Systemidentifikation und Regelung in der Medizin

8. Vorlesung

8. Einführung in die System Identification Toolbox anhand von Beispielen

Sommersemester 2020

8. Juni 2020

Thomas Schauer

Technische Universität Berlin
Fachgebiet Regelungssysteme

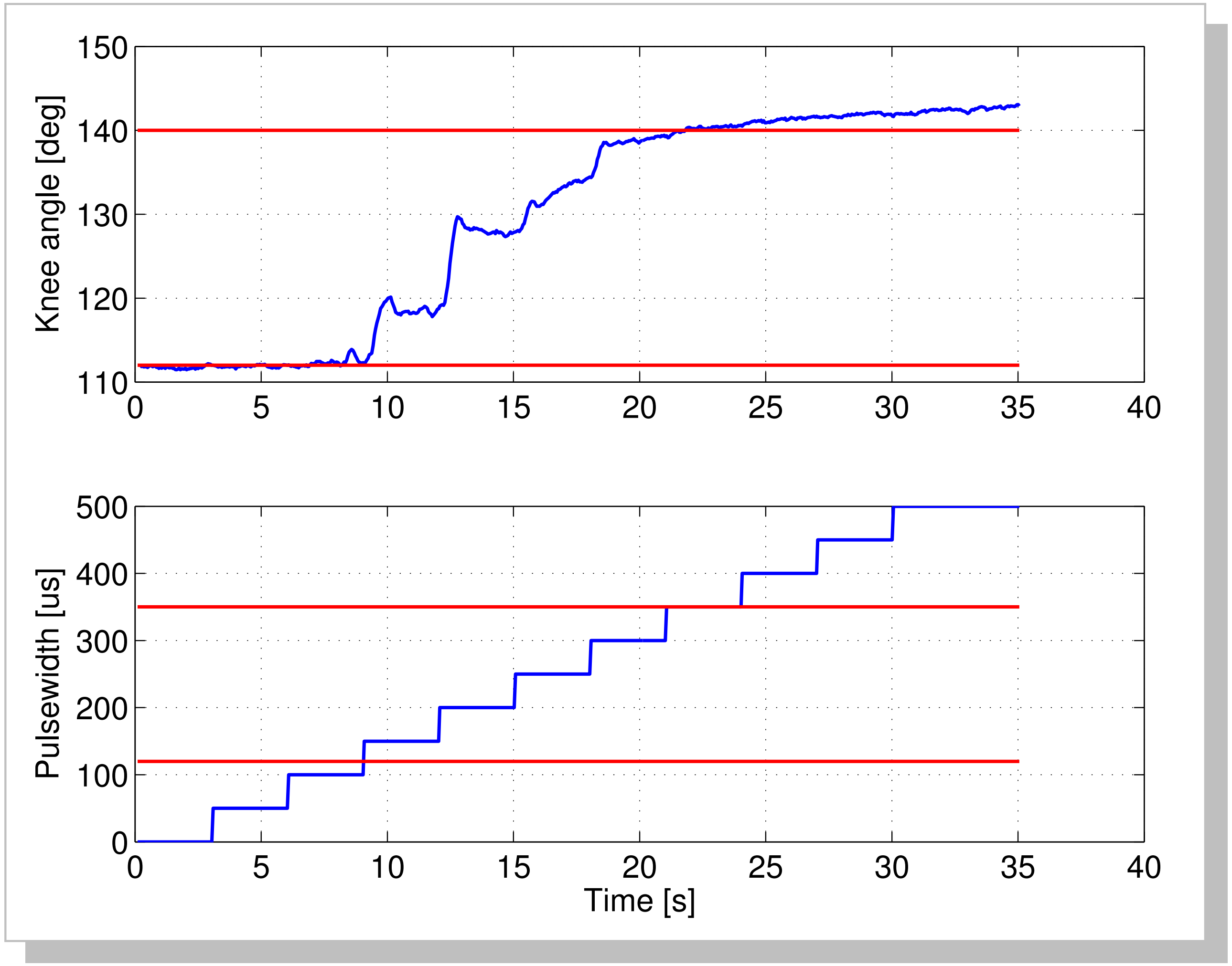
Identification of the Electrically Stimulated Quadriceps Muscle Group

- *System*: pulse width - input u, angle - output y
- *Pre-test to find operating regime and to normalise inputs and outputs*
- *Normalised output*: $y = 0$: rest angle, $y = 1$: knee full extended
- *Normalised Input*: $u = 0$: threshold pulsewidth, $u = 1$: saturation pulsewidth
- *Design of a PRB signal*: `prbs` (own routine) or `idinput` (System Identification Toolbox)
- *System Identification test*

  Hint: For Identification Toolbox: $na = \deg \mathsf{A}$, but $nb = \deg \mathsf{B} + 1$

  Hint: For Identification Toolbox and BJ-Modell: $\mathsf{A} = \mathsf{F}$
- *Extract model for controller design*

Pre-Test

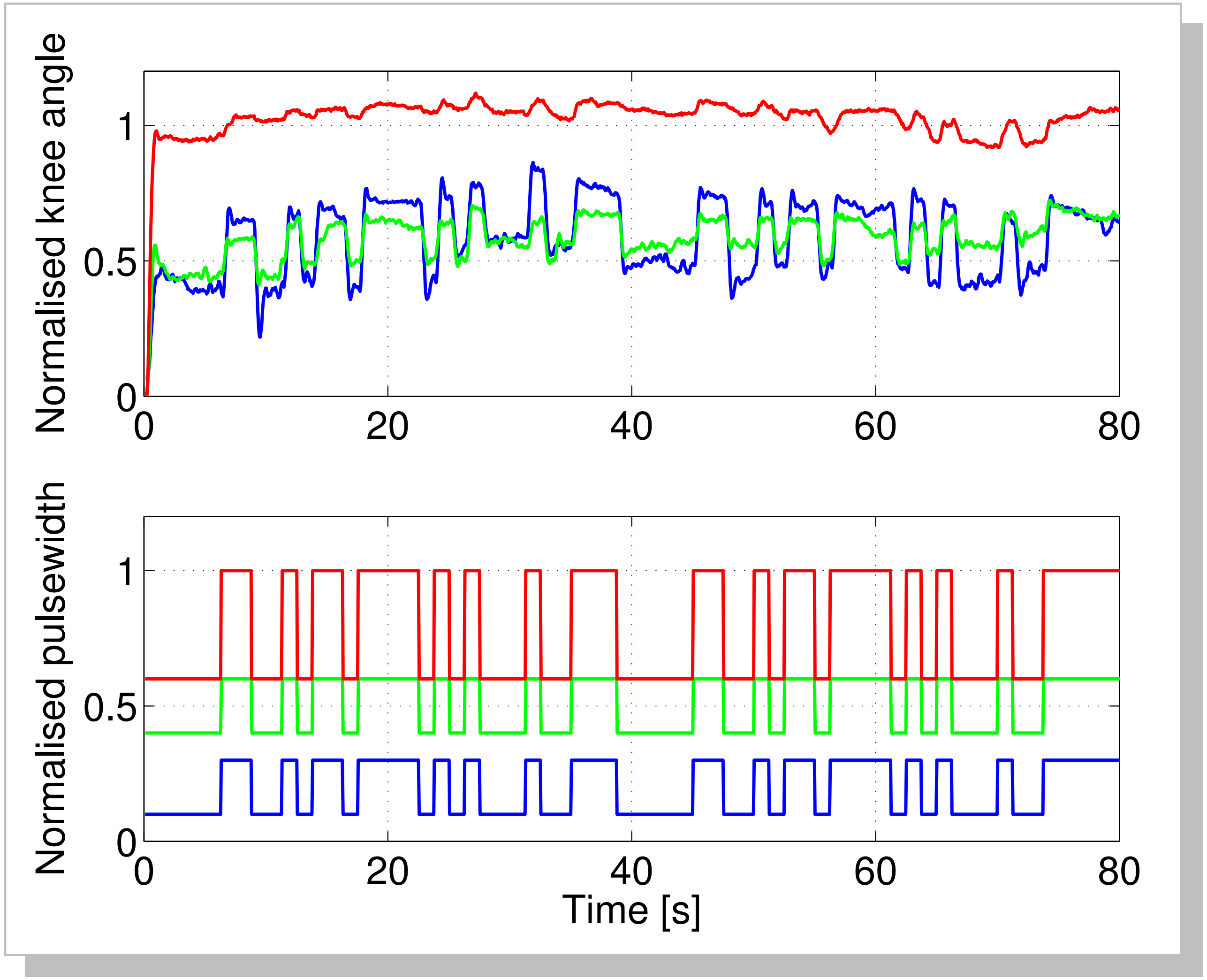


150
140
130
120
110
Knee angle [deg]
0
5
10
15
20
25
30
35
40
500
400
300
200
100
0
Pulsewidth [us]
Time [s]

