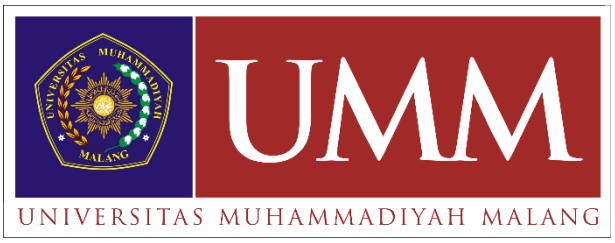


MODUL PRAKTIKUM
KOMPUTASI CERDAS

NAMA MAHASISWA
NIM MAHASISWA

LABORATORIUM TEKNIK ELEKTRO
JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNIK
UNIVERSITAS MUHAMMADIYAH MALANG
2018

DAFTAR ISI

DAFTAR GAMBAR

Halaman ini sengaja dikosongkan

DAFTAR TABEL

Halaman ini sengaja dikosongkan

BAB 1
OPTIMASI MENGGUNAKAN PSO

1.1 Latar Belakang

Pengertian optimasi secara umum yaitu proses pencarian satu atau beberapa variable agar tercapai suatu tujuan optimal (maks/min). Permasalahan optimasi merupakan permaslahan yang sering kita hadapi khususnya dalam bidang teknik. Contoh contoh permasalahan optimasi yang umum sering ditemui seperti pencarian jalur terdekat, dan penjadwalan. Sedangkan contoh-contoh permasalahan optimasi pada bidang Teknik seperti pembangkitan daya generator, penjadwalan daya energi terbaharukan (hybrid), dan pencarian parameter pada sebuah system.

Terdapat beberapa metode penyelesaian optimasi seperti penyelesaian secara matematis (gradien), metode klasik (numerik) seperti Newton Raphson, dan penyelesaian dengan metode heuristic atau metode cerdas. Metode heuristik saaat ini telah banyak digunakan untuk menyelesaiakan permaslahan optimisasi dikarenakan penyelesaian secara matematis tidak mungkin dilakukan dikarenakan keterbatasan dalam proses turunan, kemudian keterbatasan pada metode numerik yang hanya dapat menyelesaikan permasalahan linear.

1.2 Tujuan

Mahasiswa dapat mengimplementasikan PSO sebagai metode penyelesaian permasalahan optimisasi.

1.3 Dasar Teori

PSO atau particle swarm optimization merupakan salah satu algoritma cerdas atau metode heuristik yang terinspirasi oleh burung atau ikan yang sedang bermigrasi. Proses pencarian variable dalam PSO diibaratkan dengan sekumpulan burung atau ikan yang dalam pso disebut sebuah particle yang terbang atau berenang menyusuri area atau ruang yang selanjutnya diibaratkan dengan ruang permasalahan (fungsi tujuan). Saat sekumpulan burung terbang atau sekumpulan ikan sedang berenang memiliki beberapa sifat berikut:

a. Meskipun berpindah secara bersama tetapi tidak saling bertabrakan atau bersinggungan (sparasi).
b. Pencarian Bersama (Kohesi)
c. Penyesuaian (Aligment)

Pada PSO particle berpindah dengan menggunakan persamaan berikut:

$$V_{i,d}(t+1) = V_{i,d}(t) + c_1 r_1 \left(P_{i,d}(t) - X_{i,d}(t)\right) + c_2 r_2 \left(G_d(t) - X_{i,d}(t)\right) \quad (1)$$

dengan perpindahan posisi partikel menggunakan persamaan berikut:

$$X_{i,d}(t+1) = X_{i,d}(t) + V_{i,d}(t+1) \quad (2)$$

1.3.1 Algoritma PSO

Algoritma PSO dijabarkan dalam listing program berikut:

Start
Inisialisasi posisi particle awal $X_{i,d}$ *sejumlah N*
Inisialisasi kecepatan particle awal $V_{i,d}$
Inisialisasi parameter c_1 *dan* c_2
Evaluasi $X_{i,d}(0)$ *terhadap fungsi tujuan* $f\left(X_{i,d}(0)\right)$
$P_{i,d}(0) = X_{i,d}(0)$
Cari Particle terbaik $G_d(0)$ *dari kumpulan particle*
While (Iterasi < Max Iterasi)
  For i=1:N
    For d=1:D
      Hitung kecepatan dan posisi partikel dengan pers (1) *dan pers* (2)
      $V_{i,d}(t+1) = V_{i,d}(t) + c_1 r_1 \left(P_{i,d}(t) - X_{i,d}(t)\right) + c_2 r_2 \left(G_d(t) - X_{i,d}(t)\right)$
      $X_{i,d}(t+1) = X_{i,d}(t) + V_{i,d}(t+1)$
    End
  End
Evaluasi $X_{i,d}(t+1)$ *terhadap fungsi tujuan* $f\left(X_{i,d}(t+1)\right)$
Cari indeks pengalaman terbaik partikel $P_{i,d}(t+1)$
Cari indeks partikel terbaik $G_d(t+1)$
End

1.3.2 Contoh Implementasi PSO

Misal kita mempunyai sebuah permasalahan optimasi dengan fungsi tujuan berikut:

$$minimasi\ f(x)$$
$$f(x) = (100 - x)^2$$
dimana $60 \leq x \leq 120$

Iterasi 0:

1. Tentukan jumlah populasi particle N=4
2. Tentukan populasi awal secara random sesuai dengan batas atas dan batas bawah $60 \leq x \leq 120$

$$X_{i,d}(0) = \begin{bmatrix} 80 \\ 90 \\ 110 \\ 75 \end{bmatrix}$$

3. Tentukan Kecepatan awal

$$V_{i,d}(0) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

4. Tentukan $c_1 = c_2 = 1$
5. Evaluasi $X_{i,d}(0)$ terhadap $f(x)$

$$f(80) = (100 - 80)^2 = 400$$

$$f(90) = (100 - 90)^2 = 100$$

$$f(110) = (100 - 110)^2 = 100$$

$$f(75) = (100 - 75)^2 = 625$$

6. Tentukan $P_{i,d}(0) = X_{i,d}(0)$
7. Dari evaluasi no 4 cari $G_d(0)$
   Dikarenakan pada contoh ini permasalahan merupakan minimasi maka particle terbaik adalah yang menghasilkan nilai fungsi tujuan terkecil

$$G_d(0) = 90$$

Iterasi 1:

1. Hitung kecepatan dan posisi partikel. Missal nilai random yang didapat adalah $r_1 = 0.4; r_2 = 0.5$

$$v_1(1) = 0 + 1(0.4)(80 - 80) + 1(0.5)(90 - 80) = 5$$
$$v_2(1) = 0 + 1(0.4)(90 - 90) + 1(0.5)(90 - 90) = 0$$
$$v_3(1) = 0 + 1(0.4)(110 - 110) + 1(0.5)(90 - 110) = -10$$
$$v_4(1) = 0 + 1(0.4)(75 - 75) + 1(0.5)(90 - 75) = 7.5$$

   sehingga posisi partikel adalah:

$$x_1(1) = 80 + 5 = 85$$
$$x_2(1) = 90 + 0 = 90$$
$$x_3(1) = 110 - 10 = 100$$
$$x_4(1) = 75 + 7.5 = 82.5$$

2. Evaluasi $X_{i,d}(1)$ terhadap $f(x)$

$$f(85) = (100 - 85)^2 = 225$$
$$f(90) = (100 - 90)^2 = 100$$
$$f(100) = (100 - 0)^2 = 0$$
$$f(82.5) = (100 - 75)^2 = 306.25$$

3. Tentukan $P_{i,d}(1)$
   Pada persoalan minimasi untuk mencari $P_{i,d}(t)$ maka dilakukan perbandingan antara $f\left(X_{i,d}(t)\right)$ dengan $f\left(X_{i,d}(t+1)\right)$ apakah $f\left(X_{i,d}(t+1)\right)$ lebih kecil dibanding $f\left(X_{i,d}(t)\right)$ dengan ketentuan sebagai berikut:

$$P_{i,d}(t) = \begin{cases} X_{i,d}(t+1) & if\ f\left(X_{i,d}(t+1)\right) < f\left(P_{i,d}(t)\right) \\ P_{i,d}(t) & elseiff\left(X_{i,d}(t+1)\right) \geq f\left(P_{i,d}(t)\right) \end{cases}$$

$$P_{i,d} = \begin{bmatrix} 85 \\ 90 \\ 100 \\ 82.5 \end{bmatrix}$$

4. Dari evaluasi no 3 cari $G_d(1)$

$$G_d(1) = 100$$

Iterasi 2:

1. Hitung kecepatan dan posisi partikel. Missal nilai random yang didapat adalah $r_1 = 0.3; r_2 = 0.6$

$$v_1(2) = 5 + 1(0.3)(85 - 85) + 1(0.6)(100 - 80) = 14$$
$$v_2(2) = 0 + 1(0.3)(90 - 90) + 1(0.6)(100 - 90) = 6$$
$$v_3(2) = -10 + 1(0.3)(100 - 100) + 1(0.6)(100 - 100) = -10$$
$$v_4(2) = 7.5 + 1(0.3)(82.5 - 82.5) + 1(0.6)(100 - 82.5) = 18$$

   sehingga posisi partikel adalah:

$$X_{i,d}(2) = \begin{bmatrix} 99 \\ 96 \\ 90 \\ 100.5 \end{bmatrix}$$

2. Evaluasi $X_{i,d}(2)$ terhadap $f(x)$

$$f\left(X_{i,d}(2)\right) = \begin{bmatrix} 1 \\ 16 \\ 100 \\ 0.25 \end{bmatrix}$$

3. Tentukan $P_{i,d}(2)$

$$P_{i,d}(2) = \begin{bmatrix} 99 \\ 96 \\ 100 \\ 100.5 \end{bmatrix}$$

4. Dari evaluasi no 3 cari $G_d(2)$

$$G_d(2) = 100$$

1.3.3 Implementasi PSO dengan Matlab:

Pada subab ini PSO dimplementasikan untuk mengoptimalkan sebuah fungsi tujuan (*Fitness*) agar fungsi tersebut bernilai minimum.

$$\min f(x) \text{ ; dimana}$$

$$f(x) = 3x_1^2 + x_2^2 + x_3^2 - 3x_1x_2 - 2x_1x_3 - 3x_2x_3 - 5x_1 - 4x_2 - 6x_3$$

Dengan

$$0 \leq x_1, x_2, x_3 \leq 15$$

Langkah Langkah Percobaan:

1. Membuat source code fungsi tujuan pada matlab sebagai berikut:

```
function f=fitness(x)
a=3*(x(1,1))^2+4*(x(1,2))^2+2*(x(1,3))^2;
b=-3*x(1,1)*x(1,2)-2*x(1,1)*x(1,3)-3*x(1,2)*x(1,3);
c=-5*x(1,1)-4*x(1,2)-6*x(1,3);
f=a+b+c;
end
```

2. Simpan dengan nama fitness.m
3. Membuat source code program utama PSO pada matlab sebagai berikut:

```
%%Inisialisasi PSO%%
jumlah_particle=20;%%20-30 particle
D=3;
batas_atas=15;
batas_bawah=0;
x=(batas_atas-
batas_bawah).*rand(jumlah_particle,D)...
    +repmat(batas_bawah,jumlah_particle,D)
v=zeros(jumlah_particle,D)
P=zeros(jumlah_particle,D)

f=zeros(jumlah_particle,1)
Max_Iter=10000
c1=1;%%0-2
c2=1;%%0-2
minftot=[];%%index Nilai fungsi paling minimum
%%Iterasi Ke-0%%
for i=1:jumlah_particle
    f(i,:)=fitness(x(i,:));
end
P=x;
fbest=f;
[minf,idk]=min(f);
G=x(idk,:);

for t=1:Max_Iter
```

```
 for d=1:D
    for i=1:jumlah_particle
      v(i,d)=v(i,d)+(c1*rand)*(P(i,d)-x(i,d))...
           +(c2*rand)*(G(:,d)-x(i,d));
      x(i,d)=v(i,d)+x(i,d);
    end
  end
 for i=1:jumlah_particle
    f(i,:)=fitness(x(i,:));
 end
 changerow=f<fbest;
 fbest=fbest.*(1-changerow)+f.*changerow;
 P(changerow,:)=x(changerow,:);
 [minf,idk]=min(fbest);
 minftot=[minftot;minf];
 G=P(idk,:);
end
x_optimal=G
minimum_f=minf
plot(minftot)
```

4. Simpan dengan nama PSO.m
5. Run program PSO.m
6. Catat hasil pada table berikut:

Tabel 1.1 Hasil Percobaan 1.1

x_1	x_2	x_3	$f(x)$

7. Pada source code program utama PSO ubah persamaan kecepatan menjadi:

```
v(i,d)=0.75*v(i,d)+(c1*rand)*(P(i,d)-x(i,d))...
              +(c2*rand)*(G(:,d)-x(i,d));
```

8. Run kembali program PSO.m
9. Catat hasil pada table berikut:

Tabel 1.2 Hasil Percobaan 1.2

x_1	x_2	x_3	$f(x)$

10. Analisa hasil percobaan pada Tabel 1.1 dan Tabel 1.2 dengan menggunakan metode matematis

1.4 Data Hasil Percobaan

Tabel 1.1 Hasil Percobaan 1.1

x_1	x_2	x_3	$f(x)$

Tabel 1.3 Hasil Percobaan 1.2

x_1	x_2	x_3	$f(x)$

1.5 Analisa Data

1.6 Kesimpulan

BAB 2
Artificial Neural Network

2.1 Tujuan

Mahasiswa dapat mengimplementasikan ANN sebagai metode penyelesaian permasalahan identifikasi, estimasi, dan prediksi dari ebuah system.

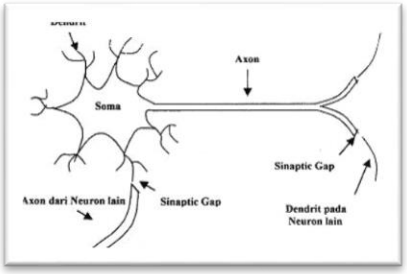
2.2 Dasar Teori

Artificial Neural Network (ANN) atau sering disebut juga dengan jaringan syaraf tiruan adalah system pemroses informasi yang memiliki karakteristik menyerupai dengan karakteristik jaringan syaraf biologi.

ANN dibentuk dengan generalisasi model dari jaringan syaraf biologi dengan asumsi sebagai berikut:

- Pemrosesan informasi terjadi pada banyak elemen sederhana (neuron)
- Sinyal dikirimkan diantara neuron neuron melalui penghubung penghubung
- Penghubung antar neuron memiliki bobot yang akan memperkuat atau memperlemah sinyal

Pada neuron jaringan syaraf biologis terdapat tiga komponen penyusun utama yaitu *Dendrit*, *Soma*, dan *Axon*. *Dendrit* berperan sebagai bagian penerima informasi yang bisa saja berasal dari neuron yang lain. *Soma* berfungsi sebagai penampung atau pengolah sinyal, sedangkan *Axon* berfungsi sebagai penerus signal. Struktur neuron pada jaringan syaraf biologis ditunjukan pada Gambar 2.1.



Gambar 2.1 neuron pada jaringan syaraf biologis

Struktur neuron pada jaringan biologis tersebut kemudian digeneralkan dalam sebuah model matematika dengan asumsi asumsi yang sudah dijelaskan dijabarkan pada persamaan (3) dan (4):

$$z = \sum_{i=1}^{N} w_i \cdot x_i + b \tag{3}$$

$$y_n = f(z) \tag{4}$$

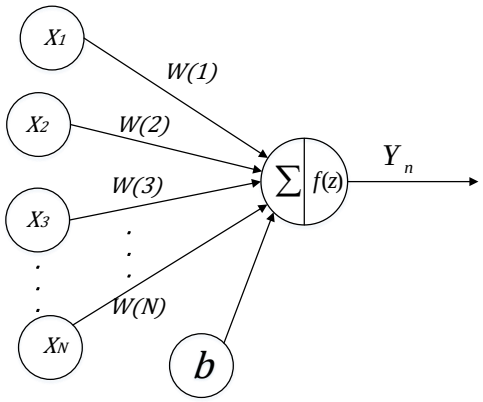
Dimana:

x_i : input neuron w_i : bobot neuron

b : bias y_n : output neuron

$f(z)$: fungsi aktifasi N : jumlah input

Model matematika pada persamaan (3) dan (4) jika digambarkan memiliki struktur yang menyerupai jaringan syaraf biologis, seperti yang ditunjukan pada Gambar 2.2



Gambar 2.2 Model neuron pada ANN

2.2.1 Jenis Jenis hidden layer

a. Fungsi Aktifasi Threshold
Persamaan fungsi aktifasi:

$$f(z) = \begin{cases} 1 \; if \; z \geq a \\ 0 \quad else \end{cases}$$

b. Fungsi Aktifasi Linear
Persamaan fungsi aktifasi:

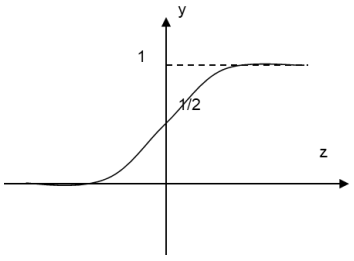
$$f(z) = \alpha z$$

Turunan fungsi aktifasi:

$$f'(z) = \alpha$$

c. Fungsi Aktifasi Sigmoid
Persamaan fungsi aktifasi:

$$f(z) = \frac{1}{(1 + e^{-z})}$$



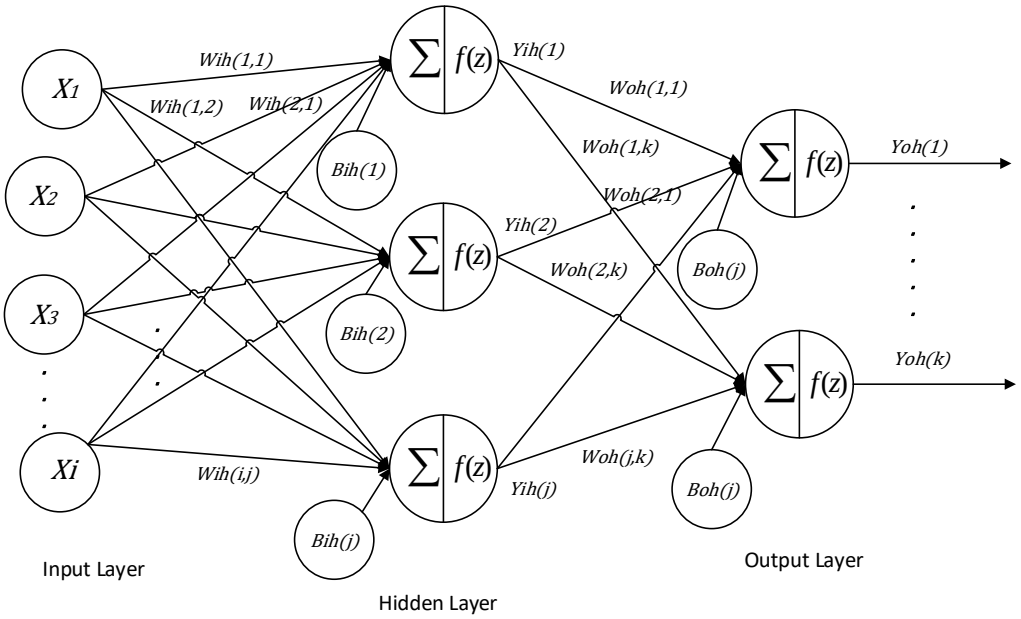
Gambar 2.3 grafik nilai keluaran fungsi aktifasi sigmoid

Turunan fungsi aktifasi:

$$f'(z) = \big(1 - f(z)\big)\big(f(z)\big)$$

2.2.2 Multilayer Neural Network (Backpropagation)

Multilayer neural network merupakan ANN yang terdapat banyak neuron pada strukturnya. Neuron tersebut disusun secara umum menjadi 3 bagian layer yaitu layer *input*, layer *hidden*, dan layer *output* yang digambarkan pada Gambar 2.4.



Gambar 2.4 Struktur Multilayer Neural Network

Multilayer neural network memiliki model matematis yang dijabarkan pada persamaan (5) hingga persamaan (8). Dimana persamaan *forward* dari input ke *hidden layer* dijabarkan pada persamaan (5) dan (6)

$$zih(j) = \sum_{i=1}^{jmlh\_input} wih(i,j) \cdot \big(x(l,i)\big) + bih(j) \tag{5}$$

$$Yih(j) = f\big(zih(j)\big) \tag{6}$$

Kemudian persamaan forward dari *hidden layer* ke *output layer* dijabarkan pada persamaan (7) dan (8)

$$zoh(k) = \sum_{i=1}^{jmlh\_hidden} woh(j,k) \cdot \big(Yih(l,k)\big) + boh(k) \tag{7}$$

$$Yoh(k) = f\big(zoh(k)\big) \tag{8}$$

Dengan

$wih(i,j)$	:	Bobot input ke *hidden layer* pada input i dan *hidden layer* ke j
$x(l,k)$	:	Nilai input ke i dan pada data ke l
$bih(j)$	:	Bobot bias input ke *hidden layer* ke j
$Yih(j)$	:	Output pada *hidden layer* ke j
$woh(j,k)$	:	Bobot *hidden* ke *output layer* pada *hidden* ke j dan *output* ke k
$boh(k)$	:	Bobot bias input ke *output layer* ke k
$Yoh(k)$	:	Output ke k

$f(zih(j))$: Fungsi aktifasi pada *hidden layer* ke j
$f(zoh(k))$: Fungsi aktifasi pada *output layer* ke k

2.2.3 Pelatihan bobot pada multilayer neural network

Pelatihan bobot pada multilayer neural network menggunakan metode *least square* dimana bobot pada *hidden* ke *output layer* dilatih terlebih dahulu berdasarkan error yang terjadi pada perbandingan nilai antara output layer dengan output pada data target atau data *training.*

Setelah proses pelatihan bobot pada *hidden* ke *output layer* dilakukan maka kemudian proses pelatihan bobot pada input ke *hidden layer* dilakukan dengan asumsi bahwa error yang terjadi pada output layer merambat *(propagate)* secara merata pada hidden layer.

Persamaan perubahan bobot pada *hidden* ke *output layer* dijabarkan pada persamaan (9) dan (10):

$$dwoh(j,k) = \lambda(error(k)) \cdot \left(f'(zoh)\right) \cdot Yih(j) \quad (9)$$

$$woh(j,k)_{baru} = woh(j,k)_{lama} + dwoh(j,k) \quad (10)$$

dengan persamaan perubahan bias pada *output layer* dijabarkan pada persamaan (11) dan (12):

$$dboh(k) = \lambda(error(k)) \cdot \left(f'(zoh(k))\right) \quad (11)$$

$$boh(k)_{baru} = boh(k)_{lama} + dboh(k) \quad (12)$$

Sedangkan persamaan perubahan bobot pada *input* ke *hidden* dijabarkan pada persamaan (13) dan (14):

$$dwih(i,j) = \lambda\left(errh(j)\right) \cdot \left(f'(zih(j))\right) \cdot x(l,i) \quad (13)$$

$$woh(i,j)_{baru} = wih(i,j)_{lama} + dwih(i,j) \quad (14)$$

dengan persamaan perubahan bias pada *hidden layer* dijabarkan pada persamaan(15) dan (16):

$$dbih(j) = \lambda\left(errh(j)\right) \cdot \left(f'(zih(j))\right) \quad (15)$$

$$bih(j)_{baru} = bih(j)_{lama} + dbih(j) \quad (16)$$

dimana

$dwih(i,j)$: Perubahan Bobot pada input ke i dan *hidden layer* ke j
$dwoh(j,k)$: Perubahan Bobot *hidden* ke j dan *output* ke k
$dbih(j)$: Perubahan Bobot bias pada *hidden layer* ke j
$error(k)$: $\left(Yt(l,k) - Yoh(k)\right)$
$boh(k)$: Bobot bias input pada *output layer* ke k
λ : Learning rate
$f'(zih(j))$: Turunan Fungsi aktifasi pada *hidden layer* ke j
$f'(zoh(k))$: Turunan Fungsi aktifasi pada *output layer* ke k

2.2.4 Backpropagation Pada Permasalahan Prakiraan Cuaca dengan Matlab

Pada Implementasi ini kita akan memodelkan system parakiraan cuaca secara sederhana. Dari data suhu, arah angin, kelembaban, dan tekanan udara diklasifikasikan pada Tabel 2.1

Tabel 2.1 Tabel kriteria cuaca didaerah X []

Unsur Cuaca	**Keadaan Cuaca**			
	Cerah	**Berawan**	**Hujan**	**Hujan Lebat**
Arah Angin (degree)	<150	150-200	>200	>200
Suhu (Derajat Celcius)	>29	26-29	26-29	<26
Kelembaban(%)	<70	70-85	>80	>80
Tekanan Udara	>1010	1007-1010	1007-1010	<1007

