Regression Analysis and Resampling Methods

Sophus Gullbekk, Alessio Canclini, Jakob Wiig Ryther
(Dated: October 6, 2024)

The performance of Ordinary Least Squares (OLS), Ridge and Lasso regression is assessed both on synthetic data generated from the Franke function and real digital terrain data. To improve the statistical accuracy of the analysis both bootstrapping and k-fold cross-validation have been implemented as resampling techniques. We use bias-variance trade off analysis to find the best complexity for OLS and a grid search to find the best regularisation parameter and model complexity for Ridge and Lasso. We find that both Ridge and Lasso show an increasing MSE when we increase the regularisation parameter. On the synthetic data, OLS performs best achieveing an MSE of 3.7×10^{-4} using a polynimial degree of 8. For the digital terrain data we find OLS to perform best on the unscaled data, also for a polynomial degree of 8, giving an MSE of 5253.97. For scaled terrain data, Ridge regression achieves the lowest MSE of 0.16 using a polynomial degree of 15 and $\lambda = 10^{-6}$.

I. INTRODUCTION

Regression analysis is often considered to be important foundational knowledge within the vast world of Machine Learning (ML). It introduces key concepts of learning algorithms that allow the user to develop some intuition about the process before delving into more complex ML methods. This is not to say that the methods aren't useful in their own right. They are suited to predict continuous target variables based on one or multiple features, making them a useful tool within many branches of science [1]. Work utilising the Method of Least Squares, which underpins all these methods, was already published in the early 1800 by Adrien-Marie Legendre[2] in his studies of planetary movement. Although Legendre published first, it is believed that Carl Friedrich Gauss discovered the method [3]. This remains a notable dispute in the history of mathematics.

We will focus on the case where the least squares method is used to fit a linear model to the data, known as Ordinary Least Squares (OLS). The fact that the functions within OLS are linear with respect to the unknown parameters makes it possible to calculate analytical expressions for both the optimal parameters and the statistical properties of the model. This not only makes the OLS method easy to implement, but also gives us the tools we need to develop a more intuitive understanding of how it works. Although the original method remains unchanged, there are many models that build upon it.

In this analysis, we will evaluate the performance of three linear regression methods: OLS, Ridge regression [4] and Lasso regression [5]. They will first be implemented on the Franke function and later on digital terrain data. We will also use Bootstrap and k-fold cross-validation resampling methods to analyze the effects of the number of data points, the polynomial order, and the regularization parameter λ introduced in Ridge and Lasso. Key aspects of our analysis include the number of data points and the model complexity, which is determined by the polynomial degree. We will investigate how these factors effect our models performance by doing a bias-variance trade off analysis. Lastly, we will use the best parameters found to implement final models for each of the datasets.

Section II introduces the necessary theory and an overview of the methods. Results are then presented in Section III and discussed in Section IV. This is followed by some concluding remarks in Section V. All code used to generate the presented results can be found in the GitHub repository.[1]

II. THEORY AND METHODS

A. Linear Regression

Linear regression is a statistical method that models the relationship between features $\mathbf{x}^T = [x_1, x_2, \ldots, x_{n-1}]$ and target variables $\mathbf{y}^T = [y_1, y_2, \ldots, y_{n-1}]$, when we assume they are related through a continuous function $\mathbf{f}$:

$$\mathbf{y} = \mathbf{f}(\mathbf{x}) + \boldsymbol{\epsilon}, \tag{1}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma^2)$ [6]. The goal of linear regression is to find an approximation $\tilde{\mathbf{y}}$ of $\mathbf{f}$ using the system of linear equations:

$$\tilde{\mathbf{y}} = \mathbf{X}\boldsymbol{\beta} = \sum_{j=0}^{p-1} \mathbf{X}_j \beta_j. \tag{2}$$

Here, $\mathbf{X}$ is the design matrix and $\boldsymbol{\beta}^T = [\beta_0, \beta_1, \ldots, \beta_{p-1}]$ are the unknown parameters. To find the optimal parameters $\boldsymbol{\beta}$ we have to define a cost function that measures the error between the prediction and the target variables. A common cost function, that we will use to derive all the different linear regression methods used, is the Mean Squared Error (MSE):

$$C_{\text{OLS}}(\mathbf{X}, \boldsymbol{\beta}) = \frac{1}{n} \|\mathbf{y} - \tilde{\mathbf{y}}\|_2^2, \tag{3}$$

where $\|\cdot\|_2$ is the ℓ^2 norm.

[1] github.uio.no/sophusbg/FYS-STK4155/tree/main/project1

1. Ordinary Least Squares

In OLS we use the approximation from equation 1 and minimise the cost function from equation 3 with respect to the unknown parameters $\boldsymbol{\beta}$. The minimisation problem becomes

$$\hat{\boldsymbol{\beta}}_{\text{OLS}} = \min_{\beta \in \mathbb{R}^p} \frac{1}{n} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2.$$

To solve the minimisation problem, we first have to calculate its derivative

$$\frac{\partial}{\partial \boldsymbol{\beta}} \frac{1}{n} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 = -2(\mathbf{y} - \mathbf{X}\beta)^T \mathbf{X}.$$

By setting the derivative equal to zero, we get a closed form solution for the optimal parameters for OLS

$$\hat{\boldsymbol{\beta}}_{\text{OLS}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}. \tag{4}$$

2. Ridge Regression

Ridge regression is the first of two shrinkage methods we will investigate. Shrinkage methods use regularisation that constrain the size of the estimated parameters. This is done to reduce the variability of the estimated parameters [7]. We can derive Ridge regression by optimising a new cost function

$$C_{\text{R}}(\mathbf{X}, \boldsymbol{\beta}) = \frac{1}{n} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \lambda \|\boldsymbol{\beta}\|_2^2,$$

where λ is the regularisation parameter [8]. In the same way as we did with OLS, we can calculate an analytical expression for the optimal parameters $\hat{\boldsymbol{\beta}}_{\text{R}}$

$$\hat{\boldsymbol{\beta}}_{\text{R}} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}. \tag{5}$$

Subject to $\sum_i \beta_i \leq t$, where t is a positive finite number. Note that, when $\lambda > 0$ then $\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}$ is a full rank matrix and thus invertible. This is the main motivation behind the original paper on ridge regression by Hoerl and Kennard [4].

3. Lasso Regression

Lasso stands for least absolute shrinkage and selection operator', it shrinks some parameters like in ridge regression, but also forces some of them to zero [5]. Again, we can derive the Lasso regression by defining a new cost function

$$C_{\text{L}}(\mathbf{X}, \boldsymbol{\beta}) = \frac{1}{n} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \lambda \|\boldsymbol{\beta}\|_1,$$

where $\|\cdot\|_1$ is the ℓ^1 norm. By calculating the derivative of the cost function and setting it to zero we get the optimal parameters $\hat{\boldsymbol{\beta}}_{\text{L}}$ for lasso regression. Unlike with $\hat{\boldsymbol{\beta}}_{\text{OLS}}$ and $\hat{\boldsymbol{\beta}}_{\text{R}}$, we have no nice analytical expression for $\hat{\boldsymbol{\beta}}_{\text{L}}$. This is because the derivative of the ℓ^1 norm is discontinuous. The derivative of the cost function C_{L} is

$$\frac{\partial C_{\text{L}}(\mathbf{X}, \boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = -\frac{2}{n}\mathbf{X}^T(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) + \lambda \operatorname{sign}(\boldsymbol{\beta}).$$

Setting the derivative to zero and rearranging gives us

$$\mathbf{X}^T\mathbf{X}\boldsymbol{\beta} + \lambda \operatorname{sign}(\boldsymbol{\beta}) = 2\mathbf{X}^T\mathbf{y},$$

which does not have a nice analytical solution [9].

B. Statistical properties of Ordinary Least Squares

The expectation value for a given element y_i of $\mathbf{y}$ is given by

$$\mathbb{E}[y_i] = \mathbb{E}[f(x_i) + \epsilon_i] = f(x_i) \approx \tilde{y}_i = \sum_j x_{ij}\beta_j = \mathbf{X}_{i,*}\boldsymbol{\beta},$$

Since $f(x_i)$ is deterministic and $\mathbb{E}[\epsilon_i] = 0$ by definition. Furthermore, the variance of $\mathbf{y}$ is

$$\text{Var}[y_i] = \text{Var}[f(x_i) + \epsilon_i] = \text{Var}[\epsilon_i] = \sigma^2.$$

This shows that $y_i \sim N(\mathbf{X}_{i,*}\boldsymbol{\beta}, \sigma^2)$, meaning that $\mathbf{y}$ is normally distributed with a mean of $\mathbf{X}\boldsymbol{\beta}$ and a variance of σ^2. For OLS, the optimal parameters $\hat{\boldsymbol{\beta}}_{\text{OLS}}$ can be found by defining the MSE as a cost function and minimizing it,

$$\min_{\boldsymbol{\beta} \in \mathbb{R}^p} \frac{1}{n} \left\{ (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})^T (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}) \right\}.$$

This results in optimal parameters given by

$$\hat{\boldsymbol{\beta}}_{\text{OLS}} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y},$$

The expectation value of the optimal parameters is given by,

$$\begin{aligned}
\mathbb{E}[\hat{\boldsymbol{\beta}}_{\text{OLS}}] &= \mathbb{E}[(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}] \\
&= \mathbb{E}[(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T(\boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon})] \\
&= \mathbb{E}[(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\beta}] \\
&= \boldsymbol{\beta}.
\end{aligned}$$

Noting that $\text{Var}[\boldsymbol{\beta}] = 0$, $\text{Var}[\boldsymbol{\epsilon}] = \sigma^2$ and using the relation that for a non-stochastic matrix $\mathbf{A}$, we have $\text{Var}[\boldsymbol{A}\boldsymbol{x}] = \boldsymbol{A}\text{Var}[\boldsymbol{x}]\boldsymbol{A}^T$, one can show that the variance of $\hat{\boldsymbol{\beta}}$ is given by:

$$\begin{aligned}
\text{Var}[\hat{\boldsymbol{\beta}}_{\text{OLS}}] &= \text{Var}[(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}] \\
&= \text{Var}[(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T(\boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon})] \\
&= \text{Var}[(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\beta} + (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{\epsilon}] \\
&= \text{Var}[(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{\epsilon}] \\
&= (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\text{Var}[\boldsymbol{\epsilon}]\boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X})^{-1} \\
&= \sigma^2(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X})^{-1} \\
&= \sigma^2(\boldsymbol{X}^T\boldsymbol{X})^{-1}.
\end{aligned}$$

