geq f(x) + g^T(y - x) \tag{4.2}$$

There may be multiple subgradients at a point x. The set of such subgradients of f at point x is called the subdifferential of x and is denoted $\partial f(x)$. A function is subdifferentiable at a point if there exists at least one subgradient at that point.

Example 4.1.4 (Subdifferential of $|x|$ at $x = 0$)**.** From the definition, we see that $\partial f(x)$ contains all $g \in \mathbb{R}$ such that

$$|y| \geq gy$$

Hence, $g \in [-1, 1]$.

Note, if f is convex and differentiable at x, $\partial f(x) = \{\nabla f(x)\}$, that is to say, its subgradient becomes its gradient [44].

Definition 4.1.4 (Minkowski Sum pg 197 [45])**.** Given two sets A and B

$$A + B \coloneqq \{a + b | a \in A, b \in B\} \tag{4.3}$$

Lemma 4.1.3 (Subgradients of sums [44])**.** *Suppose $f_1, ..., f_m$ are convex functions and $f = f_1 + ... + f_m$. Then,*

$$\partial f(x) = \partial f_1(x) + ... + \partial f_m(x) \tag{4.4}$$

Proof. Recall that g is a vector and m is a scalar.

$$\begin{aligned}
\partial f(x) &= \{g | f(z) \geq f(x) + g^T(z - x)\} \\
&= \{\frac{mg}{m} | f_1(z) + ... + f_m(z) \geq f_1(x) + ... + f_m(x) + \frac{mg^T}{m}(z - x)\} \\
&= \{\frac{g}{m} | f_1(z) \geq f_1(x) + \frac{g^T}{m}(z - x)\} \\
&+ ... + \{\frac{g}{m} | f_m(z) \geq f_m(x) + \frac{g^T}{m}(z - x)\} \\
&= \partial f_1(x) + ... + \partial f_m(x)
\end{aligned}$$

□

Example 4.1.5 (Subdifferential of the Elastic Net Penalty at $\beta_j = 0$.)**.** Let $f(\beta_j)$ be the penalty function at a single component β_j.

$$f(\beta_j) = \lambda \left(\frac{1 - \alpha}{2} \beta_j^2 + \alpha \beta_j \right)$$

We take the derivative of the Ridge component of the function to get the subdifferential.

$$\frac{\partial f_{\text{Ridge}}}{\partial \beta_j} = \lambda(1 - \alpha)\beta_j = 0$$

The Lasso subdifferential is $\{g | g \in [-\lambda\alpha, \lambda\alpha]\}$ based on the above example. Hence, using Lemma 4.1.3, the subdifferential of the Naive Elastic Net penalty at $\beta_j = 0$ is $\{g | g \in [-\lambda\alpha, \lambda\alpha]\}$.

Lemma 4.1.4 (Minimizing non-differentiable functions, page 3 [44]). *A point x is the minimiser of a function f (not necessarily convex) if and only if f is subdifferentiable at x and $0 \in \partial f(x)$, i.e 0 is a subgradient of $f(x)$.*

Proof. Note $\forall y \in$ dom f, $f(y) \geq f(x)$. This is equivalent to $f(y) \geq f(x) + 0^T(y - x)$, and hence f is subdifferentiable at x with $0 \in \partial f(x)$. Clearly, the converse is true as well. □

Note, if f is convex and differentiable at x, the condition $0 \in \partial f(x)$ reduces to $\nabla f(x) = 0$ [44].

4.2 Coordinate Descent

For the model fitting in this report, we will use the R packages glmnet [30][39] and ncvreg [31]. These packages estimate β using coordinate descent, which is a very fast algorithm [39].

Definition 4.2.1 (Coordinate-wise minimum, page 3 [46]). Let $e_i \in \mathbb{R}^n$ be the i^{th} standard basis vector. Then the coordinate-wise minimum of $f : \mathbb{R}^n \mapsto \mathbb{R}$ is defined such that

$$\forall d \in \mathbb{R} \text{ and } i \in \{1, ..., n\}, \quad f(x + de_i) \geq f(x) \tag{4.5}$$

Theorem 4.2.1. *If a function is the sum of a convex and differentiable function, as well as a convex but not necessarily differentiable function, its coordinate-wise minimum is also its global minimum.*

Proof. We prove this based on [46] and give a few more analytic details. Let $g(\beta)$ be the convex and differentiable, and let $h(\beta)$ be convex but not necessarily differentiable. Naturally, the theorem also holds if the second part is differentiable. Let

$$F(\beta) \coloneqq g(\beta) + h(\beta)$$

We want to find the β which minimises F. Note that for all $\beta \in \mathbb{R}^p$, g is convex and differentiable and h is convex but not differentiable.

Let β^* be our coordinate-wise minimum. Consider, for some $z \in \mathbb{R}^p$

$$\begin{aligned} F(z) - F(\beta^*) &= g(z) - g(\beta^*) + h(z) - h(\beta^*) \\ &\geq \nabla g^T(\beta^*)(z - \beta^*) + h(z) - h(\beta^*) \text{ as g is convex and differentiable} \\ &= \sum_{i=j}^{p} (\nabla_j g(\beta^*)(z_j - \beta_j^*) + h(z_j) - h(\beta_j^*)) \end{aligned} \tag{4.6}$$

Now, consider the problem in the view of each individual β_j^*. I.e. we optimise along one coordinate direction.

$$F(\beta_j^*) = g(\beta_j^*) + h(\beta_j^*)$$

Our assumption was that β^* was the coordinate wise minimum. Hence, we know that $F(\beta_j^*)$ is minimised at β_j^*. By Lemma 4.1.3 and Lemma 4.1.4, we see that

$$\begin{aligned} 0 &\in \partial F(\beta_j^*) \\ 0 &\in \{\nabla_j g(\beta^*)\} + \partial h(\beta_j^*) \end{aligned}$$

Now, denoting d to be the subgradients of $h(\beta_j^*)$ and γ to be $\nabla_j g(\beta^*)$, we see that

$$0 \in \{d + \gamma | h(z_j) \geq h(\beta_j^*) + (d + \gamma)^T(z_j - \beta_j^*)$$

Considering the element 0, we see that $d = -\gamma \in \partial h(\beta_j^*)$ and hence by definition of a subgradient, the following inequality must hold.

$$h(z_j) \geq h(\beta_j^*) - \nabla_j g(\beta^*)(z_j - \beta_j^*)$$

After rearranging, we see that each element of the sum in Equation 4.6 is ≥ 0, thus, $F(z) - F(\beta^*) \geq 0$ so β^* is the global minimum. □

This means that for the Lasso, Ridge and Elastic Net methods, the coordinate-wise minimum is also a global minimum, as the penalties are all convex, and the RSS and Negative Logistic Likelihood are convex and differentiable. Therefore, we can use coordinate descent to find the β which minimise the problem for specific λ and α.

Definition 4.2.2 (Coordinate Descent [46])**.** Our goal is to find the β that minimises the objective function $f(\cdot)$ for a fixed $\lambda > 0$. We start with some initial guess $\beta^{(0)}$. Next, on each k^{th} iteration, we solve the following:

$$\begin{aligned}
\beta_0^k &= \arg\min_{\beta_0} f(\beta_0, \beta_1^{k-1}, \beta_2^{k-1}, ..., \beta_p^{k-1}) \\
\beta_1^k &= \arg\min_{\beta_1} f(\beta_0^{k-1}, \beta_1, \beta_2^{k-1}, ..., \beta_p^{k-1}) \\
&\vdots \\
\beta_p^k &= \arg\min_{\beta_p} f(\beta_0^{k-1}, \beta_1^{k-1}, \beta_2^{k-1}, ..., \beta_p)
\end{aligned}$$

until the estimation difference on consecutive iterations is sufficiently small $|\beta^k - \beta^{k+1}| < \epsilon$. In essence, we calculate p optimisations on individual coordinates while fixing the rest. Once we calculate a β_i^k we immediately use it in finding the remaining β_i^k of that iteration. I.e. we update our β_i individually instead of waiting until we have finished the whole iteration.

Example 4.2.1. As a simple example, consider a 2 parameter Gaussian Ridge problem. Our objective function is

$$f(\beta_0, \beta_1, \beta_2) = \sum_{i=1}^{n} (y_i - \beta_0 - x_{i1}\beta_1 - x_{i2}\beta_2)^2 + \lambda(\beta_1^2 + \beta_2^2)$$

where β_0 is the intercept.

We generate $X \sim \mathcal{N}(0, I)$, $\epsilon \sim \mathcal{N}(0, 1)$ and $Y = X\beta + \epsilon$ with $n = 3$ observations and $p = 2$ parameters. Our values are rounded to 2 decimal places.

$$y = \begin{bmatrix} 0.83 \\ 1.43 \\ -1.92 \end{bmatrix} \quad X = \begin{bmatrix} -1.60 & -0.63 \\ -0.33 & 0.18 \\ 0.82 & -0.84 \end{bmatrix} \quad \beta = \begin{bmatrix} -1 \\ 2 \end{bmatrix} \quad \epsilon = \begin{bmatrix} 0.49 \\ 0.74 \\ 0.58 \end{bmatrix}$$

Now, we estimate β according to the coordinate descent algorithm using the objective function with $\lambda = 1$. We write down one iteration explicitly. Let k = 1, we set the initial estimate to be $\beta^{(0)} = (0, 0, 0)$.

$$\begin{aligned}
f(\beta_0, 0, 0) &= \sum_{i=1}^{n} (y_i - \beta_0)^2 \\
\frac{\partial f}{\partial \beta_0} &= -2 \sum_{i=1}^{n} (y_i - \beta_0)
\end{aligned}$$

$$
\begin{aligned}
\beta_0^{(1)} &= 0.1133 \\
f(0.1133, \beta_1, 0) &= \sum_{i=1}^{n}(y_i - \beta_0^{(1)} - x_{i1}\beta_1)^2 + \beta_1^2 \\
\frac{\partial f}{\partial \beta_1} &= -2\sum_{i=1}^{n} x_{i1}(y_i - \beta_0^{(1)} - x_{i1}\beta_1) + 2\beta_1 \\
\beta_1^{(1)} &= -0.7482 \\
f(0.1133, -0.7482, \beta_2) &= \sum_{i=1}^{n}(y_i - 0.1133 + 0.7482x_{i1} - \beta_2 x_{i2})^2 + (-0.7482)^2 + \beta_2^2 \\
\frac{\partial f}{\partial \beta_2} &= -2\sum_{i=1}^{n} x_{i2}(y_i - 0.1133 + -0.7482x_{i1} - x_{i2}\beta_2) + 2\beta_2 \\
\beta_2^{(1)} &= 0.7906
\end{aligned}
$$

