Application of Continuous Genetic Algorithm for Second-Order Singular Boundary Value Problems

Omar Abu Arqub[#1], Zaer Abo-Hammour[*2], Hassan Rashaideh[#3]

[#1]Department of Applied Science, Al-Balqa' Applied University, Salt 19117, Jordan;
o.abuarqub@bau.edu.jo

[#2]Department of Mechatronics Engineering, University of Jordan, Amman 11942, Jordan

[*3]I. T. Department, Al-Balqa' Applied University, Salt 19117, Jordan;
rashaideh@bau.edu.jo

Abstract— In this paper, the continuous genetic algorithm, previously developed by the second author, is applied for the solution of the second-order singular boundary value problems. The proposed technique might be considered as a variation of the finite difference method in the sense that each derivatives are replaced by an appropriate difference quotients approximation. This novel approach possesses main advantage as compared to other exiting methods, it can be applied without any limitation on the nature of the problem and the number of mesh points. Numerical example is included to demonstrate the efficiency, accuracy, and generality of the presented technique.

Keyword— Continuous genetic algorithm, singular boundary value problems, finite difference method.

AMS (2010)— 65K05, 34K28, 34B16.

I. INTRODUCTION

Singular boundary value problems (SBVPs) of ordinary differential equations play an important role in many fields. Accurate and fast numerical solution of second-order SBVPs is of great importance due to its wide applications in scientific and engineering research [4,6]. Since, it is usually impossible to obtain the closed form solution even when it exists, this problem must be attacked by various approximate methods [5,7].

Continuous genetic algorithm (CGA) depends on the evolutions of curves in two-dimensional space. Generally, CGA uses smooth operators and avoid sharp jumps in the parameter values. The algorithm begin with a population of randomly generated candidates and evolve to-wards better solution by applying genetic operators which is reproduction, crossover, and mutation. This novel approach is a relatively new class of optimization technique, which are generating a growing interest in the mathematics and engineering community. CGA well suited for a broad range of problems encountered in science, engineering, and economics [1–3].

In this paper, we apply the CGA to develop a novel numerical method for obtaining approximations to the solution of second-order SBVPs with the assumption that the smooth solution is unique. We consider the following equation:

$$\begin{aligned} &y''(x) = f(x, y(x), y'(x)), \ x \in I; \\ &y(a) = \alpha, \ y(b) = \beta. \end{aligned} \tag{1}$$

Here f is continuous real-valued function defined on $I \times D$, where D is an open connected subset of $\mathbb{R}^2$, I is any not closed interval with end points $\{a, b\}$, and $\alpha, \beta \in \mathbb{R}$. This general formulation includes, for example, computational fluid-dynamics, electromag-

netism, electrochemistry, nonlinear elasticity, and optimal control, etc., see [4,6].

In the survey paper [8], Kumar and Singh introduce various numerical techniques include finite difference, finite element, collocation, variational iteration, and other special approximation methods used in literature followed by own critical comments as remarks for solving some explicit forms of Equation (1). However, the previous listed methods require the use of other numerical techniques. Moreover, most of these methods are suited for a special case of Equation (1) with more constraints on the function f, see for example, [11]. In the nonlinear case some of previous methods are available and need some modifications in order to obtain the approximate solutions, see for example [5].

The work presented in this paper is motivated by the needs for a new numerical technique for the solution of the second-order SBVPs with the following characteristics: first, it does not require any modification while switching from the linear to the nonlinear case. Second, the method is not a mathematically guided scheme, which means that it does not use any basic or advanced mathematical tools and has ability to solve the Equation (1) without the use of other numerical techniques; that is, the algorithm should be simple to understand, implement, and should be thus easily accepted in the mathematical and engineering applications fields. Third, the CGA is of global nature in terms of the solutions obtained as well as it is ability to solve other mathematical problems. Finally, the proposed methodology has an implicit parallel nature which points to its implementation on parallel machines. However, being a variant of the finite difference scheme with truncation error of the order $\mathrm{O}(h^n)$, $n \in \mathbb{N}$ the method provides solutions with moderate accuracy.

The reminder of the paper is organized as follows: the formulation of the second-order SBVPs is described in section 2. Section 3 covers the description of the CGA. Numerical results and convergence analysis are given in section 4. Finally, conclusion is presented in section 5.

