[[1]](#footnote-1)

Fuzzy Identification of Systems and Its Applications to Modeling and Control

*Abstract*— This work is reproduction of the work done by Michio Sugeno and Tomohiro Takagi [1]. It presents a mathematical technique for creating a fuzzy model of a system with fuzzy implications and reasoning. The description of a fuzzy subspace of inputs is the premise of an implication. As result, the input-output relationship is linear. The identifying method the output-input data of a system is then displayed. There are two applications. The method’s applicability to industrial operations is further discussed: a body of water in the steel-making process, there is a cleaning process and a converter.

# INTRODUCTION

The primary goal of this study [1] is to propose a mathematical technique for creating a fuzzy system model. In earlier studies, the attempt was made to form a fuzzy implication containing fuzzy variable function in unimodal form. These variables were called linguistic variables as they were linguistically understood. But there was the difficulty in modeling using these linguistic variables as we may have to use multiple variables.

In this study, we tried to reduce these linguistic variables. In order to do that we must consider multidimensional thinking as we are dealing with a multi-variable system. We know that in general, building a model using input-output data must have two factors: technique for identification of the system and the mathematical tool for expressing it. As a result, we proposed a simple yet extensive mathematical tool as well as fuzzy implications based on fuzzy space partition. Also the method for identification of the system using its input-output data is shown.

This paper is divided into 6 sections. Section II describes format of if…then rule. Section III describes the algorithms. Section IV describes the mathematical background of before mentioned algorithm. The algorithm is explained with the help of few examples given in section V. After that, the two applications of the method: water cleaning process, steel-making process is discussed and simulated in section VI.

# Format of Fuzzy Implication

## A. Format of Fuzzy variables

The membership function of a fuzzy set A can be denoted as A(x), (). The fuzzy variables are assumed to be linear with minimum value of ‘0’ and maximum value of ‘1’. As a result, the truth table for the statement "x is A and y is B" may be reduced to – A(x) ^ B. (y).

here,

^ : min operation.

X : universe of discourse

## B. Format of Implications

We have defined implication R as follows:

R: If x1 is A1 and ……. and xk is Ak

Or

then

Or

here,

y : Consequence variable

x1 – xk : Premise variables

A1 - Ak : Membership function

f : Logical function

g : Output function of y with x1 – xk

: Consequence parameters

# Algorithms

## A. Algorithm of Reasoning

Let’s assume we have ‘n’ implications R (I = 1, … ,n) as shown in above and also, we have ().

In the procedures that follows, the value of y is deduced:

* For every Ri, y is given by
* Now, y = yi is given by
* Let for simplicity. Thus, the final output y is calculated by finding weighted mean as shown below

This algorithm is explained using example 1 in section V.

## B. Algorithm of Identification

This algorithm is divided into three parts:

1. Choice of Premise Variables: Out of given possible inputs premise variables are chosen then optimum consequence parameters are found using it. We then calculate errors in the output and find performance index, which is root mean square value of errors. Thus, premise variables are improved in each step choosing least performance index among different choices. To choose premise variables a heuristic search method is used. This method is further explained in section IV.
2. Premise Parameter Identification: In this step optimum premise parameters are found with respect to the above premise variables. This is done by dividing fuzzy variables into different fuzzy sub-spaces like “small, big etc.”

