International Journal of Hybrid Information Technology
Vol.9, No.1 (2016), pp. 65-80
http://dx.doi.org/10.14257/ijhit.2016.9.1.07

Performance Analysis of Active Disturbance Rejection Controlled Robotic Manipulator based on Evolutionary Algorithm

Ankur Goel[*1] and Akhilesh Swarup[2]

Department of Electrical Engineering, National Institute of Technology, Kurukshetra, Haryana (India)
[1]anckurgoel@gmail.com, [2]akhilesh.swarup@gmail.com

Abstract

Active disturbance rejection control (ADRC) is a unique control strategy that combines the effectiveness of error driven PID controller, usefulness of state observer and strength of nonlinear feedback. This control algorithm, not only actively (online) estimates but also compensates the effects of unknown internal and external disturbances, present inherently in the plant with the help of a well-tuned extended state observer (ESO). Although, the classical solution to the parameter tuning performed by using parameterization technique provides good solution, it is not optimal for having desired performance specifications. Consequently, it became imperative to have intelligent tuning technique to achieve optimized solution to parameter tuning problem. In this regards, the Evolutionary Algorithms (EA), inspired by natural system and based on swarm intelligence, are proven to be the best tool to find the optimized solution of multi-dimensional problems.

This paper presents an application of an EA optimized ADRC controller on an uncertain 2-DoF revolute-prismatic (RP) robotic manipulator for efficient trajectory tracking and parametric robustness. Eventually, the conventional ADRC design problem is converted into a special optimization problem for finding the optimal controller tuning parameters. To accomplish this task, two well-known EA's viz. Particle Swarm Optimization (PSO) and Bacteria Foraging Optimization (BFO) are implemented and performance of EA based controller-plant configuration is individually analyzed for each algorithm.

The results of this note illustrate the benefits and weakness of the EA for implementing ADRC on MIMO systems. The performance of both the optimization techniques is compared in terms of computational time and convergence efficiency. Further, the optimized controllers are tested for the robustness in presence of disturbance and sensor noise to imitate real engineering. MATLAB based simulation results are presented and compared to demonstrate the effectiveness of both the EA's in designing an ADRC controller for improving manipulator tracking ability.

***Keywords**: Active Disturbance Rejection Control, Bacteria Foraging optimization, Particle Swarm optimization, Robotic manipulator*

1. Introduction

In real time engineering problems, the most important control objective is to counter the inherent internal (parameter or unmodeled dynamics) and external (disturbance) uncertainties. In this regard, selection of relevant control methodology for obtaining desired performance specification as well as providing robustness against the acting perturbation, is the main goal of the control engineer. Some elegant methods like adaptive control [1, 2], robust control [3, 4] *etc.*, have been proposed to solve the issues pertaining

*Corresponding Author

to the internal and external disturbances in the practical systems. These control strategies gain popularity due to their simple and effective designs. However, these methods require mathematical model of the system [5] to be fully known which is not feasible practically. On the other hand, another techniques like Disturbance Observer (DO) [6], Unknown Input Observer (UIO) [7], and Perturbation Observer [8] *etc.*, provides solution of the disturbance attenuation by first estimating the perturbation and then compensating its effects[9]. Although, these methods provide simple and general method for LTI system but they also based on the assumption that precise mathematical description of the uncertain plant is available. Some other effective methods used for nonlinear and time-variant plants such as Model Estimator [2] technique and Time Delay Control [10] *etc.*, do not require a full analytical description of the plant but the requirement of the information of higher order derivatives of the plant output signals restricts their implementation in practical industrial applications.

Active Disturbance Rejection Control method (or ADRC) is a unique approach for handling the system uncertainty which do not require the complete mathematical model and consists the virtual states representing the total disturbance (internal and external disturbances) [11-13]. Its robustness and adaptive strength rely on properly tuned special observer for effective estimation and compensation of the uncertainties, forcing an unknown plant to behave like a nominal one as shown in various benchmark tests [14, 15]. However, this ability of ADRC depends on the selection of observer and controller bandwidth settings. In [11], it is shown that the ADRC tuning procedure generally use parameterization technique employing practical knowledge of the control engineer. The solution hence obtained can be good but not optimal one which is necessary to meet the challenging performance requirement in terms of stability, noise sensitivity and operation cost. Consequently, the problem of optimal solution for the ADRC parameters can be accomplish by using the tool based on Evolutionary Algorithm [16].

An Evolutionary Algorithms [17] are generic population-based meta-heuristic optimization algorithms based on the stochastic search and optimization methodology inspired by the nature. These algorithms follow the rules of the adaption mechanism combined with biological evolution, such as reproduction, mutation, recombination, and selection[18]. Many of these algorithms are inspired from molecular evolution, population genetics, immunology, *etc.*, [19]. These algorithms provide much better solutions then the traditional counterparts in case of the designs involving processing inaccuracy, noisy and complex data.

The major objective of this work is to provide comparative analysis of computational effectiveness and efficiency of well-known Evolutionary algorithms viz. PSO and BFO optimization algorithm for an ADRC designed for 2-DoF RP robotic manipulator. The design objective is to improve the tracking ability of the arm in presence of parametric uncertainties and external disturbances. The design problem is converted into an optimization problem and both PSO and BFO optimization techniques are employed to search the optimal ADRC controller parameters. The performance of both the EAs, in terms of convergence and computational time, is compared. Furthermore, the robust behavior of the Bacteria foraging based ADRC controller (BFOADRC) and PSO based ADRC (PSOADRC) in presence of external disturbances and sensor noise is verified, compared and analyzed. MATLAB simulation results are also presented to demonstrate the efficacy of the proposed EA based controller and to support the objective of improving the tracking performance of the robotic manipulator.

This paper is organized into six sections. The dynamics of 2-DOF robotic manipulator is given in section two along with design of ADRC controller in section three. The brief of PSO and BFO is shown in section four. Section five formulates the problem reflecting the objective of the paper. The sixth section gives the results obtained from the synthesis of ADRC controller for robotic manipulator from BFO and PSO with discussion. The conclusion is presented in the last.

2. Robot Dynamics

The dynamics of n-link rigid robotic manipulator, an example of highly coupled and non-linear system, can be formulated from the Euler-Lagrangian theory [20] as-

$$M(q)\ddot{q}+C(q,\dot{q})\dot{q}+G(q)=\tau \qquad (1)$$

where, $q \in R^{nx1}$ denotes the vector of *n*-joint positional coordinates, $\dot{q} \in R^{nx1}$ is a vector of joint velocity, $M(q) \in R^{nxn}$ is the bounded, symmetric, positive definite inertial matrix, $C(q,\dot{q})\dot{q} \in R^{nx1}$ represents Coriolis and centrifugal torques, $G(q) \in R^{nx1}$ is a gravity vector, and $\tau \in R^{nx1}$ represents the vector of control torques applied to the joints.

