3. Radial Basis Function-based Differential Quadrature (RBF-DQ) Method

3.1 Introduction

In this chapter, we present another mesh-free method, which combines the derivative approximation by the differential quadrature (DQ) method, and the function approximation by the radial basis functions (RBFs). As a result, the method can be used to directly approximate the derivatives of dependent variables on a scattered set of nodes.

Radial basis functions (RBFs) have been under intensive research as a technique for multivariate data and function interpolation in the past decades, especially in multi-dimensional applications. Their performance demonstrates that RBFs constitute a powerful framework for interpolating or approximating data on non-uniform grids. RBFs are attractive for pre-wavelet construction due to their exceptional rates of convergence and infinite differentiability. Since RBFs have excellent performance for function approximation, many researchers turn to explore their ability for solving PDEs. The first trial of such exploration was made by Kansa (1990). As shown by Kansa (1990), using RBFs as a meshless collocation method to solve PDEs possesses the following advantages: (1) first of all, it is a truly mesh-free method, and is independent of spatial dimension in the sense that the convergence order is of $O(h^{d+1})$ where h is the density of the collocation points and d is the spatial dimension; (2) furthermore, in the context of

scattered data interpolation, it is known that some RBFs have spectral convergence. In other words, as the spatial dimension of the problem increases, the convergence order also increases, and hence, much fewer scattered collocation points will be needed to maintain the same accuracy as compared with conventional finite difference, finite element and finite volume methods. This shows the applicability of the RBFs for solving high-dimensional problems. It should be indicated that although some excellent results were obtained, all previous works related to the application of RBFs for the numerical solution of PDEs are actually based on the function approximation instead of derivative approximation. In other words, these works directly substitute the expression of function approximation by RBFs into a PDE, and then change the dependent variables into the coefficients of function approximation. The process is very complicated, especially for non-linear problems. For the nonlinear case, some special techniques such as numerical continuation and bifurcation approach have to be used to solve the resultant nonlinear equations. Since the techniques are very complicated, it is not easy to apply them for solving practical problems such as fluid dynamics, which usually require a large number of mesh points for accurate solution.

Differential quadrature (DQ) method is a global approach for derivative approximation. It can obtain very accurate numerical results by using a considerably small number of grid points. The advantages of the DQ approximation and RBFs can be combined to provide an efficient discretization method, which is a derivative approximation approach and is mesh-free. In our method, the RBFs are taken as the test functions in the DQ approximation to compute the weighting coefficients. Once the weighting coefficients are

computed, the solution process for a PDE is exactly the same as the conventional DQ method and finite difference schemes. Moreover, the method can be consistently well applied to linear and nonlinear problems.

3.2 Radial Basis Functions (RBFs) and Function Approximation

A radial basis function, denoted by $\varphi(\left\|\mathbf{x}-\mathbf{x}_j\right\|_2)$, is a continuous spline which depends on the separation distances of a subset of scattered points $\mathbf{x} \in \Omega \subset \Re^d$, d =1, 2, or 3 denotes the spatial dimension. The "radial" is named due to RBFs' spherical symmetry about the centre point $\mathbf{x}_j$. The distances are usually taken to be the Euclidean metric. There are many RBFs (expression of φ) available. The most commonly used RBFs are

$$\text{Multiquadrics (MQ): } \varphi(r) = \sqrt{r^2 + c^2} \tag{3.1a}$$

$$\text{Thin-plate splines (TPS): } \varphi(r) = r^2 \log(r) \tag{3.1b}$$

$$\text{Gaussians: } \varphi(r) = e^{-cr^2} \tag{3.1c}$$

$$\text{Inverse multiquadrics: } \varphi(r) = \frac{1}{\sqrt{r^2 + c^2}} \tag{3.1d}$$

where $r = \left\|\mathbf{x}-\mathbf{x}_j\right\|_2$ and shape parameter c is a positive constant. Among above popular radial basis functions, the Gaussian and the inverse MQ are positive definite functions, while the TPS and the MQ are conditionally positive definite functions.

In recent years, the theory of radial basis function has undergone intensive research and enjoyed considerable success as a technique for interpolating multivariable data and functions. Simply, the RBF interpolation technique can be described as following: if the function values of a function $f(\mathbf{x})$ are known on a set of scattered points $\mathbf{x} \in \Omega \subset \Re^d$, the approximation of $f(\mathbf{x})$ can be written as a linear combination of N radial basis functions,

$$f(\mathbf{x}) \cong \sum_{j=1}^{N} \lambda_j \varphi(\left\| \mathbf{x} - \mathbf{x}_j \right\|_2) + \psi(\mathbf{x}) \tag{3.2}$$

where N is the number of centers or sometimes called *knots* $\mathbf{x}$, $\mathbf{x} = (x_1, \quad x_2, \quad ..., \quad x_d)$, d is the dimension of the problem, λ's are coefficients to be determined and φ is the radial basis function. Equation (3.2) can be written without the additional polynomial ψ. If Ψ_q^d denotes the space of d-variate polynomials of order not exceeding q, and letting the polynomials $P_1, \ldots, P_m$ be the basis of Ψ_q^d in $\Re^d$, then the polynomial $\psi(\mathbf{x})$, in equation (3.2), is usually written in the following form:

$$\psi(\mathbf{x}) = \sum_{i=1}^{m} \zeta_i P_i(\mathbf{x}) \tag{3.3}$$

where $m=(q\text{-}1+d)!/(d!(q\text{-}1)!)$. To determine the coefficients $(\lambda_1, \quad ..., \quad \lambda_N)$ and $(\zeta_1, \quad ..., \quad \zeta_m)$, extra m equations are required in addition to the N equations resulting from the collocating equation (3.2) at the N knots. This is insured by the m conditions for equation (3.2), viz

$$\sum_{j=1}^{N} \lambda_j P_i(\mathbf{x}_j) = 0 \qquad i=1, \ldots, m \tag{3.4}$$

