A new approach based on the genetic algorithm for finding a good shape parameter in solving partial differential equations by Kansa’s method

M. Esmaeilbeigi a,*, M.M. Hosseini b

a Department of Mathematics, Malayer University, Malayer, Iran
b Faculty of Mathematics, Yazd University, P.O. Box 89195-74, Yazd, Iran

1. Introduction

The ideal numerical method for partial differential equations (PDEs) should be high-order accurate, flexible with respect to the geometry, computationally efficient, and easy to implement. The methods that are commonly used usually fulfil one or two of the criteria, but not all. Finite difference methods can be made high-order accurate, but require a structured grid (or a collection of structured grids). Spectral methods are even more accurate, but have severe restrictions on the geometry and, in the Fourier case, also require periodic boundary conditions. Finite element methods are highly flexible, but it is hard to achieve high-order accuracy and both coding and mesh generation become increasingly difficult when the number of space dimensions increases [1].

A fairly new approach for solving PDEs is through radial basis functions (RBFs). An RBF depends only on the distance to a center point \( x_j \) and is of the form \( \phi(\|x - x_j\|) \). The RBF may also have a shape parameter \( c \), in which case \( \phi(r, c) \). Some of the most popular RBFs are given in Table 1 of Section 2.

A key feature of an RBF method is that it does not require a grid. The only geometric properties that are used in an RBF approximation are the pairwise distances between points. Distances are easy to compute in any number of space dimensions, so working in higher dimensions does not increase the difficulty. The method works with points scattered throughout the domain of interest, and the RBF interpolant is a linear combination of RBFs centred at the scattered points.
Let \( x_j, s(x, c) = \sum_{i=1}^{N} \lambda_i \phi(\|x - x_i\|, c) \), where the coefficients \( \lambda_i \) are usually determined by collocation with given discrete data, such as function values or derivative information. When infinitely smooth RBFs are used, the approximations feature spectral convergence as the points get denser. This has been proven strictly only for some special cases [2,3]; although numerical evidence strongly suggests that it is true in much more general settings. Furthermore, implementation of an RBF method is straightforward. However, there are some remaining issues such as computational efficiency, and stability if applied to time-dependent problems without viscosity [1].

In the last decade, the development of the RBFs as a truly meshless method for approximating the solutions of PDEs has drawn the attention of many researchers in science and engineering [4–6]. One of the domain-type meshless methods, the so-called Kansa’s method developed by Kansa in 1990 [7,8], is obtained by directly collocating the RBFs, particularly the multiquadric (MQ), for the numerical approximation of the solution.

The accuracy of the RBF-based meshless methods greatly depends on the user defined radial basis centres and the so-called shape parameter \( c \). There is a rich class of RBFs that can be chosen, and Gaussian (GS) represents one of the most popular RBFs due to their rapid convergence rate. With relatively few radial basis centres, GS can produce very accurate results. However, one of the drawbacks of using GS is that the accuracy of the RBF-based meshless methods is severely influenced by \( c \). As \( c \) gets smaller, these RBFs become flat and the resulting matrix becomes ill-conditioned [9]. The accuracy of the GS continues to improve as \( c \) decreases; however, when \( c \) is sufficiently small, the accuracy gets worse and eventually breaks down. The determination of a reasonable \( c \) is still an outstanding research problem. Over the last two decades; some progress has been reported to select a usable shape parameter [10–13].

More recent research conducted mostly by Fornberg and co-workers (see, e.g., [11,14–17]) has shown that one may be able to overcome the conditioning problems of the traditional RBF approach by using other techniques such as the Contour-Pade algorithm of [15]. This research has confirmed that—even when circumventing the ill-conditioning of the system matrix—there usually is a value of the shape parameter which results in optimal approximation errors.

In summary, regardless of whether one follows the traditional RBF approach (and therefore looks for a good balance between accuracy and stability), or whether one applies stabilization techniques such as the Contour-Pade algorithm (which is applicable only to rather small problems), the flexibility and potential for improved accuracy offered by the shape parameter present in many RBFs should be exploited by the user [18].

