FYS-STK4155 – Applied data analysis and machine learning
Project 1 - Regression analysis and resampling methods

Jon A Ottesen

(Dated: 7. oktober 2019)

The main topic of this projects is to study different regression methods including Ordinary Least Squares, Ridge regression and Lasso regression. These methods in conjunction with resampling techniques such as k-fold cross-validation are used in polynomial fitting on the 2-dimensional Franke function surface with added Gaussian noise and terrain data. The different methods give varied result in reproducing the original surface with .... having the highest MSE and R2 compared to ... with MSE being. Finally real-life data is the analysed using the same techniques as the Franke function with OLS giving the best fit.

I. INTRODUCTION

The first paper on regression methods was published by Legendre in 1805 and Gauss in 1809 about the least square method[**?**]. Albeit being a relatively simple theory developed before the computational era of statistics its importance cannot be overstated, and is still prominent and even outperforming newer more complex methods in some cases. The simple nature of the least square theory makes it a excellent starting point for further studies in regression theory while still giving decent results in real-life data-analysis. Especially when applied together with resampling methods which has a vital role in modern statistical data analysis.

Regression methods are often used in conjunction with measured data to determine the underlying patterns. Unlike in most research where linear regression is used as a tool for data-analysis, it will here be the main object of study itself. To stimulate a more real-life usage of linear regression actual data from ... is used coupled with generated data from Franke's function. The central themes are thus resampling methods, error analysis such as the mean squared error, the bias-variance tradeoff and most importantly the main linear regression methods themselves: OLS, rigde regression and lasso regression and their strengths and weaknesses.

II. THEORY

As a basis for parts of the theory below we will assume the data follows this form:

$$\boldsymbol{y} = f(\boldsymbol{x}) + \boldsymbol{\epsilon} \tag{II.1}$$

where $\boldsymbol{y}$ is the data-sample, f is the function describing the data and $\boldsymbol{\epsilon}$ is the noise of the data ... distributed with a mean of 0.

A. Moments and error analysis

1. *Moments in statistics*

To describe, analyse and understand data, statistical knowledge is essential. Various moments in statistics are therefore used when describing key elements of data, models or functions. In our case to describe and understand the model created with linear regression, and for this the the 1^{st} and 2^{and} order moments will be used.

For a real valued continuous function $g(x)$ for real-valued x, the n-th moment is given by

$$\mu_n(c) = \int_{-\infty}^{\infty} (x-c)^n g(x) dx \tag{II.2}$$

with c as a real variable. By restricting g as a normalized non-negative function the 1^{st} mo-

ment around $c = 0$ is the mean value given by

$$\mu = \int_{-\infty}^{\infty} xg(x)dx. \tag{II.3}$$

Higher order moments are often defined around the mean:

$$\mu_n(\mu) = \int_{-\infty}^{\infty} (x-\mu)g(x)dx \tag{II.4}$$

to provide more qualitative information about the distribution. The 2^{nd} moment is the variance

$$\mu_2(\mu) = \sigma^2 = \int_{-\infty}^{\infty} (x-\mu)^2 g(x)dx \tag{II.5}$$

where σ is the standard-deviation.

So far I have only looked at the continuous case where the probability distribution is know. In the non continuous case with a sample of finite size the true moments of the distribution can realistically only be approximated. There is however theoretically possible to get the correct and this is covered in section ...

For a finite sample of size n taken from the function g the 1^{st} moment i.e mean value is for the sample given by:

$$\overline{g^s} = \frac{1}{n}\sum_{i=1}^{n} g_i^s \tag{II.6}$$

The 2^{an} moment i.e variance is for the sample

$$\text{Var}(g^s) = \frac{1}{n}\sum_{i=1}^{n} (g_i^s - \overline{g^s})^2 \tag{II.7}$$

where in both cases g^s is a sample of data from the distribution $g(x)$. Note however that $\overline{g^s}$ and $\text{Var}(g^s)$ are only approximations to the real mean μ and variance μ_2 for the distribution g.

A new finite sample of size n is given from function II.1, and the function $f(\boldsymbol{x})$ is approximation from the sample by regression methods as $\hat{\boldsymbol{g}}$. From this the mean and the variance of the noise $\boldsymbol{\epsilon}$ can be approximated by

$$\overline{\epsilon} \approx \frac{1}{n}\sum_{i=1}^{n} (y_i - \hat{g}_i) \tag{II.8}$$

$$\text{Var}(\epsilon) \approx \frac{1}{n-m}\sum_{i=1}^{n} ((y_i - \hat{g}_i) - \overline{\epsilon})^2 \tag{II.9}$$

since

$$\boldsymbol{y} - \hat{\boldsymbol{g}} = g(\boldsymbol{x}) + \boldsymbol{\epsilon} - \hat{\boldsymbol{g}} \approx \boldsymbol{\epsilon}. \tag{II.10}$$

For all future reference of the moments the following convention will be used

$$\mu_n(c) = \langle (x-c)^n \rangle . \tag{II.11}$$

2. *Error analysis*

There are a multitude of different cost functions for evaluating the error in a model. Among them are the mean square error(MSE) and the R^2 score function which are respectivly given by

$$MSE(\boldsymbol{y}, \tilde{\boldsymbol{y}}) = \frac{1}{n}\sum_{i=1}^{n} (y_i - \tilde{y_i})^2 \tag{II.12}$$

$$R^2(\boldsymbol{y}, \tilde{\boldsymbol{y}}) = 1 - \frac{\sum_{i=1}^{n}(y_i - \tilde{y_i})^2}{\sum_{i=1}^{n}(y_i - \overline{y})^2}. \tag{II.13}$$

Notice how for the variance given in equation II.9 with $m = 0$ and a mean noise of $\mu = 0$ we are left with the equation for the MSE.

B. Linear regression methods

The very basis of linear regression is based around the assumption that the form of the data can be written on the form of equation II.1. This form can again be written as

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon} \tag{II.14}$$

where $\boldsymbol{y}, \boldsymbol{\epsilon} \in \mathbb{R}^{n\times 1}$ whereas $\boldsymbol{\beta} \in \mathbb{R}^{p\times 1}$ and $\boldsymbol{X} \in \mathbb{R}^{n\times p}$ is the design matrix.

The goal of linear regression is thus to approximate $\boldsymbol{y}$ with

$$\tilde{\boldsymbol{y}} = \boldsymbol{X}\boldsymbol{\beta} \tag{II.15}$$

by optimizing $\boldsymbol{\beta}$.

1. *Ordinary least squares*

The ordinary least square method optimizes $\boldsymbol{\beta}$ by minimizing the MSE cost function

$$C(\boldsymbol{\beta}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i) \tag{II.16}$$

$$= \frac{1}{n}\left[(\boldsymbol{y} - \tilde{\boldsymbol{y}})^T(\boldsymbol{y} - \tilde{\boldsymbol{y}})\right] \tag{II.17}$$

$$= \frac{1}{n}\left[(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})\right]. \tag{II.18}$$

The steps is to take following derivative

$$\frac{\partial \boldsymbol{C}(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = 0 \tag{II.19}$$

which result in the OLS formula

$$\boldsymbol{\beta} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}. \tag{II.20}$$

A more rigorous version of the same derivation is found in ...

Another way of deriving the OLS formula is by considering

$$\tilde{\boldsymbol{y}} = \text{proj}_{\text{Col X}}\boldsymbol{y}. \tag{II.21}$$

Following this there exist a $\boldsymbol{\beta}$ such that

$$\tilde{\boldsymbol{y}} = \boldsymbol{X}\boldsymbol{\beta}. \tag{II.22}$$

A consequence of having $\tilde{\boldsymbol{y}} = \text{proj}_{\text{Col X}}\boldsymbol{y}$ is that $\boldsymbol{y} - \tilde{\boldsymbol{y}}$ is orthogonal to the vector space spanned by $\boldsymbol{X}$ as stated by the Orthogonal Decomposition Theorem SITER. Therefore any column in $\boldsymbol{X}$ must be orthogonal to $\boldsymbol{y} - \tilde{\boldsymbol{y}}$ such that

$$\boldsymbol{X}_i \cdot (\boldsymbol{y} - \tilde{\boldsymbol{y}}) = 0. \tag{II.23}$$

This implies

$$\boldsymbol{X}^T(\boldsymbol{y} - \tilde{\boldsymbol{y}}) = 0 \tag{II.24}$$

and finally

$$\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\beta} = \boldsymbol{X}^T\tilde{\boldsymbol{y}} \tag{II.25}$$

which can be rearranged to equation II.20.

Both derivations are equal in the sense that they yield the OLS formula, but their interpretations are different.

For the two remaining methods: ridge and lasso regression their derivations will not be carried out. Instead both of their derivations can be found at...., but knowing them will not have any impact on the interpretations of the results.

