Estimating Error and Prior Variance in a High-Dimensional Ridge Regression Models

With Applications to Bayesian Structural Equation Models for Finding Treatment Targets.

Jurre Roan Veerman (s1601261)

Thesis advisor: Prof. Dr. Mark van de Wiel

MASTER THESIS

Defended on Month Day, Year

Specialization: Statistical Science

STATISTICAL SCIENCE
FOR THE LIFE AND BEHAVIOURAL SCIENCES
Contents

I General Introduction 4

1 Introduction 4
  1.1 High Dimensional Data 4
  1.2 Methods used throughout this thesis 6
    1.2.1 Bayesian Statistics 6
    1.2.2 Empirical Bayes 8
    1.2.3 Shrinkage 9
    1.2.4 Ridge regression 10

II Estimating Error and Prior Variance in a Ridge Setting Using Empirical Bayes 13

2 Introduction 14
  2.1 The Model and goal 14

3 Estimation methods 15
  3.1 Estimating functions of $\sigma^2$ and $\tau^2$ 15
    3.1.1 Estimating $\lambda$ by looCV 15
    3.1.2 Estimating $\lambda$ by Generalized Cross Validation 16
    3.1.3 Estimating heritability by HiLMM 17
  3.2 Direct estimation of $\sigma^2$ 18
    3.2.1 Simple estimate for $\sigma^2$ 18
    3.2.2 Principal Component Regression estimate for $\sigma^2$ 18
  3.3 Joint estimation of $\sigma^2$ and $\tau^2$ 18
    3.3.1 High dimensional Restricted Maximum Likelihood (REML) 19
    3.3.2 Optimizing Marginal Likelihood 19
    3.3.3 Empirical Bayes Method of Moments 22

4 Simulations 28
  4.1 Comparing PRESS, GCV and LooCV 28
  4.2 Selection of initial $\lambda$ estimate for MoM and simple estimate 29
4.3 Assessing performance of estimators

4.3.1 Basic High Dimensional Situation

4.3.2 Sparse model

4.3.3 Bigger N, Bigger P

4.3.4 Low dimensional setting

4.3.5 Including non penalized coefficients

4.3.6 Groupwise estimation using MoM and MML methods

4.3.7 Empirical robustness of the estimators

4.3.8 Using real data

5 Conclusion and Discussion

A Proof for $P_{\gamma}$

B Finding the root to the REML equation, (15)

C Approximation to condition number of $R$

D Improving GRridge performance

E Runtime of MML, MoM and REML

III Using a Bayesian Structural Equation Model to find treatment targets in High Dimensional expression data.

6 Introduction

6.1 Genomics Data

6.2 Goal

7 Models

7.1 BSEM + covariate

7.2 Alternative models

7.2.1 Single Shrinkage INLA

7.2.2 Differential Expression

7.2.3 Logistic Regression
8 Simulations
  8.1 Methods .................................................. 70
  8.2 Results .................................................. 71
    8.2.1 Comparison of methods ............................. 71
    8.2.2 Bayes Factors ....................................... 79
    8.2.3 No Global Shrinkage ................................. 80

9 Conclusion ................................................. 84

10 Acknowledgements ........................................ 86

F Constants added to the pertubated proteins ............. 87
Part I

General Introduction

1 Introduction

In this manuscript we are concerned with High Dimensional Data, i.e. data where the number of variables is (greatly) larger than the number of samples. At the Vrije Universiteit Medical Center this is motivated by the use of omics data. In this thesis focus lies on two distinct projects:

- Estimating error variance and prior variance using Empirical Bayes: In this project new estimators are proposed which jointly estimate the error variance and prior variance of $\beta$ in a linear ridge setting using Empirical Bayes and they are compared with existing methods. This part is more statistically mathematically focused, but also uses various simulation settings to compare the proposed methods to existing method.

- Adding a Treatment feature to the Bayesian Structural Equation Model: In this project the Bayesian Structural Equation Model (Leday et al. [2017]) will be complemented by a treatment variable. The influence of a treatment on one specific gene can be measured while correcting for the underlying gene interaction graph. This part is more applied and uses mostly simulation studies to obtain results.

While working on the Bayesian Structural Equation Model we noticed that the GRridge method estimated the prior variance of $\beta$ quite poorly. Some investigation led us to believe this was due to poor estimation of $\sigma^2$. We then decided to spent time most of or time on improving the estimation of the error and prior variance.

1.1 High Dimensional Data

Much of the statistics research done during the 20th century focuses on datasets where the number of samples ($n$) is quite large and the number of variables/features ($p$) is (much) smaller. Often new methods were researched using asymptotic setting where $p$ remained fixed (and quite small) and $n$ went to infinity. Over the last twenty (or so) years, there is a huge development of data acquisition technologies. Satellites are able to continuously collect data over long periods of time, billions of people use Google or Facebook which collect huge amounts of data per person, technologies such as fMRI and EEG are used to measure brain activity and microarrays were invented to measure the
DNA expression of individuals. New data collecting technologies can be found in many disciplines such as medicine, finance, astrophysics, environmental sciences, psychology, marketing and many more. Besides new data collecting technologies, there are also huge improvements in data storage resources and computing capabilities. This gives rise to the production, storage and processing of an exponentially growing volume of data (Giraud [2015]). The classical view of statistics does not suffice anymore.

In many research settings the amount of variables has grown dramatically, however the amount of samples does not always grow with it. It is hard to find and measure the DNA profiles of thousands of cancer patients, but it may not be hard to measure thousands of variables of one cancer patient. For this reason the number of variables can get much greater than the amount of samples, i.e. $p \gg n$. This is often called High Dimensional Data. This huge amount of data sounds like a blessing, it is however very hard to separate the signal from the noise in the data. This high dimensionality brings a curse.

Furthermore, a well-designed experiment should provide good coverage of the design space. However, the high dimensional space can get very big as there is an exponential increase in its volume causing data points to be isolated in this space. Figure 1 below shows histograms of the euclidean distances between 100 random vectors sampled from $[0, 1]^p$. The figure shows that distances between random vectors grow if $p$ increases. This means the amount of samples has to increase (drastically) to keep an adequate coverage of the sample space, unfortunately this is often practically impossible.

**Figure 1: Curse of Dimensionality**
Because the sample size is often small there is a lot of uncertainty while estimating parameters from high dimensional data, this causes the standard errors of the parameters to get big, resulting in small power and low reproducibility of the research. Fortunately there is a lot of additional information available over the large amount of variables. This information can be used to estimate parameters with less uncertainty. In this thesis, a new method that estimates the uncertainty in the data using the information of the variables will be proposed and existing methods will be explored.

1.2 Methods used throughout this thesis

To obtain a clear understanding of the research in this thesis, some basic statistical methods and ideas have to be understood. This section tries to explain some basic ideas that are often used throughout this thesis.

1.2.1 Bayesian Statistics

In modern statistics there are two dominating branches: 1) Frequentist statistics and 2) Bayesian statistics. Frequentist statistics, often associated with Fishers p-values, NeymanPearson hypothesis tests, and Neymans confidence intervals treats samples as a random function drawn from an underlying probability distribution based on a set of fixed parameters. Ad infinitum new samples can be sampled from this distribution and it’s parameter estimates based on these samples would eventually converge to the true fixed parameter set. This implies, that given strong assumptions, the true parameters can be estimated from a single sample.

In contrast, the Bayesian approach, popularized by Thomas Bayes and Pierre Simon Laplace, treats the obtained data ($y$) as fixed because we condition on this data and its underlying distribution is based on a set of random parameters ($\theta \in \Theta$), conditional on the observed values. Note that $\theta$ can be either one parameter or a vector. Philosophically a Bayesian starts a priori (before seeing the data) with a belief about the distribution of the parameters. He then obtains data and based on this data updates this prior belief into a posterior belief. This is done using Bayes rule:

$$p(\theta|y) = \frac{\mathcal{L}(y|\theta)\pi(\theta)}{p(y)} \propto \mathcal{L}(y|\theta)\pi(\theta),$$

(1)

where $\pi(\theta)$ is the a prior assumed distribution, $\mathcal{L}(y|\theta)$ is the likelihood function of the data, $p(y)$ is the probability of the observed sample and $p(\theta|y)$ is the posterior (conditional) distribution.

The following properties are central concepts in bayesian statistics:
1. **Prior Distribution** $\pi(\theta)$: The bayesian paradigm starts from the premise that a statistician can always assign prior probabilities in some reasonable manner (Grünwald [2007]). These prior distributions may represent prior information about the situation. Often in research there is no prior information available, hence the Bayesian has to resort to his or her subjective belief about the prior distribution. In some subschools of Bayesian statistics, one may also resort to expressing ignorance about the situation by using an uninformative prior, such as an uniform prior on the parameter space. This means all parameters have a priori the same probability of being right. Many different methods exist that try to model the prior as fair as possible. Another method is Empirical bayes in which the prior distribution is estimated from the data, as will be seen 1.2.2. However, in practice, a prior distribution is often chosen on the bases of analytical elegance or computational efficiency and not on how the researcher believes the prior distribution should behave. The subjectivity that modelling the prior distribution brings to the scientific method is often debated, one school of thought finds this a major flaw in the Bayesian approach, while others find it desirable.

2. **Likelihood** $L(y|\theta)$: Suppose in an experiment we observe a dataset $X = (x_1, \ldots, x_n)$. Based on some parametric statistical model/distribution parameterized by some parameters $\Theta$, the likelihood of this data is then a function of the data :

$$L(y|\theta) = \prod_{i=1}^{n} P_\theta(x),$$

which varies with $\theta$. When using a full bayesian approach, the data $y$ affects the posterior inference only through the likelihood function. However, in an empirical bayesian approach the posterior distribution is also affected by the data through the prior distribution. In frequentist statistics the Likelihood function is often maximized to find the most likely $\theta$ from the data.

3. **Posterior Distribution** $p(\theta|y)$: The posterior distribution can be interpreted as the probability of each $\theta$ given that we have observed data $y$. However, the posterior distribution is also implicitly conditioned on the covariates $x$. In bayesian statistics, conclusions about parameter $\theta$, or unobserved data/predictions are made in terms of probability statements (Gelman et al. [1995]). These probability statements are called posterior distributions and are defined by (1). On bases of this distribution we can still find point estimates, by for instance finding the mean and standard deviation of this distribution, or by using the maximum a posterior (MAP) estimate, which finds the point that maximizes the posterior and is therefore equal to the mode of the posterior distribution.
4. Marginal Distribution $p_\theta(\theta|y)$: Maybe we are not interested in all parameters, therefore suppose $\theta = (\theta_1, \theta_2)$ and we are only interested in $\theta_1$. Then $\theta_2$ may be considered a nuisance parameter. We can then look for the conditional distribution of $\theta_1$ given $y$ by first finding the posterior distribution:

$$p(\theta_1, \theta_2|y) \propto L(\theta_1, \theta_2)\pi(\theta_1, \theta_2)$$ (2)

and then averaging over all possible values for $\theta_2$:

$$p(\theta_1|y) = \int_{\theta_2} p(\theta_1, \theta_2|y) d\theta_2$$

This gives us information on $\theta_1$, while taking all uncertainty of $\theta_2$ and the data into account, instead of using a single point estimate for $\theta_2$.

Equation (2) can be rewritten to:

$$p(\theta_1|y) = \int_{\theta_2} p(\theta_1|\theta_2, y)p(\theta_2|y) d\theta_2$$ (3)

Which shows that the marginal posterior distribution of $\theta_1$ is actually a mixture distribution of the conditional posterior distributions given the nuisance parameter, $\theta_2$, using $p(\theta_2|y)$ as a weighting function of all possible different values of $\theta_2$ (Gelman et al. [1995]).

This marginal distribution can be used to maximize its likelihood function and find the most likely $\theta_1$, while taking all uncertainty of $\theta_2$ and the data into account by:

$$\hat{\theta}_1 = \arg \max_{\theta_1} \prod_{i=1}^{n} p(\theta_1|y_i)$$

1.2.2 Empirical Bayes

The roots of empirical bayes can be traced back to work by von Mises in the 1940s and work by Robbins in 1955. However, the “breakthrough” was due Efron in the 1970s (Casella [1985]).

Empirical Bayes is a method of doing bayesian inference in which the prior distribution is estimated from the data and therefore violates the principle that the prior distribution should be chosen independently of the data (Murphy [2012]). A ‘true’ Bayesian would call this non sense, as the prior is by definition the distribution prior to seeing evidence/data. Empirical bayes approximates the hyperparameters of a hierarchical model by their most likely point estimates based on the data, as such it is often seen as the frequentist interpretation of bayesian statistics. Estimating the prior distribution from the data means the data is used twice in the estimation process, which can lead to overfitting of the data.
Empirical Bayes can formally be seen as maximum marginal likelihood, where the marginal likelihood is defined in equation (3). This involves an integral over the parameter space. Generally no closed form solution to this integral exists, hence numerical approximations should be used. However, in the high dimensional case, numerical approximations are often computationally too heavy and one has to resort to specialized methods such as Markov Chain Monte Carlo (MCMC) (Casella [2001]) or Laplace approximations (Shun and McCullagh [1995]).

In contrast to maximum likelihood, there is also a moments based method in estimating the hyperparameters from the data. In this method the theoretic moments of the prior distribution are simply replaced by the empirical moments estimated from the data:

\[ m_k = \frac{1}{n} \sum_{i=1}^{n} y_i^k, \quad \mu_k = \mathbb{E}_\beta[\mathbb{E}_y[y^k|\beta]], \quad (4) \]

where the k'th population moment \( \mu_k \) with respect to expectancy of the data \( X \) and the prior on \( \beta \), which is typically a function of some parameters \( \theta \). The method of moment estimator \( \hat{\theta} \) is then estimated by solving the linear equations:

\[ m_k = \mu_k(\theta) \quad (5) \]

The Empirical Bayes method can be shown to have good frequentistic properties, (see for instance Carlin and Louis [2000], Efron [2010]), so it often used by non-Bayesians. For instance the James-Stein estimator can be derived using Empirical Bayes (Murphy [2012]).

1.2.3 Shrinkage

A shrinkage estimate is an estimate that is shrunken to a target value. An example of a shrinkage estimator is the James-Stein estimator (James and Stein [1961]), which is of the form:

\[ \hat{\theta}(\lambda) = \bar{\theta} - \lambda(\bar{\theta} - \theta_{\text{target}}), \]

where \( \lambda \) is the shrinkage factor. This estimate shrinks \( \theta \) towards \( \theta_{\text{target}} \) depending on how strong the shrinkage factor is, if \( \lambda = 0 \) then \( \hat{\theta}(\lambda) = \bar{\theta} \) and if \( \lambda = 1 \) then \( \hat{\theta}(\lambda) = \theta_{\text{target}} \).

The James-Stein estimator shows that if future predictions on \( p \) different features are to be made, there is a better way than extrapolating from the \( p \) different means, by borrowing information across all \( p \) features. This estimator is favorable because it can be shown that the Mean Squared Error (MSE) of the James-Stein estimator can be smaller (dependent on the choice of \( \lambda \)) than the MSE of the Maximum Likelihood Estimate (MLE), hence better future predictions can be made (Murphy [2012]). The difference between the MSE of the MLE method and the MSE of
the James-Stein estimator can be substantial, especially if $p$ is big and when all true means of the features are alike (Efron and Morris [1977]), which is especially the case in genomics data.

It can be shown that Empirical Bayes is a shrinkage method. Let $X_{ij} \sim N(\mu_j, 1)$ and $\mu_j \sim N(\theta_j, \tau^2)$ for $1 \leq i \leq n$ and $1 \leq j \leq p$. Then the bayes estimate for the sample mean is

$$\hat{\mu_j} = \mathbb{E}(\mu_j | X_j),$$

where, using the definition of the expected value and Bayes rule:

$$\mathbb{E}(\mu_j | X_j) = \frac{\int_{-\infty}^{\infty} \mu_j P(X_j | \mu_j) P(\mu_j) d\mu_j}{\int_{-\infty}^{\infty} P(X_j | \mu_j) P(\mu_j) d\mu_j} = \theta_j + (1 - (n\tau^2 + 1)^{-1}) \left( \frac{1}{n} \sum_{i=1}^{n} X_{ij} - \theta_j \right)$$

This is of the same form as the James-Stein estimate with, $\lambda = (1 - (n\tau^2 + 1)^{-1})$, where the sample means of $X_{ij}$ get shrunken towards $\theta_j$. In high dimensional genetics data we often set $\theta_j = \theta$ i.e. a single mean across all genes.

1.2.4 Ridge regression

Shrinkage methods are also used in regression settings, to improve prediction in comparison with ordinary least square (OLS). OLS finds the vector $\beta$, with $|\beta| = p$, which minimizes the Residual Sum of Squares, $RSS(\beta) = (y - X\beta)^T(y - X\beta)$, or equivalently maximizes the likelihood function:

$$\hat{\beta}_{ols} = \arg \max_{\beta} L(\beta)$$

Analytically minimizing the RSS, leads to:

$$\hat{\beta}_{ols} = (X^T X)^{-1} X^T y$$  \hspace{1cm} (6)

The rank of the data matrix $X$ is at most min$(n, p)$. Consequently the rank of a high-dimensional design matrix is maximally equal to $n$: rank($X$) $\leq n$. The rank of a matrix is the dimension of the column space (Strang [2009]), which implies that columns in the high dimensional design matrix $X$ are linearly dependent. This super-collinearity leads to rank of the $(p \times p)$ dimensional matrix $X^TX$ to also be smaller than $p$ and consequently does not have an inverse. Hence (6) is undefined and the regression parameter $\hat{\beta}_{ols}$ cannot be estimated.

