Imperial College London
Department of Mathematics

Group Research Project

Subset Selection and Shrinkage Methods in Regression

Group 32

Mengyu (Esther) Zhao CID: 01871903	Rijul Arora CID: 01857554
Shu Wen Hao CID: 01851455	Sruthi Vemuri CID: 01869323
Suxuan Zhang CID : 01852883	Tan Xiao Xuan CID: 01938572

Supervisor: Dr Kolyan Ray

Contents

1 Introduction

As studied in Statistical Modelling I, in a linear regression model, least squares estimators can be used to find the best fit. For example, consider

$$\mathbf{y} = \mathbf{X}\beta + \mathbf{Z}$$

where $\mathbf{y}$ is the response, $\mathbf{X}$ is the design matrix, β is the vector of parameters and $\mathbf{Z}$ is the noise. The Ordinary Least Squares(OLS) estimators is

$$\hat{\beta} = (X^T X)^{-1} X^T y$$

Several drawbacks of OLS estimators make them less preferable, however.

Firstly, the existence of OLS relies on the condition that $X^T X$ has full rank.

Secondly, it is often the case that they do not generalise well, that is, they tend to perform poorly on the 'unseen' testing data after being trained using training data. To be more specific, the OLS found with pairs of design matrices X and the response vectors y might not perform well when the design matrix is changed slightly. This is because OLS could potentially include some parameter with large magnitude so a slight change in X would have a great impact on the response.

Thirdly, although OLS estimators are unbiased, they tend to have high variance and potentially high Mean Squared Error (MSE). We note that the OLS is unstable if $X^T X$ is singular and thus has a large conditional number. As a result, $X^T X$ is sensitive under operations and a slight change in X might lead to a considerable change in $(X^T X)^{-1}$ and therefore a great change in least squares estimators.

What adjustment can be done to avoid these problems?

This report includes various ways of improving the OLS, through discrete subset selection processes and continuous shrinkage methods. Additionally, the report covers simulations and a real data set example, where many of the features are redundant.

2 Discrete Methods

One of the main reasons for subset selection is to focus on the variables with the strongest effect on the response vector. In order to understand the 'broader setting', we are willing to ignore the variables with a small effect, i.e, we preserve only a subset of the variables, and discard the rest of them. There are many different strategies to do this as listed below.

2.1 Forward-Stepwise

In forward-stepwise regression, we begin with the null model. We then add the intercept term in the model, where the intercept is set equal to the mean of the observed y-values. The next variable to be added is the one that minimises the AIC score (**4.5**), under the model that consists of that variable. The model continues adding variables by testing each of the remaining variables at every step. The process stops when adding another variable does not reduce the AIC score anymore.

The **step** function in **R** can be used to perform forward-stepwise selection. We take in the empty linear model as the starting model in the stepwise search, define the full model as the upper bound in the stepwise search and specify the direction of the stepwise search as forward.

```
step(lm(Y~1, data = data), scope = list(upper = lm(Y~., data = data)),
     direction = "forward")
```

2.1.1 Choosing a Variable in the Forward-Stepwise Model (using QR decomposition)

The implementation of forward-stepwise selection relies on the use of QR decomposition. Suppose we know the first k number of predictors of the model, which reduce the residual sum of squares. We create a N x k matrix of the k respective columns of the design matrix X and let this matrix equal X_1. Similarly we let X_2 be the N x $(p-k)$ matrix that corresponds to the rest of the predictors Given that we know the QR decomposition of X_1 ($X_1 = QR$ where Q is a k-dimensional orthogonal matrix and R is an upper triangular matrix), we fit an ordinary least squares linear model with the k variables and obtain the fitted values to be equal to

$$\hat{y} = X_1\hat{\beta}$$

Now from Statistical Modelling I, we know that

$$\hat{\beta} = (X^TX)^{-1}X^Ty$$

$$= (R^TQ^TQR)^{-1}R^TQ^Ty$$

$$= (R^TR)^{-1}R^TQ^Ty$$

$$= R^{-1}R^{-T}R^TQ^Ty$$

$$= R^{-1}Q^Ty$$

Therefore,

$$\hat{y} = QRR^{-1}Q^Ty = QQ^Ty$$

This implies that $\hat{y}$ is in the column span of Q and therefore is also in the span of X_1.

Now, we let z_i be the i^{th} column of the matrix Q for all $1 \leq i \leq k$ where k is the number of rows of Q. As $X_1 = QR$, we have that the column space of X_1 is spanned by $Z_0 = \{z_1, z_2, ..., z_k\}$ and moreover, because R is upper triangular (all columns of X_1 are linear combinations of Z_0), Z_0 is a basis of the column space of X_1. We note that $\hat{y}$ can be though of the orthogonal projection of y to Z_0.
Define x_j to be the $(k-j)^{th}$ column of X_2 and define Z_j to be the set $\{z_1, z_2, ...z_k, v_j\}$, for all $k \leq j \leq p$, where

$$u_j = x_j - \sum_{i=1}^{k} (z_i^T x_j) z_i$$

$$v_j = \frac{u_j}{||u_j||}$$

Therefore the set $\{z_1, z_2, ...z_k, v_j\}$ is an orthonormal basis of Z_j. Now, we can define a new orthogonal projection in terms of Z_0 and $Z_j \setminus Z_0$, call it $\hat{y}_j$, equal to $\hat{y} + (v_j^T y) v_j$. Including the j^{th} variable with decrease the residual sum of squares by $(v_j^T y)^2$, so the variable to pick next in our model is $argmax(|v_j^T y|)$ for all $k \leq j \leq p$ **(John L. Weatherwax and David Epstein, 2021)**.

2.2 Backward-Stepwise

As opposed to forward-stepwise selection, in backward-stepwise selection, we begin with the full model. We then remove the predictor that has the least effect on the fit, which is the variable that minimises the AIC score the most when being removed from the model. The model continues removing variables by testing each of the remaining variables at every step. The process stops when removing another variable does not reduce the AIC score anymore, by a predefined significant level anymore. **(Seber and Lee, 2003)**

Similar to forward-stepwise selection, the **step** function in **R** can also be used to perform backward-stepwise selection. We take in the full linear model as the starting model in the stepwise search, define the empty model as the lower bound in the stepwise search and specify the direction of the stepwise search as backward.

```r
step(lm(Y~., data = data), scope = list(lower = lm(Y~1, data = data)),
     direction = "backward")
```

2.2.1 Dropping a Variable in the Backward-Stepwise Model (using F-statistics)

As we start with the full linear model, we consider dropping a single variable at each step. Under the null hypothesis, this is the same as the model without one of the variables, say the k^{th} variable. We are able to calculate the F-statistic,

$$F = \frac{(RSS_k - RSS_0/(p - p_k)}{RSS_0/(N - p - 1)}$$

where $p - p_k$ represents the difference in the number of parameters under the null and alternative hypothesis, which in this case is equal to 1. Here, a relatively lower F-statistic, F, represents that the null hypothesis should be rejected. We have,

$$F = \frac{(RSS_k - RSS_1)}{RSS_1/(N - p - 1)}$$

This is just a scaling of the difference in RSS values between the null and alternative hypothesis, where the scaling constant is equal to $\frac{N-p-1}{RSS_1}$. We compute the F-statistic for all $1 \leq k \leq p$ and remove the variable with the lowest F-statistic **(P.T. Pope and J.T. Webster)**.

2.3 Drawbacks of Forward and Backward-Stepwise Methods

Firstly, we note that forward-stepwise regression is computationally inefficient. It has p choices to start with, $p-1$ in the next one and so on. We can find the algorithmic complexity by considering the sum:

$$p + (p-1) + ... + 2 + 1 = \frac{p(p-1)}{2}$$

so $O(p^2)$. On the other hand, this method is still better than iterating through every single possible subset of features (best subset selection), which has an algorithmic complexity of $O(2^p)$. As backward-stepwise regression is just the inversion of forward-stepwise while starting with the full model and removing parameters, we get that it has the same complexity.

One of the main disadvantages of using forward-stepwise is that it is a greedy algorithm as it builds on a sequence of nested models. It makes a hard selection on the next variable and once the respective weight is assigned to the variable, it cannot be changed. This makes the model optimal locally at each step, but sub-optimal in the long run (for the whole model). This is why we prefer stepwise selection (**2.4**) as the weights are always subject to change depending on the current state of the model.

Additionally, when if two variables have a high correlation, both these methods fail to give the optimal linear model. What is bound to happen is that one of the variables will get chosen to be in the regression model, and will probably have a weight too large in magnitude. Now, at the next step when we consider the other variable, the decrease in RSS will be minimal because of the correlation. Depending on this decrease in RSS, forward and backward-regression might not be able to pick up this other variable, resulting in the loss of potentially valuable data and this model would not perform well on new data.

2.4 Stepwise

Stepwise selection is the combination of both forward-stepwise selection and backward-stepwise selection. Essentially, it is a modification to forward-stepwise in the sense that once a variable has been added, the significance level of the remaining variables is compared to a pre-defined threshold level. Each variable has 2 significance levels; one for adding variables and one for removing variables. If a variable is deemed non-significant at any step, it is removed from the model. It may be the case that once a variable is removed, it gets added on at a later step.

```
step(lm(Y~1, data = data), scope = list(lower = lm(Y~1, data = data),
     upper = lm(Y~., data = data)), direction = "both")
```

2.5 Forward-Stagewise (Incremental)

Forward-stagewise regression is a version of least squares boosting for multiple linear regression, which is used to find a group of independent variables which have significant influence on the dependent variables through a series of tests, like F-tests and t-tests. This calculation is achieved iteratively.

It is similar to lasso regression. By selecting an appropriate iteration number and step size, the coefficient of some variables can be compressed to 0, which can play a role in variable selection and variance reduction. Therefore, it is more helpful when the data is high-dimensional.

The simple algorithm is shown as follows:

1. Start with $\mathbf{r} = \mathbf{y}$, $\beta_1, \beta_2, ...\beta_p = 0$, where $\mathbf{r}$ is the residual.
2. Find the predictor $\mathbf{x}_j$ most correlated with $\mathbf{r}$.
3. Update $\beta_j \leftarrow \beta_j + \delta_j$, where $\delta_j = \epsilon \cdot \text{sign}\langle \mathbf{r}, \mathbf{x}_\text{j} \rangle$ with $\epsilon > 0$ a small step size.
4. Set $\mathbf{r} \leftarrow \mathbf{r} - \delta_j \mathbf{x}_j$ and repeat steps 2 and 3 many times, until the residuals are uncorrelated with all the predictors.

The **lars** function from the **lars** package in **R** can be used to perform forward stagewise by stating the type as forward.stagewise. The function takes in the parameters X (matrix of predictors) and Y (vector of response) **(Hastie and Efron, 2013)**.

```
lars(X, Y, type = "forward.stagewise")
```

3 Continuous Shrinkage Methods

Consider

$$\mathbf{y} = \mathbf{X}\beta + \mathbf{Z}$$

where $\mathbf{y}$ is the response, $\mathbf{X}$ is the design matrix, β is the vector of parameters and $\mathbf{Z}$ is the noise. From the expression, one can see that if the norm of β is large, a slight change in the design matrix would likely result in a great change in the response, which is not ideal as this results in a poor performance on testing data. In order to fix the problem of over-fitting, we introduce penalties on the size of vector of parameters. There are four methods that are widely used: Ridge regression uses L2 norm; lasso regression uses L1 norm; elastic-net regression uses a combination of ridge regression and lasso regression; and the least angle regression is an adaption of lasso regression. Discrete methods, since they either keep or discard variables, often result in high variance, and prediction error of the model; this is mitigated by the use of more continuous methods.

3.1 Ridge Regularisation

The ridge regularisation is defined as:

$$\hat{\beta}^{ridge} = \underset{\beta}{argmin}\Big\{\sum_{i=1}^{N}(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2\Big\}, \text{ subject to } \sum_{j=1}^{p}\beta_j^2 \leq t$$

or equivalently in the Lagrangian form:

$$\hat{\beta}^{ridge} = \underset{\beta}{argmin}\Big\{\sum_{i=1}^{N}(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda\sum_{j=1}^{p}\beta_j^2\Big\}$$

An L1 penalty is put on the size of the parameters. Minimising the expression in the curly brackets by partial differentiation with respect to a vector gives that $\hat{\beta}^{ridge} = (X^TX + \lambda I)^{-1}X^T\mathbf{y}$, which is linear in Y and exists even in the case where X does not have full rank. As suggested by **Hastie et al. (2009)**, its key properties include that ridge regularisation shrinks different parameters by potentially different proportions and it cannot set a parameter to zero and thus it cannot be used to select parameters, where the former can be deduced from Singular Value Decomposition (SVD) and the latter can been seen by considering $\hat{\beta}^{ridge}$ element-wise. The detailed proof is given in B. As **Taboga (2021)** has proved, a ridge estimator always has a smaller Mean Squared Error than an Ordinary Least Squares estimator with detailed proof provided in A.

3.2 Lasso Regularisation

The Least Absolute Shrinkage Selector Operator, or lasso regularisation, is a similar shrinkage method to ridge regression, whose estimate is defined as:

$$\hat{\beta}^{lasso} = \underset{\beta}{argmin}\Big\{\sum_{i=1}^{N}(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2\Big\}, \text{ subject to the condition } \sum_{i=1}^{p}|\beta_j| \leq t$$

or equivalently, in its Lagrangian form:

$$\hat{\beta}^{lasso} = \underset{\beta}{argmin}\Big\{\frac{1}{2}\sum_{i=1}^{N}(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda\sum_{j=1}^{p}|\beta_j|\Big\}$$

Here, the L2 norm in ridge regression has been replaced by the L1 norm, and, unlike in ridge regression, $|\beta_j|$ are not linear in y_i and so, unlike in ridge regression, we cannot use partial differentiation on the expression above to find a closed form solution. In addition to mitigating the problem of over-fitting, since some of the coefficients can be set to exactly zero, if λ is made sufficiently small, this method also helps with feature selection, and gives a simpler, more easily-interpreted model. Thus, the lasso tends to perform well, and to outperform the ridge, as we will see in the numerical simulations, when the true model is sparse, ie. there is a small number of true significant parameters.

