Linear Regression methods and applications

FYS-STK4155: Project 1

Tankred Saanum, Maria Nareklishvili, Ulrik Seip
github.com/MariaRevili/FYS-STK4155

October 10, 2020

Abstract

Using synthetic and a digital terrain data we apply and test the performance of the following linear regression algorithms: ordinary least squares (OLS), Ridge and Lasso. To illustrate the performance of the methods, we use MSE and R^2 score metrics. We estimate and illustrate these measures using bootstrapping and cross-validation resampling schemes. The matrix of independent variables consists of the polynomial of up to and including degree 5. The results in the report illustrate the superior predictive performance of the OLS method. Finally, we discuss the results and possible implications thereof.

Contents

1. INTRODUCTION

Linear regression methods are fundamental in statistical modelling. These methods have gained popularity due to their simplicity, flexibility and interpretability. In the simplest form, linear regression is a framework for estimating the relation of a set of independent variables on an outcome variable (also called dependent variable) of interest. Among the underlying assumptions in this framework are the additivity of the predictors $\mathbf{X}$ and their linear dependence on the outcome variable y. By means of minimizing squared prediction error (i.e. the squared difference between the observed outcome and model predictions), we can derive a set of parameters $\boldsymbol{\beta}$ (one for each independent variable) with which we can make an approximation of the outcome variable $\tilde{y}$, defined as a linear combination of the independent variables and parameters, $\tilde{y} = \mathbf{X}\boldsymbol{\beta}$. An attractive property of this framework is that there are analytical expressions for deriving the optimal parameters for certain classes of linear regression methods.

In today's society, there is an abundance of data which lends itself to statistical modelling methods, such as linear regression. In this report, we use a range of linear regression methods to model and analyze both synthetic and real data sets. The real data set comprises terrain data (with a dependent variable encoding height, and two independent variables encoding x and y coordinates) from Møsvatn Austfjel, a rugged and hilly region in south-east Norway. The report describes the underlying statistical and mathematical theory of the methods, introduces the data, illustrates the results from the synthetic and real terrain data modelling. We then discuss the results and performance of the various regression methods.

2. PREDICTION METHODS

A. Unconstrained Linear Regression

This section introduces the linear regression algorithms. The outcome variable (i.e. the *response*) $\mathbf{y}$ represents a column vector of size n (i.e. $\mathbf{y}$ consists of n observations). The predictor matrix has dimensions $n \times p$ where n is the number of observations (items) and p is the number of columns, i.e. predictor variables. The predictor matrix is denoted by $\mathbf{X}$, and is also called the *design matrix*. Imposing a linear relation between $\mathbf{X}$ and $\mathbf{y}$ leads to the *linear regression equation*:

$$\begin{aligned} \mathbf{y} &= \tilde{\mathbf{y}} + \boldsymbol{\varepsilon} \\ \tilde{\mathbf{y}} &= \mathbf{X}\boldsymbol{\beta} \end{aligned} \tag{1}$$

$\tilde{\mathbf{y}} = \mathbf{X}\boldsymbol{\beta}$ represents the linear model (i.e. the proxy for the real data $\mathbf{y}$). β stands for the parameter vector and consists of p coefficients $\beta_j, ...\beta_p$. Each β_j describes the relation of a $j-$th column of $\mathbf{X}$ and the response $\mathbf{y}$. (1) is an *unconstrained* linear regression equation, as the unknown parameters are free to take any value from $-\infty$ to $+\infty$ (e.g. the constraint $\beta > 0$ would restrict the parameter space to be only defined for positive values in the multi-dimensional coordinate system).

In any non-trivial case, the model does not describe the outcome variable precisely, therefore, $\boldsymbol{\varepsilon}$ is an additional error term vector added to it. As such, each component of the error vector $\varepsilon_j = \tilde{y}_j - y_j$ corresponds to the difference between the $j-$th predicted observation and the real data value. Consequently, our objective in Equation (1) is to approximate the outcome variable as closely as possible, and thereby minimize the error term.

Ordinary Least Squares

Minimizing the cost or the error term requires minimizing the size of the error. A commonly used metric (i.e. function) for the size of the error is an L^2 norm/distance ($\|\mathbf{x}\|_2 = \sqrt{\sum_i x_i^2}$, which is a generalization of the straight-line distance between two points in an Euclidean space) (Horn and Johnson (1990)). The ordinary least squares algorithm minimizes the squared L_2 norm. This guarantees that the cost function is convex and there exists a minimum. Following Hastie and Tibshirani (2001) the cost function is written as follows:

$$C(\boldsymbol{\beta}) = \|\mathbf{y} - \tilde{\mathbf{y}}\|_2^2 = \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 = \quad (2)$$
$$\sum_{i=1}^{n}\Big(y_i - \beta_0 - \sum_{j}^{p} X_{ij}\beta_j\Big)^2 =$$
$$(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^T(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})$$

Minimizing (2) with respect to the parameter vector leads to the least squares solution:

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}}\big\{C(\boldsymbol{\beta})\big\} \quad (3)$$
$$0 = \frac{\partial C(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = \frac{\partial}{\partial \boldsymbol{\beta}}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^T(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})$$
$$-2\mathbf{X}^{\mathbf{T}}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) = 0 \Rightarrow$$

$$\boldsymbol{\beta}^{\text{OLS}}_{\text{estimator}} = \left(\mathbf{X}^{\mathbf{T}}\mathbf{X}\right)^{-1}\mathbf{X}^{\mathbf{T}}\mathbf{y}$$

where the notation $\boldsymbol{\beta}^{\text{OLS}}_{\text{estimator}}$ denotes the estimated optimal parameter vector obtained from the ordinary least algorithm. The procedure outlined in (3) minimizes the cost function by taking partial derivatives with respect to the unknown parameter of interest. If the design matrix $\mathbf{X}$ has full column rank (i.e. all columns are linearly independent), we are guaranteed that the least squares solution is unique. Moreover, $\mathbf{X}^{\mathbf{T}}\mathbf{X}$ is a $p \times p$ matrix where p is the dimension of the columns in $\mathbf{X}$. Assuming now that the number of columns is less than the observations (i.e. $p < n$), the matrix $\mathbf{X}^{\mathbf{T}}\mathbf{X}$ can be inverted.

B. Constrained Linear Regression

In a high-dimensional setup where the number of predictors is relatively large (and possibly highly correlated), the estimated coefficients $\hat{\boldsymbol{\beta}}$ exhibit high variability. That is, the highly negative coefficient on a variable might be cancelled by a highly positive coefficient on a correlated predictor. Furthermore, such coefficients may end up *overfitting* the outcome variable we seek to model: Seeking to explain away variability in the training set by adopting (possibly extreme) coefficient values, our estimated model may generalize poorly to test data, with the coefficients being highly tailored to the training data.

These issues can be overcome by introducing a constraint on the parameters in the minimization problem posed in (3).[1] In our setup, the general form of *constrained linear regression* can be defined as follows:

$$\arg\min_{\boldsymbol{\beta}}\big\{C(\boldsymbol{\beta})\big\} = \|\mathbf{y} - \tilde{\mathbf{y}}\|_2^2 \quad (4)$$
$$\text{s.t. } l(\boldsymbol{\beta}) \leq t$$

where $l(\boldsymbol{\beta})$ denotes the norm (i.e. the measure of the size) of the parameter vector $\boldsymbol{\beta}$. Equation 4 can be minimized by Lagrange

[1] Note that the constrained optimization reduces variability of the parameters, as the parameter space is not varying from $-\infty$ to $+\infty$, but instead from $-t$ to t where t is a constant.

multiplier methods (see Bertsekas (2014):

$$\begin{aligned} &\arg\min_{\boldsymbol{\beta}}\{C(\boldsymbol{\beta})\} \\ &\Rightarrow \arg\min_{\boldsymbol{\beta}} \|\mathbf{y}-\tilde{\mathbf{y}}\|_2^2 + \lambda l(\boldsymbol{\beta}) \\ &\Rightarrow \arg\min_{\boldsymbol{\beta}} \|\mathbf{y}-\mathbf{x}\boldsymbol{\beta}\|_2^2 + \lambda l(\boldsymbol{\beta}) \end{aligned} \tag{5}$$

Here, λ is a Lagrange multiplier which binds together the function that we are minimizing and the imposed constraint. λ is also called a *penalization parameter*, or a tuning parameter.

