IADIS International Journal on Computer Science and Information Systems
Vol. 12, No. 1, pp. 143-161
ISSN: 1646-3692

AN ADAPTIVE DIFFERENTIAL EVOLUTION ALGORITHM FOR SOLVING SECOND-ORDER DIRICHLET PROBLEMS

Hasan Rashaideh
Department of Computer Science, Prince Abdullah Ben Ghazi Faculty of Information Technology
Al-Balqa Applied University (BAU), Salt 19117, Jordan

ABSTRACT

A new adaptive Differential Evolution (DE) algorithm for finding approximate to the solutions of second-order Dirichlet problems is presented. The proposed adaptive algorithm reflected a variation of the finite difference scheme in the perspective that each of the derivatives are approximated by forward, backward, and central differences' quotients. The major advantage of the novel adaptive algorithm over other numerical methods; it has no limitations on the nature of the problem, type of classification, and the number of mesh points. A test cases that include different classes and types of Dirichlet problems to demonstrate the efficiency and simplicity of the algorithm are presented. The numerical results obtained show strong agreement with exact solutions, and demonstrate reliability and great accuracy of the method.

KEYWORDS

Algorithm, Artificial Intelligence, Numerical Optimization, Differential Evolution, Dirichlet Problems

1. INTRODUCTION

Differential evolution (DE) (Storn & Price, 1997)is considered one of the evolutionary algorithms that took inspiration from natural systems. The idea behind evolutionary algorithms is to generate a set of vectors that can be made to represent candidate solutions. These vectors then go through a cycle of evolutionary process. It is through this process that new candidate solutions are formed when the existing candidate solutions are combined via the following steps: mutation, evaluations, crossover and selection. The sequence and construction of these steps differ from one algorithm to the next.

Storn and Price in 1996 (Storn & Price, 1997) proposed the DE algorithm. The DE algorithm is a simple and powerful tools using vector differences for perturbing the vector population. It is population-based and a stochastic method of optimization. It is also considered to be an effective and efficient global optimizer for the continuous search domain. DE can be used to minimize non-differentiable and non-linear objective functions through continuous spaces. DE has had successful applications in different fields (Fleetwood, 1999; Das et al., 2011; Qin et al., 2009) such as communication (Das et al., 2011), mechanical engineering (Das et al., 2011), and pattern recognition (Das et al. 2008; Ilonen et al., 2003). The basic concept behind DE is to utilize vector differences to perturb the population of the vector during the mutation process (Chakraborty, 2008).

DE's most valuable and fundamental characteristics compared to other EAs (like EP, ACO, GP, like GA, and PSO) are as follows. First of all, DE is simple to code and implement, Moreover, in terms of performance; many studies have shown that the DE performs better than other evolutionary algorithms in terms of convergence speed, accuracy, robustness and the computational time. Moreover, DE only has a few control parameters: mutation scaling factor F, population size (NP), and the probability of crossover Cr. These parameters and their effects have also been studied well (Chakraborty 2008; Das et al., 2011). Because DE has a low space complexity, it can tackle high dimensional optimization problems more easily.

In Abou Ela et al. (2011), one can find the two main differences between the genetic algorithm and the DE. First of all, the operator for the DE mutation is self-adaptive and thus all solutions are given the chance to be chosen when coming up with new solutions. Moreover, its greedy selection process allows new solutions to be chosen if they have better fitness values than their parents. DE's general properties were enumerated by Storn and Kenneth in Price et al. (2006); Storn & Price (1997): it is capable of handling and solving non-linear, non-differentiable, and multi-model cost functions; it is easy to adapt and use; and it has good convergence speed.

DE has several trials for its vector generation strategies and a few of those may be suited for solving a certain problem. Additionally, there are three important control parameters that are involved in DE, i.e. mutation scaling factor, population size, and crossover rate (Storn & Price 1997). These parameters can have a significant influence DE's performance optimization. Thus, successfully solving a certain optimization problem requires one to conduct a long trial-and-error search for the most suitable strategy. One also has to tune the related parameter values. However, this trial-and-error searching process entails higher computational costs. Additionally, as evolution progresses, the DE population may go through varying search space regions and certain strategies correlated with specific parameter settings may end up showing more efficacy than others. Thus, it is best to adaptively identify a suitable strategy and its correlated parameter values at various stages during the evolution/search process.

Generally, the goal of global optimization is to model parameters for the systems as cost function while considering the constraints before minimising the function over a continuous space. Thus, the following function can be used to represent the global optimization problem:

$$\min_{X \in \mathbb{R}^D} f(x) \in \mathbb{R} \quad (1)$$

In Equation 1, the objective function $f : X \subseteq \mathbb{R}^D \longrightarrow \mathbb{R}$ and $\chi = [x_1, \chi_2, \chi_3, \dots, \chi_D]$ represents the solution's vector, and every x_i has bounds based on the lower and upper limits $L_i \leq x_i \leq U_i$.

DE is considered a simple parameter algorithm for optimization. It functions based on a simple stage cycle demonstrated in Figure 1. The goal of the DE algorithm is to help a population of D–dimensional real parameter vectors $\chi = [x_1, \chi_2, \chi_3, \dots, \chi_D]$ evolve. These vectors are responsible for encoding the candidate solutions, i.e. veering them to the global minimum. Once initialization is complete, the initial population for each generation will then evolve when the following operators are used: crossover, mutation, and selection.

2. OPTIMIZATION: INTRODUCTION AND BASIC CONCEPTS

Optimization is not only relevant in mathematics and engineering, but even in our everyday lives. Optimization may even be one of life's most important processes. Our brain performs optimization even when we are learning to talk or walk. Observing natural evolution gives us the impression of having optimization behind it. Theory development is also considered an optimization process. Such observations are interesting and inspiring aspects in developing procedures for numerical optimization, especially when dealing with problems that are difficult and highly complex.

In the field of mathematics, optimization is a term that is used to refer to the study of problems where one aims to minimize or maximize a real function that can meet the dominant constraints. This is achieved by systematically selecting the values of integer or real variables from within a predetermined set. When scientists and engineers propose a new idea, optimization can be used to improve that idea. The process includes trying variations for an initial concept and then utilizing the information obtained from that to improve that idea. Through the years, techniques for optimization have been used in various fields from engineering and medicine to operations and economics (Luenberger & Ye, 2008).

Optimization techniques went through significant changes recently. These changes have given us the opportunity to use these optimization techniques for the most complicated problems today. The general procedure that is utilized to formulate and address optimization problems is summarized in the following steps:

- Analyzing the process to determine the specific characteristics of interest and process variables, i e., listing down all the variables.
- Identifying the optimization criteria and specifying the objective function based on the coefficients and the above variables.
- Using mathematical expressions to develop a valid process model that can relate the process' input-output variables and their associated coefficients. This also includes the equality and inequality constraints. The independent and dependent variables are also identified in order to calculate the number of degrees of freedom.
- Using an appropriate optimization technique so that the problem can be stated mathematically.
- Evaluating how sensitive the result is to changes in the parameter values of the problem and its assumptions.