C. Error Analysis

Analysis is performed using three main methods; OLS, Ridge-, and Lasso regression. Each of these is assessed by evaluating the MSE

$$MSE(\boldsymbol{z},\tilde{\boldsymbol{z}}) = \frac{1}{n}\sum_{i=0}^{n-1}(y_i-\tilde{y}_i)^2,$$

and R^2 score,

$$R^2(\boldsymbol{z},\tilde{\boldsymbol{z}}) = 1 - \frac{\sum_{i=0}^{n-1}(y_i-\tilde{y}_i)^2}{\sum_{i=0}^{n-1}(y_i-\bar{y})^2}.$$

Here $\tilde{y}_i$ and y_i are the predicted value of the i-th element and the corresponding true value. $\bar{y}$ is the mean value of $\boldsymbol{y}$ given as,

$$\bar{y} = \frac{1}{n}\sum_{i=0}^{n-1} y_i.$$

D. Bias-Variance Trade-off Analysis

Let us derive the bias-variance trade off for OLS. We assume that we have a dataset $\mathcal{L}$ consisting of the data $\mathbf{X}_{\mathcal{L}} = \{(y_j, \mathbf{x}_j),\, j = 0,\dots,n-1\}$. We also assume that the true data is generated by a noisy model $\mathbf{y} = \mathbf{f}(\mathbf{x}) + \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(0,\sigma^2)$. The cost function that we want to minimize is the MSE. We now want to show that the MSE can be decomposed into the bias and variance of the model. First lets look at the definition of the MSE:

$$\begin{aligned}
\mathbf{C}(\mathbf{X},\boldsymbol{\beta}) &= \mathbb{E}[(\mathbf{y}-\tilde{y})^2] \\
&= \mathbb{E}[(\mathbf{f}(\mathbf{x})+\boldsymbol{\epsilon}-\tilde{y})^2] \\
&= \mathbb{E}[(\mathbf{f}(\mathbf{x})-\tilde{y})^2] + \mathbb{E}[\boldsymbol{\epsilon}^2] + 2\,\mathbb{E}[\boldsymbol{\epsilon}(\mathbf{f}(\mathbf{x})-\tilde{y})] \\
&= \mathbb{E}[(\mathbf{f}(\mathbf{x})-\tilde{y})^2] + \sigma^2.
\end{aligned}$$

Here we have used that $\mathbb{E}[\boldsymbol{\epsilon}] = 0$. Now, lets look at the definitions for the bias and variance of the model:

$$\begin{aligned}
\mathrm{Bias}(\tilde{y}) &= \mathbb{E}[(\mathbf{y}-\mathbb{E}[\tilde{y}])^2] \\
&= \mathbb{E}[(\mathbf{f}(\mathbf{x})+\boldsymbol{\epsilon}-\mathbb{E}[\tilde{y}])^2] \\
&= \mathbb{E}[\mathbf{f}(\mathbf{x})^2 + 2\mathbf{f}(\mathbf{x})\boldsymbol{\epsilon} - 2\,\mathbb{E}[\tilde{y}]\mathbf{f}(\mathbf{x}) \\
&\quad + \boldsymbol{\epsilon}^2 - 2\boldsymbol{\epsilon}\,\mathbb{E}[\tilde{y}] + \mathbb{E}[\tilde{y}]^2] \\
&= \mathbb{E}[\mathbf{f}(\mathbf{x})^2] - 2\,\mathbb{E}[\mathbf{f}(\mathbf{x})]\,\mathbb{E}[\tilde{y}] + \mathbb{E}[\tilde{y}]^2 + \mathbb{E}[\boldsymbol{\epsilon}^2] \\
&= \mathbb{E}[\mathbf{f}(\mathbf{x})^2] - 2\,\mathbb{E}[\mathbf{f}(\mathbf{x})]\,\mathbb{E}[\tilde{y}] + \mathbb{E}[\tilde{y}]^2 + \sigma^2, \\
&= \mathbb{E}[(\mathbf{f}(\mathbf{x})-\mathbb{E}[\tilde{y}])^2] + \sigma^2.
\end{aligned}$$

$$\mathrm{Var}(\tilde{y}) = \mathbb{E}[(\tilde{y}-\mathbb{E}[\tilde{y}])^2].$$

Now we have everything we need to rewrite the MSE in terms of the bias and variance of the model:

$$\begin{aligned}
\mathbf{C}(\mathbf{X},\boldsymbol{\beta}) &= \mathbb{E}[(\mathbf{f}(\mathbf{x})-\tilde{y})^2] + \sigma^2 \\
&= \mathbb{E}[(\mathbf{f}(\mathbf{x})-\mathbb{E}[\tilde{y}]+\mathbb{E}[\tilde{y}]-\tilde{y})^2] + \sigma^2 \\
&= \mathbb{E}[(\mathbf{f}(\mathbf{x})-\mathbb{E}[\tilde{y}])^2] + \mathbb{E}[(\mathbb{E}[\tilde{y}]-\tilde{y})^2] \\
&\quad + 2\,\mathbb{E}[(\mathbf{f}(\mathbf{x})-\mathbb{E}[\tilde{y}])(\mathbb{E}[\tilde{y}]-\tilde{y})] + \sigma^2 \\
&= \mathbb{E}[(\mathbf{f}(\mathbf{x})-\mathbb{E}[\tilde{y}])^2] + \mathbb{E}[(\mathbb{E}[\tilde{y}]-\tilde{y})^2] + \sigma^2 \\
&\quad + 2(\mathbf{f}(\mathbf{x})\,\mathbb{E}[\tilde{y}] - \mathbf{f}(\mathbf{x})\tilde{y} - \mathbb{E}[\tilde{y}]^2 + \mathbb{E}[\tilde{y}]\tilde{y}) \\
&= \mathrm{Bias}(\tilde{y}) + \mathrm{Var}(\tilde{y}).
\end{aligned}$$

Here, we have used the fact that $\tilde{y}$ is non-stochastic such that $\mathbb{E}[\tilde{y}] = \tilde{y}$ to show that the term on the second to last line is zero. Notice that the σ^2 term dissapears when we write out the Bias term as a function of $\mathbf{y}$.

The bias term is the error introduced by the assumptions we make when we create the model. This term is often large when we try to fit a polynomial with a low degree to a complex dataset. On the other hand, the variance is a measure of how spread out from the mean our predictions are. The variance is often high if the data is overfitted. Thus overfitting will lead to low bias and high variance, while underfitting will lead to high bias and low variance. The bias-variance tradeoff describes the balance between the two terms.

E. Splitting the data

To be able to figure out whether the model has been able to create a more general solution, we split the data set into two subsets called training and test data. The training data is the data our model uses to calculate the estimated parameters, while the testing data is strictly used to test the performance of the model. In this report we have decided to put 80% of the data for training and 20% of the data for testing.

F. Resampling Techniques

For all three regression methods, $\hat{\boldsymbol{\beta}}$ is a function of the random variable $\mathbf{X}$ and must therefore also be a random variable with a Probability Density Function (PDF) $p(\mathbf{x})$. The core idea behind resampling techniques is to try to estimate $p(\mathbf{x})$ by repeatedly drawing samples from the existing data and use the statistics of $p(\mathbf{x})$ to estimate metrics such as bias, variance and MSE [10]. In fact, if $\tilde{p}(\mathbf{x})$ is the estimate of $p(\mathbf{x})$ using m independent and identically distributed samples, and $p(\mathbf{x})$ has mean μ and standard deviation σ. Then the Central Limit Theorem (CLT) tells us that $\tilde{p}(\mathbf{x})$ will have mean $\mu_m = \mu$ and standard deviation

$$\sigma_m = \frac{\sigma}{\sqrt{m}}.$$

It is important to note that since we have assumed that the samples x_i are independent and identically dis-

tributed, then $\tilde{p}(\mathbf{x})$ will approach a normal distribution [10].

1. Bootstrapping

The idea behind the bootstrapping algorithm is to estimate the PDF $\tilde{p}(x)$ by repeatedly resampling the original dataset with replacement. A single bootstrap refers to resampling n samples from the original dataset to generate a sample estimator $\hat{\boldsymbol{\beta}}^*$. This process is typically repeated many times. The procedure for each bootstrap is as follows:

1. Draw with replacement n samples from $\mathbf{x}$ to create a new vector $\mathbf{x}^*$.
2. Using $\mathbf{x}^*$, generate a sample estimator $\hat{\boldsymbol{\beta}}^*$ by evaluating $\hat{\boldsymbol{\beta}}$ over $\mathbf{x}^*$.

Repeating this procedure k times, you end up with k estimators $\hat{\boldsymbol{\beta}}^*$ that you can use to calculate the statistical properties [10].

2. K-fold Cross Validation

K-fold Cross Validation is another commonly used resampling technique, but now without replacement. The training data is randomly split into k sets, or folds. $k-1$ folds are then used for training, whilst one is kept as a test set to asses model performance, in this case using the MSE. This process is repeated k times as shown in Figure 1 for $k = 10$. The average value of these performance estimates is then used as the final result[1].

FIG. 1. Figure 6.3 from Raschka *et al.* [1] showing how k-fold cross-validation works for 10 folds.

G. Hyperparameter tuning

Within ML it is typical to distinguish between two main types of parameters. There are the ones that the model learns from training data and the parameters related to the learning algorithm. The latter are referred to as hyperparamteres and need to be tuned to achieve an optimal model. In the case of Ridge and Lasso regression this is the regularization parameter λ. It's value dictates the amount of regularization applied to the model. In this case we also treat the complexity of the model as a tunable parameter for all three methods. This tuning can be done by eye-balling validation curves or more exhaustively with the grid search approach [1].