Commonly used norms are the L^2 norm and the L^1 norm, which lead to Ridge and Lasso regression described below.

Ridge

Hoerl and Kennard (1976) introduced the additional term λ which regularizes (i.e. penalizes) large values of $\boldsymbol{\beta}$. In Ridge regression the underlying measure of the parameter size is the L_2 norm. Thus, the cost function can be re-formulated as follows:

$$\begin{aligned} C_{\mathrm{R}}(\boldsymbol{\beta}) &= \|\mathbf{y}-\tilde{\mathbf{y}}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_2^2 \\ &= \|\mathbf{y}-\mathbf{X}\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_2^2 \\ &= \sum_{i=1}^{n}\left|y_i-\beta_0-\sum_{j=1}^{p}X_{ip}\beta_p\right|^2 + \lambda\sum_{j=1}^{p}\beta_j^2 \end{aligned} \tag{6}$$

The shrinkage parameter λ favours coefficient values that are closer to 0 and penalizes values that are more extreme. The intercept β_0 is left out from the regularization. The reasoning for this is simple. Imagine we shrink every parameter value (including the intercept) to 0, then the model becomes $\mathbf{y}=\boldsymbol{\varepsilon}$, and $\mathbb{E}(\boldsymbol{\varepsilon})=\mathbb{E}(\mathbf{y})\neq 0$ - this violates one of the main assumptions in the linear regression settings - that the expectation of the error term is 0. If we instead drop out the intercept from the regularization, then, upon shrinking the remaining parameters to 0, we have a model $\mathbf{y}=\boldsymbol{\beta}_0+\boldsymbol{\varepsilon}$. Consequently, $\mathbb{E}(\mathbf{y})=\mathbb{E}(\boldsymbol{\beta}_0)+\mathbb{E}(\boldsymbol{\varepsilon})=\mu$. In other words, the intercept is the assumed underlying model, where $\mathbb{E}(\boldsymbol{\varepsilon})=0$ and the model is unbiased (Hastie and Tibshirani (2001)).

As in the OLS algorithm, we can minimize the cost function by taking the partial derivatives with respect to the parameters.

$$\begin{aligned} \frac{\partial C_{\mathrm{R}}(\boldsymbol{\beta})}{\partial\boldsymbol{\beta}} &= \frac{\partial}{\partial\boldsymbol{\beta}}\Big[(\mathbf{y}-\mathbf{X}\boldsymbol{\beta})^T(\mathbf{y}-\mathbf{X}\boldsymbol{\beta})+\lambda\mathbb{1}\boldsymbol{\beta}^T\boldsymbol{\beta}\Big] \\ &= -2\mathbf{X}^T(\mathbf{y}-\mathbf{X}\boldsymbol{\beta})+2\lambda\mathbb{1}\boldsymbol{\beta} \overset{!!}{=} 0 \\ \Rightarrow \mathbf{X}^T\mathbf{y} &= \mathbf{X}^T\mathbf{X}\boldsymbol{\beta}+\lambda\mathbb{1}\boldsymbol{\beta} \end{aligned}$$

Hence:

$$\boldsymbol{\beta}^{\text{Ridge}}_{\text{estimator}} = \left(\mathbf{X}^T\mathbf{X}+\lambda\mathbb{1}\right)^{-1}\mathbf{X}^T\mathbf{y}.$$

Lasso

As mentioned above, the regularization is not restricted to the L^2 norm metric. We can instead choose to define the metric of the vector $\boldsymbol{\beta}$ to be the L^1 norm. This constitutes setting up the cost function as:

$$\begin{aligned} C_{\mathrm{L}}(\boldsymbol{\beta}) &= \|\mathbf{y}-\tilde{y}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_1 \\ &= \|\mathbf{y}-X\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_1 \\ &= \sum_{i=1}^{n}\left|y_i-\beta_0-\sum_{j=1}^{p}X_{ip}\beta_p\right|^2 + \lambda\sum_{j=1}^{p}|\beta_j| \end{aligned} \tag{7}$$

Originally popularized by Tibshirani (1996), the *Lasso regression* (i.e. "least absolute shrinkage and selection operator") has considerable advantages over Ridge and OLS solutions. Most notably, Lasso performs *variable selection*. That is, some

parameters β_j are exactly zero after the minimization. Deriving the cost function with respect to the parameter yields:

$$\begin{aligned}\frac{\partial C_{\mathrm{L}}(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}_j} &= \frac{\partial C_{\mathrm{OLS}}(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}_j} + \lambda \sum_{j=1}^{p} \frac{\partial}{\partial \beta_j}|\beta_j| \\ &= \frac{\partial C_{\mathrm{OLS}}(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}_j} + \lambda \frac{\beta_j}{\sqrt{\beta_j^2}} \\ &= \frac{\partial C_{\mathrm{OLS}}(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}_j} + \lambda \,\mathrm{sgn}(\beta_j) = 0 \quad (8)\end{aligned}$$

Unfortunately, (8) cannot be solved analytically. As such, to obtain the optimal parameters, we need to use numerical optimization algorithms.

C. Singular Value Decomposition

In the sections described above, we assumed that the matrix $(\mathbf{X^T X})$ was invertible. That is, we assumed that the columns of this matrix were linearly independent. In some cases however, columns of the matrix can be linearly dependent. That implies $(\mathbf{X^T X})$ is singular and therefore not invertible. In cases where we have a singular design matrix, we may still obtain the (pseudo-)inverse of $(\mathbf{X^T X})$ using a method called Singular Value Decomposition, or SVD for short. SVD allows us to decompose any $m \times n$ matrix into an $m \times m$ orthogonal matrix $\mathbf{U}$, an $m \times n$ diagonal matrix $\mathbf{\Sigma}$ and an $n \times n$ orthogonal matrix $\mathbf{V}$.

$$\mathbf{X} = \mathbf{U\Sigma V^T} \quad (9)$$

In equation (9) the elements of $\mathbf{\Sigma}$ are arranged in descending order, i.e. $\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_r > 0$ (Lay et al. (2016). Then the ordinary least squares solution can be expressed in the following way:

$$\begin{aligned}(\mathbf{X^T X})^{-1} &= (\mathbf{V\Sigma}^T\mathbf{U^T U\Sigma V^T})^{-1} \\ &= (\mathbf{V\Sigma}^T\mathbf{\Sigma}V^T)^{-1} = (\mathbf{V\Sigma}^2\mathbf{V^T})^{-1} \\ &= \mathbf{V\Sigma}^{-1}\mathbf{U^T} \quad (10) \\ \mathbf{X^T y} &= \mathbf{V\Sigma}^T\mathbf{U^T y}\end{aligned}$$

$$\begin{aligned}&\boldsymbol{\beta}^{\mathrm{OLS}}_{\mathrm{estimator}} = (\mathbf{X^T X})^{-1}\mathbf{X^T y} \\ &= \mathbf{V\Sigma}^{-1}\mathbf{U^T X}^T\mathbf{y}\end{aligned}$$

In this report, the OLS solutions are derived solely using singular value decomposition, while the Ridge estimator is obtained using both SVD and the original formula.

