Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

Differential evolution based computation intelligence solver for elliptic partial differential equations

Muhammad Faisal FATEH[1], Aneela ZAMEER[2], Sikander M. MIRZA[3], Nasir M. MIRZA[3], Muhammad Saeed ASLAM[†‡4], Muhammad Asif Zahoor RAJA[5]

[1]*Department of Electrical Engineering, Pakistan Institute of Engineering and Applied Sciences (PIEAS), Islamabad 45650, Pakistan*

[2]*Department of Computer and Information Sciences, Pakistan Institute of Engineering and Applied Sciences (PIEAS), Islamabad 45650, Pakistan*

[3]*Department of Physics and Applied Mathematics, Pakistan Institute of Engineering and Applied Sciences (PIEAS), Islamabad 45650, Pakistan*

[4]*School of Electrical and Electronic Engineering, University of Adelaide, Adelaide 5005, Australia*

[5]*Department of Electrical and Computer Engineering, COMSATS University Islamabad, Attock Campus, Attock 43600, Pakistan*

[†]E-mail: muhammad.aslam@adelaide.edu.au

Received May 1, 2019; Revision accepted July 12, 2019; Crosschecked Oct. 8, 2019

Abstract: A differential evolution based methodology is introduced for the solution of elliptic partial differential equations (PDEs) with Dirichlet and/or Neumann boundary conditions. The solutions evolve over bounded domains throughout the interior nodes by minimization of nodal deviations among the population. The elliptic PDEs are replaced by the corresponding system of finite difference approximation, yielding an expression for nodal residues. The global residue is declared as the root-mean-square value of the nodal residues and taken as the cost function. The standard differential evolution is then used for the solution of elliptic PDEs by conversion to a minimization problem of the global residue. A set of benchmark problems consisting of both linear and nonlinear elliptic PDEs has been considered for validation, proving the effectiveness of the proposed algorithm. To demonstrate its robustness, sensitivity analysis has been carried out for various differential evolution operators and parameters. Comparison of the differential evolution based computed nodal values with the corresponding data obtained using the exact analytical expressions shows the accuracy and convergence of the proposed methodology.

Key words: Differential evolution; Boundary value problems; Partial differential equation; Finite difference scheme; Numerical computing
https://doi.org/10.1631/FITEE.1900221 **CLC number:** O175

1 Introduction

Partial differential equations (PDEs) can be found in a wide variety of areas of practical interest, e.g., blood flow in human veins (Spiller et al., 2017), air flow past vehicles (Carlson et al., 2017), dynamic behavior of structures (Machado et al., 2016), magnetic resonance imaging (Chang and Tsai, 2014), thermal inhibition of tumors (Oden et al., 2016), spatial dynamics with thermo-hydraulic behavior modeling (Carotenuto et al., 2016), nanotechnology (Raja et al., 2016b), fluid dynamics (Raja et al., 2016a), astronomy (Ahmad et al., 2016), thermodynamics (Ahmad et al., 2017), and bioinformatics

‡ Corresponding author
ORCID: Muhammad Saeed ASLAM, http://orcid.org/0000-0001-6219-4910; Muhammad Asif Zahoor RAJA, http://orcid.org/0000-0001-9953-822X

(Raja et al., 2017b).

In the past, research effort focused on finding the exact solution of PDEs (Gao et al., 2017; Yin et al., 2018; Hua et al., 2019). Attempts have also been made for theoretical analysis of the solutions and their applications (Wang et al., 2016, 2017; Dai et al., 2017a, 2017b; Ding et al., 2017; Zhang et al., 2017). Since exact and closed-form analytical solutions for PDEs can be found for only a class of elliptic PDEs, efforts have been made towards finding their approximate numerical solutions. The later ones have been found to match their numerical counterparts to a high degree of accuracy. Due to the wide application of PDEs, fast and reliable solvers are of prime importance in science and engineering. Various available numerical techniques offer ease of use, coupled with the advent of high-speed digital computers, making them the method of choice for the solution of PDEs. However, most of these techniques rely on serial algorithms, which do not fully exploit the computing power of parallel hardware.

The elliptic PDEs with specified boundary conditions have a unique solution $o(h^2)$. Consequently, when multiple proposed solutions progress through iterations, the one closest to that unique solution is expected to have the dominant rate of convergence due to its lowest value of the cost function. In contrast, non-unique scenarios may lead to poor convergence rate due to flipping among multiple alternate solutions.

A literature review reveals a rich diversity of numerical techniques that have been developed over time for the solution of elliptic PDEs. A solution of a Laplace equation posed as a Dirichlet problem using five-point approximation was first proposed by Courant et al. (1928). Subsequently, Gerschgorin (1930) formulated error bounds for elliptic PDEs and these efforts were extended further by various researchers (Motzkin and Wasow, 1952; Bramble et al., 1969). For positive second-order elliptic PDEs, various iterative schemes were developed, including Jacobi, Gauss-Seidel, Frankel-Yound, and Peaceman-Rachford (Peaceman and Rachford, 1955).

In the current study, we take only second-order elliptic equations which have homogeneous and non-homogeneous forms. Homogeneous elliptic PDEs are usually called the Laplace equation, and occur in many fields, such as electrostatics, gravitation, steady state flow of inviscid fluids, and steady state heat conduction (Shao and Faltinsen, 2014; Gilbarg and Trudinger, 2015; Vétois, 2016; Webster, 2016). Similarly, the Poisson equation appears in electrostatics, elasticity theory, and elsewhere (Selvadurai, 2000; Zhao et al., 2013; Li et al., 2014; Marichal et al., 2014; Taleei and Dehghan, 2014).

2 Materials and methods

In this study, elliptic PEDs are solved by differential evolution and results are compared with the corresponding exact solutions for evaluation of the technique.

2.1 Problem formulation

A standard elliptic boundary value problem is considered as follows:

$$\begin{cases} \bigtriangledown^2 u = f(x,y), \\ u(x,y_{\min}) = f_1(x,y), \\ u(x,y_{\max}) = f_2(x,y), \\ u(x_{\min},y) = f_3(x,y), \\ u(x_{\max},y) = f_4(x,y), \end{cases} \tag{1}$$

where $x \in [x_{\min}, x_{\max}]$ and $y \in [y_{\min}, y_{\max}]$.