Pada Tabel 2.1 input merupakan unsur cuaca dan output yang dihasilkan adalah keadaan cuaca. Data input meliputi arah angin, suhu, kelembaban, dan tekanan udara. Sehingga input pada neural network berjumlah 4 buah dan output 1 buah yaitu keadaan cuaca. Agar proses learning lebih mudah untuk membuat system prakiraan cuaca dengan Neural Network berdasarkan karakteristik tersebut terlebih dahulu data dimodifikasi dengan ketentuan sebagai berikut:

Tabel 2.2 Ketentuan data input unsur cuaca dengan input neuron

Unsur Cuaca	Rentang Variable		
Arah Angin	<150	150-200	>200
X1	-1	0	1
Suhu	<26	26-29	>29
X2	-1	0	1
Kelembaban	<70%	70%-85%	>85%
X3	-1	0	1
Tekanan	<1007	1007-1010	>1010
X4	-1	0	1

Untuk data output yaitu keadaan cuaca terdapat 4 kondisi dimana pada NN fungsi aktifasi memiliki rentang output 0-1 sehingga output neuron memiliki retang nilai ¼ sehingga data output sebagai berikut:

Tabel 2.3 Data output hasil modifikasi

Keadaan Cuaca	Cerah	Berawan	Hujan	Hujan Lebat
yt	0.25	0.5	0.75	1

Sehingga data training keseluruhan disajikan pada

Tabel 2.4 Data Training

Input Neuron				**Output**
Arah Angin (X1)	suhu (X2)	Kelembaban (X3)	Tekanan Udara (X4)	
-1	1	-1	1	0.25
0	0	0	0	0.5

1	0	1	0	0.75
1	-1	1	-1	1

2.2.5 Langkah Langkah Percobaan

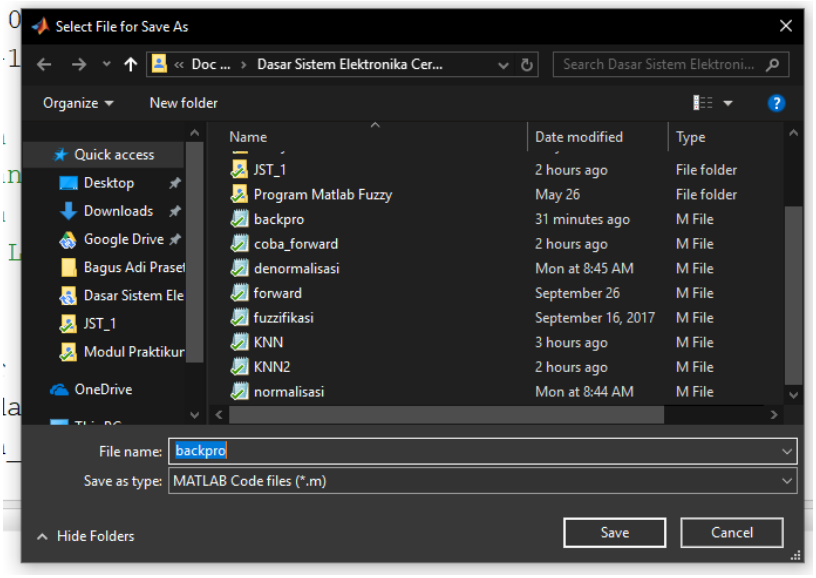
1. Buat program pada matlab dengan listing sebagai berikut

Listing program ANN untuk permasalahan Prakiraan Cuaca

```
clear
clc
%==================Inisialisali Parameter NN================
jmlh_input=4;
jmlh_hidden=15;
jmlh_output=1;
wih=zeros(jmlh_input,jmlh_hidden);
woh=zeros(jmlh_hidden,jmlh_output);
bih=zeros(1,jmlh_hidden);
boh=zeros(1,jmlh_output);
max_iter=10000;
lamdha=0.4;
jmlh_data=4;
%==================Inisialisali Parameter NN================
x=[-1   1  -1   1;
    0   0   0   0;
    1   0   1   0;
    1  -1   1  -1;];
yt=[0.25; %cerah 0-0.25
    0.5; %berawan 0.25-0.5
    0.75; %Hujan 0.5 - 0.75
    1;]; %Hujan Lebat0.75-1
max_error=[];
for t=1:max_iter
  for l=1:jmlh_data
    for j=1:jmlh_hidden
       zih(j)=0;
       for i=1:jmlh_input
       zih(j)=zih(j)+wih(i,j)*x(l,i);
       end
       yih(j)=1/(1+exp(-(zih(j)+bih(j))));
    end
    for k=1:jmlh_output
       zoh(k)=0;
       for j=1:jmlh_hidden
       zoh(k)=zoh(k)+woh(j,k)*yih(1,j);
       end
       yoh(k)=1/(1+exp(-(zoh(k)+boh(k))));
    end
   %Perhitungan Backward
   for k=1:jmlh_output
      error(k)=yt(l,k)-yoh(k)
      dboh(1,k)=lamdha*(error(k))*(1-yoh(k))*yoh(k);
      boh(1,k)=boh(1,k)+dboh(1,k); %Menghitung Bias baru
      for j=1:jmlh_hidden
         dwoh(j,k)=lamdha*(error(k))*(1-yoh(k))*yoh(k)*yih(j);%% Perubahan
bobot
         woh(j,k)=woh(j,k)+dwoh(j,k);%% Menghitung Bobot baru
         errh(j,k)=error(k)/jmlh_hidden;
      end

      for j=1:jmlh_hidden
         bih(j)=bih(j)+lamdha*errh(j,k)*(1-yih(j))*yih(k);
         for i=1:jmlh_input
            dwih(i,j)=lamdha*errh(j,k)*(1-yih(j))*yih(j)*x(l,i);
            wih(i,j)=wih(i,j)+dwih(i,j);
```

```
        end
      end
    end
  end
max_error=[max_error,max(error)];%% Mengindexkan nilai error tiap iterasi
end
plot(max_error)
```

2. Simpan Listing program dengan nama *backpropagation*

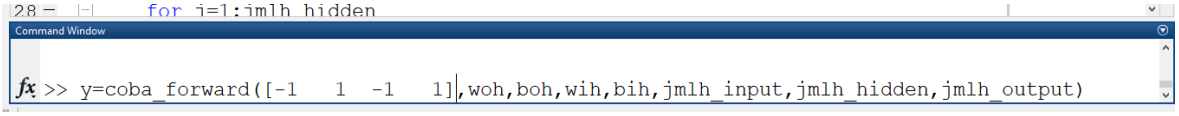


3. Buka tab baru untuk membuat listing baru
4. Buat program pada matlab dengan listing sebagai berikut

Listing program ANN untuk test data

```
function y=coba_forward(x,woh,boh,wih,bih,jmlh_input,jmlh_hidden,jmlh_output)
 for j=1:jmlh_hidden
    zih(j)=0;
    for i=1:jmlh_input
      zih(j)=zih(j)+wih(i,j)*x(1,i);
    end
    yih(j)=1/(1+exp(-(zih(j)+bih(j))));
 end
 for k=1:jmlh_output
    zoh(k)=0;
    for j=1:jmlh_hidden
      zoh(k)=zoh(k)+woh(j,k)*yih(1,j);
    end
    yoh(k)=1/(1+exp(-(zoh(k)+boh(k))));
    y(k)=yoh(k);
 end
end
```

5. *Run* listing program backpropagation dan simpan data grafik error pada hasil percobaan
6. Uji data output pada neural network dengan data training dengan mengetikan listing berikut pada *Command Window*



7. *Tekan Enter dan Catat hasil keluaran neural network pada* Tabel 2.5

Tabel 2.5 Data Hasil Percobaan 1

Input Neuron				y
Arah Angin (X1)	suhu (X2)	Kelembaban (X3)	Tekanan Udara (X4)	
-1	1	-1	1	…
0	0	0	0	….
1	0	1	0	….
1	-1	1	-1	…

8. *Ulangi langkah 6-7 untuk data masukan lain sehingga* Tabel 2.5 *terpenuhi*
9. *Jika* ***y*** *bernilai* ***0-0.25*** *adalah nilai* ***cerah(C) , berawan(B) y*** *bernilai 0.25-0.5,* ***Hujan(H) y*** *bernilai 0.5 - 0.75,* ***Hujan Lebat(HL)*** *ulangi langkah 6-7 untuk data input berikut*

Input Neuron				y	C/B/H/HL
Arah Angin (X1)	suhu (X2)	Kelembaban (X3)	Tekanan Udara (X4)		
0	1	-1	1	…	….
1	0	0	0	….	….
0	0	1	0	….	….
1	1	1	0	…	….

