Research Article

Solving Partial Differential Equations Using a New Differential Evolution Algorithm

Natee Panagant and Sujin Bureerat

*Sustainable Infrastructure Research and Development Center, Department of Mechanical Engineering,
Faculty of Engineering, Khon Kaen University, Khon Kaen 40002, Thailand*

Correspondence should be addressed to Sujin Bureerat; sujbur@kku.ac.th

Received 7 February 2014; Accepted 26 May 2014; Published 12 June 2014

Academic Editor: Khai Ching Ng

Copyright © 2014 N. Panagant and S. Bureerat. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

This paper proposes an alternative meshless approach to solve partial differential equations (PDEs). With a global approximate function being defined, a partial differential equation problem is converted into an optimisation problem with equality constraints from PDE boundary conditions. An evolutionary algorithm (EA) is employed to search for the optimum solution. For this approach, the most difficult task is the low convergence rate of EA which consequently results in poor PDE solution approximation. However, its attractiveness remains due to the nature of a soft computing technique in EA. The algorithm can be used to tackle almost any kind of optimisation problem with simple evolutionary operation, which means it is mathematically simpler to use. A new efficient differential evolution (DE) is presented and used to solve a number of the partial differential equations. The results obtained are illustrated and compared with exact solutions. It is shown that the proposed method has a potential to be a future meshless tool provided that the search performance of EA is greatly enhanced.

1. Introduction

In the field of computational mechanics, many engineering systems can be viewed as continuum domains and their physics are represented by using differential equations, for example, heat transfer, structures, and fluid mechanics. Some of such differential equations can be solved analytically; however, in most practical cases, the domains are too complicated for an analytical approach. This leads to the creation of numerical approximation of the differential equations. Some of the established approaches include finite element methods [1–4], finite difference methods [5, 6], and finite volume methods [7–9]. Those previously mentioned methods are classified as the mesh-based or grid-based methods. The concept is to approximate a complicated domain of differential equations with a finite number of simple grids enabling one to find the approximate solutions of the differential equations.

Nevertheless, although the mesh-based methods like finite element analysis have become a traditional engineering tool with numerous commercial software packages being developed around the world, they still have some

drawbacks, for example, difficulties in mesh generation and mesh distortion in large deformation analysis of structures. This causes researchers and engineers to search for alternative numerical approaches and the so-called meshless methods were originated. The method unlike the traditional finite element or finite volume methods uses only distribution of nodes on a differential equation domain to achieve its approximate solution. Over the last few decades, there have been numerous meshless methods proposed to solve a number of engineering applications. Starting with one of the earliest methods, the smooth particle hydrodynamics [10, 11] approach; then, there have been other established methods such as the element-free Galerkin method [12], the reproducing kernel particle method [13], and the local Petrov-Galerkin method [14–16]. A good review on meshless methods can be found in [17, 18]. For some up-to-date development, see [19–22]. The meshless methods, despite having some advantages, have some weak points such as difficulties in dealing with boundary conditions (BCs). Therefore, more research and ideas are still needed in this field.

This paper proposes an alternative way to achieve solutions of partial differential equations (PDEs). The approach is classified as a meshless method since only nodes are generated on a differential equation domain. A partial differential equation problem is converted into an optimisation problem while a metaheuristic (MH) or an evolutionary algorithm (EA) is employed to search for the optimum solution. Such an idea has been presented in [23, 24] but there are many obstacles to overcome if one wants to use it in practice. The most important issue is the low convergence rate of EAs which consequently results in a not-so-good approximate solution. However, the attractiveness of this approach remains due to the nature of a soft computing technique in EA. The algorithm can be used to tackle almost any kind of optimisation problem with simple random-directed evolutionary operation, which means it is mathematically simpler to use. In this paper, a new efficient differential evolution (DE) is presented and used to solve a number of the partial differential equations. The results obtained are illustrated and compared with exact solutions. The rest of this paper is organised as follows. The second section briefly reviews the differential evolution method where the new DE is proposed. Section 3 details how to solve the partial differential equations by means of evolutionary optimisation. The results are shown and discussed in Section 4 while conclusions are drawn in Section 5.

