ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

CE2 Report

System identification

Lebailly Tim
Thammisetty Devakumar

Professor : Karimi Alireza

Tuesday, 28 November 2018

Contents

List of Figures

1 FIR model identification

Code used in computation of FIR parameters

```
%% Load the data
Y = load('laserbeamdataN.mat');
N = length(Y.u);

%% Solution using FIR
m = 50;     % number of parameters to identify
n = 499;     % Number of data points to use (length of y)
start = 1;  % Starting data of vector y is start + 1

phi = zeros(n, m);
y = Y.y(start+1:start+n);
U = [zeros(m, 1); Y.u];
for i=1:n
    phi(i, :) = U(start+i:start+i+m-1, 1);
end
theta = inv(phi'*phi)*phi'*y;
y_est = phi*theta;
```

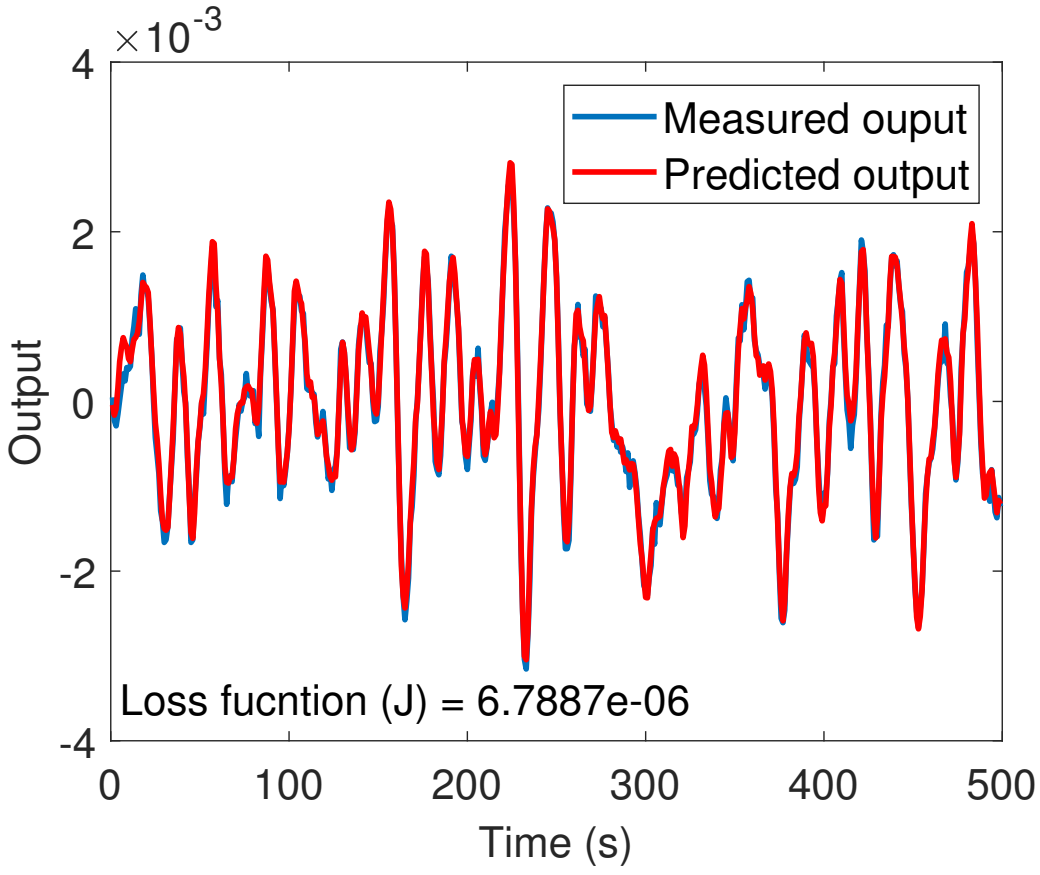

Figure 1: Measured and predicted output using FIR estimator

The value of the function J is the following

$$J(\theta) = \sum_{k=1}^{N} (y(k) - \hat{y}(k, \theta))^2 = 6.7887 * 10^{-6} \tag{1}$$

The measured output and the estimator are plotted in figure 1. Matlab code for computing the covariance of the parameters is given below.

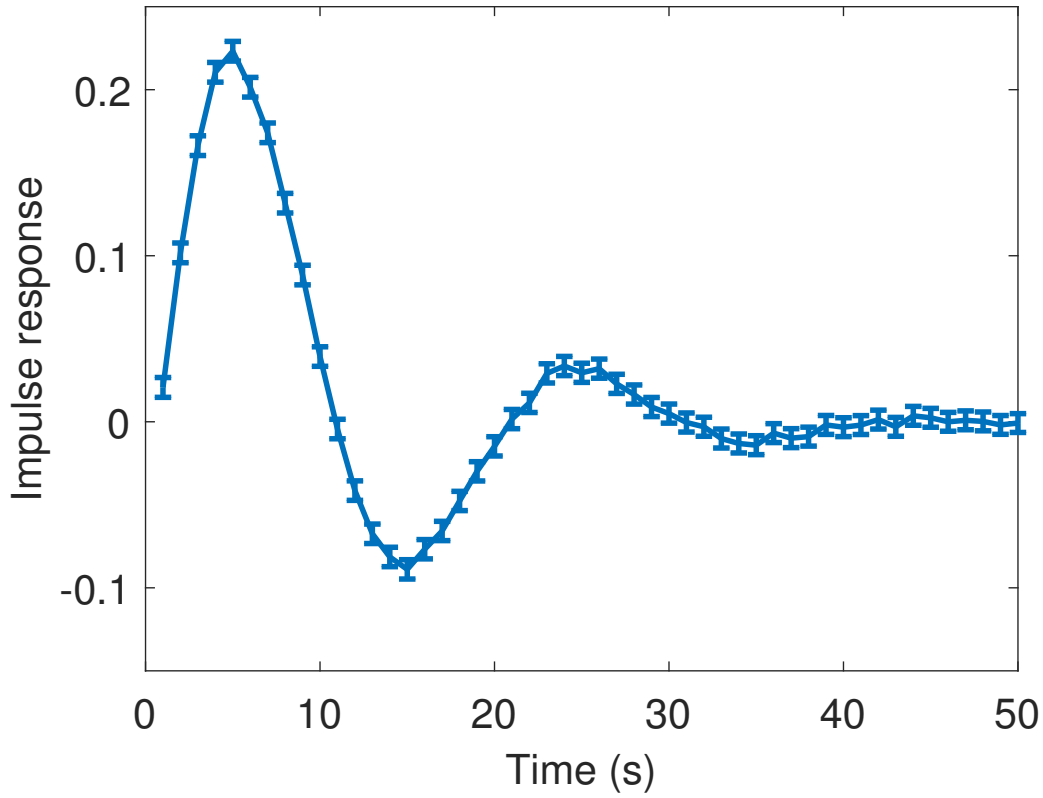

Figure 2: Identified impulse response along with confidence bounds

```
%% Loss function & covariance estimation code
J = sum((y - y_est).^2);
sigma_sq = J/(n-m);
cov_est = sigma_sq/n*inv(1/n*phi'*phi);
two_sigma = 2*sqrt(diag(cov_est));

% Plot impulse response with two_sigma
errorbar(flipud(theta), two_sigma, 'LineWidth', 2);
```

Impulse response identified using FIR method together with confidence intervals is given in figure 2

2 ARX model identification

1. *Note: the following code snippets are parts of one matlab.m file and should be ran sequentially.* This exercise focuses on ARX where an estimated parameter vector θ is computed where we minimize the error between the measured y and the estimated $\hat{y}(k,\theta)$, $\forall k \mid 1 \leq k \leq N$.

$$\hat{y}(k,\theta) = -a_1 y(k-1) - a_2 y(k-2) + b_1 u(k-1) + b_2 u(k-2) \tag{2}$$

This system of equation is solved using least squares. The computed parameter vector we obtain is the following: $\theta^T = [-1.5483, 0.6484, 0.0183, 0.0731]$.

```
% Definition of variables
Y = load("laserbeamdataN.mat");
y = Y.y;
u = Y.u;
N = length(u);
Te = 1/1000;
t = 0:Te:(N-1)*Te;

% Compute the vector theta
phi = @(k)  [-y(k-1); -y(k-2); u(k-1); u(k-2)];
matrix = [-y(1); 0; u(1); 0]*[-y(1); 0; u(1); 0]';
vec = [-y(1); 0; u(1); 0] * y(2);
for i = 3 : N
    matrix = matrix + phi(i)*phi(i)';
    vec = vec + phi(i)*y(i);
end
theta = matrix \ vec;

% Compute the estimate
output = @(k) phi(k)'*theta;
output_vec = zeros(N, 1);
output_vec(2) = [-y(1); 0; u(1); 0]'*theta;
for i = 3 : N
    output_vec(i) = output(i);
end

% Error function J
J = norm(y-output_vec)^2;

% Plot measured output and estimated output
figure;
plot(t, y, 'LineWidth', 2); hold on
plot(t, output_vec, 'LineWidth', 2);
xlabel("Time (s)")
ylabel("Output value of system")
legend("Measured output", "ARX estimator")
```

2. The value of the function J is the following:

$$J(\theta) = \sum_{k=1}^{N} (y(k) - \hat{y}(k,\theta))^2 = 2.1698 * 10^{-5} \tag{3}$$

The 2-norm of the prediction error is simply $\sqrt{J(\theta)} = 0.0047$.