The matrix formulation of equations (3.2) and (3.4) can be expressed as $\mathbf{A}\mathbf{x} = \mathbf{b}$ with the known function value on the scattered points as the components of vector **b**, and

$$\mathbf{A} = \begin{pmatrix} \varphi & P_m \\ P_m^T & 0 \end{pmatrix} \qquad \mathbf{x} = (\lambda, \zeta)^T \tag{3.5}$$

It has been proven that for a case when the nodes are all distinct, the matrix resulting from the above radial basis function interpolation is always nonsingular. In 1982, Franke published a review article evaluating the interpolation methods for scattered data available at that time. Among the methods tested, RBFs outperformed all the other methods regarding accuracy, stability, efficiency, memory requirement, and simplicity of implementation. Among the RBFs tested by Franke, Hardy's multiquadrics (MQ) were ranked the best in accuracy, followed by thin plate splines (TPS).

Though TPS radial basis functions have been considered as optimal functions for multivariate data interpolation, they do only converge linearly. Comparatively, the MQ functions converge exponentially and always produce a minimal semi-norm error. However, despite MQ's excellent performance, it contains a shape parameter *c*, which is given by end-user to control the surface shape of basis functions. When value of shape parameter *c* is small, the resulting interpolating surface forms a cone-like basis functions. As value of shape parameter *c* increases, the peak of the cone gradually flattens. The choice of the value of c can greatly affect the accuracy of the approximation. It was found that by increasing c, the root-mean-square error of the goodness-of-fit dropped to a minimum value and then grew rapidly thereafter. This is due to the fact that the MQ coefficient matrix becomes ill-conditioned when $c^2 >> r^2$. How to choose the optimal

shape parameter remains an open problem. No mathematical theory has been developed so far to determine such an optimal value. Similar difficulties are also encountered in choosing the shape parameter for the inverse MQ and Gaussian radial basis functions.

3.3 Differential Quadrature (DQ) Method for Derivative Approximation

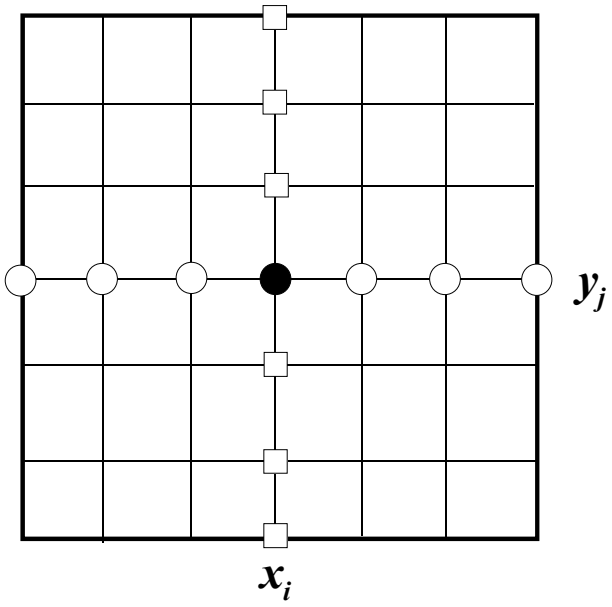


Figure 3.1 A Structured mesh for a two-dimensional problem

It is well known that any integral over a closed domain can be approximated by a linear weighted sum of all the functional values in the integral domain. Following this idea, Bellman et al. (1972) suggested that the partial derivative of a function with respect to an independent variable can be approximated by a linear weighted sum of functional values at all mesh points in that direction. As shown in Fig. 3.1, DQ approximates the derivative

of a function with respect to x at a mesh point (x_i, y_j) (represented by the symbol ●) by all the functional values along the mesh line of $y = y_j$ (represented by the symbol ○), and the derivative of the function with respect to y by all the functional values along the mesh line of $x = x_i$ (represented by the symbol □). Mathematically, the DQ approximation of the nth order derivative with respect to x, $f_x^{(n)}$, and the mth order derivative with respect to y, $f_y^{(m)}$, at (x_i, y_j) can be written as

$$f_x^{(n)}(x_i, y_j) = \sum_{k=1}^{N} w_{i,k}^{(n)} f(x_k, y_j) \tag{3.6a}$$

$$f_y^{(m)}(x_i, y_j) = \sum_{k=1}^{M} \overline{w}_{j,k}^{(m)} f(x_i, y_k) \tag{3.6b}$$

where N, M are respectively the number of mesh points in the x and y direction, $w_{i,k}^{(n)}$, $\overline{w}_{j,k}^{(m)}$ are the DQ weighting coefficients in the x and y directions. As shown by Shu (2000), $w_{i,k}^{(n)}$ depends on the approximation of the one-dimensional function $f(x, y_j)$ (x is the variable), while $\overline{w}_{j,k}^{(m)}$ depends on the approximation of the one-dimensional function $f(x_i, y)$ (y is the variable). When $f(x, y_j)$ or $f(x_i, y)$ is approximated by a high order polynomial, Shu and Richards (1992) derived a simple algebraic formulation and a recurrence relationship to compute $w_{i,k}^{(n)}$ and $\overline{w}_{j,k}^{(m)}$. When the function is approximated by a Fourier series expansion, Shu and Chew (1997) also derived simple algebraic formulations to compute the weighting coefficients of the first and second order derivatives. For simple geometry, the above DQ approach can obtain very accurate results by using a considerably small number of mesh points. However, for complex

geometry, the above scheme cannot be applied directly. The coordinate transformation technique must be introduced. To remove this drawback, we need to develop a more efficient approach.

