CHAPTER «PHISICAL AND MATHEMATICAL SCIENCES»

APPLYING CONTINUOUS GENETIC ALGORITHM AND DIFFERENTIAL EVOLUTION TO BOUNDARY VALUE PROBLEMS FOR ELLIPTIC EQUATIONS

Larysa Vakal[1]
Yevhen Vakal[2]

DOI: https://doi.org/10.30525/978-9934-26-602-7-12

Abstract. Mathematical modelling stationary processes of different physical nature, e. g. heat conduction (the distribution of temperature in a body after it has reached thermal equilibrium), electrostatics (the electrostatic potential in a region subject to boundary conditions), fluid dynamics (the study of steady, irrotational fluid flows), leads to boundary value problems for elliptic partial differential equations. *The purpose* of the paper is to introduce two methods for finding approximate solutions to boundary value problems for partial differential equations of elliptic form. In both methods, a boundary value problem is formulated as an optimization problem, namely as a differential residual minimization problem or a minimization problem of a boundary condition error. To find a solution to the optimization problem, the continuous genetic algorithm or the differential evolution algorithm can be used. Both algorithms are reliable and general function optimizers based on population, and mutation, crossover and selection operation are its three core operations. *Methodology* of the study is based on modelling stationary processes of different physical phenomena by boundary value problems, on methods to solving boundary

[1] Candidate of Technical Sciences,
Senior Researcher at the Department of Microprocessor Technology,
V.M. Glushkov Institute of Cybernetics, Ukraine
[2] Candidate of Physical and Mathematical Sciences, Associate Professor,
Associate Professor at the Department of Computer Graphics and Visualisation,
Taras Shevchenko National University of Kyiv, Ukraine

value problems for partial differential equations, on evolutionary algorithms, and on numerical methods of analysis. *Results* of the survey showed that evolutionary algorithms, including genetic algorithms and differential evolution, are novel powerful techniques for solving optimization problems. They differ from conventional optimization methods in several key aspects. They operate on a population of candidate solutions rather than a single solution, rely solely on the objective function without requiring auxiliary information, and use probabilistic rather than deterministic transition rules. Differential evolution and continuous genetic algorithms have been successfully applied to solve numerous global optimization problems over continuous spaces. *Practical implications.* Due to their simplicity and powerful search the proposed algorithms can be apply for obtaining optimal parameters of approximate solutions for boundary value problems which are used to model various physical phenomena in a steady state. *Value/originality.* The proposed methods can be considered as viable alternatives to existing approximate analytical methods for solving boundary value problems. They can be applied to obtain solutions both linear and nonlinear problems, and various norms (uniform, quadratic, mean square) can be given for measuring approximation errors.

1. Introduction

Optimization is the process of choosing the best value from a set of possible alternatives and is used in various fields: statistics, computer science, economics, engineering, etc. In mathematics, an optimization problem is the challenge of finding the best possible outcome for a given scenario by maximizing or minimizing an objective function, which is subject to a set of constraints.

In solving optimization problems, Evolutionary Algorithms (EAs) play an important role (see e.g. [2])). As their name suggests, EAs are inspired by natural evolution, a process where organisms highly adapted through many generations of incremental change to thrive in their ecological niche. EAs are population-based algorithms. Each individual of a population represents a search point in the space of potential solutions to a given problem.

Evolutionary Algorithms have distinct advantages over many traditional optimization methods. Unlike methods that search from a single point, EAs evaluate a population of potential solutions simultaneously.

This parallel approach allows them to explore the solution space more broadly and efficiently. EAs only need the objective function values to operate; they don't require any other information about the behavior of the objective function. This makes them ideal for problems where the objective function is non-differentiable, discontinuous, or computationally expensive. EAs are relatively resistant to getting stuck in local optima. They are conceptually and computationally simple to implement.

A well-known group of EAs are Genetic Algorithms (GAs), originally proposed by J.H. Holland [8]. The core idea of GAs is to simulate the natural evolution on a computer by iteratively refining a population of potential solutions through key processes: random initialization, evaluation of a fitness function (which measures the quality of a solution), selection, crossover (recombination), and mutation. This process ultimately produces a new generation of solutions that are better suited to the problem at hand.

Depending on the method of representing variables, GAs are divided into binary GAs and real-coded GAs. A binary GA represents variables as strings of 0s and 1s, making them suitable for problems with discrete variables. A Continuous GA (CGA) or a real-coded GA uses real (floating-point) numbers directly to represent variables, offering a more natural approach for continuous optimization problems and often leading to faster convergence [7].

Differential Evolution (DE), proposed by R. Storn and K. Price [17], is one of the best EAs for solving global optimization problems over continuous spaces. Due to its simplicity and powerful search, it has exhibited remarkable results on many optimization problems. In the First International IEEE Competition on Evolutionary Optimization, which was held in May 1996, it was showed that DE is one of the fastest evolutionary algorithms [17]. In the past years, quite a few DE variants have emerged (e.g. [11; 12]).

Due to their powerful functions, DE and GAs are successfully used to solve problems in various fields (e.g. [10; 25; 19; 21; 5; 23]), including mathematical physics [1; 20; 6; 13; 22; 24].

The purpose of the paper is to introduce two methods for finding approximate solutions to Boundary Value Problems (BVPs) for Partial Differential Equations (PDEs) of elliptic form. In these methods a boundary value problem is formulated as an optimization problem based on the minimization of a differential residual or a boundary condition error. To find solutions to these optimization problems, it is proposed to use a continuous genetic algorithm or differential evolution.

The BVPs for elliptic PDEs are used to model various physical phenomena in a steady state, including heat conduction (the distribution of temperature in a body after it has reached thermal equilibrium), electrostatics (the electrostatic potential in a region subject to boundary conditions), fluid dynamics (the study of steady, irrotational fluid flows), and others.

Methodology of the study is based on modelling stationary processes of different physical phenomena by BVPs, on methods to solving BVPs for PDEs, on evolutionary algorithms, and on numerical methods of analysis.

The remaining sections of this paper are organized as follows. In Section 2, the BVP for the elliptic PDEs is formulated and the review of methods for obtaining its solution is given. In Section 3, two methods for finding approximate solutions of the BVP are introduced. Section 4 covers the description of CGA for solving the BVP. In Section 5, the DE algorithm is described in detail. Numerical examples and discussion are given in Section 6. Finally, some concluding remarks are presented in Section 7.

