FYS-STK4155 H20 - Project 1: An assessment of OLS, Ridge and LASSO Regression on Aerially Distributed Data

Olav Fønstelien

October 10, 2020

Abstract

In this report I evaluate the OLS, Ridge and LASSO regression methods on aerially distributed data. Regression methods are powerful tools for data analysis and modeling, and I show in this report that these methods may reveal underlying truths about sampled data, specifically a quantification of the data's error or noise, and equally useful a model for simulating unseen data. This is shown first in a controlled manner on simulated data with known error using the Franke function. Then on geological data representing altitude above sea level of a mountain range. I show that for small data sets, Ridge and LASSO may give better estimates of these data, while for larger data sets, where the number of predictors is much smaller than the number of samples, OLS is preferrable.

1 Introduction

OLS, Ridge and LASSO are three popular methods for linear regression. They are related to each other in that Ridge and LASSO (least absolute shrinkage and selection operator) are refinements of OLS (ordinary least squares). They aim to *shrink* the coefficient estimates $\hat{\beta} \in \mathbb{R}^p$ towards zero by penalizing their value and hence reducing their variance [5].

In this report we will study each of these methods and how they perform when they are applied on nth degree polynomial fit with two variables. First, in Chapter 2 we will do a controlled, qualitative study where we will fit each of them to *Franke*'s function, and look at how they perform when varying the number of samples, the number of predictors and the method of resampling. Then, in Chapter 3, we will move on to an evaluation of their suitability when fitted to a highly irregular data set of unknown quality. For this we will use altitude levels of a mountain range. I conclude my evaluation by discussing the report's findings in Chapter 4.

2 Methods

2.1 Cost Functions and Bias-Variance Trade-Off

Given a data set $\mathcal{L} = \{(y_i, \mathbf{x}_i), i = 0, 1, ..., n-1\}$ where $\mathbf{y} = \{y_i\}$ are the response variables and $\mathbf{X} = \{\mathbf{x}_i\}$ are the predictors. Let's assume that the predictor $\mathbf{X}$ in $\mathcal{L}$ only partially explains the response variable $\mathbf{y}$, and that there is some underlying function $\mathbf{f}(\mathbf{X})$ in $\mathbf{y}$, such that $\mathbb{E}(\mathbf{y}) = \mathbf{f}(\mathbf{X})$ and

$$\mathbf{y} = \mathbf{f}(\mathbf{X}) + \epsilon \quad \text{, where} \quad \mathbf{f}(\mathbf{X}) = \mathbf{X}\beta \quad \text{and} \quad \epsilon \sim \mathcal{N}(0, \sigma^2). \tag{1}$$

In linear regression we aim find the best approximation $\tilde{\mathbf{y}} = \mathbf{X}\hat{\beta}$ to $\mathbf{f}$ by minimizing a cost function $C(\mathbf{X}, \beta)$, which reflects the expected test error of the approximation [5].

For OLS, the cost function is simply given by the mean squared error (MSE) of the approximation

$$C_{OLS}(\mathbf{X}, \beta) = \frac{1}{n}\sum_{i=0}^{n-1}(y_i - \tilde{y}_i)^2. \tag{2}$$

Now, to find the components making up the MSE, we use equation (1) and split the cost function into

$$\begin{aligned} C_{OLS}(\mathbf{X}, \beta) = \frac{1}{n}\sum_{i=0}^{n-1}(y_i - \tilde{y}_i)^2 &= \mathbb{E}[(\mathbf{y} - \tilde{\mathbf{y}})^2] = \mathbb{E}[(\mathbf{f} + \epsilon - \tilde{\mathbf{y}})^2] \\ &= \mathbb{E}[(\mathbf{f} - \tilde{\mathbf{y}})^2 + 2(\mathbf{f} - \tilde{\mathbf{y}})\epsilon + \mathbb{E}(\epsilon^2), \end{aligned}$$

and since $\mathbb{E}(\mathbf{f}) = \mathbb{E}(\tilde{\mathbf{y}})$ we get OLS's cost function on the form [5]

$$C_{OLS}(\mathbf{X}, \beta) = \mathbb{E}[(\mathbf{f} - \tilde{\mathbf{y}})^2] + \sigma^2. \tag{3}$$

The first term is the MSE of our approximation $\tilde{\mathbf{y}}$ to $\mathbf{f}$, which we aim to minimize. The second term is dependent on the data quality (instrument precision, round-off error, etc.) and is irreducible, or rather; reduction of the cost function beyond σ^2 leads to overfitting of the model. As a consequence, the quantity that we aim to determine in linear regression is the limit for the fit of our model, which is the irreducible error σ^2.

The first term in equation (3) can be reduced into model *bias* and model *variance*, as outlined in [2]. An approximation $\tilde{\mathbf{y}} = \mathbf{X}\hat{\beta}$ to $\mathbf{f}$ with too low complexity, say $\hat{\beta} \in \mathbb{R}^2$ for a parabolic function, will have low variability in its response to different training sets, but the error will be heavily dependent on the choice of model, which is the bias. A too complex approximation, on the other hand, will have high variability in its test response, but generally be less biased. The first term in equation (3) can be reduced further to reflect this:

$$\begin{aligned} \mathbb{E}[(\mathbf{f} - \tilde{\mathbf{y}})^2] &= \mathbb{E}[[(\mathbf{f} - \mathbb{E}(\tilde{\mathbf{y}})) - (\tilde{\mathbf{y}} - \mathbb{E}(\tilde{\mathbf{y}}))]^2] \\ &= \mathbb{E}[(\mathbf{f} - \mathbb{E}(\tilde{\mathbf{y}}))^2] - 2\mathbb{E}[(\mathbf{f} - \mathbb{E}(\tilde{\mathbf{y}}))(\tilde{\mathbf{y}} - \mathbb{E}(\tilde{\mathbf{y}}))] + \mathbb{E}[(\tilde{\mathbf{y}} - \mathbb{E}(\tilde{\mathbf{y}}))^2], \end{aligned} \tag{4}$$

