UNIVERSITETET I OSLO

University of Oslo
Department of Physics

FYS-STK4155 - Applied Data Analysis and Machine Learning

Project 1:

Regression analysis and resampling methods

Authors:

Rebecca Nguyen, Federico Santona, Jonas Semprini and Mathias Svoren

October 15, 2023

Contents

ABSTRACT

This project aims to examine different regression methods in the analysis of topographic data gathered from a region close in Møstvatn Austfjell. In particular the following methods are put in application: Ordinary Least Squares (OLS), Ridge regression and Lasso regression. An assessment is performed on these methods by studying their bias-variance trade-off through resampling techniques such as cross-validation and bootstrap, in addition to evaluating their mean squared error (MSE) and R^2 score. The regression methods are tested and assessed on Franke's function, a widely known function used for testing interpolation and fitting algorithms, before proceeding with fitting the topographic data. Our findings suggest Ordinary Least Squares (OLS) to be the best method for fitting the terrain data. It performed well under the assessment with results such as represented in table IV . Finally it is worth noting that the model is heavily data-dependent so discretionary means are advised in potential implementation.

Keywords: **Mean Squared Error (MSE), Ordinary Least Squares (OLS), Ridge Regression, Lasso Regression, Cross-Validation, Bootstrap resampling, Bias-Variance-Tradeoff**

I. INTRODUCTION

In the 1800s, Sir Francis Galton did a study on sweet peas, a self-fertilizing plant, to understand how strongly the characteristics of one generation would manifest in the following generation. It all started when he noticed the sweet pea packets distributed to his friends had substantial variations. A data set was created where he looked at the size of daughter peas against the size of mother peas and illustrated the basic foundation of what statisticians still call regression [1]. This report presents a comprehensive analysis of function fitting techniques applied to a two-dimensional function known as the Franke function (III A). The primary goal is to assess the performance of three different regression methods, namely Ordinary Least Squares (OLS) (II A 1), Ridge Regression (II A 2), and Lasso Regression (II A 3), in terms of model accuracy, bias-variance trade-off (II B), and generalization capabilities. The study also incorporates resampling techniques (II C 1) and cross-validation (II C 1) to enhance the model evaluation process. Initially, the report outlines the theoretical background of the Franke function and the mathematical formulations of OLS, Ridge, and Lasso regression. The study explores the bias-variance trade-off, to understand the trade-offs between model complexity and generalization performance. To assess these trade-offs, the report employs resampling techniques such as cross-validation. In addition to synthetic data generated from the Franke function, this report extends its analysis to real-world data, offering practical insights into the application of these regression techniques in authentic scenarios.

II. THEORY

A. Linear regression

Linear regression is a fundamental statistical method used for modeling the relationship between a dependent variable and one or more independent variables. In this technique, the aim is to find the following relation[2]:

$$\tilde{\boldsymbol{y}} = \boldsymbol{X}\boldsymbol{\beta} \tag{1}$$

Where X is the design matrix that represents the independent variables, often including a column of ones for the intercept term, every column represents a feature of the model that has been implemented and allows us to express the relationship between the variables in matrix form. The parameter vector β contains the coefficients that the linear regression model aims to estimate. These coefficients determine the intensity and direction of the influence of each independent variable on the dependent variable. In order to understand how good and fitting the is model implemented is important to minimize the cost function

$$C(\boldsymbol{X}, \boldsymbol{\beta}) = \frac{1}{n}\sum_{i=0}^{n-1}(y_i - \tilde{y})^2 \tag{2}$$

finding therefore the optimal set of parameters $\tilde{\beta}$. The cost function depends on the type of regression implemented, In this report, Ordinary Least Square (From now on referred to as simply OLS), Ridge and Lasso regression are analyzed.
We make the assumption that our data can be described by a non-stochastic function $f(\boldsymbol{x})$ and normal distributed noise $\boldsymbol{\varepsilon} \sim \mathcal{N}(0,\, \sigma^2)$

$$\boldsymbol{y} = f(\boldsymbol{x}) + \boldsymbol{\varepsilon} \tag{3}$$

By utilizing linear regression, the function $f(\boldsymbol{x})$ can be approximated by eq. 1. The expectation value of y_i is given by

$$\mathbb{E}(y_i) = \sum_j x_{ij}\beta_j = \boldsymbol{X}_{i,*}\boldsymbol{\beta} \tag{4}$$

and we find the variance of y_i is

$$\mathrm{Var}(y_i) = \sigma^2 \tag{5}$$

Thus, $y_i \sim \mathcal{N}(\boldsymbol{X}_{i,*}\boldsymbol{\beta},\, \sigma^2)$. The full mathematical derivation of these quantities can be found in Appendix A.

1. *Ordinary Least Square (OLS)*

In the case of the OLS, the cost function is constituted by the mean squared error[3]:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\tilde{y} - y)^2 \tag{6}$$

Both use eq. 1 and the matrix representation becomes:

$$C(\boldsymbol{X}, \boldsymbol{\beta}) = \frac{1}{n}\left[(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})\right] \tag{7}$$

Which is now our cost function $\boldsymbol{C(X, \beta)}$. In order to find the optimal parameter $\hat{\beta}$ we need to minimize the cost function, which can be considered as the variance of the quantity y if we interpret the latter as the mean value. Thinking in this way, the cost function represents our model's error, so we want to minimize it to obtain the best-fitting model for our data. We therefore compute the following :

$$\min_{\beta \in R^p} = \frac{1}{n}\left[(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})\right] \tag{8}$$

We can obtain computing the derivative of the cost function $\boldsymbol{C}(X, \beta)$ with respect to β and set it equal to 0 .We therefore get the following:

$$\begin{aligned} \frac{\partial C(\boldsymbol{X}, \boldsymbol{\beta})}{\partial \beta} = \boldsymbol{X}^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}) = 0 \\ \boldsymbol{X}^T\boldsymbol{y} = \boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\beta} \end{aligned} \tag{9}$$

It follows that in the $\boldsymbol{X^T X}$ is invertible we get :

$$\hat{\boldsymbol{\beta}} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y} \tag{10}$$

Which represents the optimal set of parameters. The expectation value of $\hat{\boldsymbol{\beta}}$ is

$$\mathbb{E}(\hat{\boldsymbol{\beta}}) = \boldsymbol{\beta} \tag{11}$$

and its variance

$$\mathrm{Var}(\hat{\boldsymbol{\beta}}) = \sigma^2(\boldsymbol{X}^T\boldsymbol{X})^{-1} \tag{12}$$

2. *Ridge regression*

Ridge regression is often implemented for cases where we encounter singularity problems. This leads to $\boldsymbol{X^T X}$ not being invertible. To overcome the problem a modified Cost Function is implemented, adding the regularization parameter λ multiplied by the square L^2 norm of β [4]:

$$C(\boldsymbol{X}, \boldsymbol{\beta}) = \frac{1}{n}\left[(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})\right] + \lambda\boldsymbol{\beta}^T\boldsymbol{\beta} \tag{13}$$

Analogously with what is done with OSL we get the optimal parameter $\hat{\boldsymbol{\beta}}_{\text{Ridge}}$

$$\hat{\boldsymbol{\beta}}_{\text{Ridge}} = (\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I})^{-1}\boldsymbol{X}^T\boldsymbol{y} \tag{14}$$

where $\boldsymbol{I}$ is a $p \times p$ identity matrix with the following constraint

$$\sum_{i=0}^{p-1} \beta_i^2 \leq t \tag{15}$$

3. *Lasso regression*

Lasso stands for least absolute shrinkage and selection operator. The concept is similar to what has already been analyzed regarding Ridge regression. The difference is that in Lasso regression we had the regularization parameter multiplied by the L^1 norm of β, leading therefore to the following cost function [4].

$$C(\boldsymbol{X}, \boldsymbol{\beta}) = \frac{1}{n}\left[(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})\right] + \lambda\|\boldsymbol{\beta}\|_1 \tag{16}$$

The previous procedure followed with OLS and Ridge cannot be implemented here, because the derivative of the L^1 norm is not continuous. From the derivative we get:

$$\begin{aligned} \frac{d}{d\boldsymbol{\beta}}|\beta| = sgn(\boldsymbol{\beta}) = \begin{cases} 1 & \beta > 0 \\ -1 & \beta < 0 \end{cases} \\ \frac{\partial C(\boldsymbol{X}, \boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = -\frac{2}{n}\boldsymbol{X}^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}) + \lambda sgn(\boldsymbol{\beta}) = 0 \end{aligned} \tag{17}$$

Which brings us to:

$$\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\beta} + \lambda sgn(\boldsymbol{\beta}) = \boldsymbol{X}^T\boldsymbol{y} \tag{18}$$

Which has no analytical solution.

