COURSE 02457

Non-Linear Signal Processing: Answers to Exercise 4

Checkpoint 4.1

In this checkpoint we investigate the effect of the training and test errors per example as a function of the training set size.

First we simulate training and test data using a three dimensional weight vector w_t, and then we make a new set of training and test set that is equal to the old one, but without the last (third) dimension. This leaves us with two sets of training and test data sets that are almost the same, the only difference being that one of them has one extra dimension.

Then, for a subset of the training sets that we vary the size of (from N_{min} to N_{max}), the weights for the two training sets are estimated using least squares, and the weights are used to predict another set of data that can be compared to the training and test sets to estimate the training and test error errors. This is done 10 times for each size of the subset of the training set and the errors are averaged. The result can be seen in Fig. 1.

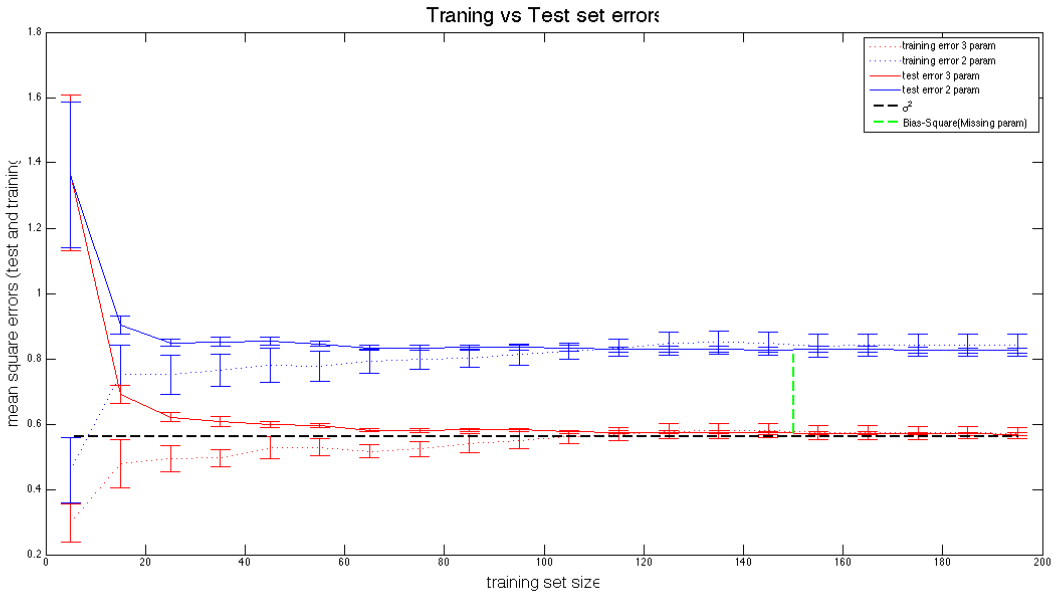


Figure 1: $N_{min} = 5$, $N_{max} = 200$ $N_{test} = 10\,000$

In Fig. 1 the smallest size of training set is 5, and for this value we see that the training error is quite low for both training sets. Had the training set only comprised of 3 data points, the training error would have been zero, because it would have been possible to fit a plane containing all three points. For 5 data points, the error is not zero any more, but it is still small.

Conversely, for small sizes of the training set, the test error is quite large because the plane is almost perfectly fitted to the small number of training data points. The test data points, being slightly different from the training data points due to the noise, are not as well described by the plane, and therefore produces a large test error.

As the size of the training set increases, the training and test error converges. The two different data sets (with different dimensionality) converges to different values. The data set with the original three dimensions converge to the value $(noiselevel = \epsilon)^2$, because when we have a large training set, the fitted model will approximately be equal to the true model, so the resulting generalization error becomes

$$E_G(w) = \frac{1}{2N}\sum_{n=1}^{N}(w^T x_n - t_n)^2 \tag{1}$$

$$= \frac{1}{2N}\sum_{n=1}^{N}(w^T x_n - w_{true}^T x_n + \epsilon_n)^2 \tag{2}$$

$$= \frac{1}{2N}\sum_{n=1}^{N}\epsilon_n^2 \tag{3}$$

The model of the smaller data set with one dimension missing produces errors that converges to a higher value than the model of the bigger training set. This is because the smaller model is missing information in the data about the third parameter w_3, so these errors converges to $(noiselevel)^2 + w_3^2$.