Hoerl and Kennard [1970] proposed an ad-hoc fix to solve the singularity of $X^TX$ by replacing $X^TX$ by $X^TX + \lambda I_{p \times p}$, where $\lambda \in [0, \infty)$. The scalar $\lambda$ is called the penalty parameter. This leads to the ridge estimator of the regression parameters to be defined as:

$$\hat{\beta}(\lambda) = (X^TX + \lambda I_{p \times p})^{-1} X^Ty$$  \hspace{1cm} (7)
Using singular value decomposition we can show that \((X^TX + \lambda I_{p \times p})^{-1}\) is invertible. Let \(X = UDV^T\) be the singular value decomposition of \(X\), where \(D\) is an \(n \times n\) diagonal matrix with the singular values on its diagonal. Then

\[
(X^TX + \lambda I_{p \times p})^{-1} = (VDU^TVD + \lambda I_{p \times p})^{-1}
\]

\[
= (V D^2 V^T + \lambda V V^T)^{-1}
\]

\[
= V (D^2 + \lambda I_{n \times n})^{-1} V^T
\]

Clearly \((D^2 + \lambda I_{n \times n})\) is non singular, hence \((X^TX + \lambda I_{p \times p})^{-1}\) is invertible.

The ridge loss function is defined as:

\[
\hat{\beta}_{ridge}(\lambda) = \arg \min_{\beta} (y - X\beta)^T (y - X\beta) + \lambda ||\beta||^2
\]

### 1.2.4.1 Bayesian Ridge Regression

Ridge regression has a close relation with Bayesian linear regression with a gaussian prior for the regression coefficients. Assume the following conjugate prior within Bayesian Linear Regression: 

\(\beta \sim N(0, \tau^2 I)\) and likelihood \(L(y|\beta) \sim N(X\beta, \sigma^2 I)\). The posterior of \(\beta\) is then:

\[
P(\beta|Y, X) = N(y|X\beta, \sigma^2 I)N(\beta|0, \tau^2)
\]

Then setting \(\tau^2 = \frac{\sigma^2}{\lambda}\), we get

\[
P(\beta|Y, X) = \frac{1}{\sqrt{2\pi \sigma^2}} e^{-\frac{||y - X\beta||^2}{2\sigma^2}} \frac{1}{\sqrt{2\pi \tau^2}} e^{-\frac{||\beta||^2}{2\tau^2}}
\]

\[
\propto e^{-\frac{||y - X\beta||^2}{2\sigma^2} + \frac{\lambda ||\beta||^2}{2\tau^2}}.
\]
Taking the logarithm and maximizing the posterior leads to:

\[
\log(P(\beta|Y, X)) \propto (||y - X\beta||^2 + \lambda||\beta||^2) \frac{1}{2\sigma^2}
\]

\[
\frac{\partial \log(P(\beta|Y, X))}{\partial \beta} = (-2X^T(y - X\beta) + 2\lambda\beta) \frac{1}{2\sigma^2}
\]

\[
0 = (-2X^T(y - X\beta) + 2\lambda\beta) \frac{1}{2\sigma^2}
\]

\[
0 = X^T(y - X\beta) - \lambda\beta
\]

\[
\lambda\beta = X^Ty - X^TX\beta
\]

\[
\lambda = X^Ty\beta^{-1} - X^TX
\]

\[
X^TX + \lambda I_{p \times p} = X^Ty\beta^{-1}
\]

\[
\hat{\beta}_{RIDGE} = (X^TX + \lambda I_{p \times p})^{-1}X^Ty
\]

\[
= (X^TX + \frac{\sigma^2}{\tau^2}I_{p \times p})^{-1}X^Ty.
\]

Thus \(\hat{\beta}_{RIDGE}\) is the MAP estimate when using a zero centered gaussian prior for \(\beta\). Because \(P(\beta|Y, X)\) is a guassian distribution, \(\hat{\beta}_{RIDGE}\) is also the posterior mean.

The parameters are encouraged to be small and shrink to zero by using a zero mean gaussian prior on \(\beta\). Equation (8) shows how the ridge parameter varies with \(\sigma^2\) and \(\tau^2\) by setting \(\lambda = \frac{\sigma^2}{\tau^2}\). When we have prior knowledge to believe \(\beta\) is close to 0 i.e. \(\tau^2\) is small, then \(\lambda\) is big and \(\beta\) will be penalized a lot. If there is a lot of signal in the data and little noise i.e. \(\sigma^2\) is small, then \(\lambda\) is small and there will be a small penalty.

Ridge regression is a shrinkage method for prediction, shrinking the size of the \(\beta_j\) coefficients towards zero, as the penalty parameter \(\lambda \to \infty\): the penalty term then floods the loss function. Because we are trying to minimize this loss function, the beta’s will shrink towards zero. In this thesis ridge regression (Hoerl and Kennard [1970]) will be often used, as ridge regression especially performs well in a case where there are many small correlated effects in the data (Van Wieringen [2015]), which is often the case in genomics data.

There are many different ways of using shrinkage in a regression setting, so called penalized regression (for an overview of penalized regression methods see for example Hastie et al. [2001]). In all penalized regression methods a penalty is added to the residual sum of squares or equivalently subtracted from the likelihood function:

\[
\mathcal{L}(\beta|\lambda) = \mathcal{L}(\beta) - \lambda \sum_{j=1}^{P} f(\beta_j),
\]
where $\lambda \in [0, \infty)$ is the shrinkage/penalty parameter, which determines the strength of the penalty to shrink the estimates for $\beta$ to some target value, which is often zero. Again it can be shown that there always is a $\lambda$ for which the ridge regression estimator outperforms the OLS estimator in terms of MSE (Van Wieringen [2015]). Penalized regression is often used to combat overfitting, which is a typical threat in high-dimensional data analysis. Overfitting refers to the phenomenon of modelling the noise rather than the signal in the data. If a true model only uses few covariates, but data on many more variables is available it is more likely that a linear combinations of all covariates leads to a high likelihood than a combination of the 'true' effective covariates. Hence the model models more than the signal and overfits the data. Overfitting often leads to large regression coefficients, which is then remedied by penalizing these big effects. Many other penalized regression techniques exist, most notably are Lasso regression:

$$\mathcal{L}(\beta | \lambda) = \mathcal{L}(\beta) - \lambda \sum_{j=1}^{p} |\beta_j|,$$

which corresponds to Bayesian regression when emposing a Laplace prior on $\beta$. Lasso Regression has the added benefit that some of the regression parameters will be shrunken to exactly zero and therefore removing these variables from the model. Lasso regression thus has an automatic model selection procedure built in. Elastic Net Regression combines both Ridge regression and Lasso regression:

$$\mathcal{L}(\beta | \lambda) = \mathcal{L}(\beta) - \lambda_1 \sum_{j=1}^{p} (\beta_j)^2 - \lambda_2 \sum_{j=1}^{p} |\beta_j|,$$

which corresponds to Bayesian regression with the elastic net prior (Li and Lin [2010]).

There are multiple reasons we focus on ridge regression in this manuscript:

- **Empirical Bayes is easier for ridge regression as ridge regression has exact moments for it’s estimator.**

- **Ridge Regression is often a good prediction model.** This is especially the case in omics data, where there are very many small correlated effects and the sparsity assumption does not hold.

- **Posterior selection is often a good alternative to Lasso regression.**
Part II

Estimating Error and Prior Variance in a Ridge Setting Using Empirical Bayes

2 Introduction

An estimate of the error variance $\sigma^2$ is often required for statistical methods. However, high dimensional data poses challenges to the estimation procedure of $\sigma^2$. Because the sample size, $n$, is often small in such data, there is a large amount of uncertainty in parameter estimates. Shrinkage estimators can be used to borrow information across variables such that in a ridge model it may be beneficial to estimate the variance over the parameters, $\tau^2$, together with the error variance, $\sigma^2$, to obtain better estimates, as there is much information across genes and little across samples.

2.1 The Model and goal

We focus on high-dimensional linear regression and index variables by $j = 1, \ldots, p$ and samples by $i = 1, \ldots, n$. Then:

\[ y = X\beta + \epsilon \]

\[ \beta_{p \times 1} \sim N(0, \tau^2 I_{p \times p}) \]

\[ \epsilon_{n \times 1} \sim N(0, \sigma^2 I_{n \times n}), \]

where $y = [y_1, \ldots, y_n]^T \in \mathbb{R}^n$ is the vector of observations, $\beta = [\beta_1, \ldots, \beta_p]^T$ corresponds to the effects of the variables and $\epsilon = [\epsilon_1, \ldots, \epsilon_n]^T$ is a vector of gaussian error. Furthermore $X \in \mathbb{R}^{p}$ is a fixed $n \times p$ matrix, where generally $p \gg n$.

The goal in this manuscript is to estimate the error variance, $\sigma^2$, and the variance of the prior on $\beta$, $\tau^2$, and functions based on these parameters such as: a) the ridge penalty parameter: $\lambda = \frac{\sigma^2}{\tau^2}$ and b) heritability: $h^2 = \frac{p \tau^2}{p \tau^2 + \sigma^2}$ (Jiang et al. [2014], Bonnet et al. [2015a]). Here $p$ is the number of genes in the data. Heritability measures the fraction of variation between individuals within a population that is due their genotypes (Visscher et al. [2008]). Furthermore, we want to compare methods that jointly estimate $\sigma^2$ and $\tau^2$ to methods that estimate $\sigma^2$ or functions of $\tau^2$ and $\sigma^2$ directly, using Monte Carlo simulation. This goal was motivated by the Empirical Bayesian Structural Equation Model explained in part II of this manuscript. This method uses a
Bayesian linear ridge regression model for which we initially used the GRridge method (Van de Wiel et al. [2016]), to estimate \( \tau^2 \). The original version of GRridge estimates \( \tau^2 \) after estimating \( \sigma^2 \) by a simple estimate. The method showed that the bad estimation of \( \sigma^2 \) had a big effect on the estimation of \( \tau^2 \). This caused the Bayesian Structural Equation model to perform poorly.

We propose two methods that estimate both \( \sigma^2 \) and \( \tau^2 \) in a fully automatic way using an Empirical Bayes setting. One method uses numerical optimization of the marginal likelihood, the other uses an analytical derivation of the moments starting from the residual sum of squares. Furthermore, we explore extensions to these methods, such as having unpenalized covariates in the model and partitioning covariates into groups based on co-data, and estimating groupwise \( \tau^2_g \).

Performance of the methods is evaluated using various simulation settings.

3 Estimation methods

The literature discusses several methods which either:

1. estimate functions of \( \sigma^2 \) and \( \tau^2 \) directly, without having an estimate for either \( \sigma^2 \) and \( \tau^2 \) (Golub et al. [1979], Bonnet et al. [2015a]);

2. estimate \( \sigma^2 \) directly without estimating \( \tau^2 \) (Cule et al. [2011], Cule and De Iorio [2012]);

3. estimate \( \sigma^2 \) and \( \tau^2 \) jointly (maximizing the marginal likelihood, using a Methods of Moment estimate and a REML estimate by Jiang et al. [2014]) in high dimensional settings, which can be used to calculate the functions in 1.

Below, we discuss these methods.

3.1 Estimating functions of \( \sigma^2 \) and \( \tau^2 \)

3.1.1 Estimating \( \lambda \) by looCV

A benchmark method that is used extensively to estimate \( \lambda \) is leave one out cross validation (looCV). LooCV is an approximately unbiased estimate for the true expected prediction error (Hastie et al. [2001]). In general:

\[
CV(\hat{f}, \lambda) = \frac{1}{n} \sum_{i=1}^{n} L(y_i, \hat{f}^{-i}(x_i, \lambda)),
\]

where \( L \) is a loss function, \( \hat{f}^{-i}(x_i, \text{lambda}) \) is a model fit where the \( i \)’th observation is removed, fitted using each separate \( \lambda \in \lambda \). The \( CV(\hat{f}, \lambda) \) function then provides an estimate of the test
error curve and we pick \( \hat{\lambda} \) that minimizes this function (Hastie et al. [2001]). In our linear ridge setting only a single regression for each \( \lambda \) has to be performed, from which the predicted residual error sum of squares (PRESS) statistic (Allen [1974] can be computed (for a proof see page 593 in Montgomery et al. [2012]).

\[
\text{PRESS} = \sum_{i=1}^{n} \left( \frac{y_i x_i^T \hat{\beta}(\lambda)}{1 - h_{ii}(\lambda)} \right),
\]

where \( h_{ii} \) is the diagonal of the “hat” matrix, \( H = X(X^TX + \lambda I)_{p \times p}^{-1}X^T \). In practice, the first step is finding the optimal \( \lambda \) using cross validation and then the expected prediction error given this \( \lambda \) has to be calculated using cross validation. The PRESS statistics obtains the expected prediction error using only cross validation to specify the optimal \( \lambda \).

We will implement the LooCV using the ‘glmnet’ package as this is a very popular package used for penalized regression (Friedman et al. [2010]). The CV glmnet package standardizes the data, so that to get a comparable \( \lambda \) estimate, \( \lambda_{\text{glmnet}} = \lambda \cdot n/S_y \), where \( S_y = \sqrt{\hat{\sigma}^2(n-1)/n} \) is the empirical standard error of \( y \).

### 3.1.2 Estimating \( \lambda \) by Generalized Cross Validation

Generalized Cross Validation (GCV) is an approximation to the leave one out cross validation estimate for the ridge parameter \( \lambda \) in a linear regression setting, using squared error loss (Hastie et al. [2001]).

Golub et al. [1979] present a GCV equation to directly obtain \( \hat{\lambda} \) without needing an estimate for \( \sigma^2 \) or \( \tau^2 \):

\[
\hat{\lambda}_G = \arg \min_{\lambda} \left[ \frac{1}{n} \| I_{n \times n} - A(\lambda)y \|_2^2 \right] / \left[ \frac{1}{n} \text{Tr}(I_{n \times n} - A(\lambda)) \right]^2,
\]

(10)

where \( A(\lambda) = X(X^TX + n\lambda I)_{p \times p}^{-1}X^T \) and \( \text{Tr}(I_{n \times n} - A(\lambda)) \), can be computed efficiently as explained in Section 3.3.3.3 below.

Equation (10) can be understood as being an invariant and weighted version of the PRESS estimate as is shown in Golub et al. [1979]. Because Golub et al. [1979] uses the mean square error \( \frac{1}{n} \| X\beta - X\hat{\beta}(\lambda) \|_2^2 \) to derive their GCV estimate, in which the sum of squares is divided by \( n \) the authors set \( \lambda_{\text{Golub}} = \frac{\sigma^2}{n\tau^2} \) (Golub et al. [1979]). This is different from equation (8), which does not divide by \( n \). To compare the method by Golub with other methods, from now on we set \( \hat{\lambda}_{\text{Golub}} = \hat{\lambda}_G \cdot n \).
3.1.3 Estimating heritability by HiLMM

Heritability is estimated by $\frac{p \cdot p_1 \tau_0^2}{p \cdot p_1 \tau_0^2 + \sigma^2}$, Not all columns in $X$ necessarily have an effect on $y$. Then the distribution of the coefficients can be a mixture distribution with a dirac delta function, i.e. a spike and a gaussian 'slab' component. In context of this sparse model the previous definition of heritability reduces to:

$$h^2 = \frac{p \cdot p_1 \tau_0^2}{p \cdot p_1 \tau_0^2 + \sigma^2},$$

where $p_1 = 1 - p_0$. Because under prior $\pi(\beta) = p_0 \delta_0 + (1 - p_0)N(0, \tau_0^2)$, we have:

$$\text{var}(\beta) = E(\beta^2) - E(\beta)^2 = (1 - p_0)\tau_0^2 - 0 = p_1 \tau_0^2$$

Bonnet et al. [2015a] propose a method which estimates heritability directly, using maximum likelihood.

Let $p_0 = 0$ i.e. the coefficients come from a regular gaussian distribution. Then:

$$y \sim N(0, h^2 \sigma^2 R + (1 - h^2)\sigma^2 I_{n \times n},)$$

where $\sigma^2 = p \tau^2 + \sigma^2$. $R = XX^T/p$ and $U$ is an orthogonal matrix with $U^T U = UU^T = I_{n \times n}$ and $URU^T = \text{diag}(\lambda_1, \ldots, \lambda_n)$, where the $\lambda_i$’s are the eigenvalue of $R$. The heritability is then estimated as:

$$h^2 = \arg \max_{h^2} \left( - \log \left( \frac{1}{n} \sum_{i=1}^{n} \frac{\tilde{y}_i^2}{h^2(\lambda_i - 1) + 1} \right) - \frac{1}{n} \sum_{i=1}^{n} (\log(h^2(\lambda_i - 1) + 1)) \right),$$

where $\tilde{y}_i$ are the components of $\tilde{y} = U^T y$.

Bonnet et al. [2015b] propose an extension to the above method in a sparse mixed model setting. This method starts with a variable selection step, followed by the above maximum likelihood setting on the reduced set of variables. In the variable selection step components are selected that are most correlated to the response $y$, using a Sure Independence Screening approach. The resulting reduced design matrix is then used to further select non null columns. This is done by randomly splitting the data in subsamples of size $n/2$, on each sub sample LASSO regression is implemented and the selected variables are stored. Then for a given threshold, the final set of selected variables appearing on average more often than this threshold will be kept in the final estimation. This threshold is selected by finding a range of threshold values for which the heritability becomes stable. This is a non automated way and this makes it hard to implement in a simulation setting. Furthermore, there are many settings in our simulations which are not sparse. Hence the HiLMM algorithm without the variable selection is used in our simulations.
3.2 Direct estimation of $\sigma^2$

The methods in this section estimate $\sigma^2$ without either estimating $\tau^2$ or taking any information about $\tau^2$ into account.

3.2.1 Simple estimate for $\sigma^2$

The most simple way of estimating $\sigma^2$, and often used in practice, is using equation 5 in Cule et al. [2011]:

$$\hat{\sigma}^2 = \frac{(y - X\hat{\beta})^T(y - X\hat{\beta})}{\nu},$$

(13)

which is the residual mean square error in a linear model, with $\nu$ is the residual effective degrees of freedom. Often, $\nu = n - \text{tr}(2H - HH^T)$ is used (Hastie and Tibshirani [1990]). Here $H = X(X^TX + \lambda I)^{-1}X^T$ is the “hat matrix”. $\lambda$ in the hat matrix is counterintuitive as in a bayesian sense $\lambda = \frac{\sigma^2}{\tau^2}$. Hence to estimate $\sigma$ we implicitly need an estimate for $\tau^2$. Fortunately, as will be empirically shown in Section 4.3.1, the $\lambda$ estimate does not affect the final $\sigma^2$ much.