1. Grid Search

As discussed in Section 11.4.3 of Goodfellow *et al.* [11], the grid search algorithm is typically used in cases of three or fewer parameters, mainly due to it's computational cost. A finite selection of values is chosen for each hyperparamter, here polynomial degree and regularization parameter λ. The algorithm subsequently trains a model for every combination of hyperparamters. The combination that results in the smallest test error is then chosen as the optimal combination. The MSE is found using k-fold cross-validation. This process is repeated several times to find a suitable domain of hyperparameters and a suitable resolution once a promising region is found.

H. Franke Function

We will apply the regression models on two different datasets. The first one is data generated with the Franke function shown in Figure 2, which is given by:

$$\begin{aligned} f(x,y) = &\frac{3}{4}\exp\left(-\frac{(9x-2)^2}{4} - \frac{(9y-2)^2}{4}\right) \\ &+ \frac{3}{4}\exp\left(-\frac{(9x+1)^2}{49} - \frac{(9y+1)}{10}\right) \\ &+ \frac{1}{2}\exp\left(-\frac{(9x-7)^2}{4} - \frac{(9y-3)^2}{4}\right) \\ &- \frac{1}{5}\exp\left(-(9x-4)^2 - (9y-7)^2\right). \end{aligned} \tag{6}$$

This is a weighted sum of exponentials that make up two Gaussian peaks. Note that this function can not be represented by a finite degree polynomial. Thus, it is a great function to test linear regression methods. We will evaluate the function for $x, y \in [0, 1]$.

I. Application on Real Data

After having tested the code analyzing the Franke function, we perform a further analysis on real terrain data. An example of the terrain used is shown in Figure 3, which corresponds to an area close to Stavanger, Norway.

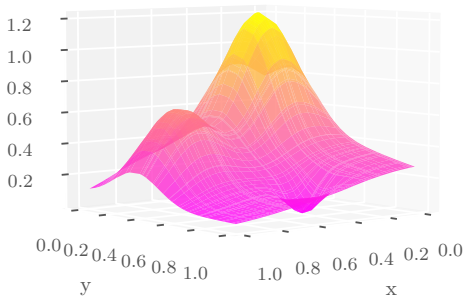


FIG. 2. Surface plot of the Franke function 6.

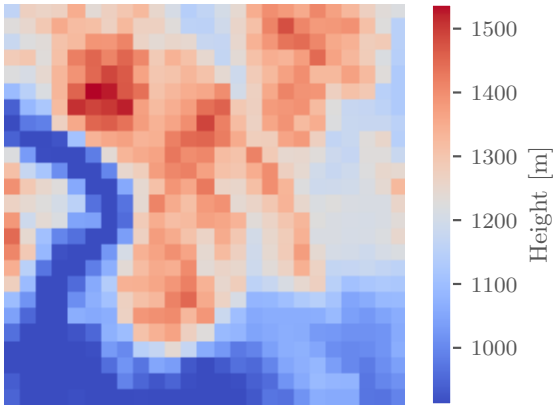


FIG. 3. Terrain data from a region close to Stavanger, Norway. The resolution of the data is 25 by 25 pixels, resulting in 625 data points.

J. Scaling

It is important to scale our data properly before we test each algorithm. It might seem that scaling is not important on our datasets because all the features have the same dimension. However, both Ridge and Lasso regression include regularisation terms that can be impacted by how the data is scaled. Especially, if the intercept is high the regularisation term will affect the cost function of Lasso and Ridge because the corresponding beta value has to be high. In addition, scaling is key for standardisation, making it easier to compare the results we get for each of the different datasets.

For these reasons, we have decided to scale all our data in two ways. First we make sure that the data is centered around zero to remove the intercept and secondly we scale the data to unit variance. These two operations are implemented using the StandardScaler() function in scikit-learn [12].

K. Multiple Regression

Both the Franke function (Equation 6) and the terrain data from Figure 3 exist in 3D space. This means that our function $f(\mathbf{x}) = \mathbf{z}$ is a function of the two variables x and y. And it means that we have to cleverly create the design matrix so that it can handle all the possible points on the surface. We do this by putting all the points on the x-axis and y-axis in two vectors $\mathbf{x}$ and $\mathbf{y}$ and let the design matrix consist of all the different combinations of $\mathbf{x}$ and $\mathbf{y}$ up to the degree n of the polynomial we are fitting. thus the design matrix becomes

$$\mathbf{X} = \begin{bmatrix} \mathbf{1} & \mathbf{x} & \mathbf{y} & \mathbf{xy} & \mathbf{x}^2 & \mathbf{y}^2 \cdots \end{bmatrix}$$

L. Tools

The code is written in Python using standard packages in addition to functions from `scikit-learn` [12]. All plots have been made using the `matplotlib` [13] package in python.

III. RESULTS

A. The Franke Function

In the following section we will present the results of fitting OLS, Ridge, and Lasso regression models to the Franke function (see Figure 2). First off all Figure 4 shows how test and train MSE behave for increasing OLS model complexity. The MSE is always greater for test than train data. Both initially drop as the complexity increases but then start to diverge for higher polynomial orders, with the train MSE continuing to decrease and the test MSE increasing. Figure 5 shows The MSE to the left and R2 score to the right, for OLS, Ridge and Lasso. In Figure 5(a) the MSE for OLS is shown to decrease as the complexity of the model increases and the R2 score in Figure 5(b) grows closer to 1 for higher complexity. Figures 5(c) through 5(f) show how the Ridge and Lasso MSE and R2 score change depending on the regularization parameter λ for a fixed polynomial degree of five. For both methods, the MSE increases and R2 score decreases for growing values of λ. Generally Ridge and Lasso show higher MSE and lower R2 score than OLS.

Figure 6(a) shows how the magnitude and spread of β_i values in $\boldsymbol{\beta}$ increases with model complexity for OLS. β_i values in the case of a fifth order polynomial and varying λ are shown for Ridge and Lasso in Figure 6(b) and Figure 6(c) respectively. In both cases the spread and largest β_i value decreases as λ becomes larger. This behaviour is most pronounced for Lasso regression.

The bias-variance trade-off analysis for OLS is shown in Figure 7. Here Figure 7(a) depicts how MSE, bias and

TABLE I. Optimal model parameters for the Franke function.

Model	**Optimal degree**	**Optimal λ**
OLS	8	NA
Ridge	15	10^{-6}
Lasso	10	1.58×10^{-5}

variance change with model complexity. The MSE is at a minimum, and bias and variance intersect at degree 8. Dependence on the number of points used is studied in Figure 7(b). MSE, bias and variance are all high for few points and drop off to lower values for 20 to 45 points.

Figure 8 compares MSE values given by k-fold cross-validation and the bootstrap method for all three regression methods. Most notably, in Figure 8(a), the MSE for OLS is at its lowest at polynomial degree 8 using the Bootstrap and then increases again. The MSE using the k-fold resampling method on the other hand shows less variability and seems to stabilise for polynomial values 9 to 15. Using a regularization parameter $\lambda = 0.01$ for both Ridge and Lasso regression results in similar behaviour for the k-fold method, although the MSE values stabilise at higher values with Lasso clearly performing worst. For Ridge (see Figure 8(b)) the Bootstrap and k-fold methods perform similarly, whereas there is notably more variability using the Bootstrap method for Lasso regression and a low point at polynomial degree 8. Overall, k-fold resampling gives a smoother curve.

To find an optimal combination of polynomial degree and regularization parameter λ, grid searches in Figure 9 are performed for Ridge and Lasso regression. Both figures mark the optimal combination with a red cross in the parameter space. For Ridge regression in Figure 9(a), the optimal combination of degree 15 and $\lambda = 10^{-6}$ is located in the bottom right corner. Although the optimal combination of degree 10 and $\lambda = 1.58 \cdot 10^{-5}$ is slightly more central in the shown parameter space for Lasso in Figure 9(b), both figures show the same tendency of lower MSE towards the lower right corner, i.e. higher degree and lower λ.

Lastly for the Franke function, Figure 10 shows the actual function (10(a)) compared to the predictions by OLS, Ridge and Lasso. Figure 10(b), 10(c) and 10(d) show the OLS, Ridge and Lasso surface respectively. These use the ideal parameters shown in Table I and include the data points they are tested against in blue and the resulting MSE in Table II. Here, we see that the MSE is lowest for OLS and highest for Lasso. Intuitively the surface looks most smoothed in the Lasso case and aligns best with the test points in the OLS case.

B. Terrain Data

We will now present results from a similar analysis to the previous section, but here applied to real digital terrain data shown in Figure 3. A bias-variance analysis

TABLE II. MSE values for all three regression methods applied to scaled Franke fucntion data.

Model	**Scaled MSE**
OLS	3.7×10^{-4}
Ridge	4.71×10^{-4}
Lasso	7.31×10^{-3}

TABLE III. Optimal model parameters for real digital terrain data.

Model	**Optimal degree**	**Optimal λ**
OLS	8	NA
Ridge	15	10^{-6}
Lasso	15	10^{-5}

for the OLS method is presented in Figure 11, both for polynomial degree and the number of data points used. Figure 11(a) shows the lowest MSE to be achieved for a polynomial degree of 8 where the bias and variance intersect. A general tendency for MSE, bias and variance to decrease as the number of points increases is shown in Figure 11(b).

Figure 12(a) presents results of a parameter grid search for Ridge regression resulting in a optimal model using a 15th degree polynomial and $\lambda = 10^{-6}$. This is represented by a red cross in the bottom right corner. Equivalently, Figure 12(b) shows the results of a parameter grid search for Lasso regression with optimal parameters being, 15th degree polynomial and $\lambda = 10^{-6}$.

Using the optimal parameters shown in table Table III, we predict the digital terrain data using all three regression methods. Figure 13 shows the final results both with and without scaling. Figures of the true and scaled terrain are shown Figures 13(h) and 13(g) for visual comparison. Values presented in Table IV show very poor performance from Lasso regression, with much larger MSE values compared to OLS and Ridge regression in the scaled and unscaled case. This can also be seen clearly in Figures 13(c) and 13(f), where Lasso has overly smoothed out the elevation gradients. Visually, results for OLS and Ridge regression are remarkably similar, but the MSE in Table IV reveals that OLS performs best on the unscaled data, whereas Ridge achieves the lowest MSE when the data is scaled. Further figures of the terrain data analysis can be found in Appendix A.