B. Bias-variance trade-off

The relationship between model complexity and the amount of training data needed to train it is known as the bias-variance trade-off. The cost function defined in eq. 2 can be rewritten as

$$C(\boldsymbol{X}, \boldsymbol{\beta}) = \mathbb{E}[(\boldsymbol{y} - \tilde{\boldsymbol{y}})^2] = \mathrm{Bias}[\tilde{\boldsymbol{y}}] + \mathrm{var}[\tilde{\boldsymbol{y}}] + \sigma^2 \tag{19}$$

One can think of the expression as the expected prediction error caused by simplifying assumptions in the model [5] and the full derivation can be found in Appendix A 5. The expected prediction error is made up of three terms:

- The mean value of the model (bias).
- The model's deviation from the true data (variance)
- The variance of the noise term.

In order to minimize the prediction error, we should choose a method that achieves both low variance and bias. It can often be useful to opt for a less complex model with higher bias due to limited data [5].

C. Resampling techniques

Resampling methods mark the next phase in our exploration of various regression techniques. They provide a systematic approach for in-depth model performance evaluation, an important step in ensuring that our models can make accurate predictions on new data without overfitting the training dataset. This aids in preventing the common pitfall of overfitting, where models become too tailored to the training data.
Moreover, these resampling techniques are indispensable in our project. They enable a methodical comparison of different algorithms and their configurations, helping us choose the most effective model for our specific task.
Resampling involves the iterative selection of samples

from the training dataset, allowing us to repeatedly refine a specific model. This iterative approach yields more insights into the fitted model, leading to improved accuracy and uncertainty estimation. One notable advantage of resampling is its ability to optimize model performance by repeatedly drawing samples from the same dataset, saving both time and resources compared to acquiring new data [6].

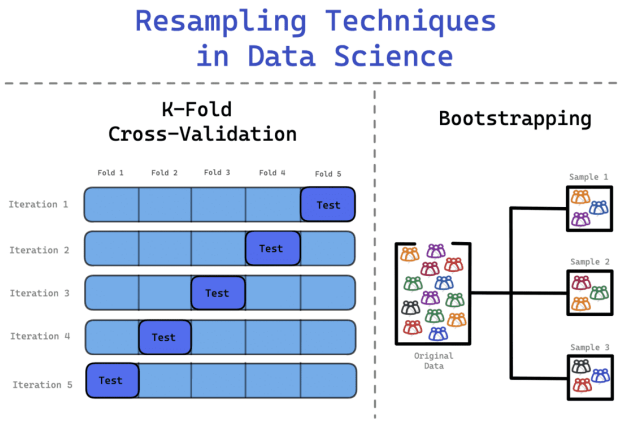


Figure 1: Resampling techniques allow for examination of how results may differ based on different samples. Left panel shows the cross-validation technique while the right panel shows the Bootstrapping technique ref. [7].

1. *Bootstrap resampling technique*

Resampling methods, particularly the Bootstrap method, play a pivotal role in data analysis, especially when dealing with datasets that deviate from the typical normal distribution. The Bootstrap method offers a non-parametric approach to statistical inference, replacing conventional distributional assumptions and asymptotic results with computational power.
Unlike traditional statistical methods that rely on specific distributional assumptions, Bootstrapping thrives in scenarios where data may not conform to standard behavior or when dealing with small sample sizes.
The core process of Bootstrapping involves repeatedly drawing sample observations from the dataset, allowing for replacements to maintain the original dataset's size. This means that an observation can appear multiple times or not at all in these drawn samples.
Furthermore, Bootstrapping comes to the rescue in cases where deriving sampling distributions for statistics is challenging, even asymptotically. Its simplicity also extends to complex data-collection plans, including stratified and clustered samples [6][8].

2. *Cross validation*

Cross-validation is a critical technique in data analysis, addressing the challenges of randomly splitting datasets, which can lead to imbalanced influences on model building and evaluation. To counter this issue, *k-fold cross-validation* introduces a structured approach that will be used in the course of this project.
In this method, the dataset is divided into k roughly equal-sized, exhaustive, and mutually exclusive subsets. During each iteration, one of these subsets serves as the test set, while the remaining subsets unite to form the training set. This structured division ensures that each sample has a balanced representation in both the training and test sets across all splits.
The core process of k-fold cross-validation involves iteratively training the model on $(k-1)$ subsets and testing it on the remaining set. This cycle continues until each subset has acted as the test set, ensuring that all data contributes to both model enhancement and performance assessment [6][8].

Relevance of the Central limit theorem

While the boostrap and cross-validation methods may offer a tidy and canny way of analyzing non-conforming data, it is worth to be noted that in accordance with richer quantities of re-samples the behaviour of the respective sample means will conform to a standard normal distribution. This is a direct consequence of what is called the central limit theorem which infers that one is able to statistically model the sample data under the assumption of being normally distributed albeit it not necessarily being so. The phenomenon will prove especially helpful in the analysis of the MSE, seeing as the residuals of the implemented model will, in accordance to, larger resamples behave normally. To contextualise it in more mathematical terms one can define it as.

Suppose we have a sequence of random statistical data $\{X_n\}$ that is identically and independently distributed with expectation $\mathbb{E}(X_i) = \mu$ and finite variance σ^2. Then as the number of samples n increase towards infinity one has that

$$\sqrt{n}(\bar{X}_n - \mu) \to \mathcal{N}(0, \sigma^2) \tag{20}$$

where $\bar{X}_n$ denotes the sample mean. [12]

III. METHOD

A. Franke´s function

Before attempting to fit polynomials to topographic data, we study the specific case of a two-dimensional

function named Franke´s function where $x, y \in [0, 1]$.

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

The function is a weighted sum of four exponential where $x, y \in [0, 1]$ and a widely used function for testing interpolation and fitting functions [11]. The Franke function is displayed in Fig. 2. Our aim is to test OLS (section II A 1), Ridge regression (sec. II A 2) and Lasso regression (section II A 3) on the function to assess our models. For OLS and Ridge regression, we use their analytical expression. As Lasso regression has no analytical solution we utilize the `scikit-learn` library.

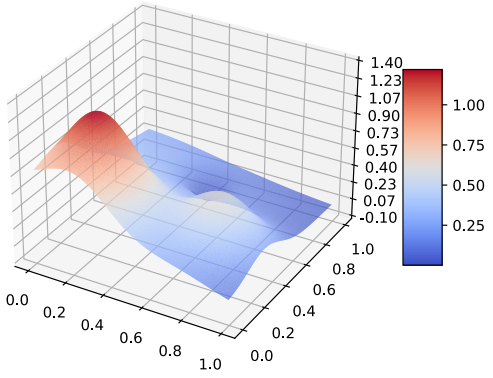


Figure 2: The figure displays the surface of the Franke function without noise. The figure is based on $200 \times 200 = 40000$ samples to create a smooth function.

A dataset of $20 \times 20 = 400$ samples based on the Franke function (eq. 1) was generated with added stochastic noise $\varepsilon \sim N(0, \sigma^2)$, where we use $\sigma = 0.1$.

Figure 3 shows the data, generated by Franke's function, that we will analyse with our regression models and methods.

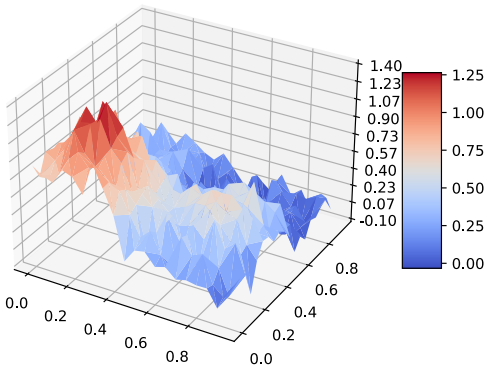


Figure 3: Surface plot of the Franke function. The figure is based on $20 \times 20 = 400$ samples.

B. Error analysis

A way to assess how "good" our models are, to look at their MSE and R^2 scores.

1. *MSE*

The mean squared error (MSE) measures the average of the squared errors [10] and is defined in equation 6. Ideally, we want the value to be zero as lower MSE scores correspond to a better fit.