3.2.2 Principal Component Regression estimate for $\sigma^2$

Cule and De Iorio [2012] introduce a new estimator for $\sigma^2$ based on Principal Component Regression (PCR), in which they consider a regression situation where most of the variance in $X$ is explained by it’s first $r$ principal components. In PCR an eigendecomposition of $X^TX$ is done, such that $X^TX = Q\Lambda Q^T$. Taking $Z = XQ$ and $\hat{\alpha} = Q^T\hat{\beta}$, we have that $y = Z\alpha + \epsilon$. Here, $Z$ is reduced from $p$ columns to $r$ principal components, which is upperbounded by $\min(n,p)$. Here, $r$ is a hyperparameter that affects the estimation procedure. The method in Cule and De Iorio [2012] tunes $r$ such that the degrees of freedom for variance of the model, $\text{tr}(HH^T)$, is equal to the number of components used in the computation of the ridge penalty $\lambda_r$. In ridge regression this is done by noting that $\text{tr}(H(\lambda)H(\lambda)^T) = \sum_{j=1}^{p} \frac{\Lambda_j^2}{(\lambda_j + \lambda)^2}$, where $\Lambda$ are the eigenvalues and $\lambda$ is the ridge penalty and then using numerical methods to solve this. Using this model, again $\sigma^2$ is estimated by the residual mean square error of the model:

$$\hat{\sigma}_r^2 = \frac{(y - Z_r\hat{\alpha}_r)^T(y - Z_r\hat{\alpha}_r)}{n - r}.$$  

(14)

3.3 Joint estimation of $\sigma^2$ and $\tau^2$

In a high dimensional setting it could be beneficial to use the information in the horizontal direction (over the variables) while estimating the variance in the vertical direction (over the samples) when
\( p \gg n \). In addition, the two parameters are intrinsically connected. Therefore it could be very useful to estimate \( \sigma^2 \) and \( \tau^2 \) together directly. Methods that implement joint estimation will be explored in the section below.

### 3.3.1 High dimensional Restricted Maximum Likelihood (REML)

Jiang et al. [2014] use REML to estimate the error variance in a linear mixed model (LMM) in a high dimensional setting. REML uses a contrast to kick out the influence of fixed effects. Let \( A \) be an \( n \times m \) matrix such that \( \text{rank}(A) = m \) and \( A^T X = 0 \). Where \( m \) is the number of fixed effects in the model. Define \( z = A^Ty \).

In general the effect of \( A \) is shown below: Let, \( H_{\text{fixed}} = X_{\text{fixed}}(X_{\text{fixed}}^TX_{\text{fixed}})^{-1}X_{\text{fixed}}^T \) be the “Hat” matrix of \( X_{\text{fixed}} \) in regular linear regression. Then setting \( A = I - H \), gives:

\[
(I - H)X = X - HX = X - X = 0
\]

\[
(I - H_{\text{fixed}})y = y - H_{\text{fixed}}y = y - \hat{y}_{\text{fixed}}
\]

By which we remove the influence of the fixed effects on \( y \). Then \( z \sim N(0, A^TVA) \), with \( V = \tau^2 I_{n \times n} + \sigma^2 ZZ^T \). By differentiating the restricted log-likelihood they then obtain an estimate for \( \gamma = \lambda^{-1} = \tau^2 / \sigma^2 \) and use this parameter to further estimate \( \sigma^2 \).

Using the above method Jiang et al. [2014] then proceeds by setting \( y = \tilde{X}\beta + \epsilon \), where \( \tilde{X} = p^{-1/2}X \) and \( X \) is the data matrix. The REML estimator, \( \hat{\gamma} \), is then the solution to:

\[
\frac{y^TP_\gamma \tilde{X}^TP_\gamma y}{\text{tr}(P_\gamma)} = \frac{y^TP_\gamma^2y}{\text{tr}(P_\gamma)}
\] (15)

In context of model (9) \( P_\gamma \) reduces to \( P_\gamma = V_\gamma^{-1}(I_{n \times n} - \tilde{V}V) \), with \( V_\gamma = I_{n \times n} + \gamma \tilde{X} \tilde{X}^T \) and \( \tilde{V} \), a \( n \times n \) matrix with all elements equal to \( \frac{1}{\sum_{i,j=1}^n v_{ij}} \) (see Appendix A for a proof). The solution is used to obtain the REML estimator, \( \hat{\sigma}^2 \) according to: \( \hat{\sigma}^2 = \frac{y^TP_\gamma^2y}{\text{tr}(P_\gamma)} \).

### 3.3.2 Optimizing Marginal Likelihood

A very simple Empirical Bayes estimate is obtained by maximizing the marginal likelihood (MML) over \( \tau^2 \) and \( \sigma^2 \). In machine learning literature this is often called the evidence procedure (Murphy [2012]). This method corresponds to:

\[
\arg \max_\theta \int_\beta L(y; \beta)p_\theta(\beta)d\beta,
\]

19
where $\theta = \{\sigma^2, \tau^2\}$. If we assume a gaussian prior on the coefficients, the marginal likelihood can be found analytically due to the conjugacy with the gaussian likelihood of the data. Hence no integration over $\beta$ space is needed. Since $P(y|\beta) \sim N(X\beta, \sigma^2 I_{n \times n})$ and $P(\beta) \sim N(0, \tau^2)$, $P(y)$ is normally distributed, $P(y)$ can be derived analytically, by deriving the first two moments.

$$E_y[y] = E_{\beta}[E_y[y]] = E_{\beta}[X\beta] = 0$$

$$\text{Var}_y[y] = E_{\beta}[\text{Var}_y[y]] + \text{Var}_{\beta}[E_y[y]]$$

Where, $E_{\beta}[\sigma^2 I_{n \times n}] = \sigma^2 I_{n \times n}$ and $\text{Var}_{\beta}[X\beta] = XX^T \tau^2$, so

$$P(y) = N(y, \mu = 0, \Sigma = XX^T \tau^2 + \sigma^2 I_{n \times n})$$

$$= \det (2\pi \Sigma)^{\frac{1}{2}} e^{-\frac{1}{2}(y-\mu)^T \Sigma^{-1} (y-\mu)}$$

This can easily be maximized over $\sigma^2$ and $\tau^2$ while requiring the estimates to be nonnegative. In some settings $n$ and $p$ can both grow very large. As can be seen in Equation: (17), both $\Sigma$ has to be inverted and the determinant of $2\pi \Sigma$ has to be calculated at every optimization step. Matrix inversion has complexity $O(n^3)$ using Gauss-Jordan elimination and calculating the determinant of $\Sigma$ also has complexity $O(n^3)$ using LU decomposition, which is implemented in the R function ‘det()’.

The optimization could possibly be done more efficient using a singular value decomposition:

$$\Sigma^{-1} = (XX^T \tau^2 + \sigma^2 I_{n \times n})^{-1}$$

$$= \tau^{-2}(XX^T + \lambda + I_{n \times n})^{-1}$$

$$= \tau^{-2}V(D^2 + \lambda I_{n \times n})^{-1}V^T,$$

in which no longer the inverse of the $n \times n$ variance matrix has to be calculated, but an inverse on a diagonal matrix containing the singular values $D^2$. However, $V$ is a $n \times n$ matrix, and matrix multiplication has complexity: $O(n^2)$. So nothing is won if $n$ is large.

This marginal likelihood method is equal to the above REML method in context of model: 32 and difference in the results are likely due to computational differences. In a mixed effects model, where there are fixed effects added to the model the two methods differ. As REML uses a contrast to kick out the influence of these fixed effects, for the marginal likelihood we studied other methods which aim to remove the influence of these effects and will be shown in Paragraph: 3.3.2.1. Jiang [2007] and Jiang et al. [1996] show asymptotics of the REML and MML method as the number of fixed effects: $m \to \infty$. They show that the marginal maximum likelihood estimator has similar asymptotic properties to the REML method, if the number of fixed effects, $m$ remains bounded.
or increases at a slower rate than the sample size, $n$, as the normalized difference converges to zero. Furthermore, they show that the MML estimate may be inconsistent if $m$ grows with $n$. The MML generally has a bias that occurs because the MML has to estimate all variance parameters and fixed effect (nuisance) parameters. This bias does not disappear if the sample size increases, when $m$ stays proportional to the sample size. The REML method does not estimate the fixed effect coefficients parameters and therefore only estimates the parameters of main interest. In practical settings $m$ is often (much) smaller than $p$ and perhaps not much care has to be given to the effect of the fixed effects.

3.3.2.1 Including Non Penalized coefficients In a practical setting one may not want to put a shrinkage prior on all coefficients, but perhaps have a few unpenalized “fixed” coefficients in the model. This means these coefficients should not be used while estimating the ridge penalty $\lambda$, or equivalently $\tau^2$ and $\sigma^2$. The unpenalized coefficients can be interpreted as fixed as they do not have the zero centered gaussian shrinkage prior that the penalized coefficients do have. This leads to a mixed effects model.

The MML method has to be adjusted to estimate the variances given these fixed effects. We consider various methods:

- The most simple method naively uses the original method and treats the fixed effects as random.
- Estimating the fixed effects in the optimization: Then the optimization process is no longer on the two variance parameters, but on the $2+m$ parameters which will be quite computationally complex if $m$ is large.
- Trying to remove the zero fixed coefficients in a stepwise manner using a criterion such as the AIC or the BIC.

3.3.2.2 Groupwise $\tau^2$ estimation Model (9) corresponds to a model where all $p$ variables come from the same prior distribution. One may have external data or prior information. For instance a previous study that found that some variables are more important and some variables are less important, meaning there are two group of variables that could come from a different prior distribution. We know for instance that genes don’t act on their own, but they act in pathways, these could be a priori defined and the variables could be partitioned into $G$ groups, $(G_1 \ldots G_G)$ of sizes $(A_1 \ldots A_G)$. For which $\forall \theta_{g \in G}$ have a different prior.
The \( G + 1 \) parameters \((\tau^2_g, \sigma^2)\) can be found by rewriting equation 16 to a solution for \( \tau^2 \):

\[
P(Y) = N(Y, \mu = 0, \Sigma = XTX^T + \sigma^2 I_{n \times n}),
\]

where \( T \) is a \( p \times p \) diagonal matrix with all \( \tau^2_g \)'s on the diagonal.

### 3.3.3 Empirical Bayes Method of Moments

In this section an alternative estimation method is using the method of moments (MoM). To estimate \( \sigma^2 \) we start by interpreting the residual sum of squares (RSS) as a measure of variance, with \( \hat{\beta} = \beta_{\text{init}} \) which can be estimated from the data by an initial ridge regression fit. The expectation of this RSS is taken with respect to \( Y \) as is often done in an empirical bayes setting.

\[
\|y - X\hat{\beta}\|^2 \approx E_y[\|y - X\beta\|^2]
\]

\[
= E_y[(y - X\beta + X\beta - X\hat{\beta})^T(X\beta - X\hat{\beta})] + 2E_y[(y - X\beta)^TX\beta] - 2E_y[y^TX\hat{\beta}]
\]

\[
= E_y[\|y - X\beta\|^2] + E_y[\|X\beta\|^2] + E_y[\|X\hat{\beta}\|^2] - 2E_y[(X\beta)^TX\hat{\beta}] - 2E_y[y^TX\hat{\beta}] := A + B + C + D
\]

Where both the cross-term \( 2E_y[(y - X\beta)^TX\beta] \) has cancelled because \( E_y[y] = X\beta \) and \( E_y[(X\beta)^TX\hat{\beta}] \) has cancelled. Each term in (20) will be worked out separately.

For A:

\[
E_y[\|y - X\beta\|^2] = E_y[\varepsilon^T\varepsilon] = \text{tr}(\Sigma) = n\sigma^2,
\]

where we used that the expectation of the quadratic form of a multivariate random variable, \( \varepsilon \), is \( E[\varepsilon^T\Lambda\varepsilon] = \text{tr}(\Lambda \Sigma) + \mu^T\Lambda \mu\), with \( \mu = E[\varepsilon] \) and \( \Sigma = \text{cov}[\varepsilon] \), see Mathai and Provost [1992].

As we are in a bayesian setting, a prior is put on \( \beta \), to take the uncertainty of the coefficients into account. This means we can take the expected value with respect to \( \beta \). In a linear ridge
regression setting, \( \beta \sim N(0, \tau^2) \). For \( B \) in (20):

\[
E_\beta[E_y[\|X\beta\|^2]] = E_\beta[E_y[\|X\beta\|^2]] = E_\beta[\|X^T X \beta\|] = \tau^2 tr(X^T X) \quad (22)
\]

Continuing with \( C \) in (20):

\[
E_y[\|X\hat{\beta}\|^2] = E_y[\hat{\beta}^T X^T X \hat{\beta}]
\]

\[
= E_y[y^T X (X^T X + \lambda I_{p \times p})^{-1}] X^T X [(X^T X + \lambda I_{p \times p})^{-1} X^T y]
\]

\[
= E_y[y^T H^2 y]
\]

\[
= \sigma^2 tr(H^2) + \beta^T X^T H^2 X \beta
\]

Again we take the expectation with respect to \( \beta \):

\[
E_\beta[\sigma^2 tr(H^2) + \beta^T X^T H^2 X \beta] = \sigma^2 tr(H^2) + \tau^2 tr(X^T H^2 X) \quad (23)
\]

where, \( H \) is again the Ridge "Hat" matrix: \( X(X^T X + \lambda I_{p \times p})^{-1} X^T \).

For \( D \) in (20):

\[
-2E_\beta[E_y[y^T X \hat{\beta}]] = -2E_\beta[E_y[y^T H y]]
\]

\[
= -2(\sum_{j=1}^p \lambda_j) + \beta^T X^T H X \beta
\]

\[
= -2\sigma^2 tr(H) - 2\tau^2 tr(X^T H X) \quad (24)
\]

Adding (21), (22), (23) and (24) together yields the final solution:

\[
E_y[\|y - X\hat{\beta}\|^2] = n\sigma^2 + \tau^2 tr(X^T X) + \sigma^2 tr(H^2) + \tau^2 tr(X^T H^2 X) - 2\sigma^2 tr(H) - 2\tau^2 tr(X^T H X) \quad (25)
\]

Van de Wiel et al. [2016] derived a solution to \( \tau^2 \) given below:

\[
\tau^2 = \frac{\sum_{j=1}^p (\hat{\beta}_j^2)/\hat{\Sigma}_{jj} - p\sigma^2}{\sum_{j,l=1}^p C_{jl}/\hat{\Sigma}_{jj}} \quad (26)
\]

Where \( C = (X^T X + \lambda I_{p \times p})^{-1} X^T X \) and \( \hat{\Sigma} = (X^T X + \lambda I_{p \times p})^{-1} X^T X (X^T X + \lambda I_{p \times p})^{-1} \)

Using (25) and (26) yields two linear equations and two unknowns (\( \sigma^2 \) and \( \tau^2 \)) which can be solved. Slightly rewriting (25) and (26):

\[
\sum_{j=1}^p (\hat{\beta}_j^2)/\hat{\Sigma}_{jj} = \sigma^2 [n + tr(H^2) - 2tr(H)] + \tau^2 [tr(X^T X) + tr(X^T H^2 X) - 2tr(X^T H X)]
\]

\[
\sum_{j,l=1}^p C_{jl}/\hat{\Sigma}_{jj} = \sigma^2 p \left( \sum_{j,l=1}^p C_{jl}/\hat{\Sigma}_{jj} \right)^{-1} + \tau^2 \quad (27)
\]
The solution to this set of equations yields the new $\sigma^2$ and $\tau^2$ estimates. However, these estimates may be negative. As a simple fix a negative estimate is set to zero and plugged into the linear equations.

This method is very similar to the GRridge method, in that it uses $\sigma^2$ to estimate $\tau^2$ while using the simple estimate for $\sigma^2$. However, there are some differences. This new method simultaneously estimates both $\tau^2$ and $\sigma^2$, where GRridge first estimates $\sigma^2$ by the Simple method formulas and then $\tau^2$. So this $\sigma^2$ estimate does not depend on $\tau^2$. The GRridge method is effectively a simple $\sigma^2$ and a MoM $\tau^2$.

There are also some similarities with looCV and GCV. LooCV approximates the expected prediction error $\text{EPE} = \mathbb{E}_y(y - X^T\beta)$. Furthermore, GCV is a weighted version of the PRESS statistic, which can be shown to be equal to LooCV (Montgomery et al. [2012]). Hence both methods try to approximate the EPE and explicitly minimizing this quantity. The MoM estimate approximates the EPE including a prior on $\beta$ and taking the expectation over this distribution by the residual sum of squares in the data. Equal to the GRridge method, this method can (easily) be extended to a logistic regression setting and survival cox regression setting as the theoretical moments are also known for these types of regression.

As the derivations show, both $\tau^2$ and $\sigma^2$ are connected to each other. Even though one might expect, it is important to note that while the expectancy with regards to $\beta$ is being taken, the proposed method does not need strong parametric assumptions on $\beta$, the prior distribution of $\beta$ needs a defined expected value and covariance matrix for the result from Mathai and Provost [1992] to hold. An initial ridge fit is done on the data, which implicitly means the errors come from a gaussian distribution. Thus some implicit distributional assumptions are needed.

