Available online at www.sciencedirect.com
SCIENCE DIRECT®
ELSEVIER Computers and Mathematics with Applications 51 (2006) 1297–1310
An International Journal
computers & mathematics with applications
www.elsevier.com/locate/camwa

Numerical Comparison of Least Square-Based Finite-Difference (LSFD) and Radial Basis Function-Based Finite-Difference (RBFFD) Methods

C. SHU*
Department of Mechanical Engineering
National University of Singapore
Singapore 119260
mpeshuc@nus.edu.sg

H. DING
Department of Mechanical Engineering
National University of Singapore
Singapore 119260

N. ZHAO
Department of Aerodynamics
Nanjing University of Aeronautics and Astronautics
29 Yudao Jie, Nanjing 210016, P.R. China

Abstract—In this paper, two mesh-free methods, i.e., least square-based finite difference (LSFD) and radial basis function-based finite difference (RBFFD), are compared numerically in terms of their accuracy and efficiency. These two mesh-free methods are based on different approximation schemes, that is, the least square approximation and radial basis function (RBF) approximation. The two mesh-free methods exhibit very different behaviors in many ways. In this study, we examine the performance of the two methods by applying them to two example problems: Poisson equation and two-dimensional incompressible viscous lid-driven cavity flow, and some interesting findings are observed.

1. INTRODUCTION

In the past decade, the so-called *mesh-free* methods have become one of the hottest research areas in computational mechanics. The term *mesh-free* or *meshless* is used to describe the special ways of constructing the approximation or interpolation scheme for the spatial discretization. That is, the function and its derivatives at one central node are approximated entirely from the information of a set of scattered nodes within its local support, and there is no prespecified connectivity or relationships among the nodes.

The interest in these schemes is instinctively spurred by the perceived difficulty of generating appropriate meshes in the standard schemes for problems characterized by complex geometry or

*Author to whom all correspondence should be addressed.

0898-1221/06/$ - see front matter Typeset by AMS-TEX
doi:10.1016/j.camwa.2006.04.015

complex physics. Instead of mesh generation, mesh-free methods usually require node generation. From the point of view of computational efforts, node generation is considered as an easier and faster job compared to the former one. After generating a set of nodes within the analysis domain, a local cloud of neighboring points, which is called "local support", is selected for the interior node. Then, a local approximation is constructed based on the local support to approximate the function and/or its derivatives in terms of point values. Finally, a set of algebraic equations can be obtained by substituting the approximation scheme into the governing equations. This is a common procedure to discretize partial differential equations by the mesh-free method. In addition, we can also enjoy the computational ease of adding and deleting nodes from the pre-existing of nodes. This property is highly appreciated in the flow problems with large deformation or moving boundaries.

In the viewpoint of kernel interpolation/approximation techniques, the mesh-free methods to date can be grouped into two categories. One is based on the least square (LS) technique or its equivalents. This interpolation scheme has been adopted by many popular mesh-free methods [1–8]. The least-squares technique allows an optimized approximation derived from an overdetermined set of equations, and generally the resultant coefficient matrix has good properties such as positive, symmetric, and definite. Thus, by means of using more local supporting-points (more than the unknowns), the problem of singular/ill-conditioned coefficient matrix arising from the polynomial interpolation and local point distribution can be ultimately circumvented. Another is based on the radial basis functions (RBFs) interpolant. Employed as base functions for multivariate data interpolation, RBFs have already shown their capability to construct a scheme with favorable properties such as high efficiency and good quality. Madych *et al.* [9] have shown that the multiquadric-RBF interpolation scheme converges faster as the dimension increases, and converges exponentially as the density of the nodes increases. Motivated by the above attractive merits, many researchers cast their sights on the development of RBFs-based methods in the past decades and a lot of literatures are available now in this field [10–18].

It is of great interest to make a comparison between the mesh-free schemes based on these two kernel approximation techniques in terms of accuracy and efficiency. In this study, we choose two mesh-free methods, i.e., least square-based finite difference (LSFD) [8] and radial basis function-based finite difference (RBFFD) method [18], as two examples to examine their performance. It must be noted that the RBFFD method is called a local multiquadrature-differential quadrature (LMQDQ) method in [18] which directly reflects the numerical techniques implemented in the method. However, in this paper we rename it the RBFFD method due to the fact that this method is essentially equivalent to the finite-difference formulation with radial basis functions as the trial functions. Another reason is that the new name is more simple and consistent.

For the LSFD method, Ding *et al.* [8] theoretically and experimentally showed that the use of the least-square technique does not cause a deterioration of the approximation accuracy. In other words, the order of the approximation accuracy remains the same as that obtained by the multidimensional Taylor series expansion. However, the additional points in the local support (more than the unknowns) do not contribute to the accuracy improvement. On the contrary, they will decrease the accuracy due to the increment of h. For the accuracy of RBFFD method, it is very difficult to conduct theoretical analysis at this moment. This is because RBF approximation is completely different from the polynomial approximation. So, the powerful tool for accuracy analysis, i.e., Taylor series expansion (it is implicitly based on polynomial approximation), cannot be applied in the accuracy analysis of RBFFD method. Nevertheless, the accuracy of RBFFD method has been studied through numerical tests [19], and the error of the second-order derivative approximation yields $\varepsilon \sim O((h/c)^n)$, in which h is the mesh size, c the value of free shape parameter in the certain range, and n a positive constant and determined by the number of supporting points.