Design PRBS - Matlab Script

```
%N - order
N=5;
%Ts - Sampling time
Ts=0.05;
%p - frequency divider (integer)
p=25;
%n - number of repetitions
n=2;
Mean=0.2;
Amp=0.1;

[t,u]=prbs(N,p,n,Ts,Mean,Amp);
```

3 Local Identification Experiments



Preprocess data 1st data set (mean $u = 0.2$)

First split data set into estimation (training) $(10 \leq t \leq 40\,\mathrm{s})$ and validation (testing) part $(45 \leq t \leq 80\,\mathrm{s})$

```
%load data
load data.mat

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%              Chose estimation and validation data set               %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
h=slide_figure;
plot(t,y,t,u);
xlabel time
ylabel output
Ts=0.05; %sample time
index_est=find((t>=10)&(t<=40));
index_val=find((t>=45)&(t<=80));
```

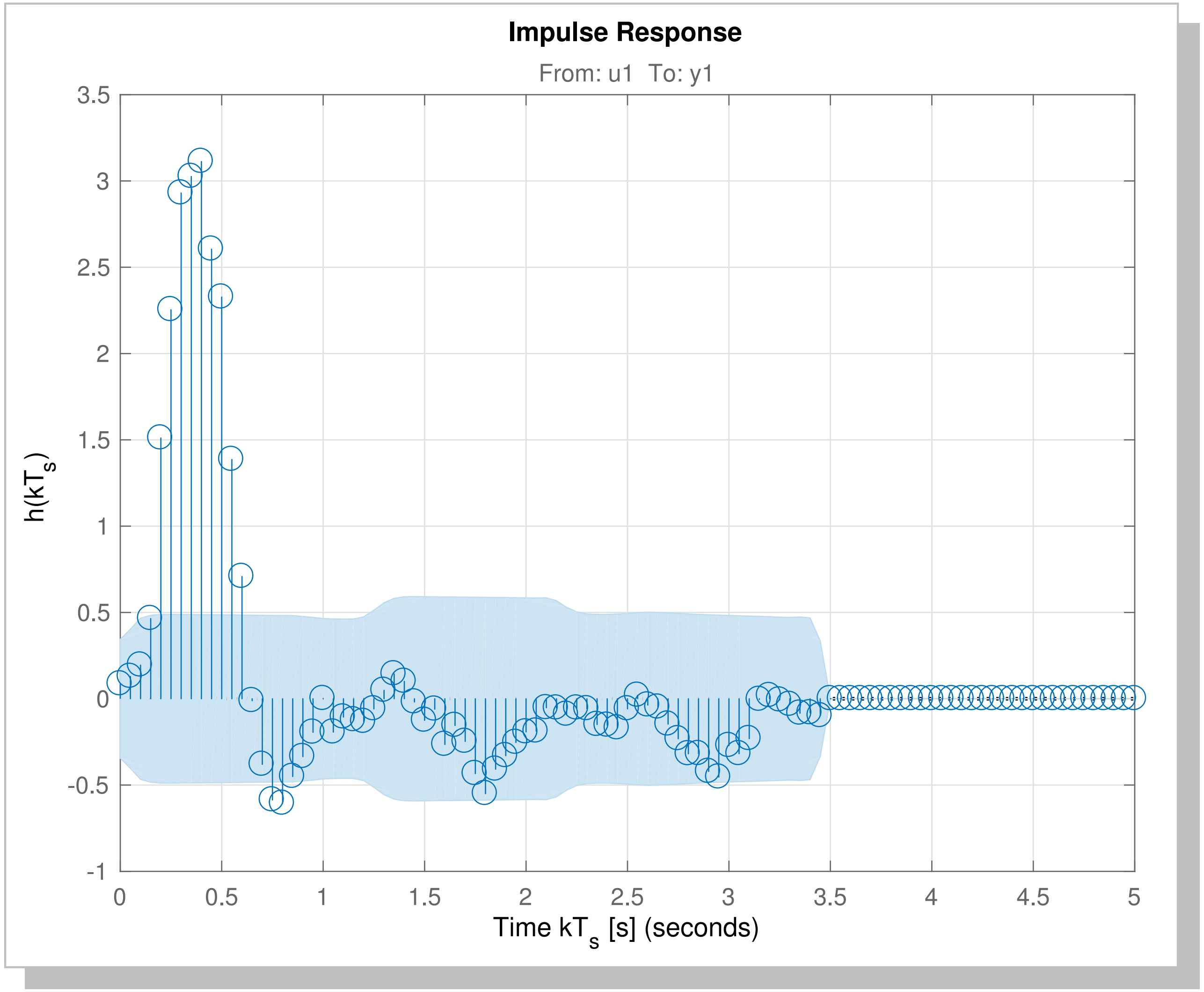
Remove offsets, normalize I/O data, create training and testing data set

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  Remove offsets and normalize to standard deviation of 1              %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
slide_figure;
u_mean=(max(u(index_est))+min(u(index_est)))/2;
y_mean=mean(y(index_est));
y_std=std(y(index_est));
u_std=std(u(index_est));
y_norm_est=(y(index_est)-y_mean)/y_std;
u_norm_est=(u(index_est)-u_mean)/u_std;

y_norm_val=(y(index_val)-y_mean)/y_std;
u_norm_val=(u(index_val)-u_mean)/u_std;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% create training and testing data set                                  %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Z_est=iddata(y_norm_est,u_norm_est,Ts);%estimation (train) data
Z_val=iddata(y_norm_val,u_norm_val,Ts);%validation (test) data
```

Guess time delay n_k from finite impulse response

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%           Estimate impulse response (FIR) to find n_k                %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

slide_figure;
sys = impulseest(Z_est)
h=impulseplot(sys);
sd = 3;
showConfidence(h,sd);
xlabel 'Time kT_s [s]';
ylabel 'h(kT_s)';
title 'Impulse Response';
grid on;
```