TABLE IV. MSE values for all three regression methods applied to both unscaled and scaled terrain data.

Model	**Unscaled MSE**	**Scaled MSE**
OLS	5253.97	0.19
Ridge	5358.48	0.16
Lasso	12327.53	0.45

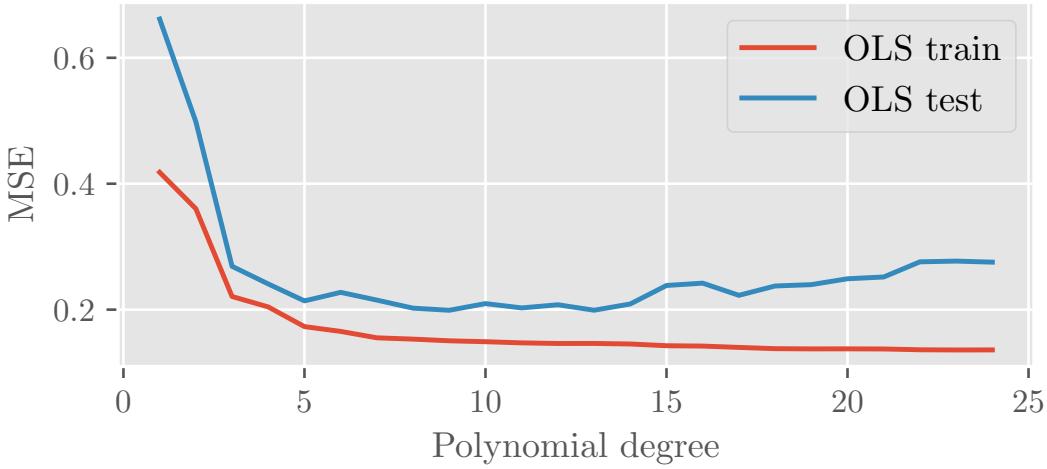


FIG. 4. A comparison of the performance of OLS on the training and test datasets applied to the Franke function 6 using $N = 625$ data points and an added i.i.d. noise with variance $\sigma^2 = 0.1$.

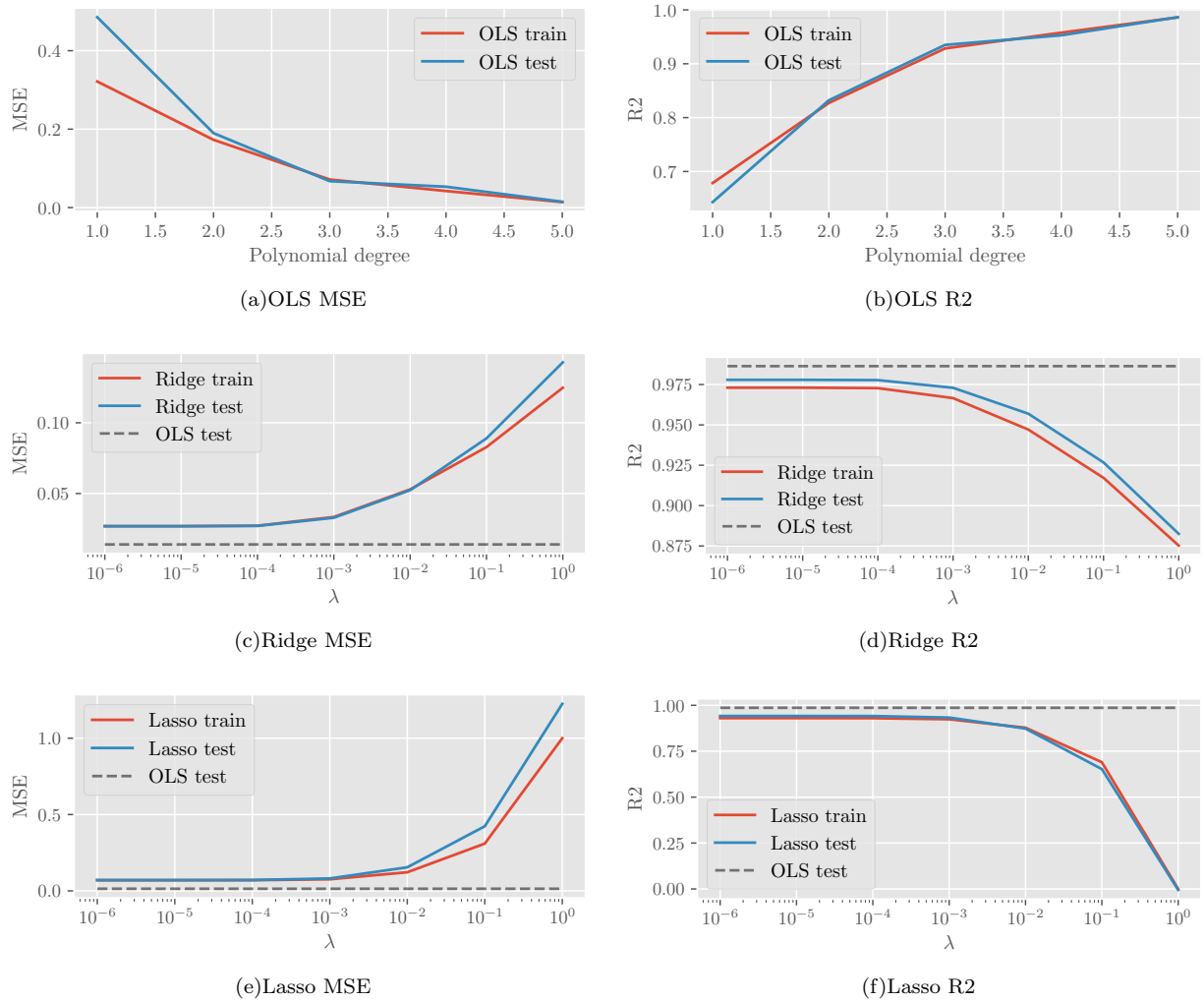
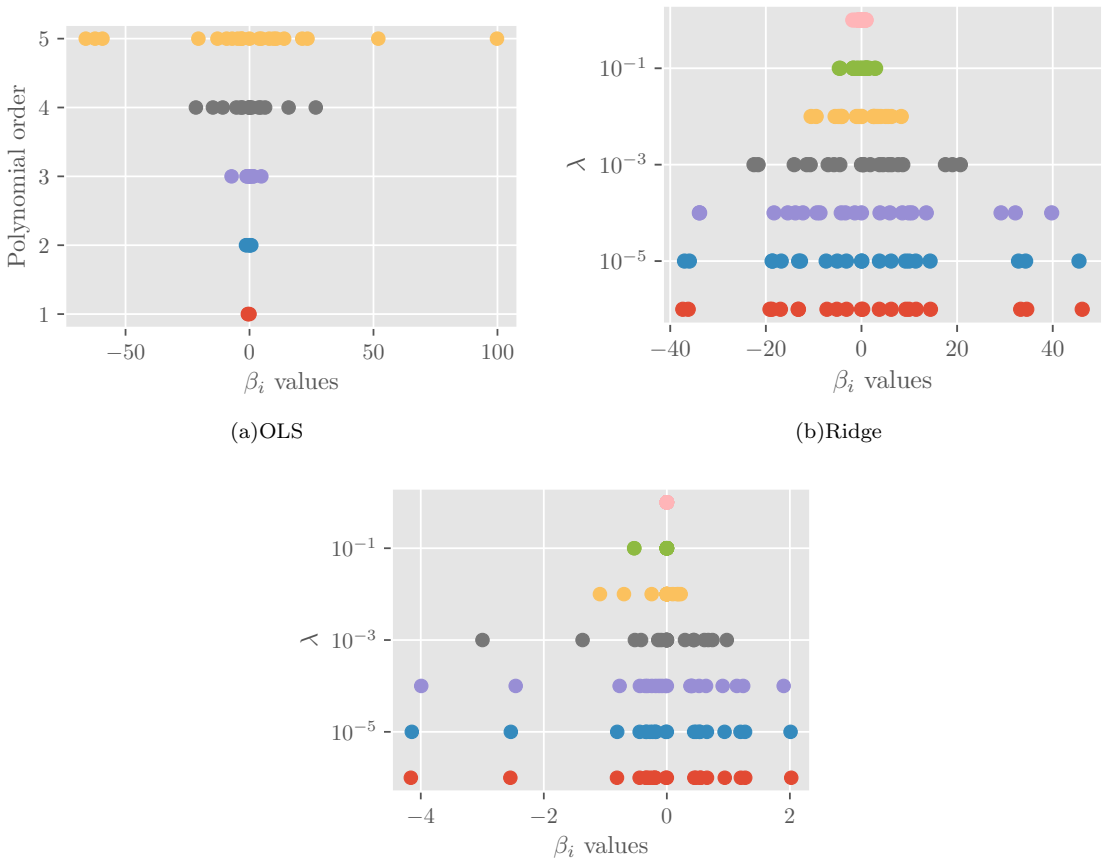


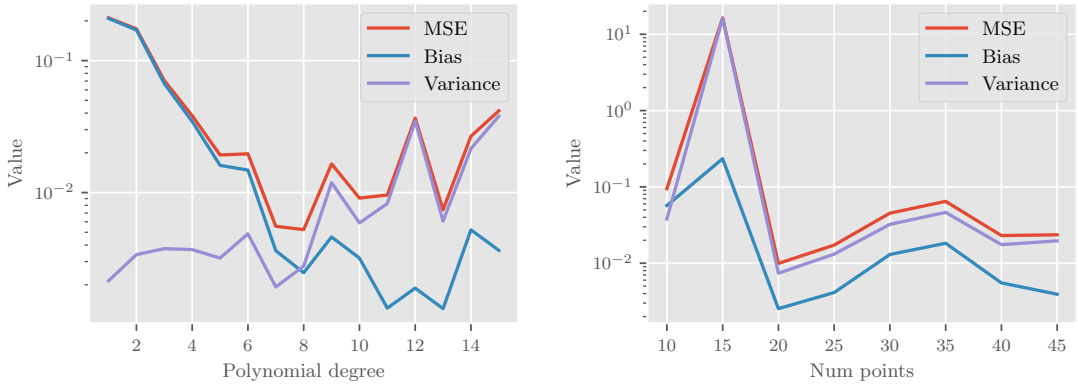
FIG. 5. A comparison of the MSE and R2 scores for OLS, Lasso and Ridge regression applied to the Franke function 6. All plots use $N = 25^2$ data points and a added i. i. d. noise with variance $\sigma^2 = 0.01$. For Ridge and lasso we have used a fifth order polynomial.



