Chapter

Modified Moving Least Squares Method for Two-Dimensional Linear and Nonlinear Systems of Integral Equations

Massoumeh Poura'bd Rokn Saraei and Mashaallah Matinfar

Abstract

This work aims at focusing on modifying the moving least squares (MMLS) methods for solving two-dimensional linear and nonlinear systems of integral equations and system of differential equations. The modified shape function is our main aim, so for computing the shape function based on the moving least squares method (MLS), an efficient algorithm is presented. In this modification, additional terms is proposed to impose based on the coefficients of the polynomial base functions on the quadratic base functions (m = 2). So, the MMLS method is developed for solving the systems of two-dimensional linear and nonlinear integral equations at irregularly distributed nodes. This approach prevents the singular moment matrix in the context of MLS based on meshfree methods. Also, determining the best radius of the support domain of a field node is an open problem for MLS-based methods. Therefore, the next important thing is that the MMLS algorithm can automatically find the best neighborhood radius for each node. Then, numerical examples are presented that determine the main motivators for doing this so. These examples enable us to make comparisons of two methods: MMLS and classical MLS.

Keywords: moving least squares, modified moving least squares, systems of integral equations, algorithm of shape function, numerical solutions

MSC 2010: 45G15, 45F05,45F35, 65D15

1. Introduction

In mathematics, there are many functional equations of the description of a real system in the natural sciences (such as physics, biology, Earth science, meteorology) and disciplines of engineering. For instance, we can point to some mathematical model from physics that describe heat as a partial differential equation and the inverse problem of it's as integro-differential equations. Also, another example in nature is Laplace's equation which corresponds to the construction of potential for a vector field whose effect is known at the boundary of Domain alone. Especially, the integral equations have wide applicability which has been cited in [1–4].

However, there are many significant analytical methods for solving integral equations but most of them especially in nonlinear cases, finding an analytical representation of the solution is so difficult, therefore, it is required to obtain approximate solutions. The interested reader can find several numerical methods for approximating the solution of these problems in [5–14] and the references therein.

Moreover, there are various numerical and analytical methods have been used to estimate the solution of integrodifferential equations or Abels integral equations [12, 15–18]. Recently the meshless based methods, particularly Moving Least Squares (MLS) method, for a solution of partial differential equations and ordinary differential equations have been paid attention. Using this approach some new methods such as meshless local boundary integral equation method [19], Boundary Node Method (BNM) [20], moving least square reproducing polynomial meshless method [21] and other relative methods are constructed. The new class of meshless methods has been developed which only relied on a set of nodes without the need for an additional mesh in the solution of a one-dimensional system of integral equations [22].

A local approximation of unknown function presented in the MLS method give us to possible choose the compact support domain for each data point as a sphere or a parallelogram box centered on a point [23, 24]. So each data point has an associated with the size of its compact support domain as dilatation parameter. Therefore the number of data point and dilatation parameter are direct effects on the MLS, Also by increasing the degree of the polynomial base function for complex data distributions give a more validated fashion. Nevertheless, in this case, it becomes more difficult to ensure the independence of the shape functions, and the least-squares minimization problem becomes ill-posed.

The common solution for increased the number of admissible node distribution is increasing the size of the support domains (a valid node distribution is referred to as an œadmissible node distribution [23]). There have been several proposed for choosing the radius of support domain [25], but one of the efficient suggestion was raised by Chen shen [26]. The author in [27] has introduced a new algorithm for selecting the suitable radius of the domain of influence. Also in [28], presented a modified MLS(MMLS) approximation on the shape function generation algorithm with additional terms based on the coefficients of the polynomial basis functions. It is an efficient method which has been proposed for handling a singular moment matrix in the MLS based methods. The advantage of this method compared to methods based on mesh such as a finite element or finite volume is this the domain of the problem is not important because this approximation method is based on a set of scattered points instead of domain elements for interpolation or approximation. So the geometry of the domain does not interfere in the MLS.

2. Methodology

2.1 Introduction of the MLS approximation

The Moving Least Square (MLS) method is a feasible numerical approximation method that is an extension of the least squares method, also it is the component of the class of meshless schemes that have a highly accurate approximation. The MlS approximation method is a popular method used in the many meshless methods [12, 19, 21, 22, 29, 30]. In many procedures used to construct the MLS shape function is used support-domain concept. The support domain of the shape function is a set of nodes in the problem domain that just those points directly contributes to the construction of the shape function, so the MLS shape function is locally supported. According to the classical least squares method, an optimization problem should be solved as follows

$$min\left(\sum_{j=1}^{m}\left(u^h(\mathbf{x}_j)-u(\mathbf{x}_j)\right)^2\right)$$

Where Ideally the approximation function $u^h(x)$ should match the function $u(\mathbf{x})$. Therefore, in the MLS approach, a weight optimization problem will be solved which is dependent on nodal points. We start, with the basic idea of taking a set of the nodal points in Ω so that $\Omega \subseteq \mathbb{R}^d$. Also $\Omega_{\mathbf{x}} \subseteq \Omega$ is neighboring nodes of point $\mathbf{x}$ and finding an approximation function with m basis functions, in a system with n equations as

$$T(U)=F$$

where T consists of linear and nonlinear operators and $U=(u_1,u_2,\dots,u_n)$ is the unknown vector of functions, also $F=\left(f_1,f_2,\dots,f_n\right)$ is the known vector of functions.

So for the approximation of any of the $u_i, i=1,2,\dots,n$ in $\Omega_{\mathbf{x}}$, $\forall x \in \Omega_{\mathbf{x}}$, $u_i^h(\mathbf{x})$ can be defined as

$$u_i^h(\mathbf{x})=\sum_{j=1}^{m}a_j(\mathbf{x})p_j(\mathbf{x})=P^T(\mathbf{x})a(\mathbf{x}). \tag{1}$$

Let $\mathbf{P}=\{p_1,p_2,\dots p_m\}$ a set of polynomial of degree at most $m, m\in\mathbb{N}$. Let $\mathbf{a}(\mathbf{x})$ is a vector containing unknown coefficients $a_j(\mathbf{x}), j=1,2,\dots m$ dependent on the intrest point position. Also m unknown functions $\mathbf{a}(\mathbf{x})=(a_1(\mathbf{x}),a_2(x),\dots a_m(\mathbf{x}))$ are chosen such that:

$$J(\mathbf{x})=\sum_{j=1}^{m}\left(\mathbf{P}^T(\mathbf{x}_j)\mathbf{a}(\mathbf{x})-u_i(\mathbf{x}_j)\right)^2 w_i(\mathbf{x})=[P.\mathbf{a}-\mathbf{u}_i]^T.W.[P.\mathbf{a}-\mathbf{u}_i], \tag{2}$$

is minimized. Note that the weight function $w_i(\mathbf{x})$ is associated with node j. As we know, each redial basis function that define in [31] can be used as weight function, we can define $w_j(r)=\phi\left(\frac{r}{\delta}\right)$ where $r=\|\mathbf{x}-\mathbf{x}_i\|_2$ (the Euclidean distance between $\mathbf{x}$ and $\mathbf{x}_j$) and $\phi:\mathbb{R}^d\to\mathbb{R}$ is a nonnegative function with compact support. In this chapter, we will use following weight functions and will compare them to each other, corresponding to the node j, in the numerical examples.

a. Guass weight function

$$w(r)=\begin{cases}\dfrac{\exp\left(\dfrac{-r^2}{c^2}\right)-\exp\left(\dfrac{-\delta^2}{c^2}\right)}{1-\exp\left(\dfrac{-\delta^2}{c^2}\right)} & 0\leq r\leq\delta\\ 0 & elsewhere.\end{cases} \tag{3}$$

b. RBF weight function

$$w(r) = \begin{cases} (1-r)^6(6+36r+82r^2+72r^3+30r^4+5r^5) & 0 \le r \le \delta \\ 0 & elsewhere. \end{cases} \tag{4}$$

c. Spline weight function

$$w(r) = \begin{cases} 1-6\left(\frac{r}{\delta}\right)^2+8\left(\frac{r}{\delta}\right)^3-3\left(\frac{r}{\delta}\right)^4 & 0 \le r \le \delta \\ 0 & elsewhere. \end{cases} \tag{5}$$

Where c is constant and is called shape parameter. Also δ is the size of support domain.

N is the number of nodes in $\Omega_{\mathbf{x}}$ with $w_i(x) > 0$, the matrices P and W are defined as

$$P = \left[\mathbf{p}^T(\mathbf{x}_1), \mathbf{p}^T(\mathbf{x}_2), \dots \mathbf{p}^T(\mathbf{x}_N)\right]^T_{N\times(m+1)} \tag{6}$$

$$W = diag((w_i(\mathbf{x})), i = 1, 2, \dots, N \tag{7}$$

and

$$\mathbf{u}^h = \left[u_1^h, u_2^h, \dots u_n^h\right]. \tag{8}$$

It is important to note that $u_i^T, i = 1, 2, \dots n,$ in (2) and (8) are the artificial nodal values, and not the nodal values of the unknown trial function $u^h(\mathbf{x})$ in general. With respect to $\mathbf{a}(\mathbf{x})$ and u_i^T will be obtained,

$$A(\mathbf{x})a(\mathbf{x}) = B(\mathbf{x})\mathbf{u}_i, \tag{9}$$

where the matrices $A(\mathbf{x})$ and $B(\mathbf{x})$ are defined by:

$$B(\mathbf{x}) = [w_1\mathbf{p}(\mathbf{x}_1), w_2\mathbf{p}(\mathbf{x}_2), \dots, w_N\mathbf{p}(\mathbf{x}_N)] \tag{10}$$

$$A(\mathbf{x}) = \sum_{i=1}^{N} w_i(\mathbf{x})\mathbf{p}^T(\mathbf{x}_i)\mathbf{p}(\mathbf{x}_i) = \mathbf{p}^T(\mathbf{x})w(\mathbf{x})\mathbf{p}(\mathbf{x}). \tag{11}$$

The matrix $A(\mathbf{x})$ in (11) is non-singular when the rank of matrix $P(\mathbf{x})$ equals to m and vice versa. In such a case, the MLS approximation is well-defined. With computing $\mathbf{a}(\mathbf{x}), u_i^h$ can be obtained as follows:

$$u_i^h(\mathbf{x}) = \sum_{j=1}^{N} \phi_j(\mathbf{x})u_i(\mathbf{x}_j) = \varphi^T.\mathbf{u}_i \tag{12}$$

$\phi_j(\mathbf{x})$ is called the shape function of the MLS approximation corresponding to the nodal point $\mathbf{x}_j$, where

$$\varphi(\mathbf{x}) = \mathbf{p}^T(\mathbf{x})A^{-1}(\mathbf{x})B(\mathbf{x}) \tag{13}$$

Also with use the weight function, matrix A, B are computed and then $\phi_i(x)$ is determined from (13), If, further, ϕ is sufficiently smooth, derivatives of U are usually approximated by derivatives of U^h,

$$D^\alpha u_i \mathbf{x} \approx D^\alpha u_i^h(\mathbf{x}) = \sum_{j=1}^{N} D^\alpha a_j(\mathbf{x}) u_i(\mathbf{x}_j), x \in \Omega \tag{14}$$

2.2 Modify algorithm of MLS shape function

In the MLS approximation method, a local evaluation of the approximating unknown function is desired, and therefore for any nodal points the compact support domain is chosen as a sphere or a parallelogram box centered on the point [23, 29, 32]. This finding which the support domains contain what points. Each data point has a connected dilatation parameter λ which is given $\delta_i = \lambda h_i$. Also, δ_i is the size of compact support domain in a node point $\mathbf{x}_i$.