To better illustrate this, the third (true) weight is set to zero and the program is run again. Now the smaller and the bigger data set contains the same information, so all the errors converges to $(noiselevel)^2$, see Fig. 2.

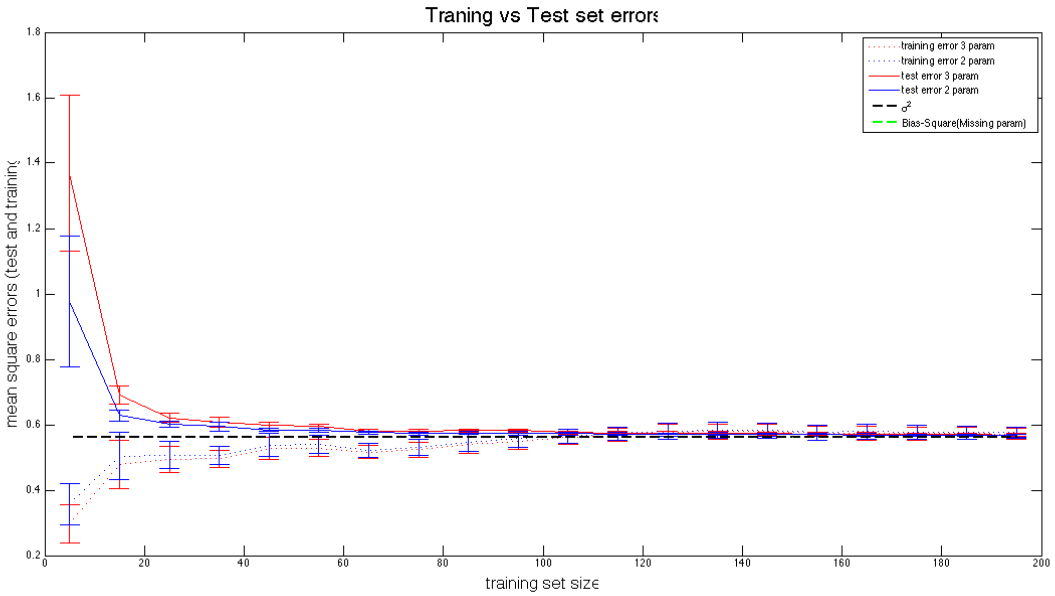


Figure 2: $w_3 = 0$, $N_{min} = 5$, $N_{max} = 200$ $N_{test} = 10\,000$

Checkpoint 4.2

This checkpoint applies the same methods as the previous checkpoint to predict the number of sunspots. We have a data set of the annual average number of sunspots measured from 1700 - 1979. The first 220 years are used as training set and the last 58 years comprise the test set. These two sets together with the prediction for $d = 9$ can be seen in Fig. 3.

