Linear Regression, Resampling and Machine Learning
FYS-STK4155 Project 1

Halvor Melkild*
Department of Physics, University of Oslo

Martin Tømterud†
Department of Physics and Technology, University of Bergen and Centre for Materials Science and Nanotechnology, University of Oslo

Code available at git rep https://github.com/martintomterud/DA-ML

(Dated: October 11, 2022)

Different methods of linear regression are compared on two datasets; one generated by the Franke function and one set of topographical data. We find that ordinary least squares regression gives the lowest mean square error, outperforming the more complex ridge and lasso regression methods. We postulate that this is due to the complexity of the data we analyse, which prevents overfitting using OLS, making the advantages of ridge and lasso unimportant for the present data.

Keywords: machine learning, ridge regression, lasso regression, bootstrap resampling, cross validation

I. INTRODUCTION

As larger and larger datasets are produced by scientists, the use of "classical" statistical methods have been debated in the community. Particularly, there has been an increasing interest in developing efficient statistical tools for the analysis of large amounts of data. Machine learning is a technique that has generated interest and enthusiasm from scientists in all fields. The collection of techniques available is rapidly growing, all from simple linear regression to unsupervised learning. In the case of supervised learning techniques, which include linear regression, the goal is to extract a model relating the in- and output, from a set of known cases.

In this paper we analyse and compare three different methods of linear regression. These are Ordinary Least Squares, Ridge regression and Lasso regression. In Section II we introduce the concept of linear regression and the details of the three methods we want to compare. The resampling techniques, used for testing the methods, are discussed in Section III. The results are presented in Section IV and here we also discuss our findings. We explore the relationship between the bias and variance of the models in each of the three cases. The methods are tested on two different datasets; the first set is generated by the analytic Franke function and the second is generated by real world terrain data.

II. THEORY

We begin by introducing the general ideas behind linear regression. Then follows a presentation of the three different implementations we are testing. We also show how the bias and variance contributes to the expected error of our models.

A. Linear Regression

Suppose we have a sample of n observed outcomes, y_i, where the features of the input is assumed to be characterised by m variables. The set of values of these variables for each sample is organised in the $n \times m$ design matrix, $\boldsymbol{X}$. The set of observations is called the response, and is a vector of length n;

$$\boldsymbol{y} = [y_1,\, y_2,\, \ldots,\, y_n]. \tag{1}$$

In general, each element of both $\boldsymbol{X}$ and $\boldsymbol{y}$ can be multidimensional variables, but for our use, they will simply be scalar variables. Finding relevant input variables to construct the design matrix, is one of the main challenges when building a regression model. Both for the Franke function and the topological data, we are considering a continuous map from two dimensions to one. The output y_i could therefore be approximated by a polynomial in the two coordinates, x_{1i} and x_{2i}. Our design matrix is therefore chosen as

$$X = \begin{pmatrix} 1 & x_{11} & x_{21} & x_{11}x_{21} & x_{11}^2 & x_{21}^2 & \cdots & x_{21}^p \\ 1 & x_{12} & x_{22} & x_{12}x_{22} & x_{12}^2 & x_{22}^2 & \cdots & x_{22}^p \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{1n} & x_{2n} & x_{1n}x_{2n} & x_{1n}^2 & x_{2n}^2 & \cdots & x_{2n}^p \end{pmatrix}, \tag{2}$$

where the number of columns, m, is decided by the polynomial degree, p, we want to work with

$$m = \frac{1}{2}(p+1)(p+2). \tag{3}$$

The aim with regression is to find a model which explains the response, $\boldsymbol{y}$, in terms of the quantities in $\boldsymbol{X}$. A

* Correspondence email address: halvor.melkild@fys.uio.no
† Correspondence email address: martin.tomterud@uib.no

good model should be able to predict y for future events, with satisfying precision. We will assume that $\boldsymbol{y}$ can be understood in terms of a continuous function $f(\boldsymbol{X})$ and some noise or error, $\boldsymbol{\epsilon}$, on the form

$$\boldsymbol{y} = f(\boldsymbol{X}) + \boldsymbol{\epsilon}. \tag{4}$$

The noise will throughout be assumed to follow a normal distribution with mean $\mu = 0$, and variance $\mathrm{Var}(\epsilon) = \sigma^2$.

In the present work, we will focus on *linear* regression, *i.e.* we will assume that f depends linearly on X,

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}. \tag{5}$$

The vector $\boldsymbol{\epsilon}$ is the deviation from the linear response model, which is given by $\boldsymbol{X}\boldsymbol{\beta}$, where $\boldsymbol{\beta}$ is the vector containing the linear regression coefficients. We denote our linear model as,

$$\tilde{\boldsymbol{y}} = \boldsymbol{X}\boldsymbol{\beta}. \tag{6}$$

The error made by the linear model can therefore be expressed as

$$\boldsymbol{\epsilon} = \boldsymbol{y} - \tilde{\boldsymbol{y}}. \tag{7}$$

The objective of our regression analysis is to determine $\boldsymbol{\beta}$ in such a way that the linear model matches the data in the "best" possible way. To define what we mean by "best", we have to introduce a measure, called the *cost* function, $C(\boldsymbol{\beta})$. In general, the cost function should be a measure of the size of $\boldsymbol{\epsilon}$ and the optimal $\boldsymbol{\beta}$ should minimise C. The definition of the cost function another of the challenge when building a regression model, where the choice heavily influences the final outcome. Depending on the dataset and selected model, different definitions of C might be more or less suitable. The definition of the cost function is what separates the three different methods we will study, ordinary least squares (OLS), ridge and lasso regression.

B. Ordinary Least Squares

The OLS method is discussed for instance in Hastie, Tibshirani, and Friedman [1, Chapter 2], and we follow the discussion presented there.