In this paper, we deal with finding a reasonable shape parameter by taking a different approach. The purpose is to use a search algorithm using genetic optimization to determine a good shape parameter \( c \). Since the resulting matrix using GS is seriously ill-conditioned for small \( c \), we also consider implementing truncated singular value decomposition (TSVD) as a regularization technique other than the standard linear solver (LU decomposition) [9]. In this study, we test the genetic optimization on the Kansa’s method to determine a good \( c \). In [19], the genetic algorithm was used to find good shape parameters in interpolation problem.

The layout of the article is as follows: In Section 2 we show that how we use the radial basis functions to approximate the solution. Then we briefly review the genetic optimization algorithm in next section. In Section 4, we introduce the promising genetic optimization algorithm for finding a good shape parameter for solving partial differential equations by Kansa’s method. The results of numerical experiments are presented in Section 5. In Section 6, we draw conclusions based on the numerical results obtained in Section 5 and address our future work direction. We have computed the numerical results by using the genetic algorithm toolbox in MATLAB.

2. Radial basis function approximation

In this section the RBFs method is defined as a technique for interpolation of the scattered data. Some well-known radial basis functions (RBFs) are listed in Table 1. Let \( r \) be the Euclidean distance between a fixed point \( x' \in \mathbb{R}^d \) and any \( x \in \mathbb{R}^d \) i.e. \( \|x - x'\|_2 \). A radial function \( \phi = \phi(\|x - x'\|_2) \) depends only on the distance between \( x \in \mathbb{R}^d \) and fixed point \( x' \in \mathbb{R}^d \). This property concludes that the radial basis function \( \phi \) is radially symmetric about \( x' \). It is clear that the functions in Table 1 are globally supported, infinitely differentiable and depend on a free parameter \( c \).

Let \( x_1, x_2, \ldots, x_N \) be a given set of distinct points in \( \mathbb{R}^d \). The main idea behind the use of RBFs is interpolation by translating a single function i.e. the RBFs interpolation is considered as

\[
F(x) = \sum_{i=1}^{N} \lambda_i \phi_i(x),
\]

where \( \phi \) is a radial basis function.

**Table 1**

<table>
<thead>
<tr>
<th>Name of function</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multiquadrics (MQ)</td>
<td>( \phi(x) = \sqrt{</td>
</tr>
<tr>
<td>Inverse multiquadrics (IMQ)</td>
<td>( \phi(x) = \left( \frac{1}{</td>
</tr>
<tr>
<td>Gaussian (GS)</td>
<td>( \phi(x) = \exp(-c</td>
</tr>
</tbody>
</table>
where \( \phi_i(x) = \phi(||x - x_i||) \) and \( \lambda_i, \ i = 1, \ldots, N \), are unknown scalars. For instance, consider the given values \( f_i = f(x_i), i = 1, \ldots, N \). To compute the unknown scalars \( \lambda_i \), we impose \( F(x_i) = f_j \) for \( j = 1, \ldots, N \) which it can be written as the following linear system equations

\[
Az = f,
\]

where \( A_{ij} = \phi(x_j), z = [\lambda_1, \ldots, \lambda_N], f = [f_1, \ldots, f_N] \).

Since most applicable \( \phi \) have global support, this method produces a dense matrix \( A \). The matrix \( A \) can be shown to be positive definite (and therefore nonsingular) for distinct interpolation points for GS and IMQ by Schoenberg’s Theorem [20]. Also using the Micchelli Theorem [21] we can show that \( A \) is invertible for distinct sets of the scattered points in the case of MQ.

Although the matrix \( A \) is nonsingular in the above cases, usually it is very ill-conditioned i.e. the condition number of \( A \)

\[
\kappa_s(A) = \frac{\|A\| \cdot \|A^{-1}\|}{\|s\|}, \quad s = 1, 2, \infty,
\]

is a very large number. Therefore a small perturbation in initial data may produce a large amount of perturbation in the solution [22].

We suppose that the partial differential equation \( Lu(x) = f(x) \) with Dirichlet boundary conditions is a well-posed equation and has a unique solution.

