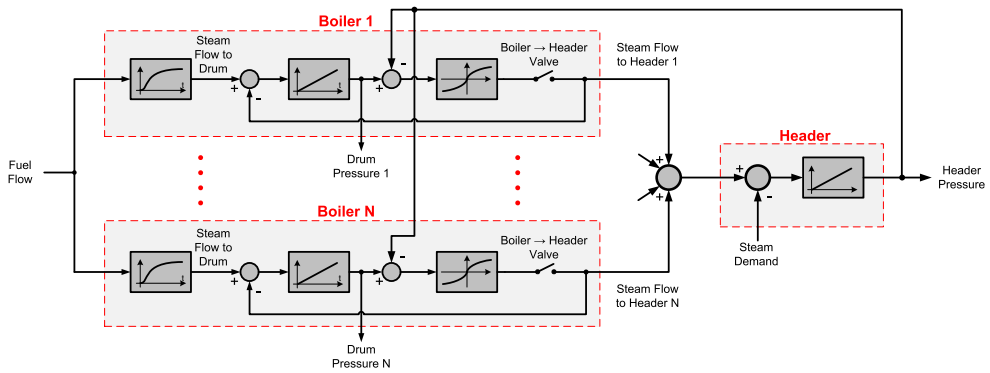
Demonstration of Large Scale Model Class `LSmodel`

This script demonstrates the use of `LSmodel` class for modeling and model management of Large Scale (LS) systems. The first part shows definition of `LSmodel` for multiple boilers connected to a single header. The second part is demonstration of `LSmodel` methods on randomly generated model with 15 submodels.

Contents

Submodels Definition

This section defines models of individual boilers and header. These models will be used to create LS model in the next section. The structure of the model is in following figure.



```
% --- Boiler parameters -----------------------------
n_boilers = 5;               % number of boilers (1-5)
T = [50 100 150 200 250];    % fuel -> steam time constants [s]
Ks = 10;                     % amount of steam [t/hrs] from unit of fuel [t/hrs]
V = [300 500 700 800 600];   % boiler volumes [m3] (1/V ~ pressure integration constant)
K = [100 130 150 130 140];   % boiler->header pipe "conductivity", =flow/(pressure difference) [t/hr
Kh = 1;                      % "conductivity" to turbine (higher number -> higher stabilization
Vh = 2e3;                    % header volume [m3]

% --- Boiler models ---------------------------------
% Inputs:
%   FF ... fuel flow - same value flows to all boilers [t/hrs]
%   ph ... header pressure [MPa]
% -------------------------------------------------
% States:
%   GS ... generated steam [t/hrs]
%   pb ... boiler pressure [MPa]
% -------------------------------------------------
% Outputs:
%   SF ... overall steam flow from all boilers [t/hrs]
%   pb ... boiler pressure [MPa]
% -------------------------------------------------
for i=1:n_boilers,
  A = [ -1/T(i) , 0
        1/V(i)  , -K(i)/V(i) ];
  B = [ Ks/T(i) , 0
        0       , K(i)/V(i) ];
  C = [ 0 , K(i)
        0 , 1    ];
  D = [ 0 , -K(i)
        0 , 0    ];
  M{i} = ss(A,B,C,D);           % models of individual boilers
  M{i}.InputName  = { 'FF' , 'ph' };
  M{i}.StateName  = { ['GS' num2str(i)] , ['pb' num2str(i)] };
  M{i}.OutputName = { ['SF' num2str(i)] , ['pb' num2str(i)] };
  M{i}.Name = ['B' num2str(i) ];
end


% --- Header model ----------------------------------
% Inputs:
%   SF ... total steam flow from boilers [t/hrs]
%   SD ... steam demand from (from turbine) [t/hrs]
% -------------------------------------------------
% Outputs:
%   ph ... header pressure [MPa]
% -------------------------------------------------
A = -Kh/Vh;
B = [1/Vh -1/Vh];
C = 1;
D = 0;
M{n_boilers+1} = ss(A,B,C,D);
M{n_boilers+1}.InputName  = { 'SF' , 'SD' };
M{n_boilers+1}.StateName  = { 'ph' };
```

```
M{n_boilers+1}.OutputName = { 'ph' };
M{n_boilers+1}.Name       = 'H';

% --- Steam flow summing block -----------------------
% sums steam flows from boilers to header
sum1 = sumblk('SF','SF1','SF2','SF3','SF4','SF5');
sum1.Name = 'SFsum';
```

Construction of LSmodel

LS model is contructed by specifying submodels, summators and a list of external inputs and outputs. Internal connections are automatically created by matching input/output names

```
mod = LSmodel(M,sum1,{'FF','SD'},{'ph','SF'})
```

```
Large scale model (total order=11, 6 subsystem(s), 1 summator(s)):
  --- Subsystems -----------
    M1: 2 inputs, 2 outputs, order=2, name="B1"
    M2: 2 inputs, 2 outputs, order=2, name="B2"
    M3: 2 inputs, 2 outputs, order=2, name="B3"
    M4: 2 inputs, 2 outputs, order=2, name="B4"
    M5: 2 inputs, 2 outputs, order=2, name="B5"
    M6: 2 inputs, 1 outputs, order=1, name="H"
  --- Summators ------------------
    Sum1: 5 inputs, name = "SFsum"
  --- External Inputs -----------
    IN01: FF               X
    IN02: SD               X
  --- External Outputs ----------
    OUT01: ph              X
    OUT02: SF              X
```

Constructor function is compatible with standard CONNECT function, where the same model would be defined as (connect does not accept cell array of models)

```
mod_connect = connect(M{1},M{2},M{3},M{4},M{5},M{6},sum1,{'FF','SD'},{'ph','SF'});
```

It is also possible to construct the model by using numeric indexing of inputs and outputs. However, name based signal referencing will be prefered as numeric indexing may be confusing for LS systems.

```
inputs = { [1 3 5] , 8 };   % extended notation of standard connect
Q = [ 2 7 0 0
      4 7 0 0
      6 7 0 0
      7 1 3 5 ];
outputs = { 7 , [1 3 5] };

inputs = { [1 3 5 7 9], 12 };
Q = [2  11 0  0  0  0
```

```
     4  11 0  0  0  0
     6  11 0  0  0  0
     8  11 0  0  0  0
     10 11 0  0  0  0
     11 1  3  5  7  9 ];
outputs = { 11 , [1 3 5 7 9] };

mod2 = LSmodel(M{1},M{2},M{3},M{4},M{5},M{6},Q,inputs,outputs);
```

