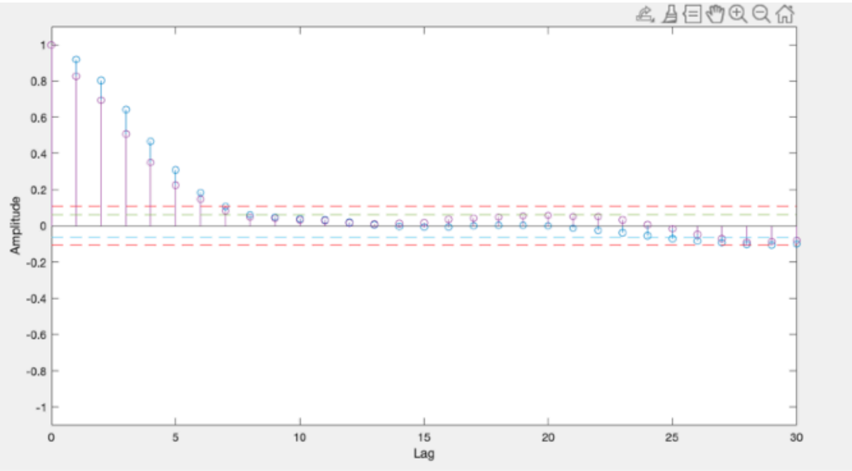
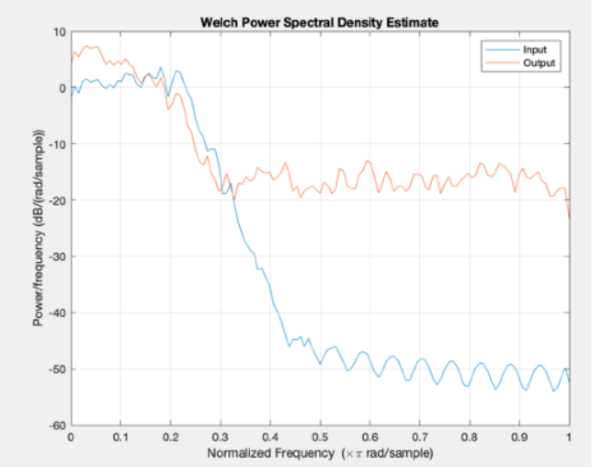
Task 2:
Task: find a good model for the SISO-system governing the given data y and u.
We begin by eyeballing the data to see if there are any outliers present. I cannot spot any clear outliers. We then proceed to do a split of modelling and validation data. This is to choose models that perform well on unseen data, in order to get a hunch of how well our model will generalise. I use the first 70% of the data for modelling and the last 30% for validation.

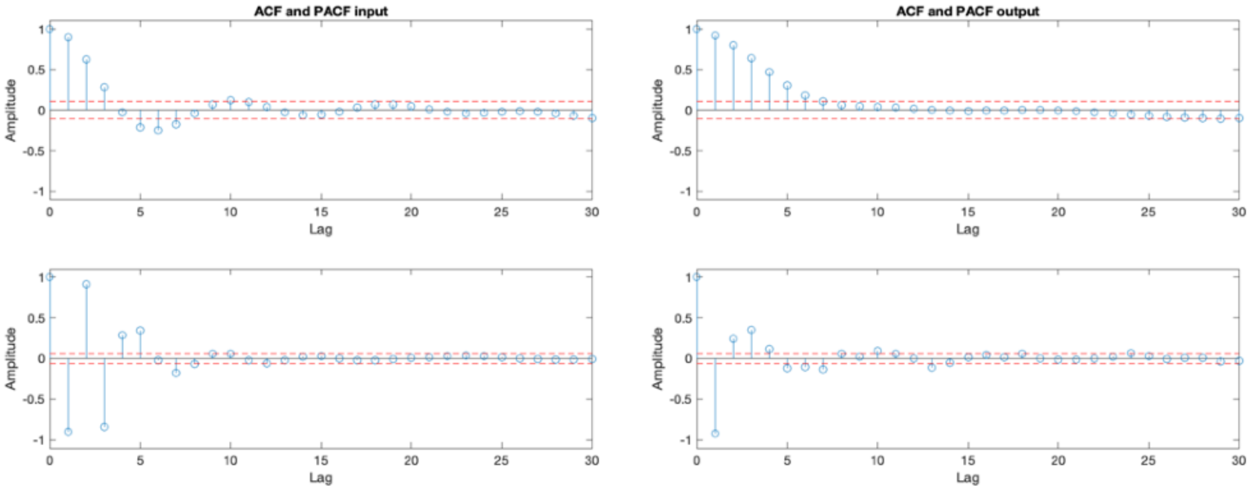
We then examine the trimmed autocorrelation function (tacf) which removes the 2% largest and smallest values and then examines the autocorrelation is a way of quantitatively find outliers. A plot of these compared to the ordinary autocorrelation function shows that the behaviour is similar, further strengthening the hypotheses that no outlier handling is necessary. (Some differences are expected, we just do not want a different characteristic).