The domain of interest is subdivided into $(N+1)$ equal intervals in both variables of space with node spacing given by h in x and l in y such that

$$\begin{cases} x_j = x_{\min} + jh, \\ y_i = y_{\min} + il, \\ u_{i,j} = y(x_j, y_i), \\ u_{i,j-1} = y(x_{j-1}, y_i), \\ u_{i,j+1} = y(x_{j+1}, y_i), \\ u_{i-1,j} = y\left(x_j, y_{i-1}\right), \\ u_{i+1,j} = y(x_j, y_{i+1}). \end{cases} \tag{2}$$

Using the three-point central finite difference scheme, the first- and second-order derivatives on a finite grid become

$$\frac{\partial^2 u}{\partial x^2} = \frac{u_{i,j-1} - 2u_{i,j} + u_{i,j+1}}{h^2}, \tag{3}$$

$$\frac{\partial^2 u}{\partial y^2} = \frac{u_{i-1,j} - 2u_{i,j} + u_{i+1,j}}{l^2}. \tag{4}$$

The nodal residual (r_i) can then be defined as

$$r_i = \nabla_i^2 u - f\left(x_i, y_j\right). \tag{5}$$

The cost function represented by the global residue (R) is given by

$$R = \sqrt{\sum_{i=1}^{n} r_i^2}. \tag{6}$$

The original elliptic PDE is now posed as a cost function minimization problem.

2.2 Differential evolution

Storn and Price (1997) first proposed the differential evolution algorithm. The salient features of differential equations (DEs) include solvability of non-differentiable, nonlinear multi-objective functions and/or constrained optimization problems with inherently parallel methodology, ease of implementation, robustness, and good convergence capability without getting trapped in local minima. In this study, a novel method of differential evolution is used to solve elliptic PDEs.

DEs, similar to other evolutionary algorithms, have various phases including encoding, cost evaluation, initialization, selection, and reproduction (Guo et al., 2016; Das et al., 2017; Fateh et al., 2017; Raja et al., 2017a, 2018). Our work is based on a fitness function that approximates second-order derivatives. Consequently, the error decrement follows the $o(h^2)$ process.

1. Encoding

Optimization algorithms are widely used for optimization of bit strings and vectors. However, the problem definition asks for the solution of two-dimensional (2D) equations; therefore, a potential solution should take the form of a 2D matrix or array. Following this, the chromosomes or candidate solutions in this method are 2D matrices, where rows and columns represent the two variables in space and the value at each cell represents the solution.

2. Initialization

The initial population for the algorithm is constructed using a three-dimensional (3D) array, where the first two indices represent the rows and columns and the third specifies the chromosome number. Initially, for each such node specified by (i, j, k), a uniform random number is generated in the range (min, max), where "min" and "max" represent the minimum and maximum values from the specified boundary values, respectively.

3. Fitness evaluation

The fitness evaluation based on the overall nodal residual error is described in the problem formulation. Individuals with small values of overall residuals have better/higher fitness values.

4. Trial vector generation

In DEs, trial vectors are used in crossover with corresponding parents to yield offspring. For each member of the parent population, a trial vector is calculated using the following relationship (Gilbarg and Trudinger, 2015; Webster, 2016):

$$\boldsymbol{T}_i = \boldsymbol{P}_j + \beta(\boldsymbol{P}_k - \boldsymbol{P}_l), \tag{7}$$

where $\boldsymbol{T}_i$ is the trial vector corresponding to the i^{th} parent, $\boldsymbol{P}_j$ is the target vector, and $\boldsymbol{P}_k$ and $\boldsymbol{P}_l$ are two randomly selected parent vectors. The parameter β is a scale factor that controls the amplification of the DE process. Clearly, all indices (i, j, k, and l) must be different from each other. A more sophisticated version of the trial vector generation procedure used in this work is

$$\boldsymbol{T}_i = \boldsymbol{P} + \beta \sum_{j=1}^{n_v} (\boldsymbol{P}_{kj} - \boldsymbol{P}_{jj}), \tag{8}$$

where $n_v = 2$, $\boldsymbol{P}_{kj}$ and $\boldsymbol{P}_{jj}$ are two random vectors in the j^{th} difference vector, and $\boldsymbol{P}$ is the best individual in the population.

5. Crossover operation

In this step, crossover of trial vectors with their corresponding parents is carried out for the generation of offspring. In each such crossover step, one offspring is created. For crossover operation, the operators used in this work include exponential crossover and binomial crossover.

Adjacent crossover points are selected and the starting crossover point is selected randomly. A reasonably high value of the crossover probability has been used in this work, since empirical study has shown its effectiveness in improving the convergence rate. The exponential crossover is used for one-dimensional (1D) chromosome or vectors. However, the solution of elliptic PDEs is represented using a space grid or a matrix type representation. To tackle the 2D crossover problem, a variant of the exponential function is introduced. A random number is selected from the indices in the vertical or horizontal direction, and each subsequent row or column is replaced by that of the trial vector until all the rows or columns are replaced or the random number token exceeds the crossover probability.

6. Replacement operator

The greedy selection operator has been employed here. The cost value of offspring is compared with that of the corresponding parent, and the one with a lower cost value is used in the next generation. Consequently, DEs keep on improving the candidate solution through competitive evolution.

7. Termination

Any one of the following can be used to terminate the DE cycles:

(1) Stagnation

The cost value of the best individual remains unchanged for a user-specified number of generations. Alternatively, the improvement in the cost value over a user-specified number of generations becomes smaller than some threshold value.

(2) Cycle limits

The number of generations reaches some user-specified maximum number of generations.

The best individual represents the optimal solution at the termination of DE cycles. At the start, the population is initialized using a suitable scheme. Subsequently, trial vectors are generated using Eq. (8) and crossover is performed between parents and the corresponding trial vectors. After fitness evaluation of offspring, a replacement operation is carried out using the greedy criteria. This process is repeated until the termination criterion is satisfied.

3 Case studies

We have implemented the methodology of differential evolution on the following five benchmark studies of elliptic PDEs and employed the exact solution for comparison.