After doing a few more iterations, we see that there is a recurrence relation between the estimates of different iterations.

$$
\begin{aligned}
\beta_0^{(k+1)} &= \frac{1}{n}\sum_{i=1}^{n}(y_i - x_{i1}\beta_1^{(k)} - x_{i2}\beta_2^{(k)}) \\
\beta_1^{(k+1)} &= \sum_{i=1}^{n} x_{i1}(y_i - \beta_0^{(k+1)} - x_{i2}\beta_2^{(k)}) \backslash \sum_{i=1}^{n} x_{i1}^2 + 1 \\
\beta_2^{(k+1)} &= \sum_{i=1}^{n} x_{i2}(y_i - \beta_0^{(k+1)} - x_{i1}\beta_1^{(k)}) \backslash \sum_{i=1}^{n} x_{i2}^2 + 1
\end{aligned}
$$

We run 100 iterations and observe that the algorithm converges to $\hat{\beta} = (0.18, -0.71, 0.84)$. Note, since the data is not standardised, the estimates will not coincide with the results of glmnet Ridge.

For SCAD and MCP, the penalties are non-convex so, in general, the coordinate descent algorithm will not give the global minimum. However, it is possible to constrain the values of γ and λ such that the convexity of the Gaussian/Logistic loss functions 'overcomes' the concavity of the penalty. We will not explore the specifics here, but more details can be found in [31] [20].

4.2.1 Gaussian Algorithms

Now we derive the coordinate descent algorithms for the Naive Elastic Net, SCAD and MCP. Adaptive Lasso, Lasso, and Ridge use the same fitting process as the Naive Elastic Net but with different α and penalty factors. Assume that the data matrix X is standardised. Assume also that the response Y is standardised, which implies that there is no intercept. The case for which Y is not standardised follows similarly.

$$
\sum_{i=1}^{n} x_{ij} = 0, \quad \frac{1}{n}\sum_{i=1}^{n} x_{ij}^2 = 1, \quad \sum_{i=1}^{n} y_i = 0
$$

Under these assumptions we see that the least squares solution $z_j \in \mathbb{R}$ to a regression problem with a single predictor $\mathbf{x}_j \in \mathbb{R}^n$ is

$$
z_j = \frac{1}{n}\mathbf{x}_j^T y \text{ since } \mathbf{x}_j^T \mathbf{x}_j = n
$$

Since the univariate case is orthogonal we can write the estimates for Naive Elastic Net, MCP and SCAD in the closed forms as presented in Section 3.

$$\hat{\beta}_j^{\,elnet} = \frac{sign(z_j)(|z_j| - \alpha\lambda)_+}{1 + \lambda(1-\alpha)}$$

$$\hat{\beta}_j^{\,scad} = \begin{cases} sign(z_j)(|z_j| - \lambda)_+ & \text{if } |z_j| \leq 2\lambda \\ \frac{(\gamma-1)z_j - sign(z_j)\gamma\lambda}{\gamma-2} & \text{if } 2\lambda < |z_j| \leq \gamma\lambda \\ z_j & \text{if } |z_j| > \gamma\lambda \end{cases}$$

$$\hat{\beta}_j^{\,mcp} = \begin{cases} \frac{\gamma sign(z_j)(|z_j| - \lambda)_+}{\gamma-1} & \text{if } |z_j| \leq \gamma\lambda \\ z_j & \text{if } |z_j| > \gamma\lambda \end{cases}$$

Now consider the objective function for unpenalised Gaussian regression. Note, $\beta \in \mathbb{R}^p$ and does not include an intercept.

$$f(\beta) = \frac{1}{2N}\sum_{i=1}^{n}(y_i - \sum_{l\neq j}^{p} x_{il}\beta_l - x_{ij}\beta_j)^2$$

Definition 4.2.3 (Partial Residuals [31])**.** Let $-j$ denote the portion of a matrix or vector that remains after column or element j has been removed. We define the partial residuals $r_{-j} \in \mathbb{R}^n$ of $\mathbf{x}_j$ as:

$$r_{-j} = y - X_{-j}\beta_{-j} \tag{4.7}$$

where each individual element can be written as

$$r_{i(-j)} = y_i - \sum_{l\neq j}^{p} x_{il}\beta_l \tag{4.8}$$

Naturally, $\mathbf{r} = Y - X\beta$.

Now, consider the j^{th} step of the m^{th} iteration of the coordinate descent algorithm. Let $\tilde{\beta}$ represent the most recently updated estimates and more specifically let $\tilde{\beta}_j^{(m)}$ represent the current estimate of β_j obtained prior to iteration m. We wish to find $\beta_j^{(m+1)}$. We write the objective function in terms of partial residuals, assuming that they have been evaluated at the current estimates β_{-j} and take the derivative.

$$f(\beta) = \sum_{i=1}^{n}(r_{i(-j)} - x_{ij}\beta_j)^2$$
$$\frac{\partial f}{\partial \beta_j} = \frac{1}{N}\sum_{i=1}^{n} x_{ij}(r_{i(-j)} - x_{ij}\beta_j)$$

Therefore, the coordinate-wise minimiser of the unpenalised least squares is:

$$\frac{1}{N}\sum_{i=1}^{n} x_{ij}r_{i(-j)} = \frac{1}{N}\sum_{i=1}^{n} x_{ij}^2 z_j$$
$$z_j = \frac{1}{N}\sum_{i=1}^{n} x_{ij}r_{i(-j)}$$
$$= \frac{1}{N}\sum_{i=1}^{n} x_{ij}\left(y_i - \sum_{l=1}^{p} x_{il}\tilde{\beta}_l + x_{ij}\tilde{\beta}_j^m\right)$$

$$= \frac{1}{N}\mathbf{x}_j^T\mathbf{r} + \tilde{\beta}_j^m$$

Then, the penalised coordinate-wise minimisers for the Naive Elastic Net, SCAD and MCP are:

$$\hat{\beta}_j^{elnet}(z_j), \quad \hat{\beta}_j^{scad}(z_j), \quad \hat{\beta}_j^{mcp}(z_j)$$

Hence, the general form of the coordinate descent algorithm for Gaussian penalised regression can be written in a few steps. Find the unpenalised coordinate wise least squares solution evaluated at the current estimates. Penalise it using the univariate closed form solution of the corresponding method. Update the new coefficient with this result and update the residuals so that they are evaluated on the new current estimates. This is summarised in Algorithm 2 below.

Algorithm 2 Coordinate Descent for Gaussian Penalised Regression [31]

On iteration m:

1. Calculate $z_j = \frac{1}{N}\mathbf{x}_j^T\mathbf{r} + \tilde{\beta}_j^m$
2. Update $\beta_j^{(m+1)} \leftarrow \hat{\beta}_j^{estimate}(z_j)$
3. Update $\mathbf{r} \leftarrow \mathbf{r} - (\beta_j^{(m+1)} - \beta_j^{(m)})\mathbf{x}_j$

where $\hat{\beta}_j^{estimate}$ is a penalised coordinate-wise minimiser.

Remark 4.2.1. Here, we have assumed that both X and y are standardised. In glmnet X is by default standardised, however y is not [39]. For ncvreg, both X and y are by default standardised [31]. The estimates are then unstandardised to introduce the intercept.

4.2.2 Logistic Algorithms

For Logistic regression, the response is binary so, standardising Y is no longer possible, and therefore we will have an intercept. Note, one does not penalise the intercept. Consider the objective function for unpenalised logistic regression:

$$f(\beta_0, \beta) = \frac{1}{2N}\sum_{i=1}^{N} y_i(\beta_0 + x_i^T\beta) - \log(1 + e^{\beta_0 + x_i^T\beta})$$

Typically, β is estimated through maximising this function through Iteratively Reweighted Least Squares (IRLS) [47]. On iteration m, the solution is:

$$\beta_{m+1} = (X^T W_m X)^{-1} X^T W_m Q_m$$

and since we use logit link, our weights and working observations are as follows [39]:

$$\begin{aligned} W_m &= \mu(1-\mu) \\ Q_m &= X\beta_m + D_m^{-1}(Y - \mu) \\ D_{m(ii)}^{-1} &= (h'(\eta_i))^{-1} = \frac{1}{\mu_i(1-\mu_i)} \end{aligned}$$

where $\mu \in \mathbb{R}^n$ with components $\mu_i = h(\eta_i)$. This is equivalent to minimising the weighted least squares problem every iteration, where, $q_i = x_i^T\beta_m + D_{m(ii)}^{-1}(y_i - \mu_i)$ and $w_i = \mu_i(1-\mu_i)$

$$\sum_{i=1}^{n} w_i(q_i - x_i^T\beta)^2$$

Hence, we can evaluate the weights and working observations at the current estimates and then solve the penalised problem and using coordinate descent as above on each iteration. Specifically, on step j of iteration, m our coordinate descent objective function is:

$$\arg\min_{\beta_j} \sum_{i=1}^{n} \tilde{w}_i \ (\tilde{q}_i - \tilde{\beta}_0^{(m)} - \sum_{l \neq j}^{p} x_{il}\tilde{\beta}_l^{(m)} - x_{ij}\beta_j)^2 + P(\tilde{\beta}_1, ..., \tilde{\beta}_{j-1}, \beta_j, \tilde{\beta}_{j+1}, ..., \tilde{\beta}_p; \theta)$$

We summarise the steps in Algorithm 3.

Algorithm 3 Coordinate descent for Logistic Penalised Regression [31]

On iteration m:

1. Update the unpenalised weighted least squares problem with the current estimates.
2. Run coordinate descent on the penalised weighted least squares problem.