Despite the previous analysis, a direct comparison of the two methods may be more appreciated. In this paper, we put the two methods under the same computational conditions in terms of

governing equation, boundary condition, node distribution, and number of supporting points. Therefore, we can examine their performance on the accuracy and efficiency. The numerical examples selected are the Poisson equation and lid-driven cavity flow. Since the analytical solution of the Poisson equation is given, this example can be used to examine the numerical error arising from the spatial discretization. The case of lid-driven cavity flow is a steady flow problem, and it was used to test the performance of two methods in the numerical simulation of complex phenomena.

2. BRIEF DESCRIPTION OF TWO MESH-FREE METHODS

In this section, a brief introduction of the two mesh-free methods is provided. For the details of two methods, one can refer to [8] and [18].

2.1. Mesh-Free Least Square-Based Finite-Difference Method

The LSFD method is based on the use of a weighted least square approximation procedure together with a Taylor series expansion of the unknown function. In theory, the multidimensional finite-difference method derived from multidimensional Taylor series expansion can be seen as a natural mesh-free method since the construction of Taylor series expansion approximation does not require the help of mesh or the connection between the local supporting points. For a smooth function f, suppose that the two-dimensional Taylor series is expanded around one reference node 0 and truncated to the third-order derivative terms. Then, we can obtain a system of equations with nine derivatives as unknowns. It can be written in the matrix form by

$$\mathbf{C}_{ij}\mathbf{d}_j = \mathbf{f}_i. \tag{1}$$

The details of the matrix $\mathbf{C}$, vector $\mathbf{d}$, and $\mathbf{f}$ yield

$$\mathbf{C} = \begin{pmatrix} \Delta x_1 & \Delta y_1 & \cdots & \frac{1}{2}\Delta x_1(\Delta y_1)^2 \\ \Delta x_2 & \Delta y_2 & \cdots & \frac{1}{2}\Delta x_2(\Delta y_2)^2 \\ \vdots & \vdots & \vdots & \vdots \\ \Delta x_9 & \Delta y_9 & \cdots & \frac{1}{2}\Delta x_9(\Delta y_9)^2 \end{pmatrix}, \tag{2a}$$

$$\begin{aligned} \mathbf{d}^\top = \Bigg[&\left(\frac{\partial f}{\partial x}\right)_0, \left(\frac{\partial f}{\partial y}\right)_0, \left(\frac{\partial^2 f}{\partial x^2}\right)_0, \left(\frac{\partial^2 f}{\partial y^2}\right)_0, \left(\frac{\partial^2 f}{\partial x\,\partial y}\right)_0, \\ &\left(\frac{\partial^3 f}{\partial x^3}\right)_0, \left(\frac{\partial^3 f}{\partial y^3}\right)_0, \left(\frac{\partial^3 f}{\partial x^2\,\partial y}\right)_0, \left(\frac{\partial^3 f}{\partial x\,\partial y^2}\right)_0 \Bigg], \end{aligned} \tag{2b}$$

$$\mathbf{f}^\top = [\, f_1 - f_0 \quad f_2 - f_0 \quad \cdots \quad f_9 - f_0 \,]. \tag{2c}$$

Then, if the coefficient matrix $\mathbf{C}$ is well conditioned, the derivative vector $\mathbf{d}$ can be uniquely determined by the node value $\mathbf{f}$. However, the method suffers from an ill-conditioned coefficient matrix due to the distribution of local supporting points, especially when the Taylor series expansion is truncated to a high-order derivative. This greatly hampers the applicability of the method to the practical problems in engineering and science.

One approach to make the coefficient matrix invertible is to introduce the least square technique. It uses more supporting/collocation points to make the modified matrix possess good

properties like symmetric, positive, and definite. Here, we skip the detailed mathematical procedure, and only give the final equation for the derivative approximation. The details can be found in [8]. By using the least square technique, equation (1) is modified as

$$\mathbf{C}^{\top}\mathbf{C}\mathbf{d} = \mathbf{C}^{\top}\mathbf{f}. \tag{3}$$

Then, the derivatives can be obtained by period

$$\mathbf{d} = \left(\mathbf{C}^{\top}\mathbf{C}\right)^{-1}\mathbf{C}^{\top}\mathbf{f}. \tag{4}$$

It is noted that now the matrix $\mathbf{C}$ is $\mathbf{C_{m\times n}}$ and $m > n$. So it is not a square matrix anymore. However, $\mathbf{C}^{\top}\mathbf{C}$ has much better condition than the previous $\mathbf{C_{n\times n}}$. In equation (4), the component of the matrix $(\mathbf{C}^{\top}\mathbf{C})^{-1}\mathbf{C}^{\top}$ contains the information of the computed derivative coefficients. Once they are computed, they are stored and used to discretize the derivative in the governing equations.

The derivative coefficients computed by using the standard least square technique (as shown in equation (3)) may not generate an optimal distribution of approximation errors. One would normally prefer the approximation error to be small in the crucial central region around the reference node, where the derivatives are evaluated, and be willing to tolerate higher errors for points further away, since the latter is expected to have smaller influence on the desired derivatives. The redistribution of errors can be achieved by introducing a distance-related weighting function that assigns larger weights to the points nearer to the reference node. That is the so-called weighted least square optimization. With the use of weighted least square technique, the derivative approximation (3) can be further improved as

$$\mathbf{d} = \left(\mathbf{C}^{\top}\mathbf{H}\mathbf{C}\right)^{-1}\mathbf{C}^{\top}\mathbf{H}\mathbf{f}, \tag{5}$$

where $\mathbf{H}$ is a diagonal matrix such as

$$\mathbf{H} = \begin{pmatrix} w_1 & & & \\ & w_2 & & \\ & & \ddots & \\ & & & w_m \end{pmatrix}. \tag{6}$$

Here w is distance-related weighting function, and m is the number of supporting points for the specified reference node. In this study, the following function is selected as a distance-related weighting function [8],

$$w_i = \sqrt{\frac{4}{\pi}}\left(1 - \bar{r}_i^2\right)^4, \tag{7}$$

where $\bar{r}_i = \sqrt{(x_i - x_0)^2 + (y_i - y_0)^2}/d_o$, the subscript i denotes the i^{th} supporting point, d_o the radius of local support, and $0 \leq \bar{r}_i \leq 1$. It should be noted that if only the first- and second-order derivative approximations are required; only the first five entries of $\mathbf{d}$ need to be considered and stored during computation. However, the inclusion of high-order terms can increase the accuracy of the method. It is also interesting to point out that at each node the coefficient matrix remains unchanged for a fixed set of supporting points, so that its inverted matrix needs to be calculated only once.