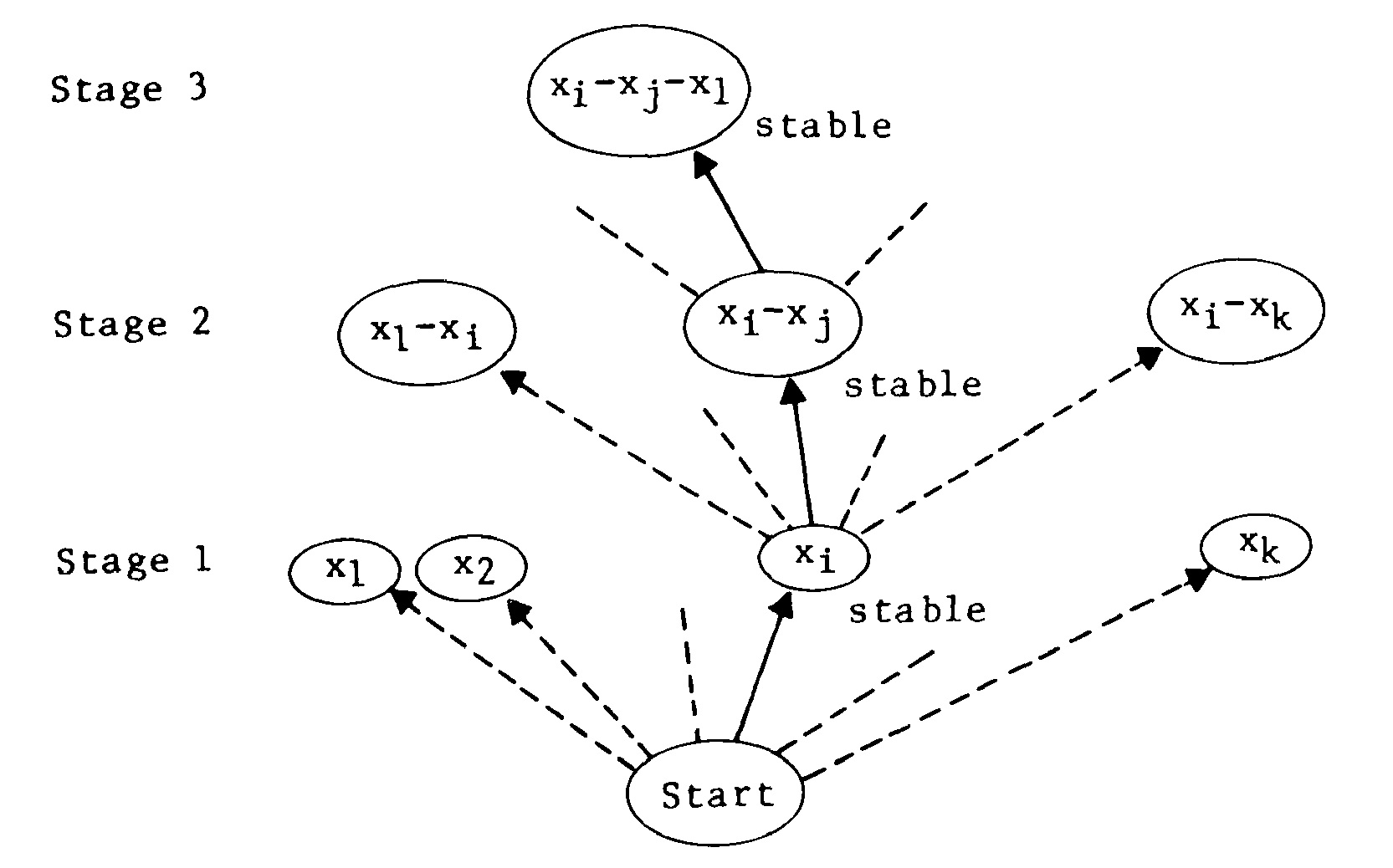
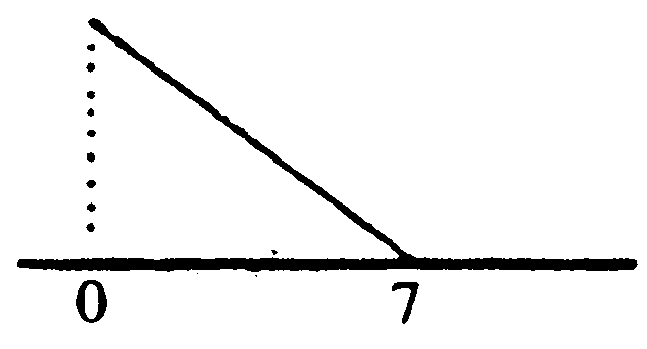


Fig 2: Choice of parameter variable

1. Consequence Parameter Identification: Now using these parameters consequences are calculated.

Now, let us see the algorithm step by step:

Step 1: A premise parameter is chosen and is divided into subspaces like “small, big etc.” Let’s assume x1 is chosen and is divided into two fuzzy subspaces as follows:

x1 is small 

x1 is big 

Let us call this model 1-1. Similarly, for xk we can have a model 1-k, which is,

xk is big 

xk is small 

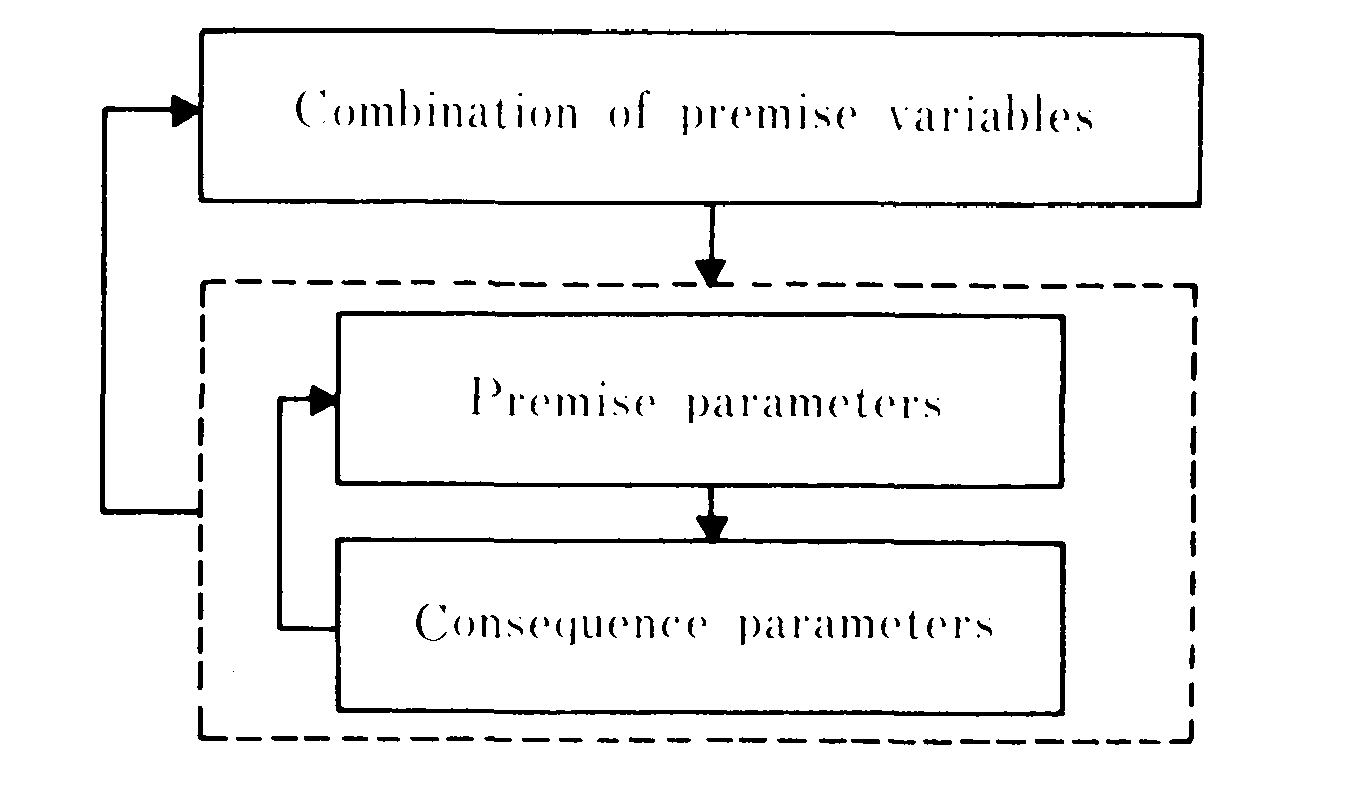


Fig 1: Algorithm outline

Step 2: Now we will find consequence parameters for every combination of subspaces calculated in step 1. This is done using the mathematical equations as stated in section IV. The process is further explained in example. Now for each rule or combination a performance index is found, which is root mean square value of the error.

Step 3: Now the model with least performance index is chosen and rest are discarded. This model is called as stable form.

Step 4: After finding stable model we will move to further divide variables in model 2-k, model 3-k etc. Let’s now have two inputs in the input premise arguments. Then all the xi – xj combination will be covered under the step and for each variable of xi and xj the subspace will be divided in two fuzzy subspaces which are generally ‘big’ and ‘small’. For model xi - xi the ranges will be divided into four subspaces namely ‘small’, ‘big’, ‘medium small’ and ‘medium big’. So, we for each implication we will get model 2-j. Then we will again go with the model having least performance index. The process is shown in fig 2.

Step 5: Now, we will repeat the process starting from step 2 until one of the following conditions are fulfilled.

* The final performance index value is less than desired value.
* Number of premise variables exceeds the maximum permissible number of rules.

# Mathematics

Let the system be defined by:

R1: If x1 is and ……. and xk is Ak then

⁞ ⁞ ⁞

⁞ ⁞ ⁞

Rn: If x1 is and ……. and xk is Ak then

Therefore, output can be written as

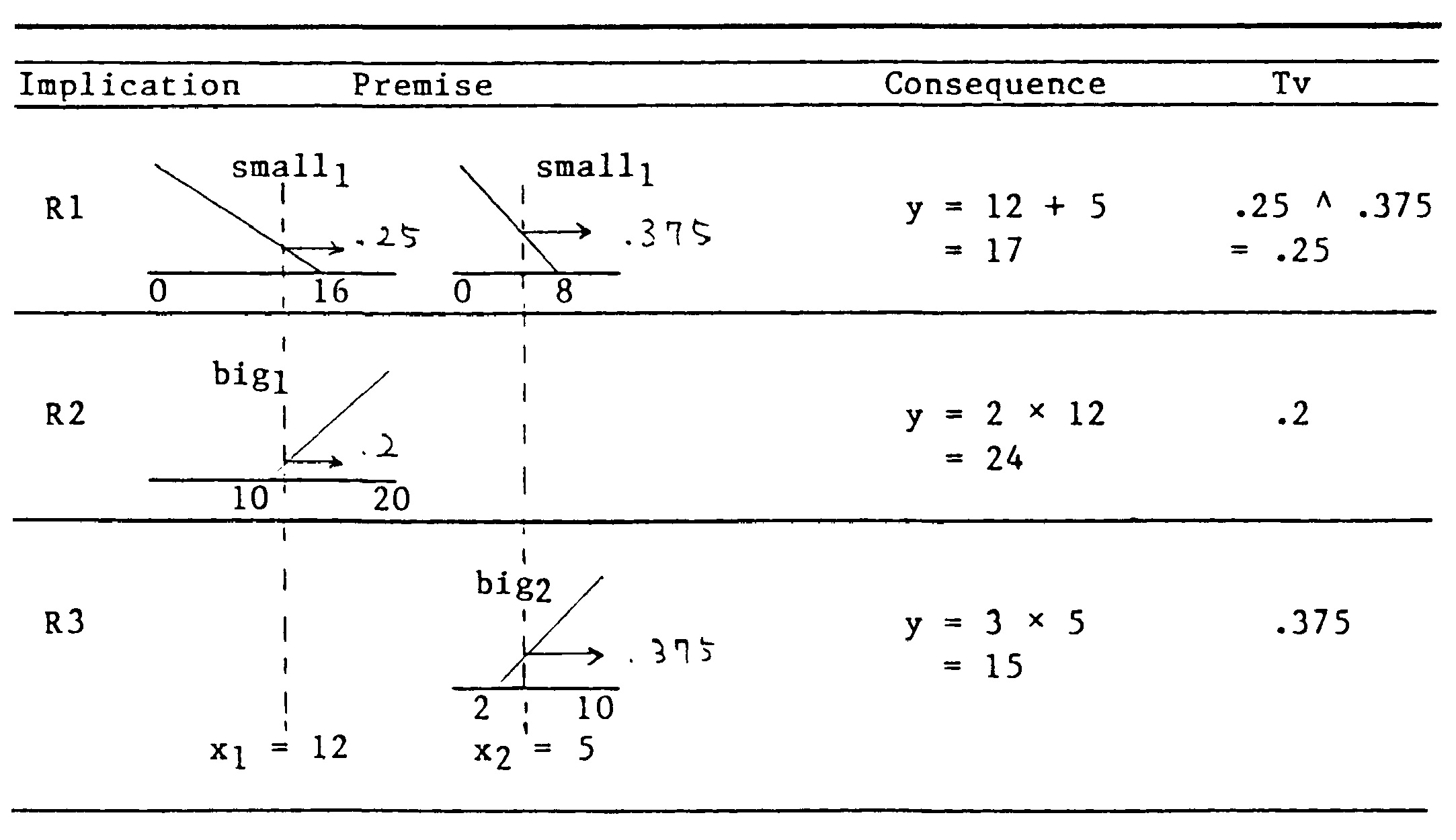


Fig 3: Tabulated data

Thus,

here,

Let,

here,

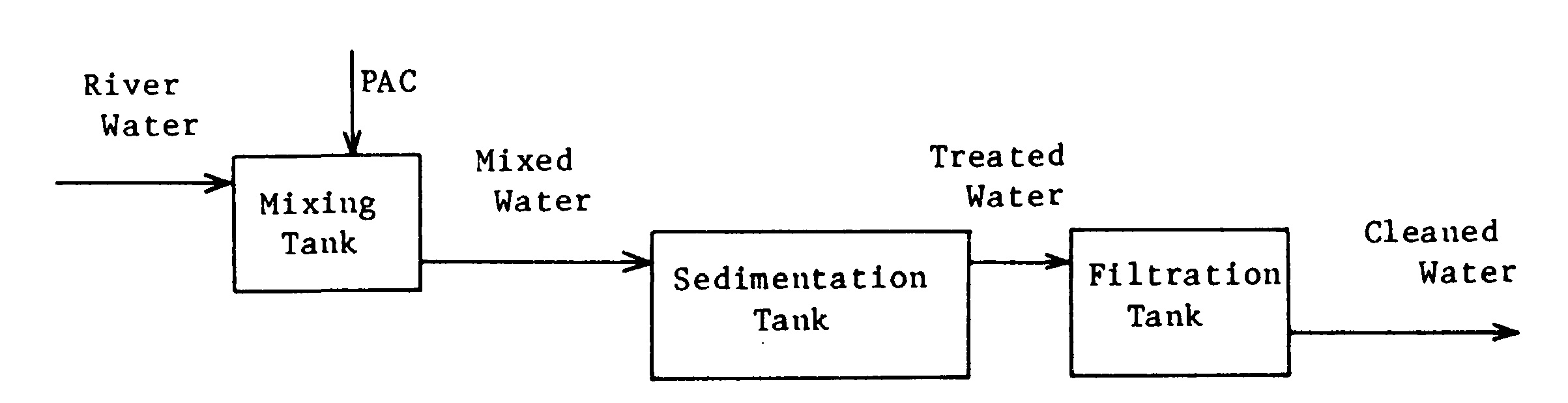
If

and

such that,

Then,

# Examples



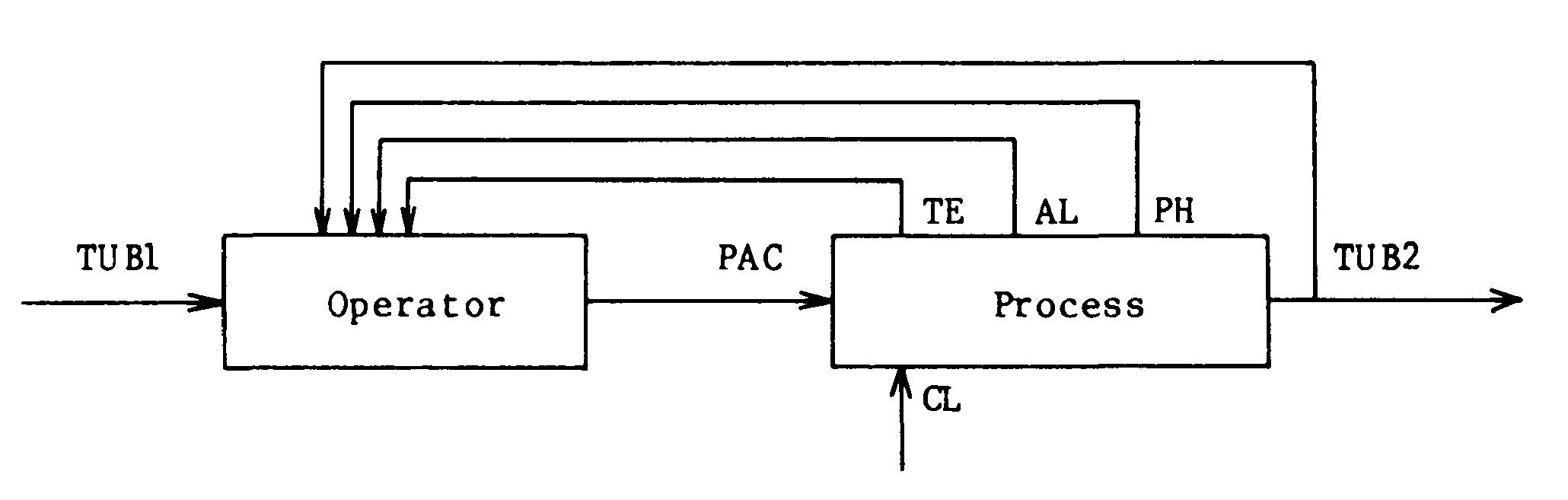


Fig 4: Water Cleaning Process

## A. Example 1

Let us have following three relations:

R1: If x1 is small and x2 is small then y = x1 + x2

R1: If x1 is big then y = 2×x1

R1: If x1 is big then y = 3×x2

Fig 3 shows reasoning process of each relation when x1 = 12, x2 = 5.

= 0.25

Now,

y = 17.8

# Simulation Results

Let’s assume we have ‘n’ implications

## A. Water Cleaning Process

In the process of cleaning water turbidity, PH etc. are analyzed then substances like PAC and chlorine are introduced to murky river water and blended with it in a mixing tank. After that, the combined water is pushed into a sedimentation tank, where the turbid portion is blended with PAC and settles to the bottom. After sedimentation, the purified water is sent into a filter tank, where it is cleaned. The process is shown in the fig 4

* MATLAB Code

close all; clear; clc;

% TUB1 PH TE AL PAC TUB2

data = [10 7.1 18.8 53 1300 1;

17 7 18.6 50 1300 1;

22 7.3 19.4 46 1400 2;

50 7.1 19.5 40 1400 1;

9 7.3 23.3 48 900 4;

11 7.1 20.7 50 900 1;

12 7.2 21.3 50 900 3;

14 7.2 23.6 53 900 4;

35 7 17.8 35 1200 1;

20 7 16.6 40 1100 1;

20 6.9 17.8 42 1100 1;

18 7.1 17.3 40 1100 1;

12 7.2 18.8 55 900 3;

8 7.2 18 50 1000 1.5;

11 7.1 19.2 49 1000 2;

50 7 18 37 1200 1.5;

35 7 17.7 42 1200 1.5;

30 7 17.3 41 1100 1.5;