The measured output and the estimator are plotted in figure 3.

3. The plot of the measured output and the simulated output can be found in figure 4. The 2-norm of the error is 0.0107.

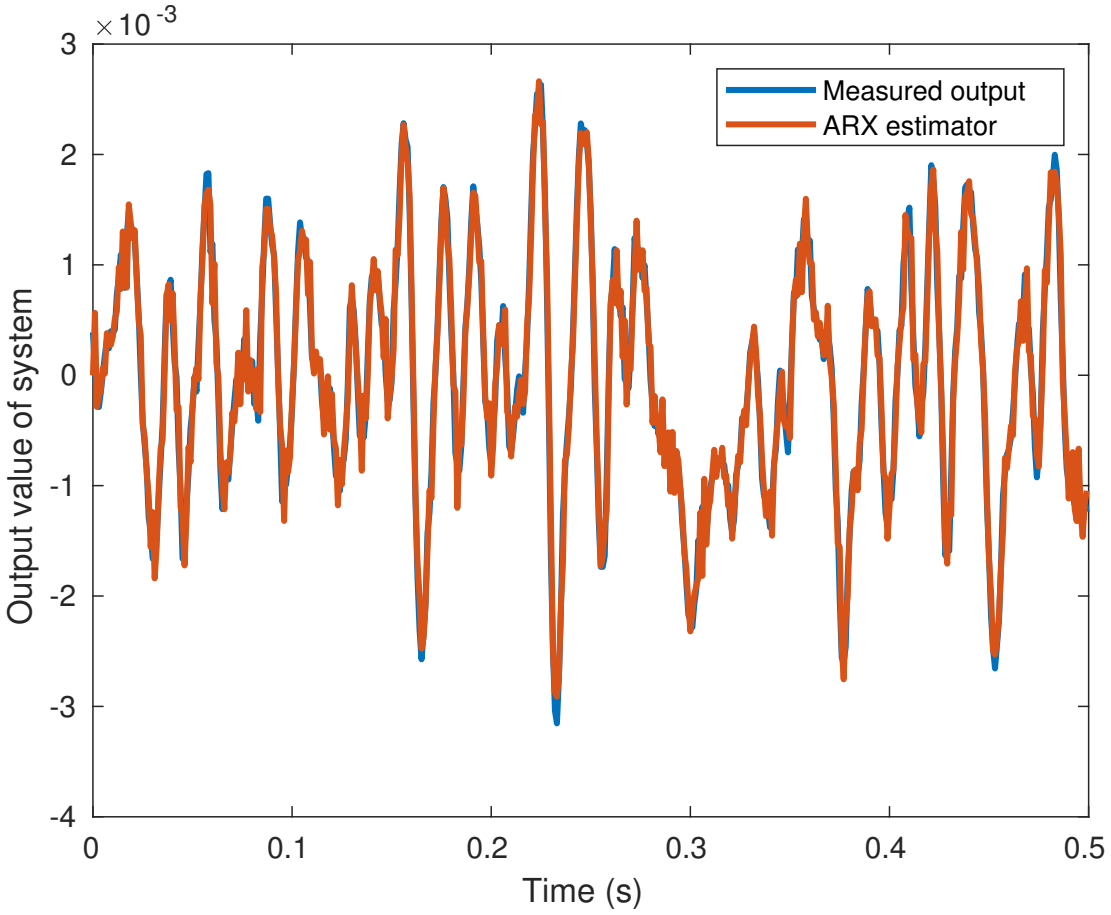

Figure 3: Measured output and ARX estimator

```
% Simulation of system with found theta
H = tf([theta(3) theta(4)], [1 theta(1) theta(2)], 1);
simulated = lsim(H, u);

% Error function J
J_simulated = norm(y-simulated)^2;

% Plot measured output and simulated output
figure;
plot(t, y, 'LineWidth', 2); hold on
plot(t, simulated, 'LineWidth', 2);
xlabel("Time (s)")
ylabel("Output value of system")
legend("Measured output", "Simulated output")
```

4. The parameter vector $\theta_{iv}^T = [-1.7748, 0.8704, 0.0174, 0.0685]$. It is worth noting that the simulated output of the system using θ_{iv} is much better than the simulated output using θ. The parameter vector is unbiased if 2 conditions are met. One of them is achieved at $R_{\phi e}(0)$ = 0. This means that the vector ϕ's should not be correlated with the error vectors $e(k)$. This means that $y(k)$ should not be correlated with $\phi(k)$ because $y(k)$ contains the measured noise. Recall that $\phi(k)$ is defined as:

$$\phi^T(k) = [-y(k-1), -y(k-2), -u(k-1), -u(k-2)] \tag{4}$$

$\phi(k)$ is thus obviously correlated with $y(k)$. One way to remove the correlation is to replace $\phi(k)$ by $\phi_{iv}(k)$ where $\phi_{iv}(k)$ is the output of the simulated system using the computed θ from

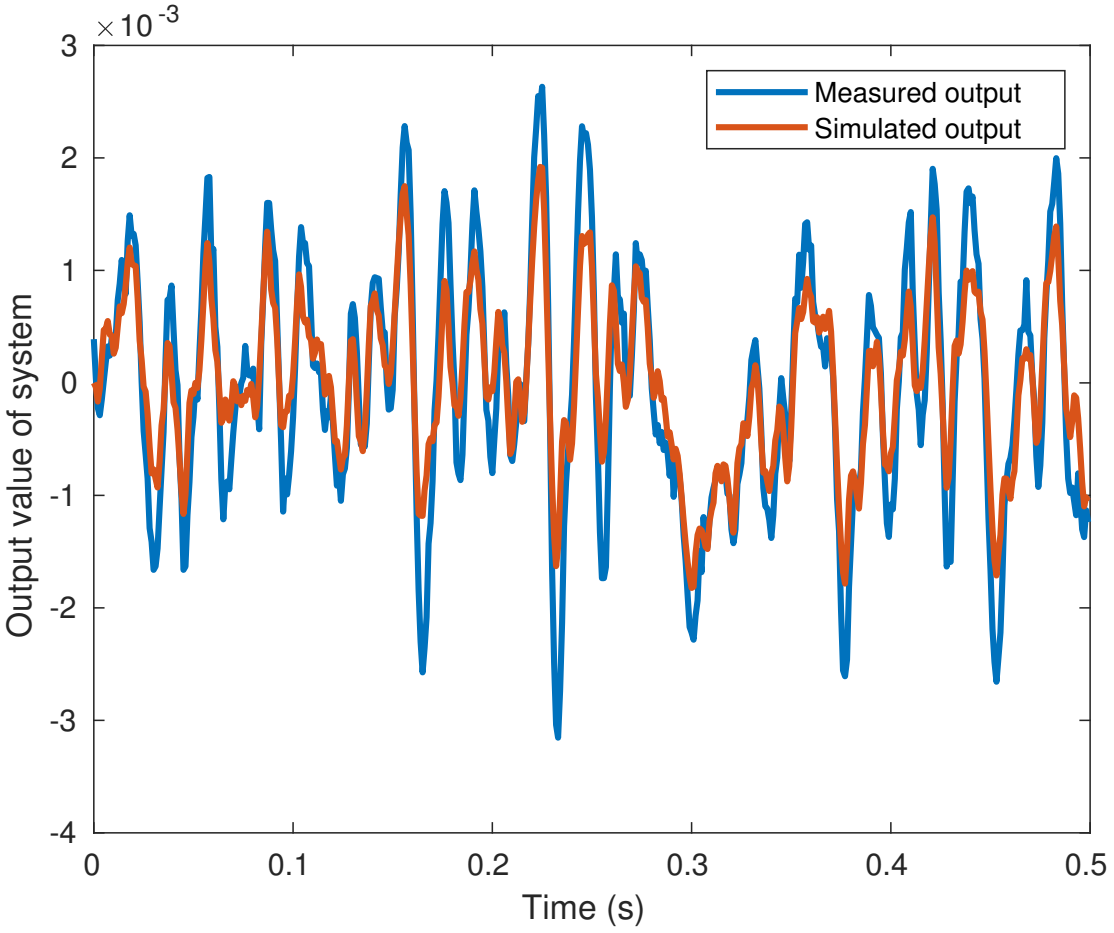

Figure 4: Measured output and simulated output

part 1. This is how the higher performance is achieved. Note that this process can be done iteratively for better performance. The output of the simulated systems and the measured output can be found in figure 5.