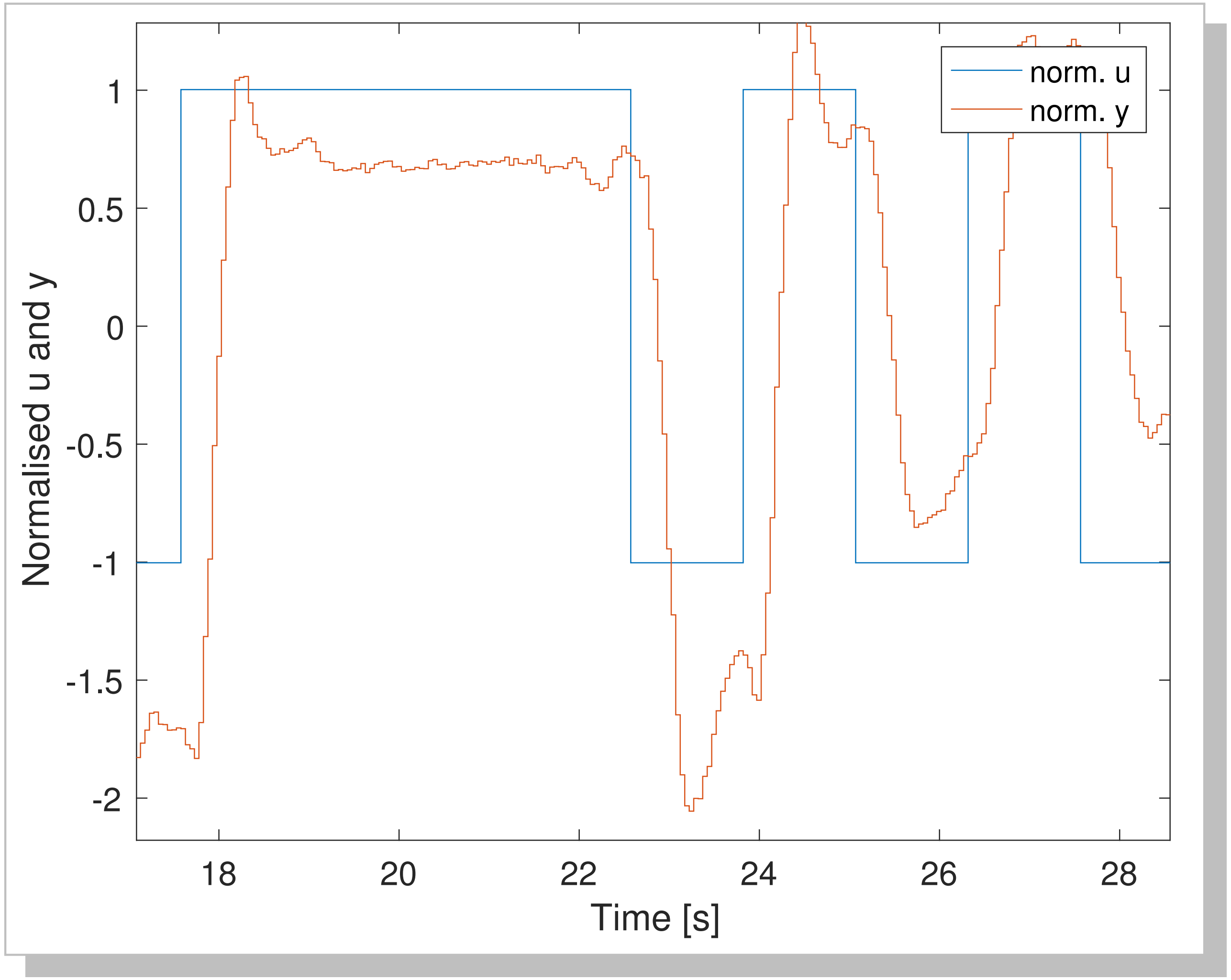
FIR for 1st est. data set (mean $u = 0.2$)



Guess for time delay: $n_k = 4$

Stairs plot to estimate time delay n_k

```
slide_figure;
stairs(t(index_est),[u_norm_est,y_norm_est]);
xlabel('Time [s]');
ylabel('Normalised u and y')
legend('norm. u','norm. y');
```



Guess for time delay: $n_k = 4$

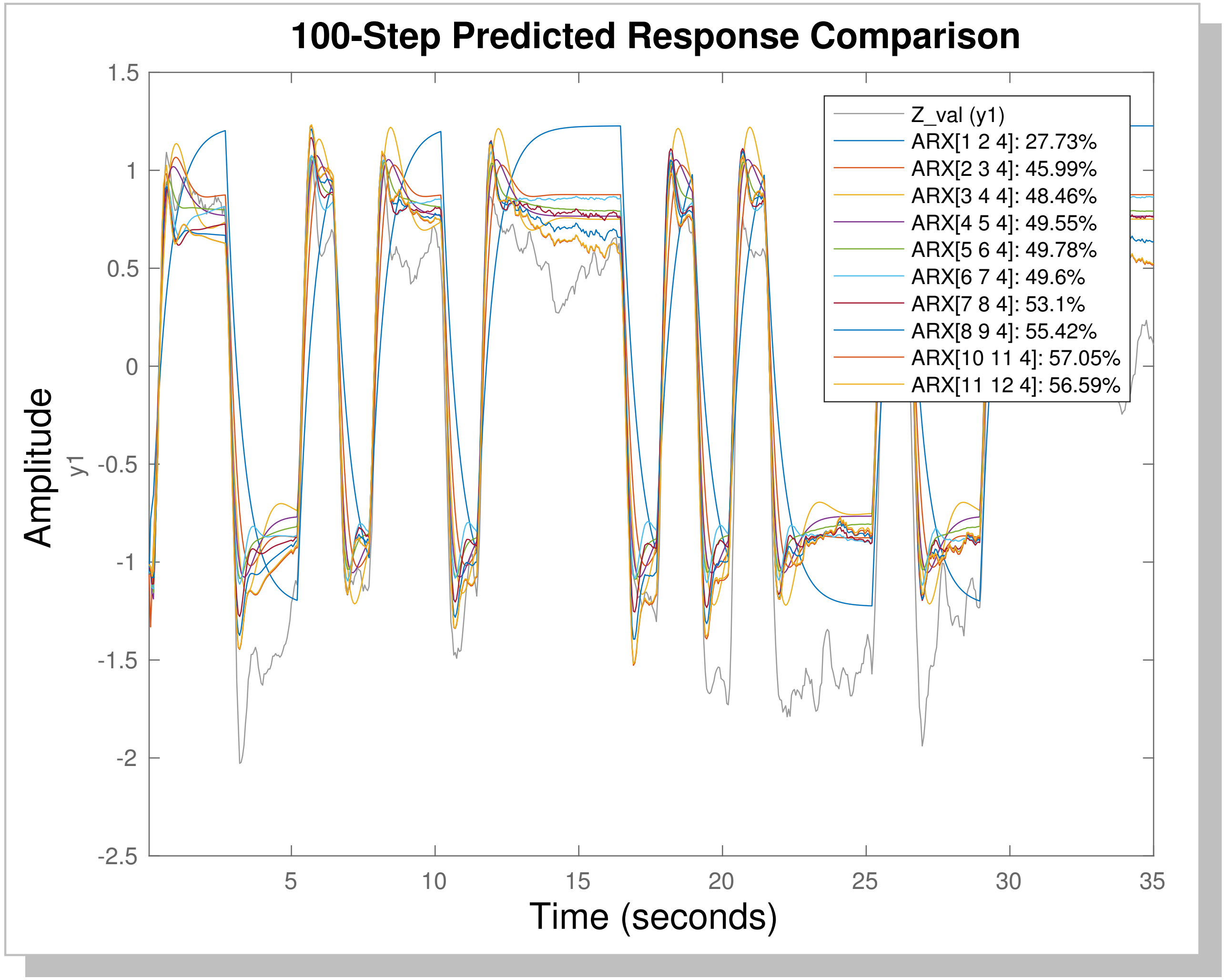
Estimate ARX models

```
  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  Choose ARX model
%  Identify different arx models and use cross validation to select
%  the best model having the best 100-step prediction fit
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%test 10 arx models with n_k=4
%na nb nk

struc=[1 2 4
       2 3 4
       3 4 4
       4 5 4
       5 6 4
       6 7 4
       7 8 4
       8 9 4
       10 11 4
       11 12 4];
```

```
Opt = arxOptions;
Opt.InitialCondition='auto';
Opt.Focus='prediction';
for i=1:10,
    [m(i).arxmodel]=arx(Z_est,struc(i,:),'Ts',Ts,...
      'initialstate','estimate','name',strcat('ARX',mat2str(struc(i,:))));
end
slide_figure;
compare(Z_val,m.arxmodel,100);
```

Cross validation to find right model order

Select model with best fit for 100-step prediction using the validation data set.