When predictors are correlated, the lasso tends to select just one from a group of correlated variables, and give full weight to it, whereas the ridge tends to group them together, and shrink them towards each other. By the triangle inequality, the third term is always non-negative, and so is minimised when it is zero, which happens exactly when all the components of the two β vectors have the same sign. So, finally, for a minimal solution, the sum of the two β vectors must be a, and the components of all of the significant parameters must have the same sign.

One drawback of the lasso regularisation is that it tends to shrink the non-zero coefficients to zero, and in general, the predicted values do not converge to their true parameters as the sample size gets large (not consistent). To reduce this type of bias and improve the convergence rate of the lasso, particularly when p is large, we use the relaxed lasso method, which involves first using the lasso to identify the set of significant parameters, and then applying the lasso again only to this set.

Cross-validation is used to find the tuning parameter at each stage, and λ generally decreases after the first stage, since the noise variables have been eliminated. A smaller λ in the second step should, in turn, lead to a smaller bias, reducing the shrinkage of these parameters and the instability of the lasso method. [**Hastie** , **Tibshirani**, **Friedman**, **2001**].

3.3 Elastic-net Regularisation

Apart from L2 norm in ridge regularisation and L1 norm in lasso regularisation, L_q norm for $q \in (1, 2)$ can also be used in the penalty term. A disadvantage of using L_q penalty, however, is that it cannot set parameters to zero as the constraint region $\sum_{j=1}^{p} |\beta_j|^q \leq t$ has differentiable corners. To fix this problem, the elastic-net regularisation is introduced:

$$\hat{\beta}^{elastic-net} = \underset{\beta}{argmin}\Big\{ \sum_{i=1}^{N}(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} \big(\alpha\beta_j^2 + (1-\alpha)|\beta_j|\big) \Big\}$$

Elastic-net regularisation uses a combination of L1 norm and L2 norm as penalty, and originally emerged due to the instability of the lasso, resulting from the high dependency on the data for variable selection. Loosely speaking, the hyper-parameter λ decides the size of shrinkage and α decides whether it is more similar to ridge regularisation or lasso regularisation. **(Hastie, Tibshirani and Friedman, 2009)**

Since ridge regularisation tends to group correlated parameters and shrink them together, and lasso regularisation can set parameters to zero, elastic-net regularisation inherits both advantages. To be more specific, it tends to firstly select parameters, group highly correlated ones and then shrink them together.

In fact, we can turn the elastic net optimisation problem:

$$\underset{\beta}{min}\|\mathbf{y} - \mathbf{X}\beta\|^2 + \lambda\Big(\alpha\|\beta\|_2^2 + (1-\alpha)\|\beta\|_1\Big)$$

where $\|\beta\|_1$, $\|\beta\|_2$ are the L1 and L2 norms of β respectively, into a lasso problem, by augmenting $\mathbf{y}$ and $\mathbf{X}$.

Consider the augmented matrix

$$\tilde{\mathbf{X}} = \begin{bmatrix} \mathbf{X} \\ a\mathbf{I}_p \end{bmatrix}$$

where a is a constant.

So that $\tilde{\mathbf{X}}\beta = \begin{bmatrix}\mathbf{X}\beta \\ a\beta\end{bmatrix}$.

Now we augment the vector $\mathbf{y}$ with a vector of p zeroes:

$$\tilde{\mathbf{y}} = \begin{bmatrix}\mathbf{y} \\ \mathbf{0}_p\end{bmatrix}$$

which gives,

$$\|\tilde{\mathbf{y}} - \tilde{\mathbf{X}}\beta\|_2^2 = \left\|\begin{matrix}\mathbf{y} - \mathbf{X}\beta \\ a\beta\end{matrix}\right\|_2^2 = \|\mathbf{y} - \mathbf{X}\beta\|_2^2 + a^2\|\beta\|_2^2 \tag{1}$$

In this augmented case, the lasso problem can equivalently be written as the following **(Tibshirani, 1996)**:

$$\hat{\beta} = \underset{\beta}{argmin}\Big\{\|\tilde{\mathbf{y}} - \tilde{\mathbf{X}}\beta\|_2^2 + \tilde{\lambda}\|\beta\|_1\Big\} = \underset{\beta}{argmin}\Big\{\|\mathbf{y} - \mathbf{X}\beta\|_2^2 + a^2\|\beta\|_2^2 + \tilde{\lambda}\|\beta\|_1\Big\}$$

where the second equality follows from 1. By inspection, let $a^2 = \lambda\alpha$, $\tilde{\lambda} = \lambda(1-\alpha)$, to yield the original elastic-net problem. Therefore, by augmenting the original $\mathbf{X}$ and $\mathbf{y}$ as shown, calculating $\tilde{\lambda}$ and solving the lasso problem with $\tilde{\mathbf{y}}$, $\tilde{\mathbf{X}}$ and $\tilde{\lambda}$, we obtain a solution to the elastic net problem. So this elastic net optimisation problem is equivalent to a lasso problem on the augmented data set: thus, the LAR algorithm (used to solve lasso) can be used to generate the elastic net solution path, using the same order of computations as a single OLS fit. **(Zou and Hastie, 2005)**

3.4 Least Angle Regression

The least angle regression (LAR) introduced by Efron et al. (2004, p.407) has a similar strategy as forward-stepwise regression. It also starts with all coefficients set to be zero and the predictor variables are standardised with mean zero and unit variance. Then it finds the variable that is most correlated with the response (say x_1), and moves in the direction of this variable continuously until another variable (say x_2) has the same amount of correlation with the current residual as the first one. Then instead of moving in the direction of x_2, we move in a direction that is equiangular between x_1 and x_2, until a third variable x_3 is as much correlated with the current residual. The process continues until all the variables are in the model **(Hastie and Tibshirani, 2009)**.

It was also proved by **Efron et al. (2004, p.417-421)** that under certain modifications, the LAR algorithm can produce all Lasso and stagewise solutions with a lower computation cost, we will see a more detailed comparison between Lasso, stagewise regression and LAR later.

Let $\mathcal{A}_k$ be the active set (the predictors included in the model) at the beginning of the **k**-th step, $\beta_{\mathcal{A}_k}$ be the vector of parameters and $\mathbf{r}_k = \mathbf{y} - \mathbf{X}_{\mathcal{A}_k}\beta_{\mathcal{A}_k}$ be the current residual. Then the direction along which we will be moving at this step is $\delta_k = (\mathbf{X}_{\mathcal{A}_k}^T\mathbf{X}_{\mathcal{A}_k})^{-1}\mathbf{X}_{\mathcal{A}_k}^T\mathbf{r}_k$. Let α be a real number such that $0 \leq \alpha \leq 1$. Then the coefficient profile evolves in the fashion described by $\beta_{\mathcal{A}k}(\alpha) = \beta_{\mathcal{A}k} + \alpha \cdot \delta_k$ where α increases from 0 to 1. Let the fit vector at the beginning of this step be $\hat{\mathbf{f}_k}$, then the fit vector evolves as $\hat{\mathbf{f}_k}(\alpha) = \hat{\mathbf{f}_k} + \alpha \cdot \mathbf{u}_k$ where $\mathbf{u}_k = \mathbf{X}_{\mathcal{A}_k}\delta_k$ is the new fit direction. We will prove three facts in the following:

1. For all the predictors in the active set, their correlations with the current residual are identical and decreasing.

2. The new fit direction $\mathbf{u}_k$ makes the least and equal angle with all predictors in the active set.

3. The coefficient profile is piece-wise linear.

3.4.1 Correlations Tied and Monotone Decreasing

For the first claim, we need to work out the expression of the correlation of each variable with the residuals in terms of α as we move along the new direction.
Consider a regression problem with all variables and response having mean zero and standard deviation one. Suppose also that each variable has identical absolute correlation with the response **(Hastie et al. 2009, p.97)**:

$$\frac{1}{N}|\langle \mathbf{x}_j, \mathbf{y}\rangle| = \lambda \quad j = 1, ..., p$$

Let $\hat{\beta}$ be the Ordinary Least Squares(OLS) estimator and $\mathbf{u}(\alpha) = \alpha\mathbf{X}\hat{\beta}$ be a fraction towards the least squares fit $\mathbf{u}$ for $0 \leq \alpha \leq 1$. The correlation of variable $\mathbf{x}_j$ with the residual $\mathbf{y} - \mathbf{u}(\alpha)$ is given by

$$\frac{\langle \mathbf{x}_j, \mathbf{y} - \mathbf{u}(\alpha)\rangle}{N}\sqrt{\frac{\langle \mathbf{x}_j, \mathbf{x}_j\rangle}{N}\frac{\langle \mathbf{y} - \mathbf{u}(\alpha), \mathbf{y} - \mathbf{u}(\alpha)\rangle}{N}}$$

We aim to prove that this quantity is independent of the value of j. It is easy to check that the numerator has absolute value equal to $(1-\alpha)\lambda \quad \forall j = 1, ..., p$

$$\begin{aligned}
\frac{|\langle \mathbf{x}_j, \mathbf{y} - \mathbf{u}(\alpha)\rangle|}{N} &= \frac{1}{N}\Big|\langle \mathbf{x}_j, \mathbf{y}\rangle - \langle \mathbf{x}_j, \mathbf{u}(\alpha)\rangle\Big| \\
&= \frac{1}{N}\Big|\langle \mathbf{x}_j, \mathbf{y}\rangle - \alpha\langle \mathbf{x}_j, \mathbf{X}\hat{\beta}\rangle\Big| \\
&= \frac{1}{N}\Big|\langle \mathbf{x}_j, \mathbf{y}\rangle - \alpha\langle \mathbf{x}_j, \mathbf{y}\rangle + \alpha\langle \mathbf{x}_j, \mathbf{e}\rangle\Big| \\
&= (1-\alpha)\frac{1}{N}|\langle \mathbf{x}_j, \mathbf{y}\rangle| \\
&= (1-\alpha)\lambda
\end{aligned}$$

Since the variables all have unit norm, $\langle \mathbf{x}_j, \mathbf{x}_j\rangle = 1$.

$$\begin{aligned}
&\langle \mathbf{y} - \mathbf{u}(\alpha), \mathbf{y} - \mathbf{u}(\alpha)\rangle \\
&= \mathbf{y}^T\mathbf{y} - 2\alpha\mathbf{y}^T\mathbf{X}\hat{\beta} + \alpha^2(\mathbf{X}\hat{\beta})^T(\mathbf{X}\hat{\beta}) \\
&= \mathbf{y}^T\mathbf{y} - 2\alpha\mathbf{y}^T\mathbf{X}\hat{\beta} + \alpha^2\mathbf{y}^T(\mathbf{X}\hat{\beta}) && \textit{since } (\mathbf{y} - \mathbf{X}\hat{\beta})^T\mathbf{X}\hat{\beta} = 0 \textit{ implies } \mathbf{y}^T\mathbf{X}\hat{\beta} = (\mathbf{X}\hat{\beta})^T\mathbf{X}\hat{\beta} \\
&= \mathbf{y}^T\mathbf{y} + \alpha(\alpha - 2)\mathbf{y}^T\mathbf{X}\hat{\beta} \\
&= (\mathbf{y}^T\mathbf{X}\hat{\beta} + \mathrm{RSS}) + \alpha(\alpha - 2)\mathbf{y}^T\mathbf{X}\hat{\beta} && \textit{when } \alpha = 1, \langle \mathbf{y} - \mathbf{u}(\alpha), \mathbf{y} - \mathbf{u}(\alpha)\rangle = \mathrm{RSS} = \mathbf{y}^T\mathbf{y} - \mathbf{y}^T\mathbf{X}\hat{\beta} \\
& && \textit{implies } \mathbf{y}^T\mathbf{y} = \mathbf{y}^T\mathbf{X}\hat{\beta} + \mathrm{RSS} \\
&= (1-\alpha)^2\Big(\mathbf{y}^T\mathbf{y} - (\mathbf{X}\hat{\beta})^T(\mathbf{y} - \mathbf{X}\hat{\beta}) - (\mathbf{y} - \mathbf{X}\hat{\beta})^T(\mathbf{y} - \mathbf{X}\hat{\beta})\Big) + \mathrm{RSS} \\
&= (1-\alpha)^2\big(N - \mathrm{RSS}\big) + \mathrm{RSS} && \textit{since } \mathbf{y} \textit{ has zero mean and unit standard deviation} \\
& && \implies \mathbf{y}^T\mathbf{y} = N \\
&= (1-\alpha)^2N + \alpha(2-\alpha)\mathrm{RSS}
\end{aligned}$$

Therefore the magnitude of the correlation of variable $\mathbf{x}_j$ with the residual $\mathbf{y} - \mathbf{u}(\alpha)$ is

$$\frac{|\langle \mathbf{x}_j, \mathbf{y} - \mathbf{u}(\alpha)\rangle|}{N}\sqrt{\frac{\langle \mathbf{x}_j, \mathbf{x}_j\rangle}{N} \cdot \frac{\langle \mathbf{y} - \mathbf{u}(\alpha), \mathbf{y} - \mathbf{u}(\alpha)\rangle}{N}} = (1-\alpha)\lambda\sqrt{(1-\alpha)^2 \cdot \frac{\alpha(2-\alpha)}{N} \cdot \mathrm{RSS}}$$

The expression is independent of j so every variable x_j has the same magnitude of correlation with the residual.
We now prove that this is a monotone decreasing function in α so as α increases from 0 to 1, the absolute value of the correlation decreases from λ to 0. We can rewrite the expression by dividing

both the numerator and the denominator by $1-\alpha$:

$$\lambda\sqrt{1+\frac{\alpha(2-\alpha)}{(1-\alpha)^2}\cdot\frac{\mathrm{RSS}}{N}}$$

After some algebra, we can see that $\frac{\alpha(2-\alpha)}{(1-\alpha)^2}$ is a monotone increasing function in α with range $(0,\infty)$, thus the absolute value of the correlation is a monotone decreasing function in α with range $(0,1)$. To conclude, each variable in the active set has correlation with the current residual equal in magnitude and the absolute value of the correlation decreases as the coefficient profile evolves.