2. Differential Evolution Algorithm

Some of the earliest evolutionary algorithms were proposed a few decades ago where genetic algorithms are arguably the best known method. Since then, this kind of soft computing has gained increasing popularity year after year because of their simplicity to use, derivative-free feature, and robustness. So far, there have been a thousand of EAs and their variants developed and presented in research literature. Among such an incredibly large number of algorithms, differential evolution is one of the top performers for general optimisation purposes. The method is a parallel random-directed search method that was first proposed by Storn and Price [25]. DE is powerful yet very simple to understand, code, and implement which resulted in its great popularity nowadays. A large number of DE variants have been proposed, for example, in [26, 27]. Similar to most EAs, a DE computational procedure starts with a randomly generated initial real-code population. Then the DE operators (mutation, crossover, and selection) are iteratively activated to improve the population approaching to an optimum. DE since it was first introduced has various schemes. In this paper, we use 4 versions of the original differential evolution algorithm: *DE/rand/1/bin*, *DE/rand/1/exp*, *DE/current-to-best/2/bin*, and *DE/current-to-best/2/exp*. Note that the notation of DE schemes is described as

$$DE/x/y/z, \quad (1)$$

where x is the vector to be mutated (“*rand*” for a randomly chosen vector and “*current-to-best*” for a current population vector that is mutated by the best so far vector), y is a number

Input: $\mathbf{x}, \mathbf{v}, CR$

Output $\mathbf{u}$

- (1) Set $\mathbf{u} = \mathbf{x}$
- (2) Randomly choose $m \in \{1, 2, \dots, d\}$
- (3) For $i = 1$ to d
- (4) Generate a uniform random number $rand \in [0, 1]$.
- (5) If $rand \leq CR$ or $= m$, set $u_i = v_i$.
- (6) Next i

ALGORITHM 1: Binomial crossover.

of pair of different vectors used in the mutation operation, and z is a crossover type (“*bin*” for binomial crossover and “*exp*” for exponential crossover).

2.1. Differential Evolution Operators. There are three main operators for DE which can be briefly detailed as follows.

2.1.1. Mutation. For a population vector $\mathbf{x}_{i,G}$, $i = 1, 2, 3, \dots, NP$, where G is a generation number and NP is a population size, a mutant vector for the *rand* scheme is produced using (2) and for the *current-to-best* scheme is produced using (3). Consider

$$\mathbf{v}_{i,G+1} = \mathbf{x}_{r_1,G} + F \times (\mathbf{x}_{r_2,G} - \mathbf{x}_{r_3,G}), \quad (2)$$

$$\mathbf{v}_{i,G+1} = \mathbf{x}_{i,G} + \lambda (\mathbf{x}_{\text{best}} - \mathbf{x}_{i,G}) + F \times (\mathbf{x}_{r_1,G} - \mathbf{x}_{r_2,G}), \quad (3)$$

where r_1, r_2 , and r_3 are random integer indices of population vectors ($r_1, r_2, r_3 \in \{1, 2, 3, \dots, NP\}$), $\mathbf{x}_{i,G}$ is current population vector of last generation, $\mathbf{x}_{\text{best}}$ is a population vector that has the best (minimum) fitness in the previous generation, F is a scaling factor ($F \in [0, 2]$) that controls amplification of difference vectors, and λ is an additional control variable ($\lambda \in [0, 1]$) that controls influence of the best population vector.

2.1.2. Crossover. In the crossover operation, a target vector ($\mathbf{x}_{i,G+1}$) will be bred with a mutant vector ($\mathbf{v}_{i,G+1}$) leading to a trial vector ($\mathbf{u}_{i,G+1}$). The probability of crossover ($PC \in [0, 1]$) will be predefined. Once the operator is revoked, there can be two types of crossover as binomial and exponential crossovers which are given in Algorithms 1 and 2, respectively, where CR is a prespecified constant and d is a dimension of a design vector.