2.2. Radial Basis Function-Based Finite-Difference Method

The radial basis function-based finite-difference method can be considered as an extension of the conventional differential quadrature (DQ) method. The conventional DQ method is a numerical discretization technique for the derivatives of the smooth functions. The essence of the

DQ method is that the partial derivative of an unknown function with respect to an independent variable can be approximated by a weighted linear sum of function values at all discrete points within its support. Suppose that a function $f(x)$ is sufficiently smooth. Then its m^{th}-order derivative with respect to x at a point x_i can be approximated by DQ as

$$\left.\frac{\partial^m f}{\partial x^m}\right|_{x=x_i} = \sum_{j=1}^{N} w_{ij}^{(m)} f(x_j), \qquad i = 1, 2, \ldots, N, \tag{8}$$

where x_j are the discrete points in the domain, N the number of supporting points, $f(x_j)$ and $w_{ij}^{(m)}$ are the function values at these points and the related weighting coefficients. Obviously, the key procedure in the DQ method is the determination of the weighting coefficients $w_{ij}^{(m)}$. Shu and Richards [20] indicated that the weighting coefficients can be easily computed under the analysis of a linear vector space. As compared with the conventional DQ method, the RBFFD method takes the MQ RBF instead of high-order polynomials as the basis functions. The MQ RBF is defined as follows:

$$\phi_j(x, y) = \sqrt{(x - x_j)^2 + (y - y_j)^2 + c^2}, \tag{9}$$

where c is a free shape parameter and defined by the practitioner. This simple replacement combines the merits of MQ RBF and the DQ scheme together, such as simplicity of the scheme, mesh-free property, and high accuracy, etc. The coefficient matrix $\mathbf{G}$ in the RBFFD method can be obtained by substituting the MQ RBF into equation (8). For the derivative approximation at the reference node $\mathbf{x}_r$, it yields

$$\mathbf{g}_x = \mathbf{G}\mathbf{W}, \tag{10}$$

where

$$\mathbf{g}_x = \left[\frac{\partial^m \phi_1}{\partial x^m}(x_r, y_r), \frac{\partial^m \phi_2}{\partial x^m}(x_r, y_r), \frac{\partial^m \phi_3}{\partial x^m}(x_r, y_r), \ldots, \frac{\partial^m \phi_N}{\partial x^m}(x_r, y_r)\right]^{\top},$$

$$\mathbf{G} = \begin{bmatrix} \phi_1(x_1, y_1) & \phi_1(x_2, y_2) & \cdots & \phi_1(x_{N_r}, y_{N_r}) \\ \phi_2(x_1, y_1) & \phi_2(x_2, y_2) & \cdots & \phi_2(x_{N_r}, y_{N_r}) \\ \vdots & \vdots & \vdots & \vdots \\ \phi_{N_r}(x_1, y_1) & \phi_{N_r}(x_2, y_2) & \cdots & \phi_{N_r}(x_{N_r}, y_{N_r}) \end{bmatrix},$$

$$\mathbf{w} = \left[w_{r,1}^{(m)}, w_{r,2}^{(m)}, w_{r,3}^{(m)}, \ldots, w_{r,N_r}^{(m)}\right]^{\top}.$$

In the above expressions, N_r denotes the number of supporting points in the local support for the reference node $\mathbf{x}_r$. For the practical computation, it may vary at different nodes. The subscript $1, 2, \ldots, N_r$ indicates a local point within the local support similar to that in the finite-element method.

If the local support of all the reference nodes is extended to the maximum limit, i.e., the number of supporting points N_r equals the number of nodes in the domain, then the local RBFFD method becomes a global RBFFD method. This global RBFFD method is essentially equivalent to Kansa's collocation method. As observed by Dubal *et al.* [21] and Fornberg *et al.* [22], the global RBFs-based methods suffer from the progressively more ill-conditioned coefficient matrix as the rank increases. They noted that the coefficient matrix of using about 2000 knots is extremely ill-conditioned. This problem also occurs in the global RBFFD method.

A unique solution of equation (10) can be obtained only if the collocation matrix $[\mathbf{G}]$ is nonsingular. Micchelli [23] proved that matrix $[\mathbf{G}]$ is conditionally positive definite for MQ RBFs. In general, the problem of ill-conditioned/singular coefficient matrix is not serious for the RBFFD

method since the supporting points are distinct and the number is comparatively small. Then, the weighting coefficient vector $\mathbf{w}$ can be computed by

$$\mathbf{w} = \mathbf{G}^{-1}\mathbf{g}_x. \tag{11}$$

Similar to the mesh-free LSFD method, the inverted matrix of $\mathbf{G}$ is only required to compute once if the local support is fixed. The computed coefficient will be stored to discretize the partial derivatives in the governing equations.

