# Dynamic system identification using exhaustive search and FLS

Gbr / 18.05.2019

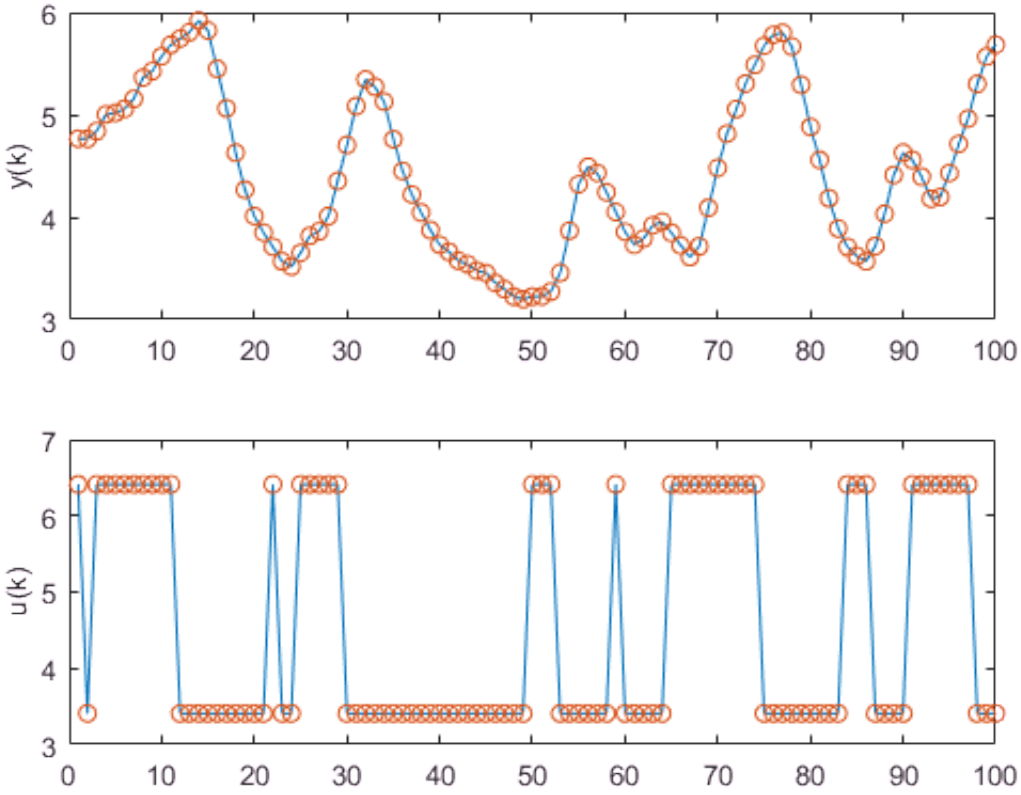
https://uk.mathworks.com/help/fuzzy/nonlinear-system-identification.html

As potential inputs for the fuzzy model, we can consider: 4 previous outputs; 6 previous inputs (in total 10)

$$\underbrace{y(k-1), y(k-2), y(k-3), y(k-4)}_{\text{iesiri anterioare}}, \underbrace{u(k-1), u(k-2), u(k-3), u(k-4), u(k-5), u(k-6)}_{\text{intrari anterioare}}$$

"exhsrch" usually involves a significant amount of computation if all combinations are tried. For instance, if 3 is selected out of 10, the total number of ANFIS models is C(10, 3) = 120.

```
%% Load data
load drydata  % u2 - inputs, y2 outputs; 1000 items in total
data_n = length(y2);
output = y2;
% input
input = [[0; y2(1:data_n-1)] ...
        [0; 0; y2(1:data_n-2)] ...
        [0; 0; 0; y2(1:data_n-3)] ...
        [0; 0; 0; 0; y2(1:data_n-4)] ...
        [0; u2(1:data_n-1)] ...
        [0; 0; u2(1:data_n-2)] ...
        [0; 0; 0; u2(1:data_n-3)] ...
        [0; 0; 0; 0; u2(1:data_n-4)] ...
        [0; 0; 0; 0; 0; u2(1:data_n-5)] ...
        [0; 0; 0; 0; 0; 0; u2(1:data_n-6)]];
data = [input output];
data(1:6, :) = [];  % remove the first 6 rows of data
input_name = char('y(k-1)','y(k-2)','y(k-3)','y(k-4)','u(k-1)','u(k-2)','u(k-3)','u(k-4)','u(k
index = 1:100;
subplot(2,1,1)
plot(index,y2(index),'-',index,y2(index),'o')
ylabel('y(k)','fontsize',10)
subplot(2,1,2)
plot(index,u2(index),'-',index,u2(index),'o')
ylabel('u(k)','fontsize',10)
```




The exhaustive search is not performed by means of matlab "exhsrch" function, but using a particular implementation to reduce the number of input combinations.