Case 1: Laplace equation with Dirichlet boundary conditions:

$$\begin{cases} \nabla^2 u = 0, & 1 < x, y < 9, \\ u(x,1) = x, & 1 \le x \le 9, \\ u(x,9) = 10 - x, & 1 \le x \le 9, \\ u(1,y) = y, & 1 \le y \le 9, \\ u(9,y) = 10 - y, & 1 \le y \le 9. \end{cases} \tag{9}$$

Exact solution:

$$u(x,y) = \frac{1}{4}(5x + 5y - xy - 5).$$

Case 2: Laplace equation with Neumann boundary conditions:

$$\begin{cases} \nabla^2 u = 0, & 1 < x, y < 9, \\ u(x,1) = x, & 1 \le x \le 9, \\ u(x,9) = 9, & 1 \le x \le 9, \\ u(1,y) = y, & 1 \le y \le 9, \\ \dfrac{\partial u(9,y)}{\partial x} = 0, & 1 \le y \le 9. \end{cases} \tag{10}$$

Exact solution:

$$u(x,y) = y.$$

Case 3: Linear Poisson equation with Dirichlet boundary conditions:

$$\begin{cases} \dfrac{\partial^2 u(x,y)}{\partial x^2} + \dfrac{\partial^2 u(x,y)}{\partial y^2} = 4, & 0 < x < 1,\ 0 < y < 2, \\ u(x,0) = x^2, & 0 \le x \le 1, \\ u(x,2) = (x-2)^2, & 0 \le x \le 1, \\ u(0,y) = y^2, & 0 \le y \le 2, \\ u(1,y) = (y-1)^2, & 0 \le y \le 2. \end{cases} \tag{11}$$

Exact solution:

$$u(x,y) = (x-y)^2.$$

Case 4: Nonlinear Poisson equation I with Dirichlet boundary conditions:

$$\begin{cases} \dfrac{\partial^2 u(x,y)}{\partial x^2} + \dfrac{\partial^2 u(x,y)}{\partial y^2} = x\mathrm{e}^y, & 0 < x < 2,\ 0 < y < 1, \\ u(x,0) = x^2, & 0 \le x \le 2, \\ u(x,1) = \mathrm{e}^x, & 0 \le x \le 2, \\ u(0,y) = 0, & 0 \le y \le 1, \\ u(2,y) = 2\mathrm{e}^y, & 0 \le y \le 1. \end{cases} \tag{12}$$

Exact solution:

$$u(x,y) = x\mathrm{e}^y.$$

Case 5: Nonlinear Poisson equation II with Dirichlet boundary conditions:

$$\begin{cases} \dfrac{\partial^2 u(x,y)}{\partial x^2} + \dfrac{\partial^2 u(x,y)}{\partial y^2} = -\cos(x+y) \\ \qquad +\cos(x-y), & 0 < x < \pi,\ 0 < y < \pi/2, \\ u(x,0) = \cos x, & 0 \le x \le \pi, \\ u(x,\pi/2) = 0, & 0 \le x \le \pi, \\ u(0,y) = \cos y, & 0 \le y \le \pi/2, \\ u(\pi,y) = -\cos y, & 0 \le y \le \pi/2. \end{cases} \tag{13}$$

Exact solution:

$$u(x,y) = \cos x \cos y.$$

4 Experimental results and comparison

The differential evolution based algorithm is employed for the solution of various linear and nonlinear second-order 2D elliptic PDEs with boundary conditions. There are two types of elliptic PDEs, namely Laplace and Poisson equations with boundary conditions. Boundary conditions for such problems are usually of the Dirichlet or Neumann type, or mixed. We consider the results in different categories.

For the Dirichlet boundary condition, the values of the dependent variable are specified at the four boundaries of the grid. In the case of the Neumann boundary condition, the solution is provided at three boundaries, while the 4^{th} boundary condition is expressed as a derivative of the solution with respect to a particular space variable. The population size (nPop) is kept at around 300 individuals pertaining to the large number of unknown nodes, and the crossover probability (p_c) is kept at a very high value of 0.99 to speed up the convergence; as per requirements of the exponential crossover, the scaling factor is kept at 0.5. The total number of generations (nGen) with the upper bound of time is taken to be 3000, and is used as an input of the differential evolution. In each case, the domain of interest is discretized into a 9×9 2D grid in DEs.

The differential evolution iterations are terminated when the value of the cost function associated with the best individual meets the convergence limit of $\varepsilon = 10^{-4}$, or when the iteration count exceeds maxGen=3000. The platform used for these calculations is Intel Core-i7 2.10 GHz CPU. The results presented below are an average of 12 simulations for each case being solved by DEs.

4.1 Results and analysis of DE solvers for the Laplace equation

DEs have been implemented for the first two cases of Laplace equations with Dirichlet and Neumann boundary conditions, separately.

1. Dirichlet boundary conditions

The results in the form of the experimental value of solution (x, y), and its absolute error from the corresponding value of the exact solution, are displayed in Tables 1 and 2 using DEs, respectively.

It can be seen that the corresponding values stay within the defined limits. The results seem to be quite in agreement with the corresponding exact solution, as is obvious from the error values of both solvers. In case of DEs, results are obtained with

Table 1 Proposed solution for $u(x, y)$ for $1 \leq x, y \leq 9$ with $h = 1$ in case 1 (Laplace equation with Dirichlet boundary conditions)

Node	Experimental value								
	1	2	3	4	5	6	7	8	9
1	1	2	3	4	5	6	7	8	9
2	2	2.7500	3.5000	4.2500	5.0000	5.7500	6.5000	7.2500	8
3	3	3.5001	4.0001	4.5001	5.0001	5.5001	6.0000	6.5000	7
4	4	4.2500	4.5001	4.7501	5.0001	5.2501	5.5001	5.7500	6
5	5	5.0001	5.0001	5.0001	5.0001	5.0001	5.0001	5.0001	5
6	6	5.7500	5.5001	5.2501	5.0001	4.7501	4.5001	4.2501	4
7	7	6.5000	6.0000	5.5000	5.0001	4.5001	4.0001	3.5001	3
8	8	7.2500	6.5000	5.7500	5.0000	4.2500	3.5000	2.7500	2
9	9	8	7	6	5	4	3	2	1

Table 2 Comparison with exact solution $u(x, y)$ for $1 \leq x, y \leq 9$ with $h = 1$ through the absolute error in case 1 (Laplace equation with Dirichlet boundary conditions)

Node	Absolute error ($\times 10^{-5}$)									Node	Absolute error ($\times 10^{-5}$)								
	1	2	3	4	5	6	7	8	9		1	2	3	4	5	6	7	8	9
1	0	0	0	0	0	0	0	0	0	6	0	3.4	8.7	10	13	14	13	7.7	0
2	0	3.3	2.5	4.3	4.1	1.7	0.67	0.76	0	7	0	0.9	2.7	2.4	5.0	6.7	7.3	5.9	0
3	0	5.2	8.3	8.1	7.4	5.3	3.9	1.6	0	8	0	1.6	0.48	1.0	0.92	2.2	3.0	4.3	0
4	0	2.9	9.4	12	10	9.4	6.8	2.6	0	9	0	0	0	0	0	0	0	0	0
5	0	5.4	10	15	16	15	11	6.6	0										

a mean error of 3.84×10^{-5} from the fitness value, the number of generations where the algorithm converges is 925 (Fig. 1), and the total CPU time for this case is 73.28 s. Graphs in the form of $u(x, y)$ in a 3D surface plot of absolute errors are demonstrated in Fig. 2.