2. *Ridge and Lasso regression*

Ridge and lasso regression are two shrinkage methods in linear regression. They work by imposing a penalty on the regression coefficients based on their size. This is done by minimizing their respective cost functions, and for ridge regression this cost function is given by **DOBBELSJEKK DETTE**

$$\left[(\boldsymbol{y} - \tilde{\boldsymbol{y}})^T(\boldsymbol{y} - \tilde{\boldsymbol{y}}) + \lambda\boldsymbol{\beta}^T\boldsymbol{\beta}\right] \tag{II.26}$$

which is variant of the MSE with $\lambda \geq 0$. The coefficients for $\boldsymbol{\beta}$ in ridge regression can be written in the following closed form

$$\boldsymbol{\beta} = (\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I})^{-1}\boldsymbol{X}^T\boldsymbol{y} \tag{II.27}$$

where $\boldsymbol{I} \in \mathbb{R}^{p\times p}$. Further analysis of equation II.27 would reveal how the λ parameter shrinks terms i beta, but such information would not allow further insight when comparing the different methods.

Lasso regression much like OLS and ridge regression is a minimization problem. However unlike OLS and ridge regression there exist no analytical solution to the following cost function minimized in lasso regression:

To solve lasso regression the implementation of gradient decent types of algorithms are a necessity. Such algorithms are outside the scope of this project and will not be covered in any depth, instead preexisting tools will be utilized.+

Before ending the subsection about Linear regression final part is to know about the confidence interval for each of the β_i terms. The 68% confidence intervals for the β_i-terms is given by

$$\sigma(\beta_i)^{\text{OLS}} = \sigma\sqrt{(\boldsymbol{X}^T\boldsymbol{X})_{ii}}. \tag{II.28}$$

Thus the variance of $\boldsymbol{\beta}^{\text{OLS}}$ is given by the diagonal of the square matrix $\boldsymbol{X}^T\boldsymbol{X}\sigma^2$ multiplied with the variance from the error in function II.1. For ridge the confidence interval is given by

$$\sigma(\beta_i)^{\text{Ridge}} = \sigma\sqrt{\left[\left(\boldsymbol{X^T X}+\lambda\boldsymbol{I}\right)^{-1}\boldsymbol{X^T X}\left(\left(\boldsymbol{X^T X}+\lambda\boldsymbol{I}\right)^{-1}\right)^T\right]_{ii}}.$$

3. SVD

Any $n\times p$ matrices $\boldsymbol{X}$ with rank r can be written as

$$\boldsymbol{X} = \boldsymbol{U\Sigma V}^T \tag{II.29}$$

with $\boldsymbol{U}$ and orthogonal $n\times n$ matrix and $\boldsymbol{V}$ a orthogonal $p\times p$ matrix. Whereas $\boldsymbol{\Sigma}$ is a $n\times p$ matrix where the first r diagonal elements are the singular values of $\boldsymbol{X}$. A small note is that $\boldsymbol{V}^{-1} = \boldsymbol{V}^T$ and $\boldsymbol{U}^{-1} = \boldsymbol{U}^T$ since they are orthogonal matrices.

By using the SVD the formulas for both OLS and ridge regression can be rewritten as

$$\boldsymbol{\beta}_{OLS} = \boldsymbol{V}(\boldsymbol{\Sigma})^{-1}\boldsymbol{U}^T\boldsymbol{y} \tag{II.30}$$

$$\boldsymbol{\beta}_{Ridge} = \boldsymbol{V}(\boldsymbol{\Sigma}^2+\lambda\boldsymbol{I})^{-1}\boldsymbol{\Sigma}^T\boldsymbol{U}^T\boldsymbol{y}. \tag{II.31}$$

Both derivations can be found in the appendix..... Further the confidence intervals for the β_i from equation II.28 can be rewritten as

$$\sigma(\beta_i)^{\text{OLS}} = \sqrt{\left(\boldsymbol{V}\left(\boldsymbol{\Sigma}^T\boldsymbol{\Sigma}\right)^T\boldsymbol{V}^T\right)_{ii}}\sigma \tag{II.32}$$

while the confidence interval for Ridge regression takes the following form:

$$\sigma(\beta_i)^{\text{Ridge}} = \sigma\sqrt{\left(\boldsymbol{V}\left(\boldsymbol{\Sigma^2}+\boldsymbol{\lambda}\right)^{-2}\boldsymbol{\Sigma^2 V^T}\right)_{ii}} \tag{II.33}$$

where $\boldsymbol{\lambda} = \lambda\boldsymbol{I}$.

C. Resampling and Bias-variance tradeoff

1. Reasmpling methods

There exist a multitude of different resampling methods, and common trope is to repeatedly refitting a model by drawing(often randomly) samples from a larger data set. By repeatedly drawing out samples from a larger set one can use the central limit theorem.

K-fold cross validation is based around the principle of dividing a data set into n-folds with equal size if possible and $n \leq$ (len of the data). Than n-1 of the folds are used a training data in linear regression while 1 is used as test data. Thereafter the prediction error, mean and standard deviation is evaluated among other things. This is redone n times but with a different fold for the test data each time such that all n-folds are used as test data. Finally the central limit theorem is applied to the measured values for the prediction error, mean, standard deviation etc. Albeit not essential the data set should be shuffled before diving into folds to avoid bias in the linear regression fit.

2. Bias-variance tradeoff

The MSE from equation II.12 can be rewritten as

$$MSE(\boldsymbol{y}, \tilde{\boldsymbol{y}}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2 \tag{II.34}$$

$$= E\left[(f + \epsilon - \tilde{f})^2\right] \tag{II.35}$$

$$\tag{II.36}$$

III. METHOD

A. Preparations

There are two sets of data used during this project. The first is self-generated through the 2D Franke function with added noise normally distributed, while the other set is terrain data from ... over ....

The self-generated data is generated with the Franke function on a $n \times n$[1] meshgrid with $x, y \in [0, 1]$ selected by random from a uniform distribution, and each point has a added normally distributed noise with $\mu = 0$ and $\sigma = 1$ in the z-direction. Thus the function used in the data-generation is

$$f_F(x, y) = F(x, y) + \mathcal{N}(0, 1) \tag{III.1}$$

with $F(x, y)$ the Franke function. Note that I used the *seed 42* when generating all the random numbers, this was to simulate a real-life data set where the measured data is static.

The terrain data is very similar to the self-generated data, the largest difference is the span of the x and y axis on a meshgrid.

[1] It's not requirement for grid to be $n \times n$, however making the grid $n \times m$ with $m \neq n$ would not have any noticable effect on the results for a reasonable sized n and m.

The last preparatory step before the analysis is to create a design matrix from the x,y datapoints in the meshgrid. As for all meshgrids x and y are $n \times n$ matrices, but these are both flattend:

$$\boldsymbol{x_f} = [x_{00} \;\; x_{01} \;\; x_{02} \ldots x_{0n} \;\; x_{01} \ldots x_{nn}]^T \tag{III.2}$$

$$\boldsymbol{y_f} = [y_{00} \;\; y_{01} \;\; y_{02} \ldots y_{0n} \;\; y_{01} \ldots y_{nn}]^T. \tag{III.3}$$

The columns in the design matrix $\boldsymbol{X}$ is than created as follows: where my intention is to illustrate that each column consist of the Hadamard product between $\boldsymbol{x_f^u}$ and $\boldsymbol{y_f^v}$ such that all polynomials in this sum is covered

$$\sum_{v=0}^{k}\sum_{u=0}^{k} \boldsymbol{x_f^{\circ u}} \circ \boldsymbol{y_f^{\circ v}}. \tag{III.4}$$

B. Regression analysis

With a design matrix in hand the SVD needed for the regression methods was calculated by the scipy module. With the SVD for a design matrix OLS is calculated by equation II.30 and ridge regression is calculated by equation II.31. The lasso regression model is created using the sklearn library, and more specific th `sklearn.linear_model.Lasso` module.

In the earlier paragraph I did not specify form which data set the design matrix originated from. This is because the analysis for both sets of data is almost identical. The few minor differences in the method will be specified when they appear.

At first the entire design matrix will be used in linear regression and not split into training and test data. The resulting models was than used to calculate the MSE and R^2 error by equation II.12 and II.13 respectively. For the Franke function data both the actual Franke function and the data with added noise was used when evaluating the error.

The next step was to split the data sets into training and test data with using the

`test_train_split` function from sklear, the ratio for the split used was 70% training data. The training data was than used to fit the linear regression models and the MSE and R^2 errors were calculated by using the test data and the Franke function without noise.

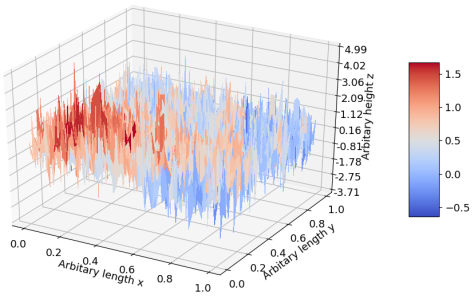
With the data split into training and test data, the resampling technique k-fold cross validation was used in fitting the linear models. The β-coefficients, MSE and R^2 values were stored for each exclusion of a fold. The error quantities were calculated both with respect to the Franke function and the test data with added noise. Finally the mean and variance for each measured quantity was calculated. These calculations were repeated for multiple different amount of data-points.