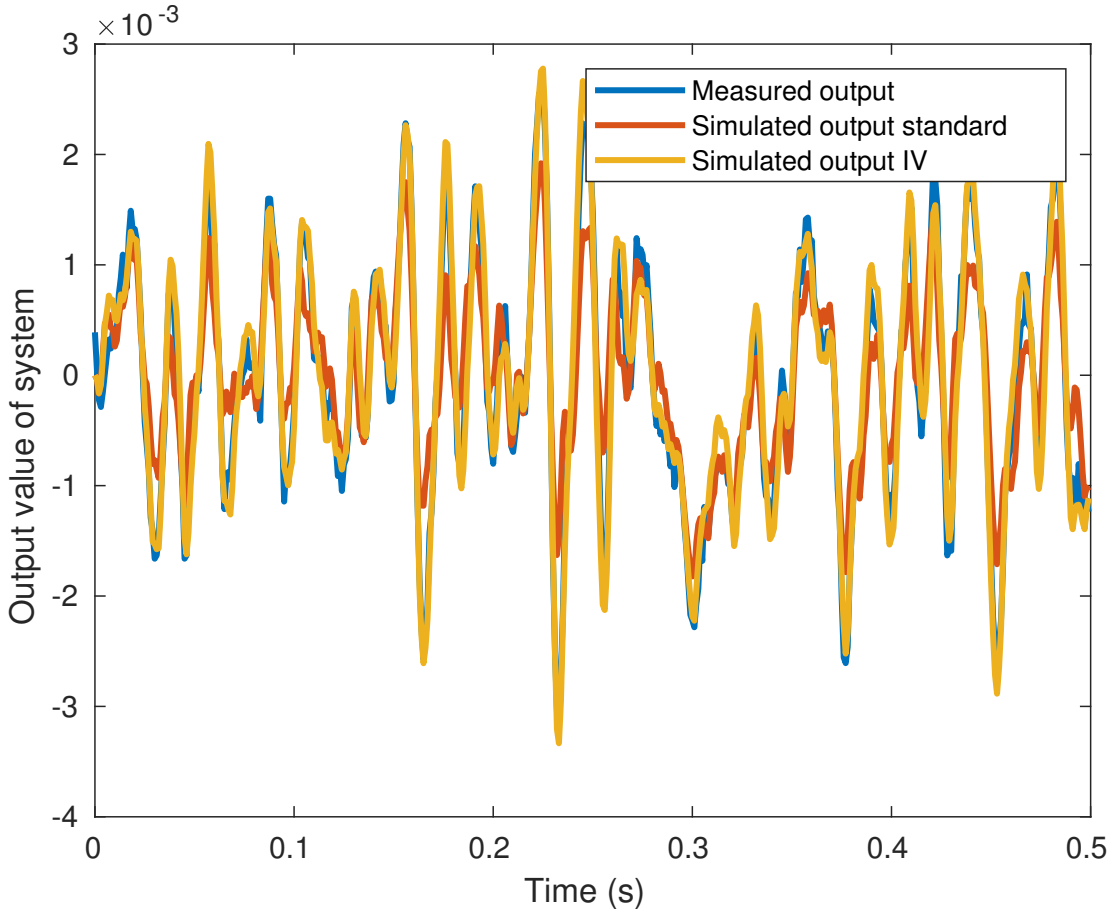

Figure 5: Measured output and simulated output

3 State-space model identification

Code for computing Q

```
%% CE 2.3

%% Loading input data
Yi = load('laserbeamdataN.mat');
y = Yi.y;
u = Yi.u;
Ni = length(y);
Te = 1;

%% State space system identification
r = 10; N = Ni - r + 1;
Y = zeros(r, N); U = zeros(r, N);

for i=1:N
    Y(:, i) = y(i:i+r-1);
    U(:, i) = u(i:i+r-1);
end
U_ortho = eye(N) - U'*inv(U*U')*U;
Q = Y*U_ortho;
Singular_values = svd(Q);
```

From the singular values, it is observed that only two values are significant and remaining values

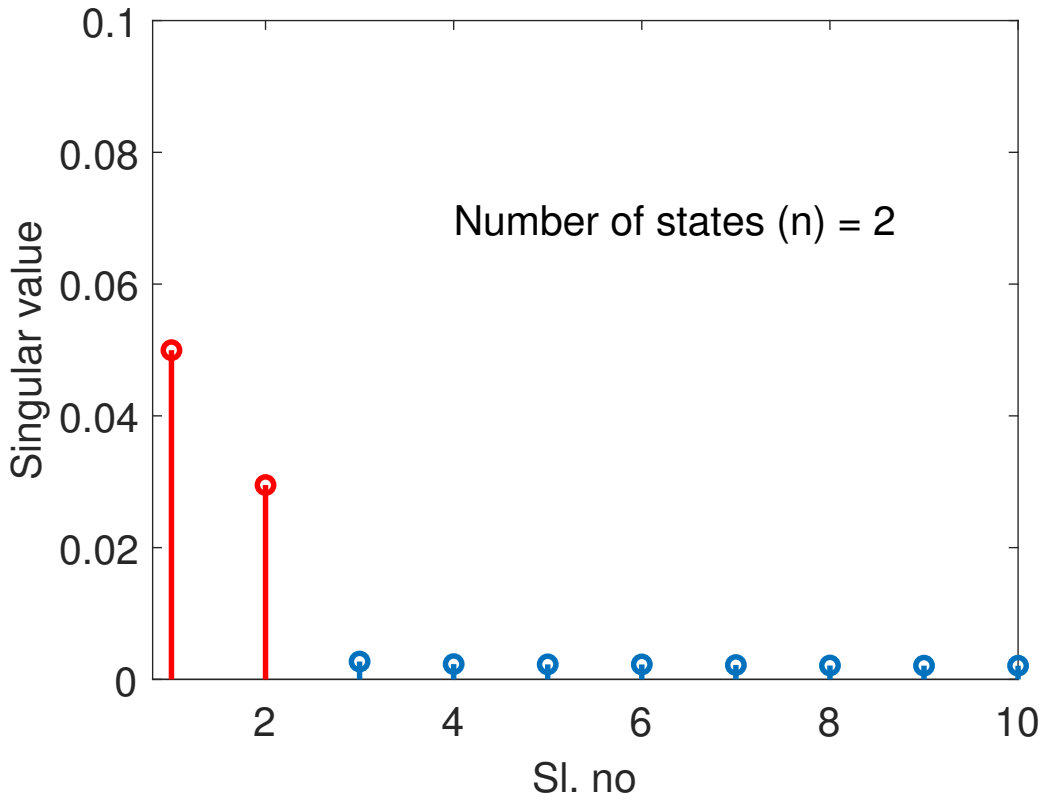

Figure 6: Singular values of Q

are very close to zero. Hence, we conclude the number of states to be equal to 2.

Code for computing A and C

```
%% Model parameters
n = 2;                                  % No. of states
Or = Q(:, 1:n);                         % Observability matrix
C = Or(1, :);                           % ny = 1,
A = Or(1:r-1, :)\Or(2:end, :);          %
err = Or(2:end, :) - Or(1:r-1, :)*A;
norm(err);
norm(A);
```

```
A =

    0.2404   -0.9869
    0.3718    1.3656


C =

   1.0e-03 *

    0.3767   -0.0184
```

Code for computing B

```
%% Estimation of B, D
D = 0;
q = tf('q', Te);
F = C*inv(q*eye(2) - A);

%% Estimating B using least squares
m = 1;    % number of parameters to identify
n = 499;    % Number of data points to use (length of y)
start = 1;  % Starting data of vector y is start + 1

% B is of size 2x1
% First element of B
phi_B1 = zeros(n, m);
y = Yi.y(start+1:start+n);
uf_B1 = lsim(F(1), Yi.u);

U = [zeros(m, 1); uf_B1];
for i=1:n
    phi_B1(i, :) = U(start+i:start+i+m-1, 1);
end
theta_B1 = inv(phi_B1'*phi_B1)*phi_B1'*y;

% Second element of B
phi_B2 = zeros(n, m);
y = Yi.y(start+1:start+n);
uf_B2 = lsim(F(2), Yi.u);

U = [zeros(m, 1); uf_B2];
for i=1:n
    phi_B2(i, :) = U(start+i:start+i+m-1, 1);
end
theta_B2 = inv(phi_B2'*phi_B2)*phi_B2'*y;
B = [theta_B1; theta_B2];
```

```
B =

  -316.2606
  -273.9756


D =

     0
```

Code for computing model output

```
%% State space model, Te=1
SStf = ss(A, B, C, D, Te);
y_est = lsim(SStf, Yi.u);

% 2 norm of the error
J = sqrt(sum((Yi.y - y_est).^2));
```

The measured output and the model are plotted in figure 7.