It is noted that the basic idea of the DQ method is that any derivative can be approximated by a linear weighted sum of functional values at some mesh points. We can keep this idea but release the choice of functional values along a mesh line in the conventional DQ approximation. In other words, for a two-dimensional problem shown in Fig. 2.1, any spatial derivative is approximated by a linear weighted sum of all the functional values in the whole two-dimensional domain. In this approximation, a mesh point in the two-dimensional domain is represented by one index, k, while in the conventional DQ approximation like equation (3.6), the mesh point is represented by two indexes i, j. If the mesh is structured, it is easy to establish the relationship between i, j and k. For the example shown in Fig. 2.1, k can be written as $k = (i-1)M + j, \;\; i = 1,2,...,N; j = 1,2,...,M$. Clearly, when i is changed from 1 to N and j is changed from 1 to M, k is changed from 1 to $NM = N \times M$. The new DQ approximation for the mth order derivative with respect to x, $f_x^{(m)}$, and the nth order derivative with respect to y, $f_y^{(n)}$, at (x_k, y_k) can be written as

$$f_x^{(m)}(x_k, y_k) = \sum_{k1=1}^{NM} w_{k,k1}^{(m)} f(x_{k1}, y_{k1}) \tag{3.7a}$$

$$f_y^{(n)}(x_k, y_k) = \sum_{k1=1}^{NM} \overline{w}_{k,k1}^{(n)} f(x_{k1}, y_{k1}) \tag{3.7b}$$

In the following, we will show that the weighting coefficients in equation (3.7) can be determined by the function approximation of RBFs and the analysis of a linear vector space.

3.4 Global Radial Basis Function-based Differential Quadrature (RBF-DQ) Method

In this section, we will show in detail the global radial basis function-based differential quadrature method. The development of this method is motivated by our desire to design a numerical scheme that is as simple to implement as traditional finite difference schemes while at the same time keeping the "truly" mesh-free nature. In the following, we will show the details of global RBF-DQ method step by step.

Among above four RBFs, MQ, which was first presented by Hardy, is used extensively. Franke did a comprehensive study on various RBFs, and found that MQ generally performs better for the interpolation of 2D scattered data. Therefore, we will concentrate on MQ radial basis functions.

The MQ RBFs are used as basis functions to determine the weighting coefficients in the DQ approximation of derivatives for a two-dimensional problem. However, the method can be easily extended to the case with other RBFs as basis functions or three-dimensional problems.

Consider a two-dimensional problem. There are N knots randomly distributed in the whole computational domain. Suppose that the solution of a partial differential equation is continuous, which can be approximated by MQ RBFs, and only a constant is included in the polynomial term $\psi(\mathbf{x})$. Then, the function in the domain can be approximated by MQ RBFs as

$$f(x,y)=\sum_{j=1}^{N}\lambda_j\sqrt{(x-x_j)^2+(y-y_j)^2+c_j^2}+\lambda_{N+1} \tag{3.8}$$

To make the problem be well-posed, one more equation is required. From equation (3.4), we have

$$\sum_{j=1}^{N}\lambda_j=0 \quad \Rightarrow \quad \lambda_i=-\sum_{j=1,j\neq i}^{N}\lambda_j \tag{3.9}$$

Substituting equation (3.9) into equation (3.8) gives

$$f(x,y)=\sum_{j=1,j\neq i}^{N}\lambda_j g_j(x,y)+\lambda_{N+1} \tag{3.10}$$

where $g_j(x,y)=\sqrt{(x-x_j)^2+(y-y_j)^2+c_j^2}-\sqrt{(x-x_i)^2+(y-y_i)^2+c_i^2}$ (3.11)

The number of unknowns in equation (3.8) is N. As no confusion rises, λ_{N+1} can be replaced by λ_i, and equation (3.8) can be written as

$$f(x,y)=\sum_{j=1,j\neq i}^{N}\lambda_j g_j(x,y)+\lambda_i \tag{3.12}$$

It is easy to see that $f(x,y)$ in equation (3.12) constitutes N-dimensional linear vector space $\mathbf{V}^N$ with respect to the operation of addition and multiplication. From the concept

of linear independence, the bases of a vector space can be considered as linearly independent subset that spans the entire space. In the space $\mathbf{V}^N$, one set of base vectors is $g_i(x,y)=1$, and $g_j(x,y)$, $j=1,...,N$ but $j\neq i$ given by equation (3.11).

From the property of a linear vector space, if all the base functions satisfy the linear equation (3.7), so does any function in the space $\mathbf{V}^N$ represented by equation (3.12). There is an interesting feature. From equation (3.12), while all the base functions are given, the function $f(x,y)$ is still unknown since the coefficients λ_i are unknown. However, when all the base functions satisfy equation (3.7), we can guarantee that $f(x,y)$ also satisfies equation (3.7). In other words, we can guarantee that the solution of a partial differential equation approximated by the radial basis function satisfies equation (3.7). Thus, when the weighting coefficients of DQ approximation are determined by all the base functions, they can be used to discretize the derivatives in a partial differential equation. That is the essence of the RBF-DQ method.