Also, the necessary condition for that the moment matrix A be nonsingular is that for any point $x_i \in \Omega, i = 1, 2, \dots, N$, [31].

$$\aleph(\{j | x_j \in \Omega_{\delta_i}\}) \geq m,\ j = 1, 2, \dots, N$$

So the dilatation parameters λ determine the number of points of support domain, Also these points participate in the production of the shape function Therefore, λ is very important and should be chosencorrectly so that the moment matrix A is nonsingular.

In the remainder of this section, we introduce the new algorithm, with the aim of avoiding the singularity of the matrix A by choosing the correct λ parameter by the algorithm.

Algorithm 1

Require: $X = \{x_i : i = 1, 2, \dots, N\}$- Coordinates of points whose MLS shape function to be evaluated.

```
1: procedure Matrix A
2:   $\lambda_{new} \leftarrow \lambda$
3:   $\alpha \leftarrow 0.01$ (This value selected experimentally.)
4:   $\delta = \lambda_{new} \times h$ (h: the fill distance is defined to be $h = \sup_{x \in \Omega} \min_{1 \leq j \leq N} \|x - x_i\|_2$)
5:   Loop
6:   set $I(x) = \{j \in \{1, 2, \dots, N\}, \|x - x_i\|_2 \leq \delta\}$ (Using set of indices $I$, by localization at a fixed point $x$)
7:   for $j \in I(x)$ do
8:     for $i = 1 : N$ do
9:       Compute $w_i$ for any $x_j \in \Omega_i$
10:      $A = A + w_i p_i^2$
11:    end
12:  end
13:  if $cond(A) \geq \frac{1}{\varepsilon}$ then
14:    {
15:    $\lambda_{new} = \lambda_{new} + \alpha\lambda$
16:    $\delta = \lambda_{new} \times h$
17:  else
18:    goto end
19:    }
20:  if $\delta_i \leq \|X_\Omega\|_2$ then
21:    goto Loop
22:  end
```

In Algorithm 1.

X: is a set containing N scattered points which are called centers or data site and I (x) is the Index of points which MLS shape function is evaluated.

α: is a small positive number that is selected experimentally.

Then in every node points, matrix A is computed.

By running the algorithm the new value is assigned to λ, this value is related to the condition number of matrix A and its amount will increase. Therefore, the size of the support domain is increased and then the matrix A with new nodal points in the support domain is reproduced. This loop is repeated until $\frac{1}{cond(A)} \geq \varepsilon$.

The main idea of the moving least squares approximation is that for every point x can solve a locally weighted least squares problem [30], it is a local approximation, thus the additional condition to stop the loop is the size of the local support domain, the value of λ should be well enough to pave the local approximation, Line 20 is said to satisfy this condition.

2.3 Modified MLS approximation method

One of the common problems in Classic MLS method is the singularity of the moment matrix A in irregularity chosen nodal points. To avoid the nodal configurations which lead to a singular moment matrix, the usual solution is to enlarge the support domains of any nodal point. But it is not an appropriate solution, in [31] to tackle such problems is proposed a modified Moving least squares(MMLS)approximation method. This modifies allows, base functions in $m \geq 2$ to be used with the same size of the support domain as linear base functions ($m = 1$). We should note that,impose additional terms based on the coefficients of the polynomial base functions is the main view of the modified technique. As we know, in the basis function $\mathbf{p}(\mathbf{x})$ is

$$\mathbf{p}(\mathbf{x}) = \left[1, x, x^2, \dots, x^m\right]^T \tag{15}$$

where $\mathbf{x} \in \mathbb{R}$, Also the correspond coefficients a_j, that should be determined are [24]:

$$\mathbf{a}(\mathbf{x}) = [a_1, a_x, a_{x^2}, \dots, a_{x^m}]^T \tag{16}$$

For obtaining these coefficients, the functional (2) rewrite as follows:

$$\bar{J}(\mathbf{x}) = \sum_{j=1}^{m} \left(\mathbf{P}^T(\mathbf{x}_j)\mathbf{a}(\mathbf{x}) - u_i(\mathbf{x}_j)\right)^2 w_i(\mathbf{x}) + \sum_{\nu=1}^{m-2} \overline{w}_\nu(x)\bar{\mathbf{a}}_\nu^2(\mathbf{x}), i = 1, 2, \dots, n \tag{17}$$

Where $\overline{w}$ is a vector of positive weights for the additional constraints, also $\bar{\mathbf{a}} = [a_{x^2}, a_{x^3}, \dots, a_{x^m}]^T$ is the modified matrix.

The matrix form of (17) is as follows:

$$\bar{J}(\mathbf{x}) = [P.\mathbf{a} - \mathbf{u}_i]^T.W.[P.\mathbf{a} - \mathbf{u}_i] + \mathbf{a}^T H\mathbf{a}, i = 1, 2, \dots, n \tag{18}$$

where H is as,

$$H = \begin{bmatrix} O_{2,2} & O_{m-2,m-2} \\ O_{2,2} & diag(\overline{w}) \end{bmatrix}, \tag{19}$$

where, $O_{i,j}$ is the null matrix. By minimizing the functional (18), the coefficients $a(\mathbf{x})$ will be obtained. So we have

$$\overline{A}(\mathbf{x})\mathbf{a}(x) = B(\mathbf{x})\mathbf{u}_i, \tag{20}$$

where

$$\overline{A} = P^T.W.P + H \tag{21}$$

And the matrics $B(\mathbf{x})$ is determined as the same before. So we have

$$\varphi_m(\mathbf{x}) = \mathbf{a}(\mathbf{x}) = p^T(\mathbf{x})\overline{A}^{-1}(\mathbf{x})B(\mathbf{x}) \tag{22}$$

where $\varphi_m(\mathbf{x})$ is the shape function of the MMLS approximation method.

3. Stiff systems of ordinary differential equations

In this section, we use MLS approximation method for numerical solution of the Stiff system of ordinary differential equations so consider the following differential equation

$$A(U) - F(\mathbf{x}) = 0, U(0) = U_0, \mathbf{x} \in \Omega \tag{23}$$

with boundary conditions,

$$B\left(U, \frac{\partial U}{\partial \mathbf{x}}\right) = 0, \mathbf{x} \in \partial\Omega.$$

where A is a general differential operator, U_0 is an initial approximation of (23), $F(\mathbf{x})$ is a vector of known analytical functions on the domain Ω and $\partial\Omega$ is the boundary of Ω. The operator can be divided by $A = L + N,$ where L is the linear part, and N is the nonlinear part of its. So (23) can be, rewritten as follows;

$$L(U) + N(U) - F(\mathbf{x}) = 0 \tag{24}$$

We assume that $\mathbf{a} = \{a_1, a_2, \ldots, a_m\}$ are the MLS shape functions so in order to solve system (24), N nodal points x_i are selected on the Ω, which $\{x_i | i = 1, 2, \ldots, N\}$ is q-unisolvent. The distribution of nodes could be selected regularly or randomly. Then instead of u_j from U, we can replace u_j^h from (13). So we have

$$u_j^h(\mathbf{x}) = \sum_{i=1}^{N} a_i(\mathbf{x}) u_j(\mathbf{x}_i) \tag{25}$$

where $j = 1, 2, \ldots, n$ is the number of unknown functions. we estimate the unknown functions u_i by (25), so the system (24) becomes the following form

$$L\left(\sum_{i=1}^{N} a_i(\mathbf{x})u_1(\mathbf{x}_i), \sum_{i=1}^{N} a_i(\mathbf{x})u_2(\mathbf{x}_i), \ldots, \sum_{i=1}^{N} a_i(\mathbf{x})u_n(\mathbf{x}_i)\right) +$$
$$N\left(\sum_{i=1}^{N} a_i(\mathbf{x})u_1(\mathbf{x}_i), \sum_{i=1}^{N} a_i(\mathbf{x})u_2(\mathbf{x}_i), \ldots, \sum_{i=1}^{N} a_i(\mathbf{x})u_n(\mathbf{x}_i)\right) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_n(\mathbf{x})) + \mathbf{r}(\mathbf{x}). \tag{26}$$

where $\mathbf{r}(\mathbf{x})$ is residual error function which vanishes to zero in collocation points thus by installing the collocation points $\mathbf{y}_r; r = 1, 2, \dots, N$, so

$$\begin{aligned}
&L\left(\sum_{i=1}^{N} a_i(\mathbf{y}_r)u_1(\mathbf{x}_i), \sum_{i=1}^{N} a_i(\mathbf{y}_r)u_2(\mathbf{x}_i), \dots, \sum_{i=1}^{N} a_i(\mathbf{y}_r)u_n(\mathbf{x}_i)\right)+\\
&N\left(\sum_{i=1}^{N} a_i(\mathbf{y}_r)u_1(\mathbf{x}_i), \sum_{i=1}^{N} a_i(\mathbf{y}_r)u_2(\mathbf{x}_i), \dots, \sum_{i=1}^{N} a_i(\mathbf{y}_r)u_n(\mathbf{x}_i)\right) =\\
&\sum_{i=1}^{N} L(a_i(\mathbf{y}_r))u_1(\mathbf{x}_i), \sum_{i=1}^{N} L(a_i(\mathbf{y}_r))u_2(\mathbf{x}_i), \dots, \sum_{i=1}^{N} L(a_i(\mathbf{y}_r))u_n(\mathbf{x}_i))+\\
&N\left(\sum_{i=1}^{N} a_i(\mathbf{y}_r)u_1(x_i), \sum_{i=1}^{N} a_i(\mathbf{y}_r)u_2(\mathbf{x}_i), \dots, \sum_{i=1}^{N} a_i(\mathbf{y}_r)u_n(\mathbf{x}_i)\right) =\\
&(f_1(\mathbf{y}_r), f_2(y_r), \dots, f_n(\mathbf{y}_r))
\end{aligned} \tag{27}$$

therefore

$$CU = \begin{bmatrix} L(a_1(y_1)) & L(a_2(y_1)) & \dots & L(a_N(y_1)) \\ L(a_1(y_2)) & L(a_2(y_2)) & \dots & L(a_N(y_2)) \\ \vdots & & & \\ L(a_1(y_N)) & L(a_2(y_N)) & \dots & L(a_N(y_N)) \end{bmatrix} \begin{bmatrix} u_1(x_1) & u_2(x_1) & \dots & u_n(x_1) \\ u_1(x_2) & u_2(x_2) & \dots & u_n(x_2) \\ \vdots & & & \\ u_1(x_N) & u_2(x_N) & \dots & u_n(x_N) \end{bmatrix} \tag{28}$$