The 2-norm value is 0.0148

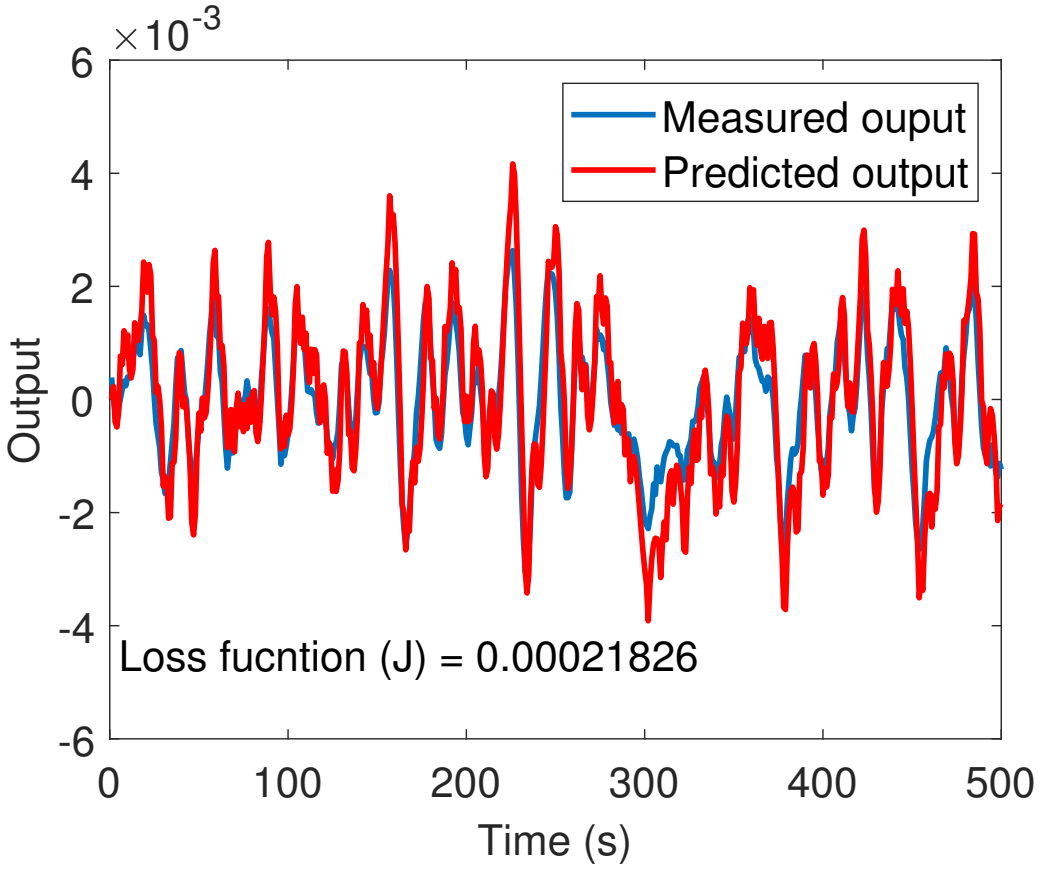

Figure 7: Measured output and identified state-space model output

4 Order estimation

Plot of loss function w.r.t model order is shown in figure 8. From the loss function, we can conclude that the order of the system must be at least 6. Hence, we say that the model order will be in the range from 6-10 based on the loss function. Since, the value of loss function is not zero from 6-10, we look for the pole zero cancellation for better identification of model order.

Zero-pole cancellation approach using ARMAX structure

From the zero-pole plot of **6th order** ARMAX model given in figure 9, we see that there are no zero-pole cancellations possible with 2σ confidence intervals. Hence, the model order is atleast 6.

From the zero-pole plot of **7th order** ARMAX model given in figure 10, we see that there are no zero-pole cancellations possible with 2σ confidence intervals. Hence, the model order is at least 7.

From the zero-pole plot of **8th order** ARMAX model given in figure 11, we see that there is a possibility for the poles and zeros on real axis to get cancelled. The 2σ confidence interval shown here extend to zeros location at +1. Hence, we look at models of higher order for any further change in the confidence intervals and cancellations.

From the zero-pole plot of **10th order** ARMAX model given in figure 12, we see the cancellation of 2 pairs of zeros and poles, with the 2σ confidence interval. However, we observer that if we increase the confidence level to 3σ, there is a probability for two more poles to get cancelled, which makes the system of order 6 (i.e 10-2-2). Since, the zero-pole plot of order 7, clearly indicates that

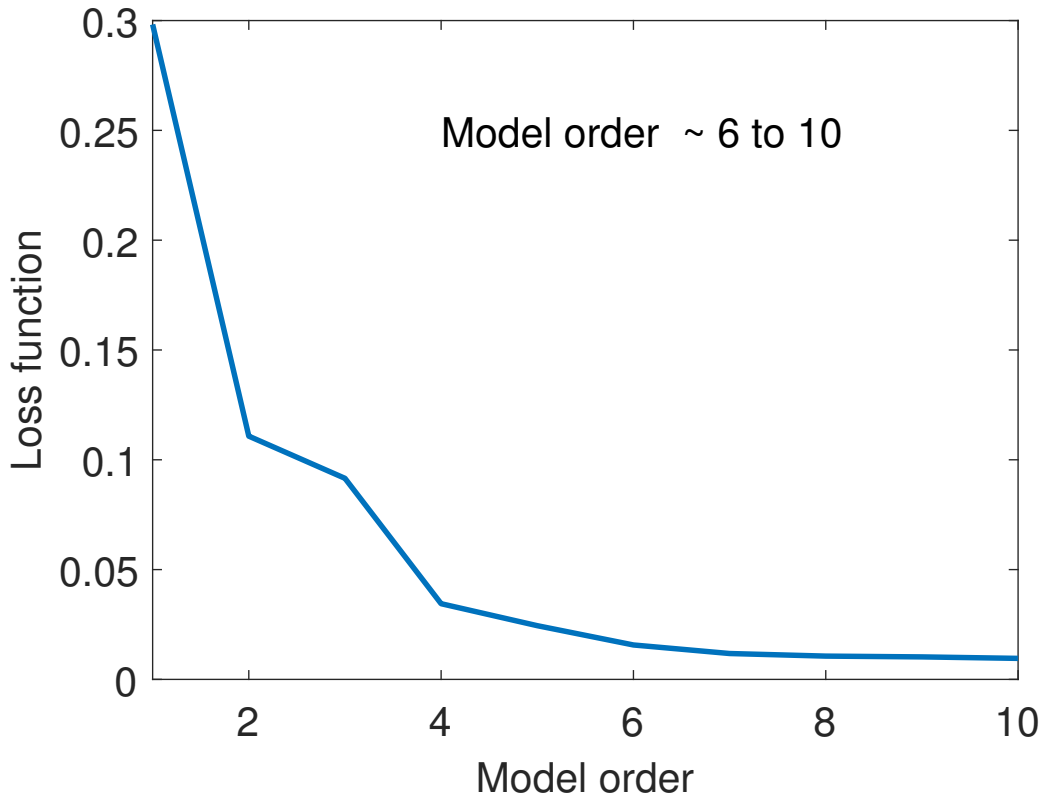

Figure 8: ARX model loss function vs model order

that order has to be greater or equal to 7, we can claim that a model with order of 7, will be a good representation of the system.