Substituting all the base functions into equation (3.7a) as an example, we can obtain

$$0=\sum_{k=1}^{N} w_{i,k}^{(m)} \tag{3.13a}$$

$$\frac{\partial^m g_j(x_i,y_i)}{\partial x^m}=\sum_{k=1}^{N} w_{i,k}^{(m)} g_j(x_k,y_k),\ j=1,2,...,N,\text{ but }\ j\neq i \tag{3.13b}$$

For the given i, equation system (3.13) has N unknowns with N equations. So, solving this equation system can obtain the weighting coefficients $w_{i,k}^{(m)}$. From equation (3.11), one can easily obtain the first order derivative of $g_j(x,y)$ as

$$\frac{\partial g_j(x,y)}{\partial x} = \frac{x - x_j}{\sqrt{(x-x_j)^2 + (y-y_j)^2 + c_j^2}} - \frac{x - x_i}{\sqrt{(x-x_i)^2 + (y-y_i)^2 + c_i^2}}$$

In the matrix form, the weighting coefficient matrix of the x-derivative can then be determined by

$$[G][W^n]^T = \{G_x\} \tag{3.14}$$

where $[W^n]^T$ is the transpose of the weighting coefficient matrix $[W^n]$, and

$$[W^n] = \begin{bmatrix} w_{1,1}^{(n)} & w_{1,2}^{(n)} & \cdots & w_{1,N}^{(n)} \\ w_{2,1}^{(n)} & w_{2,2}^{(n)} & \cdots & w_{2,N}^{(n)} \\ \vdots & \vdots & \ddots & \vdots \\ w_{N,1}^{(n)} & w_{N,2}^{(n)} & \cdots & w_{N,N}^{(n)} \end{bmatrix}, \quad [G] = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ g_1(x_1,y_1) & g_1(x_2,y_2) & \cdots & g_1(x_N,y_N) \\ \vdots & \vdots & \ddots & \vdots \\ g_N(x_1,y_1) & g_N(x_2,y_2) & \cdots & g_N(x_N,y_N) \end{bmatrix}$$

$$[G_x] = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ g_x^n(1,1) & g_x^n(1,2) & \cdots & g_x^n(1,N) \\ \vdots & \vdots & \ddots & \vdots \\ g_x^n(N,1) & g_x^n(N,2) & \cdots & g_x^n(N,N) \end{bmatrix}$$

With the known matrices [G] and [G_x], the weighting coefficient matrix $[W^n]$ can be obtained by using a direct method of LU decomposition. The weighting coefficient matrix of the y-derivative can be obtained in a similar manner. Using these weighting coefficients, we can discretize the spatial derivatives, and transform the governing equations into a system of algebraic equations, which can be solved by iterative or direct method.

One of the most attractive properties in above method is that the weighting coefficients are only related to the basis functions and the position of the knots. That character is very appealing when we deal with the nonlinear problems. Since the derivatives are directly discretized, the method can be consistently well applied to linear and nonlinear problems. Another attractive property of RBF-DQ method is that it is naturally mesh-free, i.e., all the information required about the knots in the domain is nothing but their positions.

3.5 Local RBF-DQ Method

The RBF-DQ method presented in the last section is a global approach. In other words, the function approximation form (3.12) uses all the knots in the computational domain. When the number of knots, N, is large, the matrix $[G]$ may be ill-conditioned. This limits its application. To improve it, we developed the local RBF-DQ method. To do this, at every knot in the domain, we construct a local support region. The local support in this method has the same configuration as that discussed in the LSFD method. As shown in Fig. 2.2, at any knot, there is a supporting region, in which there are N knots randomly distributed. So, equation (3.12) is applied in the local support. That is the only difference between the local RBF-DQ method and the global RBF-DQ method. All the related formulations are the same for these two versions of RBF-DQ method.

As shown in the previous section, the MQ approximation of the function contains a shape parameter c that could be knot-dependent and must be determined by the user. It is well known that the value of c strongly influences the accuracy of MQ approximation, which

is used to approximate the solution of PDEs. Thus, there exists a problem of how to select a "good" value of c so that the numerical solution of PDEs can achieve satisfactory accuracy. In general, there are three main factors that could affect the optimal shape parameter c for giving the most accurate results. These three factors are the scale of supporting region, the number of supporting knots, and the distribution of supporting knots. Among the three factors, the effect of knot distribution is the most difficult to be studied since there are infinite kinds of distribution. In this section, we will mainly discuss how to minimize the effect of two factors, that is, the scale of supporting region and the number of supporting knots, on the shape parameter *c*.

