5-2016

Computation of Least Angle Regression coefficient profiles and LASSO estimates.

Sandamala Hettigoda

COMPUTATION OF LEAST ANGLE REGRESSION COEFFICIENT PROFILES AND LASSO ESTIMATES

By

Sandamala Hettigoda
B.Sc., University of Kelaniya, Sri Lanka

A Thesis
Submitted to the Faculty of the
College of Arts and Sciences of the University of Louisville
in Partial Fulfillment of the Requirements
for the Degree of

Master of Arts in Mathematics

Department of Mathematics
University of Louisville
Louisville, KY

May 2016

COMPUTATION OF LEAST ANGLE REGRESSION COEFFICIENT PROFILES AND LASSO ESTIMATES

Submitted by

Sandamala Hettigoda

A Thesis Approved on

April 11, 2016

by the Following Examination Committee:

Professor Ryan Gill, Thesis Director

Professor Jiaxu Li

Professor K. B. Kulasekera

DEDICATION

To my family

ACKNOWLEDGEMENTS

No doubt first and foremost my deepest gratitude is to my adviser, Professor Ryan Gill one of the best teacher that I have had in my life. I am indebted to him for his encouragement, guidance and specially endless patience through out this project. How I forget his supportive and flexibility which help me not to bother at all continuing this thesis.

I would like to thank Professor Jiaxu Li and Professor K. B. Kulasekera, my committee members spending their valuable time and their highly motivated comments.

I would like to thank Professor Gamini Sumanasekara and family who put the foundation to start my higher studies in USA.

I like to express my gratitude to my dear friends Allan, Apsara, Bakeerathan, Chanchala and Udika who support me in numerous ways.

Finally I would like to thank to my family in Sri Lanka specially my eldest brother Nandana Hettigoda, my beloved husband Sujeewa, ever loving daughter Viyathma and son Vethum.

ABSTRACT

COMPUTATION OF LEAST ANGLE REGRESSION COEFFICIENT PROFILES AND LASSO ESTIMATES

Sandamala Hettigoda

May 14, 2016

Variable selection plays a significant role in statistics. There are many variable selection methods. Forward stagewise regression takes a different approach among those. In this thesis Least Angle Regression (LAR) is discussed in detail. This approach has similar principles as forward stagewise regression but does not suffer from its computational difficulties. By using a small artificial data set and the well-known Longley data set, the LAR algorithm is illustrated in detail and the coefficient profiles are obtained. Furthermore a penalized approach to variable reduction called the LASSO is discussed, and it is shown how to compute its coefficient profiles efficiently using the LAR algorithm with a small modification. Finally, a method called K-fold cross validation used to select the constraint parameter for the LASSO is presented and illustrated with the Longley data.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

Linear regression is a method of fitting straight lines in accordance to the patterns of data, and it is one of the most widely used of all statistical techniques to analyze data. Simple linear regression is used to explain the relationship between a dependent variable (y) and an independent variable (x). The model with an intercept is represented by $y = \beta_0 + \beta_1 x + \epsilon$, where ϵ is a error term with mean 0 and the variance is assumed to be a constant σ^2. Given observed data points $(x_1, y_1), \ldots, (x_n, y_n)$, the simple linear regression model for the ith dependent variable is $y_i = \beta_0 + \beta_1 x_i + \epsilon_i$ where $\epsilon_1, \ldots, \epsilon_n$ are independent and identically distributed random variables with variance σ^2. In some cases, it is preferable to use a model where the dependent variable is centered and the independent variable is rescaled; i.e., we define $x_i^* = \frac{x_i - \bar{x}}{s_x}$ and $y_i^c = y_i - \bar{y}$ where $\bar{x} = \frac{1}{n}\sum_{i=1}^n x_i$ and $\bar{y} = \frac{1}{n}\sum_{i=1}^n y_i$ are the sample means and $s_x = \sqrt{\frac{1}{n-1}\sum_{i=1}^n (x_i - \bar{x})^2}$ is the sample variance of the x variable. With the centered data representation, the simple linear regression model can be expressed as

$$y_i^c = \beta_1^* x_i^* + \epsilon_i. \tag{1.1}$$

Note that $\beta_1 = \beta_1^*/s_x$ and $\beta_0 = \bar{y} - \beta_1^* \bar{x}/s_x = \bar{y} - \beta_1 \bar{x}$.

The best fit line $\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x$ is found by minimizing the residual sum of squared errors $\sum_{i=1}^n r_i^2$ where $r_i = y_i - \hat{y}_i$ represents the i^{th} residual. The residual sum of squares can be expressed as

$$\text{RSS} = (y_1 - \hat{\beta}_0 - \hat{\beta}_1 x_1)^2 + (y_2 - \hat{\beta}_0 - \hat{\beta}_1 x_2)^2 + \cdots + (y_n - \hat{\beta}_0 - \hat{\beta}_1 x_n)^2.$$

The method of least squares chooses $\hat{\beta}_1 = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}$ and $\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}$ which minimizes the RSS. In the scaled model (1.1), the estimate is $\hat{\beta}_1^* = s_x \hat{\beta}_1$. The sample correlation is defined by

$$\text{Cor}(\boldsymbol{x}, \boldsymbol{y}) = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2} \sqrt{\sum_{i=1}^n (y_i - \bar{y})^2}}$$

which is important in prediction. An alternate formula for the slope estimate in simple linear regression is

$$\hat{\beta}_1 = \text{Cor}(\boldsymbol{x}, \boldsymbol{y}) \frac{s_y}{s_x}$$

which shows the relationship between $\hat{\beta}_1$ and the sample correlation.

When there are p distinct predictors then the multiple linear regression model is $y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p + \epsilon$. Given observed data points $(x_{11}, \ldots, x_{1p}, y_1), \ldots, (x_{n1}, \ldots, x_{np}, y_n)$, the simple linear regression model for the ith dependent variable is $y_i = \beta_0 + \beta_1 x_{i1} + \ldots + \beta_p x_{ip} + \epsilon_i$ where $\epsilon_1, \ldots, \epsilon_n$ are independent and identically distributed random variables with variance σ^2. Similar to simple linear regression, least squares estimation chooses $\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_p$ which minimizes

$$RSS = \sum_{i=1}^n \left(y_i - \beta_0 - \beta_1 x_{i1} - \ldots - \beta_p x_{ip} \right)^2 .$$

In matrix form, the goal is to estimate

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{bmatrix} .$$

and the least squares estimate can be expressed as $\hat{\boldsymbol{\beta}} = (\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{X}^\top \boldsymbol{y}$ where $\boldsymbol{X}$ is $n \times (p+1)$ matrix with columns

$$\boldsymbol{J} = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}, \boldsymbol{x}_1 = \begin{bmatrix} x_{11} \\ \vdots \\ x_{n1} \end{bmatrix}, \ldots, \boldsymbol{x}_p = \begin{bmatrix} x_{1p} \\ \vdots \\ x_{np} \end{bmatrix}, \text{ and } \boldsymbol{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} .$$

Often, each of the p predictor variables are scaled using the formulas

$$x_{ij}^* = \frac{x_{ij} - \bar{x}_j}{s_j}$$

and the response variable is centered using the formula

$$y_i^c = y_i - \bar{y}$$

where $\bar{x}_j = \frac{\sum_{i=1}^n x_{ij}}{n}$ is the sample mean for the jth variable, and $s_j = \sqrt{\frac{1}{n-1}\sum_{i=1}^n (x_{ij} - \bar{x}_j)^2}$ is the sample standard deviation for the jth variable. Letting

$$\boldsymbol{x}_1^* = \begin{bmatrix} x_{11}^* \\ \vdots \\ x_{n1}^* \end{bmatrix}, \ldots, \boldsymbol{x}_p^* = \begin{bmatrix} x_{1p}^* \\ \vdots \\ x_{np}^* \end{bmatrix}, \text{ and } \boldsymbol{y}^c = \begin{bmatrix} y_1^c \\ \vdots \\ y_n^c \end{bmatrix},$$

the least squares estimate for the centered model is given by

$$\hat{\boldsymbol{\beta}}^* = (\boldsymbol{X}^{*\top}\boldsymbol{X}^*)^{-1}\boldsymbol{X}^{*\top}\boldsymbol{y}^c.$$

Especially when there are a large number of predictor variables available in the multiple linear regression model, it is desirable to consider strategies for selectively including variables in the model. Using too many predictor variables can lead to overfitting and standard error estimates for the coefficients can become inflated as discussed in Chapter 4 of Hocking (2013). Of course, if the number of predictor variables is greater than the number of observations, it is not even possible to include all of the predictor variables since $(\boldsymbol{X}^\top\boldsymbol{X})^{-1}$ does not exist if $p \geq n$.

Different types of variable selection methods exist for regression models in statistics. The goal of each method is to identify the best subset among many variables to include in a model. Here are some basic strategies that can be used for variable selection. *Forward selection* starts with a null model (no predictors and only an intercept) and then proceeds to add one variable at a time according to the correlation until no additional variable is significant. *Backward elimination*

starts with the full model and deletes one variable at a time until all remaining variables are significant. *Stepwise regression* is a combination of forward selection and backward elimination methods. This method requires two significance levels. After each step, the significance level is checked, and it is determined whether a variable should be added to or removed from the model. *All subsets regression* builds all 2^p possible models (including a model with only the intercept, all one variable models, all two variables models, and so on).

Forward stagewise regression takes a different approach. It starts like forward selection with no variables included (usually the predictors are scaled and the responses are centered so this corresponds to a model with only an intercept) by setting the estimates for all coefficients equal to 0. Then the current residuals r_i are equal to the centered values y_i^c, and the predictor $\boldsymbol{x}_j$ most correlated with $\boldsymbol{r}$ is selected. However, instead of fully adding the predictor $\boldsymbol{x}_j^*$ to the model, the coefficient estimate for β_j is only incremented by a small amount $\varepsilon \cdot \text{sign}\langle \boldsymbol{r}, \boldsymbol{x}_j^* \rangle$ and the residuals are updated. This step is repeated many times until the remaining residuals are uncorrelated with each of the predictors. More discussion on forward selection, backward elimination, forward stagewise, all subsets, and forward stagewise regression is given in Hastie, Tibshirani, and Friedman (2013).

If the value of ε used in forward stagewise regression is small, the coefficient estimates are updated very slowly from step to step and the number of steps required to complete the algorithm can be very large. In this thesis, a closely related method called *least angle regression* (LAR) which is motivated by the same principles but more computationally efficient will be discussed in detail. The algorithm presented here is equivalent to that developed and described in Efron *et al.* (2004) and Hastie, Tibshirani, and Friedman (2013). However, the notation used herein differs significantly from those classic references, and the paths for the coefficient profiles are parametrized differently.

In Chapter 2, the LAR algorithm is presented in detail, custom R code is provided implementing the presented version of the algorithm, and the method is illustrated using a small artificial data example as well as the well-known Longley data set. In Chapter 3, a penalized approach to variable reduction called the *LASSO* is discussed and a method for computing the LASSO estimates using a modification of the LAR algorithm is presented and illustrated using the Longley data set. Finally, in Chapter 4, a method called *k-fold cross validation* for choosing a model along the coefficient path for the LAR or LASSO algorithm is described and illustrated with the Longley data set.

CHAPTER 2
LEAST ANGLE REGRESSION

Just as in forward stagewise regression, the idea behind least angle regression is to move the coefficient estimates in the direction in which the predictor variable(s) is most correlated with the remaining residual. Instead of moving in steps of size ε, the coefficient path for LAR changes continuously as it moves from a vector of zeros to the least squares solution.

Finding the variable that is most highly correlated (in absolute terms) with the current residual is equivalent to finding the vector(s) $\boldsymbol{x}_j^*$ which makes the smallest angle with the residual $\boldsymbol{r}$. The angle θ between two vectors $\boldsymbol{x}_j^*$ and $\boldsymbol{r}$ can be determined by

$$\begin{aligned} \cos(\theta) &= \frac{\langle \boldsymbol{x}_j^*, \boldsymbol{r} \rangle}{\|\boldsymbol{x}_j^*\| \|\boldsymbol{r}\|} \\ &= \frac{\langle \boldsymbol{x}_j^*, \boldsymbol{r} \rangle}{\|\boldsymbol{r}\|} \qquad (\text{since } \|\boldsymbol{x}_j^*\| = 1) \\ &= \text{Cor}(\boldsymbol{x}_j^*, \boldsymbol{r}). \end{aligned} \tag{2.1}$$

Thus, the absolute correlation $|\text{Cor}(\boldsymbol{x}_j^*, \boldsymbol{r})|$ is maximized when $|\cos(\theta)|$ is maximized and consequently when the the absolute value of the angle, $|\theta|$, is minimized. It can be seen that the variable(s) which maximize (2.1) can be found by maximizing $\langle \boldsymbol{x}_j^*, \boldsymbol{r} \rangle$ since $\|\boldsymbol{r}\|$ does not depend on the index j.