4.3 Cross Validation (CV)

In this section, we aim to present a method to find sets of tuning parameters θ that give the 'best' model. We first present the definitions of two errors.

Definition 4.3.1 (Test/Generalization Error pg 220 [25])**.** The prediction error over an independent test sample $\mathcal{T}$ is

$$Err_{\mathcal{T}} = \mathbb{E}[L(Y, \hat{f}(X)|\mathcal{T}] \tag{4.9}$$

where $\hat{f}(X)$ represents the fitted model, and $L(\cdot)$ the loss function. The loss function quantifies how far the fitted values deviates from the true response values. Two of the most common loss functions are the squared and absolute error functions.

Definition 4.3.2 (Expected prediction/test error pg 220 [25])**.**

$$Err = \mathbb{E}[Err_{\mathcal{T}}] \tag{4.10}$$

where, the expectation averages over any random variable including randomness in the training set that produces the model $\hat{f}$.

When fitting the model, we must set our tuning parameters to some value to control the amount of shrinkage of our parameters. To do this we may consider splitting our data up into 3 components:

- Training set: To fit the model based on a set of values for the tuning parameters.
- Validation set: To assess the predictive accuracy of the fitted model by estimating the expected prediction error.
- Test set: To estimate the generalization error of the final selected model based on the "optimal" θ. This is strictly separate from the training and validation set and only used at the end. One can compare the predictive accuracy of different methods on this test set.

A general rule is to allocate data in proportions 50%, 25%, 25% respectively. However, with genomic data, we often have a small sample size, e.g. < 100 observations. Setting aside 25 samples for the test set, we are left with only 75 samples, which is not enough to train and validate over a grid of tuning values. Therefore, we introduce a resampling method.

In K-fold cross validation (CV), K groups are defined and data is randomly allocated into them such that each group has roughly the same number of samples. Then, for $k = 1, ..., K$, the k^{th} group is taken out set to be the validation set while the model is trained on the rest of the data. Hence, each observation is used $K - 1$ times in the training process and once in the validating

process. Note, we use cross validation on our data after the test set has been removed, so that we can compare our model with others later on.

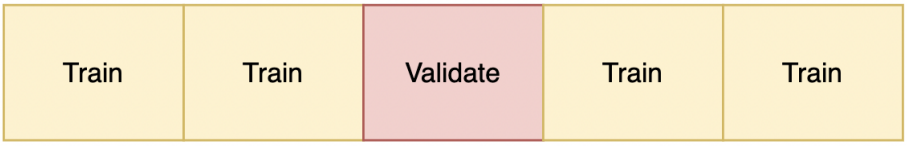

Figure 4.2: Splitting the data into 5 folds, with the 3rd fold being the validation set.

For each k^{th} step in the cross validation process, a slightly different model $\hat{f}$ is fitted, so the loss function values on the validation set will be different. The cross validation error, below, takes the average of the loss function values and provides a good estimate for the prediction error of the fitted model.

Definition 4.3.3 (Cross Validation Error, based on pg 242 [25])**.** The CV error at a specific set of tuning values θ is,

$$CV(\hat{f};\theta) = \frac{1}{N}\sum_{i=1}^{N} L(y_i, \hat{f}^{-\kappa(i)}(x_i;\theta)) \tag{4.11}$$

where $x_i \in \mathbb{R}^n$ is an observation, $y_i \in \mathbb{R}$ is the response, $\kappa : \{1,...,n\} \mapsto \{1,...,K\}$ is an indexing function which outputs which fold an observation is in and $\hat{f}^{-k}(x)$ is the fitted function with the k^{th} fold removed. The cross validation error directly estimates the expected test error Err [25].

Definition 4.3.4 (Common Loss functions pg 219 [25])**.**

$$L(Y, \hat{f}(X)) = (Y - \hat{f}(X))^2 \text{ Squared Error} \tag{4.12}$$

$$L(Y, \hat{f}(X)) = |Y - \hat{f}(X)| \text{ Absolute Error} \tag{4.13}$$

The Mean squared error is found by dividing the squared error by the sample size n. Typically, mean squared error is usually used to calculate the CV error, and it gives greater emphasis to fitted values farther away from the true response values. Note that the absolute error is on the same scale as the response. We can square root the MSE to get the root mean squared error (RMSE) which is on the same scale as well. Note, these loss functions are commonly used for a Gaussian response.

Definition 4.3.5 (Binomial Deviance pg 12 [23])**.** The binomial deviance D is defined as,

$$D = -2\sum_{i=1}^{n}\left(y_i \log\left(\frac{\hat{\mu}_i}{y_i}\right) + (1-y_i)\log\left(\frac{1-\hat{\mu}_i}{1-y_i}\right)\right). \tag{4.14}$$

Or equivalently,

$$-2\left(\sum_{i:y_i=0}\log(1-\hat{\mu}_i) + \sum_{i:y_i=1}\log(\hat{\mu}_i)\right) \tag{4.15}$$

where $\hat{\mu}_i$ are the fitted probabilities.

The deviance is the difference between the log likelihoods of the saturated and fitted models multiplied by 2. We use the binomial deviance to calculate the cross validation error for Logistic models, and the MSE for Gaussian models. Note, one could use AUROC (Definition 4.4.1) instead of binomial deviance, but the latter is still preferable as it gives a smoother CV curve. We now present the K-fold cross validation algorithm for the Lasso.

Algorithm 4 K-fold CV for Lasso

1. Split the data randomly into K roughly equal partitions.
2. At the k^{th} step, label the k^{th} partition as a validation set and fit a model, fixing λ to some value, with the remaining $K-1$ partitions as the training set.
3. Calculate and average out the MSE/binomial deviance of the fitted model on the validation set for $k = \{1, 2, ..., n\}$ to find the CV error.
4. Repeat the above steps (except step 1) for a range of values λ.

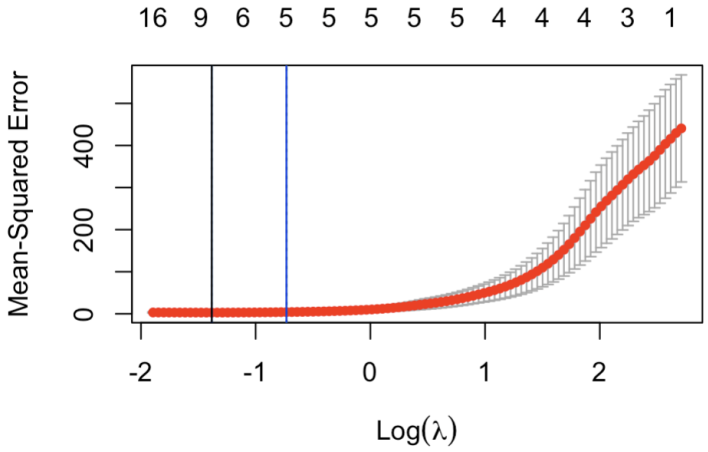

Figure 4.3: Cross Validation for the Lasso using glmnet with the same data as in Figure 3.1. The red line represents the CV error for each $\log(\lambda)$. The grey whiskers represent cover a range of one standard error above and below the CV error estimate. The black line is *lambda.min* and the blue line is *lambda.1se*. The left axis is the CV error, and the top axis describes how many coefficients are selected for a $\log(\lambda)$.

Using a built-in function from glmnet [30], we plot a cross validation curve in Figure 4.3. For $j = 1, ..., 5$, $\beta_j \neq 0$ whereas the other 45 are zero. Below is a table specifying the true and estimated β rounded to 4 decimal places.

True β	lambda.min β	lambda.1se β
-13.4088	-12.9693	-12.7952
-6.6541	-6.4441	-6.2946
3.1052	3.0106	2.7367
-8.9314	-8.5711	-8.3691
8.8749	8.3102	8.0817
Rest are 0.	$\hat{\beta}_{13} = -0.0019$, $\hat{\beta}_{27} = 0.0617$ $\hat{\beta}_{35} = 0.0278$, $\hat{\beta}_{39} = -0.0019$	Rest are 0.

In glmnet cross validation, lambda.min is the λ that produces the model with the smallest CV error, and lambda.1se is the λ that produces a model that has a CV error 1 standard error greater than the minimum CV error. lambda.1se is also greater than lambda.min, so the resulting model is usually sparser. In our example, we see that lambda.min selects 4 irrelevant parameters, whilst lambda.1se selects none. Hence, to ensure a parsimonious model, we will take our optimal λ to be lambda.1se. For the Adaptive Lasso and the Naive Elastic Net, this works too but for SCAD and MCP, the package ncvreg [31] only outputs lambda.min. However, this is not an issue as both methods generally select few parameters.

In general, one commonly uses 5 or 10 folds for cross validation. However, since we have a small sample size, the training and validation sets may be too small to produce meaningful results. In

Logistic regression, the problem is greater, as some folds may have very few observations of one binary class. An alternative method is to use Leave One Out Cross Validation.

Definition 4.3.6 (Leave One Out Cross Validation (LOOCV) [25])**.** The process here is equivalent to K-fold CV, except that each observation is assigned to an individual fold, so the number of folds is equivalent to the number of samples.

This method has no randomness in assigning observations to training and validation sets, and has less bias as the training set comprises of $n-1$ samples. The method can be computationally intensive as we fit models n times. However, since our n is small, this does not matter.

All of our methods except Ridge and Lasso have 2 tuning parameters to be determined via cross validation. Hence, we must use 2D K-fold Cross Validation for the other methods.

Definition 4.3.7 (2D K-fold Cross Validation)**.** The process is essentially the same as the Lasso CV, except that we must first specify which fold the observations go to. Then, we use a pre-specified vector of α or γ values and fix them whilst validating over different values of λ as in the Lasso. We repeat this for all the α or γ values and record the combination of $\alpha\backslash\gamma$ and lambda.1se\lambda.min that gives the lowest CV error.

4.4 Model Diagnostics

4.4.1 Prediction Accuracy

When comparing methods, we split our data into a training set $\left(\frac{2}{3}\right)$ and a test set $\left(\frac{1}{3}\right)$. We calibrate our tuning parameters and fit models using each method on the training set and evaluate the performance of the models on the test set.