And the matrix form of (27) as follows

$$C_{N\times N}U_{N\times n} + N_{N\times n}(\mathbf{a}, U) = F_{N\times n}(y_r) \tag{29}$$

where

$$\begin{aligned}
C_i &= [L(a_1(\mathbf{y}_r)), \dots, L(a_N(\mathbf{y}_r))]_{i=1}^{n}\\
U_i &= \left[(u_i(x_1), u_i(x_2), \dots, u_i(x_N))^T\right]_{i=1}^{n}\\
F(\mathbf{y}_r) &= \left(\left[(f_1(\mathbf{y}_r))_{r=1}^{N}\right]^T, \left[(f_2(\mathbf{y}_r))_{r=1}^{N}\right]^T, \dots \left[(f_n(\mathbf{y}_r))_{r=1}^{N}\right]\right)^T.
\end{aligned} \tag{30}$$

by imposing the initial conditions at $t = 0$, we have

$$\left(\sum_{i=1}^{N} a_i(0)u_1(t_i), \sum_{i=1}^{N} a_i(0)u_2(t_i), \dots, \sum_{i=1}^{N} a_i(0)u_n(t_i)\right) = U_0 \tag{31}$$

and Solving the non-linear system (29) and (31), lead to finding quantities $u_j(x_i)$. Then the values of $u_j(x)$ at any point $x \in \Omega$, can be approximated by Eq. (25) as:

$$u_j(\mathbf{x}) \simeq \sum_{i=1}^{N} a_i(\mathbf{x})u_j(\mathbf{x}_i)$$

Remark

Note that, for simplicity, the modification derivation is made for bivariate data, but can be easily extended to higher dimensions. Also, for implementation, the modified moving least squares approximation method it is sufficient to use φ_m instead of φ.

3.1 Error analysis

The convergence analysis of the method in matrix norm has been investigated for the systems of one and two-dimensional Fredholm integral equations by authors of [22]. It should be noted that The convergence analysis of the method for implementation classic moving least squares approximation method on a system of integral equations has been discussed and it should be investigated for modified Mls method. we can use the results for this type of equations.

So in continuation of this section, the error estimations for the system of differential equations is developed. In [26], has obtained error estimates for moving least square approximations in the one-dimensional case. Also in [33], is developed for functional in n-dimensional and was used the error estimates to prove an error estimate in Galerkin coercive problems. In this work, have improved error estimate for the systems of stiff ordinary differential equations.

Given $\delta > 0$ let $W_\delta \geq 0$ be a function such that $supp(w_\delta) \subset \overline{B_\delta(0)} = \{z \| z| \leq \delta\}$ and $X_\delta = \{x_1, x_2, \dots, x_n\}$, $n = n(\delta)$, a set of points in $\Omega \subset \mathbb{R}$ an open interval and let $U = (u_1, u_2, \dots, u_N)$ be the unknown function such that $u_{i1}, u_{i2}, \dots, u_{in}$ be the values of the function u_i in those points, i.e., $u_{i,j} = u_i(x_j), i = 1, \dots, N, j = 1, \dots, n$. A class of functions $W = \{\omega_j\}_{j=1}^N$ is called a partition of unity subordinated to the open covering I_N if it possesses the following properties:

- $W_j \in C_s^0, s > 0 \, or \, s = \infty,$
- $\sup(\omega_j) \subseteq \overline{\Lambda}_j,$
- $\omega_j(x) > 0, x \in \Lambda_j,$
- $\sum_{i=1}^{N} \omega_j = 1 \, for \;\; every \, x \in \overline{\Omega}$

There is no unique way to build a partition of unity as defined above [34].

As we know, the MLS approximation is well defined if we have a unique solution at every point $x \in \overline{\Omega}$. for minimization problem. And non-singularity of matrix $A(x)$, ensuring it is. In [33] the error estimate was obtained with the following assumption about the system of nodes and weight functions $\{\Theta_N, W_N\}$:

We define

$$\langle u, v \rangle = \sum_{j=1}^{n} w(x - x_j) u(x_j) v(x_j)$$

then

$$\|u\|_x^2 = \sum_{j=1}^{n} w(x - x_j) u(x_j)^2$$

Also for vector of unknown functions, we define

$$\|U\|_\infty = \, max \left\{ |u_i|_x, i = 1, 2, \dots, N \right\}$$

are the discrete norm on the polynomial space $\mathbb{P}_m^1$ if the weight function w satisfy the following properties.

a. For each $x \in \Omega$, $w(x - x_j) > 0$ at least for $(m + 1)$ indices j.

b. For any $x \in \Omega$, the moment matrix $A(x) = w(x)P^T$ is nonsingular.

Definition 3.1. *Given* $\boldsymbol{x} \in \overline{\Omega}$, *the set* $ST(\boldsymbol{x}) = \{j : \omega_j \neq 0\}$ *will be called the star of* x.

Theorem 3.1. *[34, 35] A necessary condition for the satisfaction of Property* ***b*** *is that for any* $\boldsymbol{x} \in \overline{\Omega}$

$$n = card(ST(\boldsymbol{x})) \geq card(\mathbb{P}_m) = m + 1$$

For a sample point $\mathbf{c} \in \overline{\Omega}$, if $ST(\mathbf{c}) = \{j_1, \ldots j_k\}$, the mesh-size of the star $ST(\mathbf{c})$ defined by the number is $h(ST(\mathbf{c})) = \max\{h_{j1}, \ldots h_{jk}\}$.

Assumptions. Consider the following global assumptions about parameters. There exist

(a_1) An over bound of the overlap of clouds:

$$E = \sup_{c \in \overline{\Omega}}\{card(ST(c))\}.$$

(a_2) Upper bounds of the condition number:

$$CB_q = \sup_{c \in \overline{\Omega}}\{CN_q(ST(c)), q = 1, 2\}.$$

where the numbers $CN_q(ST(\mathbf{c}))$ are computable measures of the quality of the star $ST(c)$ which defined in Theorem 7 of [19].

(a_3) An upper bound of the mesh-size of stars:

$$R = \sup_{c \in \overline{\Omega}}(hST(c)).$$

(a_4) An uniform bound of the derivatives of $\{w_j\}$. That is the constant $G_q > 0, q = 1, 2$, such that

$$\left\|D^\mu W_j\right\|_{L_\infty} \leq \frac{G_q}{R^{|\mu|}} \quad 1 < \mu < q,$$

(a_5) There exist the number $\gamma > 0$ such that any two points $\mathbf{x}, \mathbf{y} \in \Omega$ can be joined by a rectifiable curve Γ in Ω with length $|\Gamma| \leq \gamma\|\mathbf{x}\text{-}\mathbf{y}\|$.

Assuming all these conditions, Zuppa [34] proved.

Lemma 3.1. $U = (u_1, u_2, \ldots u_n)$ *such that* $u_i \in C^{m+1}(\overline{\Omega})$ *and* $\|U\|_\infty = u_k, \ 1 < k < n$, *There exist constants* $C_q, q = 1$ *or* 2,

$$C_1 = C_1(\gamma, d, E, G_1, CB_1),$$

$$C_2 = C_1(\gamma, d, E, G_2, CB_1, CB_2),$$

then

$$\left\|D^\mu U - D^\mu U^h\right\|_\infty < C_q R^{q+1-|\mu|} \|u_k^{(m+1)}\|_{L^\infty(\Omega)} \quad 0 < \mu < q \tag{32}$$

Proof: see [36].

3.2 System of ODE

If in (24) the non-linear operator N be zero, we have

$$L(U) = (f_1, f_2, \dots, f_n) \tag{33}$$

where U is the vector of unknown function and L is a matrix of derivative operators,

$$L(U(.)) = \sum_{\varsigma=1}^{n} \lambda_\varsigma \frac{\partial^\varsigma}{(.)^\varsigma} U(.). \tag{34}$$

And from (25), we define

$$U^h(t) = \sum_{i=1}^{N} a_i(t) U(t_i)$$

where $(a_i)_{i=1}^N$ are the MLs shape functions defined on the interval $[0, 1]$ satisfying the homogeneous counterparts of the boundary conditions in (23). Also if the weight function w possesses k continuous derivatives then the shape functions a_j is also in C^k [33]. By the collocation method, is obtained an approximate solution $U^h(t)$. And demand that

$$L^h(U(.)) = \sum_{\varsigma=0}^{n} \lambda_\varsigma \frac{\partial^\varsigma}{(.)^\varsigma} U^h(.) \tag{35}$$

where ($\lambda = 0$ *or* 1). It is assumed that in the system of ODE derivative of order at most $n = 2$. Each of the basis functions a_i has compact support contained in $(0, 1)$ then the matrix C in (30) is a bounded matrix. If δ be fixed, independent of N, then the resulting system of linear equations can be solved in $O(N)$ arithmetic operations.