The dynamics of a general robotic manipulator, with torques as an inputs and the actual positions as an output can be represented by re-arranging (1) as-

$$\ddot{q}=M^{-1}(q)\{\tau - C(q,\dot{q})\dot{q} - G(q)\} \qquad (2)$$

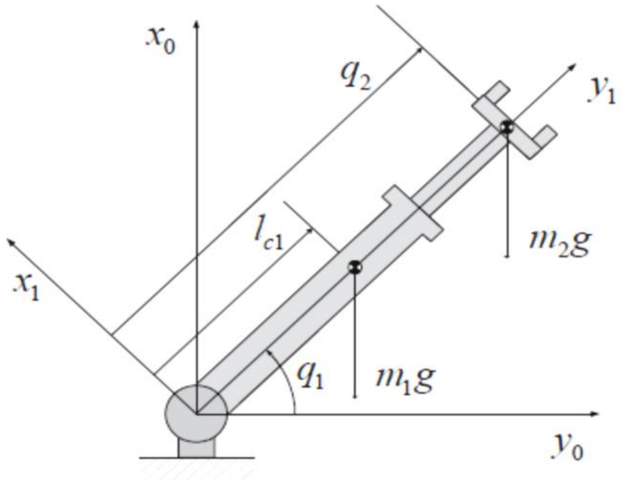


Figure 1. 2-DoF RP Joint Robotic Manipulator [21]

Consider a 2-DoF planar robotic manipulator with one revolute and prismatic joint [21] as shown in figure 1.The basic notations used for describing the manipulator are provided in Table 1. The matrices *M(q)*, $C(q,\dot{q})$ and *G(q)*, used in (2) for dynamic equation are given by

$$M(q)=\begin{pmatrix} M_{11}(q) & M_{12}(q) \\ M_{21}(q) & M_{22}(q) \end{pmatrix} \qquad (3)$$

where,

$$M_{11}=\frac{1}{4}m_1a_1^2+m_2q_2^2+m_2d_2a_1+\frac{1}{4}m_2a_1^2+I_1+I_2$$

$$M_{12}=M_{21}=0$$

$$M_{22}=m_2$$

$$C(q,\dot{q})=\begin{bmatrix} \left(m_2q_2+\frac{1}{2}m_2a_1\right)\dot{q}_2 & \left(m_2q_2+\frac{1}{2}m_2a_1\right) \\ \left(-m_2q_2-\frac{1}{2}m_2a_1\right)\dot{q}_1 & 0 \end{bmatrix} \qquad (4)$$

$$G(q)=\begin{bmatrix}\frac{1}{2}m_1 g a_1\cos \mathrm{q}_1+m_2 g(\frac{1}{2}a_1+q_2)\cos \mathrm{q}_1\\ m_2 g\sin \mathrm{q}_1\end{bmatrix} \tag{5}$$

where,

q=[q$_1$q$_2$]^T, q$_1$ is the revolute joint variable and q$_2$ isthe prismatic joint variable.

The matrices *M(q)* and $C(q,\dot{q})$ govern the inertial properties of the manipulators with following structural properties.

Property-2.1. M(q) is symmetric and positive definite which is due to the fact that kinetic energy of the manipulator is zero only when the system is at rest.

Property-2.2. $\dot{M}(q)-2C(q,\dot{q})\in R^{nxn}$ is a skew-symmetric matrix which implies that in the absence of the friction, the net energy of the robot system is conserved. This property is also known as *passivity* property.

3. Active Disturbance Rejection Control (ADRC)

The control design method of ADRC [7] depends upon the input-output behavior of the plant with feedback in each sampling time of the control system making the close loop system highly sensitive to the disturbances; actively estimated by a special type of observer known as Extended State Observer (ESO) and compensated from the control input. Consequently, this task of attenuation depends on the effectiveness of the estimation carried out by an ESO. The dynamic attenuation of these estimated disturbances, results in the reduction of the plant into double integrator type, easily controlled by the PD controller. The manipulator model given in (2) can be expressed as the second order system as-

$$\ddot{q}=f(\dot{q},q,\zeta_{ext})+bu_d \tag{6}$$

where u_d is the desired control effort, '*b*' is the system parameter.

$f(q,\dot{q},\zeta_{ext})$ or $f(\cdot)$ in (6) represents the lumped or total disturbances including internal disturbances due to un-modelled dynamics, parametric uncertainties and external nonlinear, time varying, unknown disturbances ζ_{ext}. The relation of $f(\cdot)$ can be expressed as

$$f(\cdot)=-M^{-1}(q)[C(q,\dot{q})\dot{q}+G(q)] \tag{7}$$

Now, the control objective of ADRC is to estimate $f(\cdot)$ represented as $\hat{f}(\cdot)$ by a properly tuned ESO and then it is canceled from control input it in real time. Consequently, the uncertain plant starts behaving like a nominal plant after active compensation of the disturbances.

3.1. Extended State Observer (ESO)

Let $x_1=q,\ x_2=\dot{q}\ and\ x_3=f(\cdot)$ and $x=[x_1,x_2,x_3]^T$

Assuming, lumped disturbance *f* is differentiable, then (6) in state space form can be written as

$$\begin{aligned}\dot{x}&=Ax+Bu+Eh\\ y&=C\,x\end{aligned} \tag{8}$$

where, $A=\begin{bmatrix}0&1&0\\0&0&1\\0&0&0\end{bmatrix}$; $B=\begin{bmatrix}0\\b\\0\end{bmatrix}$; $E=\begin{bmatrix}0\\0\\1\end{bmatrix}$ and $C=\begin{bmatrix}1&0&0\end{bmatrix}$.

Note that $x_3=f$ is the extended state of 2nd order system and $\dot{f}=h$ *i.e.*, rate of change of uncertainty is assumed to be unknown and bounded function.

If the consecutive time derivatives of uncertainties are available, then the observer can estimate more complex disturbances. Since, on increasing the order of ESO will have a drawback of becoming more sensitive to noise and hence tougher to tune. Although there are various methods for ESO design are available [11, 12], here linear ESO (LESO) procedure [22]similar to Luenberger observer is designed as

$$\begin{aligned} \dot{\hat{x}} &= A\hat{x} + Bu + L(y - \hat{y}) \\ \hat{y} &= C\,\hat{x} \end{aligned} \tag{9}$$

where, $L = [\kappa_1\ \kappa_2\ \kappa_3]^T$ is the observer gain.

