



## A Reliable Approach for Terminating the GA Optimization Method

L. LotfiKatooli and A. Shahsavand\*

### Abstract

Genetic algorithm (GA) has been extensively used in recent decades to solve many optimization problems in various fields of science and engineering. In most cases, the number of iterations is the only criterion which is used to stop the GA. In practice, this criterion will lead to prolong execution times to ensure proper solution. A novel approach is presented in this article as the approximate number of decisive iterations (*ANDI*) which can be used to successfully terminate the GA optimization method with minimum execution time. Two simple correlations are presented which relate the new parameter (*ANDI*) with approximate degrees of freedom (*Adf*) of the merit function at hand. For complex merit functions, a linear smoother (such as Regularization network) can be used to estimate the required *Adf*. Four illustrative case studies are used to successfully validate the proposed approach by effectively finding the optimum point by using to the presented correlation. The linear correlation is more preferable because it is much simpler to use and the horizontal axis represents the approximate (not exact) degrees of freedom. It was also clearly shown that the Regularization Networks can successfully filter out the noise and mimic the true hyper-surface underlying a bunch of noisy data set.

**Keywords:** Genetic algorithm; Termination criterion; Approximate degrees of freedom; Approximate number of decisive iteration; Linear smoother concept; Regularization Networks.

---

\*Corresponding author

Received 26 July 2015; revised 21 November 2015; accepted 28 September 2016

L. LotfiKatooli

Department of Chemical Engineering, Faculty of Engineering, Ferdowsi University of Mashhad, Mashhad, Iran. e-mail: Leyla.Lotfi@stu.um.ac.ir

A. Shahsavand

Department of Chemical Engineering, Faculty of Engineering, Ferdowsi University of Mashhad, Mashhad, Iran. e-mail: shahsavand@um.ac.ir

## 1 Introduction

Recently, there has been great interest in improving both operators and criteria of Global Optimization techniques [19, 34]. Genetic algorithm (GA) which is probably the most famous Global optimization technique, still suffers from not having a reliable automatic termination criterion. In other words, GA is usually unable to decide when or where it should terminate the optimization computations [16]. In practice, some pre-specified maximum number of generations is usually used as the termination criterion. In many practical applications, the stopping criteria can significantly influence both the final optimal solution and the overall duration of the entire optimization process [17].

Genetic algorithm is originally rooted in Hollands work [15] which presented a new approach in early sixties for problem solving that finally became widely popular. Afterwards, in late 60s, his students, Bagley [3], Rosenberg [29], and Cavicchio [6] paved the way for new generations. At the present, GA has found extensive applications in many fields of science, economy and engineering.

Many engineering design, modeling and planning problems are inherently quite complex and usually they can't be solved via conventional optimization methods. On the other hand, GA can overcome these issues and is able to successfully solve optimization problems with complex merit functions. GA has simple operators, low storage requirements, and its parallel and global outlook has been applied successfully in a wide variety of problem domains [10, 22], such as control system design [9], chemical kinetics [8] and robotic manipulator design [4].

All steps of GA are usually well defined except the number of stopping criterion iterations [5]. Several termination criteria, such as Generation Number, Evolution Time, Fitness Threshold, Fitness Convergence, Population Convergence and Gene Convergence are introduced to stop the evolutionary algorithms like GA [32]. However, the most commonly used stopping criterion for GA is Generation number which is usually determined via trial and error.

In 1992, Nix and Vose modeled a simple genetic algorithm as Markov chain [24]. They have not preserved the knowledge of previous best in their model. In the mid-90s, Rudolph [30] and Suzuki [33] preserved the knowledge of the previous best chromosomes in their model and proved faster convergence of GAs to the optimal string. In 1994, Meyer and Feng [21] developed the concept of a fuzzy stop criterion to provide a useful evaluation of GA's real-time performance. Their method is not based on a global or fixed value but it works on achieving a pre-specified user-defined level of performance for the given problem. Previously computed performances of earlier GAs were used as a frame of reference for estimation of the current GA performance.

In 1996 Aytug and Koehler [2] derived a theoretical bound on the number of iterations. They showed that a level of confidence is required to guarantee

that the GA can find an optimal solution. They reported that the bound on GA running time is a function of the mutation rate, size of the population strings and the population size but it is independent of the crossover rate and the fitness function. They used Markov chain formulation to find the bound. In a similar work, Aytug et al. [1] have also modeled run time behavior of GA using cardinality representation, as Markov chain.

In 1998 Murthy et al. [23] introduced concept of  $\varepsilon$ -optimal stopping time of Genetic algorithm. They showed that the probability of performing mutation play an important role in estimation of optimal stopping time of GA. They have provided an estimate for the number of iterations that a GA has to run to obtain an  $\varepsilon$ -optimal global solution. In 2002 Greenhalgh and Marshall [11] examined how long a genetic algorithm should run to ascertain that it can reach a reasonable optimal solution. They solved this problem by looking at the effect of mutation on the convergence of genetic algorithm.

In 2007 Hedar et al. [14] modified a genetic algorithm with new termination criteria and acceleration elements. They named the proposed method "Genetic Algorithm with Automatic Accelerated Termination (G3AT)". This method is based on some directing strategies: a) The Gene Matrix (GM) that assist the exploration process, b) The Mutagenesis a new type of mutation that accelerate the exploration and exploitation processes and c) The Memetic Algorithm in order to achieve faster convergence.

In 2011 Ong and Fukushima [25] developed another genetic algorithm that was called Genetic Algorithm with Automatic Termination and Search Space Rotation (GATR). This algorithm can terminate the search without conventional predefined criteria such as the maximum number of generations or Function evaluations. They attempted to improve performance of Hedar's work [14] more accurately and more versatile.

In 2012 Bhandari et al. [5] introduced a new stop criterion based on variance ( $\varepsilon$ ) for the best optimal solutions of the problem at hand. The proposed criterion uses only the fitness function values and takes into account the inherent properties of the objective function. They didnt suggest any automatic procedure for choosing the value of  $\varepsilon$ . Instead, the  $\varepsilon$  criterion should be chosen sufficiently small to ensure the required accuracy.

In the present article, a new stopping criterion based on the new concept of approximate number of decisive iterations (*ANDI*) and approximate degrees of freedom (*Adf*) of the merit function at hand is proposed. For a previously defined and explicitly known cost function (which are used as typical examples in this work), the maximum value of absolute error ( $MAE_y$ ) which is readily computable from the uncertainties of the input variables, can be used as a simple criterion for selecting the proper value of (*ANDI*).

In real optimizations, where the ordinary least square is used as the merit function, the above approach is extremely time consuming. For these cases, a convenient procedure is introduced to find the appropriate value of (*ANDI*) using the approximate degrees of freedom (*Adf*) for the least square objective function under consideration. A certain class of artificial neural network will

be used to compute the  $Adf$  for both discrete and continuous cost functions. It is clearly shown that a straightforward correlation exists between  $ANDI$  and  $Adf$  which can be used to estimate the GA termination criterion for any global optimization problem.

In the next sections, initially our in-house algorithm for GA,  $ANDI$  and  $Adf$  will be presented briefly. Afterwards, the so called "Regularization network" of Poggio and Girosi [26] is introduced as a vehicle to compute the approximate degrees of freedom for any desired cost function. Several simple and relatively complex case studies (merit functions) are used to propose a new correlation between our so called " $ANDI$ " criterion and the " $Adf$ " of the objective function at hand.

