Int J Adv Manuf Technol (2012) 60:1223–1235
DOI 10.1007/s00170-011-3675-x

ORIGINAL ARTICLE

Bayesian regularization-based Levenberg–Marquardt neural model combined with BFOA for improving surface finish of FDM processed part

S. S. Mahapatra · Anoop Kumar Sood

Received: 18 April 2010 / Accepted: 26 September 2011 / Published online: 18 October 2011

Abstract Fused deposition modeling has a complex part building mechanism making it difficult to obtain reasonably good functional relationship between responses and process parameters. To solve this problem, present study proposes use of artificial neural network (ANN) model to determine the relationship between five input parameters such as layer thickness, orientation, raster angle, raster width, and air gap with three output responses viz., roughness in top, bottom, and side surface of the built part. Bayesian regularization is adopted for selection of optimum network architecture because of its ability to fix number of network parameters irrespective of network size. ANN model is trained using Levenberg–Marquardt algorithm, and the resulting network has good generalization capability that eliminates the chance of over fitting. Finally, bacterial foraging optimization algorithm which attempts to model the individual and group behavior of *Escherichia coli* bacteria as a distributed optimization process is used to suggest theoretical combination of parameter settings to improve overall roughness of part. This paper also investigates use of chaotic time series sequence known as logistic function and demonstrates its superiority in terms of convergence and solution quality.

Keywords Rapid prototyping · Desirability · Multi-objective optimization · Chaotic sequence · Bacteria foraging

S. S. Mahapatra (✉)
Department of Mechanical Engineering,
National Institute of Technology,
Rourkela 769008, India
e-mail: mahapatrass2003@yahoo.com

A. K. Sood
Department of Manufacturing Engineering,
National Institute of Foundry and Forge Technology,
Ranchi 834003, India
e-mail: anoopkumarsood@gmail.com

1 Introduction

Fused deposition modeling (FDM) is a rapid prototyping (RP) technology that fabricates parts by stacking and bonding layers in one direction. This method uses heated thermoplastic filaments which are extruded from the tip of nozzle in a prescribed manner on previously deposited layer to build the parts layer by layer. For general engineering applications, the FDM processed parts sometimes become unsuitable because surface finish is inevitably excessively rough, especially on the inclined surfaces of the parts. As a result, considerable efforts have been devoted by numerous researchers to improve the surface finish of the parts. In broad spectrum of RP, investigators have proposed many empirical models through experimental data analysis [1–7]. A critical analysis of literature reveals that most of the surface roughness models consider layer thickness and build orientation neglecting many other parameters involved during actual part building stage. Anitha et al. [3] and Ahn et al. [7] have pointed out that the effect of parameters other than these may not be statistically significant but certainly influence for controlling surface roughness of the part. Further, none of the proposed model is able to predict the roughness of FDM part satisfactorily. As FDM process involves a large number of process parameters which interact themselves, it leads to a complex phenomenon for part building. Therefore, it is of utmost importance to study their influence on the surface roughness so that optimum setting for minimizing the roughness can be determined [8,9]. Previous work on FDM has shown that process parameters such as layer thickness ($x^{(1)}$), part build orientation ($x^{(2)}$), raster angle ($x^{(3)}$), raster width ($x^{(4)}$), and air gap ($x^{(5)}$) significantly influence on dimensional accuracy and mechanical strength of processed part [8,9]. Therefore, the present study considers the same process

parameters as above to study their effect on surface roughness and subsequent optimization of process parameters to minimize roughness.

Generally, design-of-experiment-based approaches are adopted to develop empirical relation between input parameters and output responses through exhaustive experimentation [8,9]. Sometimes, traditional approaches become unsuitable for developing good functional relationship particularly when a process behaves in a nonlinear manner and involves large number of interacting parameters. However, neural networks can be easily applied to situations where relationship between the predictor variables (independents, inputs) and predicted variables (dependents, outputs) is very complex and cannot be easily articulated in the usual terms of correlations. Inspired by this characteristic, present study uses back propagation algorithm (BPA)-based artificial neural network (ANN) for prediction of surface roughness of top, bottom, and side faces of a FDM built part. In order to achieve faster convergence, Levenberg–Marquardt algorithm (LMA) is used for training purpose. LMA is virtually a standard approach in nonlinear optimization, and it significantly outperforms gradient decent, conjugate gradient methods, and quasi-Newton algorithms for medium-sized problems [10]. Commonly, the network architecture which gives the minimum root mean square error is selected for formulating input–output relationship. This may not guarantee the generalization ability of trained network, especially when the training data set is smaller than the number of parameters in the network. To overcome this problem, Bayesian regularization is adopted due to its capability to minimize error using minimal weights and thus avoids cross validation. It is particularly useful to problems that would suffer if a portion of the small available data is reserved to a validation set [11].

In order to optimize process parameters using ANN, numerous researchers have proposed several evolutionary techniques, notably genetic algorithm (GA) [12] and particle swarm optimization (PSO) [13]. Usually, these algorithms simulate natural genetics and impersonate the collective behavior of social insects. They have been dominating the realm of optimization algorithms and proved their efficacy in solving several engineering optimization problems. Unlike the classical evolutionary techniques, bacteria foraging optimization algorithm (BFOA) is based on the foraging theory of natural creatures that try to optimize (maximize) their energy intake per unit time spent for foraging considering all the constraints presented by their own physiology such as sensing and cognitive capabilities and environment. Problem optimization based on the principle of bacteria foraging is easier to implement because only few parameters need to be adjusted. Every bacterium remembers its own previous best value as well as the neighborhood best, and hence, it has a more effective memory capability than other techniques [14]. Also efficient diversity of the swarm is maintained as all the bacteria use the information related to the most successful bacterium in order to improve themselves [15]. In contrast, the worse solutions are discarded, and only the good ones are saved in GA [16]. Therefore, the population revolves around a subset of the best individuals in GA. PSO can be considered as a good option because of its fast convergence, but it has the tendency to get trapped at local optimum unless some procedure is adopted to escape [17]. The efficiency of BFOA in solving real parameter optimization problems has made it a potential optimization algorithm of current interest [18]. Hence, present study uses BFOA for establishing optimal combination of process parameters to improve overall roughness of the part.

2 Proposed approach

This research proposes an integrated approach for assisting the practitioners in prediction and optimization of parameters of FDM process for improving the surface finish of fabricated part. The experimental data are used for efficiently training and testing ANN model that finely maps the relationship between the control factors and output responses. The procedure of the proposed approach consists of four phases given as follows:

Phase 1 Construct the BPA-based neural network by providing training and testing data set. For faster training, Levenberg–Marquardt algorithm is used. Over fitting is prevented by using Bayesian regularization.

Phase 2 Neural network model is used for process analysis.

Phase 3 The multiple responses are converted into single response known as fitness function.

Phase 4 Fitness function is optimized using BFOA. Output at various stages of BFOA is generated by combining phase 1 and phase 3.

Figure 1 shows the flowchart of the proposed approach. The remainder of this paper is organized as follows: Section 3 describes the data collection procedure. Section 4 presents the comprehensive discussion on Bayesian regularization-based Levenberg–Marquardt neural network model. Procedures for process analysis using trained ANN model is explained in Section 5. Section 6 presents the methodology for fitness function formulation by combining all the responses into a single response. The proposed BFOA method is covered in Section 7. Results and conclusion are given in Section 8 and Section 9, respectively.