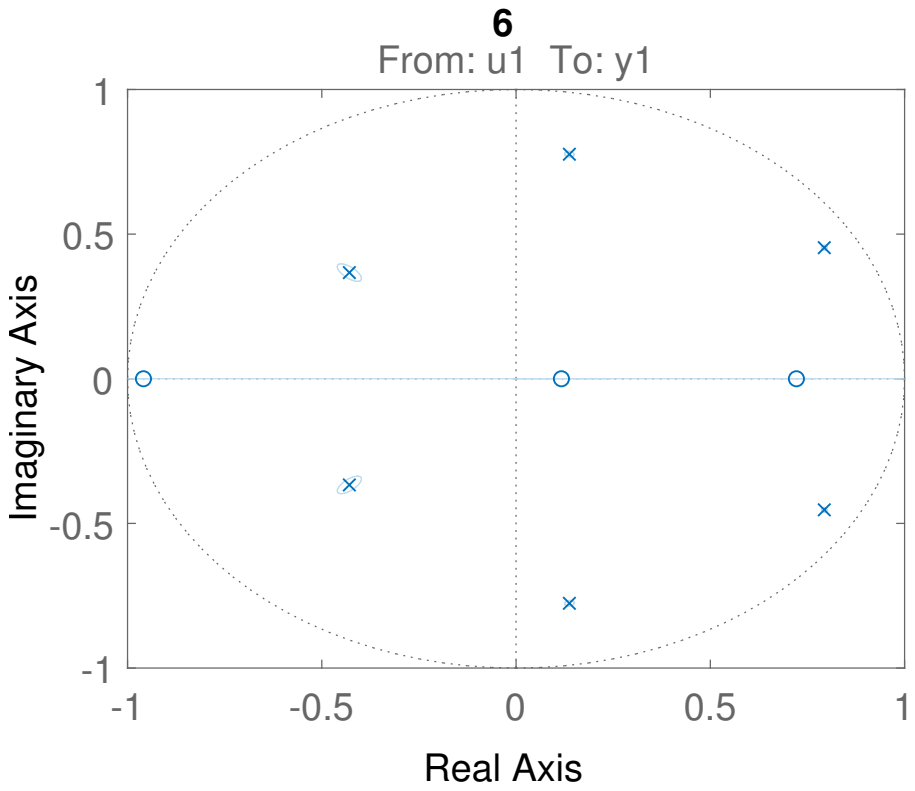

Figure 9: Zero-pole cancellation approach: Order-6

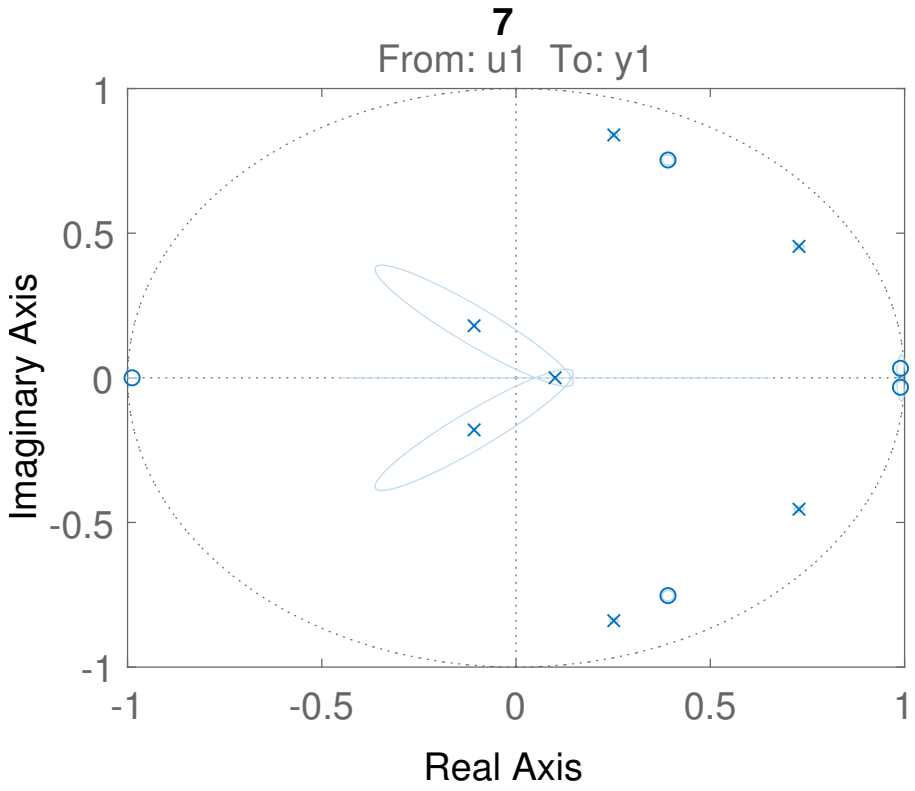

Figure 10: Zero-pole cancellation approach: Order-7

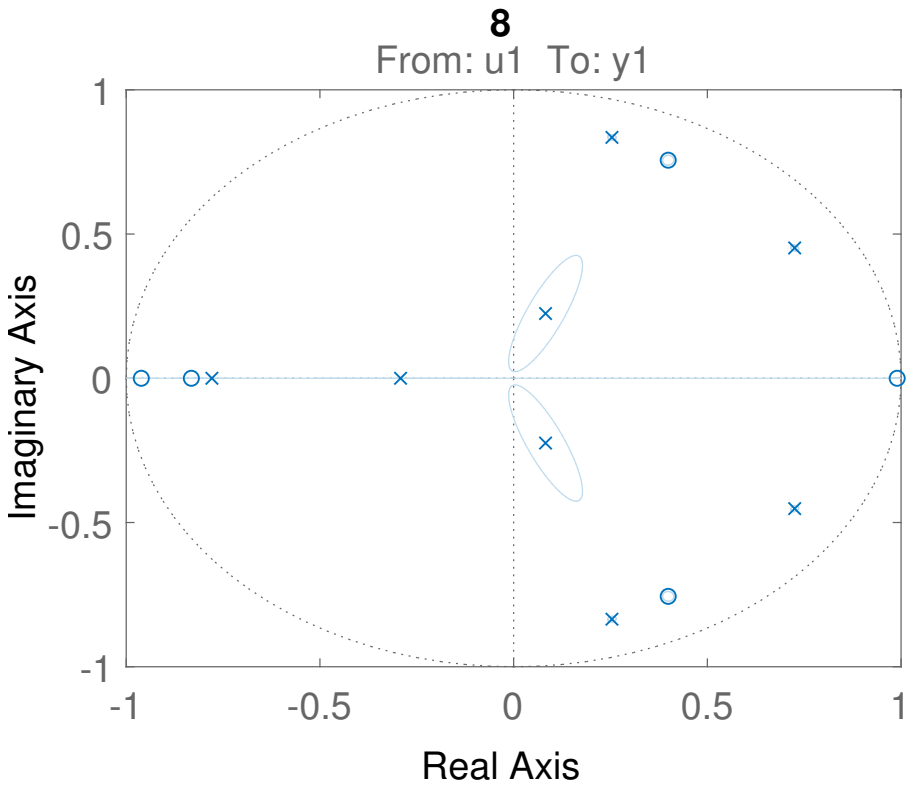

Figure 11: Zero-pole cancellation approach: Order-8

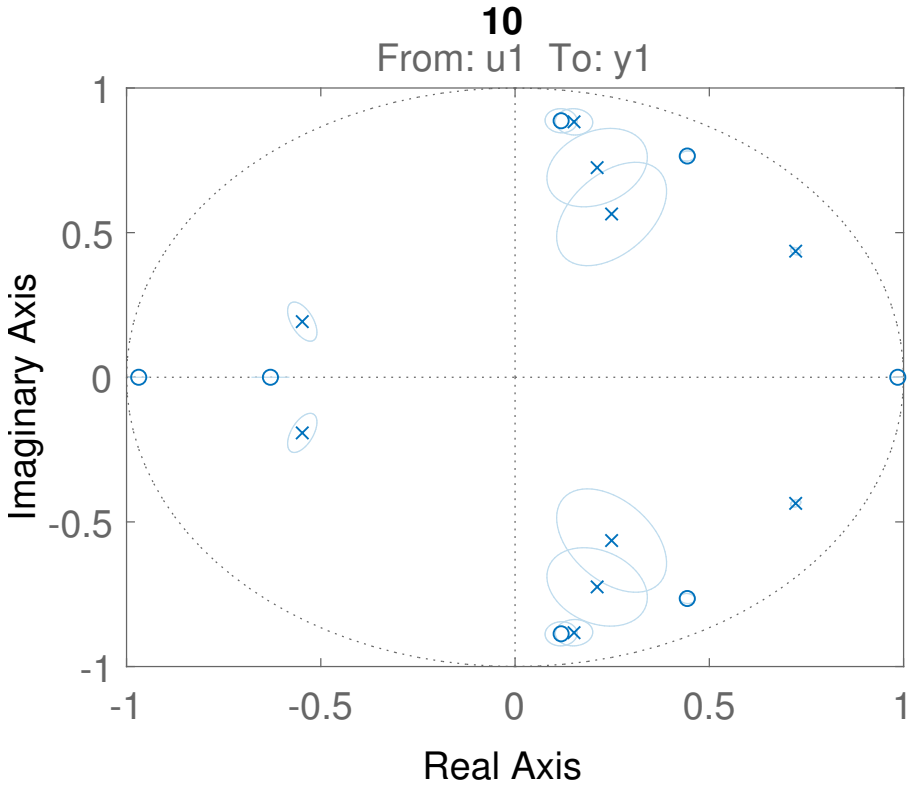

Figure 12: Zero-pole cancellation approach: Order-10

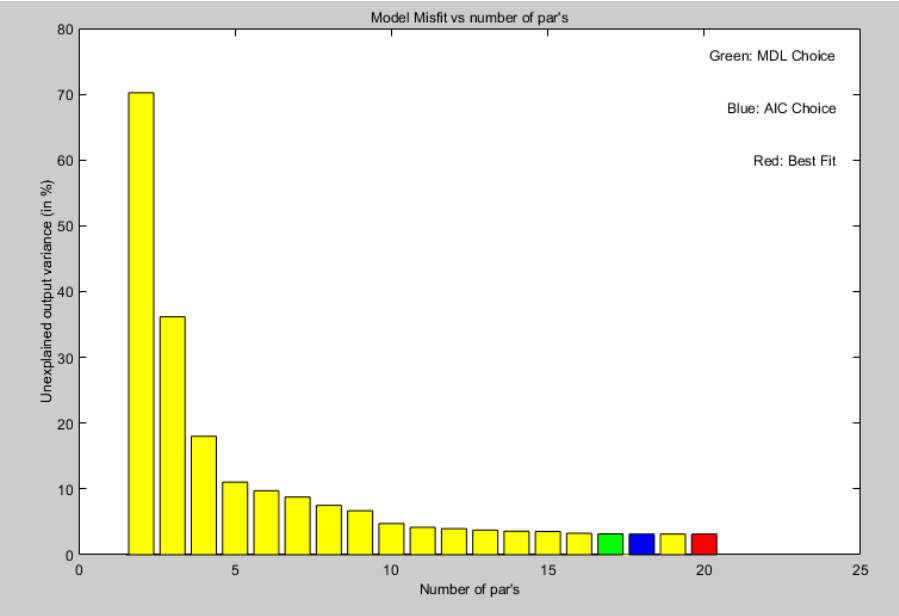

Figure 13: Number of parameters proposed by MATLAB selstruc=18

ARMAX model - estimation of delay

```
%% Order obtained using n = 7
Mamx = armax(Z, [7 7 7 1]);

Mamx.b
Mamx.db
```

From the output of M.b we find that that the second element is very close to zero and within its 2σ confidence interval (M.db). This implies that the first two elements in B are zero. Hence, we conclude that the delay of the system (nk) is 2.

Excluding the starting delay terms, we find that number of terms in B to be 7 -2 + 1 = 6

ARMAX model - Comparison with MATLAB selstruc

MATLAB code is used for identifying the model parameters. Firstly, we use 50 % of the data for estimation and remaining 50 percent for validation while using selstruc command in matlab. Grid search for the order between 1 and 10 is done, resulting variance explanation vs number of parameters as plotted by MATLAB selstruc command is shown in figure 13

```
%% Data for identification
% Data for estimation
N_id = round(0.5*N);
Z = iddata(Y.y(1:N_id), Y.u(1:N_id), 0.03);
Z = detrend(Z);

% Data for validation
Z_validation = iddata(Y.y(N_id+1:end), Y.u(N_id+1:end), 0.03);
Z_validation = detrend(Z_validation);

st = struc(1:10, 1:10, 1:3);
V = arxstruc(Z, Z_validation, st);
NN = selstruc(V, 'MDL');
selstruc(V, 'PLOT')
```

The final structure proposed by MATLAB using 'MDL' criteria is [8, 9, 2] where as with other criteria, it gives [10, 10, 2]. Hence, no clear structure can be obtained directly by MATLAB selstruc since it depends on selection criteria.