16 7.1 19.3 42 1100 3];

input = [data(:,1:4) data(:,6)];

output = data(:,5);

PH = input(:,2);

AL = input(:,4);

TE = input(:,3);

k = size(data,2) - 1; % Number of variables

m = size(data,1); % Number of data points

n = 8; % Maximum number of rules

TUB1min = min(data(:,1)); TUB1max = max(data(:,1));

PHmin = min(data(:,2)); PHmax = max(data(:,2));

TEmin = min(data(:,3)); TEmax = max(data(:,3));

ALmin = min(data(:,4)); ALmax = max(data(:,4));

PACmin = min(data(:,5)); PACmax = max(data(:,5));

TUB2min = min(data(:,6)); TUB2max = max(data(:,6));

mPH = 1/(PHmin-PHmax); cPH = PHmin/(PHmin-PHmax); cPH1 = PHmax/(PHmax-PHmin);

mAL = 1/(ALmin-ALmax); cAL = ALmin/(ALmin-ALmax); cAL1 = ALmax/(ALmax-ALmin);

mTE = 1/(TEmin-TEmax); cTE = TEmin/(TEmin-TEmax); cTE1 = TEmax/(TEmax-TEmin);

%% Model 3-k

n = 8;

beta = zeros(m,n);

b = beta;

for i = 1:n

for j = 1:m

smallPH(j) = -mPH\*PH(j)+cPH1;

smallAL(j) = -mAL\*AL(j)+cAL1;

smallTE(j) = -mPH\*TE(j)+cTE1;

bigPH(j) = mPH\*PH(j)+cPH;

bigAL(j) = mAL\*AL(j)+cAL;

bigTE(j) = mTE\*TE(j)+cTE;

switch i

case 1

b(j,i) = min([smallPH(j) smallAL(j) smallTE(j)]);

case 2

b(j,i) = min([smallPH(j) smallAL(j) bigTE(j)]);

case 3

b(j,i) = min([smallPH(j) bigAL(j) smallTE(j)]);

case 4

b(j,i) = min([smallPH(j) bigAL(j) bigTE(j)]);

case 5

b(j,i) = min([bigPH(j) smallAL(j) smallTE(j)]);

case 6

b(j,i) = min([bigPH(j) smallAL(j) bigTE(j)]);

case 7

b(j,i) = min([bigPH(j) bigAL(j) smallTE(j)]);

case 8

b(j,i) = min([bigPH(j) bigAL(j) bigTE(j)]);

end

end

beta(:,i) = b(:,i)./sum(b(:,i));

end

X = beta;

for i = 1:n

X = [X repmat(beta(:,i),1,k).\*input];

end

P = pinv(X'\*X)\*X'\*output;

p = P(1:n)';