Lemma 3.2. *Let* $U = (u_1, u_2, \dots u_n)$ *and* $F = (f_1, f_2, \dots f_n)$ *so that* $u_i \in C^{m+1}(\overline{\Omega})$ $m \geq 1$ *and* $\|u_i\|_\infty = u_k, k \in \{1, 2, \dots, n\}$ *where* Ω *be a closed, bounded set in R. Assume the quadrature scheme is convergent for all continuous functions on* Ω*. Further, assume that the stiff system of ODE (23) with the fixd initial condition is uniquely solvable for given* $f_i \in C(\Omega)$*. Moreover take a suitable approximation* U^h *of U Then for all sufficiently large n, the approximate matrix L for linearly case exist and are uniformly bounded,* $|L| \leq M$ *with a suitable constant* $M < \infty$*. For the equations* $L(U) = F$ *and* $L^h(U) = F$ *we have*

$$E_t = \|L(U(t)) - L^h(U(t))\|_\infty$$

so that

$$\|E_t\|_\infty \leq C_q K(\lambda, \varsigma) R^{m+1-\mu} \|u_k^{(m+1)}\|_{L_\infty}.$$

Proof. we have

$$\|L(U(t)) - L^h(U(t))\|_\infty = \|\sum_{\varsigma=0}^{n} \lambda_\varsigma \frac{\partial^\varsigma}{t^\varsigma} U(t) - \sum_{\varsigma=0}^{n} \lambda_\varsigma \frac{\partial^\varsigma}{t^\varsigma} U^h(t)\|_\infty$$

so due to the lemma (36),

$$\|L(U(t)) - L^h(U(t))\|_\infty \le \sum_{\varsigma=0}^{n} |\lambda_\varsigma| \| \frac{\partial^\varsigma}{t^\varsigma} U(t) - \frac{\partial^\varsigma}{t^\varsigma} U^h(t)\|_\infty$$
$$\le \max_i \sum_{\varsigma=0}^{n} |\lambda_\varsigma| | \frac{\partial^\varsigma}{t^\varsigma} u_i(t) - \frac{\partial^\varsigma}{t^\varsigma} u_i^h(t)|$$
$$\le \sum_{\varsigma=0}^{n} C_q |\lambda_\varsigma| \|u_k^{(m+1)}\|_{L_\infty} R^{m+1-\varsigma}$$

where should be $m \ge \varsigma$ so,

$$\sum_{\varsigma=0}^{n} |\lambda_\varsigma| R^{m+1-\varsigma} \le K(\lambda, \varsigma) R^{m+1-\mu}$$

where μ is the highest order derivative And $K(\lambda, \varsigma) = \sum_{\varsigma=0}^{n} |\lambda_\varsigma|$, so demanded that

$$\|E_t\|_\infty \le C_q K(\lambda, \varsigma) R^{m+1-\mu} \|u_k^{(m+1)}\|_{L_\infty}.$$

It should be noted that in the nonlinear system the upper bound of error depends on the nonlinear operator.

4. Two-dimensional linear systems of integral equations

4.1 Fredholm type

In this section, we will provide an approximation solution of the 2-D linear system of Fredholm integral equations by the MLS method. The matrix form of this system could be considered as

$$\mathbf{U}(x,y) = \mathbf{F}(x,y) + \int_\Omega K(x,y,\theta,s)\mathbf{U}(\theta,s)d\theta ds, \quad (x,y) \in \Omega, \tag{36}$$

where $\Omega = [a,b] \times [c,d]$ as $\Omega \subset \mathbb{R}^2$, Also $K(x,y,\theta,s) = [\kappa_{ij}(x,y,\theta,s)]$, $i,j = 1, 2, \dots, n$ is the matrix of kernels, $\mathbf{U}(x,y) = (u_1(x,y), u_2(x,y), \dots u_n(x,y))^T$ is the vector of unknown function and $\mathbf{F}(x,y) = (f_1(x,y), f_2(x,y), \dots f_n(x,y))^T$ is the vector of known functions.

In addition, is took that two cases for the domain, the rectangular shape, and nonrectangular one and three cases relative to the geometry of the nonrectangular domain are considered where can be transformed into the rectangular shape [35].

The first one is $\Omega = \{(\theta,s) \in \mathbb{R}^2 : a \le s \le b, g_1(s) \le \theta \le g_2(s)\}$ where $g_1(s)$ and $g_2(s)$ are continues functions of s, the second one can be consider as $\Omega = \{(\theta,s) \in \mathbb{R}^2 : c \le \theta \le d, g_1(\theta) \le s \le g_2(\theta)\}$ where $g_1(\theta)$ and $g_2(\theta)$ are continues functions of θ, Also the last one is a domain which is neither of the first nor the second kinds but could be separated to finite numbers of first or second sub-domains, it is labeled as a domain of third kind. Without loss of generality, the first kind domain can be assumed as

$$\Omega = \{(\theta, s) \in \mathbb{R}^2 : -1 \le s \le 1, g_1(s) \le \theta \le g_2(s)\} \tag{37}$$

by the following linear transformation

$$\theta(t, s) = \frac{g_2(\theta) - g_1(\theta)}{2} t + \frac{g_2(\theta) - g_1(\theta)}{2}, \tag{38}$$

the interval $[g_1(\theta), g_2(\theta)]$ is converted to the fixed interval $[-1, 1]$, so we have

$$\mathbf{U}(x,y) = \mathbf{F}(x,y) + \int_{-1}^{1}\int_{-1}^{1} \mathrm{K}(x,y,t,s)\mathbf{U}(t,s)dtds,\ \textit{such that}(x,y) \in [-1,1] \times [-1,1] \tag{39}$$

where

$$\mathrm{K}(x,y,t,s) = \frac{g_2(\theta) - g_1(\theta)}{2} \mathrm{K}(x,y,\theta,s) \tag{40}$$

Also, the second kind is straight similarly by commuting the variables and the third kind can be separated to finite numbers of sub-domains of the first or second kinds, so the method can be applied in each sub-domain as described earlier.

So, for the numerical integration $\Omega = \bigcup_{l=1}^{L} \Omega_l$ and $\Omega_l \bigcap \Omega_k \neq \varnothing$, $1 \le k, l \le L$

$$\int_{\Omega} g(s)ds = \sum_{l=1}^{L} \int_{\Omega_l} g(s)ds \tag{41}$$

Here, the MLS method is applied for the general case where the domain is $[a,b] \times [c,d]$.

To apply the method, as described in section 2.1, instead of u_i from U, we can replace u_i^h from (12). So we have

$$U^h(\mathbf{x}) = \left(u_1^h(\mathbf{x}), u_2^h(\mathbf{x}), \dots, u_n^h(\mathbf{x})\right)^T \tag{42}$$

Also, obviously from (12)

$$u_j^h(x,y) = \sum_{i=1}^{N} \phi_i(x,y) u_j(x_i, y_i) \tag{43}$$

in this section, is assumed that the domain has rectangular shape, so system (36) becomes as follows

$$\left(u_1^h(x,y), u_2^h(x,y), \dots u_n^h(x,y)\right)^T = f(x,y) + \int_c^d\int_a^b [k_{ij}(x,y,t,s)] \left(u_1^h(t,s), u_2^h(t,s), \dots u_n^h(t,s)\right)^T dtds. \tag{44}$$

By substituting (42) in (44), and it holds at points $(x_r, y_r), r = 1, 2, \dots, N$ we have

$$\mathbf{f}(x_r, y_r) = \left(\sum_{i=1}^{N} \phi_i(x_r, y_r) u_1(x_i, y_i), \sum_{i=1}^{N} \phi_i(x_r, y_r) u_2(x_i, y_i), \dots \sum_{i=1}^{N} \phi_i(x_r, y_r) u_n(x_i, y_i)\right)^T - \int_c^d \int_a^b [k_{ij}(x_r, y_r, t, s)] \left(\sum_{i=1}^{N} \phi_i(t, s) u_1(x_i, y_i), \dots, \sum_{i=1}^{N} \phi_i(t, s) u_n(x_i, y_i)\right)^T dtds. \tag{45}$$

We consider the m_1-point numerical integration scheme over Ω relative to the coefficients (τ_k, ς_p) and weights ω_k and ω_p for solving integrals in (45), i.e.,

$$(\mathrm{F}_N)_j u_j(x, y) = \sum_{p=1}^{m_1} \sum_{k=1}^{m_1} \left(\sum_{i=1}^{N} k_{ji}(x_r, y_r, \tau_k, \varsigma_k) \phi_i(\tau_k, \varsigma_k) \omega_k \omega_p\right), (x, y) \in \Omega, u_i \in (-\infty, \infty) \tag{46}$$

Applying the numerical integration rule (46) yields

$$\begin{aligned}\mathbf{f}(x_r, y_r) = \Bigg(&\sum_{i=1}^{N}\left(\phi_i(x_r, y_r) - \sum_{p=1}^{m_1}\sum_{k=1}^{m_1}\left(\sum_{j=1}^{N} k_{j1}(x_r, y_r, \tau_k, \varsigma_k)\phi_i(\tau_k, \varsigma_k)\omega_k \omega p\right)\right) u_{1i},\\ &\sum_{i=1}^{N}\left(\phi_i(\tau_k, \varsigma_k) - \sum_{p=1}^{m_1}\sum_{k=1}^{m_1}\left(\sum_{j=1}^{N} k_{j2}(x_r, y_r, \tau_k, \varsigma_k)\phi_i(\tau_k, \varsigma_k)\omega_k \omega p\right)\right) u_{2i},\\ &\dots \sum_{i=1}^{N}\left(\phi_i(\tau_k, \varsigma_k) - \sum_{p=1}^{m_1}\sum_{k=1}^{m_1}\left(\sum_{j=1}^{N} k_{jn}(x_r, y_r, \tau_k, \varsigma_k)\phi_i(\tau_k, \varsigma_k)\omega_k \omega p\right)\right) u_{ni}\Bigg)^T, r = 1, 2, \dots N\end{aligned}$$

where $(u_j)_i$ are the approximate quantities of u_j when we use a quadrature rule instead of the exact integral. Now if we set F_l, $l = 1, 2, \dots n$ as a N by N matrices defined by:

$$(F_l)_{i,j} = \phi_i(x_r, y_r) - \sum_{p=1}^{m_1}\sum_{k=1}^{m_1}\left(\sum_{j=1}^{N} k_{jl}(x_r, y_r, \tau_k, \varsigma_p)\phi_i(\tau_k, \varsigma_p)\omega_k\right)\omega_p \tag{47}$$