II. FORMULATION OF THE PROBLEM

In this work, a novel method for the solution of the SBVPs based on a CGA is introduced.

In order to apply the CGA, we must rewrite the Equation (1) in the form of the following:

$$\begin{array}{l} H\left(x, y\left(x\right), y'\left(x\right), y''\left(x\right)\right) = y'' - f\left(x, y, y'\right); \\ y\left(a\right) = \alpha,\ y\left(b\right) = \beta. \end{array} \tag{2}$$

To approximate the solution of Equation (2) we make the stipulation that the mesh points are equally distributed. This condition is ensured by setting $x_i = a + ih$, for each $i = 0, 1, ..., N$ and $h = (b-a)/N$. Thus, at the interior mesh points, x_i, $i = 1, 2, ..., N-1$, the SBVP to be approximated is given as:

$$\begin{array}{l} H\left(x_i, y\left(x_i\right), y'\left(x_i\right), y''\left(x_i\right)\right) = 0,\ i = 1, 2, ..., N-1; \\ y\left(x_0\right) = \alpha,\ y\left(x_N\right) = \beta. \end{array} \tag{3}$$

The difference quotients formulas, which closely approximates $y'\left(x_i\right)$ and $y''\left(x_i\right)$ using an $(n+1)$-point formula at the interior mesh points with error up to $\mathrm{O}(h^n)$, where $n = 2, 3, ..., N$ can be easily obtained by using Algorithm (6.1) in reference [9]. We mention here that the number n is starting from 2 and gradually increases up to N. To complete the formulation substituting the approximate value of $y'\left(x_i\right)$ and $y''\left(x_i\right)$ in Equation (3), discretized form of this equation is obtained. The resulting algebraic equations will be a function of $y\left(x_{i-n}\right)$, $y\left(x_{i-(n-1)}\right)$, $y\left(x_{i-(n-2)}\right)$, ..., $y\left(x_{i+n}\right)$, and x_i. After that, it is necessary to rewrite the discretized equation as: $f\left(x_i, y\left(x_{i-n}\right), y\left(x_{i-(n-1)}\right), ..., y\left(x_{i+n}\right)\right) \approx 0$, for $i = 1, 2, ..., N-1$. The residual of the general interior index, i, is defined as: $\mathrm{Res}(i) = f\left(x_i, y\left(x_{i-n}\right), y\left(x_{i-(n-1)}\right), ..., y\left(x_{i+n}\right)\right)$, for $i = 1, 2, ..., N-1$. The overall individual residual is a function of the residuals of all interior index. It may be stated as: $\mathrm{Oir} = \left(\mathrm{Res}^2\left(1\right) + ... + \mathrm{Res}^2\left(N-1\right)\right)^{1/2}$. A mapping of the overall individual residual into a fitness function is required in the algorithm in order to convert the minimization problem into a maximization problem. A suitable fitness function used in this work is defined as: $\mathrm{Fit} = 1/\left(1 + \mathrm{Oir}\right)$. The individual fitness is improved if a decrease in the value of the Oir is achieved. The optimal solution of the problem, nodal values, will be achieved when Oir approaches zero and Fit approaches unity.

III. DESCRIBED OF CGA

CGA was developed as an efficient method for the solution of optimization problems in which the parameters to be optimized are correlated with each

other or the smoothness of the solution curve must be achieved. It has been successfully applied in the motion planing of robot manipulators, which is a highly nonlinear, coupled problem [2], in the numerical solution of regular two point BVPs [3], and in the solution of optimal control problems [1]. Their novel development has opened the doors for wide applications of the algorithm in the filed of mathematics and engineering. The reader is asked to refer to [3] in order to know more details about CGA including their justification for use, conditions on smoothness of the functions used in the algorithms, several advantages of CGA over conventional one when it is applied to problems with coupled parameters and/or smooth solution curves, etc. The CGA proposed in this work consists of the following steps, for comparison see [3]:

1. **Initialization:** The initial population comprising of N_p smooth individuals. Two smooth functions that satisfy the boundary conditions are chosen, which include the modified normal gaussian function (MNGF) $p(j,i) = r(i) + A\exp\left(-0.5\left((i-\mu)/\sigma\right)^2\right)\sin(\pi i/N)$, and the modified tangent hyperbolic function (MTHF) $p(j,i) = r(i) + A\tanh\left((i-\mu)/\sigma\right)\sin(\pi i/N)$ for each $i = 1, 2, ..., N-1$ and $j = 1, 2, ..., N_p$, where $r(i) = \alpha + i(\beta-\alpha)/N$, $p_j(i)$ is the *i-th* variable value for the *j-th* parent, and μ, σ are a random numbers within the range $[1, N-1]$ and $(0, N/6]$, respectively.