The method imposes the standard Euclidean L^2-norm on the vector space, and defines the cost function simply as the squared norm of the error vector,

$$\begin{aligned} C^{\mathrm{OLS}}(\boldsymbol{\beta}) &= \|\boldsymbol{\epsilon}\|^2 \\ &= \|\boldsymbol{y} - \tilde{\boldsymbol{y}}\|^2 \end{aligned} \tag{8}$$

This is equivalent to the mean square error (MSE),

$$\mathrm{MSE}(\boldsymbol{y}, \tilde{\boldsymbol{y}}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2. \tag{9}$$

On vector form we therefore define our cost function as

$$C^{\mathrm{OLS}}(\boldsymbol{\beta}) = \frac{1}{n}(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}). \tag{10}$$

We find the optimal value of $\boldsymbol{\beta}^{\mathrm{OLS}}$ by differentiating C by β and equating the result to zero:

$$\frac{\partial C^{\mathrm{OLS}}(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = -\frac{2}{n}\boldsymbol{X}^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}) \tag{11}$$

$$\Rightarrow 0 = \boldsymbol{X}^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}^{\mathrm{OLS}}) \tag{12}$$

$$\Rightarrow \boldsymbol{\beta}^{\mathrm{OLS}} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y} \tag{13}$$

The predicted output, $\tilde{\boldsymbol{y}}$, is now obtained from the input $\boldsymbol{X}$, and output $\boldsymbol{y}$

$$\tilde{\boldsymbol{y}} = \boldsymbol{X}\boldsymbol{\beta}^{\mathrm{OLS}} = \boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y} \tag{14}$$

From Eq. (13) we can derive the variation in $\boldsymbol{\beta}^{\mathrm{OLS}}$.

$$\begin{aligned} \mathrm{Var}(\boldsymbol{\beta}^{\mathrm{OLS}}) &= \mathbb{E}[\boldsymbol{\beta}_{\mathrm{OLS}}\boldsymbol{\beta}_{\mathrm{OLS}}^T] - \mathbb{E}[\boldsymbol{\beta}_{\mathrm{OLS}}]\mathbb{E}[\boldsymbol{\beta}_{\mathrm{OLS}}^T] \\ &= (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T \\ &\quad \times (\mathbb{E}[\boldsymbol{y}\boldsymbol{y}^T] - \mathbb{E}[\boldsymbol{y}]\mathbb{E}[\boldsymbol{y}]^T) \\ &\quad \times \boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X})^{-1} \\ &= (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\mathrm{Var}(\boldsymbol{y})\boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X})^{-1}. \end{aligned} \tag{15}$$

From our assumption of the form of $\boldsymbol{y}$ in Eq. (5) and the normal distribution of $\boldsymbol{\epsilon}$, it follows that

$$\mathbb{E}[\boldsymbol{y}] = \mathbb{E}[\boldsymbol{X}\boldsymbol{\beta}] + \mathbb{E}[\boldsymbol{\epsilon}] = \boldsymbol{X}\boldsymbol{\beta} \tag{16}$$

and

$$\mathbb{E}[\boldsymbol{y}\boldsymbol{y}^T] = \boldsymbol{X}\boldsymbol{\beta}\boldsymbol{\beta}^T\boldsymbol{X}^T + \sigma^2\boldsymbol{I}. \tag{17}$$

This gives us

$$\mathrm{Var}(\boldsymbol{y}) = \sigma^2\boldsymbol{I}, \tag{18}$$

which we can insert in Eq. (15) and conclude that

$$\mathrm{Var}(\boldsymbol{\beta}) = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\sigma^2. \tag{19}$$

C. Ridge Regression

The approach for finding β with Ordinary Least Squares demand that $\boldsymbol{X}^T\boldsymbol{X}$ is invertible. If not, β^{OLS} is not uniquely defined. To avoid this problem, we can introduce a positive regularisation parameter λ

$$\boldsymbol{X}^T\boldsymbol{X} \to \boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I}. \tag{20}$$

making sure the matrix is invertible. This is equivalent to modifying the cost function to the one that defines ridge regression, [4]

$$\begin{aligned} C^{\mathrm{R}}(\boldsymbol{\beta}) &= \|\boldsymbol{y} - \tilde{\boldsymbol{y}}\|^2 + \lambda\|\boldsymbol{\beta}\|^2 \\ &= (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}) + \lambda\boldsymbol{\beta}^T\boldsymbol{\beta}. \end{aligned} \tag{21}$$

Again, taking the derivative with respect to $\boldsymbol{\beta}$ and equating to zero gives the optimal value, $\boldsymbol{\beta}^{\mathrm{R}}$, for the Ridge regression method,

$$\frac{\partial C^{\mathrm{R}}(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = -2\boldsymbol{X}^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}) + 2\lambda \boldsymbol{I}\boldsymbol{\beta} \tag{22}$$

$$\Rightarrow \boldsymbol{\beta}^{\mathrm{R}} = (\boldsymbol{X}^T\boldsymbol{X} + \lambda \boldsymbol{I})^{-1}\boldsymbol{X}^T\boldsymbol{y}. \tag{23}$$

We observe that in the limit $\lambda = 0$, Ridge regression reduces to OLS.

From the cost function we notice that λ penalises bigger $\boldsymbol{\beta}$'s. It's effect is therefore to shrink the all the components of $\boldsymbol{\beta}^{\mathrm{R}}$, but equally. Doing a *singular value decomposition* of $\boldsymbol{X}$, one finds that the directions in the column space of $\boldsymbol{X}$ with the smallest variance shrinks the most. As a consequence, the method does not treat all input parameters as equally important, but the main weight is put on those with higher variance. Under the assumption that the directions with low variance are of less importance, ridge regression therefore helps with reducing the effective degrees of freedom in the problem and could reduce the share of the variance in β which comes from the directions we deem less important.

D. Lasso Regression

Lasso regression differs from Ridge regression in the metric imposed on $\boldsymbol{\beta}$ in the shrinking term. Instead of using the L^2 norm we instead impose the L^1 norm on the shrinkage term only, so that the cost function reads

$$\begin{aligned} C^{\mathrm{L}}(\boldsymbol{\beta}) &= \|\boldsymbol{y} - \tilde{\boldsymbol{y}}\|^2 + \lambda\|\boldsymbol{\beta}\| \\ &= (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}) + \lambda|\boldsymbol{\beta}|. \end{aligned} \tag{24}$$

There is no explicit analytical derivation for a $\boldsymbol{\beta}$ that minimises $C^{\mathrm{L}}(\boldsymbol{\beta})$ like there is for Ridge and OLS. Instead numerical methods like gradient descent is employed to find the optimal value of $\boldsymbol{\beta}$.