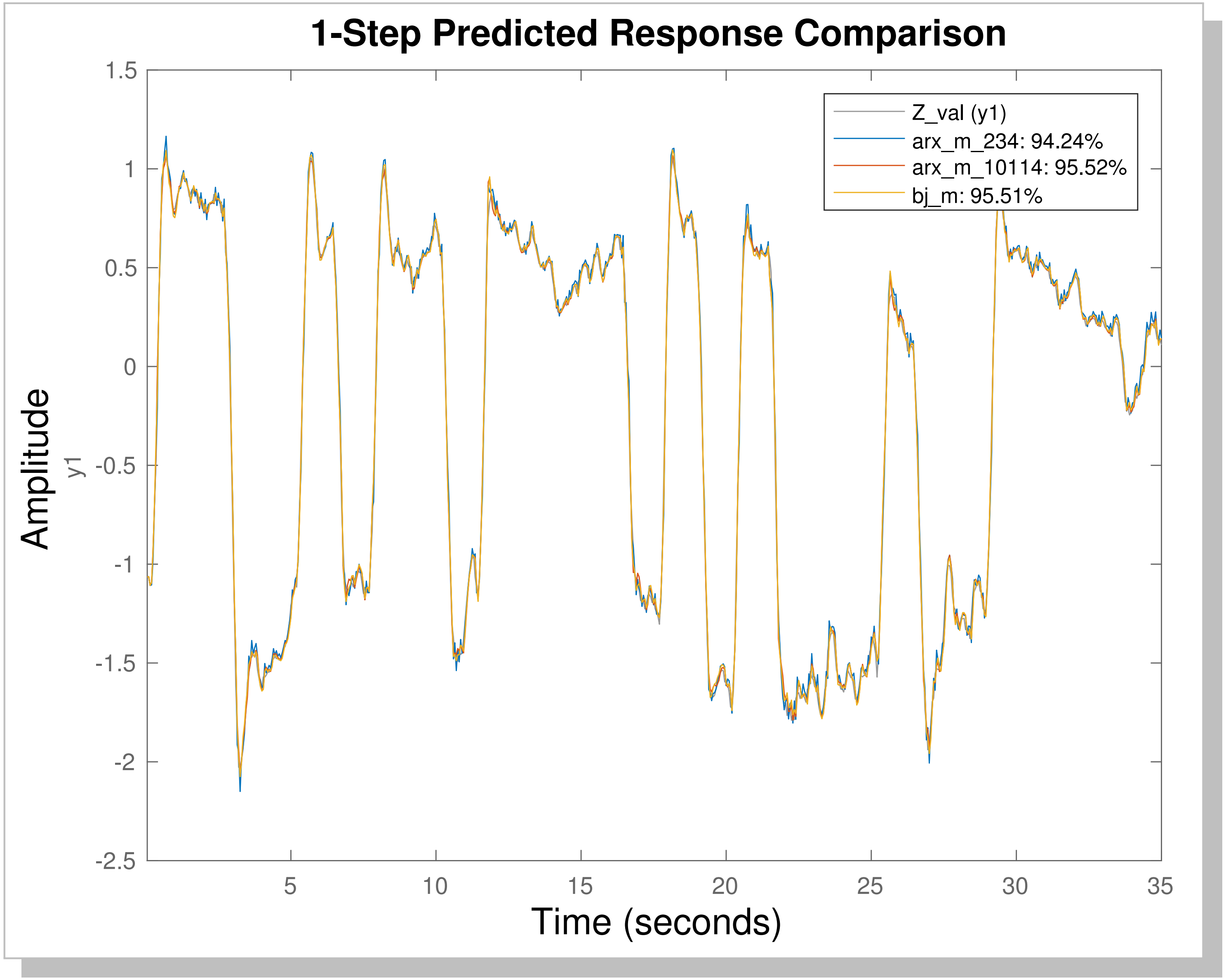
Best ARX model: na=10,nb=11,nk=4 (10th order model)

Try BJ-Model with lower order G instead

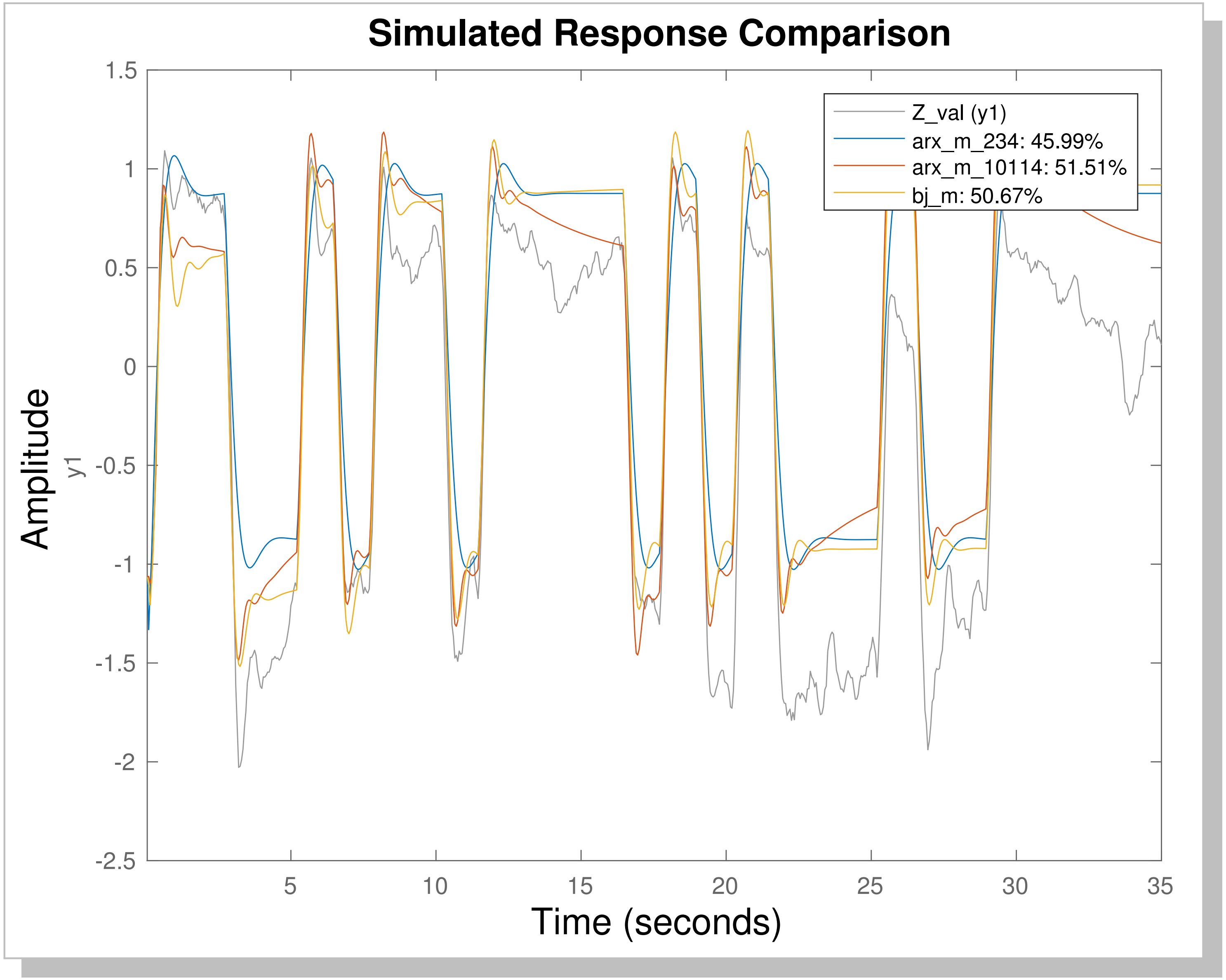
$$y(k) = \frac{\mathsf{B}(q^{-1})}{\mathsf{F}(q^{-1})}u(k-n_k) + \frac{\mathsf{C}(q^{-1})}{\mathsf{D}(q^{-1})}e(k)$$

```
%ARX: na nb nc
[arx_m_10114]=arx(Z_est,[10 11 4],'Ts',Ts,'initialstate','estimate');
[arx_m_234]=arx(Z_est,[2 3 4],'Ts',Ts,'initialstate','estimate');
%BJ: nb nc nd nf nk
[bj_m]=bj(Z_est,[3 2 4 2 4],'Ts',Ts,'initialstate','estimate');
slide_figure;
compare(Z_val,arx_m_234,arx_m_10114,bj_m,1); %1step
slide_figure;
compare(Z_val,arx_m_234,arx_m_10114,bj_m,inf); %simulation
```

