System Identification Project Part I

GROUP NR 15

STUDENTS: MATEI CRISTINA
PERPELICI ALEXANDRA
ROKALY KRISZTA

Introduction/Problem statement

- We were given a set of input and output data which is generating a 3D model of an unknown function.
- Our main goal was to find the unknown function with the help of polynomial approximation with a configurable degree m and linear regression.
- After that, we compared our approximated results with the actual output model to check if the approximation was done correctly.
- Last but not least, we performed a tuning process in order to find the best polynomial degree for our model.

Methodology

In order to obtain the requested approximation, we thought about an algorithm using linear regression .

- The main idea behind it is to find our function g(x) and compute the corresponding matrices.
- For obtaining these matrices, firstly we determined an array of powers that contains the powers for each term X1 and X2 from the polynomial approximator.
- After that, we computed the regressors matrix and the parameter column array.
- By multiplying these two, we determined the approximation matrix.
- Finally, we calculated the mean squared errors for different degrees of the polynomial.
- Analyzing the plotted errors, we obtained the best fitting degree for the approximation.

Algorithm

Computing the vector of powers p[]

First of all, we computed a vector of powers p[] corresponding to the polynomial equation that we are working with, in our case with the degree m.

We used this vector p[] in order to compute the regressors matrix PHI as follows:

- on the odd indexes of the vector p[] we have the powers of X1
- on the even indexes of the vector p[] we have the powers of X2
- we are going through the vector with a step of 2 in order to compute X1*X2 at the corresponding powers, so we'll have the elements of the regressors matrix

Algorithm

Computing the regressors matrix PHI

- The number of lines of the matrix PHI is given by: length(X1) * length(X2)
- The number of columns of the matrix PHI is given by: length(p)/2
- For taking the proper data from the 2 input vectors, we used 2 for loops => for each pair of X1&X2 we'll have a corresponding regressor vector in the matrix
- We used another for loop for the vector of powers; that index will also give us the column position
- Finally, a counter is used for the row position in the matrix

Algorithm

Computing the coefficients matrix THETA

The next step was to compute the THETA vector which contains the coefficients of the polynomial approximator for the identification data set.

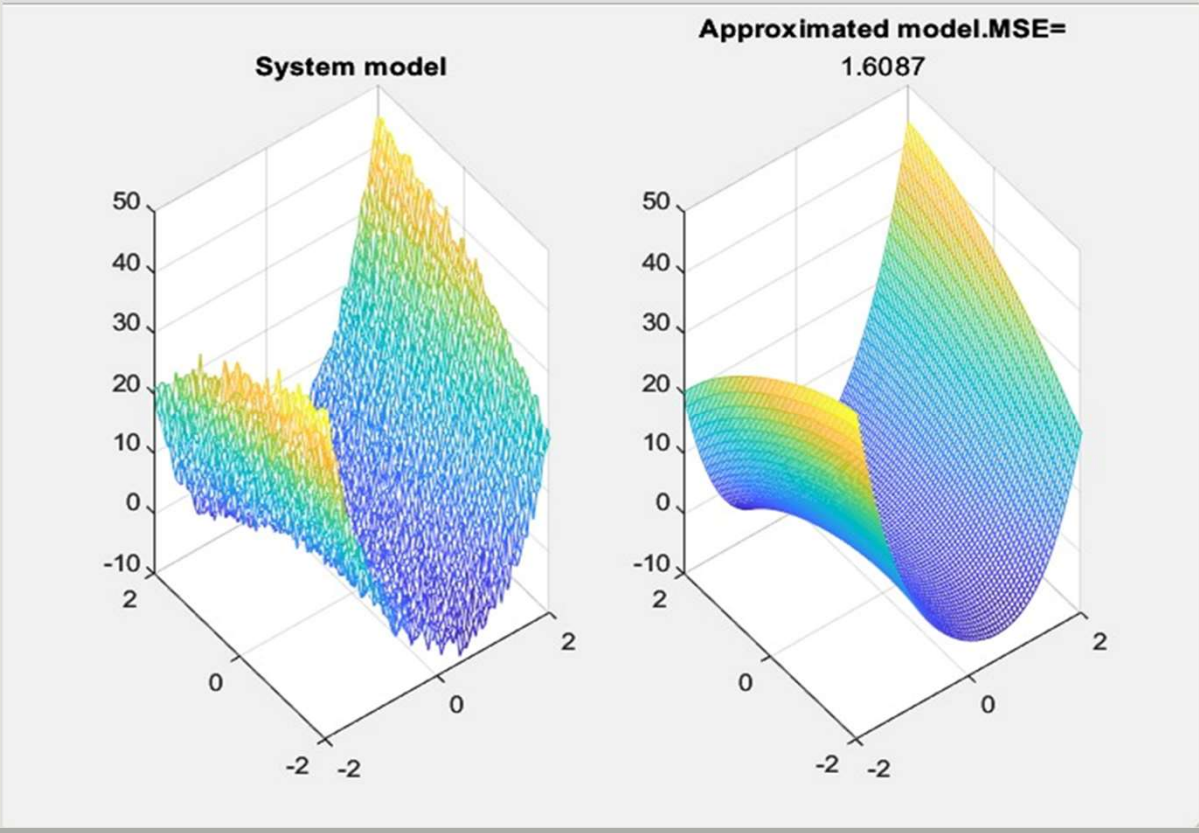
For that, we used the output identification matrix id.Y which later on we transformed into a vector v_yid, in order to be able to do the computations, and the regressors matrix PHI that we just determined.