Optimization problems could be classified into several classes. Categorisation depends on the, the modality of the fitness function, linearity of the fitness function the availability of constraints, the number of fitness functions, the linearity of the constraints (discrete or continuous), and the amount of decision variables. A general optimization problem is

generally seen as a mixture of these classifications. Optimization methods can also be categorised into different categories based on the following factors. First of all, some random components can be used to test the solution space during algorithm convergence - stochastic or deterministic methodologies (Zhou et al., 2013). The second factor is the guarantee of the calculated optimal solution - exact or heuristic methods (Nearchou & Omirou, 2006). Third, the solution's locality is considered; based on this classification, the methods can either be global or local techniques (Qin et al., 2009).

Specifically, global techniques like DE algorithm better when working with solution spaces that have constrained variables, discontinuities, nonlinear relations, or large amounts of dimensions that have several potential local optimums. Global techniques either give an optimum or near-optimum solution instead of just giving a local optimum. Thus, they can come up with useful solutions when local techniques cannot.

3. FORMULATION OF THE DIRICHLET PROBLEMS

The most important aspect of the process is normally problem formulation. Problem formulation involves the selection of design parameters, fitness function, constraints, and the discipline design's models. This section first formulates a system of differential equations to use as an optimization problem that is based on minimizing the cumulative residual error found in every unknown interior node. Afterwards, the minimization problem is converted into a maximization problem by presenting a fitness function as unity.

The Dirichlet problems presented below describe the ordinary differential equations:

$$y''(x) = f\big(x, y(x), y'(x)\big), \tag{2}$$

which is dependent on the boundary conditions

$$y(a) = \alpha, y(b) = \beta, \tag{3}$$

where $a \leq x \leq b$, α, β represent real finite constants, and f signifies the nonlinear function of y and y'.

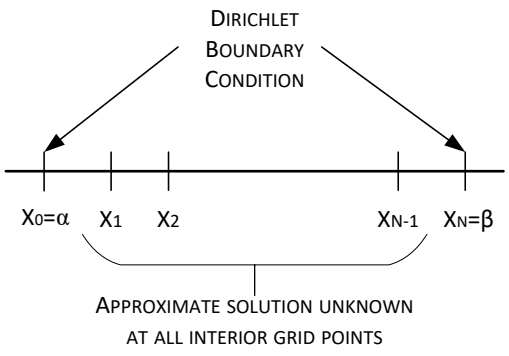

Figure 1. Mesh Grid Points

Using the approximate solutions derived from equations (2) and (3), we can have the condition that the mesh grid points (Figure 2) are distributed equally along the $[a, b]$ interval. This condition is guaranteed when we set $x_i = a + ih,\ i = 0,1, \dots, N$, where $h = \frac{b-a}{N}$. Therefore, given the interior grid points, $x_i,\ i = 1,2, \dots, N-1$, the system that needs to be approximated is as follows

$$y''(x_i) = f\big(x_i, y(x_i), y'(x_i)\big) = 0, x_1 \le x_i \le x_{N-1}, \tag{4}$$

which is subject to the boundary conditions

$$y(x_0) = \alpha, y(x_N) = \beta, \tag{5}$$

One of the simplest and oldest ways to solve differential equations is through finite difference approximations for derivatives. It involves the approximation of the differential operator through the replacement of the equation derivatives using difference quotients. This paper will use this technique to numerically approximate the solutions for system of equations (4) and (5) using DE. The difference quotients from the approximation formulas are close approximations of $y'(x_i)$ and $y''(x_i)$, $i = 1,2, \dots, N-1$, using an $(n+1)$-point found at the interior mesh points and that have an error of up to $\mathrm{O}(h^{n-m+1})$, where $n = 2,3, \dots, N$ and $m = 1,2$ represents the derivative order. This can be obtained easily using (Li, 2005). It is noted that the number n begins at 2 and increases gradually up to N.

First, as seen in (Abu Arqub et al., 2011; Abu Arqub et al., 2015; Abu Arqub & Rashaideh 2013; Abu Arqub & Rashaideh 2017) the differential equations system is transformed into a system of difference equations. To accomplish this, the approximate formulas of $y'(x_i)$ and $y''(x_i)$, $i = 1,2, \dots, N-1$ in Equation (3) are substituted to obtain the system's discretised form. The algebraic equations derived from this will serve as functions of $y\big(x_{i-(n-1)}\big)$, $y\big(x_{i-(n-2)}\big)$, ..., $y\big(x_{i+(n-1)}\big)$, and $x_i,\ i = 1,2, \dots, N-1$. Afterwards, the discretised system has to be rewritten in the following form:

$$F\left(x_i, y\big(x_{i-(n-1)}\big), y\big(x_{i-(n-2)}\big), \dots, y\big(x_{i+(n-1)}\big)\right) := y''(x_i) - f\big(x_i, y(x_i), y'(x_i)\big) \approx 0.$$

The general interior node's residual is represented by R(i) and defined as

$$\mathrm{R}(i) = F\left(x_i, y\big(x_{i-(n-1)}\big), y\big(x_{i-(n-2)}\big), \dots, y\big(x_{i+(n-1)}\big)\right).$$

The $L_2 - norm$ of vector residual, $\|\mathrm{R}\|_2$, is considered a function of all the interior nodes' residuals. It is stated as follows:

$$\|\mathrm{R}\|_2 = \sqrt{\sum_{i=1}^{N-1} R(i)^2}$$

The optimization's notation also signifies the presence of a merit function that one can improve upon and use as a measure of the design's efficacy. The merit function, fitness function, and cost function are used to name the functions being optimised. This function is also used to measure the solution's efficacy. The fitness function, Fitness, that was applied in this work can be defined as

$$\text{Fitness} = \frac{1}{1 + \|\mathrm{R}\|_2}$$

As a matter of fact, in order to convert the $\|R\|_2$ minimisation problem into a Fitness maximisation problem, the overall vector residual has to be mapped into a fitness function. Consequently, lowering the $\|\mathrm{R}\|_2$ value improves the vector fitness. The problem's optimal solution and interior grid points' values are obtained when Fitness approaches unity and $\|\mathrm{R}\|_2$ approaches zero.

4. DESCRIPTION OF THE ADAPTIVE DIFFERENTIAL EVOLUTION ALGORITHM

The DE is considered a highly parallel mathematical algorithm that is capable of converting a set (population of individuals) that has an associated fitness value into a new population (next generation). It accomplishes this by utilizing operators like mutation, recombination/crossover, and selection. This section will present a general review of the DE. A detailed DE description is then provided. Later, it will be demonstrated how DE's efficiency and performance is dependent on several factors. This includes the DE operators' design and the system parameters' settings.

As an optimization technique, DE is based on the concepts of natural selection and genetics. DE was first presented in Storn & Price (1997); Price et al. (2006); Li (2005). From then on, DE has been thought of as the most powerful evolutionary technique. Compared to any other similar techniques, it is also the most applicable stochastic search technique when it comes to optimization problems. This can be attributed to their capacity to solve problems that are multi-objective and non-differentiable (Robič & Filipič, 2005).