## 2 A brief overview of our in-house GA algorithm

Genetic algorithm is based on the survival of better solutions evolved from previous generations until a near optimal solution is acquired. It uses the main three operations of selection, crossover and mutation to produce new generations from the old ones [7].

The initial population is formed of  $N$  chromosomes that covers search domain and is usually generated in a random manner. Selection operation is performed by considering individual and cumulative probabilities, at this step stronger and better chromosomes have a greater chance of being selected. The Crossover operator works by selecting two chromosomes at random that will exchange genetic features [20]. In nature, mutation rarely happens but in a conventional GA, it happens frequently producing a new generation in each iteration.

A modified version of GA is presented here which uses a new termination criterion parameter known as  $ANDI$ . Figure 1 depicts our in-house GA algorithm which combines the standard GA with  $ANDI$  concept.

As shown in Figure 1, all conventional steps of a standard GA such as initialization, parent selection, cross over and mutation are used in our in-house algorithm. To ensure preserving the best solution throughout the entire optimization calculations, the best fitness is saved in each iteration and this chromosome is expected to bypass any GA operation (crossover or mutation). As Figure 1 depicts, our in-house algorithm stops when the best solution does not change after two consecutive approximate number of decisive iterations ( $ANDI$ ). In other words, GA runs  $ANDI$  times initially and saves the best optimal solution. Afterwards, it runs  $ANDI$  times again and compares new best fitness with previous one. If the difference between these two fitnesses is smaller than a very small pre-specified tolerance, then GA is converged. Otherwise the old fitness is replaced by the new fitness and the process will continue as long as the above condition is satisfied.

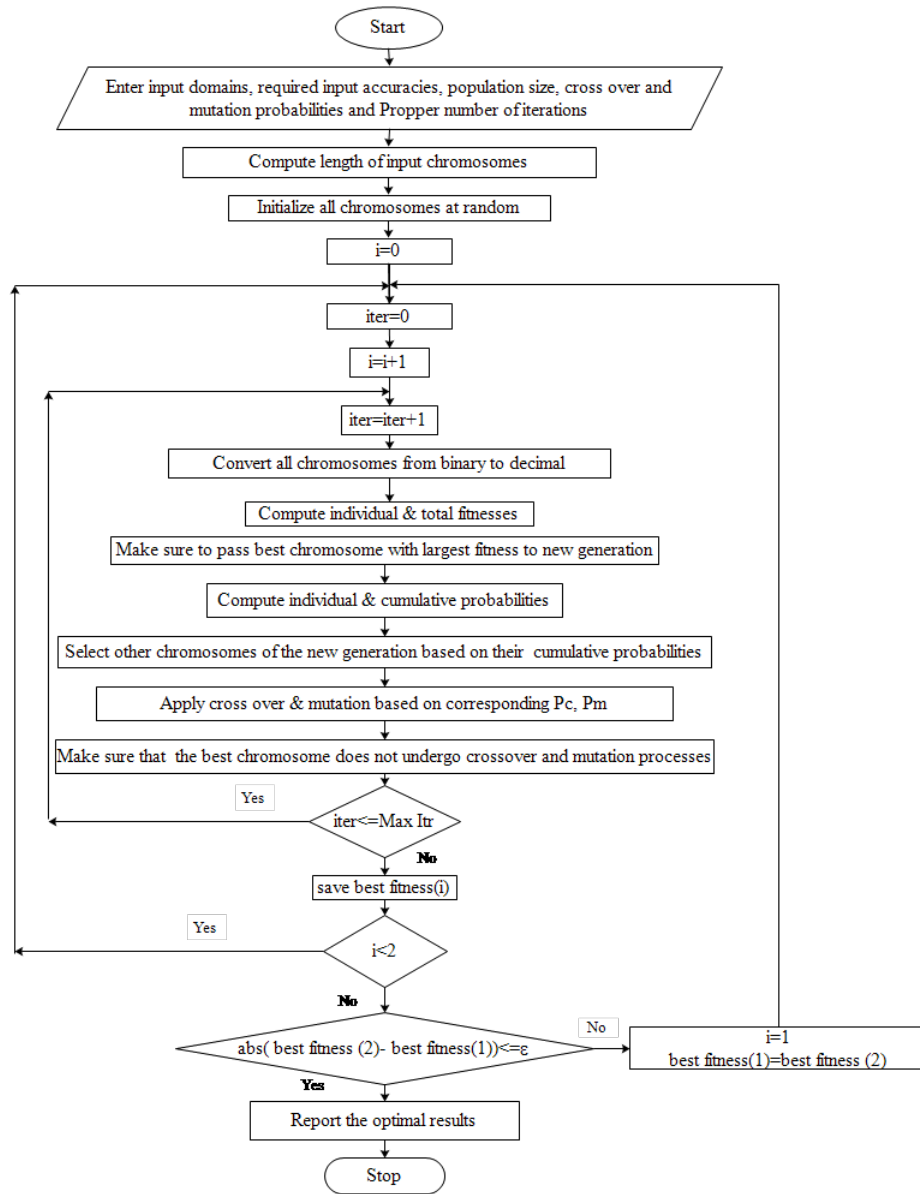


Figure 1: The in-house GA flowchart using *ANDI* concept as the termination criterion.

### 3 Description of *ANDI* and approximate *df*

As mentioned before, the approximate number of decisive iterations (*ANDI*) is defined as the minimum number of iterations that provides proper performance. This parameter has a crucial role in defining the termination criterion of our in-house algorithm. In practical optimization problems, it is essential to choose appropriate number of iterations (generations) which provides optimal answers with required accuracy in a reasonable time.

A simple but tedious way for selection of proper *ANDI* value is to plot the best optimal solution obtained after two consecutive successful number of iterations versus the corresponding number of iterations. Afterwards, a specific criterion is required to select the approximate number of decisive iterations (*ANDI*) based on difference between two optimal solutions obtained for two successive iterations. The maximum value of absolute error (MAE) which is readily computable from the uncertainties of the input variables, is a good candidate to be used as the above criterion for selecting the proper value of *ANDI*. The following section briefly explains a typical method which can be used to find MAE as an appropriate criterion for selection *ANDI*.

For a multivariate differentiable cost function the absolute error (AE) of the output variable ( $y$ ) for a given set of inputs can be readily computed from the following equation provided that sufficiently small errors exist in input variables [18]:

$$AE_y = \sum_{i=1}^p |\partial f / \partial x_i| \Delta x_i \quad (1)$$

where ( $\Delta x_i, i = 1, \dots, p$ ) are the uncertainties of the  $p$  input variables<sup>1</sup>. In GA, a vast domain of input variables is usually exist instead of fixed input values. Therefore, the value of objective function can drastically change with variations of inputs inside corresponding domain. In this work, the value of absolute error for a given cost function is initially computed when the input variables vary inside their domain. The maximum value of the absolute error encountered is then selected as the desired criteria ( $MAE_y$ ) for selection of the required *ANDI*. Figure 2 illustrates a typical plot of AE for the following objective function:

$$f(x_1, x_2) = 100(1 - 3.3x_1 + 2.9x_1^2) \exp(((x_1 - 0.5)/0.25)^2 (1 - 3.3x_2 + 2.9x_2^2) \exp(-((x_2 - 0.5)/0.25)^2) \quad (2)$$

As can be seen, the four distinct hills with different heights shown in 3D plot section of Table 1 can be distinguished in Figure 2. Furthermore, the

<sup>1</sup> For discrete functions with known singularities, the finite difference method can be used to obtain the approximate values of partial derivatives  $\frac{\partial f}{\partial x_i} = \frac{f^+ - f^-}{\partial x_i}$