In the local MQ-DQ method, the number of supporting knots is usually fixed for an application. Since the knots are randomly generated, the scale of supporting region for each reference knot could be different, and the optimal shape parameter c for accurate numerical results may also be different. Usually, it is very difficult to assign different values of *c* at different knots. However, this difficulty can be removed from the normalization of scale in the supporting region. The idea is actually motivated from the finite element method, where each element is usually mapped into a regular shape in the computational space. The essence of this idea is to transform the local support to a unit square for the two dimensional case or a unit box for the three dimensional case. So, the discussion about the optimal shape parameter is now confined to the MQ test functions in the unit square or box. The coordinate transformation has the form

$$\bar{x} = \frac{x}{D_i}, \ \bar{y} = \frac{y}{D_i} \tag{3.15}$$

where (x, y) represents the coordinates of supporting region in the physical space, $(\bar{x}, \bar{y})$ denotes the coordinates in the unit square, D_i is the diameter of the minimal circle enclosing all knots in the supporting region for the knot *i*. The corresponding MQ test functions in the local support now become

$$\varphi = \sqrt{\left(\bar{x} - \frac{x_i}{D_i}\right)^2 + \left(\bar{y} - \frac{y_i}{D_i}\right)^2 + \bar{c}^2}\ ,\ i = 1,...,N\ , \tag{3.16}$$

where *N* is the total number of the knots in the support. Compared with traditional MQ-RBF, we can find that the shape parameter c is equivalent to $\bar{c}D_i$. The coordinate transformation (3.15) also changes the formulation of the weighting coefficients in the local MQ-DQ approximation. For example, by using the differential chain rule, the first order partial derivative with respect to x can be written as

$$\frac{df}{dx} = \frac{df}{d\bar{x}}\frac{d\bar{x}}{dx} = \frac{1}{D_i}\frac{df}{d\bar{x}} = \frac{1}{D_i}\sum_{j=1}^{N} w_j^{(1x)} f_j = \sum_{j=1}^{N} \frac{w_j^{(1x)}}{D_i} f_j \tag{3.17}$$

where $w_j^{(1x)}$ are the weighting coefficients computed in the unit square, $w_j^{(1x)} / D_i$ are the actual weighting coefficients in the physical domain. Clearly, when D_i is changed, the equivalent c in the physical space is automatically changed. In our application, $\bar{c}$ is chosen as a constant. Its optimal value depends on the number of supporting knots. In the next section, we will discuss this through a test example. The following present a subroutine that is implemented to compute the derivative coefficients by local RBF-DQ method. Many parameters in this subroutine have the same meanings as those in the one for LSFD method. The programming is also very straightforward and exactly follows the description in the notes.

```
c-----------------------------------------------------------------------------------------------
c---- This program is used to calculate the derivative coefficients in the Local MQ-DQ
c---- method.

c---- INPUT: pxy, xy, c
c---- OUTPUT: r
c----  pxy: store the positions of the supporting points
c---- xy: store the position of the reference node
c---- c: shape parameter for the MQ radial basis function
c---- r:  vector of computed derivative coefficients

c---- Some important symbols and variables
c---- np: the number of supporting points
c---- A: coefficient matrix constructed from the basis functions
c---- b: derivative vectors of  the basis functions
c-----------------------------------------------------------------------------------------------

      subroutine MQRBF(pxy,xy,c,r)
      parameter(np=12,nd=np+1)
      implicit real*8(a-h,o-z)
      dimension pxy(np,2),xy(2),r(nd,5),pn(nd,2)
      dimension a(nd,nd),b(nd,5))

      do 20 i=1,nd
         if(i.ne.nd)then
                pn(i,1)=pxy(i,1)
                pn(i,2)=pxy(i,2)
         else
                pn(i,1)=xy(1)
                pn(i,2)=xy(2)
         endif
20    continue

         scaling=0.d0
         do i=1,np
                dx=pxy(i,1)-xy(1)
                dy=pxy(i,2)-xy(2)
                scaling=dmax1(scaling,dsqrt(dx*dx+dy*dy))
         enddo
         scaling=scaling*2.0

         do j=1,nd
                a(nd,j)=1.d0
         enddo

         do 19 i=1,nd-1
```

```
      do 19 j=1,nd
            dx=(pn(j,1)-pn(i,1))/scaling
            dy=(pn(j,2)-pn(i,2))/scaling
            dxk=(pn(j,1)-pn(nd,1))/scaling
            dyk=(pn(j,2)-pn(nd,2))/scaling
            a(i,j)=dsqrt(dx*dx+dy*dy+c)-dsqrt(dxk*dxk+dyk*dyk+c)
19    continue

      do 23 i=1,nd-1
            dx=(-pn(i,1)+pn(nd,1))/scaling
            dy=(-pn(i,2)+pn(nd,2))/scaling
            ffunc=dsqrt(dx*dx+dy*dy+c)
            b(i,1)=dx/ffunc
            b(i,2)=dy/ffunc
            b(i,3)=(dy*dy+c)/(ffunc**3.)-1.d0/dsqrt(c)
            b(i,5)=-dx*dy/(ffunc**3.)
            b(i,4)=(dx*dx+c)/(ffunc**3.)-1.d0/dsqrt(c)
23    continue
      b(nd,1)=0.
      b(nd,2)=0.
      b(nd,3)=0.
      b(nd,4)=0.
      b(nd,5)=0.

      do i=1,nd
      do j=1,5
            r(i,j)=b(i,j)
      enddo
      enddo
      call agjdn(a,r,nd,5,l)

c---- Recover the derivative coefficients from the scaling -----
      do ik1=1,5
      do ik2=1,nd
            if(ik1.eq.1 .or. ik1.eq.2)then
                  r(ik2,ik1)=r(ik2,ik1)/scaling
            elseif(ik1.eq.3 .or. ik1.eq.4 .or. ik1.eq.5)then
                  r(ik2,ik1)=r(ik2,ik1)/scaling/scaling
            endif
      enddo
      enddo
    return
    end


c-------This subroutine to numerically solve a linear problem Ax=b, in which A is a
```

```
c-------.nxn square matrix and b is a nxm matrix.
c------- The solution x is stored in b while the computation ends.
c------- Parameter l is used to return the information whether the computation is
c------- successfully performed. 1: Yes  0: No

      subroutine agjdn(a,b,n,m,l)
      implicit real*8(a-h,o-z)
      dimension a(n,n),b(n,m),js(n)

      l=1
      do 8100 k=1,n
        q=0.d0
        do 810 i=k,n
        do 810 j=k,n
          if (dabs(a(i,j)).gt.q) then
            q=dabs(a(i,j))
            js(k)=j
            is=i
          end if
810     continue
        if (q+1.0.eq.1.0) then
          write(*,820)
          l=0
          return
        end if
820     format(1x,'  fail  ')
        do 830 j=k,n
          d=a(k,j)
          a(k,j)=a(is,j)
          a(is,j)=d
830     continue
        do 840 j=1,m
          d=b(k,j)
          b(k,j)=b(is,j)
          b(is,j)=d
840     continue
        do 850 i=1,n
          d=a(i,k)
          a(i,k)=a(i,js(k))
          a(i,js(k))=d
850     continue
        do 860 j=k+1,n
860     a(k,j)=a(k,j)/a(k,k)
        do 870 j=1,m
870     b(k,j)=b(k,j)/a(k,k)
        do 890 i=1,n
```

```
          if (i.ne.k) then
            do 880 j=k+1,n
880         a(i,j)=a(i,j)-a(i,k)*a(k,j)
            do 885 j=1,m
885         b(i,j)=b(i,j)-a(i,k)*b(k,j)
          end if
890     continue
8100  continue
      do 8110 k=n,1,-1
      do 8110 j=1,m
        d=b(k,j)
        b(k,j)=b(js(k),j)
        b(js(k),j)=d
8110  continue
      return
      end
```