Unlike previously where the model complexity was constant, that is no longer the case. The models created will now be used to showcase the bias-variance tradeoff. This is done by varying the maximum order of the polynomial used in the creation of the design matrix $\boldsymbol{X}$. Multiple design matrices are created for $k = 0, 1, 2, 3, 4...$ maximum order of polynomials. Each design matrix is splitted into test and training data with a constant seed such that the datapoints used for test and training are the same independent of k. Multiple regression method are used to create models dependent on the design matrix, both by using the regression method alone or through k-fold cross validation. The MSE, R^2 error, the bias and variance of the model are thereafter calculated and plotted against the complexity. Remember that for the k-fold cross validation it's the mean of all the measured quantities which are plotted e.g the MSE from k-fold is the mean of all mean square errors calculated with k-fold.

IV. RESULTS

1. *Franke function*

For most of the analysis, the data set will consist of 81×81 data points except at the end of this subsection when stated otherwise. A plot of the data set of the Franke function with added noise is shown in figure 1. This is the data that will be used for further analysis.



Figur 1: The Franke function with added Gaussin noise with $\mu = 0$ and $\sigma = 1$ for a grid of 81×81 randomly distributed data points taken from a uniform distribution between $[0, 1]$.

The model created by OLS, Ridge and Lasso with a complexity of 5^{th} degree gave the following β-coefficients in table I for the full data set. Note however my choice for λ for both Ridge and Lasso, this choice will be apparent later. The confidence intervals for OLS is caluclated by II.32 while ridge and lasso is calculated by equation II.33, and all σ are multiplied with 1.96 to get the confidence at 95%.

The linear models with the β-coefficients shown in table I gives the following error estimates for MSE and R^2 shown in table II calculated by equation II.12 and II.13. The error estimates shown are both between the model and the data set, but also between the actual Franke function and the model.

β	OLS	Ridge	Lasso
β_0	0.375 ± 0.401	0.623 ± 0.241	1.06 ± 0.387
β_1	8.04 ± 4.6	4.22 ± 1.7	0.376 ± 4.38
β_2	4.28 ± 4.52	3.23 ± 1.65	0.836 ± 4.3
β_3	-32.0 ± 22.5	-15.6 ± 4.74	-4.65 ± 21.2
β_4	-10.3 ± 17.1	-1.22 ± 4.17	1.47 ± 16.4
β_5	-13.9 ± 22.9	-12.4 ± 4.74	-5.66 ± 21.5
β_6	35.9 ± 51.3	9.72 ± 6.16	3.24 ± 48.3
β_7	40.2 ± 37.4	10.1 ± 6.84	3.84 ± 35.9
β_8	4.59 ± 36.3	-4.09 ± 6.86	-3.18 ± 34.9
β_9	4.93 ± 52.8	7.01 ± 6.09	3.28 ± 49.6
β_{10}	-6.6 ± 54.3	9.03 ± 6.99	0.918 ± 51.2
β_{11}	-50.2 ± 40.7	-7.63 ± 9.15	0.0 ± 39.3
β_{12}	8.07 ± 37.2	9.88 ± 9.51	0.0 ± 35.9
β_{13}	-19.1 ± 39.4	-5.49 ± 9.29	0.0 ± 38.1
β_{14}	18.2 ± 55.9	10.4 ± 6.83	1.97 ± 52.6
β_{15}	-5.77 ± 21.5	-8.06 ± 4.27	-0.98 ± 20.4
β_{16}	17.3 ± 18.3	-0.98 ± 6.79	-2.43 ± 17.9
β_{17}	4.8 ± 17.6	-0.287 ± 8.56	0.0 ± 17.2
β_{18}	-10.5 ± 17.6	-6.51 ± 8.66	0.662 ± 17.2
β_{19}	14.8 ± 17.8	6.05 ± 6.81	0.0 ± 17.4
β_{20}	-13.6 ± 22.1	-8.49 ± 4.26	-0.915 ± 20.9

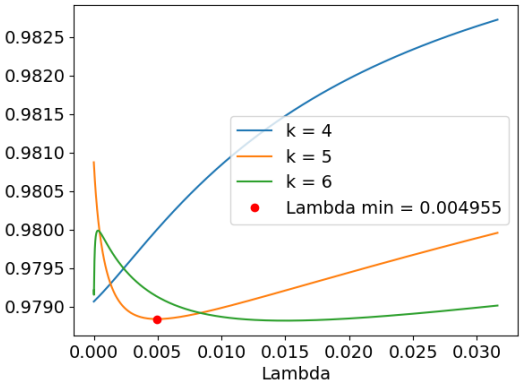
Tabell I: The β-values for the OLS, Ridge and Lasso regression methods with their respective confidence interval. For Ridge $\lambda = 4.95 \cdot 10^{-3}$ while for Lasso $\lambda = 3.66 \cdot 10^{-5}$.

Error estimates	OLS	Ridge	Lasso
MSE Franke	0.0053	0.0062	0.00895
MSE Data	0.998	0.999	1.0
R^2 Franke	0.944	0.934	0.905
R^2 Data	0.102	0.101	0.0956

Tabell II: The error estimates MSE and R^2 calculated based upon the model created by the coefficients in table I. The error estimates are calculated both with respect to the actual Franke function and the data set used to create the model.

So far all the results presented are from creating models on the entire data set. From now on the data set is split at at ratio 70% training and 30% test data. The β-coefficients from the training data is given in table XII in the appendix. The error estimates are given in table III and IV. In table III the error estimates is calculated based on the data points used for training while the error estimates in table IV is calculated based upon the test section of the data.

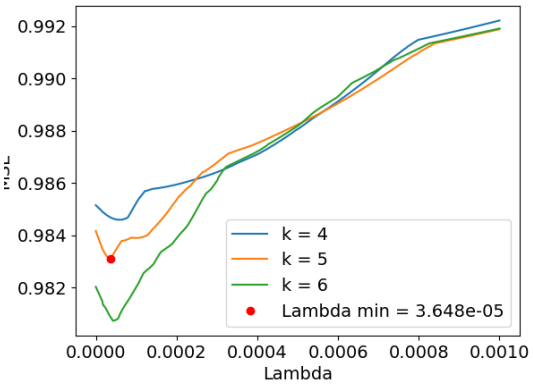
So far I have only blindly used two values for λ for both ridge and lasso regression. In figure 2 and 3 the MSE between the test set and the predicted model is plotted for a varying λ in ridge and lasso regression respectively. The same applies to figure 24 and 25, but for a wider range of λ-values. Figure 26, 27, 28 and 29 are contour versions of the previous mentioned plots, but for more polynomial degrees.



Figur 2: The MSE between the test set and the predicted model using Ridge regression for different polynomial orders and λ values. The λ which gives the minimum MSE for a 5^{th} order polynomial is highlighted.

Error estimates	OLS	Ridge	Lasso
MSE Franke	0.00783	0.00807	0.011
MSE Data	1.01	1.01	1.02
R^2 Franke	0.917	0.915	0.884
R^2 Data	0.0994	0.0979	0.092

Tabell III: The error estimates MSE and R^2 calculated based upon the model created by the coefficients in table XII. The error estimates are for the training section of the data and are calculated both with respect to the actual Franke function and the data set used to create the model.



Figur 3: The MSE between the test set and the predicted model using Lasso regression for different polynomial orders and λ values. The λ which gives the minimum MSE for a 5^{th} order polynomial is highlighted. The model is calculated using 2000 iterations pr λ-value.

Error estimates	OLS	Ridge	Lasso
MSE Franke	0.00783	0.0079	0.0106
MSE Data	0.981	0.979	0.984
R^2 Franke	0.917	0.916	0.888
R^2 Data	0.101	0.103	0.098

Tabell IV: The error estimates MSE and R^2 calculated based upon the model created by the coefficients in table XII. The error estimates are for the test section of the data and are calculated both with respect to the actual Franke function and the test section of the data set used to create the model.

Using the resampling method k-fold cross validation with 10-folds the MSE calculated between the model and the excluded fold is shown as histograms in figure 4, 5 and 6 for Ols, ridge and lasso respectively.