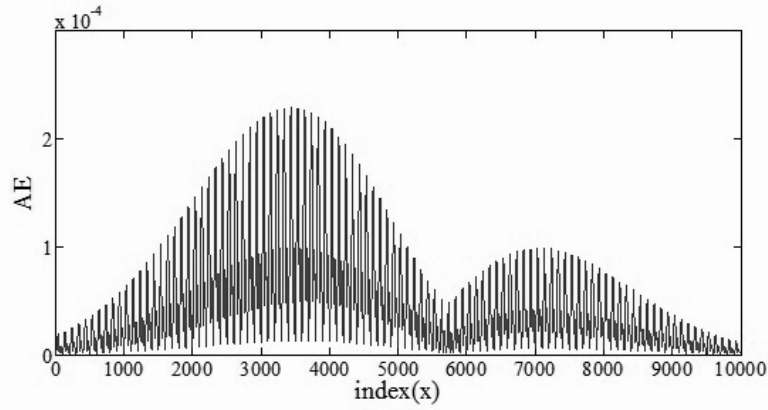


Figure 2: A typical plot of absolute error (AE) for various combinations of input variables inside input domain for cost function of equation (2).

maximum value of absolute error for the merit function described by equation (2) is around 0.022. This value will be used as the required criterion in the following sections to select the optimal value of *ANDI* for the given cost function. Figure 3 shows a typical plot of the best fitness values found after convergence versus the corresponding number of successive iterations for the given cost function inside the corresponding input domain. Assuming that the uncertainties for both input variables to be equal to 0.001, then the maximum value of absolute error ( $MAE_y$ ) for the given cost function is around 0.006, when computed from equation (1). After determining the value of ( $MAE_y$ ) and returning to Figure 3, sufficient number of consecutive successive iterations should be found which the variations in the corresponding best fitness values is less than ( $MAE_y$ ). The square boxes shown in Figure 3 belong to different number of consecutive successive iterations (2-5) which the corresponding best fitnesses variations inside those number of consecutive iterations are smaller than the  $MAE_y$ . As can be seen in Figure 3, after 3 consecutive successive iterations, the value of best fitness approaches its ultimate optimal value of around 2.95. It means that, at least 14 successive iterations are required to ensure that the optimal solution is successfully approximated. In this work, the above number of successive iterations (i.e. 14) is selected as the *ANDI* which can be used as the termination criterion for GA optimization method. To guarantee that sufficiently large values for *ANDI* is chosen, 5 consecutive successive iterations are used for all case study examples presented in latter sections. To the best of our knowledge, this procedure has not been addressed previously.

Evidently, selection of *ANDI* via the above simple but tedious procedure requires extensive computations which can be alleviated by resorting to the following technique.

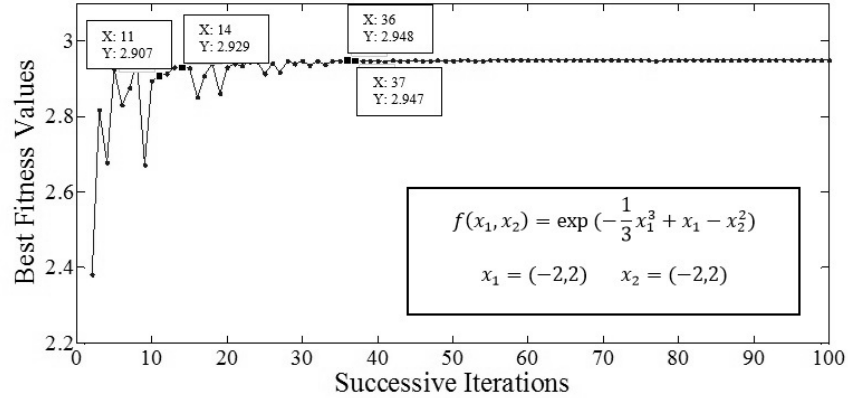


Figure 3: A typical plot of best fitness values (for the corresponding cost function) versus numbers of successive iterations.

A very useful concept usually known as approximate degrees of freedom (*Adf*) is borrowed from the literature [13] which depends on the complexity of the cost function at hand. As an example, the *Adf* is around 2 for a univariate linear function and it is around 4 for a bivariate linear objective function (in the absence of input interactions). For complex multivariate (cost) functions, the value of *Adf* can be computed by using various multivariate expansion series, which requires too many terms for successful reconstruction of the original (merit) function. In another approach, the smoother matrix concept can be used when a linear smoother (e.g. Regularization network) is replaced for the original (cost) function [13, 27]. For a linear smoother model in the form of  $f(x) = S(x)y$ , the effective or approximate degrees of freedom (*Adf*) is defined as the trace of the smoother matrix  $S(x)$ . As we know the trace of a square matrix is its sum of diagonal elements which is equal to the sum of its eigenvalues. In this work, a certain class of linear smoother known as Regularization Network (RN) is used to successfully mimic the original cost function. Then the trace of the so called smoother matrix of the trained RN, learning from various samples (exemplars) provided by the true cost function will be used as the *Adf* of the original merit function [13].

#### 4 A brief review of RN used as a linear smoother to mimic the true nonlinear cost function

Poggio and Girosi [26] illustrated that the solution of Euler-Lagrange partial differential equation (PDE) for the multivariate linear regularization problem



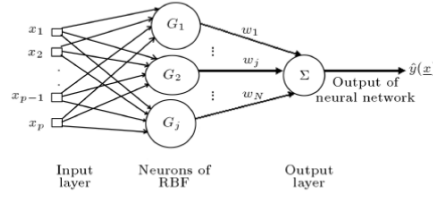


Figure 4: Schematic representation of Regularization network

leads to:

$$(G + I_N)\underline{w}_\lambda = \underline{y} \quad (3)$$

where  $G$  is  $N \times N$  symmetric green matrix with elements  $G_{(i,j)} = (x_i, x_j)$  and  $\lambda$  is the regularization level. The parameter  $\lambda$  should be sufficiently large to ensure that the matrix  $(G + I_N)$  is well behaved and hence invertible. Equation (3) can be shown as the network illustrated in Figure 4. The so called Regularization network (RN) contains  $N$  neurons or centers  $(x_j, j = 1, \dots, N)$  positioned exactly over training data points  $(x_i, i = 1, \dots, N)$ . The influence of the regularization parameter,  $\lambda$ , is embedded in the unknown synaptic weights,  $w'_j$ s.