So, the moment matrix F is defined by (47) as follows

$$\mathrm{F} = [F_1, F_2, \dots F_n]_{nN \times nN} \tag{48}$$

And

$$\begin{gathered} U = \left[(u_{11}, u_{12}, \dots u_{1N})^T, (u_{21}, u_{22}, \dots u_{2N})^T, \dots (u_{n1}, u_{n2}, \dots u_{nN})^T\right]^T \\ \mathbf{f}(x_r, y_r) = \left(\left[(f_1(x_r, y_r))_{r=1}^N\right]^T, \left[(f_2(x_r, y_r))_{r=1}^N\right]^T, \dots \left[(f_n(x_r, y_r))_{r=1}^N\right]\right)^T. \end{gathered} \tag{49}$$

So by solving the following linear system of equations with a proper numerical method such as Gauss elimination method or etc. leads to quantities, u_{ji}.

$$\mathrm{F}U = \mathbf{f} \tag{50}$$

Therefore any $u_j(x,y)$ at any arbitrary point (x,y) from the domain of the problem, can be approximated by Eq. (43) as

$$u_j(x,y) \approx \sum_{i=1}^{N} \phi_i(x,y) u_{ji}(x_i,y_i) \tag{51}$$

4.2 Volterra type

Implementation of the proposed method on the Volterra integral equations is very simple and effective. In this case, the domain under study is as $\Omega = [a,x] \times [c,y]$ such that $0 \le x \le 1$, $0 \le y \le 1$ and a, c are constant, so a Volterra system type of integral equations can be consider as

$$\mathbf{U}(x,y) = \mathbf{F}(x,y) + \int_{\Omega} \mathrm{K}(x,y,t,s)\mathbf{U}(t,s)dtds,\ (x,y) \in \Omega, \tag{52}$$

like the Fredholm type, it is the matrix form of a system, so we have

$$\begin{aligned} \mathbf{U}(x,y) &= (u_1(x,y), u_2(x,y), \ldots u_n(x,y))^T, \textit{ the vector of unknown functions} \\ \mathbf{F}(x,y) &= (f_1(x,y), f_2(x,y), \ldots f_n(x,y))^T, \textit{ the vector of known functions} \\ \mathrm{K}(x,y,t,s) &= [\kappa_{ij}(x,y,t,s)] i,j = 1,2,\ldots,n \textit{ the matrix of kernels}. \end{aligned} \tag{53}$$

By the following transformation the interval $[a,x]$ and $[c,y]$ can be transferred to a fixed interval $[a,b]$ and $[c,d]$,

$$t(x,\theta) = \frac{x-a}{b-a}\theta + \frac{b-x}{b-a}a. \tag{54}$$

$$s(y,\xi) = \frac{y-c}{d-c}\xi + \frac{d-y}{d-c}c. \tag{55}$$

Then instead of u_i from U, we can replace u_i^h from (12). So we have

$$U^h(\mathbf{x}) = \left(u_1^h(\mathbf{x}), u_2^h(\mathbf{x}), \ldots u_n^h(\mathbf{x})\right)^T \tag{56}$$

where

$$u_i^h(\mathbf{x}) = \sum_{j=1}^{N} \phi_j(\mathbf{x}) u_i(\mathbf{x}_j) \tag{57}$$

where $\mathbf{x} = (x,y) \in [a,b] \times [c,d]$, thus, system (52) becomes

$$\left(u_1^h(\mathbf{x}), u_2^h(\mathbf{x}), \ldots u_n^h(\mathbf{x})\right)^T = F(\mathbf{x}) + \int_a^{\mathbf{x}}\int_c^{\mathbf{y}} [\kappa_{ij}(x,y,t,s)].\left(u_1^h(x,y), u_2^h(x,y), \ldots u_n^h(x,y)\right)^T dtds. \tag{58}$$

Therefore from (54) and (55), the system (58) takes the following form

$$\left(u_1^h(\mathbf{x}), u_2^h(\mathbf{x}), \ldots u_n^h(\mathbf{x})\right)^T = F(\mathbf{x}) + \int_a^{\mathbf{b}}\int_c^{\mathbf{d}} [\kappa_{ij}(x,y,t,s)].\left(u_1^h(x,y), u_2^h(x,y), \ldots u_n^h(x,y)\right)^T d\theta d\xi. \tag{59}$$

Where

$$K(.,.,.,.) = \frac{x-a}{b-a}\frac{y-c}{d-c}\mathrm{K}(.,.,.,.), \tag{60}$$

Using techniques employed in the Fredholm case yields the same final linear system where

$$(F_l)_{i,j} = \phi_i\left(x_r, y_r\right) - \sum_{p=1}^{m_1}\sum_{k=1}^{m_1}\left(\sum_{j=1}^{N} k_{jl}\left(x_r, y_r, \tau_k, \varsigma_p\right)\phi_i\left(\tau_k, \varsigma_p\right)\omega_k\right)\omega_p \tag{61}$$

where $l = 1, 2, \dots, n$.

5. Nonlinear systems of two-dimensional integral equation

5.1 Fredholm type

In the nonlinear system, the unknown function cannot be written as a linear combination of the unknown variables or functions that appear in them, so the matrix form of Fredholm integral equations defined as the following form [27].

$$\mathbf{U}(x,y) = \mathbf{F}(x,y) + \int_{\Omega}\mathbf{K}(x,y,\theta,s,\mathbf{U}(\theta,s))d\theta ds, (x,y) \in \Omega, \tag{62}$$

Where $\mathbf{U}(x,y)$, K and $\mathbf{F}$ are defined as,

$$\mathbf{U}(x,y) = (u_1(x,y), u_2(x,y), \dots, u_n(x,y))^T$$

$$\mathbf{K}(x,y,\theta,s,\mathbf{U}(\theta,s)) = \left[k_{ij}(x,y,\theta,s,\mathbf{U}(\theta,s))\right], i,j = 1, 2, \dots, n$$

$$\mathbf{F} = \left(f_1, f_2, \dots, f_n\right)^T$$

As mentioned above, we assume that $\Omega = [a,b] \times [c,d]$.

To apply the aproximation MLS method, we estimate the unknown functions u_i by (12), so the system (62) becomes the following form

$$\left(u_1^h(x,y), u_2^h(x,y), \dots u_n^h(x,y)\right)^T = f(x,y) + \int_c^d\int_a^b\left[k_{ij}\left(x,y,t,s,\mathbf{U}^h(t,s)\right)\right]dtds \tag{63}$$

We consider the m_1-point numerical integration scheme over the domain under study relative to the coefficients $\left(\tau_k, \varsigma_p\right)$ and weights ω_k and ω_p for solving tow-dimentional integrals in (63), i.e.,

$$(\mathrm{F}_N)_i u_i(x,y) = \sum_{p=1}^{m_1}\sum_{k=1}^{m_1} k_{ji}\left(x, y, \tau_k, \varsigma_p, \sum_{j=1}^{N}\phi_j\left(\tau_k, \varsigma_p\right)u_i(x,y)\omega_k\omega_p\right), (x,y) \in \Omega, u_i \in (-\infty, \infty) \tag{64}$$

Applying the numerical integration rule (64) in (63) yields

$$\left(u_1^h(x_r,y_r),u_2^h(x_r,y_r),\ \dots,u_n^h(x_r,y_r)\right)^T = f(x_r,y_r)+\sum_{p=1}^{m_1}\sum_{k=1}^{m_1}\left[k_{ij}\left(x_r,y_r,\tau_k,\varsigma_p,\mathbf{U}^h\left(\tau_k,\varsigma_p\right)\right)\right]dtds \tag{65}$$

Finding unknowns $\mathbf{U}^h$ by solving the nonlinear system of algebraic Eq. (65) yields the following approximate solution at any point $(x,t)\in\Omega$.

$$u_j(x,y)\approx\sum_{i=1}^{N}\phi_i(x,y)u_{ji}(x_i,y_i) \tag{66}$$

5.2 Volterra type

Two-dimensional nonlinear system of Volterra integral equations can be considered as the following form

$$\mathbf{U}(x,y)=\mathbf{F}(x,y)+\int_a^x\int_c^y \mathrm{K}(x,y,\theta,s,\mathbf{U}(\theta,s))d\theta ds,\ (x,y)\in\Omega, \tag{67}$$

where K, **F** are known function and **U** the vector of unknown functions are defined in (63) [27]. In order to apply the MLS approximation method, as same as the linear type, the interval $[a,x]$ and $[c,y]$ transferred to a fixed interval $[a,b]$ and $[c,d]$. Then $u_i^h, i=1,2,\ \dots,n$ instead of u_i in $U=(u_1,u_2,\ \dots,u_n$ from (12) is replaced. So the nonlinear system (67) is converted to

$$\left(u_1^h(x,y),u_2^h(x,y),\ \dots,u_n^h(x,y)\right)^T = f(x,y)+\int_c^d\int_a^b \overline{K}\left(x,y,t,s,\mathbf{U}^h(t,s)\right)dtds \tag{68}$$

where

$$\overline{K}\left(x,y,t,s,\mathbf{U}^h(t,s)\right)=\frac{\mathbf{x}-a}{b-a}\frac{\mathbf{y}-c}{d-c}K\left(x,y,t,s,\mathbf{U}^h(t,s)\right), \tag{69}$$

Using the numerical integration technique (64) which applied in the Fredholm case yields the same final nonlinear system (65), so the approximation solution of **U** would be found by solving this system of equations.

6. Examples

In this section, the proposed method can be applied to the system of 2-dimensional linear and nonlinear integral equations [37] and the system of differential equations. Also, the results of the examples illustrate the effectiveness of the proposed method Also the relative errors for the collocation nodal points is used.

$$\|e_i\|_\infty=\frac{\|u_{iex}(x,y)-u_i^h(x,y)\|}{\|u_{iex}(x,y)\|}$$

where u_i^h is the approximate solution of the exact solution u_{iex}. Linear and quadratic basis functions are utilized in computations.