where the second term reduces to zero since $\mathbb{E}[\mathbb{E}(\tilde{\mathbf{y}})] = \mathbb{E}(\tilde{\mathbf{y}})$. The cost function can thus be written on the form

$$\begin{aligned} C_{OLS}(\mathbf{X}, \beta) &= \mathbb{E}[(\mathbf{f} - \mathbb{E}(\tilde{\mathbf{y}}))^2] + \mathrm{Var}(\tilde{\mathbf{y}}) + \sigma^2 \\ &= \frac{1}{n}\sum_i (f_i - \mathbb{E}[\tilde{\boldsymbol{y}}])^2 + \frac{1}{n}\sum_i (\tilde{y}_i - \mathbb{E}[\tilde{\boldsymbol{y}}])^2 + \sigma^2, \end{aligned} \tag{5}$$

where the first term is the model's bias, or actually bias^2, the second is the variance in the response $\tilde{\mathbf{y}}$ to different test sets, and the last term is the error or noise in the data set, which as we saw is irreducible. We see that to obtain the best possible expected test error we must trade bias off against variance.

For OLS, a better fit may be obtained by increasing model complexity to reduce the bias, up to the point where variance starts to influence the model negatively. To allow increasing complexity further, a larger training data set must be used. Ridge and LASSO, however, allows us to further increase model complexity without increasing the training data, by putting an *extra* cost on variance. Since

$$\mathrm{Var}(\tilde{\mathbf{y}}) = \mathbb{E}[[\mathbf{X}\hat{\beta} - \mathbb{E}(\mathbf{X}\hat{\beta})]^2] \sim \mathrm{Var}(\hat{\beta}), \tag{6}$$

reducing $\mathrm{Var}(\hat{\beta})$ reduces $\mathrm{Var}(\tilde{\mathbf{y}})$. Ridge does this by putting a cost $\sum \lambda\beta_j^2$ on the coefficient values, shrinking larger coefficients more than smaller coefficients, while LASSO puts a cost $\sum \lambda|\beta_j|$ on the values which will try to drive them all the way to zero. See equations (7) and (8). Figure 1 below shows a comparison for the simple case $\tilde{\mathbf{y}} = \mathbf{I}\hat{\beta}$.

$$C_{Ridge}(\mathbf{X}, \beta) = \mathbb{E}[(\mathbf{y} - \tilde{\mathbf{y}})^2] + \sum_{i=1}^{n} \lambda\beta_j^2 \tag{7}$$

$$C_{LASSO}(\mathbf{X}, \beta) = \mathbb{E}[(\mathbf{y} - \tilde{\mathbf{y}})^2] + \sum_{i=1}^{n} \lambda|\beta_j| \tag{8}$$

2.2 The Design Matrix

We want to approximate the Franke function, given by

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

A pth degree polynomial in two variables $p_p(x,y) = 1 + x + y + x^2 + xy + y^2 ...$ can be used to approximate any function in two variables, such as the Franke function. We begin by collecting n uniformly distributed random samples of the function $\mathcal{L}_{Franke} = \{(f(x_i, y_i), (x_i, y_i), i = 0, 1, ..., n-1\}$. We first install the response variables (the y_is) in a vector $\mathbf{y} \in \mathbb{R}^n$. Then we set up our *design*

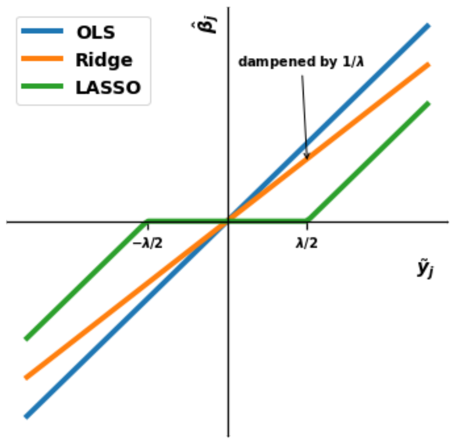

Figure 1: The effect of shrinking the coefficient estimates in Ridge and LASSO regression methods compared to OLS for the simple approximation $\tilde{\mathbf{y}} = \mathbf{I}\hat{\beta}$. We see that Ridge will try to dampen $\hat{\beta}_1$ to reduce $\tilde{y}$'s variance, while LASSO drives $\hat{\beta}_1$ to zero when it becomes sufficiently small.

matrix $\mathbf{X} \in \mathbb{R}^{n\times(p+1)}$ containing our predictors by evaluating each of the $p+1$ terms in $p_p(x_i, y_i)$ and install them in row i of $\mathbf{X}$:

$$\mathbf{x}_i = [1 \quad x_i \quad y_i \quad x_i^2 \quad xy \quad y_i^2 \quad \cdots]. \tag{10}$$

The next step is then to solve $\mathbf{y} = \mathbf{X}\beta + \epsilon$ to obtain the optimum coefficient estimates $\hat{\beta}$ with the constraints given by the linear regression method's cost function. In order not to unfairly penalize Ridge and LASSO's coefficient estimates due to the value of the predictors, we must standardize them, that is; give each of $\mathbf{X}$'s columns a standard deviation of one. We do this by dividing each of the columns by its standard deviation [5]. Also, to account for the intercept, we will center all except the first of $\mathbf{X}$'s columns by subtracting its mean.