For tuning, all the observer poles are initially placed at same position in left half of s-plane denoted by ω_0 (observer bandwidth) so that the characteristic polynomial $(s + \omega_0)^3$ is Hurwitz and L = $(3\omega_0, 3\omega_0^3, \omega_0^3)^T$ is the tuning gain parameter to be selected. Usually, if observer bandwidth is too large then estimation will be more accurate but results in the increase in noise sensitivity and if it is too small then estimation will not be accurate. Therefore, a proper choice of observer bandwidth is necessary to have the good trade-off between the tracking performance and the noise filtering.

3.2. Controller Design

The ADRC controller strategy for controlling robotic manipulator arm is shown in Figure 2. The ESO parameters are tuned by EA optimization techniques rather than manual one. Eventually, the ESO starts estimating the states accurately and its outputs track the state x_1, x_2 and x_3. Consequently, the effect of lumped disturbances f is cancelled by $\hat{x}_3$ and controller will actively compensate the disturbances in real time. The effective ADRC control law is given by

$$u_d = \frac{K_p(r - \hat{x}_1) + K_d(\dot{r} - \hat{x}_2) - \hat{x}_3 + \ddot{r}}{b} \tag{10}$$

where r is the desired trajectory of the robotic arm,b is the system parameter, K_D and K_D are the controller gains, required to be tuned properly. The closed loop system for the robotic arm can be expressed as-

$$\ddot{x} = (f - \hat{x}_3) + K_p(r - \hat{x}_1) + K_D(\dot{r} - \hat{x}_2) + \ddot{r} \tag{11}$$

It is worth noting that the R.H.S. includes the combination of simple PD controller along with $(f - \hat{x}_3)$ which is almost negligible with properly designed ESO. To imitate the real engineering, the sensor noise in form of white noise with power 0.001 and external disturbance 10sin (t) have been added in the model.

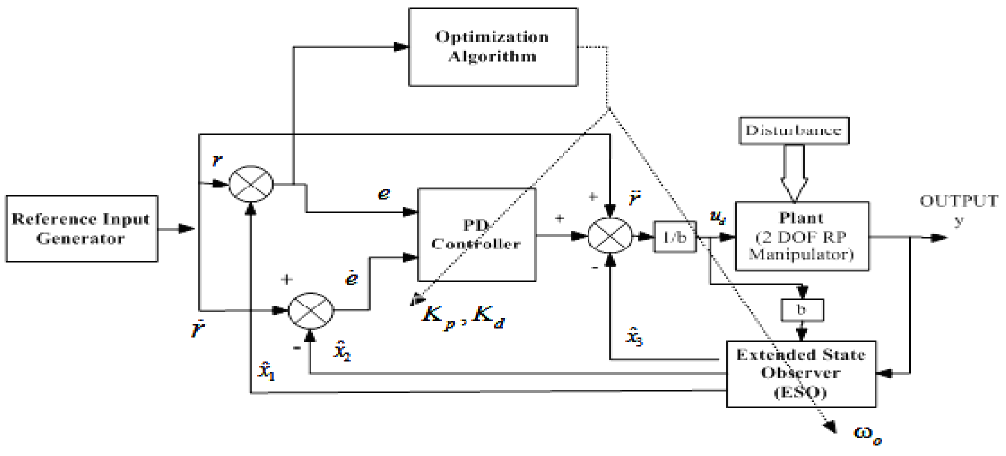


Figure 2. Modified Structure of ADRC Optimized with Optimization Algorithm

4. Overview of PSO and BFO Optimization Technique

4.1. Particle Swarm Optimization (PSO)

The PSO method is a population based stochastic search algorithm where candidate solution is referred to as particle. In the given search space, each particle move with an adaptable velocity that depends on its own as well as other particle's flying experience [23]. In PSO each particles continuously tries to follow the traits of their successful partners. Further, each particle store the best location in their memory.

The main features of PSO can be concise as follows :

- The particles (individual solutions) are randomly positioned initially which move gradually in the search space following the current best particles and come closer to the global optimum by updating the velocity and position.
- The two "best" values *i.e.*, *pbest* and *gbest* gives best solution of individual paerticle and global best solution by any particle in the population.
- The position of each particle is continuously updated by using the current velocity and position from Pbest and Gbest.
- This optimization techniques is useful for both continuous as well as discrete optimization problem sets.

The flowchart of PSO algorithm has been provided in Appendix A for reference.

4.2. Bacteria Foraging Optimization (BFO)

Bacterial foraging optimization algorithm (BFO) is widely accepted global optimization algorithm inspired by the social foraging behaviour of bacteria *E. coli*[24].Each bacteria tries to search nutrients for foraging and maximize its obtained energy per each unit of time by avoiding noxious substances. Along with this, bacteria has channel of communication among individuals. The important feature of BFO can be summarized as follows :

- Bacteria have random location in the search map of nutrients.
- Bacteria move towards nutrient rich regions where either they will die due to noxious substances or disperse due to low density of nutrients or split (reproduce) in the convenient regions.
- Best located bacteria increases their strength by attracting other bacteria, with the release of chemical attractants.
- Bacteria start searching for new location of nutrients once old location is fully utilized.
- Three main steps of bacterial foraging behaviour consists *Chemotaxis* (tumble and swimming), *reproduction* and *elimination-dispersal*.

The flowchart of PSO algorithm has been provided in Appendix A for reference.

5. Problem Formulation

The 2-DoF RP rigid robotic arm is the strongly coupled MIMO systems which require the reliable and efficient control strategy for tracking the desired trajectory. Here, this MIMO system has been split into two individual SISO systems with two independent ADRC controllers, linked together by position feedback of second arm to obtain the complete dynamics of the system. The tuning of the ADRC controllers is almost like an optimization problem, where fitness function ITAE is minimized to obtain minimum tracking error without significant output signal overshoot. The optimized controller tuning parameters are obtained by an application of the evolutionary algorithms viz. PSO and BFO algorithm on a three-dimensional solution space.

5.1. Tuning Parameters

The ADRC controller has two unknown parameters to betuned, viz. (K_p, K_d) along with the observer bandwidth (ω_0). The limit of the tuning parameter for the manipulator arm is given in Table 1. These limits are obtained by performing the manual tuning procedure provided by the Prof. Han [11].

Table 1. Range of Tuning Parameters

Tuning parameter	Lower Bound	Upper Bound
K_p	10000	360000
K_d	200	1200
ω_0	500	5000

5.2. System Design Parameters

The parameters of the manipulator used for the purpose of simulation on MATLAB Simulink platform are listed in Table 2.

