ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

CE-2 : PARAMETRIC IDENTIFICATION METHODS

SYSTEM IDENTIFICATION ME-421

PROF. ALIREZA KARIMI

Angelo Giovine 368440

Hana Catic 370754

14th August 2024

Contents

CHAPTER 1

IDENTIFICATION OF A DC SERVOMOTOR: FIR MODEL IDENTIFICATION

The first parametric identification technique that we are going to use on our DC servomotor is Finite Impulse Response (FIR). We assume that the output of the system is a function only of the past inputs, hence the best predictor, assuming 1 time instant of delay ($d = 1$) would be:

$$\hat{y}(k, \theta) = b_1 u(k-1) + \cdots + b_m u(k-m), \tag{1.1}$$

$$\theta^{\mathsf{T}} := [b_1 \ \cdots \ b_m] \tag{1.2}$$

where m is the length of our impulse response and θ^{T} is the vector of parameters in which we are interested. So our parameters are exactly the first m values of the impulse response of the system.

1.1 $\hat{\theta}$ ESTIMATION

The vector of estimated parameters $\hat{\theta}$, can be computed using the least squares algorithm, hence:

$$\hat{\theta} = (\Phi^{\mathsf{T}}\Phi)^{-1}\Phi^{\mathsf{T}}Y \tag{1.3}$$

Where Φ, is a Toeplitz matrix containing the past inputs and Y is a vector containing the measured output of our system.
Setting $m = 200$, here the code to compute $\hat{\theta}$:

LISTING 1.1
Theta FIR

```
%% FIR, estimate parameters
% assumption that delay d = 1, m = 200
d = 1;
m = 200;
k = 1;
N = length(y);
Y=y;
```

```
r = zeros(m-d+1,1);
r(k) = u(k-d+1);
c = u(k-d+1:end);
Phi = toeplitz(c,r);
theta_hat = inv(Phi'*Phi)*Phi'*Y;
```

1.2 PREDICTOR

Now that we have estimate $\hat{\theta}$, it's trivial to compute the predictor:

$$\hat{Y} = \Phi\hat{\theta} \tag{1.4}$$

now we can compute the loss function $J(\hat{\theta})$ as the 2-norm of the difference between y and $\hat{y}$, hence:

$$J(\hat{\theta}) = \sum_{k=1}^{N}(y(k) - \hat{y}(k,\hat{\theta}))^2 \tag{1.5}$$

Here is the code for computing the desired quantities:

LISTING 1.2
Prediction FIR

```
%% Prediction
y_hat = Phi*theta_hat;
Critierion = norm(y_hat - y);
loss = (1/ length(y_hat))*Critierion;
```

In Figure 1.1, we can observe that the predictor is quite accurate, but further analysis on the quality of the estimates are provided in Section 1.3. The value of the loss function is 0.0195.

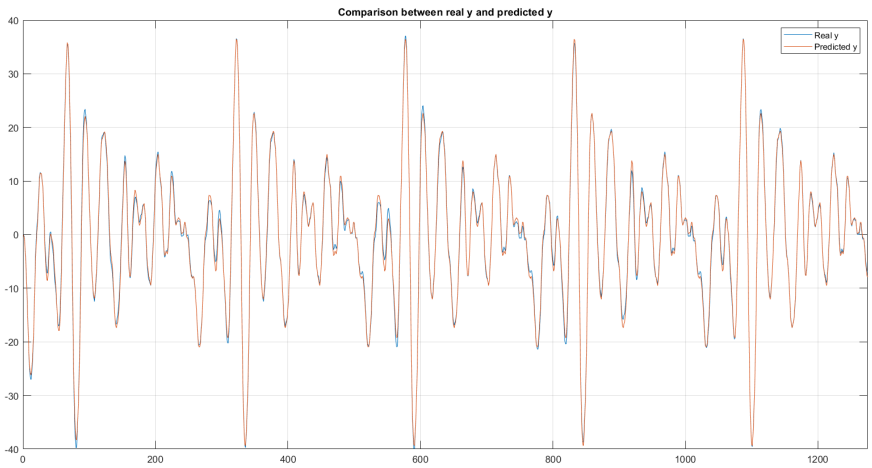


FIGURE 1.1
Comparison between y and the predictor

1.3 QUALITY OF THE ESTIMATES

To analyze the quality of the estimates, we assume that the system is subject to white noise. We are interested in estimating the noise variance and computing the covariance of the estimated parameters. For the noise variance, an unbiased estimate of σ^2 is given by:

$$\hat{\sigma}^2 = \frac{J(\hat{\theta})}{N - m} \tag{1.6}$$

Now that we have an estimate of the noise variance we can compute the covariance of $\hat{\theta}$ as follows:

$$\text{cov}(\hat{\theta}) = \hat{\sigma}^2 \left(\Phi^\mathsf{T}\Phi\right)^{-1} \tag{1.7}$$

Now by taking the diagonal of the obtained matrix, we have an estimate of the variance of the parameters that we can use to compute a confidence interval of 2 standard deviations to plot the impulse response seen in Figure 1.2.

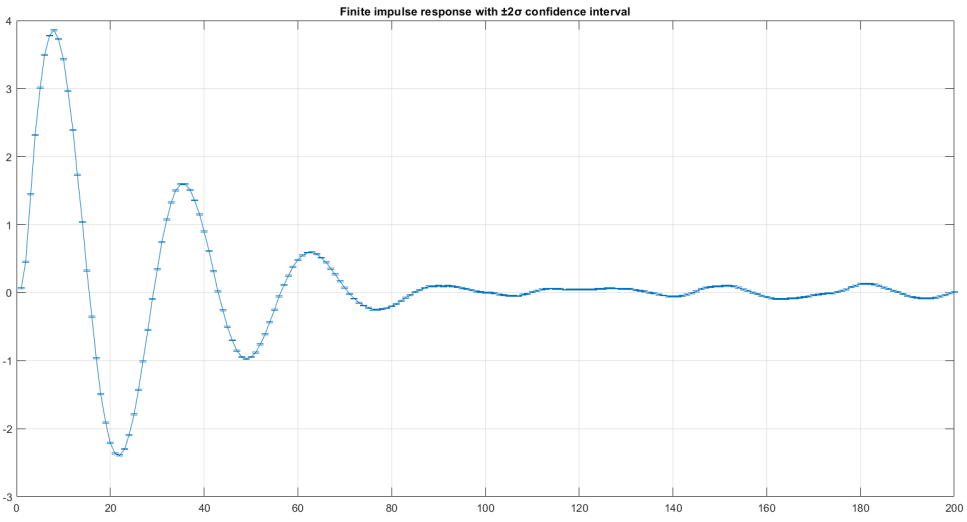


FIGURE 1.2
FIR plot with confidence interval

Here is the code for the analysis of the quality of the estimates:

LISTING 1.3
Confidence interval FIR

```
% Computing the variance
var_hat = Critierion / (N - m);
cov = var_hat *inv(Phi'*Phi);
var = diag(cov);
std = sqrt(var);
% Plotting
```

```
figure
errorbar(theta_hat', 2*std)
```

CHAPTER 2

IDENTIFICATION OF A DC SERVOMOTOR: ARX MODEL IDENTIFICATION

Unlike the FIR method, which was previously used for model identification, the ARX model structure includes the following noise model:

$$H_0(q^{-1}) = \frac{1}{A_0(q^{-1})}. \tag{2.1}$$

Taking into consideration the parametrised predictor, where $\epsilon(k,\theta)$ is the prediction error, we obtain:

$$\hat{y}(k,\theta) = G(q^{-1})u(k) + [\frac{1}{A_0(q^{-1})} - 1]\epsilon(k,\theta). \tag{2.2}$$

In this section we are assuming a second order ARX model for the data with the following predictor:

$$\hat{y}(k,\theta) = -a_1 y(k-1) - a_2 y(k-2) + b_1 u(k-1) + b_2 u(k-2), \qquad \theta^T = [a_1, a_2, b_1, b_2]. \tag{2.3}$$

2.1 ESTIMATION OF PARAMETERS $\hat{\theta}$

Parameters $\hat{\theta}$ are computed using the least squares algorithm, defined as follows:

$$\hat{\theta} = (\Phi^T\Phi)^{-1}\Phi^T Y, \tag{2.4}$$

where $\hat{\theta}$ as previously defined is the parameter vector, Y is the measurement vector and Φ is the observation matrix, containing past inputs and outputs of the system in the following form:

$$\Phi^T = [-y(k-1), ..., -y(k-n), u(k-d), ..., u(k-m)]. \tag{2.5}$$

The code to execute the estimation of the parameters is given below in listing 2.1.

