Where $\mathbf{1}$ is a unit matrix. Solving for $\mathbf{w}$ gives the optimal $\mathbf{w}$. Since $\mathbf{X}$ is an $N \times (d+1)$ matrix, $\mathbf{X}^\top\mathbf{X}$ is a $(d+1) \times (d+1)$ square matrix. Thus the solution to equation (6) is given by

$$\mathbf{w} = \left(\mathbf{X}^\top\mathbf{X} + \alpha\mathbf{1}\right)^{-1}\mathbf{X}^\top\mathbf{t}. \tag{7}$$

The generalization error is defined as the expectation

$$E_G(\mathbf{w}) = \int\int \{y(\mathbf{x};\mathbf{w}) - t\}^2 p(t|\mathbf{x})p(\mathbf{x})dtd\mathbf{x} \tag{8}$$

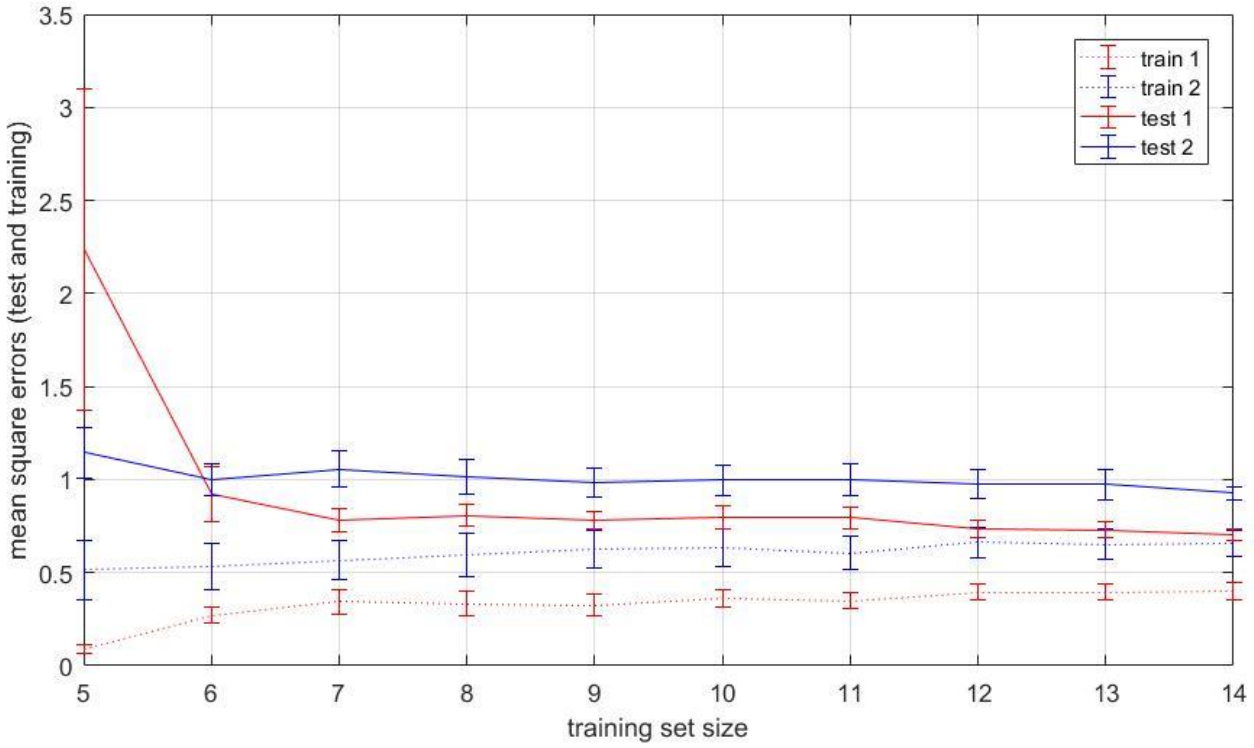
$$\approx \frac{1}{M}\sum_{m=1}^{M}\left\{\mathbf{w}^\top\mathbf{x}_m - t_m\right\}^2 \tag{9}$$

approximated by the mean value over a large *test set* consisting of M examples drawn independently from the N examples in the training set.