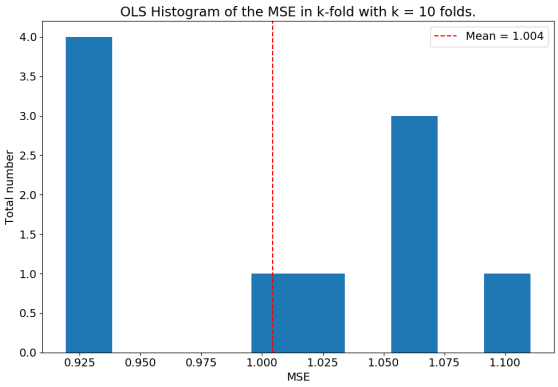
For 10-folds with OLS the MSE and R^2 scores from the test data in k-fold are

$$\text{MSE} = 1.005$$
$$R^2 = 0.0937.$$

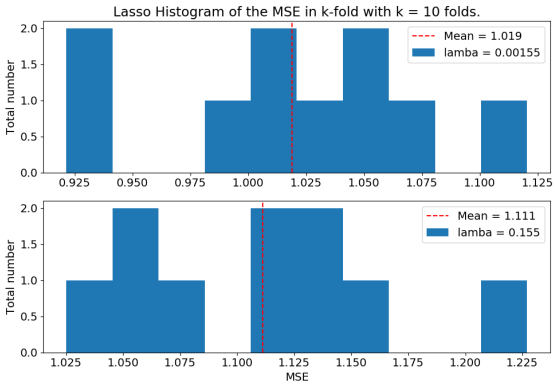
The standard deviation of the inevitable error from equation II.1 is calculated for each exclusion of a fold in k-fold cross validation. The mean of all these standard deviations is

$$\sigma = 1.0007 \pm 0.005 \qquad \text{(IV.1)}$$

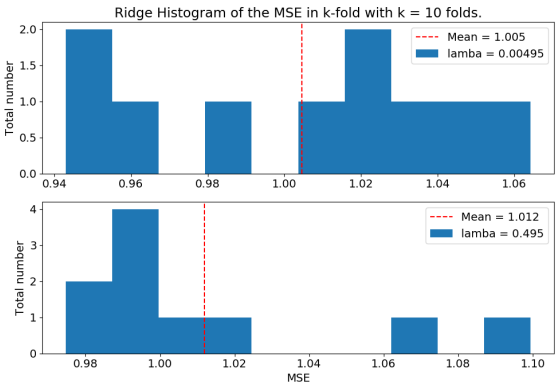
with 95% confidence interval. The mean β-coefficients of the models from k-fold is given in table XIII and the error is the standard deviation of all the calculated β_i-coefficients at a 95% confidence interval.



Figur 4: The MSE between the models created and their respective excluded folds in k-fold cross validation for 10 folds using OLS.



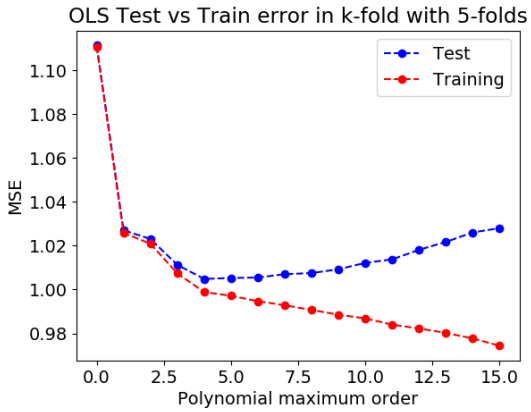
Figur 6: The MSE between the models created and their respective excluded folds in k-fold cross validation for 10 folds using Lasso with the λ-values given in the plot.



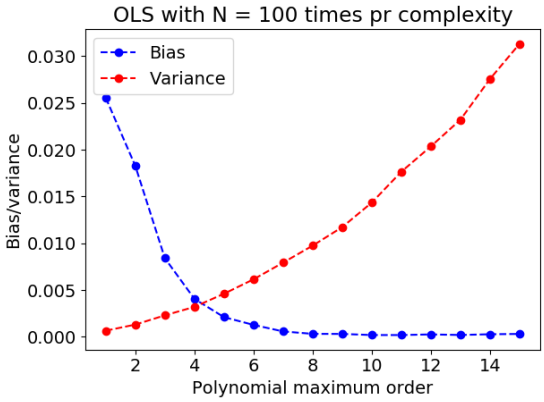
Figur 5: The MSE between the models created and their respective excluded folds in k-fold cross validation for 10 folds using Ridge with the λ-values given in the plot.

In figure 7 the MSE between a model and the training data is plotted togehter with the MSE between the model and test data not used in the creation of the model. This plotted against the complexity of the mode. In this case k-fold cross validation is used in the creation of test and training data with 5-folds. K-fold cross validation is run 50 times pr polynomial complexity but with random folds, the MSE is than the mean of the 50 runs. The same plot but for R^2 is shown in figure 8. The same plots but for ridge with $\lambda = 10^{-3}$ and $\lambda = 10^{-5}$ are shown in figure 30, 31, 32 and 33.

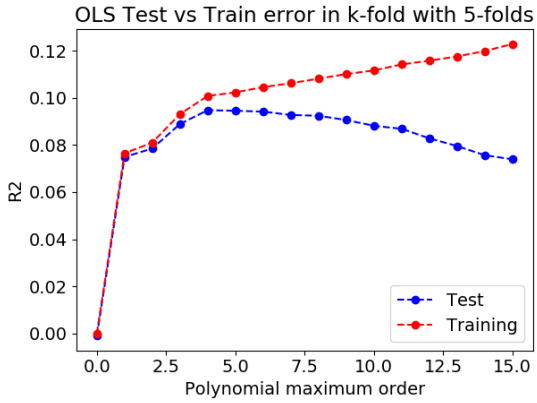
Further in figure 9 the bias and variance is plotted against the model complexity. The bias and variance is calculated between 100 models created by re randomizing the noise 100 times and fitting the OLS model on a train set. The bias and variance is calculated on the test section of the model.



Figur 7: The MSE between a OLS model for the Franke data set and both the training and test data separately in k-fold cross validation using the entire data sat as a basis for the exclusion of folds. In this case there are used 5 folds pr polynomial order. This is done $N = 50$ times with random folds and the plot shows the average.

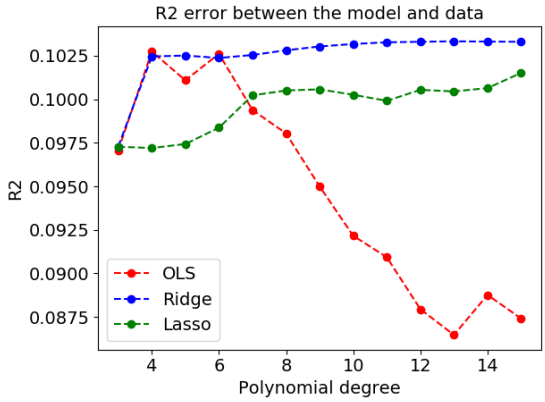


Figur 9: The R^2 score between a OLS model for the Franke data set and both the training and test data separately in k-fold cross validation using the entire data sat as a basis for the exclusion of folds. In this case there are used 5 folds pr polynomial order. This is done $N = 50$ times with random folds and the plot shows the average.

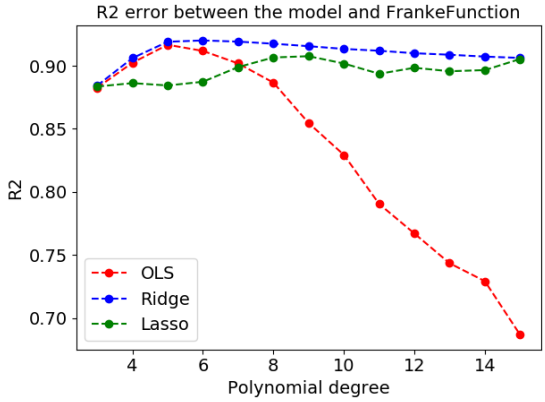


Figur 8: The R^2 score between a OLS model for the Franke data set and both the training and test data separately in k-fold cross validation using the entire data sat as a basis for the exclusion of folds. In this case there are used 5 folds pr polynomial order. This is done $N = 50$ times with random folds and the plot shows the average.

The final part of this subsection will resolve around finding the optimal solution for solving the problem. Therefore in figure 10 and 11 the R^2 score for OLS, ridge and lasso is plotted against the corresponding model complexity. The model is created using a training set whereas the error is estimated by a test set, further in figure 10 the test set used is for the real Franke function and figure 11 corresponds to the data set. For lasso and ridge regression the λ-parameter is my calculated optimal. Corresponding figures for the MSE is shown in figure 35 for the data set and 34 when comparing to the real Franke function. The minimum MSE and highest R^2 scores is found in table V with the corresponing polynomial complexity and λ-parameters.



Figur 10: The R^2 between a test set from the data set and the regression method OLS, ridge and lasso model using "optimal" λ-parameters.



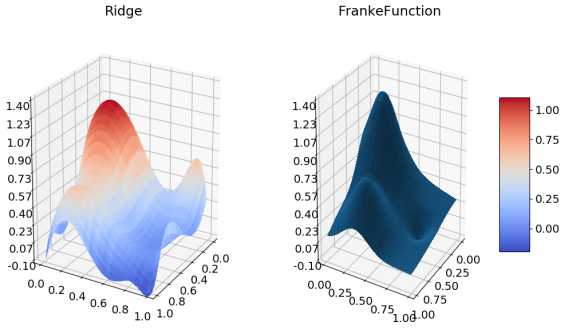
Figur 11: The R^2 between a test set and the regression method OLS, ridge and lasso model using "optimal" λ-parameters. The test set here is for the real Franke function, but it correspond to the part of the data set omitted in the training data.

