# SYSTEM IDENTIFICATION

# PROJECT SUBMISSION

**BY**

**SHARA RHAGHA WARDHAN B**

**ME11B121**

# QUESTION 1

## PART A

Code has been attached. The function used to estimate the delay is ‘delay_est.m’. The code has been made self-sustaining such that the input required is just the dataset and the delay is given as the output.

Algorithm used is as follows

1. First we evaluate the whole FRF of the system using the spectral analysis(spa) routine and find the phase for all the evaluated frequencies

2. We find the coherence for that corresponding set of frequencies and evaluate the weights

3. Then we find the contribution of the delay free portion by using the Hilbert transform

4. Then we evaluate the cost function for all integer values of delay and find the delay that corresponds to the maximum value of the objective function or the minimum value of the negative of the objective function.

## PART B

The transfer function used in this case is

$$G(z^{-1}) = \frac{4z^{-D}}{1 - 0.3z^{-1}}$$

The delay D is specified in this case as 20. The input used for this case is the full window PRBS. The SNR has been set as 10. This is done by first simulating the noise free process and evaluating the variance of that signal. Then we find $\sigma_e^2$ by using

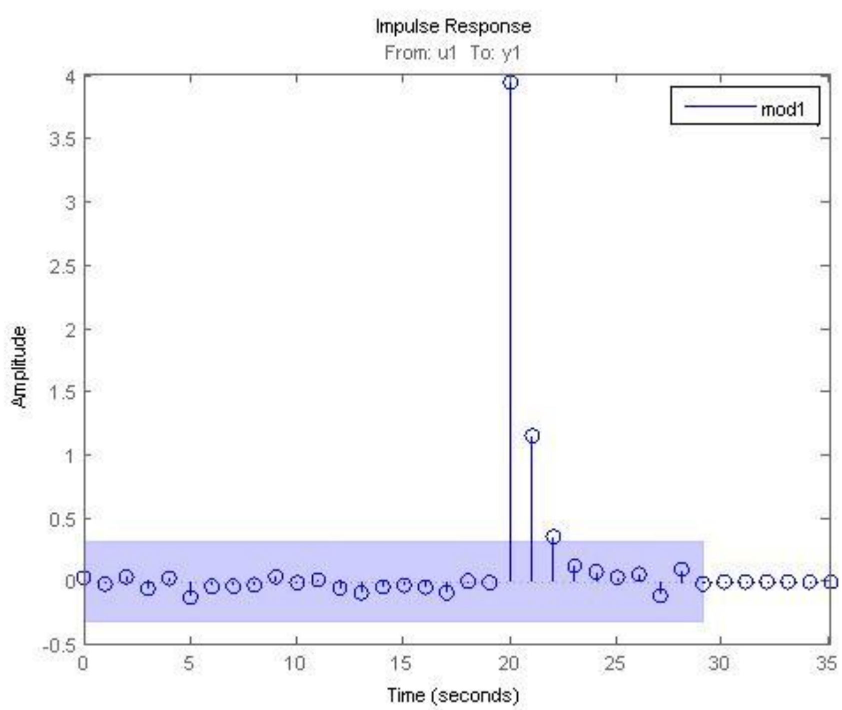
$$SNR = \frac{\sigma_x^2}{\sigma_v^2}$$

But in this case, $\sigma_v^2 = \sigma_e^2$ . Then the process is simulated with noise and 500 data points are generated.

Estimation of Delay by using the function written in Part A (FRF approach)

The delay estimation by using the FRF method reveals that the delay is 20.

Estimation by Impulse Response coefficients




The impulse response coefficients plot reveals that the delay is 20 almost with no ambiguity since all the coefficients before that are almost zero. Hence it matches with the FRF approach.