3.4.2 Least and Equal Angle between New Fit Direction and Active Variables

For the second claim, we start with the geometric definition of the dot product of two Euclidean vectors:

$$\mathbf{a}\cdot\mathbf{b}=\|\mathbf{a}\|\ \|\mathbf{b}\|\cos\theta$$

where θ is the angle between a and b.
Therefore in order to prove that each variable in the active set makes equal angle with the new direction $\mathbf{u}_k$, it suffices to prove that $\langle\mathbf{x}_j,\,\mathbf{u}_k\rangle$ is identical for all j such that $\mathbf{x}_j\in\mathcal{A}_k$.
With the notation introduced earlier, we have

$$\begin{aligned}\delta_k &= (\mathbf{X}_{\mathcal{A}_k}^T\mathbf{X}_{\mathcal{A}_k})^{-1}\mathbf{X}_{\mathcal{A}_k}^T\mathbf{r}_k\\ \beta_{\mathcal{A}k}(\alpha) &= \beta_{\mathcal{A}k}+\alpha\cdot\delta_k\\ \mathbf{u}_k &= \mathbf{X}_{\mathcal{A}_k}\delta_k\\ \hat{\mathbf{f}}_k(\alpha) &= \hat{\mathbf{f}}_k+\alpha\cdot\mathbf{u}_k\end{aligned}$$

Note that $\mathbf{x}_j^T\mathbf{u}_k$ is the j-th entry of $\mathbf{X}_{\mathcal{A}_k}^T\mathbf{u}_k$.
We can see the inner product of $\mathbf{X}_{\mathcal{A}_k}$ and $\mathbf{u}_k$ is the same as that of $\mathbf{X}_{\mathcal{A}_k}$ and $\mathbf{r}_k$ as following:

$$\begin{aligned}\mathbf{X}_{\mathcal{A}_k}^T\mathbf{u}_k &= \mathbf{X}_{\mathcal{A}_k}^T(\mathbf{X}_{\mathcal{A}_k}\delta_k)\\ &= \mathbf{X}_{\mathcal{A}_k}^T\mathbf{X}_{\mathcal{A}_k}(\mathbf{X}_{\mathcal{A}_k}^T\mathbf{X}_{\mathcal{A}_k})^{-1}\mathbf{X}_{\mathcal{A}_k}^T\mathbf{r}_k\\ &= \mathbf{X}_{\mathcal{A}_k}^T\mathbf{r}_k\end{aligned}$$

By fact 1, each variable in the active set has the same correlation with the current residual, thus $\mathbf{X}_{\mathcal{A}_k}^T\mathbf{r}_k$ has identical entries, so does $\mathbf{X}_{\mathcal{A}_k}^T\mathbf{u}_k$. Therefore each variable in the active set makes the identical angle with the new direction.
By the LARS algorithm, a new variable x_j enters the active set when the absolute value of its correlation with the current residual catches up with those of the variables already in the active set. In other words, x_j is the variable with the largest magnitude of the correlation that is not in the active set. Since cosign is decreasing within domain $(-\pi,\pi)$, a larger magnitude of inner product corresponds to a smaller angle. Thus x_j makes the least angle with the residual and therefore with the new direction $\mathbf{u}_k$.

3.5 Comparison between Different Shrinkage Methods

3.5.1 Ridge and Lasso

- Ridge regularisation is linear in Y whereas lasso regularisation is not.
- Ridge regularisation shrinks different components by different proportions whereas lasso shrinks all the components by a constant.
- Ridge regularisation cannot shrink a component to zero whereas lasso regularisation can, thus lasso regularisation is more optimal in a sparse setting where a large proportion of the parameters are zero and ridge regularisation is more optimal in a dense setting where many parameters are non-zero.

As **Starmer** has suggested, the reason why parameters can be set to zero by lasso regularisation can also be explained geometrically. For simplicity of demonstration we only consider a 2D case.

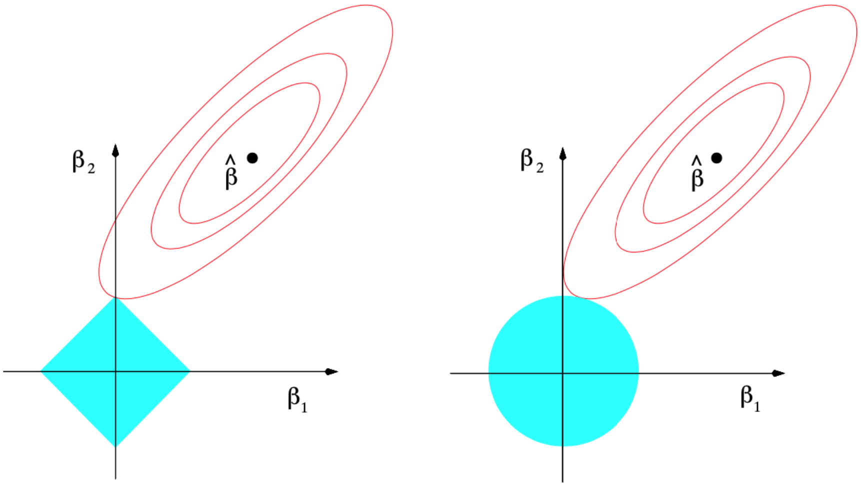


Figure 1: Contours and Constraint Regions of Ridge and Lasso Estimators

The black dot represents the Ordinary Least Squares Estimators $\hat{\beta}$. The red elliptic curves represent the contours of RSS of lasso estimators(left) and ridge estimators(right), that is, combinations of estimators on the same elliptic curve results in an identical RSS value. The light blue regions stand for the constraint region. For lasso the constraint $\sum_{i=1}^{p}|\beta_j| \leq t$ results in a diamond, and for ridge the constraint $\sum_{j=1}^{p}\beta_j^2 \leq t$ results in a disc. The minimising problem is equivalent to finding a point in the constraint region at which the RSS is as least as possible, and we can consider expanding the RSS contours and find the first time the elliptic curve hits the constraint region. It can be proved that for a diamond shape, solutions often exist at the corners, whereas for a disc, the solution does not have to be where the circle intersects the axes. In higher dimensional settings, the constraint region will have many corners, edges and faces, at which some parameters would be zero. Therefore it is much more likely for a lasso estimator to hit zero.

3.5.2 Lasso and LAR

Lasso regularisation and LAR have coefficient profiles that are almost identical: they only begin to differ after a parameter hits zero. This similarity can be seen by considering the conditions that have to be fulfilled for a variable in and not in the active set **(Hastie . 2009)**.
By the nature of the LAR algorithm, for a variable $\mathbf{x}_j$ to be added into the active set $\mathcal{A}$, it must satisfy the condition that its correlation with the current residual is equal to the current common correlation as explained in 3.4.1. On the other hand, for all $\mathbf{x}_j \notin \mathcal{A}$, they all absolute correlations less than the current common value. Assume that the design matrix X has been standardised, then the correlation is just the inner product. Using vector notation, this is equivalent to:

$$\mathbf{x}_j{}^T(\mathbf{y} - \mathbf{X}\beta) = \gamma \cdot s_j \quad \forall \mathbf{x}_j \in \mathcal{A} \tag{2}$$

$$\mathbf{x}_k{}^T(\mathbf{y} - \mathbf{X}\beta) \leq \gamma \cdot s_j \quad \forall \mathbf{x}_k \notin \mathcal{A} \tag{3}$$

where γ is the common value of absolute correlation and s_j is the sign of the inner product.
Now we consider the lasso criterion

$$\begin{aligned} R(\beta) &= \frac{1}{2}\sum_{i=1}^{N}(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda\sum_{j=1}^{p}|\beta_j| \\ &= \frac{1}{2}\|\mathbf{y} - \mathbf{X}\beta\|_2^2 + \lambda\|\beta\|_1 \end{aligned}$$

Consider the partial derivative(which exists only when $\beta_j \neq 0$ i.e. the corresponding variable $\mathbf{x}_j \in \mathcal{B}$ where $\mathcal{B}$ denotes the active set)

$$\begin{aligned}\frac{\partial R}{\partial \beta_j} &= \sum_{i=1}^{N}(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)(-x_{ij}) + \lambda \cdot sign(\beta_j) \\ &= -\mathbf{x}_j^T(\mathbf{y} - \mathbf{X}\beta) + \lambda \cdot sign(\beta_j)\end{aligned}$$

Setting the partial derivative to zero gives

$$\mathbf{x}_j^T(\mathbf{y} - \mathbf{X}\beta) = \lambda \cdot sign(\beta_j) \quad \forall \mathbf{x}_j \in \mathcal{B} \tag{4}$$

On the other hand, if $\mathbf{x}_k \notin \mathcal{B}$, then $\beta_k = 0$ and $R(\beta)$ is not differentiable with respect to β_k but we can used the generalised concept of sub-derivative. Replacing the derivative by a sub-derivative gives that

$$|\mathbf{x}_k^T(\mathbf{y} - \mathbf{X}\beta)| \leq \lambda \quad \forall \mathbf{x}_k \notin \mathcal{B} \tag{5}$$

Comparing (2) with (4), and (3) with (5) give that lasso and LAR are identical if the sign of the inner product s_j agrees with the sign of the coefficient β_j.

3.5.3 Forward-stagewise, Lasso and LAR

Interestingly, some similarities can be seen between the coefficient profiles of (infinitesimal) forward-stagewise, lasso and LAR.

As described in 2.5, forward-stagewise updates the coefficient by a small fraction, keeping the signs of the coefficient and the correlation of the variable and the residual identical, therefore the coefficient evolves monotonically. In contrast, when LAR algorithm is applied, the coefficients do not have to evolve monotonically since it is possible that they move opposite to the joint correlation.

In the situation where coefficients evolve monotonically, the coefficient profiles of infinitesimal forward-stagewise and LAR are identical. Since monotonicity implies that no coefficient hits zero in an intermediate step, lasso and LAR also have identical profiles, as discussed in 3.5.2. If the monotonicity condition is not fulfilled, forward-stagewise would look different from lasso and LAR.

4 Measures of Prediction Accuracy

There are various ways of measuring the prediction accuracy on the testing data. In this project we adopted the Residual Sum of Squares(RSS), the Euclidean norm between the estimator and the true parameter, and the True Positive Rate (TPR).

4.1 Mean Squared Error(MSE)

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(P_i - O_i)^2$$

where P_i are the predicted values, and O_i are the observed values.
RSS is a measure of how close the fitted value is to the response, so it tells us how well the model fits for the current data set.

4.2 Euclidean (L2) Norm of $\hat{\beta} - \beta$

$$\sqrt{\sum_{i=1}^{p}(\hat{\beta}_i - \beta_i)^2}$$

L2 norm is a measure of how close the estimator of the parameter is to the true parameter (it is therefore unknown in a real data setting). In other words, L2 norm, to some extent, indicates how much the fitted value would change if the design matrix is changed, i.e. how well the model generalises. The Euclidean norm is used in conjunction with the RSS, to avoid over-fitting, so that the model can be used for predicting with new data, as well as the training data.

4.3 True Positive Rate (TPR)

The True Positive Rate, also called sensitivity, is calculated as the probability that a true positive will be detected as such:

$$TPR = \frac{TP}{P} = \frac{TP}{TP + FN}$$

where TP is the number of true positive cases and P is the number of positive cases, which consists of true positive and false negative cases. In this setting, a true positive case is where both the parameter and the estimator is non-zero, and a false negative case is where the estimator is zero and the parameter is non-zero. The TPR of a model therefore measures the rate with which it correctly identifies a significant (non-zero) parameter - a TPR value of 1 is ideal.

4.4 False Discovery Rate (FDR)

The False Discovery Rate is the rate of type I errors - the proportion of coefficients that are found to be significant (non-zero), whose true parameters are zero - a low FDR is therefore ideal.

$$FDR = \frac{FP}{FP + TP}$$

where FP is the number of false positive cases and TP is the number of true positive cases. In this setting, a true positive case is where both the parameter and the estimator is non-zero, and a false positive case is where the estimator is non-zero and the parameter is zero. TPR and FDR, together, give an idea of how effectively the model correctly selects significant parameters - variable selection improves and so is important for the prediction accuracy (L2 norm) of the fitted model **(Zou, 2006)**.

4.5 Akaike Information Criterion (AIC)

$$AIC = 2K - 2ln(L)$$

where K represents the number of parameters (or number of non-zero entries in the 'true' beta vector) and L represents the likelihood of obtaining the observed y-values under the current model. Given a number of models, the model with the lowest AIC score is the desired one (as we want L to be maximised). Additionally, we note that adding too many parameters will increase the AIC, i.e, this score takes overfitting into consideration and discourages it by raising the AIC.
In regression models, it is recommended to use the following formula for the AIC instead:

$$AIC = 2K + n(ln(\frac{RSS}{n}))$$

where n represents the number of observations. Showing these two formulations are equivalent can be done by considering the maximum likelihood estimate of the log-likelihood function.

4.6 Mean Absolute Error

The Mean Absolute Error (MAE) is calculated as:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|P_i - O_i|$$

where P_i are the predicted values, and O_i are the observed values. This is a measure of the average absolute difference between predicted and observed values. This is comparable to the Root Mean Square Error ($\sqrt{MSE}$), using the L1 norm, where the RMSE uses the L2 norm, but, since the errors are squared in the RMSE, a larger error will contribute more to the overall RMSE than to the MAE, skewing the overall measure due to possible outliers. The MAE is, therefore, useful when we do not want larger errors to have a significantly greater effect on the overall sum of errors than smaller ones.

4.7 Coefficient of Determination - R^2

R^2 = 1 - $\frac{SS1}{SS2}$, where

$$SS1 = \sum_{i=1}^{n}(P_i - O_i)^2$$

and

$$SS2 = \sum_{i=1}^{n}(O_i - \bar{O_i})^2$$

ranging from 0 to 1 is a measure of the proportion of variation in the response variable which can be attributed to the independent parameters in the model. It measures how well the model fits the data, with 1 indicating a perfect fit. RSS and R^2 are both measures of how well the model fits the training data, but whereas RSS is an absolute measure, R^2 is a scaled, as a proportion of the total variance.