2.1.3. Selection. Before adding trial vectors into a new generation population, the selection operator will compare the fitness function value of a trial vector to that of its target vector. Then the vector that has a lower fitness function value is selected into the new generation population.

2.1.4. Termination Criteria. In this paper, we specify two stopping criteria. The calculation will be terminated when maximum function evaluation is reached or allowable minimum fitness function and constraint values are reached.

Input: $\mathbf{x}, \mathbf{v}, CR$
Output $\mathbf{u}$
(1) Set $\mathbf{u} = \mathbf{x}$
(2) Randomly choose $k \in \{1, 2, \dots, d\}$
(3) Set $L = 1$. Generate a uniform random number $rand \in [0, 1]$.
(4) While $rand \leq CR^L$ or $L \leq d$, do
(5) $u_k = v_k, L = L + 1$
(6) $k = k + 1$, If $k > d, k = 1$
(7) End while

ALGORITHM 2: Exponential crossover.

2.2. New Differential Evolution Algorithm. In this work, we present new DE with variation of population size during an optimisation run. This DE scheme will be termed *DE-New*. The differential operators (Mutation, crossover, and selection) and stopping criteria of *DE-New* are the same as those used in the original DE. The new DE starts searching with an initial population size. The modification is based on the premise that a larger population size tends to produce some better design solutions. Nevertheless, using a large population size throughout the optimisation run can be time consuming. Thus, the concept of population size variation is employed. As the algorithm continues, if a convergence speed of the algorithm at the current iteration is not satisfied, then additional members will be generated and added to the population. The flowchart of *DE-New* is shown in Figure 1.

During the search process in which the best fitness for all iterations are obtained, if the standard deviation of the best fitness values of the previous npv iterations is less than a predefined value pvc , then the number of solution members in a population is increased but it must be limited in such a way that the total number of population members must not exceed a predefined maximum population size. In cases that the current population size is less than maximum population size and the aforementioned criterion is met, additional solution vectors will be generated and added to the population. Given that NP is the size of the current population, other NP solutions in which half of them are randomly generated (exploration) and the rest are created by means of DE operation (exploitation) will be added to the population. Note that, in this paper, *DE/rand/1/exp* and *DE/current-to-best/2/exp* are used with this population variation concept. The *DE/rand/1/exp* scheme is used to update a regular population for each iteration while the *DE/current-to-best/2/exp* scheme is used in the additional population generation process.

3. Numerical Examples

The proposed approach is somewhat similar to the literature [24] in the sense that a partial differential equation problem is altered to be a minimisation problem with equality constraints. Initially, nodes are generated inside and on the

boundary of a PDE domain. A function for approximating a PDE solution is selected where boundary conditions are treated as equality constraints. The optimisation problem is posed to find function coefficients in such a way to minimise residuals or square errors. In the previous work, evolution strategies (ES) were used as an optimiser while an approximate function is a partial sum of Fourier cosine series. In this work, DE will be used instead of ES as it is found to be more powerful whereas a polynomial function is used to estimate the PDE solution. In order to demonstrate the performance of a new approach, 6 PDE problems are used for numerical test which can be detailed as follows:

PDE1: governing equation: $\nabla^2 \psi(x, y) = e^{-x}(x - 2 + y^3 + 6y)$,

domain: $x, y \in [0, 1]$,

BCs: $\psi(0, y) = y^3$; $\psi(1, y) = (1 + y^3)e^{-1}$;
 $\psi(x, 0) = xe^{-x}$; and $\psi(x, 1) = (x + 1)e^{-x}$,

exact solution: $\psi(x, y) = (x + y^3)e^{-x}$;

PDE2: governing equation: $\nabla^2 \psi(x, y) = -2\psi$,

domain: $x, y \in [0, 1]$,

BCs: $\psi(0, y) = 0$; $\psi(1, y) = \sin(1)\cos(y)$;
 $\psi(x, 0) = \sin(x)$; and $\psi(x, 1) = \sin(x)\cos(1)$,

exact solution: $\psi(x, y) = \sin(x)\cos(y)$;