Using the appropriate differential operator in Euler-Lagrange PDE, the infinitely differentiable factorizable multidimensional Gaussian function in the following form will be final solution and denotes the shape of the basis functions:

$$G(x_i, x_j) = \exp(\| \underline{x}_i - \underline{x}_j \|^2 / \sigma_j^2) = \prod_{k=1}^p \exp[-(x_{i,k} - x_{j,k})^2 / (\sigma_j^2)] \quad (4)$$

The performance of the RN strongly depends on both the appropriate choice of the isotropic spread ( $\sigma$ ) and the proper level of regularization (embedded in the synaptic weights) [31]. By resorting to the leave-one out cross validation (LOOCV or CV) criterion efficient and automatic computation of the optimum level of regularization ( $\lambda^*$ ) can be achieved for any given value of  $\sigma$ .

$$CV(\lambda) = 1/N \sum_{k=1}^N [(\underline{e}_k^T (I_N - H(\lambda)) \underline{y}) / (\underline{e}_k^T (I_N - H(\lambda)) \underline{e}_k)]^2 \quad (5)$$

where  $e_k$  is the  $k$ th unit vector of size  $N$ ,  $I_N$  is the  $N \times N$  unit matrix and  $H(\lambda)$  is the smoother matrix originally defined by Hastie and Tibshirani [13] for a linear smoother such as RN. As mentioned in the previous section, the trace of  $H(\lambda)$  denoted the approximate degrees of freedom of the linear smoother. By definition, the smoother matrix for the RN can be computed

from the following relation:

$$H(\lambda) = S = G(G + \lambda I)^{-1} \quad (6)$$

## 5 Predictions of $Adf$ via RN for different synthetic merit functions

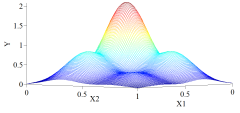
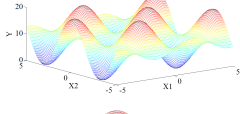
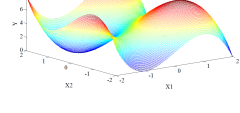
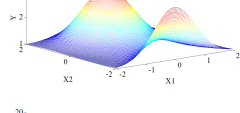
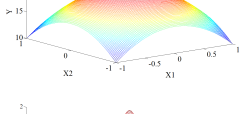
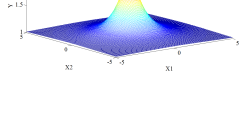
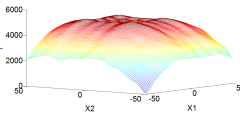
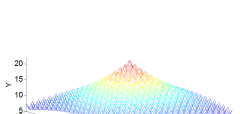
Several synthetic merit functions are used (as shown in 3D plot section of Table 1), to demonstrate the ability of the RN to successfully estimate the  $Adf$  of the original (merit) function and effectively reconstruct it.

A set of distinct exemplars are generated from the original (objective) function and then contaminated with a pre-specified level of noise. These data are initially normalized and then used to train and validate the corresponding optimized RNs with fixed spread of unity (to ensure sufficient overlap between adjacent centers). In practice, a set of distinct exemplars are usually replaces the merit function. In such cases, data generation step is not required anymore but other stages (as well as the initialization of the input variables) should be followed as they will be described in the following section.

For each merit function of Table 1, one hundred equispaced exemplars (on a  $10 \times 10$  grid) are inside the corresponding domain and then contaminated with 10 percent noise level. These data sets are then used to train the equivalent optimized RNs for all functions. As the first step, all input variables are normalized and then the so called  $100 \times 100$  Green matrix is constructed using equation (4) (with centers positioned on normalized training exemplars) and ( $\sigma = 1$ ). Then equation 5 is used to find the optimum value of regularization parameter by minimizing the LOOCV criterion. The LOOCV criterion uses all training data for both training and validation purposes. Finally the trace of the smoother matrix ( $H(\lambda)$ ) and linear weights of RN are computable from equations (6) and (3), respectively. These linear weights along with both maximum and minimum values of all training input data can then be used to compute the generalization performance of the optimally trained RN with optimum level of regularization.

A set of distinct exemplars are generated from the original (objective) function and then contaminated with a pre-specified level of noise. These data are initially normalized and then used to train and validate the corresponding optimized RNs with fixed spread of unity (to ensure sufficient overlap between adjacent centers). In practice, a set of distinct exemplars are usually replaces the merit function. In such cases, data generation step is not required anymore but other stages (as well as the initialization of the input variables) should be followed as they will be described in the following section.

Table 1: Typical bivariate non-linear functions

No.	Cost(merit)function	3D plot	Domain and Range
1	$100(1 - 3.3x_1 + 2.9x_1^2)exp(-((x_1 - 0.5)/0.25)^2)$ $(1 - 3.3x_2 + 2.9x_2^2)exp(-((x_2 - 0.5)/0.25)^2)$		$X_1 = (0, 1)$ $X_2 = (0, 1)$ $Y = (0, 2.061)$
2	$10\cos(x_1)\sin(x_2) + 10$		$X_1 = (-5, 5)$ $X_2 = (-5, 5)$ $Y = (1.7, 20)$
3	$x_1^3 - x_2^3 + 3x_1 - 3x_2 + 4$		$X_1 = (-2, 2)$ $X_2 = (-2, 2)$ $Y = (0, 8)$
4	$exp(-1/3x_1^3 + x_1 - x_2^2)$		$X_1 = (-2, 2)$ $X_2 = (-2, 2)$ $Y = (1, 2.95)$
5	$-5(x_1^2 + x_2^2) + 20$		$X_1 = (-1, 1)$ $X_2 = (-1, 1)$ $Y = (10, 20)$
6	$1 + 1/(1 + x_1^2 + x_2^2)$		$X_1 = (-5, 5)$ $X_2 = (-5, 5)$ $Y = (1, 2)$
7	$f(x) = \sum_{i=1}^D (t_i + 200D) + 5000$ $t_i = \begin{cases} -200 + x_i^2 & \text{if } x_i < 0 \\ -80(2.5 - x_i) & \text{if } 0 < x_i < 2.5 \\ -64(7.5 - x_i) & \text{if } 2.5 < x_i < 5 \\ -64(x_i - 2.5) & \text{if } 5 < x_i < 7.5 \\ -28(x_i - 7.5) & \text{if } 7.5 < x_i < 12.5 \\ -28(17.5 - x_i) & \text{if } 12.5 < x_i < 17.5 \\ -32(17.5 - x_i) & \text{if } 17.5 < x_i < 22.5 \\ -32(27.5 - x_i) & \text{if } 22.5 < x_i < 27.5 \\ -80(x_i - 27.5) & \text{if } 27.5 < x_i < 30 \\ -200 + (x_i - 30)^2 & \text{if } 30 < x_i \end{cases}$		$X_1 = (-50, 50)$ $X_2 = (-50, 50)$ $Y = (0, 5000)$
8	$20exp(-.2\sqrt{\frac{1}{D}} \sum_{i=1}^D x_i^2)$ $+exp(\frac{1}{D} \sum_{i=1}^D \cos(2\pi x_i)) + 40$		$X_1 = (-10, 10)$ $X_2 = (-10, 10)$ $Y = (5.56, 22.72)$

For each merit function of Table 1, one hundred equispaced exemplars (on a  $10 \times 10$  grid) are inside the corresponding domain and then contaminated with 10 percent noise level. These data sets are then used to train the equivalent optimized RNs for all functions. As the first step, all input variables are normalized and then the so called  $100 \times 100$  Green matrix is constructed using equation (4) (with centers positioned on normalized training exemplars) and ( $\sigma = 1$ ). Then equation 5 is used to find the optimum value of regularization parameter by minimizing the LOOCV criterion. The LOOCV criterion uses all training data for both training and validation purposes.

Finally the trace of the smoother matrix ( $H(\lambda)$ ) and linear weights of RN are computable from equations (6) and (3), respectively. These linear weights along with both maximum and minimum values of all training input data can then be used to compute the generalization performance of the optimally trained RN with optimum level of regularization. Figure 5 shows the generalization performances of all 6 trained networks (RNs) on  $100 \times 100$  grids inside the corresponding training domains.

Comparison of these generalizations performances with the original plots of Table 1 reveals that the optimal RNs provided impressive performances on reconstructing the true hyper-surface embedded in limited and noisy training data sets.

