Regression Analysis and Resampling Methods

Rashid S.

Institute of Physics, University of Oslo

sidrar@student.matnat.uio.no

October 10, 2020

Abstract

Machine Learning has taken off in many industries and has already had significant impact on our lives. This paper discusses the regression models Ordinary Least Squares, Ridge and LASSO, and their application. More specifically their ability to model the Franke function and landscape terrain from Yellowstone National Park, USA. The k-fold cross validation method is discussed, and the role it plays in reducing over-fitting and giving improved estimates for predictive accuracy. Throughout this paper the mean squared error and R2 score will be the litmus tests to measure goodness of the model. Results demonstrated that OLS performed the best in modelling terrain in Yellowstone, followed by Ridge and LASSO regression.

I Introduction

Carl Friedrich Gauss in 1794 was the first to invent and use the method of 'least squares'. It was of little importance to Gauss this method that he called it, ordinary. He was convinced that anyone engaged in numerical analysis must be using it. Often Gauss would make bets with others that least squares was used by Tobias Mayer, an elder mathematician. Later on Gauss reviewed Mayer's papers and realised he would have lost all the bets [3].

This paper builds upon Gauss' ordinary least squares (OLS) and introduces an area of data science, called Machine Learning. Machine learning is the method of building an algorithm which learns from data and makes predictions on data that was previously unknown to it. Machine learning has become extremely attractive for all industries where there exists a large amount of data. It is so attractive because one can extract knowledge from the data, enhance the performance of the industry, and make future predictions.

An example is using Machine Learning to determine if an individual will default or not on their credit card loan. The inputs or predictors are: age, gender, annual income, mortgage, assets etc. A bank would wish to create a model where they could input a new customer's data and make predictions.

The type of machine learning discussed in this paper will be regression. Regression falls under the heading, supervised learning, meaning to make predictions after having been trained on a set of data. A regression model predicts continuous-values e.g in the case of this paper to predict the terrain height of an area based upon being trained on a portion of the terrain. Another form of supervised learning is classification, which predicts discrete-values, e.g. predicting if an image is a cat or a dog after having been trained on a set of images.

The regression methods incorporated in this paper include: OLS (developed by Gauss in 1794), Ridge (developed by Arthur E. Hoerl and Robert W. Kennard in 1970) [4] and Least Absolute Shrinkage and Selection Operator (LASSO) (developed by Robert Tibshirani in 1995) [6].

The aim in this paper is to use Machine Learning to first model the Franke function and then to model a terrain data set, using the three regression types above. Each of the three, OLS, Ridge and LASSO will be discussed. This paper will also discuss the k-fold cross validation and how this can impact the predictive accuracy of a Machine Learning model in regression.

The paper is laid out by first presenting the theory behind the algorithms and statistics. Following this, the paper introduces some of the methods used to code the algorithms. The results of the modelling are presented with graphs and plots for both the Franke function and the Yellowstone Park terrain. Finally the results are discussed and critical evaluations are made on the models, ending with some conclusions.

All programs of relevance are to be found here: https://github.com/Seedsiz/FYS-STK-Project1

II Theory

The following discussion of the theoretical aspects is based on [7].

II.I A Frequentist approach to statistics

Working on a real dataset, one can never know the true value of the mean (μ), nor the variance (σ^2), due to a limited set of data and the probability density function (ρ) being

unknown. However, if integrals are interpreted as a sum of function values, a valuable and sufficient estimate can be made of these parameters given a large sample size.

One of the most common estimands to be approximated is μ. Usually, it is estimated by the population mean ($\bar{y}$),

$$\mathbb{E}[y] = \bar{y} = \frac{1}{n}\sum_{i=0}^{n-1} y_i \tag{1}$$

It is a sum of datapoints (y_i) divided by the total number of data (n).

To further describe important features of the data to be analysed, quality factors give insight into how well a model fits to its data sample. Examples of such factors are the mean square error (MSE) and the R2-score function.

MSE is a cost function, and takes the mean of the squared difference between data points predicted by an interpolation model ($\tilde{y}$) and the actual data (y_i).

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

Thus, it is a mean of the squared discrepancy between the fitted model and data. For an optimal model, MSE will approach zero as the unexplained variance is better explained by parameters in the model.

Another quality factor is the R2-score function, also known as the coefficient of determination. It normalizes the MSE by dividing on the biased estimator of the population variance ($\hat{\sigma}$^2), thus giving a ratio $\in [-\infty, 1]$.

$$R^2(\hat{y},\tilde{y}) = 1 - \frac{\sum_{i=0}^{n-1} (y_i - \tilde{y}_i)^2}{\sum_{i=0}^{n-1} (y_i - \bar{y})^2} \tag{3}$$

If the model fits the data well, or if the MSE-value is much less than σ^2, the ratio will be close to zero. As a consequence, the coefficient will approach 1. Hence, the closer R^2 gets to 1, the more variability in the dataset is explained by the fitted model.

II.II Regressions

To fit a model to a given data set assume, that the true data is depending on the features $\mathbf{x} \in \mathbb{R}^n$ with a functional dependency f

$$\mathbf{y} = f(\mathbf{x}) + \boldsymbol{\varepsilon} \tag{4}$$

and has statistical independent standard normal distributed fluctuations $\boldsymbol{\varepsilon} \in \mathbb{R}^n$ with variance σ^2 ($\boldsymbol{\varepsilon} \sim \mathcal{N}(0, \sigma^2)$)). The functional dependency can be modeled with

$$f(\mathbf{x}) = (\mathbf{X}\boldsymbol{\beta}) = \tilde{\mathbf{y}} \tag{5}$$

where $\mathbf{X} \in \mathbb{R}^{nxp}$ is called the design matrix and $\boldsymbol{\beta} \in \mathbb{R}^p$ are the parameters of the parametrization of the model.

Minimizing the cost function

$$C(\mathbf{X}, \boldsymbol{\beta}) = \frac{1}{n} \sum_{i=0}^{n-1} (y_i - \tilde{y_i})^2 = \frac{1}{n} (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^T (\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) = \mathbb{E}[(\mathbf{y} - \tilde{\mathbf{y}})^2] \tag{6}$$

with respect to the parameters leads to a parameter estimate for the ordinary least square (OLS) regression

$$\hat{\boldsymbol{\beta}} = \min_{\boldsymbol{\beta}} C(\mathbf{X}, \boldsymbol{\beta}) = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} \tag{7}$$

It can be shown, that the variance of the parameter estimates is given by

$$Var(\hat{\boldsymbol{\beta}}_i) = \sigma_i^2((\mathbf{X}^T\mathbf{X})^{-1}))_{ii} \tag{8}$$

due to the statistical independent standard normal distributed fluctuations. Thus, the estimated parameters are normal distributed with mean $\beta\hat{i}$ and variance σ_i^2. Moreover, the confidence interval for a confidence level $1 - \alpha$ is given as

$$\beta_i \in (\hat{\beta}_i - Z(1-\alpha/2)\frac{\sigma_i}{\sqrt{n}}, \hat{\beta}_i + Z(1-\alpha/2)\frac{\sigma_i}{\sqrt{n}}) \tag{9}$$

where Z(1 - $\alpha/2$) is the inverse of cumulative distribution function of the standard normal distribution, evaluated at the confidence level and n is the number of samples

The inversion of the matrix $\mathbf{X}^T\mathbf{X}^{-1}$ can lead to difficulties when its singular. Thus, a regularization is introduced. Its strength is controlled by the parameter $\boldsymbol{\lambda}$. For Ridge regression the new cost function to minimize is

$$C(\mathbf{X}, \boldsymbol{\beta}, \boldsymbol{\lambda})_{Ridge} = \frac{1}{n} (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^T (\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) + \lambda \|\boldsymbol{\beta}\|_2^2 \tag{10}$$

Here, $\|.\|_2$ is the l^2 norm (standard euclidian norm). This introduces a replacement of the matrix

$$\mathbf{X}^T\mathbf{X} \rightarrow \mathbf{X}^T\mathbf{X} + \boldsymbol{\lambda}\boldsymbol{I} \tag{11}$$

in all analytical expression, where $I \in \mathbb{R}^{pxp}$ is the identity matrix. Contrarily, LASSO regression imposes a l 1 norm regularization.

$$C(\mathbf{X}, \boldsymbol{\beta}, \boldsymbol{\lambda})_{Lasso} = \frac{1}{n}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^T(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) + \lambda\|\boldsymbol{\beta}\|_1 \tag{12}$$

Both regularization methods impose additional conditions on the magnitude of the optimal parameters. However, for the LASSO regression no analytic expression can be found because the l^1 norm is not differentiable.

II.III Bias-Variance Trade-off

Consider a set of data $\mathscr{L}$ consisting of corresponding y and x values, $\mathbf{X}_{\mathscr{L}} = \{(y_j, \mathbf{x}_j), j = 0...n-1\}$. Assume the true data can be described with a function f with a normally distributed deviation $\boldsymbol{\varepsilon}$ eq. (4). In this case, a model can be made eq. (5), by minimizing the cost function $C(\mathbf{X}, \boldsymbol{\beta})$ eq. (6).

The cost function can be rewritten using the assumption that **y = f + $\boldsymbol{\varepsilon}$**, which gives

$$\mathbb{E}[(\mathbf{y} - \tilde{\mathbf{y}})^2] = \mathbb{E}(\mathbf{f} + \boldsymbol{\varepsilon} - \tilde{\mathbf{y}})^2 \tag{13}$$

By adding and subtracting the expectation value of our model results in

$$\mathbb{E}[(\mathbf{y} - \tilde{\mathbf{y}})^2] = \mathbb{E}[((\mathbf{f} - \mathbb{E}[\tilde{\mathbf{y}}]) + (\mathbb{E}[\tilde{\mathbf{y}}] - \tilde{\mathbf{y}}) + \boldsymbol{\varepsilon})^2] \tag{14}$$

Writing out the expression yields

$$\mathbb{E}[(\mathbf{f} - \mathbb{E}[\tilde{\mathbf{y}}])^2 + 2(\mathbf{f} - (\mathbb{E}[\tilde{\mathbf{y}}])(\mathbb{E}[\tilde{\mathbf{y}}] - \tilde{\mathbf{y}}) + \boldsymbol{\varepsilon}^2 +$$

$$\boldsymbol{\varepsilon}(\mathbf{f} - (\mathbb{E}[\tilde{\mathbf{y}}]) + \boldsymbol{\varepsilon}(\mathbb{E}[\tilde{\mathbf{y}}] - \tilde{\mathbf{y}}) + (\mathbb{E}[\tilde{\mathbf{y}}] - \tilde{\mathbf{y}})^2] \tag{15}$$

Since E[ε] = 0 and E[ε^2] = σ^2, there are only three parts of the equation left to address. By writing out the parenthesises for the quadratic equation with coefficient two , gives 0, due to E[E[$\tilde{\mathbf{y}}$]] = E[$\tilde{\mathbf{y}}$] and E[$\tilde{\mathbf{y}}\mathbf{f}$] = E[$\tilde{\mathbf{y}}$]E[$\tilde{\mathbf{f}}$].

As a consequence, by using the frequentist approach on the last two equations, the cost-function can be given by

$$\mathbb{E}[(\mathbf{y}-\tilde{\mathbf{y}})^2] = \frac{1}{n}\sum_{i=0}^{n-1}(f_i - \mathbb{E}[\tilde{\mathbf{y}}])^2 + \frac{1}{n}\sum_{i=0}^{n-1}(\tilde{\mathbf{y}}_i - \mathbb{E}[\tilde{\mathbf{y}}])^2 + \sigma^2 \tag{16}$$

The first expression represents the bias error, while the variance can be described by the second term. The last term comes from the variation from the deviation *epsilon* of the data.

The bias error is the mean difference between data points and the expected values of the fitted algorithm squared. If the bias error is larger than the variance, there must be wrong or missing assumptions in how the parameters interact (underfitting).

Variance is defined as the mean difference between datapoints generated by the model and the expectation value from that model. If fluctuation in the training data easily affects a learning algorithm, the variance will be higher than the bias. As a result, the noise in a data set will influence a model's output, in such a way that the model will vary greatly between different sets of training data (overfitting).