PDE3: governing equation: $\nabla^2 \psi(x, y) = 4$,

domain: $x, y \in [0, 1]$,

BCs: $\psi(0, y) = y^2 + y + 1$; $\psi(1, y) = y^2 + y + 3$;
 $\psi(x, 0) = x^2 + x + 1$; and $\psi(x, 1) = x^2 + x + 3$,

exact solution: $\psi(x, y) = x^2 + y^2 + x + y + 1$;

PDE4: governing equation: $\nabla^2 \psi(x, y) = -\psi(x^2 + y^2)$,

domain: $x, y \in [0, 1]$,

BCs: $\psi(0, y) = 0$; $\psi(1, y) = \sin(y)$; $\psi(x, 0) = 0$;
and $\psi(x, 1) = \sin(x)$,

exact solution: $\psi(x, y) = \sin(xy)$;

PDE5: governing equation: $\nabla^2 \psi(x, y) = 4x \cos x + (5 - x^2 - y^2) \sin x$,

domain: $x^2 + y^2 \leq 1$,

FIGURE 1: Flowchart of new DE with population size variation.

FIGURE 2: Calculation of an objective function.

BCs $\psi(x, y) = 0$ in $\partial\Omega$,

exact solution: $\psi(x, y) = (x^2 + y^2 - 1) \sin x$;

PDE6: governing equation: $\nabla^2 \psi(x, y) = 2e^{(x-y)}$,

domain: $R^2(\theta) \leq \cos(2\theta) + \sqrt{1.1 - \sin^2(2\theta)}$,

BCs: $\psi(x, y) = e^{(x-y)} + e^x \cos y$ in $\partial\Omega$,

exact solution: $\psi(x, y) = e^{(x-y)} + e^x \cos y$.

The notation ∇^2 herein is a Laplacian operator. The acronym PDE i means the i th partial differential equation problem

FIGURE 3: Domain of PDE5.

while BC means boundary condition. PDE solutions are approximated by a polynomial expansion as in (4). Consider

$$\begin{aligned} \psi(x, y) = & a_1 + \sum_{i=1}^n a_{i+1} x^i + \sum_{i=1}^n a_{i+n+1} y^i \\ & + \sum_{i=1}^n \sum_{j=1}^n a_{(i-1)n+j+2n+1} x^i y^j, \end{aligned} \quad (4)$$

where a_i are polynomial coefficients to be determined and n is a maximum of the polynomial order for x and y .

The number of approximation terms is $(n+1)^2$, which means that there are $d = (n+1)^2$ design variables for an optimisation problem. Since (4) is in a polynomial form, we can calculate any orders of derivatives, which is simple to be used with a wide range of differential equations.

FIGURE 4: Domain of PDE6.

FIGURE 5: Search histories for PDE1.

A typical partial differential equation can be written in a general form as

$$\nabla^2 \psi = f(x, y) \quad \text{on domain } \Omega \quad (5)$$

with BCs $h_i(x, y) = h_{0,i}$ on Γ_i for $i = 1, \dots, k$.

Let N be the number of nodes being assigned on a PDE domain Ω and boundary where n_i is the number of nodes on the k boundaries Γ_i . If we substitute (4) into a PDE for all points on Ω , we can have N equations with d unknowns. Similarly, by substituting (4) on the BCs for all points on Γ_i , we have n_i equality constraints. This implies that we convert the PDE (5) to be a constrained optimisation problem where the N equations is replaced by an objective function to be minimised [23] which is calculated by means of a weighted

FIGURE 6: Search histories for PDE2.

residual approach. The minimisation problem can then be posed as

$$\text{minimize WRF}(\mathbf{a}) = \int_{\Omega} |W(x, y)| \cdot |R(x, y)| dA \quad (6)$$

which is subject to

$$h_i(\mathbf{a}) = 0 \quad \text{for } i = 1, \dots, n_i, \quad (7)$$

where $\mathbf{a} = \{a_1, \dots, a_{(n+1) \times (n+1)}\}^T$ is a vector of design variables or polynomial coefficients, $R(x, y)$ is a residual function, $W(x, y)$ is the weight function, in this case, which is set as unity as used in [23], A is an area of the domain, and h_i are obtained from substituting (4) into the boundary conditions for all nodes on the boundaries.