A basic description of the LAR algorithm is as follows, similar to the algorithm provided in Algorithm 3.2 on page 74 of Hastie, Tibshirani, and Friedman (2013).

1. Standardized the predictors to have mean zero and unit norm. Start with all estimates of the coefficients $\beta_1^*, \beta_2^*, \ldots, \beta_p^*$ to be equal to 0 with the residual $\boldsymbol{r}_0^{\angle} = \boldsymbol{y}^c$.

2. Find the predictor $\boldsymbol{x}_{\hat{j}_1^{\angle}}$ most correlated with the response $\boldsymbol{r}_0^{\angle}$.

3. Move the estimate of $\beta^*_{\hat{j}_1^{\angle}}$ from 0 towards the least squares coefficients until some other predictor $\boldsymbol{x}_{\hat{j}_2^{\angle}}$ has as large a correlation with the current residual $\tilde{\boldsymbol{r}}_1(\alpha)$ as $\boldsymbol{x}_{\hat{j}_1^{\angle}}$ does.

4. At this point instead of continuing in the direction based on $\boldsymbol{x}_{j_1}$, LAR proceeds in a direction of equiangularity between the two predictors $\boldsymbol{x}_{\hat{j}_1^{\angle}}$ and $\boldsymbol{x}_{\hat{j}_2^{\angle}}$. A third variable $\boldsymbol{x}_{\hat{j}_3^{\angle}}$ eventually earns its way into the most correlated (active set), and then LAR proceeds equiangularly between $\boldsymbol{x}_{\hat{j}_1^{\angle}}$,$\boldsymbol{x}_{\hat{j}_2^{\angle}}$, and $\boldsymbol{x}_{\hat{j}_3^{\angle}}$.

5. Continue adding variables to the active set in this way moving in the direction defined by least angle direction. After i steps this process gives a linear model with predictors $\boldsymbol{x}_{\hat{j}_1^{\angle}}, \boldsymbol{x}_{\hat{j}_2^{\angle}}, \boldsymbol{x}_{\hat{j}_3^{\angle}}, \ldots, \boldsymbol{x}_{\hat{j}_i^{\angle}}$. After $\min(n-1, p)$ steps, the full least squares solution is attained and the LAR algorithm is complete.

2.1 LAR Algorithm

In this section, the mathematical details of the LAR algorithm are developed in detail. This presentation of the LAR algorithm uses matrices which explicitly describe the coefficient directions on the ith step in terms of $\boldsymbol{X}^*$ and $\boldsymbol{r}_{i-1}^{\angle}$ instead of using the active set terminology described in the classic references Hastie, Tibshirani, and Friedman (2013) and Efron *et al.* (2004).

On the initial step, let $\boldsymbol{r}_0^{\angle} = \boldsymbol{y}^c$ and $\hat{\boldsymbol{\beta}}_0^{*\angle} = [\hat{\beta}_{0,1}^{*\angle}, \ldots, \hat{\beta}_{0,p}^{*\angle}]^\top = \boldsymbol{0}$. Then choose the first variable that enters the model using the formula

$$\hat{j}_1^{\angle} = \underset{j}{\operatorname{argmax}} \left| \langle \boldsymbol{x}_j^*, \boldsymbol{r}_0^{\angle} \rangle \right|.$$

Here is the algorithm for the ith step where $i = 1, \ldots, \min\{p, n-1\}$. Let $\boldsymbol{e}_j$ be the jth standard unit vector in $\mathbb{R}^p$; for example, $\boldsymbol{e}_1 = [1, 0, \ldots, 0]^\top$. The direction on the ith step is $\boldsymbol{d}_i^\angle = \boldsymbol{E}_i^\angle \left(\boldsymbol{E}_i^{\angle\top} \boldsymbol{X}^{*\top} \boldsymbol{X}^* \boldsymbol{E}_i^\angle\right)^{-1} \boldsymbol{E}_i^{\angle\top} \boldsymbol{X}^{*\top} \boldsymbol{r}_{i-1}^\angle$ where $\boldsymbol{E}_i^\angle$ is a matrix with columns $\boldsymbol{e}_{\hat{j}_1^\angle}, \ldots, \boldsymbol{e}_{\hat{j}_i^\angle}$. Then update the coefficient estimate using the formula $\tilde{\boldsymbol{\beta}}_i^{*\angle}(\alpha) = \hat{\boldsymbol{\beta}}_{i-1}^{*\angle} + \alpha \boldsymbol{d}_i^\angle$ where α is a value between $[0, 1]$ which represents how far the estimate of $\boldsymbol{\beta}$ moves in the direction $\boldsymbol{d}_i^\angle$ before another variable enters the model and the direction changes again. We choose α on the ith step by finding the smallest value of α such that the angle between the remaining residual $\tilde{\boldsymbol{r}}_i^\angle(\alpha) = \boldsymbol{r}_{i-1}^\angle - \alpha \boldsymbol{X}^* \boldsymbol{d}_i^\angle$ and one of the variables not in the model on the ith step (that is, a variable such that $\hat{\beta}_{i-1,j}^{*\angle} = 0$) equals the angle between $\tilde{\boldsymbol{r}}_i^\angle(\alpha)$ and a variable in the model.

Mathematically, we choose α as follows. The angle between $\tilde{\boldsymbol{r}}_i^\angle(\alpha)$ and the jth variable $\boldsymbol{x}_j^*$ equals the angle between $\tilde{\boldsymbol{r}}_i^\angle(\alpha)$ and $\boldsymbol{x}_{\hat{j}_i^\angle}^*$ when

$$\langle \tilde{\boldsymbol{r}}_i^\angle(\alpha), \boldsymbol{x}_j^* \rangle = \langle \tilde{\boldsymbol{r}}_i^\angle(\alpha), \boldsymbol{x}_{\hat{j}_i^\angle}^* \rangle. \tag{2.2}$$

Since it follows that

$$\begin{aligned}
\langle \tilde{\boldsymbol{r}}_i^\angle(\alpha), \boldsymbol{x}_j^* \rangle &= \langle \boldsymbol{r}_{i-1}^\angle - \alpha \boldsymbol{X}^* \boldsymbol{d}_i^\angle, \boldsymbol{x}_j^* \rangle \\
&= \langle \boldsymbol{r}_{i-1}^\angle, \boldsymbol{x}_j^* \rangle - \alpha \langle \boldsymbol{X}^* \boldsymbol{d}_i^\angle, \boldsymbol{x}_j^* \rangle \\
&= \langle \boldsymbol{r}_{i-1}^\angle, \boldsymbol{x}_j^* \rangle - \alpha \langle \boldsymbol{H}_i^\angle \boldsymbol{r}_{i-1}^\angle, \boldsymbol{x}_j^* \rangle \\
&= \langle \boldsymbol{r}_{i-1}^\angle, \boldsymbol{x}_j^* \rangle - \alpha \langle \boldsymbol{r}_{i-1}^\angle, \boldsymbol{H}_i^\angle \boldsymbol{x}_j^* \rangle
\end{aligned}$$

where $\boldsymbol{H}_i^\angle = \boldsymbol{Z}_i (\boldsymbol{Z}_i^T \boldsymbol{Z}_i)^{-1} \boldsymbol{Z}_i^T$ is a hat matrix for $\boldsymbol{Z}_i = \boldsymbol{X}^* \boldsymbol{E}_i^\angle$, the solution to (2.2) is

$$\begin{aligned}
\tilde{\alpha}_{i,j}^+ &= \frac{\langle \boldsymbol{r}_{i-1}^\angle, \boldsymbol{x}_{\hat{j}_i^\angle}^* \rangle - \langle \boldsymbol{r}_{i-1}^\angle, \boldsymbol{x}_j^* \rangle}{\langle \boldsymbol{r}_{i-1}^\angle, \boldsymbol{H}_i^\angle \boldsymbol{x}_{\hat{j}_i^\angle}^* \rangle - \langle \boldsymbol{r}_{i-1}^\angle, \boldsymbol{H}_i^\angle \boldsymbol{x}_j^* \rangle} \\
&= \frac{\langle \boldsymbol{r}_{i-1}^\angle, \boldsymbol{x}_{\hat{j}_i^\angle}^* \rangle - \langle \boldsymbol{r}_{i-1}^\angle, \boldsymbol{x}_j^* \rangle}{\langle \boldsymbol{r}_{i-1}^\angle, \boldsymbol{x}_{\hat{j}_i^\angle}^* \rangle - \langle \boldsymbol{r}_{i-1}^\angle, \boldsymbol{H}_i^\angle \boldsymbol{x}_j^* \rangle}
\end{aligned} \tag{2.3}$$

$$
\begin{aligned}
&= \frac{\langle \boldsymbol{r}_{i-1}^{\angle}, \boldsymbol{x}^*_{\hat{j}_i^{\angle}} \rangle - \langle \boldsymbol{r}_{i-1}^{\angle}, \boldsymbol{x}_j^* \rangle}{\langle \boldsymbol{r}_{i-1}^{\angle}, \boldsymbol{x}^*_{\hat{j}_i^{\angle}} \rangle - \langle \boldsymbol{H}_i^{\angle} \boldsymbol{r}_{i-1}^{\angle}, \boldsymbol{x}_j^* \rangle} \\
&= \frac{\langle \boldsymbol{r}_{i-1}^{\angle}, \boldsymbol{x}^*_{\hat{j}_i^{\angle}} \rangle - \langle \boldsymbol{r}_{i-1}^{\angle}, \boldsymbol{x}_j^* \rangle}{\langle \boldsymbol{r}_{i-1}^{\angle}, \boldsymbol{x}^*_{\hat{j}_i^{\angle}} \rangle - \langle \boldsymbol{X}^* \boldsymbol{d}_i^{\angle}, \boldsymbol{x}_j^* \rangle}.
\end{aligned}
$$

Equation (2.3) holds since $\boldsymbol{Z}_i = \boldsymbol{H}_i^{\angle} \boldsymbol{Z}_i$ which implies that

$$
\left[\boldsymbol{x}^*_{\hat{j}_1^{\angle}} \quad \cdots \quad \boldsymbol{x}^*_{\hat{j}_i^{\angle}} \right] = \boldsymbol{X}^* \boldsymbol{E}_i^{\angle} = \boldsymbol{H}_i^{\angle} (\boldsymbol{X}^* \boldsymbol{E}_i^{\angle}) = \left[\boldsymbol{H}_i^{\angle} \boldsymbol{x}^*_{\hat{j}_1^{\angle}} \quad \cdots \quad \boldsymbol{H}_i^{\angle} \boldsymbol{x}^*_{\hat{j}_i^{\angle}} \right]
$$

so $\boldsymbol{H}_i^{\angle} \boldsymbol{x}^*_{\hat{j}_k^{\angle}} = \boldsymbol{x}^*_{\hat{j}_k^{\angle}}$ for $k = 1, \ldots, i$. Similarly, the angle between $\tilde{\boldsymbol{r}}_i^{\angle}(\alpha)$ and $-\boldsymbol{x}_j^*$ equals the angle between $\tilde{\boldsymbol{r}}_i^{\angle}(\alpha)$ and $\boldsymbol{x}^*_{\hat{j}_i^{\angle}}$ when

$$
\tilde{\alpha}_{i,j}^{-} = \frac{\langle \boldsymbol{r}_{i-1}^{\angle}, \boldsymbol{x}^*_{\hat{j}_i^{\angle}} \rangle + \langle \boldsymbol{r}_{i-1}^{\angle}, \boldsymbol{x}_j^* \rangle}{\langle \boldsymbol{r}_{i-1}^{\angle}, \boldsymbol{x}^*_{\hat{j}_i^{\angle}} \rangle + \langle \boldsymbol{X}^* \boldsymbol{d}_i^{\angle}, \boldsymbol{x}_j^* \rangle}.
$$

So, the smallest value of α such that a new variable should enter the model is

$$
\hat{\alpha}_i^{\angle} = \min \left\{ \alpha \in [0,1] : \alpha = \tilde{\alpha}_{i,j}^{+} \text{ or } \alpha = \tilde{\alpha}_{i,j}^{-} \text{ for some } j \text{ such that } \hat{\beta}_{i-1,j}^{*\angle} = 0 \right\}.
$$

Then $\hat{\boldsymbol{\beta}}_i^{*\angle} = \tilde{\boldsymbol{\beta}}_i^{*\angle}(\hat{\alpha}_i^{\angle})$, $\boldsymbol{r}_i^{\angle} = \boldsymbol{y}^c - \boldsymbol{X}^* \hat{\boldsymbol{\beta}}_i^{*\angle} = \boldsymbol{r}_{i-1}^{\angle} - \hat{\alpha}_i^{\angle} \boldsymbol{X}^* \boldsymbol{d}_i^{\angle}$, and we move to the next step where $\hat{j}_{i+1}^{\angle}$ is the value of j such that $\tilde{\alpha}_{i,j}^{+} = \hat{\alpha}_i^{\angle}$ or $\tilde{\alpha}_{i,j}^{-} = \hat{\alpha}_i^{\angle}$.