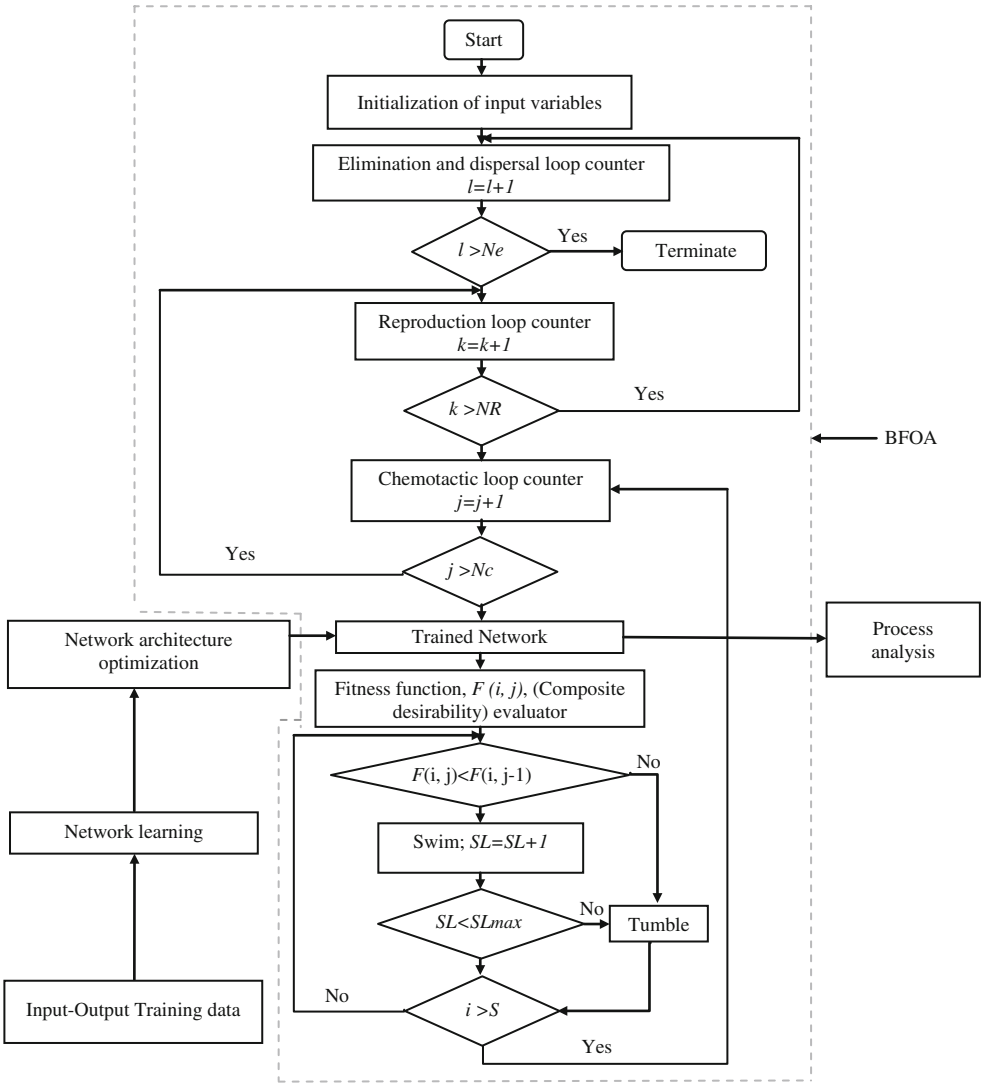

Fig. 1 Flow chart for proposed methodology

3 Data collection

Test specimens (Fig. 2) of dimension 30×10×10 mm were fabricated using FDM Vantage SE machine. For specimen fabrication, control parameters (Table 1) are set at the levels shown in Table 2 for each experimental run. For change in layer thickness, change of nozzle is required. Due to unavailability of nozzle corresponding to layer thickness value at medium level, a modified value for layer thickness is taken. The material used for specimen fabrication is acrylonitrile butadiene styrene (ABS P400). The 3D model of the specimen is modeled in CATIA V5 and exported as STL file. STL file is imported to FDM software (Insight™). Three readings of average surface roughness (R_a) on top, bottom, and left side surface of each specimen are taken along the direction shown in Fig. 2. Mean of three observations is taken as representative value of respective surface roughness. For measuring surface roughness, a contact type roughness tester Hommelwerke Turbo Wave V7.20 is used. The database is then divided into two categories such as (a) a training category, which is exclusively used to adjust the network weights, and (b) a test category, which corresponds to the set that validates the results of the training protocol.

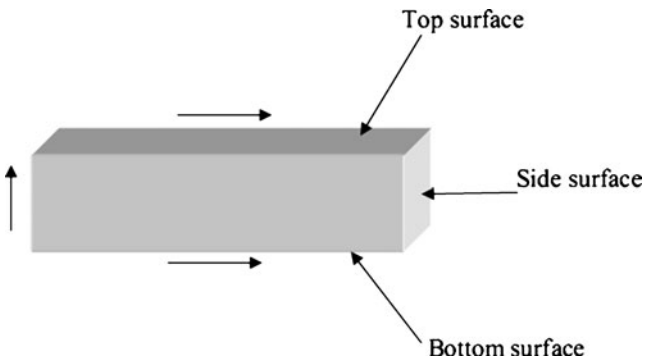

Fig. 2 Test specimen (*arrows* show direction of measurement of roughness)

Table 1 Factors and their levels

Fixed factors			Control factors					
Factor	Value	Unit	Factor	Symbol	Level			Unit
				Scaled value	Low	Medium	High	
Part fill style	Perimeter/raster	–	Layer thickness	$x^{(1)}$	0.1270	0.1780[a]	0.2540	mm
Contour width	0.4064	mm	Orientation	$x^{(2)}$	0	15	30	degree
Part interior style	Solid normal	–	Raster angle	$x^{(3)}$	0	30	60	degree
Visible surface	Normal raster	–	Raster width	$x^{(4)}$	0.4064	0.4564	0.5064	mm
XY and *Z* shrink factor	1.0038	–	Air gap	$x^{(5)}$	0.0000	0.0040	0.0080	mm
Perimeter to raster air gap	0	mm						