## PART C

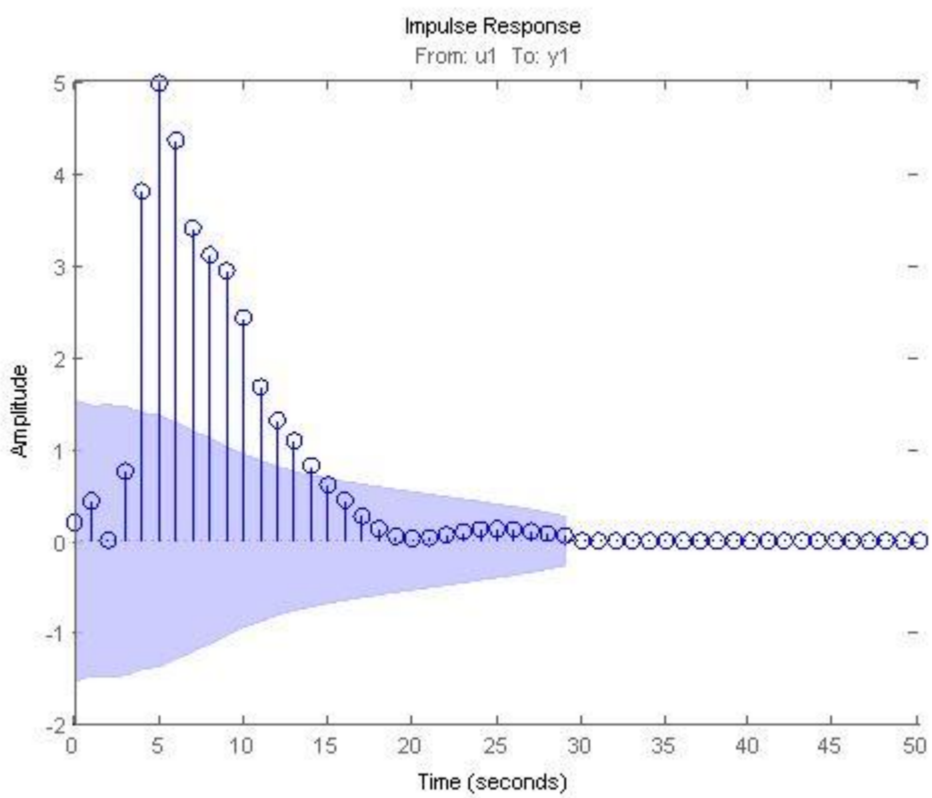
The given data set is mean centered first and then passed into the function written in part A

Estimation of Delay by using the function written in Part A (FRF approach)

The delay estimation by using the FRF method reveals that the delay is 4.

Estimation by Impulse Response coefficients

Here, the important thing to notice is that based on the confidence limit set, the delay could have been different. If no confidence limit was there, the delay would have been estimated to be 1 since a significant response can be seen.




Here we constructed a 95% confidence interval and accepts the null hypothesis until the $4^{th}$ coefficient. The impulse response coefficients plot reveals that the delay is 4. Hence IR coefficient method with 95% confidence limit matches with the FRF approach.