So, the vector of LAR coefficient profiles based on the centered responses and standardized inputs can described by

$$
\hat{\boldsymbol{\beta}}_i^{*\angle}(\alpha) = \begin{cases}
\boldsymbol{0} & \text{if } i = 0 \\
\tilde{\boldsymbol{\beta}}_1^{*\angle}(\alpha) & \text{if } i = 1, 0 \le \alpha \le \hat{\alpha}_1^{\angle} \\
\vdots & \vdots \\
\tilde{\boldsymbol{\beta}}^{*\angle}_{\min\{p,n-1\}}(\alpha) & \text{if } i = \min\{p, n-1\}, 0 \le \alpha \le \hat{\alpha}^{\angle}_{\min\{p,n-1\}}
\end{cases}
$$

and the vector of coefficient profiles based on the original scale is

$$
\hat{\boldsymbol{\beta}}_i^{\angle}(\alpha) = \left[\hat{\beta}_{i,0}^{\angle}(\alpha), \hat{\beta}_{i,1}^{\angle}(\alpha), \ldots, \hat{\beta}_{i,p}^{\angle}(\alpha) \right]^{\top}
$$

where

$$
\hat{\beta}_{i,j}^{\angle}(\alpha) = \frac{\hat{\beta}_{i,j}^{*\angle}(\alpha)}{s_{x_j}} \text{ for } j = 1, \ldots, p
$$

and

$$\hat{\beta}_{i,0}^{\angle}(\alpha) = \bar{y} - \frac{\hat{\beta}_{i,1}^{*\angle}(\alpha)}{s_{x_1}}\bar{x}_1 - \ldots - \frac{\hat{\beta}_{i,p}^{*\angle}(\alpha)}{s_{x_p}}\bar{x}_p.$$

A mathematical summary of the algorithm is given in Table 2.1.

2.2 Example

Here is a small artificial example to illustrate the LAR method. Suppose that we want to obtain the LAR coefficient profiles for

$$\boldsymbol{X} = [\boldsymbol{x}_1 \;\; \boldsymbol{x}_2 \;\; \boldsymbol{x}_3] = \begin{bmatrix} 1 & 1 & 4 \\ 5 & 3 & 5 \\ 6 & 4 & 7 \\ 6 & 4 & 1 \\ 6 & 5 & 4 \\ 6 & 7 & 3 \end{bmatrix} \text{ and } \boldsymbol{y} = \begin{bmatrix} 6 \\ 8 \\ 6 \\ 7 \\ 5 \\ 4 \end{bmatrix}.$$

Then we standardize the inputs and center the outputs to obtain

$$\boldsymbol{X}^* = [\boldsymbol{x}_1^* \;\; \boldsymbol{x}_2^* \;\; \boldsymbol{x}_3^*] = \begin{bmatrix} -2 & -1.5 & 0 \\ 0 & -0.5 & 0.5 \\ 0.5 & 0 & 1.5 \\ 0.5 & 0 & -1.5 \\ 0.5 & 0.5 & 0 \\ 0.5 & 1.5 & -0.5 \end{bmatrix} \text{ and } \boldsymbol{y}^c = \begin{bmatrix} 0 \\ 2 \\ 0 \\ 1 \\ -1 \\ -2 \end{bmatrix}.$$

We initialize $\boldsymbol{r}_0^{\angle} = \boldsymbol{y}^c = [0, 2, 0, 1, -1, -2]^\top$ and $\hat{\boldsymbol{\beta}}_0^{*\angle} = [0, 0, 0]^\top$. Then we compute $\langle \boldsymbol{x}_1^*, \boldsymbol{r}_0^{\angle} \rangle = -1$, $\langle \boldsymbol{x}_2^*, \boldsymbol{r}_0^{\angle} \rangle = -4.5$, and $\langle \boldsymbol{x}_3^*, \boldsymbol{r}_0^{\angle} \rangle = 0.5$ to determine that $\hat{j}_1^{\angle} = 2$.

Consequently, on the first step, we have $\boldsymbol{E}_1^{\angle} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$ and

initial	$\boldsymbol{r}_0^{\angle} = \boldsymbol{y}^c$
step	$\hat{\boldsymbol{\beta}}_0^{*\angle} = [\hat{\beta}_{0,1}^{*\angle}, \ldots, \hat{\beta}_{0,p}^{*\angle}]^\top = \boldsymbol{0}$
	$\hat{j}_1^{\angle} = \underset{j}{\operatorname{argmax}} \left\lvert \langle \boldsymbol{x}_j^*, \boldsymbol{r}_0^{\angle} \rangle \right\rvert$
ith	$\boldsymbol{E}_i^{\angle} = \left[\boldsymbol{e}_{\hat{j}_1^{\angle}} \quad \cdots \quad \boldsymbol{e}_{\hat{j}_i^{\angle}} \right]$
step	$\boldsymbol{d}_i^{\angle} = \boldsymbol{E}_i^{\angle} \left(\boldsymbol{E}_i^{\angle\top} \boldsymbol{X}^{*\top} \boldsymbol{X}^* \boldsymbol{E}_i^{\angle} \right)^{-1} \boldsymbol{E}_i^{\angle\top} \boldsymbol{X}^{*\top} \boldsymbol{r}_{i-1}^{\angle}$
	$\tilde{\alpha}_{i,j}^{\pm} = \frac{\langle \boldsymbol{r}_{i-1}^{\angle}, \boldsymbol{x}_{\hat{j}_i^{\angle}}^* \rangle \mp \langle \boldsymbol{r}_{i-1}^{\angle}, \boldsymbol{x}_j^* \rangle}{\langle \boldsymbol{r}_{i-1}^{\angle}, \boldsymbol{x}_{\hat{j}_i^{\angle}}^* \rangle \mp \langle \boldsymbol{X}^* \boldsymbol{d}_i^{\angle}, \boldsymbol{x}_j^* \rangle}$ for j such that $\hat{\beta}_{i-1,j}^{*\angle} = 0$
	$\hat{\alpha}_i^{\angle} = \min \left\{ \alpha \in [0,1] : \alpha = \tilde{\alpha}_{i,j}^{+} \text{ or } \alpha = \tilde{\alpha}_{i,j}^{-} \right\}$
	$\hat{\boldsymbol{\beta}}_i^{*\angle}(\alpha) = \hat{\boldsymbol{\beta}}_{i-1}^{*\angle} + \alpha \boldsymbol{d}_i^{\angle}, \alpha \in [0, \hat{\alpha}_i^{\angle}]$
	$\boldsymbol{r}_i^{\angle} = \boldsymbol{r}_{i-1}^{\angle} - \hat{\alpha}_i^{\angle} \boldsymbol{X}^* \boldsymbol{d}_i^{\angle}$
	$\hat{j}_{i+1}^{\angle}$ is the value j such that $\tilde{\alpha}_{i,j}^{+} = \hat{\alpha}_i^{\angle}$ or $\tilde{\alpha}_{i,j}^{-} = \hat{\alpha}_i^{\angle}$
	Continue until $\hat{\alpha}_i^{\angle} = 1$.
original	$\hat{\beta}_{i,j}^{\angle}(\alpha) = \frac{\hat{\beta}_{i,j}^{*\angle}(\alpha)}{s_j}$ for all i and for $j = 1, \ldots, p$
scale	$\hat{\beta}_{i,0}^{\angle}(\alpha) = \bar{y} - \frac{\hat{\beta}_{i,1}^{*\angle}(\alpha)}{s_1} \bar{x}_1 - \ldots - \frac{\hat{\beta}_{i,p}^{*\angle}(\alpha)}{s_p} \bar{x}_p$ for all i

Table 2.1 – Summary of the algorithm to obtain the coefficient profiles based on the LAR method.

$$\boldsymbol{d}_1^{\angle} = \boldsymbol{E}_1^{\angle}\left(\boldsymbol{E}_1^{\angle\top}\boldsymbol{X}^{*\top}\boldsymbol{X}^{*}\boldsymbol{E}_1^{\angle}\right)^{-1}\boldsymbol{E}_1^{\angle\top}\boldsymbol{X}^{*\top}\boldsymbol{r}_0^{\angle} = \begin{bmatrix} 0 \\ -0.9 \\ 0 \end{bmatrix}.$$

Let $\tilde{\boldsymbol{\beta}}_1^{*\angle}(\alpha) = \hat{\boldsymbol{\beta}}_0^{*\angle} + \alpha\boldsymbol{d}_1^{\angle} = \begin{bmatrix} 0 \\ -0.9\alpha \\ 0 \end{bmatrix}$ and

$$\tilde{\boldsymbol{r}}_1^{\angle}(\alpha) = \boldsymbol{y}^c - \boldsymbol{X}^{*}\tilde{\boldsymbol{\beta}}_1^{*\angle}(\alpha) = \boldsymbol{r}_0^{\angle} - \alpha\boldsymbol{X}^{*}\boldsymbol{d}_1^{\angle} = \begin{bmatrix} -1.35\alpha \\ 2 - 0.45\alpha \\ 0 \\ 1 \\ -1 + 0.45\alpha \\ -2 + 1.35\alpha \end{bmatrix}$$

for $\alpha \in [0,1]$. Then we compute $\pm\langle \boldsymbol{x}_j^{*}, \tilde{\boldsymbol{r}}_1^{\angle}(\alpha)\rangle$ for $j = 1, 2, 3$. We have $\langle \boldsymbol{x}_1^{*}, \tilde{\boldsymbol{r}}_1^{\angle}(\alpha)\rangle = -1 + 3.6\alpha$, $\langle \boldsymbol{x}_2^{*}, \tilde{\boldsymbol{r}}_1^{\angle}(\alpha)\rangle = -4.5 + 4.5\alpha$, and $\langle \boldsymbol{x}_3^{*}, \tilde{\boldsymbol{r}}_1^{\angle}(\alpha)\rangle = 0.5 - 0.9\alpha$. These line segments are plotted in Figure 2-1.

Then, we have

$$\begin{aligned}
\langle \boldsymbol{x}_1^{*}, \tilde{\boldsymbol{r}}_1^{\angle}(\tilde{\alpha}_{11}^{+})\rangle = \langle \boldsymbol{x}_2^{*}, \tilde{\boldsymbol{r}}_1^{\angle}(\tilde{\alpha}_{11}^{+})\rangle \quad &\Rightarrow \quad \tilde{\alpha}_{11}^{+} = \frac{-4.5+1}{-4.5+3.6} = \frac{35}{9} \notin [0,1) \\
\langle -\boldsymbol{x}_1^{*}, \tilde{\boldsymbol{r}}_1^{\angle}(\tilde{\alpha}_{11}^{+})\rangle = \langle \boldsymbol{x}_2^{*}, \tilde{\boldsymbol{r}}_1^{\angle}(\tilde{\alpha}_{11}^{+})\rangle \quad &\Rightarrow \quad \tilde{\alpha}_{11}^{-} = \frac{-4.5-1}{-4.5-3.6} = \frac{55}{81} \approx .679 \\
\langle \boldsymbol{x}_3^{*}, \tilde{\boldsymbol{r}}_1^{\angle}(\tilde{\alpha}_{13}^{+})\rangle = \langle \boldsymbol{x}_2^{*}, \tilde{\boldsymbol{r}}_1^{\angle}(\tilde{\alpha}_{13}^{+})\rangle \quad &\Rightarrow \quad \tilde{\alpha}_{13}^{+} = \frac{-4.5-0.5}{-4.5-0.9} = \frac{25}{27} \approx .926 \\
\langle -\boldsymbol{x}_3^{*}, \tilde{\boldsymbol{r}}_1^{\angle}(\tilde{\alpha}_{13}^{-})\rangle = \langle \boldsymbol{x}_2^{*}, \tilde{\boldsymbol{r}}_1^{\angle}(\tilde{\alpha}_{13}^{-})\rangle \quad &\Rightarrow \quad \tilde{\alpha}_{13}^{-} = \frac{-4.5+0.5}{-4.5+0.9} = \frac{10}{9} \notin [0,1).
\end{aligned}$$

