Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Mathematical Sciences

Fanny Øverbø Næss

Exact inference conditioned on the selection event

Master's thesis in Applied Physics and Mathematics
Supervisor: Øyvind Bakke
June 2024

NTNU
Norwegian University of
Science and Technology

Fanny Øverbø Næss

Exact inference conditioned on the selection event

Master's thesis in Applied Physics and Mathematics
Supervisor: Øyvind Bakke
June 2024

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Mathematical Sciences

NTNU
Norwegian University of
Science and Technology

Preface

This thesis concludes the course TMA4900, and completes my M. Sc. degree in Applied Physics and Mathematics with specialization in Industrial Mathematics. The work for this thesis was carried out at the Department of Mathematical Sciences during the spring semester of 2024.

I would like to direct a huge thanks to Øyvind Bakke for excellent supervision and clear guidance throughout the process of writing this thesis. I extend my thanks to Mette Langaas for illuminating the topic of selective inference in the course MA8701, which led me to the topic of this thesis.

Lastly, I would like to express my gratitude to Mathias Dåsvand for valuable and insightful academic discussions throughout the last years.

Fanny Øverbø Næss
Trondheim, Norway
June, 2024

Abstract

Classical statistical inference tools rely on the assumption that the models and hypotheses to be tested are specified prior to data exploration. It is common practice to choose a model by inclusion of the variables that are observed to have a strong association with the response variable. In order to perform valid inference after model selection has been carried out on the same dataset, the calculation of p-values and confidence intervals must be adjusted in order to account for the stochastic aspect of the events leading to the selection of the particular model.

In this thesis we explore a framework for post-selection inference based on conditioning on polyhedral selection events. This approach allows us to use the same dataset for model selection and corresponding inferences. The method is in closed form, and yields exact p-values and confidence intervals in the case of Gaussian errors. We introduce the necessary theoretical foundation for the polyhedral inference method, and derive selection adjusted p-values and confidence intervals for coefficients in the multiple linear regression model.

A central criterion of the polyhedral method is that the model selection procedure that has been carried out can be formulated in its entirety as a polyhedral statistical event. Forward stepwise selection with a fixed number of steps and the lasso, when used as a model selector with fixed λ, fulfill this criteria. We derive the general schemes for construction of polyhedral selection events for forward selection and the lasso. In order to clarify the applications of the polyhedral method for inference after model selection by these selection procedures, we implement the methods in R and present examples of the resulting selection adjusted confidence intervals. We expand the method by omitting the conditioning on the observed sign pattern, resulting in shorter confidence intervals with the same coverage probability.

Sammendrag

Klassiske statistiske verktøy for inferens bygger på antagelsen om at modellene som tilpasses og hypotesene som testes er forhåndsspesifiserte. I praksis er det vanlig at en modell velges ved å inkludere de forklaringsvariablene som har en observert sterk sammenheng med responsvariabelen. For å utføre gyldig inferens etter modellseleksjon på samme datasett, er det nødvendig å tilpasse utregningen av p-verdier og konfidensintervaller slik at det stokastiske aspektet av hendelsene som har ført til valget av den gitte modellen tas hensyn til.

I denne oppgaven utforsker vi et rammeverk for gyldig inferens etter modellseleksjon basert på betinging på polyhedrale seleksjonshendelser. Denne tilnærmingen tillater bruk av det samme datasettet til modellseleksjon og tilhørende inferens. Metoden er på lukket form, og gir eksakte p-verdier og konfidensintervaller ved normalfordelte residualer. Vi introduserer det nødvendige teoretiske grunnlaget for å bruke polyedermetoden, og utleder seleksjonsjusterte p-verdier og konfidensintervaller for regresjonskoeffisienter i lineære modeller.

En nødvendig forutsetning for polyedermetoden er at modellseleksjonsprosedyren som har blitt utført kan formuleres i sin helhet som en statistisk hendelse på polyederform. Variabelseleksjonsmetodene forlengs stegvis seleksjon med fiksert antall steg og lasso med fiksert λ oppfyller dette kriteriet. Vi utleder generelle metoder for konstruksjon av polyhedrale seleksjonshendelser for forlengs seleksjon og lasso. For å klargjøre anvendelsen av polyedermetoden for gyldig inferens etter seleksjon ved disse metodene implementerer vi egen kode i R og presenterer eksempler på resulterende seleksjonsjusterte konfidensintervaller. Vi generaliserer også metoden ved å ekskludere betinging på observert fortegnsmønster, noe som resulterer i kortere konfidensintervaller med samme dekningssannsynlighet.

Contents

Figures

Chapter 1

Introduction

Classical statistical theory assumes models are specified prior to data exploration and inference, which is often not the case in the practice of modern statistics. To address this, several post-selective inference frameworks have been researched in recent years, with the goal of providing valid inferences after data-driven model selection. In this thesis we focus on an approach to post-selection inference that conditions on the model choice, which results in exact inferences after adaptive model selection.

The topic of post-selection inference is linked to Goal 3 of the United Nations Sustainable Development Goals (SDGs), which aims to ensure healthy lives and promote well-being for people of all ages (United Nations 2015). In medical research, statistical inference is important when the question of interest is the identity of the features, rather than prediction of the response. In biomarker studies, for example, the goal is to identify which genes are related to certain diseases. Accurate statistical inference in these studies can aid in the discovery of critical biological markers, guiding effective treatment and prevention strategies.

The use of classical statistical inference techniques on models chosed adaptively based on data exploration can lead to biases in parameter estimation and overreporting of significant findings. This can compromise the replicability of scientific studies and lead to inaccurate findings. In the worst case, this could have negative implications for treatment strategies in the medical field. Replicability is generally considered important in the statistical community because it ensures that findings are reliable (Kuchibhotla et al. 2022). Promoting replicability and the use of properly adjusted statistical inference methods also align with Goal 17.6, which emphasizes the importance of knowledge sharing and cooperation for access to science, technology, and innovation (United Nations 2015). The use of a statistical framework that accounts for adaptive selection promotes transparency and collaboration in science and technology.

We aim our focus towards the polyhedral framework for inference after variable selection in the multiple linear regression model with Gaussian errors. In Chapter 2 we recall the multiple linear regression model and least squares estimation of model coefficients. The need for model selection in the linear regression

context is briefly discussed, and we describe two classes of model selection methods, subset selection methods and shrinkage methods. In Chapter 3, we introduce the theoretical foundation for the polyhedral approach to inference after adaptive model selection. We explain the general idea behind conditional inference and focus on the special case of inference conditional on constraints that can be formulated in polyhedral form. A central result in this framework is the Polyhedral Lemma, which is used to derive selection adjusted tests and confidence intervals with $1-\alpha$ coverage conditional on the selected model and the signs of the model coefficients. The approach is generalized to conditioning on only the model, resulting in shorter confidence intervals with the same coverage. Chapter 4 describes the application of the polyhedral inference framework from Chapter 3 to models selected by forward stepwise selection and the lasso with fixed λ. To enhance understanding of the theoretical results, we implement these schemes in R, along with the necessary functionality to perform selection adjusted tests and calculate confidence intervals conditional on the constructed polyhedral constraints. The code is used to produce examples that illustrate the need for post-selection inference techniques, and demonstrate the properties of the selection adjusted confidence intervals.

Chapter 2

General Setting

2.1 Multiple linear regression

2.1.1 The multiple linear regression model

We assume that a dataset consisting of a response variable $\mathbf{y}$ and predictor matrix X is obtained by random sampling from a population. Throughout this thesis we restrict our focus to the classical normal linear regression setting. The response variable $\mathbf{y} \in \mathbb{R}^n$ and design matrix of predictors $X = (\mathbf{x}_1, \ldots, \mathbf{x}_p) \in \mathbb{R}^{n \times p}$ are assumed to have the relation

$$\mathbf{y} = X\boldsymbol{\beta} + \boldsymbol{\varepsilon}, \qquad \boldsymbol{\varepsilon} \sim N(0, \sigma^2 I). \tag{2.1}$$

Here $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)^T \in \mathbb{R}^p$ is the vector of unkown regression coefficients, and the variance $\sigma^2 > 0$ is assumed known. Equivalently to this formulation of the linear model is that the response $\mathbf{y}$ is drawn from a multivariate normal distribution

$$\mathbf{y} \sim N(\boldsymbol{\mu}, \sigma^2 I), \tag{2.2}$$

where $\boldsymbol{\mu} = X\boldsymbol{\beta}$ is a linear function of the predictors. The design matrix X is assumed to be fixed.

2.1.2 Least squares estimation

The regression coefficients $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)^T$ of the linear model quantify the association between the variables $X = (\mathbf{x}_1, \ldots, \mathbf{x}_p)$ and the response $\mathbf{y}$. A regression coefficient β_j is interpreted as the average effect a unit increase in $\mathbf{x}_j$ has on $\mathbf{y}$ when holding all other predictors fixed (James et al. 2013). The regression coefficients are unknown, and need to be estimated in order for predictions to be made from the linear model (2.1) by $\hat{\mathbf{y}} = X\hat{\boldsymbol{\beta}}$, where $\hat{\boldsymbol{\beta}}$ are the estimated regression coefficients. This is done by minimizing the residual sum of squares (RSS)

$$\text{RSS} = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2,$$

which can also be formulated as

$$\text{RSS}(\boldsymbol{\beta}) = (\mathbf{y} - X\boldsymbol{\beta})^T(\mathbf{y} - X\boldsymbol{\beta}). \tag{2.3}$$

The RSS is a quadratic function of $\boldsymbol{\beta}$ for which there always exists a minimizer. Assuming that X has full rank, $\text{rank}\, X = p$, this minimizer is unique. We differentiate (2.3) with respect to $\boldsymbol{\beta}$

$$\frac{\partial((\mathbf{y} - X\boldsymbol{\beta})^T(\mathbf{y} - X\boldsymbol{\beta}))}{\partial \boldsymbol{\beta}} = -2X^T(\mathbf{y} - X\boldsymbol{\beta})$$
$$\frac{\partial^2((\mathbf{y} - X\boldsymbol{\beta})^T(\mathbf{y} - X\boldsymbol{\beta}))}{\partial \boldsymbol{\beta}\partial \boldsymbol{\beta}^T} = -2X^TX$$

(Hastie et al. 2009). By setting the first derivative to zero we obtain the unique least squares solution

$$\hat{\boldsymbol{\beta}} = (X^TX)^{-1}X^T\mathbf{y}. \tag{2.4}$$

2.2 Model selection

In cases where we have many observations compared to predictors, $n \gg p$, the least squares estimates tend to give a good model fit when the response and the predictors are linearly related. In this case, the full least squares model tends to have estimates with low variance, and good prediction accuracy (James et al. 2013). When it is not the case that the number of observations is much larger than the number of predictors, the full least squares model (2.1) tends to have high variance, leading to a poor prediction abilities. In the case where we have fewer observations than predictors, $n < p$, the matrix X no longer has full rank, and there no longer exists a unique solution to the least squares problem. Then there is an infinite set of solutions to the least squares problem.

All least squares estimates $\hat{\boldsymbol{\beta}} = (\hat{\beta}_1, \ldots, \hat{\beta}_p)^T$ are typically nonzero (Hastie et al. 2015). While it is not impossible for all predictors in the dataset to be associated with the response, it is more often the case that only a subset of the predictors are relevant for predicting the response (James et al. 2013). It is of interest to determine a subset of the variables to be included in a linear model, in order to improve the prediction accuracy and interpretability of the linear model. We briefly discuss two well known categories of model selection methods, the *subset selection* approach and *shrinkage* methods.

2.2.1 Subset selection

Subset selection methods use least squares to fit a model to a subset of the candidate predictors. The simplest form of this is the best subset selection method, in which all possible submodels are fitted and evaluated by a model selection criterion. The Akaike Information Criterion (AIC) and adjusted R^2 are popularly used to evaluate the model fit. While this guarantees that the chosen model is the best

fit to the data according to the selected criterion, there are several drawbacks of this approach to model selection.

There are in total 2^p models that contain subsets of p predictors (James et al. 2013). For a large number of predictors p, the best subset selection method is computationally heavy, and even infeasible for $p > 40$ (James et al. 2013). Additionally, when searching through a very large number of candidate models, we are likely to find an alternative that fits the data well. However, as the data is a sample from a larger population that we wish to infer properties about or make predictions on, we are likely to be overfitting to the realized sample of the data used to choose the model.

Selection procedures that explore a more restricted set of candidate models are often preferred to the best subset selection method. Stepwise selection is a classical approach to variable selection, and does not require nearly as much computation as best subset selection. In the forward selection procedure, our starting point is the null model containing only the intercept. At each step, the variable that lowers the model RSS the most out of all the available variables in the dataset is chosen to enter the model. Without an implemented stopping rule, the forward selection procedure will keep going until all the predictors are added when the RSS is used as the model selection criterion. For $p = 20$, the best subset selection method fits more than 1 million models, while the forward stepwise selection method only requires 211 models to be fitted in the absence of an implemented stopping rule (James et al. 2013).

A drawback of the forward stepwise selection method for variable selection is that does not guarantee that the best model, according to the chosen selection criterion, is chosen out of all the 2^p possible candidate models. Yet, it tends to perform well in practice, and is still used in practice and taught in introductory statistics courses. Backward stepwise selection is similar to the forward selection method, and takes the full least squares model containing all candidate predictors as its starting point, and iteratively removes predictors to end up with a good model. Like the forward selection procedure, backward selection does not guarantee that the best candidate model is chosen.

All forms of stepwise selection are discrete variable selection procedures, and tend to suffer from high variances of the estimates of the regression coefficients. A class of model selection methods that do not share this issue is shrinkage methods, which perform variable selection continuously.

2.2.2 Shrinkage methods

Shrinkage methods fit a model containing all p candidate predictors with constraints on the coefficient estimates. The constraints shrink the coefficients towards 0, which can reduce their variance significantly (James et al. 2013). Shrinkage methods perform well in cases of more predictors than observations, $p > n$, because they sacrifice some bias in exchange for a large decrease in variance. The two most known shrinkage techniques are ridge regression and the lasso.

The shrinkage method that we will focus on in this thesis is the least absolute shrinkage and selection operator, the lasso. The lasso fits a least squares model to the data with an additional penalty on the size of the regression coefficients. This shrinks several of the regression coefficients to zero, making the lasso a model selection tool. The lasso performs particularly well on sparse data, where many features may be irrelevant and should be excluded from the model.

2.3 Inference after variable selection in the linear model

It is often of interest to determine the statistical strength of the variables included in a model, and form confidence intervals for the model effects. Clearly, model selection procedures are necessary in order to obtain interpretable models that fit the data well. Through the use of adaptive variable selection procedures in search for an interesting model with the most significant effects included, we are "cherry picking" for the strongest relationships between the predictors and the response (Taylor and Tibshirani 2015). By the nature of most model selection methods, the variables that are chosen to enter the model tend to be the significant ones, leading to overfitting on the realized data sample from the population we wish to infer properties about (Lee et al. 2016). A much explored topic in recent statistical literature is how to perform valid inferences after model selection. While there is no complete and unified framework for post-selection inference, there are several well-known methods for avoiding overly optimistic inferences when models are chosen on the basis of the realized observations.

One of the most widely used methods for obtaining valid inferences after model selection in statistics and machine learning is sample splitting. As the name suggests, the data is split into a set of training data used to fit the model, and a set of test data to assess the model. An advantage of sample splitting compared to more restrictive methods is that it imposes no restrictions on the selection procedures used, since only the training data is used to choose the model. Sample splitting provides valid inferences after model selection regardless of the nature of the selection procedure. The resulting confidence intervals are at most $\sqrt{2}$ times wider than intervals ignoring the model selection (Kuchibhotla et al. 2022). A disadvantage of data splitting is that it results in a loss of test power, especially in cases where the number of observations is small. Another argument used against sample splitting is that the model choice depends on the particular observed split of the data. Fithian et al. (2014) argues that the inferences from data splitting are valid only if the first data split attempt is used (Lee et al. 2016). In practice, multiple splits resulting in different models might be tested in order to find a good fit to the data, which violates this criteria. Simply put, sample splitting is an effective technique for obtaining valid inferences after model selection, but it does not cover all our needs for post-selection inference tools.

Another approach to post-selection inference is to condition p-values and confidence intervals on the model selection procedure directly. In this way, the entire dataset can be used for both model selection and inference. In the next chapter, we

introduce the idea behind the conditional inference approach, and describe the polyhedral inference framework, in which the selection events can be described by a set of linear constraints on $\mathbf{y}$.

Chapter 3

Exact inference by the polyhedral lemma

In this chapter we introduce the necessary theoretical foundation for the relatively recent *polyhedral* approach to inference after adaptive model selection. In Section 3.1, we elaborate on the general idea behind the conditional inference approach. In Section 3.2, we aim our focus towards inference conditional on constraints on polyhedral form. This special case of conditional inference has several desirable properties. A remarkable property of the polyhedral inference framework is that it is a closed form method. The resulting p-values from the polyhedral method are *exact*, meaning that they are uniformly distributed when the null hypothesis is true. This approach to post-selection inference allows us to perform both model selection and inference on the entire dataset.

We give a comprehensive derivation of the polyhedral lemma, and elaborate on how to apply the result to construct a selection adjusted test statistic. We generalize the construction of this selection adjusted test statistic to include all possible sign patterns. Lastly, we specify how to invert the test statistic, giving a selection adjusted confidence interval for the model effects to be tested.

3.1 Conditional inference

Conditional inference is an approach to post-selective inference in which the constructed hypothesis tests and confidence intervals are conditioned on the selection of the model. We consider the linear regression setting from Section 2.1. The most common inference in question is the significance testing and forming of confidence intervals for the regression coefficients. In a classical setting in which the model to be tested is specified beforehand, we can expect that requiring

$$\mathbb{P}\big(\beta_j \in \mathrm{CI}_j\big) \geq 1 - \alpha$$

to be fulfilled generally yields valid p-values and confidence intervals with $1 - \alpha$ coverage. However, when a linear model has been chosen through adaptive

selection procedures, the p-values have undesired frequency properties. They tend to be significant, and their corresponding confidence intervals tend to not have the reported coverage whenever classical tests are used post-selection without accounting for selection.

Let $A = \{j : \beta_j \neq 0\} \subset \{1, \ldots, p\}$ represent a candidate model. We denote the particular model selected by the selection procedure as $\hat{A}$. The idea of the conditional inference approach is to require that the confidence intervals must be required to have $1-\alpha$ coverage conditional on the selected model $\hat{A}$,

$$\mathbb{P}\left(\beta_j \in \mathrm{CI}_j \mid \hat{A} = A\right) \geq 1 - \alpha \tag{3.1}$$

(Kuchibhotla et al. 2022). Achieving this is feasible only when the model selection is carried out through a well-defined procedure that can be fully specified as a statistical event. The selection event $\{\hat{A} = A\}$ is the set of all values of $\mathbf{y}$ that would yield the model $\hat{A}$ with the selection procedure in question. In order to obtain a confidence interval with conditional coverage such as in (3.1), we aim to characterize the distribution of

$$\boldsymbol{\eta}^T \mathbf{y} \mid \{\hat{A} = A\},$$

where $\boldsymbol{\eta} \in \mathbb{R}^n$ is a linear contrast vector specifying a direction of interest. The submatrix containing the columns specified by the set A is denoted $X_A = (\mathbf{x}_j, j \in A)$. Generally, we are interested in obtaining inference of the the β_j included in the model. We have that $\boldsymbol{\eta}^T \boldsymbol{\mu} = \beta_j$ when $\boldsymbol{\eta} = X_A(X_A^T X_A)^{-1}\mathbf{e}_j$, where $\mathbf{e}_j = (0, 0, \ldots, 1, \ldots, 0)^T$ is the basis vector where all components are zero except for the j-th component, which is 1. An unbiased estimator of $\boldsymbol{\eta}^T \boldsymbol{\mu} = \beta_j$ is $\boldsymbol{\eta}^T \mathbf{y} = \hat{\beta}_j$.

3.2 Inference conditional on polyhedral constraints

The polyhedral approach to valid inference after model selection was introduced by Lee et al. (2016). Our goal is to characterize the distribution of $\boldsymbol{\eta}^T \mathbf{y} \mid \{\hat{A} = A\}$ in order to obtain inferences that account for selection. The polyhedral lemma is a central result in this framework, and can be applied to yield valid inference after model selection in cases where the model selection event can be fully characterized by polyhedral constraints of the form $\{\Psi \mathbf{y} \leq \mathbf{b}\}$. It turns out that several model selection procedures, amongst them the forward stepwise selection method and the lasso, can be expressed by constraints on this form when the sign pattern of the coefficients is included in the selection event. In Chapter 4, we elaborate on how to derive polyhedral selection events from the selection criteria of these procedures. In order to study the distribution of $\boldsymbol{\eta}^T \mathbf{y} \mid \{\hat{A} = A\}$, we first characterize the event that a particular model along with its sign pattern is chosen, $\{\hat{A} = A, \hat{\mathbf{s}} = \mathbf{s}\} = \{\Psi \mathbf{y} \leq \mathbf{b}\}$. The polyhedral lemma states that this selection event has an equivalent formulation in terms of $\boldsymbol{\eta}^T \mathbf{y}$, and functions that are statistically independent of $\mathbf{y}$. This reformulation proves to be very useful when characterizing the distribution.

3.2.1 The polyhedral lemma

Let $\mathbf{c} \equiv \boldsymbol{\eta}(\boldsymbol{\eta}^T\boldsymbol{\eta})^{-1}$ and $\mathbf{z} \equiv (I - \mathbf{c}\boldsymbol{\eta}^T)\mathbf{y}$. The polyhedral selection event is equivalent to

$$\{\Psi\mathbf{y} \leq \mathbf{b}\} = \{\mathcal{V}^-(\mathbf{z}) \leq \boldsymbol{\eta}^T\mathbf{y} \leq \mathcal{V}^+(\mathbf{z}), \quad \mathcal{V}^0(\mathbf{z}) > 0\}, \tag{3.2}$$

where

$$\mathcal{V}^-(\mathbf{z}) = \max_{j:(\Psi\mathbf{c})_j<0} \frac{(\mathbf{b})_j - (\Psi\mathbf{z})_j}{(\Psi\mathbf{c})_j}, \tag{3.3a}$$

$$\mathcal{V}^+(\mathbf{z}) = \min_{j:(\Psi\mathbf{c})_j>0} \frac{(\mathbf{b})_j - (\Psi\mathbf{z})_j}{(\Psi\mathbf{c})_j}, \tag{3.3b}$$

$$\mathcal{V}^0(\mathbf{z}) = \min_{j:(\Psi\mathbf{c})_j=0} ((\mathbf{b})_j - (\Psi\mathbf{z})_j). \tag{3.3c}$$

We note that $\mathcal{V}^-(\mathbf{z})$, $\mathcal{V}^+(\mathbf{z})$ and $\mathcal{V}^0(\mathbf{z})$ are functions of $\mathbf{z}$, which is statistically independent of $\boldsymbol{\eta}^T\mathbf{y}$. Figure 3.1 visualizes the polyhedral lemma, providing us with some intuition of the result.