Table 2. Parameters used for Simulation of 2-DOF Robotic Manipulator

Simulation Parameter	Value	Units
Mass of Link 1& 2 (m_1 and m_2)	10.0 & 8.00	Kg
Length of Link 1& 2	0.60 & 0.40	M
Distance between center of mass of Link 1 and Link 2	variable	M
Initial position of arm 1 (q_1)	0.5	
Initial position of arm 2 (q_2)	0.0	
Moment of inertia along y-axis (I_1)	12.00	kg-m^2
Moment of inertia along z-axis (I_2)	26.00	kg-m^2
Desired Trajectory for arm 1	10t	
Desired Trajectory for arm 2	1+cos(10t)	
External disturbance	10sin(t)	
Sensor noise (Band limited white noise)	Power of 0.01	

5.3 Design Objective

The objective of this paper is to obtain the three optimized ADRC controller parameters so that the system performs well by rejecting the total disturbances acting on the system and minimize the tracking error to improve the overall system stability.

The objective function used for the minimization of tracking error of end-effector (tip of second arm) is expressed as integral of the absolute magnitude of error (ITAE) defined as

$$J = \int_{0}^{\infty} t\left|q_{2d}(t) - q_2(t)\right| dt \, . \qquad (12)$$

where q_{2d} is desired profile trajectory and q_2 is actual trajectory of the end-effector.

The function 'J' is minimized by tuning of above described three parameters to fulfill the requirements of stable and good robust performance against the total disturbances acting on the system.

6. Results and Discussion

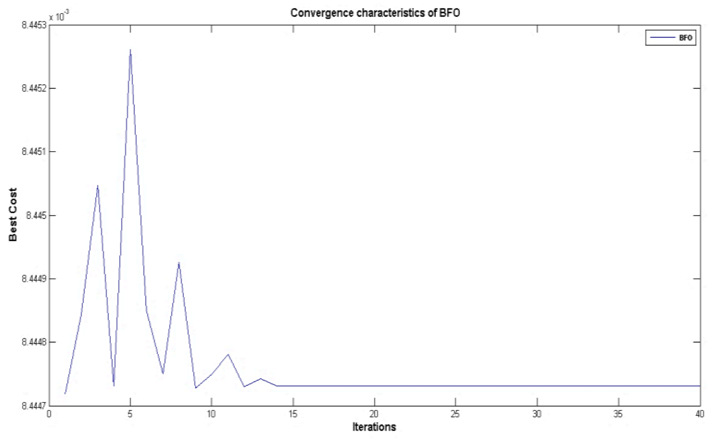
6.1. Application and Performance Comparison of BFO and PSO

The performances of the intelligent algorithm are characterized by the basic feature of optimum point convergence. The other added features like accuracy of the optimum value, and computational time distinguishes each one in terms of the simplicity of the algorithm.

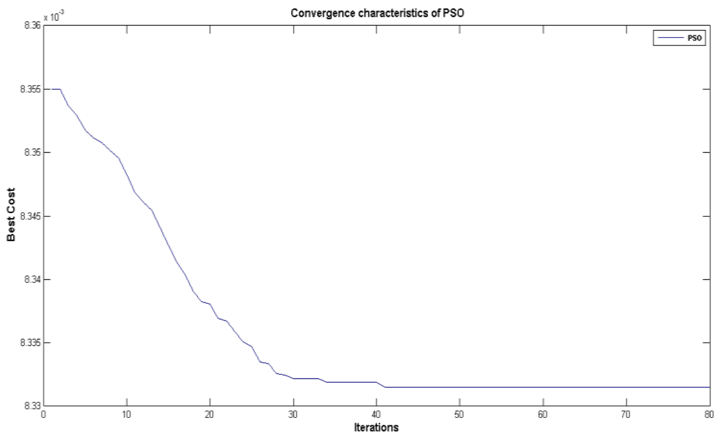
The time domain simulation of the manipulator model shown in Figure 2 is performed with each set of the selected parameter values which yields the fitness value (12). The objective function is computed by simulating the dynamic model considering the external disturbance and sensor noise effects. The performance index J reached to the finite value since the error between desired and actual position of the end-effect or becomes zero. Initially, the values of BFO and PSO parameters were selected from the literature and then adjusted to reach the final values providing best solution in terms of quality and speed. The final parameter values adopted for each of the two optimizers are listed in Table 3.

Table 3. Parameters used in PSO and BFO Optimization Algorithms

PSO	BFO
Swarm size= 10 Max. iterations=80 c_1=0.025 c_2=0.020 w=0.15	Number of Bacteria= 6 Number of chemotactic step N_c=40 Length of Swim N_s=4 Number of reproduction stepsN_{re}=4 Number of elimination-dispersal events N_{ed}=2 Probabilityof elimination /dispersal P_{ed}=0.05



(a)



(b)

Figure 3. Convergence Characteristics of (a) BFO (b) PSO

Figure 3 shows the convergence characteristics of both the techniques obtained by averaging 50 trial runs yielding the global best results of the selected performance index J. It is clear that, for this particular optimization problem, BFO converges in much lesser iterations (around 14 iterations) compared to PSO (around 40 iterations). The optimization algorithms were terminated after reaching to the pre-specified iterations selected differently for both EAs.

Table 4 gives the comparison of BFO and PSO in terms of the dynamic behavior and convergence characteristics with the help statistical indices mean (∑) and standard deviation (σ) which is given by

$$\Sigma = \frac{1}{n}\sum_{i=1}^{n} J(g_i) \tag{13}$$

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(J(g_i)-\Sigma)^2} \tag{14}$$

where $J(g_i)$ is the fitness value obtained in i^{th} iteration and n is the size of solution selected.

Table 4. Comparison of Computational Efficiency of BFO and PSO Algorithm

Algorithm	**Max**	**Min**	**Mean (∑)**	**Standard Deviation (σ)**
BFO	0.008445	0.008444	0.00844	1.6190×10^{-7}
PSO	0.008335	0.008333	0.00833	7.1676×10^{-6}

Table 4 shows that the PSO algorithm performs better in providing the fitness value than BFO algorithm.

Table 5 shows the optimized values of proposed controller tuning parameters as well as time taken for convergence by BFO and PSO.