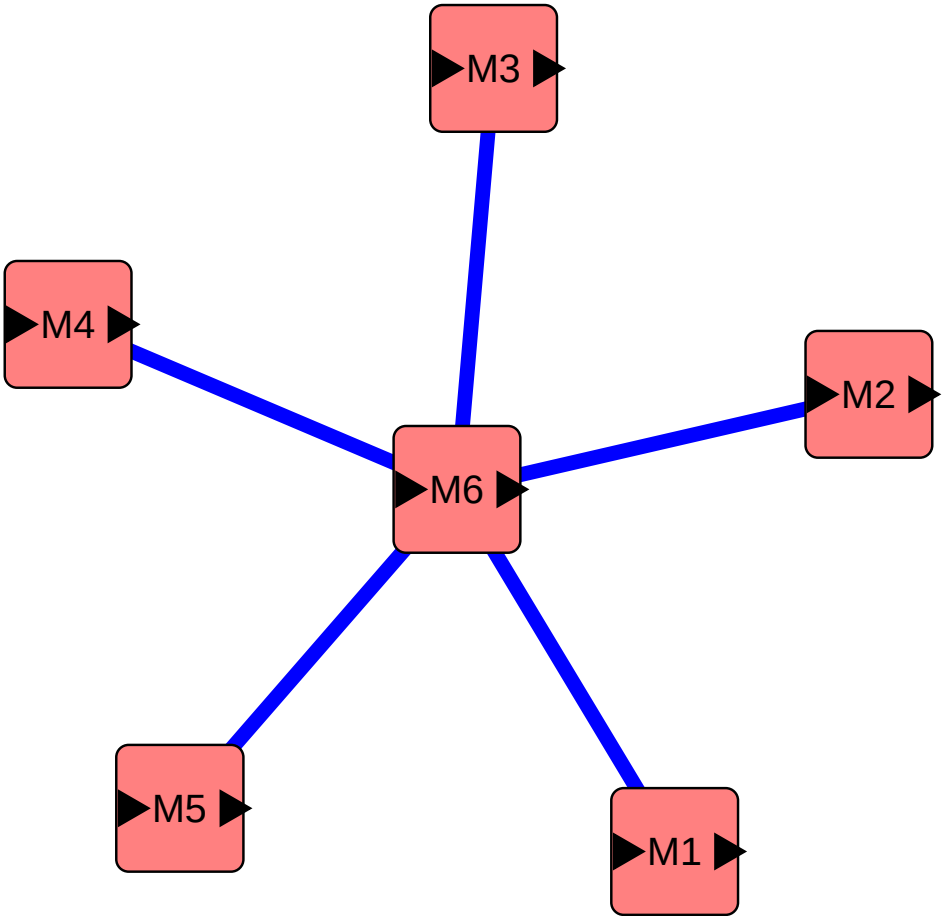
Plot model structure

Plots structure of large scale model as a graph, where vertexes are subsystems and edges indicate interaction between subsystems.

```
plot(mod);
title('\bfModel Structure');
```

```
Computing vertex positions... Done.
```

Model Structure



Random LS model with 15 sub-models

Previous model is rather small. Larger model will be randomly generated by using command `RLS` parameterized by number of sub-models, maximum sub-model order, number of external outputs and number of external inputs.

```
seed=963;rand('seed',seed);randn('seed',seed); % to get stable CL model
mod = rls(15,3,3,3)
```

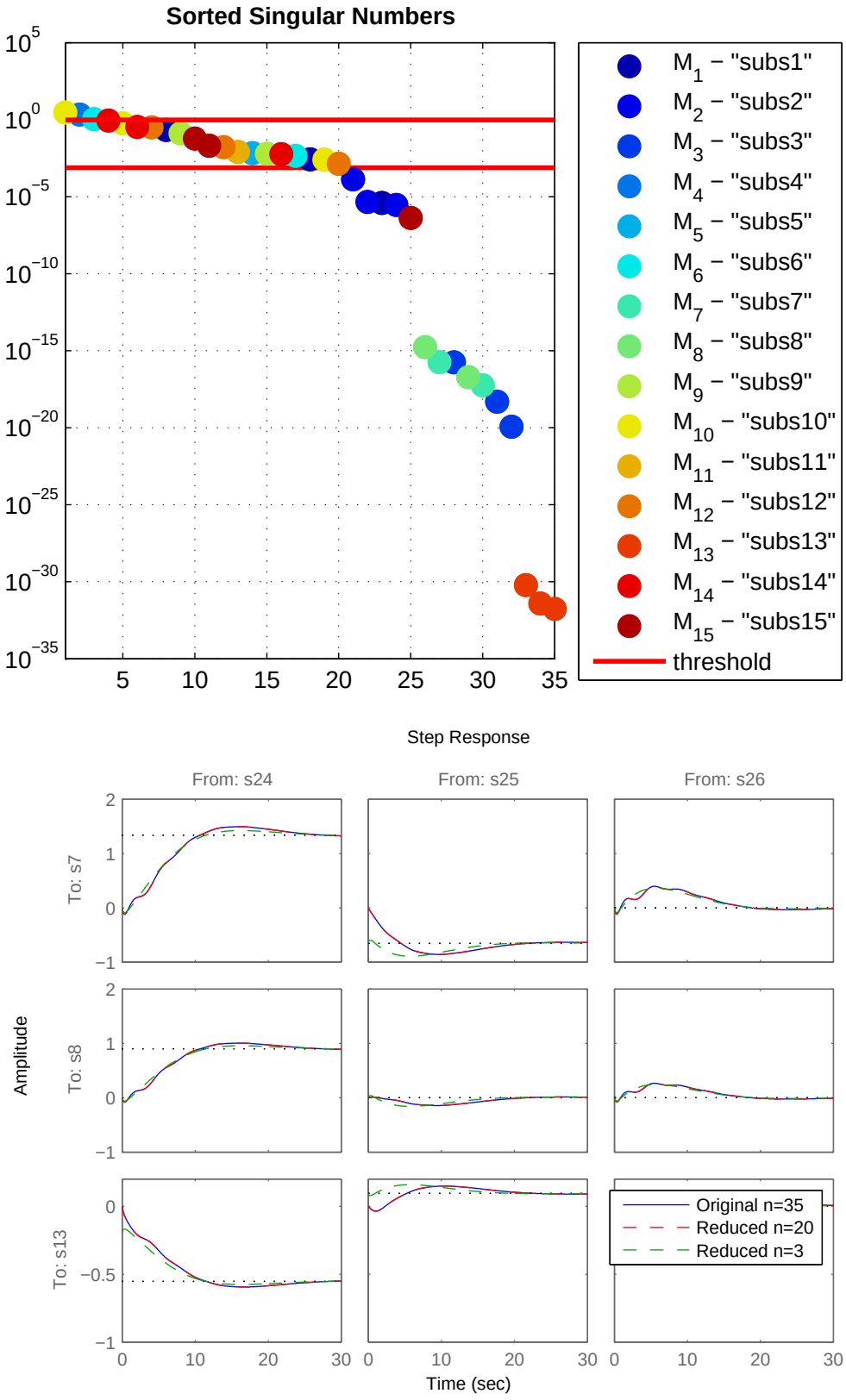
```
Large scale model (total order=35, 15 subsystem(s), 8 summator(s)):
  --- Subsystems -----------
    M1: 1 inputs, 1 outputs, order=3, name="subs1"
    M2: 1 inputs, 1 outputs, order=3, name="subs2"
```

```
  M3: 1 inputs, 2 outputs, order=3, name="subs3"
  M4: 2 inputs, 2 outputs, order=1, name="subs4"
  M5: 2 inputs, 1 outputs, order=1, name="subs5"
  M6: 1 inputs, 1 outputs, order=2, name="subs6"
  M7: 1 inputs, 1 outputs, order=2, name="subs7"
  M8: 2 inputs, 2 outputs, order=2, name="subs8"
  M9: 2 inputs, 2 outputs, order=2, name="subs9"
  M10: 1 inputs, 1 outputs, order=3, name="subs10"
  M11: 2 inputs, 2 outputs, order=1, name="subs11"
  M12: 1 inputs, 2 outputs, order=3, name="subs12"
  M13: 1 inputs, 2 outputs, order=3, name="subs13"
  M14: 2 inputs, 2 outputs, order=3, name="subs14"
  M15: 1 inputs, 1 outputs, order=3, name="subs15"
--- Summators -----------------
  Sum1: 3 inputs, name = "S1"
  Sum2: 4 inputs, name = "S2"
  Sum3: 3 inputs, name = "S3"
  Sum4: 5 inputs, name = "S4"
  Sum5: 3 inputs, name = "S5"
  Sum6: 3 inputs, name = "S6"
  Sum7: 5 inputs, name = "S7"
  Sum8: 5 inputs, name = "S8"
--- External Inputs -----------
  IN01: s24              X
  IN02: s25              X
  IN03: s26              X
--- External Outputs ----------
  OUT01: s7              X
  OUT02: s8              X
  OUT03: s13             X
```