3.3.3.1 Groupwise $\tau^2_g$ estimation The MoM can be rewritten such that groupwise estimates for $\tau^2$ can be obtained. This is done by using equation 9 in Van de Wiel et al. [2016] where estimates for $\tau^1_1 \ldots \tau^2_G$ are obtained using the following linear system:

$$
\sum_{k \in \mathcal{G}_a} (\hat{\beta}_k^2/\hat{\Sigma}_{kk} - \sigma^2) = \sum_{k \in \mathcal{G}_a} \hat{\Sigma}_{kk}^{-1} \left[ \sum_{h=1}^{G} \sum_{l \in \mathcal{G}_h} c_{hl}^2 \tau^2_h \right]
$$

(29)
The $\sigma^2$ equation can be rewritten to include a diagonal matrix $T_{p \times p}$, with on it’s diagonal $\tau_g^2$, then using the expectation of the quadratic form of a multivariate random variable:

$$
E_y[\|y - X\hat{\beta}\|^2_2] = \sigma^2[n + \text{tr}(H^2) - 2\text{tr}(H)] + [\text{tr}(X^THT^2XT) - 2\text{tr}(X^THXT)] \\
= \sigma^2[n + \text{tr}(H^2) - 2\text{tr}(H)] + \sum_{k,l \in G} \tau_k^2[\text{tr}([X^TX]_{kl}) + \text{tr}([X^TH^2X]_{kl}) - 2\text{tr}([X^THX]_{kl})]
$$

(30)

Where the indices $kl$ select a block in the matrix to which the partition belongs. Equations (29) and (30) then form a linear equation which can be solved for $\{\sigma^2, \{\tau_g^2\}_{g=1}^G\}$. The same problem as the overall $\tau^2$ of above holds, where the estimates can become negative. This was easily solved for a single $\tau^2$, but is hard to solve having multiple $\tau_g^2$’s. The diagonal matrix $T$ corresponds to a covariance matrix on the coefficients of all $p$ variables that are independent as for $\forall_{i,j,i \neq j}\text{cov}[\beta_i, \beta_j] = 0$ which is likely not the case in high dimensional data. Future investigation is possible where one adds a structure to $T$ that models the covariance between coefficients.

The set of linear equations can return some of the variances being negative. This can be solved by writing the linear equations into an optimization problem and optimizing the parameters while bounded to be nonnegative, unfortunately this goes beyond the scope of this thesis.

### 3.3.3.2 Finding initial $\lambda$ estimate

The method of moments needs an initial value for $\lambda$ to estimate $\sigma^2$ and $\tau^2$. This sounds counterintuitive, as $\lambda = \frac{\sigma^2}{\tau^2}$, hence to estimate the variance you need to know the ratio between the two. There are multiple sensible options for the selection of the initial penalty parameter.

- Setting $\lambda = 1$: This means that $\sigma^2 = \tau^2$ which can be interpreted as being a relative uninformative prior $\tau^2$
- Setting $\lambda = \lambda_{\text{CV}}$: This is done by estimating $\lambda$ from cross validation and corresponds to an informative prior on the relative size difference between $\sigma^2$ and $\tau^2$. This could most efficiently be done by using Generalized Cross Validation. GCV occasionally estimates $\hat{\lambda}_{\text{GCV}} = 0$, which leads to the undefined OLS estimates. This is fixed by setting $\hat{\lambda}_{\text{GCV}} = 1$.
- Choosing $\lambda$ on the basis of the condition number of $X^TX + \lambda I_{p \times p}$: Let $R = X^TX + \lambda I_{p \times p} = Q\Lambda Q^{-1}$, where $\Lambda$ holds the eigenvalues of $R$ and $Q$ holds the eigenvectors of matrix $\text{’}R$. Then the condition number of $R$ is $C(R) = \frac{\max(\Lambda)}{\min(\Lambda)}$. Appendix: C shows an approximation to the condition number which is computationally less expensive, as $R$ is an $p \times p$ matrix. To find an optimal $C(R)$ a condition plot has to be made, where on the X axes a grid of $\lambda$
is plotted and on the Y axes $C(R)$ is plotted. Finding $\lambda$ on basis of $C(R)$ has two different interpretations:

- Picking $\lambda$ such that $R$ becomes well conditioned: This corresponds to finding the smallest $\lambda$ such that $R$ becomes well conditioned and hence no longer singular. The ‘right’ $\lambda$ is then visually chosen at the elbow of the condition number plot. There is also an automatic selection criteria (used for our simulation studies), where a line is drawn from the first point of the curve to the last point of the curve, for each point in the curve the smallest distance is calculated to this line and the point with the biggest distance is chosen. This choice of $\lambda_{\text{init}}$ is then dependent on the points on the curve and therefore the $\lambda$ grid that is used to calculate the condition numbers.

- Picking $\lambda$ such that the approximate loss of accuracy is 2 digits: This is done by $\arg\min_{\lambda, s.t. L=2} C(R)$, where $L = \lfloor \log_{10} C(R) \rfloor$. This provides an estimate of the digits of accuracy one can expect to loose when doing calculations with $R$ (Peeters et al. [2016]). Making computations prone to very large numerical errors.

Figure: 2 shows a condition plot with all methods highlighted on a single dataset with $n = 100, p = 1000, \tau^2 = 0.01, \sigma^2 = 10$. The researcher could be visually aided by looking at an approximation to the second derivative of this curve as is shown in Peeters et al. [2016].

![Condition plot with all methods highlighted](image)

Figure 2: Condition plot highlighting all methods for selection of initial $\lambda$. The blue line is the tangent line used for the distance to line method and the black dotted line is the smallest distance between the furthest away point from the line and the line.

Section: 4.2 shows that the choice of the initial $\lambda$ estimate does not affect the final estimate for $\tau^2$ and $\sigma^2$ much.
3.3.3.3 Efficient implementation

To calculate the new $\sigma^2$ estimate, $(X^TX + \lambda I_{p \times p})^{-1}$ has to be evaluated numerically. As we are in the high dimensional data setting, $p > n$, where $p$ could easily be 40000, this will require inversing a $40000 \times 40000$ dimensional matrix. To work around this we use singular value decomposition of $X = UDV^T$, where we define $R = UD$. Because there are at most $\min(n, p)$ non zero singular values, we can reduce $R$ to an $(n \times n)$ matrix, with both $U$ and $D$ also $(n \times n)$ matrices.

The “Hat” matrix can be written in terms of $R$ and $V$:

$$H = X(X^TX + \lambda I_{p \times p})^{-1}X^T$$

$$= RV^T(VR^TV + \lambda VV^T)^{-1}VR^T$$

$$= RV^TV(R^TR + \lambda I_{n \times n})^{-1}VR^T$$

$$= R(R^TR + \lambda I_{n \times n})^{-1}R^T.$$  

This means $R^TR + \lambda I_{n \times n}$ will be $(n \times n)$ dimensional.

Furthermore, to calculate the trace of a matrix product, one is only interested in the diagonal of the matrix product. Hence it is not needed to calculate the off diagonal elements of the matrix product. An efficient way to calculate the trace of a matrix product between $A$ an $(n \times p)$ dimensional matrix and $B$ an $(p \times n)$ matrix is as follows:

$$\text{tr}(AB) = \sum_{i,j}^{n,p} (A \circ B^T)_{i,j},$$

where $\circ$ is known as the hadamard product, or element wise product.

Computation time can also be more efficient by calculating $A_\Sigma = \sum_{i,j} C_{ij}^j/\bar{\Sigma}_{jj}$, where $C$ was defined as: $C = (X^TX + \lambda I_{p \times p})^{-1}X^TX$. If we define the left matrix $L = \text{diag}(1/\sqrt{\Sigma_{kk}})(X^T X + \lambda I_{p \times p})^{-1}X^T$ and right matrix $R = X^T$, where $L$ is $1 \times n$ and $R$ is $n \times p$. Then denote the sum of all elements of matrix $A = A_\Sigma = \sum_k \sum_l A_{kl}$, then:

$$A_\Sigma = (L^TL) \circ (RR^T).$$

For a full proof see Proposition: 1 in Van de Wiel et al. [2016].

Again the left matrix $L$ can be computed efficiently using the singular value decomposition similar to the efficient calculation of $H$ as is shown above.

In comparison to the MML method, the MoM method only has to invert once which makes this method computationally more interesting, especially in a setting where both $n$ and $p$ are big.
4 Simulations

In this section results will be shown of various simulation studies, to find the effect of the initial $\lambda$ estimate on the Method of Moments estimates and the Simple estimates but also to compare all methods explained above. All simulations were done with 100 iterations. As a general guideline $\sigma^2$ will be chosen around $\tau^2 \times p$, in order for neither of the variances to flood the other, which will cause the methods to overestimate one of the variance and underestimate the other.

4.1 Comparing PRESS, GCV and LooCV

Because LooCV chooses it’s own grid for $\lambda$, the chosen $\lambda$ is not identical to the exact solution of the PRESS method. When using this $\lambda$ grid while minimizing the PRESS equation the results between glmnet and PRESS are equal. The simulation setting is as follows:

$$y = X\beta + \epsilon$$

$$X_{n \times p} \sim \mathcal{N}(0, 1)$$

$$\beta_{p \times 1} \sim p_0\delta_0 + (1 - p_0)\mathcal{N}(0, \tau_0^2)$$

$$\epsilon_{n \times 1} \sim \mathcal{N}(0, \sigma^2)$$

Figure: 3 below shows there are some difference between the three methods in estimating $\lambda$. As the differences are so small in both the sparse setting with $p_0 = 0.9$ and non sparse setting with $p_0 = 0$ and the ‘glmnet’ package is so popular we chose to further only use the GCV method and the LooCV using glmnet in our further comparisons.
4.2 Selection of initial $\lambda$ estimate for MoM and simple estimate

Both the MoM and Simple Estimate need initial estimate for $\lambda$ to yield a final result for $\sigma^2$ and $\tau^2$. In this section we look at the effect of the initial $\lambda$ on the variance estimates.

4.2.0.1 Method of Moments

Figure 4 shows the results of the initial $\lambda$ estimate on the estimation performance of $\tau^2$ and $\sigma^2$ using the MoM method. The GCV estimate for $\lambda$ performs worst even though the GCV estimate corresponds to the most informative prior on the relative size of $\tau^2$ and $\sigma^2$, as GCV has a median $\lambda_{init} = 1029$. We hypothesise that this is because GCV shrinks the data too much, removing signal in the data, making it hard to estimate $\sigma^2$ and $\tau^2$. There is barely any difference between the performance of the other estimates. Using an uninformative prior on the relative size of $\tau^2$ and $\sigma^2$ by setting the initial $\lambda = 1$, seems to be most efficient. As both methods that set the minimum condition number such that $R = X^TX + \lambda I_{p \times p}$ is well conditioned both need to calculate the condition number for each element in a grid of $\lambda$’s which is computationally quite expensive as $R$ is a $p \times p$ matrix.
Figure 4: $\tau^2 = 0.01, \sigma^2 = 10$, $n = 100$, $p = 1000$

A closer look at the chosen $\lambda_{\text{init}}$‘s of all methods shows that the methods that choose $\lambda_{\text{init}}$ based on condition number always select the same $\lambda_{\text{init}}$. As the ‘Line Distance’ method always sets $\lambda_{\text{init}} = 4.918367$ and the loss of digits method sets $\lambda_{\text{init}} = 1.979592$ (in this specific setting). This is because the condition number plots are very similar for all the 100 iterated datasets.

Figure: 5 below shows the results when the ridge penalty parameter $\hat{\lambda}$ is now approximated from the $\sigma^2$ and $\tau^2$ estimates. Estimates that were larger than $20 \cdot \lambda$ were truncated to $2E^4$. Because the GCV method often estimated either $\sigma^2 = 0$ or $\tau^2 = 0$, it’s estimates for $\lambda$ are often either 0 or $\infty$ (now trimmed to $2E^4$). Furthermore, there is very little difference in estimation performance using the other three methods.

Figure 5: $\tau^2 = 0.01, \sigma^2 = 10$, $n = 100$, $p = 1000$
4.2.0.2 Simple Estimate  To look for the effect of the initial $\lambda$ estimate on the estimation performance by the simple estimate, the GCV estimate and setting $\lambda = 1$ are used as initial $\lambda$ in the simple estimate. The plot shows that the actual estimation does not change much by using a more 'sensible' estimate for $\lambda$. The estimation process actually gets slightly worse using GCV and this method has the added problem of estimating $\sigma^2$ equal to $\infty$, if $\lambda_{GCV} = 0$. For these reasons we choose to set the initial $\lambda = 1$ while estimating the $\sigma^2$ by the simple estimate.

Figure 6: $\tau^2 = 0.01, \sigma^2 = 10$, $n = 100$, $p = 1000$

4.3 Assessing performance of estimators

To assess the performance of the estimation methods shown above in certain situations, simulations were done. During each simulation $n, p, \tau^2$ and $\sigma^2$ will be fixed. Furthermore, the proportion of null effects $p_0$, or in other words coefficients equal to zero can be fixed, making it possible to sample coefficients from a (sparse) spike and slab distribution.

4.3.1 Basic High Dimensional Situation

Starting with a very basic high dimensional situation, the data was simulated according to:

$$y = X\beta + \epsilon$$
$$X_{n \times p} \sim N(0, 1)$$
$$\beta_{p \times 1} \sim N(0, \tau^2)$$
$$\epsilon_{n \times 1} \sim N(0, \sigma^2),$$

where $n = 100, p = 1000, \tau^2 = 0.01, \sigma^2 = 10$, this is in accordance to the model as specified in equation: (9). Figure 7 below shows the result of the first simulation.
In particular it shows that while estimating $\sigma^2$ (note: the red line highlights the true value of the variances) the simple estimate using the MSE with degrees of freedom defined in Hastie and Tibshirani [1990] completely misses the true parameter value. The PCR method also overestimates the error variance, but performs slightly better than the simple estimate. The MML, MoM and REML methods seem to perform approximately equal in estimating both $\sigma^2$ and $\tau^2$. All methods estimate the variances very well. The Simple $\sigma^2 + \text{MoM } \tau^2$ method underestimates $\tau^2$, estimating it too close to zero.

Figure 7: $\tau^2 = 0.01$, $\sigma^2 = 10$, $n = 100$, $p = 1000$

Figure: 8 compares all methods in estimating functions of $\tau^2$ and $\sigma^2$. The left plot shows the estimation of the ridge penalty parameter $\lambda$. The values that were larger than $20 \cdot \lambda$ where truncated to $20 \cdot \lambda$, to make for visually better comparison of the boxplots. The (truncated) outliers were slightly jittered to show the amount of outliers each method estimates. The method of moments occasionally estimates $\lambda$ to be infinite when the MoM negatively estimates $\tau^2$ and therefore setting $\tau^2 = 0$ causing $\lambda = \frac{\sigma^2}{\tau^2} = \infty$ (in this particular setting there were no infinite $\hat{\lambda}$). The left figure shows the MML, MoM and REML methods have most outliers. These methods still do an equally good job in estimating $\lambda$. The GCV method seems to have the highest variance in it’s estimate. The leave one out cross validation (CV) slightly underestimates $\lambda$. GCV and CV are the methods that estimate $\lambda$ directly and obviously do not perform decidedly better than the methods that estimate the variances jointly.
The right plot in Figure 8 shows the heritability estimation results. The MML, MoM and REML methods perform very equal, but all have a very broad boxplot, because $h^2$ can only ranges from 0 to 1. HiLMM has some outliers and often estimates $h^2$ outside the range of the function, this is surprising as this methods estimates $h^2$ directly.

4.3.2 Sparse model

In the model specified in equation (33) above all effects come from a gaussian distribution centered around zero. Jiang et al. [2014] calls this a *misspecified model* as in reality only a small subset of the coefficients are not zero. So $\beta$ is a set consisting out of $\beta', 0'$, where $\beta'$ are the actual $m$ non zero effects $\in \beta$ and $0'$ are the true $p - m$ zero effects $\in \beta$. Jiang et al. [2014] shows their REML method is robust under misspecification of a linear mixed model.

A more realistic model is a sparse model:

$$y = X\beta + \epsilon$$

$$X_{n \times p} \sim \mathcal{N}(0, 1)$$

$$\beta_{p \times 1} \sim p_0\delta_0 + (1 - p_0)\mathcal{N}(0, \tau^2_0)$$

$$\epsilon_{n \times 1} \sim \mathcal{N}(0, \sigma^2),$$

where, $p_0 = (p - m)/p$, i.e. the proportion of coefficients that are zero and $\delta_0$ is the dirac delta
function, i.e. a point mass at 0. Furthermore, $p_0 = 0.9, \sigma^2 = 10, n = 100$ and $p = 1000$ and $\tau^2 = 0.01$, when $\tau_0^2 = \frac{0.01}{0.1}$ as is shown in Equation (12).

Figure 9: $\tau^2 = 0.01, \sigma^2 = 10, n = 100, p = 1000, p_0 = 0.9$  

Figure 9 above shows not much changes if we sample from a sparse distribution. While estimating $\tau^2$ and $\sigma^2$ the REML, MML and MoM methods still perform very similar.

It was expected that the Method of Moments estimate would outperform the MML method as the distribution on $\beta$ is no longer gaussian, which the MML method assumes and the moments method estimate does not. This is however not the case.

In a practical sparse setting $\tau^2$ can become slightly bigger relative to $\sigma^2$, this makes it harder to estimate $\sigma^2$ as more of the variance will be put towards $\tau^2$.

Figure 10: $\tau^2 = 1, \sigma^2 = 10, n = 100, p = 1000, p_0 = 0.9$
Figure 10 shows that indeed all methods have a harder time estimating $\sigma^2$. The PCR method now also completely misses the true value for $\sigma^2$. The MML, MoM and REML estimates perform relatively well. However, all models have multiple outliers and quite a lot of variance in their estimates. $\tau^2$ seems to be harder to estimate given that all methods underestimate $\tau^2$, this could be because the methods contribute too much of the total variance to the error variance, as the error variance is mostly overestimated.

Figure 11 below looks at the estimation performance while estimating $\lambda$ using a sparse model. At the left side the situation in which $\tau^2 = 0.01$, here not much has changed. Again MoM, MML and REML deal with some outliers and so does GCV albeit lesser amounts. Besides those outliers, the MoM, MML, REML and GCV perform very similar. LooCV has the most variance in it’s estimate, which is a known problem of looCV (Hastie et al. [2001]).

When $\tau^2$ gets larger we see that all methods have a harder time estimating $\lambda$ (note that outliers were truncated, as the boxplots where otherwise poorly visible). The MoM shows most variance in it’s estimator now. Surprisingly the CV method now completely overestimates $\lambda$ and also deals with some outliers.

Figure 11: $\tau^2 = 0.1\&1, \sigma^2 = 10, n = 100, p = 1000, p_0 = 0.9$

Figure: 12 below shows results of estimating heritability. HiLMM performs worst both times and often has big outliers, and performs especially bad in the case where $\tau^2 = 1$ (see right plot in figure 12) as it completely misses the true $h^2$. The other three methods seem to perform equally well in both cases and get slightly more precise when $\tau^2 = 1$. 