D. Bayesian inference

Before we proceed, we will describe the Ridge and Lasso methods from a Bayesian perspective, for the purpose of a more in-depth understanding of the algorithms.

The $\boldsymbol{\beta}$ coefficients obtained from the OLS algorithm are equivalent to to those obtained using the maximum likelihood estimation (MLE). The reason for this is that the outcome variable is assumed to be normally distributed with the mean $\mathbf{X}\boldsymbol{\beta}$ and variance σ^2, i.e. $Y \sim \mathcal{N}(\mathbf{X}\boldsymbol{\beta}, \sigma^2)$. MLE, in turn, involves setting up the likelihood function for the vector $\mathbf{y}$ and maximizing it with respect to the parameter vector $\boldsymbol{\beta}$:

$$\begin{aligned}&L(\mathbf{y}, \mathbf{X}, \boldsymbol{\beta}, \sigma^2) \\ &= \prod_{i=0}^{n-1} (\sqrt{2\pi}\sigma)^{-1} \exp\Big(-\frac{1}{\sigma^2}(\mathbf{Y_i} - \mathbf{X_i}\boldsymbol{\beta})\Big)^2 \\ &\log(L(\mathbf{y}, \mathbf{X}, \boldsymbol{\beta}, \sigma^2) = -n\log(\sqrt{2\pi}\sigma) - \\ &\frac{1}{2}\sigma^{-2}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 \quad (11)\end{aligned}$$

Maximizing this log likelihood function by taking the derivative with respect to $\boldsymbol{\beta}$

and performing simple algebraic calculations yields the same solution $\boldsymbol{\beta}^{\text{OLS}}_{\text{estimator}} = (\mathbf{X^T X})^{-1}\mathbf{X^T y}$ (see van Wieringen (2015) for further details). The likelihood function can be thought of as the joint probability of observing the data at hand. Thus, the optimal parameter should be the one maximizing the likelihood of observing the data.

Interestingly, from a probabilistic perspective, this is equivalent to using the maximum a posteriori values of $\boldsymbol{\beta}$ given the data, $\hat{\boldsymbol{\beta}} = \arg\max_{\boldsymbol{\beta}} p(\boldsymbol{\beta} \mid \mathbf{y})$, assuming that $\boldsymbol{\beta}$ is not governed by a prior distribution. That is, in the maximum-likelihood approach above, we neglected to define the prior distributions over parameters in the vector $\boldsymbol{\beta}$. However, employing a Gaussian (i.e. normal) prior distribution for this parameter and updating using Bayes theorem (Swinburne (2004)) leads to the likelihood function for the ridge estimator:

$$\begin{aligned}
\mathbf{y} \sim \mathcal{N}(\mathbf{X}\boldsymbol{\beta}, \sigma^2\mathbb{1}),\ \boldsymbol{\beta} \sim \mathcal{N}(0, \boldsymbol{\tau}^2\mathbb{1}) \\
p(\mathbf{y}|\boldsymbol{\beta}) \propto \exp\Big[-\frac{1}{2\sigma^2}(\mathbf{y}-\mathbf{X}\boldsymbol{\beta})^T(\mathbf{y}-\mathbf{X}\boldsymbol{\beta})\Big] \\
p(\boldsymbol{\beta}) \propto \exp\Big[-\frac{1}{2\tau^2}\boldsymbol{\beta}^T\boldsymbol{\beta}\Big] \\
p(\boldsymbol{\beta}|\mathbf{y}) \propto \exp\Big[-\frac{1}{2\sigma^2}(\mathbf{y}-\mathbf{X}\boldsymbol{\beta})^T(\mathbf{y}-\mathbf{X}\boldsymbol{\beta}) \\
-\frac{\boldsymbol{\sigma}^2}{\boldsymbol{\tau}^2}\boldsymbol{\beta}^T\boldsymbol{\beta}\Big] \qquad (12)
\end{aligned}$$

Intuitively, Ridge regression shrinks the variance of the parameter of interest: Since we assumed that $\boldsymbol{\beta}$ is governed by a normal distribution with variance τ^2, we penalize the likelihood of more extreme parameter values inversely proportional to the variance. The higher the variance of the distribution governing β, the more likely are more extreme β values. Interestingly, we can connect the variance τ^2 to the regularization parameter λ: To see this, we note that the optimal Ridge parameters can be described in terms of the mean squared error as well:

$$\hat{\beta}^{Ridge} = \arg\min_{\boldsymbol{\beta}\in\mathbb{R}^k} \frac{1}{N}\sum_i^N (y_i - \mathbf{X}_i\beta)^2 + \lambda\sum_j^k \beta_j^2$$

where $\lambda = \frac{1}{\tau^2}$. Consequently, as τ^2 grows, λ will decrease, and in the limiting case that $\lambda = 0$, $\hat{\beta}^{Ridge} = \hat{\beta}^{OLS}$. Conversely, as τ^2 shrinks (variance decreases), λ can grow infinitely large, and our parameter coefficients approach 0 as well.

In effect, this leads to the posterior distribution of $\boldsymbol{\beta}$ being more concentrated towards the mean. Moreover, after maximizing the likelihood with respect to the parameter, we end up with:

$$\begin{aligned}
\boldsymbol{\beta}^{\text{Ridge}}_{\text{estimator}} \sim \mathcal{N}\Big[\mathbf{w}_\lambda\mathbf{X^T y},\ \sigma^2\mathbf{w}_\lambda(\mathbf{X^T X})^{-1}(\mathbf{w}_\lambda)^T\Big] \\
\mathbf{w}_\lambda = (\mathbf{X^T X} + \lambda\mathbb{1}_{pp})^{-1} \qquad (13)
\end{aligned}$$

Equation (13) clarifies that both first and second moments (i.e. the mean and the variance) of the ridge estimator are quantitatively less than the equivalent moments for the OLS estimator (van Wieringen (2015)).

Since we are free to choose the prior distribution for $\boldsymbol{\beta}$, we can choose the double exponential (i.e. Laplace) distribution which leads to the lasso estimator (Park and Casella (2008)):

$$\begin{aligned}
p(\boldsymbol{\beta}^{\text{Lasso}}_{\text{estimator}}) \propto \prod_{j=0}^{p-1}\exp\Big[-\frac{|\beta_j|}{\tau}\Big] \\
(y|X,\beta) \sim \mathcal{N}(\mathbf{X}\boldsymbol{\beta}, \sigma^2)
\end{aligned}$$

where $\lambda = \frac{1}{\tau}$. Using Bayes rule, we get the posterior log likelihood function of $\boldsymbol{\beta}$:

$$\begin{aligned}
\log\left(p(\boldsymbol{\beta}^{\text{Lasso}}_{\text{estimator}}|X,y)\right) \propto \sum_{i=0}^{n-1}\Big[(-\frac{1}{\sigma^2}(Y_i - X_i\beta)\Big]^2 \\
+\lambda\sum_{j=0}^{p-1}(|\beta_j|) \qquad (14)
\end{aligned}$$

Maximizing the log posterior distribution function with respect to the parameter leads to the optimal Lasso estimator. The Laplace distribution is more sharply concentrated around zero than the normal distribution, and thus shrinks the posterior distribution of the parameters towards zero.