[a] Modified value

Table 2 Experimental data

Factors					Roughness (μm)		
$x^{(1)}$	$x^{(2)}$	$x^{(3)}$	$x^{(4)}$	$x^{(5)}$	Top	Bottom	Side
Training data							
0.127	00	00	0.4064	0.008	2.8600	2.5540	0.9034
0.254	00	00	0.4064	0.000	1.4170	5.1948	1.8068
0.127	30	00	0.4064	0.000	9.1780	11.046	0.4769
0.254	30	00	0.4064	0.008	9.8830	8.7280	0.5057
0.127	00	60	0.4064	0.000	4.9418	2.2170	0.4785
0.254	00	60	0.4064	0.008	1.9932	3.6863	0.8619
0.127	30	60	0.4064	0.008	4.2356	5.7563	0.8437
0.254	30	60	0.4064	0.000	4.8067	5.1333	0.8793
0.127	00	00	0.5064	0.000	1.1415	4.1950	0.4171
0.254	00	00	0.5064	0.008	3.9056	4.5153	0.9699
0.127	30	00	0.5064	0.008	8.8538	9.7465	0.4402
0.254	30	00	0.5064	0.000	4.6988	6.4857	0.9012
0.127	00	60	0.5064	0.008	5.0050	3.4303	0.5481
0.254	00	60	0.5064	0.000	2.1372	3.5212	0.8429
0.127	30	60	0.5064	0.000	9.4190	11.8750	0.4706
0.254	30	60	0.5064	0.008	6.8732	9.2910	0.6356
Testing data							
0.127	00	00	0.4064	0.004	3.0479	2.0896	0.4429
0.254	30	60	0.5064	0.004	3.6647	5.7613	0.5852
0.254	15	60	0.5064	0.008	3.7886	4.6521	0.3911
0.178	30	30	0.5064	0.004	5.6412	7.9106	0.2201
0.127	15	60	0.4564	0.000	4.6258	5.6897	0.4828
0.254	15	30	0.4064	0.000	2.6192	4.0807	1.0207
0.254	30	30	0.5064	0.008	5.4612	7.1316	0.5179
0.178	30	30	0.5064	0.004	5.6412	7.9106	0.2201
0.178	30	60	0.4564	0.004	5.8247	7.2527	0.3483
0.127	15	60	0.5064	0.008	4.6177	4.8593	0.4135
0.127	00	00	0.4064	0.008	2.5098	2.7093	0.6836
0.254	00	00	0.4064	0.004	2.5615	4.5355	1.2298
0.127	30	00	0.4064	0.004	5.4086	6.9185	0.4468
0.127	30	00	0.4064	0.008	4.5862	5.4032	0.5373

4 Neural network construction

A neural network consists of a number of simple, highly interconnected processing elements or nodes and is a computational algorithm that processes information by a dynamic response of its processing elements and their connections to external inputs. BPA-based neural networks are commonly used to model a nonlinear relationship between a system's input and output. The behavior of a neural network is determined by the architecture, the transfer functions, and the learning rule. To develop a back propagation neural network, the training and testing data sets are first collected. The data sets consist of both the input parameters and the resulting output parameters. Input vectors and the corresponding target vectors are used to train a network until it can approximate a function or associate input vectors with specific output vectors. Once the network is trained, testing data set of input and output vectors is used to confirm the generalized predictability of network [19].

4.1 ANN training

The training of neural network involves updating the weights of the connections in such a manner that the error between the outputs of the neural network and the actual output is minimized. The standard BPA with a fixed learning rate and momentum usually suffers from extremely slow convergence [20]. To achieve faster convergence, the learning rate of an algorithm that defines the shift of the weight vector has to be dynamically varied in accordance with the region that the weight vector currently stands. The Levenberg–Marquardt algorithm comes under the faster BPA and was designed to approach second-order training speed but without having to compute the Hessian matrix directly [21]. Hessian matrix (***H***) can be approximated as:

$$H = J^{\mathrm{T}}J \tag{1}$$

and the gradient (***g***) can be computed as:

$$g = J^{\mathrm{T}}e \tag{2}$$

where ***J*** is the Jacobian matrix that contains first derivatives of the network errors with respect to the weights and biases and ***e*** is a vector of network errors. The Jacobian matrix can be computed through a standard back propagation technique that is much less complex than computing the Hessian matrix [22]. The Levenberg–Marquardt algorithm uses approximation to the Hessian matrix in the following Newton-like update:

$$x_{k+1} = x_k - \left[J^{\mathrm{T}}J + \mu I\right]^{-1} J^{\mathrm{T}}e \tag{3}$$

When the scalar μ (Marquardt adjustment parameter) is zero, this is just Newton's method using the approximate Hessian matrix. When μ is large, this becomes gradient descent with a small step size. Newton's method is faster and more accurate near an error minimum so the aim is to shift toward Newton's method as quickly as possible. Thus, μ is decreased after each successful step (reduction in performance function) and is increased only when a tentative step would increase the performance function. In this way, the performance function is always reduced at each iteration of the algorithm.

One of the problems that occur during neural network training is called over fitting that is the error on the training set is driven to a very small value, but when new data are presented to the network the error is large. This problem can be addressed by using the Bayesian approach to control model complexity. Typically, training aims to reduce the sum of squared errors $F=E_{\mathrm{D}}$. Bayesian framework also considers the sum of squares of the network weight E_{W}, and the objective function is formulated as minimization of $F = \beta E_D + \alpha E_W$ where α, β are objective function parameters. If $\alpha \ll \beta$, then the training algorithm will drive the errors smaller. If $\alpha \gg \beta$, training will emphasize weight size reduction at the expense of network errors, thus producing a smoother network response. In this framework, the weights of the network are considered to be random variables. One can then apply statistical techniques to estimate distribution parameters, e.g., variances. According to Bayes' rule, the probability distribution can be written as:

$$P(w|D,\alpha,\beta,\eta) = \frac{P(D|w,\beta,\eta)P(w|\alpha,\eta)}{P(D|\alpha,\beta,\eta)} \tag{4}$$

where D corresponds to the input–output data set, η denotes the network model and architecture, and w are the network weights. $P(w|\alpha,\eta)$ is the prior density, which corresponds to our knowledge of the weights before any data are collected, $P(D|w,\beta,\eta)$ is the probability of the data occurring given the weights w. $P(D|\alpha,\beta,\eta)$ is the normalized factor which guarantees that the probability is 1. If the noise in the training set data is Gaussian and that the prior distribution for the weights is Gaussian, then

$$P(w|D,\alpha,\beta,\eta) = \frac{1}{Z(\alpha,\beta)} e^{-F} \tag{5}$$

Thus, minimization of F is equivalent to maximization of $P(w|D,\alpha,\beta,\eta)$ which gives the parameter values as: $\alpha = \frac{\gamma}{2E_W}$ and $\beta = \frac{n-\gamma}{2E_D}$ where γ (number of effective parameters) a measure of how many parameters in the neural network are effectively used in reducing the error function. γ is obtained from the relation $\gamma = N - 2\,\alpha\, tr(H)^{-1}$ where N is the total number of parameters. γ can range from zero to N. If γ is very close to the actual number of parameters, then the

minimum size network may not be large enough to properly represent the true function. In this case, simply add more hidden layer neurons and retrain. If the larger network has the same final γ, then the smaller network was large enough. Otherwise, more hidden layer neurons may need to be added. If the network is sufficiently large, then a second larger network will achieve comparable values for γ, E_W, and E_D [23]. For ANN training, input parameters as per LMA are shown in Table 3 and network is made to stop when maximum value of γ is reached or minimum value of performance function has met. The LMA with Bayesian regularization has been adopted in many engineering applications [24–27].

4.2 Network architecture

The neural network model consisted of an input, a hidden, and an output layer. In the present work, the input layer had five neurons which correspond to layer thickness ($x^{(1)}$), part build orientation ($x^{(2)}$), raster angle ($x^{(3)}$), raster width ($x^{(4)}$), and air gap ($x^{(5)}$), respectively. The output layer consists of three neurons, corresponding to top surface roughness, bottom surface roughness, and side surface roughness value, respectively. To determine the number of neurons in hidden layer, various network configurations in which hidden layer neurons are varied from five to the point when effective number of parameters almost become constant are used. The activation level of neuron is determined by tan-sigmoid transfer function except for output layer neurons for which linear output transfer function is used so that output is not limited to small values [28]. For training, input–output data are first mapped into the range [−1, 1]. Results of network training at various number of hidden layer neurons are shown in Table 4.