35
4.3.3 Bigger N, Bigger P

In this section we will look at simulation settings where both the sample size and the amount of features increases, as we will set \( n = 1000 \) and \( p = 15000 \). This will be done in two cases, first again setting \( \sigma^2 = \tau^2 p = 150 \), where \( \tau^2 \) again is set to 0.01. In the second setting, Jiang et al. [2014] is approximately replicated. This means we will simulate from a sparse model, by setting \( p_0 = 0.9 \), \( \sigma^2 = 0.4 \) and \( \tau^2 = 0.6 \cdot p/m \), where \( m \) is the amount of non zero coefficients. Since \( p_0 = 0.9 \), \( \tau^2 = 0.6 \cdot 15000/150 = 6 \). Figure: 13 and 14 show the results from the first ”basic” case.

Figure 13: \( \tau^2 = 0.01, \sigma^2 = 150, n = 1000, p = 15000, p_0 = 0 \)

Figure 13 above shows some differences from the previous simulations. Most notably is that the PCR method performs much better and got quite close to the true error variance, also having a very narrow boxplot. The Simple Estimate still performs worst and still misses the true error variance by a big margin. The MML, MoM and REML methods are very similar in performance,
where now the median of the MoM method is very close to the true value of the variances. The Simple $\sigma^2 + \text{MoM} \, \tau^2$ method still underestimates $\tau^2$.

Figure 14 shows the results of estimating $\lambda$ and $h^2$. All methods again perform very equal, again having some outliers but mostly estimating very well $\lambda$ very well. Looking at the right side in figure 14, there are also some differences seen. This time HiLMM estimates $h^2$ almost as well as the other three methods. Showing no big outliers and is full boxplot is inside the range of the function. The other three methods again perform very similar, estimating slightly better than HiLMM.

Figure 14: $\tau^2 = 0.01, \sigma^2 = 150, n = 1000, p = 15000, p_0 = 0.0$

Situation two also shows very interesting results. Figure 15 shows that most methods estimate $\tau^2$ quite nice, especially the MML method does a great job having very small variance in it’s estimate and also estimates $\tau^2$ very precise. The Simple $\sigma^2 + \text{MoM} \, \tau^2$ method stays very close to zero in estimating $\tau^2$ This is because the simple estimate completely overestimates $\sigma^2$, as is shown in the right side of the figure. Also the PCR method completely overestimates $\sigma^2$. Furthermore, the Moments estimate and the REML estimate perform very similar, both overestimating the variance quite a bit. The MML estimate relatively does a great job estimating $\sigma^2$. 
Figure 15: \( \tau^2 = 6, \sigma^2 = 0.4, n = 1000, p = 15000, p_0 = 0.9 \)

Figure 16 shows the performance while estimating \( \lambda \) and \( h^2 \). The left side that all methods and the right side gives a clearer look of the MML boxplot. The MoM and REML method perform similar and overestimate \( \lambda \) quite a bit, which makes sense given that all estimates also overestimated \( \sigma^2 \). The leave one out cross validation method also overestimates \( \lambda \) but is not very far off the true estimate. The GCV method and MML method perform very well having a very thin boxplot.

Figure 16: \( \tau^2 = 6, \sigma^2 = 0.4, n = 1000, p = 15000, p_0 = 0.9 \)

The right side of Figure: 16 shows that while estimating \( h^2 \) the MoM and REML method underestimate the heritability a bit, where the HiLMM has most trouble estimating the heritability, again performing worst. The MML method performs very well in this situation.

The right side of Figure: 17 below gives a closer look at the estimation of \( \lambda \) by two best performing methods, GCV and MML. The plot shows that the GCV method is a lot closer than the MML method and the MML still completely misses the true estimate for the ridge parameter. The left side shows how close to the true estimate of \( h^2 \) the MML method is, but still misses the
true estimate.

Figure 17: $\tau^2 = 6, \sigma^2 = 0.4, n = 1000, p = 15000, p_0 = 0.9$

4.3.4 Low dimensional setting

To see if the proposed methods also perform well in a low dimensional setting has been simulated, where $n = 100 > p = 10$, figure 18 below shows the results. As can be seen most methods perform very similar in estimating $\sigma^2$ and $\tau^2$. As the simple estimate now performs well in estimating $\sigma^2$ the Simple $\sigma^2 + \text{MoM} \tau^2$ method now estimates $\tau^2$ very well.

Figure 18: $\tau^2 = 0.01, \sigma^2 = 1, n = 100, p = 10, p_0 = 0$

Figure 19 shows similar results. The left side shows that all methods perform equally well, all methods have some outliers but mostly doing a decent job in estimating $\lambda$.

The right side shows that the HiLMM method has a lot of negative outliers when estimating $h^2$ in low dimensions, showing that HiLMM has trouble estimating a small heritability. The other methods estimate quite well.
4.3.5 Including non penalized coefficients

In this section the results of estimating $\sigma^2$ and $\tau^2$ in High Dimensional Mixed Effects models will be shown. The simulation setting is as follows:

$$
\begin{align*}
    y &= X_{fixed} \alpha + X_{random} \beta + \epsilon \\
    X_{fixed, n \times m} &\sim \mathcal{N}(0, 1) \\
    X_{random, n \times p} &\sim \mathcal{N}(0, 1) \\
    \beta_{p \times 1} &\sim p_0 \delta_0 + (1 - p_0) \mathcal{N}(0, \tau^2_0) \\
    \alpha_{m \times 1} &\sim \frac{1}{2} \delta_0 + \frac{1}{2} \mathcal{N}(0, \tau^2_0) \\
    \epsilon_{n \times 1} &\sim \mathcal{N}(0, \sigma^2),
\end{align*}
$$

where we assumed half of the fixed effects are actually null effects, $n = 100, p = 1000 - m, p_0 = 0.9$ and $\tau^2 = 0.01$. We will set the amount of unpenalized features (m) to 5 and 10 and 80.

The REML method by Jiang et al. [2014] can also deal with a mixed effect model, hence will also be implemented. Figure: 22a shows the variance estimation results for $m = 5$ and figure: 22b shows the variance estimation results for $m = 10$.
Figure 20: $\tau^2 = 0.01, \sigma^2 = 10, n = 100, p = 1000 - m$.

The plots show that overall $\sigma^2$ and $\tau^2$ are estimated quite nicely in all cases when $m = 5$. The method in which the coefficients are also optimized shows quite thin boxplots, but has some bias in $\sigma^2$. We also see some bias in the variance estimates when using MML as is expected. However, the bias in $\sigma^2$ while using REML is actually slightly bigger. Which is counterintuitive, as REML is an adjusted MML method to fix the bias that MML has. MML does estimate $\tau^2$ quite well. Using the step function works quite well in estimating the variances. When $m$ increases not all that much changes. Optimizing over the coefficients leads to even thinner boxplots and overall the best estimation. The step function method now underestimates $\tau^2$ slightly. Comparing the $m = 5$ and $m = 10$ scenarios, we see that the MML method gets less variance in it’s $\tau^2$ estimate when $m$ gets larger. However, optimizing over $m$ unpenalized coefficients might become very hard as $m$ becomes big, in most practical situations there will only be a few unpenalized coefficients. Comparing the REML results with the results from doing regular MML and keeping all coefficients random seems to show that the effect of fixed effects is quite small. Apparantly not caring about the fixed effects at all is not much of a problem, as regular MML still performs quite decent. Likely because there are much more random parameters hence the effect of the fixed effects is very small.
Figure 21: $\tau^2 = 0.01, \sigma^2 = 10, n = 100, p = 1000 - m$. 

(a) $m = 5$  
(b) $m = 10$

Figure 21 shows that the GCV method gets more variance in its estimate when $m$ gets larger as GCV does not take any influence of the fixed effects on $y$ into account. This will cause the variability of $y$ to seem inflated causing $\lambda$ to be overestimated. This same problem seems to occur with looCV, as the loss function: $||y - X\beta||_2^2$ is no longer accurate. Optimizing the coefficients shows the best results in estimating the functions of the variances. To directly estimate the heritability in a mixed effects model a more complex method is used as is explained in Section 3.1.3 . In this method a threshold has to be set a priori, this threshold set on 0.7. Simply because practice showed that setting the threshold larger the method did not always find an estimate for $h^2$. This method produces very narrow boxplots, but biased estimates. The method could in practice perform slightly better as it is then easier to tune the threshold parameter.

To empirically investigate the asymptotic situation where $m \rightarrow n$ we also ran a simulation study where $m = 80, n = 100$ and $p = 1000$. This means the random variables are high dimensional and the fixed variables approach high dimensionality. One would expect the REML method to perform best in this situation, as this is the model where REML is designed for. Figure: 22 below shows that this is not the case:
Actually the regular MML estimates $\sigma^2$ without bias, but with the most variance. The other methods completely miss $\sigma^2$. Optimizing over the coefficients estimates $\tau^2$ very precise, but underestimates $\sigma^2$ completely. Because the STEP function underestimates both $\tau^2$ and $\sigma^2$ but still keeps the ratio correct, the estimate for $\lambda$ is quite good. Also GCV estimates $\lambda$ quite well, but has broad boxplots. When estimating $h^2$ no method performs well, with all methods missing the true estimate. The EstHer method could not converge when $m$ approached $n$ and is not plotted.

### 4.3.6 Groupwise estimation using MoM and MML methods

In Section 3.3.2.2 we showed that the $p$ variables can be partitioned into $G$ groups, leading to the following model:

$$
y = X\beta + \epsilon$$

$$
\beta_{G \in G, A_G \times 1} \sim \mathcal{N}(0, \tau^2_g)
$$

$$
\epsilon_{n \times 1} \sim \mathcal{N}(0, \sigma^2),
$$

where $n = 100, p = 1000, \sigma^2 = 10$. The data was partitioned in $G = 5$ groups, with equal group size $A_G = p/G = 200$ and $\tau^2_g = \{0.01, 0.02, 0.03, 0.04, 0.05\}$.

#### 4.3.6.1 MoM results

Figure 23 shows that groupwise estimation of $\tau^2_g$ using the method of moments estimate performs poorly. Often estimates are negative, especially for $\tau^1 = 0.01$. These estimates can be set to zero,
but as most of the estimates were negative, this will lead to many zeroes. The right plot shows that $\sigma^2$ is severely overestimated. Of course this also shows in the estimation of $\lambda$ as can be seen in figure 24:

Figure 23: Results of MoM groupwise $\tau^2$ and $\sigma^2$ estimation, $n = 100$, $p = 1000$, $\sigma^2 = 10$, $\tau^2_g = \{0.01, 0.02, 0.03, 0.04, 0.05\}$, $A_1 \ldots A_5 = 200$. 
\hat{\lambda}'s were truncated to [-300, 2000]. Many \hat{\lambda}'s were smaller than zero, these estimates have to be set to zero to make sense. This leads to the OLS estimates and hence no solution is then founded for \beta.

4.3.6.2 MML results

Figure 25 shows the results of the groupwise estimation procedure of the variances using MML estimation. The MML does seem to differentiate the different \tau^2_G's, as the general trend of the true \tau^2_G are clearly visible in the results. The \tau^2_G estimates are mostly underestimated. \sigma^2 seems to be even more underestimated. As both \tau^2_g and \sigma^2 were underestimated there seems to be a lot of variance in the data that is not being estimated by the MML method. The same results are visible when we are looking at the estimates for \lambda.
Figure 25: Results of MML groupwise $\tau^2$ and $\sigma^2$ estimation, $n = 100$, $p = 1000$, $\sigma^2 = 10$, $\tau^2_g = \{0.01, 0.02, 0.03, 0.04, 0.05\}$, $A_1 \ldots A_5 = 200$.

Figure: 26 shows the results of the groupwise estimation of $\lambda_G$. $\lambda_G$ is plotted on a log scale for better comparison of the boxplots, further more the $\log(\lambda_G)$ estimates were truncated to 20 and -10. Again the trend of the increasing $\lambda_G$ values is visible. However, estimation is not great, as often $\lambda_G$ is underestimated because $\sigma^2$ was so largely underestimated.

Figure 26: Results of MML groupwise ridge penalty parameter estimation with, $n = 100$, $p = 1000$, $\sigma^2 = 10$, $\tau^2_g = \{0.01, 0.02, 0.03, 0.04, 0.05\}$, $A_1 \ldots A_5 = 200$. 
4.3.7 Empirical robustness of the estimators

As was shown in Section: 3.3.3 the method of moments estimate does not require many parametric assumptions on $\beta$ and $\epsilon$, besides the prior distribution should have a mean and variance. To test robustness of the MoM and other methods against sampling from different distributions than standard normal we conduct some simulations whereby either $\beta$ or $\epsilon$ were sampled from a non normal distribution. This was already done when sampling $\beta$ from a sparse spike and slab distribution. To further investigate the effect of non normal coefficients and errors we look at multiple distributions.

See Figure: 27 for the density curves of the used distributions:

- $\beta \sim Laplace(\mu = 0, b = 0.0707)$: The coefficients are sampled according to a laplace distribution with mean $\mu = 0$ and variance $2b^2 = 0.01$.

- $\beta \sim Uniform(a = -0.2, b = 0.2)$: Here the coefficients are sampled according to an uniform distribution with mean $\frac{1}{2}(a + b) = 0$ and variance $\frac{1}{12}(b - a)^2 = 0.013$

- $\epsilon \sim T_4$: The errors are sampled from a T distribution with 4 degrees of freedom, with mean $= 0$ and variance $\frac{\nu}{\nu-2} \approx 0.5$, where $\nu$ are the degrees of freedom.

Figure 27: View of different probability density functions used to check for robustness against non-normality

Furthermore, we will also empirically shows how robust all methods are when the random design matrix $X$ has multicollinearity.

4.3.7.1 Sampling $\beta$  Figure: 28 shows the results when $\beta$ is sampled according to a Laplace distribution. It is expected that the MoM outperforms other methods as it does not need a
distributional assumption on $\beta$. The graph seems to show that the MoM is slightly outperformed by the MML and REML methods. Also the PCR method and simple estimate now seem to be closer to the true estimate of $\sigma^2$ than in previous simulations when $\beta \sim \mathcal{N}(0, 1)$.

Figure 28: Results simulation study when $\beta \sim \text{Laplace}(0, 0.0707), n = 100, p = 1000, \tau^2 = 0.01, \sigma^2 = 10$.

![Figure 28](image)

(a) Variances  
(b) Functions of variances

Figure: 29 below shows the results when $\beta$ is sampled according to a Uniform distribution. Not much has changed from the laplace situation, all methods perform quite well.
Figure 29: Results simulation study when $\beta \sim \text{Uniform}(-0.2,0.2), n = 100, p = 1000, \tau^2 = 0.013, \sigma^2 = 10$.

(a) Variances

(b) Functions of variances

4.3.7.2 Sampling $\epsilon$  Figure: 30 shows that all variance estimation methods are quite robust against the assumption that $\epsilon \sim \mathcal{N}(0,1)$, when indeed the error terms are simulated according to $T_4$. The results do not change in comparison to earlier simulation studies. When the goal is estimation of functions of $\sigma^2$ and $\tau^2$ the results also do not differ from earlier results.
Figure 30: Results simulation study when $\epsilon \sim T_4, n = 100, p = 1000, \tau^2 = 0.01, \sigma^2 = 2$.

(a) Variances

(b) Functions of variances

4.3.7.3 Multicollinearity in $X$ In the following section the design matrix $X$ will no longer have orthogonal columns, but will be sampled such that there is a block of correlated variables, the model is then as follows:

$$y = X\beta + \epsilon$$

$$X_{n \times p} \sim \mathcal{N}(0, \Omega)$$

$$\beta_{p \times 1} \sim \mathcal{N}(0, \tau^2 I_{p \times p})$$

$$\epsilon_{n \times 1} \sim \mathcal{N}(0, \sigma^2 I_{n \times n}),$$

where $\Omega_{k,j}$ is a block in the covariance matrix of $X$ which has some correlation $\rho$ on the off diagonal for $j, k = 1 \ldots p^*$, with $p^* \leq p$. For now we set $\rho = 0.5$ and $p^* = 10$. The MoM assumes an independent prior on $\beta$, which troubles the estimation in this situation. Figure: 31 below shows that while estimating $\tau^2$ the GRridge method now slightly overestimates $\tau^2$, as does the Method of Moments estimate. The REML and MML method estimates are again very similar, as expected and estimate $\tau^2$ best. The REML algorithm did occasionally not converge in this setting and gave errors. To remedy, 200 iterations were run and 100 random converged iterations were selected. When estimating $\sigma^2$, The MoM and Simple estimates are very similar, and perform quite well. When estimating $\tau^2$ the Simple estimate returns negative values. The PCR method is now quite close to the true estimate, but again the MML and REML estimates perform best. This also
shows in estimating functions of $\tau^2$ and $\sigma^2$. An interesting result here is how the MoM and the simple $\sigma^2 + \mathrm{GRridge} \, \tau^2$ are very similar. Joint estimation does not seem to make a difference from the two stage estimation that GRridge uses. The simple estimate for $\sigma^2$, now underestimates $\sigma^2$ and is very close to the MoM error variance estimate, causing the GRridge $\tau^2$ estimate also to be similar to the MoM prior variance estimate. These results propagate further to the function of the variances. Where both MML and REML perform quite well in estimating $\lambda$, comparable or even slightly better than the GCV and looCV method that directly estimate $\lambda$. In a mixed model the HiLMM method performs better, having the least variance in it’s estimate.

Figure 31: Results simulation study when $\bm{X}$ has multicollinearity, $\rho = 0.5, p^* = 10, n = 100, p = 1000, \tau^2 = 0.01, \sigma^2 = 2$.