II.IV Re-sampling using K-fold cross validation

Often it is difficult to get an accurate estimate of the prediction error or some other parameter, due to having a limited set of data [7]. In such a case, just having one validation set and one training set taken from a small sample size, would probably give a poor parameter estimate. Therefore, methods for re-sampling are used to get a more accurate result for the predicted parameter.

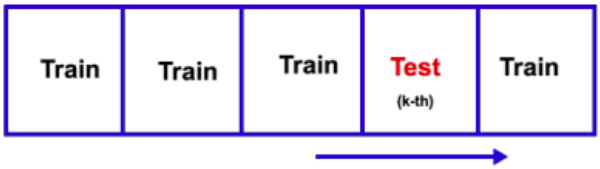


Figure 1: The K-fold cross-validation splits the dataset into test and training set, and an estimate of a given parameter is made for each k-th iteration.

K-fold cross validation are one of these methods [7]. Firstly, the entire data set is split into K equally large subgroups, where $2/3$ or $4/5$ is training data, and the rest is a validation set or test set. Let $k = \{1, 2, ..., K\}$ be the indexing of groups. For each k - use the

K-1 groups for training the model, and the k-th subgroup for validating or testing the model.

For each k-th iteration, an estimate is made for the wanted parameter, for example the MSE eq. (2). After K iterations, the average of all k estimations is calculated.

The benefit of the K-fold cross validation is that it reduces over-fitting. When training a model on the entire data set and evaluating the model on same data, the result will be an unrealistic mean squared error (high predictive accuracy). This kind of model will be influenced by noise. By holding out a section of the data set for testing, the portion of data to train the model on is reduced, however by evaluating the model on the hold out section gives a more realistic estimate of predictive accuracy [2]. This also follows from the central limit theorem, stating that the expectation value of many experiments of independent identical events (i.i.ds) will approach the true value.

III Methods

The different regression models and the k-fold cross-validation procedure described in the previous section were implemented as methods of a class Poly2DFit . The class objects stores the data points and has a method to create a design matrix for a 2D polynomial of order *poly* (If the creation of the design matrix would be generalized to any dimensional polynomial, the whole implementation could fit arbitrary dimensional data to a polynomial.).

The method implemented to find the design matrix is first to produce the list of predictors necessary for the order of the polynomial. By expanding

$$(x+y)^{poly} \propto \sum_{j=0}^{poly} \sum_{i=0}^{j} x^{j-i} y^i \tag{17}$$

where *poly* is the order of the polynomial. This gives the predictors in the design matrix which will be multiplied by the parameters β. The result of this expansion is e.g for $poly = 3$

$$x^0y^0 + x^1y^0 + x^0y^1 + x^2y^0 + x^1y^1 + x^0y^2 + x^3y^0 + x^2y^1 + x^1y^2 + x^0y^3 \tag{18}$$

The OLS and Ridge regression were implemented following eq. (7) and the substitution for Ridge in eq. (11). The inversion of the matrix was done with numpy.linalg.inv and if this fails, the inversion is done with a singular value decomposition numpy.linalg.svd.

The LASSO regression was taken from sklearn.linear_model with a tolerance of 0.001, maximal number of iterations of 10^5 and a precomputed Gram matrix to speed up computations.

A typical work-flow would be to first initialize the Poly2DFit object and then assign the data via the method Poly2DFit.givenData. Afterwards, the fit is performed by calling Poly2DFit.run_fit. Here the regression type is given as a keyword OLS, RIDGE or LASSO, the polynomial order and if needed the regularization parameter λ. This method takes care of setting up the design matrix, calculating the parameters and their variance. To evaluate the score functions (MSE, R2) of the model, Poly2DFit.evaluate_model is called.

Alternatively to calling the plain Poly2DFit.run_fit, a k-fold cross-validation method Poly2DFit.kfold_cross can be called. It takes in addition to the Poly2DFit.run_fit method the parameter k, which determines how many runs of k-fold cross validation are performed. Basically, this method splits up the data-set into k-folds and iterates over them, where each fold becomes once the test set and the remaining data the training set. With this split again the Poly2DFit.run_fit method is called. The model is evaluated, once with the training data and once with the test data. The score results are weighted with $1/(k+1)$ and stored for later access. The kFold flag tells the other methods to use appropriate design matrix for further operations. The pseudo-code for the implementation is found in algorithm 1.

Algorithm 1 K-fold Cross Validation

function POLY2DFIT.KFOLD_CROSS(self, Pol_order, regtype, lam = 0.1, k = 1)
kFold $\Leftarrow True$
self.k $\Leftarrow k+1$
data $\Leftarrow shuffled$
data $\Leftarrow splitintok+1subsets$
for int i in range (k+1) do **do**
test $\Leftarrow datasubset[i]$
train $\Leftarrow concatenatedatasubsets[\neq i]$
self $\Leftarrow Poly2DFit.run\_fit(self, Pol\_order, lam)$ ▷ Uses the train if kFold = True
self $\Leftarrow Poly2DFit.evaluate\_model(self)$ ▷ Uses train and test if kFold = True
▷ Score functions get added with weight 1/self.k

end for

The implementations are tested against the corresponding sklearn implementations.

III.I Benchmarking

For bench marking of the algorithms the Franke function is used with Gaussian noise with mean 0 and variance 1. The Franke function is a two dimensional weighted sum of four exponential functions

$$\begin{aligned} f(x,y) = \frac{3}{4}exp(\frac{(-9x-2)^2}{4} - \frac{(9y-2)^2}{4}) + \frac{3}{4}exp(-\frac{(9x+1)^2}{49} - \frac{(9y+1)^2}{10}) \\ \frac{1}{2}exp(-\frac{(9x-7)^2}{4} - \frac{(9y-3)^2}{4}) - \frac{1}{5}exp(-(9x-4)^2 - (9y-7)^2) \end{aligned} \tag{19}$$

This function displays several behaviors over a limited interval surface, hence it is interesting as an interpolation problem [5].

The various regression methods are evaluated over different sample sizes of the Franke function for different polynomial orders, regularization parameter and k-values for cross validation.

III.II Terrain data

Terrain data was obtained from https://earthexplorer.usgs.gov/, an open access resource for geological data, made available by the US. Geological Survey. STRM1 Arc-Second Global was the format used, and data was downloaded as a GeoTIF file. The data file was read using imread from python's imageio library

The TIF-file contains a two-dimensional grid, having elevations corresponding to different coordinates. I decided to reduce the matrix, due to the TIF file's original size (3601x3601), containing over $1.296 \cdot 10^7$ data points. The function load terrain thus reduces the array by picking out every i-th data point, through the sel variable. The reduced matrix is then flattened to a one-dimensional array.

Because the data does not provide x and y coordinates, they are chosen to be between 0 and 1. The pixel in the upper left corner is assigned to be (0,0) and from there on filling from left to right, top to bottom with step size $1/\text{pixels}_{row}$ and $1/\text{pixels}_{column}$, respectively. This choice is made to avoid artificial effects of large coordinates. Moreover, to increase stability and convergence of the LASSO regression the data is normalized and shifted to have a mean value of zero.

IV Results

The following subsections focus on shedding light on different features of the OLS, Ridge and LASSO regression methods, and give insight into how the models work to predict a given dataset. Firstly, the Franke function with stochastic noise ($\mathcal{N}(\mu = 0, \sigma^2 = 1)$) is taken into consideration, followed by analysing terrain data from Yellowstone National Park.

IV.I Franke Function Modelling Results

IV.I.1 Performance and Stability

A first thing to investigate is the time consumption of the Poly2DFit.run_fit method as a function of data size. As expected, with increasing number of sample points from the Franke function the time consumption increases roughly exponentially as shown in fig 2a.

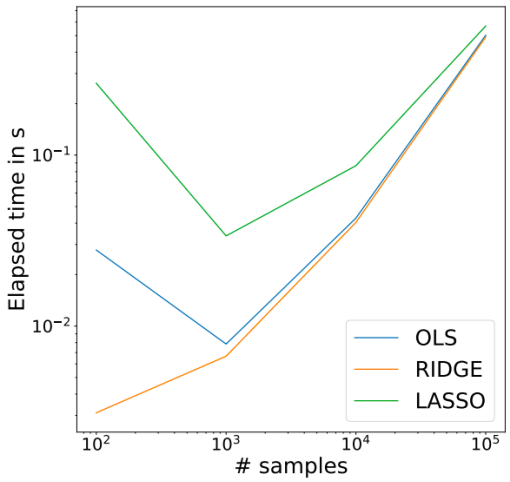
Another thing to check, is the stability of the MSE as a function of sample size and polynomial order for the different regression methods. In fig 2b the different regression methods are evaluated on different sample sizes for a polynomial of order 4. The behaviour of the MSE for different polynomial orders is investigated for OLS and Ridge regression in fig 2c and 2d. Whereas for Ridge regression the MSE of all polynomial orders seems to converge to one within the given range of sample sizes, for OLS regression it seems that above a given threshold for the polynomial order it is not converging to one within the given range of sample sizes.

(The instabilities which can be seen in Fig. 2c, Fig.3 and Fig.4 for large polynomial orders: It turns out, that these are numerical effects of direct matrix inversion from numpy.linalg. Switching to singular value decomposition and defining a cut-off for inversion of singular values below 10 - 12 resolves this issue. However, the analysis is left unchanged, but it's best to keep in mind that blow-ups for high model complexity are due to this issue.)

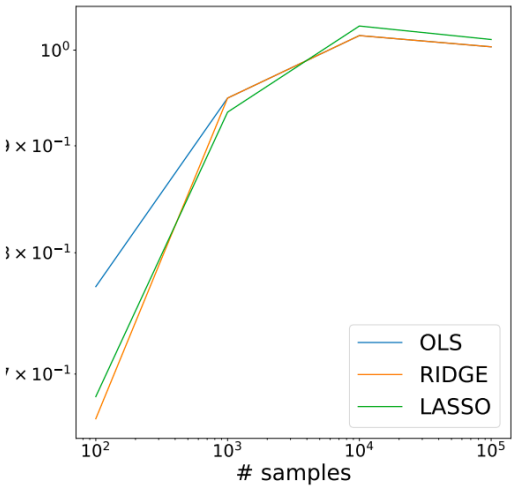
IV.I.2 OLS Results

A sample of size n = 500 and n = 5000 is chosen first to analyse the behaviour of the algorithms with relatively small and large sample sizes respectively. Each regression model results are shown separately below. In the appendix a graph of all plots is given to enable comparison with the same scale bar (Fig16).

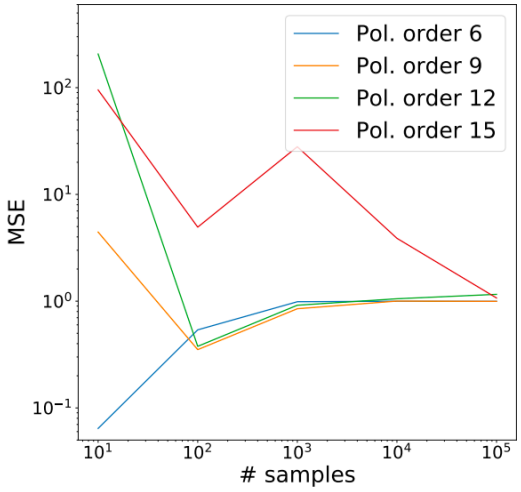
First looking at sample size n=500, the results show that without the k-fold cross validation, in Fig.3 (a), that the MSE decreases with complexity and increases suddenly at



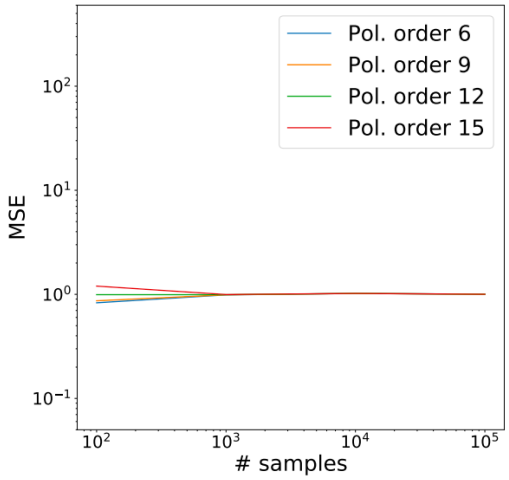
(a) Log-Log plot of the elapsed computational time for the different regression methods for an order 4 polynomial as a function of data samples.