   The choice of A depend on the boundary conditions α and β as follows: A is any random numbers within the range $[-3|\beta-\alpha|, 3|\beta-\alpha|]$ if $\alpha \neq \beta$, within the range $[-3\alpha, 3\alpha]$ if $\alpha = \beta$, and within the range $[-N/3, N/3]$ if $\alpha = \beta = 0$.

   The two initialization functions are shown in Figure 1 in red line, while the ramp function, r, appear in green line.

2. **Evaluation:** The fitness, a nonnegative measure of quality used to reflect the degree of goodness of the individual, is calculated for each individual in the population using Fit equation.

3. **Selection:** In the selection process, individuals are chosen from the current population to enter a mating pool devoted to the creation of new individuals for the next generation such that the chance of selection of a given individual for mating is proportional to its relative fitness.

4. **Crossover:** Crossover combines the features of two parent individuals, say s and h, to form two children individuals, say l and $l+1$, that may have new patterns compared to those of their parents and plays a central role in algorithm. The crossover process is expressed as: $c_l(i) = c(i)p_s(i) + (1-c(i))p_h(i)$, $c_{l+1}(i) = (1-c(i))p_s(i) + c(i)p_h(i)$, and $c(i) = \tanh\left((i-\mu)/\sigma\right)$ for each $i = 1, 2, ..., N-1$, where p_l and p_h represent the two parents chosen from the mating pool, c_l and c_{l+1} are the two children obtained through crossover process, c represents the crossover weighting function. The parameters μ, α are as given in the initialization process.

   Figure 1 shows the crossover process plotted in cia color in a solution curve for the two random parents as follows: c_1: first child in the left graph and c_2: second child in the right graph. It is clear that new information is incorporated in the children while maintaining the smoothness of the resulting solution curves.

5. **Mutation:** The purpose of mutation is to introduce occasional perturbations to the parameters to maintain genetic diversity within the population. The mutation process is governed by formulas: $m_j(i) = c_j(i) + Am(i)$, $m(i) = \exp\left(-0.5\left((i-\mu)/\sigma\right)^2\right)r_2(i)$ for each $i = 1, 2, ..., N-1$ and $j = 1, 2, ..., N_p$, where c_j represents the *j-th* child produced through the crossover process, m_j is the mutated *j-th* child, m is the gaussian mutation function. Regarding the mutation center, μ, and the dispersion factor, σ, used in the mutation function an so mutation process the reader should refer to [3] in order to know more details and the methods that used to generating them. The parameters A, μ, α are as given in the initialization process.

   The mutation processes plotted in a black color for a random child are shown in Figure 1. as follows: m_1: mutation process for the first child and m_2: mutation process for the second child. As in crossover process, some new information is incorporated in the mutated child while maintaining the smoothness of the resulting solution curves.

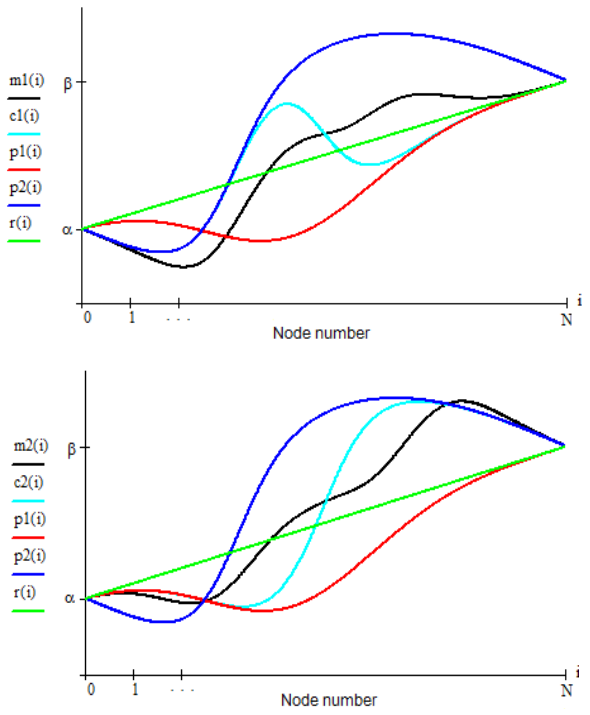

Figure 1. Geometric comparison between process in CGA.