The DE theory's general features have been widely accepted and applied. Thus, it has been able to obtain good solutions for different problems types found in various disciplines. The DE-based techniques are very versatile since they can handle discrete and/or continuous design variables. However, DE still differs from many optimization techniques because of the following reasons:

- DE moves from the design space's several points to another set of design points. Thus, DE techniques are better at finding the global minima compared to schemes that work by moving from one point to another (Fleetwood, 1999).
- DE only needs function evaluations and does not need any function derivatives. Even though derivative-based techniques help achieve faster convergence towards the optimum, the derivative is also capable of directing the search process to the local optimum (Chakraborty, 2008).
- DE utilizes probabilistic transition rules. This is an important advantage when it comes to serving as a guide for highly exploitative searches. Due to this, DE should not be thought of as the random walk approach's variant (Neri & Tirronen, 2010).
- DE is better at solving complex engineering problems compared to other techniques because of the large individual population. This provides the DE a more diverse search space that lessens the possibility of it converging to a non-global solution (Price et al., 2006).

- It is easy to parallelize DE as it can be integrated easily into the existing evaluation software. Moreover, individual sets can be simultaneously solved using parallel processors (Tasoulis et al., 2004).

The basis of DE is on the triangle of mutation and recombination and selection. Performing mutation can be done through mutation operator. Recombination can be done using the fitness function that is dependent on the specific problem. Selection is a process of choosing the parent vectors without dependence on their fitness.

Since DE is population-based, it has two major advantages compared to other optimization techniques (Storn 2008; Price et al., 2006; Storn & Price 1997; Fleetwood 1999). First, it determines DE's parallel behaviour as realized by a population of search individuals or candidate solutions that are simultaneously moving (Das et al., 2011). Implementing DE on parallel machines significantly lowers the required CPU time. This is in fact an important benefit of its inherent parallel nature. Second, the crossover procedure actively passes the information about different regions of solution space to different individuals. This exchange of information is what makes DE a robust method and efficient optimization method, especially for optimising functions that have several variables and nonlinear functions. However, DE's population-based nature also gives rise to two main drawbacks. First, it occupies more memory space. Thus, instead of just utilizing one search vector for the solution, N_p, multiple search vectors are utilized. These are used to represent the size of the population. Second, when DE is used on sequential machines, it typically suffers from computational burden. As a result, the time needed to solve particular problems through DE will be relatively higher. Moreover, the solution time is an important point of interest when dealing with real time applications. However, if any real-life problem requires off-line solutions, the major concern will then deal with solution accuracy instead of the solution time.

Because DE only utilizes objective function information without having to incorporate extremely domain-specific knowledge, one can say that it exhibits approach simplicity on one hand and versatility on the other. This signifies that when a DE is developed to solve a certain problem, one can modify it easily to address other problem types. This can be simply achieved by changing the existing algorithm's objective function. This is the reason for why DE is considered a general-purpose search strategy.

The randomized behaviours of DE are a primary aspect that provides for search efficiencies. DE uses stochastic procedures to examine response surfaces for a particular optimization problem. The benefit of this class of behaviour is the capacity to exit from local minima absent direction (Nearchou & Omirou, 2006). Nevertheless, when employing DE in optimization problems attention should focused on two issues; firstly, to see if the parametric set to be optimized is inter-correlated. Secondly, to see if constraints exist on the smoothness of the ensuing solution curves. In examples of non-correlated parametric sets or non-smooth solutions, regular DE will apply. Conversely, if the parameters are interrelated with one another or if smoothing of the solution curve is a must, then the DE is preferred in this instance.

DE relies on the developments of curves in two-dimensional spaces and surfaces in three-dimensional spaces. The algorithmic method starts with a collection of stochastically generated candidate curves and develops superior solutions by adapting differential evolution operators. The new strategy is a comparatively novel category of the optimization method, drawing interest among mathematicians and engineers. The DE recommended in this study comprises the following phases:

1. Initialization: For DE, a population of N D–dimensional real parametric vectors is stochastically generated at the start. The parametric vectors are expressed as:

$$\mathcal{X}_{G,j} = \{\mathcal{X}_{G,j,1}, \mathcal{X}_{G,j,2}, \mathcal{X}_{G,j,3}, \dots, \mathcal{X}_{G,j,D}\}$$

wherein G is the generation number and NP denotes number of candidate solutions. Initial populations should obtain coverage of the total search spaces to the extent possible by utilizing modified normal Gaussian (MNG) functions and modified-tangent hyperbolic (MTH) functions within the search spaces constrained by the stipulated minimum/maximum parametric bounds $\mathcal{X}_{\min} = \{\mathcal{X}_{1,\min}, \mathcal{X}_{2,\min}, \mathcal{X}_{3,\min}, \dots, \mathcal{X}_{D,\min}\}$ and $\mathcal{X}_{\max} = \{\mathcal{X}_{1,\max}, \mathcal{X}_{2,\max}, \mathcal{X}_{3,\max}, \dots, \mathcal{X}_{D,\max}\}$. The following expression is utilized to assemble the initial parametric values of the j^{th} vector:

$$\mathcal{X}_{G,j,i} = \mathcal{X}_{i,min} + p_k(i) * (\mathcal{X}_{i,max} - \mathcal{X}_{i,min}), \mathrm{i} = 1,2, \dots, \mathrm{D}, \mathrm{G} = 0, \mathrm{k} = 1,2$$

wherein $p_1(i)$ denotes an MNG and $p_2(i)$ denotes MTH functions.

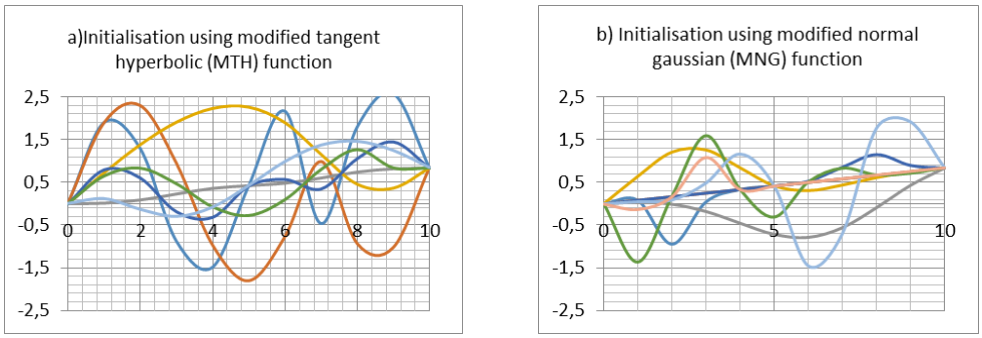

Figure 2. Initialization

Dual smoothing functions that fulfil every boundary condition are selected (Abu Arqub et al., 2011), including the modified normal Gaussian function (MNG)

$$p_1(i) = r(i) + A \sin\left(\frac{\pi}{N} i\right) e^{-0.5\left(\frac{i-\grave{\imath}}{\acute{o}}\right)^2}$$

and the modified tangent hyperbolic function(MTH)

$$p_2(i) = r(i) + A \tanh\left(\frac{i-\grave{\imath}}{\acute{o}}\right) \sin\left(\frac{\pi}{N} i\right)$$

for every $i = 1,2, \dots, D-1$ and $j = 1,2, \dots, NP$, wherein p(i) denotes i-th interior grid point value for the j-th vector, **r** denotes ramp functions of the i-th interior grid value and is described as $\mathrm{r(i)} = \alpha + \frac{\beta-\alpha}{\mathrm{N}}\mathrm{i}$, $\mathrm{N_p}$ denotes the population size, and μ, σ are randomised values within the range $[1, \mathrm{N}-1]$ and $\left]0, \frac{\mathrm{N}-1}{\eta}\right]$ provided $\eta > 0$.

