System Identification Project

MATLAB®

UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

A brief description

We are given:

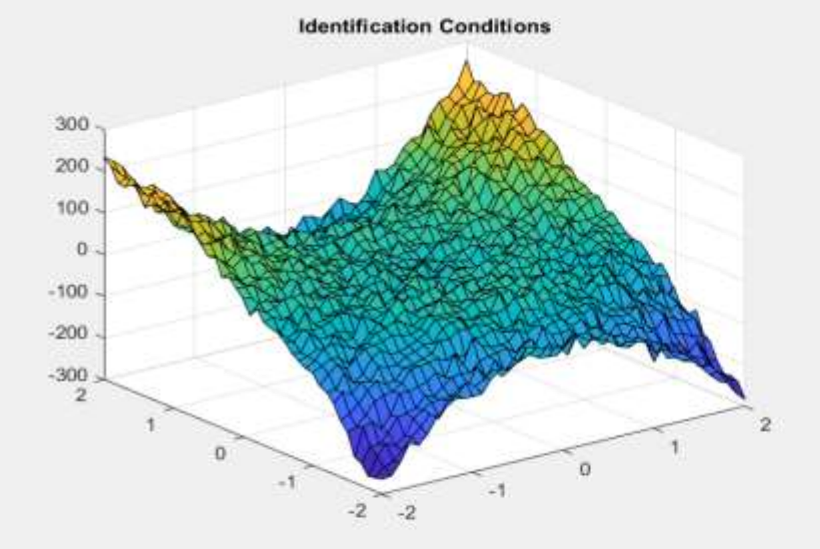
A set of grid coordinates X for the inputs, where X is a cell array of two vectors, each vector X{1}(i),X{2} containing n grid points for input dimension dim.

A set of corresponding outputs Y, a matrix of size $n \times n$, where Y(i,j) is equal to the value of f at point (X{1}(i),X{2}(j)).

Using the identification set we calculate Φ matrix (the regressors matrix) and further using that we can generate the values of θ vector.

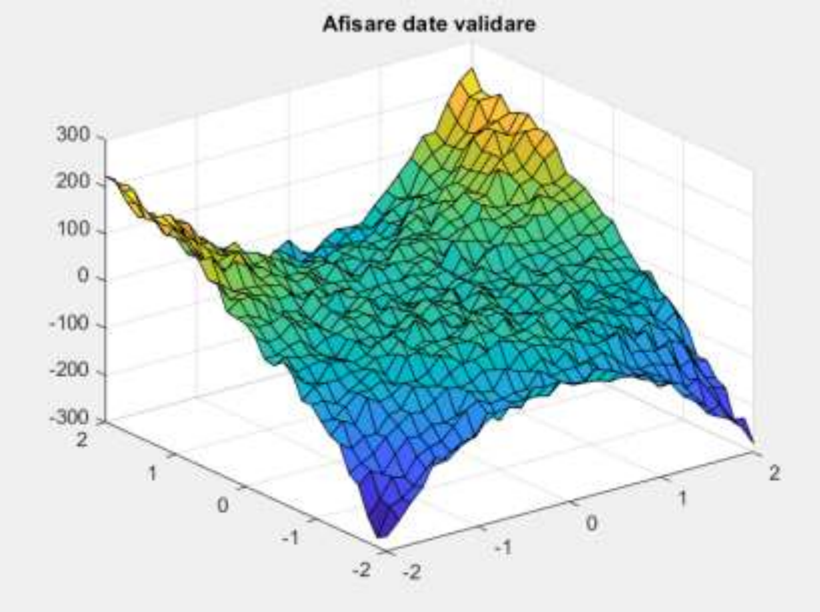
After generating the θ vector we calculate the Φ matrix for de validation set using it to create our approximation (yhat).