3.6 Sample Applications of Local RBF-DQ Method

Poisson equation

The optimal shape parameter is also related to the number of supporting knots. We will study this effect through a sample problem. Consider the two-dimensional Poisson equation in a square domain ($0 \le x \le 1, 0 \le y \le 1$),

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = g(x, y) \tag{3.18}$$

Suppose that the exact solution is given as

$$u(x, y) = \frac{\frac{5}{4} + \cos(5.4y)}{6 + 6(3x - 1)^2} \tag{3.19}$$

Equation (3.19) will be used to provide the Dirichlet condition on the boundary, the function $g(x, y)$, and to validate the numerical solution. The L_2 norm of relative error is taken to measure the accuracy of numerical results, which is defined as

$$L_2 \text{ norm of relative error: } \sqrt{\frac{\sum_{i=1}^{N}\left(\frac{u_{numerical}-u_{analytical}}{u_{analytical}+10^{-8}}\right)^2}{N}} \tag{3.20}$$

To conduct numerical experiments, the knot distribution in the square domain is fixed, which is shown in Fig. 3.2. In total, there are 673 knots in the domain. The accuracy of numerical results in terms of L2 norm of relative error is studied by changing the shape parameter $\bar{c}$ and the number of knots in the supporting region. In this study, four different support sizes (numbers of supporting knots) are used for discretization, and they are 10, 16, 22 and 28.

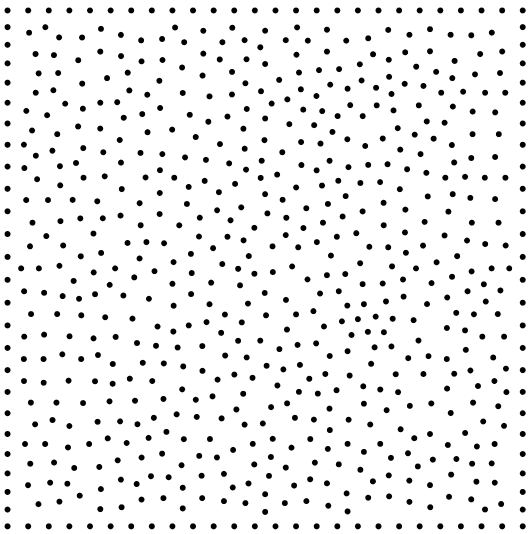


Figure 3.2 Irregular knot distribution for solution of sample PDEs

Fig. 3.3 illustrates the variation of accuracy with different shape parameter and support size (number of supporting knots). It can be seen from Fig. 3.3 that the L_2 norm of relative error depends on the value of shape parameter $\bar{c}$ and the support size. It was found that when the number of supporting knots is fixed, with increase of shape parameter $\bar{c}$, the accuracy of numerical results is improved. And when the shape parameter $\bar{c}$ is fixed, with increase of the supporting knots, the accuracy of numerical results is also improved. Another interesting phenomenon is that the shape parameter $\bar{c}$ with small number of supporting knots is less sensitive than that with large number of supporting knots. In other words, when the number of supporting knots is relatively small, the shape parameter $\bar{c}$ can be chosen in a wide range to get a convergent solution, in which the accuracy of numerical solution is changed gradually. But when the number of supporting knots is large, the shape parameter $\bar{c}$ can only be selected in a small range to get convergent solution, in which the accuracy of numerical results changes sharply. So, one has to balance the good accuracy of numerical results and the sensitivity of the shape parameter $\bar{c}$ when the number of supporting knots is chosen. From our experiences, 16 supporting knots are a suitable choice.