2.3. Comparison of LSFD and RBFFD Methods

Actually, the LSFD method shares many common features with the RBFFD method with regard to the mesh-free derivative approximation. Both of them use a fixed cloud of supporting points, and discretize the governing equations in the strong form. They even can share the same iterative solver to obtain the solution of resultant algebraic equations. However, due to the use of different kernel approximation, the first difference occurs in the way of calculating the weighting coefficients for derivative approximation. Consequently, the two methods differ in the computational aspects, for example, the contribution of the number of supporting points. Ding *et al.* [8] pointed out that the order of accuracy of the LSFD method is determined by the order of truncated Taylor series expansion. The additional supporting points do not contribute to the accuracy improvement. Instead, they are used to improve the condition of the coefficient matrix. However, this is not the case in the RBFFD method. As shown in reference [18,19], the increasing number of supporting points may directly accelerate the speed of the convergence. From the free parameter's point of view, the practitioner of the RBFFD method is required to define a not-bad value for the free shape parameter c. It is well known that the free shape parameter plays an important role in the determination of solution accuracy. An optimal shape parameter c needs no additional computation cost and can improve the accuracy of solution. On the other hand, it may be an uncertain factor in the practical applications due to the difficulties in finding a "good" shape parameter on a specified nodal distribution. For the practitioners of LSFD method, they do not have such great concern. They may have to consider the selection of weighting function to obtain an optimal error distribution. However, the effect of the weighting function on the solution accuracy in the LSFD method is much less than the shape parameter c does in the RBFFD method according to [8,18].

It is important to examine the performance of the two methods with respect to the computational efficiency and accuracy under the same conditions. Two numerical examples are designed for this purpose in the study, and they are described in the next section.

3. NUMERICAL EXPERIMENTS

In this section, numerical experiments are carried out to study the convergence properties of the two mesh-free methods by solving sample problems. The sample problems selected are the computation of two-dimensional Poisson equation and simulation of lid-driven cavity flow. The solutions of both cases are restricted in a unit square domain ($0 \leq x \leq 1$, $0 \leq y \leq 1$).

3.1. Poisson Equation

The two-dimensional Poisson equation is

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f(x, y). \tag{12}$$

For simplicity, we first specify the exact solution of equation (12), and the Dirichlet conditions are imposed on the four boundaries, i.e., $u_{\text{boundary}} = u_{\text{exact}}$. The source function $f(x, y)$ on the right side of equation (12) is determined from the given exact solution, which is also used to measure the numerical error. Relative error is taken to measure the accuracy of numerical results, which is defined as

$$\|\text{Error}\| = \frac{\sqrt{\sum_{i=1}^{N} (u_{i,\text{num}} - u_{i,\text{exact}})^2}}{\sqrt{\sum_{i=1}^{N} (u_{i,\text{exact}})^2}}. \tag{13}$$

In this study, the analytical solution of equation (12) is selected from the work of Lyche *et al.* [24], and has the following form:

$$u_2 = \left(1 - \frac{x}{2}\right)^6 \left(1 - \frac{y}{2}\right)^6 + 1000(1-x)^3 x^3 (1-y)^3 y^3 + y^6 \left(1 - \frac{x}{2}\right)^6 + x^6 \left(1 - \frac{y}{2}\right)^6. \tag{14}$$

The initial condition for the unknown function at the interior nodes is set to zero. After the spatial discretization by RBFFD or LSFD methods, the successive overrelaxation (SOR) iterative method was employed to solve the resultant algebraic equations.

To study the effect of the number of supporting points on the numerical error of the solution, we carried out numerical experiments with two numbers of supporting points, i.e., 12 and 14. Due to its great effect on the solution accuracy, the value of the shape parameter was, respectively, chosen as 0.05, 0.1, and 0.2 in the tests of the RBFFD method. The numerical experiments were performed on three uniformly distributed grids, i.e., 41×41, 57×57, and 81×81. The numerical solutions are presented in Figures 1 and 2 in terms of the error in the log-log scale. It can be obviously seen that with the increment of the number of supporting point from 12 to 14, the convergence rate of the LSFD method does not change but that of the RBFFD method is improved, which is in line with the analysis and findings in [8] and [19]. From Figures 1 and 2, it can also be found that the shape parameter c has a great influence on the accuracy of the RBFFD scheme and the optimization of c can make the solution much more accurate. The

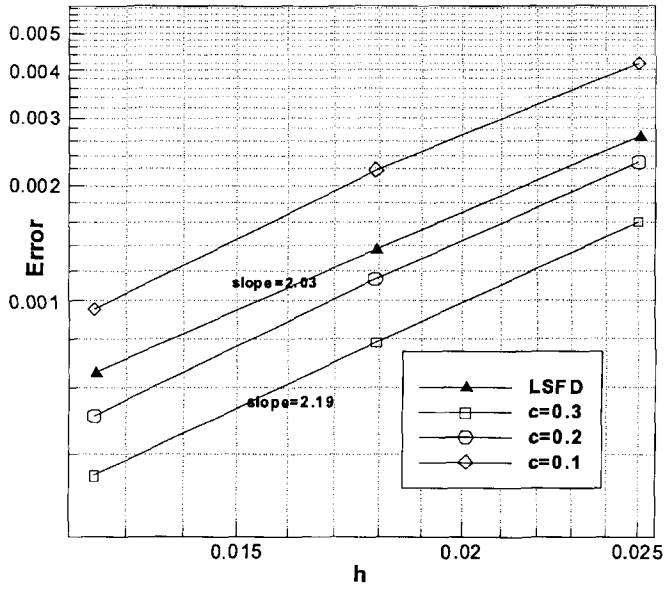


Figure 1. Convergence rate of relative error versus mesh size for 12 supporting points.