![Figure 31](image)

(a) Variances  
(b) Functions of variances

Figure 32 below shows results of a simulation setting where the block size in $\Omega$ is now larger, with $p^* = 100$. The results are similar as the setting above with $p^* = 10$ albeit that the results are more extreme. The MoM and simple $\sigma^2 + \mathrm{GRridge} \, \tau^2$ methods now overestimate $\sigma^2$ more and underestimate $\tau^2$ more, which propagates into the $\lambda$ and $h^2$ estimates. Cross validation now overestimates $\lambda$ slightly, where GCV still estimates quite well and thus seems a bit more robust. While estimating $h^2$ the HiLMM method now performs best.
Figure 32: Results simulation study when $X$ has multicollinearity, $\rho = 0.5, p^* = 100, n = 100, p = 1000, \tau^2 = 0.01, \sigma^2 = 2$.

(a) Variances

(b) Functions of variances

4.3.8 Using real data

Looking at the estimation of $\tau^2$ and $\sigma^2$ in a setting where the design matrix: $X$ is real data (i.e. non simulated), gives a good impression of how the methods perform in practice. We use protein expression data from the cancer proteome atlas (TCPA), which holds 412 tumor samples on 190 protein (as measured by reverse-phase protein arrays), which are generated by the TCGA Research Network: http://cancergenome.nih.gov/. 33 shows results of estimating the variances of functions of the variances of this setting.
Figure 33: $\tau^2 = 0.01, \sigma^2 = 10, n = 408, p = 223$, using the TCGA dataset.

As can be seen the results are quite similar to the results of the case where we sampled $X$ with multicollinearity. Genomics data often has multicollinearity, and this is also visible in the TCPA dataset as is shown in the heat map of correlation of $X$ below in Figure 34. Figure 34 shows that there are some genes with a correlation: $|\rho| > 0.5$ to other genes.

Figure 34: TCPA correlation matrix heat map.
5 Conclusion and Discussion

Some interesting comparisons can be made from the results above. First of all a comparison can be made between three methods that jointly estimate $\sigma^2$ and $\tau^2$: The Method of Moments and (Marginal) Maximum Likelihood method make for a classic comparison. In a conventional setting Maximum Likelihood is often preferred if the right model is specified for the likelihood. This is often quite easy to check. In this case the right parametric model has to be picked for the prior, which is much harder to know. Hence maximum likelihood has no clear theoretical advantage here. Indeed the results do not show that the maximum likelihood estimates $\sigma^2$ and $\tau^2$ better than the moments estimate. The MoM estimate makes less stringent distributional assumption on the distribution of the coefficients, compared to the MML method. This was expected to show in the sparse setting and the other settings where the coefficients were sampled from a non normal distribution. However, both methods behave similar in this setting. In the setting where the design matrix $X$ was sampled with multicollinearity the MML method performed slightly better. Having more accurate and more precise estimates. Which is very important in practical settings, as genomics data often has multicollinearity as was shown in Section: 4.3.8. The MoM estimate becomes very similar to the GRridge estimate for $\tau^2$ and the simple estimate for $\sigma^2$, which are also method of moment estimates. This is due to the simple estimate better estimating $\sigma^2$ and this propagates into better estimation of $\tau^2$ by the GRridge method. Making the estimation method very similar to the MoM estimates. In general maximum likelihood estimates are said to have less variance than method of moment estimates, but the results in this study show both methods have about the same variance while estimating $\tau^2$ and $\sigma^2$. The MML method clearly outperformed other methods in the high dimensional case where $\tau^2$ was very big and $\sigma^2$ was very small. Again the difference with the REML method is due to the computational implementation, as the two methods are theoretically equal in this setting. When trying to add co-data and estimate a groupwise $\tau^2_g$ the MML estimates better than MoM, but both methods can likely be improved in future work and the sections on groupwise estimation in this thesis serve merely as a start to this research. The method of moments estimate does have some big strengths over the maximum marginal likelihood method. The MoM can easily be extended to survival and binary data, which is very hard for the MML as it loses conjugacy in the likelihood and prior. A second strength is computation time, the MML estimate has to invert a $n \times n$ matrix in every optimization step, where the MoM has to only invert once. This can save a lot of time when $n$ is large as can be seen in appendix E. Furthermore, there are many models such as the BSEM model that will be touched upon in the next part of the thesis or a model in which $y$ is an $n \times p$ matrix and the full $X$ matrix has to...
be regressed on all columns in $\mathbf{y}$. In this case the variance estimates have to calculated $p$ times. Which can be done quite quick using the MoM estimate. MoM could possible be modified in using the GCV equation instead of one of the linear equations defined in Equation: 28, by noting that $\lambda = \frac{\hat{\sigma}^2}{\hat{\tau}^2}$.

A second comparison is between REML and MML. In the setting where there are no fixed effects, both methods are equal. The differences in results are then byproduct of the optimization methods. Solving equation (15) is not trivial. Equation (15) is very instable and it is therefore not feasible to solve this equation with popular/simple solving algorithms such as uniroot or optim functions in R as can be seen in Appendix B. After contacting the authors of the paper we found out the authors don’t solve equation (15) directly but use a PX-EM (parameter expanded EM) or MM algorithm to solve a surrogate function (for more information see: Zhou et al. [2015]). This was not noted in their original paper (Jiang et al. [2014]). This algorithm does not perform perfect however, in the correlated setting the algorithm gave occasional errors and could not converge.

The fact that both methods are equal is visible in the results as the performance of both methods are very similar when there are no fixed effects. In the simulation setting that tries to mimic the situation described in Jiang et al. [2014] where $\tau^2 >> \sigma^2$, the MML method outperforms the REML method. One might expect the REML method to perform better than maximizing the marginal likelihood when fixed effects are added to the model, as REML is ’built’ for this purpose. REML then theoretically has unbiased variance estimates, but these estimates have a larger variance than the maximum likelihood estimate (Raudenbush and Bryk [2002]). The results when adding fixed effects or non penalized coefficients shows, that in practical situations where the number of fixed effects ($m$) is relatively small, one does not have to care too much about correcting for these effects. As the results show that REML does not perform much better than regular MML. Optimizing the fixed coefficients while optimizing $\sigma^2$ and $\tau^2$ in the marginal likelihood seems to deliver the best results. As the variances are estimated with the least bias and variance in it’s estimates. When $m$ gets large this can become infeasible, as the marginal likelihood has to be optimized over $m + 2$ parameters. In the asymptotic situation where $m \to n$, both methods did not estimate the variances well, the MML method did estimate $\sigma^2$ without bias, but overestimates $\tau^2$ quite a lot. The REML method underestimated both $\tau^2$ and $\sigma^2$. Search for other REML implementations led to R packages such as ‘mgcv’ (Wood [2006]), ‘nlme’ (Pinheiro et al. [2017]) and ‘lme4’ (Bates et al. [2015]). Mgcov could not fit the model, as it could not estimate a single $\tau^2$ over all coefficients and all three packages had trouble estimating in a high dimensional setting.
Nlme is implemented in a low dimensional setting to make a comparison to the proposed methods in a low dimensional setting. Surprisingly the nlme method does not outperform the high dimensional methods in a low dimensional setting.

When comparing methods that estimate $\sigma^2$ directly to methods that estimate $\sigma^2$ and $\tau^2$ jointly it is clear that methods that estimate both $\tau^2$ and $\sigma^2$ perform better in estimating $\sigma^2$. The PCR method seems to require a lot of data, to perform well. This shows in the results of the setting with $n = 1000, p = 15000$, in which the PCR method estimates $\sigma^2$ quite well, being very close to the true estimate of $\sigma^2$ and having a relatively thin boxplot. However, in all other settings the PCR method has a lot of difficulty estimating $\sigma^2$ often overestimating the true estimate. It seems that with a smaller $n$ and $p$, $\hat{y}_r = Z_r \hat{\alpha}_r$ is not a good approximation for $y$. Which causes the distance between $y$ and $\hat{y}_r$ to be too big, causing the PCR method to overestimate the error variance. The simple estimate has a big bias and by far performs worst in most situations. The simple estimate performs better in the multi collinearity setting, however still performs worst. The degrees of freedom for the error variance as defined in Hastie and Tibshirani [1990] as $n - \text{tr}(2H - HH^T)$, seems to be too small to adequately account for the high dimensionality in the data, resulting in completely overestimating $\sigma^2$ and a very bad estimate. Using the vast amount of horizontal information that high dimensional data by definition has, improves the estimation of the variance of the vertical direction as these simulation studies show. It is unfortunate that the simple estimate is most often used in practical settings, as it is the worst estimate for the error variance that was implemented in this thesis.

The bad estimation of $\sigma^2$ by the simple estimate propagates into bad estimation of $\tau^2$ by the Simple $\sigma^2 + \text{MoM}\tau^2$ method estimation. The Simple $\sigma^2 + \text{MoM}\tau^2$ method uses a Method of Moment approach to estimate $\tau^2$ using as initial estimate for $\sigma^2$, the simple estimate. The results show that this method often underestimates $\tau^2$, probably caused by the overestimation of $\sigma^2$ by the simple estimate as. Using a better estimate for $\sigma^2$ boosted the Simple $\sigma^2 + \text{MoM}\tau^2$ performance a lot as can be seen in appendix D.

While estimating the heritability and ridge penalty functions of $\tau^2$ and $\sigma^2$ some interesting results were seen. GCV does a good job estimating $\lambda$, however GCV does not do a decidedly better job than the joint estimation methods, in fact the REML method seems to have slightly narrower boxplots than the GCV estimate and the MML method performed much better when $\tau^2 >> \sigma^2$. The GCV method did showcase less outliers while estimating $\lambda$. The GCV method is intuitively
quite similar to the MoM estimate. Both methods start from a squared loss function (GCV uses MSE and MoM uses RSS). In which the GCV method optimizes $\lambda$ from a weighted version of the predicted residual error sum of squares (PRESS) estimate, which intuitively seems close optimizing over prior parameters of the expected (squared) prediction error as is done by the MoM. Leave One Out Cross Validation (LooCV) performed quite similar to GCV, but did decidedly worse in the case where $\tau^2$ was larger, such as the sparse case where $\tau^2 = 1, \sigma^2 = 10, n = 100, p = 1000$ and the setting where $\beta$ was sampled from a Gamma distribution with $\tau^2 = 0.25$. In these settings LooCV overestimated $\lambda$. Furthermore, the computation time of this method is also by far the slowest. This matters especially when $p$ and $n$ get big. In these simulations there might be an unfair advantage towards the joint estimation methods, as the goal of these methods is estimating the true $\lambda$ estimate, whereas looCV has as goal minimizing prediction error in a future sample and not estimating $\lambda$ as close to the true estimate as possible. This is also the goal of generalized cross validation which effectively is an approximation to the looCV. Estimating $\lambda$ directly does not seem to perform better then joint estimation methods, whereby the joint estimation methods have the added information about the size of $\sigma^2$ and $\tau^2$. Results from the $h^2$ estimation showed that the HiLMM method which directly estimates $h^2$ generally does a worse job in estimating the heritability than the methods that jointly estimate $\sigma^2$ and $\tau^2$. HiLMM often estimates outliers, that can even be outside the range of the $h^2$ function. It seems the maximum likelihood procedure used by the HiLMM method needs a lot of data to reach accurate estimates as could be seen in the basic case where $n = 1000$ and $p = 15000$, in which case this method performed much better but still not as good as the joint estimation methods. This method seems a valid method in studies with big sample size and many features measured. However, in the harder case where $\sigma^2$ was very small and $\tau^2$ was very big, the method had a lot of difficulty estimating $h^2$. Bonnet et al. [2015a] provide an additional method to estimate $h^2$ that deals with a mixed effects model having both fixed and random effects. This method starts with a variable selection procedure, which tries to remove the null effects, before maximizing the likelihood to improve the estimation of the variances Bonnet et al. [2015b]. This method is implemented in the ‘EstHer’ package and was implemented in the mixed models case and did improve estimation. A hyper parameter has to be set, finetuning this hyper parameter could improve the $h^2$ estimation. Unfortunately this is not feasible in a simulation setting.

Overall there are interesting results from this comparative simulation study. First the simulations showed that using information in the horizontal direction improved estimation of $\sigma^2$ in a high
dimensional setting. Even in a low dimensional setting where there is only little information in the horizontal direction the joint estimation methods did not perform worse. The joint estimation methods performed equally well or better than methods that estimate the ridge penalty function directly and outperformed the HiLMM method that estimates $h^2$ directly. Furthermore, it is very surprising that the basic methods; Method of Moments and maximizing the marginal likelihood performs comparable to methods of much more complexity (REML, PCR Method, HiLMM).
A Proof for $P_\gamma$

Let $X_{n \times 1} = \begin{bmatrix} c \\ \vdots \\ c \end{bmatrix}$, be a fixed data matrix. Jiang et al. [2014] gives

$$P = V^{-1} - V^{-1} X (X^T V^{-1} X)^{-1} X^T V^{-1} = V^{-1} - V^{-1} X B X^T V^{-1} = V^{-1} - R$$

Then $X^T V^{-1} X = \begin{bmatrix} c & \ldots & c \end{bmatrix} V^{-1} \begin{bmatrix} c \\ \vdots \\ c \end{bmatrix} = c^2 \cdot V^{-1}_{\Sigma}$, where $V^{-1}_{\Sigma} = \sum_{i,j} V^{-1}_{i,j}$.

Hence, $B = \frac{1}{c^2} V^{-1}_{\Sigma}$.

Then:

$$X B X^T = B X X^T = \frac{1}{c^2} V^{-1} \Sigma C_{n \times n}^2 = (\frac{1}{V^{-1}_{\Sigma}})_{n \times n} = \hat{V}$$

where $C_{n \times n}^2$ is a $n \times n$ matrix with all elements equal to $c^2$.

So, $R = V^{-1} \hat{V} V^{-1}$, such that $P = V^{-1} (I_{n \times n} - \hat{V} V^{-1})$

B Finding the root to the REML equation, (15)

Finding the root to (15) could be problematic. The equation is instable as practice showed that different root finding algorithms find different roots.

An alternative method is subtracting the left and right part of equation (15) and (numerically) minimizing this function. A sensitivity analysis is done using both methods on the effects of the estimates. Here both the ‘optim’ function in R and the ‘uniroot’ function in R were applied on the same simulated dataset. Because ‘uniroot’ often gives an error, as it could not find the root, the iteration that produced an error was skipped and the algorithm continued until 100 complete iterations were done. This could give some bias in the estimates, but should not give major problems as this is an exploratory analysis only. Furthermore, the following parameters were fixed: $n = 100$, $p = 1000$, $\tau^2 = 0.01$, $\sigma^2 = 10$, and $\gamma = 0.001$

Figure (35) below shows the results:

There seems to be quite a big difference in estimates between both methods. Uniroot does a very good job in estimating $\sigma^2$, where optim does a lesser job. Also the estimates for $\gamma$ and $\tau^2$ seem to
be quite different. The minimizing method is always further away from the true parameter value (the red line). However, uniroot still does not perform to well in estimating $\tau^2$ and $\gamma$.

Figure 35: REML sensitivity

Because there is a distinct difference between the estimates, a possible solution could be first trying to solve using uniroot and when this fails using optim instead. Practice showed that this method also gives bad estimates of $\gamma$, which causes the estimation of especially $\tau^2$ to be very bad. The $\sigma^2$ estimate showed to be quite robust to the bad estimation of $\gamma$.

Correspondence with the authors lead to more information on the subject, as the authors agreed that equation (15) is instable and therefore they did not solve the equation directly but used PX-EM (Parameter expanded EM) or a MM algorithm. The authors generously shared their R code with us.
C Approximation to condition number of $R$

Again, let $R = X^T X + \lambda I_{p \times p}$, then the condition number of $R$ is defined as: $C(R) = \frac{\max(\Lambda)}{\min(\Lambda)}$, where $\Lambda$ are the eigenvalues. We now show $\min(\Lambda) = \lambda$, by doing using an eigenvalue decomposition:

\[
R = X^T X + \lambda I_{p \times p} = U^{-1} D U U^{-1} + D U U = U^{-1} U (\Lambda^2 + \lambda) U^{-1} U = D^2 + \lambda.
\]

Because $X$ is a singular matrix as it has more columns than rows, the smallest eigenvalue = 0. So $\min(\Lambda) = \lambda$

It is clear that doing a full eigen decomposition is overkill as we only need the biggest eigenvalue. Finding $\max(\Lambda)$ can easily be done using the power iteration algorithm (see for instance: Golub and Van Loan [2012])

D Improving GRridge performance

To look at the effect of using an improved $\sigma^2$ estimate while estimating $\tau^2$ using the GRridge approached a simple simulation was done using the model specified in equation (33) in section 4.3.1. Figure 36 below shows the effect the estimation procedure of $\sigma^2$ has on $\hat{\sigma}^2_{GRridge}$.

Comparing the GRridge procedure while setting $\sigma^2$ equal to the simple estimate, the MML estimate of $\sigma^2$ and the true value of $\sigma^2$. It is clear that when using the simple estimate GRridge performs worse, underestimating the true value. Which was also the case in all simulations situations above.

When using the MML estimate for $\sigma^2$ the performance increases tremendously, being very similar to using the true value of $\sigma^2$. However, the added uncertainty that comes from estimating $\sigma^2$ shows in the estimate. Having a much wider boxplot compared to using the true value for $\sigma^2$. 
Figure 36: $\tau^2 = 0.01, \sigma^2 = 10, n = 100, p = 1000$.

E Runtime of MML, MoM and REML

Table 1 below gives an overview of the computation time of estimating $\sigma^2$ and $\tau^2$ for specified data set sample sizes.