(b) Log-Log plot of the MSE on the test data for the different regression methods for an order 4 polynomial as a function of data samples.



(c) Log-Log plot of the MSE on the test data for the different polynomial orders as a function of data samples for OLS regression.



(d) Log-Log plot of the MSE on the test data for the different polynomial orders as a function of data samples for Ridge regression.

Figure 2: Performance and stability aspects for different sample sizes of the Franke function. The regularization parameter is set to $\lambda = 0.01$ and a 5-fold cross validation is used.

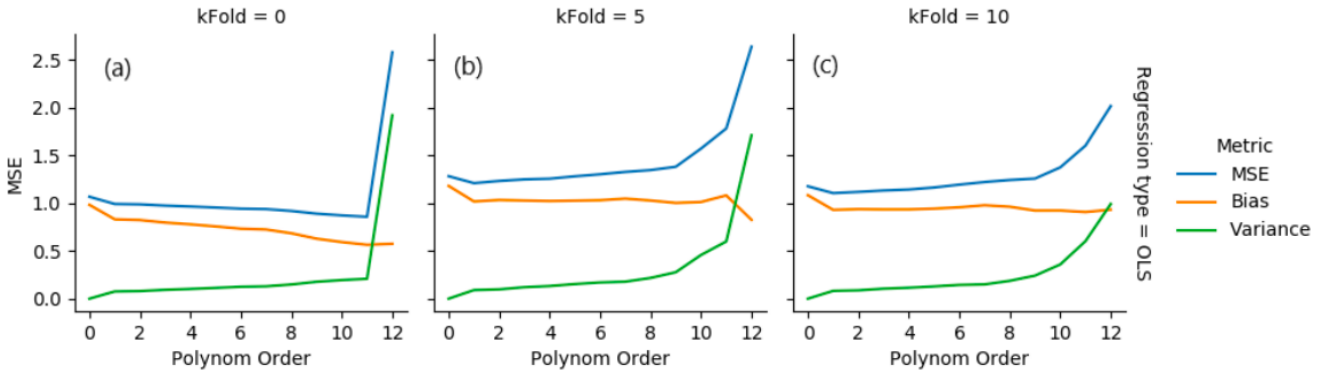


Figure 3: Bias Variance Trade-off plot for OLS regression model with and without k-fold cross validation. Model created using 500 data points.

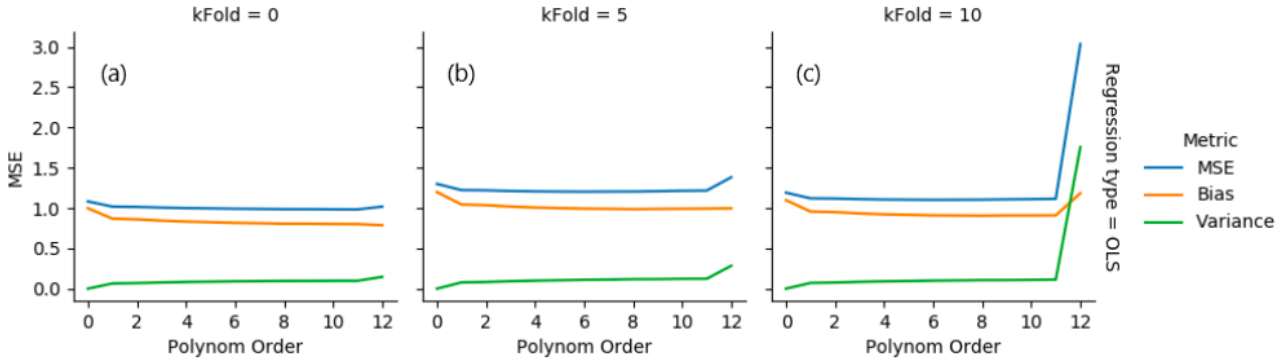


Figure 4: Bias Variance Trade-off plot for OLS regression model with and without k-fold cross validation. Model created using 5000 data points.

polynomial order 12 to an MSE of 2.60. The MSE reaches a minimum, of 0.86, at polynomial order 11. The R2-score at this minimum is 0.20. The bias decreases gradually, whilst the variance increases with complexity and suddenly increases at polynomial order 12 to var = 1.92.

As can be seen in Fig.3 (b) when 5-fold cross validation is used, the MSE is at a minimum at polynomial order 1. The MSE at this minimum is 1.21. The R2-score at this minimum is 0.07. The trend here is that MSE increases with polynomial order and more rapidly at higher orders. The same trend is seen in Fig.3 (c), here using 10-fold cross validation. The minimum MSE is 1.11 at polynomial order 1, which is lower than using 5-fold cross validation. The R2-score at this minimum is 0.06. In Fig.3 (b) and (c), the bias is fairly stable with a few fluctuations at higher orders. The variance however, shows a gradual increase with increasing orders and increases more rapidly at higher orders, similar to MSE.

Figure.4 (a) shows a more flat MSE than Fig. 3 (a). The minimum MSE in Fig.4 (a) is 0.98 is at polynomial order 11. The R2-score at this minimum is 0.09. Using 5-fold cross validation, Fig.4 (b) the minimum MSE is 1.20 at polynomial order 6 (R2-score 0.09). At polynomial order 12 there is a slight increase to 1.38. Similar trends found in Fig.4 (c), the minimum is at polynomial order 6, where MSE = 1.10 (R2-score 0.08). However there is a greater increase at polynomial order 12 to MSE = 3.03. In Fig. 4 (b) and (c) the bias is flat, the change in bias across the range of complexity is only 0.1 and 0.19 respectively (excluding polynomial order 12). The variance also has little change with complexity, with the exception in Fig.4 (c) where the variance increases suddenly.

An example of the beta parameters produced by the OLS model is given below. This is a model using 500 data points and order polynomial 5, with 5-fold cross validation. They have been rounded to one decimal place.

$$0.46.32.3-20.58.4-19.13.631.0-71.067.136.3-95.4103.936.0-88.0-25.753.3-25.9-47.86.438.5$$

IV.I.3 Ridge Results

Fig. 6(a) shows Bias Variance Trade-off of the Ridge regression without the k-fold cross validation using a sample population with 500 data points. The regularisation parameter is set to $\lambda = 0.001$. The MSE and Bias decrease from polynomial order 0 to order 1, and then show a slower decrease with more complexity. The minimum MSE is 0.93 at polynomial order 12 (R2-score 0.13). The variance has the same trend in the opposite direction.

Using the 5-fold cross validation the MSE and bias also decrease from order 0 to 1, but

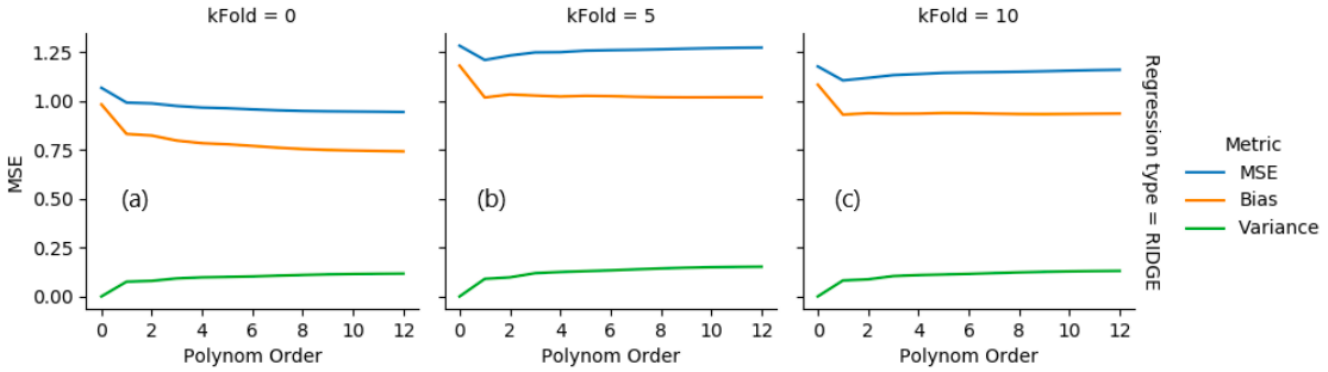


Figure 5: Bias Variance Trade-off plot for Ridge regression model with and without kfold cross validation. Model created using 500 data points. The regularisation parameter is set to $\lambda = 0.001$

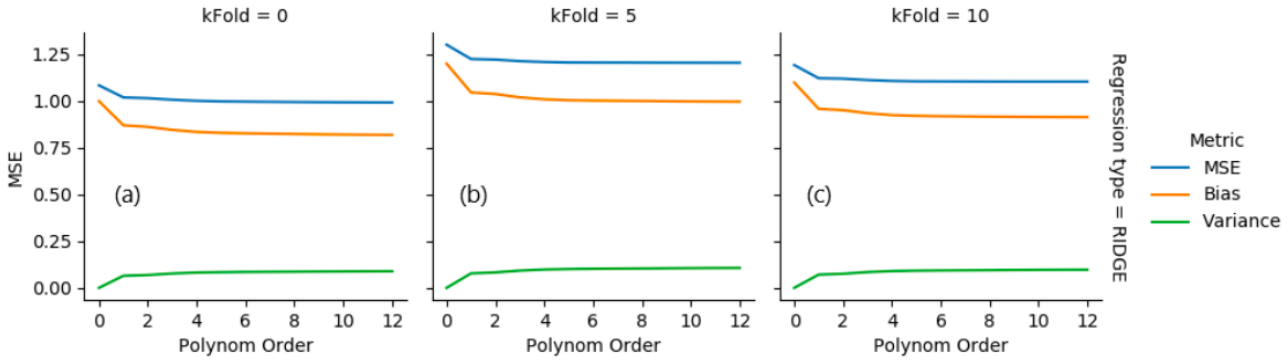


Figure 6: Bias Variance Trade-off plot for Ridge regression model with and without kfold cross validation. Model created using 5000 data points. The regularisation parameter is set to $\lambda = 0.001$

then increase gradually with complexity. The minimum MSE is 1.21 at polynomial order 1 (R2-score is 0.07). The variance shows an increase from order 0 to 1 and then a gradual increase with complexity. Using the 10-fold cross validation similar trends are found, as seen in Fig. 5. The minimum MSE is 1.11 at polynomial order 1 (R2-score is 0.06).

Fig. 6 is a model created using a sample population of 5000 data points. Much of the same trends are found as in Fig. 5. They differ in that the MSE and bias decrease gradually with complexity. The minimum MSE for 5-fold is 1.20 at polynomial order 12 (R2-score is 0.09). The minimum MSE for 10-fold is 1.10 at polynomial order 10 (R2-score is 0.08), although the difference is negligible between the high order polynomials.

An example of the beta parameters produced by the Ridge model is given below. This is a model using 500 data points, order polynomial 5, $\lambda = 0.1$ and using 5-fold cross validation. They have been rounded to one decimal place.

$$1.0 - 0.20.6 - 0.8 - 1.2 - 1.4 - 0.20.5 - 1.1 - 1.20.40.10.60.00.10.5 - 0.2 - 0.10.81.31.6$$

A heatmap was made for fitting a Ridge regression to the Franke function (Fig. 7). MSEscores varies with model complexity and λ, with the best fit being for p = 1 and $0.1 \leq \lambda \leq 10$.

IV.I.4 LASSO results

The LASSO model shows very similar trends as that in the Ridge model. It can be seen in Fig. 8 (g), (h) and (i), the MSE has become very flat for varying complexity when lambda is set to $\lambda = 0.1$.

An example of the beta parameters produced by the Ridge model is given below. This is a model using 500 data points, order polynomial 5, $\lambda = 0.1$ and using 5-fold cross validation. They have been rounded to one decimal place.

$$0.30.00.00.00.00.00.00.00.00.00.00.00.00.00.00.00.00.00.00.00.0$$

All LASSO beta parameters have been minimised to zero except one.

A heatmap was made to show the MSE-score for LASSO regression when fitting for the Franke function (Fig. 9). There is a trade-off for λ and complexity, with p = 1 and $\lambda \leq 0.01$ giving the best fit.