2. *R^2*

The coefficient of determination R^2 provides a qualitative measure of the model. It is given by

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

where $\bar{y}$ is the mean value defined as

$$
\bar{y} = \frac{1}{n}\sum_{i=0}^{n-1} y_i \tag{3}
$$

The value ranges from 0 to 1 with 1 being the best possible score [9].

C. Splitting the data

In this study, we partitioned the dataset into training and test subsets to assess model performance on independent data. The `scikit-learn` `train_test_split`

function was employed for this purpose, with a fixed test data size of 20%, as a default throughout the paper.

D. λ parameters

Throughout the article we apply λ-values ranging from 10^{-8} to 10^{4} as a default. This is to have a wide and comprehensive span of parameters for testing and illustration purposes.

E. Scaling

To evaluate the necessity of scaling the design matrices of our regression model, and hence increase the potential of more diverse models, we introduced tests and generated relevant plots (i.e Appendix B: Figure 11).

By plot 11a we can observe that the scaling makes no significant difference in our predictions. For OLS, scaling is usually not needed. The coefficients directly represent the change in the dependent variable associated with a one-unit change in the corresponding independent variable. Multiplying X_i by a constant c simply leads to a scaling of the least squares coefficient estimates by a factor of $1/c$. However, numerically, we could have concerns about the limitations of the float numbers available on a computer. In cases with very large numbers, scaling should be used, also for OLS, to avoid such matters.

From plot 11b it can be discerned that the scaled implementation of Ridge has a lower average mean squared error. Ridge regression is a type of linear regression that penalizes large β coefficients. By scaling the design matrix, we ensure that the regularization term of the ridge regression, has an equal effect on all features. We scale the data to center it by subtracting the mean of each column of the design matrix. We also scale the predicted z values so that the predicted values align with the scaling of the design matrix.

The anticipated behavior should similarly apply to Lasso regression; nevertheless, the internal mechanisms of sklearn's Lasso model remain unclear. Figure 11c llustrates that scaling the design matrix has a negligible impact in this context.

In summary, we will not scale OLS and Lasso, but adopt a mean subtraction scaling approach for Ridge, both for the columns of the design matrix and predicted z values.

IV. RESULTS OF GENERIC DATA AND FRANKE'S FUNCTION

A. Ordinary Least Squares

The MSE and R^2 values using OLS are displayed for both training and test data up to the fifteenth-order polynomial in figure 5. As the polynomial degree increases,i.e. model complexity increases, the MSE and R^2 scores improve to a certain threshold. In figure 5 we can see that we obtain our best MSE and R^2-score for polynomial degree 8. Subsequently the predictions for our test set worsen, even though the training set predictions improve.

From figure 6 it is evident that the dispersion of the optimal parameter β_{OLS} increases as the polynomial degree increases.

B. Ridge and Lasso

In the case of Ridge and Lasso, the analysis is more complex due to the presence of the hyperparameter λ. Such parameters must be tuned in order to optimize the result of the regression. Table I shows the lowest MSE found paired with the relative λ and polynomial degree follows. More descriptive plots which the values in this table are taken from can be found in Appendix B.

We perform Ridge and Lasso regression, creating heatmaps to identify the optimal lambda and polynomial order that yield the most accurate prediction to this specific dataset.

Figures 12a and 12b reveal the lambda and polynomial order that yield optimal predictive results, as summarized in Table I.

Table I: Lowest MSE scores for test data using Lasso and Ridge regression and their corresponding λ value and polynomial degree.

Model	MSE	$\log_{10}(\lambda)$	Degree
OLS	0.013962	None	8
Ridge	0.014061	-8	9
Lasso	0.01923	-8	9

Lasso regression encounters computational inefficiency issues, with convergence warnings emerging during its execution. A solution to this problem can be achieved by increasing the tolerance or reducing the maximum iterations in the `sklearn.linear_model.Lasso`-method. Though this in turn results in elevated Mean Squared Error(MSE). A more favorable MSE, similar to Ridge and OLS, is achieved with `max_iter` = 10 000, albeit

at the cost of prolonged execution time. To attain a runtime comparable to Ridge, `max_iter` needs to be reduced to 100, as depicted in (Appendix B) Figure 13.

We can observe from figure 13a that we obtain the best MSE-value to be 0.015 running 10 000 iterations, while the best MSE-value in figure 13b, is 0.022, running 100 iterations. So as a default, we use `max_iter`= 1000.

C. Resampling

From the results from OLS, Ridge and Lasso, we found that OLS performed best. Hence we implement OLS for further analysis of resampling methods.

1. *Bias-Variance Tradeoff*

We apply an analysis using Bootstrap resampling method for OLS. Figure 8 plots MSE as a function of polynomial degree using $n = 100$ bootstrap samples.

2. *Cross Validation*

We apply an analysis using Cross-Validation resampling method for OLS. Figure 9 plots MSE as a function of polynomial degree using $k = 10$ folds.

In table II, we compare the MSE of the Bootstrap and Cross-Validation resampling methods.

Table II: Lowest MSE scores for test data using Bootstrap and Cross-Validation.

Method	MSE	Degree
Bootstrap	0.015610	7
Cross-Validation	0.012008	7

V. DISCUSSION: GENERIC DATA, FRANKE'S FUNCTION

A. Linear Regression Models

1. *Ordinary Least Squares regression*

When applying the Ordinary Least Squares (OLS) model to the data generated by Franke's function, an interesting trend emerges. As we increase the polynomial degree, we observe a consistent decrease in the Mean Squared Error (MSE), accompanied by a mirrored ascent in the R^2 index as shown in figure. This phenomenon can be attributed to the complexity of Franke's function. From the figure 5, we observe that the optimal polynomial for fitting Franke's function is represented by a 8th-order polynomial.

The underlying rational is that a more complex polynomial provides a better fit for this intricate function. However, it is crucial to exercise caution when elevating the polynomial order excessively. This is because an excessively high polynomial order often leads to a common issue known as overfitting. In figure 5 we observe that for polynomial degrees > 8, the MSE increases.

Overfitting occurs when the model becomes overly adaptable and starts to lose its ability to discern the general patterns within the data. Instead, it becomes overly focused on capturing noise or fluctuations present in the dataset. Therefore, while increasing the polynomial degree can yield improved performance, striking the right balance is essential to prevent overfitting and ensure that the general patterns of the model are captured.

Another important result shown in figure 6 is how the spread of β values increase significantly at the increase of the polynomial order. This is mainly due to:

- Overfitting: Higher complexity (e.g., using high-degree polynomials) fits training data too closely, capturing noise. It leads to enhanced fitting of test data but poor generalization.

- Increased Variability: Complex models are sensitive to minor input changes, causing parameter estimates to vary significantly between data subsets or new input data. Confidence intervals widen as a result.

- Multicollinearity: High complexity may introduce multicollinearity, where input variables are strongly correlated. This complicates coefficient estimation, increasing parameter uncertainty.

2. *Ridge and Lasso regression*

Through observations of the MSE heatplots presented in figure 12a and figure 12b, it is credible to address that the respective polynomial orders (i.e 9, 9) in combination with the designated hyperparameter λ (10^{-8}) will, relative to the test data, yield the most optimal fit of Franke's function. For Ridge and Lasso regression, the best λ-parameter is 10^{-8}. This is the lowest λ-parameter we used. This looks like the best result for Ridge and Lasso is when they approach the method for doing OLS. For Ridge, when λ approaches 0, the equation 14 for Ridge becomes the equation II A 1 for OLS. This fits with our results in table I, telling us that OLS fits ouur data the best.

B. Bias-Variance trade-off

In our application of the bias-variance trade-off, we have observed the fulfillment of Equation 19 as elucidated in the theoretical framework. Examination of Figure 7 reveals a noteworthy trend: when employing a less complex model, the variance remains consistently low, but it tends to escalate as model complexity increases, indicative of overfitting.

Conversely, both error and bias exhibit initial values greater than zero. Aligning with the equation mentioned earlier, at lower model complexities, they become nearly indistinguishable due to the substantially diminished variance. However, as model complexity grows and overfitting becomes more pronounced, a noticeable divergence between error and bias emerges dictated by the increasing variance.

The observed reduction in bias with increasing model complexity is attributed to the enhanced flexibility and adaptability of the chosen fitting function, consequently diminishing bias. Furthermore, choosing appropriate scaling in accordance to the data was found to increase the bias of the estimation (and especially in the cases where the quantity of data were $\leq 20\%$ of the model), but simultaneously decreasing the variance due to a higher concentration of points being clustered around a common mean.