LISTING 2.1
Estimation of parameters $\hat{\theta}$

```
%% ARX, estimate parameters discarding the first 2 samples
n = 2;
m = 2;
k = 1;
N = length(u);
```

```
Phi = [-y(2:end-1),-y(1:end-2),u(2:end-1),u(1:end-2)];
Y = y(3:end);
theta_hat = inv(Phi'*Phi)*Phi'*Y;
```

Estimated parameters are:

$$\theta^T = [-1.8821 \quad 0.9329 \quad 0.3790 \quad 0.6396]. \tag{2.6}$$

2.2 PREDICTOR

Similar to the FRI case, with the estimated parameters it is simple to compute the predictor based on expression 1.4 and compute the loss based on expression 1.5. The code used to compute the predicted output and two-norm of the prediction error is given below in listing 2.2.

LISTING 2.2
Predictor ARX

```
y_hat = Phi*theta_hat;
loss = norm(y_hat - y(3:end));
```

It can be observed in the Fig. 2.1 that there is almost no difference between the measured output and predicted output. This is further confirmed with the two-norm difference between the two, which is equal to 9.3259.

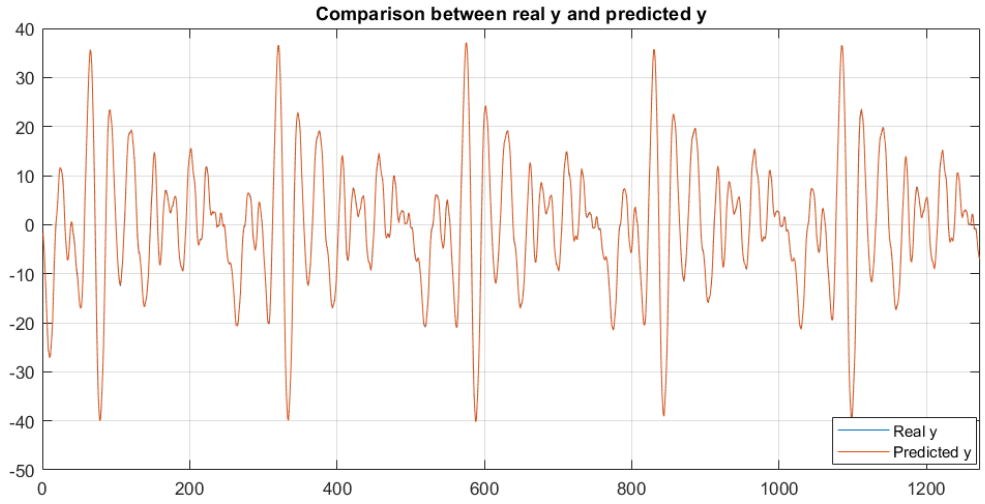


FIGURE 2.1
Comparison between the measured output and the predicted output

2.3 IDENTIFIED MODEL

Based on the previously computed vector of parameters $\hat{\theta}$, the identified model is constructed. Using the function *lsim* the output of the identified model is simulated. The results of simulation compared to the true output measurements are given in Fig. 2.2, while the code is given in listing 2.4. It can be observed that identified model follows the trend of the data consistently. However, at peak values in some instances

it underestimates the value, while in others it overestimates. This is reflected in the value of the loss function, in this case equal to 44.5101.

LISTING 2.3
Identified ARX model

```
a1 = theta_hat(1);
a2 = theta_hat(2);
b1 = theta_hat(3);
b2 = theta_hat(4);

B = [b1 b2];
A = [1 a1 a2];
G_hat = tf(B,A,Te);

y_m = lsim(G_hat,u);

loss = norm(y_m - y);
```

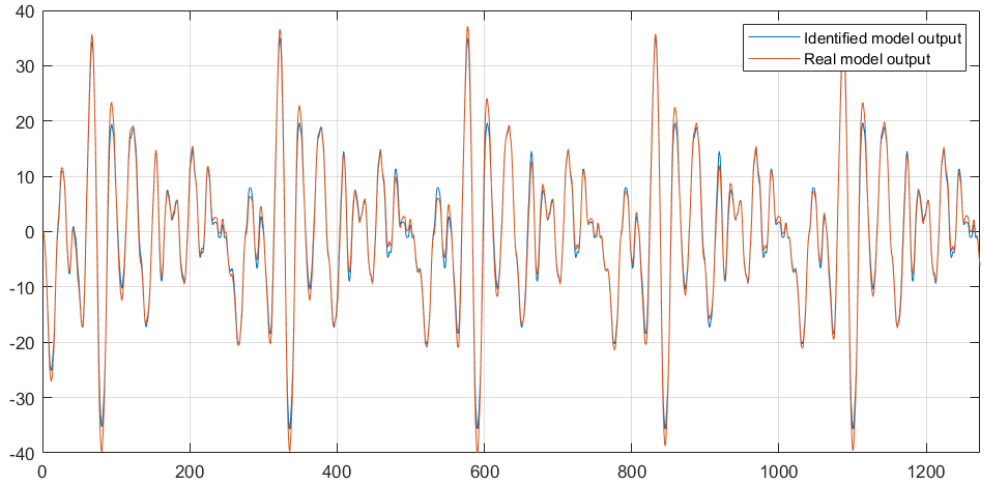


FIGURE 2.2
Comparison between the measured output and the identified output

2.4 INSTRUMENTAL VARIABLE METHOD

For asymptotical unbiasedness of the parameter estimates following conditions on cross-correlation and autocorrelation must be met:

- $R_{\phi\phi}(0)$ is not singular,
- $R_{\phi e}(0) = 0$.

The first condition is satisfied with a sufficiently rich excitation signal, while the second one is met if the vector $e(k)$ is not correlated with the vector ϕ. The vector ϕ consists of past inputs and outputs. The noise is independent of the input signal and they are thus not correlated. However, unless the noise is white, the error is going to be correlated with the output and the parameter estimate will be biased. In order to

satisfy above mentioned conditions, the observation matrix Φ is replaced by matrix Φ_{iv} of instrumental variables. The new estimates are given by:

$$\hat{\theta}_{iv} = (\Phi_{iv}^T \Phi)^{-1} \Phi_{iv} Y. \tag{2.7}$$

If the vector of instrumental variables is chosen such that it is not correlated with the noise, the parameter estiamtes will be asymptotically unbiased. Here, the noiseless output of the model is considered, as the output of the previously identified model $y_M(k)$, and the vector of instrumental variable is defined as:

$$\phi_{iv}^T(k) = [-y_M(k-1), ..., -y_M(k-n), u(k-d), ..., u(k-m)]. \tag{2.8}$$

The code for the IV method is given below in listing 2.4.

LISTING 2.4
Identified ARX model

```
Phi_iv = [-y_m(2:end-1),-y_m(1:end-2),u(2:end-1),u(1:end-2)];

Y = y(3:end);
theta_hat_iv = inv(Phi_iv'*Phi)*Phi_iv'*Y;

a1_iv = theta_hat_iv(1);
a2_iv = theta_hat_iv(2);
b1_iv = theta_hat_iv(3);
b2_iv = theta_hat_iv(4);

B_iv = [b1_iv b2_iv];
A_iv = [1 a1_iv a2_iv];
G_hat_iv = tf(B_iv,A_iv,Te);

y_m_iv = lsim(G_hat_iv,u);
loss = norm(y - y_m_iv);
```

Estimated parameters are:

$$\theta^T = [-1.8859 \quad 0.9366 \quad 0.3789 \quad 0.6381]. \tag{2.9}$$

Comparison between the measured output and the models identified with ARX and IV are given in Fig. 2.3. It can be observed that the IV method gives better results than the ARX method, the overestimates, while still there, are smaller in this case. This is further confirmed by the two-norm of the output error which is in this case slightly smaller with the value of 42.2164.