E. Statistical Metrics

To measure the error and performance of our models we employ statistical metrics that can measure the errors made by the methods and compare their performance. One such tool has already been introduced, the mean square error, which reads

$$\mathrm{MSE}(\boldsymbol{y}, \tilde{\boldsymbol{y}}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2. \tag{25}$$

Another useful metric is the R^2-score, also known as the coefficient of determination, which measures to what extent the variation in $\boldsymbol{y}$ is explained by the model $\tilde{\boldsymbol{y}}$. The R^2-score reads

$$\mathrm{R}^2(\boldsymbol{y}, \tilde{\boldsymbol{y}}) = 1 - \frac{\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y}_i)^2}, \tag{26}$$

where we have introduced the mean

$$\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i. \tag{27}$$

F. Bias-Variance

The bias and the variance of the predictions of the regression algorithms are related to the MSE, or the cost function, that we use as a criterion to optimise our $\boldsymbol{\beta}$-parameter. The variance of the resulting MSE quantifies how the prediction is expected to change if the model is trained on a different set of training data. A good model therefore should have a low variance and it should retain a low variance as the training set is altered. Flexible models are known to produce larger variances and the predictions can therefore vary more. The bias quantifies the error that is made by imposing a relation between the response $\boldsymbol{y}$ and the known quantities $\boldsymbol{X}$. In this work we assume a linear relation between the two, and the bias therefore quantifies the error made by the linear relationship, or alternatively, how non-linear the response is. Knowing that the expected value $\mathbb{E}$ is equal to the MSE,

$$\mathbb{E}\left[(\mathbf{y} - \tilde{\mathbf{y}})^2\right] = \frac{1}{n}\sum_{i=0}^{n-1}(y_i - \tilde{y}_i)^2, \tag{28}$$

we can derive that the relation between the MSE, bias and variance is

$$\begin{aligned} \mathbb{E}\left[(\mathbf{y} - \tilde{\mathbf{y}})^2\right] &= \frac{1}{n}\sum_{i=0}^{n-1}(f_i - \mathbb{E}[\tilde{y}_i])^2 \\ &+ \frac{1}{n}\sum_{i=0}^{n-1}(\tilde{y}_i - \mathbb{E}[\tilde{y}_i])^2 \\ &+ \sigma^2 \\ &= \mathrm{Bias}^2 + \mathrm{Variance} + \sigma^2. \end{aligned} \tag{29}$$

In the above we have used $f_i = y_i - \epsilon_i$, by rewriting Eq. (4) on component form. This equation therefore tells us that to lower the MSE we want a low bias and a low variance. The σ^2-term arises from the normal distribution of the error, ϵ. This is referred to as an irreducible error in the literature.

We would ultimately like to choose the regression model that gives a MSE as low as possible, and by Eq. (29) this would imply choosing a model with low bias and low variance. However, as just discussed, improving the flexibility of a model is often associated with improving the bias, but giving a higher variance. This is called the bias-variance trade-off, and implies that one often has to look for some compromise.

We use the following relations in the proof of Eq. (29) below.

$$\mathbb{E}[X^2] = \mathrm{Var}[X] + (\mathbb{E}[X])^2 \tag{30}$$

$$\mathbb{E}[f] = f \Rightarrow \mathbb{E}[y] = \mathbb{E}[f + \epsilon] = f \tag{31}$$

$$\mathrm{Var}[\epsilon] = \sigma^2 \Rightarrow \mathrm{Var}[y] = \mathbb{E}[(f + \epsilon - f)^2] = \sigma^2 \tag{32}$$

$$\begin{aligned}
\mathbb{E}[(y - \tilde{y})^2] =& \mathbb{E}[(f + \epsilon - \tilde{y})^2] \\
=& \mathbb{E}[(f + \epsilon - \tilde{y} + \mathbb{E}[\tilde{y}] - \mathbb{E}[\tilde{y}])^2] \\
=& \mathbb{E}[(f - \mathbb{E}[\tilde{y}])^2] + \mathbb{E}[\epsilon^2] + \mathbb{E}[(\mathbb{E}[\tilde{y}] - \tilde{y})^2] \\
&+ 2\mathbb{E}[(f - \mathbb{E}[\tilde{y}])\epsilon] + 2\mathbb{E}[\epsilon(\mathbb{E}[\tilde{y}] - \tilde{y})] \\
&+ 2\mathbb{E}[(\mathbb{E}[\tilde{y}] - \tilde{y})(f - \mathbb{E}[\tilde{y}])] \\
=& (f - \mathbb{E}[\tilde{y}])^2 + \mathbb{E}[\epsilon^2] + \mathbb{E}[(\mathbb{E}[\tilde{y}] - f)^2] \\
&+ 2(f - \mathbb{E}[\tilde{y}])\mathbb{E}[\epsilon] + 2\mathbb{E}[\epsilon]\mathbb{E}[\mathbb{E}[\tilde{y}] - \tilde{y}] \\
&+ 2\mathbb{E}[\mathbb{E}[\tilde{y}] - \tilde{y}](f - \mathbb{E}[\tilde{y}]) \\
=& (f - \mathbb{E}[\tilde{y}])^2 + \mathbb{E}[(\mathbb{E}[\tilde{y}] - \tilde{y})^2] + \mathbb{E}[\epsilon^2] \\
=& \mathrm{Bias}[\tilde{y}]^2 + \mathrm{Var}[\tilde{y}] + \sigma^2
\end{aligned} \tag{33}$$

G. The Franke Function

For $x, y \in [0, 1]$, the Franke function is given by [2]

$$\begin{aligned}
f(x, y) =& \frac{3}{4} \exp\left(-\frac{(9x-2)^2}{4} - \frac{(9y-2)^2}{4}\right) \\
&+ \frac{3}{4} \exp\left(-\frac{(9x+1)^2}{49} - \frac{(9y+1)^2}{10}\right) \\
&+ \frac{1}{2} \exp\left(-\frac{(9x-7)^2}{4} - \frac{(9y-3)^2}{4}\right) \\
&- \frac{1}{5} \exp\left(-(9x-4)^2 - (9y-7)^2\right).
\end{aligned} \tag{34}$$

III. METHODS

We apply two different resampling methods in the present project to test the validity of our model by computing predictions on randomly selected subsets of our data.