Checkpoint 4.1:

Use the program main4a.m to create a training-set with a 2-dimensional input variable and a 1-dimensional output variable. Evaluate the training and test errors on independent sets generated by the same true weight vector and the same noise variance, for two models: A linear model with both inputs and a linear model with only one input variable. Note: In this checkpoint the weight decay is set to zero. Compare the training and test errors *per example* as function of the size of the training set for the two models. Compare their values of the training and test errors for large training sets with the value of the noise variance.

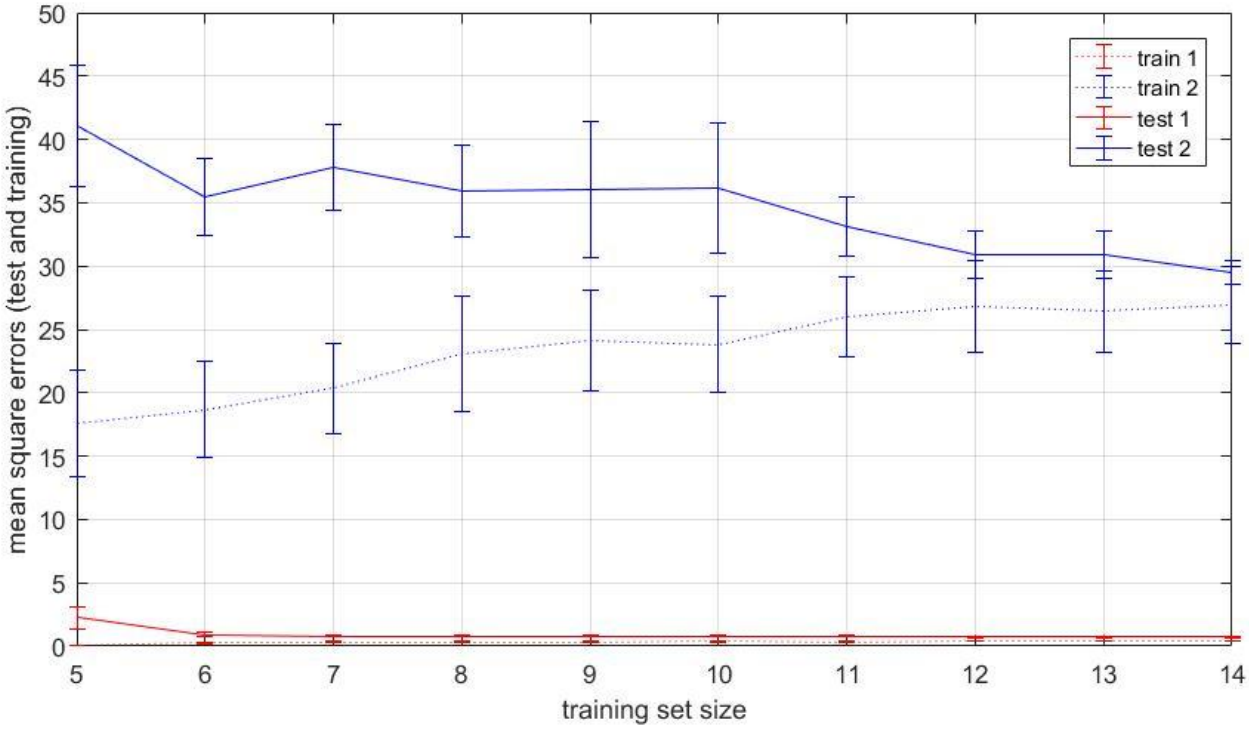
In order to evaluate the training and test errors, in the following plot we can see them plotted for both models and different training sets sizes, being the blue the model the one dimensional model and the red one the two dimensional:



From this plot, we can get a lot of information:

- The one dimensional model performs better with a small training set: this behaviour is expected, because the higher the number of parameters to train the more data you will need to train them. With a one dimensional model what we are getting is a more generalized model, more suitable for small trainings set. This idea will apply to all our experiments in the course, where we will have to select the parameters of our model accordingly.