2. Neumann boundary conditions

The next case is chosen to be a Laplace equation with Neumann boundary conditions. DEs handle it reasonably well (Tables 3 and 4), where the experimental value of $u(x, y)$ and the corresponding absolute error from the exact solution are presented for each node in the 2D space. Regarding DEs, for this case, the mean error is 2.53×10^{-5}, the number of generations for the model to achieve convergence is 923 (Fig. 3), and the total CPU time is 79.6 s. The plot of the proposed solution is given in Fig. 4. It can be observed that the same code is almost as efficient for the Dirichlet boundary conditions, as it is for Neumann conditions, which illustrates the robustness of the DE algorithm.

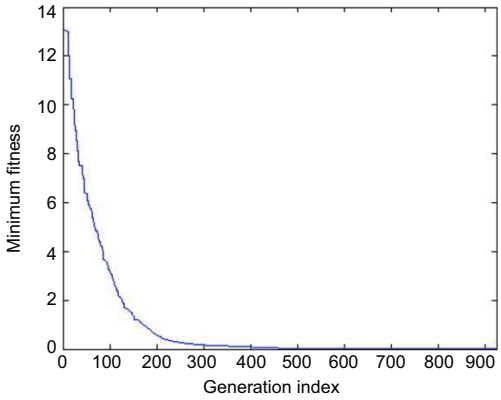

Fig. 1 Convergence curve for case 1 (Laplace equation with Dirichlet boundary conditions)

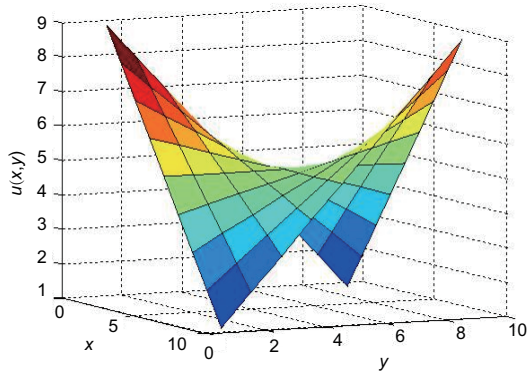

Fig. 2 Plot of the proposed solution for case 1 (Laplace equation with Dirichlet boundary conditions)

Table 3 Proposed solution for $u(x, y)$ for $1 \leq x, y \leq 9$ with $h = 1$ in case 2 (Laplace equation with Neumann boundary conditions)

Node	Experimental value								
	1	2	3	4	5	6	7	8	9
1	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
2	2.0000	2.0000	2.0000	2.0000	2.0000	2.0000	2.0000	2.0000	2.0000
3	3.0000	3.0000	3.0000	3.0000	3.0001	3.0000	3.0001	3.0000	3.0000
4	4.0000	4.0000	4.0000	4.0000	4.0001	4.0001	4.0001	4.0000	4.0000
5	5.0000	5.0000	5.0000	5.0001	5.0001	5.0001	5.0001	5.0001	5.0000
6	6.0000	6.0000	6.0000	6.0000	6.0001	6.0001	6.0001	6.0001	6.0000
7	7.0000	7.0000	7.0000	7.0000	7.0000	7.0001	7.0001	7.0001	7.0000
8	8.0000	8.0000	8.0000	8.0000	8.0000	8.0001	8.0001	8.0000	8.0000
9	9.0000	9.0000	9.0000	9.0000	9.0000	9.0000	9.0000	9.0000	9.0000

Table 4 Comparison with exact solution $u(x, y)$ for $1 \leq x, y \leq 9$, with $h = 1$ through the absolute error in case 2 (Laplace equation with Neumann boundary conditions)

Node	Absolute error ($\times 10^{-5}$)									Node	Absolute error ($\times 10^{-5}$)								
	1	2	3	4	5	6	7	8	9		1	2	3	4	5	6	7	8	9
1	0	0	0	0	0	0	0	0	0	6	0	2.6	0.19	3.5	7.4	7.4	11	8.8	0
2	0	0.6	2.0	2.9	4.8	2.8	2.5	1.0	0	7	0	2.3	2.7	0.76	10	10	10	8.6	0
3	0	0.36	0.54	2.6	4.2	4.2	5.0	1.4	0	8	0	0.99	2.8	0.44	7.9	7.6	7.6	4.8	0
4	0	0.22	0.82	2.9	5.3	5.3	5.1	1.3	0	9	0	0	0	0	0	0	0	0	0
5	0	0.77	0.34	5.1	7.0	7.0	9.5	5.0	0										

4.2 Results and analysis of the DE solver for Poisson's equations

The same codes are executed for Poisson's equations for both linear and nonlinear cases.

1. Linear case

Getting a step higher, to solve Poisson's equation, with the same techniques without any alteration, the question in case 3 is solved successfully. DE statistics show that the mean error is 3.182×10^{-7}, the number of generations to achieve convergence is 1122 (Fig. 5), and the CPU time is 99.7738 s. For DEs, the respective nodal errors from an exact solution are demonstrated in Tables 5 and 6. Fig. 6 illustrates the surface plot in 3D along with the exact solution and contour plot of the absolute error in case of DEs. It can be observed that the solutions just overlap.

2. Nonlinear case

The DE solver is executed for the nonlinear Poisson equations depicted for case 4 in Tables 7 and 8. The convergence curve and comparison with

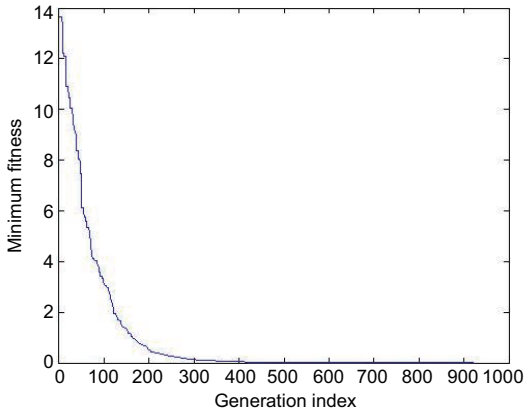

Fig. 3 Convergence curve for case 2 (Laplace equation with Neumann boundary conditions)

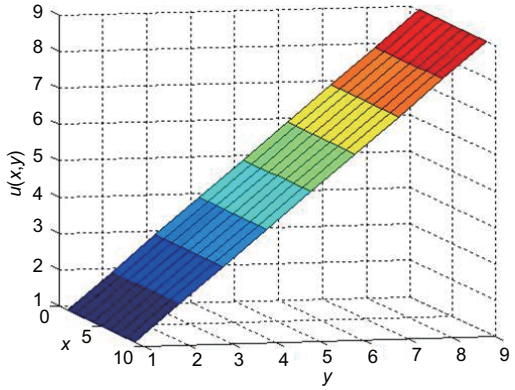

Fig. 4 Plot of the proposed solution for case 2 (Laplace equation with Neumann boundary conditions)