E. Assesing model accuracy

A fundamental aim in (supervised) machine learning applications is to predict a response variable from a set of observations. To assess a model's capacity for making correct predictions, we need to quantify how close the model's predicted response value is to the true outcome variable for each individual. A common metric is the *expected prediction error*:

$$\begin{aligned}\mathbb{E}\left(\mathbf{y}-\hat{f(\mathbf{x})}\right)^2 &= \mathbb{E}\left(f(\mathbf{x})+\boldsymbol{\varepsilon}-\hat{f(\mathbf{x})}\right)^2 \\ &= \mathbb{E}\left(f(\mathbf{x})-\hat{f(\mathbf{x})}\right)^2+\mathbb{E}(\varepsilon^2) \\ &= \mathrm{MSE}+\sigma^2 \qquad (15)\end{aligned}$$

Expected prediction error consists of the average difference between the true outcome variable $\mathbf{y}$ and the predicted target values. Expected prediction error is decomposed into the *mean squared error* and the *irreducible error* inherent to the true data. MSE is also called an *estimator error*, as it measures the deviation of the estimator function $\hat{f(\mathbf{x})}$ and the true function $f(x)$. To interpret MSE, we decompose it into *bias* and *variance* terms:

$$\begin{aligned}\mathrm{MSE} &= \mathbb{E}_\tau\left[\hat{f}(\mathbf{x})-\mathbb{E}_\tau\hat{f}(\mathbf{x})\right]^2 \\ &\quad+\left[\mathbb{E}(\hat{f}(\mathbf{x}))-f(\mathbf{x})\right]^2 \\ &= \mathrm{Var}(\hat{f(\mathbf{x})})+\mathrm{Bias}^2(\hat{f(\mathbf{x})}) \qquad (16)\end{aligned}$$

where $\mathbb{E}_\tau(\hat{f(\mathbf{x})})$ denotes the expected value of the estimated function over the test data.

Equation (16) suggests that we want to have both low variance and low bias. However, when the complexity of the model increases, bias tends to decrease, but variance increases. This phenomenon is known as the *bias-variance trade-off*. Overall, we find an optimal combination of both quantities where the MSE takes its minimum value.

In addition to the MSE, we use another measure, R^2 score, to assess the performance of the methods in this paper:

$$1-R^2 = 1-\frac{\sum_{i=0}^{n-1}(y_i-\hat{f}(y_i))^2}{\sum_{i=0}^{n-1}(y_i-\mathbb{E}(y_i))^2} \qquad (17)$$

R^2 score, also referred to as the coefficient of determination, provides a measure of how well observed outcomes are replicated by the model, based on the proportion of total variation of outcomes explained by the model. We want to maximize this value and thus minimize 1-R^2. In this paper, we calculate the metric using the test data.

3. RESAMPLING METHODS

The regression methods described in section 2 aim to minimise the error in our estimate. In general, one way of doing this is to fit a polynomial of a sufficiently high degree. The problem with using a high degree polynomial to fit the data is that we tend to run into problems with overfitting. There is granularity and noisiness to any data set. Generally, what we seek to do is to fit our model to the systematic variance in the data set, and avoid fitting the model to the noise. One way of achieving this is to employ resampling methods when fitting our models.

The main concept of a resampling method is to repeatedly train the model on randomly selected subsets of our available training data. Every trained model is evaluated, and the mean (or some other statistical quantity of interest) of the MSE (or the variance, bias, etc.) is used to assess the model. This can help us avoid overfitting in the sense that the model is trained on a training set, but evaluated on a different test set. By repeating this sufficiently many times we may be able to filter the systematic variance from the non-systematic variance in our data. The downside to this is that we have to set aside some of our data for test purposes, which gives us less data to train our model on, so a balance has to be struck. It is convention to make the test set 20% or 33.3% of the original data set. Hyperparameters (such as the penalization parameter λ in Ridge and Lasso regression) may also be tuned using such resampling methods.

F. The Bootstrap

The first resampling method we are going to look at is the bootstrap method. Given a training sample $Z = (z_1, z_2, ..., z_n)$ we create a new Z_1^* by picking n random z_i values with replacement, meaning that we can have duplicate values in Z_1^*. This is repeated B times. An expected value can then be computed from every training sample, $S(Z_1^*), S(Z_2^*), ..., S(Z_B^*)$. From this we can compute:

$$var(S(Z)) = \frac{1}{B-1}\sum_{i=1}^{B}(S(Z_i^*) - \bar{S}^*)^2 \tag{18}$$

The bootstrap method itself isn't necessarily used to improve a model but rather to infer statistical properties of a larger population from the available data. For machine learning purposes we can use all the averaged predicted values as our final model. Because bootstrapping simulates the entire population, the final model that consists of a combination of all the bootstrapped estimated values also simulate the entire population, rather than just the available data sample, and any analysis made on the final estimated values should also hold for the population.

G. Cross Validation

A slightly more complicated resampling procedure is cross validation. In this project we are going to limit ourselves to the k-fold cross validation, but other variants exist.

In a k-fold cross validation procedure the data is randomly split into k parts, where one of those parts is assigned the role of a test set. The model is then trained on the remaining data, and then tested against the test set. This process is repeated so that all k parts get to serve as a test set. The averaged error of all the $k-1$ runs is used as a performance metric of the model.

This gives us two valuable results. One is the average accuracy of the model. Suppose we have 10 folds. This gives us 9 separate fits to the data, from which we can get an idea of how well the model will perform on average given new data from the same source. More concisely we can say that the averaged error of all the $k-1$ runs is used as a performance metric of the model. The second result of interest is the actual fit. In the same way as with the bootstrap scheme, we can use the average estimated MSE values to obtain a final model.

4. DATA

We use synthetic and real data to test the aforementioned methods and illustrate their performance. We assess model performance on synthetic data before moving on to the real data. This section describes both.

H. Synthetic data

The Franke function, originally developed by Franke (1979), is a function with two Gaussian peaks. It is normally used in surface interpolation problems. Specifically, the Franke function $f_{\mathrm{F}}(x, y)$ takes the full form:

$$\begin{aligned} f_{\mathrm{F}}(x,y) = &\frac{3}{4}\exp\left\{\frac{-1}{4}\left[(9x-2)^2+(9y-2)^2\right]\right\} \\ &+\frac{3}{4}\exp\left\{\frac{-1}{49}(9x+1)^2+\frac{1}{10}(9y+1)^2\right\} \\ &+\frac{1}{2}\exp\left\{\frac{-1}{4}\left[(9x-7)^2+(9y-3)^2\right]\right\} \\ &-\frac{1}{5}\exp\left\{\frac{-1}{4}\left[(9x+4)^2+(9y-7)^2\right]\right\} \end{aligned} \tag{19}$$

This function is generally evaluated for the inputs $0 \leq x, y \leq 1$.

The synthetic data consists of the outcome variable given by the Franke function $f_{\mathrm{F}}(x, y)$ and the input matrix $\mathbf{X}$, which is the design/feature matrix. It contains linearly independent columns, which ensures unique identification of the parameters of interest.