For a continuous response, we use Gaussian penalised methods and compare them using the root mean squared error (RMSE) of the fitted values and the true responses on the test set. This is on the same scale as the response, and we call this our prediction error. During a simulation, we will run the methods several times on different data sets to observe the variance in prediction errors.

For a binary response, we use Logistic penalised methods. However, the resulting model gives probabilities, $P(Y=1)$, so we would need to determine a cutoff threshold to get a predicted binary response. One could use a threshold of 0.5, where if the probability is greater than 0.5, the predicted y would be 1. We can then determine the misclassification rate between the predicted y and true y in the test data. However, this means that the comparisons are dependent on an arbitrary threshold. A second method would be to compute the area under the receiver operator characteristic curve (AUROC). This measures the model's ability to give higher probabilities to predictions with true $y=1$ than $y=0$.

Definition 4.4.1 (AUROC pg 174[23])**.** Recall that the true positive rate (TPR) and the false positive rate (FPR) are given by

$$\text{TPR} = \frac{TP}{TP+FN}, \quad \text{FPR} = \frac{FP}{FP+TN}$$

where, TP, FP, TN, FN mean true positive, false positive, true negative and false negative respectively. These values are common referred to as sensitivity and 1-specificity, respectively. The receiver operation characteristic (ROC) curve plots the TPR against the FPR for all cutoff thresholds. The area under this curve is called AUROC and is between 0.5 and 1.

As a rule of thumb, we use the following criteria below [23].

AUROC= 0.5	No discrimination - equivalent to a coin toss
$0.5 <$ AUROC < 0.7	Poor discrimination - not much better than a coin toss.
$0.7 \leq$ AUROC < 0.8	Acceptable discrimination
$0.8 \leq$ AUROC < 0.9	Excellent discrimination
AUROC ≥ 0.9	Outstanding discrimination

Therefore, an AUROC of 0.5 suggests that the model isn't useful, so we might as well toss a coin to predict the response.

Example 4.4.1. Consider the colon cancer data set from [6] with $n = 62$ observations and $p = 2000$ genes. We take the first 42 observations as our training set and the remaining 20 to be our test set. Then, we fit a Logistic Lasso model to the training set and predict the response best on data from the test set. After that, we plot the ROC curve. We calculate the AUROC to be 0.71875, so there is acceptable discrimination.

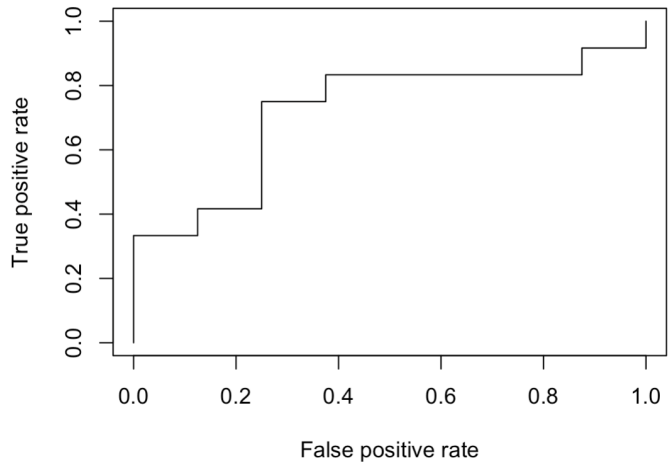

Figure 4.4: ROC curve of the logistic Lasso on the colon cancer data set from [6].

In the simulations of the next chapter, we will consider several random realizations of training and test sets and average the AUROC value across them.

4.4.2 Model Selection

Since, our applications depend on selecting relevant genes accurately, we will also run simulations to investigate the selection accuracy and variability of different methods. We will use 3 metrics.

Definition 4.4.2 (Selection Accuracy)**.** The strength of a method's selection accuracy is determined by how many correct coefficients it includes and how often.

Definition 4.4.3 (Selection Relevance)**.** The selection relevance of a method is determined by the fraction of the number of correct parameters selected over the total number of parameters selected.

Definition 4.4.4 (Selection variability)**.** A method's selection variability describes how many different coefficients the model selects under similar datasets. This can be seen as an indicator of stability.

Chapter 5

Simulation Studies

In this chapter, we conduct several simulations involving the penalised methods covered in chapter 2 on both Gaussian and Logistic frameworks. We explore their properties involving predictive accuracy, model selection consistency, parameter estimation and stability. Before extracting any sample covariance matrices from microarray data, we perform a $\log_2$ transformation on the dataset to make the data less skewed. During some Logistic simulations, we notice that the algorithms for SCAD and MCP do not converge for small γ and λ, possibly due to non-convexity. Hence, we do not carry out 2D cross validation for the Logistic versions of SCAD and MCP, but instead use the default $\gamma = 3.7$ and $\gamma = 3$ respectively.

5.1 Predictive Accuracy

We simulate our data using the sample covariance matrix $\hat{\Sigma}$ from the colon cancer data set [6] and sample X from a multivariate normal using the R package MASS [48]. $X \sim \mathcal{N}(0, \hat{\Sigma})$, $Y \sim \mathcal{N}(X\beta, I)$. We have 2000 parameters in total with five $\beta_j \sim \mathcal{U}(-5, 5)$, with j chosen randomly, and the rest are 0. We perform 100 iterations and split our data randomly so that our training set contains 40 observations and our test set contains 20 observations.

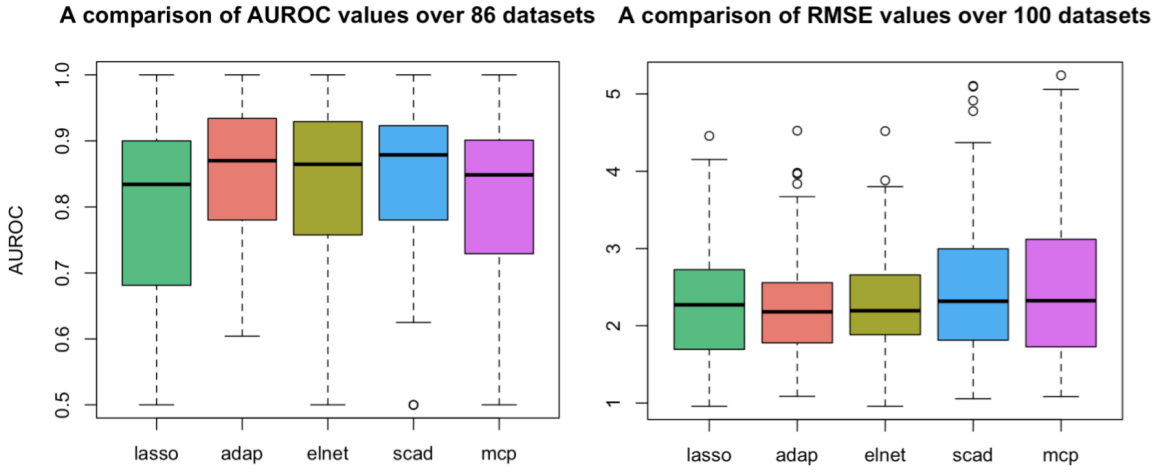

Figure 5.1: A comparison of predictive accuracies between a range of penalised methods. Logistic framework (left), Gaussian framework (right).

The mean RMSE values are roughly the same, with the Adaptive Lasso and Elastic Net performing slightly better. For Logistic models, we use the same data generation process with the additional

step of using the logit link to transform the linear predictor to the response. Since some generated Y do not have a varied number of 1's and 0's, we take some simulated data out, and this leaves us with 86 iterations. This ensures we do not have too few 1's or 0's per fold for cross validation. Overall, all methods have higher AUROC values than the Lasso, with SCAD scoring the highest. The median values are 0.83, 0.87, 0.86, 0.88, 0.84 respectively. Using the criteria presented in the previous chapter, we can see that all the methods have excellent discrimination.

5.2 Model Selection Consistency

We now consider the selection variability of the Gaussian simulation. The Naive Elastic Net exhibited 31 instances of overfitting. Since the Naive Elastic Net is a compromise between the Ridge and the Lasso overfitting is possible, however for the other methods, it is not possible. We removed those instances and display the results below.

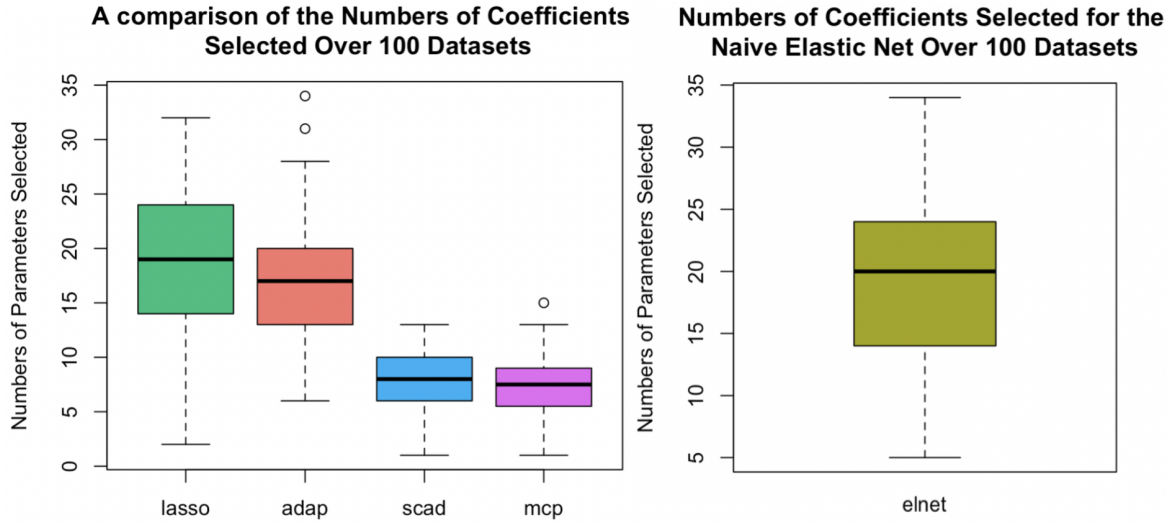

Figure 5.2: Selection Variability under the Gaussian framework.