5 Numerical Simulations

(Tibshirani, 1996)

5.1 Case 1: n = 100, p = 50, real p =15

5.1.1 Dataset

To compare the different discrete methods, we decided to analyse a simple dataset with 100 observations and 50 variables, where the ’true’ beta vector is known. Firstly, we generated a design matrix where each entry was sampled from a $N \sim (0, 1)$ i.i.d. Additionally, we created a beta vector with 35 of the entries being 0 and the rest of the 15 entries being sampled from a $Unif \sim (1, 3)$. Then let $Y = XB + Z$ where Z represents a 100-variate random vector of errors (noise), with each entry being sampled from a $N \sim (0, 1)$ i.i.d, and Y is the response vector.

```r
set.seed(42)
X <- matrix(rnorm(100*50), nrow = 100, ncol = 50)
B <- sample(c(replicate(35, 0), runif(n = 15, min = 1, max = 3)
set.seed(10)
Y <- X %*% B + rnorm(100)
```

5.1.2 Results

Discrete Methods	L2 Norm	MSE	AIC	TPR	FDR
Forward-stepwise	0.579	0.301	272.827	1	0.25
Backward-stepwise	0.685	0.330	273.668	1	0.348
Stepwise	0.579	0.301	272.827	1	0.25
Forward stagewise	1.004	0.417	NaN	1	0.7

Continuous Methods	L2 Norm	MSE	TPR	FDR
Ridge	2.736	7.466	1	0.7
Lasso	0.956	1.940	1	0.348
Relaxed Lasso	2.047	3.142	1	0.7
Elastic-net($\alpha = 0.1$)	1.393	2.568	1	0.659
Elastic-net($\alpha = 0.3$)	1.112	2.152	1	0.571
Elastic-net($\alpha = 0.5$)	0.908	1.793	1	0.516
Elastic-net($\alpha = 0.7$)	0.843	1.700	1	0.464
Elastic-net($\alpha = 0.9$)	0.857	1.760	1	0.375
LARS(step 22)	1.104	2.147	1	0.286

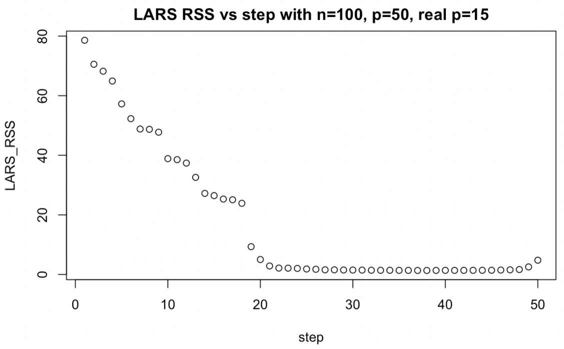


Figure 2: LARS RSS vs step for Case 1

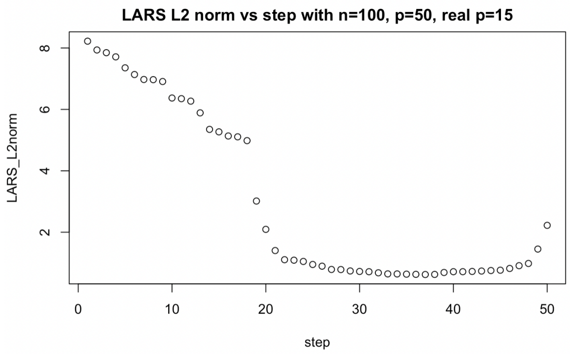


Figure 3: LARS L2norm vs step for Case 1

Above are figures showing the MSE and L2 norm generated by the LARS algorithm at each step. We choose to report the MSE and L2 norm data at step 22 as we could see from the plot that the MSE gets relatively steady after step 22.
Since we know the true β in the simulated data, we know in advance that at around step 15 (which is the number of non-zero parameters in the true β), we should observe a drop in the MSE, although this trend is not so clear in this case.

5.1.3 Discussion

Observing how the discrete methods perform on the simulated data, we note that the predicted β vector from forward-stepwise has the best L2 norm. This implies that the predicted β is closest in distance to the 'true' β and also a low L2 norm also suggests that the predicted β would provide a more accurate fit to new data, i.e. would have a better testing accuracy. Additionally, ranking these methods based on L2 norm is in line with the MSE, AIC and FDR. Forward-stagewise gives the worst MSE score, which means it has the highest RSS, highlighting the high bias of the model. It also has the highest FDR, which means that the model produces a high proportion of type I errors.

Moreover, the continuous shrinkage methods are all less effective in terms of fitting the data (greater RSS), and prediction accuracy (L2 norm) than the discrete methods. Of the continuous methods, the elastic net with $\alpha = 0.7$, produces the best result, with respect to both testing accuracy (L2 norm) and fitting the data (RSS), but also gives a high FDR - a higher proportion of non-zero coefficients, whose true values are zero, resulting in an over-parameterised model, although we can guess that the magnitudes of these coefficients are small, as the RSS and L2 norm are still small. On all counts (RSS, L2 norm, FDR, TPR), the lasso noticeably outperforms the ridge regularisation, and, with respect to variable selection (TPR and FDR), all tested variants of the elastic net regularisation. The lasso tends to outperform the ridge when, as in this case, there is a small number of significant parameters to predict, and is better than the elastic net at variable selection as it finds more zero terms. The LARS algorithm is the most effective at feature selection, identifying every significant parameter as such, and with the lowest rate of falsely identifying significant parameters; the high L2 norm and RSS suggest that the parameters identified by this algorithm as non-zero deviate more from their true values than those given by the lasso and elastic net. The relaxed lasso, although meant in theory to improve upon the lasso, only outperforms ridge regression in this case.

5.2 Case 2: n = 1000, p = 50, real p = 15

5.2.1 Dataset

To evaluate the effect of number of observations on the performance of the different subset selection and shrinkage methods, we increase the number of observations by 10 times to 1000 while keeping the other parameters constant.

```r
set.seed(42)
X <- matrix(rnorm(1000*50), nrow = 1000, ncol = 50)
B <- sample(c(replicate(35, 0), runif(n = 15, min = 1, max = 3)
set.seed(10)
Y <- X %*% B + rnorm(1000)
```

5.2.2 Results

Discrete Methods	L2 Norm	MSE	AIC	TPR	FDR
Forward-stepwise	0.222	0.0465	2822.03	1	0.348
Backward-stepwise	0.222	0.0465	2822.03	1	0.348
Stepwise	0.222	0.0465	2822.03	1	0.348
Forward-stagewise	0.246	0.0559	NaN	1	0.7

Continuous Methods	L2 Norm	MSE	TPR	FDR
Ridge	0.503	1.189	1	0.7
Lasso	0.311	1.033	1	0.167
Relaxed Lasso	0.2740	1.007	1	0.7
Elastic-net($\alpha = 0.1$)	0.389	1.087	1	0.545
Elastic-net($\alpha = 0.3$)	0.355	1.062	1	0.348
Elastic-net($\alpha = 0.5$)	0.362	1.066	1	0.167
Elastic-net($\alpha = 0.7$)	0.354	1.059	1	0.167
Elastic-net($\alpha = 0.9$)	0.316	1.036	1	0.167
LARS(Step 16)	0.433	1.112	1	0

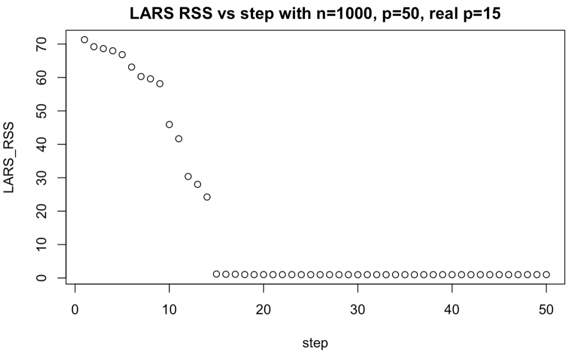


Figure 4: LARS RSS vs step for Case 2

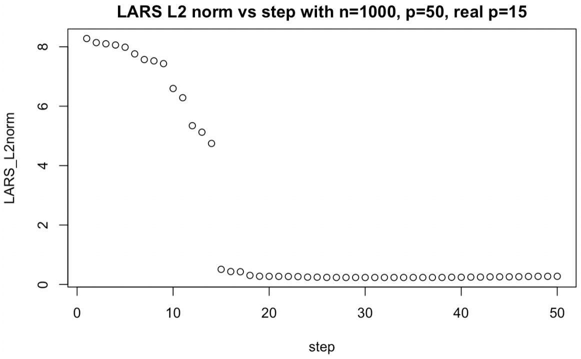


Figure 5: LARS L2norm vs step for Case 2

When we increase n from 100 to 1000, we could see that as suspected, there is a clear drop in MSE at step 15.

5.2.3 Discussion

Increasing the number of observations seems to decrease the L2 norm and MSE, in general. This makes sense, as the number of parameters stays the same, so essentially we have more data to predict the β vector. The low MSE shows that the model has a better data fit. However, we note that forward and backward-stepwise have a higher FDR, possibly because increasing the data points has led to small increase in the statistical significance of null variables. In fact, the data from Case 1 and Case 2 support the conclusion that the forward, backward and stepwise regression do not do very well when we have many redundant variables (35 in this case). However, the TPR being 1 for all discrete methods implies that if a variable is significant, it is identified by each method for all significant variables.

As in the previous case, the lasso outperforms the ridge method by all measures of prediction accuracy, and so it makes sense that the elastic net for values of $\alpha \geq 0.5$, which is more similar to the lasso, outperforms both the ridge and lasso. The LARS algorithm is found to be the best with regard to feature selection, as it has the lowest FDR of 0 of the continuous methods, meaning it does not result in any significant coefficients whose parameters are actually 0, but also a relatively large MSE and L2 norm, smaller only than those of the ridge regularisation, suggesting that the coefficients themselves are more far off. All methods here produce a TPR of 1, meaning that if a parameter is significant, it will be detected as such by all methods. The lasso is seen to be the best of the continuous methods, with respect to fitting the data and prediction accuracy, which is expected as the model is sparse. However, it is still outperformed by all 4 discrete methods.

5.3 Case 3: n = 100, p = 500, real p = 150

5.3.1 Dataset

To evaluate the effect of number of variables on the performance of the different subset selection and shrinkage methods, we increase the number of variables by 10 times to 500 as well as the number of true variables by 10 times to 150 while keeping the number of observations constant. This ensures that the ratio of true variables to total variables is constant.

```
set.seed(42)
X <- matrix(rnorm(100*500), nrow = 100, ncol = 500)
B <- sample(c(replicate(350, 0), runif(n = 150, min = 1, max = 3)
set.seed(10)
Y <- X %*% B + rnorm(100)
```

5.3.2 Results

Discrete Methods	L2 Norm	MSE	AIC	TPR	FDR
Forward-stepwise	38.715	5.033	$-\infty$	0.247	0.626
Backward-stepwise	NaN	NaN	NaN	NaN	NaN
Stepwise	38.715	5.033	$-\infty$	0.247	0.626
Forward stagewise	25.086	6.592	NaN	0.6	0.82

Continuous Methods	L2 Norm	MSE	TPR	FDR
Ridge	25.107	805.985	1	0.7
Lasso	25.100	770.913	0.007	0.667
Relaxed Lasso	25.107	758.876	0.007	0.5
Elastic-net($\alpha = 0.1$)	24.411	723.015	0.213	0.584
Elastic-net($\alpha = 0.3$)	24.845	728.256	0.093	0.391
Elastic-net($\alpha = 0.5$)	25.040	780.246	0.007	0.667
Elastic-net($\alpha = 0.7$)	25.107	805.985	0	NaN
Elastic-net($\alpha = 0.9$)	25.366	747.832	0.08	0.25
LARS(Step 23)	25.036	738.720	0.093	0.364

5.3.3 Discussion

The RSS increases significantly, by a factor in the order of 100, from case 1; this is unsurprising as the number of observations has remained constant, whilst the number of parameters has increased, and exceeded the number of observations - the same data is now needed to predict far more parameters. None of the continuous methods are particularly effective at fitting the data in this case, with high MSE and L2 norm, low TPR and high FDR, compared to the previous cases, as there are now more unknowns than observations.

We note that, although the ridge has the highest MSE, it is the only one of the continuous methods which correctly detects every significant parameter. Here, we see a limitation of the lasso regularisation: when p > N, the lasso can select at most N significant variables: the lasso estimates are computed using a slightly modified form of the LAR algorithm, which can have at most N parameters in its active set at a time. This results in many false negatives - true non-zero parameters that are not detected as significant, as shown by the low TPR.

The most effective continuous method, with respect to almost every measure (L2 norm, RSS and

TPR) is the elastic net with $\alpha = 0.1$, much closer to ridge regression than lasso, which makes sense as this model is a dense one, although the ridge does not actually outperform the lasso. The relaxed lasso fits the data slightly better than the lasso, with a lower RSS, but is (though only very slightly) less effective at generalising (L2 norm). Like the lasso, the probability of correctly identifying a significant parameter is small (0.007), but the rate of type I errors is slightly smaller, which is reasonable, as the noise parameters are first eliminated in the relaxed lasso before generating the coefficients of the significant parameters, and so we might expect fewer false positives.

The discrete methods are not very effective at testing accuracy for new data (large L2 norm), but produce coefficients that fit the data well (small RSS), suggesting that these methods generate models that have overfitted the data. They are generally a better fit to this data than the continuous methods, as the RSS values are significantly smaller, but are likely to be less effective in predicting for new data, as the L2 norms are notably greater. The stepwise algorithm seems to have overfitted the data (small MSE) and does not generalise well (large L2 norm). Forward stepwise fits the data slightly better than, but does not generalise as well as forward stagewise. All discrete methods perform poorly with respect to variable selection, with low TPR and high FDR, compared to the previous cases, which agrees with the high L2 norms and AIC statistics.