Using the models for OLS, ridge and lasso with the best test data scores shown in table V i.e: 4^{th} order for OLS, 13^{th} order for ridge and 15^{th} order for lasso. The MSE and R^2 error estimates for the entire data set for those models is shown in table VI. The ridge model is plotted in figure

Error estimates	OLS	Ridge	Lasso	Degree
MSE Franke	0.00783	0.0075	0.00868	5, 6, 9
MSE Data	0.979	0.978	0.98	4, 13, 15
R^2 Franke	0.917	0.92	0.908	5, 6, 9
R^2 Data	0.103	0.103	0.102	4, 13, 15

Tabell V: The lowest MSE and highest R^2 estimate with corresponding polynomail degress and lambda value for figure 11, 10, 35 and 34. The λ-paramters used for ridge is $\lambda = 0.002458$ and $\lambda = 0.03091$ and for lasso $\lambda = 7.2605 \cdot 10^{-6}$ and $\lambda = 3.7727 \cdot 10^{-5}$ for Franke and data respectively.

37 whereas the remaing two models from table VI is plotted in figure

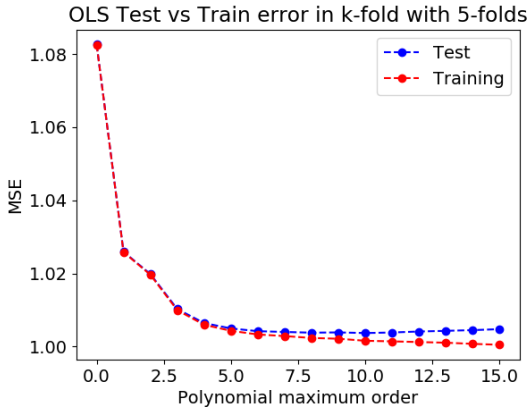


Figur 12: A 3D plot of the 13^{th} order Ridge model with $\lambda = 0.03091$ and the Franke function.

The final figure of the Franke functon data is given figure 13 and this figure is a plot of the test and training MSE error from k-fold cross validation similar to figure 7. The only difference is that the data set used is of size 400×200.

Error estimates	OLS	Ridge	Lasso
MSE Franke	0.009233	0.008609	0.009017
MSE Data	1.001	0.9985	1.001
R^2 Franke	0.9023	0.9089	0.9046
R^2 Data	0.09922	0.1012	0.09854

Tabell VI: The MSE and R^2 scores for the entire data set with the models with the best scores from the test data in table V for OLS, ridge and lasso.



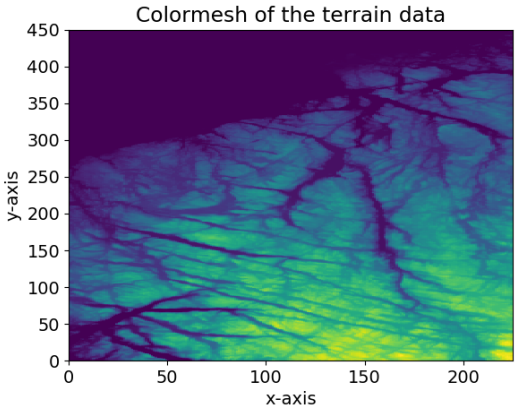
Figur 13: The MSE between a OLS model for the Franke data set and both the training and test data separately in k-fold cross validation using the entire data sat as a basis for the exclusion of folds. In this case there are used 5 folds pr polynomial order. This is done $N = 50$ times with random folds and the plot shows the average.

2. *Terrain data*

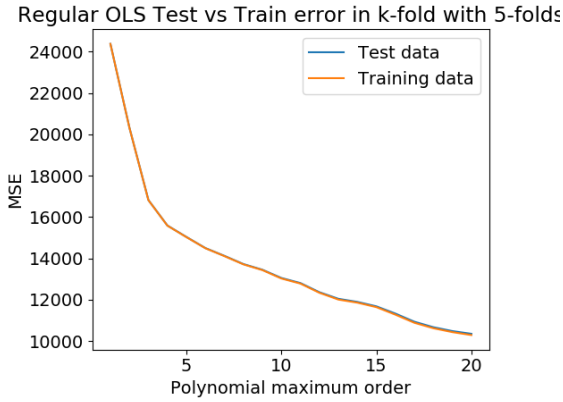
Unlike the previous subsection I will not be including any tables about the β-parameters and their confidence intervals. All of this can be found in `terrain.py` in my github.

A colormesh plot of the terrain data is shown in figure 14. As in the figure and for the major parts of this subsection of the data set is down sampled from 3600×1800 to 400×200 if not stated otherwise. The downsampling is done by taking the average of 9×9 adjutant data points.

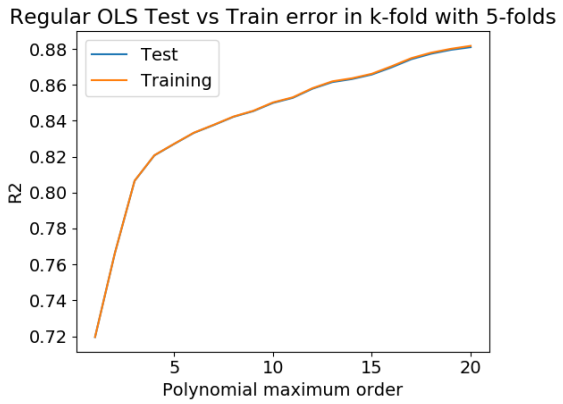
To determine the complexity of the model I will begin by finding the MSE and R^2 scores by k-fold cross validation with 5-folds. In figure 15 and 16 the mean MSE and R^2 from k-fold are plotted against the used polynomial complexity. The error estimates are for the mean errors in the test and train data in k-fold. In both cases OLS is used.



Figur 14: Colormesh plot of the terrain data used from file "SRTM_data_Norway_2" after being resampled to a size of 400×200.



Figur 15: The MSE between a OLS model for the terrain data set and both the training and test data separately in k-fold cross validation using the entire data sat as a basis for the exclusion of folds. In this case there are used 5 folds pr polynomial order, the MSE plotted is the mean of the calculated MSE i k-fold.



Figur 16: The R^2 between a OLS model for the terrain data set and both the training and test data separately in k-fold cross validation using the entire data sat as a basis for the exclusion of folds. In this case there are used 5 folds pr polynomial order, the R^2 plotted is the mean of the calculated R^2 in k-fold.

Based on figure 15 and 16 there exist no ideal polynomial complexity where further increase in complexity would have negligible effect. Thus I will be choosing a complexity of maximum 15^{th} degree[2]. The mean MSE and R^2 error for a 15^{th} degree polynomial from k-fold with 5 folds can be found in table VII.

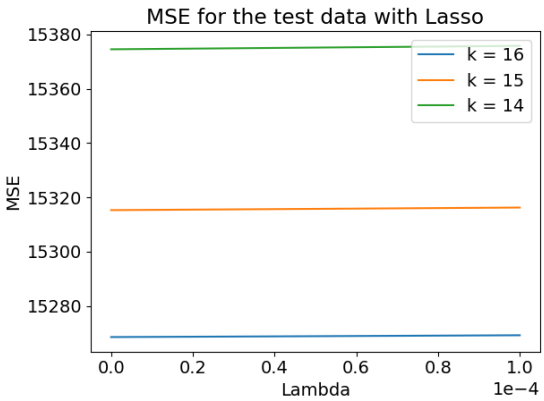
As done with the Franke function the data is splitted into training and test data. This split used to find the optimal λ parameter for ridge and lasso. The result of this is seen in figure 17 and 18 where the MSE in the test data is plotted against the corresponding λ parameter. Further increase in λ-values than demonstrated would prove pointless since it would not improve the accuracy of the model for a 15^{th} degree polynomial complexity. However for lower polynomial complexity $\lambda > 0$ does in fact have a positive effect on the error estimates of the test data.

In figure 19 the mean λ values which minimized the test error in k-fold with 4 folds ran $N = 50$ times is plotted against the model complexity. The error is the standard error $\sigma/\sqrt{N}$ at 95% confidence of the mean λ from k-fold. Notice how the standard error doesn't take into account the number of folds used in k-fold. This is because the λ value is not the mean of the λ-values which minimized to error in each fold, but rather the λ where the sum of the test errors were minimized.

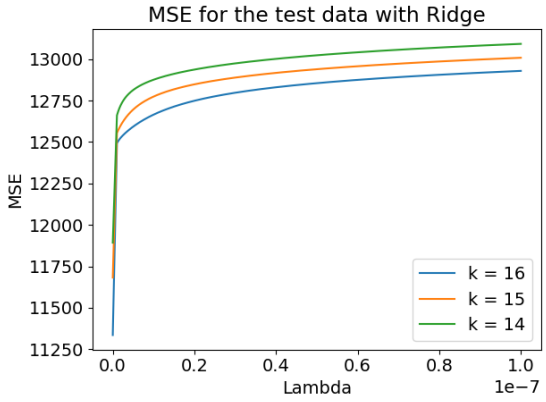
[2] I will later learn to regret this decision, so many hours running programs

Error estimates	OLS
MSE Test	11677.4
MSE Training	11636.5
R^2 Test	0.8657
R^2 Training	0.8662

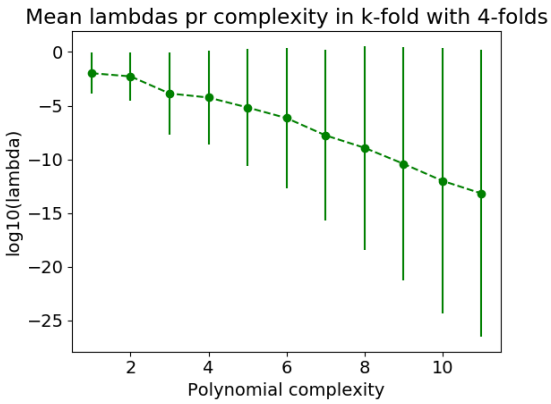
Tabell VII: The error estimates MSE and R^2 from figure 15 and 16 for 15^{th} degree polynomial.