Structure preserving model order reduction

Standard model order reduction destroys model internal structure. Structure preserving model order reduction reduces order of individual sub-models to achieve minimum mismatch with original model from external input/output point of view. Structured singular numbers are plotted to show which sub-models are reduced in order to achieve target global model order.

```
figure(2);clf;
mod_r20=mod.struct_red(20);       % model reduced to 20th order
mod_r3=mod.struct_red(3);         % model reduced to 3rd order
% Compare step responses of original and reduced order models
figure(1);
clf;step(mod,30);hold all;
step(mod_r20,'r--',30);
step(mod_r3,'g--',30);
legend(['Original n=' num2str(mod.n)],['Reduced n=' num2str(mod_r20.n)],['Reduced n=' num2str(mod_r3
```


Sorted Singular Numbers
10^{5}
10^{0}
10^{-5}
10^{-10}
10^{-15}
10^{-20}
10^{-25}
10^{-30}
10^{-35}
5
10
15
20
25
30
35
M_1 – "subs1"
M_2 – "subs2"
M_3 – "subs3"
M_4 – "subs4"
M_5 – "subs5"
M_6 – "subs6"
M_7 – "subs7"
M_8 – "subs8"
M_9 – "subs9"
M_{10} – "subs10"
M_{11} – "subs11"
M_{12} – "subs12"
M_{13} – "subs13"
M_{14} – "subs14"
M_{15} – "subs15"
threshold
Step Response
From: s24
From: s25
From: s26
To: s7
To: s8
To: s13
Amplitude
Time (sec)
Original n=35
Reduced n=20
Reduced n=3

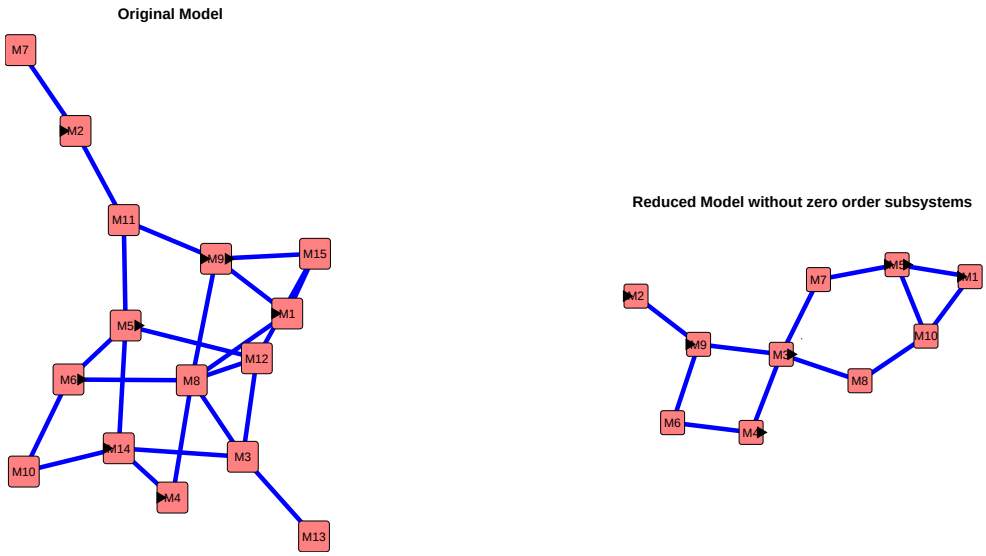
"On-line" Model changes

Removing submodel(s)

Remove subsystems, which have zero order after structured reduction (just for demonstration of large scale model manipulations).

```
ind = find(mod_r20.orders==0);
removed_models = mod.M(ind);
modX = mod.rem_mod(ind);
figure(10); clf;
subfigure(10,3,3,[1 5]);
subplot(121);plot(mod);   title('\bfOriginal Model');
subplot(122);plot(modX);  title('\bfReduced Model without zero order subsystems');
```

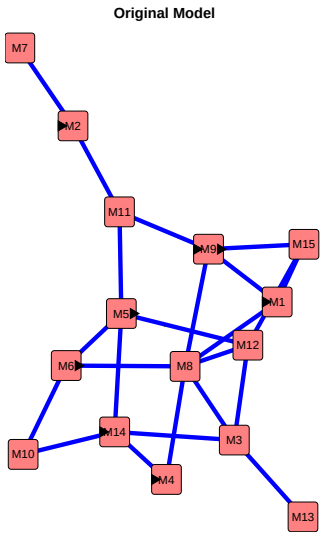
```
Computing vertex positions... Done.
Computing vertex positions... Done.
```

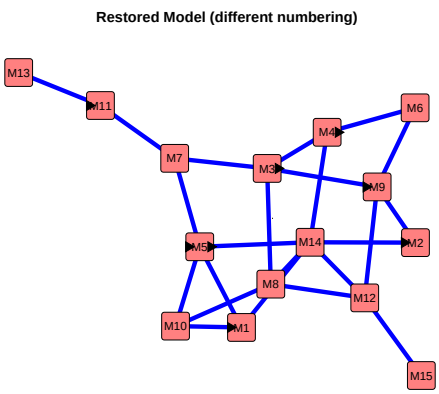


Adding submodel(s)

```
modX = modX.add_mod( removed_models );
cla; plot(modX);
title('\bfRestored Model (different numbering)');
```

```
Computing vertex positions... Done.
```



