Project 1

Isak O. Rukan[*] and Edvard B. Rørnes[†]
Institute of Physics, University of Oslo,
0371 Oslo, Norway
(Dated: October 2, 2024)

We investigate and compare three regression models[a]: Ordinary Least Squares, Ridge, and Least Absolute Shrinkage and Selection Operator. These were applied and analyzed on a two-dimensional Franke function and later tested on real terrain data from the US Geological Survey Earth Explorer [1]. The models' performances are evaluated using metrics such as the Mean Squared Error and the coefficient of determination R^2. The bias-variance trade-off for OLS is studied in detail where it was found that ... (TBD). We used resampling techniques such as bootstrapping and cross-validation to probe the quality of the evaluations and determine the predictiveness and generalizability of the models. In our analysis the best performing method was ... (TBD).

1. INTRODUCTION

Regression models are essential tools in data analysis and prediction, particularly within the realm of physics. They are used to enable understanding of the relationships between different variables and improve our ability to create predictions. In this report we study and compare three common regression techniques: Ordinary Least Squares (OLS) regression, Ridge regression and Least Absolute Shrinkage and Selection Operator (LASSO) regression. Each come with their unique strengths and weaknesses that make them suitable for different data sets.

OLS is the simplest and most foundational method which estimates relationship by minimizing the difference between observed and predicted values. A large downside with OLS is when there are many related variables as this can lead to unstable coefficient estimates [2]. Ridge regression can partially fix this issue by adding a penalty to large coefficients via a regulator λ, effectively allowing us to shrink the coefficients with a new input parameter. This in turn creates a more stable model in the event of correlated variables. LASSO regression takes this a step further by once again shrinking coefficients, but also setting some of them to zero. This allows LASSO to effectively choose important variables, which may be helpful when pursuing a simpler model.

To compare these regression model, we use a two-dimensional Franke function which allows us to test the performance of each under controlled conditions. Later we apply each model to real terrain data which is obtained from [1].

The models are evaluated by considering their Mean Squared Error (MSE) and coefficient of determination R^2. Further to make our evaluation more reliable, resampling techniques such as bootstrapping and cross-validation are used. These methods provide a quantitative description of how well each model performs on different data sets.

2. THEORY

All claims made in this section which require relatively lengthy derivations are derived in Appendix A. The general structure of all our models is that we have some data set $\{x_i, y_i\}$ where $i \in \{0, 1..., n-1\}$ where x_i are independent variables whilst y_i are dependent variables. The data is assumed to be described by

$$\boldsymbol{y} = f(\boldsymbol{x}) + \boldsymbol{\varepsilon} \tag{1}$$

where f is some continuous function which takes $\boldsymbol{x}$ as input and $\boldsymbol{\varepsilon}$ is a normal distributed error $\boldsymbol{\varepsilon} \sim \mathcal{N}(0, \sigma^2)$. The function f will then be approximated with a model $\tilde{\boldsymbol{y}}$ in which we will consider a polynomial expansion with coefficients β_i:

$$\tilde{y}_i = \sum_{j=0}^{p-1} \beta_j x_i^j \tag{2}$$

defining the $n \times p$ design matrix $\boldsymbol{X}$ with elements $X_{ij} = (x_i)^j$ we can rewrite this as

$$\tilde{\boldsymbol{y}} = \boldsymbol{X}\boldsymbol{\beta} \tag{3}$$

Further, each model will be defined with a different cost function $C(\boldsymbol{\beta})$ which we minimize to find the coefficients for each respective model.

The metrics MSE and R^2 which we use to analyze each model are defined by

$$\mathrm{MSE}(\boldsymbol{y}, \tilde{\boldsymbol{y}}) \equiv \frac{1}{n}\sum_{i=0}^{n-1}(y_i - \tilde{y}_i)^2, \tag{4}$$

$$R^2 \equiv 1 - \frac{\sum_{i=0}^{n-1}(y_i - \tilde{y}_i)^2}{\sum_{i=0}^{n-1}(y_i - \bar{y})^2} \tag{5}$$

where $\bar{y} \equiv \frac{1}{n}\sum_{i=0}^{n-1} y_i$ is the mean value of $\boldsymbol{y}$. Here we can see that if $\tilde{\boldsymbol{y}} = \boldsymbol{y}$ then $R^2 = 1$ implies that we have

[*] Insert Email
[†] e.b.rornes@fys.uio.no
[a] GitHub Repository: `https://github.com/EdvardRornes/FYS-STK4155/tree/main/Project1`

perfect accuracy. Further if $\tilde{\boldsymbol{y}}$ is simply the mean then $R^2 = 0$ implying that our fit is equally bad as a single line and if $R^2 < 0$ it means that our approximation is worse than simply taking the mean.