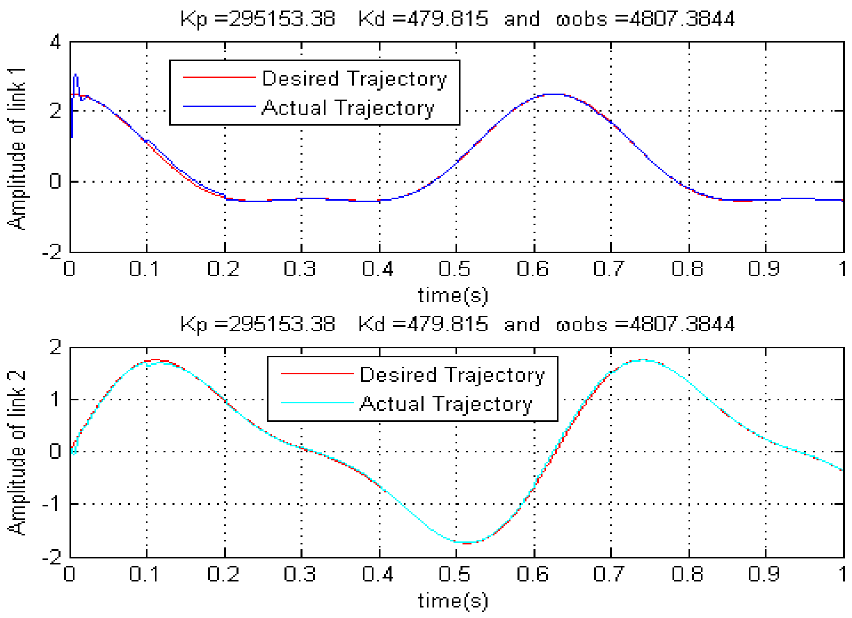
Table 5. Optimized Parameter Obtained by BFO and PSO

Algorithm	K_p	K_d	ω_0	Time Elapsed(sec)
BFO	295153.38	479.815	4807.384	20642.5
PSO	49716.58	570.991	2012.667	2286.1

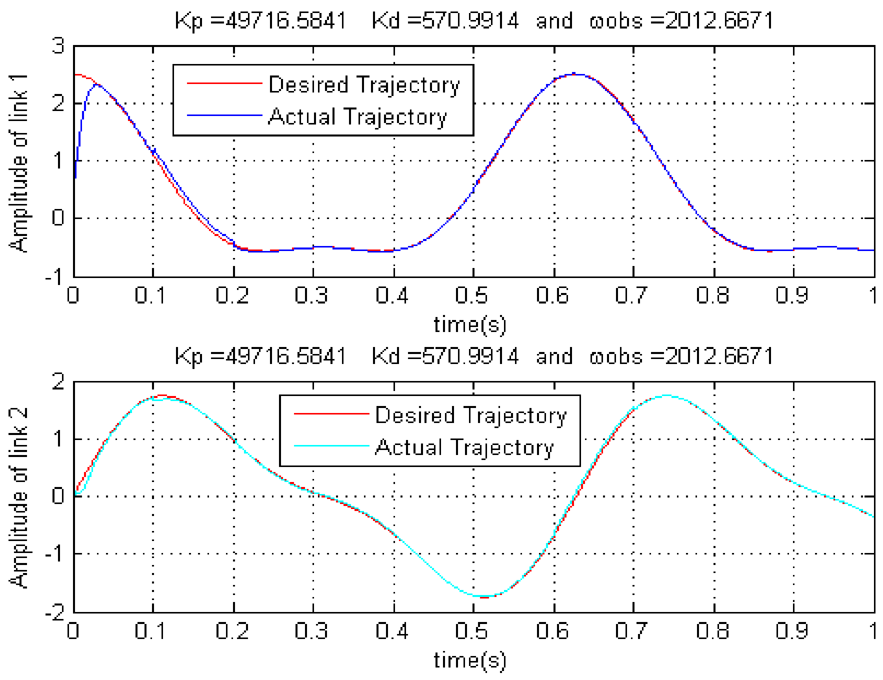
It is evident from the Table 5 that inspite of more number of iterations, computational time for PSO is comparatively much lower (1/9th times) to that of BFO optimization algorithm. This is because in BFO, bacteria require more time for reproduction and elimination-dispersal bacteria whereas in PSO, swarms are accelerated towards the solution. Also PSO algorithm has an extra advantage of tuning less number of parameter adjustment than BFO to obtain the optimized controller parameters. Further, it is analyzed that, PSO provides best value of tuning parameters than the BFO tuned parameters. It is worth noting that PSO tuned ESO bandwidth $\omega_0$is comparatively low from its counterpart which makes it more robust as higher value of ESO bandwidth increase the sensitivity to the sensor noise and other disturbances which makes tuning and hence tracking ability difficult.

6.2. Simulation Results

To evaluate the performance capabilities of the proposed BFO and PSO optimized ADRC controller, simulation is carried out on the 2 DOF planar manipulator as per the scheme shown in Figure 2. The relevant parameter considered for the purpose of simulation is listed in Table 2. The effectiveness of the proposed optimized controller is verified and compared by injecting the disturbance *d=sin (20πt)* and sensor noise in form of band limited white noise in the plant. The system responses of BFOADRC and PSOADRC for these contingencies are shown in Figure 4-7.



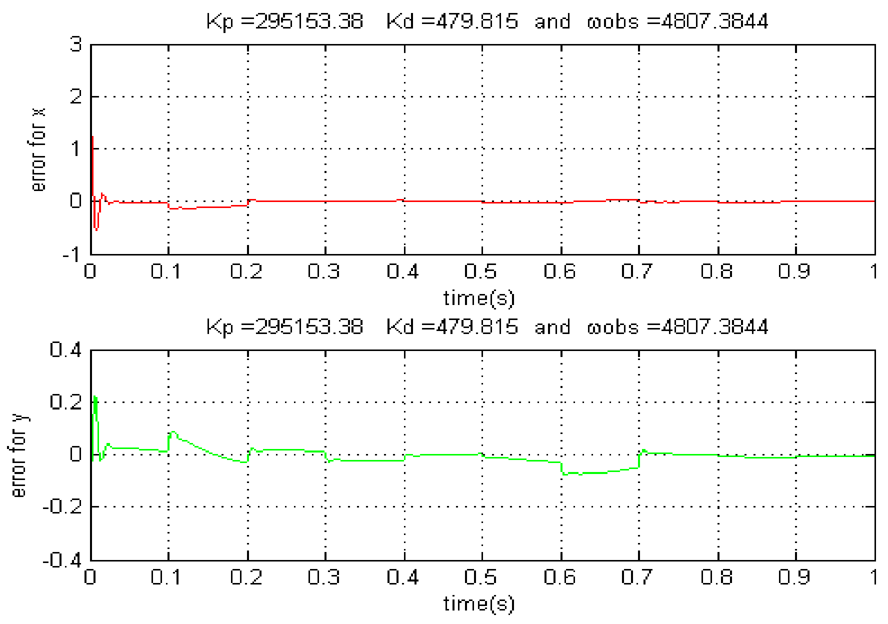
(a)