From these results, minimum size network required to properly represent the true function have seven number of neuron in the hidden layer. As noted before, the effective numbers of parameters remain approximately constant as the size of network is increased.

Table 3 LMA input parameters

Training parameter	Value
Epochs	1,000
Minimum value of performance function	10^{-9}
Minimum value of gradient	10^{-10}
Maximum validation failures	5
Marquardt adjustment parameter (μ)	0.005
Maximum value for μ	10^{10}
Decrease factor for μ	0.1
Increase factor for μ	10

5 Process analysis

Each parameter is taken at their maximum, middle, and minimum level, respectively, and other factors are maintained at their fixed levels. Responses are evaluated using ANN. The variation of each response with respect to each parameter variation is used for trend analysis. The percentage contribution of each parameter to respective response is determined as:

$$P_i^j = \frac{SS_i^j}{SS_T^j} \times 100 \tag{6}$$

where P_i^j is percentage contribution of ith factor to the jth response, SS_i^j is sum of square of ith parameter of jth response, and SS_T^j is the total sum of square. Here

$$SS_i^j = \frac{\sum_{k=1}^{n_i} \left(\overline{y}_{ik}^j - \overline{T}_j\right)^2}{n_i - 1} \tag{7}$$

where $\overline{y}_{ik}^j$ is the average response of ith parameter at kth level for jth response, $\overline{T}_j$ is the average of jth response, and n_i is the total levels of ith parameter. Adding all these sum of squares for jth response, the total sum of square (SS_T^j) is given as:

$$SS_T^j = \sum_{i=1}^{N} SS_i^j \tag{8}$$

where N is the total number of parameters.

6 Fitness function

For generating fitness function, the concept of desirability is used. Desirability represents the closeness of a response to its ideal value and lies between 0 and 1. If a response falls within the unacceptable intervals, the desirability becomes 0. The more closely the response approaches the ideal value, the closer the desirability is to 1. All the desirability can be combined to form a composite desirability which converts multi-responses into a single response [9,29]. In the present problem, it is desirable to reduce the roughness values in all the three surfaces. Therefore, "lower the better" quality characteristic is considered for measuring the desirability. The desirability can be expressed as:

$$d_{ij} = \begin{cases} 1 & \text{if} \quad y_{ij} \le low_i \\ \dfrac{High_j - y_{ij}}{High_j - low_j} & \text{if} \quad low_i \le y_{ij} \le High_i \\ 0 & \text{if} \quad y_{ij} \ge High_i \end{cases} \tag{9}$$

Table 4 Training results

Neurons	Epochs	Sum of square error	Sum of square of network weight	Total number of parameters	Effective number of parameters
5	138	0.945356	10.6488	48	32.8639
6	264	0.757393	11.3185	57	35.3729
7	169	7.00273×10^{-10}	17.0655	66	47.9996
8	134	5.23858×10^{-10}	16.5803	75	47.9996
9	217	9.46633×10^{-10}	15.9207	84	47.9996

where d_{ij} is the desirability of ith alternative of jth response, y_{ij} is the found value of ith alternative of jth response, and low_j and High_j are the minimum and the maximum values, respectively, of the experiment data for jth response. These desirability values are combined into single unit known as composite desirability [29] as:

$$C_{\mathrm{d}}^{i}=\left(\prod_{j=1}^{m} d_{ij}^{w_j}\right)^{\frac{1}{\sum_{j=1}^{m} w_j}} \tag{10}$$

where C_d^i is the composite desirability of ith alternative and w_j is the weight or importance of jth response (usually decided by the designer). For the present study, $w_j=1$, $\forall\, j$.

The fitness function is formulated as follows:

$$\text{Maximization of } F(X)=C_{\mathrm{d}}(\eta(X)) \tag{11}$$

subjected to:

$$0.127\le x^{(1)}\le 0.254$$
$$0\le x^{(2)}\le 30$$
$$0\le x^{(3)}\le 60$$
$$0.4064\le x^{(4)}\le 0.5064$$
$$0.000\le x^{(5)}\le 0.008$$

where $X=[x^{(1)}, x^{(2)}, x^{(3)}, x^{(4)}, x^{(5)}]$ are the process variables, input to neural network η, whose outputs are combined to form composite desirability C_{d}.

7 Bacteria foraging optimization algorithm

In any natural evolutionary process, survival of species depends upon their fitness criteria which rely upon their food searching and motile behavior. The law of evolution supports those species who have better food searching ability and either eliminates or reshapes those with poor search ability. This is similar to design optimization where the designer tries to find the best possible solution in the least amount of time which can be used for desired activity. Because of these similarities, Passino [30] proposed BFOA based on the foraging behavior of *Escherichia coli* bacterium. The foraging strategy of *E. coli* bacterium present in the human intestine can be explained by four processes viz., chemotaxis, swarming, reproduction, elimination, and dispersal [14,15,30].

(a) Chemotaxis: The characteristics of movement of bacteria in search of food can be defined in two ways—swimming and tumbling together known as chemotaxis. A bacterium is said to be "swimming" if it moves in a predefined direction and "tumbling" if moving in an altogether different direction. Mathematically, tumble of any bacterium can be represented by

$$\theta^{i}(j+1,k,l)=\theta^{i}(j,k,l)+C(i)\varnothing(j) \tag{12}$$

where θ^i (j,k,l) is ith bacterium position at the jth chemotactic step, kth reproduction step, and lth elimination–dispersal event. $C(i)>0$, $i=1,2,\ldots,S$ denotes a basic chemotactic step size taken in unit random direction $\varnothing(j)$. Otherwise, $C(i)$ is a basic chemotactic step size that will be used to define the lengths of steps during swimming. S is the total number of bacteria.

(b) Swarming: For the bacteria to reach at the richest food location (i.e., for the algorithm to converge at the solution point), it is desirable that optimum bacterium should try to attract other bacteria so that together they converge at the solution point more rapidly. To achieve this, a penalty function based on relative distance of each bacterium from the fittest bacterium during the search duration is added to the original cost function. Finally, when all the bacteria have merged into the solution point, penalty function assumes a value of zero. The effect of swarming is to make the bacteria congregate into groups and move as concentric patterns with high bacterial density.

Mathematically, cell-to-cell attraction and repelling effect is given by

$$\begin{aligned}F_{\mathrm{cc}}\left(\theta_{\mathrm{g}}(j,k,l),\quad \theta(j,k,l)\right)&=\sum_{i=1}^{S} F_{\mathrm{cc}}^{i}\left(\theta_{gm}(j,k,l),\quad \theta^{i}(j,k,l)\right)\\&=\sum_{i=1}^{S}\left[-d_{\text{attract}}\exp\left(-\omega_{\text{attract}}\sum_{m=1}^{P}\left(\theta_{gm}-\theta_{m}^{i}\right)^{2}\right)\right]\\&+\sum_{i=1}^{S}\left[\quad h_{\text{repellent}}\exp\left(-\omega_{\text{repellent}}\sum_{m=1}^{p}\left(\theta_{gm}-\theta_{m}^{i}\right)^{2}\right)\right]\end{aligned} \tag{13}$$

F_{cc} is called swarm attractant cost. θ_g is the position of the global optimum bacterium and m represents the mth parameter of bacterium location. $d_{attract}$ is the depth of the attractant released by the cell, $\omega_{attract}$ measures the width of the attractant signal, $h_{repellent}$ is the depth of the repellent effect, and $\omega_{repellant}$ is the measure of width of the repellent signal. Since it is not possible for two bacterium to have same location, it is assumed that $h_{repellent} = d_{attract}$.