6.1 Example 1

As the first example, we consider the following system of nonlinear Fredholm integral equations [27].

$$u_1(x,y) = f_1(x,y) + \int_\Omega u_1(s,t)u_2(s,t)dsdt$$
$$u_2(x,y) = f_2(x,y) + \int_\Omega u_1(s,t)u_2(s,t) + u_2^2(s,t)dsdt$$

where $\Omega = [0,1] \times [0,1]$. The exact solutions are $U(x,y) = (x+y, x)$ and the $F(x,y) = \left(x+y-\frac{7}{12}, x-\frac{11}{12}\right)$. The distribution of randomly nodes is shown in **Figure 1**. By attention to the irregular nodal points distribution, unsuitable δ can lead to a singular matrix A. So in this example, the adapted algorithm can tackle such problems. The MLS and MMLS shape functions are computed by using Algorithm 1, so the exact value of the radius of the domain of influence is not important; in fact, it is chosen as an initial value.

The condition numbers of the matrix A is shown in **Figure 2** and the determinant of A at sample points p is shown in **Figure 3**, where the radius of support domains for any nodal points is started from $\delta = 0.05$. Note that, there is a different radius of support domain for any node point, it might be increased due to the inappropriate distribution of scattered points by the algorithm. These variations are shown in **Table 1** for sample points (x,y), where $Cond(A)$ is the conditions number A, its initial case ($\delta = 0.005$) and final case ($New\delta,$) and $N.O.iteration$ is the number of iteration of the algorithm for determining a suitable radius of support domain.

In computing, $\delta = 2r$ where $r = 0.05$ and $c = \frac{2}{\sqrt{3}}r$. Also In MMLS, $\overline{w}_\nu = 0.1$, $\nu = 1,2,3$. It should be noted that, these values were also selected experimentally. Relative errors of the MLS method for different Gauss-Legendre number points at $m = 1,2$ and 3 compared in **Table 2**, also investigating the proposed methods shown that increasing the number of numerical integration points does not guarantee the error decreases. jkjk.

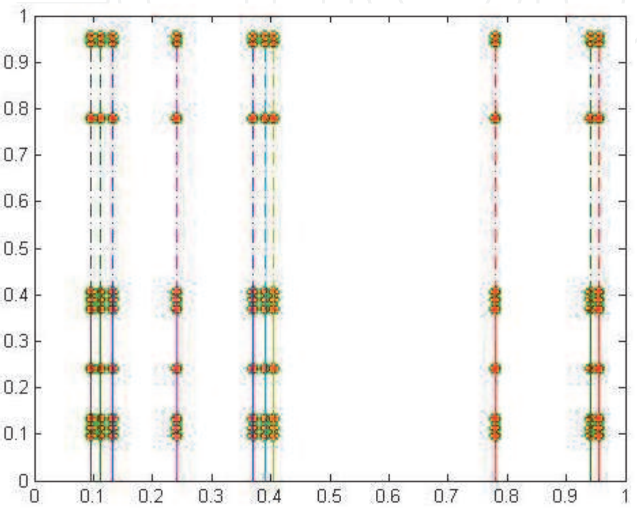

Figure 1.
The scatter data of example 1.

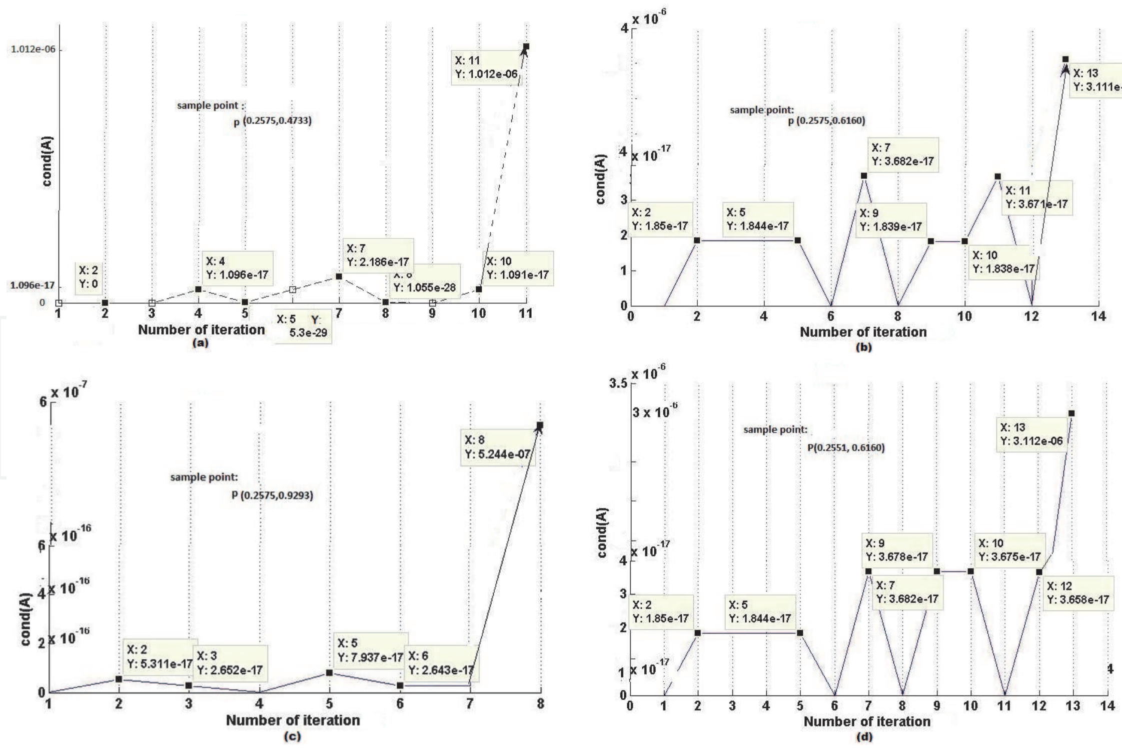

Figure 2.
The condition numbers of a at a sample point p and $\delta = 0.05$ for example 1. Using algorithm 1.

	Sample points		Cond(A)		Result of algorithm	
n	***x***	***y***	***initial***	***final***	***Newδ***	***N.O.iteration***
1	0.2575	0.4733	$1.1005e-17$	$1.0117e-06$	0.1297	11
2	0.2575	0.6160	0	$3.1111e-06$	0.1569	13
3	0.2575	0.9293	$5.3204e-17$	$5.2445e-07$	0.0974	8
4	0.2551	0.6160	0	$3.1115e-06$	0.1569	13
5	0.6991	0.2435	$2.7166e-17$	$1.4652e-07$	0.0550	2

Table 1.
Change of the radius and the condition number A at sample points (x,y) using algorithm 1, for example 1.

	m = 1 $\|e\|_\infty$		m = 2 $\|e\|_\infty$		m = 3 $\|e\|_\infty$	
N	u_1	u_2	u_1	u_2	u_1	u_2
5	6.28×10^{-4}	2.7×10^{-3}	3.45×10^{-7}	1.15×10^{-6}	3.1×10^{-6}	2.8×10^{-6}
10	1.2×10^{-4}	5.9×10^{-4}	2.46×10^{-7}	8.41×10^{-7}	2.1×10^{-7}	5.41×10^{-7}
15	2.3×10^{-4}	5.9×10^{-4}	4.47×10^{-8}	1.34×10^{-7}	4.47×10^{-8}	2.15×10^{-7}
20	2.3×10^{-4}	5.14×10^{-4}	6.12×10^{-7}	2.46×10^{-6}	3.2×10^{-7}	1.9×10^{-6}
30	3.2×10^{-4}	5.9×10^{-4}	1.84×10^{-6}	6.64×10^{-6}	2.34×10^{-6}	7.14×10^{-6}

Table 2.
Maximum relative errors for different points Gauss-Legendre quadrature rule $\delta = 2r$, for example 1.

	MMLS $\|e\|_\infty$		MLS $\|e\|_\infty$		CPU times	
N	u_1	u_2	u_1	u_2	MLS	MMLS
10	3.78×10^{-10}	8.13×10^{-10}	2.46×10^{-7}	8.41×10^{-7}	389.9574	2.9776×10^{3}
15	1.35×10^{-10}	2.00×10^{-11}	4.47×10^{-8}	1.34×10^{-7}	410.9083	2.1109×10^{3}
20	8.63×10^{-11}	2.51×10^{-9}	6.12×10^{-7}	2.46×10^{-6}	634.8373	3.2115×10^{3}
30	3.99×10^{-10}	1.58×10^{-9}	1.84×10^{-6}	6.64×10^{-6}	1.0331×10^{3}	2.5844×10^{3}

Table 3.
Compare relative errors and CPU times of MLS and MMLS for different points Gauss-Legendre quadrature rule, $(m = 2)$, for example 1.

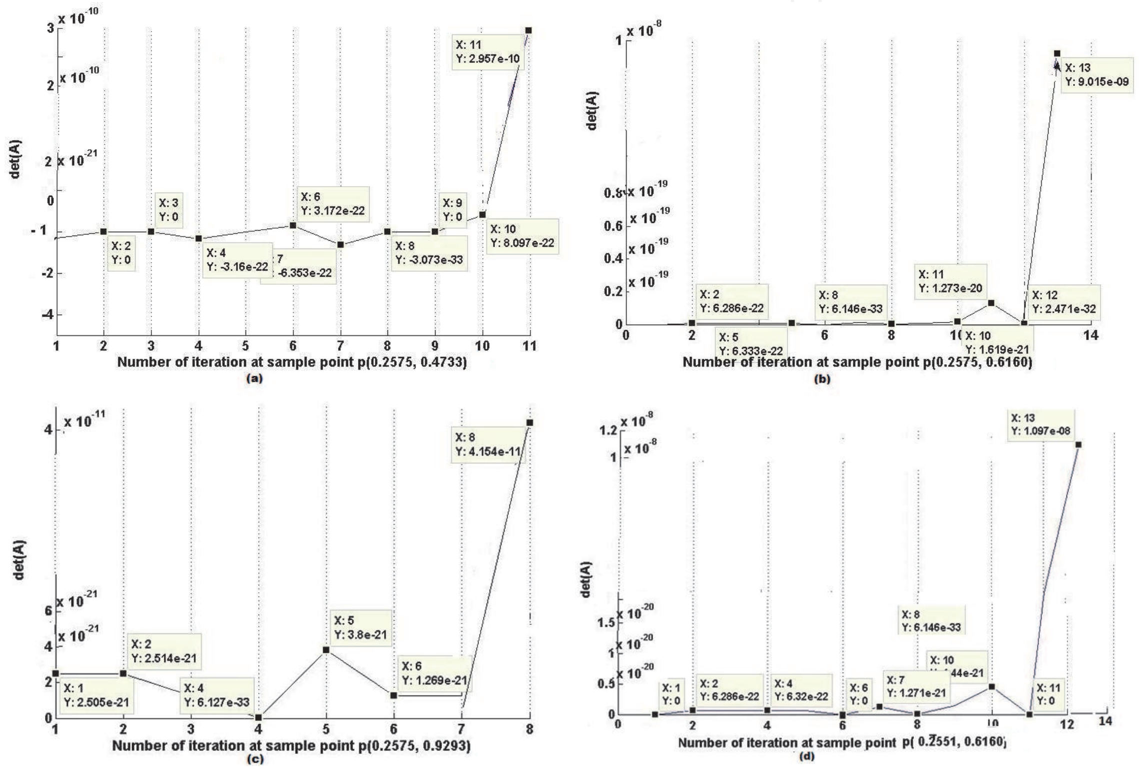

Figure 3.
The determinant of a at a sample point p and $\delta = 0.05$ for example 1. Using algorithm 1.