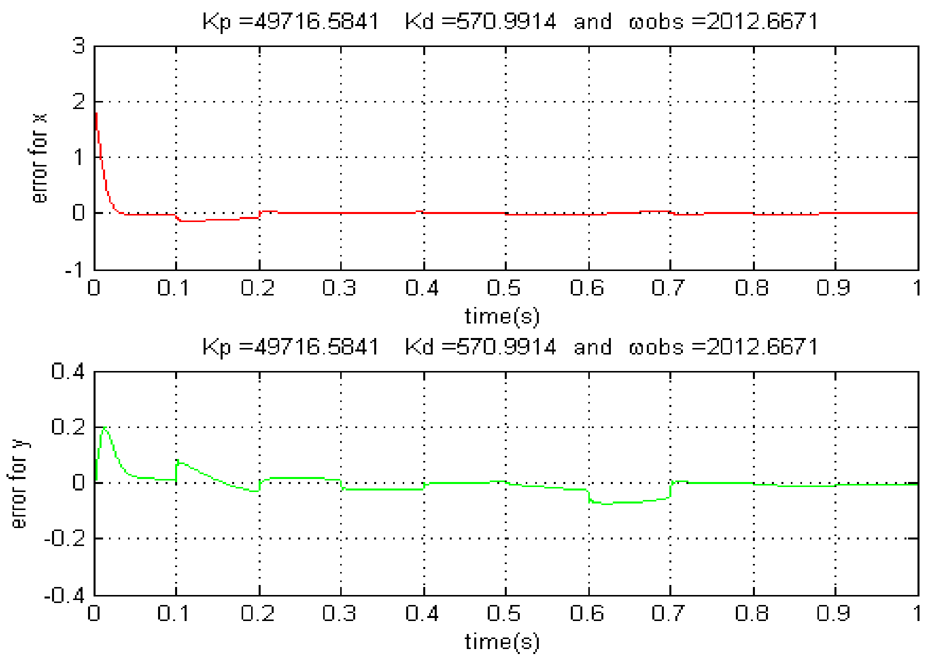
(b)

Figure 4. Position Profile for Arm 1 and Arm 2 in (a) BFOADRC (b) OADRC

Figure 4 and Figure 5 show the position and positional error profile for both the manipulator arms. It can be observed that PSOADRC performs better than BFOADRC with no overshoot and less error when time t < 0.03sec. After that both controllers provide almost the same profile due to the ESO which actively estimate and cancel the disturbance effect. It is evident by observing at t=0.07 sec in Figure 5 that the ADRC controller is highly efficient in cancellation the noise and disturbance effect at end-effector link 2.



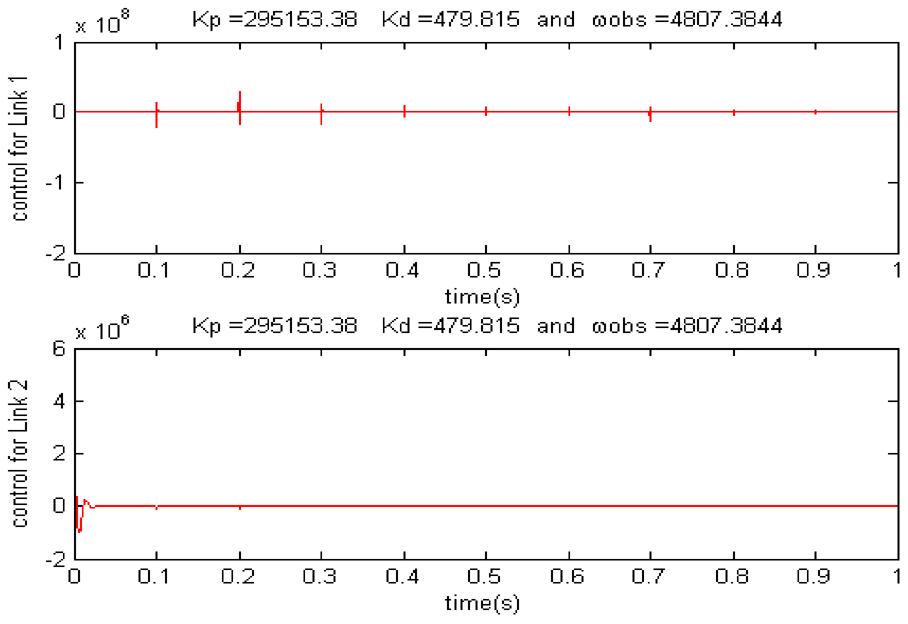
(a)



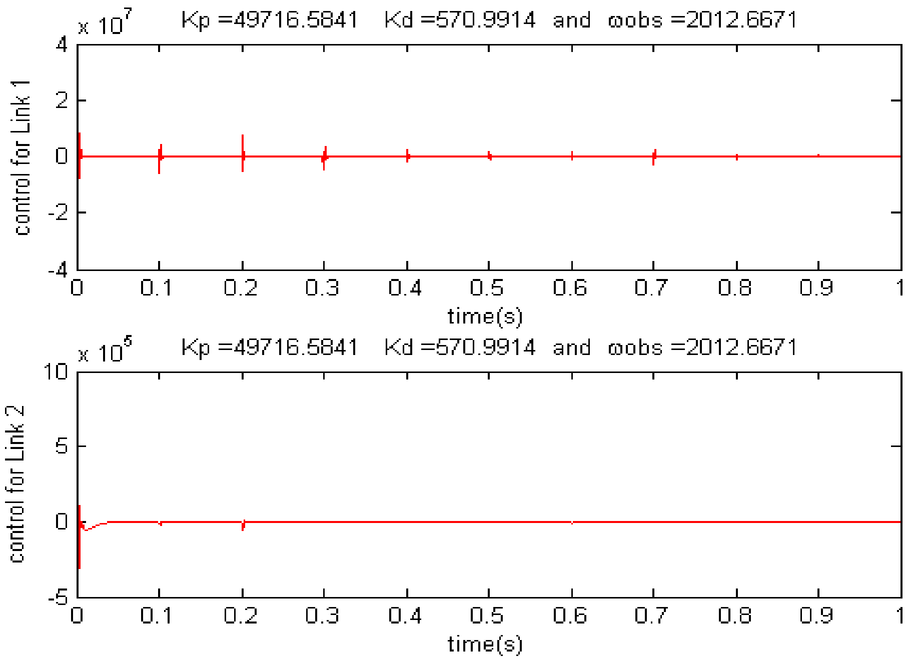
(b)

Figure 5. Error Profile for Arm 1 and 2 in (a) BFOADRC (b) PSOADRC

For more elaborate and effective analysis, the control effort experienced by the actuator on each arm is shown in Figure 6. The BFOADRC controller has to provideal most ten times higher torque than PSOADRC in both the arms which provides more wear and tear to the actuators. Hence from hardware point of view also PSO proves to be better than BFO.