SCAD and MCP select the sparsest models with the lowest variance. The Adaptive Lasso performs slightly better than the Lasso and the Naive Elastic Net. Note that the mean parameter numbers of SCAD and MCP are very close to 5 which is the true number of parameters.

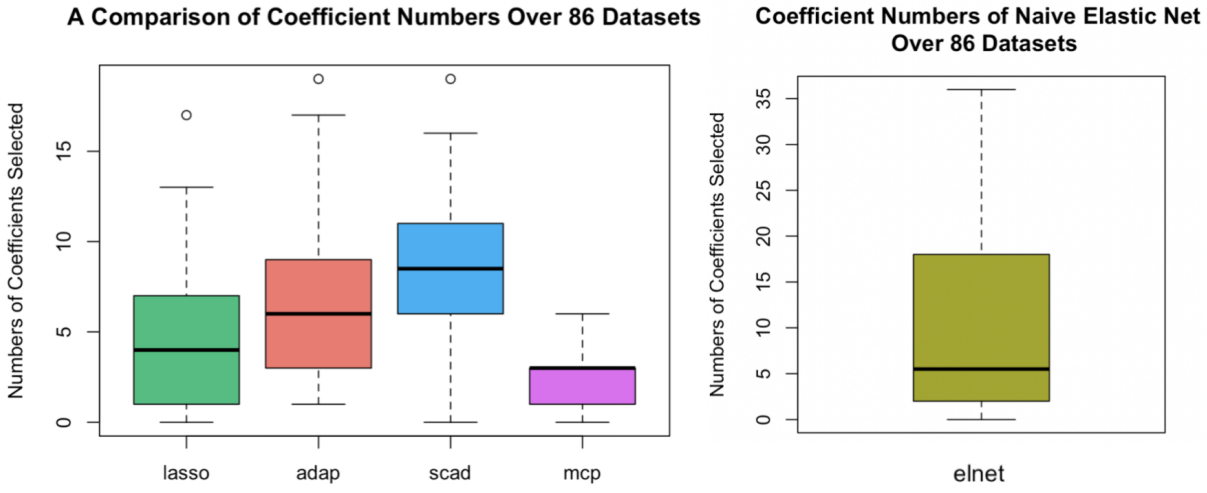

Figure 5.3: Selection Variability under the Logistic framework.

In the Logistic simulation the Elastic Net over fitted 40 times so we discard those cases. All methods select sparser models than the Gaussian case, but the Lasso and MCP in particular seem to under-fit as their mean numbers of parameters are less than 5. Overall, the Adaptive Lasso performs the best here, with low variance and a mean parameter count close to 5. Now, we analyse the selection accuracy and relevance of our methods.

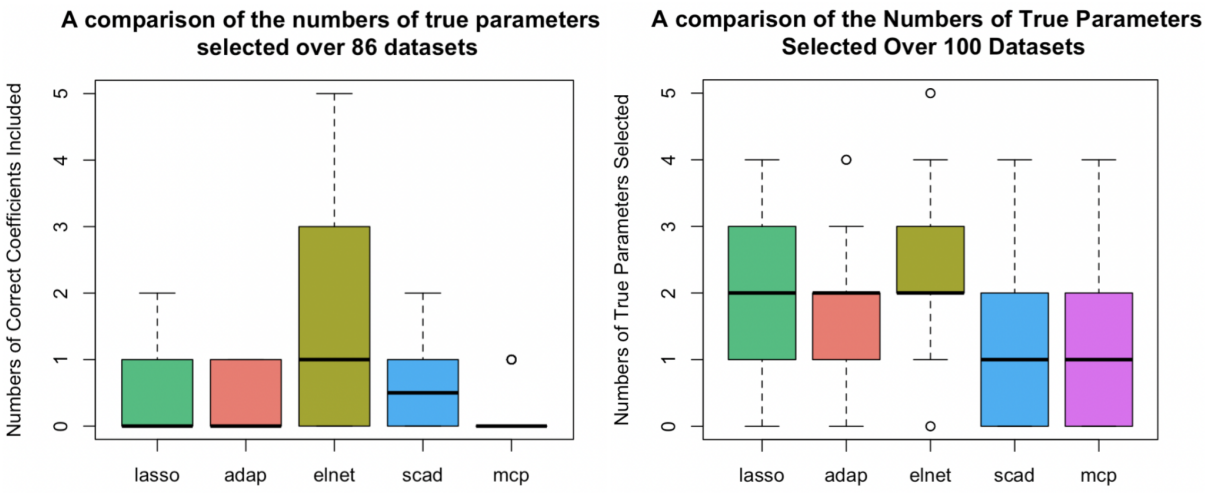

Figure 5.4: Comparisons on the selection accuracies. Logistic framework (left), Gaussian framework (right).

In both cases it seems that Naive Elastic Net performs the best, however, as noted above it suffers from frequent over-fitting so it is not reliable. Hence, that leaves the Lasso and Adaptive Lasso as being the best performers in the Gaussian case. In the Logistic case, SCAD performs the best.

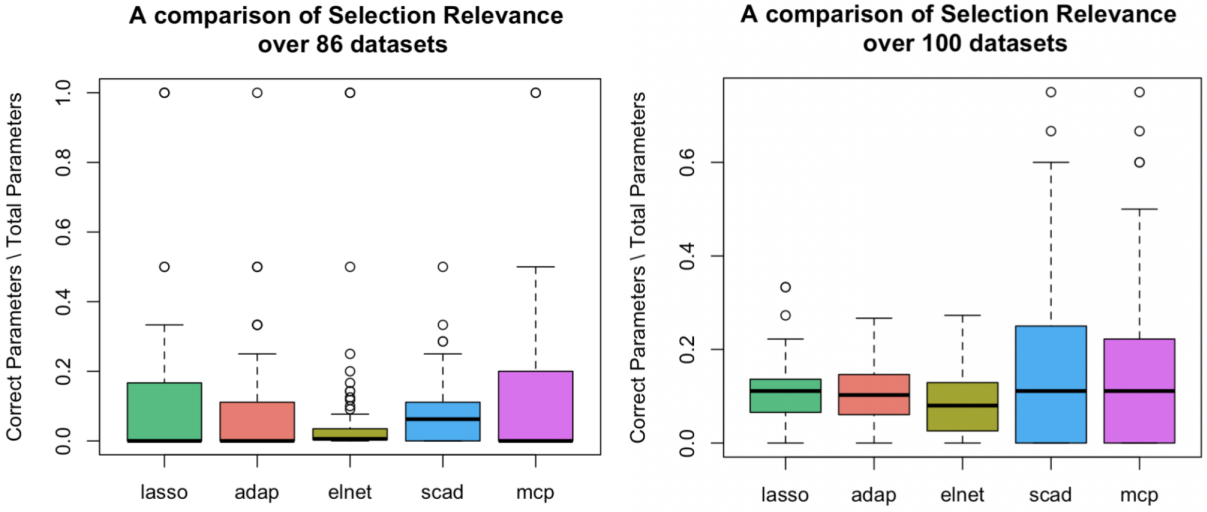

Figure 5.5: Comparisons on the selection relevance. Logistic framework (left), Gaussian framework (right).

For the Gaussian framework, all the methods perform roughly the same on average, with 1 in 10 parameters selected being relevant. SCAD and MCP have a more varied range of selection variance as they consistently select a sparse model, so not many irrelevant parameters are selected in the first place. For the Logistic case, selection relevance is very poor, with SCAD performing the best.

5.3 Parameter Estimation

In the following examples, we conduct simulations first on non-collinear data and then on multi-collinear data. We set our true β to be either very large or very small, and observe how accurately our methods estimate the coefficients. For the folded concave penalties, we use the default values for γ, namely for MCP $\gamma = 3$, and for SCAD $\gamma = 3.7$.

Example 5.3.1 (Large Coefficients, Gaussian Response)**.** We generate $X \sim \mathcal{N}(0, I)$ for 50 iterations where $n = 40$, $p = 200$. The true β is specified in the table below. $\epsilon \sim \mathcal{N}(0, 1)$ and $Y = X\beta + \epsilon$. The columns specify the average absolute difference in magnitude between our estimated coefficients and the true coefficients over all the iterations. We omit the values of any other selected parameters and give the estimates to 2 decimal places.

True β	Lasso	Naive Elastic Net	Adaptive	SCAD	MCP
$\beta_1 = 200$	12.97	18.32	15.21	2.15	0.86
$\beta_2 = -100$	14.4	20.53	18.61	2.8	1.09
$\beta_3 = 50$	12.13	16.78	16.73	16.09	6.77
$\beta_4 = -400$	13.34	18.52	13.83	1.94	0.78
$\beta_5 = 120$	13.48	18.65	17.02	2	0.84

Here we see that SCAD and MCP outperform the normed penalties greatly in terms of parameter estimation accuracy, with MCP performing the best. This could be due to the fact that the concave penalties tail off as β_j increases (see Figure 3.9) so the large coefficients are not overly penalised.

Example 5.3.2 (Small Coefficients, Gaussian Response)**.** We generate the data in the same way as above but now with a different set of β.

True β	Lasso	Naive Elastic Net	Adaptive	SCAD	MCP
$\beta_1 = 0.5$	0.46	0.45	0.43	0.39	0.42
$\beta_2 = 2$	0.73	0.75	0.38	0.16	0.16
$\beta_3 = -2$	0.68	0.7	0.37	0.18	0.17
$\beta_4 = -0.8$	0.62	0.61	0.53	0.34	0.35
$\beta_5 = 1.3$	0.67	0.69	0.46	0.18	0.21

In this case, the estimation differences of the methods are similar, with SCAD and MCP performing mildly better.

Example 5.3.3 (Small and Large Coefficients on Multi-collinear data, Gaussian Response)**.** Here, we generate our data using a covariance matrix $\hat{\Sigma}$ which satisfies the irrepresentable condition, but still has some multicollinearity as it is positive semi-definite but not positive definite. $X \sim \mathcal{N}(0, \hat{\Sigma})$, $Y \sim \mathcal{N}(X\beta, I)$. $n = 40$ observations, $p = 200$ parameters and we have 50 iterations.