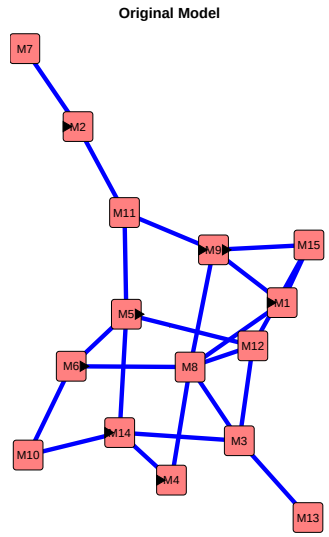


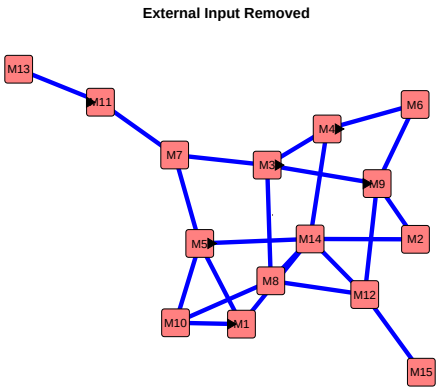
Removing external input

```
modX = modX.rem_ext_inp('s24')
cla; plot(modX);
title('\bfExternal Input Removed');
```

```
Large scale model (total order=35, 15 subsystem(s), 8 summator(s)):
  --- Subsystems -----------
    M1: 1 inputs, 1 outputs, order=3, name="subs1"
    M2: 2 inputs, 2 outputs, order=1, name="subs4"
    M3: 2 inputs, 1 outputs, order=1, name="subs5"
    M4: 1 inputs, 1 outputs, order=2, name="subs6"
    M5: 2 inputs, 2 outputs, order=2, name="subs9"
    M6: 1 inputs, 1 outputs, order=3, name="subs10"
    M7: 2 inputs, 2 outputs, order=1, name="subs11"
    M8: 1 inputs, 2 outputs, order=3, name="subs12"
    M9: 2 inputs, 2 outputs, order=3, name="subs14"
    M10: 1 inputs, 1 outputs, order=3, name="subs15"
    M11: 1 inputs, 1 outputs, order=3, name="subs2"
    M12: 1 inputs, 2 outputs, order=3, name="subs3"
    M13: 1 inputs, 1 outputs, order=2, name="subs7"
    M14: 2 inputs, 2 outputs, order=2, name="subs8"
    M15: 1 inputs, 2 outputs, order=3, name="subs13"
  --- Summators ----------------
    Sum1: 3 inputs, name = "S1"
    Sum2: 4 inputs, name = "S2"
    Sum3: 3 inputs, name = "S3"
    Sum4: 5 inputs, name = "S4"
    Sum5: 3 inputs, name = "S5"
    Sum6: 3 inputs, name = "S6"
    Sum7: 5 inputs, name = "S7"
    Sum8: 5 inputs, name = "S8"
  --- External Inputs -----------
    IN01: s25              X
    IN02: s26              X
```

```
  --- External Outputs ----------
    OUT01: s7              X
    OUT02: s8              X
    OUT03: s13             X

Computing vertex positions... Done.
```



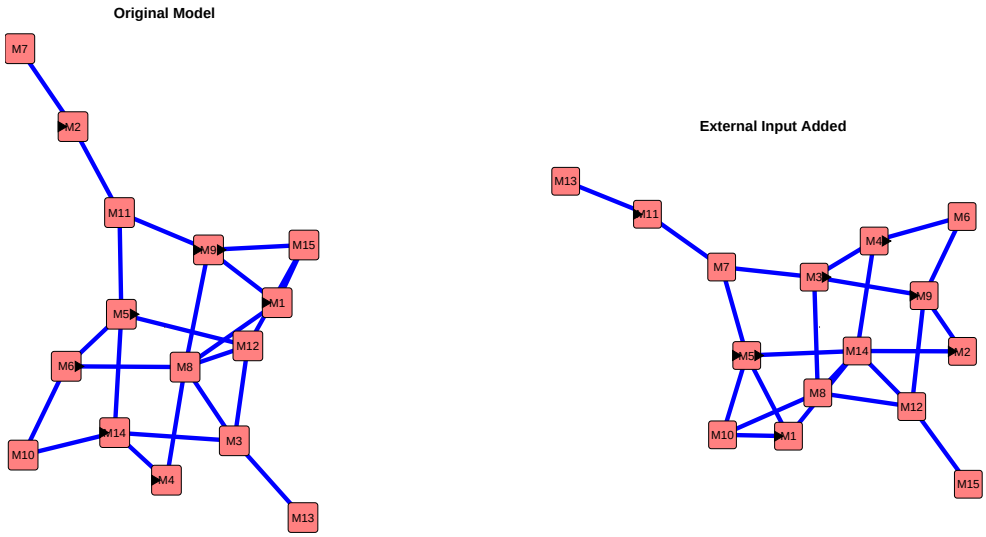


Adding external input

```
modX = modX.add_ext_inp('s24')
cla; plot(modX);
title('\bfExternal Input Added');
```

```
Large scale model (total order=35, 15 subsystem(s), 8 summator(s)):
  --- Subsystems -----------
    M1: 1 inputs, 1 outputs, order=3, name="subs1"
    M2: 2 inputs, 2 outputs, order=1, name="subs4"
    M3: 2 inputs, 1 outputs, order=1, name="subs5"
    M4: 1 inputs, 1 outputs, order=2, name="subs6"
    M5: 2 inputs, 2 outputs, order=2, name="subs9"
    M6: 1 inputs, 1 outputs, order=3, name="subs10"
    M7: 2 inputs, 2 outputs, order=1, name="subs11"
    M8: 1 inputs, 2 outputs, order=3, name="subs12"
    M9: 2 inputs, 2 outputs, order=3, name="subs14"
    M10: 1 inputs, 1 outputs, order=3, name="subs15"
    M11: 1 inputs, 1 outputs, order=3, name="subs2"
    M12: 1 inputs, 2 outputs, order=3, name="subs3"
    M13: 1 inputs, 1 outputs, order=2, name="subs7"
    M14: 2 inputs, 2 outputs, order=2, name="subs8"
    M15: 1 inputs, 2 outputs, order=3, name="subs13"
  --- Summators -----------------
    Sum1: 3 inputs, name = "S1"
    Sum2: 4 inputs, name = "S2"
    Sum3: 3 inputs, name = "S3"
```

```
    Sum4: 5 inputs, name = "S4"
    Sum5: 3 inputs, name = "S5"
    Sum6: 3 inputs, name = "S6"
    Sum7: 5 inputs, name = "S7"
    Sum8: 5 inputs, name = "S8"
  --- External Inputs -----------
    IN01: s25              X
    IN02: s26              X
    IN03: s24              X
  --- External Outputs ----------
    OUT01: s7              X
    OUT02: s8              X
    OUT03: s13             X

Computing vertex positions... Done.
```