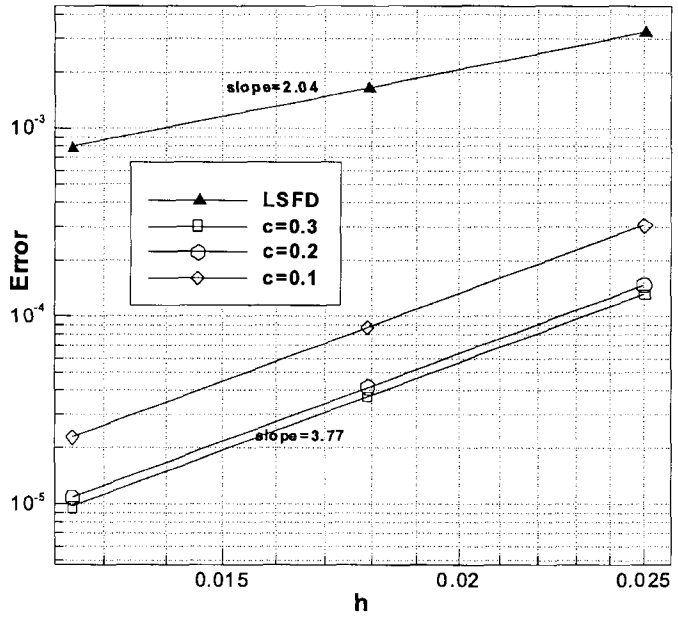


Figure 2. Convergence rate of relative error versus mesh size for 14 supporting points.

numerical solutions are also quantitatively listed in Table 1 in terms of the error. It can be observed that the accuracy of the LSFD method is actually impaired by the enlargement of local support. Ding *et al.* [8] pointed out that the accuracy of the LSFD method is proportional to h^n, where h is the radius of the local support and n the order of accuracy. In these experiments, the enlargement of local support yields a larger h but not the order of accuracy as shown in Figures 1 and 2. As a result, the solution with larger local support is less accurate. In comparison to the LSFD method, the RBFFD method shows a completely different behavior. It can be seen in Table 1 that the solution with 14 supporting points is generally more accurate than that with 12 supporting points. It is also interesting to make a comparison of solution-accuracy between the RBFFD and LSFD methods. From Table 1, it can be observed that the LSFD method is generally less accurate than the RBFFD method, especially for the cases in which more supporting points are used to improve the condition number of the coefficient matrix. On the other hand, the RBFFD method achieves good accuracy at the price of efficiency. The iteration number of using the SOR approach to obtain the converged solution is listed in Table 2. It can be observed that the RBFFD method requires many more iterations to reach the final solution than the LSFD method. Since the operation counts are almost the same in one cycle of iteration using either the LSFD or RBFFD method, the iteration number also indicates the required CPU time. In this sense, LSFD method is more efficient than the RBFFD method. In Table 2, it is also clearly seen

Table 1. Numerical result for the Poisson equation.

"Meshes"		41×41		57×57		81×81	
Number of Supporting Points		12	14	12	14	12	14
RBFFD	$c = 0.1$	4.17×10^{-3}	3.04×10^{-4}	2.19×10^{-3}	8.73×10^{-5}	9.55×10^{-4}	2.26×10^{-5}
	$c = 0.2$	2.30×10^{-3}	1.47×10^{-4}	1.14×10^{-3}	4.19×10^{-5}	5.06×10^{-4}	1.08×10^{-5}
	$c = 0.3$	1.61×10^{-3}	1.32×10^{-4}	7.81×10^{-4}	3.73×10^{-5}	3.53×10^{-4}	9.70×10^{-6}
LSFD		2.68×10^{-3}	3.27×10^{-3}	1.37×10^{-3}	1.63×10^{-3}	6.55×10^{-4}	7.92×10^{-4}

Table 2. Iteration number for the Poisson equation.

"Meshes"		41 × 41		57 × 57		81 × 81	
Number of Supporting Points		12	14	12	14	12	14
RBFFD	$c = 0.1$	4775	4977	9302	9680	18921	19676
	$c = 0.2$	4745	4935	9272	9639	18892	19635
	$c = 0.3$	4734	4921	9262	9625	18883	19622
LSFD		2550	2001	4998	3917	10202	8001

that the use of larger local support can reduce the iteration number in the LSFD method while increasing the iteration number in the RBFFD method.

3.2. Lid-Driven Cavity Flow

The two-dimensional lid-driven cavity flow is a simple viscous incompressible flow. This flow problem is often employed as a test problem since it has many features suitable for examining the performance of a numerical scheme. In general, it is characterized by primary and secondary vortices, wall boundary layers, flow separation, and reattachment. This problem has been studied by many researchers. Among them, Ghia *et al.* [25] reported a set of accurate reference solutions to the steady flow over a range of Reynolds numbers. In this work, they are considered as benchmark data.

The flow in the cavity are governed by the following nondimensional equations in terms of stream function ψ and vorticity ω:

$$\frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} = \omega, \tag{15}$$

$$\frac{\partial \omega}{\partial t} + u\frac{\partial \omega}{\partial x} + v\frac{\partial \omega}{\partial y} = \frac{1}{\mathrm{Re}}\left(\frac{\partial^2 \omega}{\partial x^2} + \frac{\partial^2 \omega}{\partial y^2}\right), \tag{16}$$