2. Formulation of BVPs for elliptic PDEs

In case of two independent variables *x* and *y* a BVP for elliptic PDEs can be written in the following formal form:

$$F\left(x, y, u, \frac{\partial u}{\partial x}, \frac{\partial u}{\partial y}, \frac{\partial^2 u}{\partial x^2}, \frac{\partial^2 u}{\partial y^2}, \frac{\partial^2 u}{\partial x \partial y}\right) = 0 \text{ in domain } D, \tag{1}$$

$$S\left(u, \frac{\partial u}{\partial \vec{n}}\right) = 0 \text{ on boundary } \Gamma \text{ of domain } D, \tag{2}$$

where $\vec{n}$ is the external normal to Γ.

One of the most common methods for integrating the BVPs for the **PDEs** is the Fourier method. Its application requires both the differential equation and the boundary conditions to be linear [14].

The linear BVP for elliptic partial differential equations is formulated as follows: it is necessary to find the function $u = u(x, y)$ of the class $C^2(D) \cap C^1(\bar{D})$ that satisfies the equation

$$Lu \equiv \Delta u + p(x, y)\frac{\partial u}{\partial x} + q(x, y)\frac{\partial u}{\partial y} + r(x, y)u = g(x, y)\,, \tag{3}$$

in domain $D \subset R^2$, and on its boundary Γ

$$\alpha u + \beta \frac{\partial u}{\partial \vec{n}} = h(x,y), \qquad (4)$$

where $\Delta u \equiv \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}$ is the Laplace operator or Laplacian, *p*, *q*, *r*, *g*, *h* are the continuous functions, α and β are the given numbers, and $\alpha^2 + \beta^2 > 0$.

If Green's function is known, the solution of the BVP for elliptic **PDEs** with arbitrary boundary conditions can be written in an explicit form. For simple canonical domains, the method of mirror reflections, the method of conformal reflections, and some special methods are used to construct Green's function for the Laplace operator [26, p. 148].

A well-known analytical method for solving BVPs is the method of integral transforms, which often reduces **PDEs** to equations with a smaller number of variables, and sometimes to ordinary differential or algebraic equations [26, p. 211]. Like the Fourier method, this approach is applicable only to linear equations with linear boundary conditions.

To solve BVP (3)–(4), various approximate analytical methods (the collocation method, the Galerkin method, the least square method and others can be applied, similar to their use for approximately solving BVPs for ordinary differential equations [18; 20; 22].

An effective approach for obtaining an approximate solution to BVP (1)–(2) was proposed by L. Collatz [3; 4]. It is based on using the best approximation tool for functions of several variables.

In this paper, we introduce two methods in which the BVP is formulated as an optimization problem based on the minimization of a differential residual or a boundary condition error. To find solutions to these optimization problems, CGA or DE are used.

3. Two methods for finding approximate solutions

This section describes two methods for finding approximate solutions to BVPs (1)–(2) and (3)–(4).

In the first method, the BVP is formulated as a minimization problem of a differential residual. There is selected a function $v(x,y;c_1,\ldots,c_n)$ such that at any values of parameters $c_1,\ldots,c_n$ satisfies boundary condition (2).

The function v is substituted into the differential equation (1), resulting in the differential residual R:

$$R(x, y;\ c_1, c_2, \ldots, c_n) = F\left(x, y, v, \frac{\partial v}{\partial x}, \frac{\partial v}{\partial y}, \frac{\partial^2 v}{\partial x^2}, \frac{\partial^2 v}{\partial y^2}, \frac{\partial^2 v}{\partial x \partial y} \right) \tag{5}$$

Note that for BVP (3)–(4), the differential residual R is written as [20]

$$R(x, y;\ c_1, c_2, \ldots, c_n) = Lv - g(x, y)\ . \tag{5$'$}$$

Next, it is necessary to find such values of $c_1, c_2, \ldots, c_n$ that the differential residual R could be the least deviating from zero in a given norm in domain D.

To find these parameter values, we cover the domain D with a two-dimensional grid $E_m = \{(x_l, y_l),\ l = 1, \ldots, m\}$ and give a norm $\|\cdot\|$ for the residual R. The most commonly used are the quadratic norm

$$\|R(x,\ y;\ c_1, \ldots, c_n)\| = \sum_{l=1}^{m} R^2(x_l,\ y_l;\ c_1, \ldots, c_n)\,, \tag{6}$$

the root mean square norm

$$\|R(x,\ y;\ c_1, \ldots, c_n)\| = \sqrt[2]{\frac{1}{m} \sum_{l=1}^{m} R^2(x_l,\ y_l;\ c_1, \ldots, c_n)}\ , \tag{7}$$

and the uniform (Chebyshev) norm

$$\|R(x,\ y;\ c_1, \ldots, c_n)\| = \max_{l=1,\ldots,m} |\ R(x_l,\ y_l;\ c_1, \ldots, c_n)\ |\,. \tag{8}$$

Thus, the above problem of determining the values of the parameters $c_1, c_2 \ldots, c_n$ for the approximate solution $v(x, y; c_1, \ldots, c_n)$ can be considered as the problem of minimizating the differential residual R:

$$\|R(x, y; c_1, \ldots, c_n)\| \to \min_{c_1, \ldots, c_n}\ . \tag{9}$$

The parameter values, at which this minimum is achieved, are called the best or optimal.

To find the optimal parameter values for the approximate solution v, CGA or DE can be used (these algorithms are described in detail below).

In the second proposed method, there is selected a function $v(x, y; c_1, \ldots, c_n)$ such that at any values of parameters $c_1, \ldots, c_n$ exactly satisfies differential equation (1) (or equation (3) for the linear BVP). The function v is substituted into boundary conditions (2), resulting in the boundary condition error ε:

$$\varepsilon(x, y; c_1, \dots, c_n) = S\left(v, \frac{\partial v}{\partial \vec{n}}\right). \tag{10}$$

For BVP (3)–(4), the error ε is written as

$$\varepsilon(x, y; c_1, \dots, c_n) = \alpha v + \beta \frac{\partial v}{\partial n} - h(x, y), \tag{10$'$}$$

It is necessary to find such values of $c_1, c_2, \dots, c_n$, that the boundary condition error ε could be the least deviating from zero in a given norm on the boundary Γ.