## 6 Predictions of *ANDI* values for various synthetic examples

The 3D case studies shown in Table 1 will be ultimately used to find a relatively simple correlation between *Adf* and *ANDI*. As mentioned earlier in section 3, the maximum value of absolute error ( $MAE_y$ ) is used to select the proper value of *ANDI* for each and every (cost) function. The uncertainties in input values and the corresponding  $MAE_y$  values are depicted in Table 2.

As previously mentioned, the  $MAE_y$  values provide proper criteria for estimation of the *ANDI* values or the number of successive iterations that the GA requires to provide the optimum value of cost function, as described in the calculation procedure shown in Figure 1. In our present approach, the Genetic Algorithm is initially executed with different number of successive iterations (2 to 100) and the best fitness values are found for each and every iteration (after convergence) and plotted against the number of successive iterations as shown in Figure 6 for all (merit) functions of Table 1. Afterwards, the  $MAE_y$  values for the each (merit) function is used to select the proper value for *ANDI* for corresponding (merit) function.

Table 3 shows the corresponding best fitness values at different number of successive iterations for various numbers of consecutive successive iterations for all (merit) functions of Table 1. As mentioned before, to guarantee that

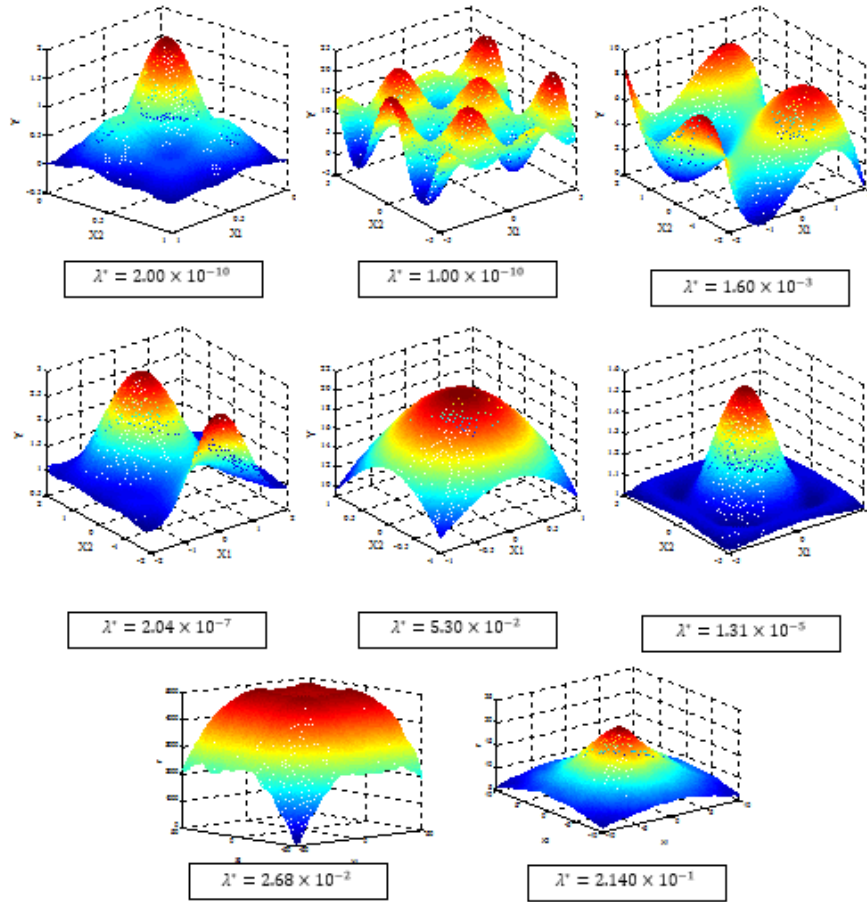


Figure 5: Generalization Performances of RNs ( $\sigma = 1$ ) for all merit functions of Table 1 trained with 100 equi-spaced exemplars contaminated with 10 percent noise.

Table 2: MAE values for functions depicted in Table 1.

Function No.	$MAE_{X_1}$	$MAE_{X_2}$	$MAE_y$
1	.001	.001	.022
2	.001	.001	.010
3	.001	.001	.018
4	.001	.001	.006
5	.001	.001	.014
6	.001	.001	.010
7	.001	.001	.200
8	.001	.001	.070

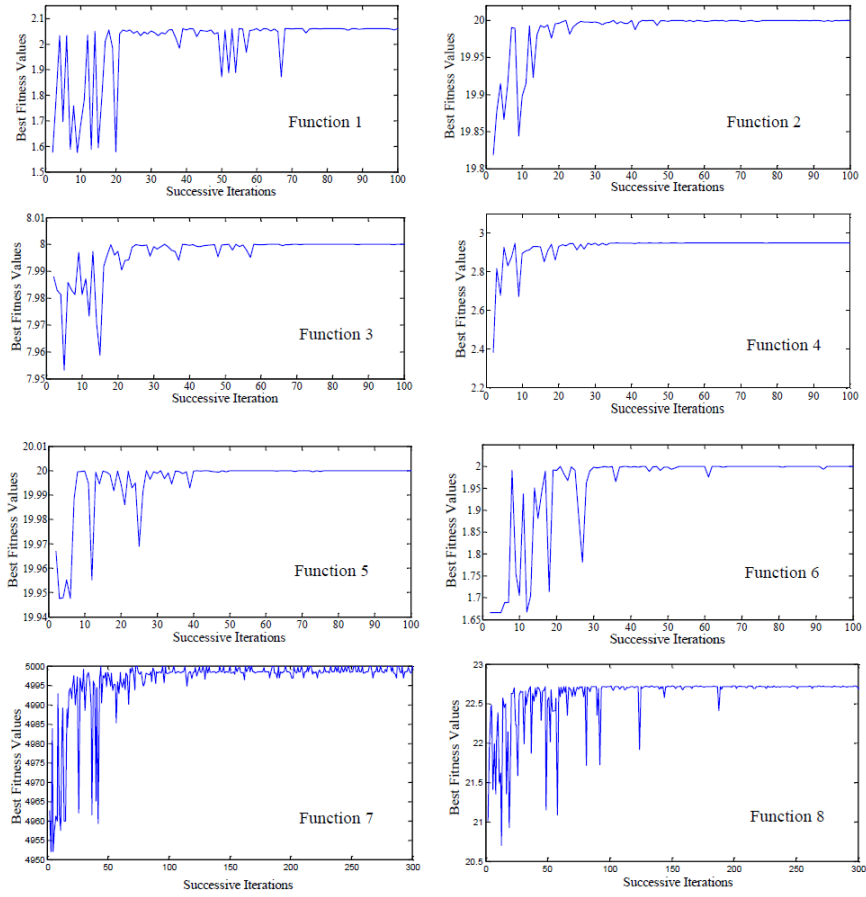


Figure 6: Plots of best fitness values versus number of successive iterations for all merit functions of Table 1.

Table 3: Best fitness values (BFV) at different number of successive iterations (NCSI) for various numbers of consecutive successive iterations (NCSI).

Function No. in Table 1	NCSI=2		NCSI=3		NCSI=4		NCSI=5	
	NSI	BFV	NSI	BFV	NSI	BFV	NSI=ANDI	BFV
1	21	2.055	22	2.050	24	2.042	71	2.060
2	27	19.997	53	19.998	78	19.999	79	19.999
3	2	7.982	3	7.981	18	7.996	19	7.997
4	11	2.907	14	2.929	36	2.948	37	2.947
5	2	19.947	3	19.947	5	19.947	10	19.994
6	2	1.666	3	1.666	4	1.666	33	1.999
7	14	4959.90	108	4988.29	137	4998.14	175	4998.42
8	38	21.98	71	22.71	135	22.65	148	22.72

sufficiently large values for *ANDI* is chosen, five consecutive successive iterations (CSI) are used for all (merit) functions as depicted in the left (NSI: number of successive iterations) column of the NCSI=5 of Table 3. For example, *ANDI* value for the first (merit) function of Table 1 is 71 where the corresponding best fitness value is around 2.060.