A. Bootstrap Resampling

With bootstrap resampling, the approach is to compute averages of several small data samples that are drawn from the main training set and then replaced. We then compute the regular statistics on the smaller data sets and compute the averages of all the smaller data set statistics. The bootstrap is particularly useful for evaluating bias and variance, and we use it to investigate the bias variance crossover in our model, to determine at which complexity the linear model breaks down and where we obtain the lowest MSE.

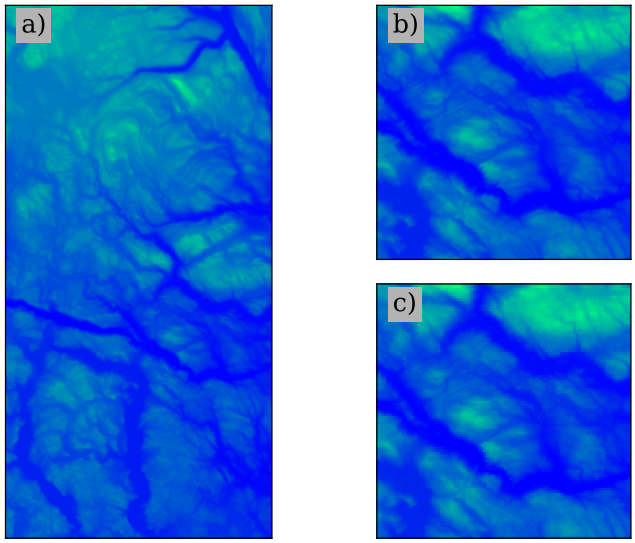

Figure 1. Illustration of the terrain selecting process. Panel a) shows the full terrain dataset. Panel b) Shows the snipped out region of the dataset. Panel c) is a selection of every 10-th datapoint in panel b).

B. k-fold Cross Validation

Cross validation is a method for evaluating the performance of a machine learning model. We split the total dataset into k folds (or groups), and retain one as a testing set, while we use the rest as a training set. After fitting the model to the training set, we evaluate it on the test set and compute the evaluation scores, *i.e.* the MSE, R^2 score, bias and variance. This we do for each fold, before calculating the mean of the evaluation scores we obtain. We stick to $k = 5$- and $k = 10$-fold cross validation, both of which has been shown to neither suffer from too high a bias, nor too high a variance [3].

C. Scaling the terrain data

The terrain data supplied is a rather large dataset, and we therefore take steps to scale down the number of datapoints and the extrema of the values. We start by making a selection of a random region within the terrain data itself. We chose a size of $N_T = 1000$ datapoints, such that the selected region measured 1000×1000 dataponts in total. From these datapoints we then selected every 10-th point and constructed a terrain landscape measuring 100×100 points. An illustration of this process is shown in Figure 1 After selecting the final terrain region, we normalise the data. This is done by finding the maximum and minimum value of the dataset, z_{max} and z_{min}, and for every datapoint z in the dataset, performing the following transformation:

$$z \to \frac{z - z_{min}}{z_{max} - z_{min}}. \tag{35}$$

IV. RESULTS AND DISCUSSION

All the test data is produced by a noisy Franke function, $F(x,y) = f(x,y) + N(x,y,\mu = 0, \sigma = 0.1)$, where $f(x,y)$ is the Franke function in Eq. (34), and N is randomly distributed noise drawn from the normal distribution with mean $\mu = 0$ and standard deviation $\sigma = 0.1$ at coordinate (x,y).

A. OLS

The MSE and R^2 scores for the OLS regression on the Franke function are shown in Figure 2. We investigate polynomials up to degree $p = 14$ to explore the region where the MSE flattens out and to observe it increasing again as the complexity (*i.e.* polynomial degree) becomes sufficiently large. The MSE decreases with polynomial degree and flattens out. The data was split into a train and test interval, with test size = 0.2. We have also plotted the MSE from the k-fold cross validation and bootstrap algorithms, as well as the R^2 score for the cross validation. For sufficiently large polynomial degrees, neither of these algorithms are seen to improve the MSE above the value obtained using just the simple split, shown in red with circular markers.

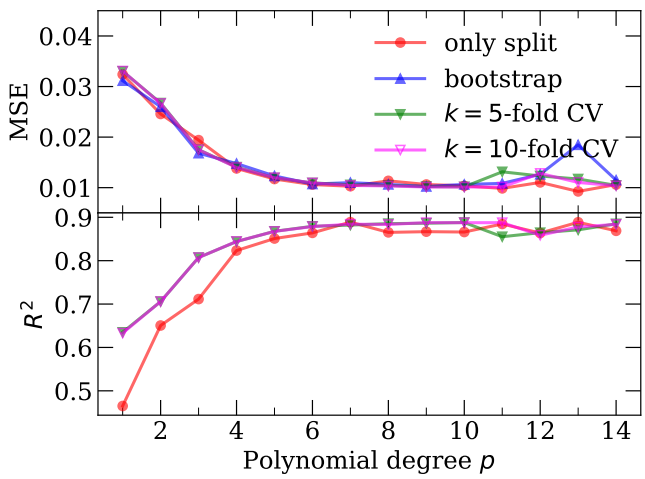

Figure 2. The mean square error (upper panel) and the R^2-score (lower panel) produced by OLS regression on a dataset generated by the Franke function. A grid size of $N = 50 \times 50$ was used.

The bias-variance trade off for OLS regression is studied in Figure 3. We study two different grid sizes (*i.e.* number of data points) as the difference is quite dramatic. We have employed bootstrap resampling with the number of bootstrap runs $k_B = 100$. Both grid sizes overall feature a low variance and a squared bias almost equal to the MSE. However, for the small grid size, there is a cross over between the bias and the variance for polynomial degree $p = 9$. The variance overtakes the bias and increases dramatically, whereas the bias is decreased. This does not happen for a the larger data point grid, where the variance remains almost at 0 for every polynomial degree studied.