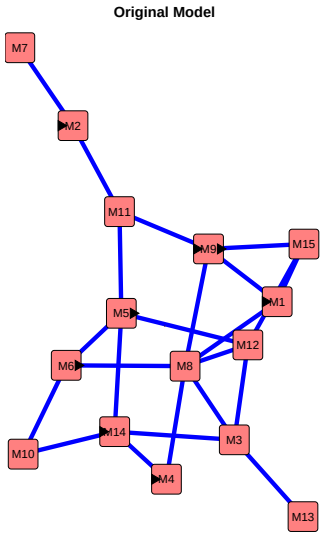


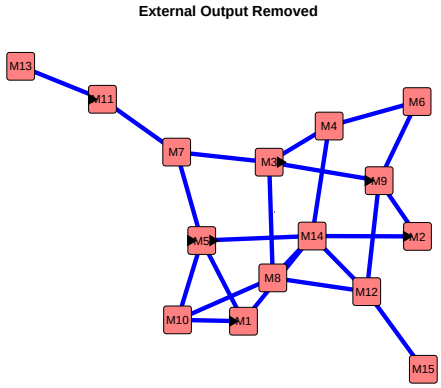
Removing external output

```
modX = modX.rem_ext_out('s8')
cla; plot(modX);
title('\bfExternal Output Removed');
```

```
Large scale model (total order=35, 15 subsystem(s), 8 summator(s)):
  --- Subsystems -----------
    M1: 1 inputs, 1 outputs, order=3, name="subs1"
    M2: 2 inputs, 2 outputs, order=1, name="subs4"
    M3: 2 inputs, 1 outputs, order=1, name="subs5"
    M4: 1 inputs, 1 outputs, order=2, name="subs6"
    M5: 2 inputs, 2 outputs, order=2, name="subs9"
    M6: 1 inputs, 1 outputs, order=3, name="subs10"
    M7: 2 inputs, 2 outputs, order=1, name="subs11"
    M8: 1 inputs, 2 outputs, order=3, name="subs12"
    M9: 2 inputs, 2 outputs, order=3, name="subs14"
    M10: 1 inputs, 1 outputs, order=3, name="subs15"
```

```
    M11: 1 inputs, 1 outputs, order=3, name="subs2"
    M12: 1 inputs, 2 outputs, order=3, name="subs3"
    M13: 1 inputs, 1 outputs, order=2, name="subs7"
    M14: 2 inputs, 2 outputs, order=2, name="subs8"
    M15: 1 inputs, 2 outputs, order=3, name="subs13"
  --- Summators -----------------
    Sum1: 3 inputs, name = "S1"
    Sum2: 4 inputs, name = "S2"
    Sum3: 3 inputs, name = "S3"
    Sum4: 5 inputs, name = "S4"
    Sum5: 3 inputs, name = "S5"
    Sum6: 3 inputs, name = "S6"
    Sum7: 5 inputs, name = "S7"
    Sum8: 5 inputs, name = "S8"
  --- External Inputs -----------
    IN01: s25              X
    IN02: s26              X
    IN03: s24              X
  --- External Outputs ----------
    OUT01: s7              X
    OUT02: s13             X

Computing vertex positions... Done.
```



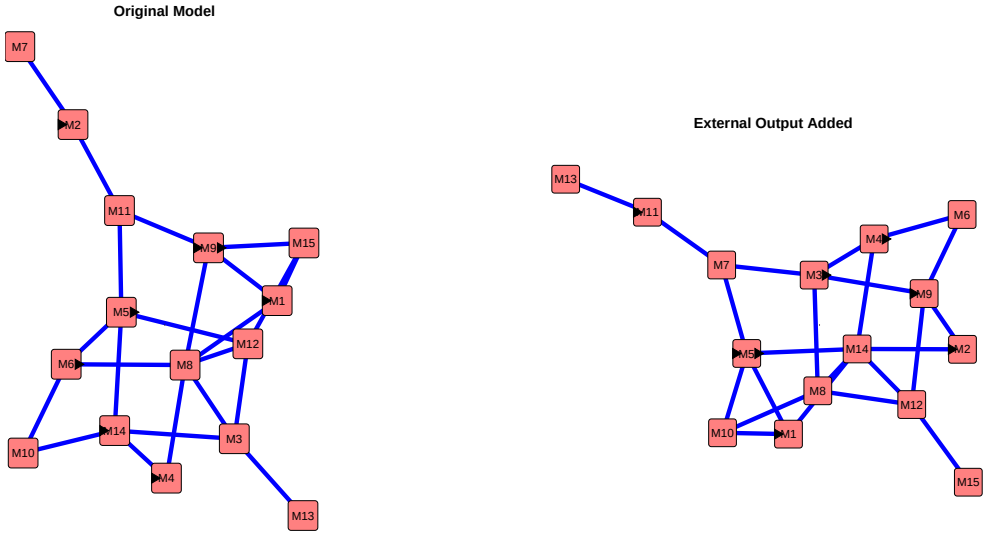


Adding external output

```
modX = modX.add_ext_out('s8')
cla; plot(modX);
title('\bfExternal Output Added');
```

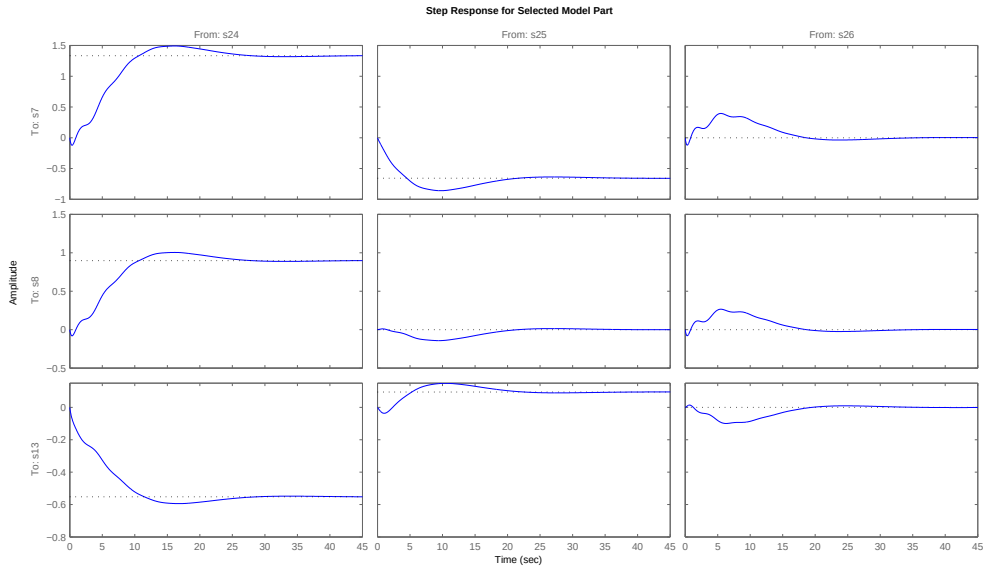
```
Large scale model (total order=35, 15 subsystem(s), 8 summator(s)):
  --- Subsystems -----------
    M1: 1 inputs, 1 outputs, order=3, name="subs1"
    M2: 2 inputs, 2 outputs, order=1, name="subs4"
```

```
  M3: 2 inputs, 1 outputs, order=1, name="subs5"
  M4: 1 inputs, 1 outputs, order=2, name="subs6"
  M5: 2 inputs, 2 outputs, order=2, name="subs9"
  M6: 1 inputs, 1 outputs, order=3, name="subs10"
  M7: 2 inputs, 2 outputs, order=1, name="subs11"
  M8: 1 inputs, 2 outputs, order=3, name="subs12"
  M9: 2 inputs, 2 outputs, order=3, name="subs14"
  M10: 1 inputs, 1 outputs, order=3, name="subs15"
  M11: 1 inputs, 1 outputs, order=3, name="subs2"
  M12: 1 inputs, 2 outputs, order=3, name="subs3"
  M13: 1 inputs, 1 outputs, order=2, name="subs7"
  M14: 2 inputs, 2 outputs, order=2, name="subs8"
  M15: 1 inputs, 2 outputs, order=3, name="subs13"
--- Summators -----------------
  Sum1: 3 inputs, name = "S1"
  Sum2: 4 inputs, name = "S2"
  Sum3: 3 inputs, name = "S3"
  Sum4: 5 inputs, name = "S4"
  Sum5: 3 inputs, name = "S5"
  Sum6: 3 inputs, name = "S6"
  Sum7: 5 inputs, name = "S7"
  Sum8: 5 inputs, name = "S8"
--- External Inputs -----------
  IN01: s25              X
  IN02: s26              X
  IN03: s24              X
--- External Outputs ----------
  OUT01: s7              X
  OUT02: s13             X
  OUT03: s8              X

Computing vertex positions... Done.
```