Table 5 Proposed solution for $u(x, y)$ for $0 \leq x \leq 2, 0 \leq y \leq 1$ in case 3 (linear Poisson's equation with Dirichlet boundary conditions)

Node	Experimental value								
	1	2	3	4	5	6	7	8	9
1	0	0.2500	0.5000	0.7500	1.0000	1.2500	1.5000	1.7500	2.0000
2	0	0.2833	0.5666	0.8499	1.1332	1.4165	1.6998	1.9831	2.2663
3	0	0.3211	0.6421	0.9632	1.2842	1.6998	1.9262	2.2472	2.5681
4	0	0.3638	0.7276	1.0914	1.4552	1.9262	2.1827	2.5464	2.9100
5	0	0.4122	0.8245	1.2367	1.6490	2.1827	2.4734	2.8855	3.2974
6	0	0.4671	0.9343	1.4014	1.8685	2.4734	2.8027	3.2697	3.7365
7	0	0.5293	1.0596	1.5879	2.1172	2.8027	3.1757	3.7049	4.2340
8	0	0.5993	1.1195	1.7993	2.7183	3.1757	3.5985	4.1982	4.7978
9	0	0.6796	1.3591	2.0387	2.0387	3.5985	4.0774	4.7570	5.4366

Table 6 Comparison with exact solution for $u(x, y)$ for $0 \leq x \leq 1, 0 \leq y \leq 2$ through the absolute error in case 3 (linear Poisson's equation with Dirichlet boundary conditions)

Node	Absolute error ($\times 10^{-5}$)									Node	Absolute error ($\times 10^{-5}$)								
	1	2	3	4	5	6	7	8	9		1	2	3	4	5	6	7	8	9
1	0	0	0	0	0	0	0	0	0	6	0	6.5	12	19	24	28	29	21	0
2	0	2.5	5.1	7.5	9.6	11	11	8.7	0	7	0	5.4	10	15	20	23	24	19	0
3	0	4.5	9.1	13	17	19	19	15	0	8	0	3.3	6.6	9.7	12	14	15	12	0
4	0	5.9	11	17	22	25	25	19	0	9	0	0	0	0	0	0	0	0	0
5	0	6.6	13	19	24	28	29	21	0										

the exact solution are presented in Figs. 7 and 8, respectively. Additionally, respective results for case 5 are presented in Figs. 9 and 10 as well as in Tables 9 and 10. In case 4, as expected due to nonlinearity, DE takes more generations, i.e., 2461, to achieve convergence, and hence the CPU time of 210.867 s is greater as well. The mean error is 9.4438×10^{-5}.

For the second case of the nonlinear Poisson problem, the statistics do not differ much in comparison to case 4. DE converges after 2342 generations with the mean error 4.3355×10^{-4} and CPU time 208.129 s.

Comparing linear and nonlinear cases, the time for execution increases significantly from the linear to the nonlinear case because of obvious complexity of the problem at hand. The same is true regarding the number of generations for convergence. We can see the difference in mean errors, which is much smaller than that of the linear Poisson equation.

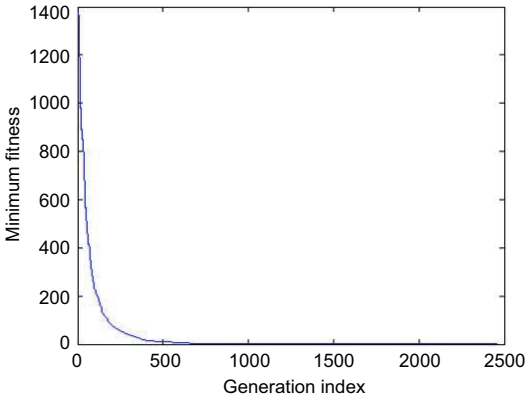

Fig. 5 Convergence curve for case 3 (linear Poisson's equation with Dirichlet boundary conditions)

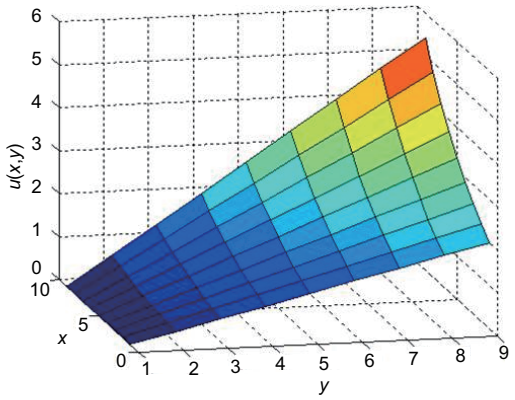

Fig. 6 Plot of the proposed solution for case 3 (linear Poisson's equation with Dirichlet boundary conditions)

Table 7 Proposed solution for $u(x, y)$ for $0 \leq x \leq 1, 0 \leq y \leq 2$ in case 4 (nonlinear Poisson's equation I with Dirichlet boundary conditions)

Node	Experimental value								
	1	2	3	4	5	6	7	8	9
1	0	0.0156	0.6250	0.1406	0.2500	0.3906	0.5625	0.7656	1.0000
2	0.0625	0.0156	0.0000	0.0156	0.0625	0.1406	0.2500	0.3906	0.5625
3	0.2500	0.1406	0.0625	0.0156	0.0000	0.0156	0.0625	0.1406	0.2500
4	0.5625	0.3906	0.2500	0.1406	0.0625	0.0156	0.0000	0.0156	0.0625
5	1.0000	0.7656	0.5625	0.3906	0.2500	0.1406	0.0625	0.0156	0
6	1.5625	1.2656	1.0000	0.7656	0.5625	0.3906	0.2500	0.1406	0.0625
7	2.2500	1.8906	1.5625	1.2656	1.0000	0.7656	0.5625	0.3906	0.2500
8	3.0625	2.6406	2.2500	1.8906	1.5625	1.2656	1.0000	0.7656	0.5625
9	4.0000	3.5156	3.0625	2.6406	2.2500	1.8906	1.5625	1.2656	1.0000

Table 8 Comparison with exact solution $u(x, y)$ for $0 \leq x \leq 1, 0 \leq y \leq 2$ through the absolute error in case 4 (nonlinear Poisson's equation I with Dirichlet boundary conditions)

