CHAPTER 8

CONCLUSION AND SCOPE

8.1 CONCLUSION

This thesis employs various intelligent techniques to design the power system stabilizer. The PSS parameters tuning problem is formulated as an optimization problem such as soft computing techniques are used to seek for optimal parameters. In the presented work, the PSS is designed for a two area four machine interconnected power system.

Three different design of PSS, multiband PSS (MB-PSS), conventional delta w PSS (Delta w) and conventional acceleration power PSS (Delta Pa). The simulated responses of the system power transfer from area 1 to 2, M1 speed, M1 acceleration power and M1 terminal voltage are observed. All PSSs do a good job stabilizing the naturally unstable system. However, it is clear that the multiband PSS (MB-PSS) is superior to the other two PSSs, providing significantly more damping to all modes, in particular on the Delta w PSS and Delta Pa PSS. The system lost its synchronism while the MB-PSS and the Delta W PSS succeed in maintaining stability.

Dynamic stability analysis of power system is investigated considering Proportional-Integral-Derivative power system stabilizer for modern power systems. Gain settings of PID-PSS are optimized by minimizing an objective function using Bat algorithm (BAT). The analysis reveals that the proposed PSS gives better dynamic performances as compared to that all of the mentioned methods. Controller design will be tested on the power system to prove its effectiveness.
A systematic approach for the design of fuzzy logic power system stabilizers in a multi-machine power system. Investigations reveal that the performance of fuzzy logic power system stabilizer in a multi-machine power system is quite robust under variations in load conditions as compared to the conventional controller.

From the responses, it is found that the designed PSS provides excellent damping enhancement for various operating conditions of the system. The PSS parameters are tuned within the given limits by using PID controller, PSO, BAT and fuzzy logic control strategies. In this thesis, the responses of the system are compared based on settling time, peak amplitude, power loss and speed deviation. Since all the above techniques are random search, the PSS designed based on fuzzy logic control is having better performance than PSSs designed based on other proposed control strategies.

8.2 SUGGESTIONS FOR FUTURE RESEARCH WORK

In this study, PSS is intended for a system using PID controller with various intelligence techniques. The work can be further extended by considering voltage dependent loads. The intelligent based optimum design of PSS proposed in this thesis is an interesting area of research. The Artificial Intelligent techniques like ANFIS and Neuro algorithms and dynamic relaxation are attracting much attention in the recent years. The above techniques may be used for the robust design of Power System Stabilizer (PSS).
APPENDIX

A. TWO AREA FOUR MACHINE POWER SYSTEM DATA

Two area four machines power system

**Distributed Line Parameters**
- Phases="3"
- Frequency(Hz)="60"
- Resistance=(0.0001*5291.61)
- Inductance(H)=(0.001*529/(377) 0.0061)
- Capacitance(F)=(0.00175/529/(377) 5.2489e-9)
- Length(M)=110
- Line-3

**Synchronous Machine**
- Library block: Powerlib/machines/synchronous machine
- Pu standard
- Mechanical Load= Mechanical Power Pm
- Rotor Type="Round"
- Nominal Parameters=(900E6 20000 60)
- Reactances 1[1.8 .3.25 1.7 .55 .25 .2]
  - D Axis Time Constants=Open-circuit
  - Q Axis Time Constants=Open-circuit
  - Time Constants 2=(8 .03 .4 05)
  - Stator Resistance=0.0025
  - Mechanical=(6.17504)
- Initial Conditions=(0-56 .3243 0.8041 0.8041 0.8041 -22.4087 -142.409 97.5913 1.8 5092)
- Set Saturation="off"
- Units=Per Unit Standard Parameters
- TsBlock=-1
- Iterative Model=Forward Euler
- Qmin=Inf
- Qmax=Inf

B. PSO PROGRAM FOR PSS

```matlab
function [gbest,Bfit,R] = pso(N,iter_max,n,x_max,x_min)
m_i_a=1.5;
m_i_b=1.5;
W=1;
v=zeros(N,n);
```

135
v_min = -((max(x_max) - min(x_min))/(N*5));
v_max = (max(x_max) - min(x_min))/(N*5);

gBest=zeros(1,n);
gbestvalue=0;
gaux=ones(N,1);
xBest=zeros(N,n);
bfit=zeros(N,1);
obj_r = zeros(N,1);

x=zeros(N,n);
%Swarm initialization
for i=1:N
    for j=1:n
        x(i,j)=rand(1,1) * (x_max(j) - x_min(j)) + x_min(j);
    end
end