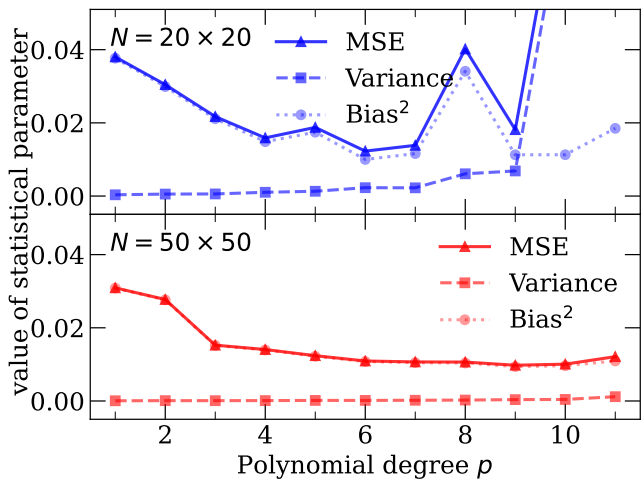

Figure 3. The bias-variance tradeoff studied using the bootstrap algorithm for OLS regression on two different datagrid sets for the Franke function.

The low variance we observe for the larger grid size is further supported by Figure 4. Here we study the variance of the OLS regression coefficients β_j for the same two grid sizes as in Figure 3. We have used polynomial degree $p = 5$ to obtain the coefficients. The error bars are taken to be the $\mathrm{Var}(\beta_j)$. As stated earlier, this is proportional to the irreducible error σ^2, which arises from the noise added to the Franke function, and the diagonal element of the matrix $(\mathbf{X}^T\mathbf{X})^{-1}$.

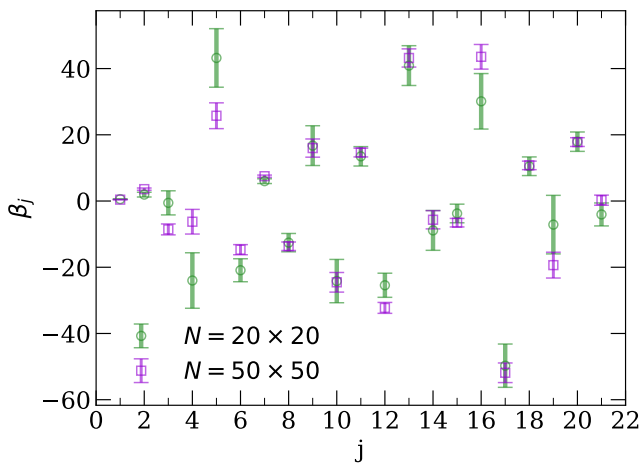

Figure 4. The coefficients β_j of the OLS regression vector $\boldsymbol{\beta}$ for a polynomial of degree $p = 5$. The coefficients are computed on two different data point grids as indicated by the legend. The error bars are taken to be $\mathrm{Var}(\beta_\mathrm{j})$. Observe that the variance decreases with the number of points.

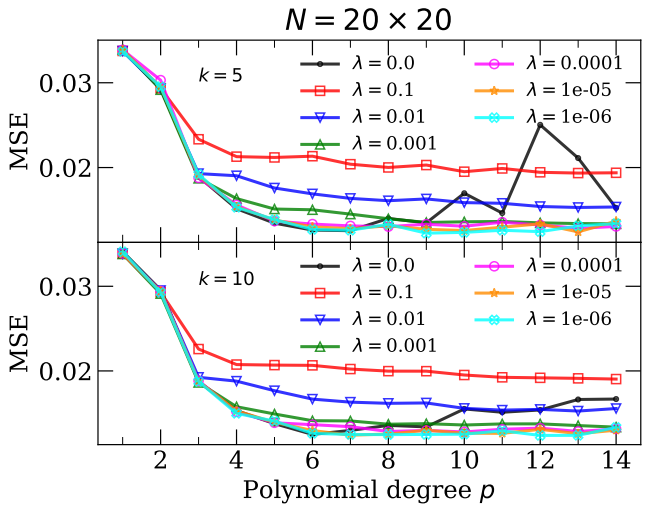

Figure 5. Mean square error for ridge regression on the Franke function with a gridsize of $(N_x, N_y) = (20, 20)$. The upper panel shows a $k = 5$-fold cross validation, the lower $k = 10$.

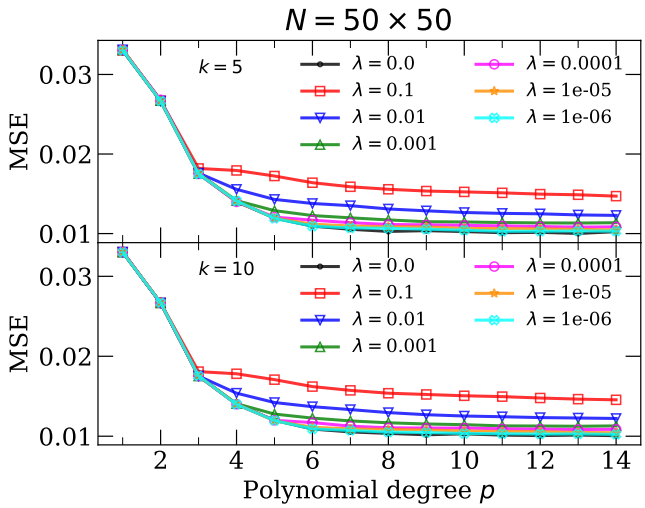

Figure 6. Mean square error for ridge regression on the Franke function with a gridsize of $(N_x, N_y) = (50, 50)$. The upper panel shows a $k = 5$-fold cross validation, the lower $k = 10$.

B. Ridge

MSE plots for ridge regression on the Franke function are shown in Figures 5 and 6. The two figures differ in the grid of (x, y)-points used to generate the Franke function, and are the same two studied for OLS. The upper panel of the figures show the MSE using a $k = 5$ fold cross validation resampling, whereas the lower panel uses $k = 10$.