Figure 2 shows how to compute the integration (6) on a particular subarea which is discretised by the nodes on the domain. The value of the residual function $|R|$ on the subarea dA is the average value of $|R|$ from those 4 points. The ideal solution is to have the objective function WRF becoming zero.

In order to handle the equality constraints h_0 , a penalty function is used which can be expressed as

$$\text{PFV}(\mathbf{a}) = \sum_{i=1}^{n_i} k_i h_i^2(\mathbf{a}), \quad (8)$$

where k_i are penalty parameters (set to be 50 in this study). Thus, the fitness function (FFV) for DE can be calculated as

$$\text{FFV} = \text{WRF} + \text{PFV}. \quad (9)$$

The accuracy of the final solutions is measured using a root mean squared error (RMSE) which can be computed as

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^N \|\psi(x_i, y_i) - \psi_{\text{exact}}(x_i, y_i)\|^2}{N}}, \quad (10)$$

TABLE 1: Fitness and RMSE.

Case	Algorithm	Fitness	RMSE
PDE1	DE/rand/1/bin	9.4210×10^{-2}	1.0269×10^{-2}
	DE/rand/1/exp	4.6360×10^{-4}	7.2340×10^{-4}
	DE/current-to-best/2/bin	1.9190×10^{-1}	1.6346×10^{-2}
	DE/current-to-best/2/exp	1.5922×10^1	5.6488×10^{-1}
	DE-New	4.6345×10^{-4}	7.2560×10^{-4}
	Result from [24]	—	6.3700×10^{-3}
PDE2	DE/rand/1/bin	1.2251×10^{-1}	1.0285×10^{-2}
	DE/rand/1/exp	1.1404×10^{-4}	2.6039×10^{-4}
	DE/current-to-best/2/bin	7.3667×10^{-1}	2.8889×10^{-2}
	DE/current-to-best/2/exp	2.6998×10^1	7.4573×10^{-1}
	DE-New	1.1351×10^{-4}	2.4529×10^{-4}
	Result from [24]	—	1.1600×10^{-3}
PDE3	DE/rand/1/bin	1.0013×10^{-1}	1.0973×10^{-2}
	DE/rand/1/exp	4.1618×10^{-6}	4.2589×10^{-5}
	DE/current-to-best/2/bin	1.3256×10^{-1}	1.2053×10^{-2}
	DE/current-to-best/2/exp	9.5633	1.4867×10^{-1}
	DE-New	6.0052×10^{-8}	9.4888×10^{-6}
	Result from [24]	—	5.9000×10^{-3}
PDE4	DE/rand/1/bin	9.1112×10^{-2}	7.5679×10^{-3}
	DE/rand/1/exp	3.0683×10^{-3}	6.6047×10^{-3}
	DE/current-to-best/2/bin	5.2632×10^{-1}	2.1121×10^{-2}
	DE/current-to-best/2/exp	6.6558	1.1495×10^{-1}
	DE-New	3.0661×10^{-3}	6.6054×10^{-3}
	Result from [24]	—	1.2300×10^{-3}
PDE5	DE/rand/1/bin	1.0649	3.6933×10^{-2}
	DE/rand/1/exp	8.2231×10^{-1}	3.7063×10^{-2}
	DE/current-to-best/2/bin	5.1483	6.8413×10^{-2}
	DE/current-to-best/2/exp	7.4642×10^1	1.0544
	DE-New	8.2224×10^{-1}	3.7243×10^{-2}
	Result from [24]	—	9.0600×10^{-4}
PDE6	DE/rand/1/bin	7.3267×10^{-1}	3.7452×10^{-1}
	DE/rand/1/exp	4.3440×10^{-1}	3.7732×10^{-1}
	DE/current-to-best/2/bin	4.6867	3.6634×10^{-1}
	DE/current-to-best/2/exp	6.2369×10^1	6.5142×10^{-1}
	DE-New	4.3435×10^{-1}	3.8225×10^{-1}
	Result from [24]	—	1.7900×10^{-2}

where $\psi(x_i, y_i)$ is an approximate function value at node i , $\psi_{\text{exact}}(x_i, y_i)$ is the exact function value at node i , and N is the number of nodes.