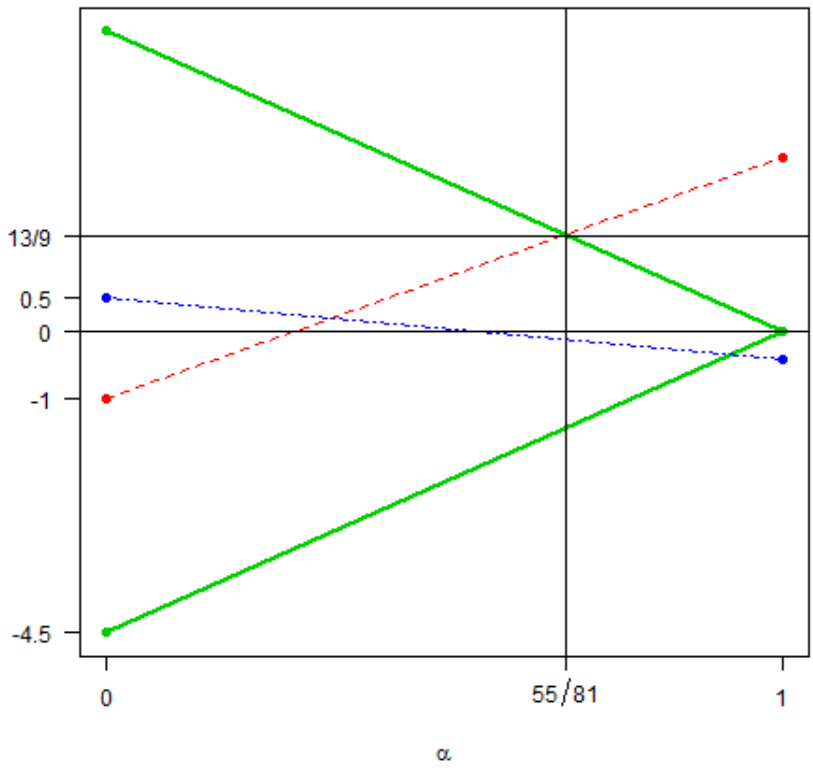

Figure 2.1 – Line segments $\langle \boldsymbol{x}_1^*, \tilde{\boldsymbol{r}}_1^{\angle}(\alpha) \rangle$ (red), $\pm\langle \boldsymbol{x}_2^*, \tilde{\boldsymbol{r}}_1^{\angle}(\alpha) \rangle$ (green), and $\langle \boldsymbol{x}_3^*, \tilde{\boldsymbol{r}}_1^{\angle}(\alpha) \rangle$ (blue) for step 1 of the LAR algorithm.

Thus, we have $\hat{\alpha}_1^{\angle} = \frac{55}{81}$ so that $\hat{\boldsymbol{\beta}}_1^{*\angle} = \tilde{\boldsymbol{\beta}}_1^{*\angle}(\hat{\alpha}_1^{\angle}) = \begin{bmatrix} 0 \\ -\frac{11}{18} \\ 0 \end{bmatrix} \approx \begin{bmatrix} 0 \\ -0.611 \\ 0 \end{bmatrix}$,

$$\boldsymbol{r}_1^{\angle} = \tilde{\boldsymbol{r}}_1^{\angle}(\hat{\alpha}_1^{\angle}) = \begin{bmatrix} -\frac{11}{12} \\ \frac{61}{36} \\ 0 \\ 1 \\ -\frac{25}{36} \\ -\frac{13}{12} \end{bmatrix},$$

and $\hat{j}_2^{\angle} = 1$.

Then, on the second step, we have $\boldsymbol{E}_2^{\angle} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 0 & 0 \end{bmatrix}$ and

$$\boldsymbol{d}_2^{\angle} = \boldsymbol{E}_2^{\angle}\left(\boldsymbol{E}_2^{\angle\top}\boldsymbol{X}^{*\top}\boldsymbol{X}^{*}\boldsymbol{E}_2^{\angle}\right)^{-1}\boldsymbol{E}_2^{\angle\top}\boldsymbol{X}^{*\top}\boldsymbol{r}_1^{\angle} = \begin{bmatrix} \frac{13}{9} \\ -\frac{13}{9} \\ 0 \end{bmatrix}.$$

Let $\tilde{\boldsymbol{\beta}}_2^{*\angle}(\alpha) = \hat{\boldsymbol{\beta}}_1^{*\angle} + \alpha\boldsymbol{d}_2^{\angle} = \begin{bmatrix} \frac{13}{9}\alpha \\ -\frac{11}{18} - \frac{13}{9}\alpha \\ 0 \end{bmatrix}$ and

$$\tilde{\boldsymbol{r}}_2^{\angle}(\alpha) = \boldsymbol{y}^c - \boldsymbol{X}^*\tilde{\boldsymbol{\beta}}_2^{*\angle}(\alpha) = \boldsymbol{r}_1^{\angle} - \alpha\boldsymbol{X}^*\boldsymbol{d}_2^{\angle} = \begin{bmatrix} -\frac{11}{12} + \frac{13}{18}\alpha \\ \frac{61}{36} - \frac{13}{18}\alpha \\ -\frac{13}{18}\alpha \\ 1 - \frac{13}{18}\alpha \\ -\frac{25}{36} \\ -\frac{13}{12} + \frac{13}{9}\alpha \end{bmatrix}$$

for $\alpha \in [0, 1]$. Then we compute $\pm\langle \boldsymbol{x}_j^*, \tilde{\boldsymbol{r}}_2^{\angle}(\alpha)\rangle$ for $j = 1, 2, 3$. We have $\langle \boldsymbol{x}_1^*, \tilde{\boldsymbol{r}}_2^{\angle}(\alpha)\rangle = \frac{13}{9} - \frac{13}{9}\alpha$, $\langle \boldsymbol{x}_2^*, \tilde{\boldsymbol{r}}_2^{\angle}(\alpha)\rangle = -\frac{13}{9} + \frac{13}{9}\alpha$, and $\langle \boldsymbol{x}_3^*, \tilde{\boldsymbol{r}}_2^{\angle}(\alpha)\rangle = -\frac{1}{9} - \frac{13}{12}\alpha$. These line segments are plotted in Figure 2-2.

Then, we have

$$\langle \boldsymbol{x}_3^*, \tilde{\boldsymbol{r}}_2^{\angle}(\tilde{\alpha}_{23}^{+})\rangle = \langle \boldsymbol{x}_1^*, \tilde{\boldsymbol{r}}_2^{\angle}(\tilde{\alpha}_{23}^{+})\rangle \quad \Rightarrow \quad \tilde{\alpha}_{23}^{+} = \frac{\frac{13}{9} - \frac{1}{9}}{\frac{13}{9} + \frac{13}{12}} = \frac{48}{91} \approx .527$$

$$\langle -\boldsymbol{x}_3^*, \tilde{\boldsymbol{r}}_2^{\angle}(\tilde{\alpha}_{23}^{-})\rangle = \langle \boldsymbol{x}_1^*, \tilde{\boldsymbol{r}}_2^{\angle}(\tilde{\alpha}_{23}^{-})\rangle \quad \Rightarrow \quad \tilde{\alpha}_{23}^{-} = \frac{\frac{13}{9} + \frac{1}{9}}{\frac{13}{9} - \frac{13}{12}} = \frac{56}{13} \notin [0, 1).$$

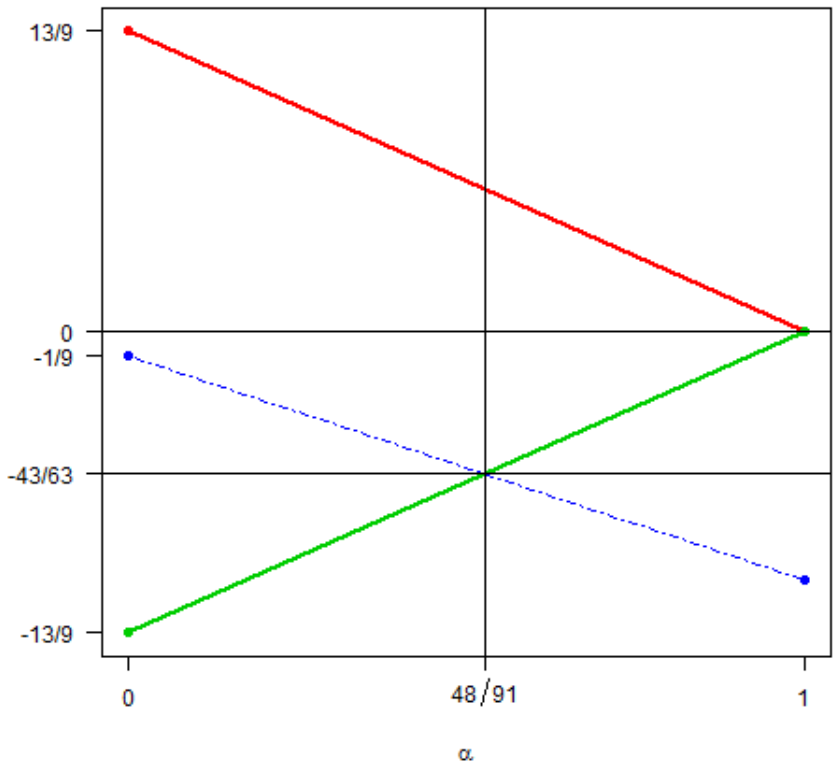

Figure 2.2 – Line segments $\langle \boldsymbol{x}_1^*, \tilde{\boldsymbol{r}}_2^{\angle}(\alpha) \rangle$ (red), $\langle \boldsymbol{x}_2^*, \tilde{\boldsymbol{r}}_2^{\angle}(\alpha) \rangle$ (green), and $\langle \boldsymbol{x}_3^*, \tilde{\boldsymbol{r}}_2^{\angle}(\alpha) \rangle$ (blue) for step 2 of the LAR algorithm.

Thus, we have $\hat{\alpha}_2^{\angle} = \frac{48}{91}$ so that $\hat{\boldsymbol{\beta}}_2^{*\angle} = \tilde{\boldsymbol{\beta}}_2^{*\angle}(\hat{\alpha}_2^{\angle}) = \begin{bmatrix} \frac{16}{21} \\ -\frac{173}{126} \\ 0 \end{bmatrix} \approx \begin{bmatrix} 0.762 \\ -1.373 \\ 0 \end{bmatrix}$,

$$\boldsymbol{r}_2^{\angle} = \tilde{\boldsymbol{r}}_2^{\angle}(\hat{\alpha}_2^{\angle}) = \begin{bmatrix} -\frac{45}{84} \\ \frac{331}{252} \\ -\frac{8}{21} \\ \frac{13}{21} \\ -\frac{25}{36} \\ -\frac{9}{28} \end{bmatrix},$$

and $\hat{j}_3^{\angle} = 3$.