Table 1: Computation time joint estimation methods

<table>
<thead>
<tr>
<th>Method</th>
<th>n</th>
<th>p</th>
<th>Runtime in seconds</th>
</tr>
</thead>
<tbody>
<tr>
<td>REML</td>
<td>10000</td>
<td>15000</td>
<td>3588.451</td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>15000</td>
<td>60.929</td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>5000</td>
<td>25.887</td>
</tr>
<tr>
<td>MML</td>
<td>10000</td>
<td>15000</td>
<td>116651.361</td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>15000</td>
<td>3985.787</td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>5000</td>
<td>886.644</td>
</tr>
<tr>
<td>MoM</td>
<td>10000</td>
<td>15000</td>
<td>74303.098</td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>15000</td>
<td>750.053</td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>5000</td>
<td>272.952</td>
</tr>
</tbody>
</table>
Part III

Using a Bayesian Structural Equation Model to find treatment targets in High Dimensional expression data.
6 Introduction

It is of interest to researchers to know precisely which genes or proteins are directly affected by a drug or other treatment. Knowing which genes are directly affected by a drug can yield a better understanding of the working and side effects of specific drugs. It could also enable scientists to fine tune medicines to specific genes. Omics data is often very noisy and many genes are related to each other. Hence when the expression of one gene is perturbated, then often this effect is indirectly propagated to other genes. The statistical description of this interplay between genes is naturally carried out with Gaussian Graphical Models, where nodes represent genes and edges represent the interactions between these genes. The Bayesian Structural Equation Model such as defined in Leday et al. [2017] estimates this graph. Adding a treatment covariate effectively corresponds to regressing a treatment on each gene in a dataset while correcting for this interplay between genes and hopefully finding the direct effect of a treatment on each gene. In this section of the manuscript we will explore this BSEM + covariate model and compare it with existing methods using simulation studies.

6.1 Genomics Data

To understand the data we are dealing with in this thesis, a short biological introduction is needed: Within molecular biology, genomics is the study of the genome, the complete set of genetic material or Deoxyribonucleic acid (DNA) within an organism. A DNA molecule is a double helix shaped molecule, made out of two twisting, paired strands. Each strand is made out of four nucleotide bases: adenine (A), thymine (T), guanine (G) and cytosine (C). Bases of opposite strands of the DNA pair are complementary such that an A always pairs with a T and a C always pairs with a G. Thus if the sequence of nucleotide bases on one strand is 'ATGAGA', the opposite strand will be 'TACTCT'. The order of As, Ts, Cs and Gs determines the information encoded in that part of the DNA molecule. Almost every cell in the body contains a complete copy of all DNA base pairs that make up the human genome. A gene is defined as the unit of DNA that carries the instructions for making a specific set of proteins, each gene codes on average three proteins. These genes are located on chromosomes, that are within the nucleus of a human cell in which the genes direct production of proteins with help from enzymes and messenger ribonucleic acid (mRNA). Figure 37 gives a visual overview.
The central dogma of biology (Crick [1958]) explains how information that is encoded within DNA is transferred to proteins in the cell. These proteins are used to make up tissue and organs, but also control chemical reactions and carry signals between cells. If a cell’s DNA is mutated, an abnormal protein may be produced which disrupts the body’s usual process which can lead to diseases such as cancer. The central dogma of biology explains that enzymes transcribe the DNA’s information into a molecule called messenger RNA (mRNA), this mRNA travels outside the cell’s nucleus into the cell’s cytoplasm. The mRNA is read and the information is translated by which amino acids are linked together in the right order to form the right protein.

New high-throughput technologies make it possible to determine the exact order of the bases in a strand of DNA within a gene. An example of such a high-throughput technology is a microarray. A microarray is a measurement device that can quantify tens of thousands biological sequences (such as a DNA or mRNA sequence). Researchers can use this data to search for genetic variations or mutations that may play a role in the development of diseases. This data does bring many difficulties. The expression of many genes can be measured in parallel, therefore often many more genes are measured than there are individuals or samples in the dataset. Afterall it is very expensive and hard to come by enough cancer patients to measure the gene expression of tens of thousands cancer patients. In more technical terms this means the data matrix \( X \), with on its columns the \( p \) genes/proteins and on its rows the \( n \) samples is short and broad. In other words the matrix has more columns than rows, which means the the product: \( X^T X \) is singular and no longer invertible. Therefore classical techniques such as linear regression do no longer have a unique
solution. Another problem is the dependencies between the measured genes or proteins. Figure 38 shows how this interrelationship between genes/proteins can make analysis harder. Giving a patient a treatment, may directly affect the expression of gene A and gene B. These genes can propagate this effect to other genes, such as gene C, gene D and gene E. Which also causes these genes to express themselves. Finding the direct effects of the treatment on the gene level can be hard. In technical terms the variables have many (small) correlated effects and statistical analysis may suffer from this collinearity between variables.

Figure 38: Possible interaction graph between genes

6.2 Goal

The goal in this part of the thesis is to find the set of target (genetical) variables that were directly affected by a treatment within a big set of (genetical) variables, which could constitute to finding gene A and gene B in the set of Genes A to F in Figure 38. This is done by adding a treatment covariate to the Bayesian Structural Equation Model (BSEM) (Leday et al. [2017]). This framework models the effect of a drug on a gene’s expression and accounts for indirect effects by including and estimating a gene interaction network (GIN) in the same model framework. The BSEM model is implemented and compared with other models explored in Section 7 below. The comparison is performed using a simulation study explained in Section 8.1.

7 Models

In this section the models to analyse omics data will be explained, which will be used to compare the performance of the BSEM + covariate model with.
7.1 BSEM + covariate

The Bayesian Simultaneous-Equation Model (BSEM) expresses each genetic variable as a function of other genetic variables. Consider high dimensional data on \( p \) features from \( n \) samples. Hence \( y_j \) is a \( n \times 1 \) outcome vector for genetic variable \( j \in \{ J = 1, \ldots, p \} \). An additional treatment indicator vector \( T = (T_1, \ldots, T_n) \), where \( T_i = 1 \) when sample \( i \) has been treated, and zero, otherwise, is added to the model in order to find variables that were directly targeted by a treatment.

The BSEM model with covariate \( T \) is then defined as (34).

\[
y_j = \sum_{k \in J \setminus j} y_k \beta_{jk} + \gamma_j T + \epsilon_j, \quad j = 1, \ldots, p
\]  

(34)

The priors on the coefficients in (34) are as follows:

\[
\begin{align*}
\epsilon_j &\sim \mathcal{N}_n(0, \sigma_j^2 I_n), \\
\beta_{jk} &\sim \mathcal{N}(0, \sigma_j^2 \omega_j), \\
\tau_j^{-2} &\sim \mathcal{G}(a, b), \\
\sigma_j^{-2} &\sim \mathcal{G}(c, d), \\
\gamma_j &\sim \mathcal{N}(0, \sigma_j^2 \tau_j^2), \\
\tau_j^{-2} &\sim \mathcal{G}(a, b),
\end{align*}
\]  

(35)

Generally the hyperparameters \( a, b \) will be initialized at 0.001 in our simulations and then updated in an empirical bayes setting as is explained in Leday et al. [2017]. \( c, d \) will be set to 0.001 but not be updated.

Because all genetic variables are added in the model, BSEM models corrects for all correlations between these genetic variables. Furthermore, the BSEM approximates a Gaussian Graphical Model (GGM) as it can be shown that \( \beta_{jk} = -\omega_j^{-1} \omega_{jk} \), if \( \epsilon_j \sim \mathcal{N}(0, \sigma_j^2) \) is independent of \( (Y_k : k \neq j) \). Where \( Y \sim \mathcal{N}_p(0, \Omega^{-1}) \) (Leday et al. [2017]).

This model shrinks the coefficients in two different ways: First it shrinks \( \beta_{jk} \) in a local sense, due to the Gaussian prior on \( \beta_{jk} \) centered around 0. A small value of the prior variance \( \tau_j^2 \) forces the posterior distributions of the genetic variables to be shrunken towards zero. This ridge type shrinkage is a good first step in variable selection (Bondell and Reich [2012]), which is very important in a high dimensional setting. Secondly the coefficients are shrunken in a global manner by the empirical Bayes estimation of the hyperparameters \( a \) and \( b \). This causes the coefficients to be shrunken by borrowing information across all genetic variables. When the hyper prior is informative (i.e. small or moderate value for its variance \( a/b^2 \)), the hyper prior shrinks the
posterior distributions of $\tau_{jk}^2$ towards a common distribution, centered around the prior mean $a/b$. This type of shrinkage stabilizes estimation of the coefficients (Leday et al. [2017]).

The model is fitted using a Variational approximation to the posteriors which is computationally very fast. The full implementation of the Variational Bayes algorithm can be found in Leday et al. [2017] and its supplementary material.

7.2 Alternative models

To compare performance of the BSEM + covariate method various simpler models have also been implemented on the data. A BSEM implementation of a standard procedure named “Differential Expression Analysis” has been implemented. This model is similar to the BSEM model without the proteins in the model, thus just the treatment covariate. A single shrinkage empirical bayes method has been implemented and logistic regression. Logistic regression is also an often used method in high dimensional data analysis.

7.2.1 Single Shrinkage INLA

The Single Shrinkage INLA model is very similar to the BSEM model defined in (34). This model shrinks locally by the ridge type prior on the coefficients, but does not shrink globally as the empirical bayes hyper prior on $\tau_{jk}^2$ is lost. The prior variance of $\beta_{jk}$ is now directly estimated using the Empirical Bayes procedures explained in the Part II of this thesis. More precisely the MML method is used as multicollinearity is often present in High Dimensional Data. The prior variance of $\gamma$ is set to 0.001. The Single Shrinkage INLA model is also not fitted using variational bayes, but using Integrated Nested Laplace Approximation (INLA) (Rue et al. [2009]). The model then uses the following priors:

$$
\epsilon_j \sim N(0, \sigma_{\epsilon}^2 I_n),
$$

$$
\beta_{jk} \sim N(0, \tau_{jk}^2),
$$

$$
\sigma_{\epsilon}^{-2} \sim G(1, 0.001),
$$

$$
\gamma_j \sim N(0, 0.001).
$$

7.2.2 Differential Expression

Ignoring the effect of the other genetical variables leads to a differential expression model:

$$
y_j = \gamma_j T + \epsilon_j, \quad j = 1, \ldots, p
$$

(37)
This model is fit using the BSEM variational bayes algorithm. The priors on the coefficients in (37) are as follows:

\[ \epsilon_j \sim N_n(0, \sigma^2_j I_n), \]
\[ \gamma_j \sim N(0, \sigma^2_j \tau^2_j), \]
\[ \tau_{-j}^{-2} \sim G(a, b), \]
\[ \sigma_{-j}^{-2} \sim G(c, d). \] (38)

It is important to note that the differential expression model is theoretically uncomparable to the other models. The differential expression model models marginal effects of the treatment, while the other implemented methods model the conditional effects of the treatment. Interest in the comparison with the differential expression model is because this model is an often used method in practice.

7.2.3 Logistic Regression

In this method the role of \( y \) and \( T \) are reversed, where now the proteins are regressed on the treatment. This leads to one single logistic regression model, which is advantageous as only one model has to be fitted instead of \( p \) models. However, the correlations between variables are not modelled. The logistic ridge estimator of \( \theta \) does take the collinearity of the proteins in account through the \( X^T X \) matrix in the likelihood function.

The full model is as follows:

\[ T_i \sim \text{Bern}(p_i), \]
\[ \logit(p_i) = \theta_0 + \sum_{j=1}^p \theta_j y_{ij}, \] (39)
\[ \theta_j \sim N(0, \tau^2_j). \]

Again a ridge like prior is put on the coefficients, to shrink the coefficients in a local manner. There is no global shrinkage in this model, just like the BSEM empirical bayes model. This model seems to answer a slightly different question, namely: What features have an effect on the treatment, instead of on which features does the treatment have an effect? Another difference with the other models is the inclusion of an intercept in the model. Because all \( p \) proteins are normalized the other models do not need to include an intercept in the model. However, as in the Logistic Regression case, all proteins are regressed on the treatment vector thus an intercept will be included.
8 Simulations

In this section the (data-based) simulation methods will be explained and the simulation results will be presented.

8.1 Methods

The simulations were based on real data. We used protein expression data on Ovarian serous cystadenocarcinoma (OV) from the Cancer Preteome Atlas Portal (TCPA). The data consists of 408 tumor samples, \( n = i \ldots 408 \), and 223 proteins, \( p = j \ldots 223 \), measured by reverse-phase protein arrays. These datasets were used to compare the performance of the BSEM model in finding treatments target with the simplified models, defined above. The simulations were done in a data-based way: First the dataset was standardized, then \( n/2 \) randomly sampled rows were taken from the expression data. This establishes the control group. The treatment group was sampled as described below, where \( m = 10 \) proteins were pertubated with the small effect vector \( \gamma = \{-0.2, -0.156, -0.111, -0.067, -0.022, 0.0667, 0.111, 0.156, 0.2\} \). Over multiple simulation runs \( \gamma \) was enlarged by a constant in \( C = \{0.2, 1.4, 2.6, 3.8, 5.0, 6.5, 8.0, 10.0\} \) to study the consequences of magnifying the effect. For each value in \( C \) a 100 iterations were run. See Appendix: F for a full table of all added effects to the proteins over the iterations. The effect of these \( m \) proteins were further propagated to the other \( p - m \) proteins in the treatment group. This was done using the Conditional Multivariate Normal Distribution.

- **Conditional Multivariate Normal:**

  The penalized covariance matrix \( \Sigma \) is estimated from the data using the ‘rags2ridges’ R package (Peeters et al. [2017]). Rags2ridges estimates the ridge penalized precision matrix (\( \Omega(\lambda) \)) by amending the log likelihood of \( \Omega \) by the ridge penalty \( \frac{1}{2} \text{tr}[(\Omega - T)^T(\Omega - T)] \), where \( T \) is a symmetric positive definite target matrix and \( \lambda \in (0, \infty) \), then:

  \[
  \hat{\Omega}(\lambda) = \left[ (\lambda I_{p \times p} + \frac{1}{4}(S - \lambda T)^2)^{1/2} + \frac{1}{2}(S - \lambda T) \right]^{-1}
  \]

  (Peeters et al. [2017]), where \( \lambda \) is chosen using cross validation.

  The treatment group is sampled by first pulling \( m \) proteins from \( \mathcal{N}(0, \hat{\Omega}(\lambda)^{-1}) \). These \( m \) proteins were ‘affected’ by adding a small constant to the matching \( m \) columns in \( X \). The complementary \( p - m \) proteins were sampled from a multivariate normal distribution, conditional on the affected proteins. This is done using Theorem 6.5 of Bickel and Doksum.
\[ [2001]: \]
\[
X | Z \sim \mathcal{N}(\mu_X + \Sigma_{XZ} \Sigma_{ZZ}^{-1} (Z - \mu_Z), \Sigma_{XX} - \Sigma_{XZ} \Sigma_{ZZ}^{-1} \Sigma_{ZX})
\]

This is done for \( n/2 \) cases. Binding the \( n/2 \) cases with the \( n/2 \) controls makes the final dataset.

The simulations were performed 100 times \( \forall c \in C \). On each dataset, the models discussed in Section 7 are implemented. Because each model is a Bayesian model, each model yields a posterior distribution for all covariates in the model. Therefore the posterior mean and posterior standard deviation of the treatment variable in each regression model per protein are available. The absolute values of these coefficients are then used. It is to be expected that the \( m \) proteins have the largest absolute standardized posterior means, \( |\hat{\beta}_{jk}| \), which will be used as a selection method. The coefficients of the logistic model should be standardised because all covariates are on the right side of the model which are not on similar scale the treatment vector. This is not the case in the other models as the error variance: \( \sigma^2 \) is included in the prior of \( \beta_j \). Furthermore, Variational Bayes algorithms are known to underestimate the standard deviation of the posterior distribution (Opper et al. [2015]). Therefore only the posterior mean treatment coefficients of Logistic Regression will be divided by the posterior treatment standard deviations (note: we have considered both standardised and non-standardised results and found that Logistic regression actually benefits from standardization, whereas the other methods do not. This is in line with our understanding of the models). ROC curves were created for each simulation, and the median (over the 100 iterations) ROC curves are reported in the results section of this thesis. Furthermore, the mean Area Under the Curve (AUC) values over the iterations are also reported.

### 8.2 Results

#### 8.2.1 Comparison of methods

The grid of plots in figure: 39 shows the median ROC curve (over the 100 iterations) for each multiplier constant, table: 2 shows the mean AUC for all \( i \in C \) of all implemented methods. Both the plot and table show that the BSEM performs very constant over the effects sizes. This causes the method to perform, relative to the other methods, best with small effects, however worst when the effects are big. The single shrinkage method is also quite constant, however does perform better when the effects get larger. The differential expression and logistic regression models benefit a lot from the larger effect sizes. We can see that the differential expression ROC curve starts quite poorly, performing just as well as randomly picking proteins (with \( AUC \approx 0.5 \),...
however progressively gets better. The logistic regression seems to perform best with the highest multiplier constant. These effects are not very practical in an omics setting, as effects on expression data are very rarely this large.
Figure 39: Results of all methods using the conditional multivariate normal distribution to propagate the effect of the $m$ proteins to the other $p_m$ proteins.
Table 2: Area under curve values (AUC) for all methods with all multiplication values using the multivariate normal simulation scheme

<table>
<thead>
<tr>
<th>Multiplier</th>
<th>BSEM</th>
<th>Single Shrinkage INLA</th>
<th>Differential Expression</th>
<th>Logistic Regression</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2</td>
<td>0.88</td>
<td>0.85</td>
<td>0.55</td>
<td>0.78</td>
</tr>
<tr>
<td>1.4</td>
<td>0.87</td>
<td>0.84</td>
<td>0.57</td>
<td>0.74</td>
</tr>
<tr>
<td>2.6</td>
<td>0.87</td>
<td>0.81</td>
<td>0.67</td>
<td>0.71</td>
</tr>
<tr>
<td>3.8</td>
<td>0.86</td>
<td>0.82</td>
<td>0.73</td>
<td>0.74</td>
</tr>
<tr>
<td>5.0</td>
<td>0.86</td>
<td>0.85</td>
<td>0.78</td>
<td>0.77</td>
</tr>
<tr>
<td>6.5</td>
<td>0.86</td>
<td>0.87</td>
<td>0.82</td>
<td>0.81</td>
</tr>
<tr>
<td>8.0</td>
<td>0.86</td>
<td>0.89</td>
<td>0.84</td>
<td>0.85</td>
</tr>
<tr>
<td>10.0</td>
<td>0.86</td>
<td>0.90</td>
<td>0.86</td>
<td>0.92</td>
</tr>
</tbody>
</table>
Figure 40: Barplot of the number of times protein $j$ has been selected as to be in top 10 largest absolute coefficients over 100 iterations, for different methods and different multiplier values. The true positives are the first $m = 10$ proteins. The logistic regression model also fits an intercept. This is shown as the highest line in the Logistic Regression plot (Logistic Regression has thus 1 extra line).