With that in mind, an analysis was henceforth conducted in plotting error, bias, and variance as a function of the numbers of bootstraps; using the best polynomial degree equal to 7 in order to find the best number of bootstrap samples for the purpose of obtaining the most accurate fit. In this exact data set, the best number of bootstraps is 65, as seen in figure 8. The error scores exhibit a decrease as the number of bootstrap samples are increased until reaching a point of stability, beyond which further reductions in error values were not observed. It can be deduced that employing a number of bootstrap samples significantly greater than 100 is essentially unnecessary, as the mean squared error (MSE) ceases to decrease further. Instead, it results in increased computational costs without providing substantial improvement in MSE reduction.

C. Bootstrap and Cross Validation

An analysis of Franke's function was ran using the Cross-Validation method to enhance our Ordinary Least Squares (OLS) model and minimize the Mean Squared Error (MSE). The MSE was plotted against both the polynomial order (Figure 9) and the number of K-folds (Figure 10). The first plot reveals that the optimal polynomial degree for fitting Franke's function remains at the previously determined 7th order polynomial. Additionally, from Figure 10, it becomes evident that $k = 10$ folds is the optimal number of folds for achieving the most optimal fit.

From table II we observe that our Cross-Validation method using $k = 10$ folds yields a lower MSE than our OLS method. However the Bootstrap resampling method yields a higher MSE than our OLS method. This did not align exactly with our initial expectations, since Bootstrap performed significantly worse than our OLS method. Bootstrap resampling can lead to overfitting, particularly when the dataset is small. When we repeatedly sample with replacement, we introduce randomness into your training data, potentially including outliers and noise. Overfitting can result in models that fit the random noise in the data, leading to poor generalization and higher MSE on unseen data.

VI. TERRAIN DATA

We have chosen to conduct our real data analysis of the region in Møstvatn Austfjell. The terrain can be seen in the Figure 4.

A. Handling terrain data

The size of the terrain data set is 6485401 points. As a consequence, we had to downsample the terrain data. We chose to do this by selecting every N-th row and column. We opted for $N = 85$, resulting in $\frac{6485401}{85 \cdot 85} \approx 897$ data points. Despite downsampling, we retained a 20% test data size, amounting to around 175 data points..

The regressions are done using polynomial orders from 1 to 25, and λ-parameters in $\log_{10}$ from -8 to 4.

Due to the vast range of altitude values within the terrain dataset, spanning from minuscule to substantial magnitudes, the application of a Min-Max scaler becomes particularly advantageous. We do this because it is easier defining x and y, but also to try to avoid too large numbers in our design matrix, potentially resulting in overflow. First we create x and y in a uniform range between 0 and 1, then we scale z using `sklearn`'s `MinMaxScaler`, so that the terrain data is also in the range $(0, 1)$. This scaling was applied during the data sampling process and remains consistent throughout all subsequent analyses.

VII. RESULTS WITH TERRAIN DATA

A. Ordinary Least Squared regression

The MSE and R2 values using OLS are displayed in figure 14. As the polynomial degree increases, i.e. model complexity increases, the MSE and R2 scores improve to a certain threshold. From figure 14 we can see that we obtain our best MSE and R2-score for polynomial degree 15. After this, the predictions of our test set get worse, even though the training set predictions improve.

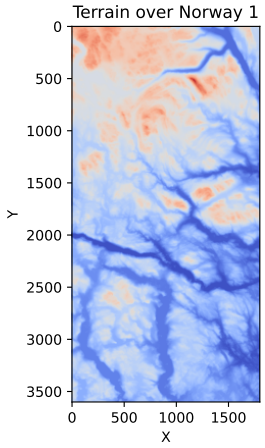


Figure 4: Two-dimensional plot of the region in Møstvatn Austfjell.

B. Ridge and Lasso regression

For the cases of Ridge and Lasso regression, the analysis is as mentioned more complex due to the impact of the hyperparameter λ. Similarly to the instance of modelling generic data to Franke's function this has to be finely tuned in order to optimise the regression fitting capabilities.
Table IV shows the lowest MSE using OLS, Ridge and Lasso. The plots of these results are heatmaps of the MSE of Ridge, found in figure 15a, and of the MSE of Lasso, found in figure 15b.

Table III: Lowest MSE scores for test data using OLS, Ridge and Lasso regression, along with their corresponding λ-parameters and polynomial degree.

Model	MSE	$\log_{10}(\lambda)$	Degree
OLS	0.032656	None	15
Ridge	0.034900	-8	25
Lasso	0.038870	-6	25

C. Bias Variance Trade-off

The Bias Variance Trade-off analysis has likewise been conducted on the terrain data. We study the magnitude of error, variance and bias as a function of the polynomial order as shown in Figure 16

In like manner of the Bias-Variance analysis presented for generic data, further analysis was conducted to study the evolution of error as a function of the number of bootstrap samples. The resulting plot is displayed in Figure 17.

D. Cross Validation

In this section, we provide an analysis of mean squared errors (MSE) conducted through cross-validation resampling with ordinary least squares (OLS) regression. The primary focus is, as earlier, on measures of error relative to polynomial order, as depicted in Figure 18.

Table IV presents a comparison of Bootstrap and Cross-Validation, in terms of their Mean Squared Error (MSE) values when applied to Ordinary Least Squares (OLS) regression. The analysis focuses on how the MSE changes with varying model complexity. The table showcases results obtained from $n = 100$ bootstrap samples and $k = 10$ folds for Cross-Validation. The MSE is plotted as a function of polynomial degree for Bootstrap in figure 16 and for Cross-Validation in figure 18.

Table IV: Results from bootstrap and cross-validation using terrain data.

Method	MSE	Degree
Bootstrap	0.040989	7
Cross-Validation	0.029772	14

VIII. DISCUSSION: TERRAIN DATA

The terrain image comprises an extensive dataset with over 6 million data points, rendering the computational demands of our regression algorithm highly resource-intensive. To alleviate this computational burden, we adopted a deterministic sampling approach. This entailed selecting every Nth element along both the rows and columns of the data matrix. We made a deliberate choice to avoid random sampling, as this decision facilitates reproducibility, particularly in situations where debugging is necessary. However, we observed a certain sensitivity when plotting our data. Despite using the same dataset for sampling, the random nature of the train-test data split can significantly influence the resulting plots, especially with a low number of points sampled. This sensitivity in the train-test split can be effectively reduced by increasing the number of initially sampled data points. Nevertheless, an excessive increase in the number of sampled data points not only burdens the algorithm in terms of computation but also diminishes the overfitting effect, which is a key aspect of our analysis in this project. Therefore, we identified an optimal balance at $N = 85$, which yields a dataset containing 897 data points for our analysis.

A. Linear regression model

1. Ordinary Least Squares regression

When applying the OLS regression method to real data, an evident trend emerges: as the polynomial degree increases, a consistent decrease in the Mean Squared Error (MSE) is observed until a certain point where it starts increasing again. This phenomenon echoes the discussion on overfitting, as previously elaborated in the section on Franke's function. The graphical representation in Figure 14 distinctly illustrates that the optimal polynomial degree for fitting the terrain dataset is the 15th order.

2. Ridge and Lasso regression

Examining the heatmap 15a for Ridge and the heatmap 15b for Lasso, allows us to identify the most suitable polynomial for fitting the terrain dataset using Ridge or Lasso regression. This choice is influenced by the λ-parameter, which has been elaborated upon in the following section (i.e Table IV). We can see from this table that OLS makes the best method for fitting the terrain data.

B. Bias-Variance trade-off

During our study of the bias-variance trade-off, we have empirically validated the applicability of Equation 19, as outlined in our theoretical framework, even when dealing with real data from the terrain dataset. Upon a close examination of Figure 16, a notable tendency becomes apparent, likewise as in the case of generic data: as we opt for simpler models, the variance consistently remains low. Yet, with more complex models, variance increases, suggesting overfitting.

Conversely, error and bias both initially start above zero. With simpler model fittings, variance decreases, making it hard to distinguish between error and bias. However, as complexity grows, thus leading to substantial overfitting, the gap between error and bias becomes more pronounced, primarily due to increasing variance.