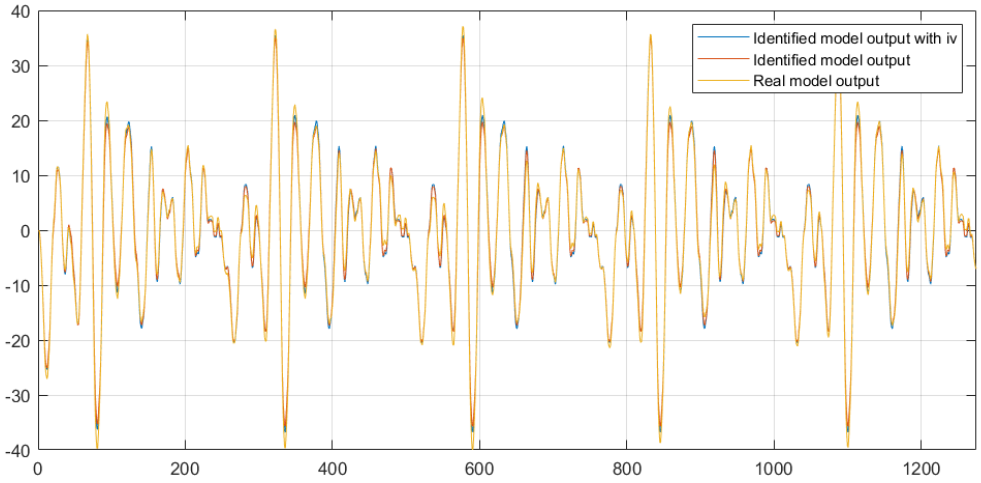


FIGURE 2.3
Comparison between the measured output, the output of ARX model and the output of ARX model with IV

CHAPTER 3

IDENTIFICATION OF A DC SERVOMOTOR: STATE-SPACE MODEL IDENTIFICATION

In this section, we aim to identify the state space model of our system using the Subspace method. We assume that we can represent our system as:

$$\mathbf{x}(k+1) = A\mathbf{x}(k) + B\mathbf{u}(k) \tag{3.1}$$
$$\mathbf{y}(k) = C\mathbf{x}(k) + D\mathbf{u}(k) \tag{3.2}$$

Hence we are interested in finding A, B, C, D.

3.1 Q DOUBLE EXTENDED OBSERVABILITY MATRIX

The first step is to find the extended observability matrix O_r of our system. Since our system is SISO, we can write:

$$Y = [Y_r(1), Y_r(2), \ldots, Y_r(N)]_{r\times N}$$
$$U = [U_r(1), U_r(2), \ldots, U_r(N)]_{r\times N}$$
$$X = [x(1), x(2), \ldots, x(N)]_{n\times N}$$

Then rewrite

$$Y_r(k) = O_r x(k) + S_r U_r(k) \text{ for } k = 1, \ldots, N \tag{3.3}$$

as:

$$Y = O_r X + S_r U \tag{3.4}$$

Where:

$$Y_r(k) = \begin{bmatrix} y(k) \\ y(k+1) \\ \vdots \\ y(k+r-1) \end{bmatrix}, \quad U_r(k) = \begin{bmatrix} u(k) \\ u(k+1) \\ \vdots \\ u(k+r-1) \end{bmatrix}$$

And r is a chosen parameter bigger than the true state of our system (in our case we have chosen $r = 100$) and N is the number of data points.

In equation 3.4 O_r, X, S_r are unknown so we can multiply every element by $U^\perp$ and obtain our double extended observability matrix Q:

$$Q \equiv YU^\perp = O_r X U^\perp \tag{3.5}$$

Here the relative MATLAB code for finding Q:

LISTING 3.1
Q Double extended observability matrix

```
%% Q computation
r = 100;
Y = [];
U = [];
N = length(y);
ny = size(y,2);
nu = size(u,2);

for i = 1 :1 : N - r +1
    Y(:,end+1) = y(i:i+r-1);
    U(:,end+1) = u(i:i+r-1);
end

U_ort = eye(size(U,2)) - U'*(U*U')^(-1) *U;
Q = Y * U_ort;
```

3.2 O_r EXTENDED OBSERVABILITY MATRIX

Now to obtain the extended observability matrix O_r, we first need to estimate the state of our system. To do that we have computed and plotted the singular value of Q (Figure 3.1), and only 2 of them are relevant, so the system could be approximated as a second-order system ($n = 2$). Theoretically speaking, we should have taken as order of our system the number of singular values of Q that are strictly positive.

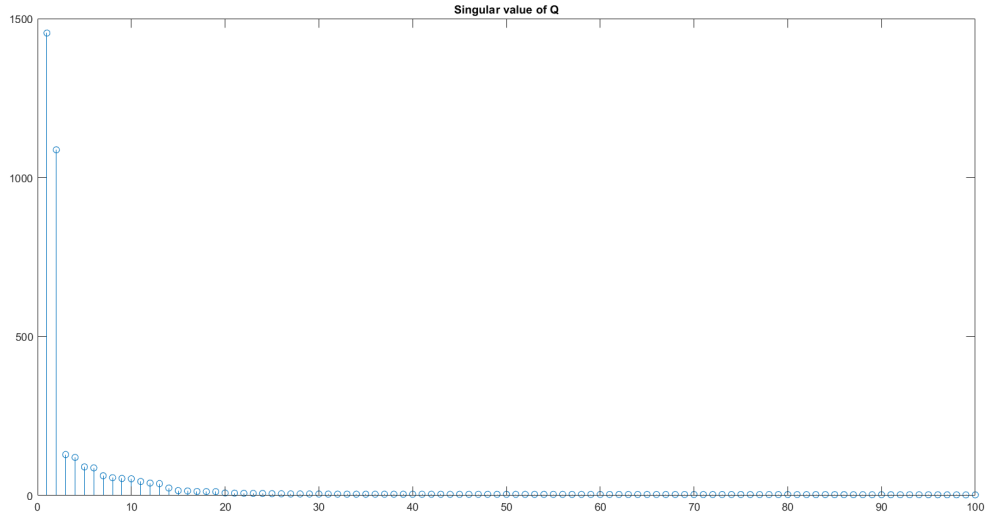


FIGURE 3.1
Singular value of Q

The extended observability matrix is now given as the first n raws of UU, where $Q = UU \cdot S \cdot V^T$. So we have used the MATLAB command svd to compute the extended observability matrix:

LISTING 3.2
Singular value decomposition

```
[UU,S,V] = svd(Q);
n = 2 ;
ext_obs = UU(:,1:(n));
```

3.3 A, C MATRICES

Obtained O_r we can now find A and C, since:

$$O_r = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{r-1} \end{bmatrix}_{rn_y \times n}$$

- Then, C is the first n_y rows of O_r
- A can be computed from:

$$[\text{the last } (r-1)n_y \text{ rows of } O_r] = [\text{the first } (r-1)n_y \text{ rows of } O_r] \times \hat{A}$$

where n_y is the number of output of our system (in our case 1).