5.4 Case 4: n = 100, p = 50, real p = 30

5.4.1 Dataset

To evaluate the effect of the ratio of true variables to total variables on the performance of the different subset selection and shrinkage methods, we double the number of true variables to 30.

```
set.seed(42)
X <- matrix(rnorm(100*50), nrow = 100, ncol = 50)
B <- sample(c(replicate(20, 0), runif(n = 30, min = 1, max = 3)
set.seed(10)
Y <- X %*% B + rnorm(100)
```

5.4.2 Results

Discrete Methods	L2 Norm	MSE	AIC	TPR	FDR
Forward-stepwise	0.790	0.361	293.576	1	0.167
Backward-stepwise	0.988	0.395	292.929	1	0.231
Stepwise	0.790	0.361	293.576	1	0.167
Forward stagewise	1.003	0.417	NaN	1	0.4

Continuous Methods	L2 Norm	MSE	TPR	FDR
Ridge	3.310	12.246	1	0.4
Lasso	1.306	2.206	1	0.333
Relaxed Lasso	0.950	2.452	1	0.268
Elastic-net($\alpha = 0.1$)	1.678	2.750	1	0.375
Elastic-net($\alpha = 0.3$)	1.503	2.780	1	0.348
Elastic-net($\alpha = 0.5$)	1.407	2.634	1	0.348
Elastic-net($\alpha = 0.7$)	1.345	2.258	1	0.333
Elastic-net($\alpha = 0.9$)	1.312	2.268	1	0.333
LARS(Step 46)	1.368	2.438	1	0.333

5.4.3 Discussion

Comparing this case to Case 1 (for discrete methods), we see that, as is reasonably expected, increasing the proportion of non-zero parameters has increased the L2 norm by a small amount. Again, as in

Case 1 and 2, we see that forward-stagewise is outperformed by all three of the other discrete methods. We can speculate that forward-stagewise performs better when there is a sharp increase in the number of parameters and non-zero parameters, possible when $p > n$, as seen in the L2 norm in Case 3. Additionally, we note a small decrease in FDR across all discrete methods, which makes sense as a higher proportion of parameters are non-zero, so the model is less likely to discover a false positive.

The lasso regularisation continues to outperform the ridge method, and, this time, all variants of the elastic net too, with respect to fitting the data (RSS), prediction accuracy (L2 norm), and feature selection (TPR and FDR). The lasso is the best at fitting the data, but is slightly outperformed by the relaxed lasso, which has a better testing accuracy, and identifies fewer coefficients as significant which are truly zero (lower FDR), which makes sense, since the second stage of the relaxed lasso is carried out after the noise parameters are eliminated.

5.5 Case 5: n = 100, p = 200, real p = 60

5.5.1 Dataset

Here, we explore the effect of $p > n$ but the number of true variables stays below n. For the same reason as above, we will not be able to perform backward-stepwise regression.

```
set.seed(42)
X <- matrix(rnorm(100*200), nrow = 100, ncol = 200)
B <- sample(c(replicate(140, 0), runif(n = 60, min = 1, max = 3)
set.seed(10)
Y <- X %*% B + rnorm(100)
```

5.5.2 Results

Discrete Methods	L2 Norm	MSE	AIC	TPR	FDR
Forward-stepwise	15.824	6.466	$-\infty$	0.667	0.596
Backward-stepwise	NaN	NaN	NaN	NaN	NaN
Stepwise	15.824	6.466	$-\infty$	0.667	0.596
Forward stagewise	11.985	0.971	NaN	0.983	0.705

Continuous Methods	L2 Norm	MSE	TPR	FDR
Ridge	16.745	199.14	1	0.4
Lasso	16.745	199.14	1	0.333
Relaxed Lasso	16.688	199.14	1	0.333
Elastic-net($\alpha = 0.1$)	16.294	175.25	1	0.375
Elastic-net($\alpha = 0.3$)	16.745	199.14	1	0.348
Elastic-net($\alpha = 0.5$)	16.745	199.14	1	0.348
Elastic-net($\alpha = 0.7$)	16.586	194.23	1	0.333
Elastic-net($\alpha = 0.9$)	16.745	199.14	1	0.333
LARS(Step 11)	15.62	187.93	1	0.333

5.5.3 Discussion

We see a significant increase in L2 norms across all methods. In the discrete cases, the TPR, FDR and MSE also get worse. The FDR being high results in including redundant variables, additionally the TPR being lower than 1 causes important variables to be ignored, which we would expect to increase the RSS. This is supported by the high MSE values for stepwise and forward-stepwise regression. Furthermore, forward-stagewise does relatively better than then the other methods. This supports the conclusion from Case 3 for forward-stagewise performing well when $p > n$. However, it still has the highest FDR yet the MSE is less than 1, which can possibly be explained by the magnitude of each of the the falsely discovered parameters being very small. The TPR being close to 1 further supports the low MSE.

The continuous methods are all less effective, with respect to fitting the data and testing accuracy, than the discrete methods, but perform considerably better at variable selection, identifying every truly significant parameter as such (TPR = 1), and with a lower proportion of false positives (lower FDR). This, combined with the high RSS and L2 norms suggest that although the continuous methods are better at selecting the correct parameters, the predicted coefficients themselves deviate much more from their true values. The relaxed lasso improves upon the testing accuracy (L2 norm) of the lasso, though only slightly. The LAR algorithm is the most effective continuous method in this case, in terms of variable selection and prediction accuracy, but is outperformed by the elastic net with $\alpha = 0.1$ in terms of fitting the data.

5.6 Conclusions

Overall, we see that it is not possible to conclude that one method is unanimously the most effective; we must consider it on a case-by-case basis, and decide which aspects of model prediction (which measures of prediction accuracy) are most important.
In the cases where p > N, the design matrix cannot have full rank, and so, by the rank-nullity theorem, there is at least one non-zero c, such that $\mathbf{X}c = 0$. It follows that there are some β_1 and β_2 such that

$$\beta_2 = \beta_1 + c$$

, and so

$$\mathbf{X}\beta_1 = \mathbf{X}\beta_2$$

, and so the model is not identifiable, as these two different values of β result the same data distribution.

This lack of identifiability is reflected in the poor performance of the continuous shrinkage methods in Case 3 (n=100, p=500, real p = 150) - particularly in the low TPR and high FDR values, which supports the idea that (at least) two distinct β vectors have resulted in the same distribution, and so the algorithms for each method cannot tell them apart, resulting in poor variable selection. This is also seen in the large RSS and L2 norms, which have each increased by a couple of orders of magnitude from case 1. This, in conjunction with the TPR and FDR values, suggests that the shrinkage methods have not only missed some significant parameters (and falsely picked up some others) due to lack of identifiability, but also that these indistinguishable vectors differ significantly in magnitude.

We see similar results for the FDR, RSS and L2 norm in case 5, where p exceeds n, but the number of non-zero parameters is less than n, suggesting again that there is a problem of identifiability. In case 5, and indeed all cases where the number of non-zero parameters is less than n, the TPR values of the continuous methods are all 1, implying that every significant parameter is detected as such. However, in case 5, (since FDR is not close to 0), some null parameters are also detected as significant, contributing to the large RSS and L2 norm.

In cases 3 and 5, where p exceeds n, although none of the continuous methods are particularly successful, the elastic net (with $\alpha = 0.1$, close to ridge) and LAR were the best methods, with respect to fitting the training data, and testing accuracy - moreover, the relaxed lasso, as intended, is more effective than the lasso in this case, but, as expected, neither one is a competitor for the most effective, as the lasso tends to perform well when the model is sparse. Although the ridge does not outperform the lasso in case 3 in testing accuracy, as might be expected, since we have a dense model, it comes far closer to doing this than any of the cases where n > p.

Looking at variable selection for continuous methods on the whole, excepting case 3, the continuous methods do not miss any significant parameters, with TPR values of 1 throughout. In cases where $n > p$, ridge regression inevitably falsely selects several parameters as significant, more than the other continuous methods, as it does not shrink any parameters to 0, and the true model is sparse in these cases, and so results in an unnecessarily over-complicated model. With the exception of case 5, the LAR algorithm is the best for feature selection, as it has the lowest rate of falsely-detected significant coefficients (FDR) in each case. The relaxed lasso produces a smaller FDR in cases 3, 4 and 5, which makes sense as the noise parameters are eliminated before the second-stage of the relaxed lasso, which should lead to fewer insignificant parameters identified as non-zero in the second step.

6 Ozone Interaction Data

6.1 Data Source

The Ozone Interaction Dataset is obtained from the **spikeslab** package in **R**, consisting of 203 observations and 134 variables. The outcome is the reading of maximum daily ozone collected from the Los Angeles basin. The readings are integers and lie within the range of 1 to 38. The variables are named x1, x2, ..., x134. **(Ishwaran , Rao and Kogalur, 2022)**

6.2 Scaling the Data

We use the **scale** function in **R** to normalise the dataset. The function standardises each input using the following formula

$$x_{scaled} = \frac{x - \bar{x}}{s}$$

where $\bar{x}$ is the sample mean and s is the sample standard deviation.

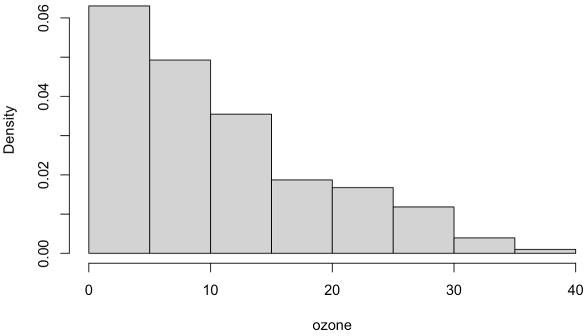


Figure 6: Histogram of ozone variable before scaling

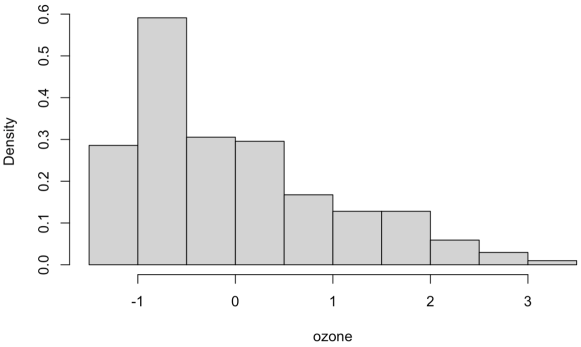
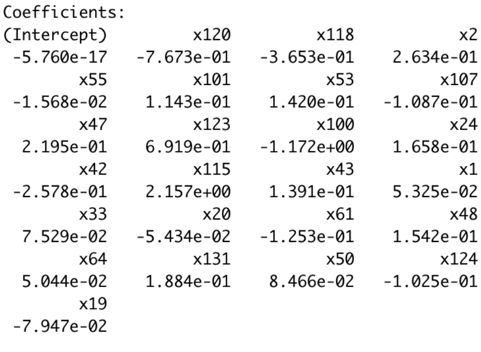


Figure 7: Histogram of ozone variable after scaling

6.3 Methods

6.3.1 Forward-stepwise Selection

The forward-stepwise selection algorithm reduces the number of variables to 24. From Figure 9, we can observe a big drop in the AIC value when the variable x120 is added to the null model, thus we can infer that x120 is one of the most impactful predictors on the response. The procedure stops when it reaches an AIC score of -367 at step 25.



```
Coefficients:
(Intercept)         x120         x118           x2
 -5.760e-17   -7.673e-01   -3.653e-01    2.634e-01
        x55         x101          x53         x107
 -1.568e-02    1.143e-01    1.420e-01   -1.087e-01
        x47         x123         x100          x24
  2.195e-01    6.919e-01   -1.172e+00    1.658e-01
        x42         x115          x43           x1
 -2.578e-01    2.157e+00    1.391e-01    5.325e-02
        x33          x20          x61          x48
  7.529e-02   -5.434e-02   -1.253e-01    1.542e-01
        x64         x131          x50         x124
  5.044e-02    1.884e-01    8.466e-02   -1.025e-01
        x19
 -7.947e-02
```

Figure 8: Coefficients of the linear model after implementing forward-stepwise selection

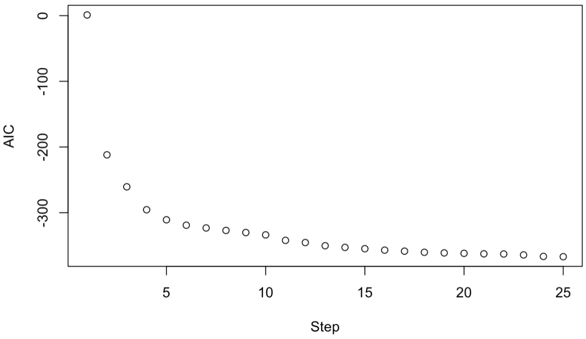
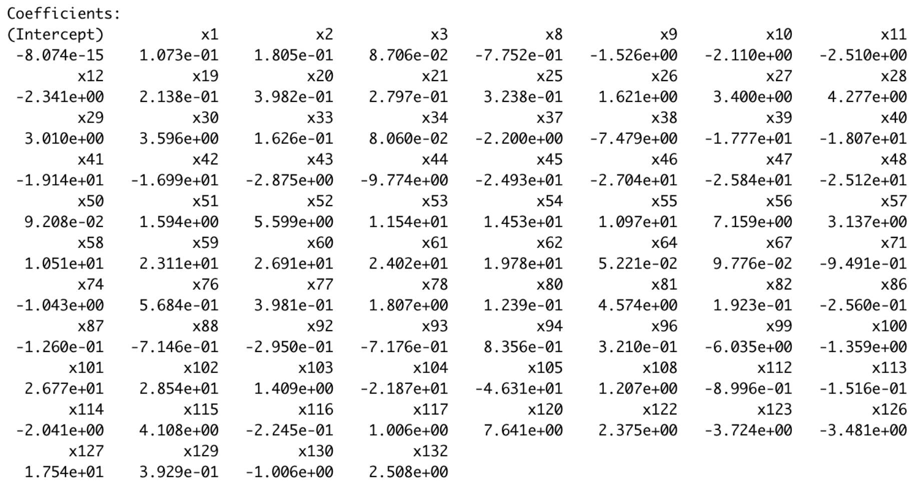