The decline in bias with increasing model complexity can be attributed to the heightened adaptability and flexibility afforded by our chosen fitting function. Our analysis explored error, bias, and variance in relation to the number of bootstrap samples. This analysis was carried out while maintaining the optimal polynomial degree at 7, aiming to identify the ideal number of bootstrap samples for achieving the best fit. As depicted in Figure 17, the error first increases to about 40 bootstrap samples and them decrease steadily with the rising number of bootstrap samples. This observation implies that employing a significantly larger number of bootstrap samples than 100 is an unwished-for choice, as it fails to yield substantial reductions in MSE while resulting in heightened computational costs. The reason for better MSE at low number of bootstrap samples can be explained by the introduction of randomness into your training data, making bootstrap resampling unreliable for such a small data set.

C. Bootstrap and Cross-Validation

We utilized Cross-Validation to refine the polynomial degree in our OLS model for the terrain dataset, optimizing it from 15 to 14, as illustrated in Figure 18. Furthermore, exploring K-folds indicated that $k = 10$ yields the optimal fit, as shown in Figure 19.

Our approach shifted when we combined bootstrap and cross-validation to minimize MSE in combination with OLS. The results, shown in table IV, deviated from our initial expectations.

The values for Bootstrap exceeded the MSE obtained through regular OLS. It becomes apparent that Cross-Validation exhibits greater stability compared to Bootstrap. We conclude that Cross-Validation is a more resilient method for model evaluation.

IX. CONCLUSION

In this research project, our primary objective was to investigate the sufficiency of linear regression methods, specifically OLS, Ridge, and LASSO, in fitting a nth-order polynomial function to the given data. We employed test-MSE measurements on both generic and actual terrain data and integrated resampling techniques, such as cross validation and bootstrap, to assess the quality of these fits.

Our findings revealed that OLS regression consistently yielded slightly better results when compared to Ridge and LASSO for both generic and real-world terrain data. While the performance difference between OLS and Ridge was not substantial, the slight advantage of OLS, coupled with its less demanding implementation, suggests that OLS is an advisable choice for such applications.

It is essential to emphasize that the fitting of real-world terrain data presented additional challenges due to its inherent irregularity. Given the highly varying landscape, the adoption of spline-based methods could be recommended. Such approaches allow for the segmentation of the terrain into subregions, enabling more accurate regressions within these segmented areas.

Moreover, a note of caution is advised. The values and results presented in this report should not be understood as precise measurements. This limitation arises from constraints related to computational resources, the use of rudimentary approaches, and a lack of comprehensive probability analysis concerning the produced estimates.

Instead, they should be viewed as valuable insights into the application of regression models, providing a foundation for further research and analysis in this domain.

X. REFERENCES

[1] Stanton, J. M. (2017). *Galton, Pearson, and the Peas: A Brief History of Linear Regression for Statistics Instructors*. Journal of Statistic Education, **9:3**, 1-2. DOI

[2] Jensen, M. H. (n.d.). *Week 34: Introduction to the course, Logistics and Practicalities*. Applied Data Analysis and Machine Learning. Retrieved September 22, 2023, from Jupyter book

[3] Jensen, M. H. (n.d.). *Week 35: From Ordinary Linear Regression to Ridge and Lasso Regression*. Applied Data Analysis and Machine Learning. Retrieved September 22, 2023, from Jupyter book

[4] Jensen, M. H. (n.d.). *Week 36: Statistical interpretation of Linear Regression and Resampling techniques*. Applied Data Analysis and Machine Learning. Retrieved September 22, 2023, from Jupyter book

[5] Jensen, M. H. (n.d.). *Week 37: Statistical Interpretations and Resampling Methods*. Applied Data Analysis and Machine Learning. Retrieved September 22, 2023, from Jupyter book

[6] Arya, N. (2023, February 20). *The Role of Resampling Techniques in Data Science*. KDnuggets. Retrieved September 23, 2023, from [Website]

[7] Arya, N. (2021). *Resampling Techniques in Data Science*. KDnuggets. [Photograph]

[8] Jensen, M. H. (n.d.). *Week 37: Statistical Interpretations and Resampling Methods*. Applied Data Analysis and Machine Learning. Retrieved September 8, 2023, from Jupyter book

[9] Jensen, M. H. (n.d.). *Week 34: Introduction to the course, Logistics and Practicalities*. Applied Data Analysis and Machine Learning. Retrieved September 8, 2023, from Jupyter book

[10] *Mean squared error*. (2023, August 15). In Wikipedia. URL

[11] Jensen, M. H. (n.d.). *Project 1 on Machine Learning, deadline October 9 (midnight)*, 2023. Applied Data Analysis and Machine Learning. Retrieved September 7, 2023, from Jupyter Book

[12] *Central Limit Theorem*. (This page was last edited on 27 September 2023, at 23:34 (UTC).)In Wikipedia. URL

Appendix A: Statistics

1. Expectation value of y_i

$$\mathbb{E}(y_i) = \sum_j x_{ij}\beta_j = \boldsymbol{X}_{i,*}\boldsymbol{\beta} \tag{A1}$$

The result follows from the fact that $\boldsymbol{X}_{i,*}\boldsymbol{\beta}$ is non-stochastic.

2. Variance of y_i

$$\begin{aligned}
\mathrm{Var}(y_i) &= \mathbb{E}\{[y_i - \mathbb{E}(y_i)^2]\} = \mathbb{E}(\underbrace{y_i^2}_{(eq.3)^2}) - [\underbrace{\mathbb{E}(y_i)}_{eq.A1}]^2 \\
&= \mathbb{E}[(\boldsymbol{X}_{i,*}\boldsymbol{\beta} + \varepsilon_i)^2] - (\boldsymbol{X}_{i,*}\boldsymbol{\beta})^2 \\
&= \mathbb{E}[(\boldsymbol{X}_{i,*}\boldsymbol{\beta})^2 + 2\varepsilon_i\boldsymbol{X}_{i,*}\boldsymbol{\beta} + \varepsilon_i^2] - (\boldsymbol{X}_{i,*}\boldsymbol{\beta})^2 \\
&= \cancel{(\boldsymbol{X}_{i,*}\boldsymbol{\beta})^2} + 2\underbrace{\mathbb{E}(\varepsilon_i)}_{=0}\boldsymbol{X}_{i,*}\boldsymbol{\beta} + \mathbb{E}(\varepsilon_i^2) - \cancel{(\boldsymbol{X}_{i,*}\boldsymbol{\beta})^2} \\
&= \mathbb{E}(\varepsilon_i^2) = \mathrm{Var}(\varepsilon_i) = \sigma^2
\end{aligned} \tag{A2}$$

3. Expectation value $\hat{\beta}$

$$\begin{aligned}
\mathbb{E}(\hat{\boldsymbol{\beta}}) &= \mathbb{E}[(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}] \\
&= (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\underbrace{\mathbb{E}(\boldsymbol{y})}_{eq.A1} \\
&= (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\beta} \\
&= \boldsymbol{\beta}
\end{aligned} \tag{A3}$$

The estimator of the regression parameters is unbiased meaning the expected value is equal to the true value, i.e. the difference between $\hat{\beta}$ and β is zero.

4. Variance of $\hat{\beta}$

$$\begin{aligned}
\mathrm{Var}(\hat{\boldsymbol{\beta}}) &= \mathbb{E}\{[\boldsymbol{\beta} - \underbrace{\mathbb{E}(\boldsymbol{\beta})}_{eq.A3}][\boldsymbol{\beta} - \underbrace{\mathbb{E}(\boldsymbol{\beta})}_{eq.A3}]^T\} \\
&= \mathbb{E}\{[(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y} - \boldsymbol{\beta}][(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y} - \boldsymbol{\beta}]^T\} \\
&= (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\mathbb{E}[\boldsymbol{y}\boldsymbol{y}^T]\boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X})^{-1} - \boldsymbol{\beta}\boldsymbol{\beta}^T
\end{aligned} \tag{A4}$$

Before we proceed, we calculate $\mathbb{E}[\boldsymbol{y}\boldsymbol{y}^T]$

$$\begin{aligned}\mathbb{E}[\boldsymbol{y}\boldsymbol{y}^T] &= \mathbb{E}[(\boldsymbol{X}\boldsymbol{\beta}+\boldsymbol{\varepsilon})(\boldsymbol{X}\boldsymbol{\beta}+\boldsymbol{\varepsilon})^T] \\ &= \mathbb{E}[\boldsymbol{X}\boldsymbol{\beta}\boldsymbol{\beta}^T\boldsymbol{X}^T + 2\boldsymbol{X}\boldsymbol{\beta}\underbrace{\boldsymbol{\varepsilon}}_{=\mathbf{0}} + \underbrace{\boldsymbol{\varepsilon}^2}_{=\sigma^2}] \\ &= \boldsymbol{X}\boldsymbol{\beta}\boldsymbol{\beta}^T\boldsymbol{X}^T + \sigma^2\end{aligned} \tag{A5}$$