Fortunately, for dynamic system identification, we do know that the inputs should not come from either of the following two sets of input candidates exclusively:

Y = {y(k-1), y(k-2), y(k-3), y(k-4)}

U = {u(k-1), u(k-2), u(k-3), u(k-4), u(k-5), u(k-6)}

A reasonable guess would be to take two inputs from Y and one from U to form the inputs to ANFIS; the total number of ANFIS models is then C(4,2)*6=36, which is much less

```
group1 = [1 2 3 4]; % y(k-1), y(k-2), y(k-3), y(k-4)
group2 = [1 2 3 4]; % y(k-1), y(k-2), y(k-3), y(k-4)
group3 = [5 6 7 8 9 10]; % u(k-1) through y(k-6)

anfis_n = 6*length(group3);
index = zeros(anfis_n,3);
trn_error = zeros(anfis_n,1);
chk_error = zeros(anfis_n,1);
trn_data_n = 300;
% ======= Training options
```

```
% Create option set for generating initial FIS.
% genOpt = genfisOptions('GridPartition','NumMembershipFunctions',2, ...
%                         'InputMembershipFunctionType','gbellmf');
% Create option set for |anfis| command and set options that remain constant
% for different training scenarios.
% anfisOpt = anfisOptions('EpochNumber',1,...
%                         'InitialStepSize',0.1,...
%                         'StepSizeDecreaseRate',0.5,...
%                         'StepSizeIncreaseRate',1.5,...
%                         'DisplayANFISInformation',0,...
%                         'DisplayErrorValues',0,...
%                         'DisplayStepSize',0,...
%                         'DisplayFinalResults',0);
% ====== Train ANFIS with different input variables
fprintf('\nTrain %d ANFIS models, each with 3 inputs selected from 10 candidates...\n\n',...
    anfis_n);
```

```
Train 36 ANFIS models, each with 3 inputs selected from 10 candidates...
```

```
model = 1;
for i = 1:length(group1)
    for j = i+1:length(group2)
        for k = 1:length(group3)
            in1 = deblank(input_name(group1(i),:));
            in2 = deblank(input_name(group2(j),:));
            in3 = deblank(input_name(group3(k),:));
            index(model, :) = [group1(i) group2(j) group3(k)];
            trn_data = data(1:trn_data_n, [group1(i) group2(j) group3(k) size(data,2)]);
            chk_data = data(trn_data_n+1:trn_data_n+300, [group1(i) group2(j) group3(k) size(d
            in_fismat = genfis1([trn_data(:,1:end-1),trn_data(:,end)], 2, 'gbellmf');
            % Set initial FIS and validation data in option set for ANFIS training.
%             anfisOpt.InitialFIS = in_fismat;
%             anfisOpt.ValidationData = chk_data;
%           [~, t_err, ~, ~, c_err] = anfis(trn_data,anfisOpt);
            [~, t_err, ~, ~, c_err] = anfis(trn_data, in_fismat, [1], [0, 0, 0, 0], chk_data);

            trn_error(model) = min(t_err);
            chk_error(model) = min(c_err);
            fprintf('ANFIS model = %d: %s %s %s  --> trn=%.4f, chk=%.4f',model,in1,in2,in3, tr
            fprintf('\n');
            model = model+1;
        end
    end
end
```

```
ANFIS model = 1: y(k-1) y(k-2) u(k-1)  --> trn=0.0990, chk=0.0962
ANFIS model = 2: y(k-1) y(k-2) u(k-2)  --> trn=0.0852, chk=0.0862
ANFIS model = 3: y(k-1) y(k-2) u(k-3)  --> trn=0.0474, chk=0.0485
ANFIS model = 4: y(k-1) y(k-2) u(k-4)  --> trn=0.0808, chk=0.0822