Throughout the paper, we use two predictors x_1 and x_2 to generate the outcome variable. They take values between 0 and 1 and are sampled from a uniform distribution of size n.

$$x_1, x_2 \sim \mathcal{U}(0, 1) \tag{20}$$

To set up a design matrix, we will employ a homogeneous polynomial mapping.[2] Polynomial P defines the mapping $(x_1, x_2) \mapsto P(x_1, x_2)$. P can be thought of a function which maps (x_1, x_2) to the design matrix X of the dimension $n \times p$. The Franke function is a 2D function, thus we will consider x_1 and x_2 in all possible homogeneous combinations up to and including degree p ($p = 5$ in this paper). Polynomial of degree p consists of the $(\frac{p(p+3)}{n})$ terms and is formally given as follows:

$$\sum_{i=1}^{p}\sum_{j=0}^{i} x_1^{i-j}x_2^j \tag{21}$$

For example, when the polynomial degree $p = \{1, 2\}$, the linear regression equation becomes:

$$\begin{aligned} f_{\mathrm{F}}(x_1,x_2)_{p=1} &= \boldsymbol{\beta}_0 + \beta_1\mathbf{x_1} + \boldsymbol{\beta}_2\mathbf{x_2} + \boldsymbol{\varepsilon} \\ f_{\mathrm{F}}(x_1,x_2)_{p=2} &= \boldsymbol{\beta}_0 + \beta_1\mathbf{x_1} + \boldsymbol{\beta}_2\mathbf{x_2} + \boldsymbol{\beta}_3\mathbf{x_1^2} + \\ &\quad + \boldsymbol{\beta}_4\mathbf{x_1x_2} + \boldsymbol{\beta}_5\mathbf{y^2} + \boldsymbol{\varepsilon} \end{aligned}$$

Fig. 1 and Fig. 2 depict the Franke function surface with and without noise.

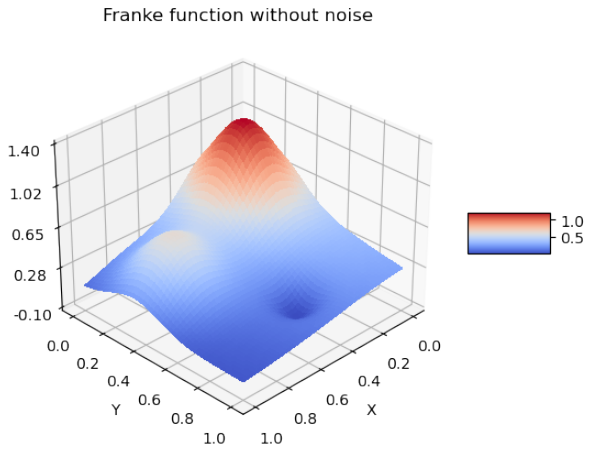


FIG. 1. The Franke function without noise plotted for $0 \leq x_1, x_2 \leq 1$.

[2] A homogeneous polynomial is a polynomial in which all terms have the same total degree, e.g. $x_1^2 + x_2^2 + x_1x_2$ is a homogeneous polynomial of degree 2.

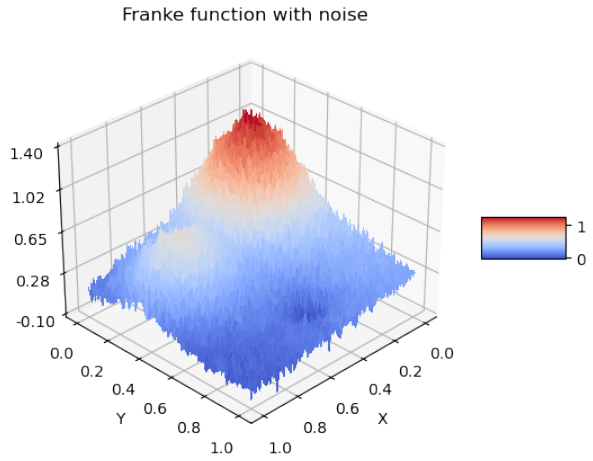


FIG. 2. The Franke function with noise plotted for $0 \leq x_1, x_2 \leq 1$. Noise $\sim 0.05 * \mathcal{N}(0, 1)$ here.

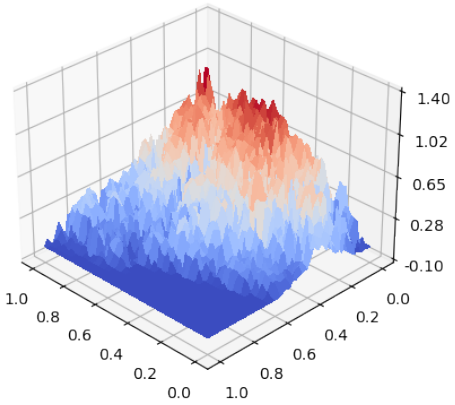


FIG. 3. The terrain data in use in the present work, taken from the Møsvatn Austfjell area. Retrieved using the USGS EarthExplorer website. The outcome variable (i.e.height) is scaled to fit in $0 \leq z \leq 1$.

I. Terrain data

The terrain data is extracted from the U.S Geological Survely (USGS). [3] Digital Terrain Elevation Data (DTED®) is a standard mapping format designed by the NGA. Each file or cell contains a matrix of vertical elevation values spaced at regular horizontal intervals measured in geographic latitude and longitude units. File size is approximately 25 MB for 1-arc-second data files and approximately 3 MB for 3-arc-second data files.

In this project we seek to model the Møsvatn Austfjell terrain, a region close to Stavanger, Norway. The terrain data can be seen in Fig. 3.

[3] The website: https://earthexplorer.usgs.gov

5. RESULTS AND DISCUSSIONS

J. Bias-Variance trade-off

First, we visualize and compare the bias-variance trade-off of the linear regression methods when normally distributed noise is added to the Franke function. Instead of $f_{\mathrm{F}}(x_1, x_2)$, a noisy Franke function takes the form:

$$f_{\mathrm{F,\ noise}}(x_1, x_2) = f_{\mathrm{F}}(x_1, x_2) + \eta\mathcal{N}(0, 1)$$

where $\mathcal{N}(0, 1)$ is the normally distributed vector of noise with 0 mean and standard deviation 1. η is a scaling term that controls the amount of noise to be added to the Franke function.

In general, when the data becomes more noisy, the fit of the methods worsens. This is captured by the increased mean squared error and the decreased coefficient of determination R^2, when the function is fitted with

ordinary least squares. When the model fits the data well, these quantities remain low. Once the noise becomes sufficiently large, they start to increase. This mechanism is depicted in Fig. 4. The plot shows that when the noise scale η is below 0.1, the MSE and $1 - R^2$ are low. Though, when the scale approaches 1, they increase substantially. That means that the noise is too high for the model to handle, so the fit is no longer representative of the underlying function. In this paper, we use η noise scaling terms up to 0.1.

Fig. 5 illustrates the MSE evaluated at the training and test data as a function of the polynomial degree with which we have fitted the Franke function, using OLS. Train and test MSE follow a similar decreasing trend up to and including polynomial degree 6. Afterwards, the MSE evaluated at the test data increases.

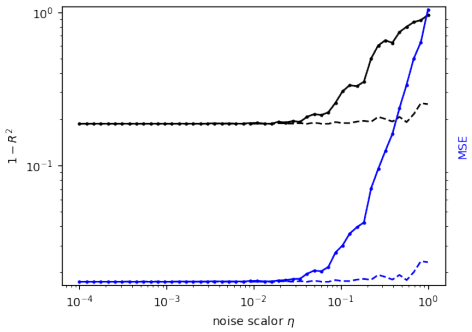


FIG. 4. The mean squared error and one minus the coefficient of determination R^2 as a function of the noise scale. The dotted line represents the MSE and $1 - R^2$ measured on the original Franke function without noise. Ordinary least squares is used in the fit with 1000 observations.

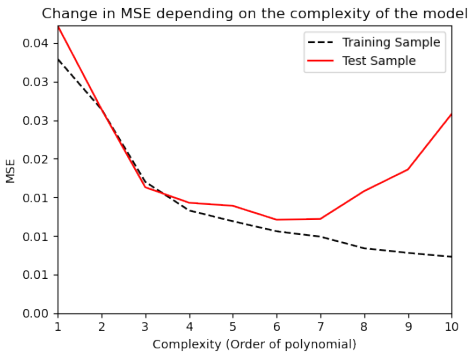


FIG. 5. The mean squared error on the train and test data for the OLS fit on the noisy Franke function. For illustrative purposes we use 300 observations here.