Node	Absolute error ($\times 10^{-7}$)									Node	Absolute error ($\times 10^{-7}$)								
	1	2	3	4	5	6	7	8	9		1	2	3	4	5	6	7	8	9
1	0	0	0	0	0	0	0	0	0	6	0	4.3	7.7	7.2	4.8	3.2	2.1	0.3	0
2	0	3.5	6.8	9.0	8.4	6.5	3.9	1.1	0	7	0	2.8	5.4	6.4	5.6	4.2	2.4	1.2	0
3	0	5.5	8.5	10	11	10	5.7	2.4	0	8	0	0.57	2.7	3.8	4.2	2.6	2.7	0.7	0
4	0	5.6	7.7	9.3	9.4	8.6	4.7	0.82	0	9	0	0	0	0	0	0	0	0	0
5	0	3.2	5.3	7.6	9.0	8.0	5.6	2.5	0										

4.3 Performance of differential evolution

Satisfactory performance of differential evolution, in terms of the absolute error and CPU time for convergence, is reflected in all five cases of both Poisson and Laplace equations. A monotonic decreasing trend of the cost values is observed for all the cases. The cost values decrease substantially for the first few generations in case of both entirely random and scattered initial population, but attain a uniform trend during evolution. This behavior is in compliance with other evolutionary optimization strategies. Also, the DE methodology tends to tweak transition step sizes according to the diversity of the population. The initial population is widely dispersed admitting a larger step size during the initial evolution phase. Near convergence of various individuals, i.e., steady state of optimization, the population becomes less diverse and hence adapts to a smaller step size. This adaptability of step sizes leads to higher computational efficiency of the DE scheme and gives it strength.

1. Time analysis of DE

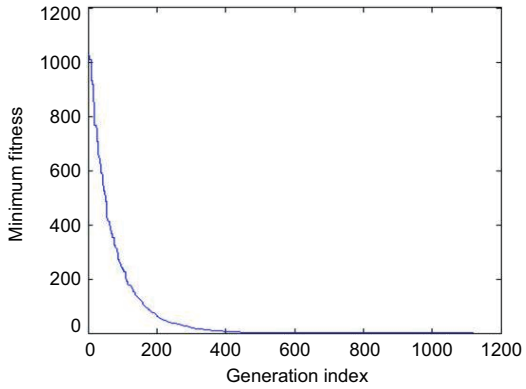

Fig. 7 Convergence curve for case 4 (nonlinear Poisson's equation I with Dirichlet boundary conditions)

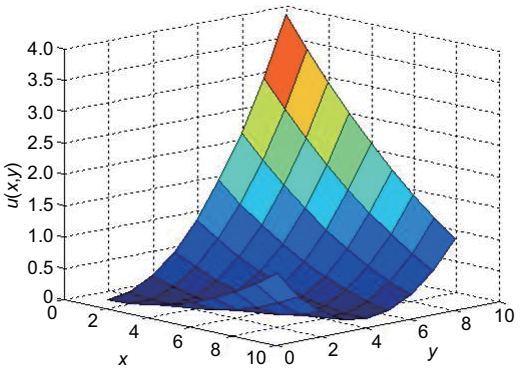

Fig. 8 Plot of the proposed solution for case 4 (nonlinear Poisson's equation I with Dirichlet boundary conditions)

Table 9 Proposed solution for $u(x, y)$ for $0 \leq x \leq \pi, 0 \leq y \leq \pi/2$ in case 5 (nonlinear Poisson's equation II with Dirichlet boundary conditions)

Node	Experimental value								
	1	2	3	4	5	6	7	8	9
1	0	0	0	0	0	0	0	0	0
2	1.0000	0.9239	0.7071	0.3827	0	−0.3827	−0.7071	−0.9239	−1.0000
3	0.9808	0.9609	0.6943	0.3758	−0	−0.3758	−0.6943	−0.9069	−0.9808
4	0.9239	0.8547	0.6545	0.3543	−0	−0.3543	−0.6545	−0.8547	−0.9239
5	0.8315	0.7694	0.5893	0.3190	−0	−0.3190	−0.5893	−0.7694	−0.8315
6	0.7071	0.6545	0.5013	0.2714	−0	−0.2714	−0.5013	−0.6545	−0.7071
7	0.5556	0.5143	0.3940	0.2133	0	−0.2133	−0.3940	−0.5143	−0.5556
8	0.3827	0.3543	0.2784	0.1469	0	−0.1469	−0.2714	−0.3543	−0.3827
9	0.1951	0.1806	0.1384	0.0749	0	−0.0749	−0.1384	−0.1806	−0.1951

Table 10 Comparison with exact solution for $u(x, y)$ for $0 \leq x \leq \pi, 0 \leq y \leq \pi/2$ through the absolute error in case 5 (nonlinear Poisson's equation II with Dirichlet boundary conditions)

Node	Absolute error ($\times 10^{-4}$)									Node	Absolute error ($\times 10^{-4}$)								
	1	2	3	4	5	6	7	8	9		1	2	3	4	5	6	7	8	9
1	0	0	0	0	0	0	0	0	0	6	0	10	11	7	0	7	11	10	0
2	0	8	8	5	0	5	8	8	0	7	0	7	8	5	0	5	8	7	0
3	0	11	12	7	0	7	12	11	0	8	0	4	4	3	0	3	4	4	0
4	0	13	14	8	0	8	14	13	0	9	0	0	0	0	0	0	0	0	0
5	0	12	13	8	0	8	13	12	0										

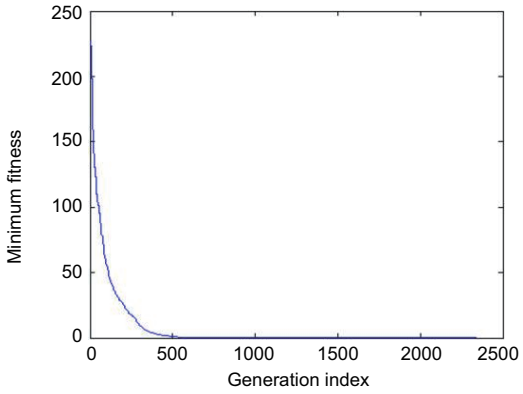

Fig. 9 Convergence curve for case 5 (nonlinear Poisson's equation II with Dirichlet boundary conditions)

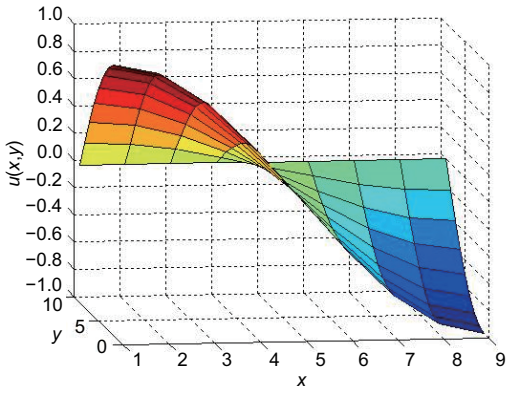

Fig. 10 Plot of the proposed solution for case 5 (nonlinear Poisson's equation II with Dirichlet boundary conditions)

