Chapter 10

Bacteria Foraging Tuned TS-Fuzzy PSS Design for damping Low Frequency Oscillations

A new algorithm from the family of evolutionary computation, known as bacteria foraging algorithm (BFA) [124] emerged recently in 2002 the process in which natural selection tends to eliminate animals with poor foraging strategies and favor the propagation of genes of those animals that have successful foraging strategies since they are more likely to enjoy reproductive success. After many generations, poor foraging strategies are either eliminated or shaped into good ones. The E. coli bacteria that are present in our intestines also undergo a foraging strategy. The control system of these bacteria that dictates how foraging should proceed can be subdivided into four sections namely Chemotaxis, Swarming, Reproduction and Elimination and Dispersal. Foraging theory is based on the assumption that animals search for and obtain nutrients in a way that maximizes their energy intake $E$ per unit time $T$ spent foraging.

BFA recently proposed was applied for harmonic estimation problem in power systems [125]. The algorithm is based on the foraging behavior of E. coli bacteria present in human intestine. The UPFC location, series injection voltage, and transformer tap positions are simultaneously optimized as control variables, so that the multiple objectives are fulfilled, keeping an eye to all specified constraints. The results so obtained show its strength in solving highly nonlinear epistatic problems. Also BFA is used to solve a combined CPF-OPF problem of real power loss minimization and VSL maximization of the system [126].

Unfortunately, recent research has identified some deficiencies in GA performance [111]. This degradation in efficiency is apparent in applications with highly epistatic objective function (i.e., where the parameters being optimized are highly correlated). Also, the premature convergence of GA degrades its performance and reduces its search capability, paving a search to improved optimization algorithms.
10.1 BFA in TS-Fuzzy Logic Controller Design

In this scheme, the foraging (methods for locating, handling, and ingesting food) behavior of E. coli bacteria present in our intestines is mimicked [124]. They undergo different stages such as chemotaxis, swarming, reproduction, and elimination and dispersal. In the chemotaxis stage, it can have tumble followed by a tumble or a tumble followed by a run. On the other hand, in swarming, each E. coli bacterium will signal other via attractants to swarm together. Furthermore, in reproduction the least healthy bacteria die and the other healthiest bacteria each split into two bacteria, which are placed in the same location. Besides, in elimination and dispersal, any one bacterium is eliminated from the total set just by dispersing it to a random location on the optimization domain.

The proposed BFA Tuned TS-fuzzy [127] approach as shown in Figure 10.1 is based on foraging algorithms and integrates the simultaneous design of three stages of the TS-fuzzy controller: the universe of discourse of the linguistic variables, the shape of the membership functions and the set of fuzzy rules. Starting from a collection of fuzzy controllers whose parameters are randomly chosen this approach allows the simultaneous determination of the fuzzy controller parameters, except the number of membership functions of the linguistic variables, the inference method and the fuzzification and defuzzification operators.

10.2 Bacteria Foraging Tuned TS-Fuzzy PSS

This Power System Stabilizer designed to damp low frequency oscillations termed as Bacteria Foraging Tuned Neuro-Fuzzy (BNFPSS) is designed with two inputs, the generator speed deviation $\Delta \omega$ and its derivative $\Delta \dot{\omega}$, and one control output $u_c$. The input variables generator speed deviation and its derivative are converted to linguistic variables specified by Gaussian membership functions and as a result 49 rules are devised. The rule-base contains the fuzzy IF-THEN rules of Takagi-Sugeno's first order type [4] in which the output of each rule is a linear combination of input variables plus a constant term. The final output is the weighted average of each rule's output.
Figure 10.1. BFA TS-Fuzzy optimization algorithm
10.2.1 BNFPSS Architecture

The architecture of the BNFPSS sensing $\Delta \omega$ and $\dot{\Delta \omega}$ is shown in Figure 10.2, where node functions in each layer are as described below.