(c) Reproduction: The original set of bacteria, after getting evolved through several chemotactic stages, reaches the reproduction stage. Here, the best set of bacteria (chosen out of all the chemotactic stages) gets divided into two groups. The healthier half replaces the other half of bacteria, which gets eliminated, owing to their poorer foraging abilities. This makes the population of bacteria constant during the evolution process. Mathematically, for reproduction, the population is sorted in terms of accumulated cost (F_{sw}) which is sum of cost function value (F) added with the swarm attractant cost (F_{cc}).

$$F_{sw}(i,j,k,l) = F(i,j,k,l) + F_{cc}\left(\theta_g(j,k,l), \theta(j,k,l)\right) \tag{14}$$

Half of the bacteria population having better cost will survive and remaining half are replaced by randomly generated new population.

(d) Elimination and dispersal: In the evolution process, a sudden unforeseen event can occur, which may drastically alter the smooth process of evolution and cause the elimination of the set of bacteria and/or disperse them to a new environment. From a broad perspective, elimination and dispersal are parts of the population level long distance motile behavior. In its application to optimization, it helps in reducing the behavior of stagnation, i.e., being trapped in a premature solution point or local optima.

Suppose that θ is the position of a bacterium and $F(\theta)$ represent the objective function with the basic goal to find the minimum. Let $P(j,k,l) = \left\{\theta^i(j,k,l) | i = 1,2\ldots..,S\right\}$ represent the position of each bacterium in the population of s bacteria at the jth chemotactic step, kth reproduction step, and lth elimination–dispersal event. Let $F(i, j, k, l)$ denote the cost at the location of the ith bacterium $\theta^i(j, k, l)$. The steps of the algorithm are presented below:

Step 1: Initialization:

1. Number of parameters (p) to be optimized
2. Number of bacteria (s) to be used for searching the total region
3. Maximum swimming length (SL_{max}) after which tumbling of bacteria will be undertaken in a chemotaxis step
4. Number of iteration (N_c) to be undertaken in a chemotaxis loop
5. Maximum number of reproduction (N_R) cycles
6. Maximum number of elimination–dispersal (N_e) events imposed on bacteria
7. Probability (P_{ed}) with which elimination–dispersal will continue
8. Location $P(p, s, 1)$ of initial set of bacteria
9. Random swim direction ($\emptyset(j)$) and step length ($C(i)$)
10. Swarming coefficients ($d_{attract}$, $w_{attract}$, $h_{repelent}$, and $w_{repelent}$)

Step 2: Elimination–dispersal loop

With some probability (P_{ed}), the existing set of bacteria gets eliminated and dispersed in a new random direction. Increment $l = l + 1$. Continue step 2 if $l < N_e$, else go to step 3.

Step 3: Reproduction loop:

1. For given k and l and for each i=1,2,…,S sort accumulated total cost (F_{sw}) in the ascending order of cost. Let F_{health}=sort $\{F_{sw}(j, k, l)\}$. Higher cost of any bacteria means poor health (objective is minimized).
2. Out of the total S bacteria, the better halves having lower F_{health} values sustain the evolution process and replace the other less healthy bacteria.

Increment $k = k+1$. Repeat step 3 if $k < N_R$, else go to step 4.

Step 4: Chemotaxis loop:

1. Calculate cost function value $F(i, j, k, l)$.
2. Find the global minimum bacteria (θ_g) from all the cost functions evaluated till that point.
3. Calculate $F_{sw}(i, j, k, l)$.
4. If $j = 1$, tumble.
5. For $j > 1$

If $F_{sw}(i, j, k, l) < F_{sw}(i, j-1, k, l)$ and $SL < SL_{max}$
Swim and increment SL=SL+1
Else,
tumble and reset SL= 0.
The next bacterium (i+1) is taken for swimming or tumbling process till i=S.

Step 5: Stop if stopping condition is met else go to step 2.

In order to ensure stability of the chemotactic dynamics in BFOA, the step-size parameter $C(i)$ must be adjusted according to the current location of the bacterium and its current fitness to improve the convergence performance as a result proposed algorithm uses adaptive step size as given by:

$$C(i) = \frac{|F^i(\theta)|}{1 + \frac{1}{|F^i(\theta)|}} \tag{15}$$

where $F^i(\theta)$=fitness of ith bacterium.

If $F^i(\theta)\rightarrow 0$, then $C(i)\rightarrow 0$ and when $F^i(\theta)\rightarrow$large, $C(i)\rightarrow 1$. This implies the bacterium which is in the vicinity of noxious substance associates with higher cost function. Hence, it takes larger steps to migrate to a place with higher nutrient concentration.

In BFOA, two types of random number generation approaches are adopted. First approach uses commonly used technique which generates a sequence of random numbers r_1, r_2, $r_3\ldots$ between 0 and $m-1$ using following relation:

$$r_{j+1} = \left(ar_j + c\right) \bmod m \tag{16}$$

where a and c are positive integers and m is the modulo.

The generated sequence is characterized by the values of a, c, m, and by the seed r_0. This series will produce values that will be repeated with a period no longer than m, starting from the seed r_0. From a practical perspective, this generator is very easy to implement. But limitation of these generators is that low-order bit is often much less random than their high-order bit [31]. In second approach of random number generation, chaotic sequences are used. Chaotic sequences are the type of random number generator whose choice is justified by their erratic and stochastic behavior. Recently, chaotic sequences have been adopted in evolutionary algorithms for random number generation [32]. Instigated from this concept, chaotic time series sequence known as logistic function is used in this work to investigate their faster convergence toward optimal solution. Logistic function is defined as:

$$C_{t+1} = R \times C_t \times (1 \quad C_t) \tag{17}$$

where C_t is the value of chaotic variable in tth iteration and R shows the bifurcation parameter of the system. Here, C_0= 0.1 and R=4.

8 Results and discussions

After training, the topology 5–7–3 is selected as the optimum. In order to evaluate the competence of this trained network, the training data set was presented to the trained network. Figure 3 shows the regression analysis between the network response and the corresponding targets. High correlation coefficient (R value) between the outputs and targets establish the performance of network.