3.2.2 Deriving the polyhedral lemma

In order to show how to arrive at the polyhedral lemma from our base assumptions, we adapt the proof from p. 917 of Lee et al. (2016), as well as explanations from pp. 151–152 of Hastie et al. (2015). Parts of this section have been adapted from previous work from Næss (2023). It is assumed that the response has a Gaussian distribution, $\mathbf{y} \sim N(\boldsymbol{\mu}, \sigma^2 I)$, where the variance σ^2 is known. We decompose $\mathbf{y}$ into the sum of its projection onto η and its projection onto the subspace orthogonal to η,

$$\mathbf{y} = P_{\boldsymbol{\eta}}\mathbf{y} + (I - P_{\boldsymbol{\eta}})\mathbf{y},$$

where $P_{\boldsymbol{\eta}} = \boldsymbol{\eta}(\boldsymbol{\eta}^T\boldsymbol{\eta})^{-1}\boldsymbol{\eta}^T$. By defining $\mathbf{c} \equiv \boldsymbol{\eta}(\boldsymbol{\eta}^T\boldsymbol{\eta})^{-1}$, the decomposition of $\mathbf{y}$ can be written as

$$\mathbf{y} = \mathbf{c}\boldsymbol{\eta}^T\mathbf{y} + (I - \mathbf{c}\boldsymbol{\eta}^T)\mathbf{y}.$$

Defining $\mathbf{z} \equiv (I - \mathbf{c}\boldsymbol{\eta}^T)\mathbf{y}$ allows us to rewrite the decomposition of $\mathbf{y}$ as

$$\mathbf{y} = \mathbf{c}\boldsymbol{\eta}^T\mathbf{y} + \mathbf{z}.$$

The $\mathbf{z}$ term of this sum is of interest because it is independent of $\mathbf{y}$ because the vectors are uncorrelated. The polyhedral lemma (3.2) rewrites the selection event $\{\Psi\mathbf{y} \leq \mathbf{b}\}$ in terms of $\boldsymbol{\eta}^T\mathbf{y}$, and three functions of $\mathbf{z}$, $\mathcal{V}^-(\mathbf{z})$, $\mathcal{V}^+(\mathbf{z})$ and $\mathcal{V}^0(\mathbf{z})$. Inserting $\mathbf{y} = \mathbf{c}\boldsymbol{\eta}^T\mathbf{y} + \mathbf{z}$ for $\mathbf{y}$ in the selection event yields

$$\begin{aligned}\{\Psi\mathbf{y} \leq \mathbf{b}\} &= \{\Psi(\mathbf{c}\boldsymbol{\eta}^T\mathbf{y} + \mathbf{z}) \leq \mathbf{b}\} \\ &= \{\Psi\mathbf{c}\boldsymbol{\eta}^T\mathbf{y} \leq \mathbf{b} - \Psi\mathbf{z}\}.\end{aligned}$$

This inequality can be equivalently formulated componentwisely as

$$\{(\Psi\mathbf{c})_j \boldsymbol{\eta}^T\mathbf{y} \leq (\mathbf{b})_j - (\Psi\mathbf{z})_j \quad \text{for all } j\},$$

where $(\mathbf{b})_j$ denotes component j of vector $\mathbf{b}$. Dividing each side of the inequality by $(\Psi\mathbf{c})_j$ yields the event

$$\left\{ \begin{array}{ll} \boldsymbol{\eta}^T\mathbf{y} \leq \dfrac{(\mathbf{b})_j - (\Psi\mathbf{z})_j}{(\Psi\mathbf{c})_j}, & \text{for all } j : (\Psi\mathbf{c})_j > 0, \\ \boldsymbol{\eta}^T\mathbf{y} \geq \dfrac{(\mathbf{b})_j - (\Psi\mathbf{z})_j}{(\Psi\mathbf{c})_j}, & \text{for all } j : (\Psi\mathbf{c})_j < 0, \\ 0 \leq (\mathbf{b})_j - (\Psi\mathbf{z})_j, & \text{for all } j : (\Psi\mathbf{c})_j = 0 \end{array} \right\}. \tag{3.4}$$

Here the components j are sorted in three categories depending on the sign of $(\Psi\mathbf{c})_j$, as this decides the direction of the inequality. From this it is clear that $\boldsymbol{\eta}^T\mathbf{y}$ must lie in the interval between the maximum value of its lower bound and the minimum value of its upper bound. To achieve a more concise description of this set of inequalities, we define the lower and upper bound as

$$\mathcal{V}^-(\mathbf{z}) = \max_{j:(\Psi\mathbf{c})_j<0} \frac{(\mathbf{b})_j - (\Psi\mathbf{z})_j}{(\Psi\mathbf{c})_j}, \qquad \text{and } \mathcal{V}^+(\mathbf{z}) = \min_{j:(\Psi\mathbf{c})_j>0} \frac{(\mathbf{b})_j - (\Psi\mathbf{z})_j}{(\Psi\mathbf{c})_j},$$

respectively. Additionally, the criteria from the last line of (3.4) can be formulated as

$$\mathcal{V}^0(\mathbf{z}) = \min_{j:(\Psi\mathbf{c})_j=0} ((\mathbf{b})_j - (\Psi\mathbf{z})_j).$$

Then (3.4) can be formulated as $\{\mathcal{V}^-(\mathbf{z}) \leq \boldsymbol{\eta}^T\mathbf{y} \leq \mathcal{V}^+(\mathbf{z}), \quad \mathcal{V}^0(\mathbf{z}) > 0\}$, and we conclude that

$$\{\Psi\mathbf{y} \leq \mathbf{b}\} = \{\mathcal{V}^-(\mathbf{z}) \leq \boldsymbol{\eta}^T\mathbf{y} \leq \mathcal{V}^+(\mathbf{z}), \quad \mathcal{V}^0(\mathbf{z}) > 0\}. \tag{3.5}$$

3.2.3 The selection adjusted test statistic

The polyhedral lemma can be applied to derive a test statistic for $H_0 : \boldsymbol{\eta}^T\boldsymbol{\mu} = 0$ that accounts for the selection procedure that chose the model. Figure 3.1 shows the equivalence of events stated by the polyhedral lemma. Since the events are equivalent, they are equal in distribution,

$$\boldsymbol{\eta}^T\mathbf{y} \mid \{\Psi\mathbf{y} \leq \mathbf{b}\} \overset{d}{=} \boldsymbol{\eta}^T\mathbf{y} \mid \{\mathcal{V}^-(\mathbf{z}) \leq \boldsymbol{\eta}^T\mathbf{y} \leq \mathcal{V}^+(\mathbf{z}), \quad \mathcal{V}^0(\mathbf{z}) > 0\}.$$

The formulation of the selection event on the right hand side explicitly states the interval $[\mathcal{V}^-(\mathbf{z}), \mathcal{V}^+(\mathbf{z})]$ to which $\boldsymbol{\eta}^T\mathbf{y}$ is truncated. As we already know that $\boldsymbol{\eta}^T\mathbf{y} \sim N(\boldsymbol{\eta}^T\boldsymbol{\mu}, \sigma^2 I)$, this suggests that truncating the unconditional distribution of to this interval would provide the distribution adjusted for selection. We emphasize the fact that we have conditioned on the value $\mathbf{z} = (I - \mathbf{c}\boldsymbol{\eta}^T)\mathbf{y}$. Even though $\mathcal{V}^-(\mathbf{z})$,

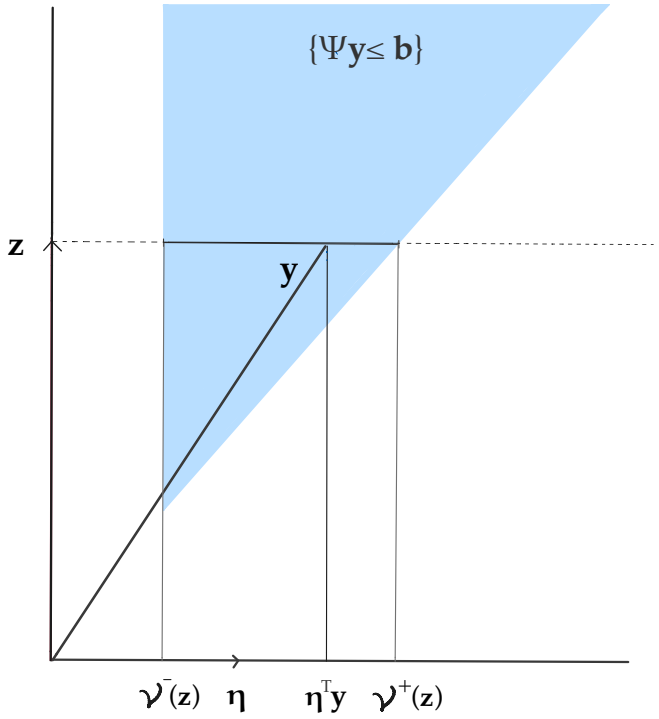


Figure 3.1: A geometric interpretation of the stated equivalence of the events $\{\Psi\mathbf{y} \leq \mathbf{b}\} = \{\mathcal{V}^-(\mathbf{z}) \leq \boldsymbol{\eta}^T\mathbf{y} \leq \mathcal{V}^+(\mathbf{z})\}$ in the polyhedral lemma in the case of $n = 2$, $\sigma^2 = 1$ and $\|\boldsymbol{\eta}\|_2 = 1$. Inpired by Lee et al. (2016), and recreated from Næss (2023).

$\mathcal{V}^+(\mathbf{z})$ and $\mathcal{V}^0(\mathbf{z})$ are functions of $\mathbf{z}$, which is independent of $\boldsymbol{\eta}^T\mathbf{y}$, the distribution of $\eta^T\mathbf{y} \mid \{\mathcal{V}^-(\mathbf{z}) \leq \boldsymbol{\eta}^T\mathbf{y} \leq \mathcal{V}^+(\mathbf{z}), \quad \mathcal{V}^0(\mathbf{z}) > 0\}$ still depends on $\mathbf{z}$. This is intuitive from Figure 3.1, where a $\mathbf{z}$ of smaller magnitude in the same direction would give a smaller interval $[\mathcal{V}^-(\mathbf{z}), \mathcal{V}^+(\mathbf{z})]$. For the fixed value of $\mathbf{z} = \mathbf{z}_0 = (I - \mathbf{c}\boldsymbol{\eta}^T)\mathbf{y}$, we have

$$\boldsymbol{\eta}^T\mathbf{y} \mid \{\mathcal{V}^-(\mathbf{z}) \leq \boldsymbol{\eta}^T\mathbf{y} \leq \mathcal{V}^+(\mathbf{z}), \;\; \mathcal{V}^0(\mathbf{z}) > 0, \mathbf{z} = \mathbf{z}_0\} \sim \mathrm{TN}\big(\boldsymbol{\eta}^T\boldsymbol{\mu}, \sigma^2 I, \mathcal{V}^-(\mathbf{z}), \mathcal{V}^+(\mathbf{z})\big)$$

Then the cumulative distribution function (CDF) of $\boldsymbol{\eta}^T\mathbf{y} \mid \{\hat{A} = A, \hat{\mathbf{s}} = \mathbf{s}, \mathbf{z} = \mathbf{z}_0\}$ is

$$F_{\mathbf{z}_0}(x) = \mathbb{P}(\boldsymbol{\eta}^T\mathbf{y} \leq x \mid \mathcal{V}^-(\mathbf{z}_0) \leq \boldsymbol{\eta}^T\mathbf{y} \leq \mathcal{V}^+(\mathbf{z}_0), \quad \mathcal{V}^0(\mathbf{z}_0) > 0, \;\; \mathbf{z} = \mathbf{z}_0)$$

$$= \frac{\mathbb{P}(\mathcal{V}^-(\mathbf{z_0}) \leq \boldsymbol{\eta}^T\mathbf{y} \leq x)}{\mathbb{P}(\mathcal{V}^-(\mathbf{z_0}) \leq \boldsymbol{\eta}^T\mathbf{y} \leq \mathcal{V}^+(\mathbf{z_0}))} \quad \text{for } \mathcal{V}^-(\mathbf{z_0}) \leq x \leq \mathcal{V}^+(\mathbf{z_0}), \;\; \mathcal{V}^0(\mathbf{z_0}) > 0.$$

From this we know that

$$\mathbb{P}(F_{\mathbf{z}_0} \leq \alpha \mid \hat{A} = A, \hat{\mathbf{s}} = \mathbf{s}, \mathbf{z} = \mathbf{z}_0) \leq \alpha.$$

We integrate over all values of $\mathbf{z}$,

$$\int_{-\infty}^{\infty} \mathbb{P}(F_{\mathbf{z}_0} \leq \alpha \mid \hat{A} = A, \hat{\mathbf{s}} = \mathbf{s}, \mathbf{z} = \mathbf{z}_0) f_{\mathbf{z}}(\mathbf{z}) d\mathbf{z} \leq \alpha \int_{-\infty}^{\infty} f_{\mathbf{z}}(\mathbf{z}) d\mathbf{z} = \alpha.$$

Hence, by the law of total probability we may conclude that

$$\mathbb{P}(F_{\mathbf{z}_0} \leq \alpha \mid \hat{A} = A, \hat{\mathbf{s}} = \mathbf{s}) \leq \alpha.$$

With $\{\hat{A} = A, \hat{\mathbf{s}} = \mathbf{s}\} = \{\Psi\mathbf{y} \leq \mathbf{b}\}$, we conclude that

$$\boldsymbol{\eta}^T\mathbf{y} \mid \{\Psi\mathbf{y} \leq \mathbf{b}\} \sim \mathrm{TN}\big(\boldsymbol{\eta}^T\boldsymbol{\mu}, \sigma^2 I, \mathcal{V}^-(\mathbf{z}), \mathcal{V}^+(\mathbf{z})\big).$$

We let Φ denote the cumulative distribution function (CDF) of the standard normal distribution $N(0,1)$. The CDF of a variable with truncated Gaussian distribution with support only on the interval $[a, b]$ is defined as

$$F_{\mu,\sigma^2}^{[a,b]}(x) = \frac{\Phi((x-\mu)/\sigma) - \Phi((a-\mu)/\sigma)}{\Phi((b-\mu)/\sigma) - \Phi((a-\mu)/\sigma)} \tag{3.6}$$

(Tibshirani et al. 2016, p. 604). Then the selection adjusted test statistic for $\{\hat{A} = A, \hat{\mathbf{s}} = \mathbf{s}\} = \{\Psi\mathbf{y} \leq \mathbf{b}\}$ is

$$F_{\boldsymbol{\eta}^T\boldsymbol{\mu}, \sigma^2\|\boldsymbol{\eta}\|_2^2}^{[\mathcal{V}^-(\mathbf{z}), \mathcal{V}^+(\mathbf{z})]}(\boldsymbol{\eta}^T\mathbf{y}) \mid \{\Psi\mathbf{y} \leq \mathbf{b}\} \sim U(0,1). \tag{3.7}$$

This selection adjusted test statistic can be inverted in order to obtain confidence intervals of a parameter β with $1-\alpha$ coverage conditional on $\{\hat{A} = A, \hat{\mathbf{s}} = \mathbf{s}\}$.

We let $\boldsymbol{\eta} = X_A(X_A^T X_A)^{-1}\mathbf{e}_j$. If L and U are the unique values satifying

$$F^{[\mathcal{V}^-(\mathbf{z}),\mathcal{V}^+(\mathbf{z})]}_{L,\ \sigma^2\|\boldsymbol{\eta}\|_2^2}(\boldsymbol{\eta}^T\mathbf{y}) = 1-\frac{\alpha}{2}, \qquad F^{[\mathcal{V}^-(\mathbf{z}),\mathcal{V}^+(\mathbf{z})]}_{U,\ \sigma^2\|\boldsymbol{\eta}\|_2^2}(\boldsymbol{\eta}^T\mathbf{y}) = \frac{\alpha}{2}, \tag{3.8}$$

then the interval $[L,U]$ is a confidence interval for $\boldsymbol{\eta}^T\boldsymbol{\mu} = \beta_j$, with $1-\alpha$ coverage conditional on $\{\hat{A}=A, \hat{\mathbf{s}}=\mathbf{s}\}$ (Lee et al. 2016).

3.2.4 Generalization to all sign patterns

So far we focused on the special case in which the conditioning on $\{\hat{A}=A, \hat{\mathbf{s}}=\mathbf{s}\}$ is equivalent to conditioning on the event that the response $\mathbf{y}$ falls into a single polyhedral region $\{\Psi\mathbf{y} \le \mathbf{b}\}$.

In some cases it is useful to generalize these results to cover all sign patterns $\mathbf{s}$. We explain how to condition on the selected model only $\{\hat{A}=A\}$. This event can be described as a union of polyhedra over the possible sign patterns,

$$\{\hat{A}=A\} = \bigcup_{\mathbf{s}} \{\Psi_{\mathbf{s}}\mathbf{y} \le \mathbf{b}_{\mathbf{s}}\}.$$

For every sign pattern $\mathbf{s}$, there exists a selection matrix $\Psi_{\mathbf{s}}$ and a vector $\mathbf{b}_{\mathbf{s}}$. There is a total of $2^{|A|}$ different sign patterns for a model containing $|A|$ predictors. The selection adjusted test statistic for

$$\boldsymbol{\eta}^T\mathbf{y} \,|\, \bigcup_{\mathbf{s}} \{\Psi_{\mathbf{s}}\mathbf{y} \le \mathbf{b}_{\mathbf{s}}\}.$$

is given by the CDF of a normal variable truncated to the union of intervals $[\mathcal{V}_{\mathbf{s}}^-(\mathbf{z}), \mathcal{V}_{\mathbf{s}}^+(\mathbf{z})]$,

$$F^{\bigcup_{\mathbf{s}}[\mathcal{V}_{\mathbf{s}}^-(\mathbf{z}),\mathcal{V}_{\mathbf{s}}^+(\mathbf{z})]}_{\boldsymbol{\eta}^T\boldsymbol{\mu},\ \sigma^2\|\boldsymbol{\eta}\|_2^2}(\boldsymbol{\eta}^T\mathbf{y}) \,|\, \bigcup_{\mathbf{s}} \{\Psi_{\mathbf{s}}\mathbf{y} \le \mathbf{b}_{\mathbf{s}}\} \sim U(0,1), \tag{3.9}$$

by the same argument as in the previous section. Under $H_0 : \boldsymbol{\eta}^T\boldsymbol{\mu} = 0$, the value of this statistic can be calculated by

$$F^{\bigcup_{\mathbf{s}}[\mathcal{V}_{\mathbf{s}}^-(\mathbf{z}),\mathcal{V}_{\mathbf{s}}^+(\mathbf{z})]}_{\boldsymbol{\eta}^T\boldsymbol{\mu},\ \sigma^2\|\boldsymbol{\eta}\|_2^2}(\boldsymbol{\eta}^T\mathbf{y}) = \frac{\sum_{\mathbf{s}}\left(\Phi\left(\frac{\eta^T\mathbf{y}}{\sigma\|\boldsymbol{\eta}\|_2}\right) - \Phi\left(\frac{\mathcal{V}_{\mathbf{s}}^-(\mathbf{z})}{\sigma\|\boldsymbol{\eta}\|_2}\right)\right)}{\sum_{\mathbf{s}}\left(\Phi\left(\frac{\mathcal{V}_{\mathbf{s}}^+(\mathbf{z})}{\sigma\|\boldsymbol{\eta}\|_2}\right) - \Phi\left(\frac{\mathcal{V}_{\mathbf{s}}^-(\mathbf{z})}{\sigma\|\boldsymbol{\eta}\|_2}\right)\right)} \tag{3.10}$$

3.2.5 Selection adjusted confidence intervals

In order to calculate the general selection adjusted confidence intervals CI_j conditional on $\{\hat{A}=A\}$, we evaluate the test statistic (3.10) for different values of β_j. For brevity we write $\beta = \beta_j$. The test statistic

$$F^{\bigcup_{\mathbf{s}}[\mathcal{V}_{\mathbf{s}}^-(\mathbf{z}),\mathcal{V}_{\mathbf{s}}^+(\mathbf{z})]}_{\beta,\ \sigma^2\|\boldsymbol{\eta}\|_2^2}(\boldsymbol{\eta}^T\mathbf{y}) = \frac{\sum_{\mathbf{s}}\left(\Phi\left(\frac{\boldsymbol{\eta}^T\mathbf{y}-\beta}{\sigma\|\boldsymbol{\eta}\|_2}\right) - \Phi\left(\frac{\mathcal{V}_{\mathbf{s}}^-(\mathbf{z})-\beta}{\sigma\|\boldsymbol{\eta}\|_2}\right)\right)}{\sum_{\mathbf{s}}\left(\Phi\left(\frac{\mathcal{V}_{\mathbf{s}}^+(\mathbf{z})-\beta}{\sigma\|\boldsymbol{\eta}\|_2}\right) - \Phi\left(\frac{\mathcal{V}_{\mathbf{s}}^-(\mathbf{z})-\beta}{\sigma\|\boldsymbol{\eta}\|_2}\right)\right)} \tag{3.11}$$

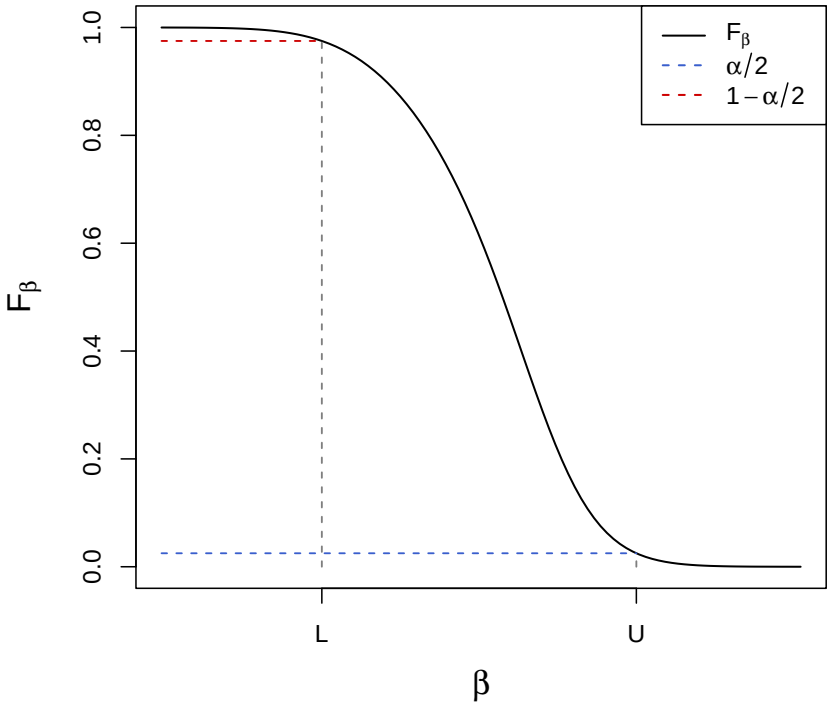


Figure 3.2: Illustration of how a $1-\alpha$ confidence interval $[L,U]$ for a parameter β is obtained from its truncated Gaussian test statistic. Here F_β is an abbreviation of $F^{\bigcup_s[\mathcal{V}_s^-(\mathbf{z}),\mathcal{V}_s^+(\mathbf{z})]}_{\beta,\sigma^2\|\boldsymbol{\eta}\|_2^2}(\boldsymbol{\eta}^T\mathbf{y})$.

is a truncated Gaussian, which is monotonically decreasing with respect to β (Lee et al. 2016). This selection adjusted test statistic can be inverted in order to obtain confidence intervals of a parameter β with $1-\alpha$ coverage conditional on $\{\hat{A}=A\}$.