10. Ubah nilai *lamdha* menjadi 0.6 dan run kembali listing backpropagation, Simpan dan amati data grafik error.

2.3 Data Hasil Percobaan

Tabel 2.6 Data Hasil Percobaan 1

Input Neuron				y
Arah Angin (X1)	suhu (X2)	Kelembaban (X3)	Tekanan Udara (X4)	
-1	1	-1	1	…
0	0	0	0	….
1	0	1	0	….
1	-1	1	-1	…

Tabel 2.6 Data Hasil Percobaan 2

Input Neuron				y	C/B/H/HL
Arah Angin (X1)	suhu (X2)	Kelembaban (X3)	Tekanan Udara (X4)		
0	1	-1	1	…	….
1	0	0	0	….	….
0	0	1	0	….	….
1	1	1	0	…	….

2.4 Analisa Data

2.5 Kesimpulan

BAB 3

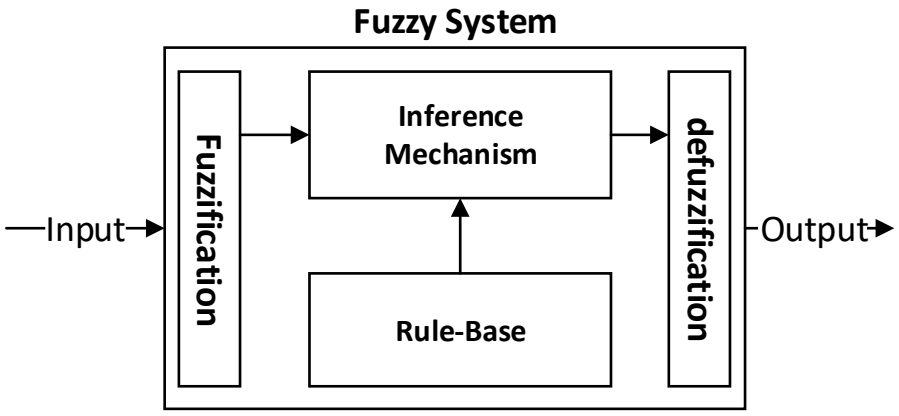
FUZZY LOGIC

3.1 Tujuan

Mahasiswa dapat mengimplementasikan PSO sebagai metode penyelesaian permasalahan optimisasi.

3.2 Dasar Teori

Fuzzy Logic merupakan peningkatan dari logika bolean, dimana pada logika bolean keanggotaan hanya dinyatakan dengan “iya” dan “tidak”. Fuzzy Logic salah satu pendekatan dimana representasi suatu kejadian didistribusikan kedalam sejumlah istilah bahasa (yang menyatakan level kualitatif). Fuzzy logic memiliki pendekatan kepada intuisi manusia yang menyatakan sesuatu dengan tingkatan seperti menyatakan kondisi suhu dengan tingkatan “sangat dingin”, “dingin”, “sedang”, “tidak dingin”, “panas”. Sistem fuzzy logic digambarakan dalam diagram pada Gambar 3.1, dimana pada digram tersebut terdapat 4 bagian utama yaitu 1). Fuzzification 2). Inference Mechanism 3). Rule-Base 4). Defuzzification.



Gambar 3.1 Struktur Sistem Fuzzy

3.2.1 Fuzzification

Fuzzifikasi merupakan proses penggolongan atau perubahan nilai pada variable input kedalam fuzzy set (himpunan fuzzy). Input pada fuzzy bisa terdiri dari banyak variable, dimana pada masing masing variable akan digolongkan pada masing masing himpunan fuzzy. Contoh 3.1:

Masukan fuzzy berupa variable suhu dan kecepatan motor, dengan himpunan variable input tersebut adalah sebagai berikut:

$$suhu = \{\text{"sangat dingin"}, \text{"dingin"}, \text{"sedang"}, \text{"panas"}, \text{"sangat panas"}\}$$

$$kecepatan\ motor = \{\text{"sangat cepat"}, \text{"cepat"}, \text{"sedang"}, \text{"lambat"}, \text{"sangat lambat"}\}$$

a. Fuzzy Set

Fuzzy set atau himpunan fuzzy adalah suatu himpunan yang beranggotakan sejumlah istilah dalam pengertian bahasa yang menyatakan level kualitatatif dari semesta pembicaraan X, seperti pada contoh 3.1 semesta suhu digolongkan menjadi 5 tingkatan himpunan. Keanggotaan dalam fuzzi set dengan logika manusia sangat kompleks, sehingga tidak dapat dinyatakan dalam bentuk tertentu dan berbeda untuk tiap individu.

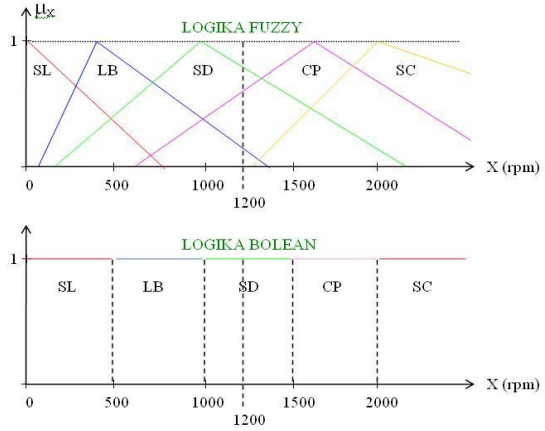
b. Fuzzy Membership function

Fungsi keanggotaan fuzzy adalah suatu fungsi yang didefinisikan untuk suatu anggota himpunan fuzzy yang menggambarkan derajat kebenaran suatu kejadian dalam semesta pembicaan X, dinyatakan dalam tingkat keanggotaan (grade of membership) dengan nilai antara 0 s/d 1. Untuk menggambarkan bagaimana fingsi keanggotaan fuzzy perhatikan contoh 3.2:

Perbandingan Nilai Kebenaran antara Logika Bolean dengan Logika Fuzzy dalam kasus kejadian tertentu.

X: Semesta Pembicaraan "Kecepatan Putaran Motor (rpm)"

Kejadian x: kecepatan 1200 rpm



Gambar 3.2 Perbandingan keanggotaan bolean dan fuzzy

Tabel 3.1 Derajat keanggotaan bolean dan fuzzy

Himpunan Pendukung	Tingkat Keanggotaan	
	Logika Bolean	Logika Fuzzy
Sangat Lambat (SL)	0	0.0
Lambat (LB)	0	0.2
Sedang (SD)	1	0.8
Cepat (CP)	0	0.6
Sangat Cepat (SC)	0	0.0

Pada Gambar 3.2 dan Tabel 3.1 menggambarkan perbandingan keanggotaan dari logika bolean dan fuzzy dimana pada kejadian kecepatan motor berputar 1200 nilai derajat keanggotaan sedang pada bolean bernilai 1 sedangkan pada fuzzy dinyatakan dalam tingkatan derajat keanggotaan yang berbeda. Pada kondisi tersebut nilai derajat keanggotan 0.8 pada himpunan sedang, 0.2 pada keanggotan lambat dan 0.6 pada keanggotan lambat.

c. Fuzzy Membership function Representation

Dalam bentuk representasi umum Himpunan dinyatakan dalam bentuk sebagai berikut:

S :{himpunan fuzzy semesta pembicaraan}

Si E S, i=1,2 ..n ; Si : himpunan pendukung ke i, dan n jumlah himpunan pendukung.