ANFIS model = 5: y(k-1) y(k-2) u(k-5)  --> trn=0.1023, chk=0.0991
ANFIS model = 6: y(k-1) y(k-2) u(k-6)  --> trn=0.1021, chk=0.0974
ANFIS model = 7: y(k-1) y(k-3) u(k-1)  --> trn=0.1231, chk=0.1206
ANFIS model = 8: y(k-1) y(k-3) u(k-2)  --> trn=0.1047, chk=0.1085
ANFIS model = 9: y(k-1) y(k-3) u(k-3)  --> trn=0.0587, chk=0.0626
ANFIS model = 10: y(k-1) y(k-3) u(k-4)  --> trn=0.0806, chk=0.0836
ANFIS model = 11: y(k-1) y(k-3) u(k-5)  --> trn=0.1261, chk=0.1311
```

```
ANFIS model = 12: y(k-1) y(k-3) u(k-6)  --> trn=0.1210, chk=0.1151
ANFIS model = 13: y(k-1) y(k-4) u(k-1)  --> trn=0.1420, chk=0.1353
ANFIS model = 14: y(k-1) y(k-4) u(k-2)  --> trn=0.1224, chk=0.1229
ANFIS model = 15: y(k-1) y(k-4) u(k-3)  --> trn=0.0700, chk=0.0765
ANFIS model = 16: y(k-1) y(k-4) u(k-4)  --> trn=0.0817, chk=0.0855
ANFIS model = 17: y(k-1) y(k-4) u(k-5)  --> trn=0.1337, chk=0.1405
ANFIS model = 18: y(k-1) y(k-4) u(k-6)  --> trn=0.1421, chk=0.1333
ANFIS model = 19: y(k-2) y(k-3) u(k-1)  --> trn=0.2393, chk=0.2264
ANFIS model = 20: y(k-2) y(k-3) u(k-2)  --> trn=0.2104, chk=0.2077
ANFIS model = 21: y(k-2) y(k-3) u(k-3)  --> trn=0.1452, chk=0.1497
ANFIS model = 22: y(k-2) y(k-3) u(k-4)  --> trn=0.0958, chk=0.1047
ANFIS model = 23: y(k-2) y(k-3) u(k-5)  --> trn=0.2048, chk=0.2135
ANFIS model = 24: y(k-2) y(k-3) u(k-6)  --> trn=0.2388, chk=0.2326
ANFIS model = 25: y(k-2) y(k-4) u(k-1)  --> trn=0.2756, chk=0.2574
ANFIS model = 26: y(k-2) y(k-4) u(k-2)  --> trn=0.2455, chk=0.2400
ANFIS model = 27: y(k-2) y(k-4) u(k-3)  --> trn=0.1726, chk=0.1797
ANFIS model = 28: y(k-2) y(k-4) u(k-4)  --> trn=0.1074, chk=0.1157
ANFIS model = 29: y(k-2) y(k-4) u(k-5)  --> trn=0.2061, chk=0.2133
ANFIS model = 30: y(k-2) y(k-4) u(k-6)  --> trn=0.2737, chk=0.2836
ANFIS model = 31: y(k-3) y(k-4) u(k-1)  --> trn=0.3842, chk=0.3605
ANFIS model = 32: y(k-3) y(k-4) u(k-2)  --> trn=0.3561, chk=0.3358
ANFIS model = 33: y(k-3) y(k-4) u(k-3)  --> trn=0.2719, chk=0.2714
ANFIS model = 34: y(k-3) y(k-4) u(k-4)  --> trn=0.1763, chk=0.1808
ANFIS model = 35: y(k-3) y(k-4) u(k-5)  --> trn=0.2132, chk=0.2240
ANFIS model = 36: y(k-3) y(k-4) u(k-6)  --> trn=0.3460, chk=0.3601
```

```
% ====== Reordering according to training error
[~, b] = sort(trn_error);
b = flipud(b);  % List according to decreasing trn error
trn_error = trn_error(b);
chk_error = chk_error(b);
index = index(b,:);

% ====== Display training and checking errors
figure
x = (1:anfis_n)';
subplot(2,1,1)
plot(x, trn_error,'-',x,chk_error,'-', ...
     x,trn_error,'o',x,chk_error,'*')
tmp = x(:, ones(1,3))';
X = tmp(:);
tmp = [zeros(anfis_n,1) max(trn_error,chk_error) nan*ones(anfis_n,1)]';
Y = tmp(:);
hold on
plot(X,Y,'g')
hold off
axis([1 anfis_n -inf inf])
h_gca = gca;
h_gca.XTickLabel = [];
```

```
% ====== Add text of input variables
for k = 1:anfis_n
    text(x(k), 0, ...
        [input_name(index(k,1),:) ' ' ...
         input_name(index(k,2),:) ' ' ...
         input_name(index(k,3),:)]);
end
h = findobj(gcf,'type','text');
set(h,'rot',90,'fontsize',11,'hori','right');

drawnow

% ====== Generate input_index for bjtrain.m
[a, b] = min(trn_error);
input_index = index(b,:);
title('Training (Circles) and Checking (Asterisks) Errors','fontsize',10)
ylabel('RMSE','fontsize',10)
```