2.3 The OLS Regression Method

The ordinary least square's cost function $C_{OLS}(\mathbf{X}, \beta)$ in equation (2) has a closed form solution for obtaining the optimum coefficient estimates given by

$$\hat{\beta} = (\mathbf{X}^\intercal\mathbf{X})^{-1}\mathbf{X}^\intercal\mathbf{y}. \tag{11}$$

See [8] for a thorough outline. Since $\mathbf{X}^\intercal\mathbf{X}$ may be uninvertible, we will use singular value decomposition $\mathbf{X} = \mathbf{U\Sigma V}^\intercal$ to calculate the pseudoinverse $\mathbf{X}^+ = \mathbf{V\Sigma}^{-1}\mathbf{U}^\intercal$, and then solve (11) by $\hat{\beta} = \mathbf{X}^+\mathbf{y}$. See [6].

Let us first use the OLS method on some randomly drawn samples from the Franke function (9), where we have added some normally distributed noise, ϵ. We see in Figure 2 that the variance in the approximation improves dramatically with increasing number of samples. This is explained by the decreased variance in the coefficient estimates, and we can illustrate this by plotting the confidence interval for each coefficient. As seen in Figure 3, for higher number of samples, the 95 % confidence interval shrinks, meaning that the range around the estimated coefficient $\hat{\beta}_j$ in which the true unknown coefficients β_j lays is narrowed [5]. The confidence interval is calculated by

$$\mathrm{CI}_{95}^{(\mathrm{j})} : \hat{\beta}_j \pm \mathrm{Var}(\hat{\beta}_j) \quad \text{where} \quad \mathrm{Var}(\hat{\beta}_j) = \sigma^2 \sqrt{[(\mathbf{X}^\intercal \mathbf{X})^{-1}]_{jj}}. \tag{12}$$

In the last term, due to [3], $\sigma^2 = \mathrm{Var}(\epsilon)$, which is known and set to 0.05 in this example.

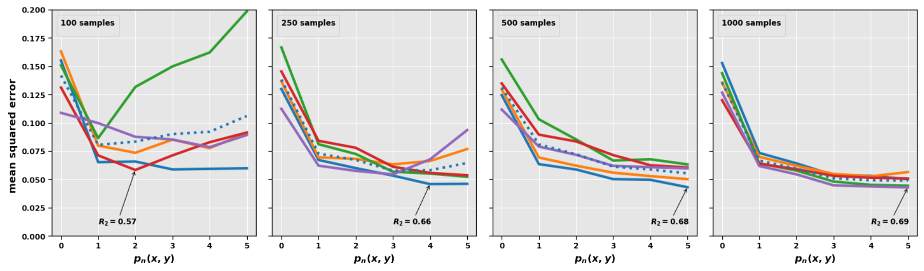

Figure 2: Some approximations to the Franke function (9) with added noise $\epsilon \sim \mathcal{N}(0, 0.05)$ for 100, 250, 500 and 1000 samples. Train-test split is 70/30 in all cases. Each curve represents a recalculation the estimated coefficients $\hat{\beta}$ for a new randomly drawn data set. We see that the variance in the approximation decreases with increasing number of samples. The dotted blue line is the average.

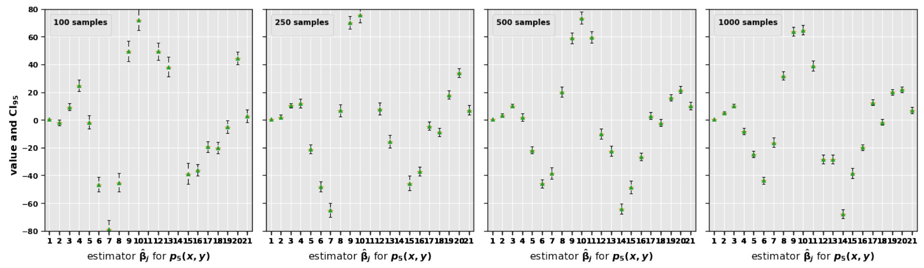

Figure 3: 95 % confidence interval (CI_{95}) for the coefficient estimators $\hat{\beta}_j$ of $p_5(x, y)$ approximating the Franke function (9) with 100, 250, 500 and 1000 randomly drawn samples. Added noise is $\epsilon \sim \mathcal{N}(0, 0.05)$ and the intervals are calculated using equation (12). We see that $\mathrm{CI}_{95}^{(\mathrm{j})}$ narrows for increasing number of samples, which is reflected in decreased $\mathrm{Var}(\tilde{y}_j)$. (See Figure 2).

Figure 2 above illustrates the importance of re-sampling in linear regression model evaluation. Especially for the 100 and 250 samples, it is difficult to draw any conclusion about the suitability of the model, but if we look at the average (dotted blue line), we get a result which is easier to interpret. From the

100-sample results, we should expect best performance for a first-order polynomial $p_1(x, y)$ *on average*, while for the 250-sample results, $p_3(x, y)$ seems to be best. For 500 and 1000, the results suggest that we could try to increase the model complexity beyond $p_5(x, y)$. Two more methodical ways re-sampling are Bootstrap and cross validation.

In Bootstrap re-sampling we split the data into a training and a test set, usually keeping 2/3 to 4/5 of the samples for training [4]. Then, assuming the training set has m samples, we collect from it the same number m samples *with* replacement, and calculate from those our coefficient estimates. Then we evaluate the model fit using the test data, note the result, and repeat the process for a number of times depending on the data set.

In k-fold cross-validation we also split the data into a training and a test set, but they are not fixed as for Bootstrap. Instead, we split the data into k equally sized parts, keep one of the k parts for test and train our model on the rest. Then we evaluate the model fit, note the result, and re-sample by picking the next of the k parts, and handing the already tested part back to the training set. The process is repeated k times (5-10 times in [4]) so that all of the data has been tested once.

In Figures 4 and 5 we see again the error estimates for the Franke function, but this time estimated with Bootstrap and cross validation re-sampling, respectively. Note that the scaling is different, so that even if the shape of the curves are similar, the actual error presented by Bootstrap in the overfitting region is at least a factor 10-100 larger than for cross validation. However, for the best fit, they both largely estimate the irreducible error σ^2 correctly (blue-shaded area) on the 500- and 1000-sample sets. On the 100- and 250-sample sets, however, there is a marked overestimation by Bootstrap. It is also worth to note that for the 100- and 250-sample sets, the best fit for the test data is achieved approximately when the training fit enters the irreducible error area.