We let $\boldsymbol{\eta}=X_A(X_A^TX_A)^{-1}\mathbf{e}_j$. If L and U are the unique values satifying

$$F^{\bigcup_s[\mathcal{V}_s^-(\mathbf{z}),\mathcal{V}_s^+(\mathbf{z})]}_{L,\sigma^2\|\boldsymbol{\eta}\|_2^2}(\boldsymbol{\eta}^T\mathbf{y})=1-\frac{\alpha}{2},\qquad F^{\bigcup_s[\mathcal{V}_s^-(\mathbf{z}),\mathcal{V}_s^+(\mathbf{z})]}_{U,\sigma^2\|\boldsymbol{\eta}\|_2^2}(\boldsymbol{\eta}^T\mathbf{y})=\frac{\alpha}{2},\tag{3.12}$$

then the interval $[L,U]$ is a confidence interval for $\boldsymbol{\eta}^T\boldsymbol{\mu}=\beta_j$, with $1-\alpha$ coverage conditional on $\{\hat{A}=A\}$ (Lee et al. 2016).

Chapter 4

Applications of polyhedral inference

The polyhedral approach to post selective inference described in Chapter 3 can be applied in cases where the entire model selection procedure can be characterized by a polyhedral selection event $\{\Psi \mathbf{y} \leq \mathbf{b}\}$. In this chapter we elaborate on the application of the polyhedral inference method for two well known model selection procedures that fit this criteria; forward selection and the lasso. We illustrate the need for a selection adjusted test in the absence of data splitting for models selected by adaptive selection methods, and derive their general selection events for use within the polyhedral inference framework. The closed form specification of these selection events is what allows us to derive tests and confidence intervals that account for the model selection made by the data.

As the forward selection procedure is very simple, it allows for a light introduction to the application of the polyhedral inference approach, and provides suitable examples of the negative effects of naive testing on adaptively chosen models. Its characterizing selection event is derived directly from the selection criteria, and we provide examples by implementation in R.

Inference on linear models *selected by* the lasso is the main focus of this thesis. Deriving the lasso selection event requires more rigor, and some results from convex optimization theory. We follow the lead of Lee et al. (2016) for deriving the lasso selection event. We put special emphasis on the generalization to conditioning on the model only, $\{\hat{A} = A\}$, omitting the conditioning on the signs of the model coefficients. We show that this yields shorter confidence intervals with the same $1-\alpha$ coverage in cases where null signals or very weak signals are included in the selected model.

4.1 Forward selection

Forward selection is one of the simplest forms of variable selection for multiple linear regression models. In the absence of data splitting, inference on the regres-

sion coefficients of a linear model chosen by forward selection requires special attention, as classical t-tests rely on the assumption that a model is specified prior to examining the data.

We give a short introduction to the forward selection procedure for linear regression models, illustrate the need for a selection adjusted test, and derive the general scheme for constructing its descriptive polyhedral selection events.

4.1.1 Introduction to forward selection

Forward selection is a stepwise model selection procedure for linear regression. The starting point of the algorithm is the null model, $y_i = \beta_0 + \epsilon_i,\ i = 1,\dots,n$. At each step of forward selection procedure, the candidate predictors $\mathbf{x}_1,\dots,\mathbf{x}_p$ in the dataset are entered to the model sequentially according to a chosen selection criterion.

There are many reasonable selection criteria to choose when performing model selection by the forward selection procedure. One of the most widely used in implementations is the Akaike Information Criterion (AIC). AIC is commonly defined as

$$\text{AIC} = \frac{1}{n}(\text{RSS} + 2|A|\hat{\sigma}^2),$$

where $|A|$ is the number of covariates included in the model, $\hat{\sigma}^2$ is the least squares estimate of the variance in the linear model, and the residual sum of squares (RSS) is defined as

$$\text{RSS} = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{4.1}$$

(James et al. 2013). Here $\hat{y}_i$ is the predicted response from the linear model. A lower AIC score indicates a better model fit, and the penalization of model complexity provides a natural stopping rule for the forward selection procedure. RSS may be used directly as the selection criterion, which is the case we consider in the following subsections.

4.1.2 Hypothesis testing after forward selection

We consider variable selection by the forward selection procedure for multiple linear regression with RSS as the selection criterion of choice. Of interest is the significance testing of the variable $\mathbf{x}_j$ entered to the model at step $k = 1,2,\dots$, where the null and alternative hypotheses are

$$H_0 : \beta_j = 0, \quad H_1 : \beta_j \neq 0.$$

Let A_k denote the ordered set of indices of predictors chosen by forward selection after the k first steps. X_{A_k} is the matrix consisting of the active predictors after k steps of forward selection. The RSS of a linear regression model with response $\mathbf{y}$ and design matrix X is denoted by $\text{RSS}(\mathbf{y}, X)$. At each step k, the predictor that

yields the largest drop in RSS is added to the model. Assuming known variance, this criterion is equivalent to maximizing the statistic

$$R_{k,j} = \frac{1}{\sigma^2}\left(\text{RSS}(\mathbf{y}, X_{A_{k-1}}) - \text{RSS}(\mathbf{y}, X_{A_{k-1}\cup j})\right) \tag{4.2}$$

(Lockhart et al. 2014). $X_{A_{k-1}\cup j}$ is the matrix of active predictors from the previous step, with $\mathbf{x}_j$ concatenated horizontally to $X_{A_{k-1}}$. Under the null hypothesis, the $R_{k,j}$ statistic follows a χ^2_1 distribution when the inclusion of the variable $\mathbf{x}_j$ is fixed. Its maximum over all potential added covariates, $\max_{j\in\{1,\dots,p\}}(R_{k,j})$, does not share this property, as

$$\max_{j\in\{1,\dots,p\}}(R_{k,j}) \geq R_{k,j},$$

and must necessarily be stochastically larger than the χ^2_1-distribution.

This explains why using a standard $\mathcal{X}^2_1$ test to determine the significance of a variable entered by forward selection will result in larger type I error than the nominal level α. We illustrate the issue of performing tests that assume fixed predictors on models chosen by forward selection. We simulate the first step of forward selection in linear regression on a random dataset with no underlying signal. For any number of predictors p, we assume the null hypothesis $H_0 : \beta_1 = \beta_2 = \cdots = \beta_p = 0$ is true.

Figure 4.1 shows a simulated example of the observed p-values plotted against expected p-values under H_0 for the effect added by the first step of forward selection. For each point p_j in the plot, the observed value denotes the proportion of p-values smaller than p_j,

$$\frac{1}{n}\sum_{i=1}^{n} I(p_i \leq p_j),$$

where I is the indicator function. The setup is inspired by the simulated example presented by Tibshirani (2016) in the Leo Breiman lecture of 2015 and uses the `selectiveInference` library by Tibshirani et al. (2022) to perform the selection adjusted test. The R code to produce the plot is given in Appendix A.1.

In Figure 4.2 we compare the Type I error rate from using a naive method to that of a selection adjusted test statistic. We calculate the type I error by

$$\text{Type I error rate} = \frac{1}{n}\sum_{i=1}^{n} I(p_i \leq \alpha).$$

In Figure 4.1 we can note that the dashed vertical line at $\alpha = 0.05$ crosses the observed simulated p-values for the selection adjusted test in blue at around 0.05, and at just above 0.2 for the naive test in red. This corresponds with the values in Figure 4.2, where we see that for 5 predictors, the type I error of the naive method is at just above 0.2, while it stays at the nominal level of 0.05 for the selection adjusted method.

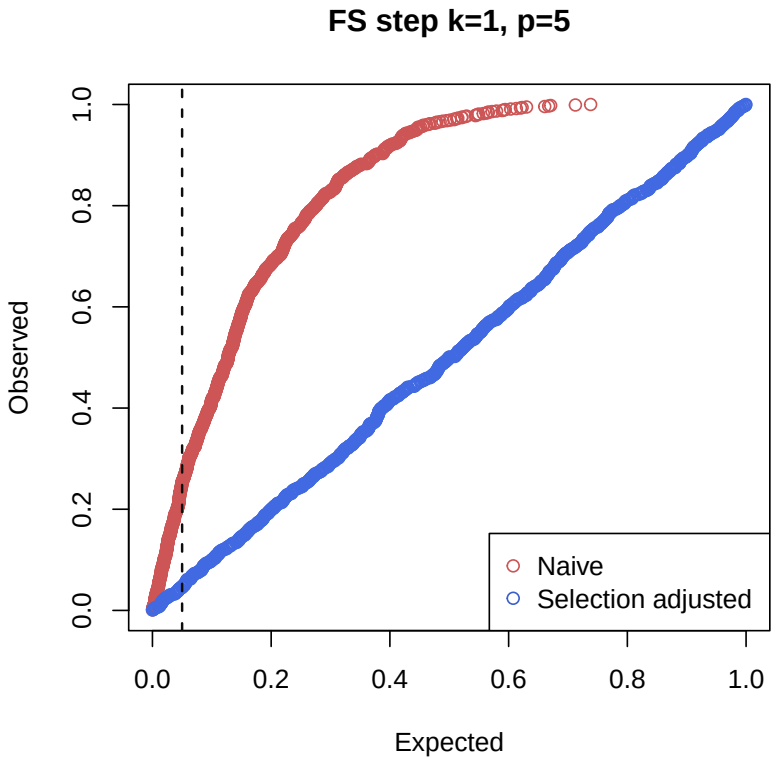


Figure 4.1: Quantile-quantile plot of 1000 simulations of the χ^2-statistic, $R_{1,j}$. Recreated after inspiration from Tibshirani (2016), and adapted from previous figure from Næss (2023).

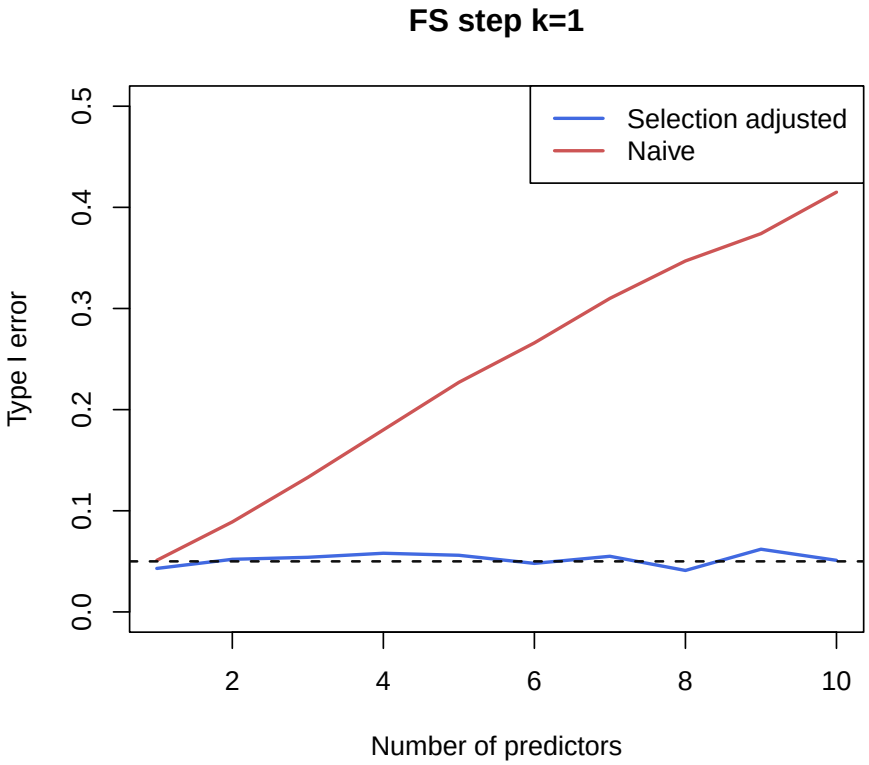


Figure 4.2: Type I errors from significance test of the variable entered at the first step of forward selection for an increasing number of candidate predictors. For the naive test, the type I error increases linearly, while the selection adjusted test from the polyhedral inference framework stays at the nominal type I error level of $\alpha = 0.05$.

4.1.3 Polyhedral selection events for forward selection

The forward selection procedure can in its entirety be represented by a set of linear inequalities forming a selection event $\{\Psi \mathbf{y} \leq \mathbf{0}\}$. The polyhedral selection event defines the set of all response vectors $\mathbf{y}$ that would lead to a particular model to be selected by forward selection. In this section we describe the general scheme for deriving the model selection event for linear models chosen by forward selection with RSS as the chosen selection criteria. Throughout the construction of the selection matrix Ψ for forward selection, we follow the lead of Tibshirani et al. (2016), pp. 605–606.

For simplicity of notation we assume no intercept. In the first step of forward selection, the matrix X consists only of the first covariate, the column vector $\mathbf{x}_j$. The jth predictor is chosen to enter the model in the first step $k = 1$ if and only if

$$\text{RSS}(\mathbf{y}, \mathbf{x}_j) \leq \text{RSS}(\mathbf{y}, \mathbf{x}_i) \text{ for all } i \in \{1, \ldots, p\} \setminus j. \tag{4.3}$$

The RSS of a linear regression model with response $\mathbf{y}$ and design matrix X can be written compactly in matrix notation as

$$\text{RSS}(\mathbf{y}, X) = \|\mathbf{y} - X\hat{\boldsymbol{\beta}}^L\|_2^2.$$

Inserting $\hat{\boldsymbol{\beta}}^L = (X^T X)X^T \mathbf{y}$ into the RSS expression above expands it to

$$\begin{aligned} \text{RSS}(\mathbf{y}, X) &= \|\mathbf{y} - X(X^T X)X^T \mathbf{y}\|_2^2 \\ &= \|\mathbf{y}^T (I - X(X^T X)X^T)\mathbf{y}\|_2^2 \\ &= \mathbf{y}^T \mathbf{y} - \frac{\mathbf{y}^T X X^T \mathbf{y}}{\|X\|_2^2}. \end{aligned}$$

We expand the inequality (4.3) as above above to obtain the selection criterion

$$\mathbf{y}^T \mathbf{y} - \frac{\mathbf{y}^T \mathbf{x}_j \mathbf{x}_j^T \mathbf{y}}{\|\mathbf{x}_j\|_2^2} \leq \mathbf{y}^T \mathbf{y} - \frac{\mathbf{y}^T \mathbf{x}_i \mathbf{x}_i^T \mathbf{y}}{\|\mathbf{x}_i\|_2^2} \text{ for all } i \in \{1, \ldots, p\} \setminus j,$$

which is equivalent to

$$\frac{\mathbf{y}^T \mathbf{x}_j \mathbf{x}_j^T \mathbf{y}}{\|\mathbf{x}_j\|_2^2} \geq \frac{\mathbf{y}^T \mathbf{x}_i \mathbf{x}_i^T \mathbf{y}}{\|\mathbf{x}_i\|_2^2} \text{ for all } i \in \{1, \ldots, p\} \setminus j.$$

Either side of the inequality above can for any design matrix X be simplified as

$$\begin{aligned} \frac{\mathbf{y}^T X X^T \mathbf{y}}{\|X\|_2^2} &= \frac{\|X^T \mathbf{y}\|_2^2}{\|X\|_2^2} \\ &= \frac{|X^T \mathbf{y}|^2}{\|X\|_2^2} \\ &= \frac{|X^T \mathbf{y}|}{\|X\|_2}. \end{aligned}$$

Then the criteria for $\mathbf{x}_j$ to be chosen in the first step of forward selection can be written as

$$\frac{|\mathbf{x}_j^T\mathbf{y}|}{\|\mathbf{x}_j\|_2} \geq \frac{|\mathbf{x}_i^T\mathbf{y}|}{\|\mathbf{x}_i\|_2} \quad \text{for all } i \in \{1,\dots p\} \setminus j.$$

In order to express this set of inequalities more compactly we introduce a sign notation $s_j = \text{sign}(\mathbf{x}_j^T\mathbf{y})$. This allows us to omit the absolute value signs in the numerator and set up the linear inequalities on polyhedral form. The event that this set of linear inequalities are fulfilled is equivalent to $\Psi\mathbf{y} \leq \mathbf{0}$ with

$$\Psi_1 = \begin{bmatrix} -s_j \frac{\mathbf{x}_j^T}{\|\mathbf{x}_j\|_2} + \frac{\mathbf{x}_{I_{1,1}}^T}{\|\mathbf{x}_{I_{1,1}}\|_2} \\ -s_j \frac{\mathbf{x}_j^T}{\|\mathbf{x}_j\|_2} - \frac{\mathbf{x}_{I_{1,1}}^T}{\|\mathbf{x}_{I_{1,1}}\|_2} \\ \vdots \\ -s_j \frac{\mathbf{x}_j^T}{\|\mathbf{x}_j\|_2} + \frac{\mathbf{x}_{I_{1,p-1}}^T}{\|\mathbf{x}_{I_{1,p-1}}\|_2} \\ -s_j \frac{\mathbf{x}_j^T}{\|\mathbf{x}_j\|_2} - \frac{\mathbf{x}_{I_{1,p-1}}^T}{\|\mathbf{x}_{I_{1,p-1}}\|_2} \end{bmatrix}, \tag{4.4}$$

where $I_{k,i}$ is the ith index from the set of indices of the predictors not yet included in the model at step k, I_k. The order of the rows in Ψ_1 is commutative. The one-step selection matrix Ψ_1 is of dimension $2(p-1)\times n$. When considering the effects in the model after the first step of forward selection, conditioning on this selection event provides valid inferences.

The stepwise construction of the final selection event is necessary because the order in which the predictors enter the model matters when specifying the selection criteria for forward selection. In the following steps, $1 < k < p$, a predictor is chosen to enter the model if and only if it reduces the model RSS more than the addition of any other predictor. In order to account for the entire selection procedure, we need to append $2(p-k)$ rows for each step. Letting I_k denote the set of inactive predictors after step k, $I_k = \{1,\dots,p\} \setminus A_k$,

$$\frac{s_j\varepsilon_j^T\varepsilon}{\|\varepsilon_j\|_2} \geq \pm\frac{\varepsilon_i^T\varepsilon}{\|\varepsilon_i\|_2} \quad \text{for all } i \in I_{k-1},$$

where ε is the residual from regressing $\mathbf{y}$ onto $X_{A_{k-1}}$, $\varepsilon = (I - X_{A_{k-1}}(X_{A_{k-1}}^T X_{A_{k-1}})^{-1}X_{A_{k-1}}^T)\mathbf{y}$, and ε_j is the residual from regressing X_k onto X_{A_k-1}. This is equivalent to

$$\frac{-s_j\varepsilon_j^T\varepsilon}{\|\varepsilon_j\|_2} \pm \frac{\varepsilon_i^T\varepsilon}{\|\varepsilon_i\|_2} \leq 0 \quad \text{for all } i \in I_{k-1},$$

or

$$\frac{-s_j\varepsilon_j^T(I - X_{A_{k-1}}(X_{A_{k-1}}^T X_{A_{k-1}})^{-1}X_{A_{k-1}}^T)\mathbf{y}}{\|\varepsilon_j\|_2} \pm \frac{\varepsilon_i^T(I - X_{A_{k-1}}(X_{A_{k-1}}^T X_{A_{k-1}})^{-1}X_{A_{k-1}}^T)\mathbf{y}}{\|\varepsilon_i\|_2} \leq 0,$$

for all $i \in I_{k-1}$. Setting $P_{A_{k-1}} = I - X_{A_{k-1}}(X_{A_{k-1}}^T X_{A_{k-1}})^{-1} X_{A_{k-1}}^T$ we obtain

$$\frac{-s_j \varepsilon_j^T P_{A_{k-1}} \mathbf{y}}{\|\varepsilon_j\|_2} \pm \frac{\varepsilon_i^T P_{A_{k-1}} \mathbf{y}}{\|\varepsilon_i\|_2} \leq 0,$$

so that

$$\Psi_k = \begin{bmatrix} \frac{-s_j \varepsilon_j^T P_{A_{k-1}}}{\|\varepsilon_j\|_2} + \frac{\varepsilon_{I_{k,1}}^T P_{A_{k-1}}}{\|\varepsilon_{I_{k,1}}\|_2} \\ \frac{-s_j \varepsilon_j^T P_{A_{k-1}}}{\|\varepsilon_j\|_2} - \frac{\varepsilon_{I_{k,1}}^T P_{A_{k-1}}}{\|\varepsilon_{I_{k,1}}\|_2} \\ \vdots \\ \frac{-s_j \varepsilon_j^T P_{A_{k-1}}}{\|\varepsilon_j\|_2} + \frac{\varepsilon_{I_{k,p-k}}^T P_{A_{k-1}}}{\|\varepsilon_{I_{k,p-k}}\|_2} \\ \frac{-s_j \varepsilon_j^T P_{A_{k-1}}}{\|\varepsilon_j\|_2} - \frac{\varepsilon_{I_{k,p-k}}^T P_{A_{k-1}}}{\|\varepsilon_{I_{k,p-k}}\|_2} \end{bmatrix}. \tag{4.5}$$

Above we assumed that $k < p$. In the last possible step of forward selection, when $k = p$, we need only add one row to Ψ. If adding the last possible predictor lowers the RSS of the model, the row

$$\Psi_p = \begin{bmatrix} \frac{-s_j \varepsilon_j^T P_{A_{p-1}}}{\|\varepsilon_j\|_2} + \frac{\varepsilon_{I_{p,1}}^T P_{A_{p-1}}}{\|\varepsilon_{I_{p,1}}\|_2} \end{bmatrix} \tag{4.6}$$

is appended to Ψ. The final selection matrix Ψ has $2pk - k^2 - k$ rows (Tibshirani et al. 2016). Stacking the selection matrices Ψ_j, $j = 1, \ldots, k$ yields the final selection matrix Ψ that is needed in order to apply the polyhedral inference framework to a model chosen by forward selection with k steps. The resulting forward selection event $\{\Psi \mathbf{y} \leq \mathbf{0}\}$ is always a polyhedral region in the $\mathbb{R}^n$ space. Taking advantage of these affine constraints on $\mathbf{y}$ allows us to apply the polyhedral lemma (3.2) and derive p-values and confidence intervals that take the forward selection procedure into account by the scheme presented in Section 3.2.3.

