TECHNICAL UNIVERSITY OF CLUJ-NAPOCA ROMANIA

Faculty of Automation and Computer Science

Time series modeling using Fourier basis functions in Systems Identification

2022 - 2023

Students: Nagy Timea, Nemes Raluca, Oprea Florin Octavian

Group 30333

Datafile index: 16

Coordinating teachers:

Prof. dr. ing. Lucian Busoniu

As. dr. ing. Zoltan Nagy

1. Description of the problem

The purpose of this report is to describe the identification process of an approximator corresponding to a time series.
Our project includes a dataset which consists of two vectors that contain information about the amount of sold items of a specific product in a building engineering store, *time* and *y*, where *time* represents the months of a year and the vector *y* contains the amount of the products that was sold for each month. The data is collected over several years.
The objective is to analyze the behavioural pattern of the regressed variable, y, and find a linear model for it, $\hat{y}$, which fits best the given data.

2. The approximator structure and the procedure to find the parameters

First, we have to divide the received data from the laboratory, the first 80% of the data as identification data and the remaining 20% as validation data, to ensure that we have a different dataset on which we can perform the verification of the found approximation. For an easier understanding, we choose to index the dataset with the index *k*, from 1 to up to *N*.

Next, we have to choose the regressors, $\boldsymbol{\varphi}$ (phi in Matlab). Find detailed explanation in section 3.

Then, the Θ (Theta) parameters are calculated automatically by the help of equation 2.1. The parameter vector, Θ, has *2*m+2* parameters: Θ = [t_0, t_1, a_1, b_1, . . . , a_m, b_m]

$$\boldsymbol{\theta} = \boldsymbol{\varphi} \backslash \boldsymbol{y}$$

Equation 2.1

The approximator vector is modelled after Equation 2.1

$$\hat{y}(k) = t_0 + t_1 k + \sum_{i=1}^{m} \left[a_i \cos\left(\frac{2\pi i k}{P}\right) + b_i \sin\left(\frac{2\pi i k}{P}\right) \right]$$

Equation 2.1

The approximation is done by multiplying the above calculated parameter vector, Θ, by the regressors for the identification data.

Then, we can verify our model on the validation data. We build the regressor matrix for the validation data too, then we use the obtained Θ from the identification data to build the approximator vector, y_hat_val. Mean – square errors are calculated for both sets of data, for the purpose of verification. Further explications about MSE can be found in section 4.

All theese calculations are done for more values of *m*, in order to obtain multiple models, so we get to choose the best one. Here, *m* means the number of terms in the Fourier series. In this case *m* has values between 1 and 7 and the best fit for the data is in case of *m* = 3.

3. Key features of our own individual solution

3.1 Choosing the regressors

The regressor $\boldsymbol{\varphi}$ is a *N x n* matrix which contains *n* regressors ($\boldsymbol{\varphi}_1$, $\boldsymbol{\varphi}_2$, … , $\boldsymbol{\varphi}_n$). Each regressor has N elements. Each row of $\boldsymbol{\varphi}$ is a regressor vector.

Our regressors contain a linear trend component and also a Fourier basis. Since the input data is expected to reflect periodicity, the Fourier basis suits our data the best.

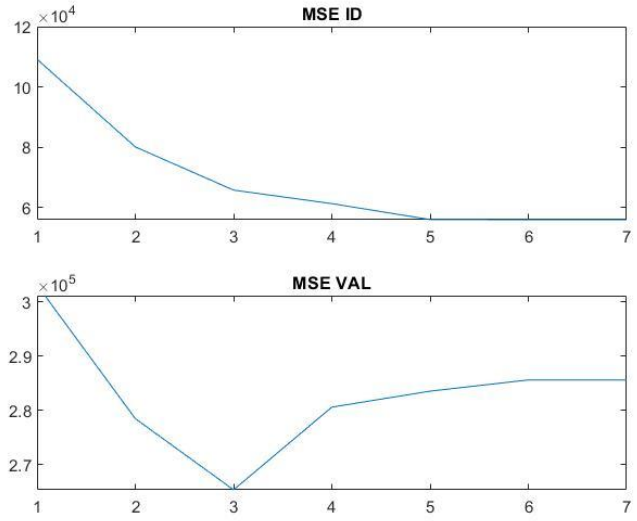
We created the regressors for the identification data *(phi_id)* which will have as indexes *i (for lines)* and *j (for columns).* After that, we noticed a rule that we followed: the number of columns of the matrix is *2+2*m,* m being the number of Fourier terms. The first column will always have the value 1 and the second column will have the value i. From here, we notice that the odd columns will have the value with *cosine (2*pi*i*(j-1)/2/p) ,* where p = 12 (number of months) and the even columns will have the value with *sine (2*pi*i*(j-2)/2/p)*. The regressors for the validation data *(phi_val)* are similar, but are built for the last 20% of the data.