Figure 9: Plot of step vs AIC scores for forward-stepwise selection

6.3.2 Backward-stepwise Selection

The backward-stepwise selection algorithm reduces the number of variables to 83. From Figure 11, we notice that the backward-stepwise method reduces the AIC score by a small amount for each step. We notice that the variable x120 has not been removed from model throughout the process, which suggests that removing it will not cause a significant decrease in the AIC score. The procedure stops when it reaches an AIC score of -365 at step 52.



```
Coefficients:
(Intercept)          x1          x2          x3          x8          x9         x10         x11 
 -8.074e-15   1.073e-01   1.805e-01   8.706e-02  -7.752e-01  -1.526e+00  -2.110e+00  -2.510e+00 
        x12         x19         x20         x21         x25         x26         x27         x28 
 -2.341e+00   2.138e-01   3.982e-01   2.797e-01   3.238e-01   1.621e+00   3.400e+00   4.277e+00 
        x29         x30         x33         x34         x37         x38         x39         x40 
  3.010e+00   3.596e+00   1.626e-01   8.060e-02  -2.200e+00  -7.479e+00  -1.777e+01  -1.807e+01 
        x41         x42         x43         x44         x45         x46         x47         x48 
 -1.914e+01  -1.699e+01  -2.875e+00  -9.774e+00  -2.493e+01  -2.704e+01  -2.584e+01  -2.512e+01 
        x50         x51         x52         x53         x54         x55         x56         x57 
  9.208e-02   1.594e+00   5.599e+00   1.154e+01   1.453e+01   1.097e+01   7.159e+00   3.137e+00 
        x58         x59         x60         x61         x62         x64         x67         x71 
  1.051e+01   2.311e+01   2.691e+01   2.402e+01   1.978e+01   5.221e-02   9.776e-02  -9.491e-01 
        x74         x76         x77         x78         x80         x81         x82         x86 
 -1.043e+00   5.684e-01   3.981e-01   1.807e+00   1.239e-01   4.574e+00   1.923e-01  -2.560e-01 
        x87         x88         x92         x93         x94         x96         x99        x100 
 -1.260e-01  -7.146e-01  -2.950e-01  -7.176e-01   8.356e-01   3.210e-01  -6.035e+00  -1.359e+00 
       x101        x102        x103        x104        x105        x108        x112        x113 
  2.677e+01   2.854e+01   1.409e+00  -2.187e+01  -4.631e+01   1.207e+00  -8.996e-01  -1.516e-01 
       x114        x115        x116        x117        x120        x122        x123        x126 
 -2.041e+00   4.108e+00  -2.245e-01   1.006e+00   7.641e+00   2.375e+00  -3.724e+00  -3.481e+00 
       x127        x129        x130        x132 
  1.754e+01   3.929e-01  -1.006e+00   2.508e+00 
```

Figure 10: Coefficients of the linear model after implementing backward-stepwise selection

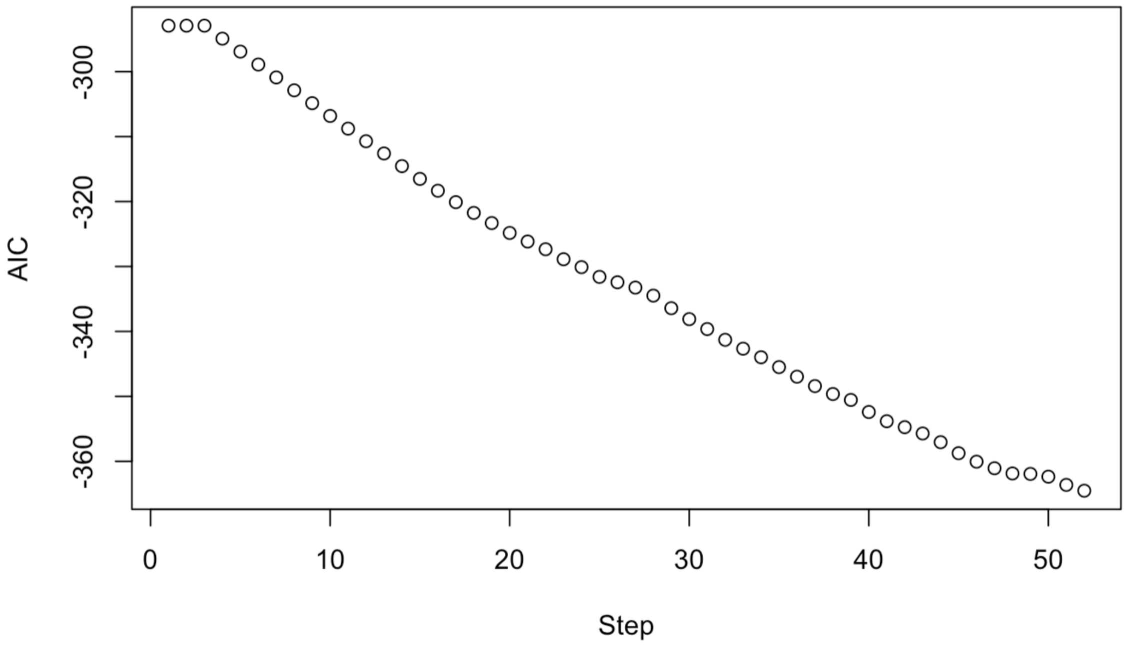
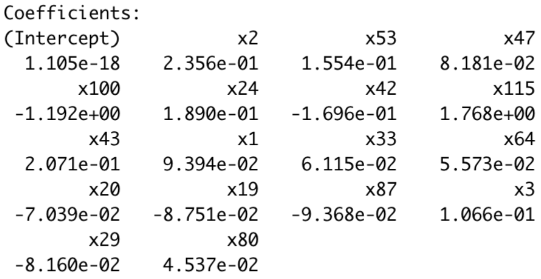


Figure 11: Plot of step vs AIC scores for backward-stepwise selection

6.3.3 Stepwise Regression

The stepwise regression algorithm reduces the number of variables to 17. Here, we find out that the variable x120 has not been included in the final model. Apparently, x120 has been added to the null model at step 2 but has been removed from the model at step 22, which implies that removing x120 from the model has a significant effect on the AIC score after adding and removing some of the predictors. The procedure stops when it reaches an AIC score of -375 at step 34.



```
Coefficients:
(Intercept)          x2          x53          x47
  1.105e-18   2.356e-01    1.554e-01    8.181e-02
       x100         x24          x42         x115
 -1.192e+00   1.890e-01   -1.696e-01    1.768e+00
        x43          x1          x33          x64
  2.071e-01   9.394e-02    6.115e-02    5.573e-02
        x20         x19          x87           x3
 -7.039e-02  -8.751e-02   -9.368e-02    1.066e-01
        x29         x80
 -8.160e-02   4.537e-02
```

Figure 12: Coefficients of the linear model after implementing stepwise regression

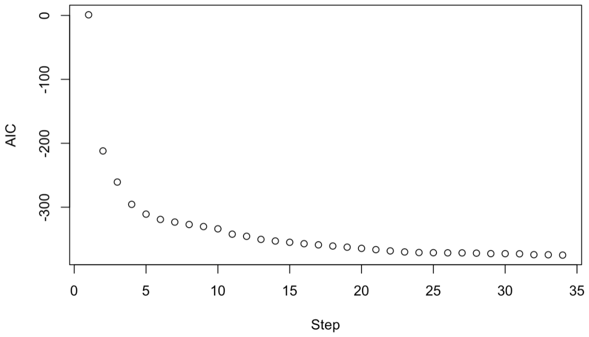


Figure 13: Plot of step vs AIC scores for stepwise regression

6.3.4 Forward Stagewise

The forward stagewise procedure takes a total of 825 steps. From Figure 14, we inspect that the variables x101 and x105 are highly correlated with the response variable since their coefficients have greatly increased and decreased respectively.

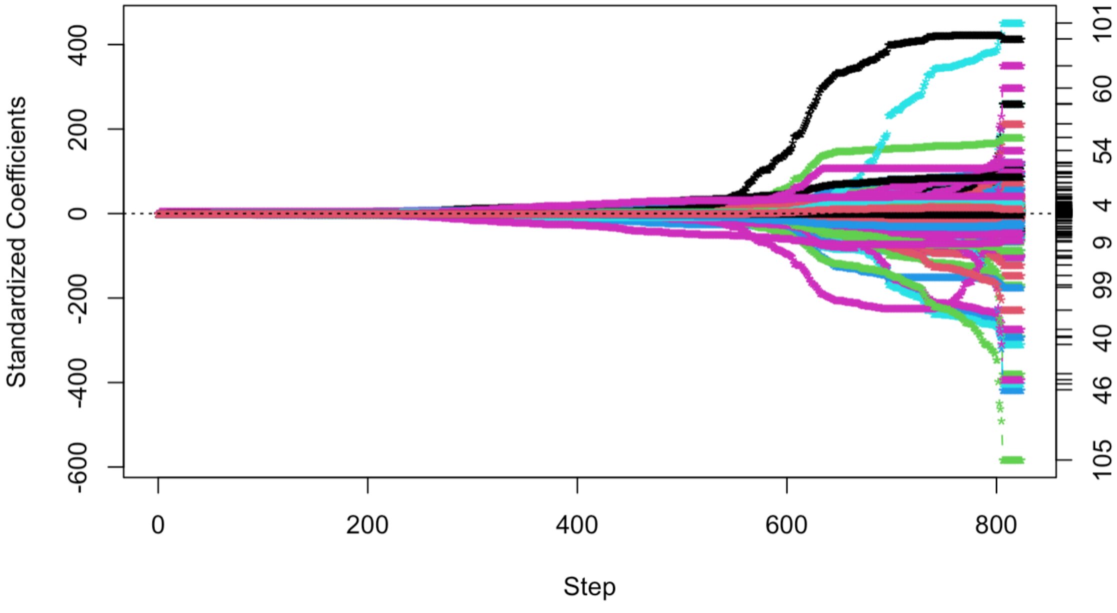


Figure 14: Coefficient profile of forward stagewise

6.3.5 Ridge Regularisation

According to Figure 15, we run cross-validation to get the best lambda value which minimises the mean square error as 0.685. From Figure 16, we can infer that variables x2, x47 and x115 have great effect on the response variable.

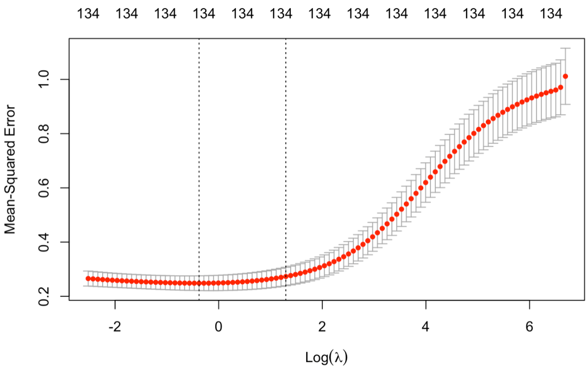


Figure 15: Plot of ridge regularisation cross-validation

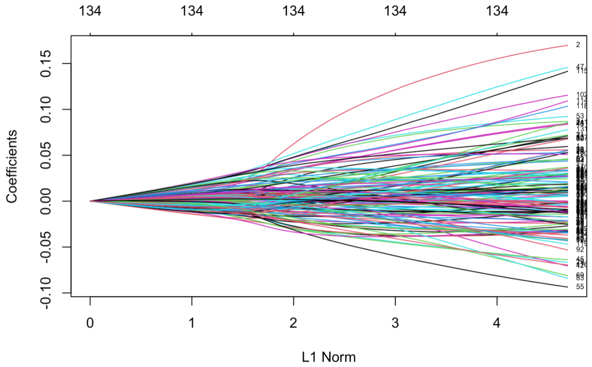


Figure 16: Coefficient profile of ridge regularisation

6.3.6 Lasso Regularisation

According to Figure 18, we run cross-validation to get the best lambda value which minimises the mean square error as 0.034. Lasso regularisation reduces the number of variables to 16. From Figure 19, we are able to detect that most of the coefficients are shrunk to zero eventually, whereas predictors such as x115 will remain in the model.

```
          x2           x24           x29           x40           x47           x50           x53           x63           x64 
 0.169984662  0.040913474 -0.020596813  0.008409874  0.141527702  0.045353476  0.108982392  0.001698499  0.009339129 
         x66           x79          x102          x115          x116          x120          x127 
-0.039298385 -0.005543172  0.152135726  0.223485644 -0.005506363  0.218322941  0.101079471 
```

Figure 17: Coefficients of the linear model after implementing lasso regularisation

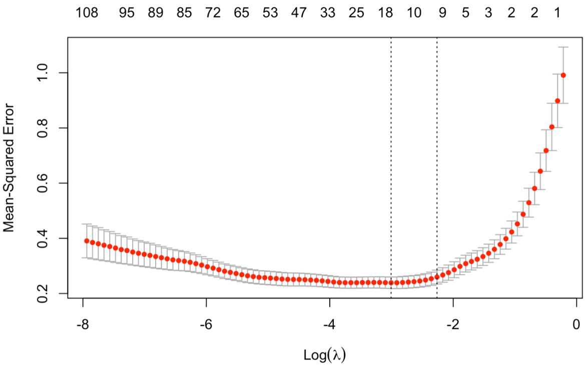


Figure 18: Plot of lasso regularisation cross-validation

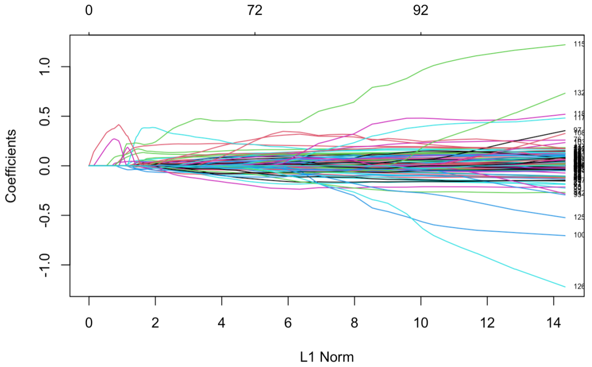


Figure 19: Coefficient profile of lasso regularisation

6.3.7 Elastic Net Regularisation

Firstly, we run cross-validation to find the best alpha value that minimises the mean square error as 0.82. According to Figure 21, we then run another cross-validation to find the best lambda value which minimises the mean square error as 0.0603. Elastic net regularisation reduces the number of variables to 30. Here, we can also discover that the coefficient profile of elastic net regularisation is similar to the coefficient profile of lasso regularisation, but stopping at a larger l_1-norm of the whole coefficient vector.

```
           x1            x2            x3           x14           x16           x20           x24           x29 
 0.0288253056  0.1929700128  0.0517885016  0.0008682475 -0.0004146798 -0.0136562754  0.0583328409 -0.0510785884 
          x31           x40           x44           x45           x46           x47           x48           x50 
-0.0116646300  0.0558787488 -0.0080490454 -0.0048025919  0.0072513018  0.1685601446  0.0522051584  0.0379771397 
          x52           x53           x55           x56           x63           x64           x66           x70 
 0.0097195562  0.1313111427 -0.0017035751 -0.0162441833  0.0205834070  0.0225238102 -0.0381526486 -0.0079007569 
          x85          x102          x115          x116          x120          x134 
 0.0044717520  0.3281739827  0.2145130869 -0.0347150562  0.0345956797 -0.0043850815 
```