The reason for this is that, as the model complexity increases, it will employ more parameters which are highly tailored to the training data, but which generalize poorly to test cases. As such, prediction error increases after introducing a certain amount of complexity, indicating that we are overfitting.

The next quantity of interest in the paper is the bias-variance decomposition of the MSE.

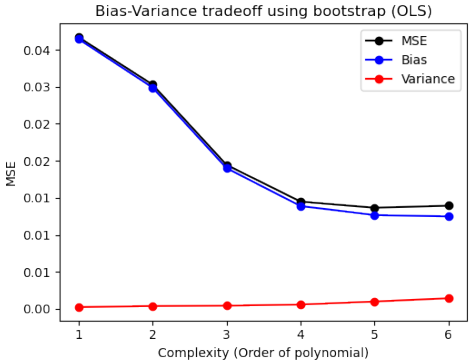


FIG. 6. Bias-variance decomposition for the OLS method (n=500 observations). Bootstrap scheme uses 500 replications of the train data.

OLS, Ridge and Lasso methods, depicted in Fig. 6, Fig. 7 and Fig. 8 respectively, exhibit similar trends. As already discussed, increasing the complexity of the model reduces bias but increases variance. Since the expected prediction error is the sum of the squared bias, variance and the irreducible error, there must exist a point where the MSE takes its minimum value. For all the methods, bias decreases rapidly but the variance increases rather slowly.

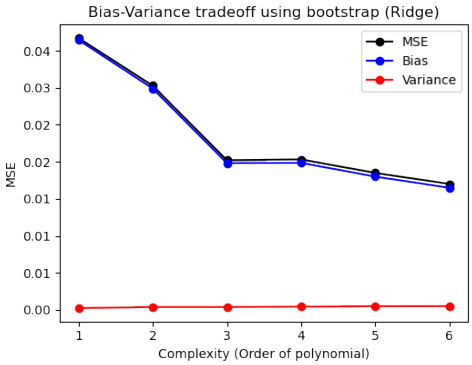


FIG. 7. Bias-variance decomposition for the Ridge regression (n=500 observations). Bootstrap scheme uses 500 replications of the train data. Shrinkage parameter $\lambda = 0.01$.

OLS seems to outperform the Ridge and Lasso regressions. A possible reason for this is that Ridge and Lasso regressions perform well in variance dominated regions, as they shrink the variability of the parameter coefficients employed by the model, and exhibit less variance in their predictions for novel data points. However, when the region exhibits highly systematic variability which can be approximated by introducing more model complexity, there may not be a lot of improvement in MSE offered by regularization schemes over the conventional OLS method. Furthermore, the very stable MSE values plotted over the complexity parameter λ for the Lasso method indicate that the shrinkage parameter is too high, as it sets most parameters to zero.

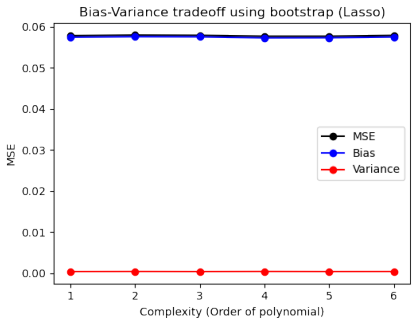


FIG. 8. Bias-variance decomposition for the Lasso regression (n=500 observations).Bootstrap scheme uses 500 replications of the train data. Shrinkage parameter $\lambda = 0.01$ (*note*: Stabilized values over the complexity parameter indicate that the shrinkage parameter is too high as it sets most parameters to zero).

The performance of the Ridge and Lasso methods largely depend on the hyperparameter λ. As such, Fig. 9 and Fig. 10 depict the dependence of the MSE on the shrinkage parameter λ. Up to and including polynomial degree 12, the MSE decreases for the OLS scheme. Additionally, we sought to estimate the effect of L^2-regularization on model bias, variance and train MSE with the bootstrap method. The results for a Ridge regression model with a polynomial design matrix of degree 5 can be seen in Figure 11. We estimated these quantities with 100 λ values regularly spaced on the logscale between -3 and 4, using 100 bootstraps.

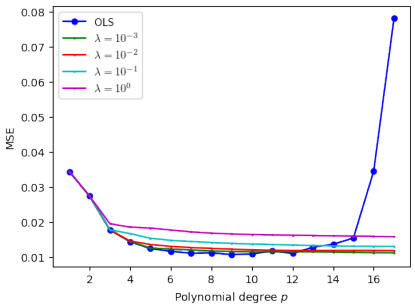


FIG. 9. The mean squared error as a function of the polynomial degree p for the Ridge regression. The OLS in the dotted blue line is shown for the comparison ($n = 1000$ observations). 10-fold Cross-validation is used to compute the MSE.

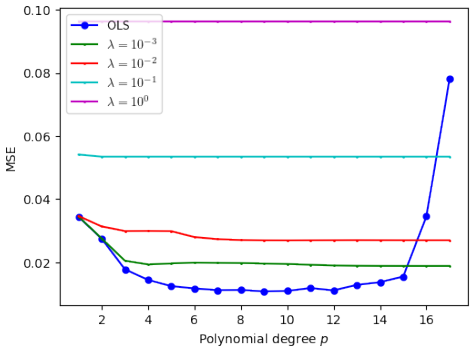


FIG. 10. The mean squared error as a function of the polynomial degree p for the Lasso regression. The OLS in the dotted blue line is shown for the comparison ($n = 1000$ observations).10-fold Cross-validation is used to compute the MSE.

Here we see that the bootstrap method's estimate of MSE increase monotonically with higher values of λ, suggesting that Ridge regularization may unwarranted for the regression problem posed by fitting the Franke function with a polynomial of fifth degree. However, it is too early to draw such conclusions as we only assessed train MSE. Moreover, the bootstrap method illustrates nicely how the variance of the predictions decrease with higher values of λ (as we regularize more), and the variance of the estimated variance decreases as well.

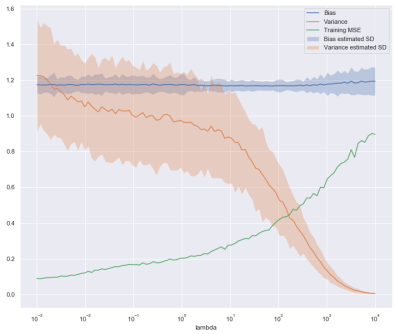


FIG. 11. Bias, variance and train MSE as a function of λ, for a Ridge regression model with polynomial features of degree 5. Shaded regions around a quantity x indicate that quantity +- σ_x. Quantities were estimated with 100 bootstraps for 100 λ values regularly spaced on the logscale between -3 and 4.

We then sought to assess the bootstrap and cross-validation methods when the objective is to find a λ which minimizes test MSE. We employed a bootstrap method identical to the one described above, this time estimating train and test MSE. Moreover, we estimated the same quantities with a 5-fold cross validation scheme. The results are visualized in Figure 12 and Figure 13, respectively. Here we see that the cross-validation method gives a different estimate of the efficacy of Ridge regularization.

Indeed, the 5-fold cross-validation scheme points to an optimal λ living in the region of 10^{-2}, whereas the bootstrap method points to a slightly lower λ as optimal. Interestingly, the bootstrap method shows that the estimated test (and train) MSE is quite variable, as indicated by the shaded region(s) in Figure 12, which comprise MSE +- σ of the estimated MSE.