If \( f(x) \in L^2(\mathbb{R}^2) \) then the solution of \( Lu(x) = f(x) \) belongs to \( L^2(\mathbb{R}^2) \) where \( L^2(\mathbb{R}^2) \) is a Hilbert space. If \( \{p_i(x)\} \) is an orthonormal basis for \( L^2(\mathbb{R}^2) \), then one can prove that each \( u(x) \in L^2(\mathbb{R}^2) \) has the representation

\[
u(x) = \sum_{i=1}^{\infty} \lambda_i p_i(x).
\]

This means that (2.4) is convergent to \( u(x) \) by definition:

\[
\left\| u(x) - \sum_{i=1}^{N} \lambda_i p_i(x) \right\| \to 0, \quad \text{as } N \to \infty.
\]

But about radial basis function we do not have this situation in general. For example, Gaussian RBFs with variable shape parameters does not form an orthonormal basis for \( L^2(\mathbb{R}^2) \) [23]. Now, the expansion property for radial basis functions is investigated.

The family of RBFs considered here consists of functions \( q : \mathbb{R}^r \to \mathbb{R} \) represented by

\[
q(x) = \sum_{i=1}^{N} w_i K\left( \frac{x - z_i}{\sigma} \right),
\]

where \( N \in \mathbb{N}, \sigma > 0, w_i \in \mathbb{R} \) and \( z_i \in \mathbb{R}^r \) for \( i = 1, \ldots, M \). We call this family \( S_K \).

**Theorem 1.** Let \( K : \mathbb{R}^r \to \mathbb{R} \) be an integrable bounded function such that \( K \) is continuous almost everywhere and \( \int_{\mathbb{R}^r} K(x)dx \neq 0 \). Then the family \( S_K \) is dense in \( L^p(\mathbb{R}^r) \) for every \( p \in [1, \infty) \).

**Proof** [24].

Note that there is no requirement for radial symmetry of the kernel function \( K \) in the above theorem. Thus, the theorem is stronger than necessary for radial basis functions, and might be useful for other purposes [24].

By \( K \) radially symmetric, we mean that \( ||x||_2 = ||y||_2 \) implies \( K(x) = K(y) \).

In this case, the function \( g : [0, \infty) \to \mathbb{R} \) is obtained by defining \( g(d) = K(z) \). Where \( z \) is any element of \( \mathbb{R}^r \) such that \( ||z||_2 = d \).

Therefore, in the case of radial symmetry, Eq. (2.6) can be written as

\[
q(x) = \sum_{i=1}^{N} w_i K\left( \frac{x - z_i}{\sigma} \right) = \sum_{i=1}^{N} w_i g\left( \frac{||x - z_i||_2}{\sigma} \right).
\]

If we choose the function \( K \) such that

\[
K\left( \frac{x - z_i}{\sigma} \right) = g\left( \frac{||x - z_i||_2}{\sigma} \right) = e^{-\left( \frac{||x - z_i||_2}{\sigma} \right)^2},
\]

Then the Gaussian radial basis functions have been introduced and all conditions in this theorem have been satisfied, thus, the family \( S_K \) is dense in \( L^4(\mathbb{R}^2) \) and it means that for each \( u \in L^2(\mathbb{R}^2) \) we have

\[
\left\| \sum_{i=1}^{N} w_i g\left( \frac{||x - z_i||_2}{\sigma} \right) - u \right\| \to 0, \quad \text{as } N \to \infty.
\]
3. Genetic optimization algorithm

The genetic algorithm (GA) is an optimization technique that is categorized as a global search heuristic. Genetic algorithms are based on an abstraction of the natural evolutionary behaviour which was originally proposed in [25]. They are a robust and flexible approach that can be applied to a wide range of optimization problems (see, for example, [26–28]). Main aim of GA is to achieve better results by removing bad results during production of population from current generation to next generation and using only good results to achieve the better results. The fitness function that defines what is the “better” has to be prepared very carefully depending on the problem. Possible solutions in GA are presented by chromosomes and generally the first solutions are produced randomly. Chromosomes (individuals) together generate a set of solution populations.