4.1.4 Example of polyhedral inference for forward selection

We present a simple example of how to construct the polyhedral region for forward selection. We simulate a dataset with $p = 3$ predictors, and $n = 80$ observations, where $x_{ij} \sim N(0,1)$ for $i = 1, \ldots, 80$ and $j = 1, 2, 3$ are the entries of the design matrix X. For simplicity of notation, we standardize X by centering and scaling each column. The response vector is generated by assuming a linear relationship between the three predictors and the response,

$$\mathbf{y} = X\boldsymbol{\beta} + \boldsymbol{\varepsilon}, \quad \boldsymbol{\varepsilon} \sim N(0, I),$$

where the vector of true coefficients is $\beta = (0.6, 0.3, 0)^T$. As the design matrix is standardized, the forward selection procedure starts with an empty model with

no intercept. For each step $k = 1, 2, 3$, of this forward selection procedure, we want to summarize the selection criteria that determined which variable should enter the model as affine constraints on the response $\{\Psi_k \mathbf{y} \leq \mathbf{0}\}$. In step $k = 1$, $\mathbf{x}_1$ is chosen to enter the model as it lowers the RSS of the model more than the addition of any of the other predictors,

$$\text{RSS}(\mathbf{y}, \mathbf{x}_1) \leq \text{RSS}(\mathbf{y}, \mathbf{x}_2) \quad \text{and} \quad \text{RSS}(\mathbf{y}, \mathbf{x}_1) \leq \text{RSS}(\mathbf{y}, \mathbf{x}_3).$$

As shown in Section 4.1.3, these inequalities can be reformulated as

$$\frac{|\mathbf{x}_1^T \mathbf{y}|}{\|\mathbf{x}_1\|_2} \geq \frac{|\mathbf{x}_2^T \mathbf{y}|}{\|\mathbf{x}_2\|_2} \quad \text{and} \quad \frac{|\mathbf{x}_1^T \mathbf{y}|}{\|\mathbf{x}_1\|_2} \geq \frac{|\mathbf{x}_2^T \mathbf{y}|}{\|\mathbf{x}_2\|_2},$$

respectively. Introducing $s_1 = \text{sign}(\mathbf{x}_1^T \mathbf{y})$ we have that these inequalities can be written as $\Psi_1 \mathbf{y} \leq \mathbf{0}$, where

$$\Psi_1 = \begin{bmatrix} -s_1 \frac{\mathbf{x}_1^T}{\|\mathbf{x}_1\|_2} + \frac{\mathbf{x}_2^T}{\|\mathbf{x}_2\|_2} \\ -s_1 \frac{\mathbf{x}_1^T}{\|\mathbf{x}_1\|_2} - \frac{\mathbf{x}_2^T}{\|\mathbf{x}_2\|_2} \\ -s_1 \frac{\mathbf{x}_1^T}{\|\mathbf{x}_1\|_2} + \frac{\mathbf{x}_3^T}{\|\mathbf{x}_3\|_2} \\ -s_1 \frac{\mathbf{x}_1^T}{\|\mathbf{x}_1\|_2} - \frac{\mathbf{x}_3^T}{\|\mathbf{x}_3\|_2} \end{bmatrix}.$$

The matrix Ψ_1 has dimensions $2(p-k) \times n = 4 \times 80$. In step $k = 2$, the comparison of model RSS is made in the context of the existing model after step 1, which contains $\mathbf{x}_1$. This means that we need to adjust for $\mathbf{x}_1$ when comparing the drops in model RSS. What we are evaluating now is which of the variables yields a two-variable model with the lowest possible model RSS. We enter $\mathbf{x}_2$ because it lowers the model RSS more than entering $\mathbf{x}_3$,

$$\text{RSS}(\mathbf{y}, X_{[1,2]}) \leq \text{RSS}(\mathbf{y}, X_{[1,3]}).$$

Here $X_{[1,2]} = (\mathbf{x}_1, \mathbf{x}_2)$. This selection criterion can be written as

$$\frac{s_2 \varepsilon_2^T \varepsilon}{\|\varepsilon_2\|_2} \geq \pm \frac{\varepsilon_3^T \varepsilon}{\|\varepsilon_3\|_2}, \tag{4.7}$$

where ε is the residual from regressing $\mathbf{y}$ onto $\mathbf{x}_1$, and ε_2 is the residual from regressing $\mathbf{x}_2$ onto $\mathbf{x}_1$ (Tibshirani et al. 2016). In order to express this set of inequalities by $\Psi_2 \mathbf{y} \leq \mathbf{0}$, we need to isolate $\mathbf{y}$. The residual from regressing $\mathbf{y}$ onto the design matrix from the last step $X_{A_{k-1}}$ is $\varepsilon = (I - X_{A_{k-1}}(X_{A_{k-1}}^T X_{A_{k-1}})^{-1} X_{A_{k-1}}^T)\mathbf{y}$. In this case, $X_{A_1} = \mathbf{x}_1$, so $\varepsilon = (I - \mathbf{x}_1(\mathbf{x}_1^T \mathbf{x}_1)^{-1} \mathbf{x}_1^T)\mathbf{y}$. Setting $P_{A_1} = I - \mathbf{x}_1(\mathbf{x}_1^T \mathbf{x}_1)^{-1} \mathbf{x}_1^T$, we can write Equation (4.7) as $\Psi_2 \mathbf{y} \leq \mathbf{0}$, where

$$\Psi_2 = \begin{bmatrix} \frac{-s_2 \varepsilon_2^T P_{A_1}}{\|\varepsilon_2\|_2} + \frac{\varepsilon_3^T P_{A_1}}{\|\varepsilon_3\|_2} \\ \frac{-s_2 \varepsilon_2^T P_{A_1}}{\|\varepsilon_j\|_2} - \frac{\varepsilon_3^T P_{A_1}}{\|\varepsilon_3\|_2} \end{bmatrix}.$$

In the last step $k = 3$, the remaining candidate predictor $\mathbf{x}_3$ is added simply if adding it lowers the total model RSS. This can be encoded in multiple ways, but we choose to keep the formulation of this criterion similar to the previous ones for simplicity. By a similar argument as for step 2, $\mathbf{x}_3$ is chosen to enter the model in step $k = 3$ if and only if $\Psi_3\mathbf{y} \leq \mathbf{0}$, where

$$\Psi_3 = \left[-s_3\varepsilon_3^T P_{A_2}\right].$$

Stacking the selection matrices Ψ_1, Ψ_2 and Ψ_3 into Ψ gives us our final forward selection event $\{\Psi\mathbf{y} \leq \mathbf{0}\}$ for the model. This polyhedral set fully captures the criteria for the model and the signs of the coefficients to be chosen by forward selection, in other words $\{\Psi\mathbf{y} \leq \mathbf{0}\} = \{\hat{A} = A, \hat{\mathbf{s}} = \mathbf{s}\}$. We implement the selection matrix Ψ in the exact manner described above, and present the code in Appendix A.3.2. Now we have the essential components to apply the polyhedral lemma (3.2) for deriving p-values and confidence intervals that appropriately account for the forward selection procedure by conditioning on the event $\{\Psi\mathbf{y} \leq \mathbf{0}\} = \{\hat{A} = A, \hat{\mathbf{s}} = \mathbf{s}\}$.

In Chapter 3 we went into detail on how to derive a selection adjusted test statistic from the result that a polyhedral selection event $\{\Psi\mathbf{y} \leq \mathbf{b}\}$ can be rewritten as a truncation of the values of a linear contrast of $\mathbf{y}$, $\{\mathcal{V}^-(\mathbf{z}) \leq \boldsymbol{\eta}^T\mathbf{y} \leq \mathcal{V}^+(\mathbf{z}),\ \mathcal{V}^0(\mathbf{z}) > 0\}$. It is shown in Section 3.2.3 that the conditional test statistic (3.7) has a $U(0,1)$ distribution, and can be inverted to obtain confidence intervals of model parameters with $1-\alpha$ coverage conditional on $\{\hat{A} = A, \hat{\mathbf{s}} = \mathbf{s}\}$. For significance testing of the model effects β_1, β_2, and β_3, we will use the truncated Gaussian test statistic

$$F^{[\mathcal{V}^-(\mathbf{z}),\mathcal{V}^+(\mathbf{z})]}_{\boldsymbol{\eta}^T\boldsymbol{\mu},\sigma^2\|\boldsymbol{\eta}\|_2^2}(\boldsymbol{\eta}^T\mathbf{y}) \mid \{\Psi\mathbf{y} \leq \mathbf{0}\} \sim U(0,1), \tag{4.8}$$

where the truncation limits $\mathcal{V}^-(\mathbf{z})$, and $\mathcal{V}^+(\mathbf{z})$ are calculated from

$$\mathcal{V}^-(\mathbf{z}) = \max_{j:(\Psi\mathbf{c})_j<0} \frac{-(\Psi\mathbf{z})_j}{(\Psi\mathbf{c})_j}, \qquad \text{and } \mathcal{V}^+(\mathbf{z}) = \min_{j:(\Psi\mathbf{c})_j>0} \frac{-(\Psi\mathbf{z})_j}{(\Psi\mathbf{c})_j}.$$

We specify $\boldsymbol{\eta} = X(X^TX)^{-1}\mathbf{e}_j$ so that $\boldsymbol{\eta}^T\boldsymbol{\mu} = \beta_j$, and $\boldsymbol{\eta}^T\mathbf{y} = \hat{\beta}_j$ for $j = 1,2,3$. The vectors $\mathbf{z}$ and $\mathbf{c}$ depend on $\boldsymbol{\eta}$ by their definitions,

$$\mathbf{c} = \boldsymbol{\eta}(\boldsymbol{\eta}^T\boldsymbol{\eta})^{-1} \quad \text{and} \quad \mathbf{z} = (I - \mathbf{c}\boldsymbol{\eta}^T)\mathbf{y}.$$

We implement a truncated Gaussian test that conditions on the model and the signs of the coefficients $\{\Psi\mathbf{y} \leq \mathbf{0}\} = \{\hat{A} = A, \hat{\mathbf{s}} = \mathbf{s}\}$. The code for this implementation is provided in Appendix A.3. To obtain selection adjusted confidence intervals, we invert the test as outlined in Section 3.2.3. An interval halving algorithm to find the confidence limits.

We extend our approach by omitting the conditioning on the signs of the coefficients. Since the signs of the estimated coefficients are used in the construction of Ψ, there is a selection matrix $\Psi_{\mathbf{s}}$ for every possible sign pattern. As argued in

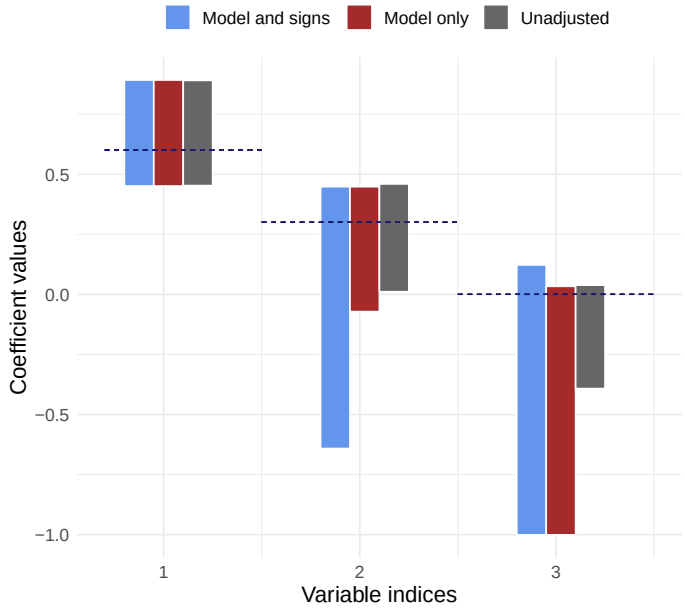


Figure 4.3: Comparison of confidence intervals for coefficients of a model chosen forward selection with $n = 80$, $p = 3$ and $\boldsymbol{\beta} = (0.6, 0.3, 0)^T$. The dashed lines indicates the true signal. Selection adjusted confidence intervals conditioned on $\{\hat{A} = A, \hat{\mathbf{s}} = \mathbf{s}\}$ and $\{\hat{A} = A\}$ are shown alongside unadjusted confidence intervals. The difference in width of the intervals decreases as the magnitude of the underlying signal β_j decreases.

Section 3.2.4 we can condition on only $\{\hat{A} = A\}$ by using the Gaussian test statistic truncated to a union of confidence intervals $\bigcup_{\mathbf{s}}[\mathcal{V}^-_{\mathbf{s}}(\mathbf{z}), \mathcal{V}^+_{\mathbf{s}}(\mathbf{z})]$,

$$F^{\bigcup_{\mathbf{s}}[\mathcal{V}^-_{\mathbf{s}}(\mathbf{z}),\mathcal{V}^+_{\mathbf{s}}(\mathbf{z})]}_{\eta^T\mu,\sigma^2\|\eta\|_2^2}(\eta^T\mathbf{y}) \,|\, \bigcup_{\mathbf{s}}\{\Psi_{\mathbf{s}}\mathbf{y} \leq \mathbf{0}\} \sim U(0,1). \tag{4.9}$$

Inverting this test statistic yields confidence intervals with $1-\alpha$ coverage conditional on the selected model $\{\hat{A} = A\}$ for β_j, $j = 1, 2, 3$.

Figure 4.3 displays that for this particular example, there is no visible difference between the confidence intervals conditioned on $\{\hat{A} = A\}$ and $\{\hat{A} = A, \hat{\mathbf{s}} = \mathbf{s}\}$ for β_1, but they are not identical. However, it is not unreasonable to get two confidence intervals that are exactly identical, because it can occur that all alternative sign patterns $\mathbf{s}$ besides the realized pattern $\hat{\mathbf{s}}$, yields $\mathcal{V}^-_{\mathbf{s}}(\mathbf{z}) \geq \mathcal{V}^+_{\mathbf{s}}(\mathbf{z})$. In these cases,

$$\mathbb{P}(\mathcal{V}^-_{\mathbf{s}}(\mathbf{z}) \leq \boldsymbol{\eta}^T\mathbf{y} \leq \mathcal{V}^+_{\mathbf{s}}(\mathbf{z})) = 0,$$

and hence these intervals do not contribute to the calculation of the test statistic. The confidence intervals for β_2 and β_3 that are conditional on the model only are closer to the unadjusted least squares intervals, but offer the same $1-\alpha$ coverage as the intervals that condition on the model as well as the observed sign pattern.

4.2 Lasso regression

The lasso method was first proposed by Tibshirani (1996) as a method for estimation in linear models. Several extensions has been made since then, and the method frequently taught in statistics courses and used in practical settings. The method combines least squares minimization with an l_1 constraint on the size of the coefficients. The name *lasso* is an acronym for *least absolute shrinkage and selection operator*, and it is also said to be a suitable name since the method constrains the coefficients, in a figurative sense similarly to the way in which a lasso rope is used to throw a loop around horses and cattle (Hastie et al. 2015).

The lasso produces sparse solutions as it automatically shrinks some effects to zero, making it a popular tool for model selection. The lasso optimization problem is convex, and always has at least one minimizer (Hastie et al. 2015). We will consider the case in which the lasso is used as a selection tool for linear models. The exact lasso estimates are not used directly, but the model choice is defined by the variables chosen to have non-zero effects by the lasso.

We give a brief introduction to the lasso for linear models and derive the general polyhedral selection event for the lasso with fixed λ. The formulation of lasso selection as a polyhedral event allows for the use of the polyhedral inference method for deriving a truncated Gaussian test and corresponding selection adjusted confidence intervals.

4.2.1 Introduction to the lasso for linear models

We consider the linear regression setting presented in Section 2.1. We assume that the response $\mathbf{y}$ is centered, and that the design matrix X is standardized, meaning that

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

The centering and standardization of the data allows us to omit the intercept β_0 from the lasso optimization problem. The lasso problem is the least squares problem with an added linear constraint on the absolute size of the coefficients,

$$\min \sum_{i=1}^{n}\left(y_i - \sum_{j=1}^{p} x_{ij}\beta_j\right)^2 \quad \text{subject to } \sum_{j=1}^{p}|\beta_j| \leq t. \tag{4.10}$$

This formulation is typically referred to the *constrained* form or the *budget* form of the lasso problem. The lasso problem can be written in its equivalent Lagrangian form,

$$\hat{\boldsymbol{\beta}}^L = \arg\min_{\boldsymbol{\beta}} \frac{1}{2}\|\mathbf{y} - X\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_1, \quad \lambda > 0 \tag{4.11}$$

(Hastie et al. 2009). There is a one-to-one relation between the budget parameter t from the constrained lasso problem (4.10) and the λ parameter from (5.1). This

means that for any value of t such that $\sum_{j=1}^{p}|\beta_j| \leq t$, there exists a value of λ that yields the same solution $\hat{\boldsymbol{\beta}}^L$.

The lasso problem (5.1) is a quadratic programming problem with a convex constraint (Hastie et al. 2015). In general, there is no closed form solution of the minimization problem (5.1). The only cases in which we can find $\hat{\boldsymbol{\beta}}^L$ analytically is the case of only one or two covariates, or when the design matrix X is orthogonal, $X^TX = (X^TX)^{-1} = I$ (Tibshirani 1996). The issue of solving quadratic programs is a well-explored topic in optimization with many solutions. The necessary and sufficient conditions for a solution to (5.1) can be summarized as

$$\langle \mathbf{x}_j, \mathbf{y} - X\boldsymbol{\beta} \rangle + s_j\lambda = 0, \qquad \text{for all } j = 1, \ldots, p \tag{4.12}$$

(Hastie et al. 2015). Here s_j is defined by

$$\begin{aligned} s_j &= \operatorname{sgn} \hat{\beta}_j^L \qquad \text{if } \hat{\beta}_j^L \neq 0 \\ s_j &\in [-1, 1] \qquad \text{if } \hat{\beta}_j^L = 0. \end{aligned}$$

In the latter case, when $\hat{\beta}_j^L = 0$, s_j is a subgradient of the absolute value function (Hastie et al. 2015). The penalty parameter $\lambda > 0$ is continuous. As each choice of λ yields a new solution for the lasso estimator $\hat{\boldsymbol{\beta}}^L$, we note that the lasso regression estimator is a sequence of estimators for $\boldsymbol{\beta}$. λ is often chosen by cross-validation. In the following subsections, we will focus on inference after lasso selection for fixed values of λ.

Larger values of λ, or equivalently, smaller values of t imply stricter penalization of the size of the regression coefficients, and hence the possibility of more sparse models as more coefficients are set to zero. For small enough λ or large enough t, the lasso estimate $\hat{\boldsymbol{\beta}}^L$ is equal to the least squares estimate. More specifically, this is the case when

$$t \geq \sum_{j=1}^{p} |\hat{\beta}_j^L|,$$

or equivalently, when $\lambda = 0$. Figure 4.4 gives an intuitive explanation of the estimation of the lasso estimator. For larger values of t, the constraint region drawn red expands, and can come to contain the least squares estimate $\hat{\boldsymbol{\beta}}$ itself, which is the minimum of the RSS function illustrated with blue contours.

4.2.2 Polyhedral selection event for the lasso

In this approach, we consider the lasso as a selection tool for linear regression. Hence, the exact lasso estimates (5.1) are not used directly for inference, but is used to determine which variables should be included in the model. Analogously to Lee et al. (2016), we define a model chosen by the lasso by the indices corresponding to the coefficients of the lasso estimator (5.1) that are different from 0,

$$\hat{A} = \{j : \hat{\beta}_j^L \neq 0\}$$

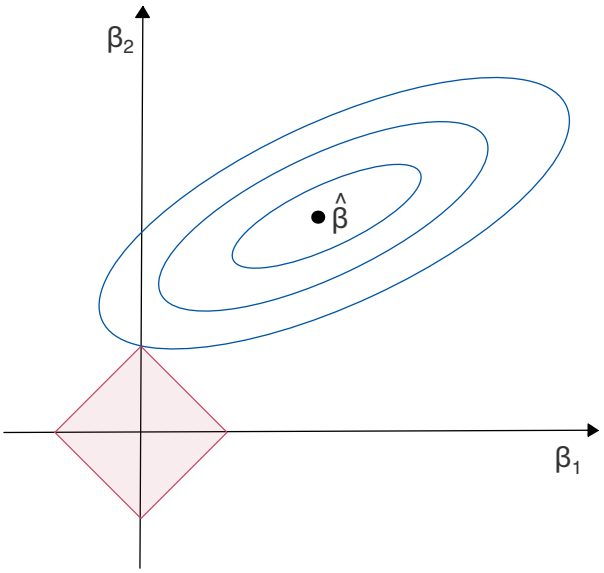


Figure 4.4: Geometric interpretation of the lasso. The constraint region $|\beta_1| + |\beta_2| \leq t$ is marked in red, and the contours of the RSS in blue. Inspired by and drawn after Figure 6.7 of James et al. (2013).

We assume that the design matrix X has full rank, rank $X = p$. In this case, the lasso solution is unique because the criterion is strictly convex (Tibshirani 2013). The Karush–Kuhn–Tucker (KKT) conditions are sufficient and necessary conditions for the existence of a lasso solution. We can reformulate the KKT conditions (4.12) as

$$X^T(X\hat{\boldsymbol{\beta}}^L - \mathbf{y}) + \lambda\hat{\mathbf{s}} = 0, \tag{4.13}$$

where s_j is defined by

$$s_j = \operatorname{sgn} \hat{\beta}_j^L \qquad \text{if } \hat{\beta}_j^L \neq 0, \tag{4.14}$$

$$s_j \in [-1, 1] \qquad \text{if } \hat{\beta}_j^L = 0 \tag{4.15}$$

(Lee et al. 2016).

We use the KKT conditions directly to derive the selection event of the lasso procedure. In this section we characterize the lasso selection event $\{\hat{A} = A, \hat{\mathbf{s}} = \mathbf{s}\}$. This selection event corresponds to a polyhedral region in the $\mathbb{R}^n$-plane. This fact allows us to use the polyhedral inference method presented in Chapter 3 to obtain conditional p-values and confidence intervals for the model effects that account for the lasso selection procedure. The characterization of the event $\{\hat{A} = A\}$ will be done by taking the union over the possible sign patterns for the model. This is expanded upon in Section 4.2.3.

In order to characterize the lasso selection event $\{\hat{A} = A, \hat{\mathbf{s}} = \mathbf{s}\}$, we first use

the definitions define the model selection $\hat{A}$ implicitly by an *equicorrelation* set

$$\hat{A} \equiv \{i \in \{1, \dots, p\} : |\hat{s}_i| = 1\}$$

which contains all the predictors with non-zero lasso coefficients, since $|\hat{s}_i| = 1$ for any i such that $\hat{\beta}_i^L \neq 0$ (Lee et al. 2016). It is not impossible for the equicorrelation set to include variables with lasso coefficients set to zero, this happens for almost no values of λ, and we do not consider it further. We adapt and expand upon the method from pp. 912-914 of Lee et al. (2016) for deriving the general lasso selection event $\{\hat{A} = A, \hat{\mathbf{s}} = \mathbf{s}\}$.