![Figure 10.2 Architecture of BNFPSS.](image)

This TS-fuzzy network is composed of five layers containing fuzzification of the input variables, fuzzy rule operation and defuzzification functions. The detailed structure of Takagi Sugeno Fuzzy [93] as described in section 5.1 is used to design the TS-Fuzzy System with seven Gaussian shaped membership functions for the inputs speed deviation $\Delta \omega$ and its derivation $\dot{\Delta \omega}$. The TS-fuzzy model of a system, can be expressed by

$$\text{IF } (\Delta \omega \text{ is } A_1 \text{ and } \Delta \omega \text{ is } B_1) \text{ THEN}$$

$$\text{Output } u_E = p_i * \Delta \omega + q_i * \dot{\Delta \omega} + r_i$$

$A_1, \ldots, A_k, B_1, \ldots, B_k$ denote fuzzy sets with appropriate membership functions, which are also referred to as premise parameters. $\{p_i, q_i, r_i\}$ represent parameters of consequence.

Layer 1 calculates the value of the membership function that is of gaussian-shaped:

$$O_{ij} = \mu_{A_j}(x_i) = \exp \left[ -\frac{(x_i - c_{ij})^2}{\sigma_j^2} \right]$$

(10.1)
where $i=1, 2$, $j=1, 2... ,7$

$x_i$, is the input to this layer ($\Delta \omega, \Delta \omega$) and $c_{ij}$, is the center of the membership function, $\sigma_j$ are the width of the $j^{th}$ fuzzy set $A_i$.

**Layer 2** performs and operation of Fuzzy AND operation of the input signals speed deviation and its derivative. It is equivalent to the meaning of the firing strength in fuzzy system $w_i$.

**Layer 3** calculates the $i$-th firing strength proportional to the sum of all the firing strengths $\bar{w}_i$.

**Layer 4** The nodes in this layer output the weighted consequent part of the rule table.

$$O_{4,i} = \bar{w}_i f_i = \bar{w}_i (p_i \Delta \omega + q_i \Delta \omega + r_i) \quad i=1,...,49. \quad (10.2)$$

where $\{p_i, q_i, r_i\}$ is the parameter set of this node.

**Layer 5** The single node in this layer computer the overall output as the summation of all the incoming signals.

$$O_{5,i} = \sum \bar{w}_i f_i \quad i=1,...,49. \quad (10.3)$$

where $O_{5,i}$ denote the output in layer 5.

The Learning Scheme of the TS-Fuzzy system [91] is done as described in detail in Section 4.10. The BNFPSS training is done assuming that there is no expert available and the initial values of the membership functions parameters are equally distributed along the universe of discourse and all consequent parts of the rule table set to zero. The BNFPSS starts from zero output and during training it gradually learns the rules and functions as close to the desired controller. Thus during training the network structure
update membership functions and rule base parameters according to the gradient descent update procedure.

10.2.2 Bacteria Foraging based Optimization of TS-Fuzzy PSS

Natural selection tends to eliminate animals with poor foraging strategies [124] and favor the propagation of genes of those animals that have successful foraging strategies since they are more likely to enjoy reproductive success. After many generations, poor foraging strategies are either eliminated or shaped into good ones. The E. coli bacteria that are present in our intestines also undergo a foraging strategy. Therefore, in this approach, the bacteria foraging has been used for optimizing the scaling coefficient (G1, G2 and Gu) of TS fuzzy scheme for the multimachine power system PSS. The control system of these bacteria that dictates how foraging should proceed can be subdivided into four sections namely Chemotaxis, Swarming, Reproduction and Elimination and Dispersal.

The scaling coefficients input gains (G1, G2) and output gain (Gu) of the TS_Fuzzy [79] are taken as individuals in BF Algorithm, and are represented by as real coded for each individual.

A) Chemotaxis: This process is achieved through swimming and tumbling via Flagella. Depending upon the rotation of Flagella in each bacterium, it decides whether it should move in a predefined direction (swimming) or altogether in different directions (tumbling), in the entire lifetime. To represent a tumble, a unit length random direction, say \( \phi(j) \), is generated; this will be used to define the direction of movement after a tumble. In particular

\[
\theta'(j + 1, k, l) = \theta'(j, k, l) + C(i)\phi(j) \tag{10.4}
\]

where \( \theta'(j, k, l) \) represents the \( i^{th} \) bacterium at \( j^{th} \) chemotactic, \( k^{th} \) reproductive and \( l^{th} \) elimination and dispersal step. \( C(i) \) is the size of the step taken in the random direction specified by the tumble (run length unit).
B) **Swarming**: During the process of reaching towards the best food location it is always desired that the bacterium which has searched the optimum path should try to provide an attraction signal to other bacteria so that they swarm together to reach the desired location. In this process, the bacteria congregate into groups and hence move as concentric patterns of groups with high bacterial density.