Here is the MATLAB code to estimate A and C:

LISTING 3.3
A and C

```
%% A and C computation
C = ext_obs(1:ny,:);
A = ext_obs(1:(r-1)*ny,:) \ ext_obs(ny+1:end,:);
```

3.4 B, D MATRICES

Having an estimate of A and C, we can now estimate B and D, by constructing our prediction, which is a linear regression:

$$\hat{y}(k) = \hat{C}(qI - \hat{A})^{-1}B_u(k) + D_u(k) \tag{3.6}$$

$$= \begin{bmatrix} B^T & D \end{bmatrix} \begin{bmatrix} y_f(k) \\ u(k) \end{bmatrix} \tag{3.7}$$

$$= \theta^T \phi(k) \tag{3.8}$$

where:

$$u_f(k) = \hat{C}(qI - \hat{A})^{-1}u(k) \tag{3.9}$$

By assuming $D = 0$, the relative code is:

LISTING 3.4
B and D

```
%% B and D computation
D = zeros(ny,nu);
z = tf('z',Te);
Filter = C*(z*eye(size(A))- A)^(-1);
Filter_red = reduce(Filter,2);
uf1 = lsim(Filter_red(1,1),u);
uf2 = lsim(Filter_red(1,2),u);
phi = [uf1'; uf2'];
B = (phi*phi')^(-1) * phi * y;
```

3.5 PLOTTING AND VALIDATION

Those are the obtained matrices:

$$\hat{A} = \begin{bmatrix} 0.9506 & -0.2524 \\ 0.1889 & 0.9322 \end{bmatrix}, \quad \hat{B} = \begin{bmatrix} 10.14 \\ -8.074 \end{bmatrix},$$

$$\hat{C} = \begin{bmatrix} 0.2463 & 0.2594 \end{bmatrix}, \quad \hat{D} = 0$$

In Figure 3.2 it's possible to observe both the identified model and the real one.

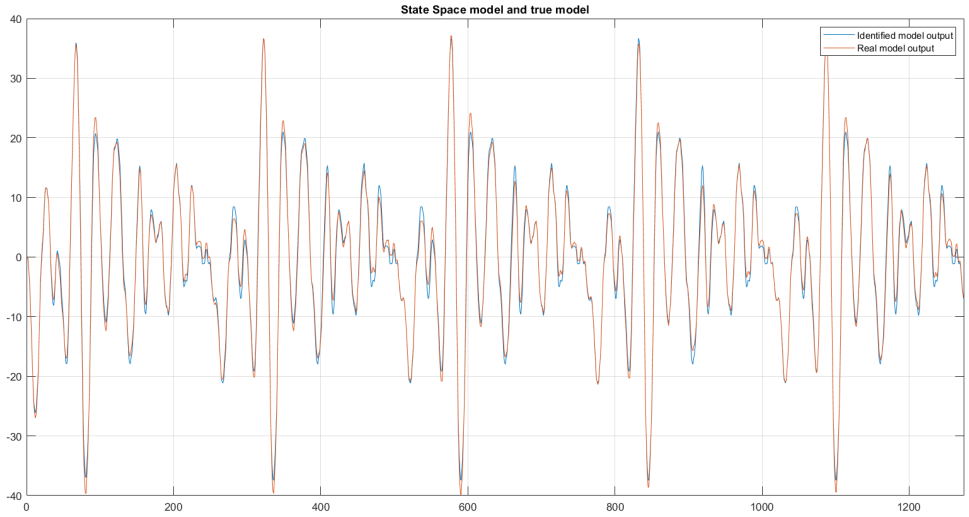


FIGURE 3.2
Comparison between the response of the identified model and true one

and the normalized sum of squared errors is 0.0304. Here is the code for computing the output of the identified model:

LISTING 3.5
SS Output

```
%% Output SS
G_hat = ss(A,B,C,D,Te);
y_m = lsim(G_hat,u);
```

COMPARISON OF THE NORMALIZED SUM OF SQUARED ERRORS

In the tables below are summarized the normalized sum of squared errors of the different methods:

TABLE 3.1
Residual Analysis for Different System Identification Techniques

Technique	Normalized sum of squared errors
ARX	0.1622
ARX with IV	0.0331
SS	0.0304

CHAPTER 4

PARAMETRIC IDENTIFICATION OF A FLEXIBLE LINK SYSTEM: ORDER ESTIMATION

4.1 FREQUENCY RESPONSE

Based on the shape of the Bode magnitude plot number of zeros and poles of a system can be estimated. Every slope increase of 20dBs per decade can be correlated with one zero and every slope decrease of 20dBs per decade can be correlated with one pole. Thus if we go along the Bode magnitude plot, we start counting the zeros and poles. Based on the slope of the curve until 10Hz there is one zero, after that there is a change in the slope, which now becomes negative, we estimate this with two poles. Around 80 Hz there is a change in trend, a resonance frequency and an increase of the slope, which can be estimated with 2 zeros. Lastly around 150 Hz there is another change in the slope trend, going from positive to negative trend, which can be explained by additional four poles. Thus based on this analysis the estimated number of zeros is 3 and number of poles is 6, and the estimated order is 6. The Bode diagram in question is given in Figure 4.1. The code is given in the Listing below.

LISTING 4.1
Frequency response

```
load("CE2.mat")

Z = iddata(y,u,Ts);
Z = detrend(Z);

figure
bode(spa(Z))
grid on
```

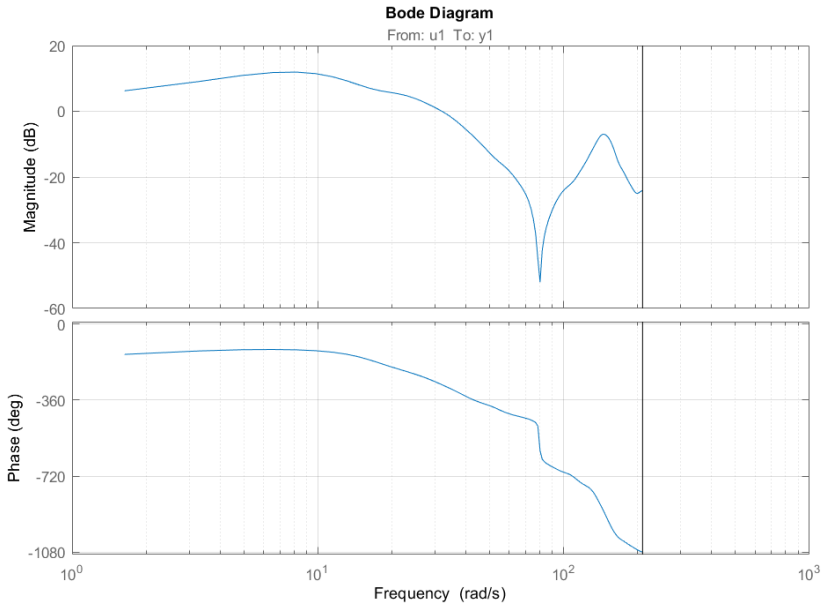


FIGURE 4.1
Bode diagram of the model

4.2 LOSS FUNCTION

Model order selection is always performed with an ARX structure because the least square algorithm gives the global minimum of the identification criterion and the monotonically non-increasing property of the loss function is preserved. The other structures (e.g. OE, ARMAX, BJ) that are based on nonlinear optimization methods are not appropriate for order selection. If the noise level is too large the instrumental variables method can be used instead. Thus by analyzing the loss function of an ARX structure, the model order can be estimated as $\delta = \max(m, n)$. The code used to compute the loss for ARX models of different orders is given in Listing 4.2.

LISTING 4.2
Loss function of ARX structure for structures of different orders

```
nk = 1; % assumed delay
loss_array = [];
for i = 1: 10 % assuming models of different orders
    loss_array(:,end + 1) = arx(Z,[i i nk]).EstimationInfo.LossFcn;
end

plot(1:length(loss_array),loss_array)
grid on
title('Loss function over sigma');
```

The loss function evolution is given with Figure 4.2. It can be observed in this figure that loss rate decrease is very small for orders larger than 7, and it can be said that further increase of order is not justified since there is no significant improvement, while there is improvement for all orders smaller than 7 by increasing the order up to 7. With this we estimate the order to be 7.