The reason for utilizing these functions is driven by (Abu-arqub et al., 2011; Abu Arqub et al., 2015) to construct adaptive DE. As well, the continuity property of DE allows it to readily manage smooth solution curve candidates where all interior grid points' values that will be optimised are correlated.

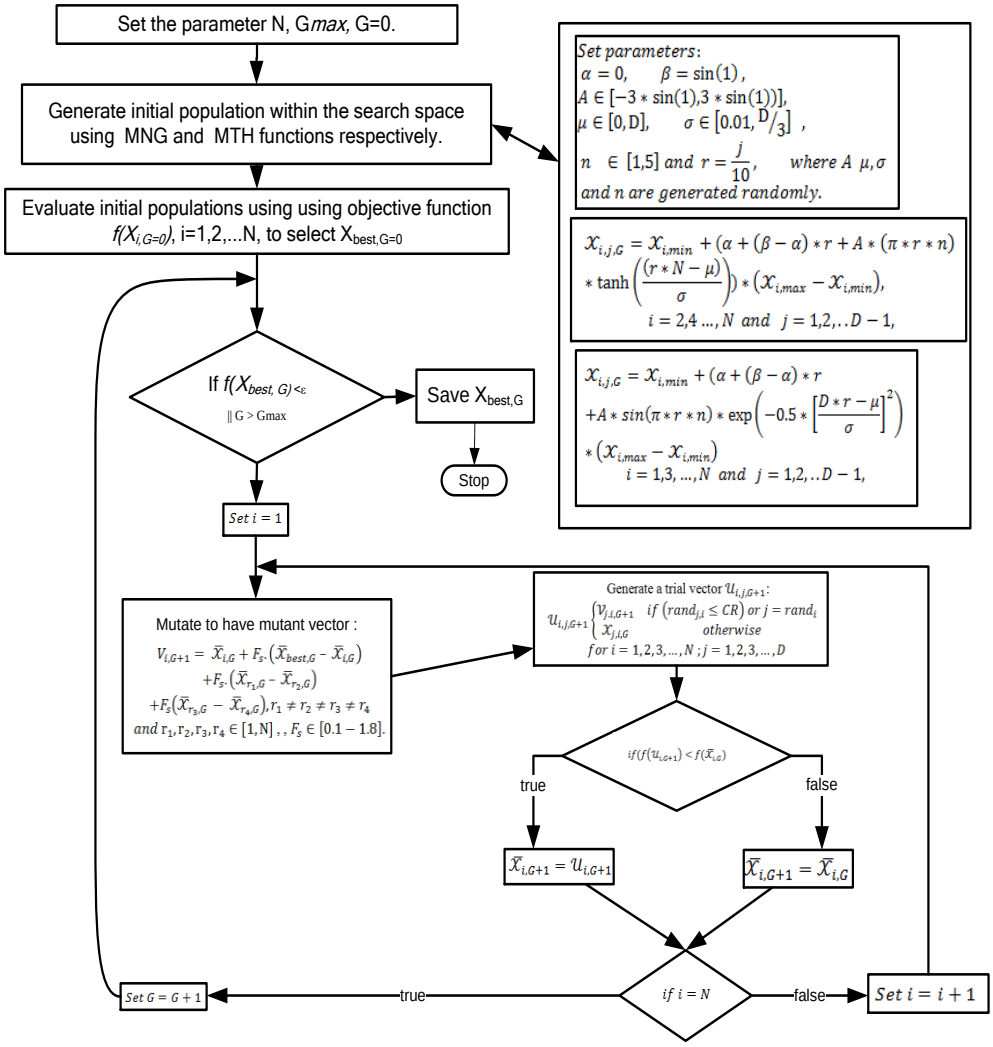

Figure 3. An Adaptive Differential Evolution Algorithm

2. Mutation: With respect to each candidate solution, the so-called target vector $\bar{X}_{G,i}$ at current generation G, a mutant vector $V_{G,i}$ is produced by mutation operator. For each target vector $\bar{X}_{G,i}$ in the current generation, a mutant vector $V_{i,G} = \{v_{i,1,G}, v_{i,2,G}, v_{i,3,G}, \dots, v_{i,D,G}\}$ is produced via the following mutation strategy "DE/rand-to-best/2/bin":

$$V_{i,G+1} = \bar{X}_{i,G} + F_s.(\bar{X}_{best,G} - \bar{X}_{i,G}) + F_s.(\bar{X}_{r_1,G} - \bar{X}_{r_2,G}) + F(\bar{X}_{r_3,G} - \bar{X}_{r_4,G}), r_1 \neq r_2 \neq r_3 \neq r_4,$$

Wherein r_1, r_2, r_3, r_4 are indices randomly selected by random within the range [1, N] from present generation. $F_s \in [0.1 - 1.8]$ denotes mutation scaling factor, and the $\overline{X}_{G,best}$ represents the candidate solutions in accordance with the values of the fitting function in present generation G.

Values of the mutation scaling factor F_s are auto-adaptive, beginning with F_{lower} , and increasing by 0.1 in K iterations until F_{upper}. Values of F_s are utilized in sequence in the Fs series if these verified a certain level of effectiveness in fitness values.

3. Recombination/Crossover:

Successive to the mutation phase, crossover operation are implemented in all pairs of the target vector $\mathcal{X}_{i,G}$ and their respective mutant vector $\mathcal{V}_{i,G+1}$ to produce trial vector $\mathcal{U}_{i,j,G+1}$:

$$\mathcal{U}_{i,j,G+1} \begin{cases} \mathcal{V}_{j.i,G+1} & if\ (rand_{j,i} \leq CR)\ or\ j = rand_i \\ \mathcal{X}_{j,i,G} & otherwise \end{cases}$$

$$for\ i = 1, 2, 3, \dots, NP\ ; j = 1, 2, 3, \dots, D$$

wherein j = 1,2, … ,D; $rand_{j,i} \in [0, 1]$ denotes random numbers, CR denotes crossover rates ∈ [0, 1] and $rand_i \in (1, 2, \dots, D)$ denotes the randomly selected indices.