Let $X_{\hat{A}}$ be the submatrix of X consisting of the columns of X corresponding to the indices in the equicorrelation set $\hat{A}$. Correspondingly, we define $X_{\hat{A}^C}$ to be the submatrix of X consisting of the columns corresponding to the indices in $\hat{A}^C = \{1, \dots, p\} \setminus \hat{A}$. The KKT conditions (4.13) can be reformulated by using this partitioning of the design matrix X, as the conditions must imply that

$$X_{\hat{A}}^T (X_{\hat{A}} \hat{\boldsymbol{\beta}}_{\hat{A}}^L - \mathbf{y}) + \lambda \hat{\mathbf{s}}_{\hat{A}} = 0, \tag{4.16}$$

$$\text{where } \hat{\mathbf{s}}_{\hat{A}} = \text{sign}(\hat{\boldsymbol{\beta}}_{\hat{A}}^L), \tag{4.17}$$

and that

$$X_{\hat{A}^C}^T (X_{\hat{A}} \hat{\boldsymbol{\beta}}_{\hat{A}}^L - \mathbf{y}) + \lambda \hat{\mathbf{s}}_{\hat{A}^C} = 0, \tag{4.18}$$

$$\text{with } \|\hat{\mathbf{s}}_{\hat{A}^C}\|_\infty < 1. \tag{4.19}$$

We note that the condition in (4.19) implies that no subgradient s_j from (4.15) is exactly 1 in absolute value. The KKT conditions are necessary and sufficient conditions for a lasso solution. Any lasso solution consisting of a model choice A and sign pattern $\mathbf{s}$ must necessarily fulfill the conditions above, and hence also fulfills

$$X_A^T (X_A \hat{\boldsymbol{\beta}}_A^L - \mathbf{y}) + \lambda \mathbf{s} = 0, \tag{4.20}$$

$$\mathbf{s} = \text{sgn } \hat{\boldsymbol{\beta}}_A^L, \tag{4.21}$$

$$X_{A^C}^T (X_A \hat{\boldsymbol{\beta}}_A^L - \mathbf{y}) + \lambda \mathbf{s}_{A^C} = 0, \tag{4.22}$$

$$\|\mathbf{s}_{A^C}\|_\infty < 1. \tag{4.23}$$

Previously we assumed that X has full rank. This implies that $\text{rank}(X_A) = |A|$. Then, for any $\mathbf{y}$, X, and $\lambda > 0$, the lasso solution is unique. For an equicorrelation set A and a sign pattern $\mathbf{s}$, the lasso solution is given by

$$\hat{\boldsymbol{\beta}}_A^L = (X_A^T X_A)^{-1} (X_A^T \mathbf{y} - \lambda \mathbf{s}), \quad \text{and } \hat{\boldsymbol{\beta}}_{A^C}^L = 0$$

(Tibshirani (2013), p. 1462). This solves equation (4.20). Solving equation (4.22) for $\mathbf{s}_{A^C}$ yields

$$\begin{aligned} \mathbf{s}_{A^C} &= X_{A^C}^T X_A (X_A^T X_A)^{-1} \mathbf{s} + \frac{1}{\lambda} X_{A^C}^T (I - P_A) \mathbf{y} \\ &= X_{A^C}^T (X_A^T)^+ \mathbf{s} + \frac{1}{\lambda} X_{A^C}^T (I - P_A) \mathbf{y}, \end{aligned} \tag{4.24}$$

where $P_A = X_A(X_A^T X_A)^{-1} X_A$ is the projection onto the column span of X_A (Lee et al. 2016). With these definitions of $\hat{\boldsymbol{\beta}}_A^L$ and $\mathbf{s}_{A^C}$, along with their respective conditions

$$\mathbf{s} = \text{sign}(\hat{\boldsymbol{\beta}}_A^L), \quad \text{and } \|\mathbf{s}_{A^C}\|_\infty < 1,$$

we can rewrite the model selection in terms of $\hat{\boldsymbol{\beta}}_A^L$ and $\mathbf{s}_{A^C}$ as

$$\{\hat{A} = A, \hat{\mathbf{s}} = \mathbf{s}\} = \{\mathbf{s} = \text{sign}(\hat{\boldsymbol{\beta}}_A^L), \|\mathbf{s}_{A^C}\|_\infty < 1\}. \tag{4.25}$$

We wish to show that this formulation of the lasso selection event is equivalent to a polyhedral region in the $\mathbb{R}^n$-plane, or equivalently a set of affine constraints on $\mathbf{y}$, $\{\Psi\mathbf{y} \leq \mathbf{b}\}$.

$$\begin{aligned}\{\mathbf{s} = \text{sign}(\hat{\boldsymbol{\beta}}_A^L)\} =& \{(\text{diag } \mathbf{s})\hat{\boldsymbol{\beta}}_A^L > 0\} \\ =& \{(\text{diag } \mathbf{s})(X_A^T X_A)^{-1}(X_A^T \mathbf{y} - \lambda \mathbf{s}) > 0\}\end{aligned}$$

By defining a matrix Ψ_1 and vector $\mathbf{b}_1$ as

$$\Psi_1 = -(\text{diag } \mathbf{s})(X_A^T X_A)^{-1} X_A^T \tag{4.26}$$

$$\mathbf{b_1} = -\lambda\,(\text{diag } \mathbf{s})(X_A^T X_A)^{-1}\mathbf{s}, \tag{4.27}$$

we can write

$$\{(\text{diag } \mathbf{s})(X_A^T X_A)^{-1}(X_A^T \mathbf{y} - \lambda \mathbf{s}) > 0\} = \{\Psi_1 \mathbf{y} < \mathbf{b}_1\}.$$

This makes up the *active* constraints on $\mathbf{y}$. For the *inactive* constraints $\{\|\mathbf{s}_{A^C}\|_\infty < 1\}$, we insert the expression (4.24) and expand

$$\begin{aligned}\{\|\mathbf{s}_{A^C}\|_\infty < 1\} =& \{\|X_{A^C}^T (X_A^T)^+ \mathbf{s} + \frac{1}{\lambda} X_{A^C}^T (I - P_A)\mathbf{y}\|_\infty < 1\} \\ =& \{-\mathbf{1} < X_{A^C}^T (X_A^T)^+ \mathbf{s} + \frac{1}{\lambda} X_{A^C}^T (I - P_A)\mathbf{y} < \mathbf{1}\} \\ =& \begin{Bmatrix} \frac{1}{\lambda} X_{A^C}^T (I - P_A)\mathbf{y} < \mathbf{1} - X_{A^C}^T (X_A^T)^+ \mathbf{s}, \\ -\frac{1}{\lambda} X_{A^C}^T (I - P_A)\mathbf{y} < \mathbf{1} + X_{A^C}^T (X_A^T)^+ \mathbf{s} \end{Bmatrix}\end{aligned}$$

By defining

$$\Psi_0 = \frac{1}{\lambda}\begin{pmatrix} X_{A^C}^T (I - P_A) \\ -X_{A^C}^T (I - P_A) \end{pmatrix}, \tag{4.28}$$

$$\mathbf{b}_0 = \begin{pmatrix} \mathbf{1} - X_{A^C}^T (X_A^T X_A)^{-1} X_A^T \mathbf{s} \\ \mathbf{1} + X_{A^C}^T (X_A^T X_A)^{-1} X_A^T \mathbf{s} \end{pmatrix}, \tag{4.29}$$

we are able to write the inactive constraint as

$$\{\|\mathbf{s}_{A^C}\|_\infty < 1\} = \{\Psi_0 \mathbf{y} < \mathbf{b}_0\}.$$

The final selection event, capturing the entire lasso selection procedure is

$$\{\hat{A} = A, \hat{\mathbf{s}} = \mathbf{s}\} = \{\Psi \mathbf{y} \leq \mathbf{b}\}, \tag{4.30}$$

where

$$\Psi = \begin{pmatrix} \Psi_0 \\ \Psi_1 \end{pmatrix}, \qquad \mathbf{b} = \begin{pmatrix} \mathbf{b_0} \\ \mathbf{b_1} \end{pmatrix}, \tag{4.31}$$

with Ψ_1, $\mathbf{b}_1$,Ψ_0, and $\mathbf{b}_0$, defined as in Equations (4.26), (4.27), (4.28) and (4.29), respectively. The polyhedral lasso event fully captures the the lasso selection procedure for a fixed value of the penalty parameter λ. This allows us to perform selection adjusted inference at any point of the lasso path, by use of the polyhedral method from Chapter 3. The order in which the predictors enter the model on the lasso path does not matter in the final selection event. This is different from the selection event for forward selection, which must be constructed with attention to the order of inclusion of each variable.

When using the polyhedral method to calculate selection adjusted p-values and confidence intervals for a model selected by the lasso, Ψ and $\mathbf{b}$ are necessary components. We remark that the lasso selection matrix Ψ and vector $\mathbf{b}$ only depend on the submatrices X_A, X_{A^C}, λ, and the sign pattern $\mathbf{s}$ of the active lasso coefficients. This makes implementation straightforward, as this information is readily available after a model is chosen by lasso regression.

4.2.3 Extending conditioning to a union of lasso polyhedra

In the previous section, we used the necessary and sufficient conditions of a lasso solution to derive a concise characterization of the event that a particular model is selected by the lasso, along with its sign pattern, $\{\hat{A} = A, \hat{\mathbf{s}} = \mathbf{s}\} =$. It was shown that this lasso selection event can be described by a polyhedral set $\{\Psi \mathbf{y} \leq \mathbf{b}\}$. A key focus of this thesis is the extension to conditioning solely on the model selected by the lasso, $\{\hat{A} = A\}$, omitting the conditioning on the sign pattern $\mathbf{s}$. As Ψ and $\mathbf{b}$ depend on $\mathbf{s}$, there exists a unique $\Psi_\mathbf{s}$ and $\mathbf{b}_\mathbf{s}$ for every possible sign pattern. The more general lasso selection event $\{\hat{A} = A\}$ is defined by the union of sign conditional lasso polyhedra over all possible sign patterns,

$$\{\hat{A} = A\} = \bigcup_{\mathbf{s}} \left\{\Psi_\mathbf{s} \mathbf{y} \leq \mathbf{b}_\mathbf{s}\right\}$$

(Lee et al. 2016). What is important to note is that not all sign patterns are possible for a given model. The lasso partitions the sample space $\mathbf{y} \in \mathbb{R}^n$ into polyhedra $\{\hat{A} = A, \hat{\mathbf{s}} = \mathbf{s}\} = \{\Psi \mathbf{y} \leq \mathbf{b}\}$ (Lee et al. 2016). Figure 4.5 depicts a geometrical interpretation of the partition the lasso makes of the space in the case of $n = 2$ observations and $p = 3$ predictors. Every possible value of $\mathbf{y}$ corresponds to a

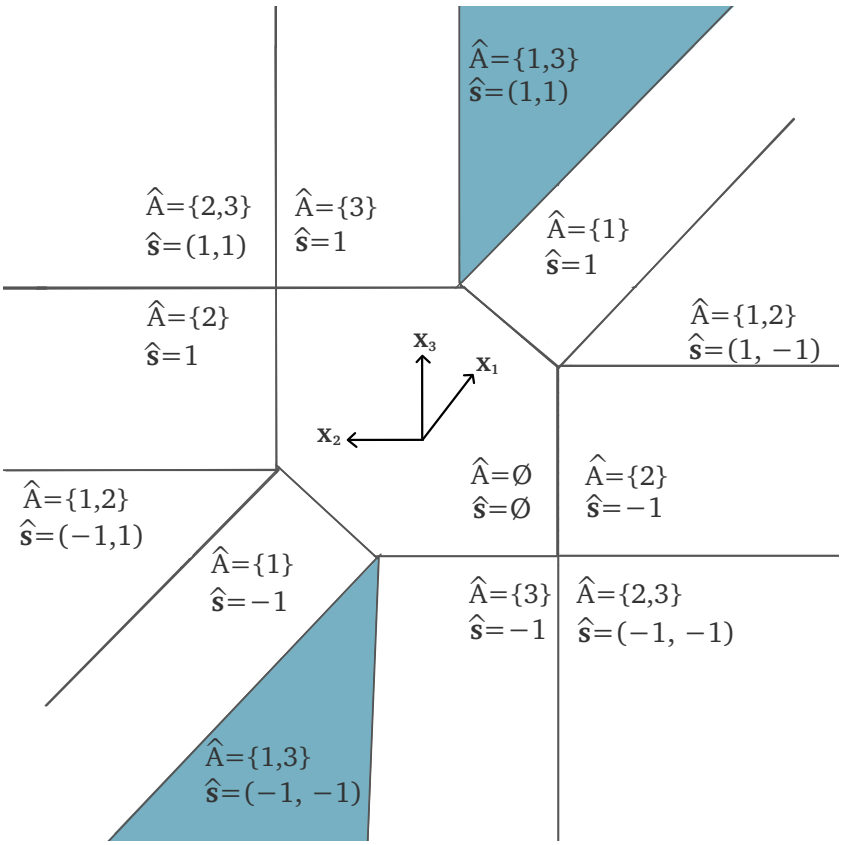


Figure 4.5: Illustration of the partition of the $\mathbb{R}^2$ sample space according to model and sign pattern by the lasso for $n = 2$ and $p = 3$. Here $\mathbf{x}_1 = (\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2})^T$, $\mathbf{x}_2 = (-1, 0)^T$, and $\mathbf{x}_3 = (0, 1)^T$. Adapted and drawn with inspiration from Lee et al. (2016) and Kivaranovic and Leeb (2021).

model $\hat{A}$ and a sign pattern $\hat{\mathbf{s}}$, but the converse is not true. For example, in Figure 4.5, the model $\hat{A} = \{1, 3\}$ cannot be chosen along with the sign patterns $\hat{\mathbf{s}} = (-1, 1)$ or $\hat{\mathbf{s}} = (1, -1)$. This is a direct reflection of the nature of the shrinkage of the coefficients by the lasso. In the center of the figure, we see that the selected model is the empty model, $\hat{A} = \emptyset, \hat{\mathbf{s}} = \emptyset$. In this case, the penalty parameter λ is set to a value so large that no predictors are added to the model. For a smaller value of λ, this polyhedral region is smaller. For $\lambda = 1$, the faces of the region of the empty model touch the unit vectors.

We implement a solution in R that automatically constructs the selection event for a model selected by lasso, and performs a selection adjusted test conditional on the active variables and the chosen sign pattern. The code is provided in Appendix A.4, along with an extension that allows the construction of confidence intervals conditioned on the union of lasso polyhedra for all possible sign patterns. A natural concern might be how to identify the impossible sign patterns for a given model, in order to exclude them from the calculation of the test statistic

and confidence intervals conditioned on the model only. It turns out that this is not necessary, and that their inclusion even tend to have some very advantageous properties for the confidence intervals conditional on $\{\hat{A} = A\}$, at the cost of more computation.

When calculating truncation intervals $[\mathcal{V}_{\mathbf{s}}^-(\mathbf{z}), \mathcal{V}_{\mathbf{s}}^+(\mathbf{z})]$ for the $|A|$ sign patterns, some of which that cannot happen for any $\mathbf{y}$, it is not uncommon that we get some $\mathcal{V}_{\mathbf{s}}^-(\mathbf{z})$, $\mathcal{V}_{\mathbf{s}}^+(\mathbf{z})$, and p-values that may look suspicious at first glance. This is of no consequence to the final p-values and confidence intervals conditioned on the union of the polyhedral events $\bigcup_{\mathbf{s}} \{\Psi_{\mathbf{s}}\mathbf{y} \leq \mathbf{b}_{\mathbf{s}}\}$. If $\mathcal{V}_{\mathbf{s}}^-(\mathbf{z}) \geq \mathcal{V}_{\mathbf{s}}^+(\mathbf{z})$, the values for the particular sign pattern are omitted from the calculation of the truncated Gaussian test statistic

$$F^{\bigcup_{\mathbf{s}}[\mathcal{V}_{\mathbf{s}}^-(\mathbf{z}), \mathcal{V}_{\mathbf{s}}^+(\mathbf{z})]}_{\boldsymbol{\eta}^T\boldsymbol{\mu}, \sigma^2\|\boldsymbol{\eta}\|_2^2}(\boldsymbol{\eta}^T\mathbf{y}).$$

This is handled in the code implementation. One should note that obtaining $\mathcal{V}_{\mathbf{s}}^-(\mathbf{z}) = -\infty$ or $\mathcal{V}_{\mathbf{s}}^+(\mathbf{z}) = \infty$ for one or more sign patterns is reasonable, and can occur even when conditioning on $\{\hat{A} = A, \hat{\mathbf{s}} = \mathbf{s}\}$. When $\bigcup_{\mathbf{s}}[\mathcal{V}_{\mathbf{s}}^-(\mathbf{z}), \mathcal{V}_{\mathbf{s}}^+(\mathbf{z})]$ is unbounded from above and below, the expected length of the conditional confidence interval is finite (Kivaranovic and Leeb 2021). In Section 4.2.4, two examples of polyhedral inference for the lasso are presented. The confidence intervals conditioned on $\{\hat{A} = A, \hat{\mathbf{s}} = \mathbf{s}\}$ are compared to the ones omitting the conditioning on the sign patterns of the lasso coefficients. In Section 5.1, the properties of these confidence intervals, specifically their expected length, is discussed further.

4.2.4 Examples of polyhedral inference for lasso

Now that we have described a general scheme for constructing polyhedral events for the lasso, we have what we need to use the polyhedral lemma and derive test statistics and confidence intervals. For illustrative purposes, we simulate two example datasets and perform lasso variable selection. We simulate two examples, one with $n > p$ and one with $n < p$. In both examples, we keep λ fixed, and fit a lasso regression model using the `glmnet` package. With the described approach from Section 4.2.2 and 4.2.3, we calculate selection adjusted confidence intervals conditioned on the model and the signs of the coefficients, as well as intervals conditioned on the model only. We compare these intervals to the unadjusted least squares intervals, that do not take into account the adaptive selection of the lasso.

As in the forward selection example presented in Section 4.1.4, the entries of the design matrix X are simulated by $x_{ij} \sim N(0,1)$ for $i = 1, \ldots, n$ and $j = 1, \ldots, p$. We standardize and scale X, and generate the response $\mathbf{y}$ from the linear relationship

$$\mathbf{y} = X\boldsymbol{\beta} + \boldsymbol{\varepsilon}, \quad \boldsymbol{\varepsilon} \sim N(0, I).$$

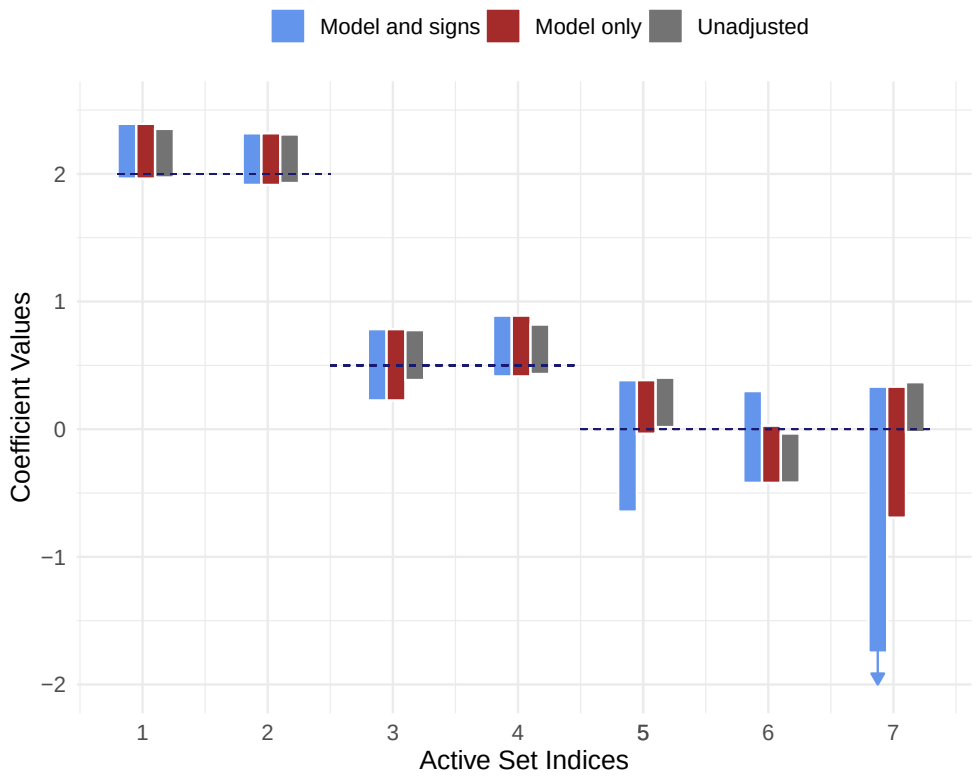


Figure 4.6: Comparison of confidence intervals for model coefficients in a linear model chosen by the lasso with fixed $\lambda = 12$, conditioned on the model and signs,$\{\hat{A} = A, \hat{\mathbf{s}} = \mathbf{s}\}$ and only the model $\{\hat{A} = A\}$. The simulated data has $n = 100$ and $p = 16$. Formatting inspired by Lee et al. (2016). The dashed line shows the true signal. The blue arrow indicates that the lower bound of the confidence interval cannot be computed, and defaults to $-\infty$.

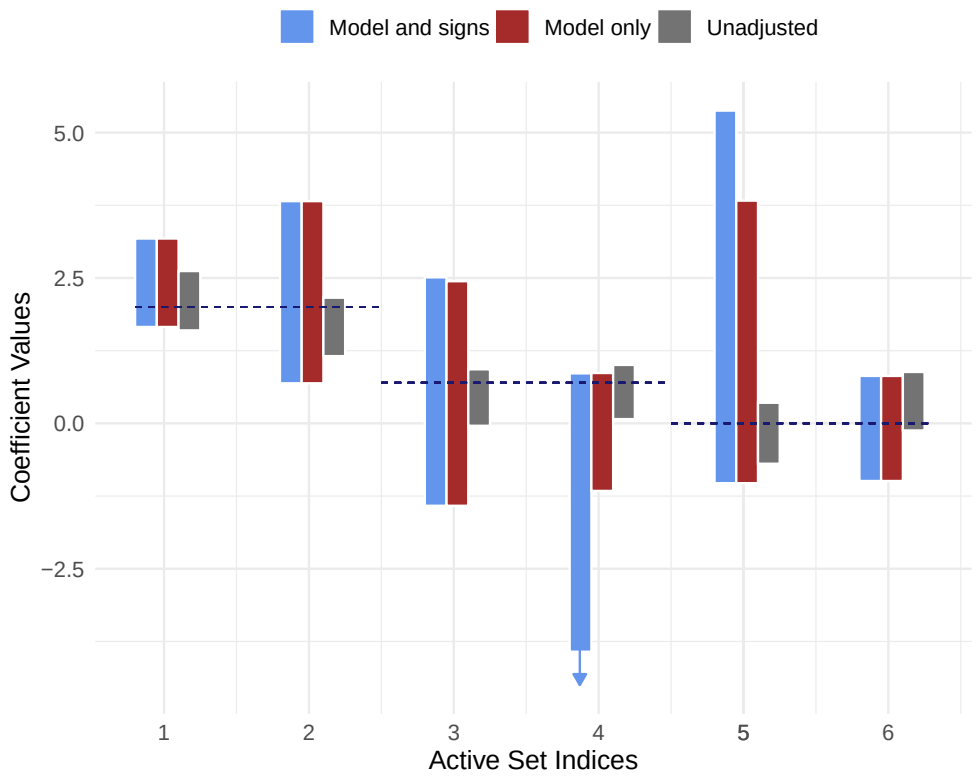


Figure 4.7: Comparison of confidence intervals for model coefficients in a linear model chosen by the lasso with fixed $\lambda = 12$, conditioned on the model and signs$\{\hat{A} = A, \hat{\mathbf{s}} = \mathbf{s}\}$ and only the model $\{\hat{A} = A\}$. The simulated data has $n = 25$ and $p = 25$. Formatting inspired by Lee et al. (2016). The dashed line shows the true signal. The blue arrow indicates that the lower bound of the confidence interval cannot be computed, and defaults to $-\infty$.

We observe that for strong signals, there is no significant difference between confidence intervals that condition on the model as well as the signs of the coefficients, compared to the ones conditioned on the model only. For weaker signals, there can be larger differences in the width of the confidence intervals, and even intervals with infinite length. This usually happens when the observed test statistic $\boldsymbol{\eta}^T\mathbf{y}$ is very close to one of the truncation limits $\mathcal{V}^-(\mathbf{z})$, $\mathcal{V}^+(\mathbf{z})$. Figure 3.2 illustrates that the confidence interval $\text{CI}_j = [L, U]$ is found from a monotonically decreasing test statistic. In order to obtain the confidence interval, we need the statistic to be defined and continuous around $1-\frac{\alpha}{2}$ and $\frac{\alpha}{2}$. When the observed $\boldsymbol{\eta}^T\mathbf{y}$ is too close to the truncation limits, the search becomes unstable, and the interval cannot be computed because the test statistic used to calculate the confidence intervals is not defined at one of the critical points (Tibshirani et al. 2022). This aligns with theoretical results from Kivaranovic and Leeb (2021) on the expected length of the confidence intervals from the polyhedral inference method, which we discuss in further detail in Section 5.1.1.