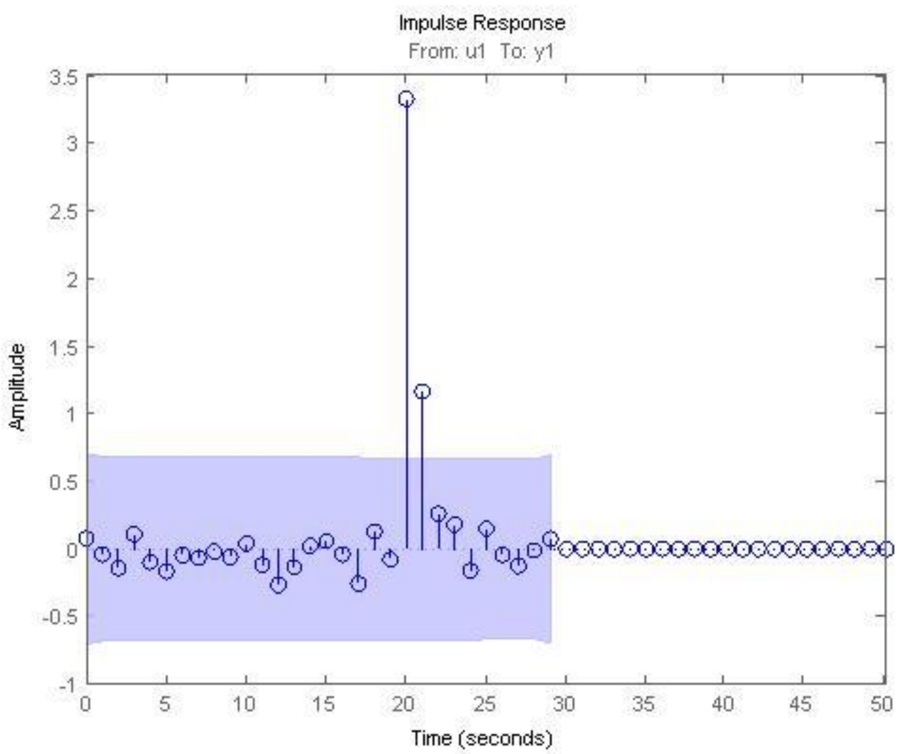
## PART D

The data set has been modified and 10% of the number of data points has been calculated. Then we generate a uniform random number and calibrate it to find the location of the outlier and then a normal random number to set a magnitude for the outlier

<u>Estimation of Delay by using the function written in Part A (FRF approach)</u>

The delay estimation by using the FRF method reveals that the delay is 20.

<u>Estimation by Impulse Response coefficients</u>




Here we constructed a 95% confidence interval and accepts the null hypothesis until the 20$^{th}$ coefficient. The impulse response coefficients plot reveals that the delay is 20. Hence IR coefficient method with 95% confidence limit matches with the FRF approach.

The issue with the IR coefficient method is that based on the confidence limit set and the SNR the delay could be varied and we could end up with a bad estimate of delay as the outliers and noise will add to the error in IR coefficient estimates. Moreover since the method used to evaluate the IR coefficient is by least squares, it is necessary that the residuals are white otherwise the normal distribution to verify the null hypothesis of the IR coefficients is not valid. But the FRF approach is more robust to outliers as it uses an optimization algorithm and gives a unique answer without any ambiguity.

## QUESTION 2

### PART A

Formulating the problem for an ARX model:

First, we have to determine the maximum order of the model that we want to fit. Say we want to fit an $M^{th}$ order model.

$$y[k] = \frac{B(q^{-1})}{A(q^{-1})}u[k] + \frac{1}{A(q^{-1})}e[k]$$

$$\Rightarrow y[k] = \frac{b_0 + b_1 q^{-1} + \cdots b_M q^{-M}}{1 + a_1 q^{-1} + \cdots a_M q^{-M}}u[k] + \frac{1}{1 + a_1 q^{-1} + \cdots a_M q^{-M}}e[k]$$

$$y[k] = (-a_1 y[k-1] - a_2 y[k-2] \ldots - a_M y[k-M]) + (b_0 u[k] + b_1 u[k-1] \ldots b_M u[k-M]) + e[k]$$

$$\Rightarrow y[k] = [-y[k-1] \quad -y[k-2] \quad \ldots \quad -y[k-M] \quad u[k] \quad u[k-1] \quad \ldots \quad u[k-M]]\begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_M \\ b_0 \\ b_1 \\ \vdots \\ b_M \end{bmatrix} + e[k]$$

Here the first matrix is called the regressor matrix or the basis matrix on which we want to project our data onto and the power of each basis is given by the parameters which are contained in the second matrix. The extent of sparsity comes in the coefficient matrix. The prime requirement is that the number of parameters should be very high compared to the number of data points. So all the data points are collected and represented as a single entity. The predictor will have the same equation with the exception of e[k].

$$\begin{bmatrix} y[k] \\ y[k-1] \\ \vdots \\ y[k-N] \end{bmatrix} = \begin{bmatrix} -y[k-1] & -y[k-2] & \ldots & -y[k-M+0] & u[k] & u[k-1] & \ldots & u[k-M] \\ -y[k-2] & -y[k-3] & \ldots & -y[k-M-1] & u[k-1] & u[k-2] & \ldots & u[k-M-1] \\ & & & \vdots & & & & \\ -y[k-N-1] & -y[k-N-2] & \ldots & -y[k-M-N] & u[k-N] & u[k-1-N] & \ldots & u[k-M-N] \end{bmatrix}\begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_M \\ b_0 \\ b_1 \\ \vdots \\ b_M \end{bmatrix}$$