4. Selection:

Every solution in the population has the same likelihood of being chosen as a parent. Target vector $\mathcal{X}_{i,G}$ compares with the trial vector $\mathcal{U}_{i,G+1}$ and the term resulting in the lowest fitness function value is included in succeeding generations. Selections can be formulated as:

$$\mathcal{X}_{i,G+1} = \begin{cases} \mathcal{U}_{i,G+1} & if(\mathcal{U}_{i,G+1}) \leq f(\mathcal{X}_{i,G}) \\ \mathcal{X}_{i,G} & otherwise \end{cases} \qquad for\ i = 1, 2, 3 \dots, N$$

Mutations, crossovers, and selections progress until the stopping conditions are attained. The algorithmic description of adaptive DE is outlined in Figure 3.

5. NUMERICAL RESULTS

Optimization problems are performed by utilizing an adaptive DE that is among of the more advanced optimization methods due to its evolutionary property; it can address most types of objective functions and limits. DE does not possess mathematical necessities for optimization problems and is also quite successful in conducting globalised probability searches, providing a much versatility. Nevertheless, so as to confirm the computational effectiveness of the constructed adaptive DE, certain numerical experimentation is carried out on well-known problems. The resulting terms were contrasted with the precise solutions and were determined to agree with one another.

The DE generates populations in succession, which can be implemented as an infinitely looping process. Once a user stipulates the fitness values to be attained, these processes can be ended the moment at least one item demonstrates a value in excess of the preferred value. Users are frequently unaware of precisely how large the fitness values of desirable solutions must be. Thus, it would be preferable to end the DE once a user cannot expect further superior solutions. With knowledge of conventional algorithmic methods for optimization, a user may be enticed to observe convergences and then end the DE once maximum fitness values remain relatively continuous over certain populations. Conversely, the ending conditions applied in every problem are reliant on each and do vary between cases. Nevertheless, the DE ends when any of these criteria are encountered:

1. The value of fitness of the best vectors of the populations reaches 0.9999999.
2. The maximum absolute value of residual for the best vector of the population is less than or equal to 1.10×10^{-8}.
3. A maximum number of 1000 iterations is reached.
4. Improvement in the fitness value of the best vector in the population over 100 iterations is less than 1.1×10^{-3}.

The adaptive DE in this paper are utilized to resolve the provided system of Dirichlet problems**.** Inputs to the proposed algorithm are partitioned into dual portions: the adaptive DE-associated parameters and the system-related parameters. Nevertheless, inputs to the proposed algorithm are as shown:

Table 1. The Adaptive DE-Associated Parameters

Parameter	Description
$G_{max} = 1000$	Population size
$N = 10$	Number of grid points
$F_{lower} = 0.1$	Mutation lower bound scaling factor
$F_{upper} = 1.8$	Mutation upper bound scaling factor
$crossover/selection = 0.9$	crossover rate

Blended techniques for initialization methods are utilized when one-half of a population is produced by the MNGF, with the other one-half produced utilizing the MTHF. A set of selected problems of second-order Dirichlet problems are provided as follows, wherein the first problem is linear, with the secondary and tertiary problems of non-linear property.

Problem 1. In the linear system as expressed below:

$$y''(x) = 2y^3(x) - 6y(x) + 4 - 6x^2 - 2x^3$$

constrained by the boundary conditions,

$$y(0) = 1, y(1) = 2$$

wherein $0 \leq x \leq 1$. The precise solutions are expressed as:

$$y(x) = x + 1$$

Problem 2. In the non-linear system as expressed below:

$$y''(x) = 2y^3(x) - 6y(x)y'(x) + 12x^3 - 6x^4 - 2x^6 - 6x^2 + 12x$$

constrained by the boundary conditions

$$y(0) = 1, y(1) = 2$$

wherein $0 \leq x \leq 1$. The precise solutions are expressed as:

$$y(x) = x^2 + 1$$

Problem 3. In the non-linear system as expressed below:

$$y''(x) = \frac{1}{3}(2 - x)e^{2y} + \frac{1}{3(x+1)}$$

constrained by the boundary conditions,

$$y(0) = 0, y(1) = -\ln 2$$

wherein $1 \leq x \leq 2$. The precise solutions are expressed as.

$$y(x) = -\ln(x + 1)$$

Utilising the adaptive DE algorithmic method, with $x_i = 0.1\,i, i = 1, 2, \ldots, 9$ through the fitness function Fitness, and the previously noted ending criteria, the numerical outcomes of estimating $y(x_i)$ for Problem 1 are charted in Tables 1.

Table 2. Numerical Results of y(x) for Problem 1 Utilizing Adaptive DE

Grid Point	*Exact value*	*Approximate value*	*Absolute error*	*Absolute residual*
0	1.00000000	1.00000000000000	0.00000000E+00	0.00000000E+00
0.1	1.10000000	1.09999999999899	1.01096909E-12	-7.36659622E-11
0.2	1.20000000	1.19999999999723	2.77111667E-12	1.71406889E-10
0.3	1.30000000	1.29999999999711	2.89057667E-12	-1.31719080E-10
0.4	1.40000000	1.39999999999555	4.44666526E-12	2.14861906E-10
0.5	1.50000000	1.49999999999589	4.11048973E-12	3.74456022E-11
0.6	1.60000000	1.59999999999629	3.70814490E-12	2.36234365E-10
0.7	1.70000000	1.69999999999871	1.29052324E-12	-2.30481190E-10
0.8	1.80000000	1.79999999999868	1.32382993E-12	-1.96280103E-10
0.9	1.90000000	1.89999999999650	3.49786866E-12	6.21990459E-10
1	2.00000000	2.00000000000000	0.00000000E+00	0.00000000E+00

The numerical outcomes through every grid point of $y(x)$ in Problem 2 are shown in Table 2. It is remarkable that the precision of some grid points varies inversely with their distance (in grid point number) from the boundaries. Conversely, from each table referred to, it is obvious that superior approximations can be attained with the precise solutions.

Table 3. Numerical Results of y(x) for Problem 2 Utilizing Adaptive DE

Grid point	*Exact value*	*Approximate value*	*Absolute error*	*Absolute residual*
0	1.00000	1.00000000000000	0.00000000E+00	0.00000000E+00
0.1	1.01000	1.00999999999789	2.11031193E-12	-1.47183155E-10
0.2	1.04000	1.03999999999555	4.44844162E-12	-2.27947883E-10
0.3	1.09000	1.08999999999245	7.55062679E-12	8.57407922E-10
0.4	1.16000	1.15999999999714	2.86348723E-12	-2.32168063E-10
0.5	1.25000	1.24999999999761	2.38808973E-12	-2.33303155E-10
0.6	1.36000	1.35999999999607	3.92752497E-12	4.06302547E-10
0.7	1.49000	1.48999999999820	1.79789517E-12	3.08615355E-10
0.8	1.64000	1.64000000000109	1.08779652E-12	-1.19458221E-10
0.9	1.81000	1.81000000000132	1.31739064E-12	-2.25442776E-10
1	2.00000	2.00000000000000	0.00000000E+00	0.00000000E+00

The detailed inputs for $y(x)$ in Problem 3 comprise the accurate nodal values, the DE nodal values, absolute errors, and the absolute nodal residuals, as shown in Table 3. Clearly, the precision attained utilizing adaptive DE is good, given its truncation error is of the order $O(h^N)$.