For both grid sizes and both k-values the lowest MSE is achieved by $\lambda = 0$, which is equivalent to OLS. For large polynomial degrees, the low-point density OLS MSE starts increasing again above $p = 9$, but for the high-

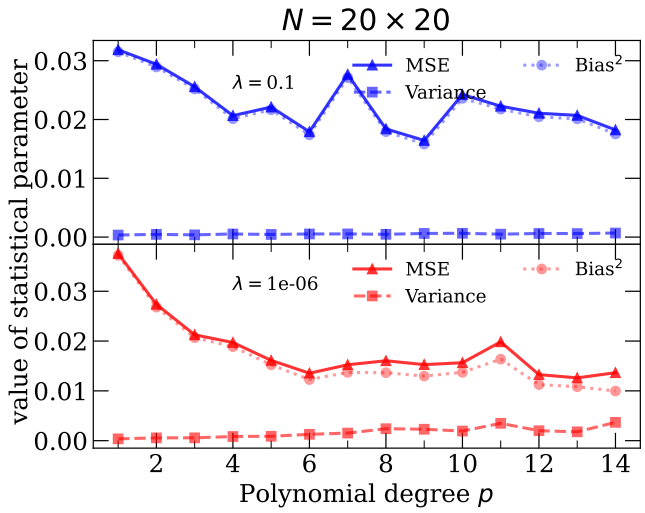

Figure 7. The bias-variance trade off investigated using the bootstrap algorithm with $k_B = 100$ and ridge regression with λ as indicated in the panels. Data generated by the Franke function on a grid of points $(N_x, N_y) = (20, 20)$.

point density grid the OLS case remains the best. Apart from the OLS, the lowest MSE is achieved for the lowest value of λ and the highest MSE is obtained for the largest value of λ. This is unsurprising considering OLS seemingly gives the best result, and a smaller λ implies the ridge regression gives results closer to OLS.

We have also applied the bootstrap algorithm to investigate the bias-variance trade off for ridge regression. The results are shown in Figures 7 and 8. We investigated two of the λ values used in computing the MSE; the one giving the best results and the one giving the worst. The worst (*i.e.* largest) lambda is shown in the upper panels and the best in the lower. The two figures again represent the two different grid sizes we are investigating. As we saw with for OLS regression, the smaller grid in Figure 7, especially for the "good" λ in the lower panel, shows that the variance becomes finite for large polynomial degrees, decreasing the bias.

C. Lasso

MSE plots of lasso regression on the Franke function are shown in Figures 9 and 10. Both are generated using k-fold cross validation, with $k = 5$ in the upper panel and $k = 10$ in the lower panel. The two figures, as before, show the MSE for the two (x, y) grids we have studied throughout this text. As with ridge regression, the MSE improves with smaller values of λ. The lasso-algorithm is not well defined for $\lambda = 0$ (OLS) and this result is therefore not included in the figures. Nevertheless it is evident when compared to the results obtained for ridge and OLS regression that lasso regression gives a larger MSE than the methods studied earlier.

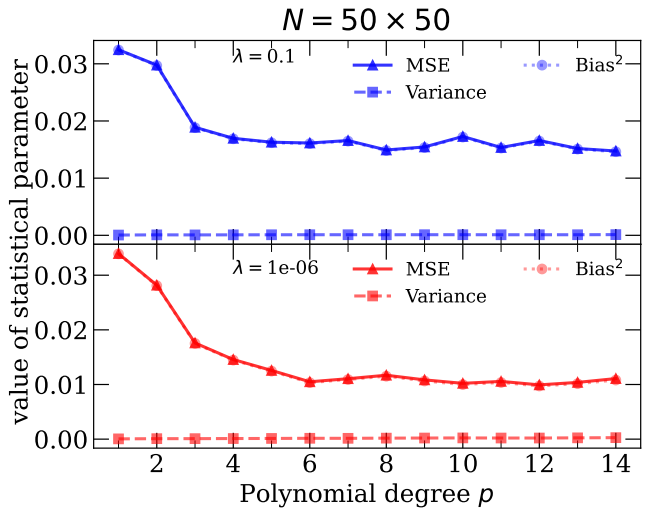

Figure 8. The bias-variance trade off investigated using the bootstrap algorithm with $k_B = 100$ and ridge regression with λ as indicated in the panels. Data generated by the Franke function on a grid of points $(N_x, N_y) = (50, 50)$.

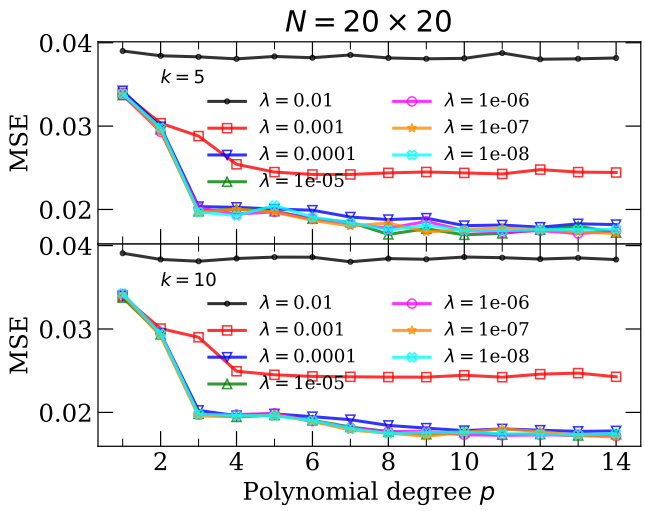

Figure 9. Mean square error for lasso regression on the Franke function with a gridsize of $(N_x, N_y) = (20, 20)$. The upper panel shows a $k = 5$-fold cross validation, the lower $k = 10$.

We have also looked at the bias-variance trade off using lasso regression. The results are presented in Figure 11. This time we only looked at the low point density grid, $(N_x, N_y) = (20, 20)$, as the results were practically identical between the two, and not much different from the high point density grid we studied using ridge regression in Figure 8. Again we chose to study a high and low value of λ. For the high value of λ, which gives the worst MSE, the variance remains approximately zero for each studied polynomial degree, whereas the square bias closely follows the MSE. For the low value of λ we obtain a finite, but small, variance implying that the square bias

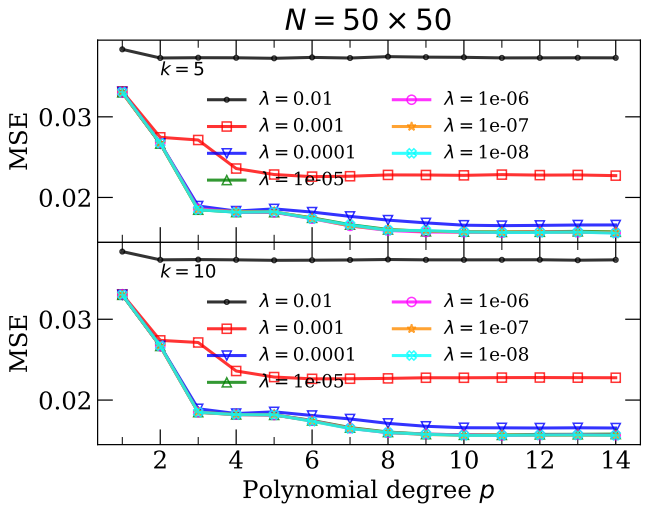

Figure 10. Mean square error for lasso regression on the Franke function with a gridsize of $(N_x, N_y) = (50, 50)$. The upper panel shows a $k = 5$-fold cross validation, the lower $k = 10$.