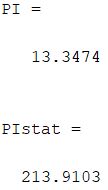


Fig 6: Results of water cleaning process

for i = 1:k

p = [p; P(n\*i+1:n\*(i+1))'];

end

x = [ones(m,1) input];

y = x\*p;

y = b.\*y;

Y = [];

for i = 1:m

Y = [Y; sum(y(i,:))./sum(b(i,:))];

end

Y = X\*P;

PI = sqrt(sum((Y - output).^2)/m)

% Statistical model

% PAC = 9.11\*sqrt(TB1) - 79.8\*PH + 12.7\*CL + 1255.6

Ystat = [994.7; 995.9; 1119.6; 1151.1; 1409.4; 1066.4; 1068.9; 1012.3; 1286.8; 1246.8; 1151.4; 1199.5; 1159.4; 985.7; 1009.3; 1038.2; 1398.3; 1290.6; 1038.5];

PIstat = sqrt(sum((Ystat - output).^2)/m)

%Plot

figure(1);

scatter(output,Y);

grid on;

hold on;

plot([500 1500],[500 1500]);

xlabel('Operator');

ylabel('Model');

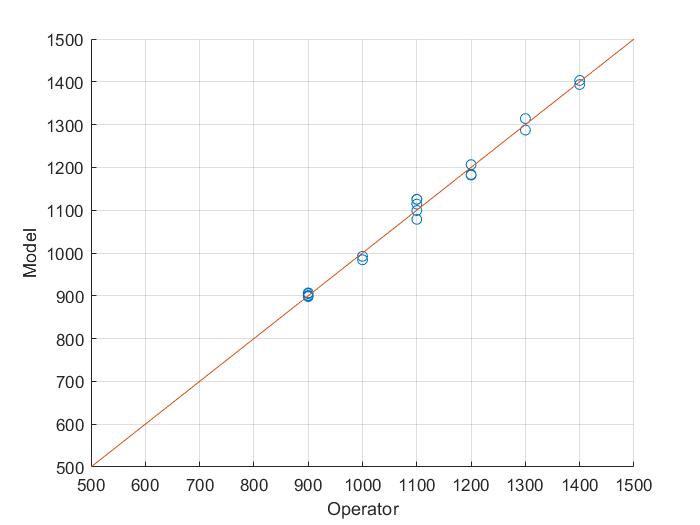
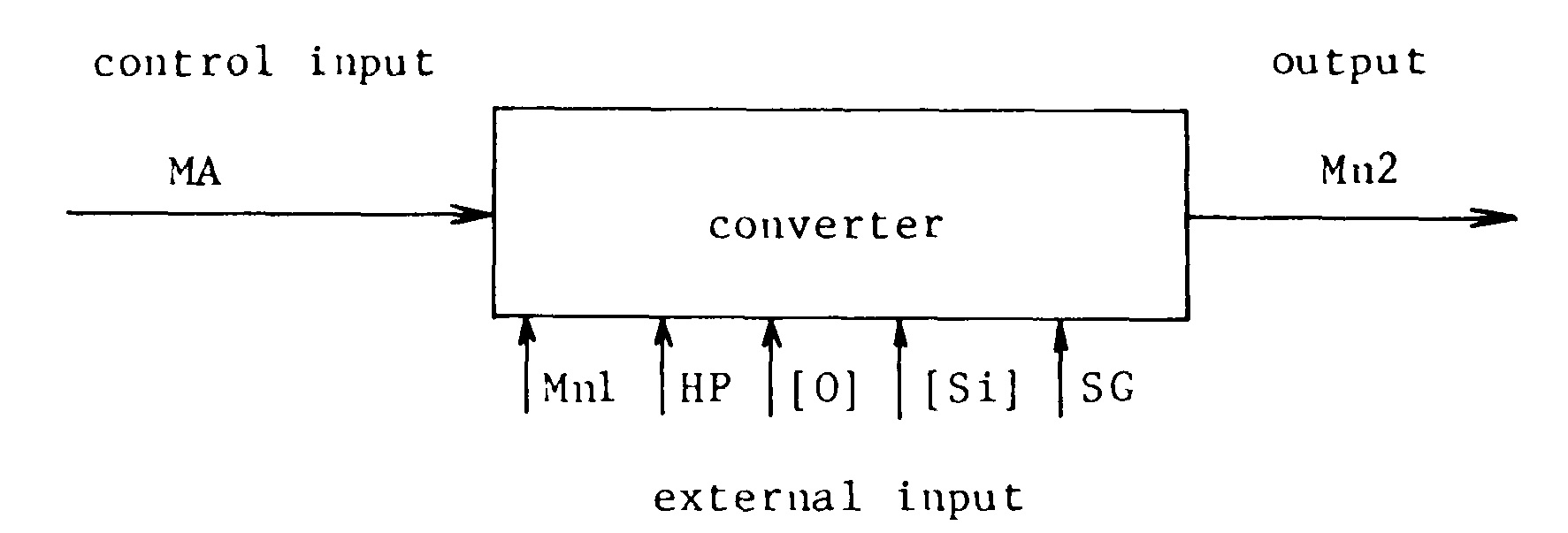