Table 4. Numerical Results of y(x) for Problem 3 Utilizing Adaptive DE

Grid point	*Exact value*	*Approximate value*	*Absolute error*	*Absolute residual*
0	0.0000000000	0.0000000000	0.00000000E+00	0.00000000E+00
0.1	0.1001667500	0.1001990414	3.22913745E-05	1.52596698E-03
0.2	0.2013360025	0.2013682939	3.22913780E-05	1.74098703E-03
0.3	0.3045202934	0.3045525848	3.22913821E-05	1.82894495E-03
0.4	0.4107523258	0.4107846172	3.22913821E-05	1.95256797E-03
0.5	0.5210953055	0.5211275969	3.22913807E-05	2.11365770E-03
0.6	0.6366535821	0.6366858735	3.22913807E-05	2.31457105E-03
0.7	0.7585837018	0.7586159932	3.22913779E-05	2.55825361E-03
0.8	0.8881059822	0.8881382736	3.22913807E-05	2.84827180E-03
0.9	1.0265167257	1.0265490171	3.22913551E-05	7.80555033E-06
1	1.1752000000	1.1752000000	0.00000000E+00	0.00000000E+00

6. STATISTICAL ANALYSIS

As a result of the randomised property of DE, nine distinct runs were conducted for each acquired result in this study, utilizing a dissimilar randomly-generated seed; the outcomes are the mean values of the runs. It is implied that every DE run will end in a slightly dissimilar result. Nevertheless, the convergence data of the trio of problems is shown in Table 4. The table clearly show that problems require 671 repetitions on average, converging on a fitness value of 0.999999993 with a mean absolute nodal residual of value $1.076667E-05$ and a mean absolute difference between the precise values and those acquired through the utilization of DE valued at $6.266668E-04$.

Table 5. Convergence Data of the Three Problems

Problem	*Average iterations*	*Average fitness*	*Average absolute error*	*Average absolute residual*
1	505	0.999999998	2.78E-12	7.22E-11
2	550	0.999999998	3.05E-12	*3.87E-10*
3	959	0.999999983	*3.23E-05*	*1.88E-03*

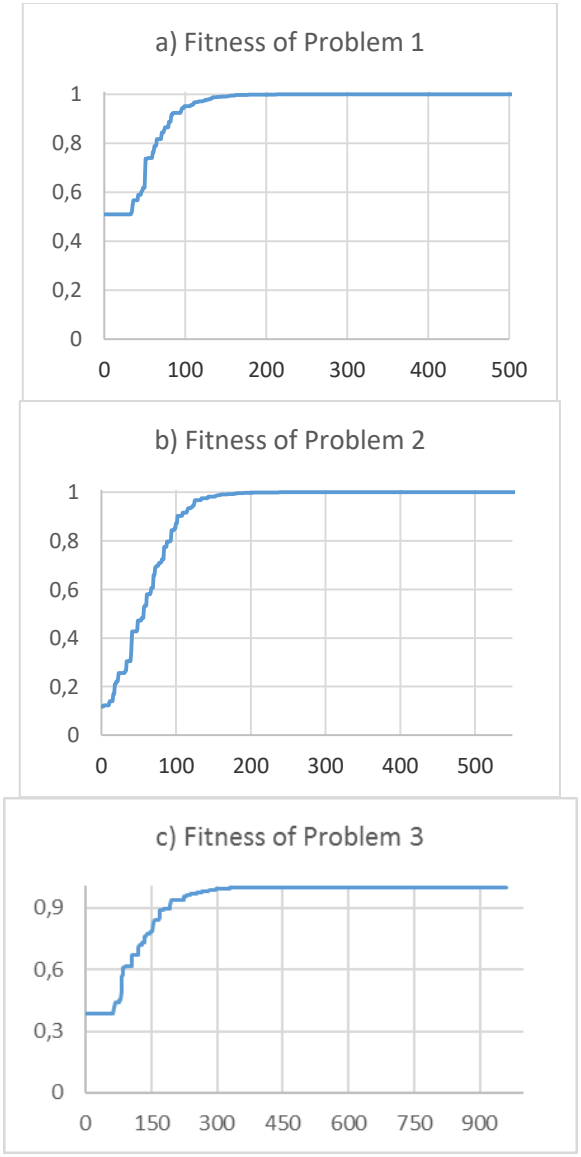

Figure 4. Progress Plots Regarding To Iterations for the Best-Of-Generation Vector for: (A) Problem 1; (B) Problem 2; (C) Problem 3