However, it is not always recomendable to get a more generalized model for smalls training set, as we will see in the following image.



For this graph we choose a right parameter of w3=5 instead of the previous w3=0.5. This causes our one dimensional model to fail when tring to model a 2 dimensional distribution. The explaination behind it is as follows:

The case where the number of model parameters is equal to numbe of real parameters (inputs) was covered in the previous example. In that case we proved that the expected value will be converging to the variance of the nois esignal. In case where the model is smaller this is not a case as we additionally get a systematic error that exsists because the model doesn't have enough parameters. Below are the calculations that allow us to determine both errors:

$$E\left\{((w_1x_1) - (w_{o1}x_1 + w_{02}x_2 + \varepsilon))^2\right\}$$
$$E\left\{(w_{02}x_2 + \varepsilon)^2\right\}$$
$$E\left\{\left({w_{02}}^2{x_2}^2 + \varepsilon^2 + 2\varepsilon w_{02}x_2\right)\right\}$$

The expected value of a product of uncorrelated variables is a product of their expected values. This is why 2*epsilon*w02*x2 is 0 (because they are uncorrelated and mean of epsilon is 0). If they were correlated the noise would be taken into the model.

$$\delta^2 + E\left\{{w_{02}}^2{x_2}^2\right\}$$
$${w_{02}}^2E\left\{{x_2}^2\right\} + \delta^2$$

x2 has mean of 0 and variance of 1 this is why expected value of x2^2 is 1. This gives us:

$${w_{02}}^2 + \delta^2$$

This is the systematic error and the error coming from the noise. This is why when n->inf the smaller model's error is higher than the bigger model's error. Additionally what we see is that the test error is bigger than the training error which is natural as the model was trained for the training set.

As we can see, the estimation error is not based only on the variance anymore, and the w02 parameter influences also this error greatly when it is bigger than 1.

The weights can be found using equation (7), and the predicted value, x_{n+1}, can be found from

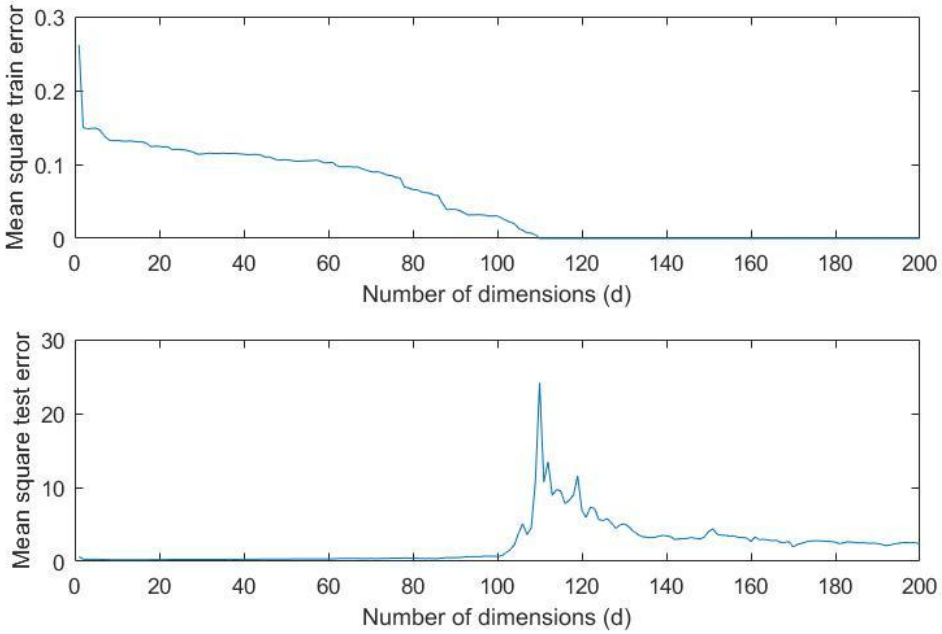
$$x_n = y(\mathbf{x}_n) = \mathbf{w}^\top \mathbf{x}_n. \tag{13}$$

In the context of sunspot time series prediction, the data set from 1700-1920 is used for training while the data from 1921-1979 is used to test performance.