We calculated the THETA vector by using left division between these two vectors mentioned above: THETA =PHI\v_yid.

Algorithm

Computing the approximated function y_hat

- We used the same approach in finding the regressors matrix PHI1 for the validation data set as for the identification set.
- After that, we computed the approximated output matrix for the validation data set and plotted the results.



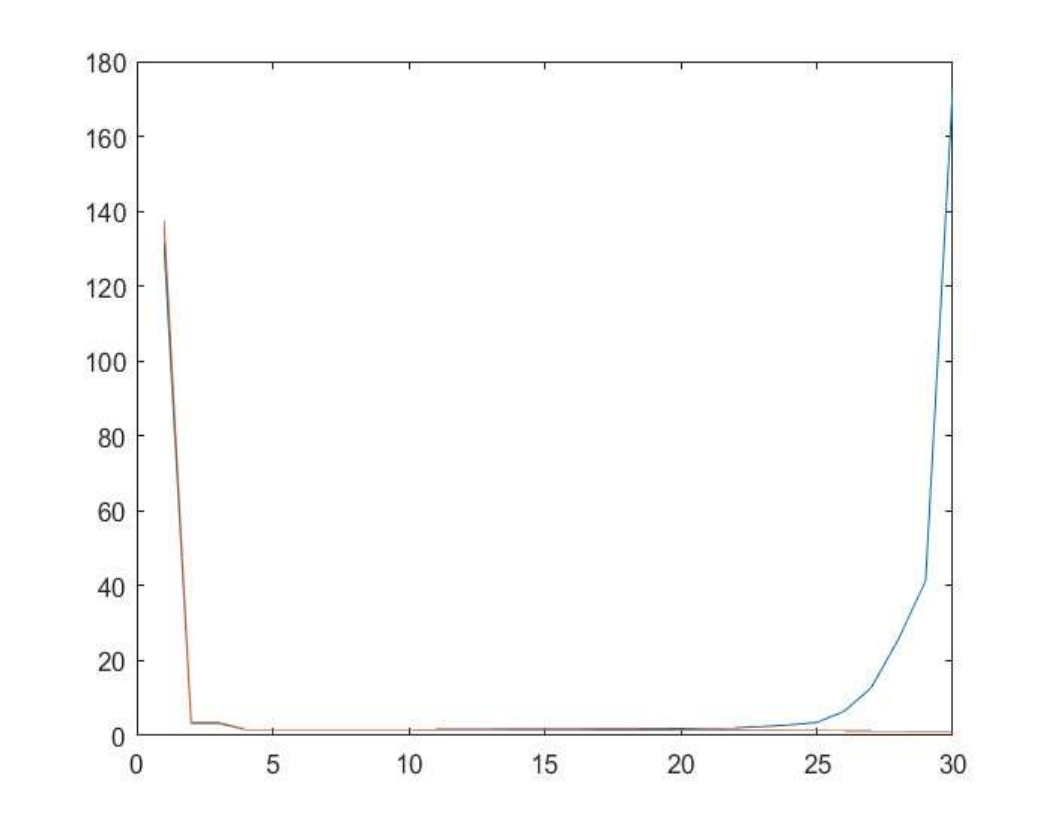
Algorithm

Tuning process

For the tuning process we put the previously described code into a for loop, so that we could verify our system with several degrees of power covering the interval [1,30].

In the end, we calculated two vectors of errors for both sets in order to plot them and see the evolution of the mean squared error corresponding to them.

By doing this, we were able to find the best fitting degree of the polynomial m which in our case is 5.



Conclusions

- To sum up all the points mentioned above, we were able to find the best polynomial approximation for our given system using the method of linear regression.
- We also plotted the errors and our best approximation next to the real system for better visual representation.

Thank you for your attention!

Any questions?

Appendix

```
%%
%%we looked at the data that we where given
clear all
load('proj_fit_15.mat')
mesh(id.X{1},id.X{2},id.Y)
%%
m=5;
for e=1:30
%we calculate an array that contains the powers of each term
c=1;
for i=1:e+1
   for j=1:e+1
      if (i+j)<=(e+2)
          p(c)=i-1;
          p(c+1)=j-1;
          c=c+2;
      end
   end
end
```

Appendix

```
%we compute the matrix phi with the identification data
phi=ones(length(id.X{1})*length(id.X{2}),length(p)/2);
nr=1;
for i=1:length(id.X{1})