The fitness function formed properly to the problem, presents a solution quality of the individuals. To produce new and good solutions, GA uses operators such as selection, crossover and mutation. To produce new generation crossover and mutation operators are applied on the two individuals that are selected from population by selection mechanism [29]. The genetic algorithm uses the individuals in the current generation to create the children that make up the next generation. Besides elite children, which correspond to the individuals in the current generation with the best fitness values, the algorithm creates crossover children by selecting vector entries, or genes, from a pair of individuals in the current generation and combines them to form a child and mutation children by applying random changes to a single individual in the current generation to create a child. Both processes are essential to the genetic algorithm. Crossover enables the algorithm to extract the best genes from different individuals and recombine them into potentially superior children. Mutation adds to the diversity of a population and thereby increases the likelihood that the algorithm will generate individuals with better fitness values. Crossover fraction specifies the fraction of the next generation that crossover produces. Mutation produces the remaining individuals in the next generation.

You can specify how many of each type of children the algorithm creates as follows:

Elite count, in reproduction options, specifies the number of elite children. Crossover fraction, in reproduction options, specifies the fraction of the population, other than elite children, that are crossover children.

The pseudo code of the general genetic algorithm is as follows [30]:

Algorithm 1. The pseudo code of the general genetic algorithm

Generate an initial population;
Evaluate fitness of individuals in the population;
while A satisfactory solution has been found do
  Select parents from the population;
  Recombine (mate (crossover and mutation operators)) parents to produce children;
  Evaluate fitness of the children;
  Replace some or all of the population by the children;
end while

4. Algorithm based on genetic optimization for finding a good shape parameter in Kansa’s method

We are now ready to propose an efficient algorithm based on genetic optimization for finding a good shape parameter in Kansa’s collocation method. Assume that the domain $\Omega \subset \mathbb{R}^d$ and the partial differential equation of the form

$$L[u](x) = f(x), \quad x \in \Omega$$  \hspace{1cm} (4.1)

are given with (for simplicity of description) Dirichlet boundary conditions

$$u(x) = g(x), \quad x \in \partial \Omega$$  \hspace{1cm} (4.2)

We expand $u(x)$ by radial basis functions, i.e.

$$\tilde{u}(x) = \sum_{i=1}^{N} \lambda_i \phi(||x - \xi_i||),$$  \hspace{1cm} (4.3)

where the points $\xi_1, \ldots, \xi_N$ are a set of centres for the radial basis functions which are usually selected to coincide with the collocation points $\chi = \{x_1, \ldots, x_N\} \subset \Omega$ (in our study, centres and collocation points are the same). The collocation matrix which arises when matching the differential equation (4.1) and the boundary conditions (4.2) at the collocation points $\chi$ has below form

$$A = \begin{pmatrix} \Phi & L[\Phi] \end{pmatrix},$$  \hspace{1cm} (4.4)
where the two blocks are generated as follows:

\[ \Phi_{jk} = \phi(||x_j - \zeta_k||), \quad x_j \in B, \quad \zeta_k \in \chi, \]  
(4.5)

and

\[ L[\Phi]_{jk} = L[\phi](||x_j - \zeta_k||), \quad x_j \in I, \quad \zeta_k \in \chi. \]  
(4.6)

The set \( \chi \) will be split into a set \( I \) of interior points, and \( B \) of boundary points. The problem is well-posed if the linear system \( Ac - y \), with \( y \) as a vector consisting of entries \( g(x_j) \), \( x_j \in B \), followed by \( f(x_i) \), \( x_i \in I \), has a unique solution. The collocation matrix \( A \) has not been proven to be non-singular but in [31] it has been shown that finding a numerically singular matrix is very rare [32].