In **Table 3**, we can see that the CPU times for solving the nonlinear system (65) are much larger in MMLS method; but, the errors are very smaller (**Figure 3**).

6.2 Example 2

Consider the system of linear Fredholm integral equations with [27].

$$K(x, y, t, s) = \begin{pmatrix} x(t+s) & -t \\ ts & (y+x)t \end{pmatrix}, \tag{70}$$

Such that $U(x, y) = (x + y, x)$ is the vector of The exact solutions and the vector of unknown function is $F(x, y) = \left(-\frac{1}{6}(x+y) + \frac{1}{3}, \frac{4}{3}x - \frac{1}{3} + \frac{1}{3}y\right)$. Also the domain of the problem determine by $\Omega = [0, 1] \times [0, 1]$. In this example, initial value of r as radius of support domain set by 0.05. Also Algorithm 1 is used for producing shape function at $m = 1, 2, 3$. In computing, we put $\overline{w}_\nu = 0.1, \nu = 1, 2, 3$.

	m = 1			**m = 3**		
N	u_1	u_2	*CPU.T.*	u_1	u_2	*CPU.T.*
5	2.94×10^{-4}	6.02×10^{-4}	108.957	2.08×10^{-4}	2.91×10^{-4}	152.1474
10	1.79×10^{-4}	3.89×10^{-4}	159.258	1.7×10^{-4}	2.37×10^{-4}	170.7879
15	1.86×10^{-4}	3.989×10^{-4}	198.135	2.47×10^{-4}	3.30×10^{-4}	252.75424
20	1.86×10^{-4}	3.986×10^{-4}	221.321	2.41×10^{-4}	3.29×10^{-4}	247.0093
30	1.85×10^{-4}	3.987×10^{-4}	308.987	2.25×10^{-4}	3.37×10^{-4}	314.8173
40	1.85×10^{-4}	3.980×10^{-4}	395.125	1.87×10^{-4}	2.86×10^{-4}	402.9594

Table 4.
Relative errors and CPU times of MLS for different points Gauss-Legendre quadrature rule at $m = 1, 3$, for example 2.

		MLS			**MMLS**		
m	**N**	u_1	u_2	*CPU.T.*	u_1	u_2	*CPU.T.*
2	5	1.24×10^{-7}	3.89×10^{-6}	289.75	4.47×10^{-7}	6.12×10^{-6}	297.7
	10	3.16×10^{-7}	5.23×10^{-7}	189.99	2.16×10^{-8}	6.26×10^{-8}	210.09
	15	1.68×10^{-7}	4.13×10^{-7}	389.95	2.02×10^{-8}	8.12×10^{-8}	390.15
	20	1.61×10^{-7}	3.88×10^{-7}	410.90	2.04×10^{-8}	5.24×10^{-8}	425.87
	30	1.45×10^{-7}	3.80×10^{-7}	604.80	1.98×10^{-8}	3.25×10^{-8}	618.44
	40	1.44×10^{-7}	3.84×10^{-7}	689.9574	1.04×10^{-8}	6.87×10^{-8}	697.04

Table 5.
Compare relative errors and CPU times of MLS and MMLS for different points Gauss-Legendre quadrature rule at $m = 2$, for example 2.

Table 4 shows relative errors and CPU times of MLS for different Gauss-Legendre number points at $m = 1, 3$. As shown in **Table 5**, comparing the errors of MMLS and MLS method determines the capability and accuracy of the proposed technique to solve systems of linear Fredholm integral equations. This indicates the advantage of the proposed method over these systems of equations.

Comparing the errors of MMLS and MLS method determines the capability and accuracy of the proposed method to solve systems of linear Fredholm integral equations.

6.3 Example 3

The third example that we want to approximate is the system of linear Volterra-Fredholm integral equations with [27].

$$K(x, y, t, s) = \begin{pmatrix} (x + y)\exp^{(t+s)} & (x + y)\exp^{(t+s)} \\ 1 & -1 \end{pmatrix}, \tag{71}$$

The domain is considered as $\Omega = \{(x, y) \in \mathbb{R}^2 : 0 \le x \le 1, 0 \le y \le (1 - y)\}$ so that $y \in [0, 1]$. Also the exact solutions are $(\exp^{x+y}, \exp^{x-y})$. It is important to note that the linear transformation used in the experiment is only (55) and from (40) the kernel becomes

$$K(.,.,.,.) = \frac{y-c}{d-c} \mathrm{K}(.,.,.,.). \tag{72}$$

In this example, the effect of increasing radius of the domain of influence δ_i in MLS method on error has been investigated. Therefore the δ_i was considered as follows

$$\delta_i = \lambda r \; i = 1, 2, \dots, N \tag{73}$$

In this way, useful information is obtained about the performance of the proposed method. By investigating the results in **Tables 6** and 7 we found that the relative error in MLS was also related to the radius of the domain of influence (i.e. $\delta = \lambda r$ so that $\lambda = 3, 5, 7$); however, it cannot be greater than 7. For example, the relative errors by choosing $\lambda = 10$ (i.e. $\delta = 0.05\lambda$) and 10−point GaussLegendre quadrature rule are $\|e\|_{\infty u_1} = 3.3 \times 10^{-2}$ and $\|e\|_{\infty u_2} = 1.8 \times 10^{-2}$ at $m = 1$. Also, **Table 8** depicts, the number of points in the numerical integration rule cannot be effective to increase the accuracy of the method.

	$\lambda = 3\|e\|_\infty$		$\lambda = 5\|e\|_\infty$		$\lambda = 7\|e\|_\infty$	
r	u_1	u_2	u_1	u_2	u_1	u_2
0.2	9×10^{-3}	5×10^{-3}	1.9×10^{-3}	1×10^{-3}	6.02×10^{-4}	3.36×10^{-4}
0.1	1.2×10^{-3}	6.87×10^{-4}	1.34×10^{-4}	7.46×10^{-5}	4.57×10^{-6}	2.91×10^{-6}
0.05	1.44×10^{-4}	8.14×10^{-5}	2.23×10^{-5}	1.26×10^{-5}	6.27×10^{-6}	3.6×10^{-6}

Table 6.
Maximum relative errors of MLS for 10 Gauss-Legendre points and different values of $\delta = \lambda r$ at $m = 2$, for example 3.

	$\lambda = 3\|e\|_\infty$		$\lambda = 5\|e\|_\infty$		$\lambda = 7\|e\|_\infty$	
r	u_1	u_2	u_1	u_2	u_1	u_2
0.2	1.5×10^{-3}	8.77×10^{-3}	6.56×10^{-5}	3.01×10^{-5}	1.09×10^{-4}	6.19×10^{-5}
0.1	1.08×10^{-4}	6.51×10^{-5}	3.47×10^{-5}	2.08×10^{-5}	1.24×10^{-5}	7.22×10^{-6}
0.05	6.63×10^{-6}	3.93×10^{-6}	3.9×10^{-6}	2.31×10^{-6}	2.41×10^{-6}	1.38×10^{-6}

Table 7.
Maximum relative errors of MLS for 10 Gauss-Legendre points and different values of $\delta = \lambda r$ at $m = 3$ for example 3.

	$\lambda = 3\|e\|_\infty$		$\lambda = 5\|e\|_\infty$		$\lambda = 7\|e\|_\infty$	
N	u_1	u_2	u_1	u_2	u_1	u_2
5	2.2×10^{-3}	1.5×10^{-3}	6.33×10^{-5}	4.52×10^{-5}	2.21×10^{-5}	1.59×10^{-5}
10	1.1×10^{-3}	7.4×10^{-4}	4.17×10^{-5}	2.89×10^{-5}	2.96×10^{-6}	2.72×10^{-6}
15	2×10^{-3}	1.3×10^{-3}	7.9×10^{-5}	5.07×10^{-5}	6.31×10^{-6}	5.11×10^{-6}
20	2.1×10^{-3}	1.3×10^{-3}	8.06×10^{-5}	5.17×10^{-5}	3.96×10^{-6}	3.69×10^{-6}
30	2.1×10^{-3}	1.3×10^{-3}	8.16×10^{-5}	5.17×10^{-5}	4.11×10^{-6}	3.74×10^{-6}

Table 8.
Maximum relative errors of MLS for different values of $\delta = \lambda r, r = 0.05$ and Gauss-Legendre points at $m = 1$, using algorithm 1 for example 3.

	MLS		MMLS		CPU time	
r	u_1	u_2	u_1	u_2	**MLS**	**MMLS**
0.2	1.09×10^{-4}	6.19×10^{-5}	5.52×10^{-4}	3.08×10^{-4}	4.0556	3.3893
0.1	1.24×10^{-5}	1.08×10^{-6}	1.25×10^{-5}	7.36×10^{-6}	38.0616	31.7945
0.05	1.91×10^{-6}	1.085×10^{-6}	1.54×10^{-5}	8.77×10^{-6}	496.1746	409.0342

Table 9.
Compare relative errors and CPU times of MLS and MMLS for $\overline{w}_\nu = 10$ and 10 Gauss-Legendre points and different values of $\delta = 7r$ at $m = 2$ for example 3.

Then the relative error by MMLS shape function described in section (2.3) were obtained, using $\delta = 7r$ and $\overline{w}_\nu = 10$, such that $\nu = 1, 2, 3$ as weights of additional coefficients for MMLS. We can see that in **Table 9** the errors of the system of linear Volterra-Fredholm integral equations are similar in both methods (i.e. MLS and MMLS methods).