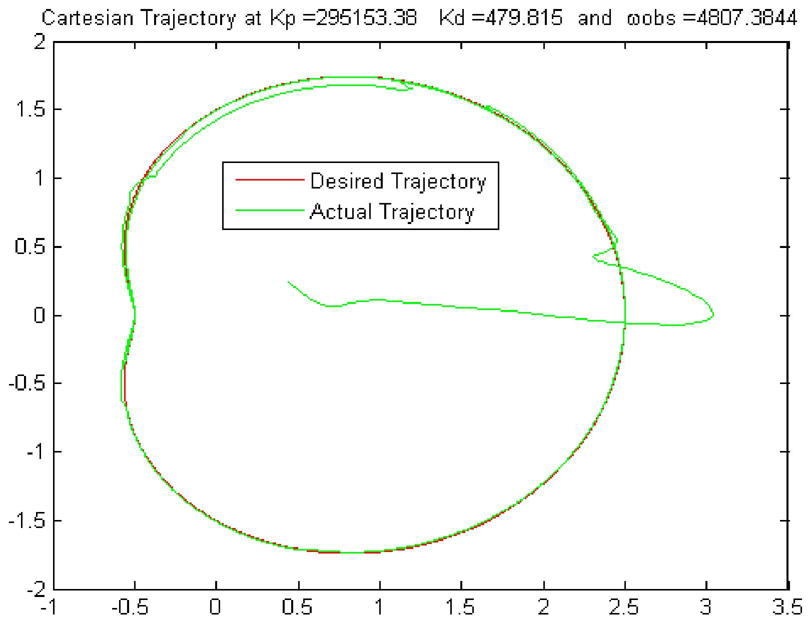
(a)



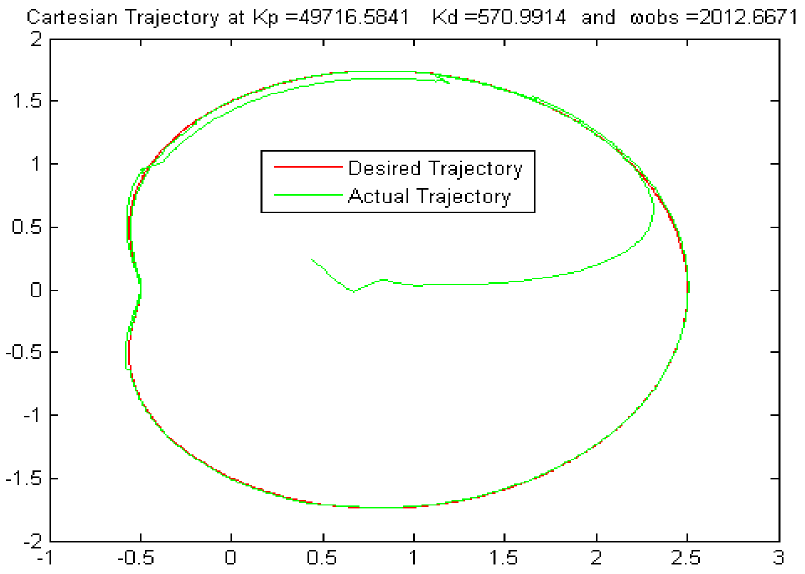
(b)

Figure 6. Control Profile for Arm 1 and Arm 2 in (a) BFOADRC (b) PSOADRC

Figure 7 presents the trajectory tracking of the manipulator end-effect or in Cartesian coordinates. It is clear from Figure 7(a) that the in comparison to BFOADRC, PSOADRC provides better trajectory tracking capabilities with fast response, more rise time and almost no overshoot in presence of the disturbance and sensor noise which is the requirement of the robust control strategy.



(a)



(b)

Figure 7. Cartesian Trajectory Profile of Roboticarmin (a) BFOADRC (b) PSOADRC

6. Conclusion

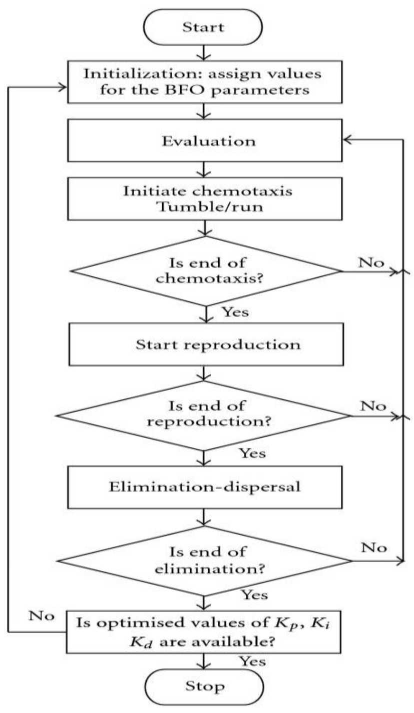
This paper presents an optimal solution technique for tuning ADRC parameters by evolutionary algorithms namely Bacteria Foraging Algorithm and Particle Swarm Optimization. The performances of these algorithms are compared in terms of processing time, and quality of solution. The two optimized ADRC controllers BFOADRC and PSOADRC are tested on the 2 DOF RP robotic manipulator arm trajectory control problem. The trajectory tracking results show that PSO algorithm outperforms the BFO algorithms for this particular problem. The BFO algorithm inspite of requiring less number of iterations to converge, it requires large processing time than PSO algorithm. Also PSO tuned ADRC parameters shows the better position tracking capabilities with high rise time and almost no overshoot in presence of disturbance and sensor noise effects. Moreover, the PSO algorithm takes appreciably less time to converge due to much simple calculation, have an added feature of having few number of initial parameters.