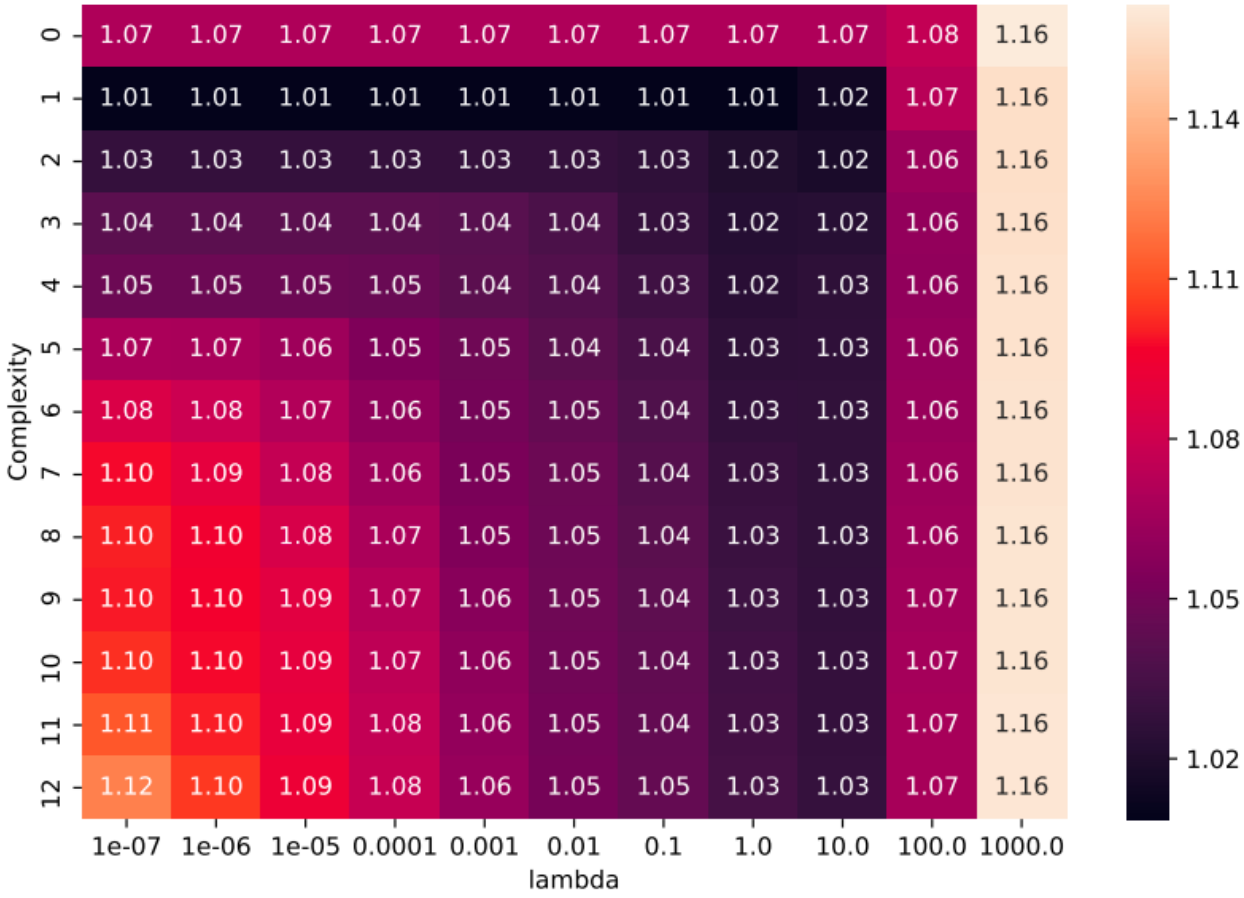


Figure 7: Showing the optimal regularization parameter λ and complexity for the Ridge regression, for a model with 500 data points and using 5-fold cross validation. MSEscores are shown from low (dark) to high (light). The optimal lambda is between 0.1 and 10.

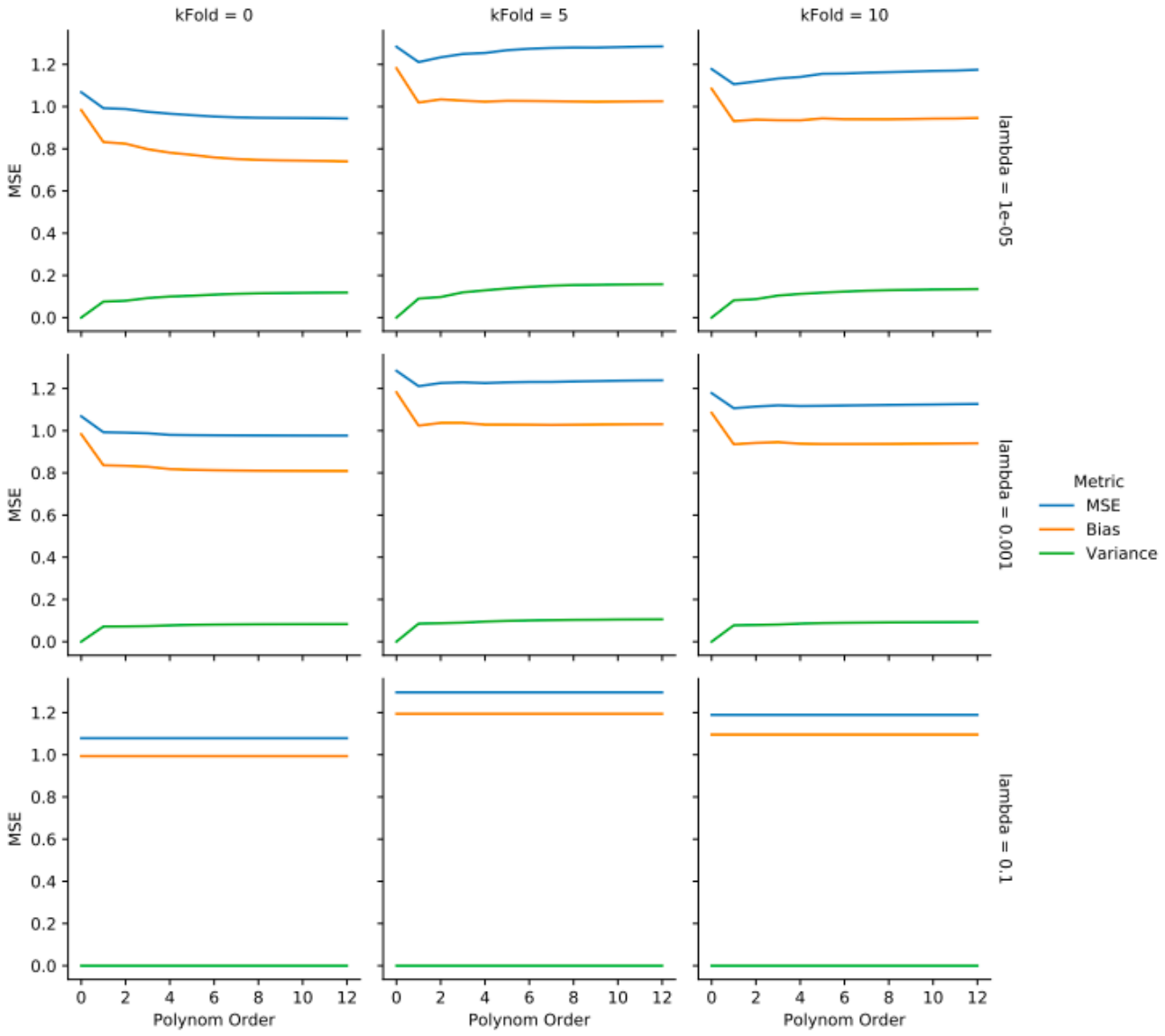


Figure 8: LASSO Bias Variance Trade-off plot with varied kfolds and regularisation parameter λ. The population size is 500 data points. LASSO MSE becomes more flat with larger lambda values.

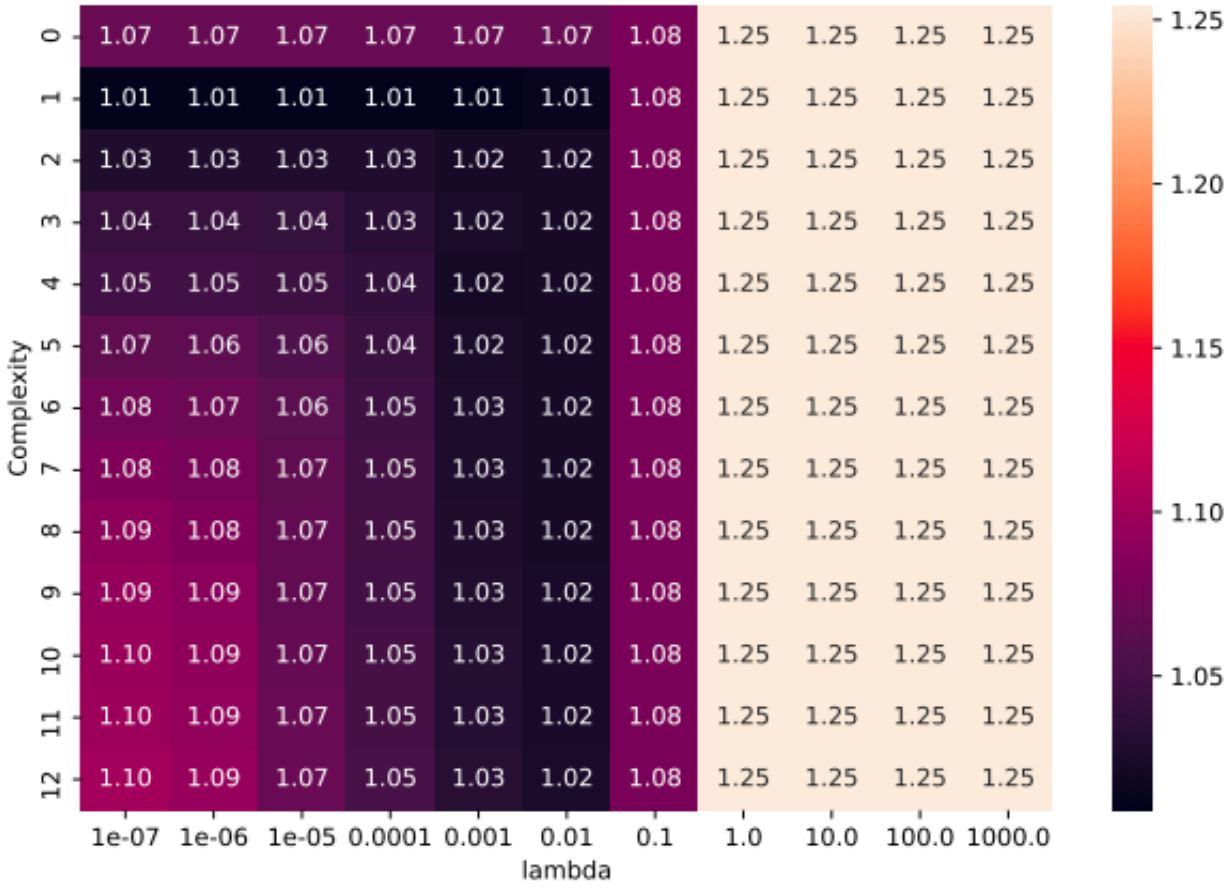
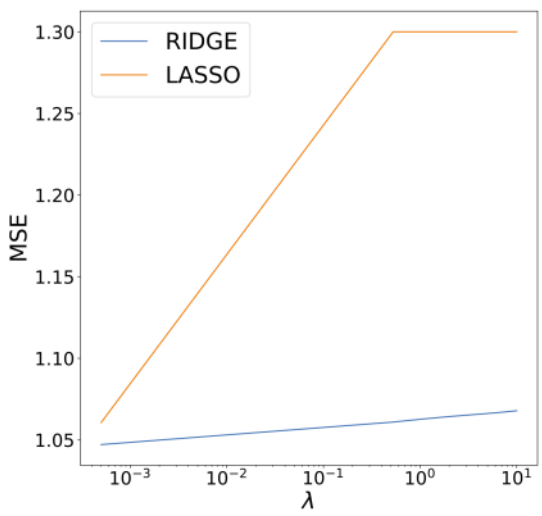
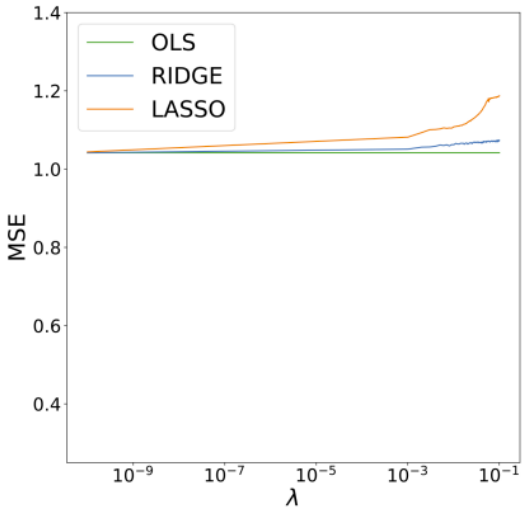


Figure 9: Showing the optimal regularization parameter λ and complexity for the LASSO regression for a model with 500 data points. MSE-scores are shown from low (dark) to high (light). Model run using 5-fold cross validation. The optimal lambda is 0.01 for 500 data points.