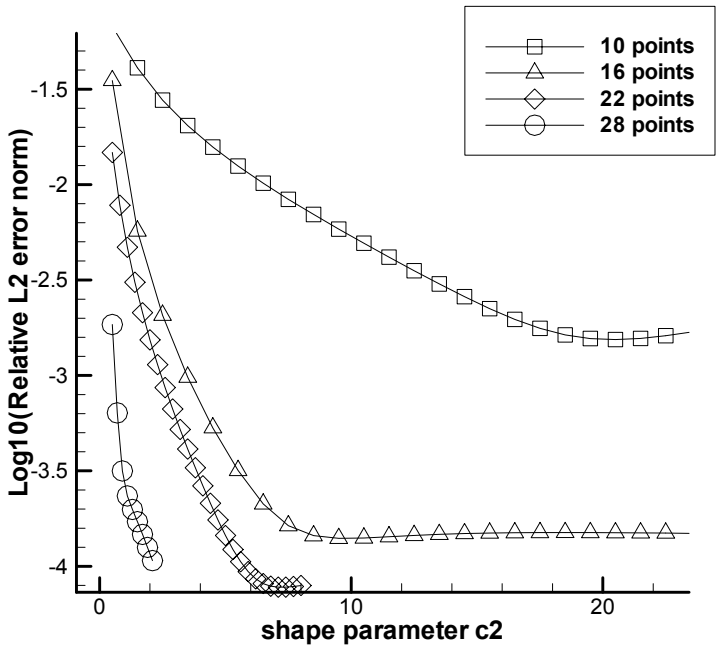


Figure 3.3 Log_{10}(error) vs $\bar{c}^2$ with irregular knot distribution for Poisson problem

Advection-diffusion equation

The discretization of the derivatives by local RBF-DQ method in the nonlinear PDEs follows the same way as that in the linear PDEs. It is interesting to see whether the effect of shape parameter $\bar{c}$ on the accuracy of numerical solution for a nonlinear equation behaves in the same way or a similar way to the linear equation. To study this, we consider the following nonlinear equation,

$$\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}+u\left(\frac{\partial u}{\partial x}+\frac{\partial u}{\partial y}\right)=q(x,y) \tag{3.21}$$

For simplicity, we suppose that the exact solution of equation (3.21) is also given by equation (3.19), which is used to determine the function $q(x,y)$ and the boundary

condition. It was found that when the same conditions such as knot distribution, shape parameter, and the number of supporting knots are used, the accuracy of numerical results for equation (3.21) is very close to the accuracy for equation (3.18). This can be clearly observed in Table 3.1, which compares the accuracy of results for linear and nonlinear equations with the use of 22 supporting knots. This observation is very interesting. It may imply that the choice of shape parameter is operator-independent. From this point of view, we can first study the sample problem in details, and get an optimal shape parameter $\bar{c}$. Then this optimal value is used to solve incompressible Navier-Stokes (N-S) equations. It is indeed that our computation of incompressible flow problems follows this process.

Table 3.1 Comparison of accuracy for linear and nonlinear equations with using 22 supporting knots

Shape parameter $\bar{c}^2$	$Log_{10}(L_2$ norm of relative error)	
	Linear equation	Nonlinear equation
0.500	-1.8332	-1.8314
1.400	-2.5138	-2.5117
2.300	-2.9455	-2.9431
3.500	-3.3879	-3.3852
5.000	-3.8410	-3.8386
6.500	-4.0905	-4.0903

3.7 Application of Local RBF-DQ Method to Flow Problems

For their physical complexity and practicality, the flow and thermal fields in enclosed space are of great importance due to their wide applications such as in solar collector-

receivers, insulation and flooding protection for buried pipes used for district heating and cooling, cooling systems in nuclear reactors, etc. The purpose of this section is to investigate how the mesh-free methods behave in the solution of the natural convection problem with complex geometry. A schematic view of a horizontal eccentric annulus between a square outer cylinder and a heated circular inner cylinder is shown in Fig. 3.4. Heat is generated uniformly within the circular inner cylinder, which is placed concentrically or eccentrically within the cold square cylinder. From the non-slip condition, the velocities u and v on both the inner and outer cylinder walls are zero. For an eccentric annulus, the stream function values on the inner and outer cylinders are different and a global circulation flow along the inner cylinder exists. The stream function value on the outer cylinder wall is set to zero. The boundary condition can be written as

$$u\big|_{inner\_wall} = u\big|_{outer\_wall} = 0, \quad v\big|_{inner\_wall} = v\big|_{outer\_wall} = 0 \tag{3.22}$$

$$\psi\big|_{inner\_wall} = \text{constant}, \quad \psi\big|_{outer\_wall} = 0 \tag{3.23}$$