6. **Replacement:** After generating the offspring's population through the application of the genetic operators to the parents population, the parents population is totally or partially replaced by the offspring's population depending on the replacement scheme used. This is known as non-overlapping, generational, replacement. This completes the "life cycle" of the population.

7. **Termination:** The algorithm is terminated when some convergence criterion is met. Possible convergence criteria are: the fitness of the best individual so far found exceeds a threshold value, the maximum number of generations is reached, or the progress limit. After terminating the algorithm, the optimal solution of the problem is the best individual so far found.

To summarize the evolution process in algorithm, an individual is a candidate solution that consists of single curve of N nodal values. The population of individuals undergoes the selection process, which results in a mating pool among which pairs of individuals are crossed over with probability P_c. This process results in an offspring generation where every child undergoes mutation with probability P_m. For example, if P_c value is set to 0.5, then one pair of parents between two pairs is likely to be crossed, and if P_m value is set to 0.5, then one child out of two children is likely to be mutated. After that, the next generation is produced according to the replacement strategy applied. The complete process is repeated till the convergence criterion is met where the N parameters of the best individual are the required nodal values. The final goal of discovering the required nodal values is translated into finding the fittest individual in genetic terms.

Two additional operators were introduced to enhance the performance of the algorithm, the "elitism" and the "extinction and immigration" operators. The reader should refer to [3] in order to know more details about these operators.

IV. NUMERICAL RESULT AND CONVERGENCE ANALYSIS

In order to evaluate the performance of the proposed algorithm for the solution of second-order SBVPs, we utilize the following example.

Example 1 *[12] Consider the nonlinear singular second-order BVP:*

$$y''(x) + \frac{60}{\sqrt{x}(x-1)^2}y'(x) + \frac{3}{\tan(x)}\cos(y(x)) = f(x);$$
$$f(x) = \frac{3\cos(\sin(\pi x)+e)}{\tan(x)} - \pi^2\sin(\pi x) + \frac{60\pi\cos(\pi x)}{\sqrt{x}(x-1)^2};$$
$$y(0) = e,\ y(1) = e,\ x \in (0,1),$$

whose analytical solution is $y(x) = \sin(\pi x) + e$. *This problem can be solved by a few numerical methods, for example, reproducing kernel space [12], cubic spline TAGE method [10], while CGA can be solve it with higher degree of accuracy. To compare the computational results, see [10,12].*

The algorithm was implemented using Visual Basic platform. The input data to the algorithm are as follows: $N_p = 500$, $P_c = P_m = 0.9$, and $N = 10$. Mixed method for initialization schemes are used where half of the population is generated by the MNGF, while the other half generated using the MTHF. The rank-based selection strategy is used where the rank-based ratio is set to 0.1. Generational replacement scheme is applied where the number of elite parents that are passed to the next generation equals one-tenth of the population size. Extinction and immigration operator is applied when the improvement in the fitness value of the best individual of the population over 100 generations is less than 0.01. The termination criterion used for each problem is problem dependent and

vary from one case to another. However, the CGA is stopped when one of the following conditions is met. First, the fitness of the best individual of the population reaches a value of 0.99999. Second, a maximum number of 5000 generations is reached. Third, the improvement in the fitness value of the best individual in the population over 500 generations is less than 0.00001. It is to be noted that the first condition indicate to a successful termination process (optimal solution is found), while the last two conditions point to a partially successful end depending on the fitness of the best individual in the population (near-optimal solution is reached) [3]. Due to the stochastic nature of CGA, sixth different runs were made using a different random number generator seed; results are the average values of these runs [3]. The convergence speed of the algorithm, whenever used, means the average number of generations required for convergence.

The convergence data are as follows: the problems take about 3000 generations, on average, to converge to a fitness value of about 0.99999467 with an average absolute nodal residual $2.8891039957 \times 10^{-6}$ and an average absolute error $1.6468545540 \times 10^{-9}$. Table 1 and Table 2 show the results obtained using CGA across all interior node.