Nilai Logika Fuzzy X dapat dinyatakan dalam representasi umum himpunan:

X={μx1/S1; μx1/S1 ..... ; μxn/Sn}

Pada contoh 3.2 Kecepatan Putaran Motor dinyatakan dalam 5 keanggotaan

$kecepatan\ motor = \{$"sangat cepat", "cepat", "sedang", "lambat", "sangat lambat"$\}$

Dimana pada kondisi 1200 nilai keanggotaanya adalah sebagai berikut:

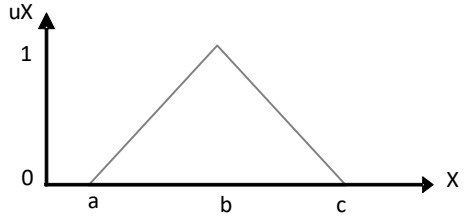
$$x(1200) = \{0.0, 0.2, 0.8, 0.6, 0.0\}$$

d. Bentuk-Bentu Fungsi Keanggotaan

Fungsi keanggotaan fuzzy memiliki bentuk yang beragam. Meskipun beragam fungsi keanggotaan tersebut tetap menyatakan derajat keanggotaan pada nilai 0 s/d 1. Bentuk bentuk fungsi keanggotaan tersebut diantaranya:

1. Distribusi-s
2. Distribusi-Z
3. Distribusi-Pi
4. Distribusi Trapesium
5. Distribusi Segitiga
6. Distribusi Gaussian

Fungsi keanggotaan dengan distribusi segitiga adalah yang paling umum digunakan karena kemudahanya. Fuzzifikasi dengan fungsi keanggotaan dengan distribusi segitiga dilakukan dengan persamaan :



Gambar 3.3 grafik fungsi keanggotaan segitiga

$$\mu x = \begin{cases} \dfrac{(x-a)}{b-a} & if(a < x \leq b) \\ \dfrac{(c-x)}{c-b} & elif\ (b < x \leq c) \\ 0 & else \end{cases} \quad (17)$$

Contoh 3.3:

Berikut perhitungan pada fungsi keanggotaan LB, SD, dan CP pada contoh 3.2, dimana kondisi x=1200

$$\mu x_1 = 0; karena\ diluar\ keanggotaan$$
$$\mu x_2 = \frac{1400-x}{1400-400} = \frac{1400-1200}{1000} = \frac{200}{1000} = 0.2$$
$$\mu x_3 = \frac{2000-x}{2000-1000} = \frac{2000-1200}{1000} = \frac{800}{1000} = 0.8$$
$$\mu x_4 = \frac{x-600}{1600-600} = \frac{1200-600}{1000} = \frac{600}{1000} = 0.6$$
$$\mu x_5 = 0; karena\ diluar\ keanggotaan$$

3.2.2 Rule Base

Basis Aturan fuzzy menyatakan hubungan kejadian yang ada pada input fuzzy dengan keputusan apa yang ada pada output fuzzy. Hubungan tersebut dapat dinyatakan dengan hubungan "jika" "maka" atau "if" "then". Contoh 3.4, sebuah system control fuzzy dengan input fuzzy berupa nilai error (e) dan nilai delta error (de) dan output berupa sinyal control (u)

jika istilah istilah dalam nilai error dan delta error

$$\{"nb","n","z","p","pb"\} = \{"negatif\ besar","negatif","zero","positif","positif\ besar"\}$$
$$error = \{"nb","n","z","p","pb"\}$$
$$delta\ error = \{"nb","n","z","p","pb"\}$$

Dengan himpunan output sinyal control adalah sebagai berikut:

$$sinyal\ control = \{"NB","N","Z","P","PB"\}$$

Maka basis aturan dari hubungan input dan output dapat dinyatakan dengan fungsi berikut:

$$\boldsymbol{if}\ e\ is\ nb\ \boldsymbol{and}\ de\ is\ nb\ \boldsymbol{than}\ \ u\ is\ NB \tag{18}$$

$$\boldsymbol{if}\ e\ is\ nb\ \boldsymbol{and}\ de\ is\ n\ \boldsymbol{than}\ \ u\ is\ NB \tag{19}$$

Dan seterusnya hingga semua kondisi terpenuhi.

Selain disajikan dengan logika "if" "then", basis aturan juga dapat disajikan dengan table sebagai berikut:

Tabel 3.2 contoh tabel basis aturan fuzzy

Delta error \ Error	nb	n	z	p	pb
nb	NB	NB	NB	N	Z
n	NB	NB	N	Z	P
z	NB	N	Z	P	PB
p	N	Z	P	PB	PB
pb	Z	P	PB	PB	PB

3.2.3 Inference Mechanism (Operasi Himpunan)

Infrensi fuzzy dilakukan untuk menghitung berapa nilai keanggotaan output berdasarkan nilai keanggotaan input dan basis aturan yang didefinisikan. Terdapat beberapa metode infrensi fuzzy diantaranya sebagai berikut:

a. Metode Generalize Modul Ponens (GMP) atau metode Mamdani:

$$\mu u(k) = \max[\mu u(k), \min\{\mu e(j), \mu de(i)\}] \tag{20}$$

b. Metode Larsent

$$\mu u(k) = 0.5[\mu u(k) + \{\mu e(j) \cdot \mu de(i)\}] \tag{21}$$

3.2.4 Defuzzification

Defuzzifikasi adalah bagian terakhir dari system fuzzy yang digunakan untuk menghitung besar nilai nyata berdasarkan hasil perhitungan infrensi dan membership output yang didefinisikan terdapat beberapa metode defuzzifikasi diantaranya Maximum of Mean (MOM), Center of Area (COA) atau center of gravity (COG). Metode COG diskrit sering digunakan untuk defuzzifikasi karena mudah dalam mengimplementasikan dalam bahas a pemrograman. Persamaan defuzzifikasi COG dinyatakan dalam persamaan

$$u = \frac{\sum_{k=1}^{m} b(k) \cdot \mu u(k)}{\sum_{k=1}^{m} \mu u(k)} \tag{22}$$

$$b = nilai\ tengah\ keanggotaan\ output$$

3.3 Fuzzy Logic Control

Pada sub bab ini akan menjelaskan bagaimana implementasi fuzzy untuk permasalaahan control. Sebagai contoh fuzzy digunakan untuk mengontrol kecepatan putar motor DC dimana model motor dc dijabarkan dalam persamaan berikut

$$J\ddot{\theta} + b\theta = Ki \tag{23}$$

$$L\frac{di}{dt} + Ri = V - K\dot{\theta} \tag{24}$$

Hasil transformasi laplace dari persamaan (23) dan (24)

$$s(Js + b)\theta = Ki \quad (25)$$

$$sLi + Ri = V - Ks\theta \quad (26)$$

Untuk menghilangkan i pada model substitusi i pada persamaan (25) ke persamaan (26)

$$\left(\frac{(sL + R)(Js + b)\dot{\theta}}{K}\right) = V - K\dot{\theta} \quad (27)$$

Kumpulkan variable sehingga menjadi

$$\left[\left(\frac{(sL + R)(Js + b)}{K}\right) + \frac{K^2}{K}\right]\dot{\theta} = V \quad (28)$$

Dari persamaan (28) didapatkan transfer function:

$$\frac{\dot{\theta}}{V} = \frac{K}{(sL + R)(Js + b) + K^2} = (LJ + (RJ + LB) + K^2 + RB \quad (29)$$

Dimana:

$$K = K_e = K_t$$

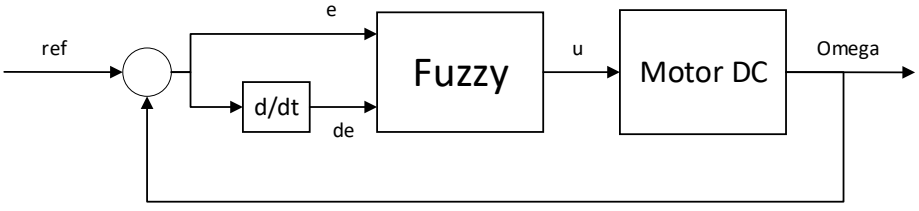
$$\omega = \dot{\theta}$$

Symbol	keterangan	Nilai parameter
(J)	moment of inertia of the rotor	0.0167 kg.m^2
(b)	motor viscous friction constant	0.0167 N.m.s
(Ke)	electromotive force constant	0.2 V/rad/sec
(Kt)	motor torque constant	0.2 N.m/Amp
(R)	electric resistance	0.6 Ohm
(L)	electric inductance	0.012 H
ω	Motor angular velocity	rad/s

Sehingga didapatkan transfer function pada persamaan (30) berikut:

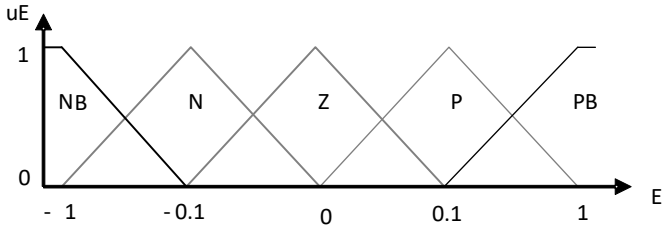
$$\frac{\omega}{V} = \frac{0.2}{0.0002004s^2 + 0.01022s + 0.05} \quad (30)$$