Figure 4 also presents the estimated bias-variance trade-off. We see that as expected, variance increases with model complexity, while bias decreases up to a point where high complexity itself makes the model biased.

Lastly, we note that the choice of k does not seem to influence the results greatly for cross validation on this data set. We will use this to limit our scope when we do a similar evaluation of Ridge and LASSO further down.

2.4 The Ridge Regression Method

We saw in equation (7) that Ridge's cost function $C_{Ridge}(\mathbf{X}, \beta)$ introduces a penalty $\sum \lambda \beta_j^2$ on the coefficient estimates in order to reduce their variance. As with OLS, Ridge's cost function also has a closed form solution for obtaining the coefficient estimates:

$$\beta(\lambda) = (\mathbf{X}^\intercal \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^\intercal \mathbf{y}. \tag{13}$$

In contrast to the solution to OLS's cost function, due to the addition of $\lambda \mathbf{I}$ the matrix $(\mathbf{X}^\intercal \mathbf{X} + \lambda \mathbf{I})$ *is* invertible, which eases its solution. Again, see [8] for a

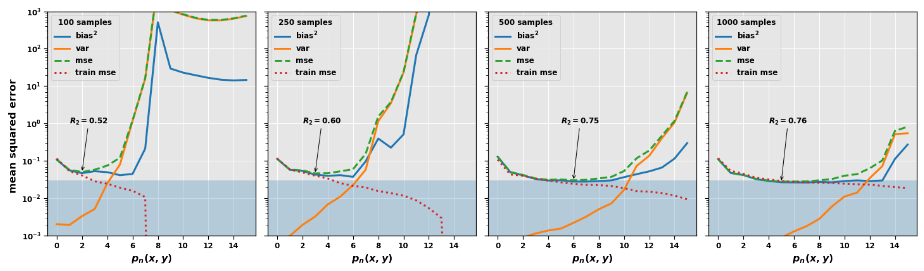

Figure 4: Estimated error for the OLS polynomial fit to the Franke function (9) with 100, 250, 500 and 1000 samples and Bootstrap re-sampling. Added noise is $\epsilon \sim \mathcal{N}(0, 0.03)$, indicated by the blue-shaded area. $\mathbf{X}$ has been centered, but not scaled. The estimates are calculated with a train-test split of 70/30 and with 25 bootstraps. We see that for the 500- and 1000-sample sets, the model estimates the irreducible error σ^2 well, while for the 100- and 250-sample sets it overestimates it. Cross validation performs a little better in this area. See Figure 5. Concerning bias-variance trade-off, we see that as expected, for all data sets variance increases with increasing model complexity, while bias is reduced until complexity itself makes the model biased.

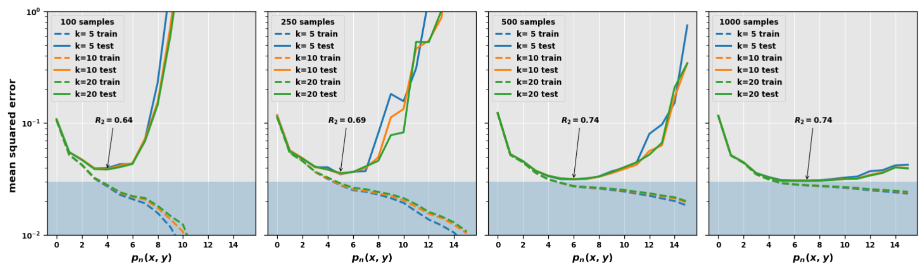

Figure 5: Estimated error for the OLS polynomial fit to the Franke function (9) with 100, 250, 500 and 1000 samples and k-fold cross validation re-sampling. Added noise is $\epsilon \sim \mathcal{N}(0, 0.03)$, indicated by the blue-shaded area. $\mathbf{X}$ has been centered, but not scaled. The estimates are calculated with ks 5, 10 and 20. We see that the choice of k does not influence the estimates except for in the overfitted area. For the 500- and 1000-sample sets, the model estimates the irreducible error σ^2 well, while for the 100- and 250-sample sets it overestimates it, but less so than does Bootstrap, though. See Figure 4.

thorough outline. Performing singular value decomposition on $\mathbf{X}$ does reveal an interesting property of Ridge regression. As shown in [7], with $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\intercal$ and $\mathbf{u}_j$ being the jth row of $\mathbf{U}$, we can write equation (13) as

$$\beta(\lambda) = \sum_{j=1}^{p} \mathbf{u}_j \frac{\sigma_j^2}{\sigma_j^2 + \lambda} \mathbf{u}_j^\intercal \mathbf{y}. \tag{14}$$

This shows that introduction of the λ term in Ridge's cost function (7) dampens or shrinks the coefficient estimates β. See Figure 1 again for an illustration of this.

Even if we have a closed form solution to equation (13), finding the *optimum* solution $\hat{\beta} = \beta(\hat{\lambda})$ must be done by trial and error. Figures 6 and 7 show plots of searches for $\hat{\lambda}$ using Bootstrap and k-fold cross validation resampling, respectively. We see that both methods estimate the error well, but that cross validation only needs 500 samples for an exact approximation of the irreducible error σ^2, whereas Bootstrap needs 1000 (blue-shaded area). They both find the best fit for $p_5(x, y)$, all though for slightly different λ value. MSE remains more or less unaffected by model complexity after that. For the other data sets, we see again as we saw for OLS that cross validation has better estimates than Bootstrap, even if the differences are less pronounced this time.