Fig 5: Variation of model from ideal values

## B. Steel making Process

Depending on the components used, a steel-making facility produces several types of steel. A variety of adjustments must be made to the manganese ratio in the goods. The controller to determine the amount of manganese alloy to produce desired type of steel is shown in the fig7.



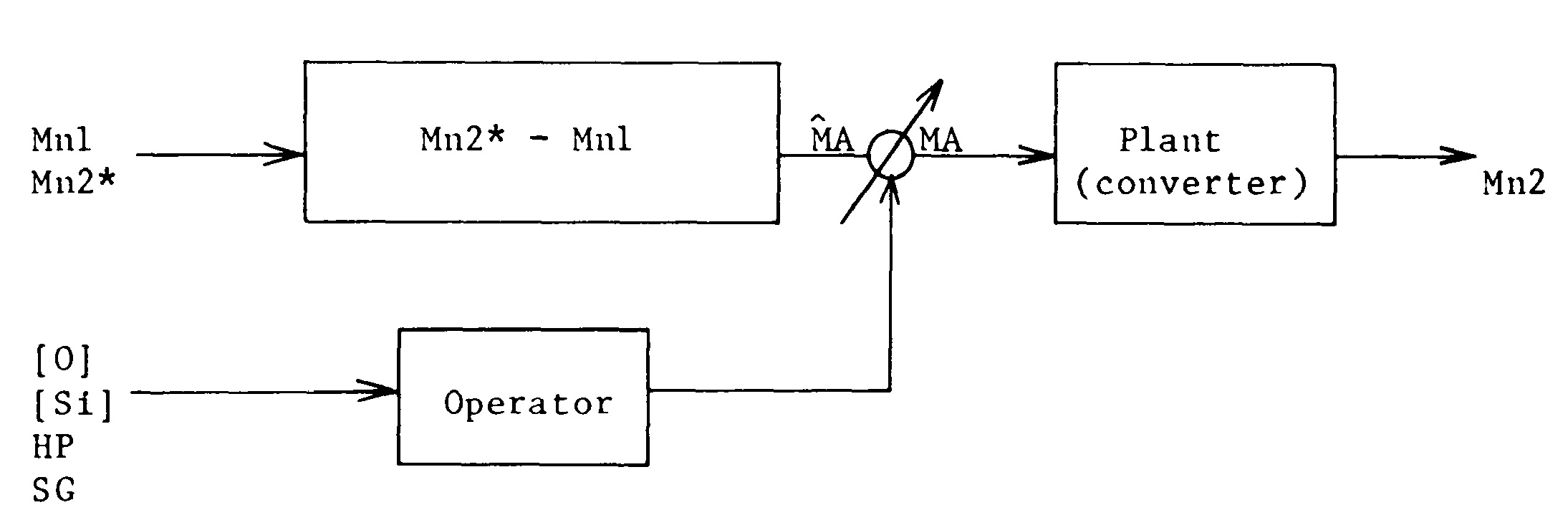


Fig 7: Steel Making Process

* MATLAB Code

close all; clear; clc;

% HP Mn SG MA

data = [93.9 1.11 0.3 14;

94 14.52 -0.08 135;

93.1 5.22 -0.05 54;

93.6 5.36 -0.43 53;

85.6 13.56 0.11 129;

85.8 11.55 0.12 106;

86.4 11.88 -0.06 113;

93.1 13.85 0.16 119;

88.5 12.27 -0.02 129;

93 11.69 -0.03 127;

93 9.86 -0.01 98;

95 4.96 -0.13 46;

94.7 2.19 0.01 25;

85 1.89 -0.1 23;

87.5 14.26 -0.09 121;

90.5 11.52 -0.1 112;

90.1 12.59 -0.06 116;

90.2 5.67 -0.28 60];

input = [data(:,1) data(:,3:4)];

output = data(:,2);

HP = input(:,1);

SG = input(:,2);

MA = input(:,3);

k = size(data,2) - 1; % Number of variables

m = size(data,1); % Number of data points

n = 4; % Maximum number of rules

HPmin = min(data(:,1)); HPmax = max(data(:,1));

Mnmin = min(data(:,2)); Mnmax = max(data(:,2));

SGmin = min(data(:,3)); SGmax = max(data(:,3));

MAmin = min(data(:,4)); MAmax = max(data(:,4));

mHP = 1/(HPmin-HPmax); cHP = HPmin/(HPmin-HPmax); cHP1 = HPmax/(HPmax-HPmin);

mSG = 1/(SGmin-SGmax); cSG = SGmin/(SGmin-SGmax); cSG1 = SGmax/(SGmax-SGmin);

mMA = 1/(MAmin-MAmax); cMA = MAmin/(MAmin-MAmax); cMA1 = MAmax/(MAmax-MAmin);