The mathematical representation for swarming can be represented by

\[
J_{sw}(\theta, P(j, k, l)) = \sum_{i=1}^{S} J_{cc}(\theta, \theta'(j, k, l))
\]

\[
= \sum_{i=1}^{S} \left[ -d_{\text{attract}} \exp(-\omega_{\text{attract}} \sum_{m=1}^{P} \theta_m - \theta'_m) \right] + \sum_{i=1}^{S} \left[ h_{\text{repellent}} \exp(-\omega_{\text{repellent}} \sum_{m=1}^{P} \theta_m - \theta'_m) \right]^{10.5}
\]

where \( J_{sw}(\theta, P(j, k, l)) \) is the cost function value to be added to the actual cost function to be minimized to present a time varying cost function. \( S \) is the total number of bacteria and \( P \) the number of parameters to be optimized which are present in each bacterium. \( d_{\text{attract}}, \omega_{\text{attract}}, h_{\text{repellent}}, \omega_{\text{repellent}} \) are different coefficients.

C) **Reproduction**: The least healthy bacteria die and the other healthiest bacteria each split into two bacteria, which are placed in the same location. This makes the population of bacteria constant.

D) **Elimination and Dispersal**: It is possible that in the local environment the live of a population of bacteria changes either gradually (e.g., via consumption of nutrients) or suddenly due to some other influence. Events can occur such that all the bacteria in a region are killed or a group is dispersed into a new part of the environment. They have the effect of possibly destroying the chemotactic progress, but they also have the effect of assisting in chemotaxis, since dispersal may place bacteria near good food sources. From a broad perspective, elimination and dispersal are parts of the population-level long-distance motile behavior.
10.2.3 Formulation of Objective Function

The Bacterial Foraging scheme has been used for the optimization of Takagi Sugeno-Fuzzy based PSS parameters. Just like any other optimization problem, a cost or an objective function needs to be formulated for the optimal PSS design. The objective in the optimal PSS design is to maximize damping; in other words minimize the overshoots and settling time in system oscillations.

The objective function is

\[ J = \int_{t=0}^{t=T} \left( \sum_{i,j} \left| \omega_i - \omega_j \right|^2 \right) dt \]

where \( i, j \)-participating generators \( (10.6) \)

The objective function is the total error of the speed signals corresponding to the inter-area and local mode of oscillations over a specific time period.

The Integral of Time Squared Error (ITSE) is considered as the cost function to be minimized by the bio-inspired algorithm. Integral of Squared Error (ISE) accounts mainly for error at the beginning of the response and to a lesser degree for the steady state duration. ITSE is a better criterion which keeps account of errors at the beginning but also emphasizes the steady state.

10.2.4 Steps involved in bacterial foraging algorithm

The algorithm of the proposed scheme is as follows:

Step 1-Initialization

i. Number of parameters \( (\rho) \) to be optimized.

ii. Number of bacteria \( (S) \) to be used for searching the total region.
iii. Swimming length $N_s$ after which tumbling of bacteria will be undertaken in a chemotactic loop.

iv. $N_c$ the number of iteration to be undertaken in a chemotactic loop. ($N_c > N_s$).

v. $N_{re}$ the maximum number of reproduction to be undertaken.

vi. $N_{ed}$ the maximum number of elimination and dispersal events to be imposed over the bacteria.

vii. $P_{ed}$ the probability with which the elimination and dispersal will continue.

viii. The location of each bacterium $P (1-p, 1-S, 1)$ which is specified by random numbers on $[-1, 1]$.

ix. The value of $C (i)$ which is assumed to be constant in our case for all the bacteria to simplify the design strategy.

x. The values of $d_{attract}, \omega_{attract}, h_{repellent}$ and $\omega_{repellent}$.

In this simulation work we have considered the foraging parameters as $S=20, p=6, N_c=20, N_s=4, N_{re}=100, N_{ed}=2, P_{ed}=0.25, d_{attract}=0.1, \omega_{attract} = 0.2, h_{repellent} = 0.1$ and $\omega_{repellent} = 10$.