Checkpoint 4.2:

Use the program main4b.m to perform a time series prediction of the number of sunspots with the data from 1700-1920 as training set. Evaluate the test error on the set 1921-1979. Normalize the test error per example by the total variance of the sunspot series. Study the test error as function of the number of weights, d, (hence years) included in the model. Which value of d do you recommend?

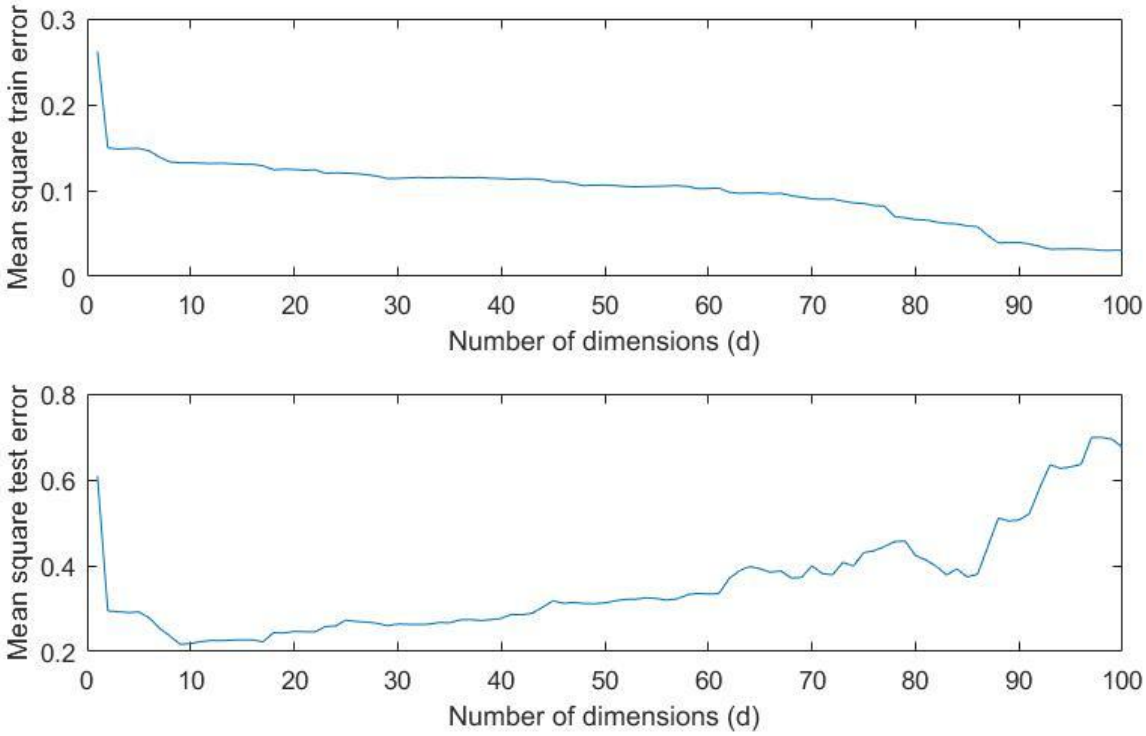
In order to study the dependence of the errors on the nummber of weights we have plotted both the train and test error for different values of d.



The first graph, the one that represents the train error is already familiar, as we show it on the previous exercise. We can see that once the dimensions reach the number of training points the train error becomes zero, but this also means that we are overfitting our model, as we can see from the test error.