Extraction of model part - eliminates submodels which are not controlable and observable for selected inputs and outputs.

```
ex_mod = mod.select('s24','s7');
clf;step(ex_mod);
title('\bfStep Response for Selected Model Part');
```

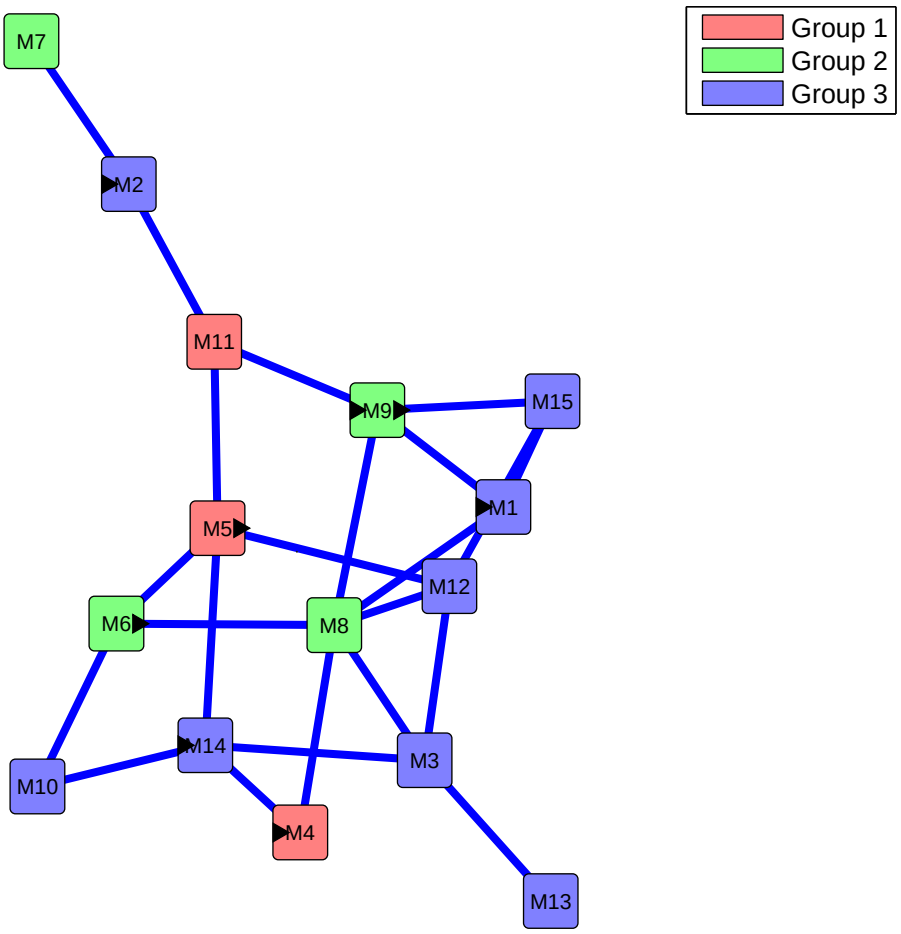


Submodels grouping

Each sub-model can be assigned to a group. Each group has its own number and sub-models are assigned to given groups in property GROUPING. Following command combines sub-models to groups according to their order.

```
mod.grouping = mod.orders;
clf; plot(mod);
```

```
Computing vertex positions... Done.
```