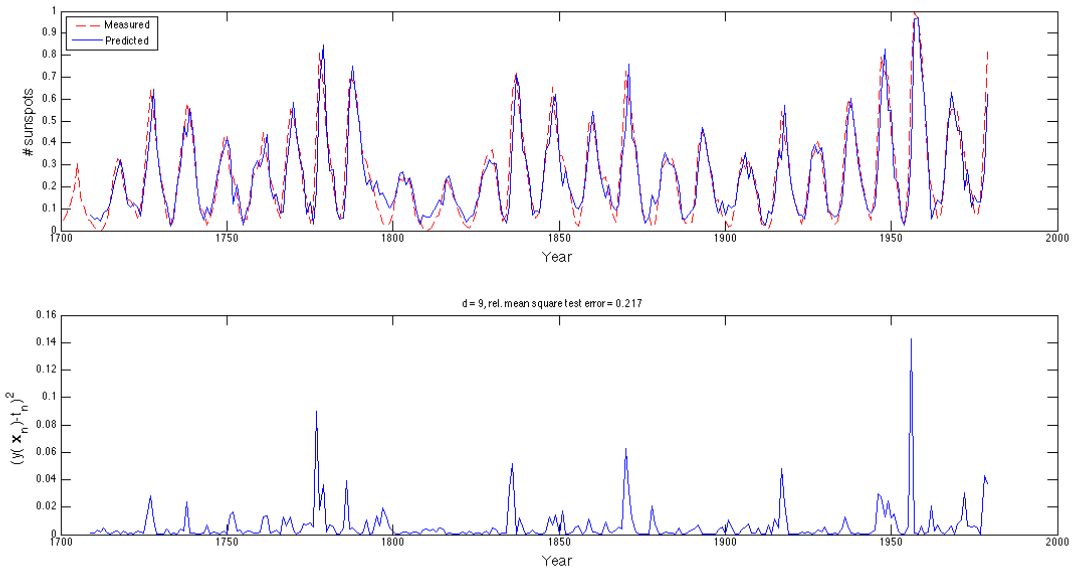
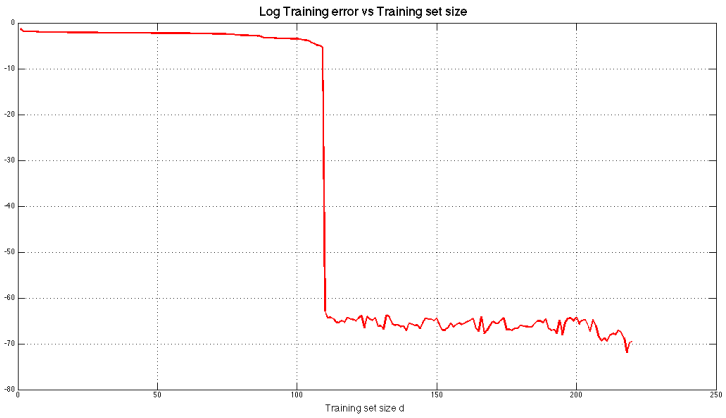
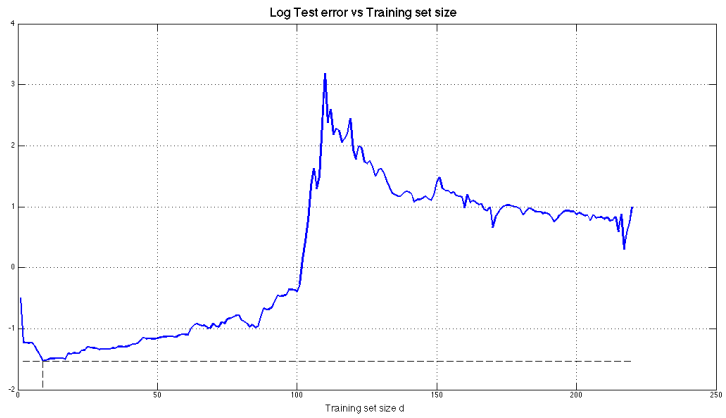


Figure 3: The sunspot data together with an example of a prediction from an AR(9) model. Notice that the first 9 years are not predicted, as the model needs 9 previous years before it can predict the next.

By evaluating the test error per example as a function of d, which is the order of the AR-model, we seek to determine the optimal order of the fit, see Fig. 4b. We see that the test error has a small global maximum at $d = 9$, which corresponds nicely with what seems to be the period of the oscillations in the time series in Fig. 3.



(a) The training error as a function of model order d. Notice the sharp drop in the error when d reaches half the size of the training set $N_{train} = 220$.



(b) The test error as a function of model order d. Note the dramatic rise in the test error when the training error goes to zero.

Checkpoint 4.3

This checkpoint illustrates the bias variance decomposition for a linear model. The bias and variance components of the error are plotted for models with increasing weight decay parameter α.

The loss/generalization error is

$$\mathbb{E}[L] = \int \int (y(x) - t)^2 p(x,t) dx dt \tag{4}$$

and our goal is to find the $y(x)$ minimizes $\mathbb{E}[L]$. After some algebraic gymnastics, the generalization error can be rewritten to

$$\mathbb{E}[L] = (bias)^2 + variance + noise \tag{5}$$

The first term on the right hand side is the $bias^2$ which represents how much the average prediction over all data sets differ from the optimal regression function. The second term is the variance as the form of it indicates, and it represents how much the solutions $y(x;\mathcal{D})$ for individual data sets vary around their average, any gives a measure of how sensitive the solution is to a particular choice of data set. The last term is the intrinsic variability of the target data and can be regarded as noise.