(a) The MSE-score function for LASSO and Ridge regression as a function of λ (K= 0)



(b) MSE-score function for OLS, LASSO and Ridge regression as function of λ (K= 5). Be aware of scale differences.

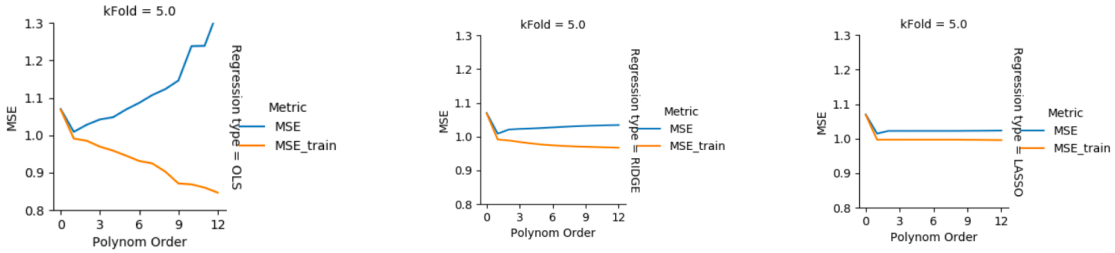
Figure 10: Characteristics of λ for different types of regression for polynomial order 4, obtained fitting models to the Franke Function with stochastic noise $N(\mu = 0, \sigma^2 = 1)$.

IV.I.5 λ dependence of LASSO and Ridge

LASSO and Ridge differ from OLS by introducing the regularization parameter λ. How it affects MSE-score values withouth K-fold cross validation (K = 0) for a polynomial 4 complexity is shown in Fig 10a.

The cost function displays a linear increase for both LASSO and Ridge with λ. Moreover, the slope for LASSO has a higher intercept and is steeper than that of Ridge regression. Another interesting feature of Fig. 10a, is that the slope for LASSO approaches a maxima when $\liminf \lambda_{\to 1}$, whereas Ridge has no such threshold. The corresponding norm-1 of $\hat{\beta}(\left|\hat{\beta}\right|)$ show a decrease for LASSO and Ridge (Appendix: Fig. 18).

Making a regression with K-fold cross-validation (K = 5), results in a similar behavior for MSE as a function of λ for the same complexity (p = 4) (Fig. 10b). However, as shown in Fig. 7 and Fig. 9 this is not the case for other model complexities. In this case, the MSE for Ridge and LASSO goes below that of OLS (p = 1).



(a) OLS regression

(b) Ridge regression at the optimal lambda λ = 1.0.

(c) Lasso regression at the optimal lambda λ = 0.01.

Figure 11: Showing test and training MSE as a function of complexity for each regression at the optimal lambda (with exception to OLS which does not depend on lambda), when population size is 500 data points. Model using 5-fold cross validation. The plots share the same y axes limits for comparison

IV.I.6 Results of Test-Train split error

Training error continues to decrease with increasing complexity (Fig.12). The exception is the LASSO regression. The LASSO test and training error drop and then level off. The test and train error for OLS and Ridge however, separate with varying complexity. Both show a minimum test error at polynomial order 1 and then a continuous increase with varying complexity

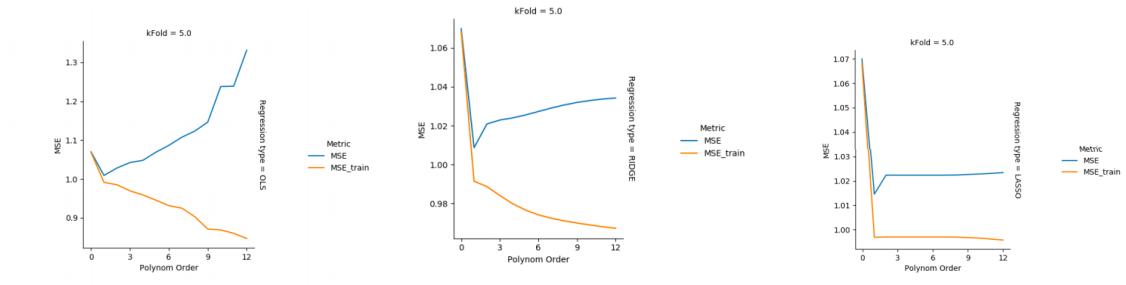
When comparing the test train splits for each regression, it can be seen that OLS has the largest separation of test and training error (Fig. 11), whilst LASSO regression has the least separation.

IV.I.7 Overview of model fitting

Table 1 sums up the results for OLS, LASSO and Ridge Regression for the Franke Function with stochastic noise ($\mathcal{N}(\mu = 0, \sigma^2 = 1)$). Ridge regression gave the best fitted model to the data, followed by LASSO and OLS.

Table 1: Best fits for OLS, Ridge and LASSO regression for the Franke Function with noise (N (μ = 0, 2 = 1)) for 500 datapoints (K = 5).

It is not very realistic to have dataset without noise, however, in this case OLS performed best, followed by Ridge and LASSO (Appendix: Fig. 19- 21). For OLS, MSE was 0.000 at polynomial degree 8, whereas for Ridge, this was not achieved before p = 12. LASSO reached MSE = 0.001 for this complexity.



(a) OLS regression shows almost linear increase in test MSE and linear decrease in training MSE.

(b) Ridge regression shows minimum test MSE at polynomial order 1. Test and training MSE slowly separate with varying complexity. Optimal λ = 1.0

(c) LASSO regression also shows minimum test MSE at polynomial order 1 and shows fairly constant test and training MSE with varying complexity. Optimal λ = 0.01.

Figure 12: Showing test and training MSE as a function of complexity for each regression at the optimal lambda (with exception to OLS which does not depend on lambda), when population size is 500 data points. Model using 5-fold cross validation. The plots here do not share the same y axes limits, this is intended to show the individual regression behaviour.

	Complexity	λ	MSE-score	Bias	Variance
OLS	1	0.00	1.009064	-0.134199	0.0750737
Ridge	1	0.1	1.009005	-0.133474	0.0742906
LASSO	1	0.0001	1.009062	-0.133799	0.0746723

IV.II Results of Yellowstone Terrain modelling

A GEOTIF-file was downloaded and reduced as described in the Methods-section. The same procedure and analysis as for the Franke Function was performed on a landscape from Yellowstone National Park, near Yellowstone Lake.

The reduced map and fitted models is presented in Fig 13. Both the details from the map, compared to that of the processed data, and the MSE-score gives an indication of how well the given model fit the data. The figure shows that OLS is the best regression method for such a varied landscape (MSE = $1.35 \cdot 10^3$, R2 = 0.6), followed by Ridge (MSE = $1.91 \cdot 10^3$, R2 = 0.4) and LASSO regression (MSE = $3.51 \cdot 10^3$, R2 < 0.01).

The R2-scores for OLS and Ridge generally increases with polynomial order before reaching a maxima, while LASSO does not display such a strong correlation between R2-score and complexity, but rather fluctuates around a constant (Appendix: Fig. 22). When it comes to the bias-variance trade-off it shows that MSE is close to constant with polynomial order for LASSO, whilst having clear trade-offs for both OLS and Ridge (Fig. 14).

Mostly MSE decreases for OLS and Ridge with complexity, and they both display a negative bias. Bias-variance trade-off had the same behavior independent of λ for both LASSO and Ridge, meaning the graphs looked similar to that of Fig. 14 (Appendix: Fig. 24, Fig. 25).

Using K-fold cross-validation to train the models, resulted in different train and test errors for OLS, Ridge and LASSO (Fig. 15, Appendix: Fig.23). Overall, OLS and Ridge has a decrease in both test error and train error, with train error having the steepest decline, with model complexity. However, for LASSO, the train error remain stagnant with higher order polynomials, whereas the test error oscillate in an irregular manner below and above the training error value.

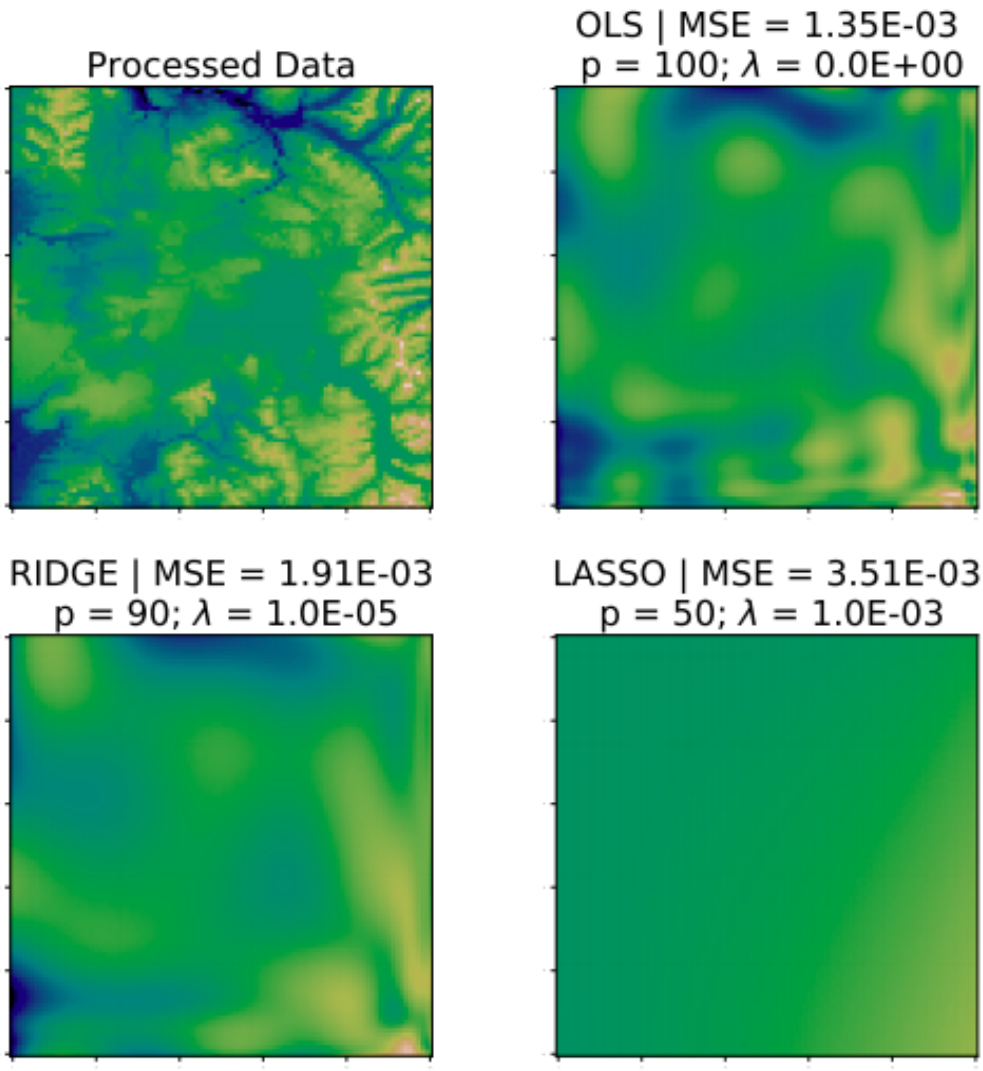


Figure 13: Heat map for elevations (blue (low) - orange(high)) in a real landscape from Yellowstone National Park (reduced with sel = 36), and for three regression models with best fit for the given data (λ:Regularization parameter, p: polynomial order, K = 5).

