Draft version October 9, 2018
Preprint typeset using LaTeX style emulateapj v. 12/16/11

SURFACE FITTING OF TWO DIMENSIONAL TERRAIN DATA USING LINEAR REGRESSION

Sigbjrn Foss, Gunnar Thoresen Liahjell, Tobias Berdiin Olesen, Thomas Sjstad

Draft version October 9, 2018

ABSTRACT

We perform the OLS, Lasso and Ridge regression methods on a two-variable function with added noise and terrain data from mt.Everest and analyze the results. Reading of the mean square error and R^2-score for the different regression methods methods show that the OLS regression method provides the best fit for this data set.

Github repository with all source files can be found at this link.

1. INTRODUCTION

With the ubiquity machine learning algorithms in the current era, the importance of a thorough understanding of such methods cannot be overstated. Typical machine learning problems often include regression methods, either for prediction or classification, and this paper is to serve as an introduction to three central linear regression methods, the ordinary least square method, ridge and lasso. These algorithms will be developed using the known two-variable Franke function as test material, and later applied to a data set containing terrain features around the peak of Mount Everest. The aim is to give an overview of the derivation of these methods and examine their strengths and weaknesses from a statistical standpoint. To demonstrate the distinctions between the methods, we will apply them to the same data sets. Calculating the R^2-score, mean square error and regression parameter variance in the fitted surfaces, we can glean some insight in to how the different algorithms perform. We will also demonstrate the difference in results resulting from implementing the k-fold cross validation resampling technique.

2. METHODS

In this section, we provide some insight into the methods used in solving and analyzing the fitting problems, and give a brief explanation of their mathematical background in varying degrees of detail.

2.1. *Linear regression methods*

The digital terrain data is to be fitted by a two variable polynomial. The data set is given in the form of an $N\times N$ matrix, containing the z-values over a grid $\{(x,y)\}$, and is fitted by a polynomial

$$\tilde{z}(x,y) = \beta_0\beta_1 x + \beta_2 y + \beta_3 xy + ... + \beta_P x^p y^p$$

of order p. The order of the fitted function is determined according to the shape of the surface, but when developing algorithms for the Franke function, $p = 5$ is used for convenience. For a given order p, the number of coefficients needed is P, so in the instance of $p = 5$, we get $P = 21$. For each point on the grid, the approximation can be expressed in terms of the (x,y)-coordinates,

$$\begin{aligned}
\tilde{z}_1 &= \beta_0 + \beta_1 x_1 + \beta_2 y_1 + \ldots + \beta_P x_1^p y_1^p + \varepsilon_1 \\
\tilde{z}_2 &= \beta_0 + \beta_1 x_2 + \beta_2 y_2 + \ldots + \beta_P x_2^p y_2^p + \varepsilon_2 \\
&\vdots \\
\tilde{z}_{N^2} &= \beta_0 + \beta_1 x_{N^2} + \beta_2 y_{N^2} + \ldots + \beta_P x_{N^2}^p y_{N^2}^p + \varepsilon_{N^2}.
\end{aligned}$$

This set of equations is compactly expressed by the matrix vector equation

$$\vec{\tilde{z}} = \mathbf{X}\vec{\beta} + \vec{\varepsilon} \tag{1}$$

With $\vec{\tilde{z}} \in \mathbb{R}^{N^2}$ and $\mathbf{X} \in \mathbb{R}^{N^2\times P}$. The matrix $\mathbf{X}$ is called the design matrix. This is the problem to be solved for $\vec{\beta}$ by regression. The general way to derive a method of fitting a function to data is to define a cost function which describes the distance between a point $z(x,y)$ in the data set and a point $\tilde{z}(x,y)$ in the fitted function and try to minimize this function for all points. The difference in the three regression methods to be discussed is essentially which cost function is chosen.