Figure 6 shows how Ridge operates: Increasing λ forces the variance in the response $\mathrm{Var}(\tilde{\mathbf{y}})$ down, with a corresponding cost to bias. At the optimum point, $\hat{\lambda} = 3.09 \cdot 10^{-5}$ here, this trade-off reaches a best-ratio, which we select to fit our model.

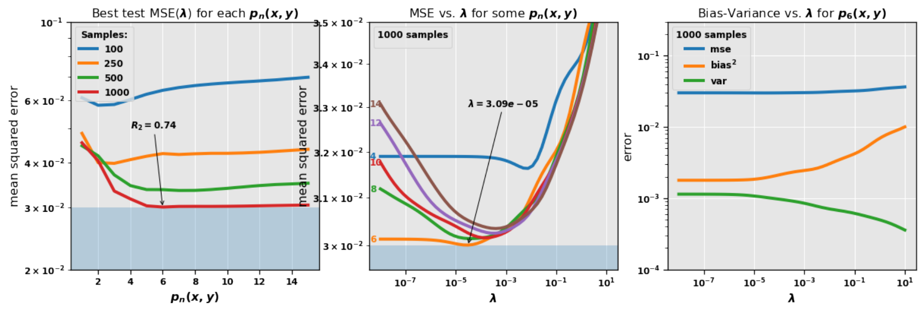

Figure 6: Estimated error for the Ridge polynomial fit to the Franke function (9) with 100, 250, 500 and 1000 samples and Bootstrap re-sampling. Added noise is $\epsilon \sim \mathcal{N}(0, 0.03)$, indicated by the blue-shaded area. $\mathbf{X}$ has been centered and scaled. The estimates are calculated with a train-test split of 70/30 and with 25 bootstraps. To the left we see that we see that severe overfitting is prevented in all cases. For the 1000-sample set, the model estimates the irreducible error σ^2 accurately at around $p_5(x, y)$ and it remains very stable for increasing model complexity. The 500-sample set has a slight overfit beginning at around $p_{10}(x, y)$, but the error approximation is still good. The 250-sample set overestimates σ^2 by about 33 %, but has about the same level of overfitting as the 500-sample set. The 100-sample has a best approximation of about $2\sigma^2$. Cross validation performs a better for small data sets. See Figure 7. To the right we see the bias-variance trade-off depending on λ. Increasing λ reduces the variance in the response, which comes with a cost in bias, as expected from Ridge.

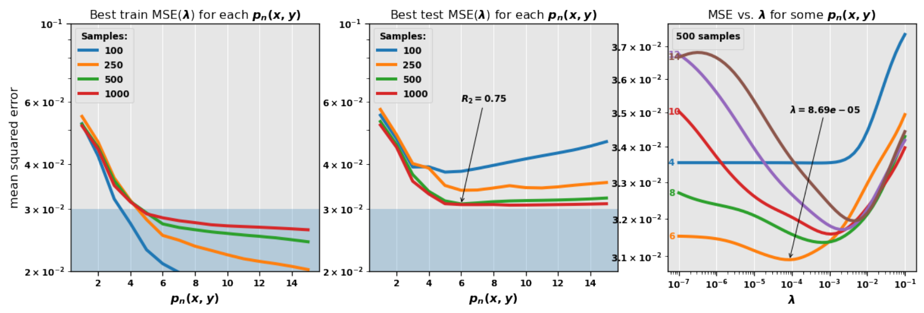

Figure 7: Estimated error for the Ridge polynomial fit to the Franke function (9) with 100, 250, 500 and 1000 samples and 5-fold cross validation re-sampling. Added noise is $\epsilon \sim \mathcal{N}(0, 0.03)$, indicated by the blue-shaded area. $\mathbf{X}$ has been centered and scaled. To the left we see that the train MSE indicates heavy overfitting, but in the middle that Ridge prevents severe overfitting in all test cases. For the 500-sample set, the model estimates the irreducible error σ^2 accurately at $p_5(x, y)$ and that it has a slight overfit after that, but the error approximation is still good. The 1000-sample remains approximation remains very stable for increasing model complexity. The 250-sample set overestimates σ^2 only by about 10 %, and has about the same level of overfitting as the 500-sample set. The 100-sample overestimates σ^2 by about 33 %. Overall, cross validation performs better than Bootstrap on this data set. See Figure 6. To the right we see test MSE as a function of λ for the 500-sample approximations.

2.5 The LASSO Regression Method

We saw in equation (8) that like Ridge's cost function, LASSO's cost function $C_{LASSO}(\mathbf{X}, \beta)$ introduces a penalty $\sum \lambda |\beta_j|$ on the coefficient estimates in order to reduce their variance. In contrast to Ridge, and OLS for that matter, LASSO's cost function does not have a closed form solution, meaning that for each λ we must search both for a solution that minimizes the cost function itself. In this report we will use `sklearn.linear_model.Lasso` in Python's `scikit-learn` module to search for this solution.

Figures 8 and 9 show plots of searches for $\hat{\beta}, \hat{\lambda}$ using Bootstrap and k-fold cross validation resampling, respectively. Again, we see that both methods estimate the irreducible error almost equally well for the larger data sets, while cross validation is a little more stable for smaller data sets. The bias-variance trade-off, seen to the right in Figure 8, behaves as we expect, with increased λ reducing the variance in the response $\mathrm{Var}(\tilde{\mathbf{y}})$. This comes of course with a cost in model bias.