Figur 18: The R^2 between a OLS model for the terrain data set and both the training and test data separately in k-fold cross validation using the entire data sat as a basis for the exclusion of folds. In this case there are used 5 folds pr polynomial order, the R^2 plotted is the mean of the calculated R^2 in k-fold.



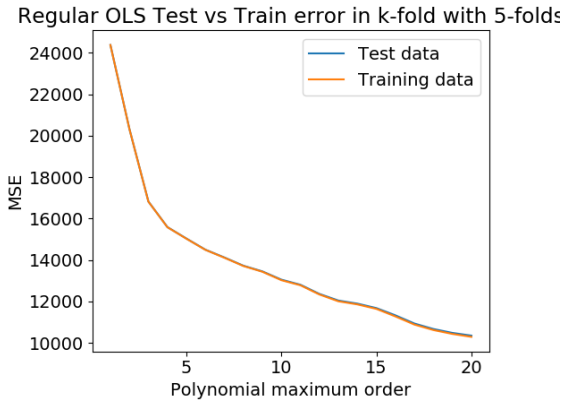
Figur 17: The R^2 between a OLS model for the terrain data set and both the training and test data separately in k-fold cross validation using the entire data sat as a basis for the exclusion of folds. In this case there are used 5 folds pr polynomial order, the R^2 plotted is the mean of the calculated R^2 in k-fold.



Figur 19: The λ-values which minimizes the test error in k-fold cross validation with 4-folds using ridge regression. The plotted λ-values are the mean of $N = 5$ k-fold runs with randomized folds pr complexity.

Using the λ-values from figure 19 and recreating figure 15 with ridge, the final result is plotted in figure 20. Further a comparison of the test

MSE scores from ridge and OLS with the same folds is shown in table VIII. Note however that after 12^{th} polynomials $\lambda = 0$ and differences are because of round off errors in the calculations.



Figur 20: The MSE between a OLS model for the terrain data set and both the training and test data separately in k-fold cross validation using the entire data sat as a basis for the exclusion of folds. In this case there are used 5 folds pr polynomial order, the MSE plotted is the mean of the calculated MSE i k-fold.

Complexity	OLS	Ridge
1	24381.92	24381.92
2	20328.777	20328.777
3	16820.989	16820.989
4	15588.154	15588.153
5	15034.744	15034.74
6	14496.119	14496.114
7	14119.902	14119.895
8	13722.009	13722.006
9	13447.197	13447.194
10	13048.584	13048.576
11	12802.96	12802.897
12	12359.427	12359.424
13	12039.967	12039.967
14	11890.025	11890.025
15	11674.806	11674.806
16	11325.064	11325.064
17	10929.79	10929.789
18	10664.982	10664.984
19	10476.987	10476.981
20	10345.836	10345.831

Tabell VIII: The error estimates MSE and R^2 from figure 15 and 16 for 15^{th} degree polynomial.

Using a 15^{th} degree complexity to create the model the MSE and R^2 error estimates for the entire data set is shown in table IX and the inevitable error is $\sigma = 107.98423$. Using a simple train test split the error estimates is shown in table X and the inevitable error for the train test case is $\sigma = 107.98418$.

Using k-fold cross validation on the entire data set with $k = 5$ folds the MSE and R^2 error estimates is given in table XI, and figure 21 is a histogram of MSE for k-fold cross validation with 10 folds for OLS. The same histogram but for ridge and lasso regression with $\lambda = 10^{-8}$ is shown in figure 38 and 39.

Error estimates	OLS	Ridge	Lasso
MSE Data	11641.0	12700.0	15349.0
R^2 Data	0.86612	0.85393	0.82346

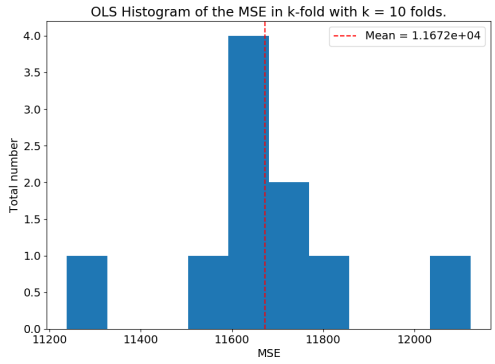
Tabell IX: The error estimates MSE and R^2 from figure 15 and 16 for 15^{th} degree polynomial.

Error estimates	OLS	Ridge	Lasso
MSE Data	11632.0	12722.0	15305.0
R^2 Data	0.86617	0.85363	0.82392
MSE Data	11676.0	12813.0	15467.0
R^2 Data	0.86581	0.85274	0.82224

Tabell X: The error estimates MSE and R^2 from figure 15 and 16 for 15^{th} degree polynomial.

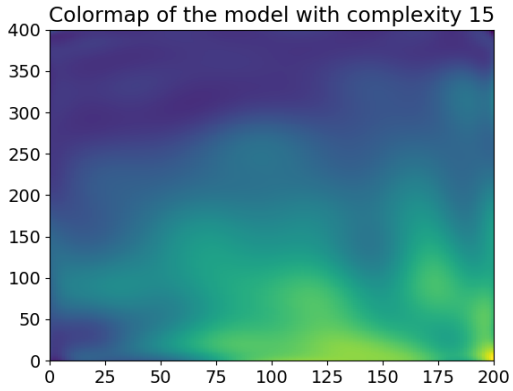
Error estimates	OLS	Ridge	
MSE Data	11700.0 ± 362.0	12700.0 ± 463.0	154(
R^2 Data	0.866 ± 0.00471	0.853 ± 0.00533	0.82
MSE Data	11600.0 ± 90.1	12700.0 ± 113.0	153
R^2 Data	0.866 ± 0.00117	0.854 ± 0.00131	0.82

Tabell XI: The error estimates MSE and R^2 from figure 15 and 16 for 15^{th} degree polynomial.

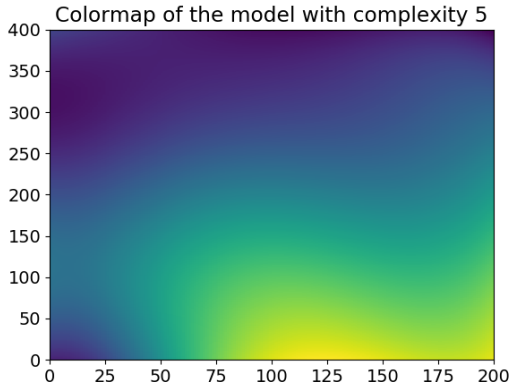


Figur 21: The R^2 between a OLS model for the terrain data set and both the training and test data separately in k-fold cross validation using the entire data sat as a basis for the exclusion of folds. In this case there are used 5 folds pr polynomial order, the R^2 plotted is the mean of the calculated R^2 in k-fold.

Using 15^{th} degree complexity for the entire data set a colormap of the model is shown in figure 22. For comparison a 5^{th} degree polynomial fit has the following error estimates $MSE = 15026$ and $R^2 = 0.8272$ and the colorplot is shown in figure 23.



Figur 22: The R^2 between a OLS model for the terrain data set and both the training and test data separately in k-fold cross validation using the entire data sat as a basis for the exclusion of folds. In this case there are used 5 folds pr polynomial order, the R^2 plotted is the mean of the calculated R^2 in k-fold.



Figur 23: The R^2 between a OLS model for the terrain data set and both the training and test data separately in k-fold cross validation using the entire data sat as a basis for the exclusion of folds. In this case there are used 5 folds pr polynomial order, the R^2 plotted is the mean of the calculated R^2 in k-fold.