2.1. OLS

OLS is a primitive method used in linear regression to estimate coefficients of a linear model. The cost function in OLS is simply defined as the residual sum of squares (RSS)

$$C_{\text{OLS}}(\boldsymbol{\beta}) = \text{RSS}(\boldsymbol{\beta}) = (\boldsymbol{y} - \tilde{\boldsymbol{y}})^2 = (y_i - X_{ij}\beta_j)^2$$

where we employ the summation notation where repeated indices are summed over. As mentioned prior, the coefficients $\boldsymbol{\beta}$ are found by minimizing the cost function, i.e. taking the derivative w.r.t. $\boldsymbol{\beta}$. This results in

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

which yields the model

$$\tilde{\boldsymbol{y}}_{\text{OLS}} = \boldsymbol{X}\boldsymbol{\beta}_{\text{OLS}} \tag{6}$$

Assuming our data takes the form of (1) then the expectation value $\boldsymbol{y}$ is

$$\mathbb{E}(y_i) = \mathbb{E}(f(x_i)) = X_{ij}\beta_j \equiv \boldsymbol{X}_{i,*}\boldsymbol{\beta}$$

since $\mathbb{E}(\varepsilon_i) = 0$ follows from its definition. The variance of $\boldsymbol{y}$ is given by

$$\text{Var}(y_i) = \text{Var}(\varepsilon_i^2) = \sigma^2$$

which yields $y_i \sim \mathcal{N}(\boldsymbol{X}_{i,*}\boldsymbol{\beta}, \sigma^2)$. The expectation value of the optimal parameters $\hat{\boldsymbol{\beta}}$ can be found to be

$$\mathbb{E}(\hat{\boldsymbol{\beta}}_{\text{OLS}}) = \boldsymbol{\beta}.$$

with the variance

$$\text{Var}(\hat{\boldsymbol{\beta}}_{\text{OLS}}) = \sigma^2(\boldsymbol{X}^T\boldsymbol{X})^{-1}$$

2.2. Ridge

Ridge regression is an extension of OLS where define the cost function as a modified version of the OLS cost function with an added penalty term which is proportional to the coefficients β_i^2:

$$C_{\text{Ridge}}(\boldsymbol{\beta}) = C_{\text{OLS}}(\boldsymbol{\beta}) + \lambda\boldsymbol{\beta}^2 \tag{7}$$

Here $\lambda \geq 0$ is a regularization parameter, or commonly referred to as a hyperparameter, which controls the strength of this additional penalty. This regulator essentially drives the magnitude of these coefficients allowing for more tweaking in the parameter space. This parametrization of course includes the constraint that $\boldsymbol{\beta}^2 \leq t$ for some $t < \infty$ such that we can choose our arbitrary parameter $\lambda \geq 0$ to be sufficiently small s.t. the cost function (7) does not diverge. The optimal parameters for Ridge regressions can again be found by the same process as for OLS:

$$\boldsymbol{\beta} = (\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I})^{-1}\boldsymbol{X}^T\boldsymbol{y}$$

Here we can see that the effect of adding this penalty term is essentially taking $(\boldsymbol{X}^T\boldsymbol{X})^{-1} \to (\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I})^{-1}$ when compared to the OLS case. In the past this was generally the starting point for Ridge regression in the cases where the matrix $\boldsymbol{X}^T\boldsymbol{X}$ was not invertible. A direct way of seeing the effect of the regulator is by considering the singular value decomposition (SVD) of $\boldsymbol{X}$. Doing so one can show that

$$\tilde{\boldsymbol{y}}_{\text{Ridge}} = \sum_{j=0}^{p-1} \boldsymbol{u}_j\boldsymbol{u}_j^T \frac{\sigma_j^2}{\sigma_j^2 + \lambda}\boldsymbol{y} \tag{8}$$

where $\boldsymbol{u}_i$ is the i-th row (column?) of the orthogonal matrix $\boldsymbol{U}$ stemming from the SVD: $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}$. Since $\lambda \geq 0$ then this added factor compared to OSL is ≤ 1. We can then see that Ridge regression effectively suppresses all the coefficients, thus λ is often called the "shrinkage" factor.

2.3. LASSO

Similarly to Ridge, LASSO also includes a penalty factor. The cost function in this case is instead defined to be

$$C_{\text{LASSO}}(\boldsymbol{\beta}) = C_{\text{OLS}}(\boldsymbol{\beta}) + \lambda||\boldsymbol{\beta}||_1 \tag{9}$$

where

$$||\boldsymbol{\beta}||_k \equiv \left[\sum_{i=0}^{n-1} |\beta_i|^k\right]^{1/k}$$

is the L^k norm of $\boldsymbol{\beta}$. Taking the derivative of (7) w.r.t. $\boldsymbol{\beta}$ and requiring that this becomes zero we have

$$0 = \frac{\partial C_{\text{LASSO}}}{\partial\boldsymbol{\beta}} = -2\boldsymbol{X}^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}) + \lambda\,\text{sgn}(\boldsymbol{\beta}) \tag{10}$$

This has the added benefit of being able to set certain parameters to be 0 instead of suppressing them, at the cost of losing analytical expressions for $\hat{\boldsymbol{\beta}}$ in non-trivial cases.