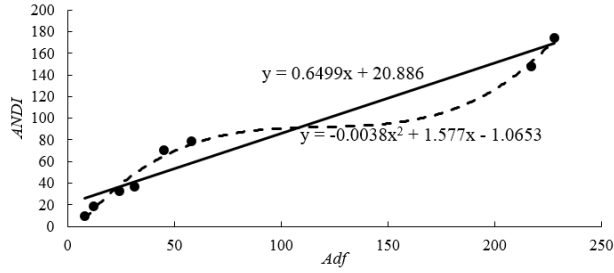
Evidently, the above procedure seems quite complex and can be avoided by resorting to the approximate degrees of freedom (*Adf*). In the absence of an explicit cost function, a linear smoother model such as RN can be used (as discussed in the previous section) to reconstruct the hyper-surface representing the original merit function and compute the *Adf* of the cost function at hand. The final goal is to present a relatively simple correlation between, *ANDI* and *Adf*. This approach has not been addressed previously.

In some cases the cost function is sufficiently simple and the exact df can be easily computed. For example, the exact degrees of freedom for the third and 5th (merit) functions of Table 1 are exactly eight and six, respectively. In the cases, no input interactions exist and two cubic or quadratic models ( $f_1(x_1)$  and  $f_2(x_2)$ ) can successfully reconstruct the true cost functions, respectively. In other cases, the Regularization network can be used to find the *Adf* for the complex cost function at hand. The following procedure is used to train six fully optimized RNs and compute the *Adf* values for all cost functions of Table 1.

1. Generate N training exemplars with sufficient noise level from the cost function at hand.
2. Normalize input variables values.
3. Compute the  $N \times N$  elements of the Green matrix by positioning all the network centers on the normalized inputs and assuming a pre-specified value for the Gaussian isotropic spreads.
4. Compute the optimal value of regularization parameter ( $\lambda^*$ ) by resorting to LOOCV criterion.
5. Calculate the *Adf* value as the trace of  $N \times N$  smoother matrix  $H(\lambda^*)$ , which is already defined by equation (6).

Table 4: Approximate Degrees of freedom ( $Adf$ ) for all merit functions of Table 1.

Func. No.	1	2	3	4	5	6	7	8
$Adf$	45	48	12	31	8	24	228	217
$ANDI$	71	79	19	37	10	33	175	148

Figure 7: Two simple correlations introduced between  $ANDI$  and  $Adf$  values.

The second row of Table 4 shows the values of  $Adf$  for all merit functions of Table 1 computed via the above procedure. As can be seen, the approximate values for the degrees of freedom ( $Adf$ ) of both third and 5th (merit) functions of Table 1 (which are: 12 and 8) are sufficiently close to their exact values (which are: 8 and 6). Other merit functions are too complex to directly compute their  $Adf$  values and the above procedure should be used to calculate the corresponding  $Adf$  values. The third row contains the  $ANDI$  values which are exactly transferred from the last column of Table 3.

## 7 Presenting simple correlations between $ANDI$ and $Adf$

The data of Table 4 are used to plot the variations of  $ANDI$  values versus the corresponding  $Adf$  values, as depicted in Figure 7. Two linear and cubic correlations are extracted by fitting the data of Table 4. Although, the cubic equation almost passes from each and every point, however we prefer to use the linear fit because it is quite simpler and it should also be remember that the horizontal axis represents the approximate (not exact) degrees of freedom. Both of these correlations will be used in the following section to compute  $ANDI$  values for a relatively complex optimization problem.



Table 5: Comparison of best fitness and maximum values found via linear and cubic correlations for 2 different cost functions of Fig. 8.

Function	$Adf$	Correlation	$ANDI$	Best Fitness value	True Maximum
8.a	14	Linear	30	2.1479	2.1481
		Cubic	21	2.1471	
8.b	30	Linear	41	13.5842	13.5846
		Cubic	43	13.5836	

## 8 Validation of the proposed correlations

Various synthetic 2 and 5 dimensional in input domains case studies are used to validate the applicability of the proposed correlations for the termination criterion of the optimal genetic algorithm method. As the results clearly shows, the proposed heuristics successfully predicts the approximate number of decisive iterations ( $ANDI$ ) for all 3D and 6D hyper-surfaces.

### 8.1 Two 3D hyper-surfaces synthetic case studies

Figure 8 illustrates two different 3D synthetic cost functions which are used to validate the effectiveness of the proposed correlations (shown on Figure 7). The first merit function (Figure 8.a) has no input interactions and the corresponding value for its degrees of freedom can be readily obtained as 14 by adding the maximum orders of each input plus number of input variables. Using both linear and cubic correlations, the  $ANDI$  values are computed and fixed at 18 and 22, respectively. Table 5 illustrates the same results for the second cost function (Figure 8.b). For such complex function, the exact degrees of freedom is not readily known because large interactions exist between two input variables. The fully optimized RN is used to estimate its approximate degrees of freedom around 30. Table 5 shows that both  $ANDI$  values computed from correlations of Figure 7 can successfully pinpoint the true maximum (optimum) point. As before, the linear correlation is much simpler and provides almost the same result as the cubic one.

### 8.2 Two 6D hyper-surface synthetic case studies

To demonstrate the robustness of the proposed algorithm and the excellent performance of the Regularization Network on filtering large noise levels and extracting the true underlying hyper-surface from a bunch of noisy data, two 6D synthetic examples are introduced in this section. Figure 9 depicts two

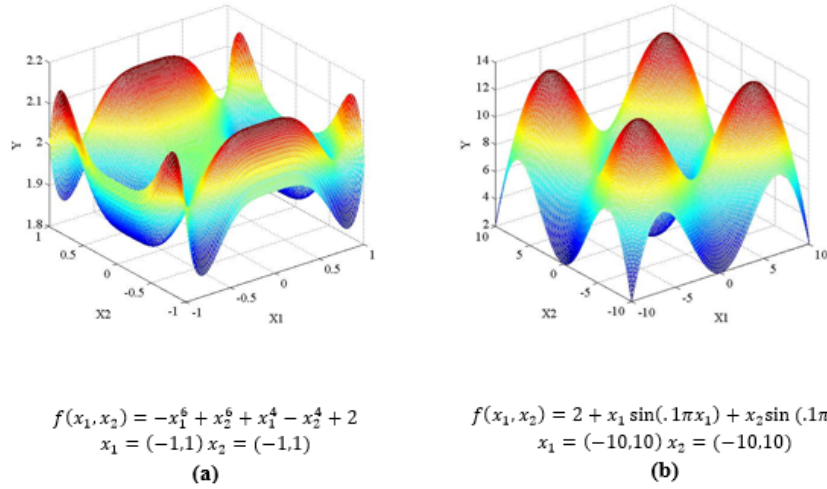


Figure 8: 3D plots and corresponding equations for two bivariate cost functions used for validation purposes.

typical contour maps of these case studies (borrowed from literature [28]). As can be seen, both functions are quite complex with relatively large oscillations and have sufficiently sizable degrees of freedom. To reduce the computational time while preserving the accuracy of the network predictions, 5 randomly spaced exemplars are generated in each input dimensions. To emphasize the excellent noise filtering capability of the Regularization network, the true generated hyper-surfaces is then contaminated with extremely large noise level of 50 percent .