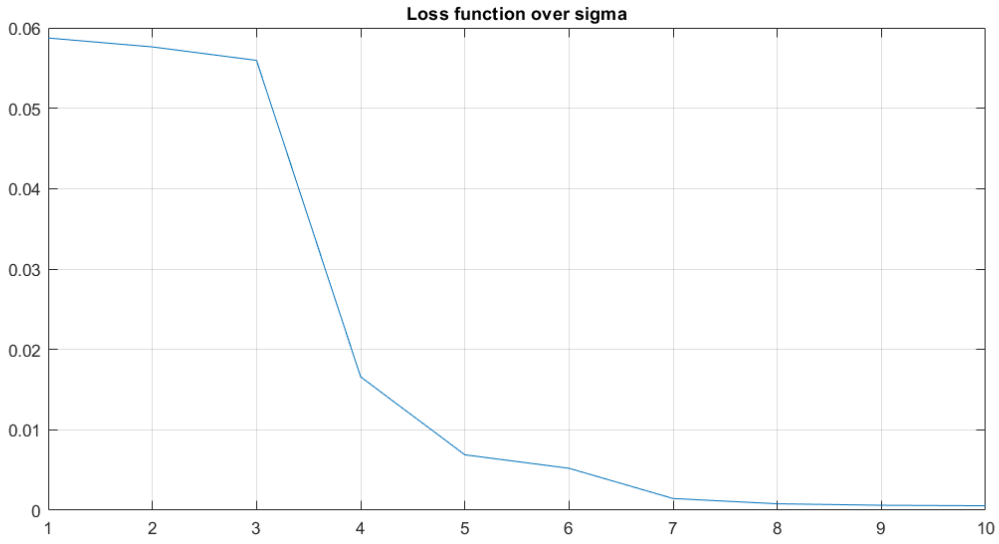


FIGURE 4.2
Loss function evolution

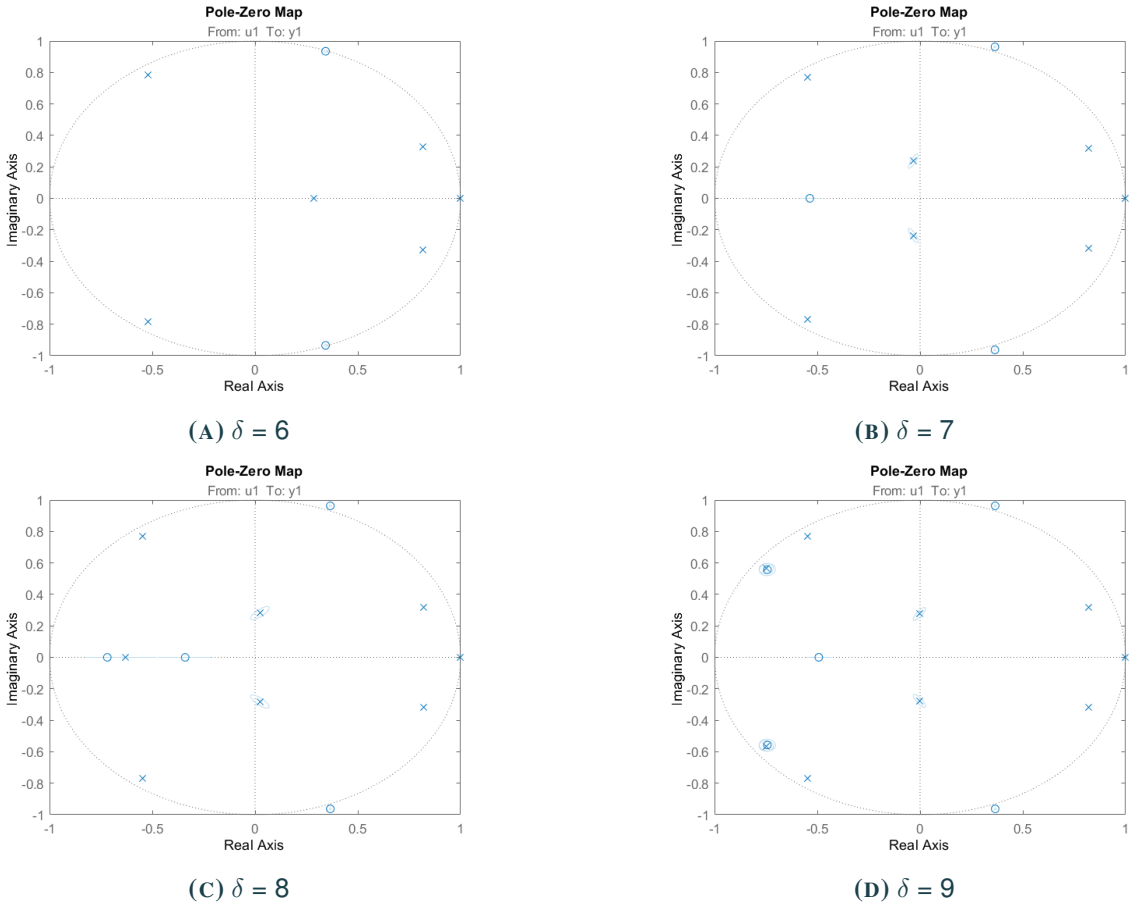
4.3 ZERO/POLE CANCELLATION

Under the assumption of delay of 1 and using the loss function evolution given with Figure 4.2, we further consider zero/pole cancellations for ARMAX structure of orders 6 to 9. Code is given in the listing below, and Figure 4.3 gives the corresponding pole-zero maps for each of the ARMAX structures. Zero/pole cancellation is possible when poles and zeros of a model are very close to each other in this way reducing the order of the model without affecting the behaviour of the plant model.

LISTING 4.3
Zero/Pole cancellation for ARMAX model structures of different orders

```
armax_array = struct([]);
for i = 4 : 10
    armax_array{i} = armax(Z,[i i i nk]);
    figure(i)
    h = iopzplot(armax_array{i});
    xlim([-1 1])
    ylim([-1 1])
    showConfidence(h,2);
end
```

In Figure 4.3a it can be observed that for order 6 there are no zero/pole. Similar can be observed in Figure 4.3b. However, in Figure 4.3c one zero/pole cancellation can be observed on the real axis and in Figure 4.3d cancellation of a complex pair of zeros/poles is observed, both based on the displayed confidence intervals. In accordance with these observations we conclude that the order of the system is 7.



(A) $\delta = 6$ (B) $\delta = 7$

(C) $\delta = 8$ (D) $\delta = 9$

FIGURE 4.3
Zero/Pole cancellation for ARMAX model structures of different orders

4.4 DELAY ESTIMATION

For delay estimation ARX structure is used, as shown in Listing 4.4. Here we consider $n_A = n_B = \delta = 7$ as the order has been estimated in the previous sections. For the ARX function since it is a digital system there is always a delay of at least 1 so we take $n_k = 1$. Estimate of the time delay can be obtained as the number of first coefficients of $B(q^{-1})$ which are equal to zero. In the presence of noise this will not be the case so the coefficient is considered zero if it is between two standard deviations of the $k - th$ coefficient. The coefficients of $B(q^{-1})$ along with two standard deviations are given with table 4.1. Due to the assumption of delay larger than 1, the coefficient $b_0 = 0$. It can be observed that only b_1 is within the two standard deviations of the coefficient and is thus considered to be equal to zero. Based on this the estimated delay is $n_k = 2$.

LISTING 4.4
Delay estimation using ARX

```
delta = 7; % Estimated order of the system
ARX_model_no_delay = arx(Z,[delta delta 1])
M = [2 *ARX_model_no_delay.db;ARX_model_no_delay.b; -2
   *ARX_model_no_delay.db]
```

TABLE 4.1
Delay estimation with ARX

	b_0	b_1	b_2	b_3	b_4	b_5	b_6	b_7
$2*\sigma$	0	0.0030	0.0030	0.0031	0.0034	0.0052	0.0052	0.0043
$B(q^{-1})$	0	0.0006	-0.0209	-0.0751	0.1693	-0.0146	0.0177	0.1587
$-2*\sigma$	0	-0.0030	-0.0030	-0.0031	-0.0034	-0.0052	-0.0052	-0.0043

With the estimated order and estimated delay, the order of the numerator can be estimated as $n_B \leq \delta - n_k + 1 = 7 - 2 + 1 = 6$. Thus, the order of the numerator is 6.