2.4. Connection to Statistics

The three aforementioned regression techniques all have relatively simple connections to statistical probability distribution functions. Assuming that the likelihood of an event y_i with the input variables $\boldsymbol{X}$ and parameters $\boldsymbol{\beta}$ is given by a Gaussian distribution:

$$p(y_i, \boldsymbol{X}|\boldsymbol{\beta}) = \prod_{i=0}^{n-1} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{(y_i - \boldsymbol{X}_{i,*}\boldsymbol{\beta})^2}{2\sigma^2}\right] \quad (11)$$

Further assuming that all events are independent and identically distributed the total probability density function takes the form

$$p(\boldsymbol{y}, \boldsymbol{X}|\boldsymbol{\beta}) = \prod_{i=0}^{n-1} p(y_i, \boldsymbol{X}|\boldsymbol{\beta}) \quad (12)$$

then taking the negative logarithm of the above and defining that as our cost function we can arrive at the exact expression for the cost function that we defined earlier, i.e. just the RSS which corresponds to OLS. A similar analysis for Ridge can be done. Using Bayes' theorem we have that $p(\boldsymbol{\beta}|\boldsymbol{X}, \boldsymbol{y}) \propto p(\boldsymbol{y}, \boldsymbol{X}|\boldsymbol{\beta})p(\boldsymbol{\beta})$ we then feed in the prior:

$$p(\beta_i) = \exp\left[-\frac{\beta_i^2}{2\tau^2}\right] \quad (13)$$

one arrives at the probability of the coefficients $\boldsymbol{\beta}$ given the data $\boldsymbol{y}$

$$p(\boldsymbol{\beta}|\boldsymbol{X}, \boldsymbol{y}) \propto p(\boldsymbol{y}, \boldsymbol{X}|\boldsymbol{\beta}) \cdot \prod_{i=0}^{p-1} \exp\left[-\frac{\beta_i^2}{2\tau^2}\right] \quad (14)$$

Again, defining the cost function as the negative logarithm of this expression one arrives at the Ridge cost function with $\lambda = 1/(2\tau^2)$. Finally for LASSO one assumes that the initial probability is given by the same as OLS but with the addition of the β coefficients following a Laplace distribution

$$p(\beta_i) = \exp\left[-\frac{|\beta_i|}{\tau}\right] \quad (15)$$

one arrives at the LASSO cost function with $\lambda = 1/\tau$. These statistical connections help us better understand the assumptions made with each technique. With OLS one assumes that each event y_i are all independent and Gaussian. Thus when this is not the case one should not be surprised if OLS performs poorly. Ridge and LASSO however do account for the chance that there are correlated events, and assumes that the coefficients that we compute are themselves Gaussian or Laplace distributed. The hyperparameter λ now has a clear interpretation, i.e. that we essentially insert a standard deviation of the β coefficients. The performance of these latter two can then also be used to indicate which distribution more accurately describes the coefficients.

2.5. Bias-Variance

The so-called Bias-Variance Trade-Off can be summarized in a single equation:

$$\mathbb{E}[(\boldsymbol{y} - \tilde{\boldsymbol{y}})^2] = \text{Bias}[\tilde{\boldsymbol{y}}] + \text{Var}[\tilde{\boldsymbol{y}}] + \sigma^2 \quad (16)$$

where the bias is defined as $\mathbb{E}[(f(\boldsymbol{x}) - \mathbb{E}[\tilde{\boldsymbol{y}}])^2]$. The LHS of (16) is the expected value of the MSE which tells us how well the model's predictions match the true data on average. The equation shows that we can decompose this expected MSE into 3 different components.

- Bias: This quantity measures how much the model's average prediction differs from its true value. A high bias implies that the assumptions and simplifications which the model is built on are not valid, i.e. that we are underfitting the data.

- Var: The variance measures how much the model's predictions vary when trained on different datasets. It captures the sensitivity of the model to small changes in the training data. A high variance suggests overfitting, meaning it performs well on the training data but may be capturing noise or false patterns.

- σ^2: This is the irreducible error or noise in the data itself which cannot be explained by the model.

The idea is to minimize the LHS of (16), so clearly we want to minimize both the bias and the variance at the same time. However these are correlated to one another, so lowering the e.g. the bias can in general increase the variance and vice-versa. So Bias-Variance Tradeoff is essentially trying to optimize the complexity of the model such that we neither overfit nor underfit the model such that it can be generalized to other cases whilst not being too simplistic. These quantities can then be used as means to fine tune a model.

2.6. Resampling

Resampling techniques are critical in understanding the performance and robustness of a model. They tend to better show how well a model will do on data it has not seen before, and are thus useful for diagnosing problems like over- and underfitting. In this report focus on two resampling methods: bootstrapping and cross-validation.