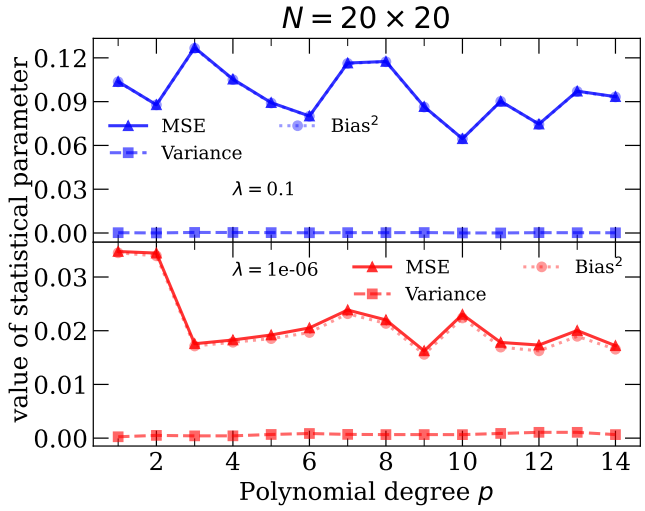

Figure 11. The bias-variance trade off investigated using the bootstrap algorithm with $k_B = 100$ and lasso regression with λ as indicated in the panels. Data generated by the Franke function on a grid of points $(N_x, N_y) = (20, 20)$.

drops below the MSE.

D. Intermezzo

Before proceeding with evaluating our model on terrain data we summarise and discuss what we have learned from applying the regression methods to the Franke function. The first observation is that OLS gives the smallest MSE overall, and that for large enough polynomial degree, the MSE flattens out. The same is true for the R^2-score, which flattens out and approaches unity. The

Franke function is a sum of exponentials, which can be Taylor expanded as a series of powers in x and y. Thus it is no surprise that the MSE decreases with increasing polynomial degree, as we can more accurately describe each power in the Taylor series with increasing polynomial degree. Furthermore, the n-th term in the Taylor series is proportional to $1/n!$, implying that they become less and less important. This is what is reflected when the MSE flattens out with increasing polynomial degree. There is a caveat however. As explained in *e.g.* Ref. [1], with larger polynomial degree we run the risk of overfitting. When this happens, the polynomials will start fitting the noise present in the data as well, not just the Franke function. This increases the variance and lowers the bias. This effect is present in the upper panel of Figure 3. However, as we observe, increasing the number of data points can mitigate the overfitting effect for a given polynomial degree. This is evident both from comparing the upper and lower panels of Figure 3, and from the variance of the regression coefficients β_j in Figure 4. When moving on from OLS to ridge and lasso regression the overfitting issue is less apparent. In all cases where we tested the bias-variance trade off for ridge and lasso regression, the variance remained close to zero. Summarised we therefore see that OLS has given us the lowest MSE, and when remaining conscious about polynomial degrees and the number of datapoints in our grid, an increasing variance has not given us issues with increasing the MSE.

E. Terrain Data

We now present the MSE of OLS, ridge and lasso regression performed on the terrain data. We performed two sets of OLS regression; one set on unnormalised terrain data and one set on normalised terrain data. For all instances we performed both 5-fold and 10-fold cross validation.

OLS on unnormalised and normalised terrain data is presented in Figures 12 and 13, respectively. Both instances give the lowest MSE for polynomial degree $p = 10$, for both 5- and 10-fold corss validation. Beyond $p = 10$, the MSE blows up. likely due to the previously discussed bias variance trade off. It is also worth nothing that the shape of the MSE in both figures is identical. The scale on the ordinate differs, but that is due to the normalisation. The MSE itself is not normalised by the scale of the data.

MSE for ridge and lasso regression is shown in Figures 14 and 15 respectively. The upper panels show $k = 5$-fold cross validation and the lower panels show $k = 10$-fold cross validation. In both instances, as for the Franke function, a λ closer to zero gives a lower MSE. We went to a higher polynomial degree to investigate if we could see the MSE flatten out, but we have not yet reached that region for $p = 18$ for the λ-values closest to zero.

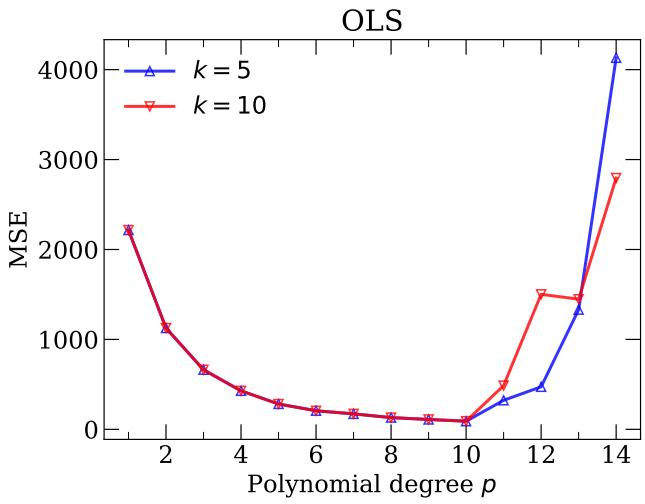

Figure 12. OLS regression on unnormalised terrain data using both 5- and 10-fold cross validation, as indicated in the legend.