The accuracy of Kansa’s method depends profoundly on the choice of the shape parameter of RBFs. Determining a good shape parameter is still an outstanding research topic. A typical way to determine a reasonable \( c \) is to minimize the residual error in some test points (we choose test points to differ from the collocation points). In practice, the information of the exact solution inside the domain is not available. Alternatively, a more practical way to measure the numerical error is to calculate the residual error in the interior points using the governing equation and in the boundary points by Dirichlet boundary condition in the following sense:

\[ R = \sqrt{\frac{\sum_{j=1}^{n_i} (L[\bar{u}](y_j) - f(y_j))^2}{n_i}} + \sum_{j=n_i+1}^{n_i+n_b} |\bar{u}(y_j) - g(y_j)|. \]  
(4.7)

where \( n_i \) is the number of the interior test points \( \{y_j\}_{j=1}^{n_i} \) in \( \Omega \) and \( n_b \) is the number of the boundary test points \( \{y_j\}_{j=n_i+1}^{n_i+n_b} \) on \( \partial\Omega \) and \( \bar{u} \) denote the approximate solution. In the Genetic algorithm, we use \( R \) as a fitness function in the computation. The fitness function \( R \) is summation of RMS error in the interior test points and norm one of error in the boundary test points. Experimentally, we found that this fitness function works very well in genetic algorithm.

A wide range of algorithms and mathematical techniques are available to solve minimization problem. Traditional optimization methods can be classified into two distinct groups: direct and gradient-based methods [33]. In direct search methods, only objective function and constraints values are used to guide the search strategy, whereas gradient-based methods use the first and/or second-order derivatives of the objective function and/or constraints to guide the search process. Since derivative information is not used, the direct search methods are usually slow, requiring many function evaluations for convergence. For the same reason, they can also be applied to many problems without a major change of the algorithm. On the other hand, gradient-based methods quickly converge to an optimal solution, but are not efficient in non-differentiable or discontinuous problems. In addition, there are some common difficulties with most of the traditional direct and gradient-based techniques:

1- The convergence to an optimal solution depends on the chosen initial solution.

2- Most algorithms tend to get stuck to a suboptimal solution.

3- An algorithm efficient in solving one optimization problem may not be efficient in solving a different optimization problem.

4- Algorithms are not efficient in handling problems having discrete variables.

5- Algorithms cannot be efficiently used on a parallel machine.

Because of the nonlinearities and complex interactions among problem variables often exist in engineering design problems, the search space may have more than one optimal solutions, of which most are undesired locally optimal solutions having inferior objective function values. When solving these problems, if traditional methods get attracted to any of these locally optimal solutions, there is no escape.

In all, traditional searching methods are not good candidates as efficient optimization algorithms. GAs are search and optimization procedures that are motivated by the principles of natural genetics and natural selection. Some fundamental ideas of genetics are borrowed and used artificially to construct search algorithms that are robust and require minimal problem information. Over the last decade, genetic algorithms have been extensively used as search and optimization tools in various problem domains, including sciences, commerce, and engineering [34]. The primary reasons for their success are their broad applicability, ease of use, and global perspective [35].

In our study, the population size is 5, the elite count is 1, and the crossover fraction is 0.8, the numbers of each type of children in the next generation are as follows:

There is one elite child. There are 4 individuals other than elite child, so the algorithm rounds from 0.8 * 4 = 3.2 to 3 to get the number of crossover children. The remaining one individual, other than elite child, is mutation child.

We chose the initial range between 0 and 2 and search the shape parameter between 0 and 5. The proposed GAs are terminated after about 50 generations.
5. Numerical results

To demonstrate the effectiveness and stability of the genetic optimization method proposed in the last section, several examples for solving the equation in 2D spaces are presented in this section. To validate the numerical accuracy, we calculate the RMS error in 1600 points that are distributed uniformly in the computational domain. The RMS error of the numerical results is defined as follow:

$$RMS = \sqrt{\frac{1}{1600} \sum_{i=1}^{1600} (u(t_i) - \tilde{u}(t_i))^2},$$

(5.1)

where $u(x)$ is the exact solution, $\tilde{u}(x)$ is the approximate solution and $\{t_i\}_{i=1}^{1600}$ are evaluation points. In this section, the truncated singular value decomposition (TSVD) is employed to solve the linear systems. In all examples the collocation points are Chebyshev data sites in 2D and the test points are uniformly gridded data sites in 2D. In all tables $N$ is the number of collocation points and $M$ is the number of test points. Throughout the rest of this paper, the notation $c'$ stands for a good shape parameter that is obtained by the proposed algorithm in previous section. Also, note that in all tables we keep just four decimal places after the decimal points of the good shape parameter.