$$\Rightarrow Y = AX$$

Brining the model to the optimizing framework, we get

$$\underbrace{min}_{X} \ J = \sum_{i=1}^{M} |a_i| + \sum_{i=0}^{M} |b_i|$$

$$subject\ to \left\| \begin{bmatrix} y[k] \\ \vdots \\ y[k-N] \end{bmatrix} - \begin{bmatrix} -y[k-1] & -y[k-2] & \cdots & u[k-M] \\ \vdots & \vdots & \ddots & \vdots \\ -y[k-N-1] & -y[k-N-2] & \cdots & u[k-M-N] \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_M \\ b_0 \\ b_1 \\ \vdots \\ b_M \end{bmatrix} \right\|_2^2 \leq \epsilon$$

## PART B

Steps followed in the algorithm

1. The A matrix and the Y vector are constructed

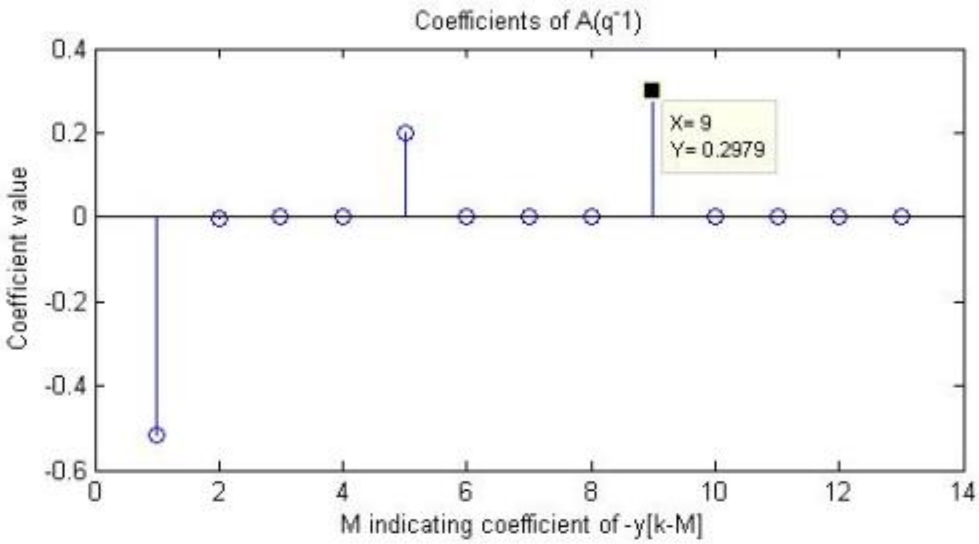
2. We solve an unconstrained optimization problem with the same objective function as above to generate a good initial guess of the parameters

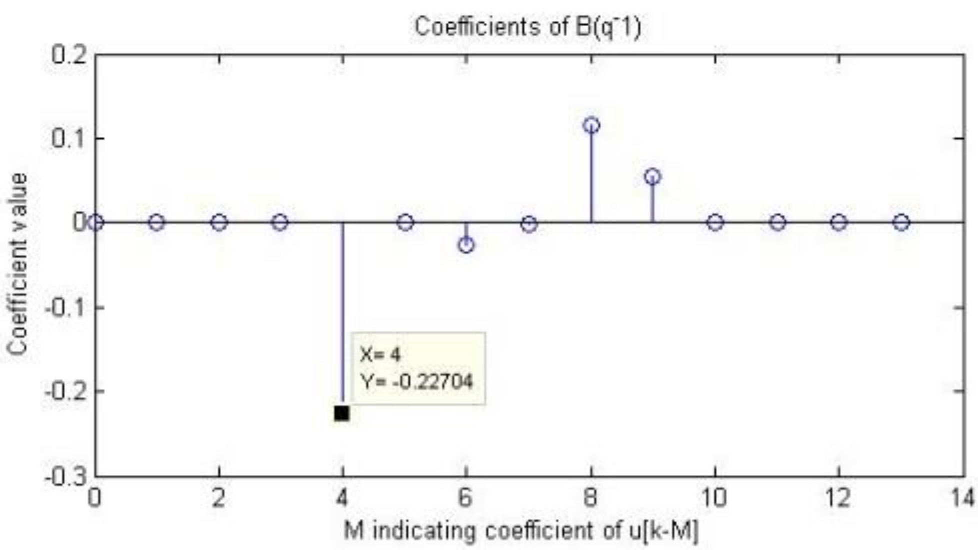
3. We use a modification of the objective function to account for the sparsity of the parameters to a large extent depending on the magnitude of their value
$$X_i = max\left(0,1 - \frac{\tau}{max(abs(X_i), 10^{-10})}\right) * X_i$$
This is done and then the objective function is evaluated. The above just accounts for that if the value of the parameter is insignificant, it is made zero. $\tau$ is a tweaking parameter to vary the threshold. In this case $\tau = 0.01$. The threshold $\epsilon = 0.01$