We cover the boundary Γ with the grid $E_m = \{(x_l, y_l),\ l = 1, \dots, m\}$ and select one of norms (6)–(8). Then, the problem of determining the best values of $c_1, c_2, \dots, c_n$ for the approximate solution *v* is reduced to the problem of minimizing the boundary condition error ε:

$$\| \varepsilon(x, y; c_1, \dots, c_n) \| \to \min_{c_1, \dots, c_n}. \tag{11}$$

The global minimum of the error ε equals 0. It is achieved at the exact solution *u*. To find the optimal parameter values for the approximate solution *v*, CGA or DE are used.

Note that using uniform norm (8) allows us to estimate the error of the approximate solution for the first BVP (when $\beta = 0$ in (4)) in the whole domain $\bar{D} = D \bigcup \Gamma$ based on the error on the boundary Γ. As the difference $u - v$ between the exact and approximate solutions is a harmonic function of *x* and *y*, the value of the boundary condition error ε is simultaneously the maximum value of $|u - v|$ in $\bar{D}$. This fact demonstrates one of the significant advantages of using the uniform norm to solving BVPs for elliptic **PDEs**, where the principle on achieving a maximum on the boundary of the domain [14, p. 213] holds for the modulus of the difference $u - v$.

It should be added that the described methods can also be used to find approximate solutions for BVPs given in polar coordinates (see Section6).

4. CGA for finding optimal parameter values

GAs are optimization methods that mimics biological evolution as a problem-solving strategy. For using GAs, an optimization problem is formulated in such a way that its solution can be presented in the form of a vector ("chromosomes"), components of which are the parameters ("genes") characterizing this solution. GAs process a population of chromosomes with three operations: selection, crossover and mutation. Chromosome evaluation is performed by means of the fitness function that depends on the specific optimization problem.

For optimization problems (9) and (11), it is more convenient to use a CGA, which offers a more natural approach to continuous optimization problems and often leads to faster convergence. In CGA, a chromosome is a vector of floating point numbers whose size is kept the same as the length of the vector, which is the solution to the problem [7]. For problems (9) and (11), a chromosome will consist of n genes. Each gene is a real number and presents a certain parameter c_i of the approximate solution $v(x, y; c_1, \ldots, c_n)$.

To construct CGA for any problem, it is necessary to determine the best genetic operators (crossovers and mutations) for this problem, the operator for selecting parent chromosomes for crossover, and the strategy for forming a new generation. It should be noted that CGA operators are usually determined by trial and error based on an analysis of the results obtained [21].

For solving minimization problem (9) and (11), we offer the following CGA.

1. The initial generation ($G = 0$) involves Np chromosomes $S_1, S_2, \ldots, S_{Np}$. The genes $s_{1i}, s_{2i}, \ldots, s_{ni}$ of each chromosome S_i $(i = \overline{1, Np})$ are random numbers generated from a specified numerical interval (by default the interval is set as $[-1, 1]$).

2. Each chromosome S_i, $i = \overline{1, Np}$, is evaluated by its value of fitness function *Fit*. For optimization problem (9), $Fit(S_i)$ is calculated by the formula

$$Fit(S_i) = \| R(x, y; s_{1i}, \ldots, s_{ni}) \| . \tag{12}$$

For optimization problem (11), $Fit(S_i)$ is computed using the formula

$$Fit(S_i) = \| \varepsilon(x, y; s_{1i}, \ldots, s_{ni}) \| . \tag{13}$$

Here $\|\cdot\|$ is one of norms (6)–(8). For example,

$$Fit(S_i) = \max_{l=1,\ldots,m} | \varepsilon(x_l, y_l; s_{1i}, \ldots, s_{ni}) | ,$$

for uniform norm (8) and the boundary condition error ε.

The closer the fitness value $Fit(S_i)$ is to zero, the fitter the chromosome, and the closer the values of parameters encoded in its genes to their optimal values.

Note that in the first proposed method, the values of derivatives are required to determine in the points of the grid $E_m = \{(x_l, y_l),\ l = 1, \ldots, m\}$. The derivatives can be calculated in two ways: numerically (with a certain order of smallness) and analytically. Since the differentiation is always performed according to strict rules, the analytical calculation of the derivative doesn't present any particular difficulties. Furthermore, the analytically calculated derivative can be more accurate.

3. The selection of parents for crossover is performed according to a paired tournament selection procedure. Two chromosomes are randomly chosen from the population, and the one with the higher fitness function value is placed in an intermediate array. After this operation is repeated *Np* times, all consecutive pairs of chromosomes from the intermediate array are subjected to crossover.

4. For recombination, linear crossover is used. Each pair of parents, S_1 and S_2, produces three offspring $0.5S_1 + 0.5S_2$, $1.5S_1 - 0.5S_2$ and $1.5S_2 - 0.5S_1$ [27]. In such problems, this crossover outperforms most crossover operators [21].

5. The mutation operator arbitrarily alters one gene of one randomly selected offspring with a prescribed probability *Pm*. The gene's value is replaced with a new value chosen randomly from a user determined range (by default the range $[-0.5, 0.5]$ is set).

The mutation probability *Pm* is a control parameter of the algorithm. A recommended value for *Pm*, determined in computational experiments [21], is 0.1.

The mutation operator allows to increase the structural variability of the population and restore lost or unexplored genetic material into the population to prevent the premature convergence of the algorithm to suboptimal solutions [7].

6. Reduction and selection. At the step of forming the next generation (*G*+1), only *Np* chromosomes with the lowest value of the fitness function are included from the extended population of parents and offspring.

7. The algorithm stops if one of the following conditions is satisfied:

– a given maximum number of generations G_{max} is reached (by default $G_{max} = 200$);

– the fitness function value of the best chromosome in a current generation is less than a user-determined constant $\propto$ (see Section 6).

If none of the above conditions is fulfilled, the passage to step 3 is carried out.