4. Tuning results (at least the MSE as a function of m, either as a graph or a table).

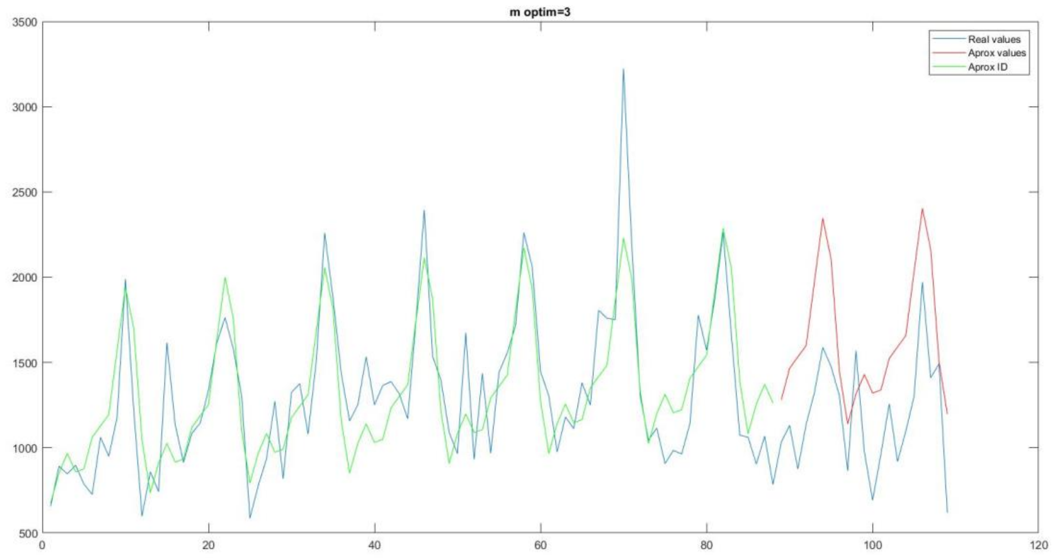
4.1 Mean squered error

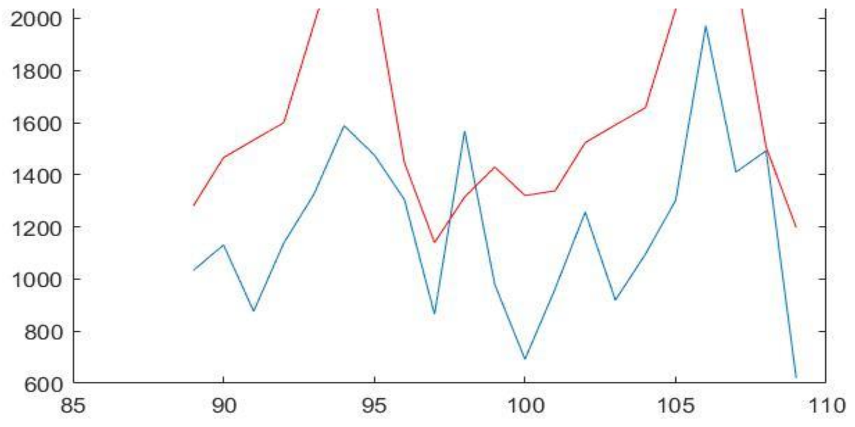
The mean-squared error is calculated for all the approximations

$$J = \frac{1}{N}\sum_{k=1}^{N}(\hat{y}(k) - y(k))^2$$



5. The representative plots pointed out above, for the optimal value of m





6. The discussion pointed out above, and an overall conclusion.

In the process of regression model selection, the mean squared error of the random regression function can be split into random noise approximation bias and variance in the estimate of the regression function.

An overfitted model is a mathematical model that contains more parameters than can be justified by the data. + modeling noise

In order to imrove the approximation, we could use a much more complex Fourier series.