Table 2
Results provided by proposed algorithm for Example 1.

<table>
<thead>
<tr>
<th>The analytical solution</th>
<th>$N$</th>
<th>$M$</th>
<th>RMS</th>
<th>Condition number</th>
<th>$c'$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$u_1(x,y)$</td>
<td>49</td>
<td>100</td>
<td>$5 \times 10^{-6}$</td>
<td>$1 \times 10^{16}$</td>
<td>0.9033</td>
</tr>
<tr>
<td>$u_2(x,y)$</td>
<td>81</td>
<td>144</td>
<td>$2 \times 10^{-5}$</td>
<td>$1 \times 10^{17}$</td>
<td>1.4908</td>
</tr>
<tr>
<td>$u_3(x,y)$</td>
<td>36</td>
<td>81</td>
<td>$1 \times 10^{-6}$</td>
<td>$1 \times 10^{19}$</td>
<td>0.3024</td>
</tr>
<tr>
<td>$u_4(x,y)$</td>
<td>36</td>
<td>81</td>
<td>$5 \times 10^{-6}$</td>
<td>$1 \times 10^{13}$</td>
<td>0.8524</td>
</tr>
</tbody>
</table>

Table 3
Results provided by the different population size in Example 1.

<table>
<thead>
<tr>
<th>The analytical solution</th>
<th>$N$, $M$</th>
<th>Population size</th>
<th>RMS</th>
<th>$c'$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$u_1(x,y)$</td>
<td>81, 144</td>
<td>5</td>
<td>$2 \times 10^{-5}$</td>
<td>1.4908</td>
</tr>
<tr>
<td>$u_1(x,y)$</td>
<td>81, 144</td>
<td>10</td>
<td>$7 \times 10^{-5}$</td>
<td>1.4775</td>
</tr>
<tr>
<td>$u_1(x,y)$</td>
<td>81, 144</td>
<td>15</td>
<td>$3 \times 10^{-5}$</td>
<td>1.4881</td>
</tr>
</tbody>
</table>

Fig. 1. Approximate solution (left) using 81 Collocation points (right) with the good shape parameter for Example 1.

Table 4
Results provided by proposed algorithm for Example 2.

<table>
<thead>
<tr>
<th>$N$</th>
<th>$M$</th>
<th>RMS</th>
<th>Condition number</th>
<th>$c'$</th>
</tr>
</thead>
<tbody>
<tr>
<td>49</td>
<td>100</td>
<td>$5 \times 10^{-6}$</td>
<td>$7 \times 10^{19}$</td>
<td>0.6294</td>
</tr>
<tr>
<td>64</td>
<td>121</td>
<td>$1 \times 10^{-5}$</td>
<td>$2 \times 10^{18}$</td>
<td>0.9420</td>
</tr>
<tr>
<td>81</td>
<td>144</td>
<td>$6 \times 10^{-6}$</td>
<td>$6 \times 10^{18}$</td>
<td>1.1919</td>
</tr>
</tbody>
</table>
It must be noted that the real coded GAs in MATLAB global optimization toolbox (the default case) are used in all examples. Moreover, the value of population size is 5 in all examples because the numerical results show that the performance of GA is not very sensitive to the population size and the consume time increases significantly, when the population size is higher.

Example 1. Consider the following Poisson problem with Dirichlet boundary conditions

\[
\nabla^2 u(x, y) = f(x, y), \quad (x, y) \in \Omega = [0, 1] \times [0, 1],
\]

\[
u(x, y) = g(x, y), \quad (x, y) \in \partial \Omega,
\]

and \(f\) and \(g\) are chosen according to the following analytical solutions [1, 12, 36],

\[
u_1(x, y) = \sin(\pi x) \cos\left(\frac{\pi y}{2}\right),
\]

\[
u_2(x, y) = \cos(2\pi(x + y)),
\]

\[
u_3(x, y) = \frac{25}{25 + (x - 2)^2 + 2y^2},
\]

\[
u_4(x, y) = \exp\left(-\left((x - 0.1)^2 + 0.5y^2\right)\right).
\]

We have applied the proposed algorithm presented in the previous section to four analytical solutions (5.4)–(5.7). Good shape parameters for different solutions are shown in Table 2, which have quite different values of \(c\).