Step-2 Iterative algorithm for optimization

This section models the bacterial population chemotaxis, swarming, reproduction, elimination and dispersal (initially, $j=k=l=0$). For the algorithm updating $\theta^t$ automatically results in updating of $'P'$.

1) Elimination-dispersal loop: $l=l+1$

2) Reproduction loop: $k=k+1$

3) Chemotaxis loop: $j=j+1$
a) For $i = 1, 2, \ldots, S$, calculate cost function value for each bacterium $i$ as follows.

- Compute value of cost function $J(i, j, k, l)$ using equation 10.6

Let $J_{sw}(i, j, k, l) = J(i, j, k, l) + J(i, j, k, l)P(j, k, l)$ (i.e., add on the cell to cell attractant effect for swarming behavior).

- Let $J_{last} = J_{sw}(i, j, k, l)$ to save this value since we may find a better cost via a run.
- End of For loop

b) For $i = 1, 2, \ldots, S$ take the tumbling/swimming decision

- Tumble: Generate a random vector $\Delta(i) \in \mathbb{R}^p$ with each element $\Delta_m(i)$, $m = 1, 2, \ldots, p$, a random number on [-1, 1].
- Move: let

$$\theta'(j + 1, k, l) = \theta'(j, k, l) + C(i) \frac{\Delta(i)}{\sqrt{\Delta^T(i)\Delta(i)}}$$

Fixed step size in the direction of tumble for bacterium $i$ is considered.

- Compute $J(i, j + 1, k, l)$ and then let

$$J_{sw}(i, j + 1, k, l) = J(i, j + 1, k, l) + J_{cc}(i, j + 1, k, l)P(j, k, l)$$

- Swim:

i) Let $m=0$; (counter for swim length)

ii) While $m < N_s$ (have not climbed down too long)

- Let $m=m+1$
- If $J_{sw}(i, j + 1, k, l) < J_{last}$ (if doing better), let $J_{last} = J_{sw}(i, j + 1, k, l)$ and let
\[ \theta'(j+1,k,l) = \theta'(j,k,l) + C(i) \frac{\Delta(i)}{\sqrt{\Delta'(i)\Delta(i)}} \]

and use this \( \theta'(j+1,k,l) \) to compute the new \( J(i, j+1, k, l) \)

- Else, let \( m=N_s \). This is the end of the while statement.

c) Go to next bacterium \((i+1)\) if \( i \neq S \) (i.e. go to b) to process the next bacterium.

d) If \( j < N_e \), go to step 3. In this case, continue chemotaxis since the life of the bacteria is not over.

5) Reproduction

a) For the given \( k \) and \( l \), and for each \( i=1, 2, \ldots, S \), let \( J'_{\text{health}} = \min_{j \in \{1, \ldots, N_n\}} \{ J_{\text{sw}}(i, j, k, l) \} \)

be the health of the bacterium \( i \) (a measure of how many nutrients it got over its life time and how successful it was at avoiding noxious substance). Sort bacteria in order of ascending cost \( J_{\text{health}} \) (higher cost means lower health).

b) The \( S_r = S/2 \) bacteria with highest \( J_{\text{health}} \) values die and other \( S_r \) bacteria with the best value split (and the copies that are made are placed at the same location as their parent)

6) If \( k < N_{re} \) go to 2, in this case, we have not reached the number of specified reproduction steps, so we start the next generation in the chemotactic loop.

7) Elimination-dispersal: For \( i=1, 2, \ldots, S \), with probability \( P_{ed} \), eliminates and disperses each bacterium (this keeps the number of bacteria in the population constant) to a random location on the optimization domain.

The detailed flow chart of the above algorithm for tuning TS-Fuzzy PSS is shown in Figure 10.3.
Initialize $S=20$, $p=6$, $N_{c}=20$, $N_{z}=4$, $N_{w}=100$, $N_{sd}=2$, $P_{sd}=0.25$, $d_{attract}=0.1$, $\omega_{attract}=0.2$, $h_{repel}=0.1$ and $\omega_{repel}=10$

Scaling Factors $G_{1}=1$, $G_{2}=1$, $G_{u}=1$

Figure 10.3 Flowchart of the Proposed SANFPSS design.
**Stopping criteria**

With various stopping criteria's reported such as maximum number of functional evaluation, convergence criteria, computation time etc, in this work, the number of iteration to be undertaken in a chemotactic loop has been used as stopping criteria.