Inserting eq. A5 in eq. A4

$$\begin{aligned}\mathrm{Var}(\hat{\boldsymbol{\beta}}) &= (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T[\boldsymbol{X}\boldsymbol{\beta}\boldsymbol{\beta}^T\boldsymbol{X}^T + \sigma^2]\boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X})^{-1} - \boldsymbol{\beta}\boldsymbol{\beta}^T \\ &= \boldsymbol{\beta}\boldsymbol{\beta}^T + \sigma^2(\boldsymbol{X}^T\boldsymbol{X})^{-1} - \boldsymbol{\beta}\boldsymbol{\beta}^T \\ &= \sigma^2(\boldsymbol{X}^T\boldsymbol{X})^{-1}\end{aligned} \tag{A6}$$

5. Bias-Variance tradeoff

$$\begin{aligned}\mathbb{E}[(\boldsymbol{y}-\tilde{\boldsymbol{y}})^2] &= \mathbb{E}[(\boldsymbol{f}+\boldsymbol{\varepsilon}-\tilde{\boldsymbol{y}})^2] \\ &= \mathbb{E}[(\boldsymbol{f}-\tilde{\boldsymbol{y}})+\boldsymbol{\varepsilon})^2] \\ &= \mathbb{E}[(\boldsymbol{f}-\tilde{\boldsymbol{y}})^2 + 2(\boldsymbol{f}-\tilde{\boldsymbol{y}})\boldsymbol{\varepsilon} + \boldsymbol{\varepsilon}^2] \\ &= \mathbb{E}[(\boldsymbol{f}-\tilde{\boldsymbol{y}})^2] + 2\mathbb{E}[(\boldsymbol{f}-\tilde{\boldsymbol{y}})\underbrace{\boldsymbol{\varepsilon}}_{=0}] + \underbrace{\mathbb{E}[\boldsymbol{\varepsilon}^2]}_{=\sigma^2} \\ &= \mathbb{E}[(\boldsymbol{f}-\tilde{\boldsymbol{y}})^2] + \sigma^2\end{aligned} \tag{A7}$$

By adding and subtracting $\mathbb{E}[\tilde{\boldsymbol{y}}]$ we obtain

$$\begin{aligned}\mathbb{E}[(\boldsymbol{y}-\tilde{\boldsymbol{y}})^2] &= \mathbb{E}[(\boldsymbol{f}-\tilde{\boldsymbol{y}}+\mathbb{E}[\tilde{\boldsymbol{y}}]-\mathbb{E}[\tilde{\boldsymbol{y}}])^2] + \sigma^2 \\ &= \mathbb{E}[((\boldsymbol{f}-\mathbb{E}[\tilde{\boldsymbol{y}}]) + (\mathbb{E}[\tilde{\boldsymbol{y}}]-\tilde{\boldsymbol{y}}))^2] + \sigma^2 \\ &= \mathbb{E}[(\boldsymbol{f}-\mathbb{E}[\tilde{\boldsymbol{y}}])^2 + 2(\boldsymbol{f}-\mathbb{E}[\tilde{\boldsymbol{y}}])(\mathbb{E}[\tilde{\boldsymbol{y}}]-\tilde{\boldsymbol{y}}) + (\mathbb{E}[\tilde{\boldsymbol{y}}]-\tilde{\boldsymbol{y}})^2] + \sigma^2 \\ &= \mathbb{E}[(\boldsymbol{f}-\mathbb{E}[\tilde{\boldsymbol{y}}])^2 + (\mathbb{E}[\tilde{\boldsymbol{y}}]-\tilde{\boldsymbol{y}})^2] + 2\mathbb{E}[(\boldsymbol{f}-\mathbb{E}[\tilde{\boldsymbol{y}}])(\mathbb{E}[\tilde{\boldsymbol{y}}]-\tilde{\boldsymbol{y}})] + \sigma^2\end{aligned} \tag{A8}$$

where

$$\begin{aligned}\mathbb{E}[(\boldsymbol{f}-\mathbb{E}[\tilde{\boldsymbol{y}}])(-\tilde{\boldsymbol{y}})] &= \mathbb{E}[\boldsymbol{f}\mathbb{E}[\tilde{\boldsymbol{y}}] - \boldsymbol{f}\tilde{\boldsymbol{y}} - \mathbb{E}[\tilde{\boldsymbol{y}}]^2 + \tilde{\boldsymbol{y}}\mathbb{E}[\tilde{\boldsymbol{y}}]] \\ &= \boldsymbol{f}\mathbb{E}[\tilde{\boldsymbol{y}}] - \boldsymbol{f}\mathbb{E}[\tilde{\boldsymbol{y}}] - \mathbb{E}[\tilde{\boldsymbol{y}}]^2 + \mathbb{E}[\tilde{\boldsymbol{y}}]^2 \\ &= 0\end{aligned} \tag{A9}$$

Thus, we are left with

$$\begin{aligned}\mathbb{E}[(\boldsymbol{y}-\tilde{\boldsymbol{y}})^2] &= \mathbb{E}[(\boldsymbol{f}-\mathbb{E}[\tilde{\boldsymbol{y}}])^2 + (\mathbb{E}[\tilde{\boldsymbol{y}}]-\tilde{\boldsymbol{y}})^2] + \sigma^2 \\ &= \underbrace{\mathbb{E}[(\boldsymbol{f}-\mathbb{E}[\tilde{\boldsymbol{y}}])^2]}_{\mathrm{Bias}[\tilde{y}]} + \underbrace{\mathbb{E}[(\tilde{\boldsymbol{y}}-\mathbb{E}[\tilde{\boldsymbol{y}}])^2]}_{\mathrm{Var}[\tilde{y}]} + \sigma^2 \\ &= \mathrm{Bias}[\tilde{y}] + \mathrm{Var}[\tilde{y}] + \sigma^2\end{aligned} \tag{A10}$$

Appendix B: Figures

1. Franke's function

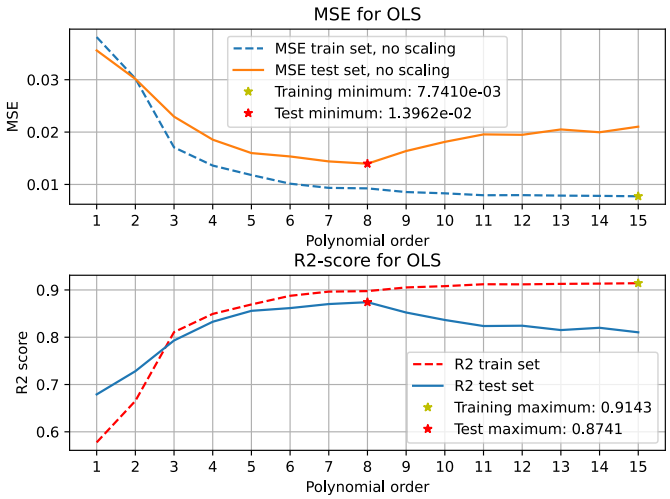


Figure 5: Upper panel: MSE as a function of polynomial order using OLS. Lower panel: R^2- score as a function of polynomial order using OLS.

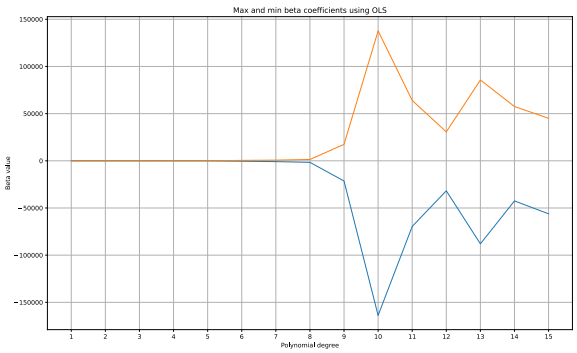


Figure 6: Minimum and maximum β_{OLS} coefficients as a function of polynomial order.

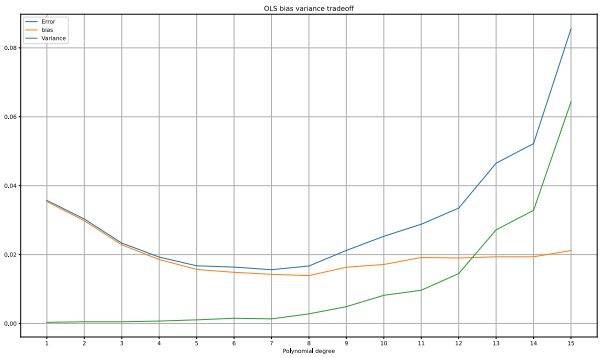


Figure 7: Error, bias, and variance as a function of model complexity using OLS of the Franke function.