True β	Lasso	Naive Elastic Net	Adaptive	SCAD	MCP
$\beta_1 = 200$	5.61	5.53	5.35	0.84	0.84
$\beta_2 = -200$	6.28	6.2	5.99	0.66	0.66
$\beta_3 = 5$	4.88	4.76	5	5	5
$\beta_4 = 1$	1	1	1	1	1
$\beta_5 = -2$	2	2	2	2	2

All 5 methods miss out the small coefficients β_3, β_4 and β_5 over the 50 iterations except the Lasso and Elastic Net which select β_3 occasionally. This may be because of the large coefficients, which mask the effects of the parameters with small coefficients. In terms of parameter estimation for the large coefficients, SCAD and MCP perform better than the normed penalties.

5.4 Stability

Example 5.4.1 (Stability of OLS stepwise selection and Lasso to single observation removal)**.** We sample non-collinear data $X \sim \mathcal{N}(0, I)$ with 100 observations and 30 parameters. We sample 5 uniform $\beta_j \sim \mathcal{U}(-10, 10)$ and the rest of the β_j are 0. $\epsilon \sim \mathcal{N}(0, 1)$ and $Y = X\beta + \epsilon$. We remove the observation at index $i \in \{1, ..., 100\}$ and compare OLS with backward stepwise selection against Lasso regression 100 times to select models. We present a table concerning the numbers of parameters selected each time for both Lasso and stepwise OLS.

No. of parameters	5	6	7	8	9	10	11	13
Stepwise OLS	0	0	11	58	5	17	8	1
Lasso	33	67	0	0	0	0	0	0

Not only is the Lasso less sensitive to observation removal, it also selects models closer to the number of true parameters. In this example, 33% of its selected models have the same number as the true model, whilst none of the stepwise selection have. This matches Breiman's analysis that stepwise methods have high variability, as their selection is discrete [29]. On the other hand, the Lasso has less variability the continuous shrinkage allows some compromise between the coefficients. Using collinear data, the OLS method performs poorly as expected (see Section 2.4) and the Lasso is less stable.

No. of parameters	4	5	6	7	8	9	29
Stepwise OLS	0	0	0	0	0	0	100
Lasso	1	14	33	18	28	6	0

Example 5.4.2 (Sensitivity to observation removal on Microarray-Data, Logistic Response)**.** Let the 2000 parameter sample covariance matrix from the colon dataset in [6] be denoted $\hat{\Sigma}$. We generate 60 observations using $X \sim \mathcal{N}(0, \hat{\Sigma})$. We then generate 5 non-zero beta's using the same $\mathcal{U}(-10, 10)$ distribution and 60 $\epsilon_i \sim \mathcal{N}(0, 1)$. We take $Y = X\beta + \epsilon$. We remove a single observation i, for $i = 1, ..., 60$ leaving behind 59 observations, and compare how many parameters are selected at a time.

No. Parameters p	$p < 5$	$5 \leq p < 10$	$10 \leq p < 15$	$15 \leq p < 30$	$30 \leq p \leq 60$	$p > 60$
Lasso	3	20	37	0	0	0
Elastic Net	3	1	6	4	10	36
Adaptive Lasso	5	29	20	6	0	0
SCAD	2	33	24	1	0	0
MCP	43	17	0	0	0	0

Lasso and MCP perform the best in terms of stability. The Elastic Net performs the worst and suffers from some over-fitting as well due to the compromise with Ridge.

Chapter 6

Application to Microarray Data

In this chapter, we apply all the penalised methods explored above except Ridge regression on real life data sets. Recall that we have two objectives:

1. To identify candidate genetic biomarkers for cancers.
2. To identify genes which may affect candidate genetic biomarker for cancers.

6.1 Microarray Data

A microarray dataset is a gene expression matrix of several samples. Fundamentally, genes are small sections of DNA that provide instructions for the production of specific proteins. The amount of gene expression determines the characteristics of a cell, so tumour and non-tumour cells well have different levels of gene expression. DNA contains four bases, adenine (A), thymine (T), cytosine (C), and guanine (G) where A strictly binds to T and C strictly binds to G. No other interactions are possible. Two strands of DNA are complementary if, for all bases on one strand, the other strand contains the complements of those bases. The two strands react in a process called hybridization to form a double helix.

Microarray technology works by extracting mRNA from a tissue, e.g. tumour tissue, labelling it to form complementary DNA and then hybridizing it to the genes on the microarray chip. The mRNA represents the characteristics of the tissue, and only the genes which contribute to those will be hybridized. In this report, we investigate data from Affymetrix chips, which use oligonucleotide sequences thought to be representative of specific genes and photolithographic technology. Further details can be found in [49]. We use the following data sets:

Record	Title	Samples	Platform
GDS 4102	Pancreatic Tumor and Normal Tissue Samples	36 Tumor, 16 Non-Tumor	GPL 570
GDS 4336	Pancreatic Ductal Adenocarcinoma Tumor and Adjacent Non-Tumor Tissue	45 Tumor, 45 Non-Tumor	GPL 6244
GDS 4103	ICF cohort: Whole Tissue Pancreatic Ductal Adenocarcinoma	36 Tumor, 36 Non-Tumor & 6 Replicates	GPL 570

Figure 6.1: Microarray Datasets from GEO Omnibus [7]: GDS4102 [9], GDS4336 [10], GDS4103 [11]. Platform denotes the Affymetrix chip used. After the removal of control genes, the datasets have 54613, 28829, and 54613 parameters respectively.

To find candidate biomarkers, we remove any sample replicates and control genes, as these are used to reduce background and systemic variation, and hence have irrelevant responses. For our second objective, we remove the control genes but keep the sample replicates, as our analysis is only based on interactions between genes. Note, all the datasets are already RMA normalised to reduce background noise and allow for dataset cross-comparison. Before analysing the datasets, we also perform a log2 transformation to reduce data skewness. After splitting the data randomly into training and test sets such that the test set contains roughly $25\% - 33\%$ of the observations, we fit the models to the training sets and compare their performances on the test set.

6.2 Methods

For our first application, we fit models using the logistic version of the Lasso, (Naive) Elastic Net, Adaptive Lasso, SCAD and MCP. From here onwards, we drop the 'Naive' distinction and simply refer to it as the Elastic Net. We do not use Ridge as it does not give sparse solutions. We use 1D LOOCV for the Lasso, SCAD and MCP. For the Elastic Net and Adaptive Lasso, we use k-fold 2D cross validation to reduce the computation time. We use the packages GEOquery [8], glmnet [30], ncvreg [31] and ROCR [50].

For our second application, we use the Gaussian version of the penalised methods given above. However, we need to ensure that the distribution of the response is Gaussian-like. Hence, we perform a Box-Cox transformation on Y beforehand and constrain the transformation parameter $\lambda \in [-5, 5]$. After that, we standardise Y so that the RMSE values are cross-comparable between datasets.

6.3 Results

We begin with the results for the first objective. The tables below show the Affymetrix probe set IDs of the parameters selected for each method per dataset. We also provide the AUROC values of the models per dataset in Figure 6.5.

Method	Parameters Selected, Dataset GDS 4103
Lasso	203021_at, 212353_at, 217428_s_at, 231993_at
Elastic Net	1557080_s_at, 1560228_at, 1563034_at, 203021_at, 203700_s_at, 205422_s_at, 205713_s_at, 212353_at, 212354_at, 214927_at, 217428_s_at, 218856_at, 220377_at, 223690_at, 226237_at, 227140_at, 231240_at, 231993_at, 243372_at
Adaptive Lasso	1553266_at, 1554298_a_at, 1559307_s_at, 1560760_s_at, 1561962_at, 1562083_at, 202078_at, 203170_at, 204587_at, 204663_at, 207793_s_at, 210433_at, 216190_x_at, 216605_s_at, 216881_x_at, 217126_at, 217136_at, 217577_at, 219262_at, 219643_at, 221506_s_at, 221619_s_at, 230113_at, 233683_at, 234115_s_at, 240177_at, 240179_at, 243372_at, 243607_at, 55093_at
SCAD	1560228_at, 1563034_at, 203021_at, 203700_s_at, 204848_x_at, 212285_s_at, 212353_at, 214927_at, 216605_s_at, 224499_s_at, 227745_at, 231993_at, 243372_at
MCP	201474_s_at, 205422_s_at, 243372_at

Figure 6.2: Affymetrix probe set IDs of parameters with non-zero coefficients in the models fitted per penalised method on the dataset GDS 4103.

According to Figure 6.5, all the methods produce models with at least excellent discrimination, and specifically, the Lasso, SCAD and Elastic Net perform outstandingly. These three methods

share the elements, '203021_at' and '212353_at' in common. Also note, '231993_at' is selected by the Lasso, Elastic Net and MCP and '243372_at' is selected by all the methods except the Lasso. Looking at the feature data, we see that the gene symbols are SLPI, SULF1, ITGBL1, and HSPD1, respectively.

Method	Parameters Selected, Dataset GDS 4102
Lasso	1568892_at, 228233_at, 236972_at, 237390_at, 243379_at
Elastic Net	1555731_a_at, 1557437_a_at, 1560322_at, 1562546_at, 1568892_at, 1570422_at, 208659_at, 209968_s_at, 210844_x_at, 211301_at, 215713_at, 219340_s_at, 220210_at, 222154_s_at, 222392_x_at, 222702_x_at, 223249_at, 225573_at, 228233_at, 229377_at, 230042_at, 230417_at, 230790_x_at, 232510_s_at, 234381_at, 236972_at, 237361_at, 237390_at, 237396_at, 238964_at, 239548_at, 240581_at, 241328_at, 243379_at, 244697_at
Adaptive Lasso	1558849_at, 1561274_at, 220210_at, 230088_at, 232943_at
SCAD	1560322_at, 1568892_at, 205303_at, 209968_s_at, 220210_at, 225207_at, 228233_at, 230042_at, 230088_at, 230417_at, 234381_at, 236511_at, 236972_at, 237390_at, 237396_at, 241328_at
MCP	1557437_a_at, 1568892_at, 236972_at

Figure 6.3: Affymetrix probe set IDs of parameters with non-zero coefficients in the models fitted per penalised method on the dataset GDS 4102.