4. Then the stem plot of the parameters is displayed and the appropriate order is chosen from the user.

## PART C

We first run the function in Part B on the data set to get the estimates of the coefficients.



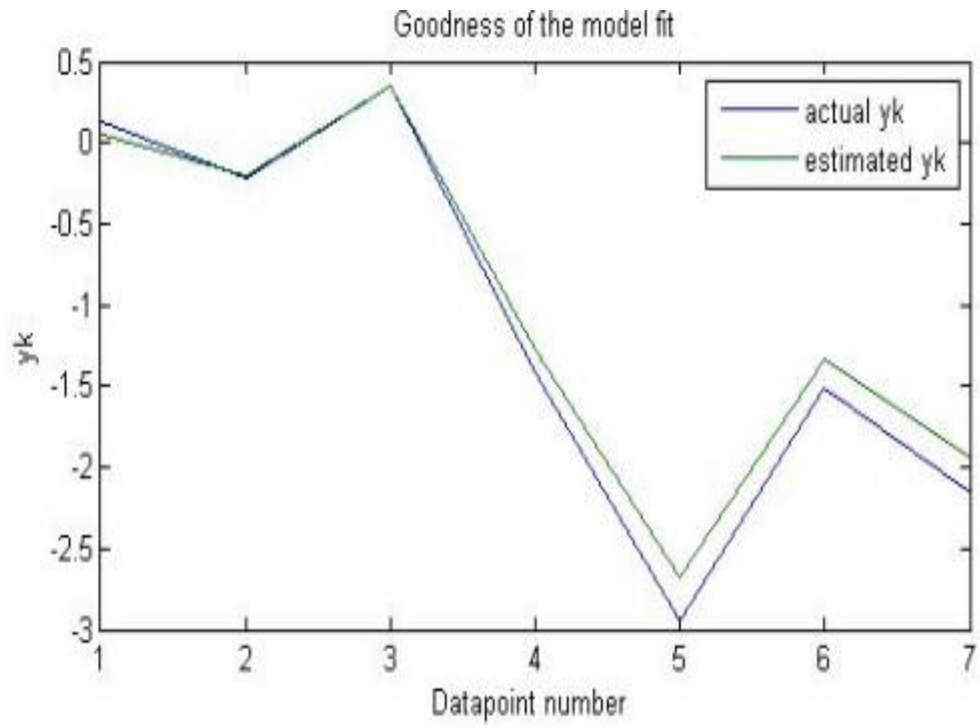




From the above plot it can be estimated that the minimum significant parameter of B is 4. Hence the delay is 4.

The maximum significant parameter of A is 9. Hence the order is 9.

$$Delay = 4$$

$$Order = 9$$




With the above estimates, the fit to the actual data seems pretty good and hence we proceed further and try to build an arx model with the significant coefficients in the previous figures.

But since the order of B should be less than the order of A, we restrain ourselves to $4^{th}$ parameter alone in B.