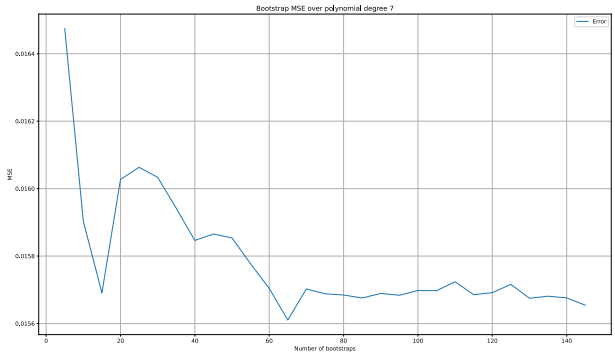


Figure 8: MSE using Bootstrap over the number of bootstraps samples for the optimal polynomial degree found in figure 7.

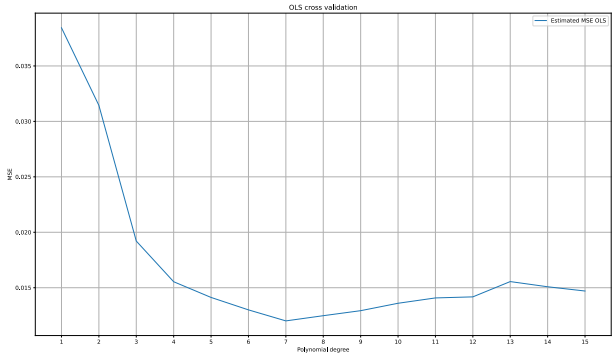


Figure 9: MSE using Cross Validation as a function of the polynomial order with $k = 10$ folds.

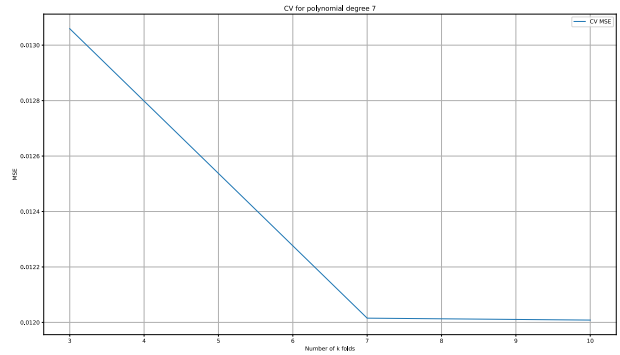
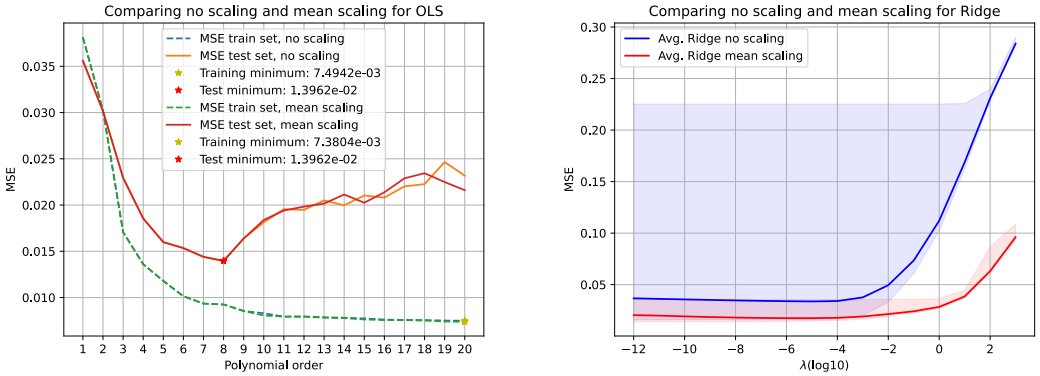
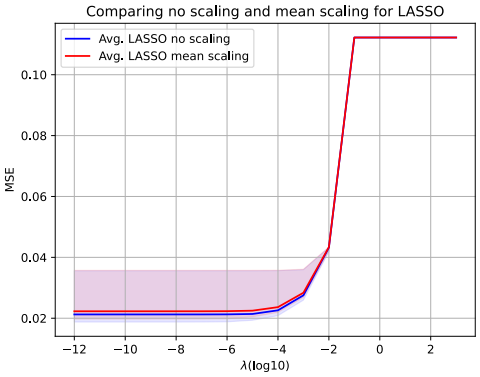


Figure 10: MSE using Cross Validation as a function of the number of K-Folds for the optimal polynomial degree found in figure 9, testing $k = [3, 7, 10]$.



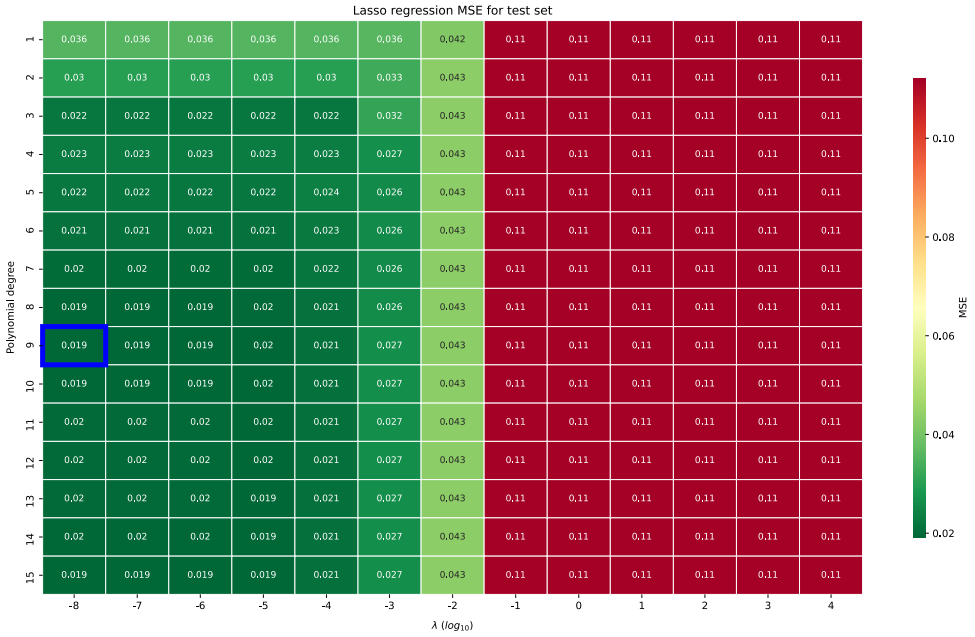
(a) OLS **(b)** Ridge



(c) LASSO

Figure 11: Comparing no scaling with scaling by subtracting the mean for the different regression methods. 11a plots the MSE of the test and training sets of both scaling and not scaling. Figure 11b plots the average MSE over the all the polynomial degrees for each λ, comparing scaling and not scaling. Figure 11c plots the same for lasso regression.

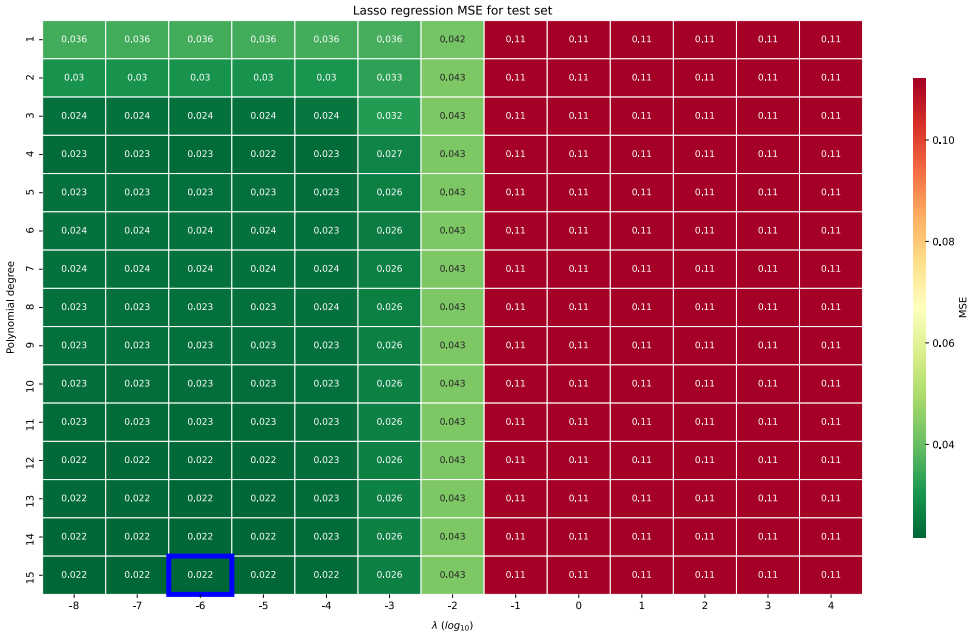
(a) Heatmap of MSE using Ridge on data from Franke's function.