2.1.1. *Ordinary least square (OLS) regression*

The OLS method is derived by defining the cost function

$$Q(\vec{\beta}) = \sum_{i=0}^{N^2-1} (z_i - \tilde{z}_i)^2.$$

This can be written as a matrix product, and by expressing $\vec{\tilde{z}}$ as in (1), we get

$$Q(\vec{\beta}) = (\vec{z} - \mathbf{X}\vec{\beta})^T(\vec{z} - \mathbf{X}\vec{\beta})$$

To minimize the cost function we set

$$\frac{dQ(\vec{\beta})}{d\vec{\beta}} = 0,$$

and we state without proof that this derivative can be found to be

$$\frac{dQ(\vec{\beta})}{d\vec{\beta}} = \mathbf{X}^T(\vec{z} - \mathbf{X}\vec{\beta}) = 0.$$

If $\mathbf{X}^T\mathbf{X}$ is invertible, we find the solution

$$\vec{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\vec{z}, \tag{2}$$

which is inserted into (1) to find the fitted function $\vec{\tilde{z}}$. The OLS method has some drawbacks that are relevant to this particular data set. Particularly, since the cost function is based on the square error, eventual outliers have the potential to disrupt the results significantly. Additionally, there are some constraints on $\mathbf{X}^T\mathbf{X}$. If this matrix is singular or near-singular, which is more likely for high dimensional design matrices, the inversion of this matrix in the expression for the regression coefficients can behave erratically, and the precision of the fit will suffer and in some cases the coefficients cannot be estimated at all. These issues are addressed by an adjustment to the cost function, which is discussed in the following.

2.1.2. *Ridge regression*

The way to resolve the issues that may arise in the OLS method is to simply add a penalty term $\vec{\beta}^T\vec{\beta}\lambda$ to the cost function. Following the the steps as for the OLS method, the regression solutions are now

$$\vec{\beta}^{\text{ridge}} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\vec{z}, \tag{3}$$

The addition of this error term ensures that the system is solvable by adding a positive value to the diagonal of $\mathbf{X}^T\mathbf{X}$ and thus ensures that the matrix is non-singular. It also gives us greater control over the bias and variance of the fit. The discussion in Hastie et. al. [1], shows how the regression coefficients are penalized according to size. This leads to a dampening effect where, if we choose a larger value of λ, the outlying data points with be weighted less and the variance will be reduced. This of course comes with a cost. When increasing the value of λ, the fit will approach a linear function and the bias increases. As a consequence, implementation of this method requires some amount of finesse in tuning the λ parameter, in contrast to the brute force OLS method.

2.1.3. *Lasso regression*

Another similar adjustment can be made to the cost function by adding the penalty $\lambda\sum_{j=1}^{p}|\lambda_j|$ to the cost function. The cost function is then given by

$$Q(\vec{\beta}) = \sum_{i=1}^{n}\Big(y_i - \sum_{j=1}^{p}x_{ij}\beta_j\Big)^2 + \lambda\sum_{j=1}^{p}|\beta_j|, \tag{4}$$

which we once again want to minimize. In this case the regression coefficients will be nonlinear in z_i, so they cannot be expressed in closed form, and must be solved by the implementation of some root finding algorithm. Luckily for us, the Scikit-learn python package includes and efficient function to perform this regression. The behaviour of the results with varying λ is much less transparent than in the ridge method, and will not be touched upon in detail here, but the main strength of the lasso is that it can reduce the coefficients for non important output parameters to zero. This amounts to an automatic feature selection integrated in the algorithm itself. It also has a regularization effect similar to the ridge method.

2.2. *Brief Statistical overview*

A few statistical concepts have been mentioned in the above discussion, and for the sake of completeness, they will be discussed briefly. These methods are essential when analyzing the results of the surface fitting.

2.2.1. *Variance, standard deviation and confidence intervals*

The variance , defined as

$$\sigma^2 = Var(y) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})^2, \tag{5}$$

is a measure of how spread out a given dataset $\{y_i\}$ is from the mean. The standard deviation σ is defined as the square root of the variance. In our analysis of the regression, we will estimate the variance of each regression coefficient, and in turn establish a 95% confidence interval, which means that each coefficient has a 95 % chance of being in the interval $\beta_i \pm \sigma(\beta_i)$

2.3. *Error estimates*

In order to properly analyse and interpret the results of our approach, error analysis is very important. There are several possible ways to do an error estimate, but in this project we will mainly concern ourselves with the mean square error (MSE) and the R^2 score function.

2.3.1. *Mean squared error*

The mean squared error tells us how close the regression curve is the to set of datapoints. It is called the mean squared error because one is finding the mean of a set of errors. In essence one would typically want the MSE to be as small as possible.