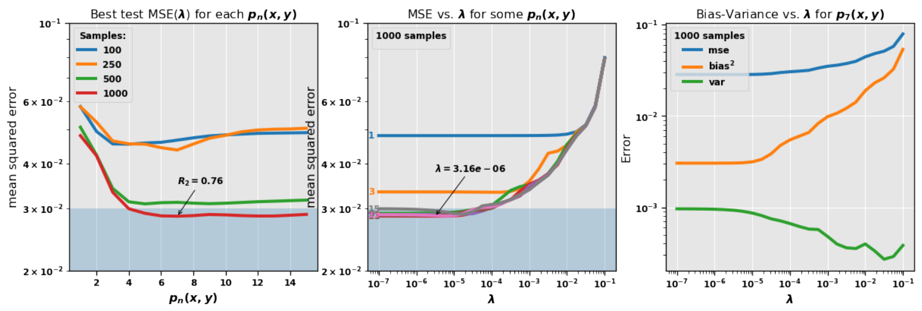

Figure 8: Estimated error for the LASSO polynomial fit to the Franke function (9) with 100, 250, 500 and 1000 samples and Bootstrap re-sampling. Added noise is $\epsilon \sim \mathcal{N}(0, 0.03)$, indicated by the blue-shaded area. $\mathbf{X}$ has been centered and scaled. The estimates are calculated with a train-test split of 70/30 and with 25 bootstraps. To the left we see that severe overfitting is prevented in all cases. For the 1000-sample set, the model estimates the irreducible error σ^2 fairly well at around $p_4(x, y)$ and it remains very stable for increasing model complexity. It may seem a little strange that the fit in this case is *better* than σ^2, but this comes from the randomness of the test set samples, indicating that they are ever so slightly off-center. Repeating the experiment with a different test set could yield a best fit slightly above the line. The 500-sample approximation performs almost equally well as the 1000-sample approximation, and show no sign of overfitting. The 100- and 250-sample sets overestimates σ^2 by about 50 %, and show slight signs of overfitting. Bootstrap performs somewhat better than cross validation on smaller data sets, and worse on larger. See Figure 9. To the right we see the bias-variance trade-off depending on λ. Increasing λ reduces the variance in the response, which as we see comes with a quite significant cost in bias, leading to a relatively small $\hat{\lambda}$. `scikit-learn Lasso()` options: `{'max_iter':1E4, 'fit_intercept':False, 'tol':.001}`

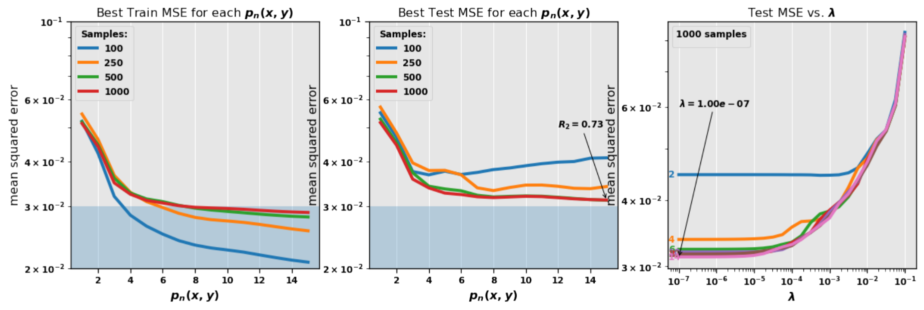

Figure 9: Estimated error for the LASSO polynomial fit to the Franke function (9) with 100, 250, 500 and 1000 samples and 5-fold cross validation resampling. Added noise is $\epsilon \sim \mathcal{N}(0, 0.03)$, indicated by the blue-shaded area. $\mathbf{X}$ has been centered and scaled. To the left we see that the train MSE indicates heavy overfitting, but in the middle that LASSO prevents severe overfitting in all test cases. For the 500- and 1000-sample sets, the model estimates the irreducible error σ^2 quite accurately at $p_5(x, y)$ and do not show any overfitting after that. Increasing model complexity beyond $p_{15}(x, y)$ did not improve the R^2 value when that was tested (not shown). The 250-sample set also estimates σ^2 well at $p_8(x, y)$, before it gets a slight overfit. The 100-sample overestimates σ^2 by about 20 % at $p_4(x, y)$, and then. Bootstrap performs somewhat better than cross validation on smaller data sets, and worse on larger. See Figure 8. To the right we see test MSE as a function of λ for the 1000-sample approximations. `scikit-learn Lasso()` options: `{'max_iter':1E4, 'fit_intercept':False, 'tol':.01}`

3 Results

To finish our qualitative evaluation of the OLS, Ridge and LASSO regression methods, lets first look at what each of them do to the coefficient estimates $\hat{\beta}_j$ when we do the polynomial fit to the Franke function (9). After that we will evaluate their performance on *real* data, altitude levels of a mountain range.

3.1 OLS, Ridge and LASSO Coefficient Estimates

We expect that Ridge shall *shrink* the coefficients more or less evenly, but not all the way to zero as we expect LASSO to do, since LASSO shall also pick those predictors that best explain $\mathbf{f}(\mathbf{X})$, and preferably drive the rest to zero. In addition, and as a consequence of this, we expect that the variance shall be greatly reduced, and more so for LASSO than Ridge, since it is more agressive.

Figure 10 shows the coefficient estimates $\hat{\beta}_j$ for OLS, Ridge and LASSO when fitting $p_6(x, y)$ to the Franke function with 1000 samples. λs are chosen to give the best fit, according to our evaluation with cross validation above in Figures 7 and 9. We see that indeed, the Ridge and LASSO do reduce the values of the estimates, and the variance is also reduced. In both cases also more aggressively for LASSO than for Ridge. However, with this model on this data set LASSO does not drive any of the coefficient estimates to zero, so we cannot derive some underlying truth about $\mathbf{f}(\mathbf{X})$ from them here, which may be reasonable since OSL's fit is just as good as Ridge and LASSO's for this model complexity.