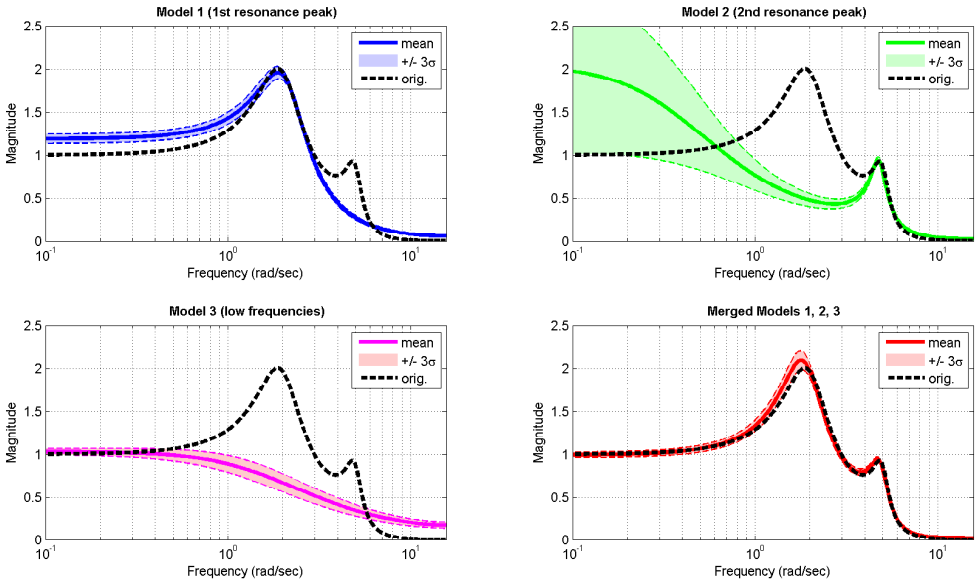
Time domain models merging

Merging of multiple models describing the same system. The merging is done in time domain and allows to combine information from uncertain ARX models of different orders (`idarx` or `idpoly`)

```
[id_mod,sys] = demo_models; % 3 models of the same system with different
                            % different orders identified from different data

for i=1:3,
  LS{i} = LSmodel(id_mod{i},'u1','y1');
end

LS12 = LS{1}.merge('M1',id_mod{2},4,4);
LS123 = LS12.merge('M1',id_mod{3},4,4);
```

```
% --- Plot the result ----------------------------
sc = 3; % variance scale
figure(10);clf;
w = logspace(-1,log10(pi/id_mod{1}.Ts),300);
X = [w w(end:-1:1)];
axx = [w(1) w(end) 0 2.5];

for i=1:3,
  [meanG{i}  ,varG{i}  ] = LS{i}.freq_uncert('M1',w);
end
[meanG123,varG123] = LS123.freq_uncert('M1',w);

line_c = { 'b','g','m' };
fill_c = { [.8 .8 1],[.8 1 .8],[1 .8 .8]};
for i=1:3,
  subplot(2,2,i);
  h(i) = semilogx(w,meanG{i},line_c{i},'LineWidth',3); hold on;
  Y = meanG{i}+sc*sqrt(varG{i}); Y = [meanG{i}-sc*sqrt(varG{i}) Y(end:-1:1)];
  hf(i) = fill(X,Y,fill_c{i},'EdgeAlpha',0,'FaceAlpha',0.5,'LineStyle','none');
  semilogx(w,[meanG{i}-sc*sqrt(varG{i}) ; meanG{i}+sc*sqrt(varG{i})]',[line_c{i} '--'],'LineWidth',1
  switch i,
    case 1, title('\bf Model 1 (1st resonance peak)');
    case 2, title('\bf Model 2 (2nd resonance peak)');
    case 3, title('\bf Model 3 (low frequencies)');
  end
end

subplot(2,2,4);
h(4) = semilogx(w,meanG123,'r','LineWidth',3); hold on;
Y = meanG123+sc*sqrt(varG123); Y = [meanG123-sc*sqrt(varG123) Y(end:-1:1)];
hf(4) = fill(X,Y,[1 .8 .8],'EdgeAlpha',0,'FaceAlpha',0.5,'LineStyle','none');
semilogx(w,[meanG123-sc*sqrt(varG123) ; meanG123+sc*sqrt(varG123)]','r--','LineWidth',1);
title('\bf Merged Models 1, 2, 3');

[mag,phase] = bode(sys,w);
for i=1:4,
  subplot(2,2,i);
  horig(i) = semilogx(w,squeeze(mag),'k--','LineWidth',3);
  axis(axx); grid on; xlabel('Frequency (rad/sec)'); ylabel('Magnitude');
  legend([h(i) hf(i) horig(i)] ,'mean',['+/- ',num2str(sc),'\sigma'],'orig.');
end

Current iteration error = 30568.7304
Current iteration error = 30568.7304
Current iteration error = 30568.7304
Current iteration error = 370494.737
Current iteration error = 410827.2347
Current iteration error = 68383.1445
Current iteration error = 30568.7304
Current iteration error = 30568.7304
Current iteration error = 30568.7304
Current iteration error = 203078.9615
```

```
Maximum number of iterations exceeded.
Precission matrix quadratic error (normalized)= 0.02566
Current iteration error = 75719.3471
Current iteration error = 914945.2182
Current iteration error = 870103.539
Current iteration error = 54613.5888
Current iteration error = 155621.7346
Current iteration error = 92255.2874
Current iteration error = 137884.8909
Current iteration error = 863230.4205
Current iteration error = 66342.421
Current iteration error = 947935.3184
Maximum number of iterations exceeded.
Precission matrix quadratic error (normalized)= 0.068154
Current iteration error = 4469951.5209
Current iteration error = 2424856.2825
Current iteration error = 2315073.8367
Current iteration error = 2853276.1104
Current iteration error = 2511910.3461
Current iteration error = 2487229.5931
Current iteration error = 2455166.663
Current iteration error = 2764858.2822
Current iteration error = 2181018.6175
Current iteration error = 2194488.5943
Maximum number of iterations exceeded.
Precission matrix quadratic error (normalized)= 1.1806
 **********  Warning: Fictive data quality may be low.  **********
Current iteration error = 2.4564e-020
Precission matrix quadratic error (normalized)= 1.4569e-025
```



Overloaded functions

Many standard functions are overloaded for LSmodel class.

```
dcgain(mod)
```

```
Warning: Matrix is singular, close to singular or badly scaled.
         Results may be inaccurate. RCOND = NaN.

ans =

   NaN   NaN   NaN
   NaN   NaN   NaN
   NaN   NaN   NaN
```

```
pole(mod)
```

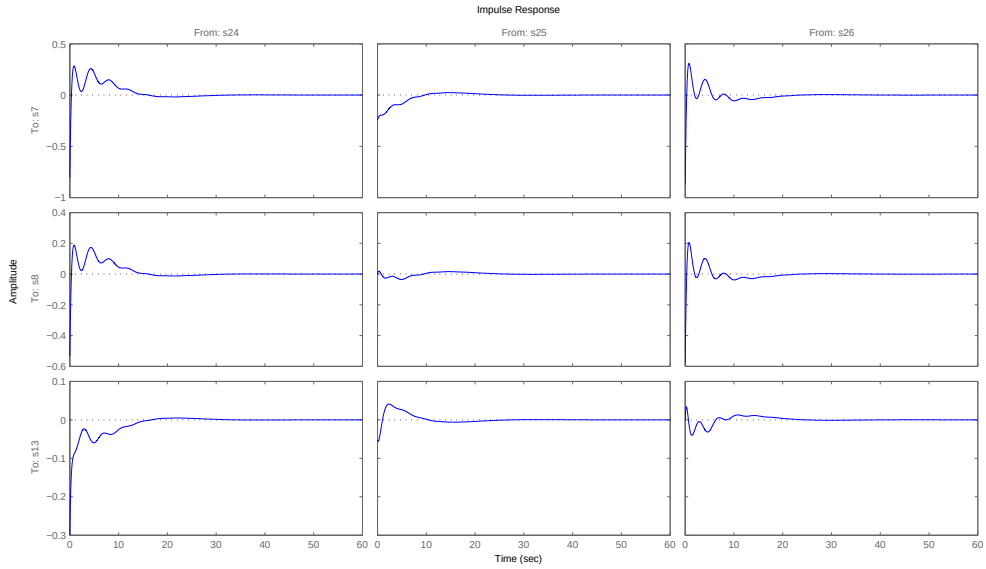
```
ans =

  -6.6715
  -4.4112 + 1.3459i
  -4.4112 - 1.3459i
  -0.2786 + 1.6244i
  -0.2786 - 1.6244i
  -2.2857
  -2.1010 + 0.2367i
  -2.1010 - 0.2367i
  -0.1335 + 0.1818i
  -0.1335 - 0.1818i
  -0.3904
  -1.8142
  -0.7314
  -0.7882
  -0.9910
  -1.6759
  -1.1631
  -1.1784
  -1.5414
  -1.3276
  -1.3349
  -1.4047
  -0.5040 + 2.7656i
  -0.5040 - 2.7656i
  -0.3553
  -6.6356
  -0.3594
  -3.4410
  -3.8862
  -3.7298
  -1.2730
  -3.6092
  -9.9454
  -2.3938
  -0.7164
```

```
zero(mod)
```

```
ans =

  -7.3180
  -6.2397
  -0.5040 + 2.7655i
  -0.5040 - 2.7655i
   1.1169
  -0.0229 + 1.2125i
  -0.0229 - 1.2125i
  -2.8319
  -0.0010
  -0.8764 + 0.6952i
  -0.8764 - 0.6952i
  -0.3551
  -0.4630
  -0.7853
  -2.0245
  -1.9684
  -1.9588
  -1.1695
  -1.7014
  -1.5418
  -1.4130
  -1.4413
  -6.6356
  -0.3594
  -3.4410
  -3.8862
  -3.7298
  -1.2730
  -3.6092
  -9.9454
  -2.3938
  -0.7164
```