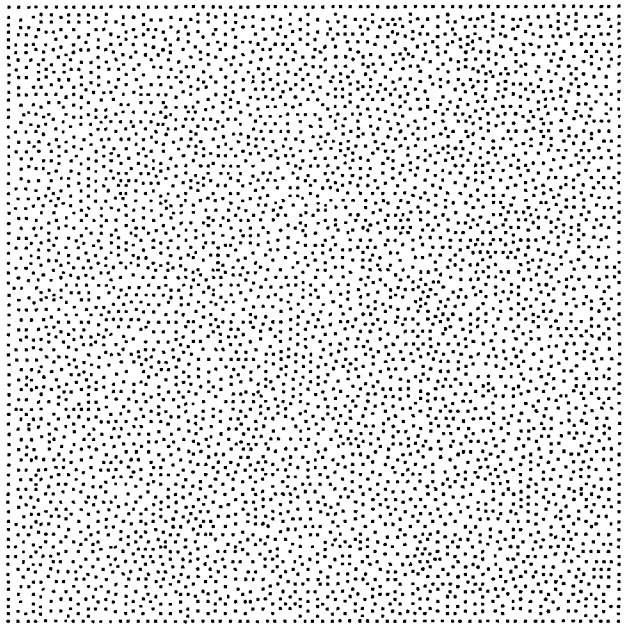
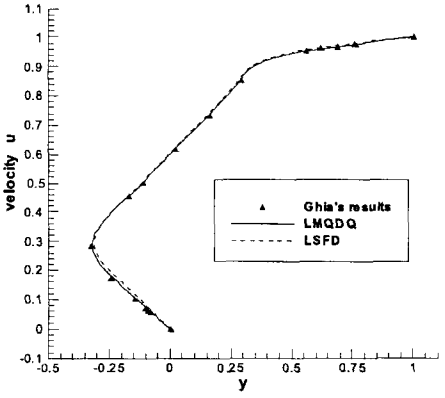
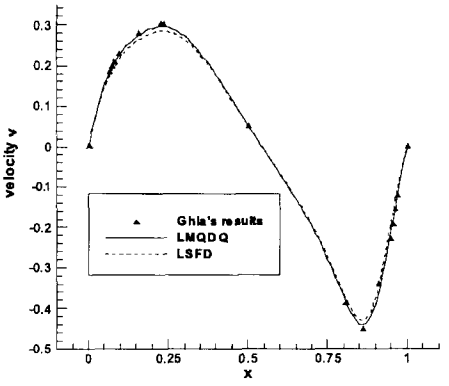


Figure 3. Randomly-distributed nodes in a square domain.

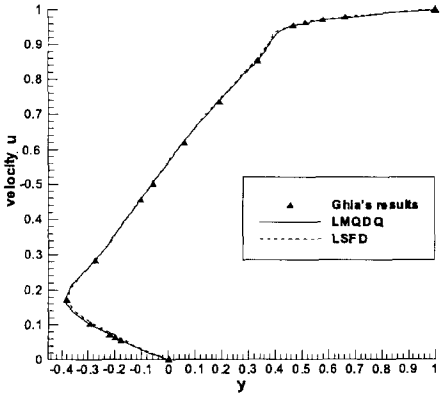


(a) u along the vertical centerline.

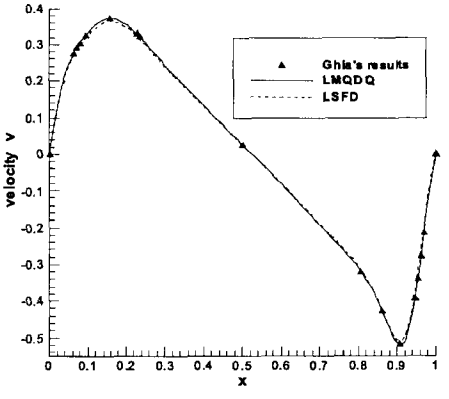


(b) v along the horizontal centerline.

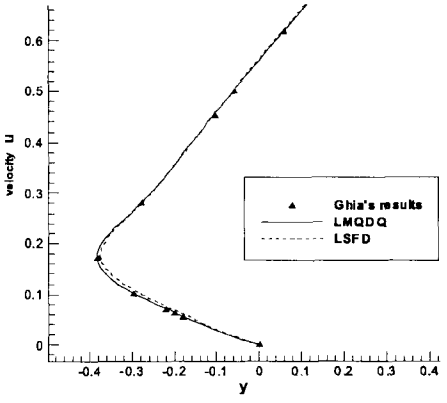
Figure 4. Velocity profiles at Re = 400 on a grid of 41 × 41.



(a) u along the vertical centerline.



(b) v along the horizontal centerline.



(c) Zooming-in view of (a).

Figure 5. Velocity profiles at Re = 1000 on a grid of 81 × 81.

where u and v denote the components of velocity in the x and y direction, which can be calculated from the stream function by $u = \frac{\partial \psi}{\partial y}$ and $v = -\frac{\partial \psi}{\partial x}$. $\mathrm{Re} = UL/v$ is the Reynolds number.

More specifically, the lid-cavity flows at Reynolds number Re = 400 and 1000 are considered as the test cases in this study. The numerical simulations are performed on the uniform grids of 41 × 41 and 81 × 81, as well as on a random node distribution (including 4786 nodes) which is shown in Figure 3. For the spatial discretization, the number of supporting point is fixed to 13.