8.1 Process analysis

For determining the variation in response with change in factor levels, a response is noted by setting a factor at its low, medium, and high levels and remaining factors fixed at low level. Figure 4 shows the factor variation results on three responses. In Fig. 4, $x(i)j$ denotes ith factor at jth level. If factor 2 is set at high level and remaining factors at low level, $x(i)j$ is written as $x(2)3$. Now, the response obtained at this setting is plotted in Fig. 4. For top surface (Fig. 4a), it is observed that roughness becomes minimum when $x^{(1)}$=0.254 mm, $x^{(2)}$=0°, $x^{(3)}$=0°, $x^{(4)}$=0.5064 mm, and $x^{(5)}$=0.008 mm for minimizing surface roughness. For bottom surface, the roughness (Fig. 4b) becomes minimum when factors are set at $x^{(1)}$=0.127 mm, $x^{(2)}$=0°, $x^{(3)}$= 30°, $x^{(4)}$=0.4564 mm, and $x^{(5)}$=0.008 mm. Similarly, surface roughness (Fig. 4c) can be minimum when factors are set at $x^{(1)}$=0.127 mm, $x^{(2)}$=30°, $x^{(3)}$=30°, $x^{(4)}$= 0.5064 mm, and $x^{(5)}$=0.000 mm for side surface. Percentage contribution of each factor to respective face surface roughness is given in Table 5. It can be concluded that raster width ($x^{(4)}$, raster angle) is the most important factor for top surface. Similarly, part orientation ($x^{(2)}$) and layer thickness ($x^{(1)}$) contribute more on the bottom surface and side surface roughness, respectively, in comparison to other factors.

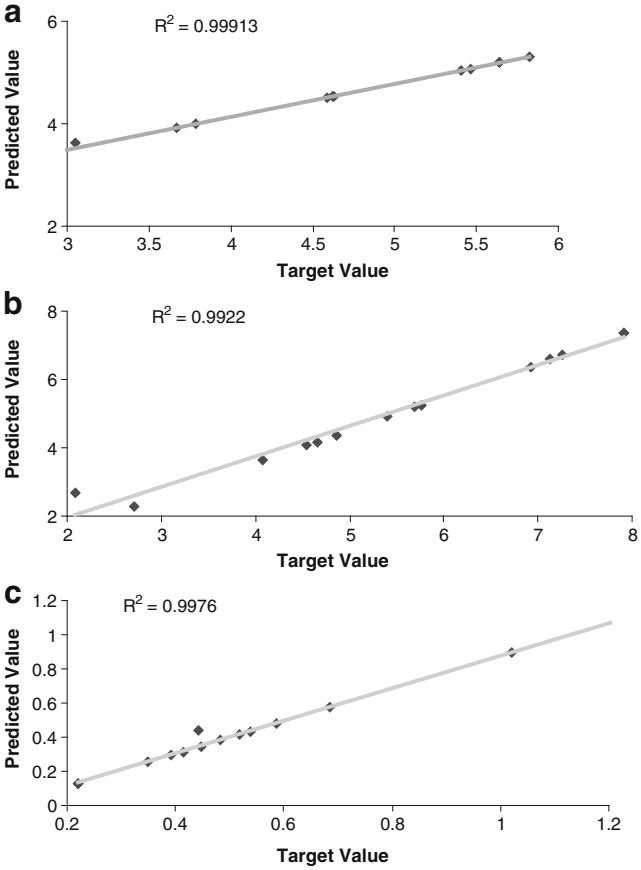

Fig. 3 Performance of neural model. **a** Top surface roughness predicted versus target value. **b** Bottom surface roughness predicted versus target value. **c** Side surface roughness predicted versus target value

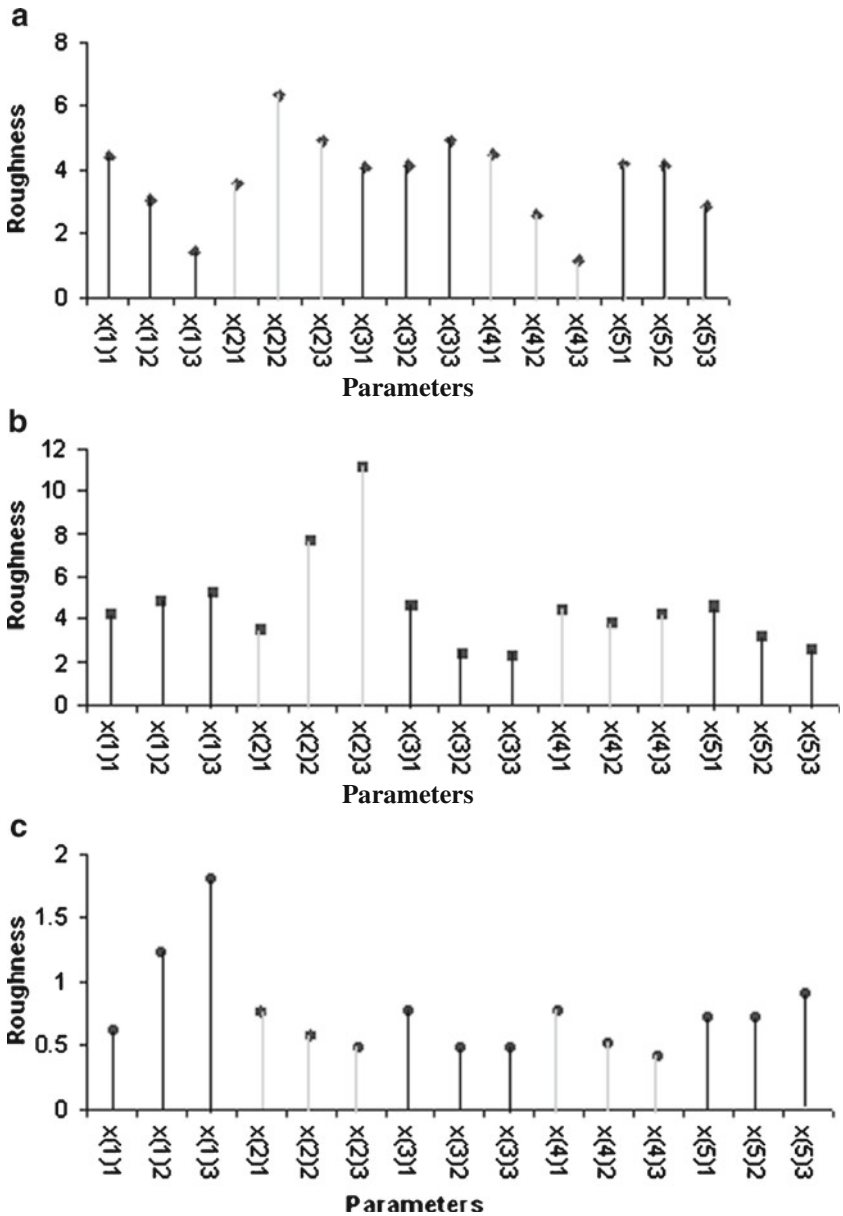

Fig. 4 Factor variation effect on roughness. **a** Top surface. **b** Bottom surface. **c** Side surface ($x(i)j$ is ith factor at jth level)

Table 5 Percentage contribution

Parameter	TOP surface		Bottom surface		Side surface	
	Sum of square	Percentage contribution	Sum of square	Percentage contribution	Sum of square	Percentage contribution
$x^{(1)}$	4.1873	29.9119	0.5102	1.4411	0.7138	78.3751
$x^{(2)}$	3.0949	22.1082	28.38	80.162	0.0454	4.98272
$x^{(3)}$	0.3589	2.56398	4.0961	11.57	0.0634	6.96574
$x^{(4)}$	5.5864	39.9063	0.1442	0.4072	0.0729	7.9996
$x^{(5)}$	0.7713	5.50969	2.273	6.4201	0.0153	1.67679
Total	13.999		35.404		0.9108	