Generally, the analysis of computational complexity is given in terms of fitness calls, when analyzing an evolutionary algorithm. However, as the chromosome size increases to a large number, as in our case, the candidate solution consists of 81 nodes, and fitness evaluations no longer remain a good measure of computational cost. Also, the context of this study revolves around the comparison of DE with other stochastic algorithms; it seems suitable for presenting the analysis in terms of computational time. It can be seen from all cases that the time consumed by DE for a solution to evolve is significantly less for the linear cases while more for nonlinear cases. Another important aspect is the increase in time consumption as the grid is refined from coarser to finer. It is also observed that the evolutionary progress of the DE algorithm with respect to generations of the five problems is in good agreement with the nature of the evolutionary algorithms, initially taking large steps and then refining the steps to avoid overshooting from minima.

2. Robustness of DE

As mentioned earlier, we have considered linear and nonlinear case studies of Laplace and Poisson's equations with different boundary conditions, and found that the proposed DE algorithm converges each time. It neither is trapped in any local minimum, nor gives any divergent result. Another important point about DE is that, we need not write separate codes for linear or nonlinear cases, making it a generic technique. Altogether, these features contribute towards the robustness of DE methodology for the application to an elliptic PDE solver.

3. Sensitivity to initialization

Let (min, max) define the expected range in which the solution lies. Four different sets of initialization ranges are defined to test the convergence of the differential evolution algorithm. The test cases are defined as: (1) lower bound less than min and upper bound greater than max; (2) lower bound less than min; (3) upper bound greater than max; (4) lower bound greater than min and upper bound less than max.

The time taken by the differential evolution algorithm for all these cases is almost the same, i.e., 51.6950, 50.5329, 52.7346, and 53.0481 s, which shows that the initialization ranges converge to the same fitness and also highlights the fact that the effect of the initialization method dies out quickly. This shows that the DE algorithm is less sensitive to initialization.

4. Convergence of interior and exterior nodes

The analysis of the convergence of interior and exterior nodes shows that the interior nodes that are farther from the predefined boundary nodes have more turbulence in the convergence process. However, the nodes that are closer to the predefined boundary nodes are less turbulent in the refinement stage of convergence. Also, the absolute error at nodes near the predefined boundary is less than that at the interior nodes, i.e., 2.0×10^{-6} vs. 1.1×10^{-4}.

5 Conclusions

In this study, a novel differential evolution based methodology has been developed for the solution of elliptic PDEs. The results clearly showed the effectiveness of the proposed DE methodology for various benchmark problems. CPU time required

and the amount of historical data required for convergence clearly showed that DE is an effective computing framework for solving PDEs. Also, DE can easily be applied to both linear and nonlinear problems. The random initialization scheme is easy to use and requires minimum information of the nature of the problem. A smaller step size yields more accurate answers, but it entails higher computational cost. Higher convergence rates, wider scope of applicability, inherently parallel nature, and robustness are among the attractive features of the proposed DE solver for elliptic PDEs.

Compliance with ethics guidelines

Muhammad Faisal FATEH, Aneela ZAMEER, Sikander M. MIRZA, Nasir M. MIRZA, Muhammad Saeed ASLAM, and Muhammad Asif Zahoor RAJA declare that they have no conflict of interest.

References

Ahmad I, Raja MAZ, Bilal M, et al., 2016. Bio-inspired computational heuristics to study Lane-Emden systems arising in astrophysics model. *SpringerPlus*, 5, Article 1866. https://doi.org/10.1186/s40064-016-3517-2

Ahmad I, Raja MAZ, Bilal M, et al., 2017. Neural network methods to solve the Lane-Emden type equations arising in thermodynamic studies of the spherical gas cloud model. *Neur Comput Appl*, 28(S1):929-944. https://doi.org/10.1007/s00521-016-2400-y

Bramble JH, Hubbard BE, Thomée V, 1969. Convergence estimates for essentially positive type discrete Dirichlet problems. *Math Comput*, 23(108):695-709.

Carlson HA, Verberg R, Harris CA, 2017. Aeroservoelastic modeling with proper orthogonal decomposition. *Phys Fluids*, 29(2):020711. https://doi.org/10.1063/1.4975673

Carotenuto A, Ciccolella M, Massarotti N, et al., 2016. Models for thermo-fluid dynamic phenomena in low enthalpy geothermal energy systems: a review. *Renew Sustain Energy Rev*, 60:330-355. https://doi.org/10.1016/j.rser.2016.01.096

Chang HH, Tsai CY, 2014. Adaptive registration of magnetic resonance images based on a viscous fluid model. *Comput Methods Progr Biomed*, 117(2):80-91. https://doi.org/10.1016/j.cmpb.2014.08.004

Courant R, Friedrichs K, Lewy H, 1928. Über die partiellen differenzengleichungen der mathematischen physik. *Math Ann*, 100(1):32-74 (in German). https://doi.org/10.1007/BF01448839

Dai CQ, Chen RP, Wang YY, et al., 2017a. Dynamics of light bullets in inhomogeneous cubic-quintic-septimal nonlinear media with PT-symmetric potentials. *Nonl Dynam*, 87(3):1675-1683. https://doi.org/10.1007/s11071-016-3143-0

Dai CQ, Liu J, Fan Y, et al., 2017b. Two-dimensional localized Peregrine solution and breather excited in a variable-coefficient nonlinear Schrödinger equation with partial nonlocality. *Nonl Dynam*, 88(2):1373-1383. https://doi.org/10.1007/s11071-016-3316-x

Das R, Singh K, Gogoi TK, 2017. Estimation of critical dimensions for a trapezoidal-shaped steel fin using hybrid diffierential evolution algorithm. *Neur Comput Appl*, 28(7):1683-1693. https://doi.org/10.1007/s00521-015-2155-x

Ding DJ, Jin DQ, Dai CQ, 2017. Analytical solutions of differential-difference sine-Gordon equation. *Therm Sci*, 21(4):1701-1705. https://doi.org/10.2298/TSCI160809056D

Fateh MF, Zameer A, Mirza NM, et al., 2017. Biologically inspired computing framework for solving two-point boundary value problems using differential evolution. *Neur Comput Appl*, 28(8):2165-2179.