Comparison 1-step prediction



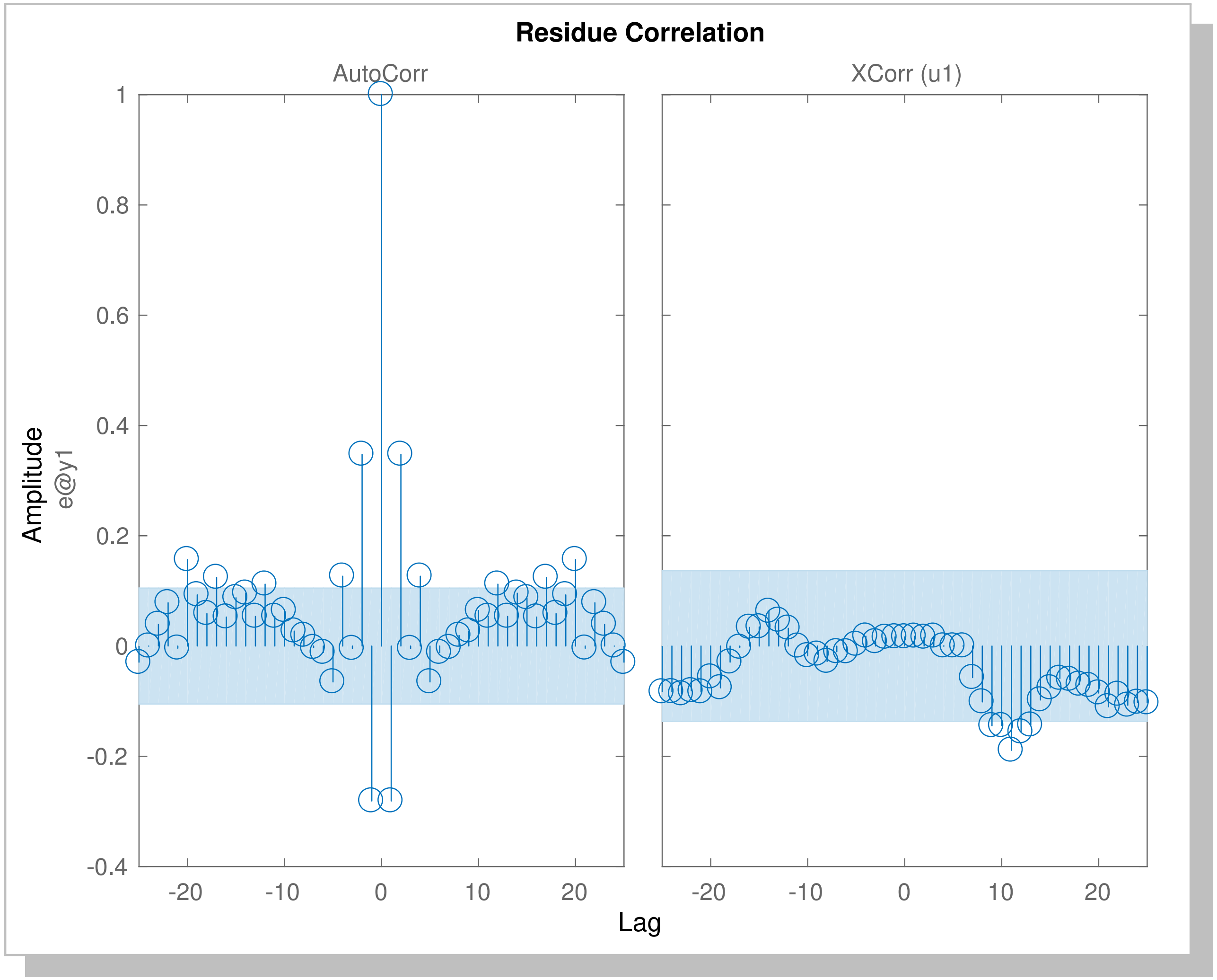
Comparison simulation



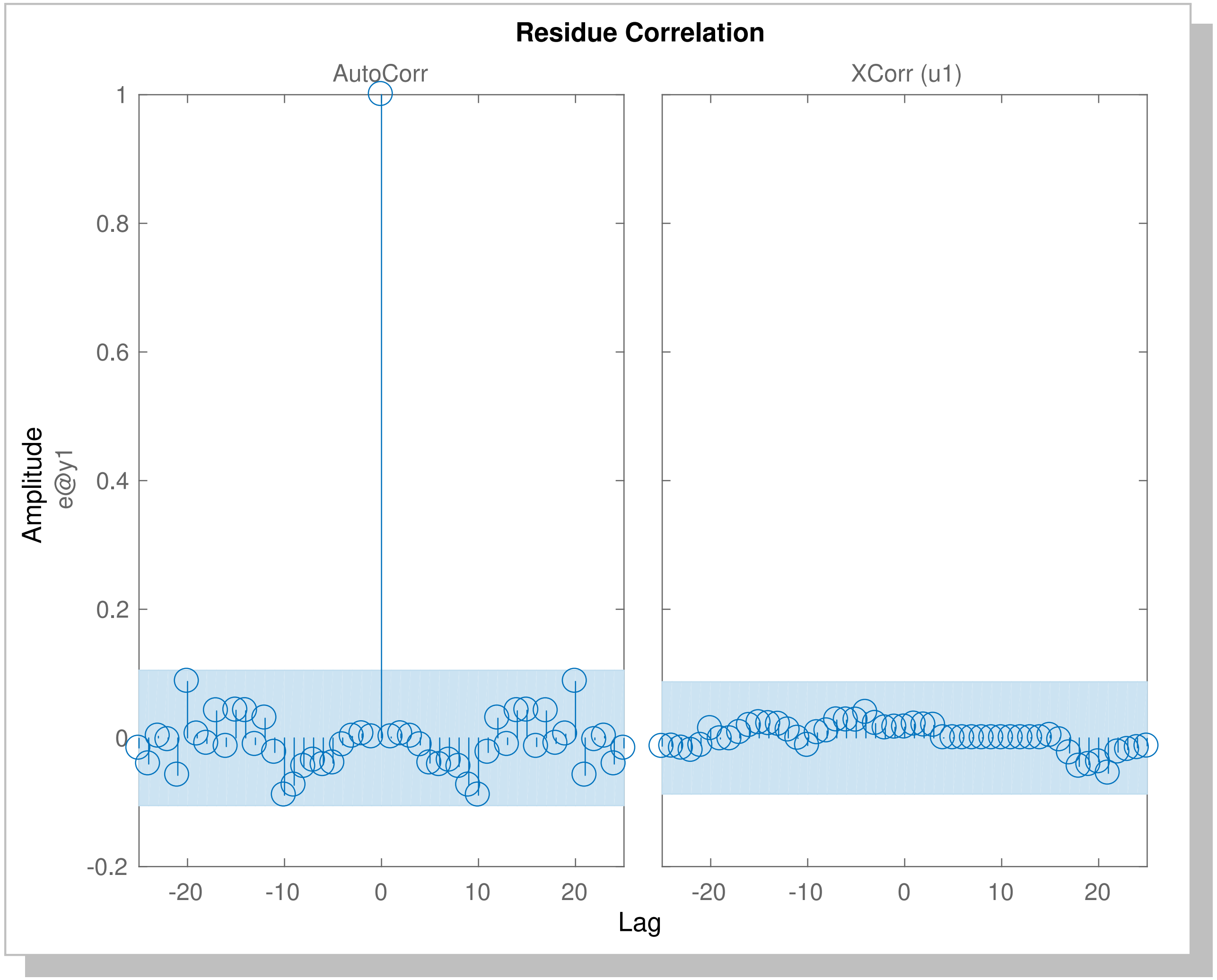
Prediction error analysis

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%               Prediction Error Analysis                                %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