To investigate the sensitivity of the GA to the population size, different sizes of population are used in the proposed algorithm, and results are shown in Table 3.

Fig. 1 shows the profiles of the approximate solution using Gaussian radial basis functions with the good shape parameter for which the analytical solution is \(\cos(2\pi(x + y))\).

Example 2. Consider the following 2D modified Helmholtz equation

\[
\nabla^2 u(x, y) - u(x, y) = f(x, y), \quad (x, y) \in \Omega = [0, 1] \times [0, 1],
\]

\[
u(x, y) = g(x, y), \quad (x, y) \in \partial \Omega,
\]

and \(f\) and \(g\) are chosen according to the following analytical solution [9].
The new approach based on the genetic algorithm for various values of the collocation points is applied to find a good shape parameter. In Table 4, the RMS error and a good shape parameter are presented for 49, 64 and 81 collocation points. The numerical results show that the values for $c^*$ are quite different as we change the number of collocation points.

To investigate the sensitivity of the GA to the population size, different sizes of population are used in the proposed algorithm, and results are shown in Table 5.

The maximum error of approximate solution using 81 collocation points with the good shape parameter is shown in Fig. 2.

Example 3. Consider the following 2D convection–diffusion equation

$$\nabla^2 u(x,y) + (x^2 + y^2) u(x,y) + y \cos \left( \frac{\pi y}{C_0} \right) \sinh \left( \frac{\pi x}{C_0} \right) \frac{\partial u}{\partial x} + \sinh \left( \frac{\pi x}{C_0} \right) \frac{\partial u}{\partial x} = f(x,y), \quad (x,y) \in \Omega, \quad \nabla^2 u(x,y) + (x^2 + y^2) u(x,y) + y \cos \left( \frac{\pi y}{C_0} \right) \sinh \left( \frac{\pi x}{C_0} \right) \frac{\partial u}{\partial x} + \sinh \left( \frac{\pi x}{C_0} \right) \frac{\partial u}{\partial x} = f(x,y), \quad (x,y) \in \Omega, (5.11)$$

$$u(x,y) = g(x,y), \quad (x,y) \in \partial \Omega, \quad (5.12)$$

and where $f$ and $g$ are generated from the function [9],

$$u(x,y) = \sin \left( \frac{\pi x}{C_0} \right) \cosh \left( \frac{\pi y}{C_0} \right) - \cos \left( \frac{\pi x}{C_0} \right) \sinh \left( \frac{\pi y}{C_0} \right) \quad (5.13)$$

We solve this equation in different computational domains, see Table 6. The values for the good shape parameter are quite different as we change the computational domain. In large domains, we require more collocation points to get desirable accuracy.

### Table 6
Results provided by proposed algorithm for Example 3.

<table>
<thead>
<tr>
<th>$N$</th>
<th>$M$</th>
<th>RMS</th>
<th>Condition number</th>
<th>$c^*$</th>
<th>Computational domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>64</td>
<td>121</td>
<td>$1 \times 10^{-5}$</td>
<td>$1 \times 10^{18}$</td>
<td>0.9803</td>
<td>[0.1] x [0.1]</td>
</tr>
<tr>
<td>81</td>
<td>144</td>
<td>$8 \times 10^{-5}$</td>
<td>$6 \times 10^{16}$</td>
<td>0.7544</td>
<td>[0.2] x [0.2]</td>
</tr>
<tr>
<td>121</td>
<td>196</td>
<td>$2 \times 10^{-4}$</td>
<td>$5 \times 10^{18}$</td>
<td>0.6294</td>
<td>[0.3] x [0.3]</td>
</tr>
</tbody>
</table>