4.5 COMPARISON

Structure selection can also be done using Matlab functions, result of using this function for the given dataset is given with Figure 4.4. Matlab suggests the model with the lowest unexplained output variance which has 20 parameters, while the model structure that we estimated has 13 parameters. It can also be observed that for number of parameters larger than 12 the misfit is less than 0.05%, for 20 parameters the misfit is 0.025%. It can be considered unreasonable to increase the number of parameters 1.5 times for such a small improvement.

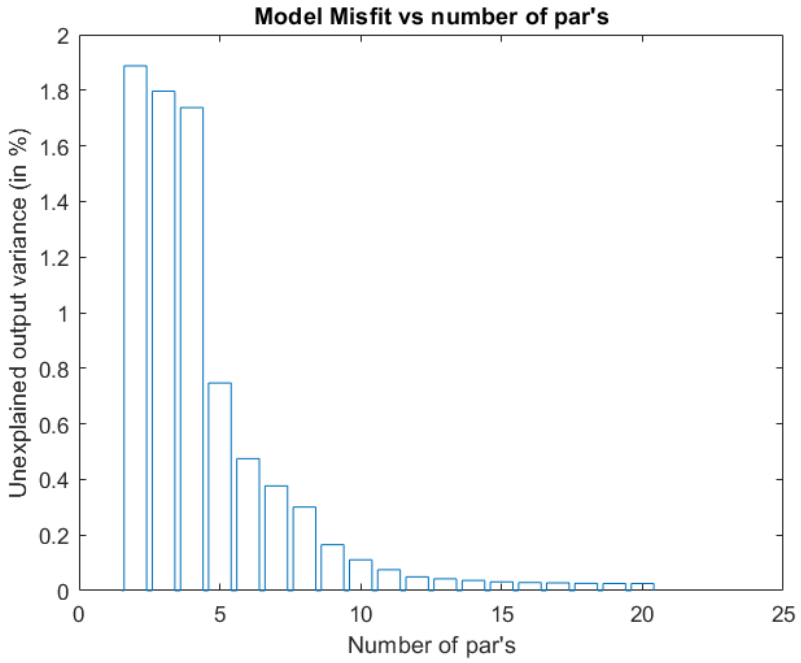


FIGURE 4.4
ARX Matlab model structure selection

CHAPTER 5

PARAMETRIC IDENTIFICATION OF A FLEXIBLE LINK SYSTEM: PARAMETRIC IDENTIFICATION

5.1 DIVISION OF THE DATA

To validate our identification, we need to split our original dataset Z into a validation set $Z_{validation}$ and a testing set for identification $Z_{identification}$. To do that we have chosen a test-to-train ratio of 0.5; Here is the relative MATLAB code:

LISTING 5.1
Data splitting

```
%% 2.2.2.1 Datadivision in 2 part
data_length = length(y);
ratio = 0.5;
validation_length = floor(ratio * data_length);
Z_validation = Z(validation_length+1:end);
Z_identification = Z(1:validation_length);
```

5.2 PARAMETRIC IDENTIFIFCATION

To perform model validation (Chapter 6), we first need to generate different models based on the dataset $Z_{identification}$. We will compare ARX, ARX with IV, ARMAX, OE, BJ, and SS. For the model that requires a structure of the error we will use $n_c = n_d = n_a$, and for state space identification we will use the global order δ as the number of states. Here is the relative code:

LISTING 5.2
Parametric ID

```
%% 2.2.2.1 Parametric ID
nk = 2;
na = 7;
nb = 6;
nc = 7;
nd = 7;
```

```
states_number = 7;
sys_arx = arx(Z_identification,[na nb nk]);
sys_iv4 = iv4(Z_identification,[na nb nk]);
sys_armax = armax(Z_identification, [na nb nc nk]);
sys_oe = oe(Z_identification,[nb na nk]);
sys_bj = bj(Z_identification,[nb nc nd na nk]);
sys_n4sid = n4sid(Z_identification,states_number);
```

CHAPTER 6

PARAMETRIC IDENTIFICATION OF A FLEXIBLE LINK SYSTEM: MODEL VALIDATION

6.1 TIME DOMAIN COMPARISON

By comparing the model in time domain it appears that output error is the best model with an accuracy of 95.14 % (Figure 6.1).

Here is the relative MATLAB code:

LISTING 6.1
Time domain comparison

```
%% Time domain comparison
compare(Z_validation,sys_arx,sys_iv4,sys_armax,sys_oe,sys_bj,sys_n4sid);
```

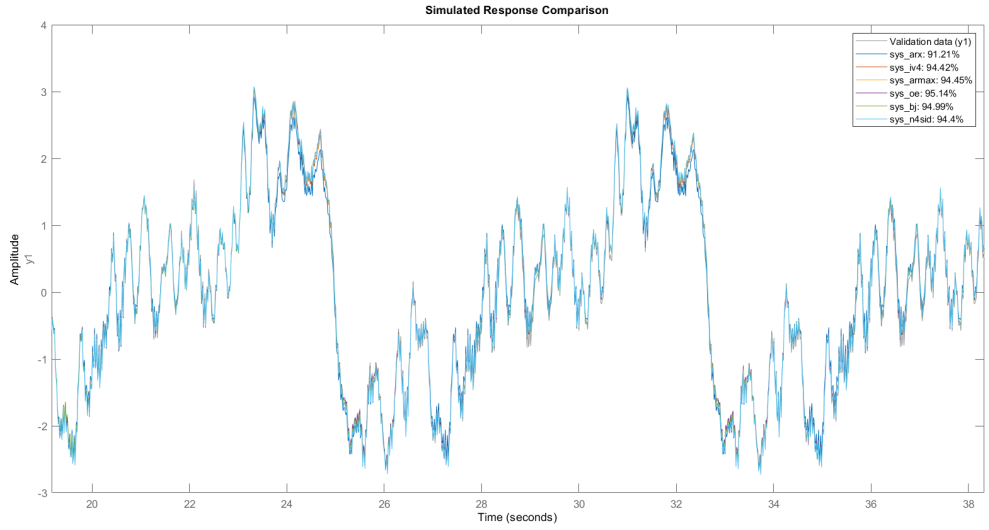


FIGURE 6.1
Time domain comparison

6.2 FREQUENCY DOMAIN COMPARISON

To compare the models in the frequency domain, obtaining a nonparametric frequency-domain identified model is first necessary. To do that, we first analyzed the input signal (Figure 6.2) and discovered that not only it's periodic but it is a PRBS signal with $n = 9$ and $p = 5$. So our first approach was to use the Fourier analysis method, by computing the fft for each period and using the average of if (Computer Exercise 1.5).

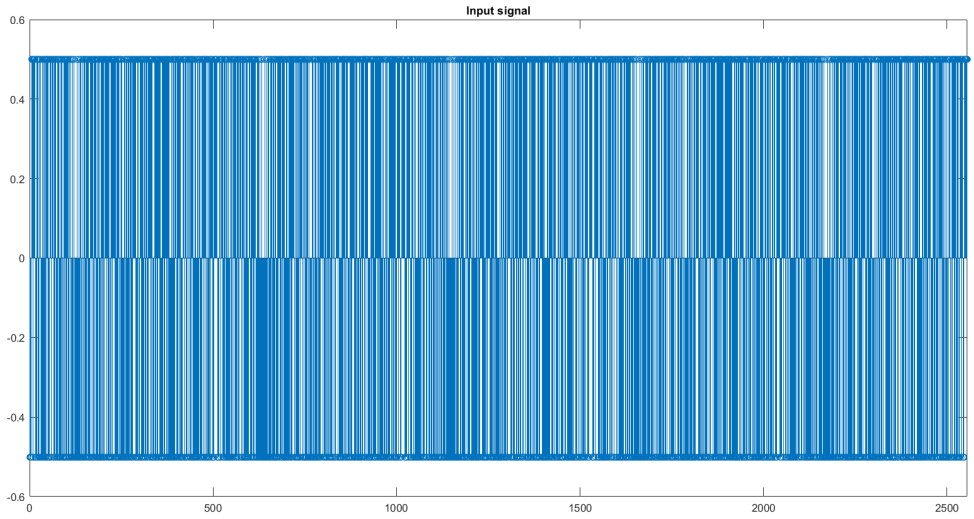


FIGURE 6.2
Input signal

The problem with this approach is that noise is not periodic and we have only 5 periods on which we are going to averaging, hence we can clearly see in Figure 6.3, that we need to smooth the curve. A solution is to use a window, so we have used the command spa with a window of length 300 to obtain another nonparametric frequency-domain model (Figure 6.4), From the bode magnitude of this plot we can observe that the best model seems to be the State space ones with an accuracy of 94,09 %.