Five versions of DE (DE/rand/1/bin, DE/rand/1/exp, DE/current-to-best/2/bin, DE/current-to-best/2/exp, and DE-New) are employed to tackle those six optimisation problems. The optimisation parameter settings are given as $PC = 0.7$, $F = 0.5$, $CR = 0.8$, and $\lambda = 0.9$. An optimiser is stopped when either the maximum number of function evaluations (500,000) is reached or the objective and penalty function values are less than 0.001. The best solution of the last generation of DE is classified as an optimum. Population variation criteria are $npv = 10$ and $pvc = 0.001$. The population size for the original DE is set to be 400 while the initial population size for DE-New is set as 100. The maximum

FIGURE 7: Search histories for PDE3.

FIGURE 8: Search histories for PDE4.

population size for DE-New is 400. For all PDE problems, the number of nodes is 100. Nodes are equispaced throughout the square domains for PDE1-4. The nodal distributions for PDE5-6 are illustrated in Figures 3 and 4, respectively.

4. Results and Discussion

The numerical experiment was carried out on a personal computer where the program is written using MATLAB computing language. Having employed five DE strategies for tackling six optimisation problems of PDE1-6, the results obtained are compared in Table 1. In the table, there are

FIGURE 9: Search histories for PDE5.

FIGURE 10: Search histories for PDE6.

two indicators for comparison, the fitness function value calculated using (5) and RMSE calculated using (10). The former is used to measure DE performance while the latter is used to measure the accuracy of the PDE solution approximation. The search histories of the five DEs for solving PDE1-6 are illustrated in Figures 5, 6, 7, 8, 9, and 10, respectively. From Table 1 based on the fitness value, it is shown that the proposed DE with population variation gives the best optimisation results for all test cases with only *DE/rand/1/exp* being comparable in cases of PDE4-6. The worst performer is *DE/current-to-best/2/exp*. From the search histories, it can be seen that *DE/current-to-best/2/exp* always gets stuck at a local minimum and its search histories for PDE5-6 are discarded from the figures. For PDE5-6, all DEs cannot approach to zero. This could possibly be due to unevenly distributed nodes as shown in Figures 3 and 4.

For approximation accuracy based on RMSE, the results from the literature [24] are compared. It should be noted that, in [24], different fitness function is used, thus, optimisation convergence from [24] and this work cannot be compared. Also, the nodal distributions of PDE5-6 from [24] are also different from those used in this work. From the table, *DE-New* gives the best results for PDE1-3 while the approach in [24] gives the best results in cases of PDE4-6. Nevertheless, it should be noted that the previous work used ES in combination with a partial sum Fourier cosine series with approximately 1,000,000 function evaluations which is twice the total number of function evaluations used in this paper. *DE-New* significantly outperforms all other DE versions for PDE1-3. It is said to be as good as *DE/rand/1/exp* in cases of PDE4 while *DE/rand/1/exp* along with *DE/rand/1/bin* is slightly better for the last two test cases.

Graphical comparisons of the approximate solutions from using *DE-New* and the exact solutions of PDE1-6 are displayed in Figures 11, 12, 13, 14, 15, and 16, respectively. The

left-hand side is an approximate solution while the right-hand side is the exact solution. It is shown that *DE-New* gives good approximation for PDE1-4 while the approximation of PDE5-6 is not so good. One of the notable reasons is that PDE5-6 have nodal distribution with poor quality. Figure 15 (PDE5) shows that the number of nodes on the boundary is insufficient. A similar conclusion can be said for PDE6 (Figure 16).