In Figure 4.6 we see that for indices 5 and 6 of the active set, the unadjusted least squares intervals did not cover the true signal in the simulated data, while both versions of the selection adjusted intervals did. In the case of $n \gg p$ the selection adjusted confidence intervals are usually close to the unadjusted intervals for strong signals. Figure 4.7 shows that this is not the case for $n < p$, where there are larger differences between the intervals that adjust for selection and the unadjusted intervals for both strong and weak signals in the data. This suggests that we stand to gain more from the use of the polyhedral inference method after lasso selection when we have few observations compared to predictors.

Chapter 5

Discussion

In this chapter, we discuss the properties and limitations of the polyhedral inference framework. We clarify the assumptions we have relied on throughout this thesis, and acknowledge existing generalizations of the polyhedral inference approach, as well as related approaches to obtaining valid post-selection inferences.

5.1 Properties of selection adjusted confidence intervals

5.1.1 Width of selection adjusted confidence intervals

Selection adjusted confidence intervals are generally wider than least squares confidence intervals that do not take model selection into account. We have seen from the examples in Figures 4.6 and 4.7 that we at times obtain very wide confidence intervals when conditioning on $\{\hat{A} = A, \hat{\mathbf{s}} = \mathbf{s}\}$. When the truncated Gaussian random variable $\boldsymbol{\eta}^T\mathbf{y}$ is very close to the endpoints of the truncation interval $[\mathcal{V}_{\mathbf{s}}^-(\mathbf{z}), \mathcal{V}_{\mathbf{s}}^+(\mathbf{z})]$, we get wide confidence intervals because there are many values of $\boldsymbol{\eta}^T\boldsymbol{\mu}$ that are consistent with the observation of $\boldsymbol{\eta}^T\mathbf{y}$ (Lee et al. 2016). This usually occurs when the signal is weak or null. For strong signals, the observed statistic $\boldsymbol{\eta}^T\mathbf{y}$ is usually far from both $\mathcal{V}_{\mathbf{s}}^-(\mathbf{z})$ and $\mathcal{V}_{\mathbf{s}}^+(\mathbf{z})$, giving relatively narrow confidence intervals with $1-\alpha$ coverage conditional on $\{\hat{A} = A, \hat{\mathbf{s}} = \mathbf{s}\}$.

Conditioning on more leads to wider confidence intervals and less powerful tests (Tibshirani et al. 2016). Omitting the conditioning on the sign pattern $\mathbf{s}$ yields shorter confidence intervals with the same $1-\alpha$ coverage. The option of computing p-values and confidence intervals conditional only on $\{\hat{A} = A\}$ is not available in the `selectiveInference` library in R (Tibshirani et al. 2022). Hence, extending our approach to cover this was a natural focus point for this thesis.

The confidence intervals conditional on $\{\hat{A} = A, \hat{\mathbf{s}} = \mathbf{s}\}$ are shown by Kivaranovic and Leeb (2021) to have infinite expected length. However, confidence intervals with infinite expected length can still have a high probability of turning out short, or of appropriate lengths (Kivaranovic and Leeb 2021). It is also shown that under a set of necessary and sufficient conditions, the confidence intervals conditional on only $\{\hat{A} = A\}$ share the property of infinite expected length. When these

conditions are not fulfilled, the expected length of these intervals is finite. From our examples in Figures 4.6 and 4.7 we cannot determine the expected length of the different types of confidence intervals, but we do observe two instances of intervals that go to $-\infty$ when conditioned on $\{\hat{A} = A, \hat{\mathbf{s}} = \mathbf{s}\}$, that are finite when omitting the conditioning on the realized sign pattern. While this does not make any guarantees regarding length properties of the confidence intervals, the realizations shown in the examples align with the general results presented by Kivaranovic and Leeb (2021).

5.1.2 Unbiased confidence intervals

The confidence intervals $\text{CI}_j = [L, U]$ for β_j found from test inversion of the selection adjusted test statistic (3.11) have at least $1-\alpha$ coverage. While many alternative confidence intervals will share this property, CI_j is one of the shortest possible alternatives with this conditional coverage, which is preferable. Unbiasedness is also valued when evaluating confidence intervals. An unbiased confidence interval CI for a parameter θ is an interval which covers no other parameter θ_I with probability greater than $1-\alpha$, in other words,

$$\mathbb{P}(\theta_I \in \text{CI}) \leq 1-\alpha \quad \text{for all } \theta_I \neq \theta$$

(Lee et al. 2016). Fithian et al. (2014) show that the CI_j conditional on $\{\hat{A} = A\}$ from the polyhedral inference framework are close to the shortest unbiased intervals for β_j among all intervals with the same $1-\alpha$ coverage (Lee et al. 2016).

5.2 Generalizations

5.2.1 Extension to unknown variance

Throughout this thesis we have assumed that the variance σ^2 is known. In practice, this is rarely the case. In the simulated examples in Sections 4.1.4 and 4.2.4, the variance is known to be $\sigma^2 = 1$. In our implementations in R, we have relied on the knowledge of the value of σ^2, and have not generalized to cover unknown variances. The `selectiveInference` library estimates the variance, and uses this estimate as a known variance. In the cases of more observations than predictors, $n > p$, the estimated variance $\widehat{\sigma^2}$ of the residuals from fitting the full model is a consistent estimator of σ^2 (Lee et al. 2016). In settings where $p > n$, estimating σ^2 is difficult. Tibshirani et al. (2018) discuss the asymptotics of selective inference in further detail, and suggest an efficient bootstrap approach for when σ^2 is unknown.

5.2.2 Further extensions

Variable selection in the linear model with Gaussian errors is often chosen as the target of post-selection inference partly because it is widely used, and partly because it is one of the most familiar examples of model selection. There exist many

methods for variable selection for linear regression. Forward selection with a fixed number of steps k, the lasso with fixed λ, and least angular regression are known to be compatible with the polyhedral method (Tibshirani et al. 2016). There still remain many interesting generalizations of conditional inference beyond the linear model, and some of these are explored in literature and implementations. The R package `selectiveInference` contains extensions of the polyhedral inference framework to logistic regression, the graphical lasso, and the Cox proportional hazards model (Tibshirani et al. 2022).

Hyun et al. (2018) extends the polyhedral inference framefork to inference conditioned on model selection events defined by the generalized lasso path. The generalized lasso estimate is given as

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \frac{1}{2}\|\mathbf{y} - X\boldsymbol{\beta}\|_2^2 + \lambda\|D\boldsymbol{\beta}\|_1, \quad \lambda \geq 0 \tag{5.1}$$

where $D \in \mathbb{R}^{m\times p}$ is a penalty matrix and $m \leq n$. The inference tools presented in the article holds for any penalty matrix D, and the method has many possible applications.

5.3 Related approaches to post-selection inference

5.3.1 The covariance test

The covariance test for the lasso is proposed by Lockhart et al. (2014) as a solution for valid significance testing of variables that enter a linear model along the lasso solution path. When the true model is linear, the covariance test statistic has an Exp(1) distribution under the null hypothesis that all significant predictors are already contained in the current lasso model (Lockhart et al. 2014). The covariance test is based on the least angular regression algorithm. It takes advantage of the knots of the lasso, which are the values of λ for which the set of active predictors changes. The covariance test statistic measures how much of the covariance between the fitted model and the outcome is associated with the predictor that was last entered by the lasso. For forward stepwise selection, the $R_{k,j}$ statistic (4.2) is the covariance test statistic (Hastie et al. 2015).

5.3.2 Simultaneous inference

A central requirement of the conditional approach to post selection inference is that the model selection is realized by the use of well-defined selection methods that can be specified by statistical events. It is impossible to condition on a selection procedure that can not be summarized mathematically. In practice, it is often the case that a model is selected partially by informal selection methods, e.g. by inspection of visual representations of the data or post hoc considerations. A post-selection inference approach that addresses this challenge is simultaneous inference.

The simultaneous inference approach was proposed by Berk et al. (2013). The method yields universally valid inferences by considering all possible selection procedures that could produce the submodel in question. Let $\mathcal{A}$ denote the set of all possible candidate models. We regard the regression coefficients β_j^A relative to their respective submodels A, not relative to the full model (Berk et al. 2013). The method constructs a set of confidence intervals for all possible submodels such that

$$\mathbb{P}\left(\bigcap_{A\in\mathcal{A}}\{\beta_j^A \in \mathrm{CI}_j^A\}\right) \geq 1-\alpha$$

(Kuchibhotla et al. 2022). The resulting inferences do not depend on the selected model being correct (Tibshirani et al. 2016). Simultaneous inference is a lot more computationally expensive to perform than polyhedral inference. It is recommended by (Berk et al. 2013) to impose restrictions on the set of candidate models $\mathcal{A}$ to manage computation. However, in contrast to polyhedral inference, simultaneous inference can be used to obtained valid inference after any model selection procedure has taken place, which is a notable advantage when the model selection cannot be formally defined.

Chapter 6

Conclusions and further work

Post-selection inference is a highly relevant field in modern statistics. Techniques for handling inference after adaptive model selection are continuously being developed and researched, and existing frameworks are being improved and expanded. The polyhedral framework provides a general scheme to perform valid inference after model selection whenever the selection event can be fully characterized by a set of linear inequalities in $\mathbf{y}$, $\{\Psi\mathbf{y} \leq \mathbf{b}\}$. The polyhedral lemma states that this selection event can be rewritten as the event that the linear contrast $\boldsymbol{\eta}^T\mathbf{y}$ lies in the interval $[\mathcal{V}^+(\mathbf{z}), \mathcal{V}^-(\mathbf{z})]$. The truncation limits are functions of Ψ, $\mathbf{b}$, and the part of $\mathbf{y}$ orthogonal to the projection onto $\boldsymbol{\eta}$, $\mathbf{z}$, but they are statistically independent of $\mathbf{y}$. This key result allows the construction of a selection adjusted test statistic conditional on $\{\Psi\mathbf{y} \leq \mathbf{b}\}$, which is shown to follow a $U(0,1)$ distribution under the null hypothesis $\boldsymbol{\eta}^T\boldsymbol{\mu} = 0$. This provides us with a scheme to perform exact inferences after the model selection has taken place.

A remarkable property of the property of the polyhedral method is that it is in closed form. As it requires no sampling or estimation, it is very computationally convenient. However, through the use of the polyhedral inference framework, we do condition on both the model and signs $\hat{A} = A, \hat{\mathbf{s}} = \mathbf{s}$ and on the vector $\mathbf{z}$. As discussed in Section 5.1, the resulting confidence intervals are generally wide, and can have infinite expected length. Conditioning on less results in shorter confidence intervals, but requires more computation when we require the same level of coverage proability. We have successfully omitted the conditioning on the sign pattern of the coefficients for forward selection and the lasso, and reviewed theory and examples that show that this results in shorter confidence intervals with the same $1-\alpha$ coverage probability. An interesting question is whether it would be possible to omit the conditioning on the component $\mathbf{z}$ of $\mathbf{y}$ orthogonal to the direction $\boldsymbol{\eta}$ of interest. However, this condition is necessary for the polyhedral method to be a closed form method for post-selection inference. In other words, including the conditioning on $\mathbf{z}$ is done for computational reasons, and it is not clear how to exclude it without losing the very properties that make the polyhedral method convenient to use for inference after model selection.

Two significant assumptions we have relied on are that the number of steps

k in the forward selection procedure and the tuning parameter λ in the lasso are fixed. In practice, the number of steps for forward selection is usually determined by a stopping rule. The lasso tuning parameter λ is normally chosen by cross validation, and hence it is a random variable. An interesting extension to consider for further work is how to include a conditioning on the choice of λ, instead of assuming that it is fixed.

Another natural extension of this thesis could be to include the construction of polyhedral selection events for least angular regression. This is elaborated by Tibshirani et al. (2016), pp. 606–607, and implementations for this are included in the `selectiveInference` package. Similarly to forward selection and the lasso, the selection adjusted tests and confidence intervals are calculated in the same way.

Bibliography

Berk, Richard, Lawrence Brown, Andreas Buja, Kai Zhang and Linda Zhao (2013). 'Valid post-selection inference'. In: *The Annals of Statistics*, pp. 802–837.

Fithian, William, Dennis Sun and Jonathan Taylor (2014). 'Optimal inference after model selection'. In: *arXiv preprint arXiv:1410.2597*.

Hastie, Trevor, Robert Tibshirani, Jerome H Friedman and Jerome H Friedman (2009). *The elements of statistical learning: data mining, inference, and prediction*. Vol. 2. Springer.

Hastie, Trevor, Robert Tibshirani and Martin Wainwright (2015). *Statistical learning with sparsity: the lasso and generalizations*. CRC press.

Hyun, Sangwon, Max G'Sell and Ryan J. Tibshirani (2018). 'Exact post-selection inference for the generalized lasso path'. In: *Electronic Journal of Statistics* 12.1, pp. 1053–1097. DOI: 10.1214/17-EJS1363. URL: https://doi.org/10.1214/17-EJS1363.

James, Gareth, Daniela Witten, Trevor Hastie, Robert Tibshirani et al. (2013). *An introduction to statistical learning*. Vol. 112. Springer.

Kivaranovic, Danijel and Hannes Leeb (2021). 'On the length of post-model-selection confidence intervals conditional on polyhedral constraints'. In: *Journal of the American Statistical Association* 116.534, pp. 845–857.

Kuchibhotla, Arun K, John E Kolassa and Todd A Kuffner (2022). 'Post-selection inference'. In: *Annual Review of Statistics and Its Application* 9, pp. 505–527.

Lee, Jason D, Dennis L Sun, Yuekai Sun and Jonathan E Taylor (2016). 'Exact post-selection inference, with application to the lasso'. In:

Lockhart, Richard, Jonathan Taylor, Ryan J Tibshirani and Robert Tibshirani (2014). 'A significance test for the lasso'. In: *Annals of statistics* 42.2, p. 413.

Næss, Fanny Øverbø (2023). *Exact inference conditioned on the selection event*. Project report in TMA4500. Department of Mathematical Sciences NTNU – Norwegian University of Science and Technology.

Taylor, Jonathan and Robert J Tibshirani (2015). 'Statistical learning and selective inference'. In: *Proceedings of the National Academy of Sciences* 112.25, pp. 7629–7634.

Tibshirani, Robert (1996). 'Regression shrinkage and selection via the lasso'. In: *Journal of the Royal Statistical Society Series B: Statistical Methodology* 58.1, pp. 267–288.

Tibshirani, Robert (2016). *Invited Talk: Post-selection Inference for Forward Stepwise Regression, Lasso and other procedures*. Microsoft Research. URL: https://youtu.be/RKQJEvc02hc?si=%200dlktxA7rY-%20K05R7.

Tibshirani, Ryan J. (2013). 'The lasso problem and uniqueness'. In: *Electronic Journal of Statistics* 7.none, pp. 1456–1490. DOI: 10.1214/13-EJS815. URL: https://doi.org/10.1214/13-EJS815.

Tibshirani, Ryan J., Alessandro Rinaldo, Rob Tibshirani and Larry Wasserman (2018). 'Uniform asymptotic inference and the bootstrap after model selection'. In: *The Annals of Statistics* 46.3, pp. 1255–1287. DOI: 10.1214/17-AOS1584. URL: https://doi.org/10.1214/17-AOS1584.

Tibshirani, Ryan J, Jonathan Taylor, Richard Lockhart and Robert Tibshirani (2016). 'Exact post-selection inference for sequential regression procedures'. In: *Journal of the American Statistical Association* 111.514, pp. 600–620.

Tibshirani, Ryan, Rob Tibshirani, Jonathan Taylor, Joshua Loftus, Stephen Reid and Jelena Markovic (2022). *Package 'selectiveInference'*. R package version 1.2.5. URL: https://cran.r-project.org/web/packages/selectiveInference/selectiveInference.pdf.

United Nations (2015). *Transforming our world: The 2030 Agenda for Sustainable Development*. Accessed: 2024-05-28. URL: https://sustainabledevelopment.un.org/post2015/transformingourworld.

Appendix A

R code

A.1 Simulating p-values for forward selection

The following code snippets are recycled from previous work (Næss 2023).

```
library(selectiveInference)
set.seed(123)
n<-1000
preds <- 5
p <- apply(matrix(runif(n*preds), nrow = n), 1, min)
obs_naive <- sapply(1:n, function(i) sum(p <= p[i])) / n
pvec <- numeric(n)
X <- matrix(rnorm(n*preds), nrow = n)
for (i in 1:n) {
  y <- rnorm(n)
  fs_fit <- fs(X, y, maxsteps = 1)
  fs_inf <- fsInf(fs_fit)
  pvals <- fs_inf$pv
  pval <- min(pvals)
  pvec[i] <- pval
}
obs_selective <- sapply(1:n, function(i) sum(pvec <= pvec[i])) / n
```

The results from the code above is plotted to create Figure 4.1 as follows:

```
plot(p, obs_naive, xlim = c(0, 1), ylim = c(0, 1), ylab = "Observed",
     xlab = "Expected", main = "FS␣step␣k=1,␣p=5", col="indianred3")
points(pvec, obs_selective, xlim = c(0, 1), ylim = c(0, 1), ylab = "Observed",
       xlab = "Expected", col = "royalblue")
legend("bottomright", legend = c("Naive", "Selection␣adjusted"),
       col = c("indianred3", "royalblue3"), pch = c(1, 1))
abline(v=0.05,col="gray5", lty=2, lwd=1.5)
```

A.2 Simulating type I error for forward selection

```
library(selectiveInference)
set.seed(123)
type1forp_FS <- function(n, preds){
  p <- apply(matrix(runif(n*preds), nrow = n), 1, min)
  obs_naive <- sapply(1:n, function(i) sum(p <= p[i])) / n
  type1naive <-(sum(p <= 0.05))/n       #Type 1 error for the Naive test
  pvec <- numeric(n)
  X <- matrix(rnorm(n*preds), nrow = n) #Random design matrix
  for (i in 1:n) {
    y <- rnorm(n)                       #Random response vector
    fs_fit <- fs(X, y, maxsteps = 1)    #Perform first step of FS
    fs_inf <- fsInf(fs_fit)             #From selectiveInference library
    pvals <- fs_inf$pv                  #Selection adjusted p-value
    pval <- min(pvals)
    pvec[i] <- pval
  }
  obs_selective <- sapply(1:n, function(i) sum(pvec <= p[i])) / n
  type1tg <- (sum(pvec <= 0.05))/n       #Type 1 error for the TG test
  returnlist <- list("TG"=obs_selective, "p"=p, "naive"=obs_naive,
  "type1_TG"=type1tg, "type1_naive"=type1naive)
  return(returnlist)
}

#Number of simulations and observations
n<-1000
#Vector containing values for the number of predictors
preds <- 1:10
#Initialize result vectors
type1_TG_results <- numeric(length(preds))
type1_naive_results <- numeric(length(preds))

for (i in seq_along(preds)) {
  p_result <- type1forp_FS(n, preds[i])
  type1_TG_results[i] <- p_result$type1_TG
  type1_naive_results[i] <- p_result$type1_naive
}
```

The results from the code above is plotted to create Figure 4.2 as follows:

```
plot(preds, type1_TG_results, type = "l", col = "royalblue",
xlab = "Number␣of␣predictors", ylab = "Type␣I␣error",
main = "FS␣step␣k=1", ylim=c(0,0.5), lwd=2)
lines(preds, type1_naive_results, col = "indianred3", lwd=2)
legend("topright", legend = c("Selection␣adjusted", "Naive"),
col = c("royalblue", "indianred3"), lty = 1, lwd=c(2,2))
abline(h=0.05,col="gray5", lty=2, lwd=1.5)
```

A.3 Polyhedral inference for forward selection

This section contains the necessary functions to use polyhedral post-selection inference on models chosen by forward stepwise selection. The example described in Section 4.1.4 and shown in Figure 4.3 is also provided.

A.3.1 Implemented functions

For error checking purposes, all p-values and confidence intervals have been compared to equivalent results produced by the `selectiveInference` library. The p-values from this code are equal to the ones from the library up to the fifth decimal. The confidence intervals differ slightly from the second decimal. This is likely due to different choices of grid size in binary search algorithms. Throughout the code, the selection matrix Ψ is named A.