Figure 20: Coefficients of the linear model after implementing elastic net regularisation

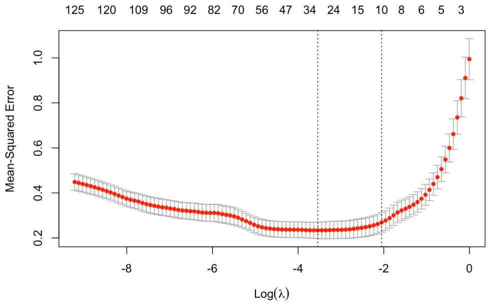


Figure 21: Plot of elastic net regularisation cross-validation

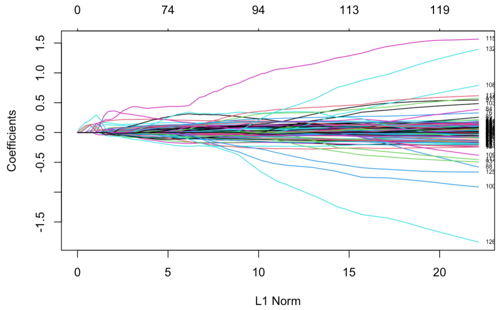


Figure 22: Coefficient profile of elastic net regularisation

6.3.8 Least Angle Regression

It can be observed that the coefficient profile of least angle regression is similar to the coefficient profile of forward stagewise. The main difference is that least angle regression terminates at step 135, which is much faster than forward stagewise.

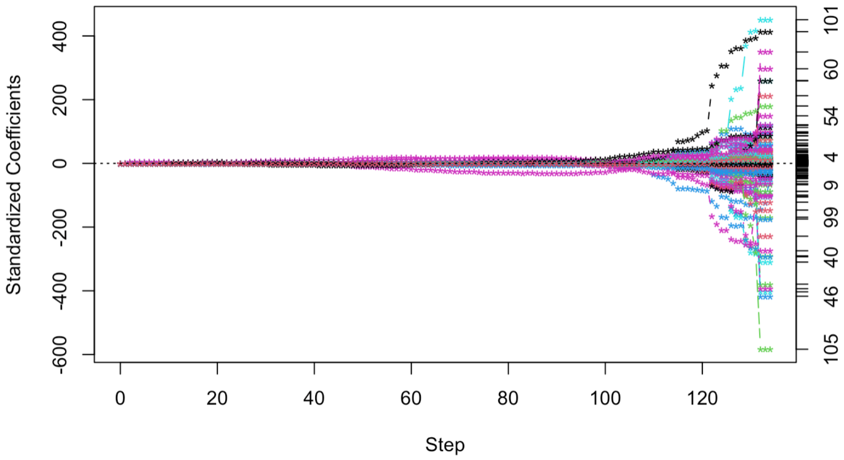


Figure 23: Coefficient profile of least angle regression

6.4 Results

As the sample size of the ozone interaction data is relatively small, we calculate the mean squared error using 10-fold cross-validation. The 10-fold cross-validation procedure is as follows:

1. The data is split into 10 cross-validation folds $T_1, T_2, ..., T_{10}$.
2. For each fold k = 1, 2, ..., 10
   (a) We leave out the fold T_k and fit the model to the other 9 parts of the data.
   (b) We calculate the prediction error of the fitted model when predicting the fold T_k.
3. We calculate the mean and standard deviation of the prediction errors from all 10 folds.

(Hastie et al., 2009, p. 241-249)
This process is implemented using the **train** function from the **caret** package in **R**. First, we set seed for reproducibility and define the training control using the **trainControl** function from the **caret** package in **R**. The method is defined as ”cv” (cross-validation) and the number of folds is fixed as 10. In the **train** function, we specify the method as ”lmStepAIC” for forward-stepwise, backward-stepwise and stepwise methods, ”lars” for forward stagewise and least angle regression as well as ”glmnet” for ridge, lasso and elastic net regularisation. **(Kuhn, 2021)**

```
set.seed(123)
train.control <- trainControl(method = "cv", number = 10)
model <- train(ozone˜., data = df, method = "lm", trControl = train.control)
```

	RMSE	R^2	MAE
Full model	0.9371 (0.3267)	0.4997 (0.2185)	0.7422 (0.1968)

Discrete Methods	RMSE	R^2	MAE
Forward-stepwise	0.8027 (0.1666)	0.5687 (0.0699)	0.6330 (0.1163)
Backward-stepwise	0.7917 (0.1883)	0.5781 (0.1590)	0.6537 (0.1387)
Stepwise	0.8549 (0.2546)	0.5317 (0.1118)	0.6470 (0.1143)
Forward stagewise	0.7047 (0.0790)	0.6268 (0.0555)	0.5708 (0.0534)

Continuous Methods	RMSE	R^2	MAE
Ridge ($\lambda = 0.64$)	0.4893 (0.0974)	0.7761 (0.0759)	0.3790 (0.0743)
Lasso ($\lambda = 0.033$)	0.4620 (0.1225)	0.8100 (0.0927)	0.3587 (0.0890)
Elastic net ($\alpha = 0.99, \lambda = 0.041$)	0.4737 (0.0792)	0.7922 (0.0783)	0.3556 (0.0452)
Least Angle	0.6214 (0.1417)	0.6910 (0.0910)	0.4968 (0.1032)

Note: standard errors are given in parentheses.

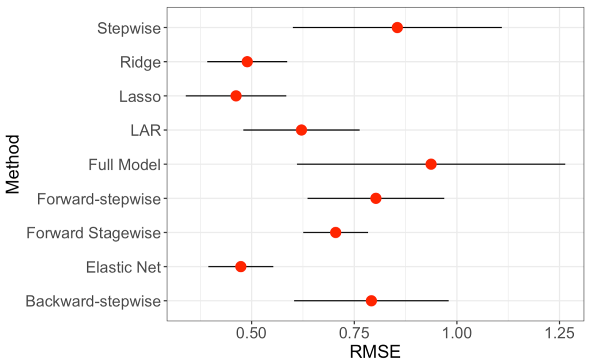


Figure 24: Summary of RMSE for each method

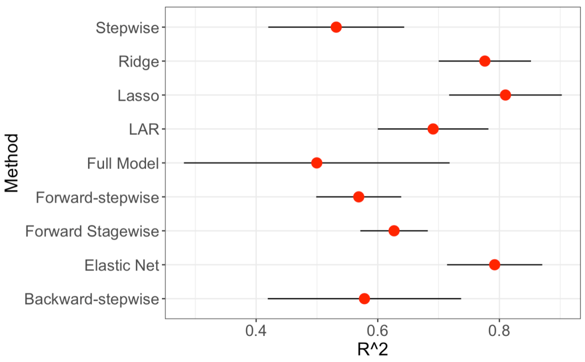


Figure 25: Summary of R^2 values for each method

6.5 Discussion

Overall, we can conclude that all the subset selection and shrinkage methods improve the root mean square error(RMSE), R-squared(R^2) value and the mean absolute error(MAE).

Among the discrete methods, forward-stagewise which has the lowest RMSE and highest R^2 value performs the best, followed by backward-stepwise, forward-stepwise and stepwise regression. Forward-stagewise doing well in this case where $p < n$ is not in line with the results obtained from the simulations. Compared to forward-stepwise selection which takes 25 steps, backward-stepwise selection which takes 52 steps, and stepwise regression which takes 34 steps, forward stagewise which takes 825 steps might be seen as inefficient. Nevertheless, forward stagewise is computationally cheap as it involves simple repetitions, more stable in terms of component paths and has a quite favourable statistical performance. **(Tibshirani, 2015)**

Within the continuous methods, lasso regularisation which has the lowest RMSE and highest R^2 value performs the best, followed by elastic net regularisation, ridge regularisation and least angle regression. This result aligns with the findings obtained from the numerical simulation case where $p < n$. Furthermore, unlike ridge regularisation and least angle regression where we have to estimate 134 parameters, lasso regularisation reduces the number of variables to 16, which makes it more computationally efficient than all the other methods. **(Hastie, Tibshirani and Wainwright, 2015)**

It can be observed that the continuous methods outperform the discrete methods. Lasso regularisation possesses some of the useful advantages of both subset selection and ridge regularisation, i.e. it generates interpretable models like subset selection and demonstrates similar stability to ridge regularisation. **(Tibshirani, 1996)** This might be the reason why lasso regularisation exhibits the best results among all the methods. Hence, we would suggest applying lasso regularisation to the ozone interaction dataset.

Elastic net regularisation and ridge regularisation also display fairly well performances (only slightly higher RMSE and slightly lower R^2 value than lasso regularisation). Elastic net regularisation is a combination of both ridge regularisation and lasso regularisation, thus demonstrating both good prediction performance like ridge regularisation and variable selection properties like lasso regularisation. **(Zou and Hastie, 2005)** Ridge regularisation is less preferred here as it keeps all the predictor variables in the model.

On the other hand, there are many fatal problems with stepwise methods such as the decisions of adding or removing a specific variable are based on the current optimal step which is not consequently a global optimisation, highly biased regression coefficients are obtained throughout the process as well as the number of true predictor variables and noise variables included in the final model are impacted by the degree of correlation between the predictor variables and number of predictor variables respectively. **(Harrell, 2015, p. 67-70)** Thus, the poor performance of the stepwise methods in this simulation supports the notion that the use of these methods should be discouraged. **(Smith, 2018)**

From Figures 8, 10, 12, 17 and 20, the predictors which are always present even after undergoing variable selection procedures are x2, x47, x53, x64, x115. This finding suggests that these variables have significant effect on the response variable.

The lasso outperforms both the ridge and the elastic net, suggesting that the model is sparse; this seems to support the conjecture that there are only 5 significant parameters (2, 47, 53, 64, 115). However, since the ridge method also performs well (only slightly higher RMSE and coefficient of determination than lasso and elastic net), we might surmise that the true values of these parameters are small, as ridge regression generally shrinks coefficients towards 0. Least Angle Regression is the least effective, with the highest RMSE and MAE and smallest coefficient of determination - in contrast, in all cases where n > p in the numerical simulations, the LARS algorithm outperforms the ridge, with respect to L2 norm and RSS. Looking at the coefficient profiles for, eg. ridge and

LARS, we see that, both of these methods do not eliminate any variables, but the ridge coefficients are, typically, a couple of orders of magnitude smaller than the LARS coefficients, which explains why the ridge regression has a notably smaller RMSE, comparable to the lasso and elastic net. Moreover, the ridge and lasso, though performing well individually, produce coefficients that differ (like the ridge and LARS) by 2 orders of magnitude on average. If there is a degree of correlation between some of the parameters, the lasso might choose just one of these, which is strongly correlated to the others, and assign all of the weight of this group to this one variable, whereas the ridge is more likely to select all parameters of that group, and shrink their coefficients towards zero, and each other. One possible explanation, therefore, of the fact that ridge and lasso perform well separately but produce very different coefficients is some correlation in the data. However, since in this case n > p, high correlations between predictors would have resulted in the ridge dominating the lasso, which is not the case. **(Zou, Hastie, 2005)**.

7 Conclusion

In conclusion, the various subset selection and shrinkage methods have different performances for distinct cases. For example, lasso regularisation outperforms all the other methods for the ozone interaction data. However, in Case 3 where $p >> n$, elastic net regularisation outperforms all the other methods. Likewise, the discrete methods perform outstandingly in most of the cases in the numerical simulations section. Nonetheless, they demonstrate rather unsatisfactory performances for the ozone interaction data. Thus, neither method is constantly better than the other, so we should test each method to select the best method for a particular dataset.

A Why does a ridge estimator always have a smaller MSE than an OLS estimator?

Since ridge estimators are linear in y, we can work out its covariance matrix readily easily.

$$\begin{aligned}Cov(\hat{\beta}^{ridge}|X) &= Cov((X^TX+\lambda I)^{-1}X^Ty)\\ &= ((X^TX+\lambda I)^{-1}X^T)Cov(y)((X^TX+\lambda I)^{-1}X^T)^T\\ &= \sigma^2(X^TX+\lambda I)^{-1}X^TX(X^TX+\lambda I)^{-1}\end{aligned}$$

Define $W = X^TX(X^TX+\lambda I)^{-1}$, then we can write

$$\begin{aligned}Cov(\hat{\beta}^{ridge}|X) &= \sigma^2(X^TX+\lambda I)^{-1}(X^TX)(X^TX)^{-1}(X^TX)(X^TX+\lambda I)^{-1}\\ &= \sigma^2W^T(X^TX)^{-1}W\end{aligned}$$

Since the OLS has covariance matrix $\sigma^2(X^TX)^{-1}$, the difference between the covariance matrices of these two estimators would be

$$\begin{aligned}Cov(\hat{\beta}) - Cov(\hat{\beta}^{ridge}) &= \sigma^2(X^TX)^{-1} - \sigma^2W^T(X^TX)^{-1}W\\ &= \sigma^2W^T(W^T)^{-1}(X^TX)^{-1}W^{-1}W - W^T(X^TX)^{-1}W\\ &= \sigma^2W^T(W^T)^{-1}(X^TX)^{-1}W^{-1} - (X^TX)^{-1}W\\ &= \sigma^2W^T(X^TX)^{-1}(X^TX+\lambda I)(X^TX)^{-1}(X^TX+\lambda I)(X^TX)^{-1} - (X^TX)^{-1}W\\ &= \sigma^2W^T(I+\lambda(X^TX)^{-1})(X^TX)^{-1}(I+\lambda(X^TX)^{-1}) - (X^TX)^{-1}W\\ &= \sigma^2W^T((X^TX)^{-1}+\lambda(X^TX)^{-2})(I+\lambda(X^TX)^{-1}) - (X^TX)^{-1}W\\ &= \sigma^2W^T(X^TX)^{-1} + 2\lambda(X^TX)^{-2} + \lambda^2(X^TX)^{-3} - (X^TX)^{-1}W\\ &= \sigma^2W^T2\lambda(X^TX)^{-2} + \lambda^2(X^TX)^{-3}W\\ &= \sigma^2(X^TX+\lambda I)^{-1}X^TX2\lambda(X^TX)^{-2} + \lambda^2(X^TX)^{-3}X^TX(X^TX+\lambda I)^{-1}\\ &= \sigma^2(X^TX+\lambda I)^{-1}(2\lambda I + \lambda^2(X^TX)^{-1})(X^TX+\lambda I)^{-1}\end{aligned}$$

B Why does ridge regularisation not shrink parameters to zero?