5. Conclusions and Future Work

An alternative numerical meshless approach for solving PDEs based on evolutionary computation is proposed. A PDE problem is converted into an optimisation problem while, in this paper, DE is used as an optimiser. A new DE with a population variation concept is proposed. The results show that the proposed method in this paper is comparable to the previous work while using approximately twice less the number of function evaluations in the optimisation process. Most of the approximate solutions of PDE are accurate except for some cases which use poorly distributed nodes. The performance of an optimiser is probably the most important factor for this type of meshless analysis. The choice of approximate function also affects the accuracy of estimating the PDE solutions. Apart from that, nodal distribution also plays a vital role. The weak points can be listed as difficulty in dealing with equality constraints of a metaheuristic, and its slow convergence rate. Moreover, the calculation of WRF is, to some extent, difficult in the sense that adjacent nodes for each subarea have to be identified initially.

For our future work, the performance enhancement of EAs or MHs is the first issue to be dealt with. EAs for optimisation with equality constraints should be rigorously investigated. The hybrid between EAs and some efficient gradient-based optimisers will be first focused. The choice of

FIGURE 11: Approximate and exact solutions for PDE1.

FIGURE 12: Approximate and exact solutions for PDE2.

FIGURE 13: Approximate and exact solutions for PDE3.

FIGURE 14: Approximate and exact solutions for PDE4.

FIGURE 15: Approximate and exact solutions for PDE5.

FIGURE 16: Approximate and exact solutions for PDE6.

approximate functions will be investigated. Applications to real-world problems will be demonstrated.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