Understanding the project

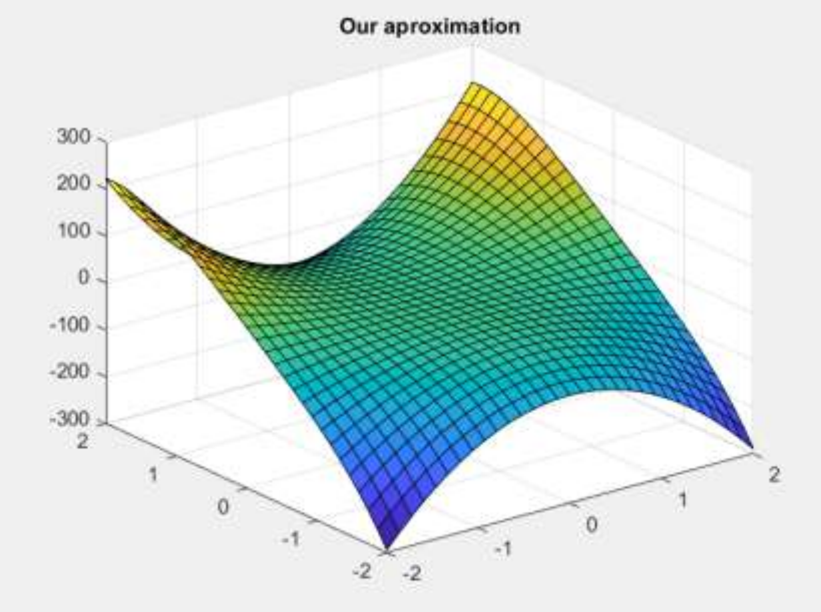


Step 1

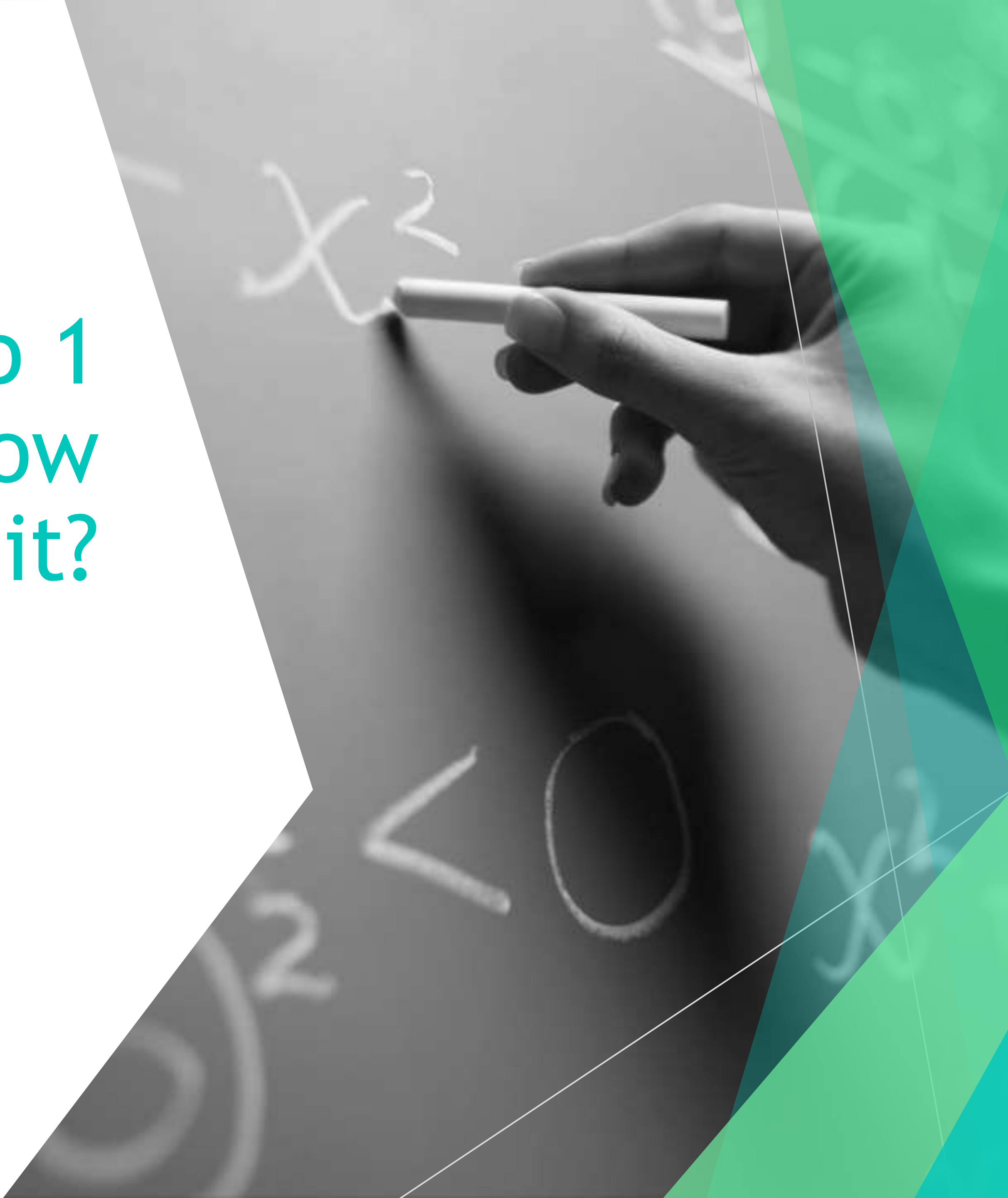
θ



Step 2



Step 1
Who is θ and how do we calculate it?



To determine how many columns we have in the Φ matrix we observe:

$n=1 \Rightarrow 3$ terms
$n=2 \Rightarrow 6$ terms
$n=3 \Rightarrow 10$ terms
$n=4 \Rightarrow 15$ terms

We can see that for the n-th degree we use the formula:

$$\text{No. of Terms on each line} = \frac{(n+1)(n+2)}{2}$$

The matrix of regressors for a specific degree has this formula for each line

$X^2 + xy + y^2 + x + y + 1$ $\quad n=2$

$x^3 + x^2y + xy^2 + y^3 + x^2 + xy + y^2 + x + y + 1$ $\quad n=3$

$x^4 + x^3y + x^3 + x^2y^2 + x^2y + x^2 + xy^3 + xy^2 + xy + x + y^4 + y^3 + y^2 + y + 1$ $\quad n=4$

We observe that the sum is symmetric

For this example we consider grade = 4

$x^{n-0}y^0 + x^{n-1}y^1 + x^{n-2}y^2 + x^{n-3}y^3 + x^{n-4}y^4 +$ pt $n = grad$