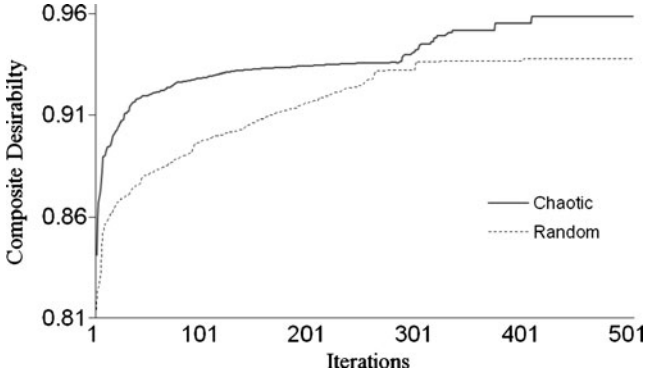

Fig. 5 Convergence curve for proposed algorithm

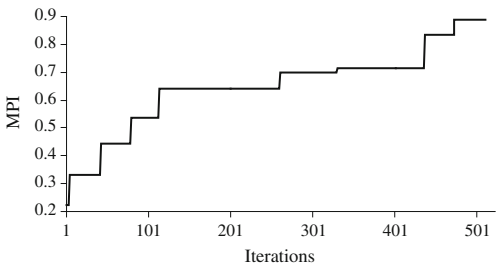

Fig. 6 Convergence curve for MPI

8.2 Parameter optimization

Process parameter values for maximization of C_d are found by bacteria foraging method. Parameters of swarming such as $d_{attract}$, $w_{attract}$, $h_{repelent}$, and $w_{repelent}$ are set at 1.9, 0.2, 1.9, and 10, respectively. In order to apply BFOA, initial parameters are set as: number of bacteria S=10, chemotactic loop limit N_c=20, maximum swim length SL_{max}=250, reproduction loop limit N_R=10, elimination–dispersal loop limit N_e=20, and probability of elimination dispersal P_{ed}= 0.001. For calculation of composite desirability (fitness function), high and low values for each response are taken from experimental data (Table 2). The high values of surface roughness are 19, 39, and 1 μm for top, bottom, and side surfaces, respectively. Similarly, low values of surface roughness are 2, 2, and 0.2 μm for top, bottom, and side surfaces, respectively. The algorithm is coded in MATLAB and run on HP Compaq dc7600 Pentium IV desktop computer.

The BFOA algorithm is run for 500 iterations, and results are shown in Fig. 5. It can be observed that chaotic function not only has better convergence but also gives the good result as compared to that based on random number function. The fitness function value of 0.9567 is obtained in 408 iterations if chaotic time series sequence (logistic function) is adopted for random number generation. The fitness value approaches 0.9357 in 401 iterations if congruential random generator is adopted. The final optimum results under the assumptions considered in the present work are given in Table 6.

Sood et al. have optimized process parameters for improving dimensional accuracy and mechanical strength of FDM built parts [8,9]. Improvement of surface finish of FDM built parts using BFOA has been reported in [33]. However, the approach uses weighted principal component for converting multiple responses into a single response known as multi-response performance index (MPI), input–output relations are established through regression analysis, and process parameters are optimized via BFOA. Employing the method proposed in [33] to the present data set, the regression equation is developed as:

$$\begin{aligned}\mathrm{MPI} = {} & 0.847 + 0.0532x^{(1)} - 0.237x^{(2)} + 0.0321x^{(3)} - 0.0383x^{(4)} + 0.0071x^{(5)} + 0.145\left(x^{(1)}\right)^2 \\ & -0.144\left(x^{(2)}\right)^2 - 0.110\left(x^{(3)}\right)^2 - 0.042\left(x^{(4)}\right)^2 - 0.161\left(x^{(5)}\right)^2 + 0.0325x^{(1)}x^{(2)} + 0.0230x^{(1)}x^{(3)} \\ & +0.0592\,x^{(1)}x^{(4)} - 0.107\,x^{(1)}x^{(5)} + 0.0304\,x^{(2)}x^{(3)} - 0.0291\,x^{(2)}x^{(4)} + 0.0250x^{(2)}x^{(5)} \\ & -0.0755\,x^{(3)}x^{(4)} + 0.0346x^{(3)}x^{(5)} - 0.0492\,x^{(5)}x^{(4)}\end{aligned} \tag{18}$$

Table 6 BFOA results

Random number							Chaotic sequence						
Fitness function	Iteration	$x^{(1)}$ (mm)	$x^{(2)}$ (deg)	$x^{(3)}$ (deg)	$x^{(4)}$ (mm)	$x^{(5)}$ (mm)	Fitness function	Iteration	$x^{(1)}$ (mm)	$x^{(2)}$ (deg)	$x^{(3)}$ (deg)	$x^{(4)}$ (mm)	$x^{(5)}$ (mm)
0.9357	401	0.1296	2.3955	17.6334	0.4894	0.0045	0.9567	408	0.1402	12.9290	15.9366	0.4170	0.0073
Roughness (μm)							Roughness (μm)						
Top surface				3.6049			Top surface				3.4861		
Bottom surface				3.9852			Bottom surface				3.9649		
Side surface				0.2351			Side surface				0.1892		

The coefficient of determination which indicates the percentage of total variation in the response explained by the terms in the model is 94.3%. An Anderson–Darling normality test result signifies that residuals follow normal distribution and the model given by Eq. 18 is suitable for practical engineering applications. Figure 6 gives the convergence curve for optimal MPI using BFOA proposed in present work.

The optimal parameter setting found out is layer thickness=0.178 mm, part build orientation=29.06°, raster angle=5.4°, raster width=0.5018 mm, and air gap=0.00057. MPI value at this parameter setting is 0.8901. The surface roughness of top, bottom, and side surfaces at this level of parameter is 4.3362, 3.5660, and 0.6695 μm, respectively. Comparison of these values with values predicted using proposed methodology (Table 6), it is observed that modeling using ANN approach gives better prediction as compared to statistical approach proposed in [33]. When functional relations are established through neural network as in this study, the optimal parameter settings are layer thickness=0.1402 mm, part build orientation=12.9290°, raster angle=15.9366°, raster width=0.4170 mm, and air gap=0.0073 mm. The surface roughness values at top, bottom, and side surfaces are 3.4861, 3.9649, and 0.1892 μm. The resulting process parameters seem to be convenient to set in the machine, and overall surface roughness is uniform in top and bottom surfaces and side surface becomes much smoother. The major advantage of the present work rests on use of neural network having high-order generalization capability to develop functional relationship among inputs and outputs in a complex situation to avoid shortcomings of empirical relations.