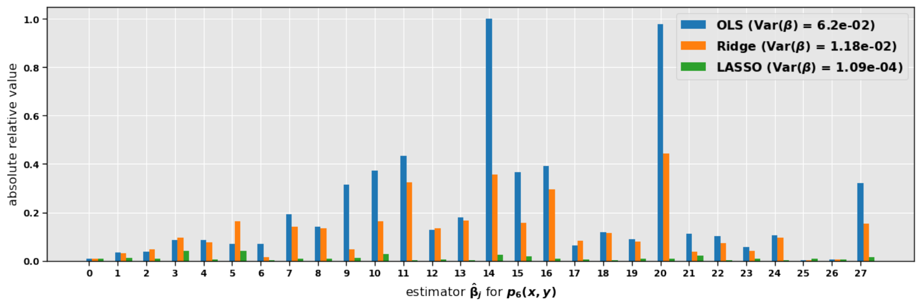

Figure 10: OLS, Ridge and LASSO coefficient estimates $\hat{\beta}_j$ when fitting $p_6(x, y)$ to the Franke function with 1000 random samples and noise $\epsilon \sim \mathcal{N}(0, 0.03)$. λ values for Ridge and LASSO are chosen to give the best fit, as found in the cross validation evaluation: $\lambda_{Ridge} = 8.7 \cdot 10^{-5}$; $\lambda_{LASSO} = 3.2 \cdot 10^{-6}$. We see that LASSO shrinks the estimates more aggressively than does Ridge, and that $\text{Var}(\hat{\beta})$ follows the same pattern. However, none of LASSO's coefficients are driven to zero.

3.2 Geological Data: Comparison of the OLS, Ridge and LASSO Regression Methods

We will now investigate the performance of each of the regression methods on a highly irregular data set. The data set represents the altitude above sea level of a mountain range in the central United States, collected from [1]. See Figure 14, left part. The data has unknown quality; that is, the irreducible error level σ^2 is unknown. We will however assume that the error is normally distributed with zero mean, such that the data again can be represented as in equation (1), repeated here for convenience:

$$\mathbf{y} = \mathbf{f}(\mathbf{X}) + \epsilon \quad \text{, where} \quad \mathbf{f}(\mathbf{X}) = \mathbf{X}\beta \quad \text{and} \quad \epsilon \sim \mathcal{N}(0, \sigma^2).$$

The data set contains $3601 \times 3601 \approx 13$ million data points, which is prohibitively large. We therefore draw uniformly distributed random samples from the data set to form subset of n samples which we do our study on, as for the Franke function. Figure 11 shows that the noise level σ^2 probably lies a little bit above $3 \cdot 10^{-3}$, and that using a larger data set than 5000 samples does not bring any benefits with regards to model fit parameter R^2, so we will base our study on 5000 samples.

Ridge regression on the 5000-sample data set give slightly worse fit, as we see in Figure 12. The model R^2 is only 0.80, compared to 0.83 for simple OLS, and this on the 30th polynomial, as opposed to the 20th for OLS. We see that $\hat{\lambda}$ for the Ridge's $p_{30}(x, y)$ is very small at $3.7 \cdot 10^{-10}$, indicating that there is not much variance in the response for Ridge to trade off against model bias in this data set. Beyond this level, it might be numerical instability which causes the estimates to increase.

Figure 13 shows the results for LASSO regression on the same data set. We see that the fit is markedly worse than both Ridge and OLS. R^2 fit is only 0.71 at the 30th polynomial, but as we see the trend suggests that it would be possible to achieve at least some improvement to the fit. As for Ridge, $\hat{\lambda}$ for the best polynomial $p_{30}(x, y)$ is small, but stabilizing at around $1.0 \cdot 10^{-7}$. Again as for Ridge, this suggests that the data set does not offer the opportunity to trade bias against variance.

In Figure 14, we see how the three approximations reproduces the original data set. For OLS and, if to a lesser extent, also for Ridge, we clearly see the main features are preserved, but that all smaller details disappears. For LASSO, the reproduction rather poor with all original features completely blurred.

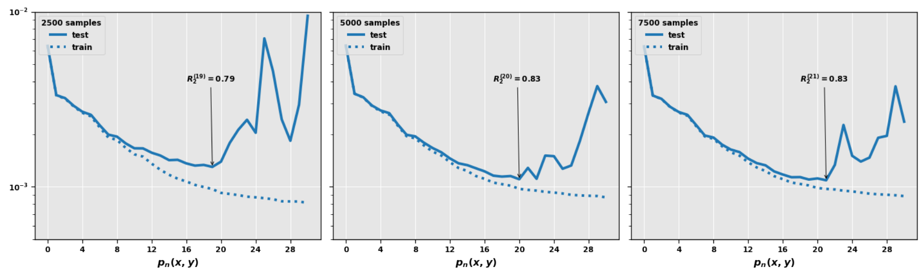

Figure 11: OLS polynomial fit to geological data with 2500, 5000 and 7500 randomly selected samples. Error approximation by 5-fold cross validation. We see that the noise level probably lies a little bit above $3 \cdot 10^{-3}$, and that collecting more than 5000 samples does not give better fit for the polynomial. Hence, we base our evaluation of the suitability of OLS, Ridge and LASSO on the data set with 5000 samples.

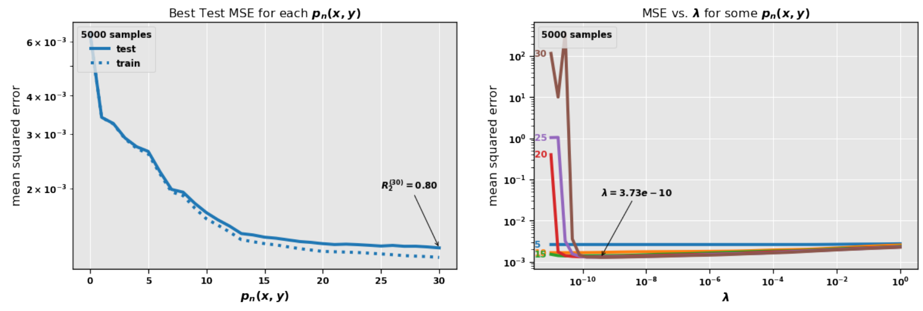

Figure 12: Ridge polynomial fit to geological data with 5000 randomly selected samples. Error approximation by 5-fold cross validation. Noise level approximated to about $3 \cdot 10^{-3}$ using OLS (see Figure 11). The model fit $R^2 = 0.80$ is markedly poorer than OLS and Ridge. The small $\hat{\lambda}$ for $p_{30}(x, y)$ suggests that bias-variance trade-off does not pay off greatly in this case.