Then $\boldsymbol{E}_3^{\angle} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ and $\boldsymbol{d}_3^{\angle} = \boldsymbol{E}_3^{\angle}\left(\boldsymbol{E}_3^{\angle\top}\boldsymbol{X}^{*\top}\boldsymbol{X}^{*}\boldsymbol{E}_3^{\angle}\right)^{-1}\boldsymbol{E}_3^{\angle\top}\boldsymbol{X}^{*\top}\boldsymbol{r}_2^{\angle} = \begin{bmatrix} \frac{10664}{14007} \\ -\frac{33497}{42021} \\ -\frac{172}{667} \end{bmatrix}$. Let $\tilde{\boldsymbol{\beta}}_3^{*\angle}(\alpha) = \hat{\boldsymbol{\beta}}_2^{*\angle} + \alpha\boldsymbol{d}_3^{\angle} = \begin{bmatrix} \frac{16}{21} + \frac{10664}{14007}\alpha \\ -\frac{173}{126} - \frac{33497}{42021}\alpha \\ -\frac{172}{667}\alpha \end{bmatrix}$ and

$$\tilde{\boldsymbol{r}}_3^{\angle}(\alpha) = \boldsymbol{y}^c - \boldsymbol{X}^*\tilde{\boldsymbol{\beta}}_3^{*\angle}(\alpha) = \boldsymbol{r}_2^{\angle} - \alpha\boldsymbol{X}^*\boldsymbol{d}_3^{\angle} = \begin{bmatrix} -\frac{45}{84} + \frac{3053}{9338}\alpha \\ \frac{331}{252} - \frac{22661}{84042}\alpha \\ -\frac{8}{21} + \frac{86}{14007}\alpha \\ \frac{13}{21} - \frac{10750}{14007}\alpha \\ -\frac{25}{36} + \frac{215}{12006}\alpha \\ -\frac{9}{28} + \frac{6407}{9338}\alpha \end{bmatrix}$$

for $\alpha \in [0, 1]$. Then we compute $\pm\langle \boldsymbol{x}_j^*, \tilde{\boldsymbol{r}}_3^{\angle}(\alpha)\rangle$ for $j = 1, 2, 3$. We have $\langle \boldsymbol{x}_1^*, \tilde{\boldsymbol{r}}_3^{\angle}(\alpha)\rangle = -\frac{43}{63} + \frac{43}{63}\alpha$, $\langle \boldsymbol{x}_2^*, \tilde{\boldsymbol{r}}_3^{\angle}(\alpha)\rangle = \frac{43}{63} - \frac{43}{63}\alpha$, and $\langle \boldsymbol{x}_3^*, \tilde{\boldsymbol{r}}_3^{\angle}(\alpha)\rangle = \frac{43}{63} - \frac{43}{63}\alpha$. These line segments are plotted in Figure 2-3. When $\alpha = 1$, we get the least squares estimate

$$\hat{\boldsymbol{\beta}}_3^{*\angle} = \begin{bmatrix} \frac{1016}{667} \\ -\frac{2895}{1334} \\ -\frac{172}{667} \end{bmatrix} \approx \begin{bmatrix} 1.523 \\ -2.170 \\ -0.258 \end{bmatrix}.$$

Since $\bar{y} = 6$, $\bar{x}_1 = 5$, $\bar{x}_2 = 4$, $\bar{x}_3 = 4$ and $s_{x_1} = s_{x_2} = s_{x_3} = 2$, the LAR coefficient profiles are

$$\hat{\beta}_{i,0}(\alpha) = \begin{cases} 6 & \text{if } i = 0 \\ 6 + \frac{9}{5}\alpha & \text{if } i = 1, 0 \le \alpha \le \frac{55}{81} \\ \frac{65}{9} - \frac{13}{18}\alpha & \text{if } i = 2, 0 \le \alpha \le \frac{48}{91} \\ \frac{431}{63} + \frac{8686}{42021}\alpha & \text{if } i = 3, 0 \le \alpha \le 1 \end{cases},$$

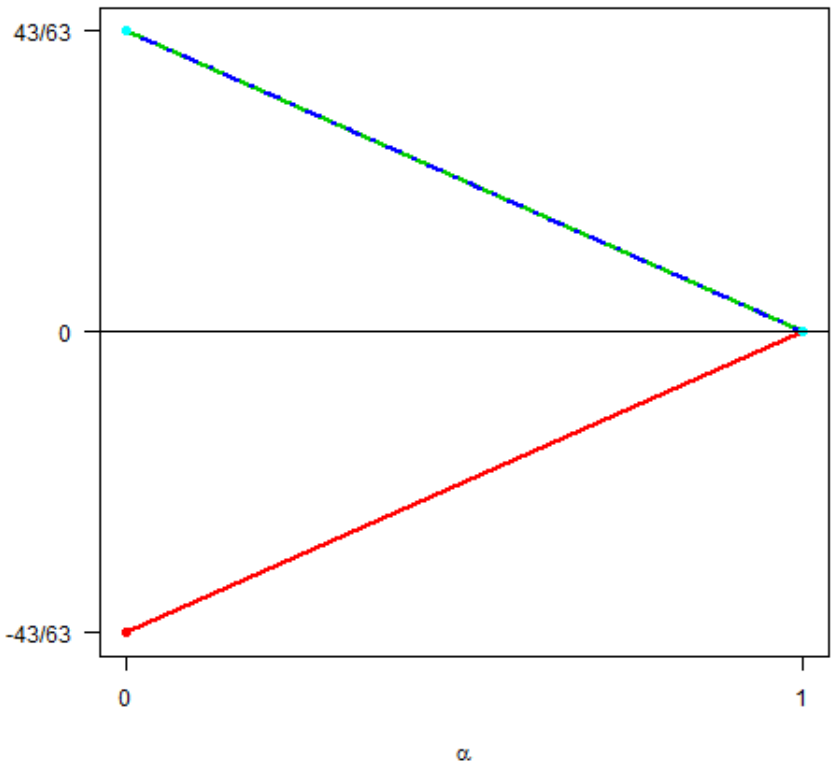

Figure 2.3 – Line segments $\langle \boldsymbol{x}_1^*, \tilde{\boldsymbol{r}}_3^{\angle}(\alpha) \rangle$ (red), $\langle \boldsymbol{x}_2^*, \tilde{\boldsymbol{r}}_3^{\angle}(\alpha) \rangle$ (green), and $\langle \boldsymbol{x}_3^*, \tilde{\boldsymbol{r}}_3^{\angle}(\alpha) \rangle$ (blue) for step 3 of the LAR algorithm.

$$\hat{\beta}_{i,1}(\alpha) = \begin{cases} 0 & \text{if } i \leq 1 \\ \frac{13}{18}\alpha & \text{if } i = 2, 0 \leq \alpha \leq \frac{48}{91} \\ \frac{8}{21} + \frac{5332}{14007}\alpha & \text{if } i = 3, 0 \leq \alpha \leq 1 \end{cases},$$

$$\hat{\beta}_{i,2}(\alpha) = \begin{cases} 0 & \text{if } i = 0 \\ -\frac{9}{20}\alpha & \text{if } i = 1, 0 \leq \alpha \leq \frac{55}{81} \\ -\frac{11}{36} - \frac{13}{18}\alpha & \text{if } i = 2, 0 \leq \alpha \leq \frac{48}{91} \\ -\frac{173}{252} - \frac{33497}{84042}\alpha & \text{if } i = 3, 0 \leq \alpha \leq 1 \end{cases},$$

and

$$\hat{\beta}_{i,3}(\alpha) = \begin{cases} 0 & \text{if } i \leq 2 \\ -\frac{86}{667}\alpha & \text{if } i = 3, 0 \leq \alpha \leq 1 \end{cases}.$$

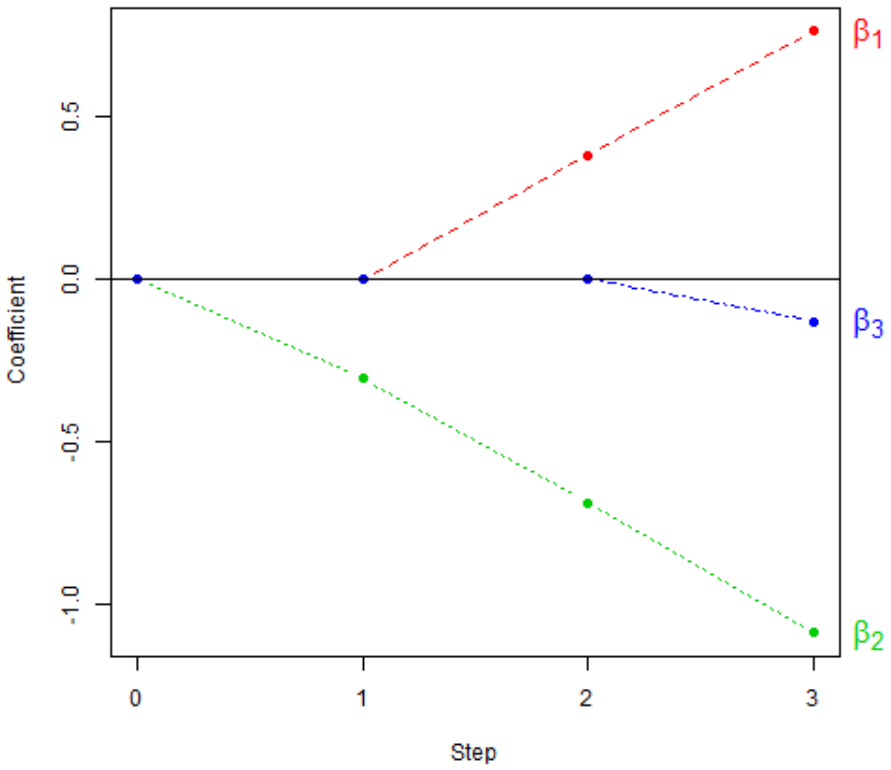

Figure 2.4 – Coefficient profiles for the artificial data example in Section 2.2.

The LAR coefficient profiles for β_1, β_2, and β_3 are illustrated in Figure 2.4.

2.3 Code

All of the computational work in the thesis was performed using the R statistical software environment (R Core Team, 2015). The following custom function `our.lar` implements the LAR algorithm as described in Section 2.1.

```
# LAR code from scratch
our.lar=function(X,y,epsilon=1e-8){
 n=nrow(X)
 p=ncol(X)

 #compute the mean and standard deviation of each column of X
```

```
 X.means=apply(X,2,mean)
 X.sds=apply(X,2,sd)

 #center and scale the columns of X
 Xstar=X
 for (i in 1:p)
  Xstar[,i]=(X[,i]-X.means[i])/X.sds[i]

 #center the y variable
 yc=y-mean(y)

 #step up matrix to store the parameters that will be returned
 #from the function
 beta.hat=rep(0,p+1)
 alpha.hat=NULL

 #initial step
 r=yc

 #compute inner product and choose the first variable to enter
 #the model
 inner.products=t(X)%*%r
 j.hat=which.max(abs(inner.products))

 #algorithm on the ith step
 i=1
 while ((i==1)||(alpha.hat[i-1]<1)){
  beta.hat=rbind(beta.hat,rep(0,p+1))
  alpha.hat=c(alpha.hat,1)
  njhat=length(j.hat)
  j.hat=c(j.hat,0)
  XE=Xstar[,j.hat]
  d=rep(0,p)
  d[j.hat]=solve(t(XE)%*%XE)%*%t(XE)%*%r
  Xd=Xstar%*%d
  #find alpha.hat[i]
  for (j in 1:p){
   alpha=1
   if (j%in%j.hat==FALSE){
    if (abs(sum(Xstar[,j.hat[1]]*r)-sum(Xstar[,j]*Xd))>epsilon){
     alpha=(sum(Xstar[,j.hat[1]]*r)-sum(Xstar[,j]*r))/
(sum(Xstar[,j.hat[1]]*r)-sum(Xstar[,j]*Xd))
     if ((alpha<epsilon)|(alpha>1-epsilon)){
      alpha=1
```

```
      if (abs(sum(Xstar[,j.hat[1]]*r)+sum(Xstar[,j]*Xd))>epsilon){
       alpha2=(sum(Xstar[,j.hat[1]]*r)+sum(Xstar[,j]*r))/
(sum(Xstar[,j.hat[1]]*r)+sum(Xstar[,j]*Xd))
      if ((alpha2>epsilon)&(alpha2<1-epsilon))
       alpha=alpha2
      }
     }
     if (alpha+epsilon<alpha.hat[i]){
      alpha.hat[i]=alpha
      j.hat[njhat+1]=j
     }
    }
   }
  }
  beta.hat[i+1,2:(p+1)]=beta.hat[i,2:(p+1)]+alpha.hat[i]*d
  r=r-alpha.hat[i]*Xd
  i=i+1
 }

 #translate coefficient estimates back to original scale
 beta.hat[,-1]=t(t(beta.hat[,-1])/X.sds)
 beta.hat[,1]=mean(y)-beta.hat[,-1]%*%X.means

 #output relevant results
 list(beta=beta.hat,alpha=alpha.hat)
}
```

Here is code that can be used to compute the LAR coefficient profiles for the artificial data example in Section 2.2.

```
X=rbind(
c(1,1,4),
c(5,3,5),
c(6,4,7),
c(6,4,1),
c(6,5,4),
c(6,7,3))
y=c(6,8,6,7,5,4)

print(our.lar(X,y)$beta,digits=4)
print(our.lar(X,y)$alpha,digits=4)
```

Here is the output for `print(our.lar(X,y)$beta,digits=4)`.

```
> print(our.lar(X,y)$beta,digits=4)
          [,1]   [,2]    [,3]    [,4]
beta.hat 6.000 0.0000  0.0000  0.0000
         7.222 0.0000 -0.3056  0.0000
         6.841 0.3810 -0.6865  0.0000
         7.048 0.7616 -1.0851 -0.1289
```

The rows give the values of $\hat{\boldsymbol{\beta}}_0^{\angle}$, $\hat{\boldsymbol{\beta}}_1^{\angle}$, $\hat{\boldsymbol{\beta}}_2^{\angle}$, and $\hat{\boldsymbol{\beta}}_3^{\angle}$, respectively. There is an excellent R package lars (Hastie and Efron, 2013) for implementing LAR, the LASSO, and forward stagewise regression. Using the package lars, the following command `coef(lars(X,y,type="lar"))` verifies that the custom function above obtains the same coefficient profile as the classic LAR algorithm.