Auxiliary functions

```
#Libraries
library(selectiveInference)
library(glmnet)
library(ggplot2)
library(knitr)
library(MASS)
library(plotrix)


ej <- function(j, n){
  ej <- rep(0, n)
  ej[j] <- 1
  return(ej)
}

euclidean_norm <- function(v) {
  return(sqrt(sum(v^2)))
}

#Simple binary search algorithm
interval_halving_search <- function(f, target, lower, upper, tolerance = 1e-6,
max_iterations = 1000) {
  iteration <- 0
  while (upper - lower > tolerance && iteration < max_iterations) {
    middle <- (lower + upper) / 2
    if (f(middle) < target) {
      upper <- middle
    } else {
      lower <- middle
    }
    iteration <- iteration + 1
  }
  return((lower + upper) / 2)
}

#Generates all possible sign patterns of n active coefficients
sign_patterns <- function(n) {
```

```
  #Only one active variable
  if (n == 1) {
    return(list(c(-1), c(1)))
  } else {
    #Use recursion to generate sign patterns for n-1 coefficients
    subpatterns <- sign_patterns(n-1)
    signpatterns <- list()
    #Make new sign patterns for every subpattern by adding -1 and 1
    for (signpattern in subpatterns) {
      signpatterns <- c(signpatterns, list(c(-1, signpattern)))
      signpatterns <- c(signpatterns, list(c(1, signpattern)))
    }
    return(signpatterns)
  }
}
```

Truncated Gaussian test for models chosen by forward selection

```
#Performs a FS adjusted test from the polyhedral lemma
myselinf_FS <- function(X,y,A,b,actives,j,s){
  #X: design matrix
  #y: response vector
  #A: current FS selection matrix
  #b: vector of zeros
  #actives: vector of indices of active variables
  #j: the variable index to be tested
  #s: current sign pattern vector
  X_A <- X[, actives]
  XTX <- t(X_A) %*% X_A
  XTXinv <- solve(XTX)
  pinv <- (XTXinv) %*% t(X_A)  #Pseudoinverse of X_A
  pinvt <- t(pinv)
  e <- ej(j,length(actives))
  e <- matrix(e, ncol=1)
  eta <- pinvt %*% e         #eta
  etat <- t(eta)
  etateta <- etat %*% eta
  etatetainv <- solve(etateta)
  c <- eta %*% etatetainv
  z<- y-(c%*%etat)%*%y
  Ac <- A %*% c
  Az <- A %*% z
  #Extract indices for which Ac < 0 and Ac > 0
  minjs <- which(Ac < 0)
  maxjs <- which(Ac > 0)
  zerojs <- which(Ac ==0)
  #Calculate the truncation limits
  vmin<- max((b[minjs]-Az[minjs])/Ac[minjs])
  vmax<- min((b[maxjs]-Az[maxjs])/Ac[maxjs])
  etaty <- etat %*% y
  etanorm <- euclidean_norm(eta)
  #Calculate selection adjusted p-value and F
  #Truncated gaussian CDF
  if(etaty >= 0){
    F_selectionadjusted <-(pnorm((etaty)/etanorm)-pnorm(vmin/etanorm))/
      (pnorm(vmax/etanorm)-pnorm(vmin/etanorm))
    p_selectionadjusted <- 1-F_selectionadjusted
  }else{
```

```
    p_selectionadjusted <-(pnorm((etaty)/etanorm)-pnorm(vmin/etanorm))/
      (pnorm(vmax/etanorm)-pnorm(vmin/etanorm))
    F_selectionadjusted <- 1-p_selectionadjusted
  }
  #Handling values too close to 0 in denominator
  if (is.nan(p_selectionadjusted)) {
    p_selectionadjusted <- 0
  }

  if (is.nan(F_selectionadjusted)) {
    F_selectionadjusted <- 1
  }
  returnlist <- list("pv"=p_selectionadjusted, "F"=F_selectionadjusted,
  "vmin"=vmin,"vmax"=vmax, "etaty"=etaty, "etanorm"=etanorm)
  return(returnlist)
}
```

Selection adjusted test statistic for calculation of confidence intervals

```
F_selectionadjusted_CIsearch <- function(L, selinf.results){
  etaty <- selinf.results$etaty
  etanorm <- selinf.results$etanorm
  vmin <- selinf.results$vmin
  vmax <- selinf.results$vmax
  F_val <- (pnorm((etaty-L)/etanorm)-pnorm((vmin-L)/etanorm))/
    (pnorm((vmax-L)/etanorm)-pnorm((vmin-L)/etanorm))
  return(F_val)
}
```

Selection adjusted test statistic conditional on only the model

```
F_union <- function(L, etatys, etanorm, vminvals, vmaxvals){
  n_signpatterns <- length(vminvals)
  numvec <- c(rep(0,n_signpatterns))
  demvec <- c(rep(0,n_signpatterns))
  for(i in 1:n_signpatterns){
    if(vminvals[i] < vmaxvals[i]){
      if(etatys[i] < vminvals[i]){
        numvec[i] <- 0
      }else{
        if(etatys[i] > vmaxvals[i]){
          numvec[i] <- demvec[i] <- pnorm((vmaxvals[i]-L)/etanorm[i])
          -pnorm((vminvals[i]-L)/etanorm[i])
        }else{
          numvec[i] <- pnorm((etatys[i]-L)/etanorm[i])-
            pnorm((vminvals[i]-L)/etanorm[i])
        }
      }
      demvec[i] <- pnorm((vmaxvals[i]-L)/etanorm[i])-
        pnorm((vminvals[i]-L)/etanorm[i])
    }
    else{
      numvec[i] <- 0
      demvec[i] <- 0
    }
```

```
  }
  numerator <- sum(numvec)
  denominator <- sum(demvec)
  F_val_union <- numerator/denominator
  return(F_val_union)
}
```

Test inversion

```
union_CI_FS <- function(X, y, A_list, b, signpatterns,j, buffer, alpha, result.orig,
                        actives){
  #A_list: List of selection matrices A_s for the different sign patterns
  #Signpatterns: List of all possible sign patterns for the active variables
  results <- list()
  for (i in seq_along(signpatterns)) {
    s <- signpatterns[i]
    result <- myselinf_FS(X, y, A_list[i], b, actives,j, s)
    results <- c(results, list(result))
  }
  vmins <- c()
  vmaxs <- c()
  etatys <- c()
  etanorm <- c()
  for (result in results) {
    vmins <- c(vmins, result$vmin)
    vmaxs <- c(vmaxs, result$vmax)
    etatys <- c(etatys, result$etaty)
    etanorm <- c(etanorm, result$etanorm)
  }
  lb <- result.orig$etaty - buffer
  ub <- result.orig$etaty + buffer
  lower <- interval_halving_search(function(L)
  F_union(L, etatys, etanorm , vmins, vmaxs),
                        1-alpha/2,
                        lb,
                        ub)
  upper <- interval_halving_search(function(L)
  F_union(L, etatys, etanorm , vmins, vmaxs),
                        alpha/2,
                        lb,
                        ub)
  CI_union <- c(lower, upper)
  return(list("CI_union"=CI_union, "vmins"=vmins, "vmaxs"=vmaxs, "etatys"=etatys,
              "etanorm"=etanorm))

}
```

A.3.2 Forward selection example on simulated data

We note that when implementing selection events manually for small p, it is useful to check that $\Psi \mathbf{y} \leq \mathbf{0}$ holds.

```
set.seed(1)
n=80
p=3
sigma=1
X <- matrix(rnorm(n*p),n,p)
X <-scale(X, center = TRUE, scale = TRUE)
beta <- c(0.6,0.3,0)
y <- X%*%beta + sigma*rnorm(n)

x1 <- matrix(X[, 1], ncol = 1)
x2 <- matrix(X[, 2], ncol = 1)
x3 <- matrix(X[, 3], ncol = 1)

#Creating the rows of A
#A is 2(p-k) x n
#Rows of selection matrix, k=1, x1 chosen
A1 <- -(t(x1) / euclidean_norm(x1)) + t(x2) /euclidean_norm(x2)
A2 <- -(t(x1) / euclidean_norm(x1)) - t(x2) /euclidean_norm(x2)
A3 <- -(t(x1) / euclidean_norm(x1)) + t(x3) /euclidean_norm(x3)
A4 <- -(t(x1) / euclidean_norm(x1)) - t(x3) /euclidean_norm(x3)

#Collect the rows to form the selection matrix
A_1 <- rbind(A1, A2, A3, A4)
b_1 <- c(rep(0,4))
s1 <- sign(t(x1) %*% y)
actives <- c(1)
#Selection adjusted test cond. on model and signs
selinf.results <-myselinf_FS(X,y,A_1,b_1,actives,1,s1)

#Calculating confidence intervals
alpha <- 0.05
lb <- selinf.results$etaty -1.5
ub <- selinf.results$etaty +1.5
lower_1 <- interval_halving_search(function(L)
F_selectionadjusted_CIsearch(L, selinf.results),
                          1-alpha/2,
                          lb,
                          ub)
upper_1 <- interval_halving_search(function(L)
F_selectionadjusted_CIsearch(L, selinf.results),
                          alpha/2,
                          lb,
                          ub)

myCI1 <- c(lower_1, upper_1)
s <- sign(t(x1) %*% y)
buffer <- 1
#STEP 2: x2 chosen
s2 <- sign(t(x2) %*% y) #sign
X_A <- X[,1]
XTX <- t(X_A) %*% X_A
XTXinv <- solve(XTX)
#Projection onto the column space of X_A1
P <- X_A %*% (XTXinv) %*% t(X_A)
X_A2 <- X[, c(1,2)]
```

```
#Regress x2 onto x1
lm_2 <- lm(X_A2[,2] ~ X_A2[,1], data=as.data.frame(X_A2))
res_2 <- residuals(lm_2)
res_2<- matrix(res_2)
#The submatrix if 3 had been chosen in step 2
X_A23 <-X[, c(1,3)]
#Regress x3 onto x1
lm_23 <- lm(X_A23[,2] ~ X_A2[,1], data=as.data.frame(X_A23))
res_23 <- residuals(lm_23)
res_23<- matrix(res_23)
#Orthogonal to P
P_ortho <- diag(n)-P
#Rows for the selection matrix at k=2, x2 chosen
A5 <- -s2%*%(t(res_2)%*%P_ortho / euclidean_norm(res_2)) +
t(res_23)%*%P_ortho /euclidean_norm(res_23)
A6 <- -s2%*%(t(res_2)%*%P_ortho / euclidean_norm(res_2)) -
t(res_23)%*%P_ortho /euclidean_norm(res_23)
#Append the new rows to A
A_2 <- rbind(A_1, A5, A6)
A_2_11 <- A_2

actives_2 <- c(1,2)
s2<- c(s,s2)
b_2<- c(rep(0,6))
selinf.results_2 <-myselinf_FS(X,y,A_2,b_2,actives_2,2,s2)

lb_2 <- selinf.results_2$etaty -1.2 #adjusted manually for precision
ub_2 <- selinf.results_2$etaty +1.2
lower_2 <- interval_halving_search(function(L)
F_selectionadjusted_CIsearch(L, selinf.results_2),
                          1-alpha/2,
                          lb_2,
                          ub_2)
upper_2 <- interval_halving_search(function(L)
F_selectionadjusted_CIsearch(L, selinf.results_2),
                          alpha/2,
                          lb_2,
                          ub_2)

myCI2 <- c(lower_2, upper_2)

#THIRD STEP: x3 chosen
s3 <- sign(t(x3) %*% y)
XTX_2 <- t(X_A2) %*% X_A2
XTXinv_2 <- solve(XTX_2)
#Projection onto the column space of X_A2
P_2 <- X_A2 %*% (XTXinv_2) %*% t(X_A2)
P_ortho2 <- diag(n)-P_2
X_A3 <- X[, c(1,2,3)]
#Regress X[,3] onto the other columns
lm_3 <- lm(X_A3[,3] ~ X_A3[,1] +X_A3[,2]  , data=as.data.frame(X_A3))
res_3 <- residuals(lm_3)
res_3<- matrix(res_3)

#Adding row for p=k=3
A7 <- -s3%*%(t(res_3)%*%P_ortho2 / euclidean_norm(res_3))
#Bind to the final selection matrix
A_3 <- rbind(A_2,A7)

b_3 <- c(rep(0,7))
```

```
actives_3 <- c(1,2,3)
s_3 <- c(s2,s3)
selinf.results_3 <-myselinf_FS(X,y,A_3,b_3,actives_3,3,s_3)
lb_3 <- selinf.results_3$etaty -1
ub_3 <- selinf.results_3$etaty +1
lower_3 <- interval_halving_search(function(L)
F_selectionadjusted_CIsearch(L, selinf.results_3),
                         1-alpha/2,
                         lb_3,
                         ub_3)
upper_3 <- interval_halving_search(function(L)
F_selectionadjusted_CIsearch(L, selinf.results_3),
                         alpha/2,
                         lb_3,
                         ub_3)
myCI3 <- c(lower_3, upper_3)

#Rows of selection matrix, k=1, x1 chosen, signs reversed
A1r <- (t(x1) / euclidean_norm(x1)) + t(x2) /euclidean_norm(x2)
A2r <- (t(x1) / euclidean_norm(x1)) - t(x2) /euclidean_norm(x2)
A3r <- (t(x1) / euclidean_norm(x1)) + t(x3) /euclidean_norm(x3)
A4r <- (t(x1) / euclidean_norm(x1)) - t(x3) /euclidean_norm(x3)
A_1_rev <- rbind(A1r, A2r, A3r, A4r)

selinf.results_rev <-myselinf_FS(X,y,A_1_rev,b_1,1,1,s=c(-1))
etatys <- c(selinf.results$etaty, selinf.results_rev$etaty)
etanorm <- c(selinf.results$etanorm, selinf.results_rev$etanorm)
vmins <- c(selinf.results$vmin, selinf.results_rev$vmin)
vmaxs <- c(selinf.results$vmax, selinf.results_rev$vmax)

selinf.results_rev <-myselinf_FS(X,y,A_1_rev,b_1,c(1,2),1,s=c(-1))
etatys <- c(selinf.results$etaty, selinf.results_rev$etaty)
etanorm <- c(selinf.results$etanorm, selinf.results_rev$etanorm)
vmins <- c(selinf.results$vmin, selinf.results_rev$vmin)
vmaxs <- c(selinf.results$vmax, selinf.results_rev$vmax)

lb <- selinf.results$etaty - 1.1
ub <- selinf.results$etaty + 1.1
lower <- interval_halving_search(function(L)
F_union(L, etatys, etanorm , vmins, vmaxs),
                       1-alpha/2,
                       lb,
                       ub)
upper <- interval_halving_search(function(L)
F_union(L, etatys, etanorm , vmins, vmaxs),
                       alpha/2,
                       lb,
                       ub)
CI_union <- c(lower, upper)

s2 <-1
#Rows for the selection matrix at k=2, x2 chosen, signs reversed
A5r <- s2%*%(t(res_2)%*%P_ortho / euclidean_norm(res_2)) +
t(res_23)%*%P_ortho /euclidean_norm(res_23)
A6r <- s2%*%(t(res_2)%*%P_ortho / euclidean_norm(res_2)) -
t(res_23)%*%P_ortho /euclidean_norm(res_23)

#sign pattern x1 pos x2 neg
A_2_10 <- rbind(A_1, A5r, A6r)
#sign pattern x1 neg x2 neg
```

```
A_2_00 <- rbind(A_1_rev, A5r, A6r)
#sign pattern x1 neg x2 pos
A_2_01 <- rbind(A_1_rev, A5, A6)

#Store results for each
selinf.results_2_10 <-myselinf_FS(X,y,A_2_10 ,b_2,actives=c(1,2),2,s=c(1,-1))
selinf.results_2_00 <-myselinf_FS(X,y,A_2_00,b_2,actives=c(1,2),2,s=c(-1,-1))
selinf.results_2_01 <-myselinf_FS(X,y,A_2_01,b_2,actives=c(1,2),2,s=c(-1,1))

etatys2 <- c(selinf.results_2_00$etaty,
selinf.results_2_10$etaty,selinf.results_2_01$etaty,selinf.results_2$etaty)
etanorm2 <- c(selinf.results_2_00$etanorm,
selinf.results_2_10$etanorm,selinf.results_2_01$etanorm,selinf.results_2$etanorm)
vmins2 <- c(selinf.results_2_00$vmin,
selinf.results_2_10$vmin,selinf.results_2_01$vmin,selinf.results_2$vmin)
vmaxs2 <- c(selinf.results_2_00$vmax,
selinf.results_2_10$vmax,selinf.results_2_01$vmax,selinf.results_2$vmax)

lb2 <- selinf.results_2$etaty - buffer
ub2 <- selinf.results_2$etaty + buffer
lower2 <- interval_halving_search(function(L)
F_union(L, etatys2, etanorm2 , vmins2, vmaxs2),
                        1-alpha/2,
                        lb2,
                        ub2)
upper2 <- interval_halving_search(function(L)
F_union(L, etatys2, etanorm2 , vmins2, vmaxs2),
                        alpha/2,
                        lb2,
                        ub2)
CI_union2 <- c(lower2, upper2)

#step 3 signs reversed
A7r <- s3%*%(t(res_3)%*%P_ortho2 / euclidean_norm(res_3))

#All possible selection matrices
A_3_000 <- rbind(A_2_00, A7)
A_3_100 <- rbind(A_2_10, A7)
A_3_101 <- rbind(A_2_10, A7r)
A_3_110 <- rbind(A_2_11, A7)
A_3_001 <- rbind(A_2_00, A7r)
A_3_010 <- rbind(A_2_01, A7r)
A_3_011 <- rbind(A_2_10, A7)
A_3_111 <- rbind(A_2_11, A7r)
selinf.results_3_000 <-myselinf_FS(X,y,A_3_000
,b_3,actives=c(1,2,3),3,s=c(-1,-1,-1))
selinf.results_3_100 <-myselinf_FS(X,y,A_3_100
,b_3,actives=c(1,2,3),3,s=c(1,-1,-1))
selinf.results_3_101 <-myselinf_FS(X,y,A_3_101
,b_3,actives=c(1,2,3),3,s=c(1,-1,1))
selinf.results_3_110 <-myselinf_FS(X,y,A_3_110
,b_3,actives=c(1,2,3),3,s=c(1,1,-1))
selinf.results_3_001 <-myselinf_FS(X,y,A_3_001
,b_3,actives=c(1,2,3),3,s=c(-1,-1,1))
selinf.results_3_010 <-myselinf_FS(X,y,A_3_010
,b_3,actives=c(1,2,3),3,s=c(-1,1,-1))
selinf.results_3_011 <-myselinf_FS(X,y,A_3_011
,b_3,actives=c(1,2,3),3,s=c(-1,1,1))
selinf.results_3_111 <-myselinf_FS(X,y,A_3_111
,b_3,actives=c(1,2,3),3,s=c(1,1,1))
```

```
etatys3 <- c(selinf.results_3_000$etaty, selinf.results_3_100$etaty,
selinf.results_3_101$etaty,
selinf.results_3_111$etaty,
selinf.results_3_001$etaty,
selinf.results_3_010$etaty,
selinf.results_3_011$etaty,
selinf.results_3_111$etaty)
etanorm3 <- c(selinf.results_3_000$etanorm,
selinf.results_3_100$etanorm,
selinf.results_3_101$etanorm,
selinf.results_3_111$etanorm,
              selinf.results_3_001$etanorm, selinf.results_3_010$etanorm,
              selinf.results_3_011$etanorm,
              selinf.results_3_111$etanorm)
vmins3 <- c(selinf.results_3_000$vmin, selinf.results_3_100$vmin,
selinf.results_3_101$vmin,
selinf.results_3_111$vmin,
            selinf.results_3_001$vmin, selinf.results_3_010$vmin,
            selinf.results_3_011$vmin,
            selinf.results_3_111$vmin)
vmaxs3 <- c(selinf.results_3_000$vmax, selinf.results_3_100$vmax,
selinf.results_3_101$vmax,
selinf.results_3_111$vmax,
            selinf.results_3_001$vmax, selinf.results_3_010$vmax,
            selinf.results_3_011$vmax,
            selinf.results_3_111$vmax)
etatys3 <- c(selinf.results_3_000$etaty, selinf.results_3_100$etaty,
selinf.results_3_101$etaty,
selinf.results_3_110$etaty,
             selinf.results_3_001$etaty, selinf.results_3_010$etaty,
             selinf.results_3_011$etaty,
             selinf.results_3_111$etaty)
etanorm3 <- c(selinf.results_3_000$etanorm, selinf.results_3_100$etanorm,
selinf.results_3_101$etanorm,
selinf.results_3_110$etanorm,
              selinf.results_3_001$etanorm, selinf.results_3_010$etanorm,
              selinf.results_3_011$etanorm,
              selinf.results_3_111$etanorm)
vmins3 <- c(selinf.results_3_000$vmin, selinf.results_3_100$vmin,
selinf.results_3_101$vmin,
selinf.results_3_110$vmin,
            selinf.results_3_001$vmin, selinf.results_3_010$vmin,
            selinf.results_3_011$vmin,
            selinf.results_3_111$vmin)
vmaxs3 <- c(selinf.results_3_000$vmax, selinf.results_3_100$vmax,
selinf.results_3_101$vmax,
selinf.results_3_110$vmax,
            selinf.results_3_001$vmax, selinf.results_3_010$vmax,
            selinf.results_3_011$vmax,
            selinf.results_3_111$vmax)

lb3 <- selinf.results_3$etaty-0.9
ub3 <- selinf.results_3$etaty +0.9
lower3 <- interval_halving_search(function(L)
F_union(L, etatys3, etanorm3 , vmins3, vmaxs3),
                        1-alpha/2,
                        lb3,
                        ub3)
upper3 <- interval_halving_search(function(L)
```

```
F_union(L, etatys3, etanorm3 , vmins3, vmaxs3),
                        alpha/2,
                        lb3,
                        ub3)
CI_union3 <- c(lower3, upper3)

#Unadjusted least squares confidence intervals
model_x1 <- lm(y ~ x1-1)
naiveCI1<- confint(model_x1, level = 0.95)
x2 <- matrix(X[, 2], ncol = 1)
X12 <- cbind(x1, x2)
model_x1_x2 <- lm(y ~ X12-1)
naiveCI2<-confint(model_x1_x2)[2,]
X123 <- cbind(x1, x2, x3)
model_x1_x2_x3 <- lm(y ~ X123-1)
naiveCI3<- confint(model_x1_x2_x3)[3,]

modelonly <- data.frame(index = c(1, 2, 3),
                        lower_ci = c(CI_union[1], CI_union2[1], CI_union3[1]),
                        upper_ci = c(CI_union[2], CI_union2[2], CI_union3[2]))
naives <- data.frame(index = c(1, 2, 3),
                     lower_ci = c(naiveCI1[1], naiveCI2[1], naiveCI3[1]),
                     upper_ci = c(naiveCI1[2], naiveCI2[2], naiveCI3[2]))
modelandsigns <- data.frame(index = c(1, 2, 3),
                            lower_ci = c(myCI1[1], myCI2[1], myCI3[1]),
                            upper_ci = c(myCI1[2], myCI2[2], myCI3[2]))
combined_data <- rbind(modelandsigns, modelonly, naives)
combined_data$method <-
rep(c("Model␣and␣signs", "Model␣only","Unadjusted"), each = 3)

# Plot with ggplot2
ggplot(combined_data, aes(x = index, ymin = lower_ci, ymax = upper_ci,
fill = method)) +
  geom_rect(aes(xmin = index - 0.2, xmax = index - 0.05),
  data = subset(combined_data, method == "Model␣and␣signs"), color = "gray100") +
  geom_rect(aes(xmin = index - 0.05, xmax = index + 0.1),
  data = subset(combined_data, method == "Model␣only"), color = "gray100") +
  geom_rect(aes(xmin = index + 0.1, xmax = index + 0.25),
  data = subset(combined_data, method == "Unadjusted"), color = "gray100") +
  labs(x = "Variable␣indices", y = "Coefficient␣values", fill = "␣") +
  scale_x_continuous(breaks = c(1, 2, 3)) +
  scale_fill_manual(values = c("cornflowerblue", "brown", "gray40")) +
  theme_minimal() +
  theme(legend.position = "top", text = element_text(size=14))+
  geom_segment(aes(x = 0.7, xend = 1.5, y = 0.6, yend = 0.6),
  linetype = "dashed", color = "midnightblue", size=0.35) +
  geom_segment(aes(x = 1.5, xend = 2.5, y = 0.3, yend = 0.3),
  linetype = "dashed", color = "midnightblue", size=0.35) +
  geom_segment(aes(x = 2.5, xend = 3.5, y = 0, yend = 0),
  linetype = "dashed", color = "midnightblue", size=0.35)
```

A.4 Polyhedral inference for the lasso

We use the same auxiliary functions for the basis vector, euclidean norm and binary search as in the previous section.