On this dataset, the Lasso and MCP perform less well, possibly due to the small number of non-tumour samples. By contrast, the other methods still perform well. Note, that the microarray platform for this dataset is the same as the one above, so the data parameters are the same. However, the experimental conditions may be different. '1568892_at' and '236972_at' are selected by all the methods except the Adaptive Lasso. '228233_at' and '237390_at' are selected by the Lasso, Elastic Net, and SCAD. '220212_at' is selected by the Elastic Net, Adaptive and SCAD. The corresponding gene symbols are: LOC100996251, TRIM63, FREM1, ADRA1A. The feature does not include the gene symbol for '220212_at'.

Method	Parameters Selected, Dataset GDS 4336
Lasso	7983718, 8037408, 8062545, 8093950, 8098637, 8101366, 8166266
Elastic Net	7909164, 7962579, 7977409, 7981514, 7983718, 7998784, 8020551, 8028924, 8037408, 8049487, 8058627, 8062545, 8063028, 8093950, 8098637, 8098654, 8101366, 8166266, 8169504
Adaptive Lasso	7911444, 7915949, 7919028, 7919940, 7925851, 7936612, 7940171, 7963208, 7963588, 7975813, 7981968, 8006502, 8015049, 8019308, 8020802, 8037309, 8045585, 8051185, 8062545, 8064382, 8064462, 8095161, 8097036, 8152213, 8155451, 8157608, 8168569, 8170633, 8175252
SCAD	7923034, 7962579, 7981514, 7983718, 7987365, 8017098, 8028924, 8037408, 8062545, 8093950, 8098637, 8101366, 8104461, 8166266
MCP	7983718, 8037408

Figure 6.4: Affymetrix probe set IDs of parameters with non-zero coefficients in the models fitted per penalised method on the dataset GDS 4336.

Here, all models have excellent discrimination except the Adaptive Lasso. '7983718' and '8037408' were selected by all models except the Adaptive Lasso. '8062545' was selected by all except MCP. '8093950', '8098637', '8101366', '8166266' were all selected by Lasso, Elastic Net and SCAD. The corresponding gene symbols are: SCG3, KCNN4, ACTR5, S100P, CYP4V2, SCD5, NHS.

Dataset	Lasso	Elastic Net	Adaptive Lasso	SCAD	MCP
GDS 4103	0.9030	0.9091	0.8061	0.9212	0.8606
GDS 4102	0.7188	0.8750	0.9375	0.9375	0.6563
GDS 4336	0.8824	0.9186	0.7557	0.9004	0.9004

Figure 6.5: A table showing the AUROC values for models fitted by a range of penalised methods on a single random realisation of training and test partition of the data.

We now consider the second objective. The gene MLH1 is a known biomarker of pancreatic cancer [51]. For, GDS 4103 and GDS 4102, the identifier is 202520_s_at and for GDS 4336, the identifier is 8078544. We regress this parameter against the other parameters using the Gaussian version of our penalised methods. We present the results in Figure 6.6, Figure 6.7, and Figure 6.8, and a table of RMSE values for reference in Figure 6.9.

Method	Parameters Selected, Dataset GDS 4103, Training Sample Size: 52
Lasso	1555483_x_at, 203115_at, 203261_at, 206746_at, 213324_at, 221475_s_at, 221791_s_at, 222844_s_at, 223474_at, 224060_s_at, 230326_s_at, 238731_at
Elastic Net	Overfit, 143 Parameters selected
Adaptive Lasso	1552993_at, 1554886_a_at, 1555483_x_at, 1555985_at, 1556937_at, 1557457_at, 1558375_at, 1561959_x_at, 1562544_at, 1564628_at, 1567031_at, 1569449_a_at, 1570306_at, 1570523_s_at, 201707_at, 202395_at, 203115_at, 203261_at, 206746_at, 210726_at, 210966_x_at, 211023_at, 212559_at, 216520_s_at, 219819_s_at, 221475_s_at, 222844_s_at, 222975_s_at, 223892_s_at, 226366_at, 229756_at, 231383_at, 233038_at, 233958_at, 238540_at, 239463_at, 241188_at, 241284_at, 243184_at, 244041_at
SCAD	1555483_x_at, 203115_at, 203261_at, 206746_at, 221475_s_at, 222844_s_at, 228332_s_at, 233038_at, 238731_at, 239463_at, 241188_at, 244041_at
MCP	1555483_x_at, 203115_at, 203261_at, 206746_at, 221475_s_at, 222844_s_at, 228332_s_at, 233038_at, 238731_at, 239463_at, 241188_at, 244041_at

Figure 6.6: Affymetrix probe set IDs of parameters with non-zero coefficients in the models fitted per penalised method on the dataset GDS 4103 with 202520_s_at as the response.

Since the Elastic Net model overfitted, we omit it in our analysis. In all of the other methods, the following were selected: '1555483_x_at', '203115_at', '203261_at', '206746_at', '221475_s_at', '222844_s_at'. The corresponding gene symbols are: FBLIM1, FECH, DCTN6, BFSP1, RPL15, SRR. Also note that the low RMSE values of the Adaptive Lasso, SCAD and MCP may suggest they perform well. However, note that the Adaptive Lasso selects many parameters. SCAD and MCP also select similar parameters to the Lasso, so the increase in predictive accuracy may be due to better parameter estimation.

Method	Parameters Selected, Dataset GDS 4102, Training Sample Size: 42
Lasso	201632_at, 207040_s_at, 211036_x_at, 221475_s_at, 227298_at
Elastic Net	201632_at, 201805_at, 203409_at, 203829_at, 204808_s_at, 205429_s_at, 207040_s_at, 211036_x_at, 220742_s_at, 221475_s_at, 227298_at
Adaptive Lasso	1557889_at, 1560928_at, 1562703_at, 203409_at, 205600_x_at, 207805_s_at, 210930_s_at, 211411_at, 213054_at, 213870_at, 215241_at, 218097_s_at, 220274_at, 220312_at, 222609_s_at, 223541_at, 223770_x_at, 225788_at, 227298_at, 228162_at, 229862_x_at, 230559_x_at, 238223_at, 242913_at, 244289_at
SCAD	201632_at, 203115_at, 203409_at, 205601_s_at, 211214_s_at, 218377_s_at, 219718_at, 221475_s_at, 225398_at
MCP	201632_at, 203409_at, 205600_x_at, 210627_s_at, 211214_s_at, 215411_s_at, 215757_at, 226691_at, 227850_x_at 229279_at

Figure 6.7: Affymetrix probe set IDs of parameters with non-zero coefficients in the models fitted per penalised method on the dataset GDS 4102 with 202520_s_at as the response.

Here, we see that '201632_at' is selected by all methods except the Adaptive Lasso and '203409_at' is selected by all methods except the Lasso. '227298_at' is selected by the Lasso, Elastic Net and Adaptive Lasso, and echoing the previous dataset '221475_s_at' is selected by the Lasso, Elastic Net and SCAD. The corresponding gene symbols are: IF2B1, DDB2, TRAM2-AS1, RPL15.

Method	Parameters Selected, Dataset GDS 4336, Training Sample Size: 60
Lasso	7898084, 7915758, 7971039, 7976412, 7997533, 8003719, 8006170, 8017476, 8023080, 8030339, 8037166, 8068578, 8078110, 8084219, 8085815, 8088151, 8103106, 8105801, 8116195, 8116530, 8121429, 8127141, 8129045, 8129943, 8176782, 8177347
Elastic Net	Overfit, 140 Parameters Selected
Adaptive Lasso	7899392, 7901867, 7916616, 7918323, 7930413, 7935951, 7938834, 7939954, 7945228, 7945680, 7952086, 7961820, 7976412, 7993622, 7993798, 7997048, 7997533, 8003719, 8005879, 8006170, 8013484, 8014259, 8019954, 8023080, 8030339, 8036207, 8044295, 8053427, 8068578, 8070239, 8079867, 8084219, 8088065, 8088151, 8105801, 8115476 8116530, 8127141, 8129943, 8150175, 8155458, 8166213, 8175642, 8176782
SCAD	7898084, 7915758, 7976412, 8023080, 8030339, 8078110, 8088151, 8103106, 8105801, 8116530, 8127141, 8129943, 8157700
MCP	7899392, 7914805, 7938834, 8005231, 8005879, 8087453, 8088065, 8116530, 8130580, 8140814, 8144600, 8175642

Figure 6.8: Affymetrix probe set IDs of parameters with non-zero coefficients in the models fitted per penalised method on the dataset GDS 4102 with 8078544 as the response.

Since the Elastic Net overfitted we omit it. '8116530' is selected by all the methods. Lasso, Adaptive Lasso and SCAD select: '7976412', '8030339', '8023080', '8088151', '8105801', '8127141', '8129943'. The corresponding gene symbols are: SNORD96A, LINC00521, FLT3LG, LOXHD1, ACTR8, SLC30A5 with the last two being unnamed.

Dataset	Lasso	Elastic Net	Adaptive Lasso	SCAD	MCP
GDS 4103	1.0400	1.0386	0.5265	0.7572	0.7389
GDS 4102	1.0818	0.9025	1.1659	0.7979	0.7835
GDS 4336	0.6207	0.6240	0.7907	0.6376	0.9070

Figure 6.9: A table showing the RMSE values for models fitted by a range of penalised methods on a single random realisation of training and test partition of the data.

Chapter 7

Conclusion

In this report, we have compared the theoretical and practical aspects of the Lasso, Adaptive Lasso, Naive Elastic Net, SCAD and MCP. We have presented the theoretical motivations behind the methods and compared their shrinkage effects through closed form, orthogonal solutions. We have also explored their parameter estimation, model selection and stability properties. Additionally, we have explored key model fitting algorithms process such as penalised coordinate descent and K-fold cross validation, for both Gaussian and Logistic frameworks. We have also included a brief overview of our model diagnostics.