The efficiency and performance of CGAs depend on the settings of the control parameters, in particular, the population size (*Np*). It effects on the convergence speed of the CGA. Small population sizes suffer from a greater number of generations needed for convergence and a higher probability of getting stuck in local minima, while large population sizes suffer from a larger number of fitness function evaluations, which leads to an increase in the execution time of the CGA. The optimal *Np* values depends on the number *n* of genes in a chromosome. For optimization problems (9) and (11), based on the results of computational experiments [21], the following values of the population size are offered: $Np = 150 \div 200$ for $n = 3$ and $Np = 200 \div 300$ for $n = 4$.

5. DE algorithm for obtaining optimal parameter values

The DE algorithm operates on a population of *n*-dimensional real-parameter vectors, each of them encodes a potential solution. Similar to CGA, the DE algorithm begins with a randomly generated population of vectors. The population then evolves iteratively through the application of mutation, crossover, and selection operators until a stopping criterion is met.

For each vector (known as the target vector), the mutation operator creates a mutant vector by combining other vectors from the current population. The crossover operator mixes the coordinates of the mutant and target vectors to generate a a so-called trial vector. The mutation and crossover operators are aimed at diversifying the search, providing a wider overview of the search space and a higher probability of localizing the global extremum of the objective function.

The selection operator compares the objective function values of the trial vector and the target vector. The vector with the better objective function value is chosen to become a member of the next generation. This process guarantees that the population size remains constant throughout the algorithm's operation.

In each generation, the best vector is identified to monitor the progress of the search for an optimal solution. The DE algorithm stops when a specific condition (or conditions) is met, for example, reaching a satisfactory value of the optimization criterion, exhaustion of a predefined maximum number of generations, etc.

The DE algorithm can be considered as a modification of CGA. However, its key distinguishing feature is the source of the "noise" used during mutation. Unlike genetic algorithms, which relies on an external random number generator, DE uses an "internal" noise source. This noise is generated from the difference between two or more randomly selected vectors from the current population. This approach allows the algorithm to effectively model the "terrain" of the objective function and quickly pass through ravines. This is why DE is so effective even in complex terrains.

Furthermore, unlike genetic algorithms, in DE each vector in the generation is not compared against all the vectors in the current generation, but only against its counterpart in the current generation which replaces if better fitted.

A computational scheme of the DE algorithm for finding optimal parameters of approximate solutions for boundary value problem (1)–(2) (or problem (3)–(4)) is given below.

1. The generation number $G=0$ is established and the population consisting of vectors $A_i=(a_{1i},\dots,a_{ni})$, $i=1,\dots,Np$, is created. Here vector coordinates a_{ji}, $j=1,\dots,n$, are random real numbers from $[-1,\ 1]$.

2. The values of the objective function $F(V_i)$ is calculated by the formula

$$F(A_i)=\|R(x,y;a_{1i},\dots,a_{ni})\|,\ i=1,\dots,Np, \tag{14}$$

for problem (9), and by the formula

$$F(A_i)=\|\varepsilon(x,y;a_{1i},\dots,a_{ni})\|,\ i=1,\dots,Np, \tag{15}$$

for problem (11), where $\|\cdot\|$ is one of norms (6)–(8). For example,

$$F(A_i)=\sum_{l=1}^{m}R^2(x_l,y_l;a_{1i},\dots,a_{ni}),$$

for uniform norm (6) and the differential residual R.

3. For each target vector A_i, $i=1,\dots,Np$, a mutant vector $\tilde{A}_i=(\tilde{a}_{1i},\dots,\tilde{a}_{ni})$ is generated according to

$$\tilde{A}_i=A_{r_1}+Fm\cdot(A_{r_2}-A_{r_3}),$$

where r_1, r_2 and r_3 are random integer numbers from the interval $[1,Np]$, $r_1\uparrow r_2\uparrow r_3\uparrow i$, Fm is the scaling factor (or the mutation force), a real-valued constant chosen from $(0,2]$.

4. The trial vector $B_i = (b_{1i},\ldots,b_{ni})$ is generated by the formula

$$b_{ji} = \begin{cases} \tilde{a}_{ji}, \text{ if } rand_j \le Cr \text{ or } j = j_{rand}, \\ a_{ji} \text{ otherwise,} \end{cases}$$

where Cr is the crossover constant, $rand_j$ is a random real number from [0,1], $j = 1,...,n$, and index $rand \in 1,2,\ldots,n$ in j_{rand}.

5. If the trial vector B_i has less or equal objective function value $F(B_i)$ than the corresponding target vector A_i, the trial vector will replace the target vector and enter the next generation $G+1$. Otherwise, A_i will remain in the next generation.

6. The DE algorithm stops if one of the following conditions is satisfied:

– the maximum number of generations G_{max} is reached (by default $G_{max} = 200$);

– the objective function value of the best chromosome in a current generation is less than a user-determined constant $\propto$.

If none of the above conditions is fulfilled, the passage to step 3 is carried out.

The DE algorithm performance depends mainly on appropriately choosing its control parameters: the population size (*Np*), the scaling factor (*Fm*) and the crossover constant (*Cr*). The parameter *Np* generally doesn't require fine-tuning. It’s suggested that a reasonable value for *Np* should be in the range of 5*n* to 10*n*, where *n* is the number of problem variables [17].

The scaling factor determining the magnitude of the perturbation during the mutation phase is closely related to the convergence speed. The value of *Fm* is crucial for balancing the algorithm's exploration (searching new areas of the solution space) and exploitation (refining the search in promising areas). A large value of *Fm* (e.g., closer to 1 or greater) results in a larger perturbation, causing the algorithm to take bigger steps. This promotes exploration, helping to avoid local optima by widely sampling the search space. A small value of *Fm* (e.g., closer to 0) results in a smaller perturbation. This promotes exploitation, allowing the algorithm to finely tune solutions and converge on a specific optimum. A *Fm* value typically ranges from 0.4 to 1 [17; 5]. For the given problem, recommended values of *Fm* are between 0.5 and 0.7 [23].

The crossover constant (or crossover rate) takes values in the range $[0, 1]$, acting as a probability. A low *Cr* value (close to 0) results in the trial vector being very similar to the original target vector, with only a few components being inherited from the mutant. This promotes a more local, exploitative search, refining existing solutions. Conversely, a high *Cr* value (close to 1) means that the trial vector will likely inherit most of its components from the mutant vector. This promotes a high degree of exploration, allowing the algorithm to make large changes and potentially escape local optima. For a given problem, recommended values for *Cr* are between 0.9 and 1 [23].