(a)OLS (b)Ridge

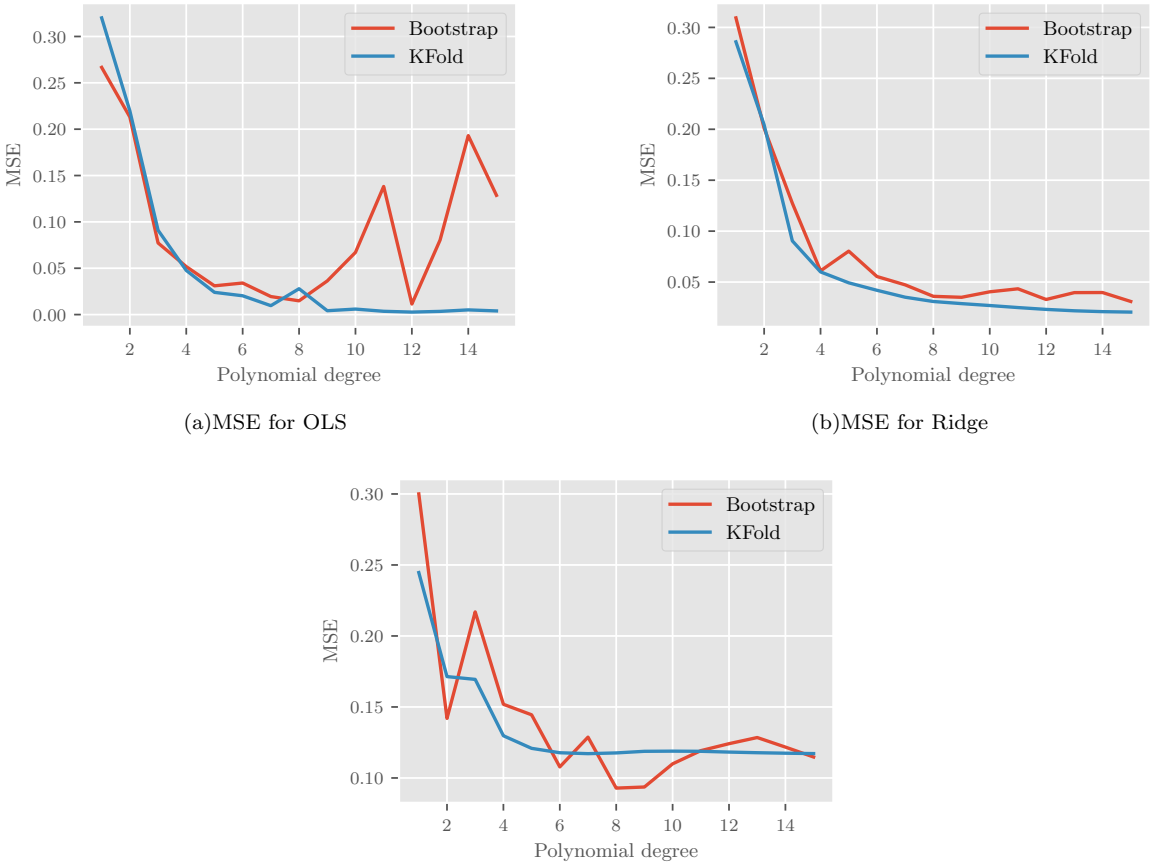
(c)Lasso

FIG. 6. Polynomial weight β_i values in $\boldsymbol{\beta}$ for OLS, Ridge and Lasso regression on the Franke function 6 are displayed for different values of λ or polynomial degree. Each color corresponds to a λ value. All plots use $N = 25^2$ data points and a added i. i. d. noise with variance $\sigma^2 = 0.01$. Ridge and Lasso are run using a fifth order polynomial.



(a)Polynomial degree (b)Number of datapoints

FIG. 7. Bias-variance trade-off analysis of the Ordinary Least Squares (OLS) on the Franke function. For Figure 7(a), an increasing number of polynomial degrees is used with 625 datapoints ($N = 25^2$), and for Figure 7(b), the analysis is done with an increasing number of datapoints for a polynomial of degree 5. In both cases, the i.i.d. noise has variance $\sigma^2 = 0.01$, and 100 bootstraps with 200 samples each are applied.

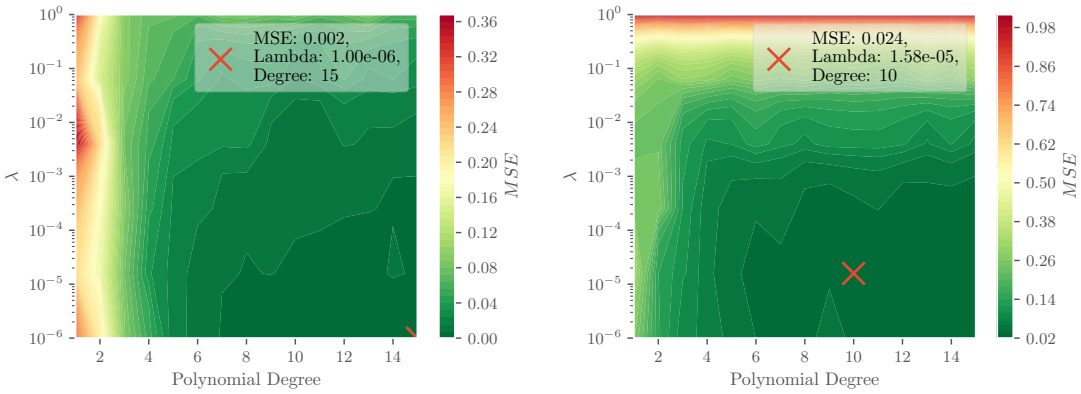


(a)MSE for OLS

(b)MSE for Ridge

(c)MSE for Lasso

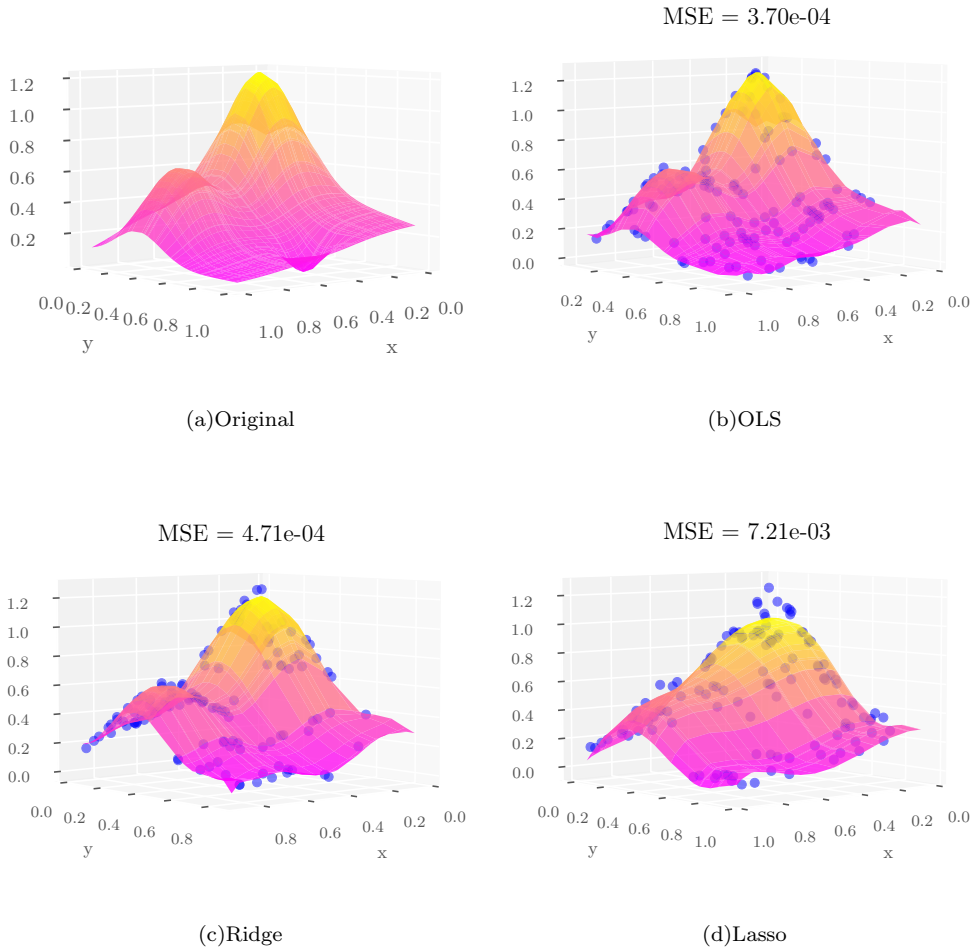
FIG. 8. A comparison of the MSE calculated using bootstrapping and k-fold cross-validation on the Franke function (Equation 6) for increasing polynomial degrees. 100 Bootstraps with 100 samples are performed and 10 k-folds. Here the i.i.d. noise has variance $\sigma^2 = 0.01$ and both Ridge an Lasso regression use a $\lambda = 0.01$.