The noisy data set is then used to train the optimal RN and Figure 10 illustrates the impressive performance of such fully optimized RN on capturing the true hyper-surface from heavily contaminated noisy data. The leave one out cross validation technique is used to select the optimum level of regularization [12]. Figure 10 clearly illustrates that the optimal RN which uses linear regularization theory (also known as Tikhonov or Phillips and Twomey method [27]) with optimum level of regularization can successfully pull away the fitted hyper-surface from the noisy training data and reconstruct the true multidimensional surface by filtering out the large noises available in that data set. As table 6 shows, the RN predicts that the fitted hyper-surfaces have Adf s of 53 and 154 respectively, which are quite reasonable for such sophisticated functions. As before, both equations of Figure 7 can provide excellent maximums compared to the real maximum values of the cost functions at hand.

Figure 11 shows that both values of approximate numbers of decisive iterations computed from correlations of Figure 7 provide same results for the fitness values of both functions. Although, several oscillations may occur

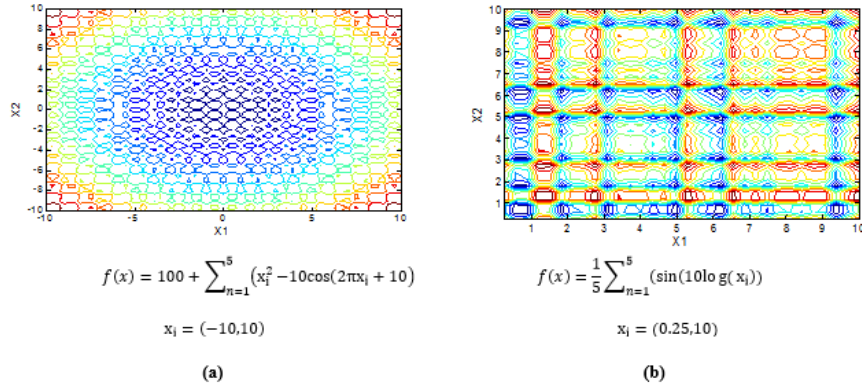


Figure 9: Typical two dimensional contour maps of two 5D synthetic case studies.

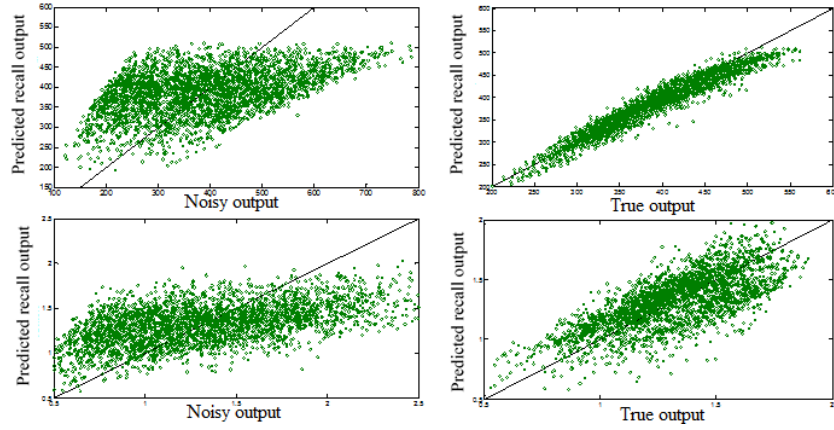


Figure 10: Effectiveness of RN on filtering the noise and reconstructing the true hyper surface in large multi-dimensional case studies.

Table 6: Comparison of best fitness and maximum values found via linear and cubic correlations for 2 different cost functions of Fig. 9.

Function	<i>Adf</i>	Correlation	<i>ANDI</i>	Best Fitness value	True Maximum
9.a	53	Linear	54	642.7	642.8
		Cubic	72	642.6	
9.b	154	Linear	121	1.993	1.993
		Cubic	152	1.993	

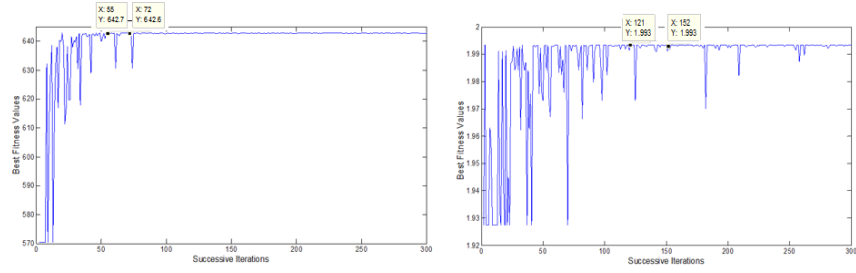


Figure 11: Plots of best fitness values versus number of successive iterations for all both 6D hyper-surfaces.

after the predicted  $ANDI$  values, however the maximum value of the cost function remains essentially constant. In other words, practically no better maximum exists for any iterations greater than  $ANDI$  values.

All above 3D and 6D hyper-surfaces case studies clearly demonstrate that the proposed method can successfully be used to predict the maximum value of any large dimensional cost function via genetic algorithm technique by resorting to approximate number of decisive iterations computable from correlations of Figure 7.

## 9 Conclusion

A new concept was introduced as the approximate number of decisive iterations ( $ANDI$ ) which could be used as the termination criterion for genetic algorithm global optimization method. Two simple correlations were presented which related the novel  $ANDI$  value with the approximate degrees of freedom. A convenient method based of linear smoother concept was used to predict the approximate degrees of freedom for complex merit functions. The proposed approach was successfully tested for 4 different case studies. It was clearly illustrated that the Regularization Network (RN) can successfully filter out large noise levels and reconstruct the true hyper-surface from noisy data. The proposed approach has not addressed previously and can drastically reduce the computation time for complex optimization problems.

## References

1. Aytug, H., Bhattacharrya, S. and Koehler, G.J. *A Markov chain analysis of genetic algorithms with power of 2 cardinality alphabets*, Eur. J. Oper.