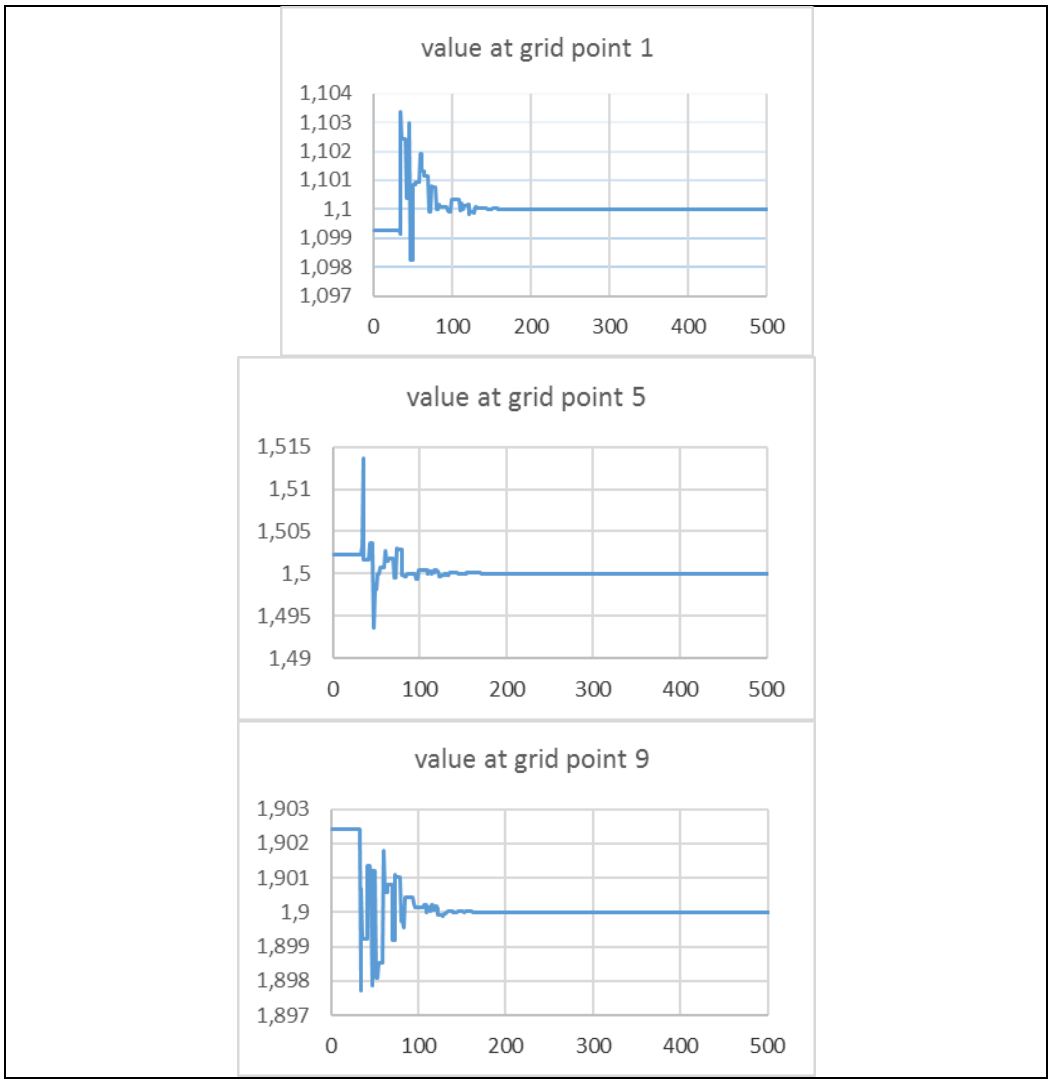

Figure 5. Evolution of the Grid Point Value of y(x) at First Grid Point, Middle Grid Point, and Last Grid Point for Problem 1

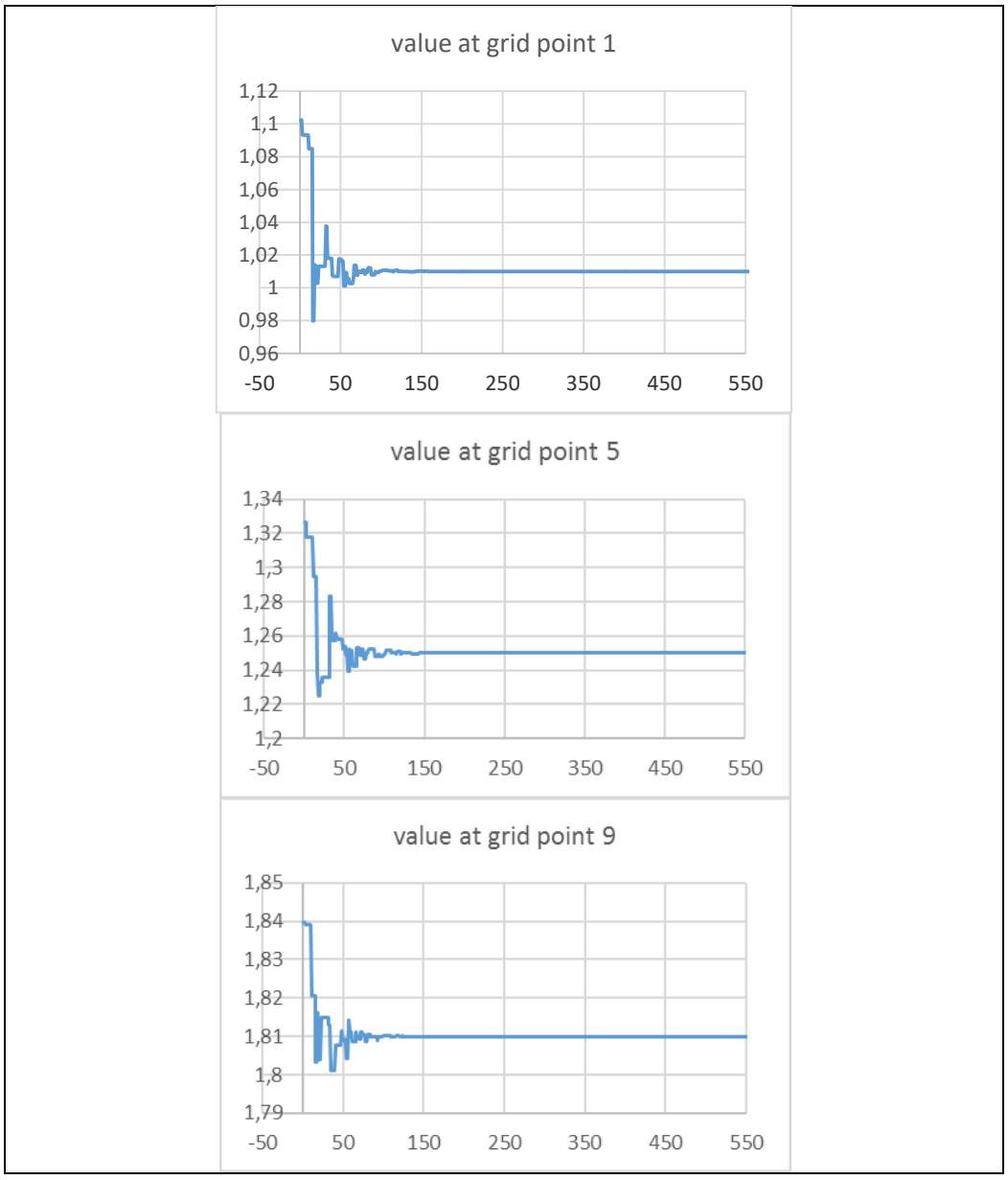

Figure 6. Evolution of the Grid Point Value of y(x) at First Grid Point, Middle Grid Point, and Last Grid Point for Problem 2

The developmental progress plotting of the best-fitness individual values of Problems 1, 2, and 3 are depicted in Figure 4. The figures clearly demonstrate that, in the initial 40% of repetitions, the best-fitness strategies to unity occurred very fast, after which the strategies to

unity occurred at slower rates. It implies that the DE converges to the nearly optimised solutions most rapidly in the initial 40% of repetitions.

The manner by which the internal values for the grid points for Problem 1, problem 2, as well as problem 3 are next examined. Figures 5 depicts the computational value of the initial grid point x_1, middle, x_5, and ninth, x_9 values of the interior grid points of $y(x)$, whereas Figures 6 and 7 demonstrate the computational values of problem 2 and problem 3, respectively, through every repetition.