Gao LN, Zi YY, Yin YH, et al., 2017. Bäcklund transformation, multiple wave solutions and lump solutions to a (3+1)-dimensional nonlinear evolution equation. *Nonl Dynam*, 89(3):2233-2240. https://doi.org/10.1007/s11071-017-3581-3

Gerschgorin VS, 1930. Fehlerabschätzung für das differenzenverfahren zur Lösung partieller differentialgleichungen. *J Appl Math Mech*, 10(4):373-382 (in German). https://doi.org/10.1002/zamm.19300100409

Gilbarg D, Trudinger NS, 2015. Elliptic Partial Differential Equations of Second Order. Springer, Berlin, Germany.

Guo W, Xu T, Lu ZL, 2016. An integrated chaotic time series prediction model based on efficient extreme learning machine and differential evolution. *Neur Comput Appl*, 27(4):883-898. https://doi.org/10.1007/s00521-015-1903-2

Hua YF, Guo BL, Ma WX, et al., 2019. Interaction behavior associated with a generalized (2+1)-dimensional Hirota bilinear equation for nonlinear waves. *Appl Math Model*, 74:184-198. https://doi.org/10.1016/j.apm.2019.04.044

Li YY, Zhao Y, Xie GN, et al., 2014. Local fractional Poisson and Laplace equations with applications to electrostatics in fractal domain. *Adv Math Phys*, 2014:590574.

Machado MR, Pereira VS, Costa DI, et al., 2016. Dynamics analysis of 1D structure including random parameter via frequency-domain state-vector equations. Proc 37th Iberian Latin-American Congress on Computational Mehods in Engineering, p.1-14.

Marichal Y, Chatelain P, Winckelmans G, 2014. An immersed interface solver for the 2-D unbounded Poisson equation and its application to potential flow. *Comput Fluids*, 96:76-86. https://doi.org/10.1016/j.compfluid.2014.03.012

Motzkin TS, Wasow W, 1952. On the approximation of linear elliptic differential equations by difference equations with positive coefficients. *J Math Phys*, 31(1-4):253-259. https://doi.org/10.1002/sapm1952311253

Oden JT, Lima EA, Almeida RC, et al., 2016. Toward predictive multiscale modeling of vascular tumor growth. *Arch Comput Methods Eng*, 23(4):735-779. https://doi.org/10.1007/s11831-015-9156-x

Peaceman DW, Rachford HHJr, 1955. The numerical solution of parabolic and elliptic differential equations. *J Soc Ind Appl Math*, 3(1):28-41. https://doi.org/10.1137/0103003

Raja MAZ, Khan MAR, Mahmood T, et al., 2016a. Design of bio-inspired computing technique for nanofluidics based on nonlinear Jeffery-Hamel flow equations. *Can J Phys*, 94(5):474-489. https://doi.org/10.1139/cjp-2015-0440

Raja MAZ, Farooq U, Chaudhary NI, et al., 2016b. Stochastic numerical solver for nanofuidic problems containing multi-walled carbon nanotubes. *Appl Soft Comput*, 38:561-586. https://doi.org/10.1016/j.asoc.2015.10.015

Raja MAZ, Shah FH, Alaidarous ES, et al., 2017a. Design of bio-inspired heuristic technique integrated with interior-point algorithm to analyze the dynamics of heartbeat model. *Appl Soft Comput*, 52:605-629. https://doi.org/10.1016/j.asoc.2016.10.009

Raja MAZ, Ahmad I, Khan I, et al., 2017b. Neuro-heuristic computational intelligence for solving nonlinear pantograph systems. *Front Inform Technol Electron Eng*, 18(4):464-484. https://doi.org/10.1631/FITEE.1500393

Raja MAZ, Aslam MS, Chaudhary NI, et al., 2018. Bio-inspired heuristics hybrid with interior-point method for active noise control systems without identification of secondary path. *Front Inform Technol Electron Eng*, 19(2):246-259. https://doi.org/10.1631/FITEE.1601028

Selvadurai AP, 2000. Partial Differential Equations in Mechanics 2: the Biharmonic Equation, Poissona Equation. Springer Science & Business Media, Berlin, Germany.

Shao YL, Faltinsen OM, 2014. A harmonic polynomial cell (HPC) method for 3D Laplace equation with application in marine hydrodynamics. *J Comput Phys*, 274:312-332. https://doi.org/10.1016/j.jcp.2014.06.021

Spiller C, Toro EF, Vázquez-Cendón ME, et al., 2017. On the exact solution of the Riemann problem for blood flow in human veins, including collapse. *Appl Math Comput*, 303:178-179. https://doi.org/10.1016/j.amc.2017.01.024

Storn R, Price K, 1997. Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. *J Glob Optim*, 11(4):341-359. https://doi.org/10.1023/A:1008202821328

Taleei A, Dehghan M, 2014. Direct meshless local Petrov-Galerkin method for elliptic interface problems with applications in electrostatic and elastostatic. *Comput Methods Appl Mech Eng*, 278:479-498. https://doi.org/10.1016/j.cma.2014.05.016

Vétois J, 2016. A priori estimates and application to the symmetry of solutions for critical p-Laplace equations. *J Differ Equat*, 260(1):149-161. https://doi.org/10.1016/j.jde.2015.08.041

Wang YY, Zhang YP, Dai CQ, 2016. Re-study on localized structures based on variable separation solutions from the modified tanh-function method. *Nonl Dynam*, 83(3):1331-1339. https://doi.org/10.1007/s11071-015-2406-5

Wang YY, Chen L, Dai CQ, et al., 2017. Exact vector multipole and vortex solitons in the media with spatially modulated cubica-quintic nonlinearity. *Nonl Dynam*, 90(2):1269-1275. https://doi.org/10.1007/s11071-017-3725-5

Webster AG, 2016. Partial Differential Equations of Mathematical Physics. Courier Dover Publications, Inc., New York, USA.

Yin YH, Ma WX, Liu JG, et al., 2018. Diversity of exact solutions to a (3+1)-dimensional nonlinear evolution equation and its reduction. *Comput Math Appl*, 76(6):1275-1283. https://doi.org/10.1016/j.camwa.2018.06.020

Zhang B, Zhang XL, Dai CQ, 2017. Discussions on localized structures based on equivalent solution with different forms of breaking soliton model. *Nonl Dynam*, 87(4):2385-2393. https://doi.org/10.1007/s11071-016-3197-z

Zhao LG, Liu HD, Zhao FK, 2013. Existence and concentration of solutions for the Schrödinger-Poisson equations with steep well potential. *J Differ Equat*, 255(1):1-23. https://doi.org/10.1016/j.jde.2013.03.005