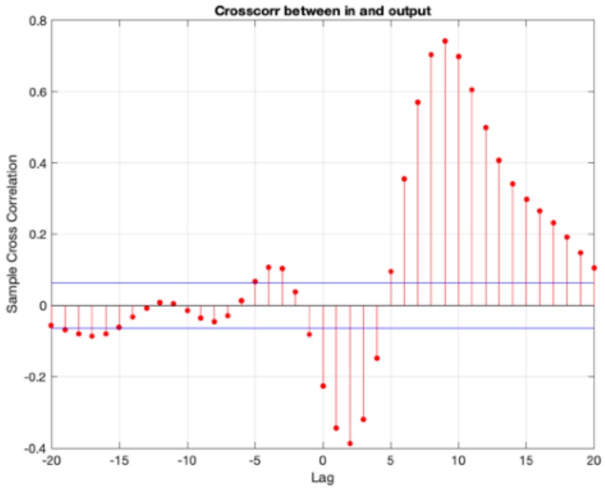


The spectrum of the input and output shows that the system is excited to about 20% of the Nyquist frequency.



We then examine the ACF of both input and output, as well as the correlation between input and output, to find any dependencies. These show that the input and output both has some ARMA-dependencies. The crosscorrelation show that we should look for an input/output relation up to lag around ten.



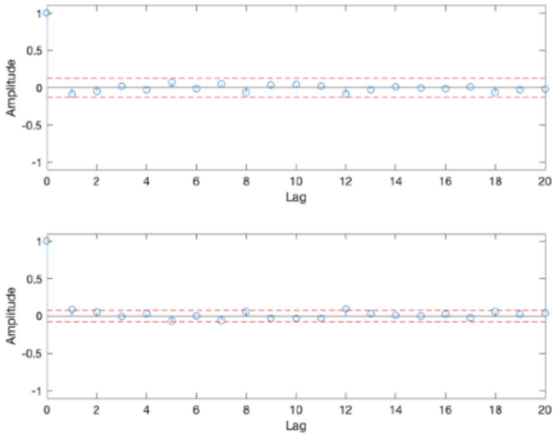


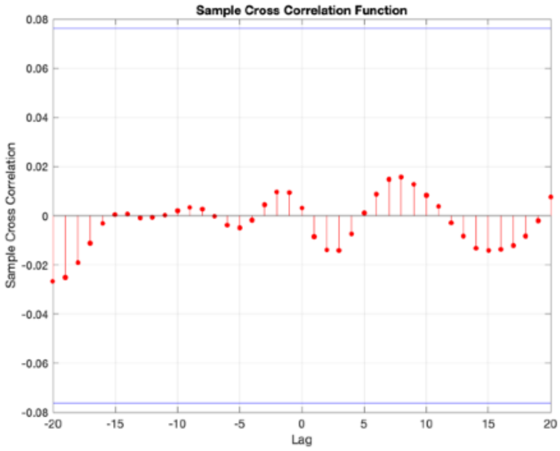
Since the given experiment with ARX of order up to 10 did not generalise well to test data, despite having a good fit on modelling data makes me think we should test for a simpler model, perhaps an OE. The heavy dependency of input for time lag starting at 2 justifies this. I use the toolbox ident to quickly find the best outlines for a model, but then proceed to use the PEM-function to make more tailored fits (see code). ident shows that an OE-model for delay 2 and model order around 4,4 gives a good fit to validation data. I proceed to model using my own algorithm (see code) and find an oe with polynomials:

B(z) = 0.05423 (+/- 0.01204) - 0.2971 (+/- 0.02901) z^-1 + 0.2836 (+/- 0.01841) z^-2

F(z) = 1 - 2.228 (+/- 0.02027) z^-1 + 1.675 (+/- 0.03569) z^-2 - 0.4273 (+/- 0.01612) z^-3

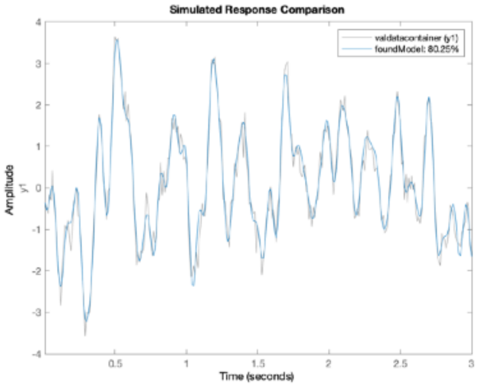
The the ACF of the residuals and the crosscorrelation between input and residuals on modelling data show that we have modelled almost all dependencies:

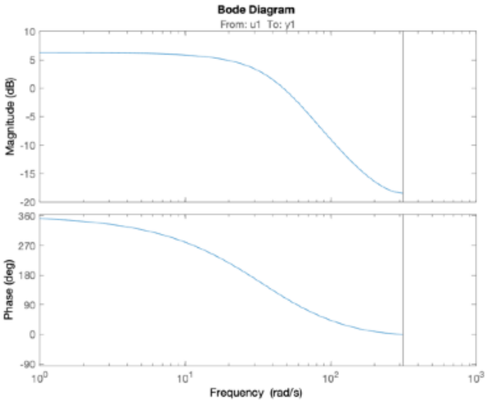




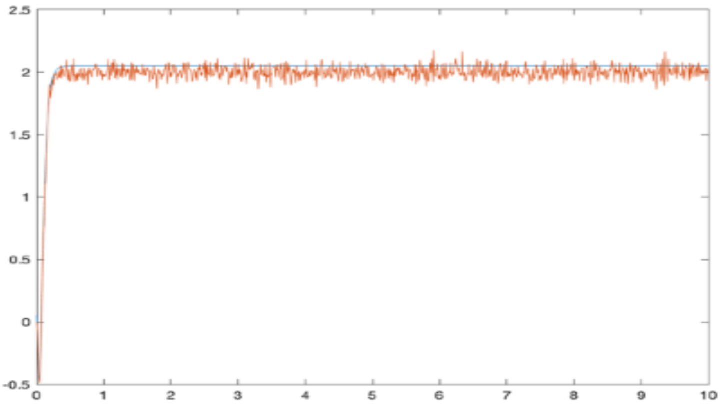
White residuals and no cross correlation.

This model has an 80.25% fit to validation data and not the same high frequency amplification as the example ARX10 had. See plots below