The convergence rate of the objective function in 73 BFA reproductions are shown in Figure 10.4.

The BNFPSS was trained by data's created from Power System Stabilizers designed for various operating conditions in which the generator output ranges from 0.2 to 1.0 pu and power factor ranging from 0.85 lead to 0.4 lag. The wide spectrum of possible disturbances used for the training are: reference voltage and infinite bus voltage disturbances in the range of -0.05 pu to 0.05 pu, torque variations from -0.15 pu to 0.15 pu, three phase fault transients, transmission line with different line reactance disturbances, different machine inertia disturbances and one transmission line outage. A total of 4672 input-output data pairs are created for the training of BNFPSS.

![Fitness evaluation graph in Optimizing TS-Fuzzy using BFA.](image-url)
10.3 Simulation Studies

The performance of the designed BNFPSS was investigated on a power system model of the three machine nine bus system [3] with third generator considered as the infinite bus. The multimachine parameters are detailed in Appendix-II. A single-machine part of the schematic diagram of the multi-machine system used for simulation studies is shown in Figure 10.5. The Simulink and Fuzzy Logic toolbox of MATLAB [96, 97] are used for modeling the power system and designing the SNFPSS respectively as shown in Appendix-IV. Using M-file the Bacteria Foraging Algorithm is developed. A number of studies have been performed to investigate the effect of PSS designed by the Adaptive Bacteria Foraging based TS-Fuzzy control approach. The conventional PSS is designed using lag-lead compensator for every operating condition as detailed in Appendix-II. The control output for both the SNFPSS and CPSS was limited to 0.10 pu.

![Schematic Diagram](https://example.com/schematic_diagram.png)

**Figure 10.5 Simulation Studies of System using BNFPSS**

The designed BNFPSS is tested for various dynamic and transient disturbances and their performances are compared with conventional PSS. The various case studies performed discussed below proves the effectiveness of the proposed BNFPSS controller.
i. Light load test:

With the generator working under a light load condition, 0.4 pu active power and 0.2 pu lag, a 3% step increase to torque was applied. The disturbance is large enough to cause the system to operate in the non-linear region. System performance under such non-linear condition is shown in Figure 10.6. It can be seen that the BNFPSS damps out the oscillations very efficiently.

![Graph 1](image1)

![Graph 2](image2)

Figure 10.6 (a) System Response in generator 1 for light load test at an active power of 0.4 pu and reactive power 0.2 pu a 3% step increase in input torque.
Figure 10.6 (b) System Response in generator 2 for light load test at an active power of 0.4 pu and reactive power 0.2 pu a 3% step increase in input torque.
ii. Load test:

With the generator operating at an active power of 0.9 pu and 0.7 pu reactive power a 1% step increase in input torque was applied. The disturbance is large enough to cause the system to operate in the non-linear region. System response without PSS and with the CPSS and BNFPSS under these conditions was shown in Figure 10.7. The system without stabilizer is highly oscillatory. Although the CPSS is effective in damping the oscillations, the BNFPSS settles the oscillations smoothly and quickly.

Figure 10.7 (a) System Response in generator 1 for heavy load test at an active power of 0.9 pu and reactive power 0.7 pu a 1% step increase in input torque.
Figure 10.7 (b) System Response in generator 2 for heavy load test at an active power of 0.9 pu and reactive power 0.7 pu a 1% step increase in input torque.
With the generator operating at a power of 0.6 pu and 0.6 pu reactive a 2% step decrease in input torque was applied at 5s. The disturbance was removed at 30s and the system returned to the original operating point. The disturbance is large enough to cause the system to operate in the non-linear region. System response without PSS and with the CPSS and BNFPSS under these conditions was shown in Figure 10.8. The system without stabilizer is highly oscillatory. Although the CPSS is effective in damping the oscillations, the BNFPSS settles the oscillations smoothly and quickly.