$+ x^{n-0}y^0 + x^{n-1}y^1 + x^{n-2}y^2 + x^{n-3}y^3 +$ pt $n = grad - 1$

$+ x^{n-0}y^0 + x^{n-1}y^1 + x^{n-2}y^2 +$ pt $n = grad - 2$

$+ x^{n-0}y^0 + x^{n-1}y^1 +$ pt $n = grad - 3$

$+ x^0y^0$ $\quad n = grad - 4$

What the system of equations looks like:

$$\begin{bmatrix} y(1) \\ y(2) \\ \vdots \\ y(N) \end{bmatrix} = \begin{bmatrix} f_1(1) & f_2(1) & \dots & f_n(1) \\ f_1(2) & f_2(2) & \dots & f_n(2) \\ \vdots & \vdots & & \\ f_1(N) & f_2(N) & \dots & f_n(N) \end{bmatrix} \cdot \begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix}$$

Where

$$n = \frac{(grad + 1)(grad + 2)}{2}$$

N = length of x_1 multiplied with the length of x_2

Step 2
Creating our aproximation

My biggest issue with the project. (symbolic variables)

expand((1+a+b)^degree))

degree = 2

$$\Rightarrow a^2+2ab+2a+b^2+2b+1$$

use coeffs -> [1 2 2 1 2 1]

$$\text{PHi} \rightarrow \left[\frac{a^2}{1} \quad \frac{2ab}{2} \quad \frac{2a}{2} \quad \frac{b^2}{1} \quad \frac{2b}{2} \quad 1\right]$$