V. DISCUSSION

Something something

VI. CONCLUSION

Something something

β	OLS	Ridge	Lasso
β_0	0.457 ± 0.469	0.699 ± 0.285	1.03 ± 0.449
β_1	6.04 ± 5.45	3.16 ± 1.94	0.34 ± 5.09
β_2	5.09 ± 5.38	2.92 ± 1.92	0.848 ± 5.02
β_3	-23.1 ± 26.8	-12.5 ± 5.22	-4.14 ± 24.7
β_4	-5.23 ± 20.2	1.59 ± 4.5	2.35 ± 19.1
β_5	-23.2 ± 27.3	-13.2 ± 5.24	-6.28 ± 25.1
β_6	15.3 ± 61.2	6.07 ± 5.82	1.96 ± 56.2
β_7	38.0 ± 44.5	8.67 ± 6.47	3.97 ± 42.0
β_8	-2.12 ± 43.1	-6.93 ± 6.5	-4.77 ± 40.8
β_9	29.4 ± 63.1	9.71 ± 5.76	4.52 ± 57.7
β_{10}	16.1 ± 64.8	9.88 ± 6.5	1.46 ± 59.8
β_{11}	-59.4 ± 48.7	-9.13 ± 8.51	0.0 ± 46.4
β_{12}	21.7 ± 44.4	11.1 ± 9.01	0.0 ± 42.4
β_{13}	-24.2 ± 46.8	-6.51 ± 8.69	-0.124 ± 44.8
β_{14}	-5.41 ± 66.7	8.73 ± 6.38	1.92 ± 61.2
β_{15}	-14.8 ± 25.7	-7.36 ± 4.43	-0.612 ± 23.9
β_{16}	23.6 ± 22.0	0.156 ± 6.91	-3.1 ± 21.2
β_{17}	0.97 ± 21.1	-1.3 ± 9.03	0.0 ± 20.5
β_{18}	-13.5 ± 21.1	-4.31 ± 9.1	1.85 ± 20.5
β_{19}	19.9 ± 21.2	6.43 ± 6.89	0.0 ± 20.6
β_{20}	-5.79 ± 26.3	-8.38 ± 4.45	-1.34 ± 24.3

Tabell XII: The β-values for the OLS, Ridge and Lasso regression methods with their respective confidence interval. For Ridge $\lambda = 4.95 \cdot 10^{-3}$ while for Lasso $\lambda = 3.66 \cdot 10^{-5}$.

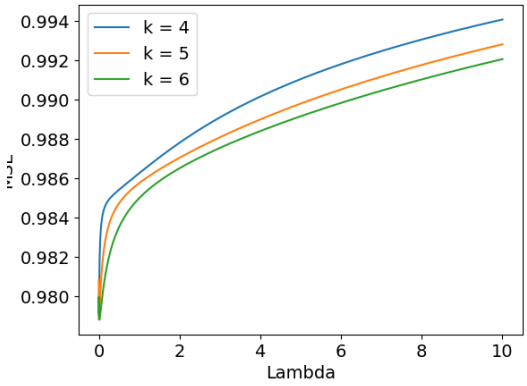
Tillegg A: Tables

$$
\begin{aligned}
\boldsymbol{\beta}_{OLS} &= (\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{X})^{-1}\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{y} \\
&= ((\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T)^T\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T)^{-1}(\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T)^T\boldsymbol{y} \\
&= (\boldsymbol{V}\boldsymbol{\Sigma}^{\boldsymbol{T}}\boldsymbol{U}^{\boldsymbol{T}}\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T)^{-1}\boldsymbol{V}\boldsymbol{\Sigma}^{\boldsymbol{T}}\boldsymbol{U}^T\boldsymbol{y} \\
&= (\boldsymbol{V}\boldsymbol{\Sigma}^{\boldsymbol{2}}\boldsymbol{V}^T)^{-1}\boldsymbol{V}\boldsymbol{\Sigma}^{\boldsymbol{T}}\boldsymbol{U}^T\boldsymbol{y} \\
&= \boldsymbol{V}\boldsymbol{\Sigma}^{-2}\boldsymbol{\Sigma}^{\boldsymbol{T}}\boldsymbol{U}^{\boldsymbol{T}}\boldsymbol{y} \\
&= \boldsymbol{V}(\boldsymbol{\Sigma})^{-1}\boldsymbol{U}^{\boldsymbol{T}}\boldsymbol{y}.
\end{aligned}
$$

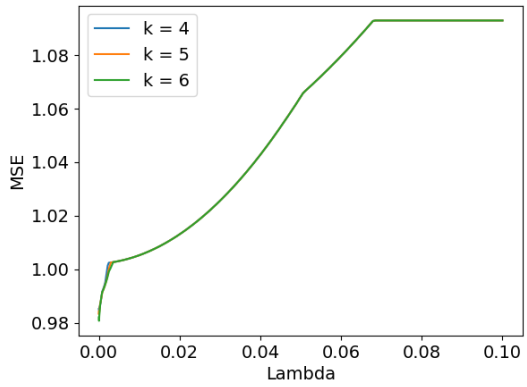
β	OLS
β_0	0.373 ± 0.115
β_1	8.05 ± 1.19
β_2	4.29 ± 1.43
β_3	-32.1 ± 6.52
β_4	-10.3 ± 5.55
β_5	-13.9 ± 7.13
β_6	36.0 ± 15.6
β_7	40.2 ± 10.1
β_8	4.65 ± 17.4
β_9	4.94 ± 17.1
β_{10}	-6.61 ± 16.5
β_{11}	-50.2 ± 13.6
β_{12}	8.09 ± 12.5
β_{13}	-19.2 ± 19.1
β_{14}	18.2 ± 18.7
β_{15}	-5.78 ± 6.2
β_{16}	17.4 ± 6.43
β_{17}	4.77 ± 6.87
β_{18}	-10.5 ± 8.23
β_{19}	14.8 ± 7.72
β_{20}	-13.6 ± 7.36

Tabell XIII: The β-values for the OLS regression method using 10 folds in k-fold cross validation on the training data in a train/test data set. The confidence interval is 95%, and is calculated by taking the standard deviation for all the calculated β_i-coefficients i.e the std for β_0, β_1, β_2 etc.

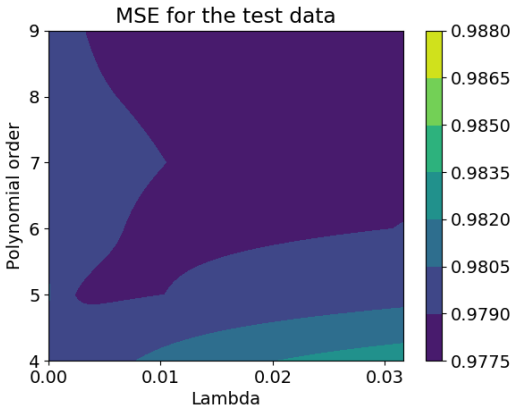
Tillegg B: Figures



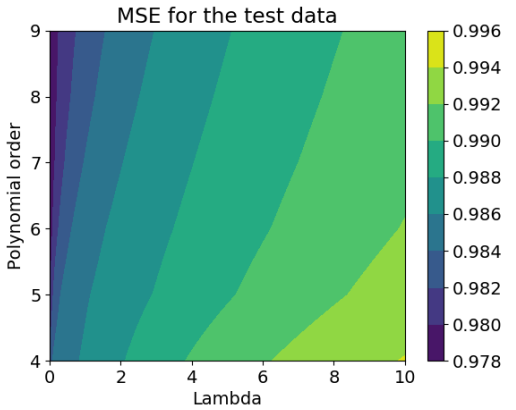
Figur 24: The MSE between the test set and the predicted model using Ridge regression for different polynomial orders and λ values. The λ which gives the minimum MSE for a 5^{th} order polynomial is highlighted.



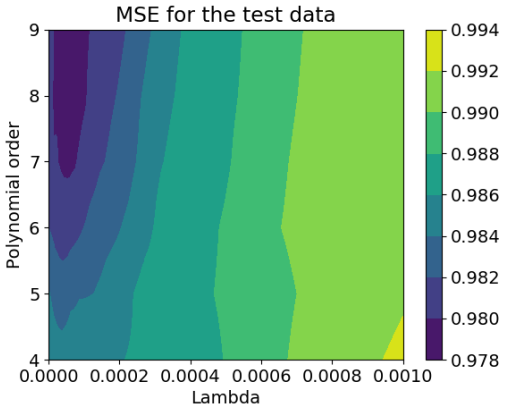
Figur 25: The MSE between the test set and the predicted model using Lasso regression for different polynomial orders and λ values. The λ which gives the minimum MSE for a 5^{th} order polynomial is highlighted. The model is calculated using 2000 iterations pr λ-value.



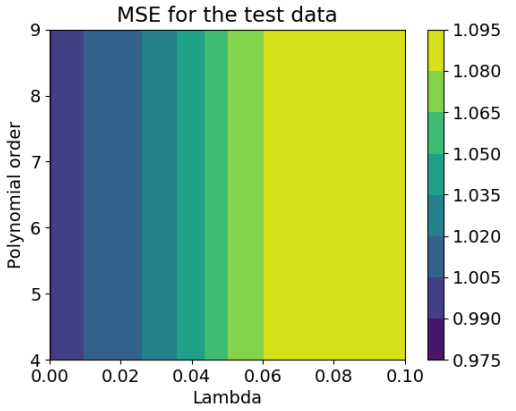
Figur 26: The MSE between the test set and the predicted model using Ridge regression for different polynomial orders and λ values. The λ which gives the minimum MSE for a 5^{th} order polynomial is highlighted.



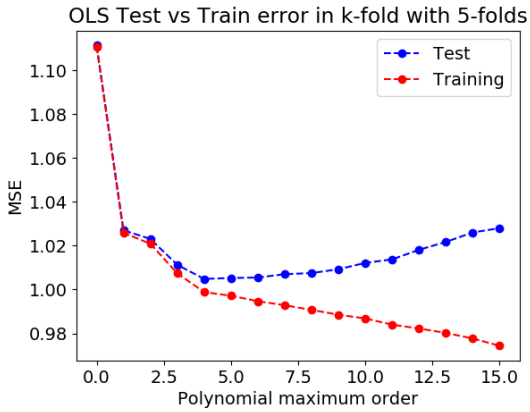
Figur 28: The MSE between the test set and the predicted model using Ridge regression for different polynomial orders and λ values. The λ which gives the minimum MSE for a 5^{th} order polynomial is highlighted.