slide_figure;
resid(Z_est,arx_m_234);
slide_figure;
resid(Z_est,arx_m_10114);
slide_figure;
resid(Z_est,bj_m);
```

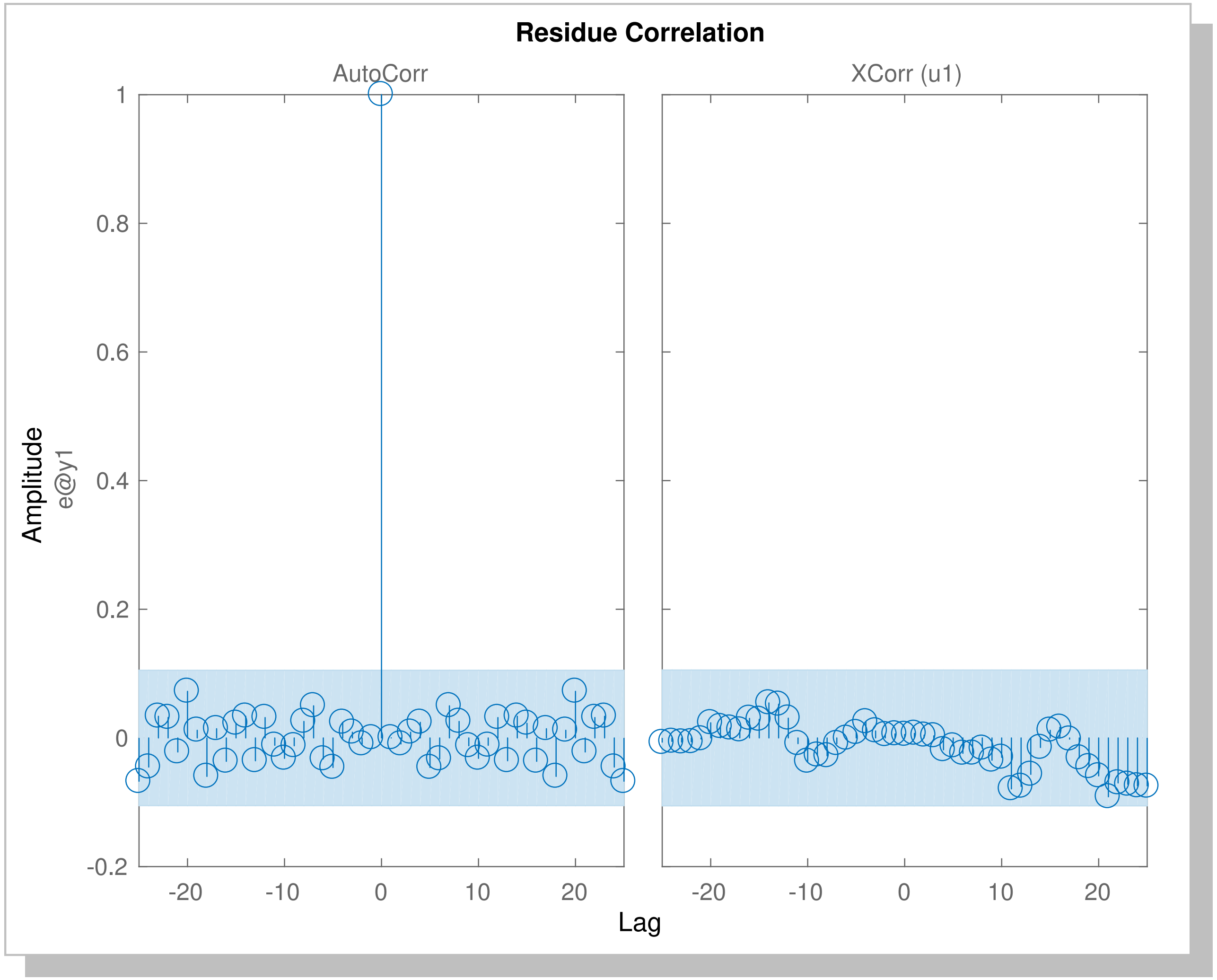
Prediction error analysis (ARX 234)



Prediction error analysis (ARX 10114)



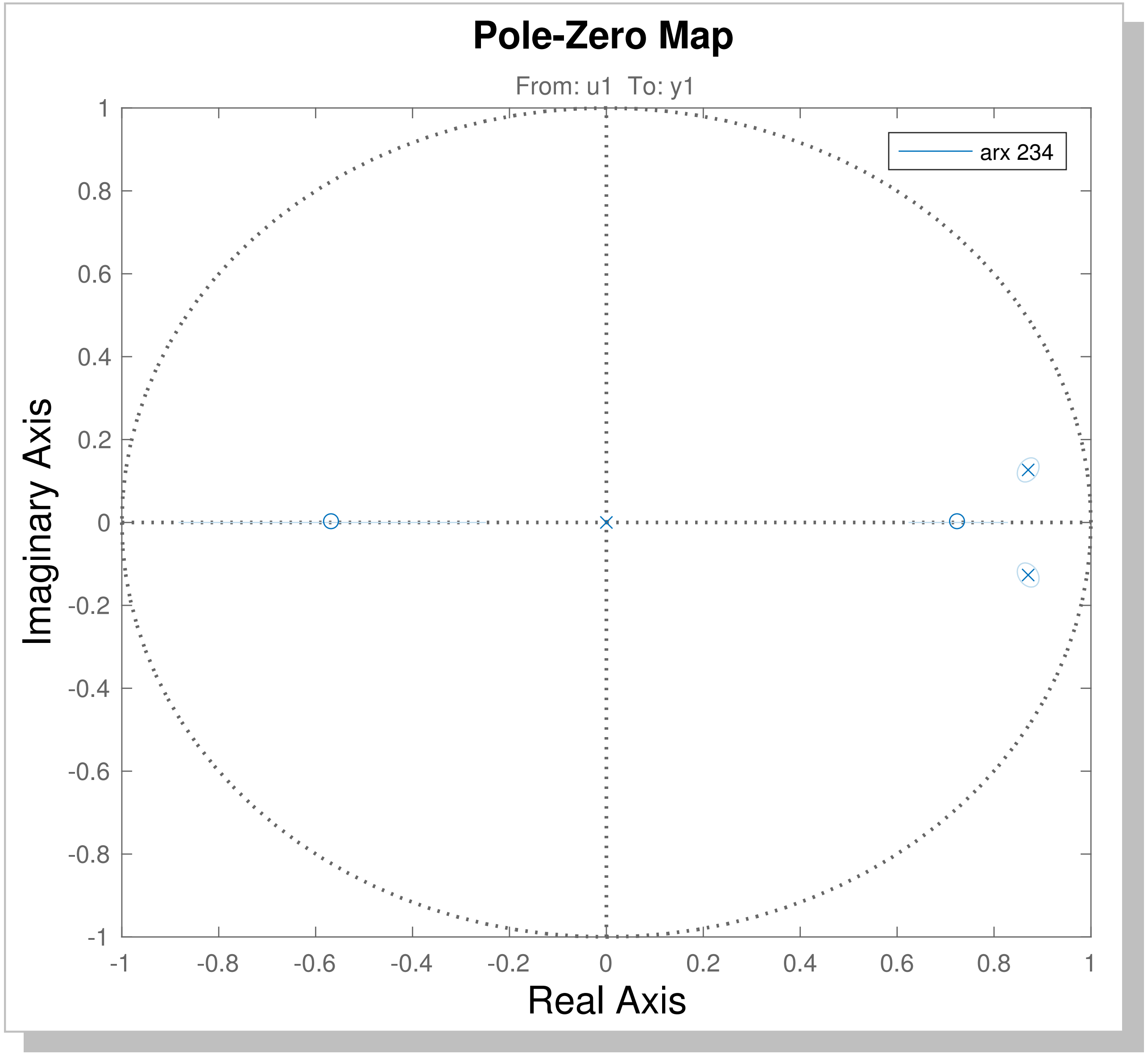
Prediction error analysis (BJ)



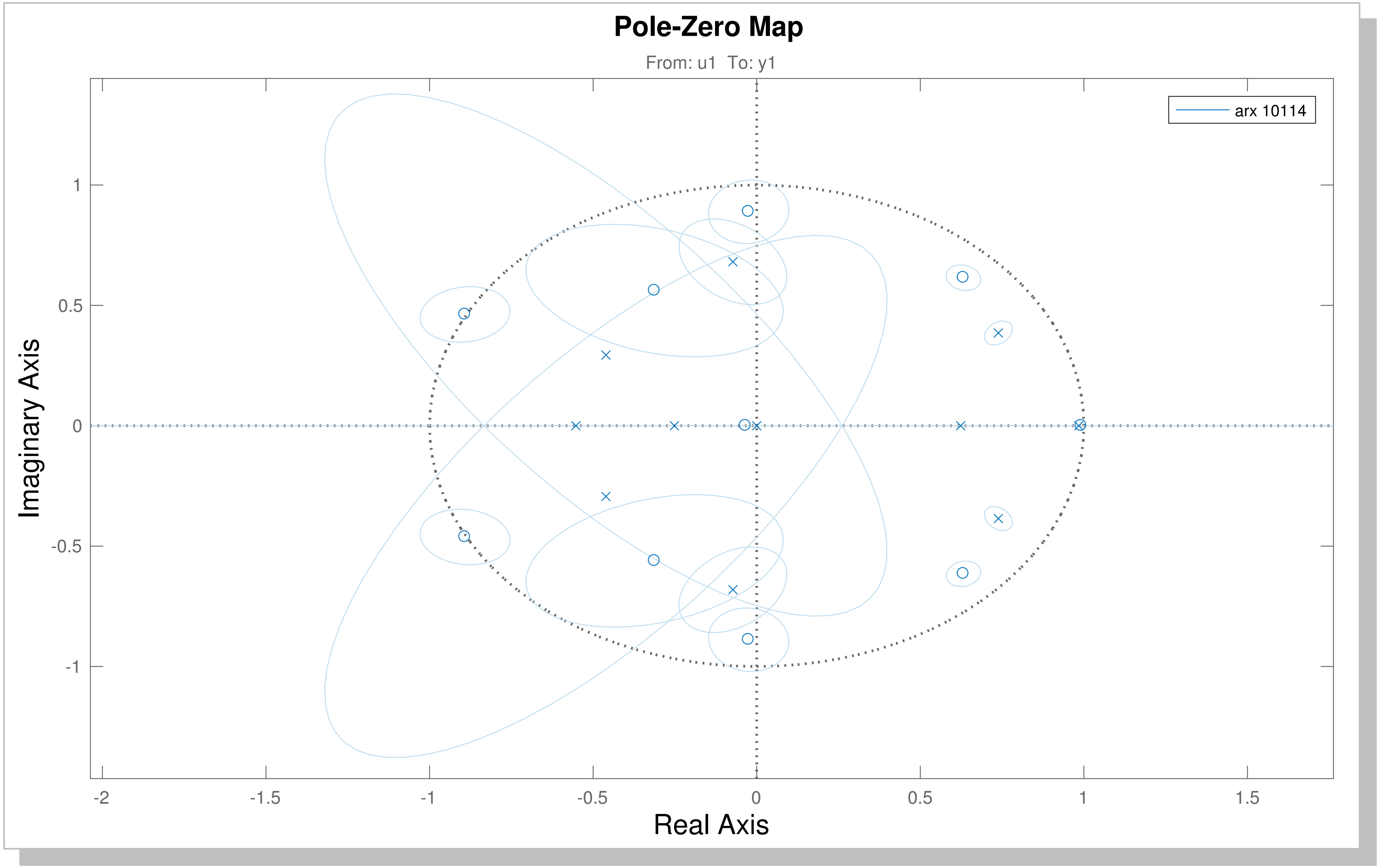
PZ maps

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%              PZMAPS                                                  %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
slide_figure
h=iopzplot(bj_m);legend('bj nf=2 nb=2 nk=4');
sd=2;
showConfidence(h,sd);
slide_figure
h=iopzplot(arx_m_234);legend('arx 234');
sd=2;
showConfidence(h,sd);
slide_figure
h=iopzplot(arx_m_10114);legend('arx 10114');
sd=2;
showConfidence(h,sd);
```