5 Parametric identification

The code for partitioning (we have a 80%, 20% partition, we could have chosen something else like 50% 50% but it doesn't conceptually make a difference and 20% 80% is what's common for machine learning so we are just using it out of habit) the data and identifying the different models is the following:

```
%% Load the data
Y = load('CE2.mat');
N = length(Y.u);
N_id = round(0.8*N);

%% Data for identification
Z = iddata(Y.y(1:N_id), Y.u(1:N_id), 0.03);
Z = detrend(Z);
% Data for validation
Z_validation = iddata(Y.y(N_id+1:end), Y.u(N_id+1:end), 0.03);
Z_validation = detrend(Z_validation);

%% System identification using ARX, iv4, armax, oe, bj, n4sid
% We select na = nb = 7 for ARX based on zero-pole cancellation and nk = 1
na=7; nb=na; nc=na; nd = na; nf = na; nk=1;
Marx = arx(Z, [na nb nk]);
Mamx = armax(Z, [na nb nc nk]);
Miv4 = iv4(Z, 'na', na, 'nb', nb, 'nk', nk);
Moe = oe(Z, [nb nf nk]);
Mbj = bj(Z, [nb nc nd nf nk]);
Mn4sid = n4sid(Z, na);

% Bode plot
figure(); hold('on');grid('on');
p = bodeoptions('cstprefs');
p.FreqScale = 'linear';
bode(Marx, p);bode(Mamx, p);bode(Miv4, p);bode(Moe, p);bode(Mbj, p);bode(Mn4sid,
    p);bode(sys_computed,p)
legend('Location', 'SouthWest')
%% Comparison of the models
figure();
compare(Z_validation, Marx, Mamx, Miv4, Moe, Mbj, Mn4sid);

%% Statistical tests
figure();
resid(Z_validation, Marx, Mamx, Miv4, Moe, Mbj, Mn4sid)
legend();
```

6 Model validation

The simulated output based on the different models along with the validation data are shown in figure 14. This is a zoomed-in plot of about 1 second. This figure clearly shows that all the models are superposed except the ARX model. This is also to be seen based on the fit. All methods achieve 78% fit whereas the ARX model only gets 75%. From the output model, it is thus not possible to chose a best model. We can however eliminate ARX.

Plots of the frequency response of all the identified models, both in linear and log scale, are shown in figure 16 and 17. The frequency response of the system identified by the non-parametric Fourier analysis is also shown. It was computed by averaging over the 4 periods of the 9-bit PRBS input

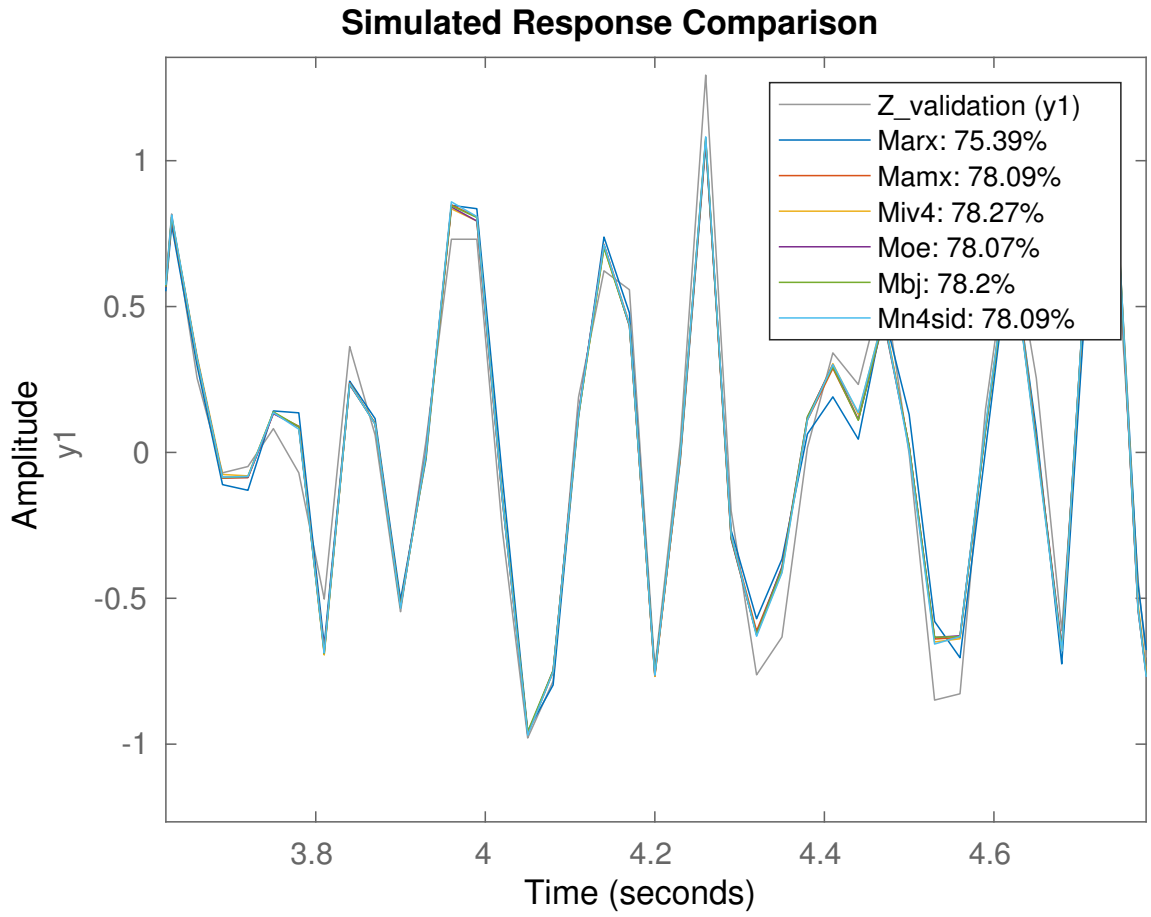

Figure 14: Output comparison of different identified models

signal. It is important to look at the bias distribution in the frequency domain. Recall that the optimal $\hat{\theta}$ is obtained by minimizing the sum of the squared residuals $\epsilon(k, \theta)$ at each k. Applying Parseval's relation for structures with noise models leads to:

$$\hat{\theta} = argmin \frac{1}{2\pi} \int_{-\pi}^{\pi} |H^{-1}(e^{j\omega})|^2 [|G_0(e^{j\omega}) - G(e^{j\omega})|^2 \Phi_{uu}(\omega) + |H_0(e^{j\omega}) - H(e^{j\omega})|^2 \Phi_{ee}(\omega)] d\omega \quad (5)$$

And for structures without noise models:

$$\hat{\theta} = argmin \frac{1}{2\pi} \int_{-\pi}^{\pi} [|G_0(e^{j\omega}) - G(e^{j\omega})|^2 \Phi_{uu}(\omega) + \Phi_{nn}(\omega)] d\omega \quad (6)$$

Recall that the input signal is a PRBS and that its spectral density function $\Phi_{uu}(\omega)$ is uniform across the spectrum except at low frequencies. The identification error $G_0(e^{j\omega}) - G(e^{j\omega})$ will thus be much higher at very low frequencies. This is also to be seen on the bode plots. The linear bode plot was given in order not to highlight the low frequencies too much and indeed, the "fit" looks much better on the linear scale.

For structures with noise models like ARX, ARMAX and BJ, the identification error $G_0(e^{j\omega}) - G(e^{j\omega})$ will higher where the magnitude of the inverse noise model is low. Fortunately, the inverse noise models of these structures are not high pass so they don't make the bad identification in low frequencies worse. The bode plot of the inverse noise model of the identified ARMAX structure can be found in figure 15.

Based on the frequency response, it it again hard to distinguish a clear winner. However, it is clear that ARX and BJ, don't follow the peaks between 15 and 45 Hz as well as the other structures. It can