It is easy to see that the eigenvalues of $X^TX+\lambda I$ and those of X^TX are related.
Suppose t is an eigenvalue of X^TX, then

$$\det(tI - X^TX) = 0$$

Rearranging gives:

$$\det((t+\lambda)I - (X^TX+\lambda I)) = 0$$

$$\implies t+\lambda \text{ is an eigenvalue of } X^TX+\lambda I.$$

Claim: X^TX is positive definite if X has full rank.
Proof. X has full rank, by the rank-nullity theorem, dim(ker(X))=0. Thus for $v \neq 0$, $Xv \neq 0$, which then implies that $v^tX^TXv = (Xv)^tXv = \langle Xv\ ,\ Xv\rangle \neq 0$, i.e. X^TX is positive definite.

As λ is positive and X^TX has positive eigenvalues, $X^TX+\lambda I$ has positive eigenvalues. Consider the Singular Value Decomposition(SVD) of X:

$$X = UDV^T$$

where U is an $N \times p$ orthogonal matrix, D is a $p \times p$ diagonal matrix with diagonal entries being the eigenvalues of X, and V is a $p \times p$ orthogonal matrix.

After some matrix algebra, we have:

$$\begin{aligned}
\hat{\beta}^{ridge} &= (X^T X + \lambda I)^{-1} X y \\
&= \left(V(D^2 + \lambda I)V^T\right)^{-1} V D U^T y \\
&= V(D^2 + \lambda I)^{-1} D U^T y \\
&= \sum_{i=1}^{p} u_j \frac{d_j^2}{d_j^2 + \lambda} u_j^t y
\end{aligned}$$

where u_j is the j-th column of U and d_j is the j-th diagonal entry of D.
As $d_j \neq 0 \quad \forall j = 1, 2, ..., p$, $\hat{\beta}^{ridge}$ does not have zero entries.

As $\lambda \longrightarrow \infty$, $\hat{\beta}^{ridge} \longrightarrow 0$ but it never reaches zero for positive real λ.

C R Code for Continuous Shrinkage Method for Numerical Simulation Case 1

We first set the standardised design matrix X and our true β value, and add some noise to produce Y = Xβ + Z.

```
set.seed(42)
library(glmnet)

## Loading required package:  Matrix
## Loaded glmnet 4.1-2

n <- 100
p <- 50
real_p <- 15
X <- matrix(rnorm(n*p), nrow=n, ncol=p)
B_1 <- replicate(35, 0)
B_2 <- runif(n=15, min=1, max=3)
B <- sample(c(B_1, B_2))
```

Split into training set and test set.

```
set.seed(10)
Y = X %*% B + rnorm(n)
train_rows <- sample(1:n, .66*n)
x.train <- X[train_rows, ]
x.test <- X[-train_rows, ]
y.train <- Y[train_rows]
y.test <- Y[-train_rows]
```

Ridge Regression

```
set.seed(10)
ptm <- proc.time()
alpha0.fit <- cv.glmnet(x.train, y.train, type.measure="mse", alpha=0,
                        family="gaussian")
alpha0.predicted <- predict(alpha0.fit, s=alpha0.fit$lambda.1se, newx=x.test)
```

```r
timetaken <- (proc.time() - ptm)[3]
Ridge_RSS <- mean((y.test-alpha0.predicted)^2)
Ridge_L2norm <- norm(coef(alpha0.fit)[2:51,1]-B, type="2")
TP <- 0
FP <- 0
P <- real_p
for (i in 1:p) {
  if (coef(alpha0.fit)[i+1,1] != 0 & B[i] != 0) {
    TP <- TP + 1
  }
  if (coef(alpha0.fit)[i+1,1] != 0 & B[i] == 0) {
    FP <- FP + 1
  }
}
Ridge_TPR <- TP/P
Ridge_FDR <- FP/(TP+FP)
```

Lasso Regularisation

```r
set.seed(10)
ptm <- proc.time()
alpha1.fit <- cv.glmnet(x.train, y.train, type.measure="mse", alpha=1,
                        family="gaussian")
alpha1.predicted <- predict(alpha1.fit, s=alpha1.fit$lambda.1se, newx=x.test)
timetaken <- (proc.time() - ptm)[3]
LASSO_RSS <- mean((y.test-alpha1.predicted)^2)
LASSO_L2norm <- norm(coef(alpha1.fit)[2:201,1]-B, type="2")
TP <- 0
FP <- 0
P <- real_p
for (i in 1:p) {
  if (coef(alpha1.fit)[i+1,1] != 0 & B[i] != 0) {
    TP <- TP + 1
  }
  if (coef(alpha1.fit)[i+1,1] != 0 & B[i] == 0) {
    FP <- FP + 1
  }
}
Lasso_TPR <- TP/P
Lasso_FDR <- FP/(TP+FP)
```

Elastic-net Regularisation

```r
set.seed(10)
list.of.fits <- list()
results <- data.frame()
for (i in 0:10){
  fit.name <- paste0("alpha", i/10)
  ptm <- proc.time()
  list.of.fits[[fit.name]] <-
    cv.glmnet(x.train, y.train, type.measure="mse", alpha=i/10, family="gaussian")
  predicted <- predict(list.of.fits[[fit.name]],
                       s=list.of.fits[[fit.name]]$lambda.1se, newx=x.test)
  timetaken <- (proc.time() - ptm)[3]
  L2norm <- norm(coef(list.of.fits[[fit.name]])[2:(p+1),1]-B, type="2")
  mse <- mean((y.test-predicted)^2)
  TP <- 0
  FP <- 0
  P <- real_p
  for (j in 1:p) {
    if (coef(list.of.fits[[fit.name]])[j+1,1] != 0 & B[j] != 0) {
      TP <- TP + 1
    }
    if (coef(list.of.fits[[fit.name]])[j+1,1] != 0 & B[j] == 0) {
      FP <- FP + 1
    }
  }

  elastic_TPR <- TP/P
  elastic_FDR <- FP/(TP+FP)
  temp <- data.frame(alpha=i/10, mse=mse, TPR=elastic_TPR, FDR=elastic_FDR,
                     L2norm=L2norm, timetaken=timetaken, fit.name=fit.name)
  results <- rbind(results, temp)
}

results

##            alpha      mse TPR       FDR    L2norm timetaken fit.name
## elapsed     0.0 7.465585   1 0.7000000 2.7361701     0.087   alpha0
## elapsed1    0.1 2.568437   1 0.6590909 1.3934507     0.086 alpha0.1
## elapsed2    0.2 3.436088   1 0.5945946 1.5902021     0.078 alpha0.2
## elapsed3    0.3 2.151500   1 0.5714286 1.1119542     0.082 alpha0.3
## elapsed4    0.4 2.038819   1 0.5454545 1.0353769     0.079 alpha0.4
## elapsed5    0.5 1.792574   1 0.5161290 0.9078314     0.077 alpha0.5
## elapsed6    0.6 1.903764   1 0.4827586 0.9461912     0.075 alpha0.6
## elapsed7    0.7 1.699744   1 0.4642857 0.8433144     0.073 alpha0.7
## elapsed8    0.8 1.948978   1 0.3478261 0.9563155     0.075 alpha0.8
## elapsed9    0.9 1.759953   1 0.3750000 0.8574792     0.074 alpha0.9
## elapsed10   1.0 1.708326   1 0.3750000 0.8296458     0.085   alpha1
```

Least Angle Regression

```
set.seed(10)
LARS_L2norm <- c()
LARS_RSS <- c()
LARS_TPR <- c()
step <- c()
ptm <- proc.time()
lars.fit <- lars(x.train, y.train, type="lar", intercept=FALSE)
lars.predicted.coef <- predict.lars(lars.fit, x.test, type="coefficient")
timetaken <- (proc.time() - ptm)[3]
lars.predicted.fit <- predict.lars(lars.fit, x.test, type="fit")
for (i in 2:51){
  LARS_L2norm[i-1] <- norm(lars.predicted.coef$coefficients[i,]-B, type="2")
  LARS_RSS[i-1] <- mean((y.test-lars.predicted.fit$fit[,i])^2)
}
TP <- 0
FP <- 0
P <- real_p
s <- 22
```

```
for (i in 1:p) {
    if (lars.predicted.coef$coefficients[s,i]!=0 & B[i] != 0) {
      TP <- TP + 1
    }
    if (lars.predicted.coef$coefficients[s,i]!=0 & B[i] == 0) {
      FP <- FP + 1
    }
}
lars_TPR <- TP/P
lars_FDR <- FP/(TP+FP)
```

```
set.seed(10)
# ridge regression, with alpha = 0
relaxed_lasso_fit = cv.glmnet(x.train, y.train, type.measure = "mse", alpha = 0,
                              famiy = "gaussian", relax = TRUE)

relaxed_lasso_predicted = predict(relaxed_lasso_fit,
                                  s = relaxed_lasso_fit$lambda.1se, newx = x.test)
relaxed_lasso_rss = mean((y.test - relaxed_lasso_predicted)^2)
relaxed_lasso_norm <- norm(coef(relaxed_lasso_fit)[2:51,1]-B, type="2")
```

References

[1] Anon. *Ridge Regression.* STAT 508 Applied Data Mining and Statistical Learning. PennState Eberly College of Science. `https://online.stat.psu.edu/stat508/lesson/5/5.1`

[2] Efron et al., 2004, 'Least Angle Regression', *The Annals of Statistics*, 32, 407-421.

[3] Harrell, Frank E. (2015). *Regression Modeling Strategies : With Applications to Linear Models, Logistic and Ordinal Regression, and Survival Analysis.* Springer Series in Statistics. 2nd edn. Cham, Springer.

[4] Hastie, T., Tibshirani, R., Friedman, J. H. (2009). *The Elements of Statistical Learning : Data Mining, Inference, and Prediction.* Springer Series in Statistics. 2nd edn. New York, Springer.

[5] Hastie, T., Efron, B. (2013). *lars: Least Angle Regression, Lasso and Forward Stagewise.* R package version 1.2. `https://CRAN.R-project.org/package=lars`

[6] Hastie, T., Tibshirani, R., Wainwright, M. (2015). *Statistical Learning with Sparsity: The Lasso and Generalizations.* Chapman and Hall/CRC.

[7] Ishwaran, H., Rao, J., Kogalur, U. (2022). *spikeslab : Prediction and Variable Selection Using Spike and Slab Regression.* R package version 1.1.6. `https://cran.r-project.org/package=spikeslab`.

[8] Jackson, S. *Choosing λ.* Machine Learning module MATH42815 of the Masters of Data Science course. Durham University.
`https://bookdown.org/ssjackson300/Machine-Learning-Lecture-Notes/the-lasso.html#different-lambdas`

[9] Kuhn, M. (2021). *caret: Classification and Regression Training.* R package version 6.0-88. `https://CRAN.R-project.org/package=caret`

[10] Pope, P. T., Webster, J.T. (1972). The Use of an F-Statistic in Stepwise Regression Procedures. Technometrics, 14(2), 327–340. `https://www.jstor.org/stable/1267425`

[11] Seber, G.A.F., Lee, A.J. (2003) *Linear Regression Analysis.* Wiley Series in Probability and Statistics. 2nd edn. Hoboken, New Jersey, John Willy and Sons, Inc.

[12] Starmer, J. Regularisation Part 1-3
`https://www.youtube.com/watch?v=Q81RR3yKn30`, `https://www.youtube.com/watch?v=NGf0voTMlcs`, `https://www.youtube.com/watch?v=1dKRdX9bfIo`

[13] Smith, G. (2018). Step away from stepwise. *Journal of Big Data.* 5(32). `https://doi.org/10.1186/s40537-018-0143-6`.

[14] Taboga, M. (2021). "Ridge regression", Lectures on probability theory and mathematical statistics. Kindle Direct Publishing. Online appendix.
`https://www.statlect.com/fundamentals-of-statistics/ridge-regression`

[15] Tibshirani, R. (1996). Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society.* Series B (Methodological), 58(1), 267–288. `http://www.jstor.org/stable/2346178`

[16] Tibshirani, R. (2015). A General Framework for Fast Stagewise Algorithms. *The Journal of Machine Learning Research.* 16(1), 2543-2588.

[17] Weatherwax, J. L., Epstein, D. (2021). *A Solution Manual and Notes for: The Elements of Statistical Learning by Jerome Friedman, Trevor Hastie, and Robert Tibshirani.*
`https://waxworksmath.com/Authors/G_M/Hastie/WriteUp/Weatherwax_Epstein_Hastie_Solution_Manual.pdf` [Accessed 12th June 2022].

[18] Zou, H., Hastie, T. (2005). Regularization and Variable Selection via the Elastic Net. *Journal of the Royal Statistical Society.* Series B (Statistical Methodology), 67(2), 301–320. `http://www.jstor.org/stable/3647580`.