- [1] E. Onate, J. Rojek, M. Chiumenti, S. R. Idelsohn, F. D. Pin, and R. Aubry, "Advances in stabilized finite element and particle methods for bulk forming processes," *Computer Methods in Applied Mechanics and Engineering*, vol. 195, no. 48-49, pp. 6750-6777, 2006.
- [2] B. T. Tang, Z. Zhao, H. Hagenah, and X. Y. Lu, "Energy based algorithms to solve initial solution in one-step finite element method of sheet metal stamping," *Computer Methods in Applied Mechanics and Engineering*, vol. 196, no. 17-20, pp. 2187-2196, 2007.
- [3] J. Nam, M. Behr, and M. Pasquali, "Space-time least-squares finite element method for convection-reaction system with transformed variables," *Computer Methods in Applied Mechanics and Engineering*, vol. 200, no. 33-36, pp. 2562-2576, 2011.
- [4] S. R. Idelsohn, E. Onate, F. D. Pin, and N. Calvo, "Fluid-structure interaction using the particle finite element method," *Computer Methods in Applied Mechanics and Engineering*, vol. 195, no. 17-18, pp. 2100-2123, 2006.
- [5] H. P. Chu and C. L. Chen, "Hybrid differential transform and finite difference method to solve the nonlinear heat conduction problem," *Communications in Nonlinear Science and Numerical Simulation*, vol. 13, no. 8, pp. 1605-1614, 2008.
- [6] J. Zhao and R. M. Corless, "Compact finite difference method for integro-differential equations," *Applied Mathematics and Computation*, vol. 177, no. 1, pp. 271-288, 2006.
- [7] I. Bijelonja, I. Demirdzic, and S. Muzaferija, "A finite volume method for incompressible linear elasticity," *Computer Methods in Applied Mechanics and Engineering*, vol. 195, no. 44-47, pp. 6378-6390, 2006.
- [8] X. Nogueira, L. C. Felgueroso, I. Colominas, and H. Gomez, "Implicit large eddy simulation of non-wall-bounded turbulent flows based on the multiscale properties of a high-order finite volume method," *Computer Methods in Applied Mechanics and Engineering*, vol. 199, no. 9-12, pp. 615-624, 2010.
- [9] A. G. Malan and O. F. Oxtoby, "An accelerated, fully-coupled, parallel 3D hybrid finite-volume fluid-structure interaction scheme," *Computer Methods in Applied Mechanics and Engineering*, vol. 253, pp. 426-438, 2013.
- [10] L. B. Lucy, "A numerical approach to the testing of the fission hypothesis," *The Astronomical Journal*, vol. 82, pp. 1013-1024, 1977.
- [11] R. A. Gingold and J. J. Monaghan, "Smoothed particle hydrodynamics: theory and application to non-spherical stars," *Monthly Notices of the Royal Astronomical Society*, vol. 181, pp. 375-389, 1977.
- [12] T. Belytschko, Y. Y. Lu, and L. Gu, "Element-free Galerkin methods," *International Journal for Numerical Methods in Engineering*, vol. 37, no. 2, pp. 229-256, 1994.
- [13] W. K. Liu, S. Jun, and Y. F. Zhang, "Reproducing Kernel particle methods," *International Journal for Numerical Methods in Fluids*, vol. 20, no. 8-9, pp. 1081-1106, 1995.
- [14] S. N. Atluri and S. Shen, "The Meshless Local Petrov-Galerkin (MLPG) method: a simple & less-costly alternative to the finite element and boundary element methods," *Computer Modeling in Engineering & Sciences*, vol. 3, no. 1, pp. 11-51, 2002.
- [15] S. N. Atluri and T. Zhu, "A new Meshless Local Petrov-Galerkin (MLPG) approach in computational mechanics," *Computational Mechanics*, vol. 22, no. 2, pp. 117-127, 1998.
- [16] S. N. Atluri and T. Zhu, "The Meshless Local Petrov-Galerkin (MLPG) approach for solving problems in elasto-statics," *Computational Mechanics*, vol. 25, no. 2, pp. 169-179, 2000.
- [17] K. M. Liew, X. Zhao, and A. J. M. Ferreira, "A review of meshless methods for laminated and functionally graded plates and shells," *Composite Structures*, vol. 93, no. 8, pp. 2031-2041, 2011.
- [18] V. P. Nguyen, T. Rabczuk, St. Bordas, and M. Duflot, "Meshless methods: a review and computer implementation aspects," *Mathematics and Computers in Simulation*, vol. 79, no. 3, pp. 763-813, 2008.
- [19] S. Wang, S. Li, Q. Huang, and K. Li, "An improved collocation meshless method based on the variable shaped radial basis function for the solution of the interior acoustic problems," *Mathematical Problems in Engineering*, vol. 2012, Article ID 632072, 20 pages, 2012.
- [20] S. Lyu, C. Wu, and S. Zhang, "Application of the RBF method to the estimation of temperature on the external surface in laminar pipe flow," *Mathematical Problems in Engineering*, vol. 2013, Article ID 205169, 11 pages, 2013.
- [21] S. F. Tu and C. S. Hsu, "Constructing HVS-based optimal substitution matrix using enhanced differential evolution," *Mathematical Problems in Engineering*, vol. 2013, Article ID 824239, 10 pages, 2013.
- [22] S. L. Mei, "HAM-based adaptive multiscale meshless method for burgers equation," *Journal of Applied Mathematics*, vol. 2013, Article ID 248246, 10 pages, 2013.
- [23] M. Babaei, "A general approach to approximate solutions of nonlinear differential equations using particle swarm optimization," *Applied Soft Computing*, vol. 13, no. 7, pp. 3354-3365, 2013.
- [24] J. M. Chaquet and E. J. Carmona, "Solving differential equations with Fourier series and evolution strategies," *Applied Soft Computing*, vol. 12, no. 9, pp. 3051-3062, 2012.
- [25] R. Storn and K. Price, "Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces," Tech. Rep. TR-95-012, International Computer Science Institute, 1995.
- [26] X. Wang and S. Zhao, "Differential evolution algorithm with self-adaptive population resizing mechanism," *Mathematical Problems in Engineering*, vol. 2013, Article ID 419372, 14 pages, 2013.
- [27] Z. Yang, K. Tang, and X. Yao, "Self-adaptive differential evolution with neighborhood search," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '08)*, pp. 1110-1116, Hong Kong, June 2008.