Figure 10.8 (a) System Response in generator 1 for heavy load test at an active power of 0.6 pu and reactive power 0.6 pu a 2% step decrease in input torque applied at 5 sec and removed at 30 sec.
Figure 10.8 (b) System Response in generator 2 for heavy load test at an active power of 0.6 pu and reactive power 0.6 pu a 2% step decrease in input torque applied at 5 sec and removed at 30 sec.
Leading pf Load test:

When the generator is operating at a leading power factor, the situation is much more difficult because the stability margin is reduced. However, in order to absorb the capacitive charging current in a high voltage power system, it may become necessary to operate the generator at a leading power factor. It is therefore desirable that the controller be able to guarantee stable operation of the generator under leading power factor condition. With the generator operating at an active power of 0.3 pu and leading reactive power 0.2 pu, a 2.5% step decrease in torque was applied. The results given in Figure 10.9 show that the oscillation of the system is damped out rapidly by the BNFPSS.

Figure 10.9 (a) System Response in generator 1 for leading power factor test at an active power of 0.3 pu and reactive power 0.2 pu a 2.5% step increase in input torque
Figure 10.9 (b) System Response in generator 2 for leading power factor test at an active power of 0.3 pu and reactive power 0.2 pu a 2.5% step increase in input torque
iv. **Switching line Disturbance test:**

At an operating point of 0.5 pu active power, 0.6 pu reactive power, one circuit of the double circuit transmission line between buses 7, 8 and 9 was switched off at 5s and re-energized at 25s, thereby returning the system to the original operating point. System response without PSS and with the CPSS and BNFPSS under these conditions was shown in Figure 10.10. The response with BNFPSS shows less oscillations demonstrating better performance.

![Graph](image)

Figure 10.10 (a) System Response in generator 1 for switching line disturbance test at an active power of 0.5 pu and reactive power 0.6 pu.
Figure 10.10 (b) System Response in generator 2 for switching line disturbance test at an active power of 0.5 pu and reactive power 0.6 pu.
**Different machine Inertia test:**

With the generator operating at 0.8 pu active power, 0.6 pu reactive power test was conducted by change in machine inertia of generator 1 from \( H = 6.4 \) s to 12.8 s and of generator 2 from \( H = 3.01 \) s to 6.02 s. For this system the oscillation frequencies gets different with the change of inertia constant. System response with the CPSS and BNFPSS was shown in Figure 10.11. Because the BNFPSS is trained for wide range of conditions it yields best performance than the CPSS with fixed parameters.

![Graph](image_url)

**Figure 10.11 (a) System Response in generator 1 for different machine inertia change at an active power of 0.8 pu and reactive power 0.6 pu.**
Figure 10.11 (b) System Response in generator 2 for different machine inertia change at an active power of 0.8 pu and reactive power 0.6 pu.
vi. Fault test:

The behavior of the BNFPSS under transient conditions was verified by applying a fault. At an operating condition of 0.8 pu active power and 0.8 pu reactive power a three phase to ground short circuit is applied at 3 sec in the middle of transmission line in bus 7 and 8, cleared 100 ms later by the disconnection of the faulted line and successful reclosure after 6 sec. The response is shown in Figure 10.12. Results show that BNFPSS help the system to reach the new operating point very quickly.

Figure 10.12 (a) System Response in generator 1 for three phase fault applied at 3 s and cleared after 100 ms and reclosure at 6 sec.
Figure 10.12 (b) System Response in generator 2 for three phase fault applied at 3 s and cleared after 100 ms and reclosure at 6 sec.
10.4 Summary

The Bacterial foraging algorithm is proposed to the robust TS-Fuzzy PSS design problem. The proposed design approach employs BFA to search for optimal settings of the TS-Fuzzy power system stabilizers. The effectiveness of the suggested technique in enhancing stability of three machine nine bus multimachine power systems is verified through simulation results with different disturbances. Simulation results show that, application of TS-Fuzzy PSS tuned by the proposed bacteria foraging algorithm approach gives the best response in terms of overshoot and settling time compared to conventional PSS design. The first swings in the power angle, speed deviation and the electrical power are also greatly suppressed and the settling time is greatly reduced with the simultaneous design approach in comparison with the other tuning algorithms.