(a)Ridge

(b)Lasso

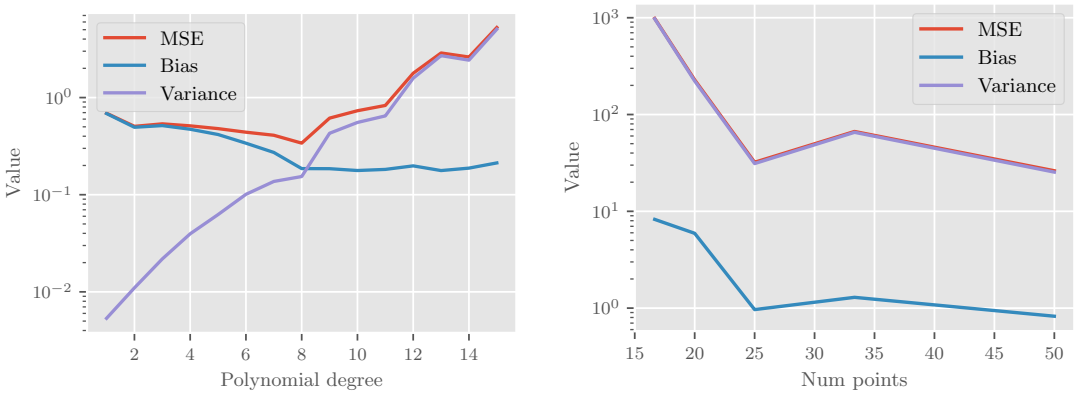
FIG. 9. Grid search over complexity and λ-values for Ridge and Lasso regression on the Franke function 6 using k-fold with 10 folds. We have used $N = 25^2$ data points and noise with variance $\sigma^2 = 0.01$. The best parameters are marked with a red cross.



(a)Original (b)OLS

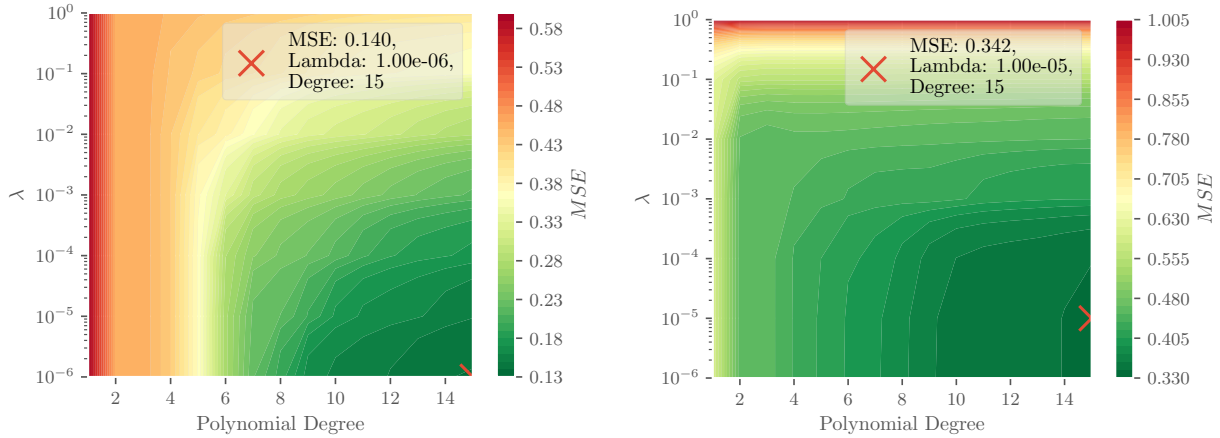
(c)Ridge (d)Lasso

FIG. 10. A comparison of the surfaces predicted by OLS, Ridge and Lasso on the Franke function 6. There are $N = 25^2$ data points and a added i.i.d. noise with variance $\sigma^2 = 0.01$ on all figures. The ground truth from the test data is shown with blue scatter points, while the surface shows the prediction. OLS use an eight order polynomial, Ridge uses a 15th order polynomial and Lasso uses a 10th order polynomial. Ridge uses $\lambda = 1 \cdot 10^{-6}$ and Lasso uses $\lambda = 1.58 \cdot 10^{-5}$. Note that we have not used scaling for this example to better illustrate the original shape of the function.



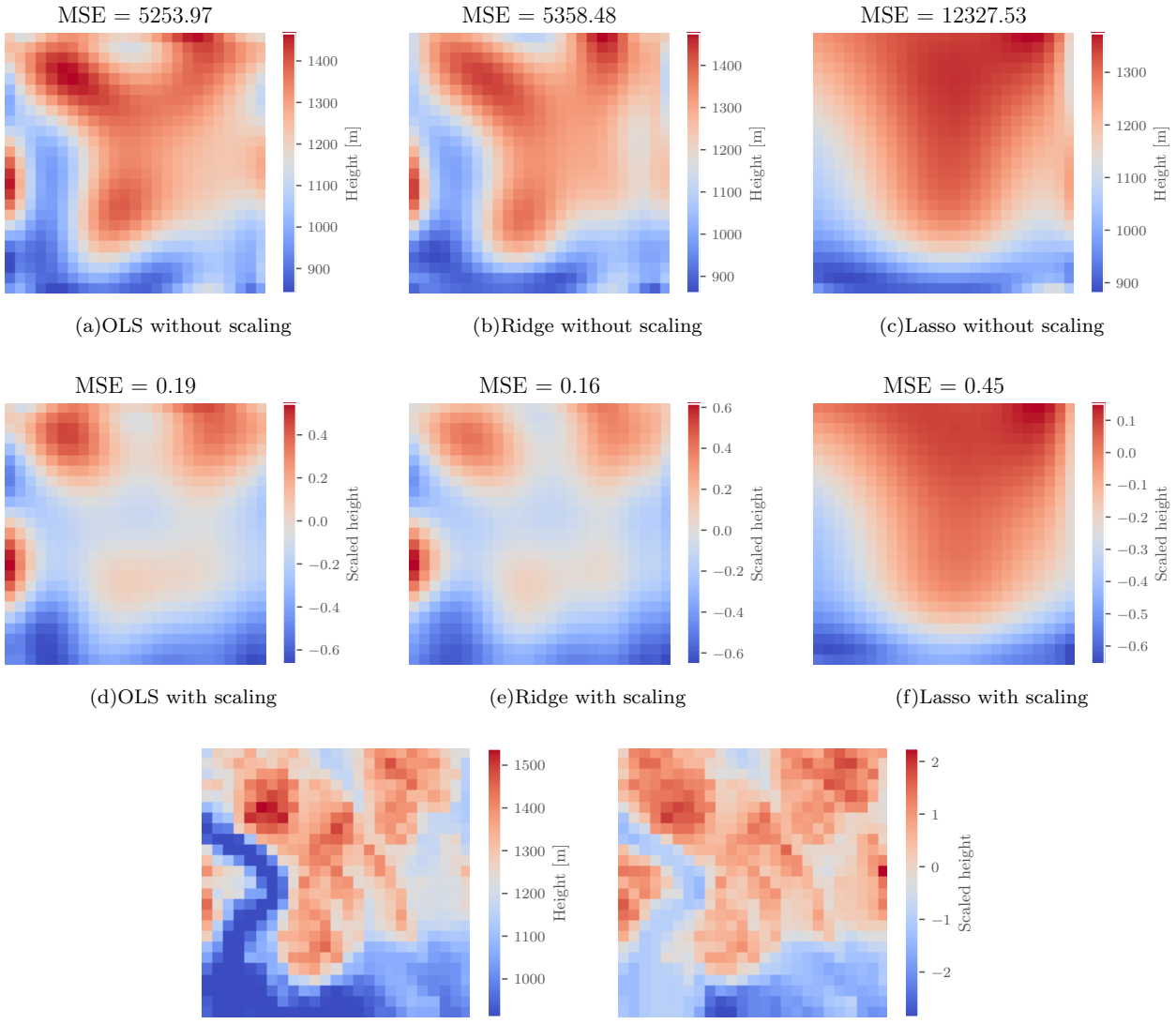
(a)Polynomial degree (b)Number of data points

FIG. 11. A bias-variance trade off analysis of the influence of the polynomial degree and number of points used for terrain data. For figure 11(a) we have used a 500x500 section of the original terrain data with a down sampling factor of 20, resulting in $N = 25^2 = 625$ data points, and for for figure 11(b) we have used order 10 polynomials. The number of points in figure 11(b) refers to the resolution in the x and y directions of a 500x500 area of the terrain. Both results are generated using 100 bootstraps with 200 samples each.



(a)Ridge (b)Lasso

FIG. 12. Grid search over complexity and λ-values for Ridge and Lasso regression on the terrain data from figure 3 using k-fold with 10 folds. The best parameters are marked with a red cross.



(a)OLS without scaling (b)Ridge without scaling (c)Lasso without scaling

(d)OLS with scaling (e)Ridge with scaling (f)Lasso with scaling

(g)Terrain without scaling (h)Terrain with scaling

FIG. 13. A comparison of the different methods final performance using the parameters found using bias-variance trade off and grid-search. The results are shown with and without scaling (removing intercept and scaling to unit variance). For OLS we fit a polynomial with degree 8. While Ridge and Lasso both fit a polynomial with degree 15. Ridge uses $\lambda = 1 \cdot 10^{-6}$ and Lasso uses $\lambda = 1 \cdot 10^{-5}$. The resolution used is 25x25 pixels resulting in 625 data points.

IV. DISCUSSION

We will discuss the performance of the three different regression methods OLS, Ridge and Lasso on both the synthetic data from the Franke function (Equation 6) and the actual terrain data shown in Figure 3. The performance is assessed using the MSE and R^2-score metrics. Note that all results use centered data scaled to unit variance unless specified otherwise.

Initial Comparison on the Franke Function

Our initial impression is that OLS will outperform both Ridge and Lasso on the Franke function. In Figure 5, we compare the performance of OLS for polynomial degrees from one to five, and different λ values for Ridge and Lasso. It is clear from the figure that OLS performs the best and that Ridge and Lasso prefer a smaller λ value that mitigates the regularisation as much as possible. It is also clear that although the λ values seem to go low enough, we need to run the models for higher polynomial degrees. This is expected, as the Franke function is a sum of exponentials that cannot be approximated exactly as a polynomial. However, since we have added noise $\epsilon \sim \mathcal{N}(0, 0.01^2)$, we can expect a value where higher complexity leads to larger MSE as we begin overfitting to the noise. Overfitting behaviour for OLS can clearly be seen in Figure 4, where the test MSE initially decreases, but then increases as the polynomial order becomes too high. This aligns well with our expectations and the idealised example shown in Figure 2.11 in Hastie *et al.* [14]. For higher complexity we expect Ridge and Lasso regression to perform better as a suitable regularization parameter λ should dampen such behaviour.