Pada model motor tersebut masukan berupa tegangan dan keluaran berupa kecepatan angular motor. Motor tersebut memiliki tegangan kerja maksismum 24 volt dan kecepatan maksimum 96 rad/s atau 916 rpm. Pada contoh ini menggunakan struktur fuzzy dengan pendekatan control PD yaitu error dan delta error sebagai masukan system fuzzy yang digambarkan pada blok diagram control fuzzy berikut:

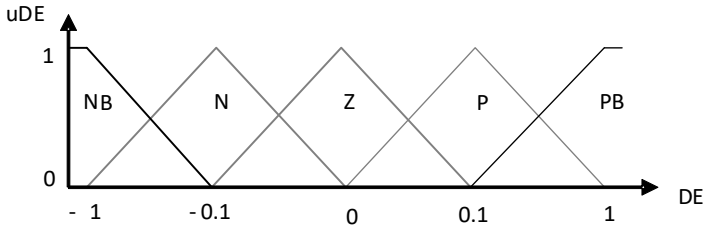


Gambar 3.4 block diagram Fuzzy logic control

Sehingga membership fuzzy input fuzzy terdiri dari dua membership yaitu error dan delta error:

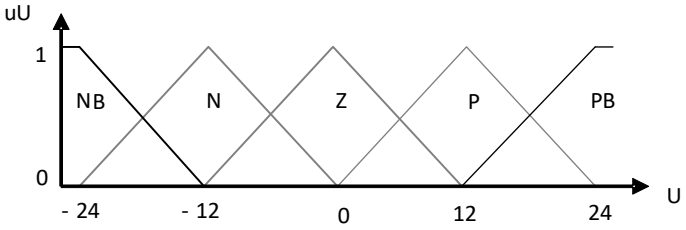


Gambar 3.5 membership function error



Gambar 3.6 membership function delta error

Dengan satu variable membership output yaitu sinyal control u.



Gambar 3.7 membership function sinyal control

Pada input maupun output jumlah membership yaitu 5 buah, sehingga basis aturan dari system fuzzy tersebut memiliki 25 kemungkinan seperti pada table berikut:

Tabel 3.3 Tabel basis aturan fuzzy logic control

Delta error \ Error	nb	n	z	p	pb
nb	NB	NB	NB	N	Z
n	NB	NB	N	Z	P
z	NB	N	Z	P	PB
p	N	Z	P	PB	PB
pb	Z	P	PB	PB	PB

Dengan metode infrensi yaitu menggunakan metode Mamdani karena memiliki perihitungan yang lebih mudah serta defuzzifikasi menggunakan metode COG. Sebagai contoh untuk mensimulasikan system fuzzy perhatikan contoh berikut:

Saat waktu t nilai error = 0.06 dan nilai delta error adalah -0.2 maka kita hitung dulu nilai keanggotaan pada masing masing membership function:

$\mu E_1(0.06) = 0$ $\mu E_2(0.06) = 0$

$\mu E_3(0.06) = \frac{(0.1-0.06)}{0.1-0} = \frac{0.04}{0.1} = 0.4$ $\mu E_4(0.06) = \frac{(0.06-0)}{0.1-0} = \frac{0.06}{0.1} = 0.6$

$\mu E_5(0.06) = 0$

$\mu DE_1(-0.2) = 1$ $\mu DE_2(-0.2) = 0$

$\mu DE_3(-0.2) = 0$ $\mu DE_4(-0.2) = 0$

$$\mu E_5(-0.2) = 0$$

Sehingga $\mu E(0.06) = \{0, 0, 0.4, 0.6, 0\}$ & $\mu DE(-0.2) = \{1, 0, 0, 0, 0\}$;

Setelah itu kita hitung nilai keanggotaan berdasarkan rule base menggunakan inferensi Mamdani, diawali dengan menghitung operasi minimum pada nilai keanggotaan input

Delta error \ Error	nb (0)	n (0)	z (0.4)	p (0.6)	pb (0)
nb (1)	NB min(0,1)=0	NB min(0,1)=0	NB min(0.4,1)=0.4	N min(0.6,1)=0.6	Z min(0,1)=0
n (0)	NB min(0,0)=0	NB min(0,0)=0	N min(0.4,0)=0	Z min(0.6,0)=0	P min(0,0)=0
z (0)	NB min(0,0)=0	N min(0,0)=0	Z min(0.4,0)=0	P min(0.6,0)=0	PB min(0,0)=0
p (0)	N min(0,0)=0	Z min(0,0)=0	P min(0.4,0)=0	PB min(0.6,0)=0	PB min(0,0)=0
pb (0)	Z min(0,0)=0	P min(0,0)=0	PB min(0.4,0)=0	PB min(0.6,0)=0	PB min(0,0)=0

Menghhitung nilai keanggotaan output dengan operasi max

$$\mu U_1 = \max(0, 0, 0.4, 0, 0) = 0.4$$

$$\mu U_2 = \max(0.6, 0, 0, 0, 0) = 0.6$$

$$\mu U_3 = \max(0, 0, 0, 0, 0) = 0$$

$$\mu U_4 = \max(0, 0, 0, 0, 0) = 0$$

$$\mu U_5 = \max(0, 0, 0, 0, 0) = 0$$

Sehingga $\mu U = \{0.4, 0.6, 0, 0, 0\}$

Selanjutnya proses terakhir pada system fuzzy adalah deffuzifikasi untuk menghitung nilai output nyata:

$$u = \frac{\sum_{k=1}^{m} b(k) \cdot \mu u(k)}{\sum_{k=1}^{m} \mu u(k)} = \frac{-24 * 0.4 + (-12 * 0.6) + 0 * 0 + (12 * 0) + (24 * 0)}{0.4 + 0.6 + 0 + 0 + 0}$$

$$u = \frac{-9.6 + (-7.2) + 0 + 0 + 0}{0.4 + 0.6 + 0 + 0 + 0} = -16.8$$

3.4 Pemrograman Fuzzy pada contoh 3.3

Langkah percobaan:

1. Membuat listing Fuzzifikasi:

```
function xf=fuzzifikasi(x,b_mf)
xf=[0,0,0,0,0];
mf=[b_mf(1),b_mf(2),b_mf(3),b_mf(4),b_mf(5)];
    if (x <=mf(1))
       xf(1)=1;
    elseif (mf(1)< x && x<=mf(2))
       xf(1)=(mf(2)-x)/(mf(2)-mf(1));
       xf(2)=(x-mf(1))/(mf(2)-mf(1));
    elseif (mf(2)< x && x<=mf(3))
       xf(2)=(mf(3)-x)/(mf(3)-mf(2));
       xf(3)=(x-mf(2))/(mf(3)-mf(2));
    elseif (mf(3)< x && x<=mf(4))
       xf(3)=(mf(4)-x)/(mf(4)-mf(3));
       xf(4)=(x-mf(3))/(mf(4)-mf(3));
    elseif (mf(4)< x && x<=mf(5))
       xf(4)=(mf(5)-x)/(mf(5)-mf(4));
       xf(5)=(x-mf(4))/(mf(5)-mf(4));
    elseif x >= mf(5)
        xf(5)=1;
    end
end
```

2. Simpan dengan nama fuzzifikasi.m
3. Membuat listing inferensi fuzzy:

```
function uU=infuzz(in1,in2)
uE=[in1(1),in1(2),in1(3),in1(4),in1(5)];
uDE=[in2(1),in2(2),in2(3),in2(4),in2(5)];
uU=[0,0,0,0,0];
rbf=[1 1 1 2 3;
     1 1 2 3 4;
     1 2 3 4 5;
     2 3 4 5 5;
     3 4 5 5 5;];
 for i=1:5
     for j=1:5
         k=rbf(i,j);
         uU(k)=max(uU(k),min(uE(i),uDE(j)));
     end
 end
end
```

4. Simpan dengan nama infuzz.m

5. Membuat listing defuzzifikasi:

```
function u=defuzzifikasi(in)
uU=[in(1),in(2),in(3),in(4),in(5)];
B=[-24,-12,0,12,24];
num=B(1)*uU(1)+B(2)*uU(2)+B(3)*uU(3)+B(4)*uU(4)+B(5)*uU(5);
denum=uU(1)+uU(2)+uU(3)+uU(4)+uU(5);
u=num/denum;
end
```