In the RBFFD method, the value of the shape parameter is selected as 0.2 for the uniformly distributed nodes and 0.03 for the randomly distributed nodes. The reason for selection of a comparatively small value for the randomly distributed nodes lies on the fact that randomly distributed nodes abound with variant configurations of local nodal distribution, which narrows the range of applicable shape parameter. Thus, we prefer to choose a smaller but "safer" shape parameter to formulate the spatial discretization. Successive overrelaxation iterative method is employed to solve the resultant algebraic equations. The convergence criterion is set as

$$\max_{i\in\Omega}\left(\left|\omega_i^{n+1}-\omega_i^n\right|\right)<0.001.$$

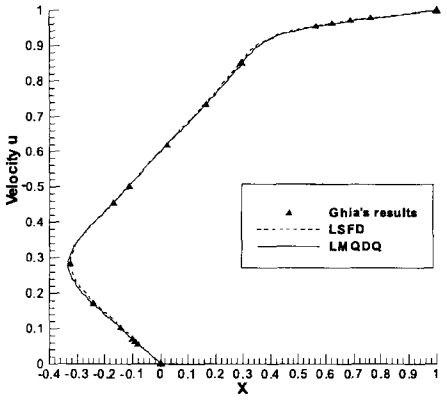
The numerical results on the uniformly distributed nodes are shown in Figures 4 and 5 in terms of the velocity profiles of the u component along the vertical centerline and the v component along the horizontal centerline. It can be clearly seen that the solutions of both methods have achieved a good agreement with Ghia's results [25] at Re = 400 and 1000. However, the result produced by the RBFFD method is comparatively more accurate since it is closer to Ghia's data [25], especially in the regions where large velocity gradient is encountered. The zooming-in view of u-velocity along the vertical centerline at Re = 1000 (Figure 5c) gives a good indication for this finding. With regard to the efficiency, the iteration number for both methods is listed in Table 3. It can be observed that the LSFD method requires only half of the iterations used by the RBFFD method. In other words, the LSFD method can obtain the solution only with half of the CPU time required by the RBFFD method though the solution may be a little bit less accurate. The computations on the randomly distributed nodes are also consistent with this observation, even working with a small shape parameter. The iteration numbers required for the computations on the randomly distributed nodes are listed in Table 4. It can be seen that the LSFD method converges faster than the RBFFD method at Re = 400. However, by observing the solution comparison in terms of u-velocity along the vertical centerline and v-velocity along the horizontal centerline in Figure 6, we can clearly see that the RBFFD solution is closer to the benchmark data than LSFD solution though both solutions generally agree very well with Ghia's results [25]. Moreover, the LSFD method cannot obtain a converged solution for the case of Re = 1000 on the same randomly distributed nodes, while the RBFFD method is still able to achieve quite a good solution as shown in Figure 7. This fact indicates that the RBFFD method not only achieves better accuracy with regard to the spatial discretization, but also has better capability in capturing the physical phenomenon in the flow region than the LSFD method.

Table 3. Iteration number for the lid-driven cavity flow on the uniformly distributed nodes.

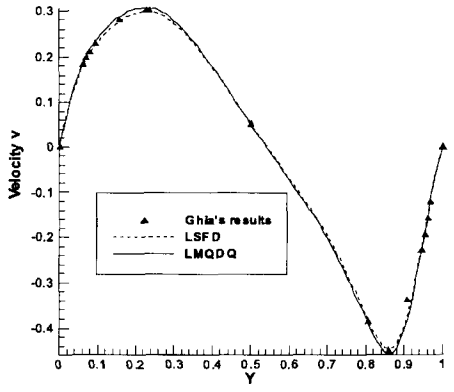
"Meshes"		41 × 41	81 × 81
Reynolds Number		400	1000
Iteration Number	RBFFD ($c = 0.2$)	4.3×10^5	2.7×10^6
	LSFD	2.1×10^5	1.2×10^6

Table 4. Iteration number for the lid-driven cavity flow on the randomly distributed nodes.

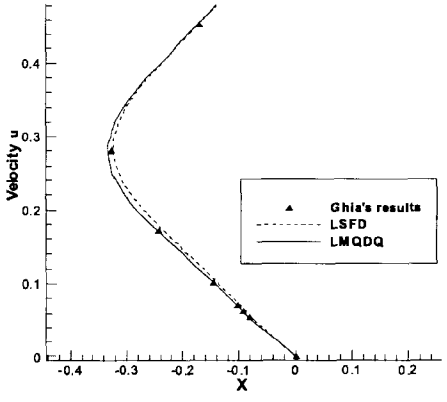
Number of Nodes		4786	4786
Reynolds Number		400	1000
Iteration Number	RBFFD ($c = 0.2$)	1.6×10^6	8.7×10^6
	LSFD	5.7×10^5	No convergence



(a) u along the vertical centerline.

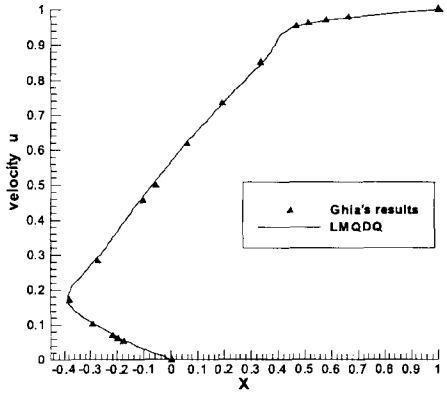


(b) v along the horizontal centerline.

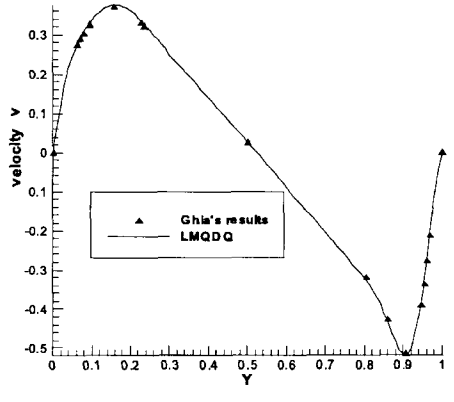


(c) Zooming-in view of (a).

Figure 6. Velocity profiles at Re = 400 on the random nodes.