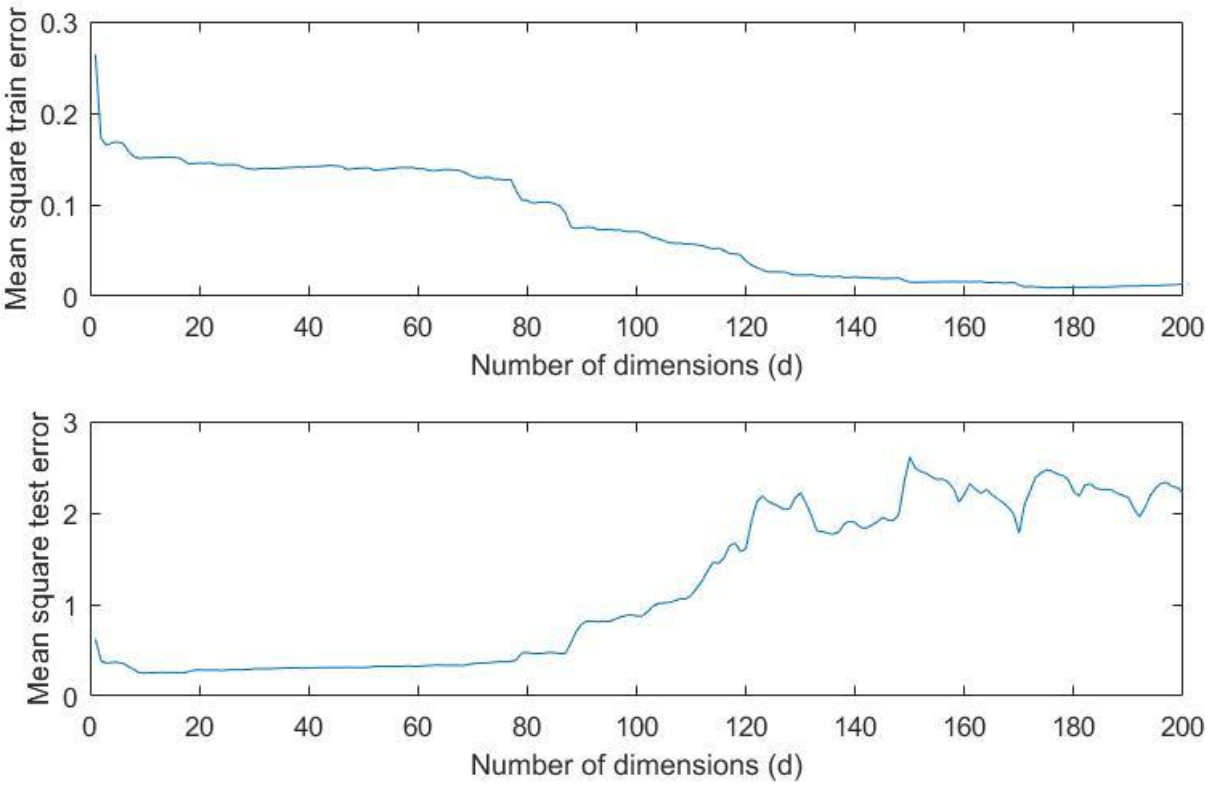
In the moment when d=nPoints there is a huge peak on the test error. This happens because the model is not general at all, with 0DoF. That means that the model also fits the noise of the dataset, and that's why the test error is so big. After this point, you start to add more DoF to your model, making the model more general.

If we want to choose the best number of dimensions for this current dataset we can take a look at the following graph, where we have limited the d so that we have similar scales.



Based on this, we can see that d=9 is the number that minimizes the test error.

Finally, and as an introduction for the next exercise, we can take a look at how to avoid that huge spike in the test error. If we want to achieve that we have to introduce some kind of weight decay, term that we will explain later. With a weight decay of 0.5 we get a graph that looks like this:



As we can see, the test error does not reach the huge values that we experienced before, even though the error is still bad when we reach the point when d=nPoints.

Bias-variance trade-off

The training set averages generalization error in the point $\mathbf{x}$ can be rewritten,

$$\begin{aligned}\mathbb{E}_{\mathcal{D}}\left[(y(\mathbf{x})-\mathbb{E}_t[t|\mathbf{x}])^2\right] &= \mathbb{E}_{\mathcal{D}}\left[\{y(\mathbf{x})-\mathbb{E}_{\mathcal{D}}[y(\mathbf{x})]\}^2\right] \\ &+ \{\mathbb{E}_{\mathcal{D}}[y(\mathbf{x})]-\mathbb{E}_t[t|\mathbf{x}]\}^2.\end{aligned}$$

Where $\mathbb{E}_{\mathcal{D}}$ is the expectation with respect to training sets. Note $y(\mathbf{x}) = y(\mathbf{x};\mathbf{w}(\mathcal{D}))$.