2.6.1. Bootstrap

Bootstrapping is a statistical method that involves taking random samples with replacement from the original dataset to create several new datasets. These datasets are the same size as the original dataset, but since our sampling was done with replacement, some observations are presented more than once whilst others are eliminated. Here we train and evaluate a model over each of these resampled datasets allowing us to estimate how the performance varies across different datasets. This method is particularly important when the datasets are small as it heavily increases the amount information that can be extracted from a given data set. Averaging across all the performance metrics from the bootstrap samples can provide a clearer description of how well our model performs.

2.6.2. Cross-Validation

k-fold cross-validation is the other resampling method which we considered in this project. In k-fold cross-validation, the dataset is divided into k subsets (or folds). The model is trained on $k-1$ of these folds and tested on the remaining fold. This process is repeated k times, with each fold being used as the test set exactly once. The performance metrics, MSE and R^2, are averaged over the k iterations to provide a more reliable estimate of how the model performs on unseen data.

Cross-validation is particularly advantageous for its ability to reduce the variance of performance estimates by using different portions of the data for both training and validation. In our analysis, we used 5-fold cross-validation to evaluate the stability and predictive power of each regression method. This method gave us a clearer picture of how each model generalizes, helping us tune the hyperparameters, such as the regularization parameter λ for Ridge and LASSO.

Both resampling techniques are used to gain insight into the models performance and helps us ensure that the conclusions drawn from our analysis are not overly dependent on a particular division of the dataset.

3. IMPLEMENTATION

In this project we look to implement the three polynomial regression models; OLS, Ridge and LASSO, on 2D surface functions; $f(x,y) = z$, with $x, y \in [0,1]$. Before testing our models on actual data, we work with the Franke function, given by:

$$\begin{aligned} f(x,y) = {} & \frac{3}{4}\exp\left(-\frac{(9x-2)^2}{4} - \frac{(9y-2)^2}{4}\right) \\ & + \frac{3}{4}\exp\left(-\frac{(9x+1)^2}{49} - \frac{(9y+1)^2}{10}\right) \\ & + \frac{1}{2}\exp\left(-\frac{(9x-7)^2}{4} - \frac{(9y-3)^2}{4}\right) \\ & - \frac{1}{5}\exp\big(-(9x-4)^2 - (9y-7)^2\big) \end{aligned} \tag{17}$$

To perform an analysis on this function we consider a polynomial fit up to degree n analogous to (2) where:

$$\begin{aligned} \tilde{\boldsymbol{z}} = {} & \frac{1}{n+1}\sum_{i=0}^{n}\Big(\boldsymbol{\beta}_{0,0} \\ & + \boldsymbol{\beta}_{1,0}x_i + \boldsymbol{\beta}_{1,1}y_i \\ & + \boldsymbol{\beta}_{2,0}x_i^2 + \boldsymbol{\beta}_{2,1}x_iy_i + \boldsymbol{\beta}_{2,2}y_i^2 \\ & + ... \\ & + \boldsymbol{\beta}_{n,0}x_i^n + \boldsymbol{\beta}_{n,1}x_i^{n-1}y + ... + \boldsymbol{\beta}_{n,n-1}x_iy_i^{n-1} + \boldsymbol{\beta}_{n,n}y_i^n\Big) \\ = {} & \frac{1}{n+1}\sum_{i,j=0}^{n}\sum_{k=0}^{j}\boldsymbol{\beta}_{j,k}x_i^{j-k}y_i^k \equiv \boldsymbol{X\beta} \end{aligned} \tag{18}$$

Here the components x_i and y_i are entries in the input vectors $\boldsymbol{x}^T = \begin{bmatrix} x_0 & ... & x_n \end{bmatrix}$ and $\boldsymbol{y}^T = \begin{bmatrix} y_0 & ... & y_n \end{bmatrix}$ respectively which are our independent variables. Each $\boldsymbol{\beta}_{i,j}$ is a $\frac{(n+1)(n+2)}{2}$ component vector with a single non-zero entry with magnitude $|\beta_{i,j}|$ and the design matrix $\boldsymbol{X}$ is then an $(n+1) \times \frac{(n+1)(n+2)}{2}$ matrix of the form:

$$\boldsymbol{X} = \frac{1}{n+1}\begin{bmatrix} 1 & x_0 & y_0 & x_0^2 & x_0y_0 & ... & x_0^n & ... & y_0^n \\ 1 & x_1 & y_1 & x_1^2 & x_1y_1 & ... & x_1^n & ... & y_1^n \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 1 & x_n & y_n & x_n^2 & x_ny_n & ... & x_n^n & ... & y_n^n \end{bmatrix} \tag{19}$$

and the $\boldsymbol{\beta}$ vector contains the $\frac{(n+1)(n+2)}{2}$ components

$$\boldsymbol{\beta}^T = \sum_{i=0}^{n}\sum_{j=0}^{i}\boldsymbol{\beta}_{i,j}^T = \begin{bmatrix} \beta_{0,0} & \beta_{1,0} & \beta_{1,1} & ... & \beta_{n(n-1)} & \beta_{n,n} \end{bmatrix}$$

It should now be clear which unit vectors correspond to each term in (18).