  for j=1:length(id.X{2})
      for k=2:2:length(p)
          phi(nr,k/2)=(id.X{1}(i).^p(k-1))*(id.X{2}(j).^p(k));
      end
      nr=nr+1;
  end
end
%we transform the identification matrix into an array
nr2=1;
v_yid=ones(length(id.Y)*length(id.Y),1);
for i=1:length(id.Y)
  for j=1:length(id.Y)
      v_yid(nr2)=id.Y(i,j);
      nr2=nr2+1;
  end
end
```

Appendix

```
theta=phi\v_yid;
%we calculate the phi matrix with the validation data
phi1=ones(length(val.X{1})*length(val.X{2}),length(p)/2);
nr3=1;
for i=1:length(val.X{1})
   for j=1:length(val.X{2})
      for k=2:2:length(p)
         phi1(nr3,k/2)=(val.X{1}(i).^p(k-1))*(val.X{2}(j).^p(k));
      end
      nr3=nr3+1;
   end
end
%we calculate the approximation for both phi and phi1
y_val=phi1*theta;
y_id=phi*theta;
%the approximation in matrix form
nr4=1;
y_hat=ones(length(val.Y));
```

Appendix

```
for i=1:71
    for j=1:71
        y_hat(i,j)=y_val(nr4);
        nr4=nr4+1;
    end
end
nr5=1;
v_yval=ones(length(val.Y)*length(val.Y),1);
for i=1:length(val.Y)
    for j=1:length(val.Y)
        v_yval(nr5)=val.Y(i,j);
        nr5=nr5+1;
    end
end
%we calculate the error arrays
E1(e)=1/(length(val.X{1})*length(val.X{2}))*sum((v_yval-y_val).^2);
E2(e)=1/(length(id.X{1})*length(id.X{2}))*sum((v_yid-y_id).^2);
%when we reach the optimal degree, we're plotting the approximation and the validation set
if e==m
    subplot(121)
```

Appendix

```
        mesh(val.X{1},val.X{2},val.Y);title('System model');
        subplot(122)
        mesh(val.X{1},val.X{2},y_hat);
    end
end
%calculate the MSE and m_optimal is the optimal degree
[MSE,m_optimal]=min(E1);title('Approximated model.MSE=',MSE)
%we plot the error arrays for later analysis
figure;
plot(E1);hold;plot(E2);
```