%% MODEL 2-1

beta = zeros(m,n);

b = beta;

for i = 1:n

for j = 1:m

smallHP(j) = -mHP\*HP(j)+cHP1;

smallSG(j) = -mSG\*SG(j)+cSG1;

smallMA(j) = -mMA\*MA(j)+cMA1;

bigHP(j) = mHP\*HP(j)+cHP;

bigSG(j) = mSG\*SG(j)+cSG;

bigMA(j) = mMA\*MA(j)+cMA;

switch i

case 1

b(j,i) = min([smallHP(j) smallSG(j)]);

case 2

b(j,i) = min([smallHP(j) smallSG(j)]);

case 3

b(j,i) = min([smallHP(j) bigSG(j)]);

case 4

b(j,i) = min([smallHP(j) bigSG(j)]);

end

end

beta(:,i) = b(:,i)./sum(b(:,i));

end

X = beta;

for i = 1:n

X = [X repmat(beta(:,i),1,k).\*input];

end

P = pinv(X'\*X)\*X'\*output;

p = P(1:n)';

for i = 1:k

p = [p; P(n\*i+1:n\*(i+1))'];

end

x = [ones(m,1) input];

y = x\*p;

y = b.\*y;

Y = [];

for i = 1:m

Y = [Y; sum(y(i,:))./sum(b(i,:))];

end

Y = X\*P;

PI1 = sqrt(sum((Y - output).^2)/m);

%% MODEL 2-2

beta = zeros(m,n);

b = beta;

for i = 1:n

for j = 1:m

smallHP(j) = -mHP\*HP(j)+cHP1;

smallSG(j) = -mSG\*SG(j)+cSG1;

smallMA(j) = -mMA\*MA(j)+cMA1;

bigHP(j) = mHP\*HP(j)+cHP;

bigSG(j) = mSG\*SG(j)+cSG;

bigMA(j) = mMA\*MA(j)+cMA;

switch i

case 1

b(j,i) = min([smallSG(j) smallMA(j)]);

case 2

b(j,i) = min([smallSG(j) bigMA(j)]);

case 3

b(j,i) = min([bigSG(j) smallMA(j)]);

case 4

b(j,i) = min([bigSG(j) bigMA(j)]);

end

end

beta(:,i) = b(:,i)./sum(b(:,i));

end

X = beta;

for i = 1:n

X = [X repmat(beta(:,i),1,k).\*input];

end

P = pinv(X'\*X)\*X'\*output;

p = P(1:n)';

for i = 1:k

p = [p; P(n\*i+1:n\*(i+1))'];

end

x = [ones(m,1) input];

y = x\*p;

y = b.\*y;

Y = [];

for i = 1:m

Y = [Y; sum(y(i,:))./sum(b(i,:))];

end

Y = X\*P;

PI2 = sqrt(sum((Y - output).^2)/m);

%% OUTPUT

if PI1 < PI2

fprintf('Stable: Model 2-2\nVariables: HP & SG\n');

else

fprintf('Stable: Model 2-2\nVariables: SG & MA\n');

end

%Plot

figure(1);

scatter(output,Y);

grid on;

hold on;

plot([0 20],[0 20]);

xlabel('Operator');

ylabel('Model');

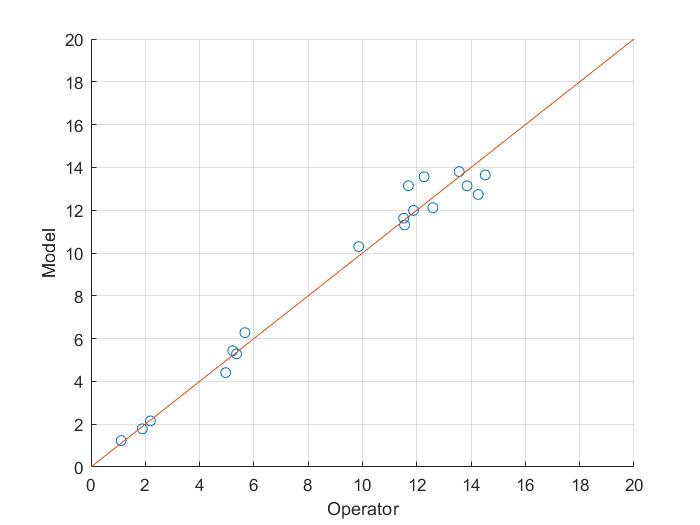


Fig 8: Variation of model from ideal values

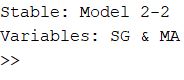


Fig 9: Results of steel making process

# Conclusion

We have fuzzy models of the two real-world systems that has been successfully constructed, and the mathematical approach has been used to evaluate the fuzzy models.

We have shown variation of model from ideal scenario in both of the cases an also least PI is used while calculating the fuzzy variables.

Reference

1. ‘Fuzzy identification of Systems and it’s applications to modeling and control’ by Tomohiro Takagi and Michio Sugeno.

1. [↑](#footnote-ref-1)