The inputs $\boldsymbol{x}$ and $\boldsymbol{y}$ are then split randomly into a training and test set with the latter comprising 1/4 of the data throughout our analysis. These are then used to calculate both the train and test variations of $\boldsymbol{z}$ and $\boldsymbol{X}$ using the function `train_test_split` from `sklearn.model_selection`. The training and testing set may then be scaled (or not), before the β-coefficient is calculated. For Ridge, β_{Ridge} is found by numpy's `np.linalg.pinv` which gives the pseudoinverse (on the R.H. side of (8)) using the SVD. Obtaining β_{OLS} is then the special case of β_{Ridge} with $\lambda = 0$. The β-coefficient for LASSO on the other hand, is computed using `sklearn.Lasso`. Sklearn's LASSO is dependent on the chosen maximum number of iterations and a tolerance parameter. Throughout this project these parameters are set to 10^5 and 10^{-1}, respectively. Lastly, the mean squared error and the R2 score is then calculated using Sklearn's `metrics` library.

4. RESULTS

Feels like this is more set up as a conclusion than results. Think it's better to talk about it chronologically, i.e. talk about OLS, then Ridge and finally LASSO, and then start to compare them ones the plots have already been established. Then we can conclude that LASSO performs the worst out of the 3 and now the λ should have been established to make it easier to write these claims.

As a general observation, we found that the LASSO-method performed the worst out of the three methods, whilst OLS and Ridge performed similarly for the optimal choice of λ. With an increasing (maximum) polynomial degree the MSE of OLS and Ridge blew up, while LASSO settled horizontally. All three models had the largest β-coefficients corresponding to an polynomial degree of approximately $2-3$. Have we even checked the betas for ridge and lasso?

Between the choice of min-max scaling, standard scaling and no scaling, it was found that no scaling gave the best result. Min-max scaling showed little to no impact on both the MSE and the R^2-score, while standard scaling, in contrast, led to a deterioration in both metrics. As far as I understand we cannot look at the values for MSE for this one as it is scale dependent. R^2 however should be reasonable to look at though.

We found that all the three models are quite dependent on the size N of the sample-data set and on p. Keeping p fixed and increasing N, we found that the MSE of both the test and the training-data decreased. To better analyzie the performance of the three models we found that with $N = 25$ and $p = 25$. Think going past deg 10 is kind of redundant as we already see things go crazy past like deg 7. Looking at previous reports people generally tend to choose 5-8 as their maximum.