- Res. 96 (1996), 195-201.
2. Aytug, H. and Koehler, G.J. *Stopping Criteria for Finite Length Genetic Algorithms*, *Inform. Jo. Comput.* 8 (1996), 183-191.
  3. Bagley, J.D. *The behavior of adaptive systems which employ genetic and correlation algorithms*. Ph.D. Thesis, University of Michigan, 1967.
  4. Bergamaschi, P.R., Saramago, S.F.P. and Coelho, L.S. *Comparative study of SQP and metaheuristics for robotic manipulator design*, *Appl. Numer. Math.* 58 (2008), 1396-1412.
  5. Bhandari, D., Murthy, C.A. and Pal, S.K. *Variance as a Stopping Criterion for Genetic Algorithms with Elitist Model*, *Fundamenta Informaticae*. 120 (2012), 145-164.
  6. Cavicchio, D.J. *Adaptive Search using Simulated Evolution*, Ph.D. Thesis, University of Michigan, 1970.
  7. Conway, B.A. *A Survey of Methods Available for the Numerical Optimization of Continuous Dynamic Systems*, *J. Optim. Theory Appl.* 152 (2012), 2713-306.
  8. Elliott, L., Ingham, D.B., Kyne, A.G., Mera, N.S., Pourkashanian, M. and Wilson, C.W. *Genetic algorithms for optimisation of chemical kinetics reaction mechanisms*, *Prog. Energ. Combust.* 30 (2004), 297-328.
  9. Fleming, P.J. and Purshouse, R.C. *Evolutionary algorithms in control systems engineering: a survey*, *Control Eng. Pract.* 10 (2002), 1223-1241.
  10. Gen, M. and Cheng, R. *Genetic Algorithms and Engineering Design*. New York, John Wiley and Sons, 2000.
  11. Greenhalgh, D. and Marshall, S. *Convergence Criteria for Genetic Algorithms*, *SIAM J. Comput.* 30(1) (2000), 269-282.
  12. Golub, G.H., Van Loan, C.G., *Matrix Computations*, third ed., Johns Hopkins University Press, Baltimore, 1996.
  13. Hastie, T.J. and Tibshirani, R.J., *Generalized Additive Models*, London, Chapman and Hall, 1990.
  14. Hedar, A.R., Ong, B.T. and Fukushima, M. *Genetic Algorithms with Automatic Accelerated Termination*, Technical Report, 2007.
  15. Holland, J.H. *Outline for a Logical Theory of Adaptive Systems*, *J. ACM.* 9 (1962), 297-314.
  16. Jackson, W.C. and Norgard, J.D. *A Hybrid Genetic Algorithm with Boltzmann Convergence Properties*, *J. Optim. Theory Appl.* 136 (2008), 431-443.

17. Kim, J. *Genetic algorithm stopping criteria for optimization of construction resource scheduling problems*, Construction Management and Economics. 31(1) (2013), 3-19.
18. Kopchenova, N.V. and Maron, I.A. *Computational Mathematics: worked examples and problems with elements of theory*, Moscow, Mir Publishers, 1975.
19. Kumar. M. and Banka, H. *Changing Mutation Operator of Genetic Algorithms for Optimizing Multiple Sequence Alignment*, International Journal of Information and Computation Technology. 5(3) (2003), 465-470.
20. Lam, X.B, Kim, Y.S., Hoang, A.D. and Park. C.W. *Coupled Aerostructural Design Optimization Using the Kriging Model and Integrated Multiobjective Optimization Algorithm*, J. Optim. Theory Appl. 142 (2009), 533556.
21. Meyer, L. and Feng, X. *A Fuzzy Stop Criterion for Genetic Algorithms Using Performance Estimation*, In: Proceedings of the Third IEEE Conference on Fuzzy Systems, Orlando, FL, IEEE Service Center, pp. 1990-1995, (1994)
22. Michalewicz, Z. *Genetic Algorithm+Data Structure =Evolution Programs*, New York, Springer-Verlag, 1992.
23. Murthy, C.A., Bhandari, D. and Pal, S.K. *Optimal Stopping Time for Genetic Algorithm*, Fundamenta Informaticae. 35 (1998), 91-111.
24. Nix, A.E. and Vose, M.D. *Modeling genetic algorithms with Markov chains*, Ann. Math. Artif. Intel. 5 (1992) 79-88.
25. Ong, B.T. and Fukushima, M. *Genetic algorithm with automatic termination and search space rotation*, Memetic Comp. 3 (2011), 111-127.
26. Poggio, T. and Girosi, F. *Regularization algorithms for learning that are equivalent to multilayer networks*, Science. 247 (1990), 978-982.
27. Press, W.H., Teukolsky, S.A., Vetterling, W.T. and Flannery, B.P. *Numerical Recipes: The Art of Scientific Computing*, Cambridge, Cambridge University Press, 2007.
28. Qu, B.Y., Liang, J.J., Wang, Z.Y., Chen, Q., Suganthan, P.N.: *Novel benchmark functions for continuous multimodal optimization with comparative results*, Swarm and Evolutionary Computation 26 (2016), 23-34.
29. Rosenberg, R.S. *Simulation of genetic populations with biochemical properties*, Ph.D. Thesis, University of Michigan, 1967.
30. Rudolph, G. *Convergence Analysis of Canonical Genetic Algorithms*, IEEE T Neural Networ. 5(1) (1994), 96-101.

31. Shahsavand, A.: *An Optimal Radial Basis Function (RBF) Neural Network for Hyper-Surface Reconstruction*. Chemistry and Chemical Engineering 16 (2009), 41-53.
32. Sumathi, S., Hamsapriya, T. and Surekha, P. *Evolutionary Intelligence An Introduction to Theory and Applications with Matlab*, Berlin, Springer-Verlag, 2008.
33. Suzuki, J. A Markov chain analysis on a genetic algorithm, IEEE T. Syst. Man Cyb. 25(4) (1995) 655659.
34. Yuan, S., Skinner, B., Huang, S. and Liu, D. *A new crossover approach for solving the multiple travelling salesmen problem using genetic algorithms*, Eur. J. Oper. Res. 228 (2013), 7282.

## رویکردی قابل اطمینان برای اختتام روش بهینه سازی الگوریتم ژنتیک

لیلا لطفی کتولی و اکبر شاهسونند

دانشگاه فردوسی مشهد، دانشکده مهندسی، گروه مهندسی شیمی

دریافت مقاله ۴ مرداد ۱۳۹۴، دریافت مقاله اصلاح شده ۳۰ آبان ۱۳۹۴، پذیرش مقاله ۷ مهر ۱۳۹۵

**چکیده:** در دهه های اخیر، الگوریتم ژنتیک به شکل گسترده ای برای حل بسیاری از مسائل بهینه سازی در زمینه های مختلف علوم و مهندسی مورد استفاده قرار گرفته است. در اکثر موارد تعداد تکرارها تنها معیار برای متوقف کردن الگوریتم ژنتیک می باشد. در عمل، استفاده از این معیار موجب طولانی شدن زمان اجرای الگوریتم جهت رسیدن به نتیجه مطلوب می گردد. در این مقاله، رویکرد جدیدی با عنوان "تعداد تقریبی تکرار های ضروری" ارائه شده است که می تواند موجب توقف موفقیت آمیز روش بهینه سازی الگوریتم ژنتیک در کمترین زمان ممکن شود. همچنین، دو رابطه کاربردی ساده جهت ارتباط پارامتر جدید با درجه آزادی تقریبی برای توابع هدف مورد نظر ارائه گردیده است. برای توابع هدف پیچیده تر می توان از یک برازشگر خطی مانند شبکه عصبی موسوم به رگولاریزاسیون جهت محاسبه مقدار تقریبی درجه آزادی مربوطه استفاده نمود. به منظور اعتبارسنجی موفق رویکرد مطرح شده از چهار مثال موردی مناسب استفاده گردیده است که در تمامی موارد روش پیشنهادی با استفاده از روابط مذکور قادر به یافتن نقطه بهینه مربوطه شده است. بدلیل سادگی و از آنجایی در رابطه خطی ارائه شده، محور افقی نشانگر مقدار تقریبی (نه دقیق) درجه آزادی می باشد، استفاده از آن ارجح تر به نظر می رسد. به وضوح می توان دید که شبکه عصبی رگولاریزاسیون می تواند با موفقیت خطای موجود در داده های واقعی را حذف نموده و سطح چند بعدی حقیقی را از میان مجموعه ای از داده های آغشته به خطا بازسازی نماید.

**کلمات کلیدی:** ژنتیک؛ شرط اختتام؛ درجه آزادی تقریبی؛ تعداد تقریبی تکرار های ضروری؛ برازشگر خطی؛ شبکه عصبی رگولاریزاسیون.