### Table 7
Results obtained by arbitrary shape parameter for Example 3.

<table>
<thead>
<tr>
<th>$N$</th>
<th>RMS</th>
<th>Condition number</th>
<th>Shape parameter</th>
<th>Computational domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>121</td>
<td>$3 \times 10^{-3}$</td>
<td>$2 \times 10^{19}$</td>
<td>0.5</td>
<td>[0.3] x [0.3]</td>
</tr>
<tr>
<td>121</td>
<td>$4 \times 10^{-3}$</td>
<td>$1 \times 10^{13}$</td>
<td>1</td>
<td>[0.3] x [0.3]</td>
</tr>
<tr>
<td>121</td>
<td>$2 \times 10^{-1}$</td>
<td>$2 \times 10^{7}$</td>
<td>2</td>
<td>[0.3] x [0.3]</td>
</tr>
</tbody>
</table>

Fig. 3. Approximate solution (left) using 121 collocation points (right) with the good shape parameter for Example 3.
In Table 7, we present the results obtained with different values of the shape parameter by 121 collocation points.

Fig. 3. shows the profiles of the approximate solution using Gaussian radial basis functions with the good shape parameter.

**Example 4.** Consider the following elliptic equation with variable coefficients and homogeneous Dirichlet boundary conditions on \( \Omega = [0, 1] \times [0, 1] \):

\[
\frac{\partial}{\partial x} \left( a(x, y) \frac{\partial}{\partial x} u(x, y) \right) + \frac{\partial}{\partial y} \left( b(x, y) \frac{\partial}{\partial y} u(x, y) \right) = f(x, y), \quad (x, y) \in \Omega,
\]

\[u(x, y) = 0, \quad (x, y) \in \partial\Omega,
\]

Where

\[
f(x, y) = -16x(1-x)(3-2y)e^{x-y} + 32y(1-y)(3x^2 + y^2 - x - 2),
\]

and the coefficients are given by

\[
a(x, y) = 2 - x^2 - y^2, \quad b(x, y) = e^{x-y}.
\]

The exact solution for this problem is given by [36],

\[
u(x, y) = 16x(1-x)(1-y).
\]

The proposed algorithm presented in the previous section is applied to find a good shape parameter. The numerical results obtained using 25 collocation points are shown in Table 8.

In Table 9, some arbitrary values of the shape parameter \( c \), the corresponding RMS error and condition number of the matrix, which is resulted from discretization of the problem are listed for 25 collocation points. Results in Table 9. show the dependence of the Kansa’s method to the shape parameter. In Fig. 4, the approximate solution and the maximum error using Gaussian radial basis functions with 25 collocation points and the good shape parameter are plotted.

6. Conclusion

The accuracy of Kansa’s method depends profoundly on the choice of the shape parameter of RBFs. Determining a good shape parameter is still an outstanding research topic. In this paper, an efficient way to select a good shape parameter is presented when the Gaussian radial basis function is used. Most significantly it is possible to select, if not an “optimal,” at least a reliably good value for the parameter \( c \). Moreover, this can be computed efficiently by using the genetic optimization algorithm which does not require any knowledge of the exact solution of the PDE.
A typical way to determine a reasonable $c$ is to minimizing the residual error in some test points. Most optimization problems have nonlinear and complex interactions among variables; therefore the search space may have many locally optimal solutions. When solving these problems, if traditional methods get attracted to any of these locally optimal solutions, there is no escape. Therefore, traditional searching methods are not good candidates as efficient optimization algorithms. The genetic algorithm is a method for solving both constrained and unconstrained optimization problems that is based on natural selection, the process that drives biological evolution. Genetic algorithms have been extensively used as search and optimization tools in various problem domains, including science, commerce, and engineering. The primary reasons for their success are their broad applicability, ease of use, and global perspective. However, the large computational time is the major drawback to use the genetic algorithm.

The numerical results show that the values for the good shape parameter are quite different as we change the number of collocation points. Also, with different computational domains the values for the good shape parameter are different.

Clear, there is still much to be done regarding the optimal choice of RBFs shape parameters. For example, many researchers have suggested the use of variable shape parameters (e.g., [37]). On the one hand, following this suggestion provides a clear potential for improved accuracy and stability of the RBF method. Extension of the proposed method to solve partial differential equations by Kansa’s method using variable shape parameters is the subject of a future research work.

References