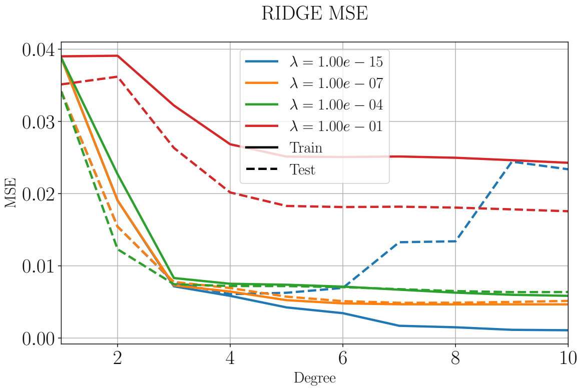

Fig. 1. Caption

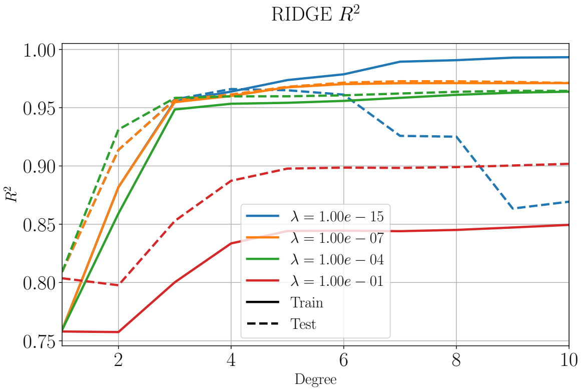

Fig. 2. Caption

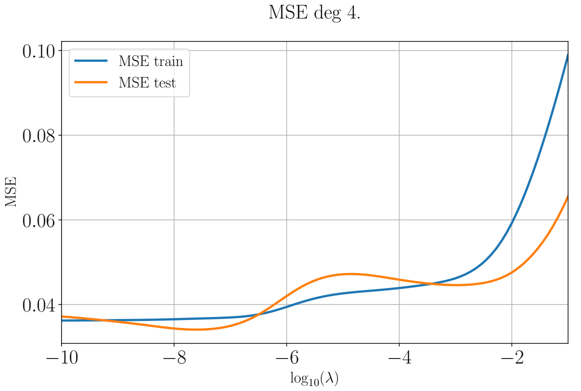

Fig. 3. Caption

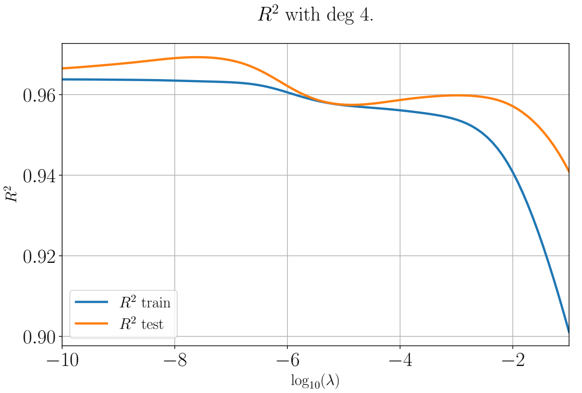

Fig. 4. Caption

4.1. OLS

The OLS-method showed a divergence in its MSE and R^2-score for higher polynomial degrees. Its β-coefficients seemed more dominant for the β-coefficients corresponding to polynomials of order 2 to 3; x^2, x^2y, y^3,etc.

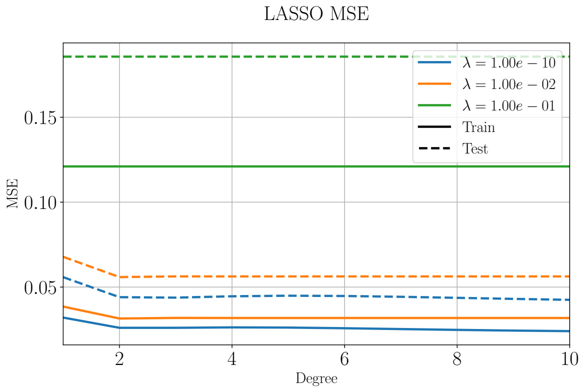

Fig. 5. Caption

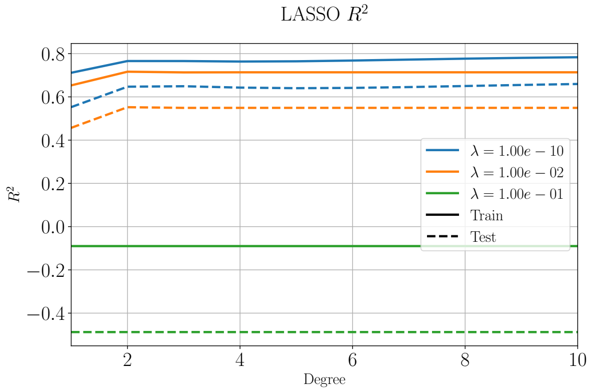

Fig. 6. Caption

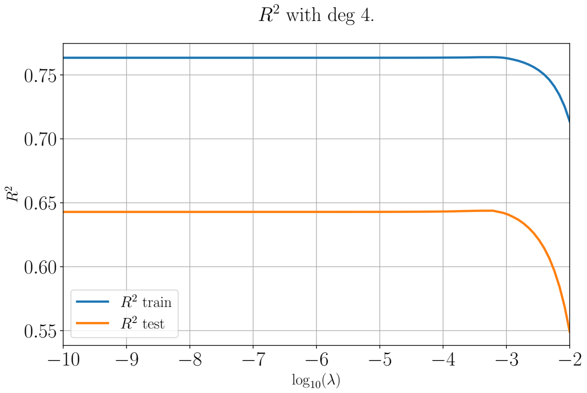

Fig. 7. Caption

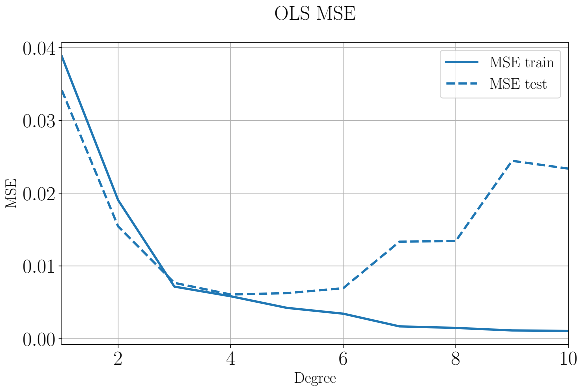

Fig. 8. The MSE for the OLS-method with $N = 25$, $p_{\max} = 25$ and $\epsilon = 0.1$.

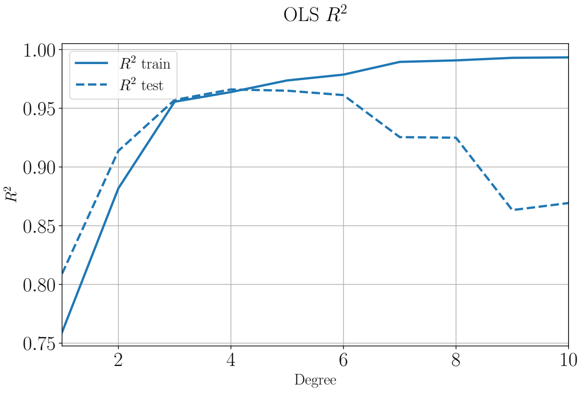

Fig. 9. The R^2-score for the OLS-method with $N = 25$, $p_{\max} = 25$ and $\epsilon = 0.1$.