Discrete-time ARX model: A(z)y(t) = B(z)u(t) + e(t)

A(z) = 1 - 0.5521 (+/- 0.6172) z^-1 + 0.1355 (+/- 0.8573) z^-5 + 0.4429 (+/- 0.7568) z^-9

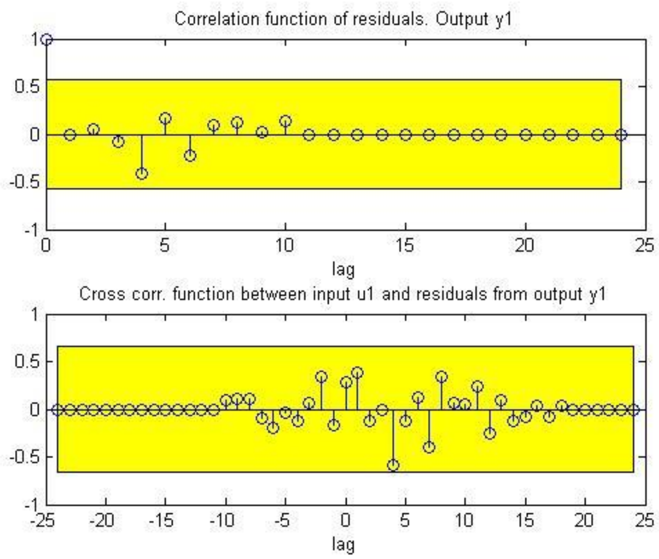
B(z) = -0.3605 (+/- 0.9387) z^-3

$$y[k] = 0.5521y[k-1] - 0.1355y[k-5] - 0.4429y[k-9] - 0.3605u[k-3]$$

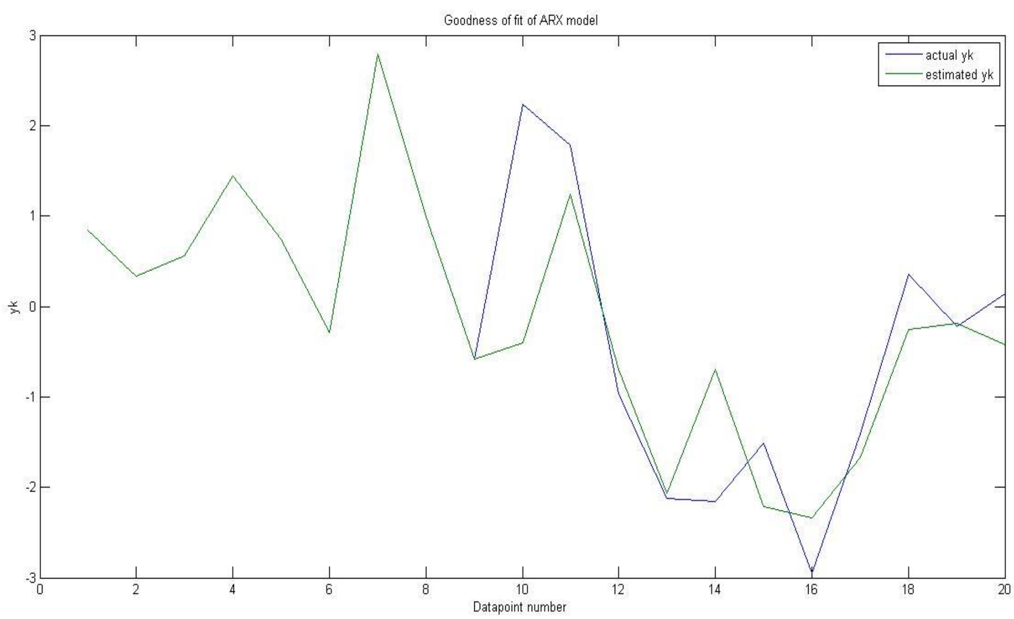
Fit to estimation data: 50.14% (prediction focus)

FPE: 1.104, MSE: 0.5812

The parameters have not passed the 95% confidence test and all parameters could be rejected by null hypothesis. But this could be due to the lack of many data points. Here our estimation is done with 20 data points and hence obviously it cannot yield a very good estimate.




The residual analysis reveals that the model has passed the whiteness test.




The fit to the actual data is pretty good. Hence, even though the model isn't good, it gives a good idea of the order and the delay.

# THE END