6.4 Example 4

Consider the following nonlinear stiff systems of ODEs [38].

$$\begin{cases} u_1'(t) = -1002u_1(t) + 1000u_2^2(t) \\ u_2'(t) = u_1(t) - u_2(t) - u_2^2(t) \end{cases}$$

With the initial condition $u_1(0) = 1$ and $u_2(0) = 1$. The exact solution is

$$u_1(t) = exp\,(-2t)$$
$$u_2(t) = exp\,(-t).$$

In this numerical example, two scheme are compared and as explained the main task of the modified method tackle the singularity of the moment matrix. **Table 10** presents the maximum relative error by MLS on a set of evaluation points (with $h = 0.1$ *and* 0.02) and $\delta = 4h$ *and* $3h$. Also in **Table 11** MLS and MMLS at different number of nodes for $h = 0.004$ and $\delta = 5h$ *and* $8h$, were compared (**Tables 10–12**).

	m = 2,δ = 4 r		m = 2, δ = 3 r	
r	u_1	u_2	u_1	u_2
0.1	5×10^{-3}	4.1×10^{-4}	8.85×10^{-4}	2.2×10^{-3}
0.02	5.8×10^{-2}	6.5×10^{-5}	5.42×10^{-4}	6.52×10^{-5}

Table 10.
Maximum relative errors by MLS, example 4.

	m = 3, δ = 5 r		m = 3, δ = 8 r	
Type	u_1	u_2	u_1	u_2
MLS	1.03×10^{0}	0.98×10^{1}	1.01×10^{0}	9.2×10^{0}
MMLS	9.23×10^{-4}	9.22×10^{-4}	6.89×10^{-4}	6.96×10^{-4}

Table 11.
Maximum relative errors for h = 0.004 by MMLS and MLS, example 4.

	MLS			MMLS		
weight type	u_1	u_2	**Cputime**	u_1	u_2	**Cputime**
Guass	3.06×10^{-3}	9.92×10^{-5}	61.1706	8.5×10^{-4}	6.5×10^{-4}	0.5598
Spline	5.06×10^{-4}	5.3×10^{-4}	64.5897	1.93×10^{-2}	4.23×10^{-4}	0.6714
RBF	6.407×10^{-4}	3.02×10^{-4}	59.1790	6.9×10^{-3}	6.9×10^{-3}	0.5768

Table 12.
Maximum relative errors by MLS $t \in [0,5], h = 0.004$,, example 2.

6.5 Example 5

In this example, we consider $U(t) = \left(\frac{1}{47}\left(95\exp^{(-2t)} - 48\,exp\,(-96t)\right), \frac{1}{47}\right.$ $\left.(48\exp(-96t) - \exp(-2t))\right)$ as the exact solution and $U(0,0) = (1,1)$ as the initial conditions for the following system of ODE,

$$\begin{cases} x'(t) = -x(t) + 95y(t) \\ y'(t) = -x(t) - 97y(t) \end{cases}$$

Table 12 presents the maximum relative norm of the errors on a fine set of evaluation points (with $h = 0.004$) and $\delta = 5h$ for MLS and MMLS at different type of weight functions. As seen in this table, one major advantage of MMLS is that the computational time used by MMLS is less than MLS.

7. Conclusion

In this paper, two meshless techniques called moving least squares and modified Moving least-squares approximation are applied for solving the system of functional equations. Comparing the results obtained by these methods with the results obtained by the exact solution shows that the moving least squares methods are the reliable and accurate methods for solving a system of functional equations. Meshless methods are free from choosing the domain and this makes it suitable to study real-world problems. Also, the modified algorithm has changed the ability to select the support range radius In fact, the user can begin to solve any problem with an arbitrary radius from the domain and the proposed algorithm can correct it during execution.

Author details

Massoumeh Poura'bd Rokn Saraei* and Mashaallah Matinfar
Department of Mathematics, Science of Mathematics Faculty, University of Mazandaran, Iran

*Address all correspondence to: m.pourabd@gmail.com and m.matinfar@umz.ac.ir

References

[1] Scudo FM. Vito Volterra and theoretical ecology. Theoretical Population Biology. 1971;**2**:1-23

[2] Small RD. Population growth in a closed model. In: Mathematical Modelling: Classroom Notes in Applied Mathematics. Philadelphia: SIAM; 1989

[3] TeBeest KG. Numerical and analytical solutions of Volterra's population model. SIAM Review. 1997; **39**:484-493

[4] Wazwaz AM. Partial Differential Equations and Solitary Waves Theory. Beijing and Berlin: HEP and Springer; 2009

[5] Mckee S, Tang T, Diogo T. An Euler-type method for two-dimensional Volterra integral equations of the first kind. IMA Journal of Numerical Analysis. 2000;**20**:423-440

[6] Hanson R, Phillips J. Numerical solution of two-dimensional integral equations using linear elements. SIAM Journal on Numerical Analysis. 1978;**15**: 113-121

[7] Babolian E, Masouri Z. Direct method to solve Volterra integral equation of the first kind using operational matrix with block-pulse functions. Journal of Computational and Applied Mathematics. 2008;**220**:51-57

[8] Beltyukov BA, Kuznechichina LN. A RungKutta method for the solution of two-dimensional nonlinear Volterra integral equations. Differential Equations. 1976;**12**:1169-1173

[9] Masouri Z, Hatamzadeh-Varmazyar S, Babolian E. Numerical method for solving system of Fredholm integral equations using Chebyshev cardinal functions. Advanced Computational Techniques in Electromagnetics. 2014:1-13

[10] Jafarian A, Nia SAM, Golmankhandh AK, Baleanu D. Numerical solution of linear integral equations system using the Bernstein collocation method. Advances in Difference Equations. 2013:1-23

[11] Singh P. A note on the solution of two-dimensional Volterra integral equations by spline. Indian Journal of Mathematics. 1979;**18**:61-64

[12] Assari P, Adibi H, Dehghan M. A meshless method based on the moving least squares (MLS) approximation for the numerical solution of two-dimensional nonlinear integral equations of the second kind on non-rectangular domains. Numerical Algorithm. 2014;**67**:423-455

[13] Brunner H, Kauthen JP. The numerical solution of two-dimensional Volterra integral equations by collocation and iterated collocation. IMA Journal of Numerical Analysis. 1989;**9**:47-59

[14] Wazwaz A. Linear and Nonlinear Integral Equations Methods and Applications. 2011

[15] Dehghan M, Abbaszadeha M, Mohebbib A. The numerical solution of the two-dimensional sinh-Gordon equation via three meshless methods. Engineering Analysis with Boundary Elements. 2015;**51**:220-235

[16] Mirzaei D, Schaback R, Dehghan M. On generalized moving least squares and diffuse derivatives. IMA Journal of Numerical Analysis. 2012;**32**:983-1000

[17] Salehi R, Dehghan M. A generalized moving least square reproducing kernel method. Journal of Computational and Applied Mathematics. 2013;**249**:120-132

[18] Saadatmandi A, Dehghan M. A collocation method for solving Abels

integral equations of first and second kinds. Zeitschrift für Naturforschung. 2008;**63a**(10):752-756

[19] Dehghan M, Mirzaei D. Numerical solution to the unsteady two-dimensional Schrodinger equation using meshless local boundary integral equation method. International Journal for Numerical Methods in Engineering. 2008;**76**:501-520

[20] Mukherjee YX, Mukherjee S. The boundary node method for potential problems. International Journal for Numerical Methods in Engineering. 1997;**40**:797-815

[21] Salehi R, Dehghan M. A moving least square reproducing polynomial meshless method. Applied Numerical Mathematics. 2013;**69**:34-58

[22] Matin far M, Pourabd M. Moving least square for systems of integral equations. Applied Mathematics and Computation. 2015;**270**:879-889

[23] Li S, Liu WK. Meshfree Particle Methods. Berlin: Springer-Verlag; 2004

[24] Liu GR, Gu YT. A matrix triangularization algorithm for the polynomial point interpolation method. Computer Methods in Applied Mechanics and Engineering. 2003;**192**: 2269-2295

[25] Belinha J. Meshless Methods in Biomechanics. Springer; 2014

[26] Chen S. Building interpolating and approximating implicit surfaces using moving least squares [Phd thesis] EECS-2007-14. Berkeley: EECS Department, University of California. 2007

[27] Matin far M, Pourabd M. Modified moving least squares method for two-dimensional linear and nonlinear systems of integral equations. Computational and Applied Mathematics. 2018;**37**:5857-5875

[28] Joldesa GR, Chowdhurya HA, Witteka A, Doylea B, Miller K. Modified moving least squares with polynomial bases for scattered data approximation. Applied Mathematics and Computation. 2015;**266**:893-902

[29] Lancaster P, Salkauskas K. Surfaces generated by moving least squares methods. Mathematics of Computation. 1981;**37**:141-158

[30] Wendland H. Local polynomial reproduction, and moving least squares approximation. IMA Journal of Numerical Analysis. 2001;**21**:285-300

[31] Zuppa C. Error estimates for moving least square approximations. Bulletin of the Brazilian Mathematical Society, New Series. 2001;**34**:231249

[32] Liu GR. Mesh Free Methods: Moving beyond the Finite Element Method. Boca Raton: CRC Press; 2003

[33] Zuppa C. Error estimates for moving least squareapproximations. Bulletin of the Brazilian Mathematical Society, New Series. 2003;**34**(2):231-249

[34] Zuppa C. Good quality point sets error estimates for moving least square approximations. Applied Numerical Mathematics. 2003;**47**:575-585

[35] Mirzaei D, Dehghan M. A meshless based method for solution of integral equations. Applied Numerical Mathematics. 2010;**60**:245-262

[36] McLain D. Two dimensional interpolation from random data. Computer Journal. 1976;**19**:178181

[37] Pourabd M. Meshless method based moving least squares for solving systems of integral equations with investigating the computational complexity of algorithms [Phd thesis], IRANDOC-2408208. IRAN: Department of mathematic, University of Mazandaran. 2017

[38] Biazar J, Asadi MA, Salehi F. Rational Homotopy perturbation method for solving stiff systems of ordinary differential equations. Applied Mathematical Modelling. 2015;**39**: 12911299