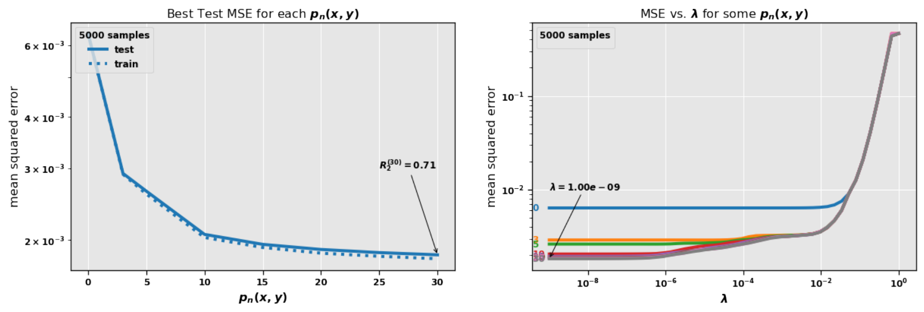

Figure 13: LASSO polynomial fit to geological data with 5000 randomly selected samples. Error approximation by 5-fold cross validation. Noise level approximated to about $3 \cdot 10^{-3}$ using OLS (see Figure 11). The model fit $R^2 = 0.71$ and very small $\hat{\lambda}$ for $p_{30}(x, y)$ suggests that bias-variance trade-off does not pay off greatly in this case. We see however that

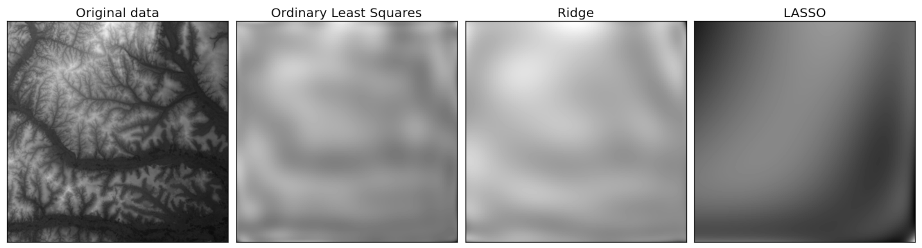

Figure 14: To the left we see the original geological data reflecting altitude above sea level of a mountain range. The three images to the right are reproductions generated by solving the optimum $p_n(x_i, y_i)$ for $i \in [0, 3600]$ obtained for each regression method OLS, Ridge and LASSO, respectively. OSL and Ridge reproduce the main features of the original, while LASSO's reproduction is completely unrecognizable.

4 Discussion and Conclusion

In the controlled case with the Franke function, we saw that the assumption that the data points $\mathbf{y}, \mathbf{X}$ could be represented by a linear function $\mathbf{f}(\mathbf{X}) = \beta\mathbf{X}$ with an error $\epsilon \sim \mathcal{N}(0, \sigma^2)$. We then saw that the best-fit solution $\hat{\beta}$ to the equation $\mathbf{y} = \beta\mathbf{X} + \epsilon$, given the cost functions for either OLS, Ridge and LASSO lets us reproduce the underlying function $\mathbf{f}$ with high accuracy, and that we also obtain a precise estimate for the error σ^2 in the data.

The performance of the three regression methods appear to not be very dependent on whether we use Bootstrap of cross validation resampling if the data set is large enough, but for smaller data sets, cross validation performed best on the Franke data. This is good for two reasons; firstly that for this data it does not matter much which method we choose, and secondly because k-fold cross validation is faster than bootstrap since it runs through the data only 5 to 10 times, as opposed to 20 to 30 times for Bootstrap. Hence, cross validation is the method of choice here.

We have also seen that Ridge and LASSO with cross validation perform better than OLS on smaller data sets, where number of n is not much greater than p. The reason for this is that these methods let us choose a model with higher complexity than OLS, since they remedy the high variance in the response $\mathrm{Var}(\mathbf{y})$ which would otherwise occur by penalizing large coefficient estimates and reducing their variance.

On the geological data, on the other hand, $n >> p$, which is ideal for OLS, since complexity can be increased without too high cost in variance, and the randomness in the test data will closely resemble that of the training data. As a result, OLS performs markedly better than Ridge here, and LASSO performs quite poorly, as we saw in the image reproduction in Figure 14.

Future expansion of this study should include more kinds of *aerially* distributed data, like fertility vs. location or income vs. a persons physical weight and height, to proper gauge where OLS, Ridge and LASSO each come to best use in these applications.

References

[1] United states geological survey. https://earthexplorer.usgs.gov/.

[2] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics New York, 2001.

[3] Morten Hjort-Jensen. Fys-stk4155 applied data analysis and machine learning, lecture notes fall 2020. https://compphysics.github.io/MachineLearning/doc/web/course.html, University of Oslo, August 2020.

[4] Morten Hjort-Jensen. Fys-stk4155 applied data analysis and machine learning, project 1. https://compphysics.github.io/MachineLearning/doc/Projects/2020/Project1/html/Project1.html, University of Oslo, August 2020.

[5] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An introduction to statistical learning*, volume 112. Springer, 2013.

[6] David C Lay, Steven R Lay, and Judi J McDonald. *Linear algebra and its applications*. Pearson, 2016.

[7] Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

[8] Wessel N van Wieringen. Lecture notes on ridge regression. *arXiv preprint arXiv:1509.09169*, 2015.