9 Conclusions

In this work, functional relationship between process parameters and three responses (top, bottom, and side surface roughness) for FDM built parts has been developed using BPA-based ANN methodology. The process parameters considered are layer thickness ($x^{(1)}$), part build orientation ($x^{(2)}$), raster angle ($x^{(3)}$), raster width ($x^{(4)}$), and air gap ($x^{(5)}$). For faster training of ANN, Levenberg–Marquardt algorithm was used. Advantage of using this algorithm is due to the fact that it approaches second-order training speed without having to compute the Hessian matrix. The common method of improving network generalization is to use a network that is just large enough to provide an adequate fit. Unfortunately, it is difficult to know beforehand how large a network should be for a specific application. To overcome this problem, Bayesian regularization which provides better generalization performance was adopted. Process analysis carried out based on prediction values of neural network shows the existence of nonlinear relationship between output and process parameters. Raster width ($x^{(4)}$) is found to be most important parameter for improving surface finish at the top surface. Similarly, part orientation ($x^{(2)}$) and layer thickness ($x^{(1)}$) are significant for reduction of surface roughness at bottom and side surfaces, respectively. Although variation in these parameters affects the surface topology but contribution of other factors, no matter how minor they are, may also have deciding effect on surface topology, hence cannot be neglected. A latest evolutionary approach known as bacterial foraging has been used to predict the optimal parameter settings. It predicts parameters in universal range of values that may not exist in the experimental system setup. Therefore, this technique suggests alternative system settings so as to produce optimum results in the process. Further, it is also demonstrated that chaotic sequence-based random number generation is a better alternative than the conventional random number generation procedure as far as convergence characteristic of the algorithm is considered. Methodology adopted here is quite general and can be relevant to other problems of function approximation and optimization, especially when multi-input–multi-output system is considered.

Acknowledgments The authors express hearty thanks to the Editor-in-Chief of International Journal of advanced Manufacturing Technology and learned reviewers for their useful suggestions that helped to improve the literal and technical content of the paper.

References

1. Luis Pérez CJ, Calvet JV, Sebastián Pérez MA (2001) Geometric roughness analysis in solid free form manufacturing process. J Mater Process Technol 119:52–57
2. Voorakarnam V, Paul KB (2001) Effect of layer thickness and orientation angle on surface roughness in laminated object manufacturing. J Manuf Process 3(2):94–101
3. Anitha R, Arunachalam S, Radhakrishnan P (2001) Critical parameters influencing the qualities of prototype in fused deposition modelling. J Mater Process Technol 118:385–388
4. Campbell RI, Martorelli M, Lee HS (2002) Surface roughness visualization for rapid prototype models. Comput Aided Des 34:717–725
5. Thrimurthulu K, Pandey PM, Reddy NV (2004) Optimum part deposition orientation in fused deposition modeling. Int J Mach Tools Manuf 44:585–594
6. Kim HC, Lee SH (2005) Reduction of post processing for stereolithography systems by fabrication direction optimization. J Mech Sci Technol 37:711–725
7. Ahn D, Kweon JH, Kwon S, Song J, Lee S (2009) Representation of surface roughness in fused deposition modelling. J Mater Process Technol 209(15–16):5593–5600
8. Sood AK, Ohdar RK, Mahapatra SS (2009) Improving dimensional accuracy of fused deposition modelling process using grey Taguchi method. Mater Des 30(10):4243–4252

9. Sood AK, Ohdar RK, Mahapatra SS (2010) Parametric appraisal of mechanical property of fused deposition modelling processed parts. Mater Des 31(1):287–295
10. Torrecilla JS, Otero L, Sanz PD (2007) Optimization of an artificial neural network for thermal/pressure food processing: evaluation of training algorithms. Comput Electron Agric 56 (2):101–110
11. Fadare DA, Ofidhe UI (2009) Artificial neural network model for prediction of friction factor in pipe flow. J Appl Sci Res 5(6):662–670
12. Debabrata M, Pal Surjya K, Partha S (2007) Modeling of electrical discharge machining process using back propagation neural network and multi-objective optimization using non-dominating sorting genetic algorithm-II. J Mater Process Technol 186:154–162
13. Chun-Yao L, Yi-Xing S, Jung-Cheng C, Yi-Yin L, Chih-Wen C (2009) Neural networks and particle swarm optimization based MPPT for small wind power generator. World Acad Sci Eng Technol 60:17–23
14. Liu Y, Passino KM, Simaan MA (2002) Biomimicry of social foraging bacteria for distributed optimization: models, principles, and emergent behaviors. J Optim Theory Appl 115(3):603–628
15. Mahapatra SS, Kumar PS, Saumyakant P, Kumar SA (2009) Optimization of fused deposition modelling (FDM) process parameters using bacterial foraging technique. Intell Inf Manag 1:89–97
16. Goldberg DE (1989) Genetic algorithm in search, optimization and machine learning. Addison-Wesley Longman, Boston
17. Biswas S, Mahapatra SS (2009) An improved metaheuristic approach for solving the machine loading problem in flexible manufacturing systems. Int J Serv Oper Manag 5(1):76–93
18. Swagatam D, Sambarta D, Arijit B, Ajith A, Amit K (2009) On stability of the chemotactic dynamics in bacterial foraging optimization algorithm. IEEE Trans Syst Man Cyber Part A Syst Hum 39(3):670–679
19. Min FL (1994) Neural network in computer intelligence. McGraw-Hill, New Delhi
20. Chidrawar K, Bhaskarwar S, Sujata M, Balasaheb P (2009) Implementation of neural network for generalized predictive control: a comparison between a Newton Raphson and Levenberg Marquardt implementation. 2009 World Congress on Comp Sci Inform England, pp 669–673
21. Hirschen K, Schafer M (2006) Bayesian regularization neural networks for optimizing fluid flow processes. Comp Methods Appl Mech Eng 195:481–500
22. Martin HT, Menhaj MB (1994) Training feed forward networks with the Marquardt algorithm. IEEE Trans Neural Netw 5(6):989–993
23. Foresee FD, Hagan MT (1997) Gauss–Newton approximation to Bayesian learning. IEEE Trans Neural Netw 3:1930–1935
24. Furferi R, Governi L (2008) The recycling of wool clothes: an artificial neural network colour classification tool. Int J Adv Manuf Tech 37(7–8):722–731
25. Basterrech S, Mohammed S, Rubino G, Soliman M (2011) Levenberg–Marquardt training algorithms for random neural networks. The Comput J 54(1):125–135
26. Tambourgi EB, Fischer GA, Fileti AMF (2006) Neural modeling for cytochrome b5 extraction. Proc Biochem 41(6):1272–1276
27. Ciurana J, Arias G, Ozel T (2009) Neural network modeling and particle swarm optimization (PSO) of process parameters in pulsed laser micromachining of hardened AISI H13 steel. Mater Manuf Proc 24(3):358–368
28. Howard D, Martin H, Beale M (2010) Neural network Toolbox™ user's guide. The Maths Works, Natick
29. Montgomery DC (2003) Design and analysis of experiments, 5th edn. Wiley, Singapore
30. Passino KM (2002) Biomimicry of bacterial foraging for distributed optimization and control. IEEE Control Sys Mag 22:52–67
31. Riccardo C, Fortuna L, Stefano F, Gabriella XM (2003) Chaotic sequences to improve the performance of evolutionary algorithms. IEEE Trans Evol Comput 7(3):289–304
32. Nitesh K, Anoop P, Ravi S, Tiwari MK (2008) Fast clonal algorithm. Eng Appl Artif Intell 21:106–128
33. Sood AK, Mahapatra SS, Ohdar RK (2011) Weighted principal component approach for improving surface finish of ABS plastic parts built through fused deposition modelling process. Int J Rapid Manuf 2(1/2):4–27