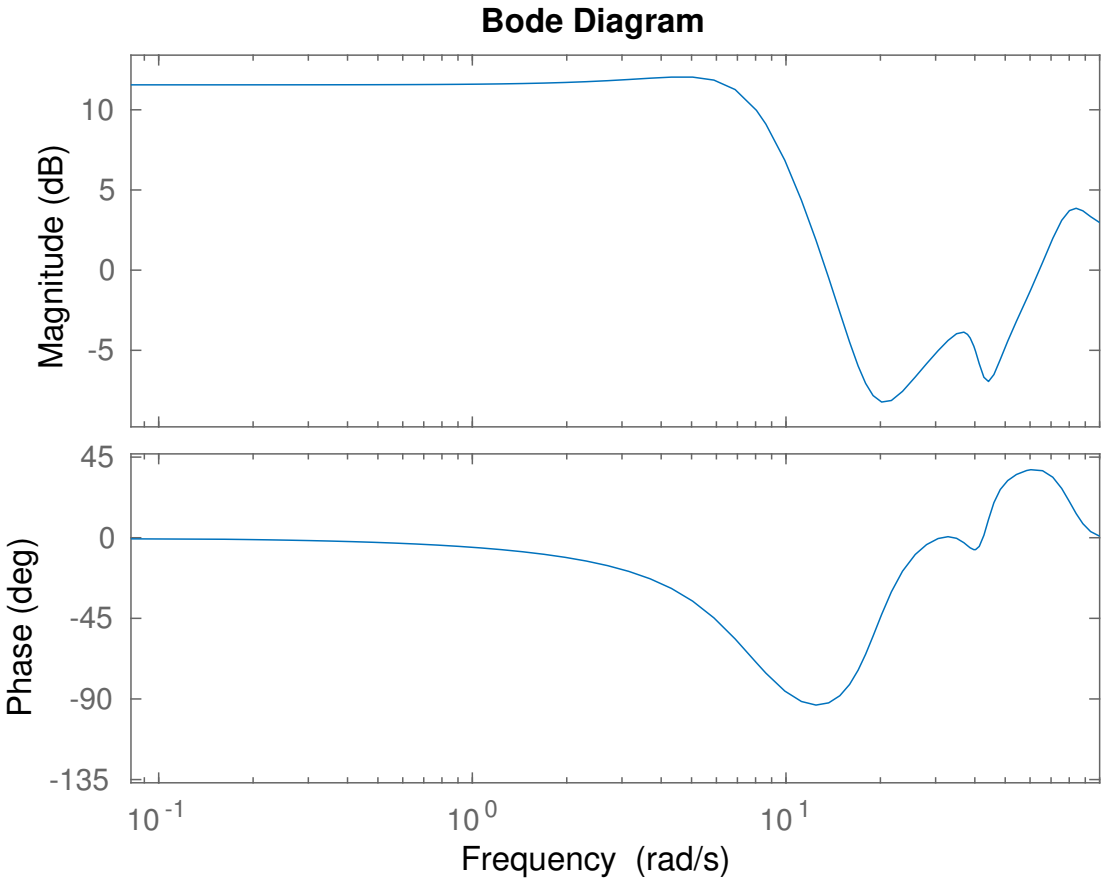

Figure 15: Frequency response of the inverse noise model of the identified ARMAX

also be noted that IV and OE don't have the same behaviour has the non-parametric estimation in the low frequencies of the phase diagram. It seems like the non-parametric estimation either has a DC phase of 180° or 0°. IV and OE have DC phase of 360° (which is 0°, so there is no difference for DC, but for low frequencies, there is a phase difference).

Structures with noise models

For the statistical validation tests, we will first have a look at the whiteness test of the residuals. This whiteness only applies to structures with noise models, in our case: ARX, ARMAX and BJ. ARX and ARMAX don't go out of the 99% confidence interval. BJ does slightly go out of the 99% confidence interval. However, we could have taken a less demanding confidence interval of for instance 2 σ (95%) and BJ would have stayed in. To us, all structures with noise models pass the whiteness test. The output of the $resid()$ function for the models with noise can be found in figure 18.

The second statistical validation test is the cross-correlation test. If there exists a substantial correlation between the past inputs and the residual, then it means that some information about the true model has not been captured. It is clear that the ARX model escapes the 99% confidence interval and for this reason, we will discard it. Note that it is not surprising as it is the simplest model of the 3. The ARMAX and BJ don't go out of the confidence interval except at 1 particular lag value. Again, the confidence interval of 99% was randomly chosen and for this reason, ARMAX and BJ pass the cross correlation test.

Structures without noise models

The output of the $resid()$ function for the models without noise can be found in figure 19. Note that only the right side is relevant here. All of the models only slightly leave the 99% confidence interval,

and therefore all of them pass the cross correlation test.

Conclusion

- output comparaison leads to elimination of ARX
- frequency response leads to elimination of ARX, BJ, OE and IV
- whiteness test leads to elimination of ARX
- cross correlation is satisfied for all

Overall, the state space and ARMAX seem to be the best. Their fit is not as high as BJ or IV but these two were eliminated due to other reasons. Out of the 2, we don't consider a clear winner. ARMAX is the best out of the structures with noise models, state space is the best out of the structures without noise models.

7 Conclusion

The system identification of a real output with noise is done using various parametric methods (as opposed to nonparametric) presented in class. We note the following differences between the 2 based on our results of CE1 and CE2 (see slides chapter 2, 2/41):

- The identified models are much less noisy using the parametric models as the measurement noise is filtered out in the optimization procedure. This is clearly visible from all the plots in the 2 reports.
- The optimization procedure indeed takes some non trivial CPU time. This was clearly noticeable for exercise 6 when all the different parametric models were optimized individually.