Conclusion

The task I had for the project was simple, yet complicated. Creating the approximation for a 3D system using linear regression appeared to be an interesting topic that made me push my mind to find the best algorithm I could and make the code work as I wanted

My conclusion regarding this project is that despite the hard beginning I had in understanding the task, with the help of my project teacher Tudor I managed to make small steps in the right direction and after a lot of work that I put into this project I an algorithm that calculates a correct approximation using linear regression that is fast and optimized.

My goal was to write a code that can be understood by everyone and I hope that I did a good job presenting my idea to you.

```
% Algoritm pentru substitutia valorilor simbolice una cate
una
syms a b;
PHI = [];
k= expand((1+a+b).^degree)

child = children(k);
L = length(child);
for index=1:L
    regresor(index) = child{index};
    coe(index) = coeffs(child{index});
    row(index) = regresor(index)/coe(index);
end


for i = 1:N
    Y = [Y, y(i,:)];
    ROW = subs(row,a,x1(i));
    for j = 1:N
        ROW2 = subs(ROW,b,x2(j));
        PHI = [PHI; ROW2];
    end
end
PHI = double(PHI);
theta = PHI \ Y';
```

```
%% INCEPUT DE PROGRAM
tic;
clear; clc;
%% INCARCAM SI CITIM DATELE
load("proj_fit_19.mat")
%Date Identificare
x1 = id.X{1};
x2 = id.X{2};
y = id.Y;


%Date Validare
valx1 = val.X{1};
valx2 = val.X{2};
valy = val.Y;


Yval = []; % Reshape matricea valy pentru a fi o singura linie
YhatFinal = []; % Matricia Yhat care este pentru degree ul cel mai optim
Y = []; % Acelasi lucru ca la Yval


MSE = []; % Vectorul MSE pentru Erori la Validare
MSEid = []; % Vectorul MSE pentru Erori la Identificare


n=30; % Aici alegem Degree ul
```

```
%% AFLAREA MATRICII PHI, YHAT, THETA, MSE
for k=1:n % for-ul care face cate o matrice PHI pentru fiecare degree
Y = reshape(y, [], 1); % dam reshape la y intr un vector coloana sa il pot afla pe theta
Yval = reshape(valy, [], 1); % dam reshape la valy ca sa putem afla yhat (avem nevoie de matricie linie)
valreshape = ((k+1)*(k+2))/2; % un fel de suma lui Gauss, ca sa stim cum trebuie sa i dam reshape lui PHI in functie de degree
PHIval = []; % declaram o matrice PHI de validare la fiecare iteratie a lui k
PHI = []; % declaram o matrice PHI la fiecare iteratie a lui k

% DATELE DE IDENTIFICARE
for i=1:length(x1) %for-ul care creaza linile matricii PHI. selecteaza primul element din x1 care se inmulteste cu toate valorile din x2, etc. Contorul lui x1
row = []; % cream un rand nou la fiecare linie noua
for j=1:length(x2) % for-ul care creaza fiecare element de pe o linie. Contorul lui x2
m = k; % m este gradul ca sa stim de unde coboara contorul
while(m >= 0) % cat timp m este mai mare decat 0
contor = 0; % avem un contor care ia 0 si va creste pana la m
while(contor <= m) % cat timp contorul este mai mic decat m
row = [row; x1(i)^(contor)*x2(j)^(m-contor)]; % contruim fiecrae element astfel: a^0*b^m | a^1*b^m-1 | a^2*b^m-2 | etc... (in coloana)
contor = contor + 1; % crestem contorul
end