Figure: 40 above shows results regarding the selection of the proteins. On the Y-axis the P different proteins are shown and on the X-axis the number of times the corresponding protein was selected to be in the top 10 biggest absolute proteins during the 100 iterations. The story that unfolded in the ROC plots in Figure: 39 can also be seen in these plots. The BSEM figure shows that the true affected proteins were not selected as often as was the case in the logistic regression model. However, there were much fewer different false positive proteins selected in this BSEM
model than in the Logistic Model. This means that a small group of non affected proteins were often falsely selected, where in the Logistic Model a large group of non affected proteins were less often falsely selected. There are also some visible peaks in the BSEM model that correspond with the same peaks in the Differential Expression models. There were some non affected proteins that were often selected in both the BSEM and Differential Expression model, but not in the other models. There seem to be two closely related but slightly different reasons for this. Both reasons correspond to the $\beta_{jk}$ in each of the $p$ regression models. In some models the sum of all $\beta_{jk}$ is very small which causes the treatment covariate to become very large. Whereas in other models the sum of full $\beta_{jk}$ is very big. This shows there is a competition between the regularised proteins and non regularised proteins, as the regularised proteins tend to zero, the effect is attributed too much to the unregularised proteins. This can be seen in Figure: 41. In this histogram the regular BSEM model without covariate was applied in a simulation setting equal to the setting shown in Section: 8.1, such that we have a clear view of the interaction between proteins in the model without interference of the treatment covariate. Figure: 41 shows that ‘protein 105’, which has the largest peak in Figure: 40, has the smallest sum of the $|\beta_{jk}|$ of all $p$ regression models, whereas ‘protein 98’ (which also has a large peak in 40) has one of the largests sums of $\beta_{jk}$.

Figure 41: Distribution of mean (over 100 iterations) $\beta_{jk}$ of the Bayesian Structural Equation Model without treatment covariate for all $j$ regression models for the smallest multiplier.

Figure: 42 shows a histogram of the rank of the absolute sum over all $\beta_{jk}$ in each iteration for specific models/proteins. Here ‘protein 64’ is randomly chosen, while not being an affected protein and not having a peak in Figure: 40. The sum over $\beta_{105,k}$, modelling ‘protein 105’ is relatively very small. ‘Protein 216’ often has a relative small sum, however the absolute treatment coefficient
Table 3: Correlations between the $m$ true positive (TP)/affected proteins and protein 98 and 105 which are often selected by the model.

<table>
<thead>
<tr>
<th>(TP) Proteins</th>
<th>Protein 98</th>
<th>Protein 105</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-0.14</td>
<td>0.11</td>
</tr>
<tr>
<td>2</td>
<td>-0.07</td>
<td>0.04</td>
</tr>
<tr>
<td>3</td>
<td>-0.12</td>
<td>-0.03</td>
</tr>
<tr>
<td>4</td>
<td>0.17</td>
<td>0.19</td>
</tr>
<tr>
<td>5</td>
<td>0.02</td>
<td>0.09</td>
</tr>
<tr>
<td>6</td>
<td>-0.09</td>
<td>0.13</td>
</tr>
<tr>
<td>7</td>
<td>0.01</td>
<td>0.07</td>
</tr>
<tr>
<td>8</td>
<td>-0.09</td>
<td>-0.17</td>
</tr>
<tr>
<td>9</td>
<td>0.01</td>
<td>-0.18</td>
</tr>
<tr>
<td>10</td>
<td>0.14</td>
<td>0.13</td>
</tr>
</tbody>
</table>

is large. This could be caused by regularization-induced confounding (Hahn et al. [2018]). This refers to the tendency of shrinkage priors to bias treatment effects by over-shrinking control effects. ‘Protein 98’ often has a large relative sum over $\beta_{98,k}$ also ‘protein 218’ often has a slightly large sum. The treatment covariates in the models of protein ‘98’ and protein ‘105’ are often (wrongfully) selected during iterations where the sum of absolute $\beta_{jk}$ of the remaining proteins in the models are either small or large. Furthermore the treatment coefficient in the model of ‘protein 98’ are positively very large and the treatment coefficient in the model of ‘protein 105’ are negatively very large. This could also be due to a spill-over effect if these proteins were highly correlated with the $m$ affected proteins. However Table 3 below shows that both proteins do not have high correlations to any of the $m$ affected proteins. However when the multiplier becomes very large, the small correlations could still cause a spill over effect as the regular effects of the proteins are small.
Figure 42: Histograms of the rank of the absolute sum of $\beta_{jk}$ of the Bayesian Structural Equation Model without treatment covariate for a few selected proteins.

The results above also show that the differential expression model performs much better with the higher effect multiplier, i.e. if the treatment has a much better effect. Figure: 43 shows a histogram of the posterior precision of $\beta$. A large precision means that the treatment regression coefficient in the $j^{th}$ regression model comes from a very narrow distribution around mean 0 and hence is often estimated close to 0. The histograms show that when the multiplier is large the precision of the affected protein gets smaller, which causes the posterior mean of the treatment coefficient to be large. Furthermore, this model shows that the precision of the treatment covariate of ‘protein 105’ is relatively small and the precision of the treatment covariate of ‘protein 98’ is relatively large.
8.2.2 Bayes Factors

To investigate whether the treatment covariate adds much to the fit of each model we consider the Bayes Factors for the models including and excluding the treatment covariate model for all $p$ regression models over 100 simulation iterations.
Figure 44: Bayes Factors of the mean Marginal Likelihood over 100 iterations, per protein using the conditional multivariate normal distribution to propagate the effect of the $m$ proteins to the other $p-m$ proteins. The red lines show the Bayes Rates of the affected proteins, the numbers over these lines show the added constant divided by the multiplier.

Figure 44 shows that most proteins benefit from the added treatment covariate as the Bayes Factors are centered above 1. Looking at the affected proteins, it is evident that the biggest effects benefit most.

8.2.3 No Global Shrinkage

To further validate the full BSEM with treatment we ran the model with and without empirical bayes estimator of the hyper parameters of the $\tau_j^{-2}$ and $\tau_{j,*}^{-2}$. Which means the parameters of these Gamma distributions were vague, set to 0.001. Figure: 45 and Table: 4 below show again that the full BSEM model performs the best when the effects on the affected proteins are very small. This shows in the AUC values when the multiplier is equal to 0.2, these are largest for the full BSEM model. However having a vague prior on the proteins precision, i.e. $\tau_j^{-2}$ did seem to improve the estimation a little bit when the effects got slightly larger. This vague precision prior can cause the precision of the coefficients to be large and hence shrink the protein effects more than the empirical bayes would. The performance of this method however worsens when the effects get larger. Again this could be due to regularization-induced confounding (Hahn et al. [2018]) as the control variables will be shrunken too much causing the treatment effect to be large.
Figure 45: Results of the BSEM methods using full shrinkage, no shrinkage on the treatment covariate, no shrinkage on the $p - 1$ proteins.

(a) Multiplier = 0.2  
(b) Multiplier = 1.4  
(c) Multiplier = 2.6  
(d) Multiplier = 3.8  
(e) Multiplier = 5  
(f) Multiplier = 6.5  
(g) Multiplier = 8  
(h) Multiplier = 10
Table 4: Area under curve values (AUC) for the BSEM models with and without global shrinkage

<table>
<thead>
<tr>
<th>Multiplier</th>
<th>Treatment without EB</th>
<th>Proteins without EB</th>
<th>Full BSEM</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2</td>
<td>0.8697</td>
<td>0.8691</td>
<td>0.8843</td>
</tr>
<tr>
<td>1.4</td>
<td>0.8667</td>
<td>0.8845</td>
<td>0.8705</td>
</tr>
<tr>
<td>2.6</td>
<td>0.8725</td>
<td>0.8912</td>
<td>0.8713</td>
</tr>
<tr>
<td>3.8</td>
<td>0.8688</td>
<td>0.8814</td>
<td>0.8609</td>
</tr>
<tr>
<td>5.0</td>
<td>0.8659</td>
<td>0.8787</td>
<td>0.8595</td>
</tr>
<tr>
<td>6.5</td>
<td>0.8608</td>
<td>0.8693</td>
<td>0.8631</td>
</tr>
<tr>
<td>8.0</td>
<td>0.8554</td>
<td>0.8697</td>
<td>0.8555</td>
</tr>
<tr>
<td>10.0</td>
<td>0.8555</td>
<td>0.8647</td>
<td>0.8572</td>
</tr>
</tbody>
</table>

Figure: 46 considers the selection of the proteins over 100 iterations. The three plots are very similar, the same false positive peaks occur in all three methods. However some peaks are slightly larger or smaller with the different priors.
Figure 46: Barplot of the number of times protein $j$ has been selected as to be in top 10 largest absolute coefficients over 100 iterations, for different shrinkage methods and different multiplier values.
9 Conclusion

A few interesting observations can be made from the simulation data. First the results seem to show that the extra layer of horizontal shrinkage that the BSEM + covariate model compared to the single shrinkage model, strengthens the model when the effects are small. However as the effects become larger the extra layer of shrinkage seems to make it harder to select the absolute standardized posterior mean of the $m$ affected proteins among the $m$ largest proteins over all $p$ proteins. This is shown by the median ROC curve plots and the mean AUC values. This is also, albeit in lesser strength, visible when running the BSEM method without global shrinkage on the proteins and thus using vague priors and not using the global shrinkage algorithm outlined in Leday et al. [2017]. Not using global shrinkage on the proteins seems to work better when the effects on the affected proteins get moderately to very large. This could be due to a small spill over effect due to the collinearity of the proteins, when the affected protein effects were much larger than the regular protein effects. Another interesting comparison can be made between the BSEM + covariate model and the Differential Expression model using the BSEM algorithm. The differential expression model, models the marginal treatment effects whereas the BSEM + covariate models the conditional treatment effect (conditioned on the proteins in the model). This makes comparing the models unfair, as the BSEM + covariate model is designed for our goal of finding which proteins are directly influenced by the covariate. Table: 2 shows that when the effect increases, the performance of the Differential Expression model comes closer to the BSEM + covariate model. Showing that, especially when effects are small, the BSEM model benefits a lot from correcting for the protein interaction graph. However when the treatment effects are large, correcting for the other proteins in the model is less important and the Differential Expression than seems to select the $m$ affected proteins better. It seems to be advantageous to include the treatment covariate, as is visible from the bayes factors being larger than one when comparing the model with and without treatment covariate. The Logistic Regression model seems to perform better when the treatment effect gets larger. The logistic regression model actually is likely to outperform the BSEM + covariate model (and the other methods) when the treatment effects are very large. In general the BSEM model outperforms other methods when the effects are small, but gets outperformed when the effects are very large as the performance of the BSEM + covariate model stays very constant over the effect sizes. In most practical setting, the large effect size of the affected features (compared to the small effects of the unaffected features) is not realistic. In an omics setting the effect sizes where the multiplier is equal to 10 rarely exist, as the protein expressions are often quite small. Hence in a practical omics setting the BSEM method is likely to outperform other methods. However in other
settings it might be wise to have some idea on the size of the effects when picking a method. As the Logistic Regression outperforms the other methods when very big effects are present, however it seems that the single shrinkage INLA model or the BSEM model performs better in smaller and medium effects. Where the BSEM + covariate model has the benefit that it is computationally much faster.

The BSEM method could be improved. The coefficients of the proteins are occasionally shrunken to much, causing the treatment covariate to overpower in some models and in some models the protein coefficients were quite large. It is not completely clear what causes either. Further work could be spend in using more datasets for the data-based simulations, also including high dimensional datasets and adding more affected genetic variables to study the behaviour of the coefficient sizes better. From the obtained information of these simulation studies a next step could be in modelling the prior distributions of the BSEM model. A spike-and-slab prior on the treatment covariate could possible give room for the treatment effects of the affected proteins to be large, but could make the treatment effects in the unaffected proteins zero. A spike-and-slab priors on the treatment covariate could be implemented using the ShrinkBayes methodology, shown in :: Van de Wiel et al. [2014].

Another simple change in the model could be simply fitting a single $\tau$ over all regression models. This causes the specific models that show very small protein coefficients in the model to have a larger prior variance than they do in our model and hence give room for the affected proteins effects to be larger. Another way to model the treatment effect is using the prior scheme outlined in Hahn et al. [2018]. This method is build to reduce “regularization-induced confounding”, the tendency of regularization priors to bias treatment effect estimates by over penalizing control variable regression coefficients. They define confouding variables, variables that both affect the treatment covariate and the outcome variable. The treatment effect is biased due to confounding by the genes. This is due to the prior preferring small $\beta$ coefficients, when confounding is strong. This is countered by over stating the strength of the treatment effect parameter. It is shown in the results that in some regression models the coefficients of the proteins are very small and the treatment coefficient is very large. Hahn et al. [2018] try to cancel this by considering the following two equation model:

\[
\begin{align*}
Z &= X\gamma + \epsilon, & \epsilon &\sim \mathcal{N}(0,\sigma^2_\epsilon) \\
y &= Z\alpha + X\beta + \nu, & \nu &\sim \mathcal{N}(0,\sigma^2_\nu)
\end{align*}
\]

The first equation models the impact that the control variables, $X$, have on the treatment level, $Z$. The second equation shows the impact of the treatment and controls on the conditional expectation of the outcome variable $y$. Because the control variables appear in both equation the first equation
reflects the confounding influence of the controls. However the results also showed that some regression models had large protein coefficients. In this case adding a sparse prior on the protein-protein effects, shrinking these effects towards zero could be a promising solution.

Furthermore, a more "proper" selection method could be found/designed, as in the above simulations a lot of calculations are done on the raw posterior coefficients for the treatment covariate. A better selection criterium was outside of the scope of this project.

10 Acknowledgements

First of, I would like to thank my supervisor Mark. His ungoing enthousiasm was very contagious and without his guidance I would not have been able to finish this thesis. Mark spend much time with me explaining and sending me into the right direction and taught me much about statistics and research in general. Also important, Mark was always in for a joke whether it was at the expense of him or me! I greatly enjoyed working at the department of epidemiology and biostatistics at the VU medical center. During my time I always felt welcome and a member of their group and I would like to especially thank the big statistics group (in alphabetical order): Armin, Carel, Gino, Iuliana, Magnus, Mirrelijn, Renee, Tim and Wessel. A special thanks to Magnus for helping me make my switch from psychology to statistics, a choice I am very pleased with. A final mention should go to the teachers of the statistical science program, who gave me plenty of work and exercises to do, enabling me to understand many statistical concepts needed to write this thesis. I am very grateful they made me work hard. Three years ago I would have never imagined writing this thesis.

September 5, 2018
F Constants added to the pertubated proteins

Table 5 shows the constants added to the pertubated \( m \) proteins for each value in the multiplier grid. The table is equal to:

\[
\gamma \otimes C,
\]

where \( \gamma = \{-0.2, -0.156, -0.111, -0.067, -0.022, 0.0667, 0.111, 0.156, 0.2\}, C = \{1, 3.25, 5.5, 7.75, 10\} \) and \( \otimes \) is the Kronecker product.

Table 5: Perturbation values added to protein \( 1 \ldots m \ \forall i \in C_i \)

<table>
<thead>
<tr>
<th>multiplier_seq</th>
<th>Protein 1</th>
<th>Protein 2</th>
<th>Protein 3</th>
<th>Protein 4</th>
<th>Protein 5</th>
<th>Protein 6</th>
<th>Protein 7</th>
<th>Protein 8</th>
<th>Protein 9</th>
<th>Protein 10</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2</td>
<td>-0.04</td>
<td>-0.03</td>
<td>-0.02</td>
<td>-0.01</td>
<td>0.00</td>
<td>0.00</td>
<td>0.04</td>
<td>0.04</td>
<td>0.04</td>
<td>0.04</td>
</tr>
<tr>
<td>1.4</td>
<td>-0.28</td>
<td>-0.22</td>
<td>-0.16</td>
<td>-0.09</td>
<td>-0.03</td>
<td>0.03</td>
<td>0.09</td>
<td>0.16</td>
<td>0.22</td>
<td>0.28</td>
</tr>
<tr>
<td>2.6</td>
<td>-0.52</td>
<td>-0.40</td>
<td>-0.29</td>
<td>-0.17</td>
<td>-0.06</td>
<td>0.06</td>
<td>0.17</td>
<td>0.29</td>
<td>0.40</td>
<td>0.52</td>
</tr>
<tr>
<td>3.8</td>
<td>-0.76</td>
<td>-0.59</td>
<td>-0.42</td>
<td>-0.25</td>
<td>-0.08</td>
<td>0.08</td>
<td>0.25</td>
<td>0.42</td>
<td>0.59</td>
<td>0.76</td>
</tr>
<tr>
<td>5.0</td>
<td>-1.00</td>
<td>-0.78</td>
<td>-0.56</td>
<td>-0.33</td>
<td>-0.11</td>
<td>0.11</td>
<td>0.33</td>
<td>0.56</td>
<td>0.78</td>
<td>1.00</td>
</tr>
<tr>
<td>6.5</td>
<td>-1.30</td>
<td>-1.01</td>
<td>-0.72</td>
<td>-0.43</td>
<td>-0.14</td>
<td>0.14</td>
<td>0.43</td>
<td>0.72</td>
<td>1.01</td>
<td>1.30</td>
</tr>
<tr>
<td>8.0</td>
<td>-1.60</td>
<td>-1.24</td>
<td>-0.89</td>
<td>-0.53</td>
<td>-0.18</td>
<td>0.18</td>
<td>0.53</td>
<td>0.89</td>
<td>1.24</td>
<td>1.60</td>
</tr>
<tr>
<td>10.0</td>
<td>-2.00</td>
<td>-1.56</td>
<td>-1.11</td>
<td>-0.67</td>
<td>-0.22</td>
<td>0.22</td>
<td>0.67</td>
<td>1.11</td>
<td>1.56</td>
<td>2.00</td>
</tr>
</tbody>
</table>
References