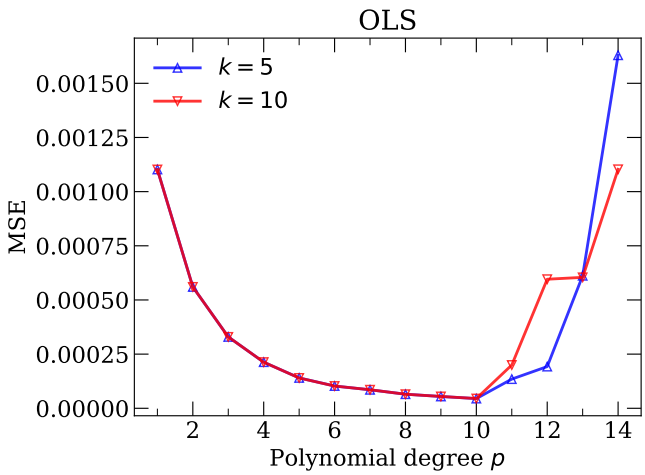

Figure 13. OLS regression on normalised terrain data using both 5- and 10-fold cross validation, as indicated in the legend.

As we observed when examining the regression results for the Franke function, the same trends are present in the regression results for the terrain data. The MSE for both ridge and lasso regression is much larger than the MSE we obtain from OLS. Furthermore, OLS obtains an MSE minimum at $p = 9$, whereas ridge and lasso still are approaching their minima for $p = 18$. Extrapolating the results we obtained for the Franke function, we do not expect the minima of neither ridge or lasso regression to give a smaller MSE than for OLS. This is further supported by the fact that the MSE of ridge and lasso regression is smaller for smaller λ. We therefore expect the minima of the MSE of both ridge and lasso to be at $\lambda = 0$; *i.e.* OLS regession. There is somewhat of a surprise associated with the fact that we observe an increasing MSE with OLS on the terrain data for large p. Intuitively, one would think that the terrain data is so

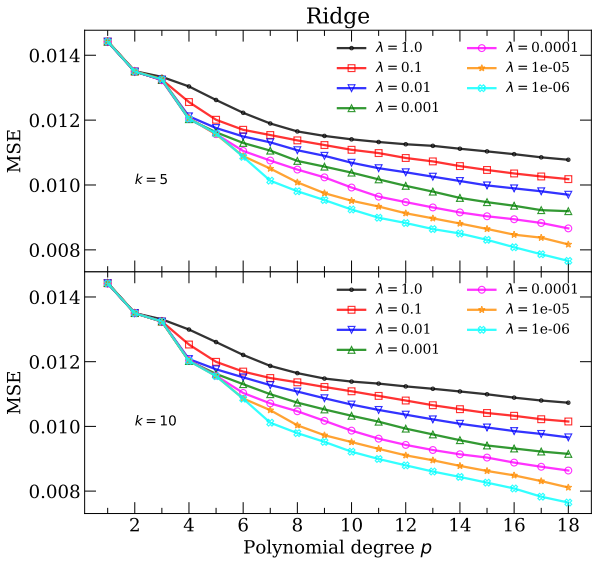

Figure 14. Ridge regression on normalised terrain data. 5-fold cross validation in the upper panel, 10-fold cross validation in the lower panel. Range of λ indicated in the figure legend.

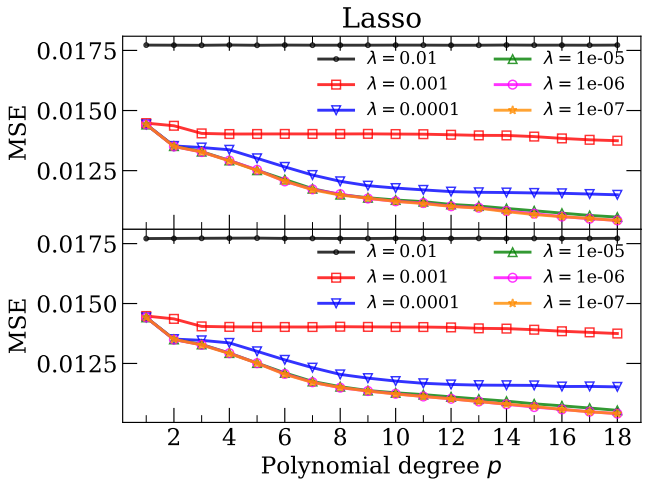

Figure 15. Lasso regression on normalised terrain data. 5-fold cross validation in the upper panel, 10-fold cross validation in the lower panel. Range of λ indicated in the figure legend.

complex that it was hard to generate overfitting. This overfitting is not reproduced by ridge and lasso regression. Separate runs, not shown herein, indicate that the overfitting issue is only an issue for $p = 13$ and $p = 14$. This could imply that we have implemented some sort of systematic error in our code that fails for polynomials of these degrees. The overall takeaway from both studying the Franke function and the terrain data is that OLS performs best, and that we only once saw a bias-variance crossover. The latter implies that the variance has remained small for all methods, in turn rendering the advantages of ridge and lasso regression over OLS small. This is reflected in the fact that lasso and ridge regression performs best for small λ – when they are close to true OLS. The likely cause is that the data is complex enough to prevent overfitting, even at large polynomial degrees, rendering the OLS method superior.

V. CONCLUSION

We have performed compared three different methods of linear regression (OLS, ridge and lasso) on two sets of data; one set generated by the analytic Franke function, and one set of real world terrain data. Analysing the Franke function, we observed the bias-variance crossover using OLS and we found that OLS gave the lowest MSE of the three methods. Furthermore, we observed that increasing the number of datapoints used to generate the Franke function reduced the variance and prevented overfitting. The observations we made on the Franke function were consistent with the ones we made analysing the terrain data, where we again find that OLS gives the best MSE. We understand this in terms of the large complexity of the data. This complexity lead to no overfitting of the data, even for large polynomial degree, which in turn cancelled the advantages of ridge and lasso regression over OLS.

[1] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, New York, 2009.

[2] R. Franke *A Critical Comparison of Some Methods for Interpolation of Scattered Data* Naval Postgraduate School, California, 1979.

[3] G. James, D. Witten, T. Hastie, and R. Tibshirani. *An Introduction to Statistical Learning*. Springer, New York, 2017.

[4] A. E. Hoerl, R. W. Rennard, (1970). Ridge Regression: Biased Estimation for Nonorthogonal Problems. *Technometrics*, 12(1), 55–67.