The other output command `print(our.lar(X,y)$alpha,digits=4)` explicitly computes the values of $\hat{\alpha}_1^{\angle}$, $\hat{\alpha}_2^{\angle}$, and $\hat{\alpha}_3^{\angle}$.

```
> print(our.lar(X,y)$alpha,digits=4)
[1] 0.6790 0.5275 1.0000
```

2.4 Longley Example

As an example for Least Angle Regression consider the Longley data set which is studied extensively in Longley (1967). This data contained seven economical variables observed annually from 1947 to 1960. There are 6 explanatory variables $x_1, \ldots, x_6$; they are the GNP implicit price deflator(`GNP.deflator`), Gross National Product(`GNP`), number of people unemployed(`Unemployed`), number of people in the armed force(`Armed Force`), non institutionalized population greater than 14 years of age(`Population`), and the time(`Year`), respectively. The dependent variable y is the number of people employed(`Employed`). The data set is also available in the R data frame `longley`. The scale for the variables in R's built-in data set is different from the scale in the original paper by Longley (1967); herein, the scale in the R data set is used.

i (Step)	$\hat{\beta}^{*\angle}_{i,1}$	$\hat{\beta}^{*\angle}_{i,2}$	$\hat{\beta}^{*\angle}_{i,3}$	$\hat{\beta}^{*\angle}_{i,4}$	$\hat{\beta}^{*\angle}_{i,5}$	$\hat{\beta}^{*\angle}_{i,6}$
0	0	0	0	0	0	0
1	0	12.59954	0	0	0	0
2	0	13.94697	-1.347432	0	0	0
3	0	14.30726	-1.662917	-0.267191	0	0
4	0	-2.52538	-4.995279	-1.661989	0	19.67211
5	1.13	-15.87012	-7.548153	-2.780428	0	34.09000
6	0.63	-13.789	-7.312	-2.784	-1.377	33.728

Table 2.2 – LAR coefficient table for the standardized Longley data.

Coefficient tables obtained from the custom R function `our.lar` are given in Table 2.2 (for the standardized predictors and centered response) and in Table 2.3 for the variables all in the original scale. It is seen that `GNP` is most highly correlated with `Employed` since it is the first variable to enter the model. Then `Unemployed` enters the model next, followed by `Armed Force`, `Year`, and `GNP.deflator`. `Population` enters the model last and finally the least squares solution

$$\hat{y} = -13.789x_2 - 7.312x_3 - 2.784x_4 + 33.728x_6 + 0.63x_1 - 1.377x_5$$

is attained.

Figure 2.1 clearly shows the LAR coefficient profiles for the standardized Longley data.

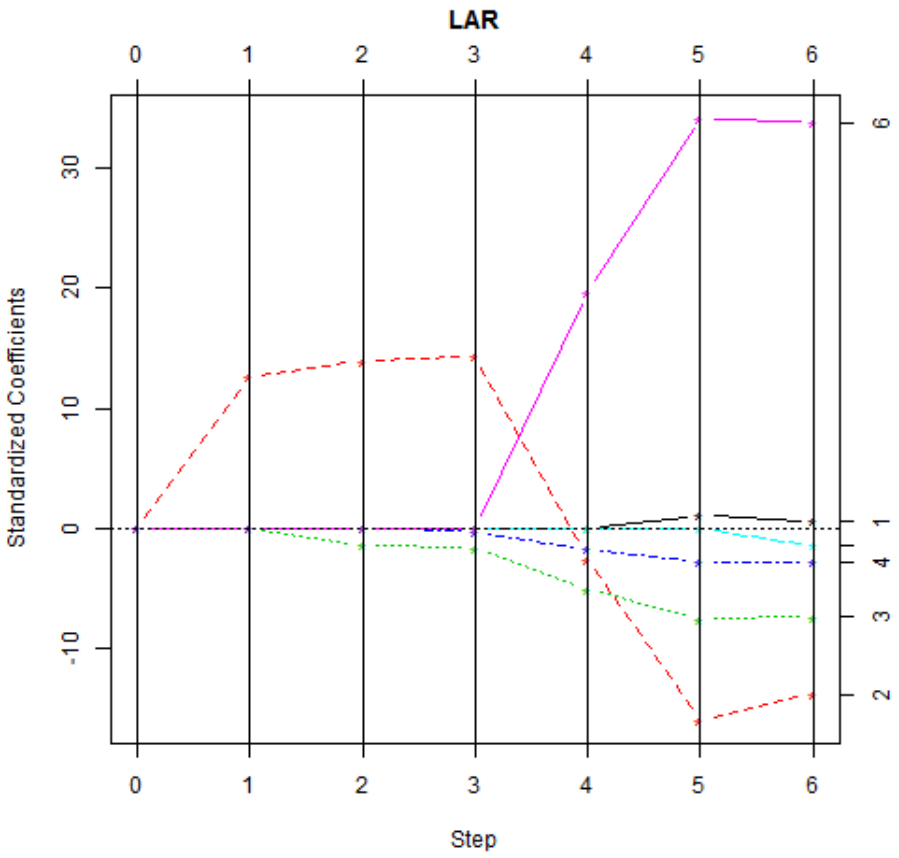

Figure 2.5 – LAR coefficient profiles for the standardized Longley data.

i (Step)	$\hat{\beta}^{\angle}_{i,0}$	$\hat{\beta}^{\angle}_{i,1}$	$\hat{\beta}^{\angle}_{i,2}$	$\hat{\beta}^{\angle}_{i,3}$	$\hat{\beta}^{\angle}_{i,4}$	$\hat{\beta}^{\angle}_{i,5}$	$\hat{\beta}^{\angle}_{i,6}$
0	65.32	0	0	0	0	0	0
1	52.63	0	0.03273	0	0	0	0
2	52.46	0	0.03623	-0.003723	0	0	0
3	52.63	0	0.03717	-0.004595	-0.0009913	0	0
4	-2011.32	0	-0.00656	-0.013802	-0.0061663	0	1.067
5	-3525.56	0.02701	-0.04123	-0.020856	-0.0103159	0	1.849
6	-3482.26	0.01506	-0.03582	-0.020202	-0.0103323	-0.0511	1.829

Table 2.3 – LAR coefficient table for the Longley data (original scale).

CHAPTER 3
PENALIZED REGRESSION VIA THE LASSO

Penalized regression methods estimate the regression coefficients by minimizing the Residual Sum of Squares(RSS) which is based on Ordinary Least Squares(OLS) as in LAR. However penalized regression methods use a penalty on the size of the regression coefficients. This penalty causes the regression coefficients to shrink towards zero. Penalized regression methods include sequence of models each associated with specific values for one or more tuning parameters. Some versions of penalized regression keep all the predictors in the model; for example, ridge regression coefficients minimize the RSS

$$\bar{\beta} = \arg\min_{\beta} \sum_{i=1}^{N} \left(y_i^c - \sum_{j=1}^{p} x_{ij}^* \beta_j^* \right)^2 \quad \text{subject to} \quad \sum_{j=1}^{p} \beta_j^{*2} \leq t$$

for some non-negative real number t.

Another method for penalized regression is the Least Absolute Shrinkage and Selection Operator(LASSO). The LASSO is a constrained version of OLS which minimizes the RSS subject to a constraint on the sum of absolute value of the regression coefficients. There is an important difference in LASSO with Ridge Regression. In Ridge Regression the L_2 ridge penalty, $\sum_{j=1}^{p} \beta_j^{*2}$ is replaced by the L_1 LASSO penalty, $\sum_{j=1}^{p} \left|\beta_j^*\right|$. So the LASSO constraint makes the solutions non linear. Making t sufficiently small will cause some of the coefficients to be zero. Often, the coefficient profiles for the LASSO are written as functions of the standardized tuning parameter $s = \dfrac{t}{\sum_{j=1}^{p} |\hat{\beta}_j^*|}$.

Figure 3.1 illustrates the LASSO constraint for the artificial example using

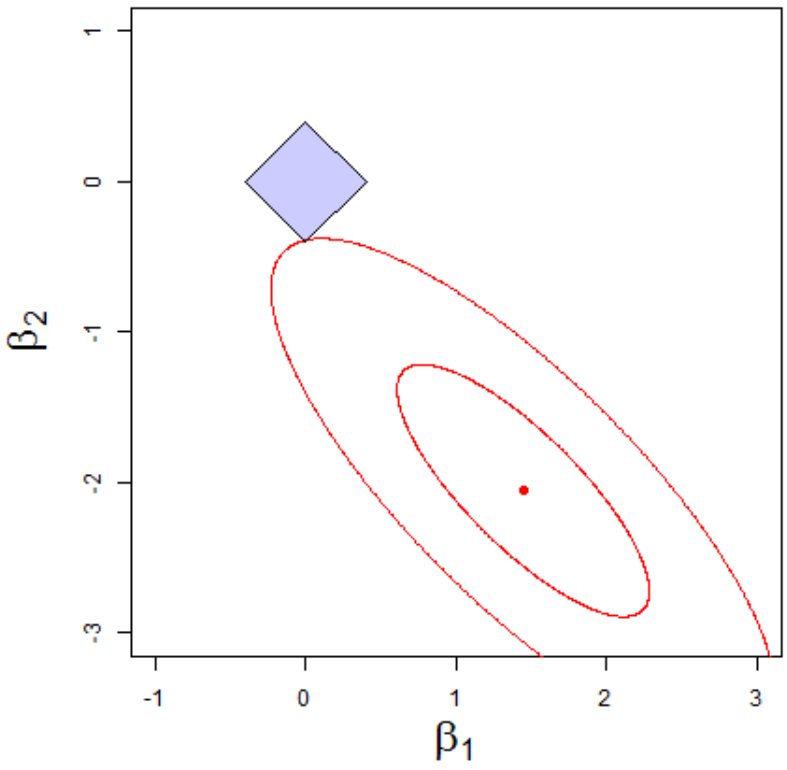

Figure 3.1 – Contour plot of $\|\boldsymbol{y}^c - \boldsymbol{X}^*\boldsymbol{\beta}^*\|^2$ and LASSO constraint $\sum_{j=1}^{2} |\beta_j^*| \leq 0.4$ for the artificial example with $\boldsymbol{X}^* = [\boldsymbol{x}_1^* \;\; \boldsymbol{x}_2^*]$.

only $\boldsymbol{x}_1^*$ and $\boldsymbol{x}_2^*$ as explanatory variables with $t = .4$. The solid blue diamond $\sum_{j=1}^{2} |\beta_j^*| \leq 0.4$ gives the set of values of β_1^* and β_2^* which are permitted under the LASSO constraint. The point at the center of the contour plot represents the least squares estimates of β_1^* and β_2^* based on the regression model of $\boldsymbol{y}$ on $\boldsymbol{x}_1^*$ and $\boldsymbol{x}_2^*$. The red ellipses depicted in Figure 3.1 show level curves for the sum of squares function $\|\boldsymbol{y}^c - \boldsymbol{X}^*\boldsymbol{\beta}^*\|^2$; the further the ellipse is from the center of the contour plot, the larger the sum of squares function. Thus, it is seen from the contour plot that the constrained minimum of $\|\boldsymbol{y}^c - \boldsymbol{X}^*\boldsymbol{\beta}^*\|^2$ is at the corner of the diamond where $\beta_1^* = 0$. Hence, for $t = .4$, the first variable $\boldsymbol{x}_1^*$ is not included in the model.

Naive computation of the LASSO is very computationally expensive but a

simple modification of LAR algorithm gives a computationally efficient algorithm for computing the LASSO estimates. The main modification to the LAR algorithm is that if a non-zero coefficient hits zero, its variable must be dropped from the active set of variables and the current joint least squares direction should be recomputed. Thus, in the LASSO algorithm, variables can leave the model and possibly re-enter later multiple times. Hence it may take more than p steps to reach the full model, if $n-1 > p$, whereas in the LAR algorithm, variables added to the model are never removed, hence it will reach the full least squares solution using all variables in p steps or less.