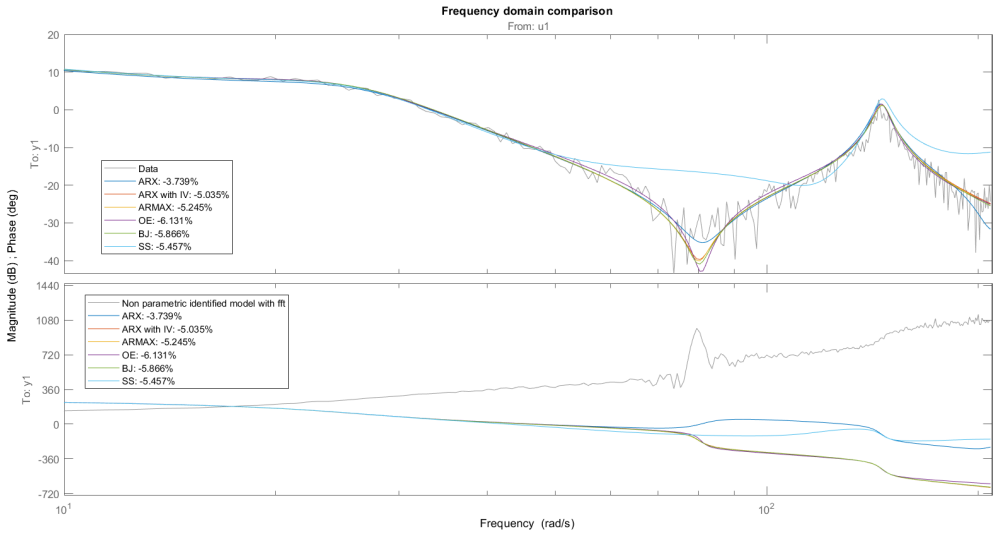


FIGURE 6.3
Frequency domain obtained with fft comparison

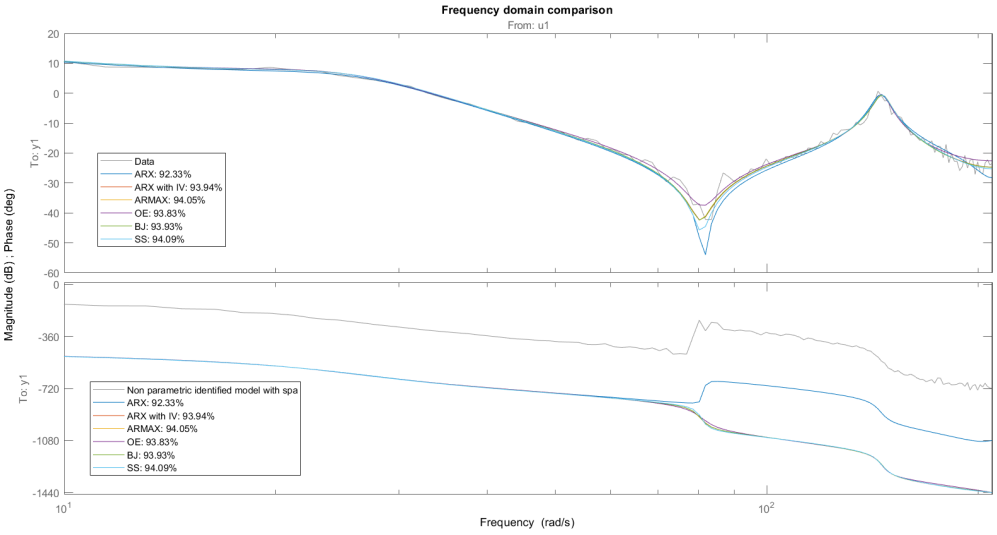


FIGURE 6.4
Frequency domain obtained with spa comparison

6.3 STATISTICAL ANALYSIS

For all the models,in order to validate the plant of the model, we will analyze the cross-correlation of the residuals and past inputs (Cross-correlation test). For the identification technique on which we have made assumptions on the noise model, like ARMAX, ARX, ARX with IV and BJ we will also conduct a whiteness test of the residual to validate the noise model structure.

ARMAX

The noise model is not validated, while the plant model is validated.

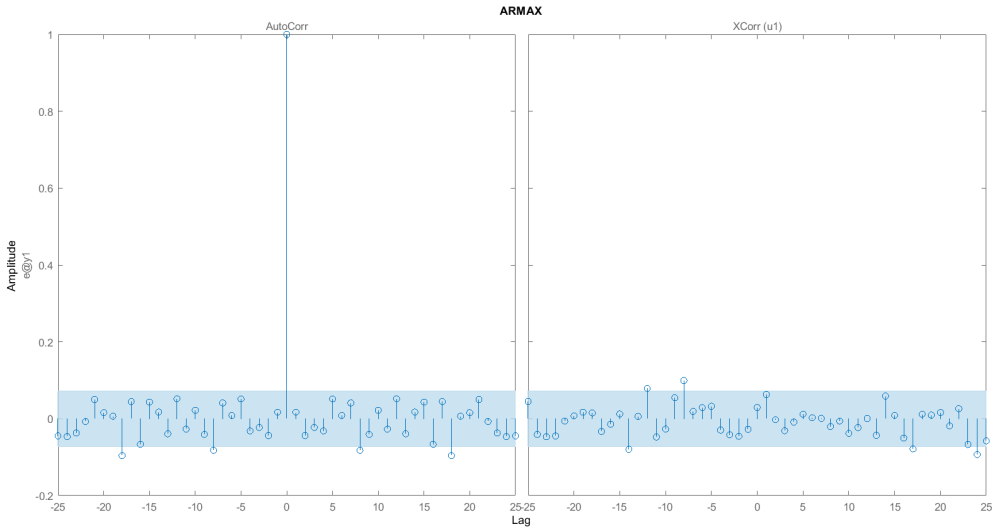


FIGURE 6.5
Statistical analysis ARMAX

ARX

The noise model is not validated, and also the plant model is not validated.

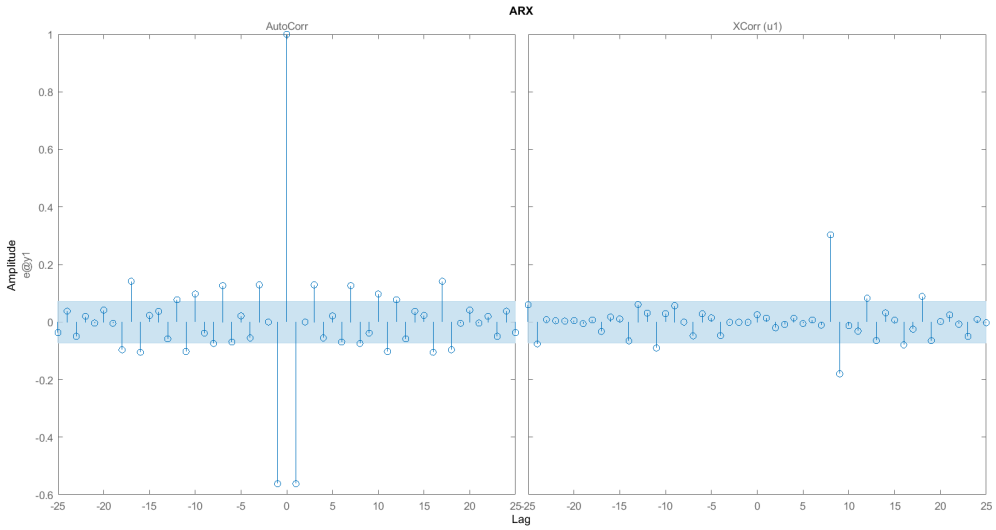


FIGURE 6.6
Statistical analysis ARX

ARX WITH IV

The usage of IV improves ARX, since now the plant model is validated. The noise model remain not validated