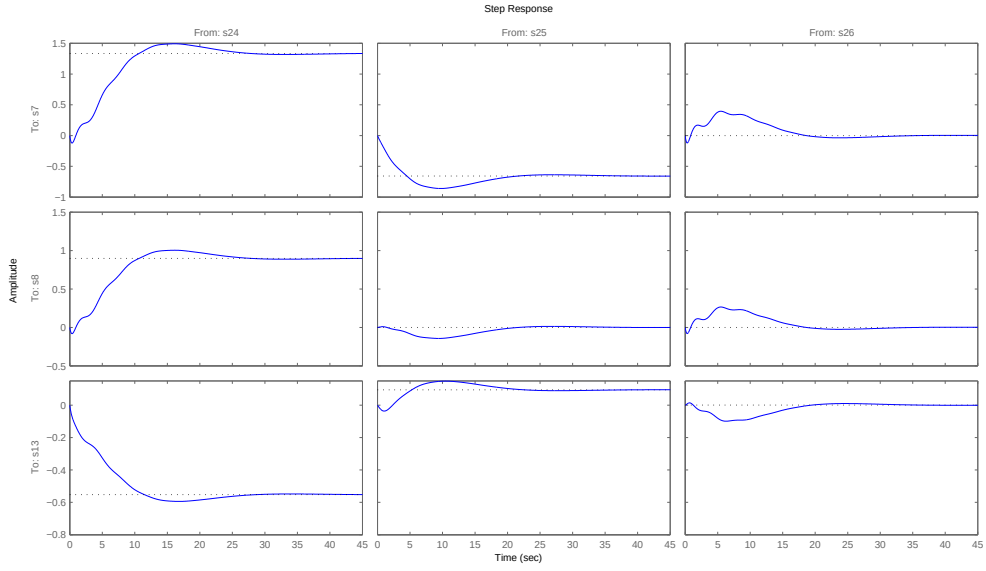
Impulse response

```
figure(10);clf;
impulse(mod);
```



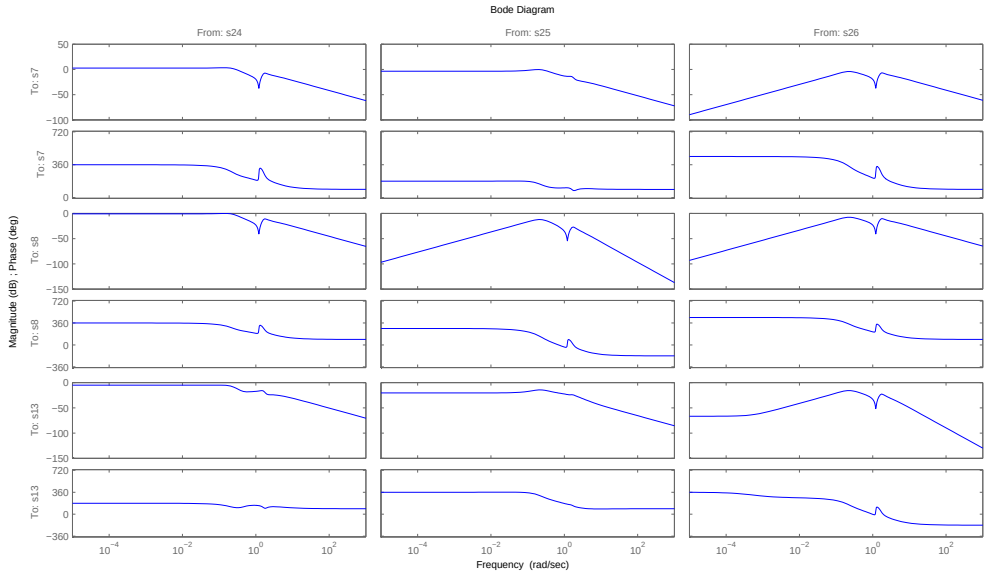
Step response

```
step(mod);
```



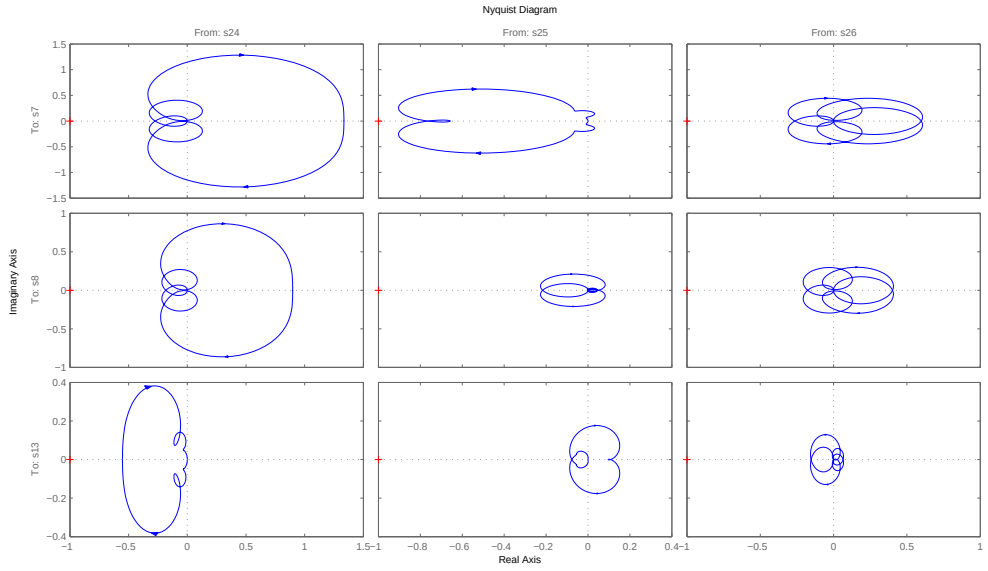
Bode frequency response

```
bode(mod);
```



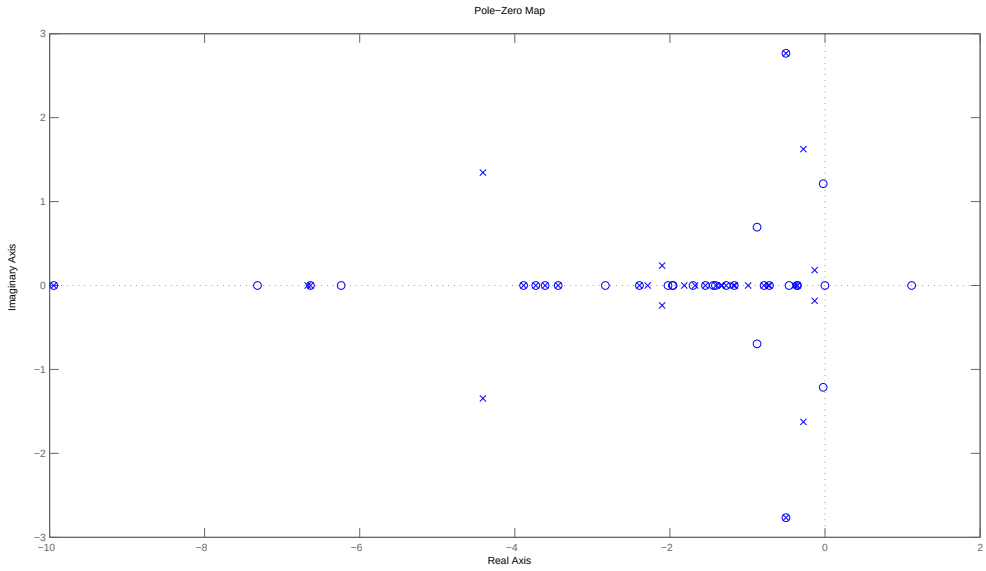
Nyquist frequency response

```
nyquist(mod);
```



Pole-zero map

```
pzmap(mod);
```



Pole-zero map for I/O pairs

```
iopzmap(mod);
```