Table 1. Numerical results using CGA

x	Exact value	Approximate value
0.1	3.027298822834	3.027298823609
0.2	3.306067080752	3.306067081574
0.3	3.527298822834	3.527298822995
0.4	3.669338344754	3.669338348798
0.5	3.718281828459	3.718281832346
0.6	3.669338344752	3.669338347402
0.7	3.527298822834	3.527298823863
0.8	3.306067080752	3.306067080856
0.9	3.027298822834	3.027298824183

Table 2. Numerical results using CGA

x	Absolute error	Absolute residue
0.1	$7.75467246 \times 10^{-10}$	$1.30090730 \times 10^{-6}$
0.2	$8.22971913 \times 10^{-10}$	$2.64175808 \times 10^{-6}$
0.3	$1.60777613 \times 10^{-10}$	$3.71315284 \times 10^{-6}$
0.4	$4.04390121 \times 10^{-9}$	$2.48448932 \times 10^{-6}$
0.5	$3.88734511 \times 10^{-9}$	$2.57037032 \times 10^{-6}$
0.6	$2.64769140 \times 10^{-9}$	$4.77295199 \times 10^{-6}$
0.7	$1.02912745 \times 10^{-9}$	$4.69600633 \times 10^{-6}$
0.8	$1.04971587 \times 10^{-10}$	$1.22938197 \times 10^{-6}$
0.9	$1.34943745 \times 10^{-9}$	$2.59291783 \times 10^{-6}$

It is clear that the accuracy obtained using CGA is moderate since it has a truncation error of the order $\mathrm{O}(h^{10})$.

The evolutionary progress plots, of the best-fitness individual is shown in Figure 2. It is observed that from the evolutionary plots that the convergence process is divided into two stages: the coarse-tuning stage and the fine-tuning stage, where the coarse-tuning stage is the initial stage in which oscillations in the evolutionary plots occur, while the fine-tuning stage is the final stage in which the evolutionary plots reaches steady-state values.

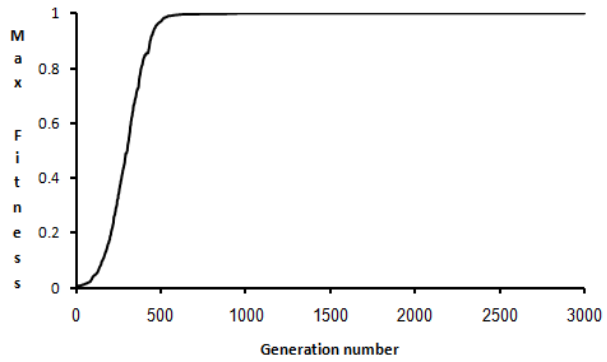

Figure 2. Evolutionary progress plots for the best-of-generation individual.

The percentage of the coarse-tuning stage till convergence from the total number of generations with respect to fitness evolution is 20%, while the remaining 80% is spent in the fine-tuning stage. That means in the first 20% of the generations the best-fitness approaches to one very fast, after that it approaches to one slower. In other words, the approximate of CGA converge to the actual solution very fast in the first 20% of the generations.

The way in which the nodal values evolve are plotted in Figure 3 and Figure 4, respectivelly.

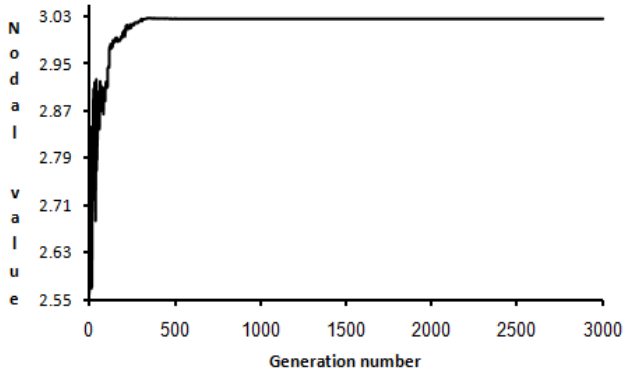

Figure 3. Evolution of the first nodal value.