6. Numerical computation results and discussion

In this section, some numerical problems are studied to demonstrate the accuracy and applicability of the proposed algorithms. Results obtained are compared with exact solutions or approximate solutions obtained by other methods.

All the numerical computations were performed using MATLAB platform. Due to the stochastic nature of the CGA and the DE algorithm, ten runs were made for obtaining every result.

Example 1. Consider the following linear BVP:

$$\Delta u = 2x(1-x)\,,\ 0 < x < 1\,,\ 0 < y < 1\,;$$

$$u(0,y) = 0\,,\ u(1,y) = 0\,,\ 0\ ''\ y\ ''\ 1\,;\ u_y(x,0) = 0\,,\ u_y(x,1) = 0\,,\ 0\ ''\ x\ ''\ 1\,.$$

This is a problem on a stationary distribution of temperature in a uniform square plate with a heat source of intensity $2x(x-1)$ acting in its middle, if the coefficient of internal thermal conductivity is equal to 1, the edges $x = 0$ and $x = 1$ of the plate are kept at zero temperature, and the other two edges are thermally insulated [15, p. 148].

For approximate solution of this BVP, we select the following function *v*:

$$v(x) = c_1 x(1-x^2) + c_2 x^3(1-x).$$

This function satisfies the boundary conditions at any values of the parameters c_1 and c_2. Note that the approximate solution is sought as a function of one variable x , since the free term in the differential equation is the function of the variable x alone, and the conditions at the edges $x = 0$ and $x = 1$ do not depend on the variable *y*) [20]. After substituting $v(x)$ into the differential equation, the following differential residual is obtained:

$$R = -6c_1 x + 6c_2 x(1-2x) - 2x(1-x)\,. \qquad (16)$$

To find the optimal values of the parameter c_1 and c_2 for residual (16), we give the uniform grid E_{101} and root mean square norm (7) for the residual R in minimization problem (9). Using the proposed CGA with stop settings $G_{\max} = 50$ and $\mu = 1.0 \cdot 10^{-12}$, the following results

$$c_1 = -0.1666666667,\ c_2 = 0.1666666667,\ \|R\| = 2.4 \cdot 10^{-13}$$

are calculated. In this case, the approximate solution is

$$\begin{aligned} v(x) &= -0.166666666667x(1-x^2) + 0.166666666667x^3(1-x) = \\ &= 0.166666666667(-x^4 + 2x^3 - x). \end{aligned} \tag{17}$$

The exact solution of the considered BVP is $u(x) = (-x^4 + 2x^3 - x)/6$ [15, p. 150]. A comparison of the values for the exact solution $u(x)$ and the approximate solution $v(x)$ at the points of computational domain showed that approximate solution (17) is well consistent with the exact solution $u(x)$ and for absolute error of the inequality $|u - v| \le 7.8 \cdot 10^{-12}$ holds.

For uniform norm (8) of residual (16), the following optimal values of the parameters c_1 and c_2:

$$c_1 = -0.1666666667,\ c_2 = 0.1666666666,\ \|R\| = \max|R| = 4.5 \cdot 10^{-13}$$

are computed using the proposed CGA with the same stop settings. In this case, for the according approximate solution v is well consistent with the exact solution $u(x)$ and the inequality $|u - v| \le 9.0 \cdot 10^{-12}$ holds.

If we choose the DE algorithm (with the stop settings $G_{\max} = 70$, $\mu = 1.0 \cdot 10^{-12}$) and uniform norm (8) to seek the optimal parameter values of differential residual (16), we obtain the following values of c_1, c_2 and $\|R\| \equiv \max|R|$:

$$c_1 = -0.1666666667,\ c_2 = 0.1666666667,\ \|R\| = 4.1 \cdot 10^{-13}.$$

The same optimal values of c_1 and c_2 are computed using the DE algorithm and root mean square norm (7). The value of $\|R\|$ equals $5.8 \cdot 10^{-13}$.

For both norms, the approximate solutions obtained using the DE algorithm can be written as (17).

Thus, the presented results show that the approximate solutions $v(x)$ obtained using the proposed CGA and DE are well consistent with the exact solution $u(x)$.

The average number of objective function evaluations required to obtain results by the DE algorithm, is 1477 for the root mean square norm and 1427 for the uniform norm. The average number of fitness function evaluations needed to obtain results using the CGA, is 3184 for the root mean square norm and 3431 for the uniform norm. The average numbers of fitness function and objective function evaluations allow us to compare the convergence speed of the CGA and the DE algorithm.

Example 2. Consider the following problem of static deflection of a homogeneous rectangular membrane fixed at the edges and unevenly loaded:

$\Delta u = x^2 - 1$ in domain $D = \{ (x,y): -1 < x < 1, -0.5 < y < 0.5 \}$;

$u(x,y) = 0$ on boundary $\Gamma = \{ (x,y): (|x| = 1, -0.5 \le y \le 0.5) \cup (-1 \le x \le 1, |y| = 0.5) \}$

The approximate solution of this BVP is chosen in the form [9]

$$v(x,y) = \frac{x^4}{12} - \frac{x^2}{2} + \sum_{k=1}^{2} c_k \phi_k(x,y), \tag{18}$$

where

$$\phi_1 = x^2 - y^2, \ \phi_2 = x^4 - 6x^2y^2 + y^4 .$$

It is easy to verify that function $v(x)$ exactly satisfies the differential equation for any c_k. According to the second proposed method, function (18) is substituted into the boundary conditions and the boundary condition error ε is obtained.

Note that based on the symmetry of the functions ϕ_k with respect to the coordinate axes, it is sufficient to minimize the boundary condition error ε on the contour $\tilde{\Gamma} = \{ (x,y): (0 \le x \le 1, y = 0.5) \cup (x = 1, 0 \le y \le 0.5) \}$, which constitutes a quarter of the boundary Γ.

To find the optimal values of the parameter c_1 and c_2, we select uniform norm (8) for measuring the error ε and cover the contour $\tilde{\Gamma}$ with the grid $E_{201} = \left\{ \left(x_l, y_l \right) \right\}_0^{200}$, where $y_l = 0.5$, $x_l = l \cdot 0.01$ ($l = \overline{0,100}$) and $x_l = 1$, $y_l = \left(200 - l \right) \cdot 0.005$ ($l = \overline{101,200}$).