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

2.3.2. *The R^2 score function*

The R^2 score function also measures how close the fitted regression curve is to the data, and is in that regard similar to the mean square error. More explicit the R^2 score function is defined as the percentage of the variaton in the dataset that is explained by the model [ref:http://blog.minitab.com/blog/adventures-in-statistics-2/regression-analysis-how-do-i-interpret-r-squared-and-assess-the-goodness-of-fit]:

$$\text{R}^2 = \frac{\text{Explained variation}}{\text{Total variation}} = 1 - \frac{\sum_{i=1}^{N}(y_i - \tilde{y})^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2} \tag{7}$$

This means that the R^2 score always will be between zero and one. If $R^2 = 0$ it means that the model doesn't explain the variations in the data (around it's mean) at all, and if $R^2 = 1$ it means that the model explains all the variations in the data (again around it's mean). Therefore one would in general like to have a R^2 score as close to one as possible.

2.3.3. *Bias*

Another important quantity is the bias, which measures how much the model $\tilde{y}$ differs from the actual dataset y. Because the bias measures the errors due to simplifying assumptions of our model $\tilde{y}$, a high bias is related to underfitting. Our dataset could for example make out some kind of curve, but because our model isn't complex enough or haven't been trained enough, it fails to follow the curve in an explanatory way. This is a serious problem because it can make us miss important characteristics of the data set.

The bias can be calculated from the expression

$$\text{bias}(\tilde{y}) = \frac{1}{n}\sum_{i=1}^{N}(\tilde{y}_i - y_i) \tag{8}$$

2.3.4. *The bias-variance decomposition*

The connection between MSE, variance and bias is one of importance. It can be shown that the so-called bias-variance tradeoff(or bias-variance decomposition) can be written:

$$\text{MSE}(\tilde{y}) = \text{bias}(\tilde{y})^2 + \text{var}(\tilde{y}) + \sigma^2 \tag{9}$$

where the bias and variance are as given by the aforementioned expressions. The σ^2 term is an irreducible error. It comes directly from the noise in our training data, and because real data always will have some noise, it is unavoidable.

As mentioned above, a high bias is typically related to underfitting. When trying to avoid underfitting it is important not to start overfitting instead. A high variance often corresponds to overfitting of the model, meaning that the model is too complex compared to the dataset it is supposed to approximate. Overfit can be a consequence of overtraining, where the model have been trained too much on the training data. This makes the model inclined to follow every little twist and turn of the dataset, and "sticking too closely" to it. The problem with this is that it makes the model bad at generalizing to new data. In other words, an overfit model may perform very well on the training data, but perform very poorly when tested on new data.

3. METHODS

3.1. *Resampling: K-fold cross validation*

Resampling is a technique that is widely used in modern statistics. The technique consists of drawing repeated samples from a training set and then fitting your model to each sample. The point is to get as much use out of your finite data set as you can. This method might reveal information about the fitted model that would not have been discovered otherwise.
Although resampling has several advantages, it might also be computationally expensive because of the repetitive nature of the technique. There exists several resampling methods, such as the bootstrap method, jackknife resampling and the k-fold cross validation method. In this project we have chosen to focus on the k-fold cross validation method.

In addition to what's already been mentioned, k-fold cross validation can be used to estimate the potency of the model. Compared to other resampling methods it is relatively easy to understand and to implement. The k in the method's name refers to the number of groups to split the data in. The value of k is chosen in such a way that the training/test sets are large enough to be statistically representative for the whole data set.

TABLE 1

	Direct regression	Using 5-fold cross validation
R^2	0.9792	0.8571
MSE	0.0018	0.0119

NOTE. — results of regression with and without cross validation

The general approach is as follows:

1. Shuffle the data set randomly

2. Split the data set into k independent groups (folds)

3. Define (for example) the first $k-1$ folds as training data and leave out the k-th fold as test data. Then one can test the model on the test set (the k-th fold).

4. Next take $k-1$ folds as training data once again, but this time include the fold that was used as test data in step 3 in the training data. Then test the model again on the new test set.