There is a trade-off between the bias and the variance, typically when one of them gets lower, the other gets higher. The optimal model is the one where the two are balanced. By solving for the weights in model $y(x;\mathcal{D})$ using a regularized least squares with a weight decay parameter, we can exercise some level of control over how much the expected loss is dominated by the bias or the variance. So this checkpoint centers around finding the optimal choice of the weight decay parameter so that the bias and variance are properly balanced.

The script **main4c.m** is implemented to do this for the plane-fit example in **checkpoint 4.1**. Here, a test set is created using the true weights w_{true} as before, but now we add a loop over different weight decay values going from λ_{min} to λ_{max}, and for each λ, several (l) training sets X^l_{train} are created and the weights w^l estimated for each training set using regularized least squares,

$$w = (X^T_{train}X_{train} + \alpha I)^{-1}(X^T_{train}X_{train}) \tag{6}$$

where α is the decay parameter. This results in several predictions $Y^l_{test} = X^l_{train}w^l$ that are averaged and compared against the test target values T_{test}, and the $bias$ can be computed as

$$bias^2 = \frac{1}{N}\sum_{n=1}^{N}(\bar{y}(x) - \mathbb{E}[t|x])^2 \tag{7}$$

where $\bar{y}(x) = mean(X^i_{test}w)$ is the mean of the predictions and the optimal prediction $\mathbb{E}[t|x] = X_{test}w_{true}$. Next the mean error is calculated as

$$ME = \frac{1}{L}\sum_{l=1}^{L}\frac{1}{N}\sum_{n=1}^{N}(Y^l_{test}(x_n) - \mathbb{E}[t|x_n])^2 \tag{8}$$

and the variance is calculated as

$$variance = ME - bias^2 \tag{9}$$

These three quantities were calculated as a function of the weight decay parameter, and the result can be seen in Fig. 4. Since we are interested in the weight decay parameter that minimizes both the variance and the bias, we pick the decay parameter corresponding to the minimum mean error, which is $\alpha = 0.4481$.

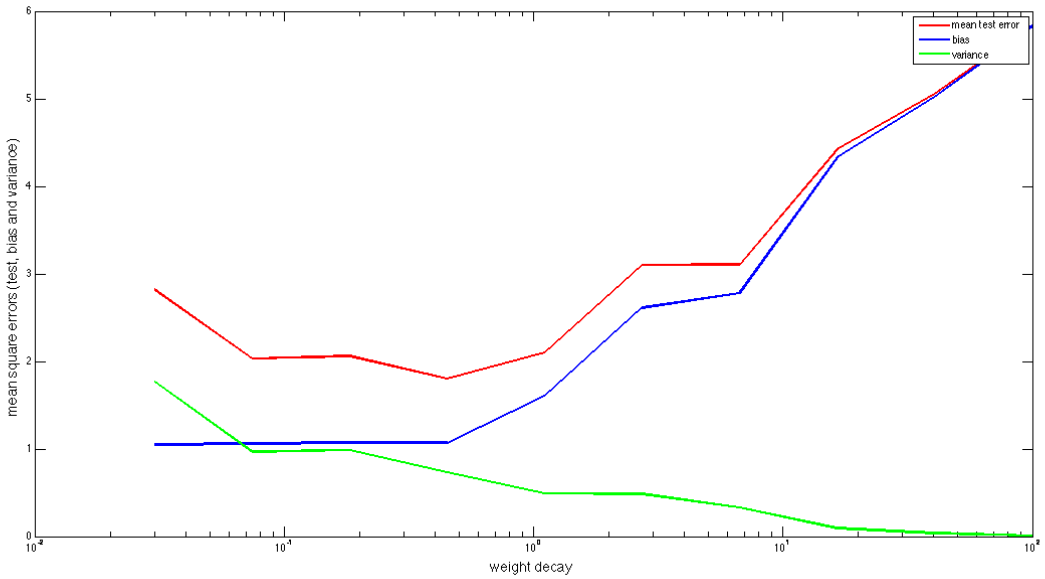


Figure 4: The bias-variance trade-off regulated by the weight decay parameter.