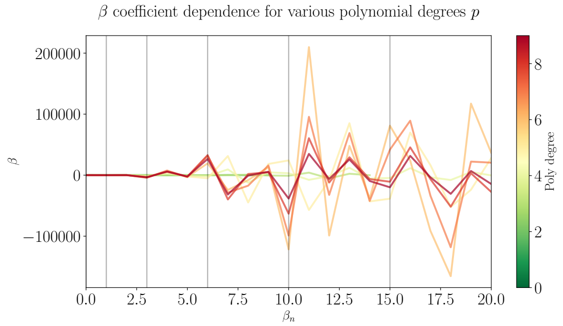

Fig. 10. The β-coefficients for the OLS-fit with $N = 25$, $p_{\max} = 25$ and $\epsilon = 0.1$. The plots shows the polynomial degree from 0 (green) to 25 (red).

4.2. Ridge

4.3. LASSO

5. DISCUSSION

For the Franke function, the fact that the most dominating β-coefficients where those corresponding to an polynomial degree of approximately $2-4$, is likely tied to the fact the the Franke function is a sum of Gaussian functions. A Gaussian function is Taylor expanded as $e^{-a(x-b)^2} = 1 - a(x-b)^2 + a^2(x-b)^4/2 + ...$, which clearly highlights the importance of terms involving polynomial degrees around $2-4$. I don't really understand the argument here. Is the argument that since factorials scale very fast and powers of $x, y \in [0,1]$ become small that higher order terms become negligible? If so then I would expect the model to be very insensitive to changes in the β coefficients corresponding to large powers and thus allowing them to become incredibly large without a large effect so it kind of seems like it's the opposite way around? This is of course only true for OLS since Ridge and LASSO have an explicit penalty term for large coefficients so they all get driven down by that.

The terrain-data, on the other hand, is very sensitive to the specific location it is extracted from. For example, mountain regions would likely be better modelled as a Gaussian, as most mountains are (at least) somewhat continuous and just visually in the form of a Gaussian. This is of course the case for Mount Everest. However, for the grand canyon, one would assume some (more) discontinuities, or cliffs with jagged edges, better modelled by say a very high degree of polynomial.

6. CONCLUSION

[1] U.S. Geological Survey, "EarthExplorer: Terrain Elevation Data." `https://earthexplorer.usgs.gov`, 2024. Accessed: 20/09/2024. Extracted terrain elevation data.

[2] C. M. Bishop, *Pattern Recognition and Machine Learning*, vol. 1. Springer, 2006.

Appendix A: Derivations

1. Model

The variance of $\boldsymbol{y}$ calculated as follows:

$$\begin{aligned}\mathrm{Var}(y_i) &= \mathbb{E}\{[y_i - \mathbb{E}(y_i)]^2\} \\ &= \mathbb{E}\{(\boldsymbol{X}_{i,*}\boldsymbol{\beta} + \varepsilon_i)^2\} - (\boldsymbol{X}_{i,*}\boldsymbol{\beta})^2 \\ &= (\boldsymbol{X}_{i,*}\boldsymbol{\beta})^2 + \mathbb{E}(\varepsilon_i^2) + 2\mathbb{E}(\varepsilon_i)\boldsymbol{X}_{i,*}\boldsymbol{\beta} - (\boldsymbol{X}_{i,*}\boldsymbol{\beta})^2 \\ &= \mathrm{Var}(\varepsilon_i^2) = \sigma^2\end{aligned}$$

A direct way of seeing the effect of the regulator is by considering the singular value decomposition of $\boldsymbol{X}$. Writing $\boldsymbol{X} = \boldsymbol{U\Sigma V}$ where $\boldsymbol{U}, \boldsymbol{V}$ are orthogonal and $\boldsymbol{\Sigma}$ only contains elements on the diagonal the proof goes as follows:

$$\begin{aligned}\tilde{\boldsymbol{y}}_{\mathrm{Ridge}} &= \boldsymbol{X}\boldsymbol{\beta}_{\mathrm{Ridge}} = \boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I})^{-1}\boldsymbol{X}^T\boldsymbol{y} \\ &= \boldsymbol{U\Sigma V}^T((\boldsymbol{U\Sigma V}^T)^T\boldsymbol{U\Sigma V}^T + \lambda\boldsymbol{I})^{-1}(\boldsymbol{U\Sigma V}^T)^T\boldsymbol{y} \\ &= \boldsymbol{U\Sigma V}^T(\boldsymbol{V\Sigma}^T\boldsymbol{\Sigma V}^T + \lambda\boldsymbol{I})^{-1}\boldsymbol{V\Sigma}^T\boldsymbol{U}^T\boldsymbol{y} \\ &= \boldsymbol{U\Sigma V}^T(\boldsymbol{V}(\boldsymbol{\Sigma}^T\boldsymbol{\Sigma} + \lambda\boldsymbol{I})\boldsymbol{V}^T)^{-1}\boldsymbol{V\Sigma}^T\boldsymbol{U}^T\boldsymbol{y} \\ &= \boldsymbol{U\Sigma}(\boldsymbol{\Sigma}^T\boldsymbol{\Sigma} + \lambda\boldsymbol{I})^{-1}\boldsymbol{\Sigma}^T\boldsymbol{U}^T\boldsymbol{y} \\ &= \sum_{j=0}^{p-1}\boldsymbol{u}_j\boldsymbol{u}_j^T\frac{\sigma_j^2}{\sigma_j^2 + \lambda}\boldsymbol{y}\end{aligned}$$

where the last step is valid due to the orthogonality of $\boldsymbol{U}$ and σ_j are the elements on the diagonal of $\boldsymbol{\Sigma}$.

2. Coefficients

The expectation value of the optimal parameters $\hat{\boldsymbol{\beta}}$ can be found to be

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

with the variance

$$\begin{aligned}\mathrm{Var}(\hat{\boldsymbol{\beta}}_{\mathrm{OLS}}) &= \mathbb{E}\{[\boldsymbol{\beta} - \mathbb{E}(\boldsymbol{\beta})][\boldsymbol{\beta} - \mathbb{E}(\boldsymbol{\beta})]^T\} \\ &= \mathbb{E}\{[(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y} - \boldsymbol{\beta}] \\ &\quad \times [(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y} - \boldsymbol{\beta}]^T\} \\ &= (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\mathbb{E}\{\boldsymbol{y}\boldsymbol{y}^T\}\boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X})^{-1} \\ &\quad - \boldsymbol{\beta}\boldsymbol{\beta}^T \\ &= (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T[\boldsymbol{X}\boldsymbol{\beta}\boldsymbol{\beta}^T\boldsymbol{X}^T + \sigma^2]\boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X})^{-1} \\ &\quad - \boldsymbol{\beta}\boldsymbol{\beta}^T \\ &= \boldsymbol{\beta}\boldsymbol{\beta}^T + \sigma^2(\boldsymbol{X}^T\boldsymbol{X})^{-1} - \boldsymbol{\beta}\boldsymbol{\beta}^T \\ &= \sigma^2(\boldsymbol{X}^T\boldsymbol{X})^{-1}\end{aligned}$$

The optimal parameters for Ridge regressions can again be found by the same process as for OLS:

$$\begin{aligned}0 = \frac{\partial C_{\mathrm{Ridge}}}{\partial\boldsymbol{\beta}} &= -\frac{2}{n}(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})^T\boldsymbol{X} + 2\lambda\boldsymbol{\beta}^T \\ &= \frac{2}{n}(\boldsymbol{\beta}^T\boldsymbol{X}^T\boldsymbol{X} - \boldsymbol{y}^T\boldsymbol{X}) + 2\lambda\boldsymbol{\beta}^T \\ 0 &= \boldsymbol{\beta}^T(\boldsymbol{X}^T\boldsymbol{X} + \tilde{\lambda}\boldsymbol{I}) - \boldsymbol{y}^T\boldsymbol{X} \\ \boldsymbol{\beta}^T &= \boldsymbol{y}^T\boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X} + \tilde{\lambda}\boldsymbol{I})^{-1} \\ \boldsymbol{\beta} &= (\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I})^{-1}\boldsymbol{X}^T\boldsymbol{y}\end{aligned}$$

where we defined $\tilde{\lambda} \equiv n\lambda$, renamed $\tilde{\lambda} \to \lambda$ and used that the matrix in the parenthesis is a symmetric matrix and thus its inverse must also be symmetric.

3. Bias-Variance

For ease of notation we write $f(\boldsymbol{x}) = f$ and simply ignore vector notation since everything is a scalar in the end. Then we have

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

where $\mathbb{E}[(f - \tilde{\boldsymbol{y}})\boldsymbol{\varepsilon}] = 0$ is justified by $\boldsymbol{\varepsilon}$ being independent and we note that the wrong definition of the Bias is given in the problem text (with that definition σ^2 gets put into the 'Bias'). All that remains is to show that the last term is 0. Since $\mathbb{E}[f] = f$ and $\mathbb{E}[f\,\mathbb{E}[\tilde{\boldsymbol{y}}]] = f\,\mathbb{E}[\mathbb{E}[\tilde{\boldsymbol{y}}]] = f\,\mathbb{E}[\tilde{\boldsymbol{y}}]$ then

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

which proves the claim.

Appendix B