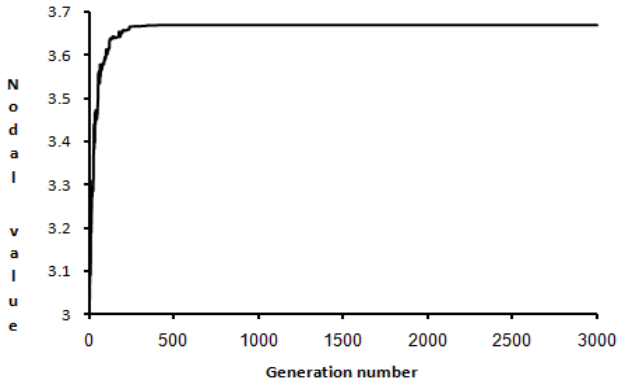

Figure 4. Evolution of the middle nodal value.

It is observed that all nodes, reaches the near optimal solution together. It is also concluded that the evolution has initial oscillatory nature for all nodes. As a result, the distance (number of nodes) from the boundary points doesn't effect in the convergence speed. In addition to that, it is clear by inspection that the coarse-tuning stage takes about 17%, on average, of the total number of generations required for convergence, while the remaining 83% is spent in the fine-tuning stage.

The effect of the different types of initialization methods on the convergence speed of the algorithm is studied next. Table 3 shows that the used initialization method has a minor effect on the convergence speed because usually the effect of the initial population dies after few tens of generations and the convergence speed after that is governed by the selection mechanism, crossover, and mutation operators. As a result, the mixed-type initialization method is used as the algorithm's default method.

Table.3. Effect of initialization functions:

Kind	Mixed-type	MNGF	MTHF
Generations	3000	2987	3003

V. CONCLUSION

In this research, a new numerical method to tackle the second-order SBVPs is proposed. Central to the approach is the novel use of CGA where smooth solution curves are used for representing the required nodal values. Compared with existing classical numerical methods, the present method is found to be simple, efficient, and attractive with a great potential in mathematical and engineering applications.

REFERENCES

1. Z. Abo-Hammour, A. Asasfeh, A. Al-Smadi, and O. Alsmadi ,A novel continuous genetic algorithm for the solution of optimal control problems, *Optimal Control, Applications and Methods*, In Press, 2010.
2. Z. Abo-Hammour, N. Mirza, S. Mirza, and M. Arif, Cartesian path planning of robot manipulators using continuous genetic algorithms. *Robotics and Autonomous Systems* 41(4), pp. 179–223, 2002.
3. Z. Abo-Hammour, M. Yusuf., N. Mirza, S. Mirza, M. Arif, and J. Khurshid, Numerical solution of second-order, two-point boundary value problems using continuous genetic algorithms, *Int. J. Numer. Meth. Engng.* 61, pp. 1219-1242, 2004.
4. R. Agarwal, and D. O'Regan, *Singular Differential and Integral Equations with Applications*, Kluwer Academic Publishers, Dordrecht, Boston, London, 2003.
5. C. Bender and S. Orszag, *Advanced mathematical methods for scientists and engineers*. New York: McGraw-Hill, 1978.
6. S. Chandrasekhar, *Hydrodynamic and Hydromagnetic Stability*, Dover, New York, 1981.
7. D. Greenspan and V. Casulli, *Numerical Analysis for Applied Mathematics*, Science, and Engineering, Addison-Wesley, 1998.
8. M. Kumar, N. Singh, A collection of computational techniques for solving singular boundary-value problems, *Advances in Engineering Software* 40, pp. 288–297, 2009.
9. J. Li, General explicit difference formulas for numerical differentiation, *Journal of Computational and Applied Mathematics* 183, pp. 29-52, 2005.
10. R. Mohanty, P. Sachdev, and N. Jha, An $O(h^4)$ accurate cubic spline TAGE method for nonlinear singular two point boundary value problems, *Applied Mathematics and Computation* 158, pp. 853–868, 2004.
11. J. Rashidinia, Z. Mahmoodi, and M. Ghasemi, Parametric spline method for a class of singular two-point boundary value problems, *Applied Mathematics and Computation* 188, pp. 58–63, 2007.
12. Y. Zhou, Y. Lin, and M. Cui, An efficient computational method for second order boundary value problems of nonlinear differential equations, *Applied Mathematics and Computation* 194, pp. 354–365, 2007. 6