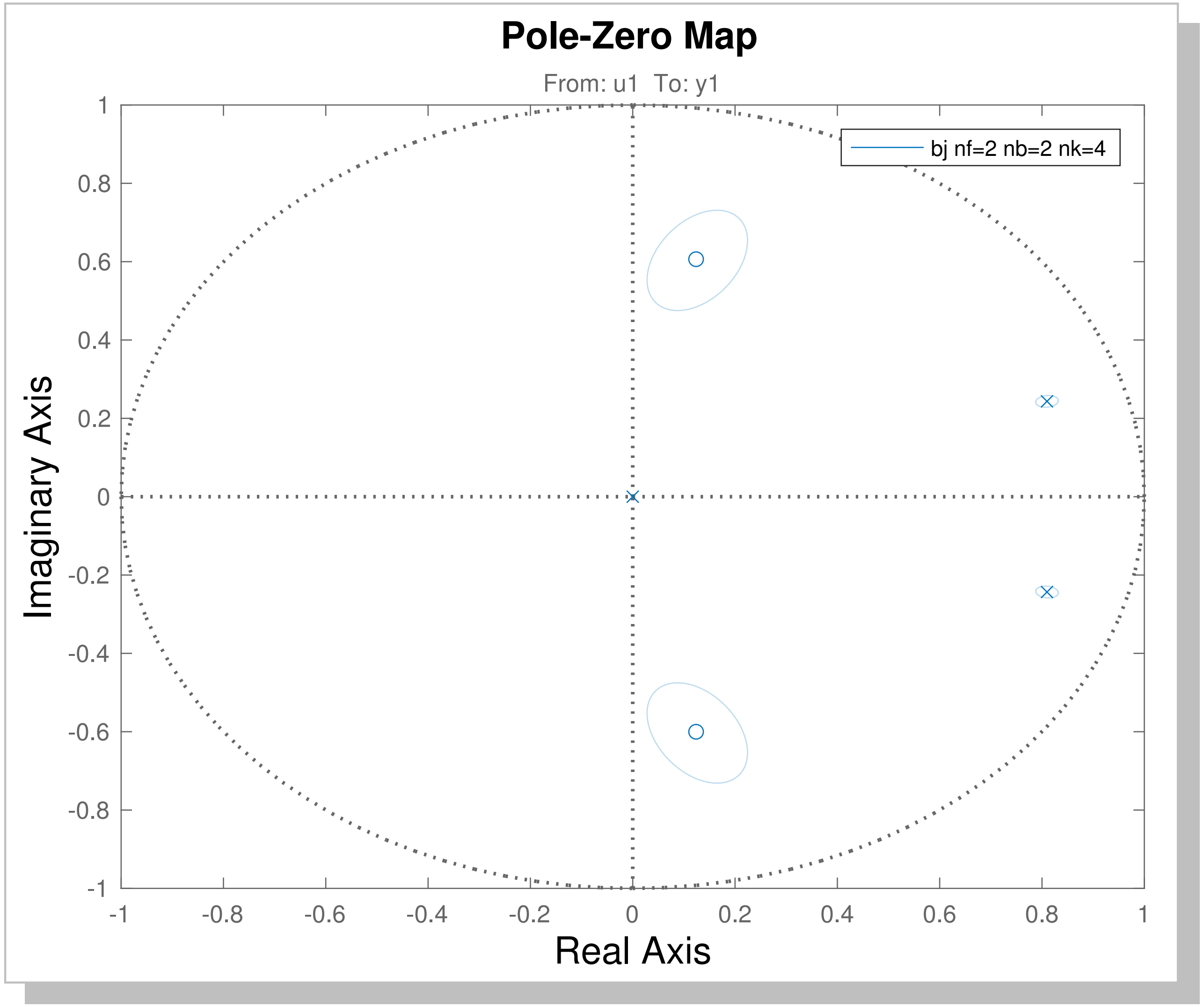
PZ map (ARX 234)



PZ map (ARX 10114)



PZ map (BJ)



Pole-Zero Map
From: u1 To: y1
bj nf=2 nb=2 nk=4
Imaginary Axis
Real Axis
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1