Applying the proposed DE algorithm with stop settings $G_{\max} = 70$ and $\mu = 0.1$, the following results

$c_1 = 0.38786,\ c_2 = -0.072905$, $\|\varepsilon\| = 0.101708$

are computed. Thus, the maximum absolute value of ε is 0.101708. For these parameter values, the approximate solution is written as

$$v(x,y) = \frac{x^4}{12} - \frac{x^2}{2} + 0.38786(x^2 - y^2) - 0.072905(x^4 - 6x^2y^2 + y^4) . \tag{19}$$

According to the maximum principle for harmonious functions the inequality $|u-v| \le 0.10171$ holds at all points of the domain $D \cup \Gamma$.

For root mean square norm (7), using the DE algorithm the following results

$$c_1 = 0.512666,\ c_2 = -0.118308,\ \|\varepsilon\| = 0.066427$$

are obtained. The according approximate solution is

$$v(x,y) = \frac{x^4}{12} - \frac{x^2}{2} + 0.512666(x^2 - y^2) - 0.118308(x^4 - 6x^2y^2 + y^4). \quad (20)$$

The maximum absolute error of approximate solution (20) equals 0.13556 on $\tilde{\Gamma}$. This is 1.3 times more than the error in the case of the uniform norm.

For both norms, the number of objective function evaluations required to obtain results is 1490, because the algorithm is stopped when the first terminal condition fulfill, i.e. maximum number of generations G_{max} is reached.

<u>Example 3</u>. It is necessary to find a solution to the problem of torsion of a beam with cross-section D:

$$\Delta u(x,y) = -1 \text{ in domain } D, \quad (21)$$

$$u(x,y) = 0 \text{ on boundary } \Gamma \text{ of domain } D, \quad (22)$$

The boundary Γ consists of two line segments $y = \pm 1$ for $|x|'' 1$ and two semicircular arcs of radius 1 with centers at points $(-1, 0)$ and $(1, 0)$ for $|x| \ge 1$ [3, p. 365].

The approximate solution $v(x,y)$ for BVP (21)-(22) is chosen in the form:

$$v(x,y;c_1,\dots,c_n) = -\frac{x^2+y^2}{4} + \sum_{k=1}^{n} c_k \phi_k(x,y), \quad (23)$$

$$\phi_k(x,y) = \mathrm{Re}(x+iy)^{2k-2},\ k = \overline{1,n}.$$

For $k = \overline{1,6}$, the functions ϕ_k can be written as:

$$\phi_1(x,y) = 1,\ \phi_2(x,y) = x^2 - y^2,\ \phi_3(x,y) = x^4 - 6x^2y^2 + y^4,$$

$$\phi_4(x,y) = x^6 - 15x^4y^2 + 15x^2y^4 - y^6,$$

$$\phi_5(x,y) = x^8 - 28x^6y^2 + 70x^4y^4 - 28x^2y^6 + y^8,$$

$$\phi_6(x,y) = x^{10} - 45x^8y^2 + 210x^6y^4 - 210x^4y^6 + 45x^2y^8 - y^{10}.$$

The function $v(x,y)$ exactly satisfies differential equation (21) for any values of coefficients c_k. According to the second method, function (23) is substituted in (22) and the boundary condition error ε is obtained. As in Example 2, taking the symmetry reasons, it is sufficient to minimize the ε on the contour $\tilde{\Gamma}$, which constitutes a quarter of the boundary Γ.

To find the optimal values of the coefficients c_k, we choose uniform norm (8) for minimizing the error ε and cover the contour $\tilde{\Gamma}$ with the grid $E_{51} = \{(x_l, y_l):\ l = \overline{0,50}\}$, where $x_l = l \cdot 0.05,\ y_l = 1,\quad l = \overline{0,19}$ and $y_l = \cos(l-20)\alpha,\quad x_l = 1 + \sin(l-20)\alpha,\quad l = \overline{20,50},\quad \alpha = \pi/60$. Using the uniform norm to solve BVPs for elliptic PDEs has the advantage that the principle of reaching a maximum at the boundary of the domain is fulfilled for the difference modulus $u - v$ (see Section 3).

Applying the proposed DE algorithm the optimal values of the coefficients c_k and the errors ε for approximate solution (23) are computed. The results obtained for $n = 2, 3, 5, 7$ are presented in Table 1. For example, for n=6, approximate solution (23) is written as

$$v(x,y) = -\frac{x^2 + y^2}{4} + 0.44240961 + 0.18099186\phi_2(x,y) - 0.01342571\phi_3(x,y) +$$

$$+0.00044738\phi_4(x,y) + 0.00018263\phi_5(x,y) - 0.00002851\phi_6(x,y).$$

Note that the coefficient c_1 gives an approximate value for the function $u(x,y)$ at the midpoint and the estimate $c_1 - \varepsilon \le u(0,0) \le c_1 + \varepsilon$ holds [3, p. 366]. For example, $0.4402 '' u(0,0)'' 0.4447$ in the case n=6.

Table 1

n	Coefficients	Error ε
2	c_1= 0.4665779, c_2= 0.150	0.066578
3	c_1= 0.44916077, c_2= 0.17384999, c_3=-	0.015318
4	c_1= 0.44243986, c_2= 0.18134097, c_3=-	0.003877
5	c_1= 0.442260, c_2= 0.18117759, c_3=-	0.003681
6	c_1= 0.44240961, c_2= 0.18099186, c_3= -	0.002244

The number of objective function evaluations required to obtain results is 650 for *n*=2, 2014 for *n*=3, 3320 for *n*=4, 4640 for *n*=5, and 7380 for *n*=6.

When we choose CGA to calculate the optimal values of the coefficients of function (23), the average numbers of fitness function evaluations were approximately four times larger than for the DE algorithm.

In paper [18], the values of the coefficients c_k for the function $v(x, y)$ and the approximation errors ε were obtained using the algorithm of the best uniform approximation for many-variable functions by generalized polynomials. The results computed by this algorithm practically coincide with the results obtained using the DE algorithm and CGA. It should be noted that the algorithm of the best uniform approximation by generalized polynomials is quite sophisticated and can be applied only in the case of linear inclusion of coefficients c_k in the approximate solution $v(x, y)$, while the DE algorithm and CGA are simple in realization and can be used in both linear and nonlinear cases.