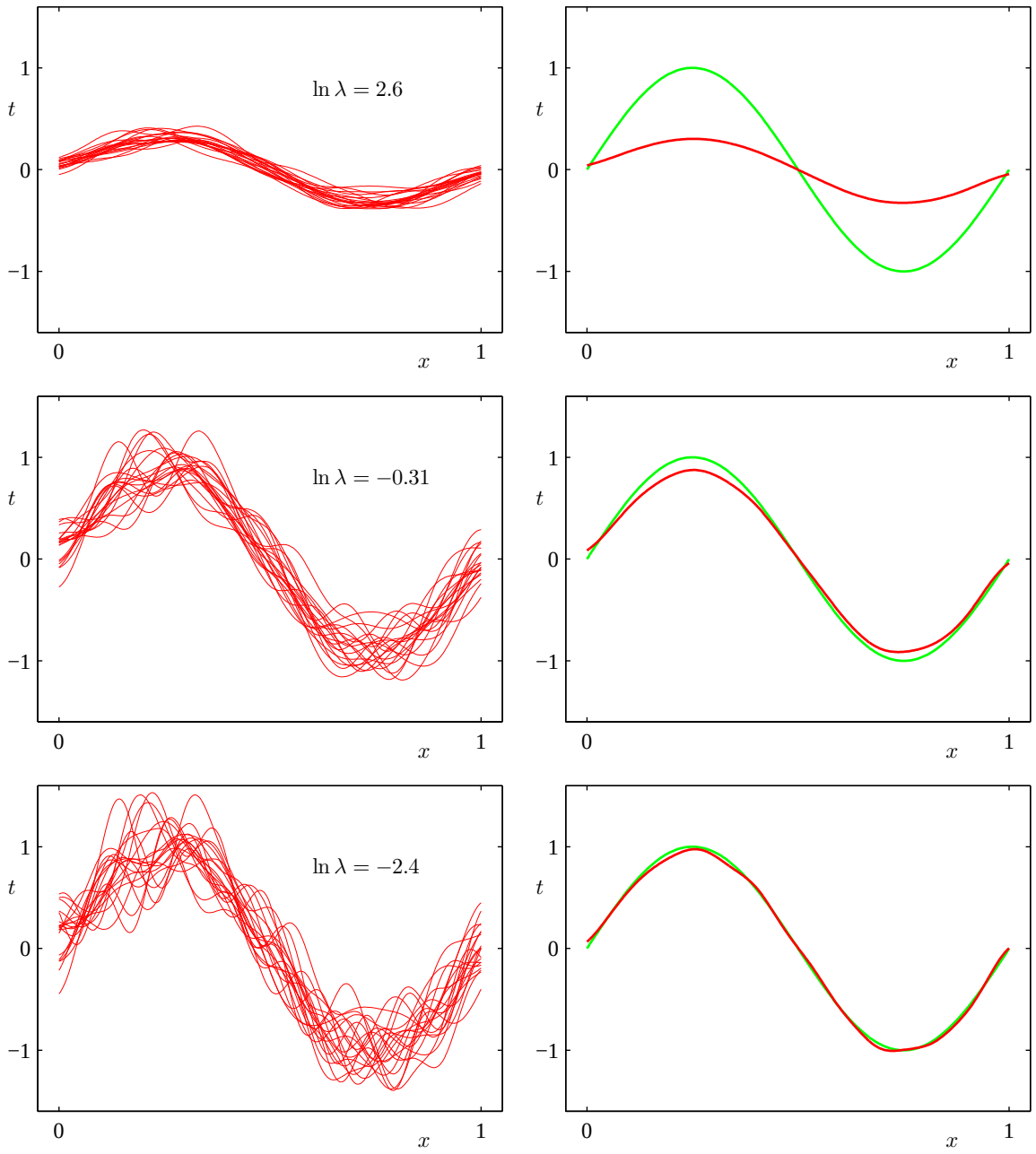


Figure 5: Example of bias-variance trade-off from the book.