We also plot the values of the parameters $\hat{\boldsymbol{\beta}}$ for each of the complexities and λ values in Figure 6. This shows three different trends: 1) As we increase the polynomial order, the spread of the parameter values increases. 2) As λ increases, the spread of the parameter values decreases for a fixed polynomial order. 3) The variance of the parameters obtained with OLS is larger than the parameters obtained with Ridge and Lasso using the same (fifth) order polynomial. Furthermore, we observe much lower $\hat{\boldsymbol{\beta}}$ values for Lasso This is expected since the ℓ^1 norm in the Lasso cost function encourages small $\hat{\boldsymbol{\beta}}$ coefficients more than the ℓ^2 norm in Ridge. An equivalent analysis of the terrain data can be found in Appendix A.

Bias-Variance Trade-off

We have shown in Section II D that we can express the MSE of our methods as the sum of the bias and variance of their predictions. We have done an investigation of the bias-variance trade off for OLS on both datasets, looking at how the complexity and number of points in the dataset impacts the bias, variance and MSE.

For the Franke function, the results are shown in Figure 7. Here, Figure 7(a) shows that as we increase the complexity of the model the bias tends to decrease while the variance increases. It is then natural that the two have to intersect at some point, and that this point represents a good trade off between them. We see that when we use an eight order polynomial we have both an intersection between the bias and variance, as well as the minimum MSE. The consequence of using higher order polynomials would most likely lead to overfitting where the model starts to model the noise instead of the underlying function. Based on this analysis, we choose an eight order polynomial when generating the final results. Another pattern can be observed in Figure 7(b). The variance and bias behave nicely as long as we choose a large enough number of data points. For a small number of points i.e. $N = 15^2$ we see that the variance explodes. On the other hand, it is important to not choose too many points as this would be too computationally expensive. Therefore, we have chosen a middle ground of $N = 25^2 = 625$ to generate all other results.

We observe similar trends when we apply OLS to the terrain data in Figure 11(a) and 11(b). The optimal polynomial degree for the complexity is also eight at the intersection between the bias and variance. In addition, as shown in Figure 11(b), we see a decrease in bias and variance as we increase the number of data points. In contrast to the results on the Franke function, the MSE is now dominated by the variance - which is more than one magnitude larger than the bias. It is clear that we should choose a resolution of at least 25 data points in each direction such that the MSE is not too high, but after this point the bias and variance seem to stabilise somewhat. To make sure that we have good balance between bias, variance and computational efficiency we choose a resolution of 25x25 data points in all experiments.

Resampling Techniques: Bootstrapping vs K-Fold Cross-Validation

In addition to using bootstrapping for bias-variance trade off and k-fold for grid searching, we do a comparison of how they perform applied to the three regression methods. The idea behind the resampling methods are, as discussed, that they simulate a larger dataset by approximating the underlying PDF of the data. Therefore, we can expect the MSE to behave in a way that would suggest more data samples in both training and test - decreasing for increased model complexity.

In Figure 8 we show how the two resampling techniques behave on the Franke function as we increase the complexity of the models (fixed $\lambda = 0.01$). We see from the figures that in general, the k-fold technique gives a smoother curve that tends to zero as we increase the complexity. The bootstrap technique seems to show more erratic behavior, but has the same trend to zero for both

Ridge and Lasso. Bootstrapping with OLS tends to increase the MSE when we increase the complexity beyond an eight order polynomial. This is expected as it is the same we saw in the bias-variance trade off (Figure 7). The smoother results when using k-fold as a resampling technique are likely due to the fact that k-fold generalizes better since it changes the test set for each fold. This motivates its further use when performing grid searches.

Finding Optimal Hyperparameters

We found the optimal complexity of OLS by studying the bias-variance trade off for different model complexities. One could do the same for Ridge and Lasso, but these have an additional parameter λ that affects their performance. Therefore we use a grid search to find the optimal combination of complexity and λ.

For the Franke function, the results from the grid search are shown in Figure 9. Figure 9(a) shows that the best combination of parameters are found in the bottom right corner where $\lambda = 1 \times 10^{-6}$ and the polynomial order is 15. However, we see that the model is much more sensitive to the complexity than the value of λ. Grid search results for Lasso regression in Figure 9(b) gives $\lambda = 1.58 \times 10^{-5}$ and a polynomial order of 10. This result is similar to Ridge, where a low value for λ and a relatively high polynomial order was chosen. This time, however, we see that the model is more sensitive to the λ value than to the complexity.

Similarly, the hyperparameters for each method when applied to the real digital terrain data are found in Figure 12. Both methods have the lowest MSE with a polynomial order of 15. For Ridge the hyperparameter is $\lambda = 1 \times 10^{-6}$ and for Lasso it is $\lambda = 1 \times 10^{-5}$. We observe generally the same tendencies as for the Franke function, but with the MSE being at least one order of magnitude higher. This likely reflects the higher variance in the terrain data, which is expected. This difference could decrease if the stochastic noise added to the Franke function is increased.

The Impact of Scaling

As mentioned in the method Section II, scaling can greatly affect the performance of the different models. We have chosen to augment the data in two different ways for both the Franke function and the Terrain data. First we center the data by removing the intercept and then we scale the data to have unit variance. In the Franke case, it is not strictly necessary to scale the data, since it is already defined in a domain close to zero with no obvious outliers. We have nonetheless chosen to scale the data as we view this as a good convention to allow easier comparison between different datasets. When it comes to the terrain data, there is a greater motivation for scaling since the original values are of a much greater magnitude and may contain more outliers. Here scaling allows for a more direct comparison between the Franke function and terrain data as well as taking advantage of the widely accepted fact that ML algorithms perform better on scaled data, especially when using regularization techniques [14].

In the final results for the terrain data shown in Figure 13 we have shown the resulting polynomial landscape as a heatmap for both scaled and unscaled data. An interesting aspect is that the best performing model changes from OLS to Ridge when we scale the data. There are most likely two reasons for this. The main reason is that, as we see on the figure, the terrain data varies between around 1000 and 1500 meters in height. This means that the intercept is large, which would naturally give Ridge a disadvantage because of the regularisation term that would penalize these more. The other reason is that as we scale the data to unit variance, the landscape naturally varies less and has less defined peaks. It is, however, important to note that Ridge needed a higher polynomial degree (15) than OLS (8) to achieve this result.

Final Evaluation of Regression Methods

Using the best hyperparameters found by grid search gives MSE values presented in Table II and Table IV for the Franke function and terrain data respectively.

It should be noted that future studies may want to consider searching a larger domain with as fine a resolution as computationally feasible. This is a definitive weakness of the presented analysis, where we only consider complexities up to a polynomial degree of 15.

The prediction of the Franke frunction is only considered for scaled data in Figure 10 as we do not expect there to be a large effect of scaling in this case. Here OLS performs best using a polynomial degree of 8 and achieving an MSE of 3.7×10^{-4}. Studying the difference between the displayed surfaces in Figure 10 gives some intuition for why this is the case. One clearly sees that the predicted surface aligns well with the test points in the OLS case. For Ridge and Lasso this becomes progressively worse, due to a smoothing of the surface. This is linked to the regularization parameter λ, which as expected, is most pronounced in the Lasso case.

As previously mentioned, scaling is expected to, and does, have a larger influence on the successful fitting of the terrain data. This should be most pronounced for the two regularization methods. We cannot assess this for Lasso as it is by far the worst with an MSE of 0.45 in the scaled case, and 12327.53 in the unscaled case. This MSE is more than twice as large as for OLS and Ridge in both cases. The Lasso regularization aggressively smooths the prediction (see Figure 13) as already seen for the Franke function and thus does not account for the variability in the terrain data.

Without scaling, OLS performs best using a polyno-

mial degree of 8 and achieving an MSE of 5253.97. As discussed, scaling allows Ridge to perform better achieving an MSE of 0.16 using a polynomial degree of 15 and $\lambda = 10^{-6}$. Here we see how the regularization allows Ridge to achieve a better prediction using a higher complexity whilst avoiding overfitting which would occur for OLS. As mentioned a broader parameter search may reveal even better parameters.

V. CONCLUSION

Based on an analysis of OLS, ridge and Lasso regression we have thus concluded that OLS performs the best fit on the Franke function with an MSE of 3.7×10^{-4} using polynomial degree 8. For the real terrain data OLS also performs best on unscaled data, again with a polynomial degree of 8, now achieving an MSE of 5253.97. When scaling the terrain data on the other hand, Ridge regression performs best using a polynomial degree of 15 and $\lambda = 10^{-6}$, resulting in an MSE of 0.16. We have deemed the model fits achieved with an analysis up to a fifteenth order polynomial satisfactory, but further studies may consider investing larger computational resources to include higher values in parameter searches. Furthermore, for the Franke function, further analysis could be done to study the effects of varying the added stochastic noise. Lastly, we have only considered a small geographic region and future research may delve into the methods performance on a larger variety of data, including how well the methods generalize to different regions.

[1] S. Raschka, Y. Liu, V. Mirjalili, and D. Dzhulgakov, *Machine Learning with PyTorch and Scikit-Learn: Develop Machine Learning and Deep Learning Models with Python*, Expert insight (Packt Publishing, 2022).

[2] A. M. Legendre, *Nouvelles méthodes pour la détermination des orbites des comètes: avec un supplément contenant divers perfectionnemens de ces méthodes et leur application aux deux comètes de 1805* (Courcier, 1806).

[3] S. M. Stigler, the Annals of Statistics , 465 (1981).

[4] A. E. Hoerl and R. W. Kennard, Technometrics **12**, 55 (1970).

[5] R. Tibshirani, Journal of the Royal Statistical Society. Series B (Methodological) **58**, 267 (1996).

[6] M. Hjort-Jensen, *Applied Data Analysis and Machine Learning* (UiO, 2021) Chap. From Regression to Support Vector Machines.

[7] F. Andreis, Shrinkage methods (ridge, lasso, elastic nets) (2017).

[8] M. Hjort-Jensen, *Applied Data Analysis and Machine Learning* (UiO, 2021) Chap. Linear Regression and Statistical Interpretations.

[9] M. Hjort-Jensen, *Applied Data Analysis and Machine Learning* (UiO, 2021) Chap. From Ordinary Linear Regression to Ridge and Lasso Regression.