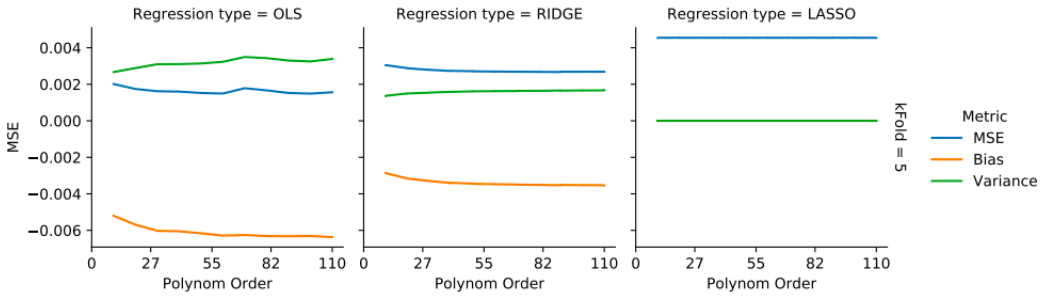
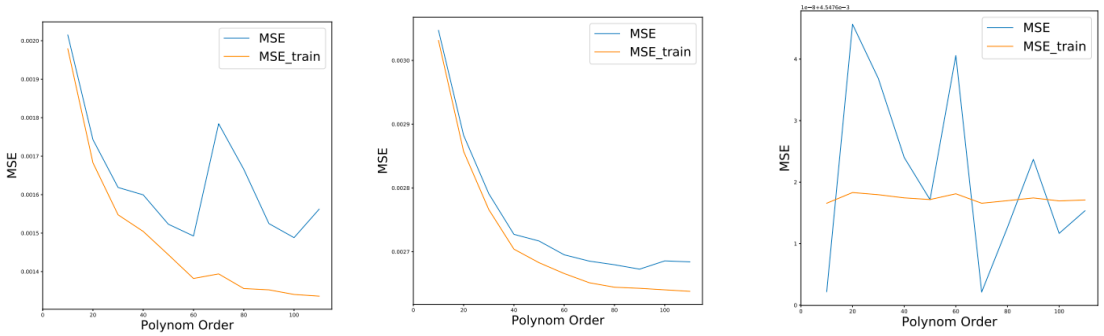


Figure 14: Bias-Variance trade-off for OLS, Ridge and LASSO respectively, when fitting a model to reproduce Yellowstone terrain data (K = 5).



(a) OLS test and training error as a function of polynomial order. Training and test error decrease with complexity. There is a peak in the test error at polynomial order 70 and 80.

(b) Ridge test and training error as a function of complexity. Training error and test error decrease gradually with complexity

(c) LASSO test and training error as a function of complexity. Test error is unstable with varying complexity and goes below training error

Figure 15: MSE of test and train samples, using K-fold cross validation (K = 5) when fitting a model to reproduce Yellowstone terrain data. Be aware of different axis scales, and possibly compare with Fig. 23 in the appendix.

V Discussion

The Franke function was used to investigate weaknesses and strengths of three methods of regression, encompassing OLS, Ridge and LASSO. An optimal model was fitted by minimizing the MSE-cost function through tuning hyper parameters, such as polynomial complexity and the regularization parameter λ. Thereafter an optimal model was made for terraindata from Yellowstone National Park, using the same method.

Firstly, let's have a look on model stability and time consumption. One crucial thing, for reasonable computational time consumption, is illustrated in Fig. 2a. The main point is that, more data points result in stable MSE but also require increasing amount of computation time. Unclear is the relative increase in computational time for small sample sizes. One should note, that the computational time of the LASSO method is depending on the chosen tolerance level, maximal number of iterations and chosen λ.

An additional time consuming part is the model complexity. The complexity directly enters the matrix size which will be inverted. Moreover, the complexity has direct effects on the MSE. For a fixed polynomial order the MSE becomes stable around one when there are many more data points than fitting parameters as figure 2b, 2c and 2d demonstrate.

On the contrary, when the number of data points is reduced to the order of the number of parameters or even below, the MSE deviates strongly. In regressions with regularization parameters, these effects are not as strong as in plain OLS regression. Thus, a regularization parameter can be tuned to soften the effects of sparse data samples.

Maybe another obvious conclusion is, that the number of data points limits the complexity, i.e. the polynomial order, of the model. This is because without k-fold cross validation the model is trained and evaluated on the entire data set, presenting a unrealistic predictive accuracy. Unrealistic because, it means with increasing polynomial order the model can fit the noise with greater accuracy. Fitting the noise is undesirable, it is more desirable to have a model that has a low generalization error (out of sample error), which is a measure of the algorithms ability to predict outcomes for previously unseen data.

The optimal λ, i.e the λ giving the lowest MSE, can roughly be seen in Fig.7 and Fig.9. The range of λ could be narrowed to give a more precise optimal value, however looking at the MSE it is not expected to give much more insight. This however, is only for modelling the Franke function with 500 data points. If another size is modelled, another optimal λ is expected. This paper has discussed a little on the number of data points, as shown above. To be critical of this papers' work, it would be a significant improvement to iterate over many data sizes and find the optimal λ for each. Otherwise OLS has an

edge over the other regressions when making a comparison prior to λ optimization, as it has no λ dependency.

It is counter-intuitive in a sense that, the data size can be optimized for better model fitting. Especially since in supervised learning, data is usually sparse and a fixed size, dependent on a physical experiments' results. Data can be costly, in classification problems each item must be hand labelled. However by modelling the Franke function, any data size is possible, leading to the conclusion that data size perhaps could be optimized to give an even lower MSE. Future improvements in this area would hope to enable the discussion of the optimal data size for modelling the Franke function. It is hypothesized that there is some trade-off present. If the data size is reduced to increase model fitting, then resolution is lost in the process.

This directly affects all benchmarking results from the Franke function because the number of generated points dictates the maximal model complexity ore vice versa. For a fixed number of Franke function sample points and without k-fold cross validation, it is expected that the MSE constantly decreases or stagnates at a low value. This hypothesis is not fully supported by the Figs. 3 - 6. While for Ridge and LASSO the MSE stabilizes, the OLS in Fig. 3 however, shows a sudden increase in MSE at polynomial order 12.

It turns out that the instabilities in Fig. 2c, Fig. 3 and Fig. 4 for high polynomial orders are numerical effects of direct matrix inversion form numpy.linalg. Switching to singular value decomposition and defining a cut-off for inversion of singular values below 10 12 resolves this issue. However, the analysis is left unchanged, but the reader should keep in mind that blow-ups for high model complexity are due to this issue. A plot is created in the appendix comparing OLS, Ridge and LASSO as a function of complexity and k-fold cross validation after this was resolved (Fig.17). Dealing with the terrain data, a trade-off between computational time and complexity needed to describe the rugged terrain leads to data reduction.

The results from the previous section give some interesting insights into the complex dependencies of regression on the different hyperparameters in different situations. Without k-fold cross validation the OLS model shows the highest predictive accuracy on a fixed number of samples from the Franke function (Fig. 3 - 6), as expected. For the Franke Function without noise, OLS also perform best with K = 5. This illustrates that when there are little noise in the dataset, or when variability is an important feature of a dataset, OLS is a good choice of model.

However, when noise is added, it is not so easy to choose the best model (table 1). Using a regularization parameter flattens out the effect of stochastic noise, but when using this penalty, you are also in risk of reducing complexity from the model that is vital to describe the given set of data. Even though Ridge performed best for the Franke function

with K = 5, LASSO and OLS differed only by a few decimal numbers.

The same conclusions could be made for the terrain data. When applied on the reduced terrain data from Yellowstone, OLS performed best amongst the three regression methods, followed by Ridge and LASSO respectively (Fig 13). The results was obtained using K-fold analysis (K = 5), for the model's capability to predict unseen data, as discussed above.

It is important to remember, that the outcome might have been different for a another number of data points. For example, by reducing the sample size, there could be more issues with overfitting due to having a simpler dataset than model complexity, so that LASSO and Ridge would have an edge compared to OLS. However, when choosing every 36 data point, OLS still performs best.

Taking a closer look, an investigation was made on how the regression models depend on λ. From Fig. 10a it becomes evident that with K = 0 and p = 4, MSE for both LASSO and Ridge increases with higher λ's. This means that OLS is a better option ($\lambda = 0$) for this complexity, by minimizing the cost function. Since LASSO has a steeper slope, Ridge should be a better fit than LASSO between the two, for the Franke Function without K-fold analysis. This is consistent with OLS being the best model with K = 0. However, it is important to note that with K = 0, the model is trained and tested on all data. This gives a superficially low MSE for OLS, having all parameters untouched by a penalty. Nonetheless, there are certain features worthy of discussion.

Since LASSO regression shrinks more coefficients to become zero as λ increases, the model would be more prone to underfitting, due to effectively increasing bias by reducing model complexity. With a dataset which display several behaviours, such as the Franke Function, LASSO might thus be a poor choice of model. In contrast Ridge does not shrink coefficients to zero, hence giving a lower MSE-score (Fig. 10a).

Another interesting feature of Fig.10a, is that the slope for LASSO approaches a maxima when $\lim_{\lambda \to 1}$, whereas Ridge has no such threshold. A possible explanation might be that with $\lambda > 100.5$ there are few β values left to reduce to zero in the LASSO regression. This is exactly what Fig. 18 displays for norm-1 of $\beta's$. Thus, increasing λ further would make only small impacts on the fitted model. As shown in the results, for polynomial order 5, the LASSO regression has minimised all but one parameter to zero. In fact, using $\lambda = 0.1$ results in all models with varying complexities to have only one non-zero beta parameter.

Interestingly, Fig. 10b demonstrates that there is not much of a difference in MSE's λ dependence with K = 5 for p = 4, compared to that of K = 0. Since Fig. 7, Fig. 9 and 1 all show that a given λ for Ridge and LASSO with p = 1, is slightly better than OLS, this

highlights the importance of tuning the right model complexity.

In the book, Hasties et al, Figure.2.11 shows a graph of test and train prediction error with increasing model complexity. Comparing this graph to Fig. , a few similarities and differences can be observed.

Firstly, it can be seen that both show a trend of decreasing training error. By increasing the polynomial order the training data will be fit with more degrees of freedom. This leads to an over-fit model and does not perform well on a test data. In both figures the test error decreases initially and then increases; a 'sweet spot' is found in between the two, the test error minimum. Fig. shows this 'sweet spot' at polynomial order 1, whereas Hastie et al predicts the 'sweet spot' somewhere in the middle, although there is no scale and the regression model is not stated.

LASSO shows different results. The test and training error are almost constant after polynomial order 2 This is most likely due to the hyperparameter λ. The beta parameters have been constrained to 0 and hence increasing the complexity does not cause over-fitting to noise, and hence does not result in increasing test error.

The trend of increasing test error with complexity is resulting from over-fitting (Fig. 12. and Fig. 3. Over-fitting occurs when the model fits the training data but has less generalization towards the unseen test data. Over-fitting increases with polynomial order as there are more degrees of freedom to fit the training data (and the noise), but again less generalization to unseen or test data.

It is important to note that, 5-fold and 10-fold include a random shuffle of the data set prior to dividing into folds. This leads to the possibility of an unlucky and lucky choice of test data, especially for a small sample size. In k-fold cross validation the means scores are taken to give a more accurate estimate of test error. With more folds it is predicted that an unlucky choice of test data can result in a skewed MSE. However Fig. 3 (c) shows that the MSE for 10-fold is lower than 5-fold cross validation at high orders.

As expected, the bias generally decrease with polynomial complexity, whereas the variance is increasing, with the exception of LASSO regression, which was mostly constant (Fig. 14). To explain this dissimilarity, one has to take the train-test errors into account (Fig. 15). It shows that for both both OLS and Ridge, the model perform better on the train data with higher polynomial order, whereas this is not the case for LASSO.

The train-test error for LASSO display an instability (15), that is most likely due to having a regularization parameter penalty that reduces many parameters to zero. Thus the model will not benefit from adding a higher complexity. As a consequence, for a highly varied landscape, where a high polynomial order might be need to explain gen-

eral trends, LASSO is unable to predict unseen data. This correspond with what is seen for the Franke Function (Fig 8).