%Initial Objective function calling
obj_r=objfun_pi(x);
bfit=obj_r;
[~, b]=min(obj_r);
gBest=x(b,:);
gbestvalue = objfun_pi(gBest);
xBest=x(b,:);

iter=1;
%Iteration
R.gbestvalue(iter)=gbestvalue;
R.xBest(iter,:)=xBest;
R.gBest(iter,:)=gBest;
while iter <= iter_max
    r1=rand(N,n);
r2=rand(N,n);
    v = (W.*v) + (2*r1.*(m_i_a.*(gaux*xBest-x)) )+( 2*r2.*(m_i_b.*(gaux*gBest-x)));
    v = (v <= v_min).*v_min ) + ( (v > v_min).*v );
    v = ( v >= v_max).*v_max ) + ( (v < v_max).*v );
    x = x+v;  %updated position of x
    for j = 1:N,
        for k = 1:n
            if x(j,k) < x_min(k)
                x(j,k) = x_min(k);
            elseif x(j,k) > x_max(k)
                x(j,k) = x_max(k);
            end
        end
    end
/obj_r=objfun_pi(x);

% for j=1:N
% if obj_r(j) < bfit(j)

136
C. FUZZY LOGIC CONTROL

A mamdani type Fuzzy logic control proposed in this work, the rule and membership details as follows as,

Name='frule'
Type='mamdani'
Version=2.0
NumInputs=2
NumOutputs=2
NumRules=25
AndMethod='min'
OrMethod='max'
ImpMethod='min'
AggMethod='max'
DefuzzMethod='centroid'

[Input1]
Name='error'
Range=[-0.01 0.01]
NumMFs=5
MF1='NB':'trimf',[-0.015 -0.01 -0.005]
MF2='NS':'trimf',[-0.01 -0.005 0]
MF3='Z':'trimf',[-0.005 0 0.005]
MF4='PS':'trimf',[0 0.005 0.01]
MF5='PB':'trimf',[0.005 0.01 0.015]

[Input2]
Name='d__error'
Range=[-0.01 0.01]
NumMFs=5
MF1='NB':'trimf',[-0.015 -0.01 -0.005]
MF2='NS':'trimf',[-0.01 -0.005 0]
MF3='Z':'trimf',[-0.005 0 0.005]
MF4='PS':'trimf',[0 0.005 0.01]
MF5='PB':'trimf',[0.005 0.01 0.015]

[Output1]
Name='Kp'
Range=[0 2]
NumMFs=5
MF1='NB':'trimf',[-0.5 0.5]
MF2='NS':'trimf',[0 0.5 1]
MF3='Z':'trimf',[0.5 1.5]
MF4='PS':'trimf',[1.5 2]
MF5='PB':'trimf',[1.5 2.5]

[Output2]
Name='ki'
Range=[0 2]
NumMFs=5
MF1='mf1':'trimf',[-0.5 0 0.5]
MF2='mf2':'trimf',[0 0.5 1]
MF3='mf3':'trimf',[0.5 1 1.5]
MF4='mf4':'trimf',[1 1.5 2]
MF5='mf5':'trimf',[1.5 2 2.5]

[Rules]
1 1, 5 5 (1) : 1
1 2, 5 5 (1) : 1
1 3, 5 5 (1) : 1
1 4, 4 4 (1) : 1
1 5, 4 4 (1) : 1
2 1, 5 5 (1) : 1
2 2, 5 5 (1) : 1
2 3, 4 4 (1) : 1
2 4, 3 3 (1) : 1
2 5, 2 2 (1) : 1
3 1, 5 5 (1) : 1
3 2, 4 4 (1) : 1
3 3, 3 3 (1) : 1
3 4, 2 2 (1) : 1
3 5, 2 2 (1) : 1
4 1, 4 4 (1) : 1
4 2, 3 3 (1) : 1
4 3, 2 2 (1) : 1
4 4, 1 1 (1) : 1
4 5, 1 1 (1) : 1
5 1, 3 3 (1) : 1
5 2, 2 2 (1) : 1
5 3, 1 1 (1) : 1
5 4, 1 1 (1) : 1
5 5, 1 1 (1) : 1