Conversion of **idmodel** to LTI model for controller design using **polydata**

Generic **idmodel** structure for transfer function models (time delay is absorbed in $\mathsf{B}(q^{-1})$):

$$\mathsf{A}(q^{-1})y(k) = \frac{\mathsf{B}(q^{-1})}{\mathsf{F}(q^{-1})}u(k) + \frac{\mathsf{C}(q^{-1})}{\mathsf{D}(q^{-1})}e(k)$$

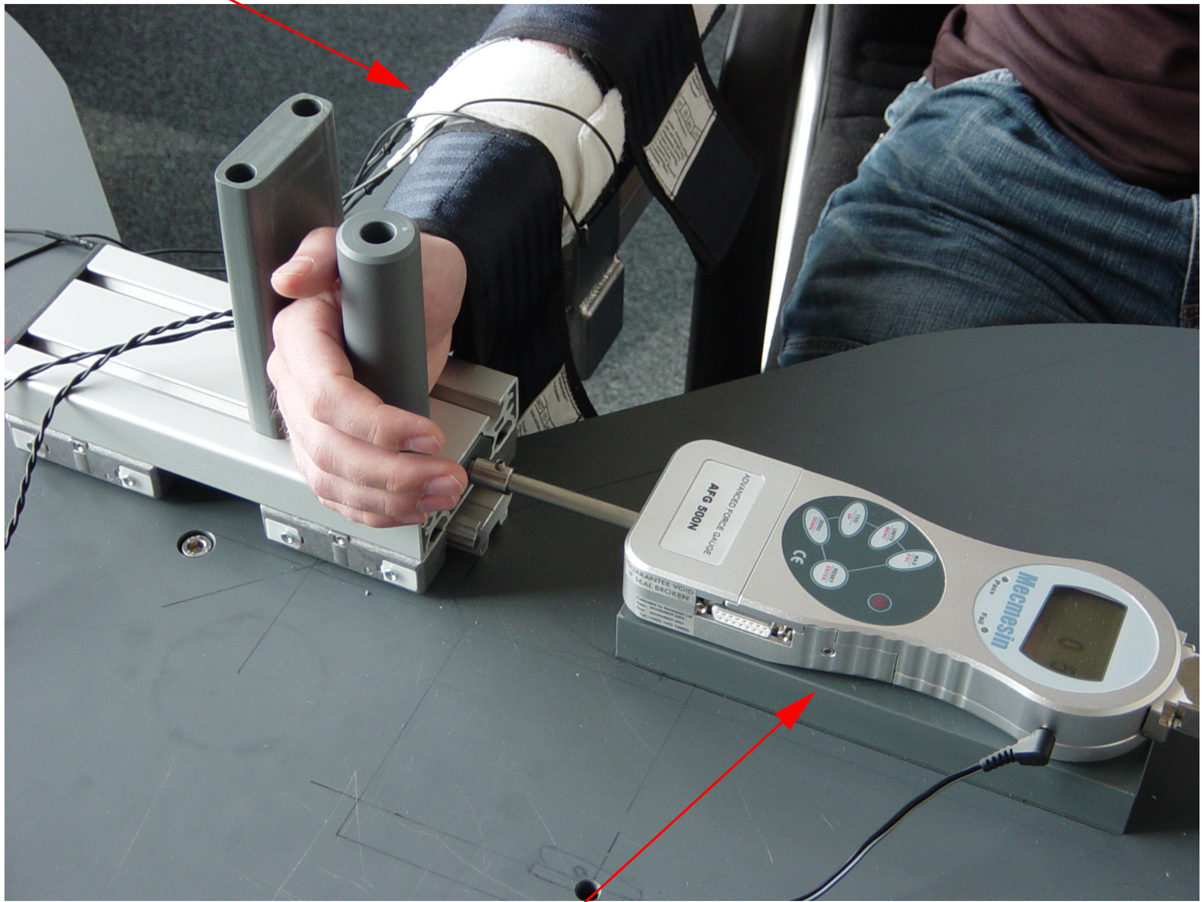
```
%%%
% Convert to lti model in q^-1 for controller design
%%%
[A,B,C,D,F]=polydata(bj_m);
T_s=0.05;
H=tf(C,conv(D,A),T_s,'variable','z^-1');
G=tf(B,conv(F,A),T_s,'variable','z^-1');

%%%
% Convert to z
%%%
set(H,'variable','z');
set(G,'variable','z');
```

Conclusions

- BJ model with low order G does a better job than a ARX model!
- High order for ARX describes noise and disturbances but not deterministic system parts.
- G of BJ model would be the best for controller design
- Identification strategy: 1st: FIR (n_k), 2nd: ARX, 3rd: BJ,OE (then ARX ist not good) or procedure outline in lecture

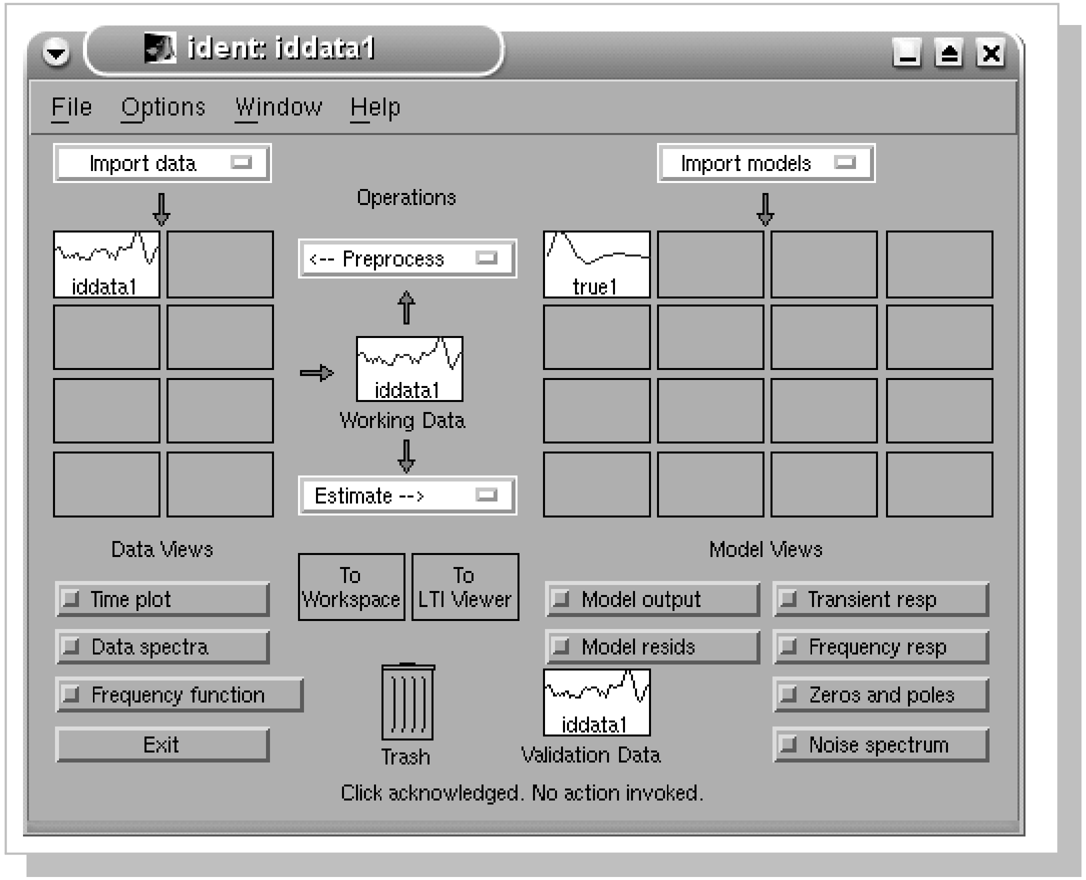
Live GUI Tutorial – Identification of the Wrist Extensor Muscle

Stimulation of the wrist extensor (pulsewidth u)



Measurement of the isometric force (y)

- *System*: pulse width – input u, force – output y
- *Sample period*: 50ms
- *Data set*: `data_io.mat` contains 4 experiments
- *Identify model using the graphical user interface of the Identification Toolbox*: start with `ident`



Ident-Tool