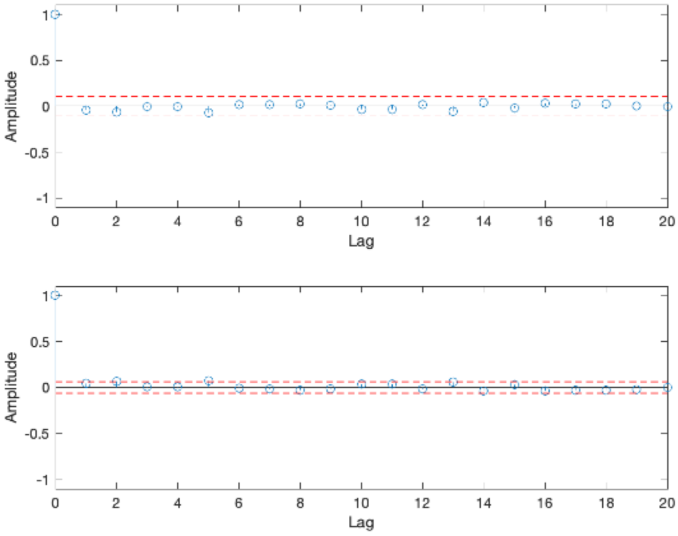




Now we move over to the test data, to se how well it performs:



The model seems quite successful! The residuals, shown below, are clearly white showing that we have modelled almost all dependencies:



Task 3
a) The ideal features to use are those who are as uncorrelated with each other as possible, but highly correlated with the target, and most importantly: we want to have a clear separation between the target population in the dimensions of the features.
From the scatter plots given I would draw the same conclusion that the example code writer, that features 2, 3 and 5 would make a good model However, feature 3 gives good decision boundary together with most features, so we will this one in combination with the others!

By using the same features and a standard scaler to make all features have zero mean and unit variance, we manage to get the accuracy to 0.85. Important: the scaler is fitted on training data and then performs the same operation on new data, meaning that the mean and variance of the training data is used to normalise validation and test data. This prevents data leakage.

By using features 2, 3 and 6 we get 0.857 accuracy on validation data and 0.885 on test data. This is satisfactory.

b) Since a random tree works such that it begins to in each node split the data as good as possible. Each node will do the split that maximises the reduction of for example the mean squared error. Therefore, if we want to have for example tre features, we can do three splits and see what features it uses to do these splits. In most machine learning libraries, the methods for a random forest has methods to extract the feature importance. We can see that the most important features were not the ones we used, apart from 6. However, testing the features that are presented as best from the random forest (1, 4 and 6) gives the same performance as our model. This could have to do with that the node splits are made on individual features, and does not consider the combination of three.

c) The ROC curve shows the alternatives we have for FP-rate, that is we can move along this curve without training just by specifying another acceptable probability to have a false positive. It is normally 0.5, that is we want the same probability to predict a false negative as a false positive, however if it is more crucial to not have as little as possible false positives then we can just make this higher. This comes at a cost of higher lower true positives however, since we want to be more certain before we deem a class as positive.

Task 4:
a) My line of code: Ypred(k) = [height(k) leg(k)]*thetahat; I get a RMSE for leave one out of 5.1077.

b) Since there probably is very high correlation between leg length and height, a linear regression on these features could then result in a negative value for one of the parameters since maybe a tall person with long legs maybe have slightly smaller shoe size than a tall person with shorter legs. If we were to estimate using only leg lenght, we would surely get a positive correlation.

c) Gamma is a regularisation parameter, which makes the optimization prefer smaller parameters theta.

d) It seems to be working worse. From the singular value decomposition we see that it is mostly the first parameter that is important, the height. It is better to just use this one, the second probably just ads noise. As we see from the histogram the estimates of theta2 span a very large interval, between -16 and -11 which further strengthens the theory that it is unnecessary and just ads noise.

Task 5:

a) See paper.

b) When we are only regressing on X, we don't take into consideration the dependence from other variables. When we are intervening and making X a fixed value, we break this dependence and therefore get an unexpected result. A regression on X and B, that is we condition on B, also breaks the backdoor dependence from X to C via B (fork/confounder) and then gives the correct estimate of the dependence between X and Y. They are hard to measure and cannot be included in the regression, but it would give a correct estimate.

c) As in b), if we use the data to estimate the relation between Y and X together with B. That is we make an OLS regression on the relationship Y = a1X + a2B.
This gives the correct estimate c6 = 2, which means that an intervention with x = 7.5 would give the correct result. The mean of Y from the data confirms this.