In addition, for OLS and Ridge, the bias was negative. This might be due to calculating the bias squared from simply rewriting 16, since f_i remains unknown. In theory, this should be a positive value, but if calculations of σ^2 are affected by SVD, this might lead to negative bias. However, since the same calculations are done for Ridge, LASSO and OLS, the conclusion still holds.

Few articles explore terrain data with Machine Learning. There are some groups using Machine Learning for mapping terrain, for the use of low aerial vehicles [1]. It was found however their Machine Learning algorithms were based on Neural Networks (NN), which is beyond the scope of this paper. A future improvement could be to compare their models and some of their data (if available) if the study in this paper employed NN. This would really give a benchmark to test the methods in this paper and seek improvements from an experimental project.

To sum things up, all results are strongly dependent on the chosen hyperparameters, such as number of data points, regularization parameter and obviously on the regression type and polynomial order.

VI Conclusion

The main conclusion made from the discussion of the regression methods has been that the model applied is very influenced by the data. This perhaps is quite a logical statement that could have been made beforehand, but this paper has been able to explore this. The model which fitted the Yellowstone terrain data the 'best' was the OLS regression. The MSE calculated is 0.00135, the lowest among the three regression types. This is likely that such a steep terrain requires a model that doesn't restrict the parameters.

The model which fits the Franke function the 'best', is not so clear. OLS, Ridge and LASSO models show a reduced MSE at low complexity, and the difference between them is not significant. The test train splitting and the bias variance trade-off show other trends in addition to this.

Possible future improvement for this Machine Learning model would be to investigate further the regression models as a function of noise. This was not explored fully. By cutting out the noise, OLS showed to be the 'best' model for the Franke function, although this was not analysed fully and not examining the other parameters in depth.

References

[1] Guang-Bin Huang A. Agarwal C. . T. Yeu, Meng-Hiot Lim and Yew-Soon Ong. A. *New machine learning paradigm for terrain reconstruction.* IEEE Geoscience and Remote Sensing Letters, 3(3):(382–386), July (2006).

[2] Richard Franke. *Information visualizations used to avoid the problem of overfitting in supervised machine learning.* In HCI in Business, Government and Organizations. Supporting Business, pages 373–385. Springer International Publishing,, 2017.

[3] J. Gray G.W. Dunnington and F.E. Dohse. *Carl Friedrich Gauss: Titan of Science.* Mathematical Association of America, 2004.

[4] Arthur E. Hoerl and Robert W. Kennard. *Ridge regression: Biased estimation for nonorthogonal problems.* Technometrics, 12(1):(55–67), (1970).

[5] Robbie T. Nakatsu. *A critical comparison of some methods for interpolation of scattered data.* NPS Archive Calhoun,, 1979.

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

[7] Jerome H. Friedman. Trevor Hastie, Robert Tibshirani. *The Elements of Statistical Learning.* Springer, 2016.

VII Appendix

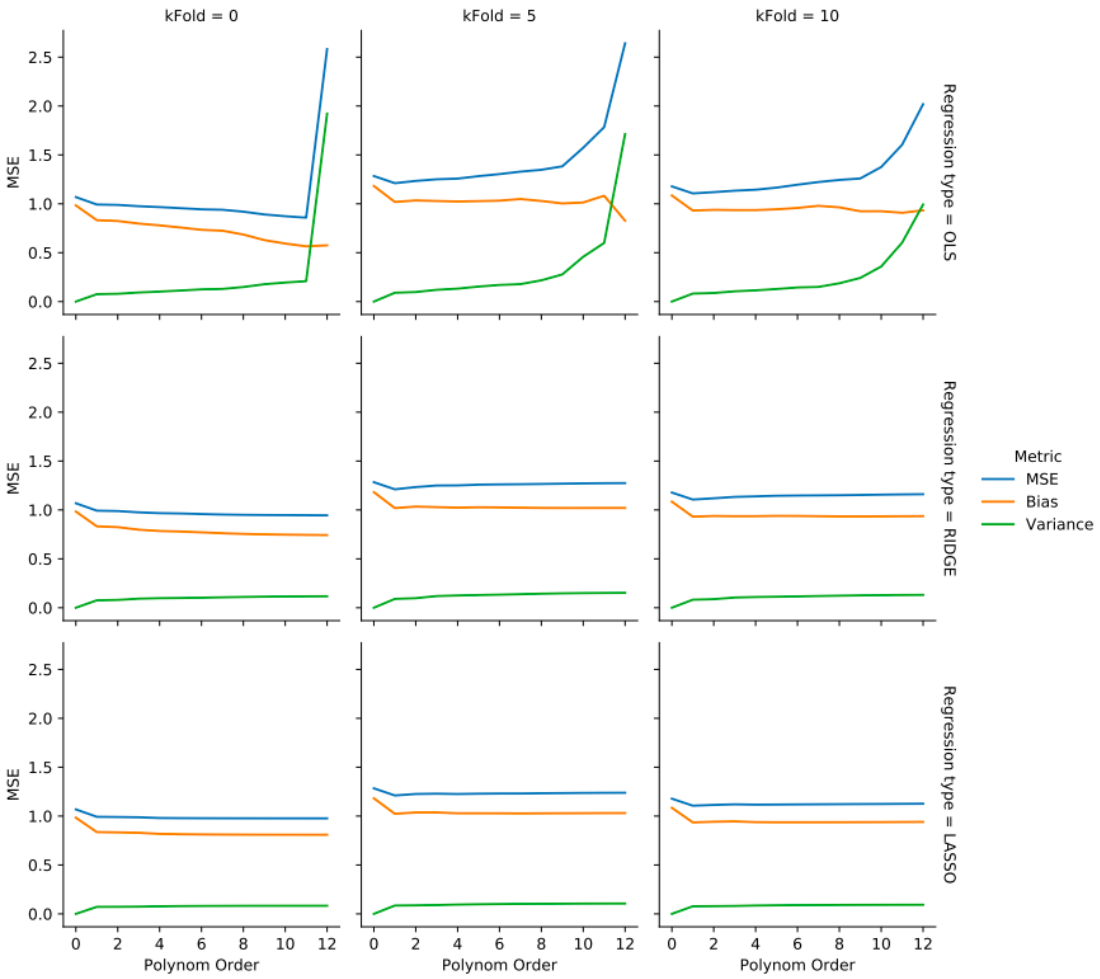


Figure 16: Bias Variance Trade-off comparing OLS, Ridge and LASSO. The numberof data points in the population sample is 500. The regularisation parameter is set to $\lambda = 0.001$

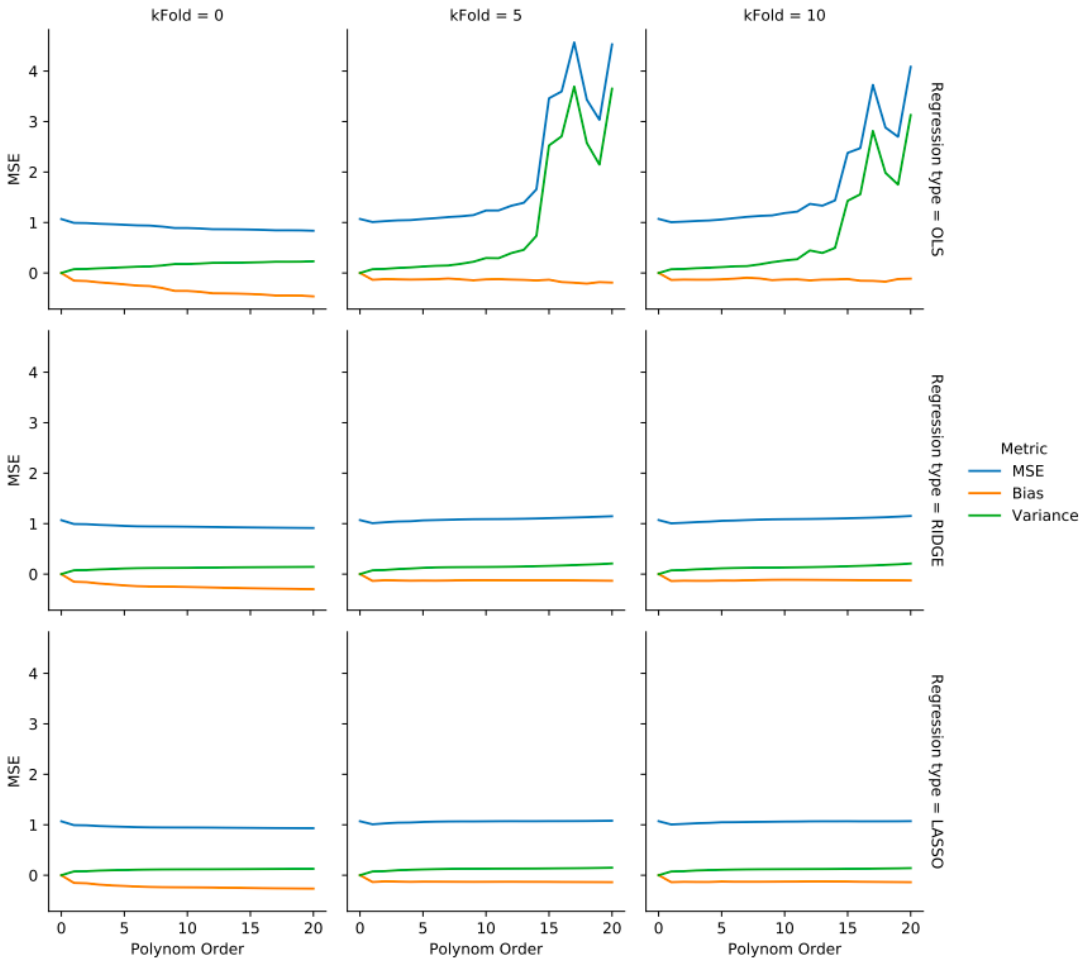


Figure 17: Bias Variance Trade-off comparing OLS, Ridge and LASSO. The number of data points in the population sample is 500. The regularisation parameter is set to $\lambda = 10^5$

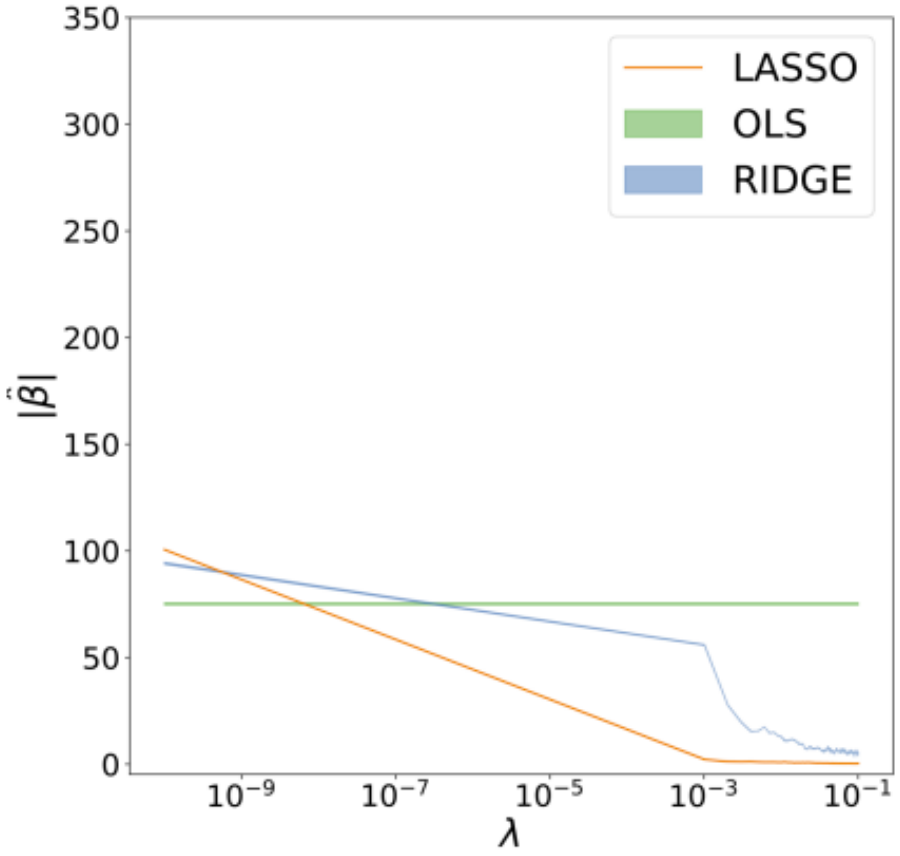


Figure 18: Norm 1 of β as a function of λ for LASSO, Ridge and OLS regression (polynomial order 4).