5. Repeat until all k folds have been used as test data. One will then in the end have accumulated a sample of model evaluation scores (for example by using MSE or the R^2 score)

6. Lastly the skill of the model can be estimated by using the sample of model evaluation scores.

4. RESULTS

4.1. *Regression on the Franke function*

4.1.1. *OLS regression of the Franke function*

We start off by implementing a standard OLS regression on the Franke function with some added noise drawn from the normal distribution, and an OLS regression with 5-fold cross validation. The MSE and R^2-scores are shown in table 1, and as expected, the scores are somewhat poorer after k-fold cross validation. This is to be expected, but the more important fact is that none the values in the folds differ significantly, and therefore the regression model provides a good fit of the whole surface. The bias and variance of the OLS fit using different orders of polynomial are shown in figure 1, with 100-fold cross validation. The bias and variance steadily decrease when increasing the order up to 10. The variance shown here is the mean of the variance in the folds for each point on the surface. The bias behaves as anticipated when increasing to order 10. We expect a tighter fit for higher order polynomials. The variance is negligibly low for orders lower than 11. Because of the low variance compared to bias, the MSE is essentially equal to the bias. We do see, however, that for higher orders, the variance is larger than the bias, which is what we would expect.

4.1.2. *Comparison of ridge, lasso and OLS*

For the lasso and ridge methods, the results' dependence on λ is of interest when choosing the best model.

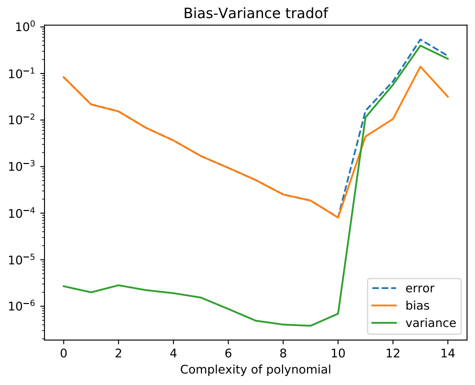
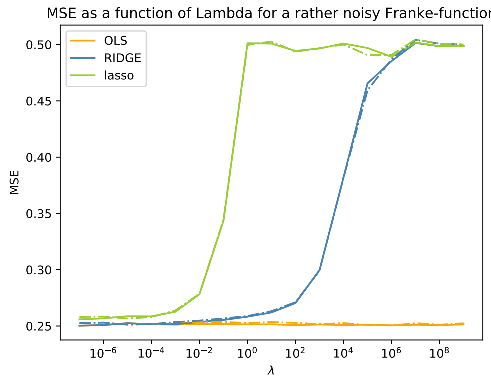


FIG. 1.— Bias-variance tradeoff for standard OLS with N=100

Figure 2 shows R^2-score and MSE for the three regression methods when increasing λ.

4.2. *Regression on terrain data*



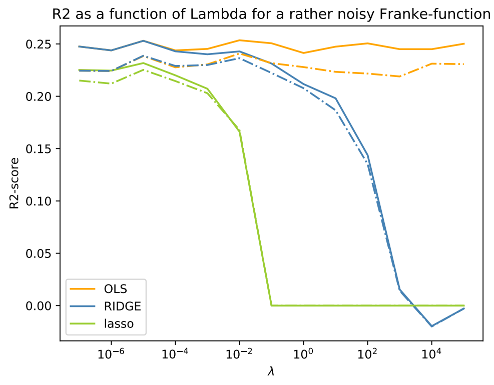
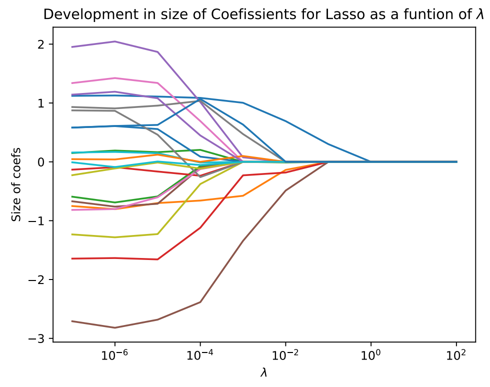


FIG. 2.— MSE and R2 vs λ

The OLS obviously does not change, and both the ridge and lasso provide worse fits for higher values of lambda. The Lasso fit has a steeper drop off in both measurements. Both of these measurements provide information of the fits compared to the input data, and because the coefficients in the ridge and lasso fits are dampened by increasing λ, which leads to the increase in MSE and decrease in R2. The size of the regression coefficients for the Lasso and Ridge methods are shown in figure 3.