Hence the average error is split into a *variance* part, quantifying the variation among solutions for different training sets and a *bias* part quantifying the performance of the average model with respect to best possible model $\mathbb{E}_t[t|\mathbf{x}] = \int tp(t|\mathbf{x})dt$ (the conditional mean of the output given the input).

Checkpoint 4.3:

Use the program main4c.m to measure the relative amount of variance and bias for a linear model as in checkpoint 4.1 with two inputs and controlled by weight decay. Plot the average generalization error, the bias error, and the variance error for a large range of weight decay values. Comment on the two regimes where the generalization error stems from variance and bias respectively. What is the role of the weight decay in these two regimes. Which weight decay value would you recommend?

1.1 Conceptual Definition

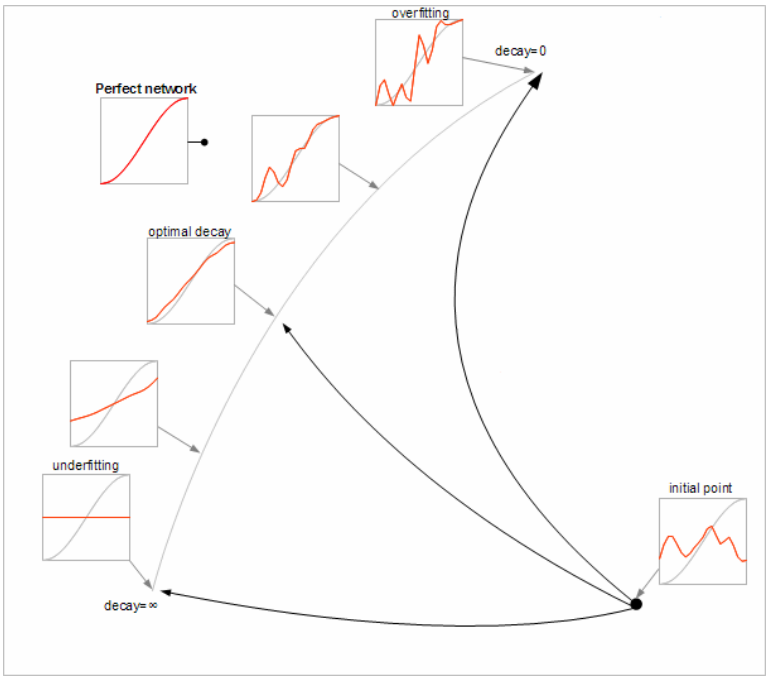
- **Error due to Bias**: The error due to bias is taken as the difference between the expected (or average) prediction of our model and the correct value which we are trying to predict. Of course you only have one model so talking about expected or average prediction values might seem a little strange. However, imagine you could repeat the whole model building process more than once: each time you gather new data and run a new analysis creating a new model. Due to randomness in the underlying data sets, the resulting models will have a range of predictions. Bias measures how far off in general these models' predictions are from the correct value.
- **Error due to Variance**: The error due to variance is taken as the variability of a model prediction for a given data point. Again, imagine you can repeat the entire model building process multiple times. The variance is how much the predictions for a given point vary between different realizations of the model.

From the expectation with respect to datasets we can see how we have two different parts, variance and squared bias part. It is important to point out that this is all based on the best posible model, and not in the real model. If we would be working with the real model we will have to include an additional source of error, an irreducible one, defined as follows:

$$\text{noise} = \int \{h(\mathbf{x})-t\}^2 p(\mathbf{x},t)\,\mathrm{d}\mathbf{x}\,\mathrm{d}t$$

with h(x) being the best possible model. In this exercise we will exclude this additional source of noise and we will focus on the influence of the variance and bias.