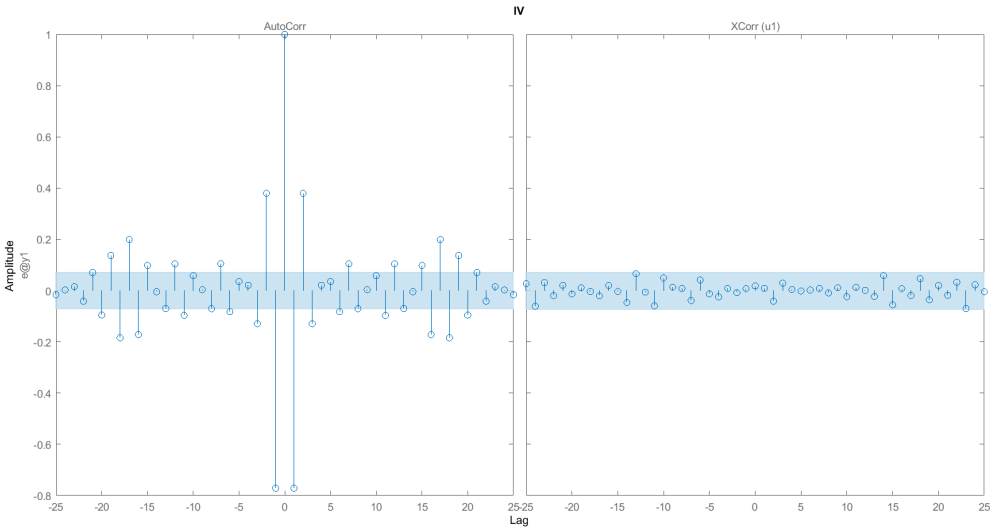


FIGURE 6.7
Statistical analysis ARX with IV

BJ

Here can validate the noise model, and, even if 1 point slightly exceeds the variance boundary, in the cross-correlation test, we have decided to validate also the plant model.

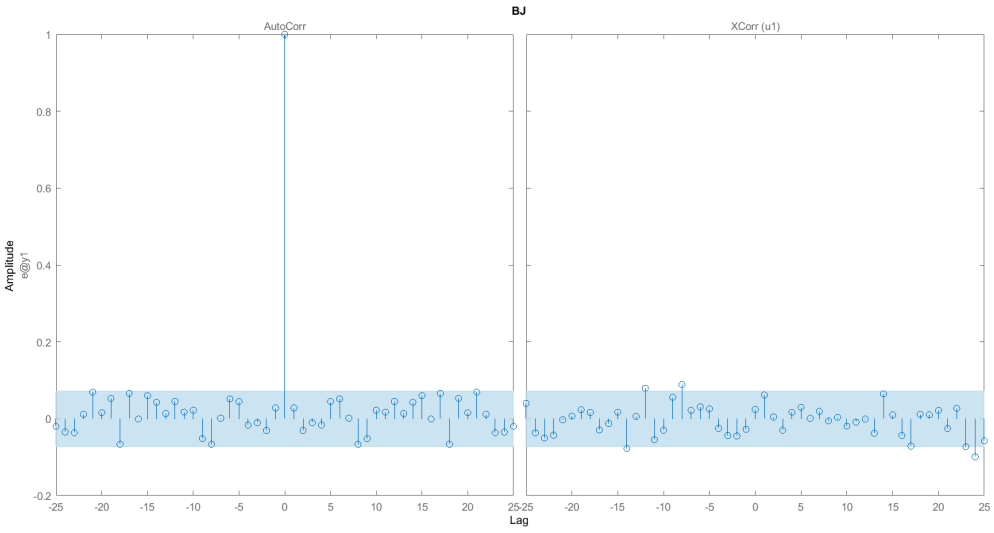


FIGURE 6.8
Statistical analysis BJ

OE

Here the plant model is verified.

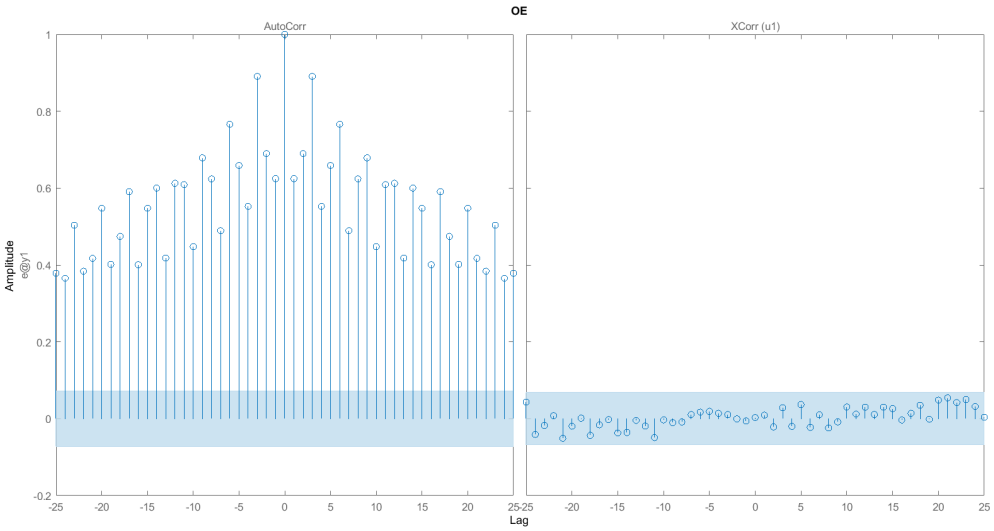


FIGURE 6.9
Statistical analysis OE

SS

Also in state space we can validate the plant model.

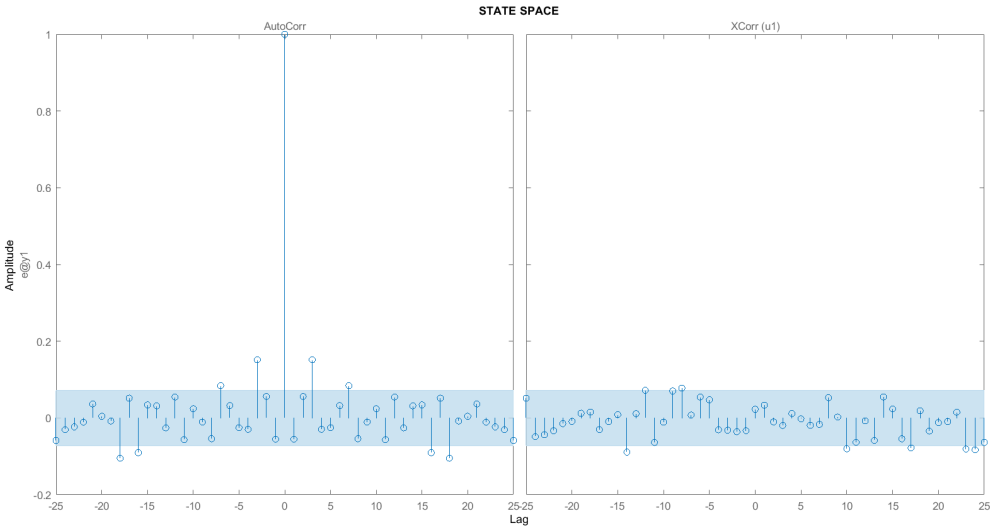


FIGURE 6.10
Statistical analysis SS

RECAP STATISTICAL

Below we summarize all the results:

Structure	Uncorrelation Test	Whiteness test
OE	passed	-
SS	passed	-
ARMAX	passed	**non passed**
ARX	**non passed**	**non passed**
ARX with IV	passed	**non passed**
BJ	passed	passed

6.4 FINAL DECISION

The final decision is based on all three tests that we have conducted in this section, so the structure OE with $n_k = 2, n_a = 7, n_b = 6$