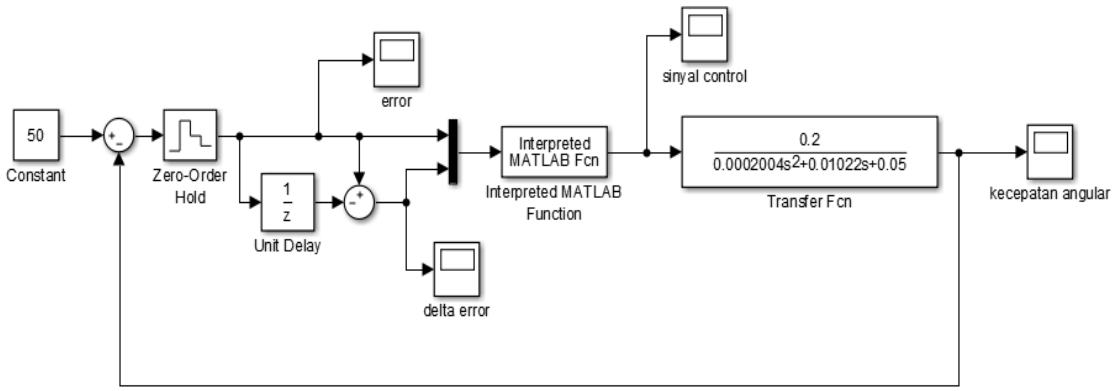
6. Simpan dengan nama defuzzifikasi.m
7. Membuat Program utama fuzzy

```
function u=fuzzy(in)
e=in(1);
de=in(2);
mE=[-1,-0.1,0,0.1,1];%membership Error
mDE=[-1,-0.1,0,0.1,1];%membershib delta Error
uE=fuzzifikasi5(e,mE);
uDE=fuzzifikasi5(de,mDE);
uU=infuz(uE,uDE);
u=defuzzifikasi(uU);
end
```

8. Simpan dengan nama fuzzy.m
9. Run program utama fuzzy.m
10. Abaikan jika muncul hasil running sebagai berikut:

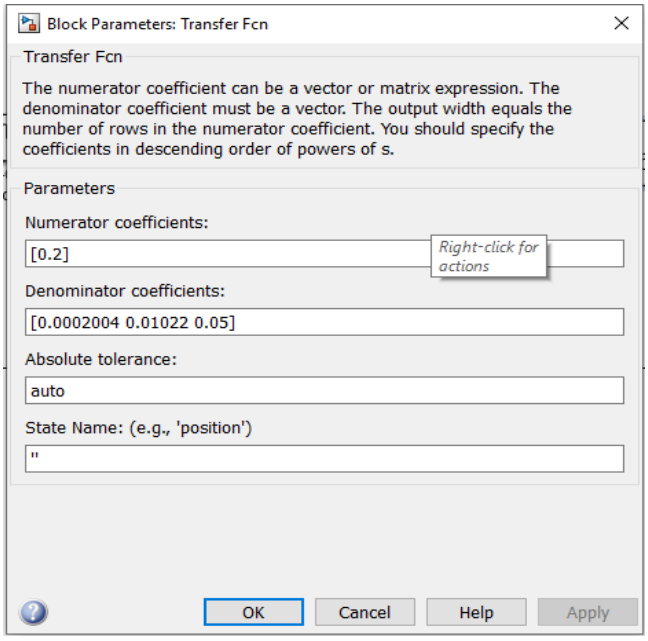
```
Command Window
>> fuzzy
Not enough input arguments.

Error in fuzzy (line 2)
e=in(1);
```

11. Buat Simulink sebagai berikut:

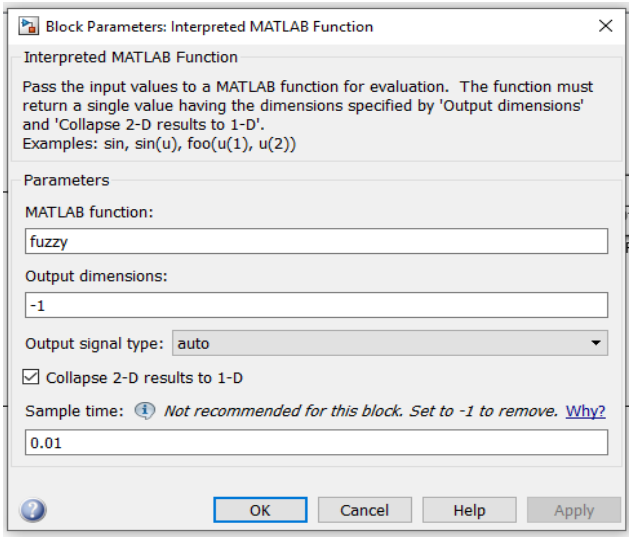


Gambar 3.8 blok Simulink fuzzy logic control pada kecepatan motor

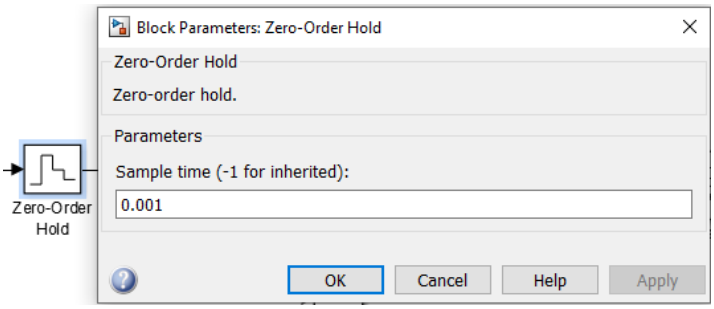
12. Pada blok treansfer fcn double click dan isikan dengan parameter berikut:



13. Click Ok
14. Pada Block *Interpreted MATLAB Function* double click dan isikan dengan parameter berikut:

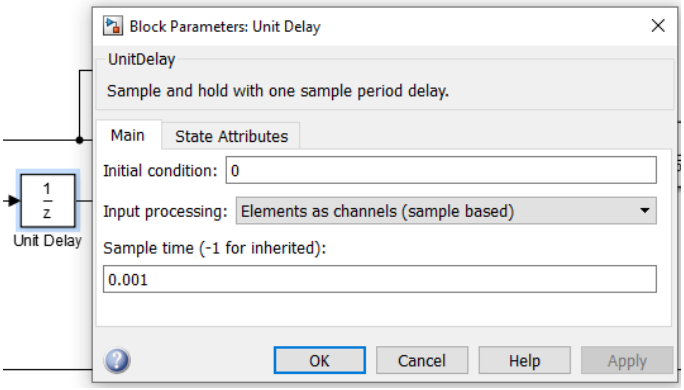


15. Click Ok
16. Pada block *zero order hold* double click dan isi time sampling seperti gambar berikut:

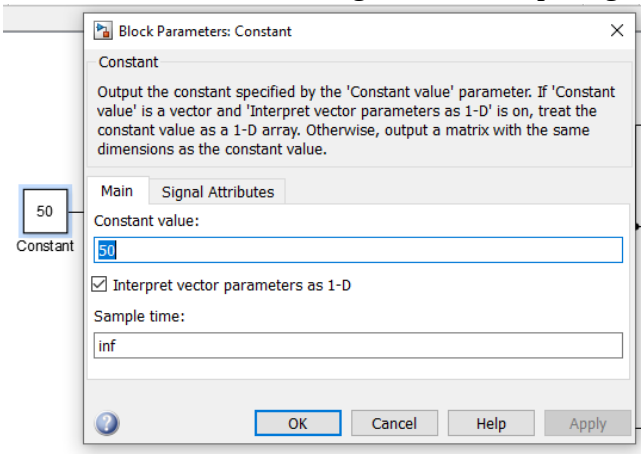


17. Click Ok

18. Pada block *unit delay* double click dan isi time sampling seperti gambar berikut:



19. Click Ok
20. Pada block constant isi dengan nilai 50 seperti gambar berikut:



21. Click ok
22. Run program dan simpan hasil grafik pada kecepatan, sinyal control, error, delta error
23. Pada listing program utama fuzzy rubah batas nilai keanggotaan dengan nilai berikut:

```
mE=[-1,-0.05,0,0.05,1];%
mDE=[-1,-0.05,0,0.05,1];
```

24. Run program kembali dan simpan hasil grafik pada kecepatan, sinyal control, error, delta error
25. Analisa dari perubahan tersebut

3.5 Data Hasil Percobaan

3.6 Analisa Data

3.7 Kesimpulan

DAFTAR PUSTAKA

[1] Santosa, B,"Metode Metaheuristic",Guna Widya,Surabaya,2011 .

[2] Siang, JJ," Jaringan Syaraf Tiruan", Andi, 2004.

[3] Pasino,K, "Fuzzy Logic", Adison Wesley, 1997.

Halaman ini sengaja dikosongkan