(a)



(b)

Figure 7. Velocity profiles at Re = 1000 on the random nodes.

4. CONCLUSIONS

In this paper, two mesh-free methods based on the different kernel approximations, i.e., LSFD and RBFFD methods, have been numerically examined and compared in terms of their accuracy

and efficiency. The numerical experiments indicate that under the same conditions like number of supporting points, node distribution, and iterative solver, the RBFFD method can generally achieve a more accurate solution while the LSFD method is more efficient in terms of iteration number. From the viewpoint of practical applications, we suggest that the additional supporting points used to improve the condition number of the matrix in the LSFD method should be as small as possible so as to exploit the best accuracy. From the viewpoint of safety and convenience, the range of "good" shape parameter in the RBFFD method is more preferred than the best one.

REFERENCES

1. T. Belytschko, Y.Y. Lu and L. Gu, Element-free Galerkin methods, *International Journal for Numerical Methods in Engineering* **37**, 229–256, (1994).
2. W. Liu, S. Jun and Y. Zhang, Reproducing kernel particle methods, *International Journal for Numerical Methods in Fluids* **20**, 1081–1106, (1995).
3. I. Babuska and J. Melenk, The partition of unity method, *International Journal for Numerical Methods in Engineering* **40**, 727–758, (1997).
4. C.A. Duarte and J.T. Oden, Hp clouds—A meshless method to solve boundary-value problems, TICAM Report 95-05.
5. E. Oñate, S. Idelsohn, O.C. Zienkiewicz and R.L. Taylor, A finite point method in computational mechanics. Application to convective transport and fluid flow, *International Journal for Numerical Methods in Engineering* **39**, 3839–3866, (1996).
6. S.N. Atluri and T. Zhu, New meshless local Petrov-Galerkin (MLPG) approach in computational mechanics, *Computational Mechanics* **22** (2), 117–127, (1998).
7. T. Liszka, An interpolation method for an irregular net of nodes, *International Journal for Numerical Methods in Engineering* **20**, 1599–1612, (1984).
8. H. Ding, C. Shu and K.S. Yeo, Development of least square-based two-dimensional finite difference schemes and their application to simulate natural convection in a cavity, *Computer & Fluids* **33**, 137–154, (2004).
9. W.R. Madych and S.A. Nelson, Multivariate interpolation and conditionally positive definite functions, II, *Math. Comput.* **54**, 211–230, (1990).
10. E.J. Kansa, Multiquadrics—A scattered data approximation scheme with applications to computational fluid-dynamics—I. Surface approximations and partial derivative estimates, *Computers Math. Applic.* **19** (6–8), 127–145, (1990).
11. E.J. Kansa, Multiquadrics—A scattered data approximation scheme with applications to computational fluid-dynamics—II. Solutions to parabolic, hyperbolic, and elliptic partial differential equations, *Computers Math. Applic.* **19** (6–8), 147–161, (1990).
12. G.E. Fasshauer, Solving partial differential equations by collocation with radial basis functions, In *Surface Fitting and Multi-Resolution Methods*, (Edited by A.L. Mehaute, C. Rabut and L.L. Schumaker), pp. 131–138, (1997).
13. A.H.-D. Cheng, M.A. Golberg, E.J. Kansa and G. Zammito, Exponential convergence and H-c multiquadric collocation method for partial differential equations, *Numerical Methods for Partial Differential Equations* **19**, 571–594, (2003).
14. W. Chen, New RBF collocation schemes and kernel RBFs with applications, *Lecture Notes in Computational Science and Engineering* **26**, 75–86, (2002).
15. R. Schaback and C. Franke, Convergence order estimates of meshless collocation methods using radial basis functions, *Advances in Computational Mathematics* **8** (4), 381–399, (1998).
16. T.A. Driscoll and B. Fornberg, Interpolation in the limit of increasingly flat radial basis functions, *Computers Math. Applic.* **43**, 413–422, (2002).
17. Y.C. Hon and Z.M. Wu, A quasi-interpolation method for solving stiff ordinary differential equations, *International Journal for Numerical Methods in Engineering* **48**, 1187–1197, (2000).
18. C. Shu, H. Ding and K.S. Yeo, Local radial basis function-based differential quadrature method and its application to solve two-dimensional incompressible Navier-Stokes equations, *Comput. Methods Appl. Mech. Engrg.* **192**, 941–954, (2003).
19. H. Ding and C. Shu, Empirical error estimates of local multiquadric-based differential quadrature (LMQDQ) method through solution of Poisson equation, *International Journal for Numerical Methods in Engineering* (to appear).
20. C. Shu and B.E. Richards, Application of generalized differential quadrature to solve two-dimensional incompressible Navier-Stokes equations, *Int. J. Numer. Methods Fluids* **15**, 791–798, (1992).
21. M.R. Dubal, S.R. Oliveira and R.A. Matzner, *Approaches to Numerical Relativity*, Cambridge University Press, Cambridge, UK, (1993).
22. B. Fornberg, T.A. Driscoll, G. Wright and R. Charles, Observations on the behavior of radial basis function approximations near boundaries, *Computers Math. Applic.* **43**, 473–490, (2002).
23. C.A. Micchelli, Interpolation of scattered data: Distance matrices and conditionally positive definite functions, *Constr. Approx.* **2**, 11–22, (1986).

24. T. Lyche and K. Morken, Knot removal for parametric B-spline curves and surfaces, *Comput. Aided Geom. Design* **4**, 217–230, (1987).
25. U. Ghia, K.N. Ghia and C.T. Shin, High-Re solutions for incompressible flow using the Navier-Stokes equations and a multi-grid method, *Journal of Computational Physics* **48**, 387–411, (1982).