References

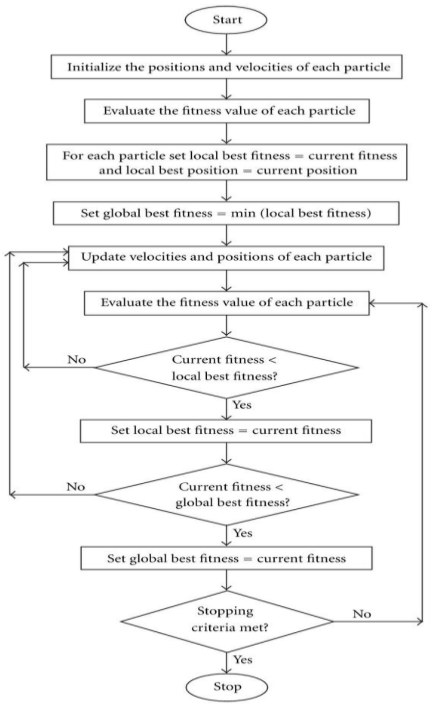
[1] J. J. Craig, P. Hsu and S. S. Sastry, "Adaptive control of mechanical manipulators", The International Journal of Robotics Research, vol. 6, **(1987)**, pp. 16-28.
[2] J.-J. E. Slotine and W. Li, "On the adaptive control of robot manipulators", The International Journal of Robotics Research, vol. 6, **(1987)**, pp. 49-59.
[3] J.-J. E. Slotine, "The robust control of robot manipulators", The International Journal of Robotics Research, vol. 4, **(1985)**, pp. 49-64.
[4] C. Abdallah, D. Dawson, P. Dorato and M. Jamshidi, "Survey of robust control for rigid robots", Control Systems, IEEE, vol. 11, **(1991)**, pp. 24-30.
[5] W. S. Levine, "The Control Handbook", CRC Press Book, **(1999)**.
[6] E. Schrijver and V. J. Dijk, "Disturbance Observers for Rigid Mechanical Systems: Equivalence, Stability, and Design", Journal of Dynamics Systems, Measurement and Control, vol. 124, **(2002)**, pp. 539- 548.
[7] J. A. Profeta, W. G. Vogt and M. H. Mickle, "Disturbance Estimation and Compensation in Linear Systems", IEEE transaction on Aerospace and Electronics Systems, vol. 26, **(1990)**, pp. 225 - 231.
[8] S. Kwon and W. K. Chung, "Combined Synthesis of State Estimator and Perturbation Observer", Journal of Dynamics Systems, Measurement, and Control, vol. 125, **(2003)**, pp. 19-26.

[9] A. Radke and Z. Gao, "A Survey of State and Disturbance Observers for Practitioners", in American Control Conference, **(2006)**.
[10] K. Youcef-Toumi and O. Ito, "A Time Delay Controller for Systems with Unknown Dynamics", Journal of Dynamics Systems, Measurement, and Control, vol. 112, **(1990)**, pp. 133-142.
[11] J. Han, "From PID to Active Disturbance Rejection Control", IEEE Transactions on Industrial Electronics, vol. 56, **(2009)**, pp. 900-906.
[12] Z. Gao, Y. Huang and J. Han, "An Alternative Paradigm for Control System Design", IEEE Conference on Decision and Control, vol. vol. 5, **(2001)**, pp. 4578-4585.
[13] Z. Gao, "Active Disturbance Rejection Control: A Paradigm Shift in Feedback Control System Design", presented at the American Control Conference, **(2006)**, pp. 2399-2405.
[14] W. Zhang and H. Xu, "Active Disturbance Rejection based Pitch Control of Variable Speed Wind Turbine", in 30th Chinese Control Conference (CCC), **(2011)**, pp. 5094-5098.
[15] Z. Qing, D. Lili, L. Dae Hui and G. Zhiqiang, "Active Disturbance Rejection Control for MEMS Gyroscopes", IEEE Transactions on Control Systems Technology, vol. 17, **(2009)**, pp. 1432-1438.
[16] P. J. Fleming and R. C. Purshouse, "Evolutionary Algorithms in Control Systems Engineering: A Survey", Control Engineering Practice, vol. 10, **(2002)**, pp. 1223-1241.
[17] D. Ashlock, "Evolutionary computation for modeling and optimization", Springer Science & Business Media, **(2006)**.
[18] D. E. Goldberg, "Genetic algorithms in search, optimization, and machine learning", Addison-Wesley, Reading, MA, **(1989)**.
[19] R. Poli, J. Kennedy and T. Blackwell, "Particle swarm optimization", Swarm intelligence, vol. 1, **(2007)**, pp. 33-57.
[20] M. W. Spong, S. Hutchinson and M. Vidyasagar, "Robot Modeling and Control", Wiley New York, vol. 3, **(2006)**.
[21] J. Kasac, B. Novakovic, D. Majetic and D. Brezak, "Global Positioning of Robot Manipulators with Mixed Revolute and Prismatic Joints", IEEE Transactions on Automatic Control, vol. 51, **(2006)**, pp. 1035-1040.
[22] Z. Gao, "Scaling and Bandwidth Parameterization based Controller Tuning", presented at the American Control Conference, **(2003)**, pp. 4989-4996.
[23] J. Kennedy and R. Eberhart, "Particle Swarm Optimization", in Proceedings of IEEE International Conference on Neural Networks, **(1995)**, pp. 1942-1948.
[24] K. M. Passino, "Biomimicry of Bacterial Foraging for Distributed Optimization and Control", IEEE Transaction on Control Systems, vol. 22, **(2002)**, pp. 52-67.

Appendix A: Flowchart for Bacteria Foraging and Particle Swarm Optimiztion



(a) BFO Optimization


Start
Initialize the positions and velocities of each particle
Evaluate the fitness value of each particle
For each particle set local best fitness = current fitness and local best position = current position
Set global best fitness = min (local best fitness)
Update velocities and positions of each particle
Evaluate the fitness value of each particle
No
Current fitness < local best fitness?
Yes
Set local best fitness = current fitness
No
Current fitness < global best fitness?
Yes
Set global best fitness = current fitness
Stopping criteria met?
No
Yes
Stop

(b) PSO Optimization

Authors

Akhilesh Swarup, He received his Ph.D. in 1993 from the Indian Institute of Technology (IIT), New Delhi, India. He is currently working as Professor in the Department of Electrical Engineering at the National Institute of Technology (N.I.T), Kurukshetra, India. He is a Senior Member of the Institute of Electrical and Electronics Engineers (IEEE). His research interests include robotics and artificial intelligence, system identification, computer networking and control systems.

Ankur Goel, He had received the degree Masters in Technology in the field of Electrical Engineering with specialization “Control Systems” from National Institute of Technology, Kurukshetra (India) in 2001 and now, he is pursuing research in the field of Optimal Motion Control from National Institute of Technology, Kurukshetra (India). His fields of interest lie in Nonlinear Control, Soft Computing, Robotics and Intelligent Control Techniques.