7. Table of Contents

To do

-rename figures Fig 1.2: Caption + reference in text

Appendix

```
clear all;
load('product_16.mat');


p=12;
t_id = time(1:88);  %timpul pt primele 80%
t_val = time(89:109); %timpul pt ultimele 20%

y_id = y(1:88); %datele pt primele 80%
y_val = y(89:109); %datele pt ultimele 20%

for m=1:7
    figure;
    %Crearea matricii ID
    for i=1:length(t_id) % adica i = 1->88
        for j=1:2+2*m
            phi_id(i,1)=1;
            phi_id(i,2)=i;
            r = mod(j,2);
                if r==1
                    phi_id(i,j)=cos(2*pi*i*(j-1)/2/p);
                else
                    phi_id(i,j)=sin(2*pi*i*(j-2)/2/p);
                end
        end
    end


        teta=phi_id\y_id; %teta e aproximatorul, se calculeaza doar pt primele
80% din valori
        yhat_id=phi_id*teta; %aproximarea pt primele 80%


    %Crearea matricii val
    for i=1:length(t_val) % adica i = 1->21
        for j=1:2+2*m
            phi_val(i,1)=1;
            phi_val(i,2)=t_val(i);
            r = mod(j,2);
                if r==1
                    phi_val(i,j)=cos(2*pi*t_val(i)*(j-1)/2/p);
                else
                    phi_val(i,j)=sin(2*pi*t_val(i)*(j-2)/2/p);
                end
        end
    end

    %teta nu se mai calculeaza pt ultimele 20%, ci se foloseste cea de
    %dinainte pt a calcula aproximarea pe ultimele 20%
    yhat_val=phi_val*teta; %aproximarea pt ultimele 20%

    %MSE pt primele 80%
    s=0;
    for k=1:length(t_id)
        s=s+(y_id(k)-yhat_id(k)).^2;
    end
    MSE_id(m)=s/length(t_id);


    %MSE pt ultimele 20%
    s=0;
    for k=1:length(t_val)
        s=s+(y_val(k)-yhat_val(k)).^2;
    end
    MSE_val(m)=s/length(t_val);

    plot(time,y),title("m=",m);
    hold on;
    plot(t_val,yhat_val,'r');
    plot(t_id,yhat_id,'g');legend('Real values','Aprox values','Aprox ID');


end
```

```
figure
m=1:7;
subplot(211);plot(m,MSE_id);title("MSE ID");
hold on
subplot(212);plot(m,MSE_val);title("MSE VAL");

figure;
clear all
m=3;
load('product_16.mat');
p=12; %folosit mai mult pt formula la matrice, puteam sa in locuiesc direct cu
12, dar arata mai frumos asa :)


t_id = time(1:88);  %timpul pt primele 80%
t_val = time(89:109); %timpul pt ultimele 20%


y_id = y(1:88); %datele pt primele 80%
y_val = y(89:109); %datele pt ultimele 20%


%Crearea matricii ID
for i=1:length(t_id) % adica i = 1->88
    for j=1:2+2*m
        phi_id(i,1)=1;
        phi_id(i,2)=i;
        r = mod(j,2);
            if r==1
                phi_id(i,j)=cos(2*pi*i*(j-1)/2/p);
            else
                phi_id(i,j)=sin(2*pi*i*(j-2)/2/p);
            end
    end
end
    teta=phi_id\y_id; %teta e aproximatorul, se calculeaza doar pt primele 80%
din valori
    yhat_id=phi_id*teta; %aproximarea pt primele 80%

%Crearea matricii ID
for i=1:length(t_val)
    for j=1:2+2*m
        phi_val(i,1)=1;
        phi_val(i,2)=t_val(i);
        r = mod(j,2);
            if r==1
                phi_val(i,j)=cos(2*pi*t_val(i)*(j-1)/2/p);
            else
                phi_val(i,j)=sin(2*pi*t_val(i)*(j-2)/2/p);
            end
    end
end
yhat_val=phi_val*teta; %aproximarea pt ultimele 20%


plot(time,y),title("m optim=3");
hold on;
plot(t_val,yhat_val,'r');
plot(t_id,yhat_id,'g');legend('Real values','Aprox values','Aprox ID');


figure;
plot(time(89:109),y(89:109)),title("m optim=3 validation data");
hold on;
plot(t_val,yhat_val,'r');
```