A.4.1 Implemented functions

Lasso selection event and truncated Gaussian test

```
myselinf_lasso <- function(X,y,lambda,actives,j,s){
  #Split the design matrix into active and inactive columns
  X_A <- X[, actives]
  X_I <- X[, -actives]
  #Define the projection onto the column span of X_A
  XTX <- t(X_A) %*% X_A
  XTXinv <- solve(XTX)
  P <- X_A %*% (XTXinv) %*% t(X_A)
  #Inactive constraints A_0
  I <- diag(nrow(X))
  elem <- t(X_I) %*% (I-P)
  A_0 <- 1/lambda * rbind(elem, -elem)
  pseudoinv <- ginv(t(X_A))
  pseudoinvs <- pseudoinv %*% s
  #Inactive constraints b_0
  ones <- matrix(1, nrow = dim(t(X_I)%*%pseudoinvs)[1],
  ncol = dim(t(X_I)%*%pseudoinvs)[2])
  b_0 <- matrix(c(ones - t(X_I)%*%pseudoinvs,
  ones + t(X_I)%*%pseudoinvs), nrow=2*dim(ones)[1], ncol=dim(ones)[2] )
  b_0 <- rbind(ones - t(X_I)%*%pseudoinvs, ones + t(X_I)%*%pseudoinvs)
  #Sign vector
  s_vec <- c(s)
  mat <- diag(s_vec)
  #Active constraints
  A_1 <- -mat %*% XTXinv %*% t(X_A)
  b_1 <- -lambda * mat %*% XTXinv %*% s_vec
  #Collecting the active and inactive constraints
  A <- rbind(A_0, A_1) #Lasso selection matrix
  b <- rbind(b_0, b_1) #Lasso selection vector

  #Using the polyhedral lemma to derive a selection adjusted test
  #Note that the test is the same as for FS
  XTX <- t(X_A) %*% X_A
  XTXinv <- solve(XTX)
  pinv <- (XTXinv) %*% t(X_A)
  pinvt <- t(pinv)
  e <- ej(j,length(actives))
  e <- matrix(e, ncol=1)
  eta <- pinvt %*% e
  etat <- t(eta)
  etateta <- etat %*% eta
  etatetainv <- solve(etateta)
  c <- eta %*% etatetainv
  z <- (diag(1, nrow=nrow(X), ncol=nrow(X))-c%*%etat)%*%y
  Ac <- A %*% c
  Az <- A %*% z
  minjs <- which(Ac < 0)
  maxjs <- which(Ac > 0)
```

```
  zerojs <- which(Ac ==0)
  vmin<- max((b[minjs]-Az[minjs])/Ac[minjs])
  vmax<- min((b[maxjs]-Az[maxjs])/Ac[maxjs])
  etaty <- etat %*% y
  etanorm <- euclidean_norm(eta)
  if(etaty >= 0){
    F_selectionadjusted <-(pnorm((etaty)/etanorm)-pnorm(vmin/etanorm))/
      (pnorm(vmax/etanorm)-pnorm(vmin/etanorm))
    p_selectionadjusted <- 1-F_selectionadjusted
  }else{
    p_selectionadjusted <-(pnorm((etaty)/etanorm)-pnorm(vmin/etanorm))/
      (pnorm(vmax/etanorm)-pnorm(vmin/etanorm))
    F_selectionadjusted <- 1-p_selectionadjusted
  }
  #Handling values too close to 0 in denominator
  if (is.nan(p_selectionadjusted)) {
    p_selectionadjusted <- 0
  }
  if (is.nan(F_selectionadjusted)) {
    F_selectionadjusted <- 1
  }
  returnlist <- list("pv"=p_selectionadjusted, "F"=F_selectionadjusted,
  "vmin"=vmin, "vmax"=vmax, "etaty"=etaty, "etanorm"=etanorm)
  return(returnlist)
}
```

We note that the selection adjusted test is derived in the same way for forward selection and the lasso. The function above includes the construction of the lasso selection event.

```
CIfunc_tent <- function(result, buffer, alpha){
  lb <- result$etaty -buffer
  ub <- result$etaty +buffer
  lower <- interval_halving_search(function(L)
  F_selectionadjusted_CIsearch(L, result),
                         1-alpha/2,
                         lb,
                         ub)
  upper <- interval_halving_search(function(L)
  F_selectionadjusted_CIsearch(L, result),
                         alpha/2,
                         lb,
                         ub)
  CI <- c(lower, upper)
  return(CI)
}
CIfunc_custom <- function(result, lb, ub, alpha){
  lower <- interval_halving_search(function(L)
  F_selectionadjusted_CIsearch(L, result),
                         1-alpha/2,
                         lb,
                         ub)
  upper <- interval_halving_search(function(L)
  F_selectionadjusted_CIsearch(L, result),
                         alpha/2,
                         lb,
                         ub)
  CI <- c(lower, upper)
  return(CI)
```

```
}
```

Selection adjusted test statistic generalized to all sign patterns

```
F_union <- function(L, etatys, etanorm, vminvals, vmaxvals){
  n_signpatterns <- length(vminvals)
  numvec <- c(rep(0,n_signpatterns))
  demvec <- c(rep(0,n_signpatterns))
  for(i in 1:n_signpatterns){
    if(vminvals[i] < vmaxvals[i]){

      if(etatys[i] < vminvals[i]){
        numvec[i] <- 0
      }else{
        if(etatys[i] > vmaxvals[i]){
          numvec[i] <- demvec[i] <- pnorm((vmaxvals[i]-L)/etanorm[i])
          -pnorm((vminvals[i]-L)/etanorm[i])
        }else{
          numvec[i] <- pnorm((etatys[i]-L)/etanorm[i])-
          pnorm((vminvals[i]-L)/etanorm[i])
        }
      }
      demvec[i] <- pnorm((vmaxvals[i]-L)/etanorm[i])-
      pnorm((vminvals[i]-L)/etanorm[i])
    }
    else{
      numvec[i] <- 0
      demvec[i] <- 0
    }
  }
  numerator <- sum(numvec)
  denominator <- sum(demvec)
  F_val_union <- numerator/denominator
  return(F_val_union)
}
```

```
union_CI <- function(X, y, lambda, signpatterns,j, buffer, alpha, result.orig,
    actives){
  results <- list()
  for (s in signpatterns) {
    result <- myselinf_lasso(X, y, lambda, actives,j, s)
    results <- c(results, list(result))
  }
  vmins <- c()
  vmaxs <- c()
  etatys <- c()
  etanorm <- c()
  for (result in results) {
    vmins <- c(vmins, result$vmin)
    vmaxs <- c(vmaxs, result$vmax)
    etatys <- c(etatys, result$etaty)
    etanorm <- c(etanorm, result$etanorm)
  }

  lb <- result.orig$etaty - buffer
  ub <- result.orig$etaty + buffer
  lower <- interval_halving_search(function(L)
```

```
  F_union(L, etatys, etanorm , vmins, vmaxs),
                        1-alpha/2,
                        lb,
                        ub)
  upper <- interval_halving_search(function(L)
  F_union(L, etatys, etanorm , vmins, vmaxs),
                        alpha/2,
                        lb,
                        ub)
  CI_union <- c(lower, upper)
  return(list("CI_union"=CI_union, "vmins"=vmins, "vmaxs"=vmaxs, "etatys"=etatys,
  "etanorm"=etanorm))
}
```

```
#Generates all possible sign patterns of n active coefficients
sign_patterns <- function(n) {
  #Only one active variable
  if (n == 1) {
    return(list(c(-1), c(1)))
  } else {
    #Use recursion to generate sign patterns for n-1 coefficients
    subpatterns <- sign_patterns(n-1)
    signpatterns <- list()
    #Make new sign patterns for every subpattern by adding -1 and 1
    for (signpattern in subpatterns) {
      signpatterns <- c(signpatterns, list(c(-1, signpattern)))
      signpatterns <- c(signpatterns, list(c(1, signpattern)))
    }
    return(signpatterns)
  }
}
```

A.4.2 Lasso examples on simulated data

More observations than predictors

```
#Generate simulated data
set.seed(1)
n=100
beta <- c(2,2,0.5,0.5,0,0,0,0,0,0,0,0,0,0,0)
p=length(beta)
sigma=1
X <- matrix(rnorm(n*p),n,p)
X <-scale(X, center = TRUE, scale = TRUE)
y <- X%*%beta + sigma*rnorm(n)

#Fit the lasso with glmnet
lassofit <- glmnet(X, y, standardize = FALSE)
#Set a fixed value of lambda
lambda <-  12
#Extract coefficients for specified lambda
betas <- coef(lassofit, s = lambda/n, exact=TRUE, x=X, y=y)[-1]
#Indices of the active variables
actives <- which(betas != 0)
s <- c(1,1,1,1,1,-1,1)
#Polyhedral inference for lasso
results <- lapply(1:7, function(i) {
```

```
  myselinf_lasso(X, y, lambda, actives, i, s)
})
result.x1 <- results[[1]]
result.x2 <- results[[2]]
result.x3 <- results[[3]]
result.x4 <- results[[4]]
result.x5 <- results[[5]]
result.x6 <- results[[6]]
result.x7 <- results[[7]]

#Confidence intervals conditioned on model and signs
x1_ci_ms <-CIfunc_tent(result.x1, 1, 0.05)
x2_ci_ms <-CIfunc_tent(result.x2, 1, 0.05)
x3_ci_ms <-CIfunc_tent(result.x3, 1, 0.05)
x4_ci_ms <-CIfunc_tent(result.x4, 1, 0.05)
x5_ci_ms <-CIfunc_custom(result.x5, -0.73,1, 0.05)
x6_ci_ms <-CIfunc_tent(result.x6, 2, 0.05)
x7_ci_ms <-CIfunc_custom(result.x7, -0.75,0.6, 0.05)
#Adjusting only for plot. Cannot be computed
x7_ci_ms[1]<- -1.75
#Conditioned on model only
signpatterns <- sign_patterns(length(actives))
x1_ci_m <- union_CI(X, y, lambda, signpatterns =
signpatterns, j=1, buffer=1, alpha=0.05, result.orig = result.x1, actives=actives)
x2_ci_m <- union_CI(X, y, lambda, signpatterns =
signpatterns, j=2, buffer=1, alpha=0.05, result.orig = result.x2, actives=actives)
x3_ci_m <- union_CI(X, y, lambda, signpatterns =
signpatterns, j=3, buffer=1, alpha=0.05, result.orig = result.x3, actives=actives)
x4_ci_m <- union_CI(X, y, lambda, signpatterns =
signpatterns, j=4, buffer=1, alpha=0.05, result.orig = result.x4, actives=actives)
x5_ci_m <- union_CI(X, y, lambda, signpatterns =
signpatterns, j=5, buffer=1, alpha=0.05, result.orig = result.x5, actives=actives)
x6_ci_m <- union_CI(X, y, lambda, signpatterns =
signpatterns, j=6, buffer=1, alpha=0.05, result.orig = result.x6, actives=actives)
x7_ci_m <- union_CI(X, y, lambda, signpatterns =
signpatterns, j=7, buffer=1, alpha=0.05, result.orig = result.x7, actives=actives)
x1_ci_m<-x1_ci_m$CI_union
x2_ci_m<-x2_ci_m$CI_union
x3_ci_m<-x3_ci_m$CI_union
x4_ci_m<-x4_ci_m$CI_union
x5_ci_m<-x5_ci_m$CI_union
x6_ci_m<-x6_ci_m$CI_union
x7_ci_m<-x7_ci_m$CI_union

#Naive selection: The lasso has chosen the model
X_actives <- X[, actives]
linear_model <- lm(y ~ X_actives-1)
naives <- confint(linear_model)

modelonly <- data.frame(index = c(1, 2, 3,4,5,6,7),
                        lower_ci = c(x1_ci_m[1], x2_ci_m[1],
                        x3_ci_m[1],x4_ci_m[1], x5_ci_m[1],
                        x6_ci_m[1],x7_ci_m[1]),
                        upper_ci = c(x1_ci_m[2], x2_ci_m[2],
                        x3_ci_m[2], x4_ci_m[2], x5_ci_m[2],
                        x6_ci_m[2],x7_ci_m[2]))

modelandsigns <- data.frame(index = c(1, 2, 3,4,5,6,7),
                            lower_ci = c(x1_ci_ms[1],
                            x2_ci_ms[1], x3_ci_ms[1],
```

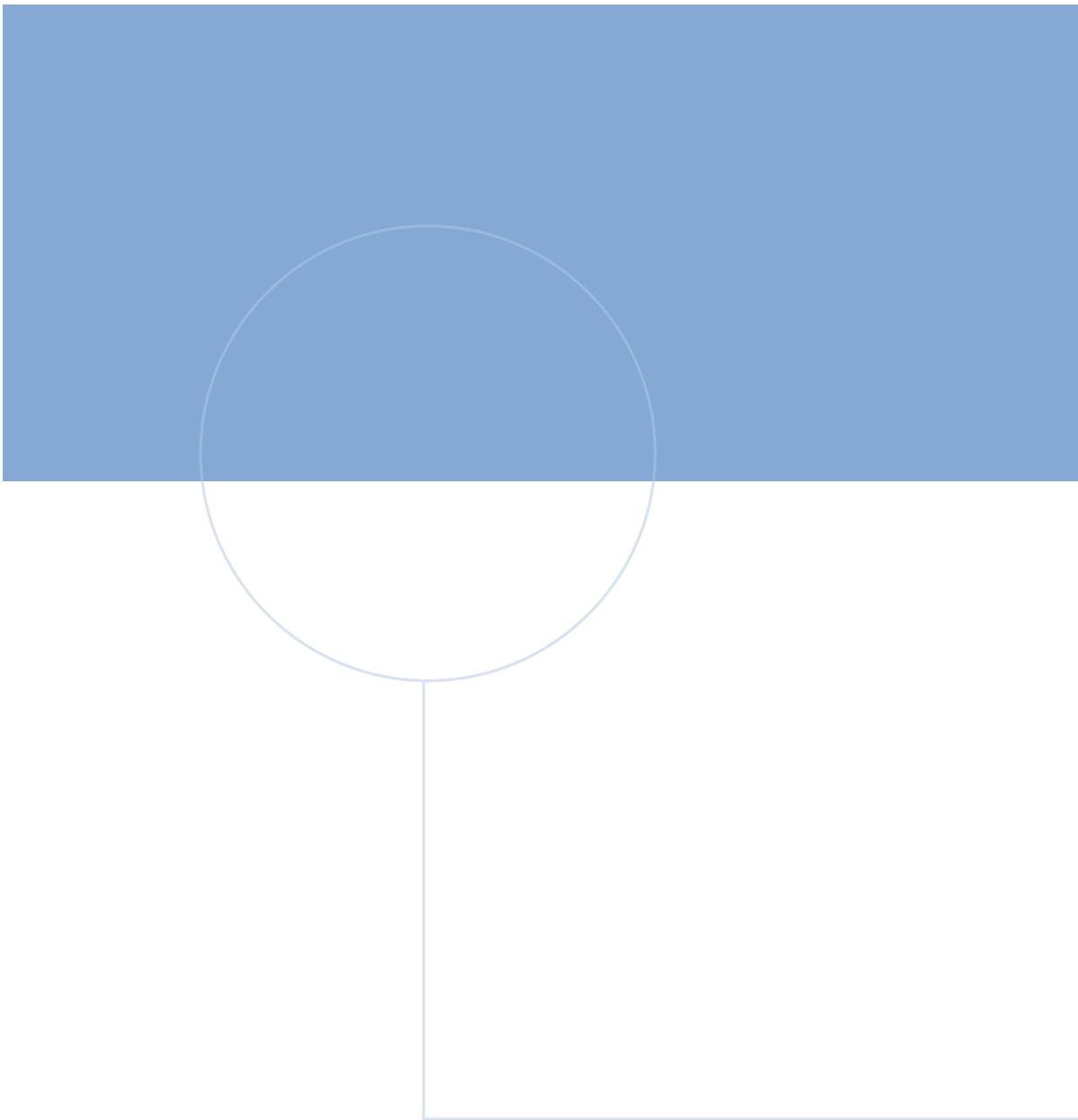
```
                            x4_ci_ms[1], x5_ci_ms[1],
                            x6_ci_ms[1],x7_ci_ms[1]),
                            upper_ci = c(x1_ci_ms[2],
                            x2_ci_ms[2], x3_ci_ms[2],
                            x4_ci_ms[2], x5_ci_ms[2],
                            x6_ci_ms[2],x7_ci_ms[2]))

naive <- data.frame(index = c(1, 2, 3,4,5,6,7),
                        lower_ci = c(naives[1,1], naives[2,1], naives[3,1],
                        naives[4,1],
                        naives[5,1], naives[6,1],
                        naives[7,1]),
                        upper_ci = c(naives[1,2],
                        naives[2,2], naives[3,2],
                        naives[4,2],naives[5,2],
                        naives[6,2],naives[7,2]))

ggplot(combined_data_all, aes(x = index, ymin = lower_ci,
ymax = upper_ci, fill = method)) +
  geom_rect(aes(xmin = index - 0.2, xmax = index - 0.05),
  data = subset(combined_data_all, method == "Model␣and␣signs"),
  color = "gray100") +
  geom_rect(aes(xmin = index - 0.05, xmax = index + 0.1),
  data = subset(combined_data_all, method == "Model␣only"),
  color = "gray100") +
  geom_rect(aes(xmin = index + 0.1, xmax = index + 0.25),
  data = subset(combined_data_all, method == "Unadjusted"),
  color = "gray100") +
  labs(x = "Active␣Set␣Indices", y = "Coefficient␣Values", fill = "␣") +
  scale_x_continuous(breaks = c(1, 2, 3,4, 5, 5, 6, 7)) +
  scale_fill_manual(values = c("cornflowerblue","brown",
  "gray45")) +
  theme_minimal() +
  theme(legend.position = "top", text = element_text(size =
  12))  +
  ylim(c(-2, 2.5)) +
  geom_segment(aes(x = 0.8, xend = 2.5, y = 2, yend = 2),
  linetype = "dashed", color = "midnightblue", size=0.35) +
  geom_segment(aes(x = 2.5, xend = 4.5, y = 0.5, yend = 0.5),
  linetype = "dashed", color = "midnightblue", size=0.35) +
  geom_segment(aes(x = 4.5, xend = 7.3, y = 0, yend = 0),
  linetype = "dashed",
  color = "midnightblue", size=0.35)+
  annotate("segment", x = 6.8745, xend = 6.8745, y = -1.5, yend = -2,
           arrow = arrow(type = "closed", length = unit(0.08,
           "inches")), color = "cornflowerblue")
```

More predictors than observations

```
set.seed(1)
n=25
beta <- c(2,2,0.7,0.7,rep(0,46))
p=length(beta)
sigma=1

X <- matrix(rnorm(n*p),n,p)
X <-scale(X, center = TRUE, scale = TRUE)
```

```
y <- X%*%beta + sigma*rnorm(n)

lassofit <- glmnet(X, y, standardize = FALSE)
lambda <-  12
betas <- coef(lassofit, s = lambda/n, exact=TRUE, x=X, y=y)[-1]
actives <- which(betas != 0)
s<- c(1,1,1,1,-1,1)
results <- lapply(1:6, function(i) {
  myselinf_lasso(X, y, lambda, actives, i, s)
})
result.x1 <- results[[1]]
result.x2 <- results[[2]]
result.x3 <- results[[3]]
result.x4 <- results[[4]]
result.x5 <- results[[5]]
result.x6 <- results[[6]]

#CIs conditioned on model and signs
x1_ci_ms <-CIfunc_tent(result.x1, 1.5, 0.05)
x2_ci_ms <-CIfunc_tent(result.x2, 3, 0.05)
x3_ci_ms <-CIfunc_tent(result.x3, 2.55, 0.05)
x4_ci_ms <-CIfunc_custom(result.x4,-2,0.86,0.05)
x5_ci_ms <-CIfunc_custom(result.x5, -1.5,6, 0.05)
x6_ci_ms <-CIfunc_tent(result.x6, 2, 0.05)
#Adjusting for plot. Cannot be computed.
x4_ci_ms[1]<- -3.9269364

signpatterns <- sign_patterns(6)
#Conditioned on model only
x1_ci_m <- union_CI(X, y, lambda, signpatterns = signpatterns,
j=1, buffer=1.5, alpha=0.05, result.orig = result.x1, actives=actives)
x2_ci_m <- union_CI(X, y, lambda, signpatterns = signpatterns,
j=2, buffer=3, alpha=0.05, result.orig = result.x2, actives=actives)
x3_ci_m <- union_CI(X, y, lambda, signpatterns = signpatterns,
j=3, buffer=2, alpha=0.05, result.orig = result.x3, actives=actives)
x4_ci_m <- union_CI(X, y, lambda, signpatterns = signpatterns,
j=4, buffer=1.7, alpha=0.05, result.orig = result.x4, actives=actives)
x5_ci_m <- union_CI(X, y, lambda, signpatterns = signpatterns,
j=5, buffer=4, alpha=0.05, result.orig = result.x5, actives=actives)
x6_ci_m <- union_CI(X, y, lambda, signpatterns = signpatterns,
j=6, buffer=2, alpha=0.05, result.orig = result.x6, actives=actives)

x1_ci_m<-x1_ci_m$CI_union
x2_ci_m<-x2_ci_m$CI_union
x3_ci_m<-x3_ci_m$CI_union
x4_ci_m<-x4_ci_m$CI_union
x5_ci_m<-x5_ci_m$CI_union
x6_ci_m<-x6_ci_m$CI_union
#x7_ci_m<-x7_ci_m$CI_union

#Naive selection: The lasso has chosen the model
X_actives <- X[, actives]
linear_model <- lm(y ~ X_actives-1)
naives <- confint(linear_model)

modelonly <- data.frame(index = c(1, 2, 3,4,5,6),
                        lower_ci = c(x1_ci_m[1],
                        x2_ci_m[1],
                        x3_ci_m[1],x4_ci_m[1],
                        x5_ci_m[1], x6_ci_m[1]),
```

```
                        upper_ci = c(x1_ci_m[2],
                        x2_ci_m[2], x3_ci_m[2],
                        x4_ci_m[2], x5_ci_m[2],
                        x6_ci_m[2]))

modelandsigns <- data.frame(index = c(1, 2, 3,4,5,6),
                            lower_ci = c(x1_ci_ms[1],
                            x2_ci_ms[1], x3_ci_ms[1],
                            x4_ci_ms[1], x5_ci_ms[1],
                            x6_ci_ms[1]),
                            upper_ci = c(x1_ci_ms[2],
                            x2_ci_ms[2], x3_ci_ms[2],
                            x4_ci_ms[2], x5_ci_ms[2],
                            x6_ci_ms[2]))

naive <- data.frame(index = c(1, 2, 3,4,5,6),
                    lower_ci = c(naives[1,1],
                    naives[2,1], naives[3,1],
                    naives[4,1], naives[5,1],
                    naives[6,1]),
                    upper_ci = c(naives[1,2],
                    naives[2,2], naives[3,2], naives[4,2],naives[5,2],
                    naives[6,2]))

combined_data_all <- rbind(modelandsigns, modelonly, naive)
combined_data_all$method <- rep(c("Model␣and␣signs","Model
only","Unadjusted"), each = 6)

ggplot(combined_data_all, aes(x = index, ymin = lower_ci,
ymax = upper_ci, fill = method)) +
  geom_rect(aes(xmin = index - 0.2, xmax = index - 0.05),
  data = subset(combined_data_all, method == "Model␣and
␣␣signs"), color = "gray100") +
  geom_rect(aes(xmin = index - 0.05, xmax = index + 0.1),
  data = subset(combined_data_all, method == "Model␣only"),
  color = "gray100") +
  geom_rect(aes(xmin = index + 0.1, xmax = index + 0.25),
  data = subset(combined_data_all, method == "Unadjusted"),
  color = "gray100") +

  labs(x = "Active␣Set␣Indices", y = "Coefficient␣Values",
  fill = "␣") +

  scale_x_continuous(breaks = c(1, 2, 3,4, 5, 5, 6, 7)) +
  scale_fill_manual(values = c("cornflowerblue","brown",
  "gray45")) +
  theme_minimal() +
  theme(legend.position = "top", text = element_text(size = 12))+
geom_segment(aes(x = 0.8, xend = 2.5, y = 2, yend = 2),
linetype = "dashed", color = "midnightblue", size=0.35) +
  geom_segment(aes(x = 2.5, xend = 4.5, y = 0.7, yend = 0.7),
  linetype = "dashed", color = "midnightblue", size=0.35) +
  geom_segment(aes(x = 4.5, xend = 6.3, y = 0, yend = 0),
  linetype = "dashed", color = "midnightblue", size=0.35)+
annotate("segment", x = 3.87, xend = 3.87, y = -3,
yend = -4.5, arrow = arrow(type = "closed", length = unit(0.08, "inches")),
color = "cornflowerblue")
```



NTNU

Norwegian University of
Science and Technology