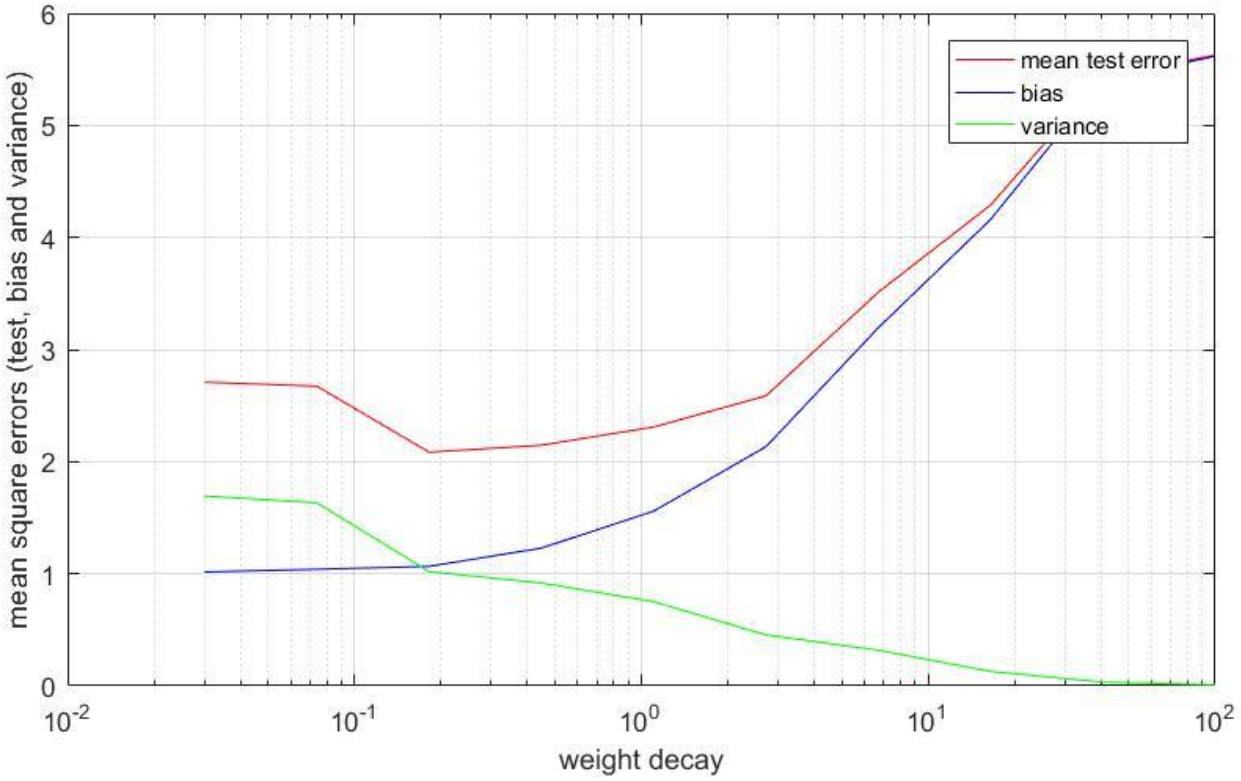
Next thing we need to understand is the influence of weight decay. For this, we can take a look at the following image



In this image we can see the response of a ML model (NN) to different weight decays. Without weight decay we risk to overfit our model, and at this point the model will try to fit also the noise present in each measurement, making it difficult to generalize. One way of solving this problem is to introduce this weight decay term. This term penalizes large weight values, which are not desirable, as those are the ones allowing for fitting the noise (quick changes mean that the weights have to be large, as otherwise it is not possible to produce those abrupt changes in the data).

However, when choosing this weight decay, you also risk to underfit your model, causing your model to keep your weights zero, causing a huge bias error, and zero variance error.

All of this can be observed from the graph out of the exercise 3:



On the left side, the weight decay is too small, which causes the mean square error out of the variance to increase. In the right side, the weights tend to zero, which causes the mean square error to increase due to the bias error. In this case, we should choose a weight decay of around 0.1, that is when the combined mean test error is minimized.