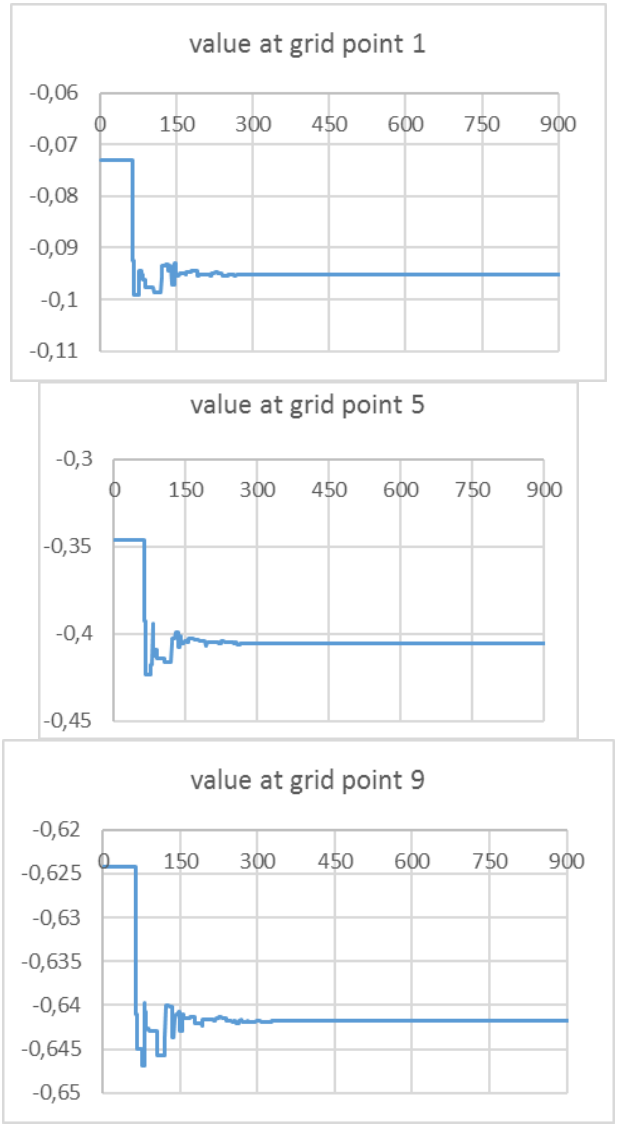

Figure 7. Evolution of the Grid Point Value of y(x) at First Grid Point, Middle Grid Point, and Last Grid Point for Problem 3

7. CONCLUSION

In this paper, solving systems of second-order Dirichlet problems using adaptive DE algorithm is discussed. To validate the proposed algorithm, test cases are designed and solved using adaptive DE algorithm. The results showed that proposed adaptive DE is able to find optimal solutions for test cases in a more reasonable accuracy and promising convergence. Consequently, such algorithm is expected to fit significantly the mathematical and engineering applications. Whilst, the influence of different parameters, including the initialization methods, the evolution of grid points values, the probability crossover and adaptive mutation parameter, the maximum number of iterations, and the maximum nodal residual is studied. The solving procedure reveals that the adaptive DE method is a straightforward, and promising tool for solving linear and nonlinear systems of ordinary differential equations.

ACKNOWLEDGMENT

The author would like to thank Omar Abu Arqub, Habes Alkhraisat, and Shadi Aljawarneh for their valuable comments and suggestions. The author gratefully acknowledges financial support of the Al-Balqa Applied University

REFERENCES

Abou El Ela, A. A., Abido, M. A., & Spea, S. R. (2011). Differential evolution algorithm for optimal reactive power dispatch. *Electric Power Systems Research*, *81*(2), 458–464. http://doi.org/10.1016/j.epsr.2010.10.005

Abu Arqub, O., Abo-hammour, Z., & Rashaideh, H. (2011). Application of Continuous Genetic Algorithm for Second-Order Sigular Boundary Value Problems. In *ICIT 2011 The 5th International Conference on Information Technology*. Zarqa.

Abu Arqub, O., & Rashaideh, H. (2013). Solution of LANE-EMDEN Equation by Resedual Power Series Method. In *ICIT 2013 The 6th International Conference on Information Technology* (pp. 2–7). Amman.

Abu Arqub, O., & Rashaideh, H. (2017). The RKHS method for numerical treatment for integrodifferential algebraic systems of temporal two-point BVPs. *Neural Computing and Applications*, 1–12. http://doi.org/10.1007/s00521-017-2845-7

Abu Arqub, O., Rashaideh, H., & Aljawarneh, S. (2015). Numerical Simulation for Fuzzy Fredholm Integral Equations Using Reproducing Kernel Algorithm. *The 7th International Conference on Information Technology*, *2015*, 497–501. http://doi.org/10.15849/icit.2015.0090

Chakraborty, U. K. (2008). *Advances in Differential Evolution*. Springer. http://doi.org/10.1007/978-3-540-68830-3

Das, S., Abraham, A., & Konar, A. (2008). Automatic Clustering Using an Improved Differential Evolution Algorithm. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, *38*(1), 218–237. http://doi.org/10.1109/TSMCA.2007.909595

Das, S., & Suganthan, P. N. (2011). Differential evolution: A survey of the state-of-the-art. *IEEE Transactions on Evolutionary Computation*, *15*(1), 4–31. http://doi.org/10.1109/TEVC.2010.2059031

Fleetwood, K. (1999). An Introduction to Differential Evolution. *New Ideas in Optimization*, 79–108. http://doi.org/10.1038/155531c0

Ilonen, J., Kamarainen, J.-K., & Lampinen, J. (2003). Differential Evolution Training Algorithm for Feed-Forward Neural Networks. *Neural Processing Letters*, *17*(1), 93–105. http://doi.org/10.1023/A:1022995128597

Li, J. (2005). General explicit difference formulas for numerical differentiation. *Journal of Computational and Applied Mathematics*, *183*(1), 29–52.

Luenberger, D. G., & Ye, Y. (2008). *Linear and nonlinear programming*. Springer.

Nearchou, A. C., & Omirou, S. L. (2006). Differential evolution for sequencing and scheduling optimization. *Journal of Heuristics*, *12*(6), 395–411. http://doi.org/10.1007/10732-006-3750-x

Neri, F., & Tirronen, V. (2010). Recent advances in differential evolution: a survey and experimental analysis. *Artificial Intelligence Review*, *33*(1–2), 61–106.

Price, K., Storn, R. M., & Lampinen, J. A. (2006). *Differential evolution: a practical approach to global optimization*. Springer Science & Business Media. http://doi.org/10.1007/3-540-31306-0

Qin, A. K., Huang, V. L., & Suganthan, P. N. (2009). Differential evolution algorithm with strategy adaptation for global numerical optimization. *IEEE Transactions on Evolutionary Computation*, *13*(2), 398–417. http://doi.org/10.1109/TEVC.2008.927706

Robič, T., & Filipič, B. (2005). DEMO: Differential Evolution for Multiobjective Optimization (pp. 520–533). Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-540-31880-4_36

Storn, R. (2008). Differential Evolution Research – Trends and Open Questions. In U. K. Chakraborty (Ed.), *Advances in Differential Evolution* (pp. 1–31). Springer. http://doi.org/10.1007/978-3-540-68830-3_1

Storn, R., & Price, K. (1997). Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. *Journal of Global Optimization*, *11*(4), 341–359. http://doi.org/10.1023/A:1008202821328

Tasoulis, D. K., Pavlidis, N. G., Plagianakos, V. P., & Vrahatis, M. N. (2004). Parallel differential evolution. In *Evolutionary Computation, 2004. CEC2004. Congress on* (Vol. 2, pp. 2023–2029). http://doi.org/10.1109/CEC.2004.1331145

Zhou, J., Love, P. E. D., Wang, X., Teo, K. L., & Irani, Z. (2013). A review of methods and algorithms for optimizing construction scheduling. *Journal of the Operational Research Society*, *64*(8), 1091–1105. http://doi.org/10.1057/jors.2012.174