[10] M. Hjort-Jensen, *Applied Data Analysis and Machine Learning* (UiO, 2021) Chap. Statistical Interpretations and Resampling Methods.

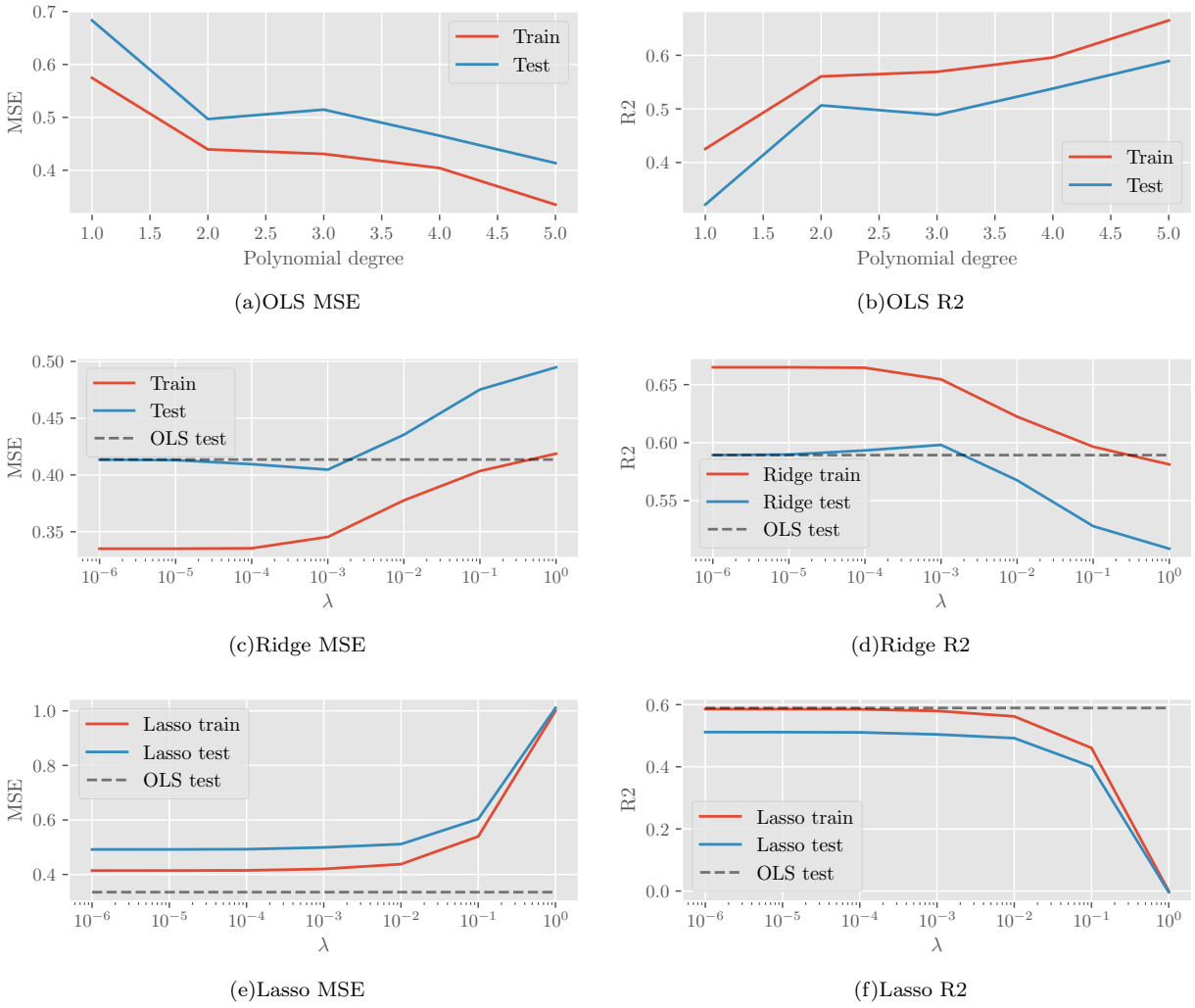
[11] I. J. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (MIT Press, Cambridge, MA, USA, 2016) http://www.deeplearningbook.org.

[12] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, Journal of Machine Learning Research **12**, 2825 (2011).

[13] J. D. Hunter, Computing in Science & Engineering **9**, 90 (2007).

[14] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: data mining, inference and prediction*, 2nd ed. (Springer, 2009).

Appendix A: Appendix



(a)OLS MSE

(b)OLS R2

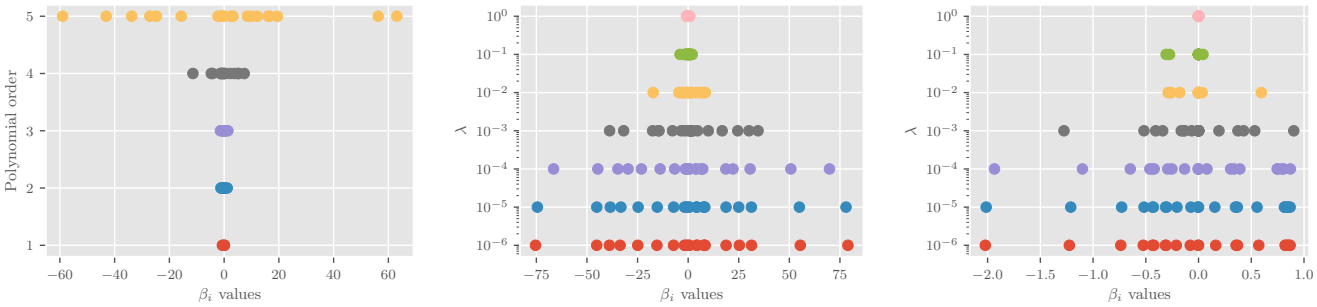
(c)Ridge MSE

(d)Ridge R2

(e)Lasso MSE

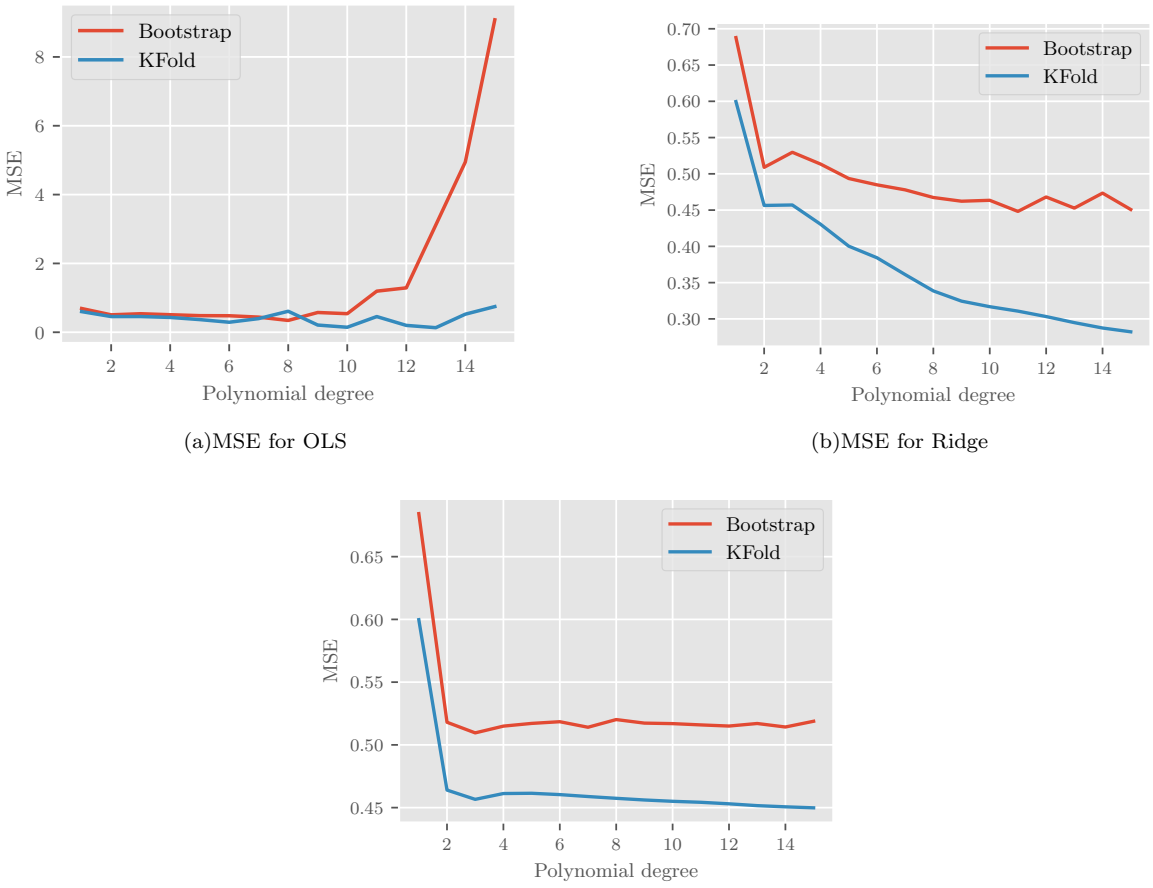
(f)Lasso R2

FIG. 14. A comparison of the MSE and R2 scores for OLS, Lasso and Ridge regression applied to the terrain data from figure 3. All plots use a resolution of 25x25 data points. For Ridge and lasso we have used a fifth order polynomial.



(a)OLS (b)Ridge (c)Lasso

FIG. 15. Polynomial weight β_i values in $\boldsymbol{\beta}$ for OLS, Ridge and Lasso regression on the terrain data shown in figure 3 are displayed for different values of λ or polynomial degree. Each color corresponds to a λ value. Ridge and Lasso are run using a fifth order polynomial.



(a)MSE for OLS (b)MSE for Ridge

(c)MSE for Lasso

FIG. 16. A comparison of the MSE calculated using bootstrapping and k-fold cross-validation on the terrain data shown in figure 3 for increasing polynomial degrees. 100 Bootstraps with 200 samples each are performed and 10 k-folds. Both Ridge an Lasso regression use a $\lambda = 0.01$.