3.1 LASSO Algorithm via LAR Modification

The LAR algorithm with a minor modification provides an efficient algorithm for computing the LASSO coefficient profiles. On the ith step, the modification requires that none of the coefficient profiles cross 0. This is equivalent to considering other candidates for $\hat{\alpha}_i^\circ$ that correspond to values of α such that $\hat{\boldsymbol{\beta}}_{i-1}^{*\circ} + \alpha \boldsymbol{d}_i^\circ = \mathbf{0}$. If $\hat{\beta}_{i-1,j} \neq 0$, then let

$$\tilde{\alpha}_{i,j}^* = -\hat{\beta}_{i-1,j}^{*\circ} / d_{i,j}^\circ.$$

Then, α is selected using the modified formula

$$\hat{\alpha}_i^\circ = \min \Big\{ \alpha \in [0,1] : \Big(\alpha = \tilde{\alpha}_{i,j}^+ \text{ or } \alpha = \tilde{\alpha}_{i,j}^- \text{ for some } j \text{ such that } \hat{\beta}_{i-1,j}^{*\circ} = 0 \Big) \text{ or } \Big(\alpha = \tilde{\alpha}_{i,j}^* \text{ for some } j \text{ such that } \hat{\beta}_{i-1,j}^{*\circ} \neq 0 \Big) \Big\}.$$

Finally, the other modification is made if $\hat{\alpha}_i^\circ = \tilde{\alpha}_{i,j}^*$ for some j such that $\hat{\beta}_{i-1,j}^{*\circ} \neq 0$; in this case, $\boldsymbol{E}_i^\circ$ is the matrix formed by removing the column $\boldsymbol{e}_j$ from $\boldsymbol{E}_{i-1}^\circ$. Complete details for this LASSO algorithm are provided in Table 3-1.

initial	$\boldsymbol{r}_0^\circ = \boldsymbol{y}^c$
step	$\hat{\boldsymbol{\beta}}_0^{*\circ} = [\hat{\beta}_{0,1}^{*\circ}, \ldots, \hat{\beta}_{0,p}^{*\circ}]^\top = \mathbf{0}$
	$\hat{j}_1^\circ = \underset{j}{\operatorname{argmax}} \left\lvert \langle \boldsymbol{x}_j^*, \boldsymbol{r}_0^\circ \rangle \right\rvert$
	$\boldsymbol{E}_1^\circ = \boldsymbol{e}_{\hat{j}_1^\circ}$
ith	$\boldsymbol{d}_i^\circ = \boldsymbol{E}_i^\circ \left(\boldsymbol{E}_i^{\circ\top} \boldsymbol{X}^{*\top} \boldsymbol{X}^* \boldsymbol{E}_i^\circ \right)^{-1} \boldsymbol{E}_i^{\circ\top} \boldsymbol{X}^{*\top} \boldsymbol{r}_{i-1}^\circ$
step	$\tilde{\alpha}_{i,j}^{\pm} = \frac{\langle \boldsymbol{r}_{i-1}^\circ, \boldsymbol{x}_{\hat{j}_i^\circ}^* \rangle \mp \langle \boldsymbol{r}_{i-1}^\circ, \boldsymbol{x}_j^* \rangle}{\langle \boldsymbol{r}_{i-1}^\circ, \boldsymbol{x}_{\hat{j}_i^\circ}^* \rangle \mp \langle \boldsymbol{X}^* \boldsymbol{d}_i^\circ, \boldsymbol{x}_j^* \rangle}$ for j such that $\hat{\beta}_{i-1,j}^{*\circ} = 0$
	$\tilde{\alpha}_{i,j}^* = -\hat{\beta}_{i-1,j}^{*\circ} / d_{i,j}^\circ$ for j such that $\hat{\beta}_{i-1,j}^{*\circ} \neq 0$
	$\hat{\alpha}_i^\circ = \min \left\{ \alpha \in [0,1] : \alpha = \tilde{\alpha}_{i,j}^+, \alpha = \tilde{\alpha}_{i,j}^-, \text{ or } \alpha = \tilde{\alpha}_{i,j}^* \right\}$
	$\hat{\boldsymbol{\beta}}_i^{*\circ}(\alpha) = \hat{\boldsymbol{\beta}}_{i-1}^{*\circ} + \alpha \boldsymbol{d}_i^\circ, \alpha \in [0, \hat{\alpha}_i^\circ]$
	$\boldsymbol{r}_i^\circ = \boldsymbol{r}_{i-1}^\circ - \hat{\alpha}_i^\circ \boldsymbol{X}^* \boldsymbol{d}_i^\circ$
	If $\tilde{\alpha}_{i,j}^+ = \hat{\alpha}_i^\circ$ or $\tilde{\alpha}_{i,j}^- = \hat{\alpha}_i^\circ$ for some j, then $\boldsymbol{E}_{i+1}^\circ = [\boldsymbol{E}_i^\circ \;\; \boldsymbol{e}_j]$.
	If $\tilde{\alpha}_{i,j}^* = \hat{\alpha}_i^\circ$ for some j, then $\boldsymbol{E}_{i+1}^\circ$ is $\boldsymbol{E}_i^\circ$ with $\boldsymbol{e}_j$ removed.
	Continue until $\hat{\alpha}_i^\circ = 1$.
original	$\hat{\beta}_{i,j}^\circ(\alpha) = \frac{\hat{\beta}_{i,j}^{*\circ}(\alpha)}{s_j}$ for all i and for $j = 1, \ldots, p$
scale	$\hat{\beta}_{i,0}^\circ(\alpha) = \bar{y} - \frac{\hat{\beta}_{i,1}^{*\circ}(\alpha)}{s_1} \bar{x}_1 - \ldots - \frac{\hat{\beta}_{i,p}^{*\circ}(\alpha)}{s_p} \bar{x}_p$ for all i

Table 3.1 – Modified LAR algorithm to obtain the coefficient profiles based on the LASSO method.

3.2 Code

The following custom function `our.lasso` implements the LASSO algorithm discussed in the previous section.

```
# LASSO code from scratch
our.lasso=function(X,y,epsilon=1e-8,max.steps=20){
 n=nrow(X)
 p=ncol(X)

 #compute the mean and standard deviation of each column of X
 X.means=apply(X,2,mean)
 X.sds=apply(X,2,sd)

 #center and scale the columns of X
 Xstar=X
 for (i in 1:p)
  Xstar[,i]=(X[,i]-X.means[i])/X.sds[i]

 #center the y variable
 yc=y-mean(y)

 #step up matrix to store the parameters that will be returned
 #from the function
 beta.hat=rep(0,p+1)
 alpha.hat=NULL

 #initial step
 r=yc

 #compute inner product and choose the first variable to enter
 #the model
 inner.products=t(X)%*%r
 j.hat=which.max(abs(inner.products))

 #algorithm on the ith step
```

```
 i=1
 while (((i==1)||(alpha.hat[i-1]<1))&(i<max.steps)){
  beta.hat=rbind(beta.hat,rep(0,p+1))
  alpha.hat=c(alpha.hat,1)
  njhat=length(j.hat)
  j.hat=c(j.hat,0)
  XE=Xstar[,j.hat]
  d=rep(0,p)
  d[j.hat]=solve(t(XE)%*%XE)%*%t(XE)%*%r
  Xd=Xstar%*%d
  #find alpha.hat[i]
  for (j in 1:p){
   alpha=1
   if (j%in%j.hat==FALSE){
    if (abs(sum(Xstar[,j.hat[1]]*r)-sum(Xstar[,j]*Xd))>epsilon){
     alpha=(sum(Xstar[,j.hat[1]]*r)-sum(Xstar[,j]*r))/
(sum(Xstar[,j.hat[1]]*r)-sum(Xstar[,j]*Xd))
     if ((alpha<epsilon)|(alpha>1-epsilon)){
      alpha=1
      if (abs(sum(Xstar[,j.hat[1]]*r)+sum(Xstar[,j]*Xd))>epsilon){
       alpha2=(sum(Xstar[,j.hat[1]]*r)+sum(Xstar[,j]*r))/
(sum(Xstar[,j.hat[1]]*r)+sum(Xstar[,j]*Xd))
      if ((alpha2>0)&(alpha2<1))
       alpha=alpha2
      }
     }
     if (alpha+epsilon<alpha.hat[i]){
      alpha.hat[i]=alpha
      j.hat[njhat+1]=j
     }
    }
   }
   else{
   #LASSO modification
    if (d[j]!=0){
     alpha=-beta.hat[i,j+1]/d[j]
     if ((alpha>0)&(alpha<alpha.hat[i])){
      alpha.hat[i]=alpha
      j.hat[njhat+1]=-j
     }
    }
   }
  }
  if (j.hat[njhat+1]<0){
```

```
   remove.j=-j.hat[njhat+1]
   j.hat=j.hat[abs(j.hat)!=remove.j]
  }
  beta.hat[i+1,2:(p+1)]=beta.hat[i,2:(p+1)]+alpha.hat[i]*d
  r=r-alpha.hat[i]*Xd
  i=i+1
 }

 #translate coefficient estimates back to original scale
 beta.hat[,-1]=t(t(beta.hat[,-1])/X.sds)
 beta.hat[,1]=mean(y)-beta.hat[,-1]%*%X.means

 #output relevant results
 list(beta=beta.hat,alpha=alpha.hat)
}
```

The LASSO coefficient profiles and the α values can be extracted from the output of `our.lasso` the same way as the LAR coefficient profiles and the α values were extracted from the output of `our.lar`.

3.3 Longley Example

Now, the LASSO method is illustrated on the Longley data set that was described in Section 2.4. The coefficient tables obtained from the custom R function `our.lasso` are given in Table 3-2 (for the standardized predictors and centered response) and Table 3-3 for the variables all in the original scale.

Figure 3.1 clearly shows the LASSO coefficient profiles for the standardized Longley data. When using the LASSO, it is preferable to parameterize the coefficient profiles by s instead of the index i for the step and the $\hat{\alpha}_i^{\angle}$ that was used for the LAR algorithm.

The LASSO coefficient profiles are the same as the LAR coefficient profiles through step 3. On step 4, the LAR path for crosses 0. This is allowed for the LAR algorithm, but it causes `GNP` to be dropped from the model when its path hits 0. From that point on, the direction for the coefficients in the LASSO differs from the

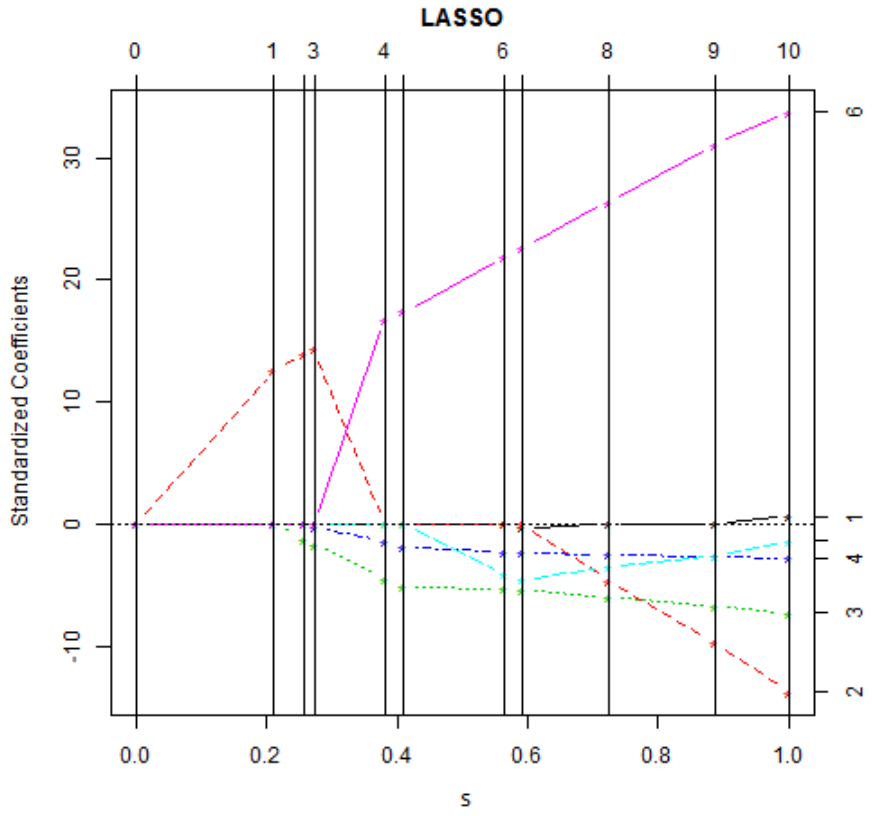

Figure 3.2 – LASSO coefficient profiles for the standardized Longley data.

direction for LAR. Eventually, at beginning of step 8, `GNP` re-enters the model, but now with a negative coefficient. At the end of step 8, the path for `GNP.deflator` hits 0, so it is dropped from the model; eventually it returns on the last step with the opposite sign for the coefficient.