<u>Example 4</u>. Consider the problem of the stationary temperature distribution in thin plate having the shape of the circular sector $0 \le \phi \le \pi/3$, $0 \,''\, r \,''\, 1$, the radii of which are maintained at zero temperature, and the arc of the circle is maintained at temperature $f(\phi)$. The following mathematical model corresponds to this problem in the polar coordinate system [10, p. 129]:

$$u_{rr} + \frac{1}{r}u_r + \frac{1}{r^2}u_{\phi\phi} = 0\,,\ 0 < r < 1\,,\ 0 < \phi < \pi/3\,, \tag{24}$$

$$u(r,\,0) = u(r,\,\pi/3) = 0\,,\ 0 \,''\, r \,''\, 1\,, \tag{25}$$

$$u(1,\,\phi) = \frac{9}{8}\pi\phi\left(\frac{\pi}{3} - \phi\right),\ 0 \le \phi \le \pi/3\,. \tag{26}$$

The approximate solution $v(r, \phi)$ for BVP (24)–(25) is selected in the form

$$v(r,\phi) = \sum_{k=1}^{n} c_k r^{3k} \sin 3k\phi\,. \tag{27}$$

For any values of coefficients c_k, function (27) satisfies differential equation (24) and boundary condition (25). When substituting the function v into boundary condition (25), the error ε is formed:

$$\varepsilon(\phi; c_1,\ldots,c_n) = v(1,\phi) - w(\phi) \ \ \varepsilon(\phi; c_1,\ldots,c_n) = \sum_{k=1}^{n} c_k \sin 3k\phi - \frac{9}{8}\pi\phi\left(\frac{\pi}{3} - \phi\right). \quad (28)$$

To compute the optimal values of the coefficients $c_1,\ldots,c_n$, the grid $E_{101} = \{(r_l,\phi_l):\ r_l = 1, \phi_l = (l-1)\pi/300, l = 1,\ldots, m\}$ is given and uniform norm (8) for the boundary condition error ε is selected.

Table 2 shows the optimal values of the coefficients c_k and the errors for approximate solution (27) computed using the DE algorithm for $n = 3, 5, 7$.

Note that for all coefficients c_k with even index *k*, zero values were obtained. For example, in the case $n = 3$ the coefficient c_2 equals zero and the approximate solution is written as

$$v(r,\phi) = 1.0002461 r^3 \sin 3\phi + 0.0406731 r^9 \sin 9\phi .$$

The number of objective function evaluations required to obtain results in the DE algorithm is 1580 for $n = 3$, 3320 for $n = 5$ and 5550 for $n = 7$.

Table 2

n	The DE algorithm		Fourier's method	
	coefficients	**error**	**coefficients**	**error**
3	c_1=1.0002461 c_3=0.0406731	0.00937	c_1=1 c_3=0.0370370	0.0115
5	c_1=1.0000409 c_3=0.0371963 c_5=0.0100136	0.00391	c_1=1 c_3=0.0370370 c_5=0.008	0.0052
7	c_1=1.0000050 c_3=0.0370805 c_5=0.0080929 c_7=0.0041670	0.00210	c_1=1 c_3=0.0370370 c_5=0.008 c_7=0.0029155	0.0029

The solution to problem (24)–(26), obtained by the Fourier method, is written as [10, p. 130].

$$u(r,\phi) = \sum_{k=0}^{\infty} \frac{1}{(2k+1)^3} r^{3(2k+1)} \sin 3(2k+1)\phi . \quad (29)$$

In Table 2, the coefficients c_k and the errors for approximate solutions formed by the partial sum consisted of the first *n* terms of infinite sum (29) are also presented.

A comparison of the results given in Table 2 shows that the error of approximate solution (27) with the coefficients obtained using the DE algorithm is smaller than the error of approximate solutions with Fourier's coefficients. This is especially noticeable for small n.

Conclusions

In the paper, two methods for obtaining approximate solutions to BVPs for elliptic PDEs were presented. In both methods, a BVP is formulated as an optimization problem. In the first method, it is the problem of differential residual minimization. In the second method, it is the problem of boundary condition error minimization. To solve these minimization problems, the CGA and the DE algorithm were employed.

As evolutionary algorithms, CGA and DE differ from conventional optimization methods in several key aspects. They operate on a population of candidate solutions rather than a single solution, they rely solely on the objective function without requiring auxiliary information, and they use probabilistic rather than deterministic transition rules.

The CGA and the DE algorithm were described in detail for finding optimal parameters of approximate solutions to BVPs for elliptic PDEs. The algorithms can be applied to both linear and nonlinear inclusions of parameters c_k in approximate solutions. Besides, various error norms (uniform, quadratic, root mean square) can be incorporated into the definitions of objective functions.

The performance of CGA and DE strongly depends on a choice of settings for control parameters such as the population size, the scaling factor, the crossover constant, and the mutation probability. The recommended control parameter settings were given.

To demonstrate the accuracy and applicability of the proposed algorithms, several examples of solving BVPs were presented. Comparison between the exact solution and the approximate solution (examples 1 and 4), obtained by the algorithms, showed a high level of agreement. It was also shown that the errors of approximate solutions computed by DE and CGA were comparable to those obtained by more sophisticated algorithms. In terms of computational efficiency, DE required fewer objective function evaluations than CGA, indicating faster convergence.

Thus, the proposed two methods for obtaining BVP approximate solutions using the CGA or the DE algorithms can be considered as viable alternatives to existing approximate analytical methods for solving BVPs for elliptic PDEs.

We see the prospects for further research in developing the proposed approach for obtaining approximate solutions to initial-boundary value problems for partial differential equations.

References:

1. Abu-Arqub O., Abo-Hammour Z., Momani Sh. (2014). Application of continuous genetic algorithm for nonlinear system of second-order boundary value problems. *Applied Mathematics & Information Sciences,* vol. 8(1), pp. 235–248. https://doi.org/10.12785/amis/080129.

2. Bhargava S. (2013). A note on evolutionary algorithms and its applications. *Adults Learning Mathematics: An International Journal*, vol. 8(1), pp. 31–45.