(b) Heatmap of MSE using Lasso on data from Franke's function.

Figure 12

(a) Heatmap of MSE using Lasso with `max_iter`=10000.



(b) Heatmap of MSE using Lasso with `max_iter`=100.

Figure 13

2. Terrain data

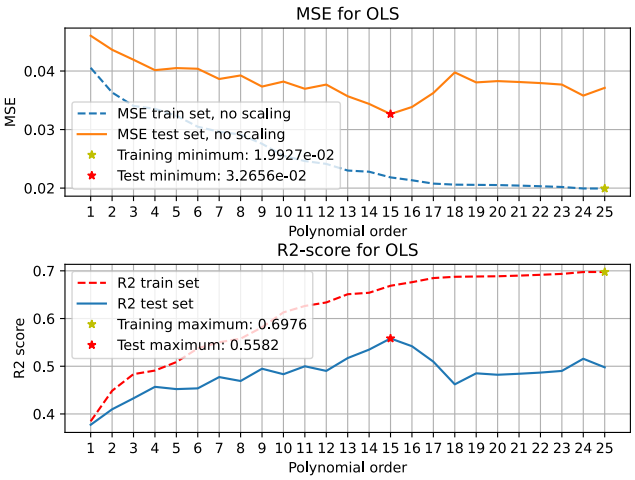
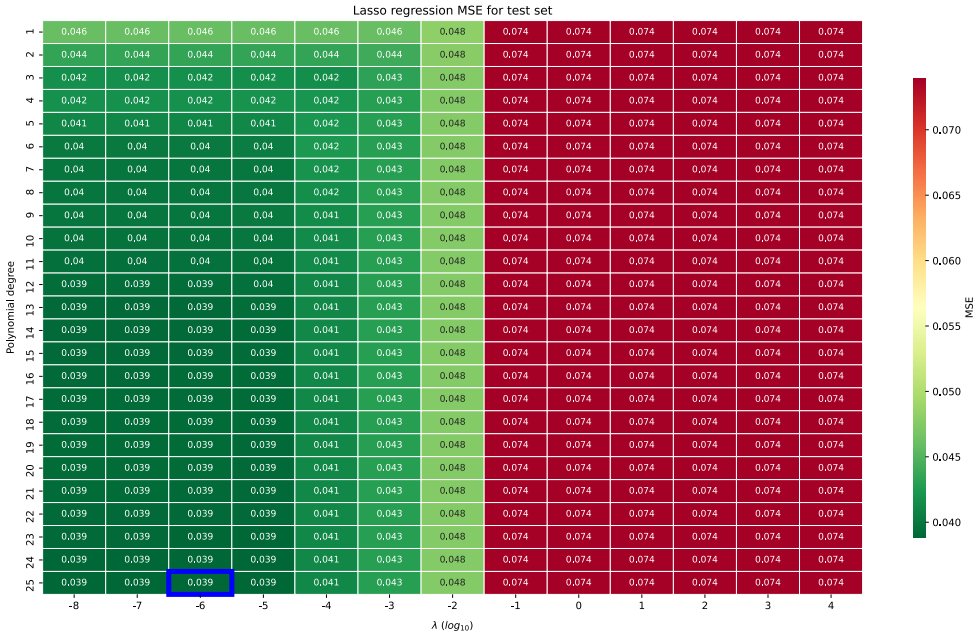


Figure 14: Upper panel: MSE as a function of polynomial order using OLS. Lower panel: R^2- score as a function of polynomial order using OLS.

(a) Heatmap of MSE using Ridge on terrain data.



(b) Heatmap of MSE using Lasso on terrain data.

Figure 15

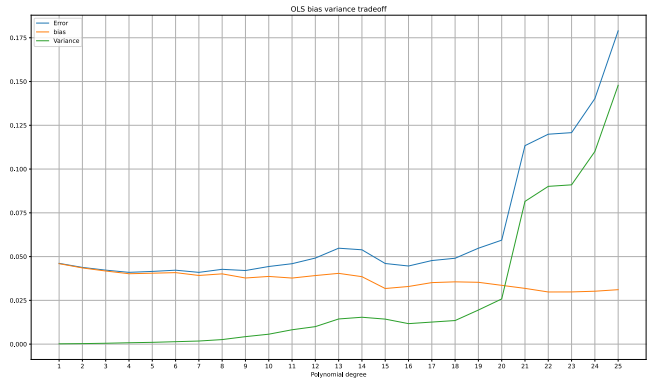


Figure 16: Values of error, variance and bias as a function of the polynomial degree with $n = 100$ bootstrap samples.

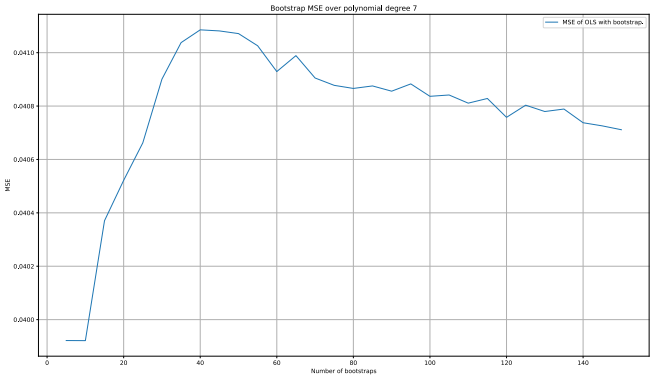


Figure 17: MSE using bootstrap plotted as a function of the number of bootstrap for the optimal polynomial degree found in figure 16.

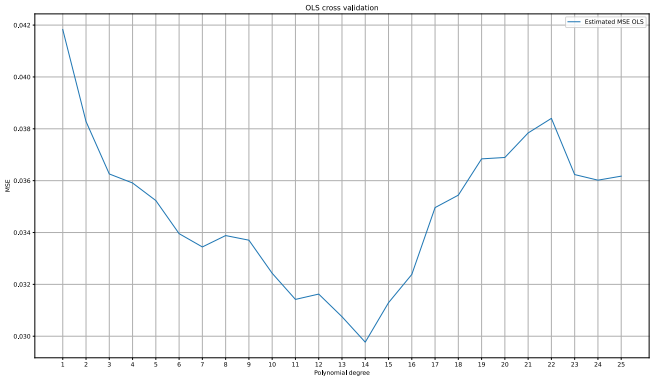


Figure 18: MSE using Cross Validation as a function of the polynomial order with $k = 10$ folds.

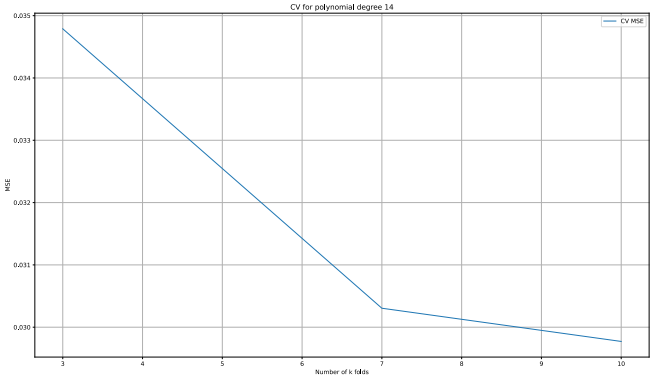


Figure 19: MSE using Cross Validation as a function of the number of K-Folds for the optimal polynomial degree found in figure 18, testing $k = [3, 7, 10]$.

Appendix C: Github repository

Code for the regression models, test runs and figure plotting available at following github repository:

- https://github.uio.no/mathihs/Project-1-Regression-analysis-and-resampling-methods/tree/master