Finally, we note that this particular effect of Ridge regularization may only hold for models whose complexity is of polynomial degree 5. As such, we obtained the cross-validated test MSE for all 100 λ values, for models with polynomial features up to and including degree 20. The results can be seen in Figure 14. Here we can see that the plane representing MSE as a 2-dimensional function of λ and model complexity has interesting non-linearities and several local minima. In the end, when we want to do model selection (for Ridge regression), we would search for the global minimum of such a plane.

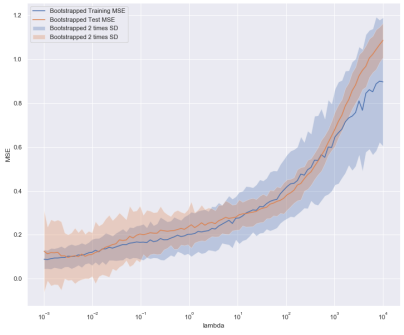


FIG. 12. Estimated train (blue) and test (orange) MSE from bootstrapping. Shaded regions around a quantity x indicate that quantity +- σ_x. Quantities were estimated with 100 bootstraps for 100 λ values regularly spaced on the logscale between -3 and 4.

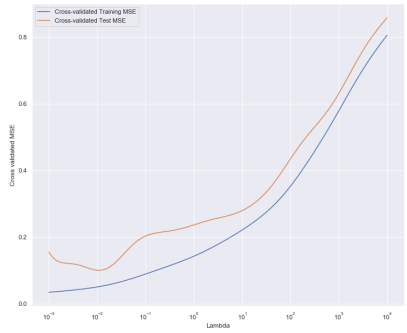


FIG. 13. Estimated train (blue) and test (orange) MSE from 5-fold cross-validation. Quantities were estimated for 100 λ values regularly spaced on the logscale between -3 and 4.

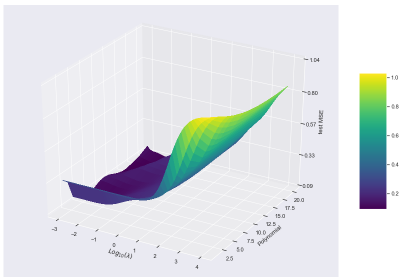


FIG. 14. Estimated test MSE from 5-fold cross-validation for polynomial features of various degrees. Models were fitted using Ridge regression. Quantities were estimated for 100 λ values regularly spaced on the logscale between -3 and 4.

We sought to make a similar analysis of the effect of L^1 regularization on MSE, i.e. with Lasso regression. We obtained the cross-validated test MSE for Lasso regression models with various complexities, anal-

ogous to the procedure described above. As per the observations made earlier in this report, we noted that too high λ values would end up eliminating all model parameters except the intercept, so we decided to use a smaller range of λ values (regularly spaced on the logscale between -6 and 1) for this analysis. As can be gathered from Figure 15, the plane representing test MSE does not display the same kinds of non-linearities as the plane obtained from L^2 regularization.

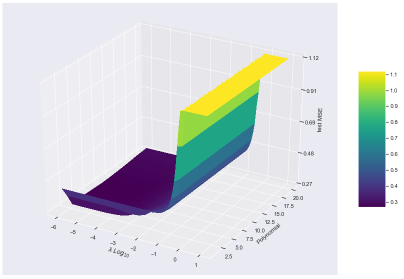


FIG. 15. Estimated test MSE from 5-fold cross-validation for polynomial features of various degrees. Models were fitted using Lasso regression. Quantities were estimated for 100 λ values regularly spaced on the logscale between -3 and 4.

K. Method fits

As a part of the validation analysis, it is natural to depict the fit of the linear methods to the underlying Franke function without noise. Fig. 18 shows the surface of the Franke function predicted by the OLS, Ridge and Lasso schemes, while Fig. 19 is an illustration of the absolute difference between the predicted outcome of the corresponding model and the real Franke function without noise. According to these results, OLS captures the true underlying outcome more precisely than the other methods. Lasso performs the worst out of the three models.

As an additional demonstration of the Ridge and Lasso hyper-parameter behaviour, we investigate the shrinkage of the coefficients as a function of λ. Fig. 16 and Fig. 17 verify that for increasing values of λ, parameters shrink and approach 0. That is, when $\lambda \Rightarrow \infty$, $\boldsymbol{\beta}^{Ridge}_{estimator}$, $\boldsymbol{\beta}^{Lasso}_{estimator} \Rightarrow 0$

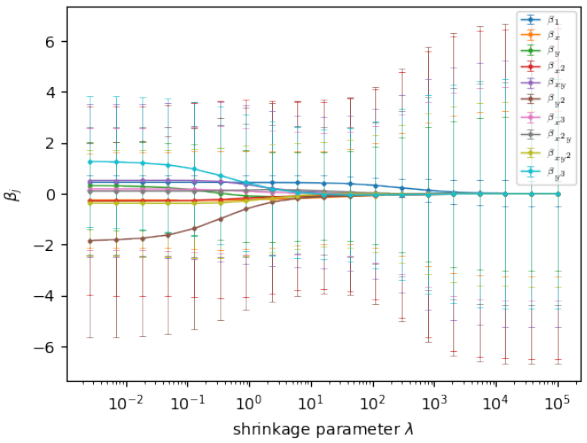


FIG. 16. The coefficients β_j for the Ridge regression as a function of varying shrinkage parameter λ. The error bars represent the confidence intervals for each β_j, calculated as $\beta_j \pm 2\sigma^2(\beta_j)$, where $\sigma^2(\beta_j) = \sigma^2(\mathbf{X^T X})^{-1}$ is the variance for each β_j . Polynomial degree $p = 3$.

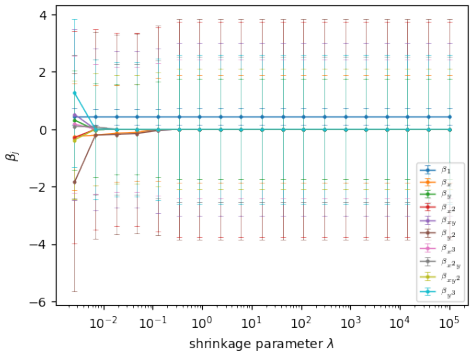


FIG. 17. The coefficients β_j for the Lasso regression as a function of varying shrinkage parameter λ. The error bars represent the confidence intervals for each β_j, calculated as $\beta_j \pm 2\sigma^2(\beta_j)$, where $\sigma^2(\beta_j) = \sigma^2(\mathbf{X^T X})^{-1}$ is the variance for each β_j . Polynomial degree $p = 3$.

L. Terrain data

Having illustrated the results for the synthetic data, we turn to the real terrain data. The size of the original data is 3601×1801. We split the train and test data into two equally sized folds.

To enhance the performance of the methods, we re-scale the outcome variable by subtracting the minimum value and dividing by the difference of its maximum and minimum values. This type of standardization is called min-max standardization. The outcome vector $\mathbf{y}$ then takes the form:

$$\tilde{\mathbf{y}} = \frac{\mathbf{y} - \min(\mathbf{y})}{\max(\mathbf{y}) - \min(\mathbf{y})} \tag{22}$$

The predictive performance of the OLS, Ridge and Lasso methods is shown in Fig. 21. Visual inspection makes it clear that the Lasso scheme is inferior to the other methods.

Fig. 20 illustrates the mean squared error for varying shrinkage parameters λ. The plot shows that OLS regression approximates the data best, Ridge comes in second, while the Lasso estimator has the highest MSE. To obtain better results for Lasso regression, one may need to use smaller λ values.