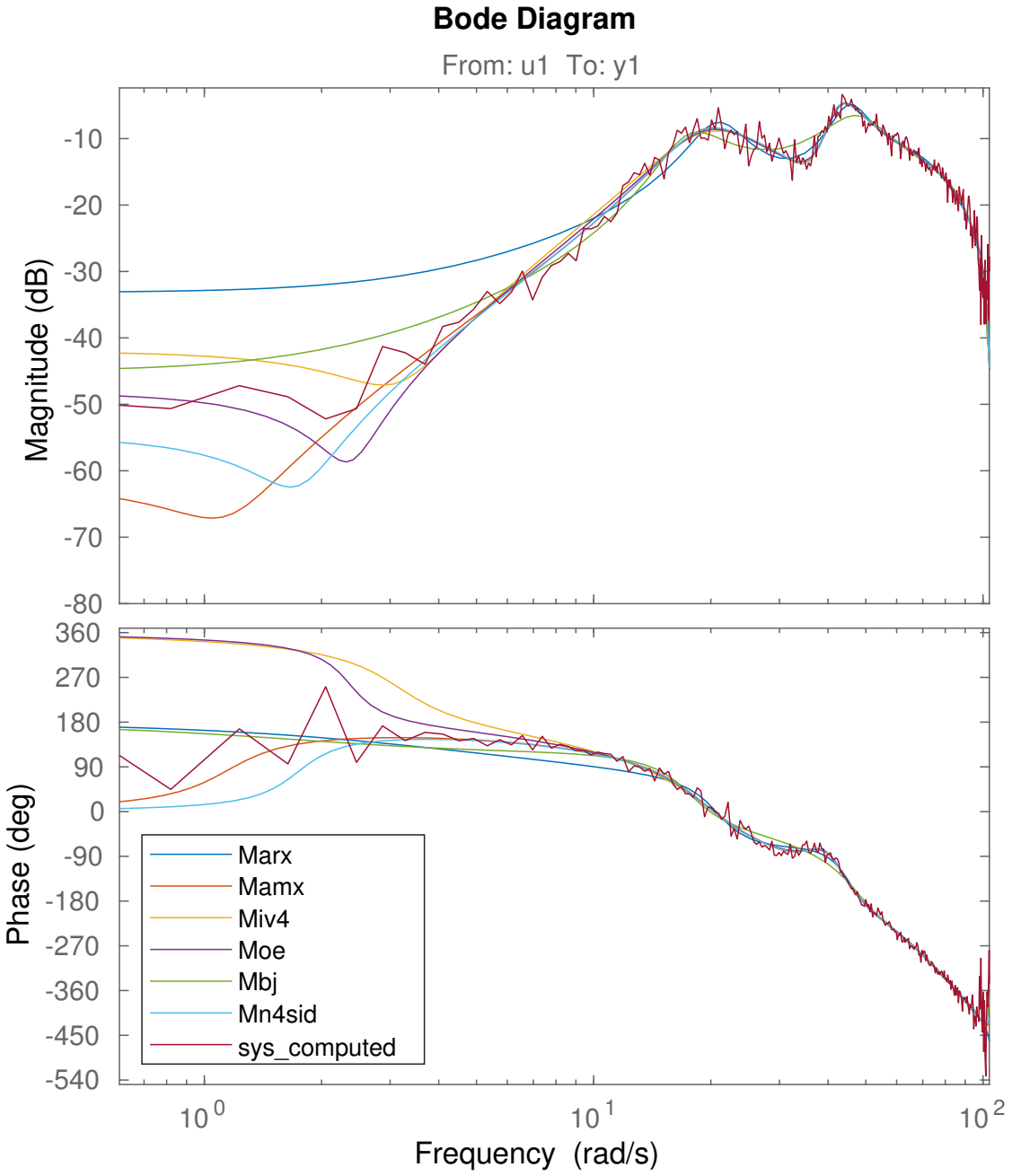

Figure 16: Frequency response of different identified models

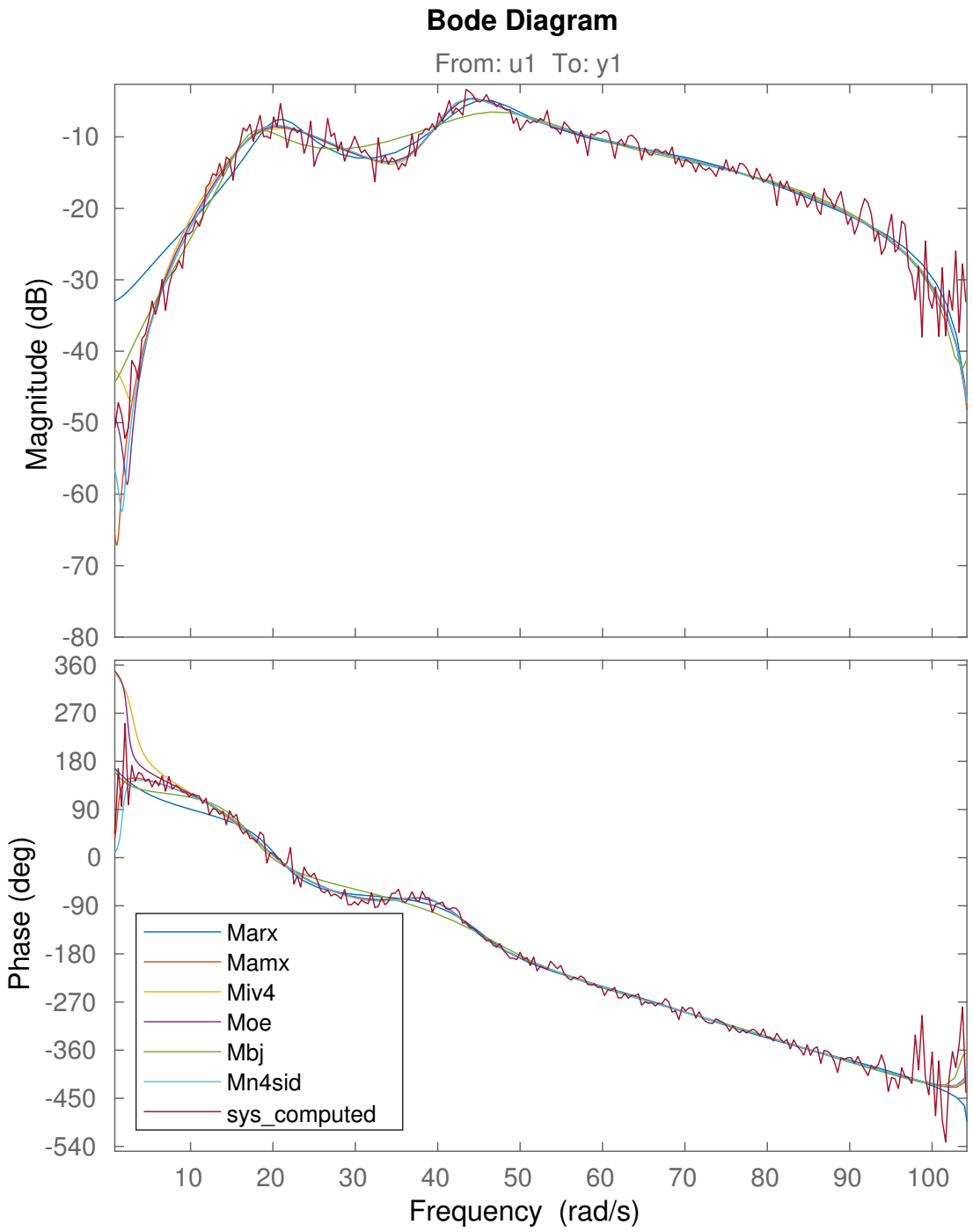

Figure 17: Frequency response of different identified models with linear scale

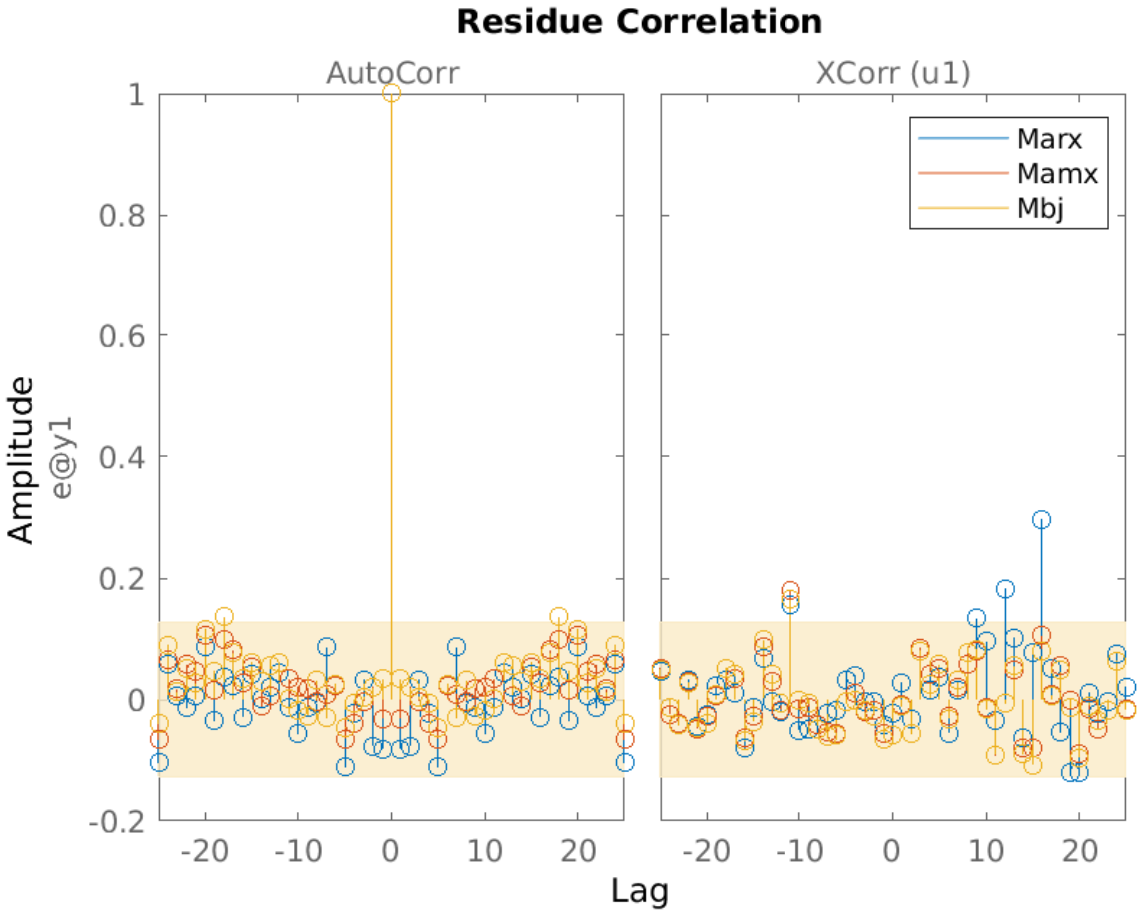

Figure 18: Statistical validation tests for structures with noise models

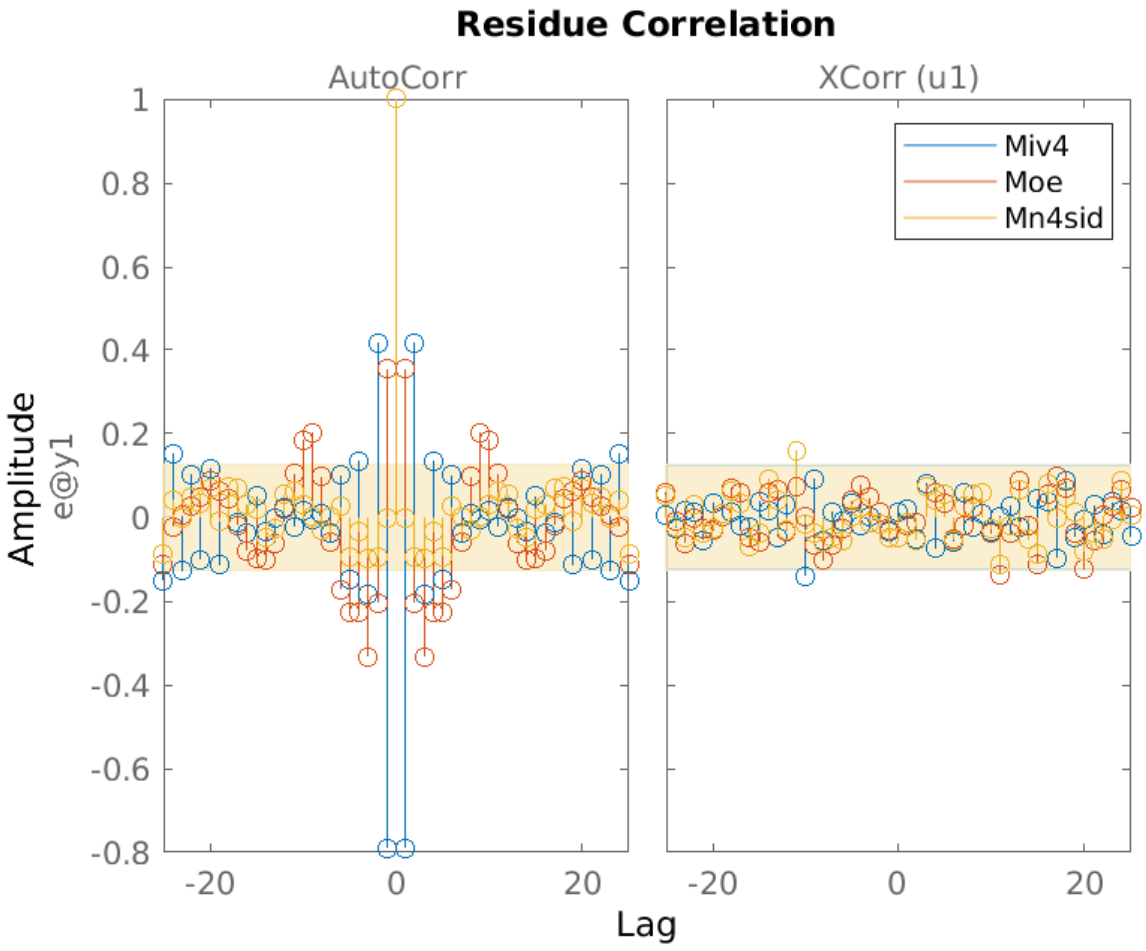

Figure 19: Statistical validation test for structures without noise models