m = m - 1; % scadem m-ul
end
end
PHI = [PHI; row]; % concatenam fiecare rand in matricea PHI (va fi un vector coloana)
end
PHI = reshape(PHI, valreshape, []); % dam reshape la matricea PHI in functie de ce degree aste folosind VALRESHAPE ce l am calculat mai sus
PHI = PHI'; % transpunem matricea PHI ca sa o putem impartii cu matricea Y
theta = PHI \ Y; % aflam THETA
Yhatid = PHI * theta; % aflam YHAT pentru datele de identificare
N = length(x1); % N este lungimea vectorului x din datele de identificare
for c=1:N % facem suma de erori cu un for de la 1 la N
mse = (Y(c)-Yhatid(c)).^2; % calculam MSE ul cu formula de Y - Yhat de Identificare
end
MSEid(k)=1/N * mse; % folosim formula finala pentru MSE care este suma inmultita cu 1/N

if k == 1
Minid = MSEid(1);
elseif MSEid(k)<Minid
Minid = MSEid(k);
YhatIDFinal = Yhatid;
end
```

```
% DATELE DE VALIDARE
% pentru datele de validare facem acelasi lucru ce am facut si pentru
% datele de identificare :)
for i=1:length(valx1)
rowVal = [];
for j=1:length(valx2)
m = k;
while(m >= 0)
contor = 0;
while(contor <= m)
rowVal = [rowVal; valx1(i)^(contor)*valx2(j)^(m-contor)];
contor = contor + 1;
end
m = m - 1;
end
end
PHIval = [PHIval; rowVal];
end
PHIval = reshape(PHIval, valreshape, []);
PHIval = PHIval';
yHat = PHIval * theta;


N = length(valx1); % N este lungimea vectorului x din datele de validare
for c=1:N
mse = (Yval(c)-yHat(c)).^2;
end
MSE(k)=1/N * mse;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%
% alegem primul minim ca fiind MSE(1) iar dupa in else if verificam %
% daca gasim un MSE mai bun decat cel gasit anterior iar daca este mai%
% bun, YhhatFinal o sa devina Yhat de degree-ul unde am gasit MSE-ul %
% cel mai mic %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%
if k == 1
Min = MSE(1);
elseif MSE(k)<Min
Min = MSE(k);
YhatFinal = yHat;
end

end
```

```
%% GRAFICELE DE MSE-uri
% plotez grafu de erori
b = 1:n;
figure;
subplot(221)
plot(b, MSE);


title(["MSE graphic (from 1 to " num2str(n)]);
subplot(223)
plot(b, MSEid);


title(["MSEid graphic (from 1 to " num2str(n)]);


subplot(222)
plot(b, MSE);
axis([1 30 -5 40])
title(["Zoomed MSE graphic (from 1 to " num2str(n)]);


subplot(224)
plot(b, MSEid);
axis([1 30 -5 40])
title(["Zoomed MSEid graphic (from 1 to " num2str(n)]);
```

```
%% GRAFICUL DATELOR DE IDENTIFICARE
%conditiile initiale (id X si Y )
figure;
surf(x1, x2, y);
title("Identification set");
%% GRAFICUL DATELOR DE VALIDARE
%conditiile de verificare (val X1 si X2 si Yhat)
YHAT = reshape(YhatFinal, 31, 31);
figure
surf(valx1,valx2,val.Y);
title("Validation set");
%% APROXIMAREA VALIDARE
figure
surf(valx1, valx2, YHAT);
title("Our aproximation for the validation set");
%% APROXIMAREA IDENTIFICARE
figure
YHATID = reshape(YhatIDFinal, 41, 41);
surf(x1, x2, YHATID);
title("Our aproximation for the identification set");
%% VALIDAREA SI APROXIMAREA PE ACELASI GRAFIC
YHAT = reshape(YhatFinal, 31, 31);
figure
surf(valx1,valx2,val.Y);
title("Afisare date validare");
hold on;
surf(valx1, valx2, YHAT);
title("The validation set and the aproximation on the same graph");

toc;
```