The main property of the Lasso regression is clearly showcased here. As λ increases, the lasso algorithm



„ ,

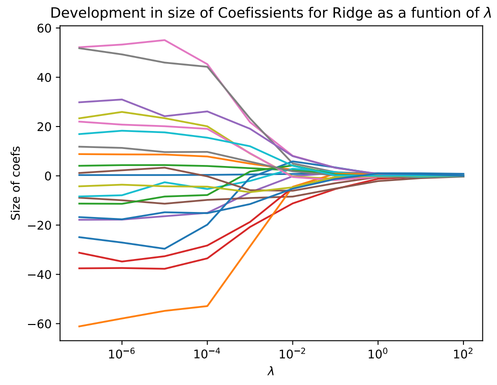


FIG. 3.— Ridge and Lasso coeffecient shrinkage with increasing λ

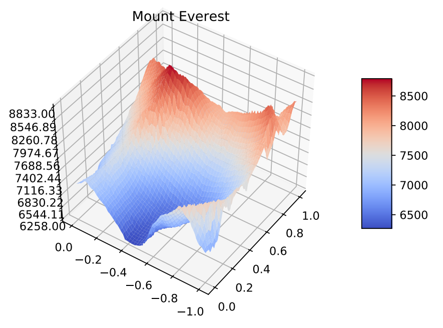
suppresses more of the coefficients completely, while in the ridge methods, the size of the coeficients decrease smoothly. The best regression method for this case than seems to be regular OLS. The ridge and lasso methods provide the better fit when $\lambda = 0$, which corresponds to the regular OLS method. The bias and variance both are at their lowest when using polynomials of order 10, so this is the order used in the chosen method.

4.3. *Regression on terrain data*

The raw and fitted data are shown in 5. from just observing the plots directly, the regression seems to reproduce the central properties of the input data when using a OLS with polynomial order 8. Looking at the R^2-score (figure 6) and MSE (figure 7), of the different methods when applied to the data give the same results as for the Franke function fit. Interestingly, for $\lambda = 10^2$, the R^2 score for the ridge method has a negative value, meaning it fits the data worse than a flat plane.

5. CONCLUSIONS

The data set of the mt. Everest terrain does not contain much noise. Additionally, in this case, the data is not used for prediction. With this in mind, an OLS method of regression is the most appropriate method. The aim here is to provide a tight fit which reproduces the central properties of the data. With the absence of significant noise and any distinctive outliers in the data, the ridge and lasso methods do not provide anything significant compared to the OLS method. The statistical quantities are shown to hold to a sufficient degree in the way we should expect. The OLS method is fairly simple



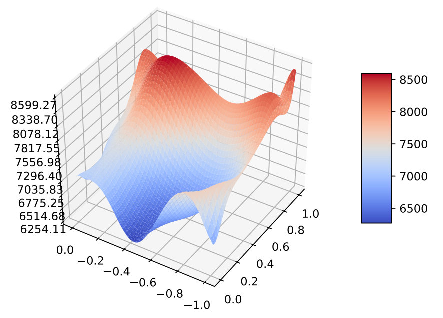
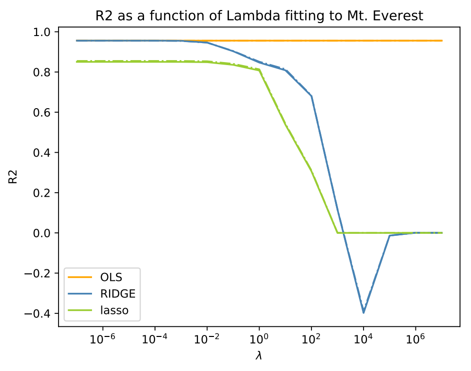


FIG. 4.— input (top) and fitted data (bottom)



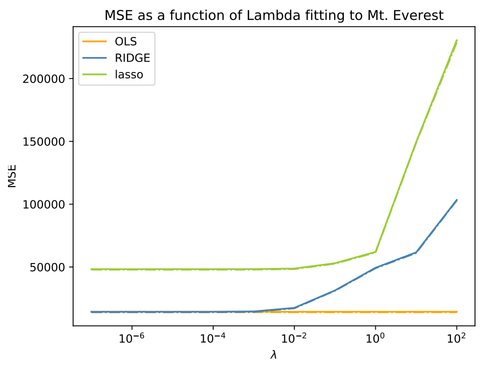


FIG. 5.— R^2 and MSE of mt.everest fit

to implement and analyze,so these results make it an obvious choice when fitting well behaved data sets such as this one

6. REFERENCES

1. Hastie, T., Tibshirani, R., Friedman, J. 2009. *The Elements of Statstical Learning Second Edition*

2. Most discussion in this paper is based on Mortens lectures and lecture notes.

3. Terrain data imported from https: //earthexplorer.usgs.gov/,