i (Step)	$\hat{\beta}^{*\circ}_{i,1}$	$\hat{\beta}^{*\circ}_{i,2}$	$\hat{\beta}^{*\circ}_{i,3}$	$\hat{\beta}^{*\circ}_{i,4}$	$\hat{\beta}^{*\circ}_{i,5}$	$\hat{\beta}^{*\circ}_{i,6}$
0	0	0	0	0	0	0
1	0	12.599536	0	0	0	0
2	0	13.946968	-1.347432	0	0	0
3	0	14.307258	-1.662917	-0.267191	0	0
4	0	0	-4.495328	-1.452729	0	16.72073
5	0	0	-5.109205	-1.921372	0	17.40197
6	0	0	-5.324667	-2.321762	-4.132012	21.83734
7	-0.3217731	0	-5.359592	-2.352200	-4.600470	22.65942
8	0	-4.664033	-6.019837	-2.498540	-3.510064	26.40331
9	0	-9.753905	-6.764499	-2.665480	-2.563231	31.09839
10	0.6295186	-13.788770	-7.311544	-2.784841	-1.376789	33.72789

Table 3.2 – LASSO coefficient table for the standardized Longley data.

i (Step)	$\hat{\beta}^{\circ}_{i,0}$	$\hat{\beta}^{\circ}_{i,1}$	$\hat{\beta}^{\circ}_{i,2}$	$\hat{\beta}^{\circ}_{i,3}$	$\hat{\beta}^{\circ}_{i,4}$	$\hat{\beta}^{\circ}_{i,5}$	$\hat{\beta}^{\circ}_{i,6}$
0	65.32	0	0	0	0	0	0
1	52.63	0	0.03273	0	0	0	0
2	52.46	0	0.03623	-0.003723	0	0	0
3	52.63	0	0.03717	-0.004595	-0.0009913	0	0
4	-1701.67	0	0	-0.012421	-0.0053899	0	0.9068
5	-1772.89	0	0	-0.014117	-0.0071286	0	0.9438
6	-2224.44	0	0	-0.014712	-0.0086142	-0.15337	1.1843
7	-2308.69	-0.007699	0	-0.014809	-0.0087271	-0.17076	1.2289
8	-2705.65	0	-0.01212	-0.016633	-0.0092700	-0.13029	1.4319
9	-3201.50	0	-0.02534	-0.018691	-0.0098894	-0.09514	1.6865
10	-3482.26	0.015062	-0.03582	-0.020202	-0.0103323	-0.05110	1.8292

Table 3.3 – LASSO coefficient table for the Longley data (original scale).

CHAPTER 4
SELECTION OF CONSTRAINT FOR THE LASSO

Selection of the constraint t in the LASSO plays an important role since it controls the amount of regularization. One approach in such circumstances is to use a cross validation method to find the optimal value. Choosing the constraint depends on how many variables are included in the model, or equivalently how many coefficients are shrunk towards zero. Therefore each value corresponds to a model selection. There are a few kinds of cross validation methods (see, for instance, Chapter 7 of Hastie, Tibshirani, and Friedman (2013)). Herein a method called K-fold cross validation is considered. In this method the data set is randomly partitioned into K equal (or approximately equal) size parts. Then the method leaves out one part as a test data set and fits the model based on the other $K-1$ parts combined together. The fitted model based on $K-1$ parts (the training data) is used to obtain predictions for the left out part (test data), and the prediction error is recorded for each observation in the part that was left out. This process is repeated using each of the K parts, and thus the prediction error is obtained for all observations in the data set. Finally, there are different approaches for selecting the final model based on the average prediction error for each candidate model. While it is natural to choose the model which minimizes the average prediction error, some instead choose the model by visually identifying the "elbow" of the curve representing average prediction error as a function of the complexity of the model.

4.1 Description of K-Fold Cross-Validation for the LASSO

First, the labels for the observations are randomly permuted to obtain a design matrix $\breve{\boldsymbol{X}}$ and response vector $\breve{\boldsymbol{y}}$. The rows of $\breve{\boldsymbol{X}}$ and $\breve{\boldsymbol{y}}$ are randomly partitioned into K parts so that

$$\breve{\boldsymbol{X}} = \begin{bmatrix} \breve{\boldsymbol{X}}_1 \\ \breve{\boldsymbol{X}}_2 \\ \vdots \\ \breve{\boldsymbol{X}}_K \end{bmatrix} \text{ and } \breve{\boldsymbol{y}} = \begin{bmatrix} \breve{\boldsymbol{y}}_1 \\ \breve{\boldsymbol{y}}_2 \\ \vdots \\ \breve{\boldsymbol{y}}_K \end{bmatrix}$$

where $\breve{\boldsymbol{X}}_k$ is an $n_k \times (p+1)$ matrix and $\breve{\boldsymbol{y}}_k$ is a n_k dimensional vector for $k = 1, \ldots, K$. Usually, $n_1, \ldots, n_K$ are chosen to be approximately equal. Then, for each k, the method proceeds to use the LASSO to estimate a coefficient profile denoted by $\hat{\boldsymbol{\beta}}^{\circ}_{-k}(s)$ based on the design matrix $\breve{\boldsymbol{X}}^{(-k)} = \begin{bmatrix} \breve{\boldsymbol{X}}_1 \\ \vdots \\ \breve{\boldsymbol{X}}_{k-1} \\ \breve{\boldsymbol{X}}_{k+1} \\ \vdots \\ \breve{\boldsymbol{X}}_K \end{bmatrix}$ and response vector $\breve{\boldsymbol{y}}^{(-k)} = \begin{bmatrix} \breve{\boldsymbol{y}}_1 \\ \vdots \\ \breve{\boldsymbol{y}}_{k-1} \\ \breve{\boldsymbol{y}}_{k+1} \\ \vdots \\ \breve{\boldsymbol{y}}_K \end{bmatrix}$ to predict $\breve{\boldsymbol{y}}_k$ using the formula $\widehat{\breve{\boldsymbol{y}}}_k = \breve{\boldsymbol{X}}_k \hat{\boldsymbol{\beta}}^{\circ}_{-k}(s)$. The K-fold cross-validation mean square error function for a LASSO model can be expressed as

$$\mathrm{CV}(s) = \frac{1}{n} \sum_{k=1}^{K} \|\breve{\boldsymbol{y}}_k - \breve{\boldsymbol{X}}_k \hat{\boldsymbol{\beta}}^{\circ}_{-k}(s)\|^2,$$

and the smallest CV can be found by minimizing $\mathrm{CV}(s)$ for s in $[0, 1]$.

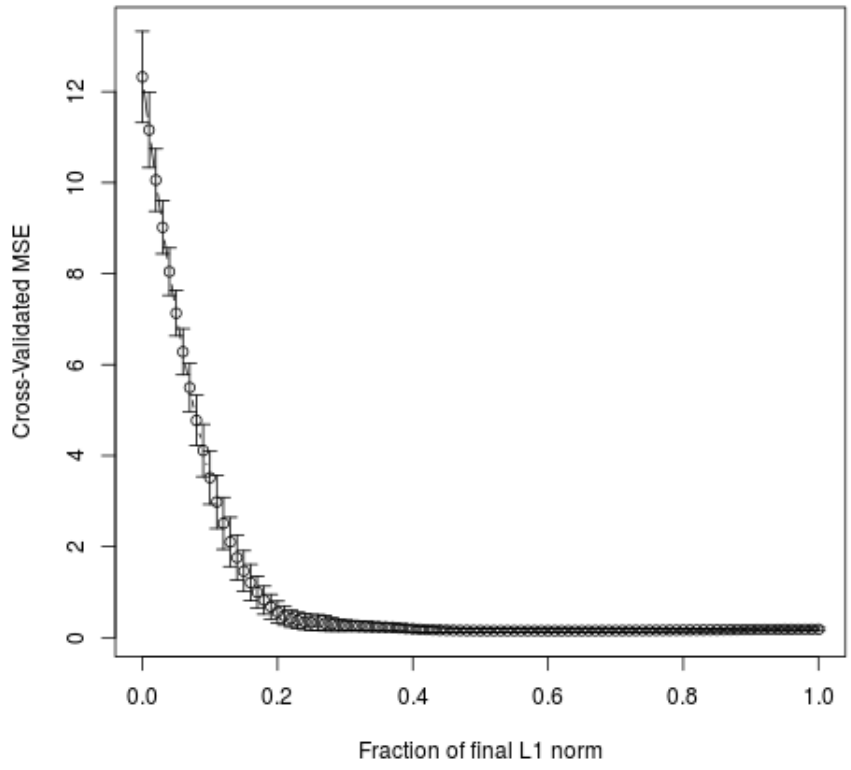

Figure 4.1 – 5-fold cross validation for the LASSO with the Longley data.

4.2 Longley Example

The built-in function `cv.lars` in the lars package can be used to obtain the cross-validation function. The following R commands can be used to compute the function $CV(s)$ with $K = 5$ and obtain the plot shown in Figure 4.1.

```
set.seed(32245)
cv.lasso.model=cv.lars(X,y,K=5,type="lasso",index=seq(0,1,by=.01))
```

In the above code, the random seed `32245` is useful to obtain reproducable results based on the random partitioning of the observation into $K = 5$ parts. The argument `index=seq(0,1,by=.01)` sets up a grid of values on which $CV(s)$ is computed.

The minimum value s of the cross-validation function can be obtained with the following R commands.

```
w=which.min(cv.lasso.model$cv)
s=cv.lasso.model$index[w]
s
```

This code outputs the value $s = 0.59$, though if one wants to use the "elbow" estimate to obtain a result with lower complexity, a value near 0.2 should be used. Finally, the vector of coefficients can be obtained with the following R code.

```
lasso.model=lars(X,y,type="lasso")
b=coef(lasso.model,s=s,mode="fraction")
b
intercept=mean(y)-sum(apply(X,2,mean)*b)
intercept
```

This code produces the fitted model

$$\hat{y} = -2302.76 - 0.00716x_1 - 0.01480x_3 - 0.00872x_4 - 0.16954x_5 + 1.22574x_6.$$

CHAPTER 5
CONCLUSION

LASSO is a recently developed well-known variable selection method in statistics. It is a regression analysis method that performs at the same time both variable selection and regularization. The purpose of this thesis has been to study Least Angle Regression in full detail and subsequently study a computationally efficient method for obtaining the LASSO coefficient estimates. Rather than giving the brief compact version of the LAR algorithm, I described it with full mathematical details which is easy to follow and understand. Furthermore, the algorithm is illustrated with the help of an artificial small example and a famous Longley data set including all necessary steps fully described.

With a small modification of the LAR algorithm, the LASSO is obtained and illustrated with the same two examples. Using my own R codes, both methods are implemented, and a comparison of the LAR and LASSO coefficient profiles are made with graphs and coefficient tables using the custom R code and the lars package.

To use the LASSO to estimate the regression coefficients, a point on the coefficient paths must be selected. That is, we must select a value for the penalty, or equivalently, the shrinkage factor. K-fold cross validation is a method which can be used to accomplish this task, and an example of selcting the shrinkage factor s using 5-fold cross validation for the Longley data set was presented.

REFERENCES

[1] Efron, B., Hastie, T., Johnstone, I., Tibshirani, R. (2004). Least angle regression. *The Annals of Statistics*, **32**, 407–499.

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

[3] Hastie, T., Tibshirani, R., and Friedman, J. (2013). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, second edition, 10th printing. Springer-Verlag.

[4] Hocking, R. R. (2013). *Methods and Applications of Linear Models: Regression and the Analysis of Variance*, third edition. Hoboken, NJ: Wiley.

[5] Longley, J. W. (1967). An appraisal of least squares programs for the electronic computer from the point of view of the user. *Journal of the American Statistical Association*, **62**, 819–841.

[6] R Core Team (2015). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/

CURRICULUM VITAE

Sandamala Hettigoda

Education

- B.Sc. in Physical Sciences, 1999, University of Kelaniya, Sri Lanka.

Teaching

- Graduate Teaching Assistant, Dept.of Mathematics, University of Louisville, USA, Jan2016 - May2016
- Math Tutor, REACH, University of Louisville, USA, 2015 Aug -Jan 2016
- Mathematics Teacher, Carey College Colombo, Sri Lanka, 2000 – 2004
- Demonstrator, Dept. of Statistics and Computing, University of Kelaniya, Sri Lanka, 1999 – 2000.

Certification

- Level I Tutor Training Certificate, REACH, University of Louisville, USA in 2015.
- Diploma in Computer Science, IDM, Sri Lanka in 1999

Research Experience

- Research Assistant Department of Meteorology, Sri Lanka, 2004 – 2007