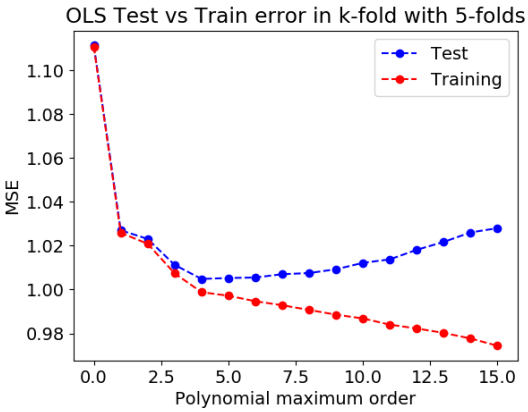
Figur 27: The MSE between the test set and the predicted model using Lasso regression for different polynomial orders and λ values. The λ which gives the minimum MSE for a 5^{th} order polynomial is highlighted. The model is calculated using 2000 iterations pr λ-value.



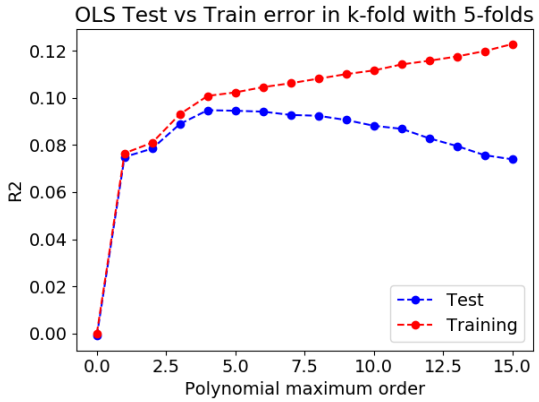
Figur 29: The MSE between the test set and the predicted model using Lasso regression for different polynomial orders and λ values. The λ which gives the minimum MSE for a 5^{th} order polynomial is highlighted. The model is calculated using 2000 iterations pr λ-value.



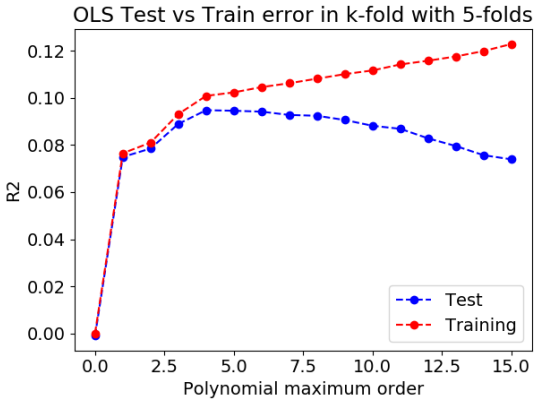
Figur 30: The MSE between a OLS model for the Franke data set and both the training and test data separately in k-fold cross validation using the entire data sat as a basis for the exclusion of folds. In this case there are used 5 folds pr polynomial order. This is done $N = 50$ times with random folds and the plot shows the average.



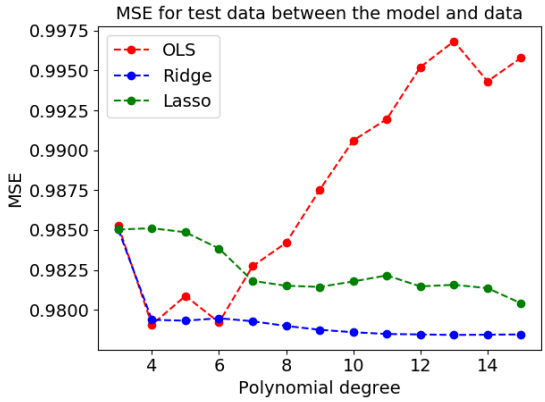
Figur 32: The MSE between a OLS model for the Franke data set and both the training and test data separately in k-fold cross validation using the entire data sat as a basis for the exclusion of folds. In this case there are used 5 folds pr polynomial order. This is done $N = 50$ times with random folds and the plot shows the average.



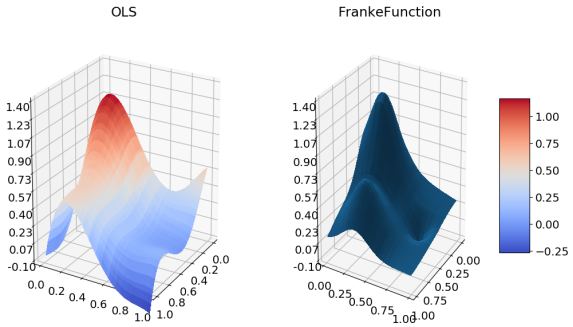
Figur 31: The R^2 score between a OLS model for the Franke data set and both the training and test data separately in k-fold cross validation using the entire data sat as a basis for the exclusion of folds. In this case there are used 5 folds pr polynomial order. This is done $N = 50$ times with random folds and the plot shows the average.



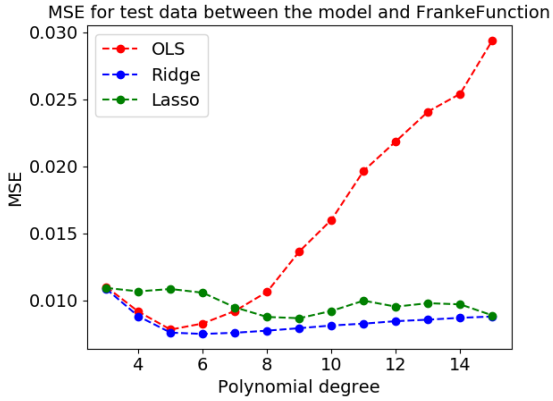
Figur 33: The R^2 score between a OLS model for the Franke data set and both the training and test data separately in k-fold cross validation using the entire data sat as a basis for the exclusion of folds. In this case there are used 5 folds pr polynomial order. This is done $N = 50$ times with random folds and the plot shows the average.



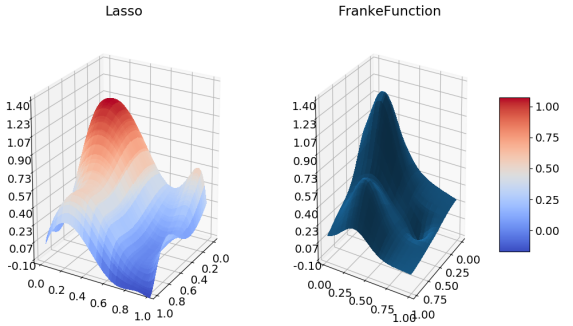
Figur 34: The R^2 between a test set from the data set and the regression method OLS, ridge and lasso model using "optimal" λ-parameters.



Figur 36: A 3D plot of the 4^{th} order OLS model from table VI beside the Franke function.

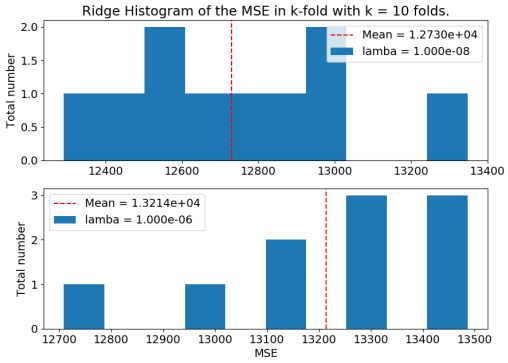


Figur 35: The R^2 between a test set and the regression method OLS, ridge and lasso model using "optimal" λ-parameters. The test set here is for the real Franke function, but it correspond to the part of the data set omitted in the training data.

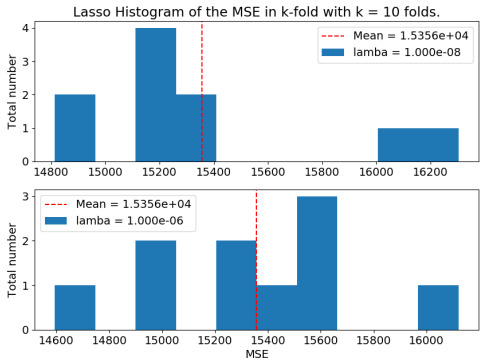


Figur 37: A 3D plot of the 15^{th} order Lasso model from table VI with $\lambda = 3.7727 \cdot 10^{-5}$ beside the Franke function.

a. Terrain



Figur 38: The R^2 between a OLS model for the terrain data set and both the training and test data separately in k-fold cross validation using the entire data sat as a basis for the exclusion of folds. In this case there are used 5 folds pr polynomial order, the R^2 plotted is the mean of the calculated R^2 in k-fold.



Figur 39: The R^2 between a OLS model for the terrain data set and both the training and test data separately in k-fold cross validation using the entire data sat as a basis for the exclusion of folds. In this case there are used 5 folds pr polynomial order, the R^2 plotted is the mean of the calculated R^2 in k-fold.

Tillegg C: Proofs