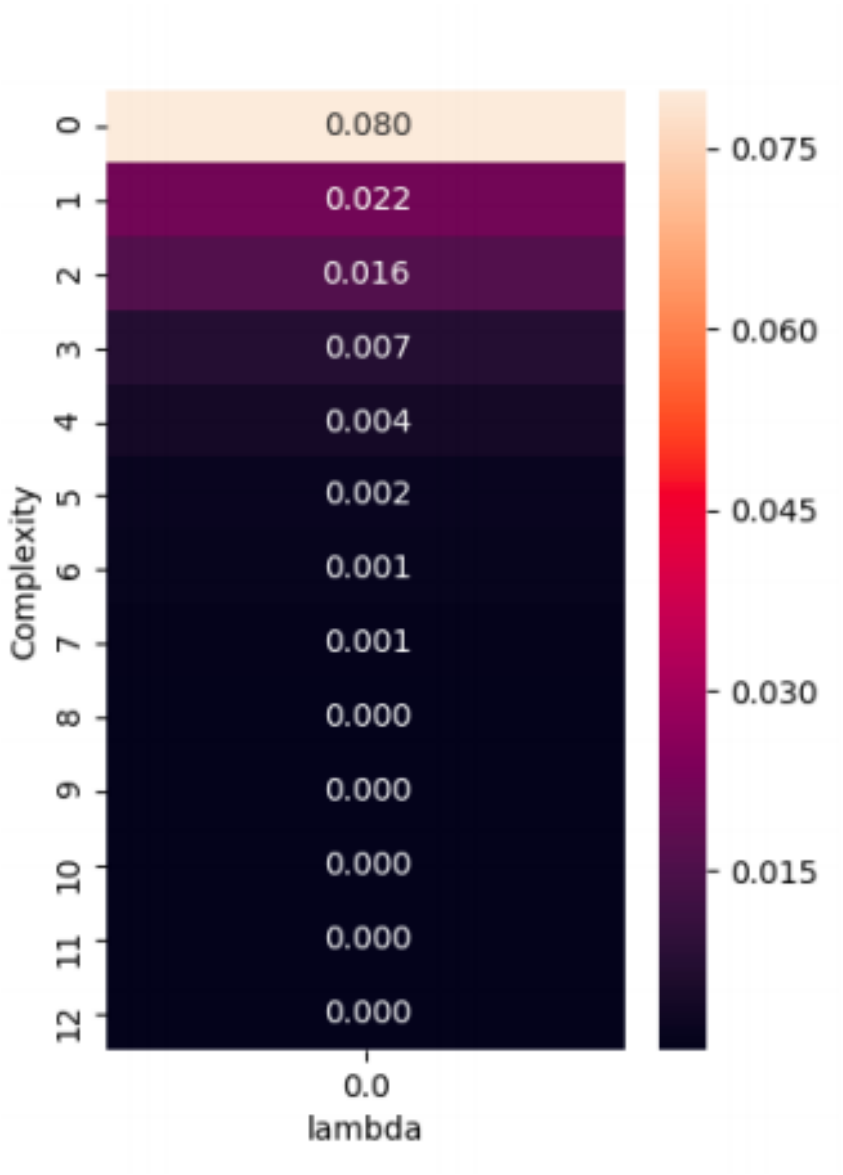


Figure 19: Heatmap showing MSE as a function complexity for OLS regression without any added noise. Population size is 500 data points. 5-fold cross validation. The heatmap shows that OLS performs better with greater complexity without added noise. In the bottom it can be seen that MSE equals 0 from polynomial order 8 and above.

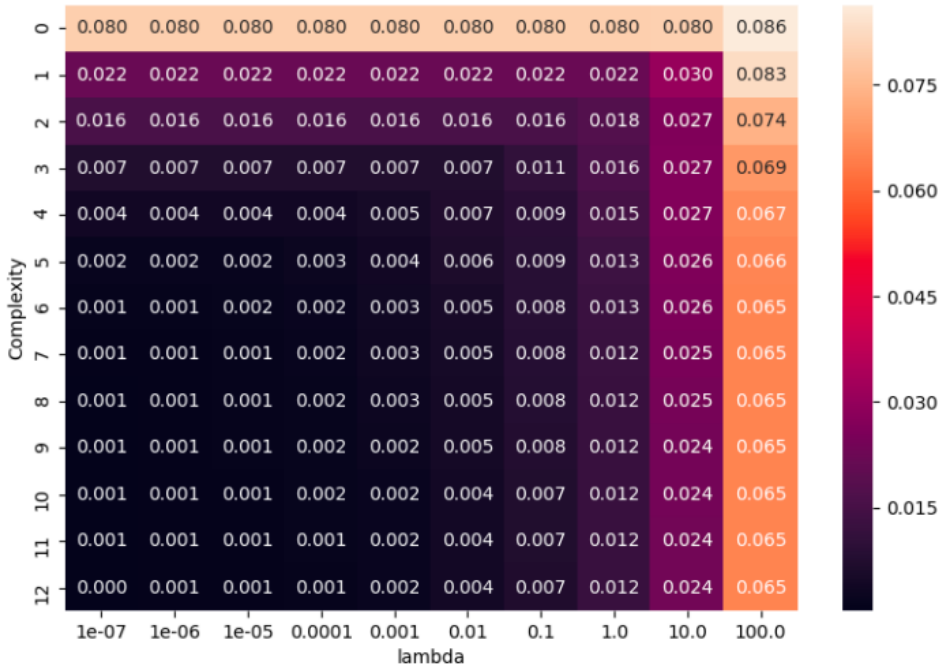


Figure 20: Heatmap showing MSE as a function of lambda and complexity for Ridge regression without any added noise. Population size is 500 data points. 5-fold cross validation. The heatmap shows that Ridge will likely continue to perform better with smaller lambda and greater complexity without added noise. In the bottom left corner it can be seen that MSE equals 0.000.

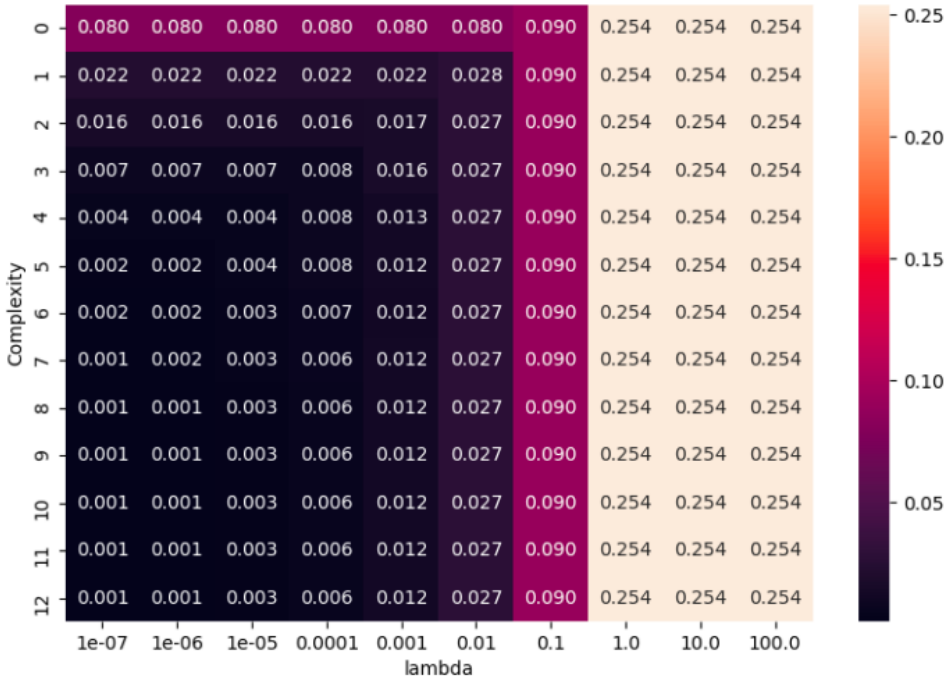


Figure 21: Heatmap showing MSE as a function of lambda and complexity for LASSO regression without any added noise. Population size is 500 data points. 5-fold cross validation. The heatmap shows that LASSO performs better with smaller lambda and greater complexity without added noise. In the bottom left corner region it can be seen that MSE equals 0.001.

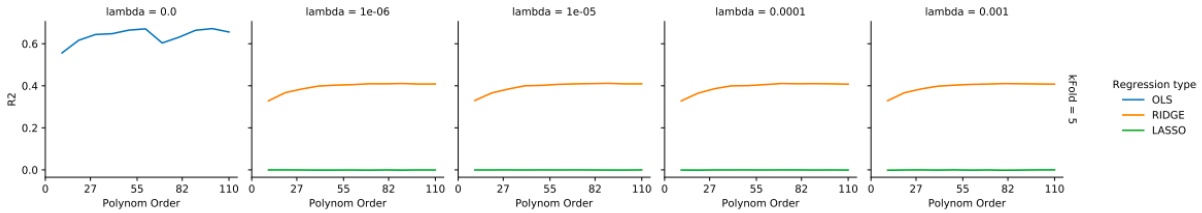


Figure 22: R2-score for different λ values for OLS, Ridge and LASSO regression when fitting for Yellowstone terrain data.

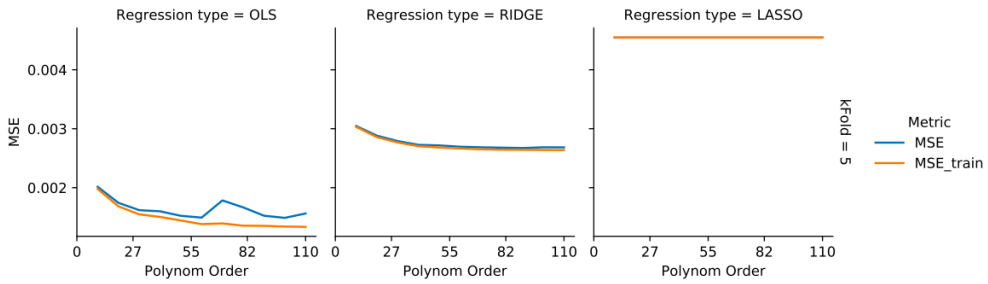


Figure 23: Train and test MSE as a function of polynomial order for OLS, Ridge and LASSO regression respectively. The regression methods were used to fit a model to reproduce Yellowstone terrain data.

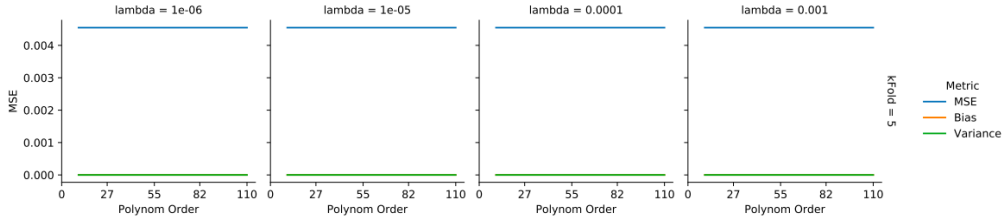


Figure 24: Bias-Variance trade-off for LASSO regression with different λ values for fitting the model to Yellowstone terrain data.

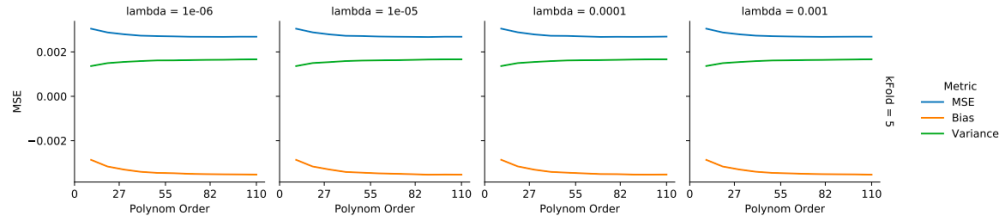


Figure 25: Bias-variance trade-off for Ridge regression with different λ values, fitting the model to Yellowstone terrain data.