3. Collatz L. (1966). *Functional analysis and numerical mathematics*. New York: Academic Press.

4. Collatz L. & Krabs W. (1973). *Approximationstheorie: Tschebyscheffsche Approximation mit Anwendungen*. Stuttgart: Vieweg+Teubner Verlag Wiesbaden.

5. Das S. & Suganthan P. (2011). Differential evolution: A survey of the state-of-the-art. *IEEE Transactions on Evolutionary Computation*, vol. 5(1), pp. 4–31.

6. Fateh M. F., Zameer A., Mirza N. M., Mirza S. M., Raja M. A. Z. (2017). Biologically inspired computing framework for solving two-point boundary value problems using differential evolution. *Neural Computing and Applications*, vol. 28, pp. 2165–2179. https://doi.org/10.1007/s00521-016-2185-z

7. Herrera F., Lozano M., Verdegay J.L. (1998). Tackling real-coded genetic algorithms: operators and tools for the behaviour analysis. *Artificial Intelligence Review*, vol. 12(4), pp. 265–319.

8. Holland J. H. (1975). *Adaptation in natural and artificial systems*, Ann Arbor: Univ. of Michigan Press.

9. Kalenchuk-Porkhanova A. O. & Vakal L. P. (2010). Zastosuvannia naikrashchoi chebyshovskoi aproksymatsii dlia modeliuvannia deiakykh fizychnykh protsesiv [Using the best Chebyshev approximation for modeling some physical processes]. *Matematychne ta kompiuterne modeliuvannia. Seriia: Tekhnichni nauky – Mathematical and Computer* Modelling, Series: Technical sciences, no. 4, pp. 111–118. (in Ukrainian)

10. Lavrenchuk V. P., Ivanyshyn S. D., Sovin P. A., Dron V. S. (1998). *Rivniannia matematychnoi fizyky (metodychnyi posibnyk)* [Equations of mathematical physics (methodological guide)]. Chernivtsi: Ruta. (in Ukrainian)

11. Lu X., Tang K., Sendhoff B., Yao X. (2014). A new self-adaptation scheme for differential evolution. *Neurocomputing*, no. 146, pp. 2–16.

12. Mallipeddi R., Suganthan P. N., Pan Q. K., Tasgetiren M. F. (2011). Differential evolution algorithm with ensemble of parameters and mutation strategies. *Applied Soft Computing*, vol. 11(2), pp. 1679–1696.

13. Nasim A., Burattini L., Fateh M. F., Zameer A. (2019). Solution of linear and non-linear boundary value problems using population-distributed parallel differential evolution. *Journal of Artificial Intelligence and Soft Computing Research*, vol. 9(3), pp. 205–218. https://doi.org/10.2478/jaiscr-2019-0004

14. Perestiuk M. O. & Marynets V. V (2001). *Teoriia rivnian matematychnoi fizyky* [Theory of mathematical physics equations]. Kyiv: Lybid. (in Ukrainian)

15. Perestiuk M. O., Marynets V. V., Reho V. L. (2012). *Zbirnyk zadach z matematychnoi fizyky* [Collection of problems in mathematical physics]. Kamianets-Podilskyi: Aksioma. (in Ukrainian)

16. Price K., Storn R., Lampinen J. (2005). *Differential evolution: a practical approach to global optimization*. New York: Springer-Verlag.

17. Storn R. & Price K. (1997). Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, vol. 11, pp. 341–35.

18. Vakal L. P. (2010). Rozviazannia kraiovykh zadach z vykorystanniam prohramnykh zasobiv chebyshovskykh nablyzhen [Solving boundary value problems using software for Chebyshev approximation]. *Kompiuterni Zasoby, Merezhi ta Systemy – Computer Means, Networks and Systems*, no. 9, pp. 47–53. (in Ukrainian)

19. Vakal L. P. (2013). Henetychni alhorytmy dlia Chebyshovskoi aproksymatsii [Genetic algorithms for Chebyshev approximation]. *Kompiuterni Zasoby, Merezhi ta Systemy – Computer Means, Networks and Systems*, no. 12, pp. 20–26. (in Ukrainian)

20. Vakal L. P. (2015). Using genetic algorithm for solving boundary value problems. *Journal of Automation and Information Sciences*, vol. 47(8), pp. 52–62. https://doi.org/10.1615/JAutomatInfScien.v47.i8.50

21. Vakal L. P. (2016). Solving uniform nonlinear approximation problem using continuous genetic algorithm. *Journal of Automation and Information Sciences*, vol. 48(6), pp. 49–59. https://doi.org/10.1615/JAutomatInfScien.v48.i6.50

22. Vakal L. P. & Vakal Ye. S. (2020). Rozviazannia kraiovykh zadach dlia zvychainykh dyferentsialnykh rivnian za alhorytmom dyferentsialnoi evoliutsii [The solution of boundary value problems for ordinary differential equations using the differential evolution algorithm]. *Matematychni mashyny i systemy – Mathematical Machines and Systems*, no. 1, pp. 43–52. https://doi.org/10.34121/1028-9763-2020-1-43-52 (in Ukrainian)

23. Vakal L. & Vakal E. (2024). Differential evolution for best uniform spline approximation. In D. Koroliouk, S. Lyashko and N. Limnios (Eds.) *Computational methods and mathematical modeling in cyberphysics and engineering applications 1*. John Wiley & Sons, Inc., pp. 355–366. https://doi.org/10.1002/9781394284344.ch14

24. Vakal L. P., Vakal Ye. S., Dovgiy B. P. (2021). Solving Fredholm integral equations of the second kind using differential evolution. *Computer Science and Applied Mathematics*, no. 1, pp. 15-21. https://doi.org/10.26661/2413-6549-2021-1-02

25. Varadarajan M. & Swarup K. S. (2008). Differential evolution approach for optimal reactive power dispatch. *Applied Soft Computing*, vol. 8(4), pp. 1549–1561.

26. Virchenko N. O. (1997). *Osnovni metody rozviazannia zadach matematychnoi fizyky* [Basic methods for solving mathematical physics problems]. Kyiv: KPI. (in Ukrainian)

27. Wright A. H. (1991). Genetic algorithms for real parameter optimization. In G. J. E. Rawlins (ed.), *Foundations of Genetic Algorithms*. San Mateo, CA: Morgan Kaufmann, pp. 205–218.