In our simulations, we observe that despite having good predictive accuracies, the methods perform poorly in terms of model selection accuracy and relevance. In particular, the Naive Elastic Net frequently overfits and suffers major stability issues. Moreover, despite being asymptotically selection consistent, we observe that the Adaptive Lasso, SCAD and MCP do not exhibit any significant improvements against the Lasso in terms of selection accuracy and relevance. This is possibly due to the small sample size compared to the number of parameters. In future, it may be beneficial to explore whether there are any selection improvements given a greater sample size. In our application, we observe several potential candidate biomarkers. However, we note that each dataset gives a different set of results with hardly any overlap, which suggests some concern on the reliability of our results.

Throughout our analysis, we have assumed that all the genes are equally likely to be significant *a priori*. However, in reality there is already known information on genetic interaction. Therefore, an interesting direction would be to consider a Bayesian approach, where one could determine a shrinkage prior to take into account known biological information. Furthermore, we have applied the methods on each dataset individually. We believe it may be more beneficial to aggregate the data through sequential analysis under a Bayesian framework so that no information is wasted.

Appendix A

Code

All the code used to produce the figures and simulations in the report can be accessed via the following link to google drive.

`https://drive.google.com/drive/folders/14A1CzqftsmYrQMLvZd9uWOjHkjEUcGf2?usp=drive_link`

Bibliography

[1] Barry Barnes and John Dupré. *Genomes and what to make of them.* University of Chicago press, 2009.

[2] Mark Schena, Dari Shalon, Ronald W Davis, and Patrick O Brown. Quantitative monitoring of gene expression patterns with a complementary dna microarray. *Science*, 270(5235):467–470, 1995.

[3] Jeremy JW Chen, Reen Wu, Pan-Chyr Yang, Jane-Yu Huang, Yuh-Pyng Sher, Meng-Hsuan Han, Wei-Chen Kao, Pei-Jung Lee, Trai Fu Chiu, Fu Chang, et al. Profiling expression patterns and isolating differentially expressed genes by cdna microarray system with colorimetry detection. *Genomics*, 51(3):313–324, 1998.

[4] Stephen PA Fodor, J Leighton Read, Michael C Pirrung, Lubert Stryer, Amy Tsai Lu, and Dennis Solas. Light-directed, spatially addressable parallel chemical synthesis. *science*, 251(4995):767–773, 1991.

[5] Bertrand Jordan. *DNA microarrays: gene expression applications.* Springer Science & Business Media, 2001.

[6] Yaohui Zeng and Patrick Breheny. The biglasso package: A memory- and computation-efficient solver for lasso model fitting with big data in r. *R Journal*, 12(2):6–19, 2021.

[7] Tanya Barrett, Stephen E Wilhite, Pierre Ledoux, Carlos Evangelista, Irene F Kim, Maxim Tomashevsky, Kimberly A Marshall, Katherine H Phillippy, Patti M Sherman, Michelle Holko, et al. Ncbi geo: archive for functional genomics data sets—update. *Nucleic acids research*, 41(D1):D991–D995, 2012.

[8] Sean Davis and Paul S Meltzer. Geoquery: a bridge between the gene expression omnibus (geo) and bioconductor. *Bioinformatics*, 23(14):1846–1847, 2007.

[9] Huadong Pei, Liang Li, Brooke L Fridley, Gregory D Jenkins, Krishna R Kalari, Wilma Lingle, Gloria Petersen, Zhenkun Lou, and Liewei Wang. Fkbp51 affects cancer cell response to chemotherapy by negatively regulating akt. *Cancer cell*, 16(3):259–266, 2009.

[10] Geng Zhang, Aaron Schetter, Peijun He, Naotake Funamizu, Jochen Gaedcke, B Michael Ghadimi, Thomas Ried, Raffit Hassan, Harris G Yfantis, Dong H Lee, et al. Dpep1 inhibits tumor cell invasiveness, enhances chemosensitivity and predicts clinical outcome in pancreatic ductal adenocarcinoma. *PloS one*, 7(2):e31507, 2012.

[11] Liviu Badea, Vlad Herlea, Simona Olimpia Dima, Traian Dumitrascu, Irinel Popescu, et al. Combined gene expression analysis of whole-tissue and microdissected pancreatic ductal adenocarcinoma identifies genes specifically overexpressed in tumor epithelia-the authors reported a combined gene expression analysis of whole-tissue and microdissected pancreatic ductal adenocarcinoma identifies genes specifically overexpressed in tumor epithelia. *Hepato-gastroenterology*, 55(88):2016, 2008.

[12] Debashis Ghosh and Arul M Chinnaiyan. Classification and selection of biomarkers in genomic data using lasso. *Journal of Biomedicine and Biotechnology*, 2005(2):147, 2005.

[13] Ke Xue, Huilin Zheng, Xiaowen Qian, Zheng Chen, Yangjun Gu, Zhenhua Hu, Lei Zhang, and Jian Wan. Identification of key mrnas as prediction models for early metastasis of pancreatic cancer based on lasso. *Frontiers in Bioengineering and Biotechnology*, 9:701039, 2021.

[14] Shao-Hua Yu, Jia-Hua Cai, De-Lun Chen, Szu-Han Liao, Yi-Zhen Lin, Yu-Ting Chung, Jeffrey JP Tsai, and Charles CN Wang. Lasso and bioinformatics analysis in the identification of key genes for prognostic genes of gynecologic cancer. *Journal of personalized medicine*, 11(11):1177, 2021.

[15] Stéphane Chrétien, Christophe Guyeux, Michael Boyer-Guittaut, Régis Delage-Mouroux, and Françoise Descôtes. Using the lasso for gene selection in bladder cancer data. *arXiv preprint arXiv:1504.05004*, 2015.

[16] Zakariya Yahya Algamal and Muhammad Hisyam Lee. Penalized logistic regression with the adaptive lasso for gene selection in high-dimensional cancer classification. *Expert Systems with Applications*, 42(23):9326–9332, 2015.

[17] Hui Zou. The adaptive lasso and its oracle properties. *Journal of the American statistical association*, 101(476):1418–1429, 2006.

[18] Jianqing Fan and Runze Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American statistical Association*, 96(456):1348–1360, 2001.

[19] Peng Zhao and Bin Yu. On model selection consistency of lasso. *The Journal of Machine Learning Research*, 7:2541–2563, 2006.

[20] Cun-Hui Zhang. Nearly unbiased variable selection under minimax concave penalty. *The Annals of Statistics*, 2010.

[21] Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani, et al. *An introduction to statistical learning*, volume 112. Springer, 2013.

[22] George EP Box and David R Cox. An analysis of transformations. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 26(2):211–243, 1964.

[23] David W Hosmer Jr, Stanley Lemeshow, and Rodney X Sturdivant. *Applied logistic regression*, volume 398. John Wiley & Sons, 2013.

[24] Peter J Green. Iteratively reweighted least squares for maximum likelihood estimation, and some robust and resistant alternatives. *Journal of the Royal Statistical Society: Series B (Methodological)*, 46(2):149–170, 1984.

[25] Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer, 2009.

[26] CL Mallows. Choosing variables in a linear regression: A graphical aid. In *Central Regional Meeting of the Institute of Mathematical Statistics, Manhattan, KS, 1964*, 1964.

[27] Gideon Schwarz. Estimating the dimension of a model. *The annals of statistics*, pages 461–464, 1978.

[28] Hirotugu Akaike. A new look at the statistical model identification. *IEEE transactions on automatic control*, 19(6):716–723, 1974.

[29] Leo Breiman. Better subset regression using the nonnegative garrote. *Technometrics*, 37(4):373–384, 1995.

[30] J. Kenneth Tay, Balasubramanian Narasimhan, and Trevor Hastie. Elastic net regularization paths for all generalized linear models. *Journal of Statistical Software*, 106(1):1–31, 2023.

[31] Patrick Breheny and S Lee. Regularization paths for scad and mcp penalized regression models. *CRAN 3.12. 0*, 2020.

[32] Loann David Denis Desboulets. A review on variable selection in regression analysis. *Econometrics*, 6(4):45, 2018.

[33] AE Horel. Application of ridge analysis to regression problems. *Chemical Engineering Progress*, 58:54–59, 1962.

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

[35] StackExchange. derivation-of-closed-form-lasso-solution, 2011. Forum accessed March 6th 2024.

[36] Hadley Wickham. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2016.

[37] Gilbert Strang. *Introduction to linear algebra*. SIAM, 2022.

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

[39] Jerome Friedman, Trevor Hastie, and Rob Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33(1):1, 2010.

[40] Geoffrey Grimmett and David Stirzaker. *Probability and random processes*. Oxford university press, 2020.

[41] Huan Xu, Constantine Caramanis, and Shie Mannor. Sparse algorithms are not stable: A no-free-lunch theorem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(1):187–193, 2012.

[42] Stephen P Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[43] Sébastien Bubeck. Theory of convex optimization for machine learning. *arXiv preprint arXiv:1405.4980*, 15, 2014.

[44] Vandenberghe Boyd. Subgradients. *Notes for EE364b, Stanford University, Spring 2021-22*, 2022.

[45] Ilya Molchanov. *Theory of random sets*, volume 19. Springer, 2005.

[46] Tibshirani Ryan Gordon Geoff. Coordinate descent, 2012. [Online slides; accessed 11-January-2024].

[47] Peter McCullagh. *Generalized linear models*. Routledge, 2019.

[48] W. N. Venables and B. D. Ripley. *Modern Applied Statistics with S*. Springer, New York, fourth edition, 2002. ISBN 0-387-95457-0.

[49] Sorin Draghici. *Statistics and data analysis for microarrays using R and bioconductor*. CRC Press, 2016.

[50] Tobias Sing, Oliver Sander, Niko Beerenwinkel, and Thomas Lengauer. Rocr: visualizing classifier performance in r. *Bioinformatics*, 21(20):3940–3941, 2005.

[51] Chunling Hu, Steven N Hart, Eric C Polley, Rohan Gnanaolivu, Hermela Shimelis, Kun Y Lee, Jenna Lilyquist, Jie Na, Raymond Moore, Samuel O Antwi, et al. Association between inherited germline mutations in cancer predisposition genes and risk of pancreatic cancer. *Jama*, 319(23):2401–2409, 2018.