The possible explanations for this trend are manifold. The data has a highly non-linear structures. Two covariates and their polynomial terms up to and including 5th degree might not be enough to account for all non-linear structures in the data set. Increasing the polynomial degree will reverse the trend. Another reason may be that the terrain data exhibits systematic non-linearities which can be captured by adopting more variability in the model parameters. As such, regularization schemes may apply too conservative smoothing and parameter shrinking in this domain. As such, Ridge and Lasso methods do not lead to improved outcomes.

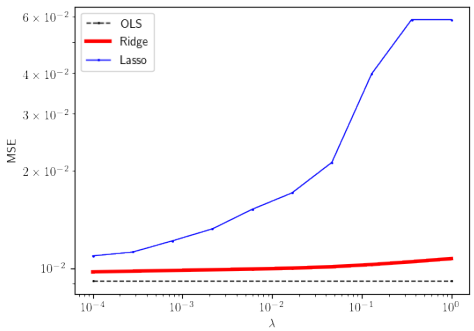


FIG. 20. The coefficients β_j for the Lasso regression as a function of varying shrinkage parameter λ. The error bars represent the confidence intervals for each β_j, calculated as $\beta_j \pm 2\sigma^2(\beta_j)$, where $\sigma^2(\beta_j) = \sigma^2(\mathbf{X^T X})^{-1}$ is the variance for each β_j . Polynomial degree $p = 3$.

6. CONCLUSION

In the present report we have performed OLS, Ridge and Lasso fits on the synthetic Franke function and the real terrain data. The results on both data sets indicate that the OLS exhibits superior prediction performance over the other methods. One of the potential reasons lies behind the complexity parameter. Increasing polynomial degree will reduce bias and increase variance, hence, the improved prediction performance of the Ridge and Lasso methods.

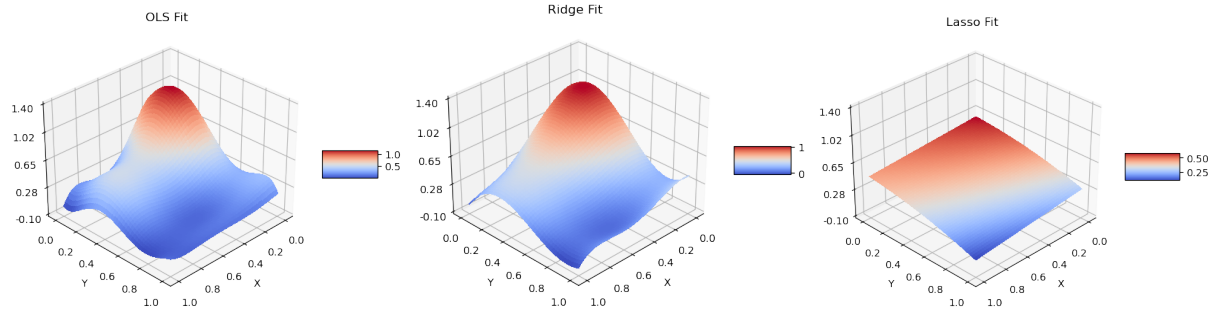


FIG. 18. Ordinarly least squares, Ridge and Lasso fits to the Franke function with no noise. Polynomial degree $p = 5$. The target function of Franke can be seen in Fig. 1.

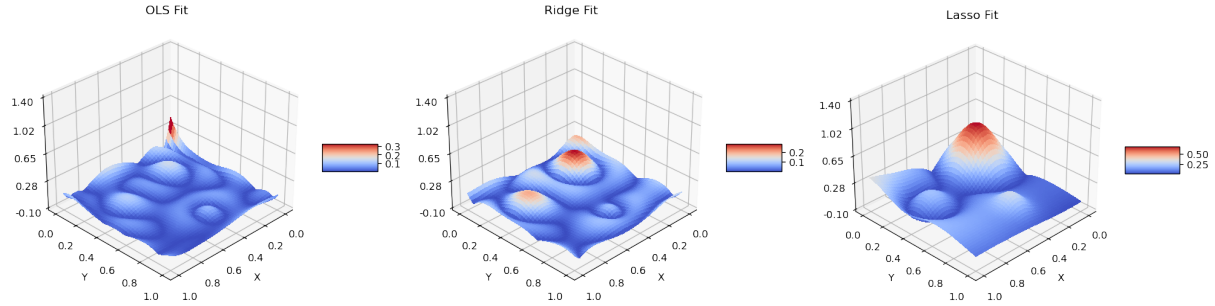
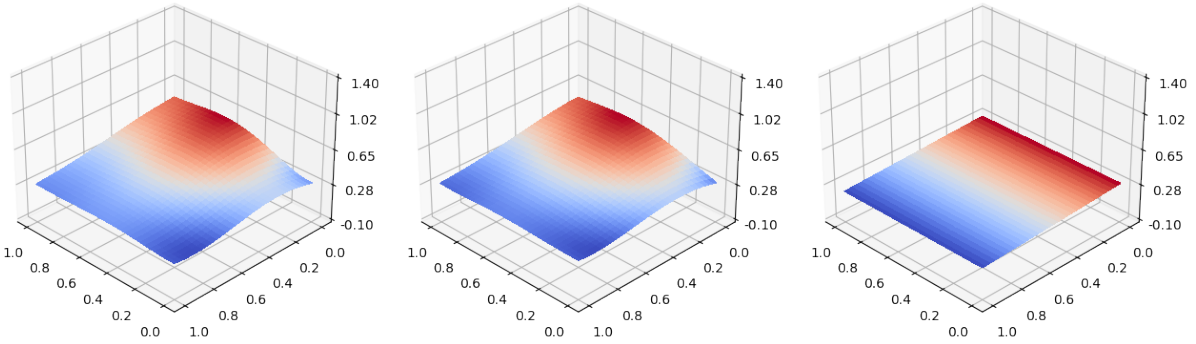


FIG. 19. Absolute difference of the ordinary least squares, Ridge and Lasso fits and the real Franke function without noise. Polynomial degree $p = 5$. The target function of Franke can be seen in Fig. 1.



OLS Ridge Lasso

FIG. 21. OLS, Ridge and Lasso predictions on the test data. A shrinkage parameter $\lambda = 0.1$ for both Ridge and Lasso schemes. Polynomial degree $p = 5$. The target test data can be seen in Fig. 3. Note that the OLS performs better than Ridge and Lasso. That is also corroborated by the fact that $\text{MSE}_{\text{OLS}} < \text{MSE}_{\text{Ridge}} < \text{MSE}_{\text{Lasso}}$

REFERENCES

Bertsekas, D. P. (2014). *Constrained optimization and Lagrange multiplier methods*. Academic press.

Franke, R. (1979). A critical comparison of some methods for interpolation of scattered data. Technical report, NAVAL POSTGRADUATE SCHOOL MONTEREY CA.

Hastie, T. and Tibshirani, R. (2001). Friedman. *The Elements of Statistical Learning," Springer*, page 52.

Hoerl, A. E. and Kennard, R. W. (1976). Ridge regression iterative estimation of the biasing parameter. *Communications in Statistics-Theory and Methods*, 5(1):77–88.

Horn, R. A. and Johnson, C. R. (1990). Norms for vectors and matrices. *Matrix analysis*, pages 313–386.

Lay, D. C., Lay, S. R., and McDonald, J. J. (2016). Linear algebra and its applications.

Park, T. and Casella, G. (2008). The bayesian lasso. *Journal of the American Statistical Association*, 103(482):681–686.

Swinburne, R. (2004). Bayes' theorem.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288.

van Wieringen, W. N. (2015). Lecture notes on ridge regression. *arXiv preprint arXiv:1509.09169*.