$$T\big|_{inner\_wall} = 1, \quad T\big|_{outer\_wall} = 0 \tag{3.24}$$

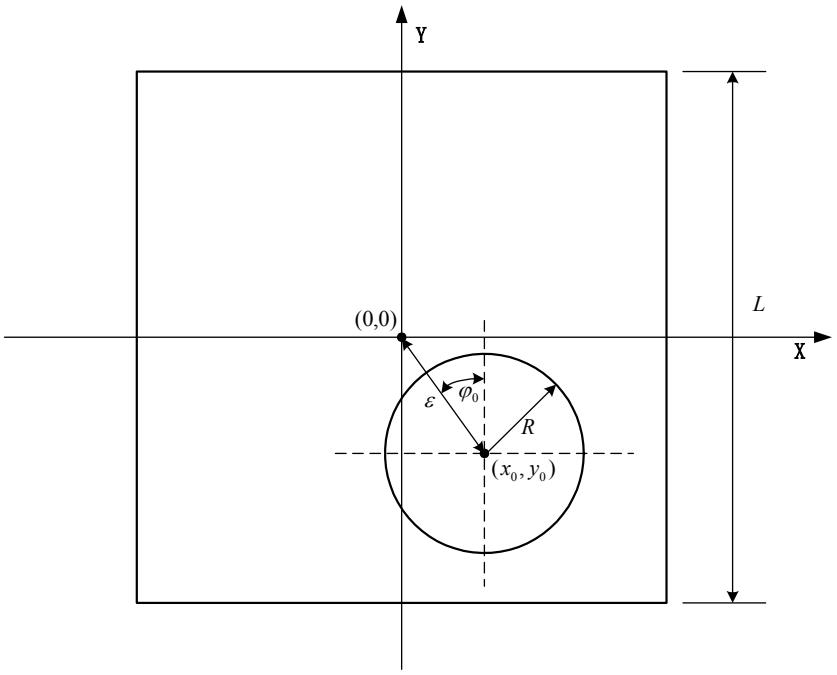
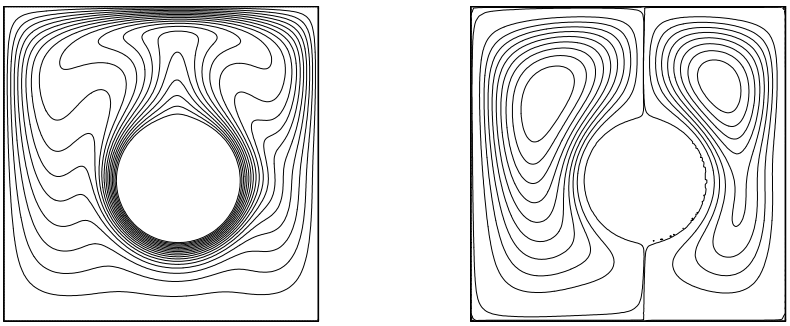


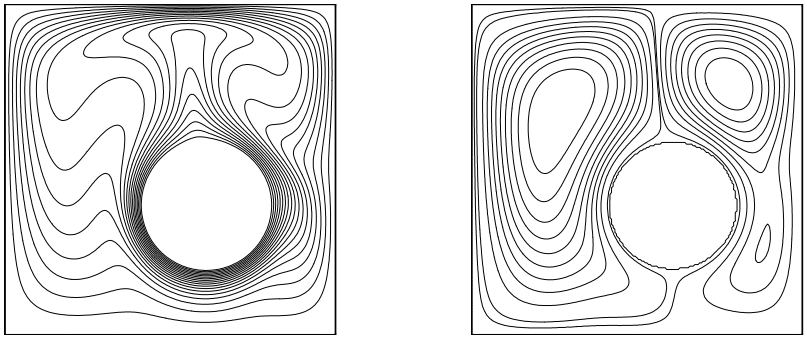
Figure 3.4 Sketch of physical domain of natural convection between a square outer cylinder and a circular inner cylinder

The governing equations for this problem are the same as equations (2.34)-(2.36). The derivatives in the governing equations are discretized by the local MQ-DQ method, and the Neumann boundary conditions are approximated by the conventional FD schemes. The number of supporting knots is taken as 17, and the shape parameter $\bar{c}^2$ is selected as 3.1. The numerical results for the cases with $\varphi_0 = 45^0$, $Ra = 3 \times 10^5$ and ($rr = 2.6$ $rr = L/(2R)$) are presented by the streamlines and vorticity contours. As shown in Fig. 3.5, the eddy on the left hand side in the flow expands in size due to the increasing space, with the center of the eddy moving downwards. The thermal plume tends to incline to the left from the vertical line as the eccentricity increases. The eddy on the RHS remains the similar size but shifts above the inner cylinder. The increasing

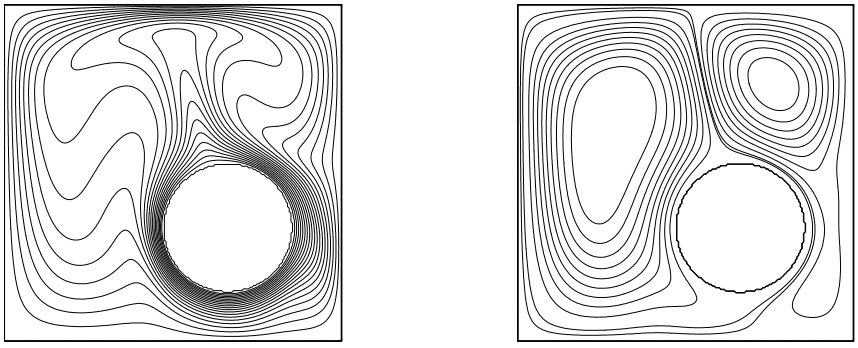
eccentricity allows larger space for the eddy on the RHS, but the increasing eddy on the left hand side limits the space for the eddy on the RHS. It is the balance between the two eddies that make the thermal plumes above the top of the inner cylinder shifts from the vertical line to the left.



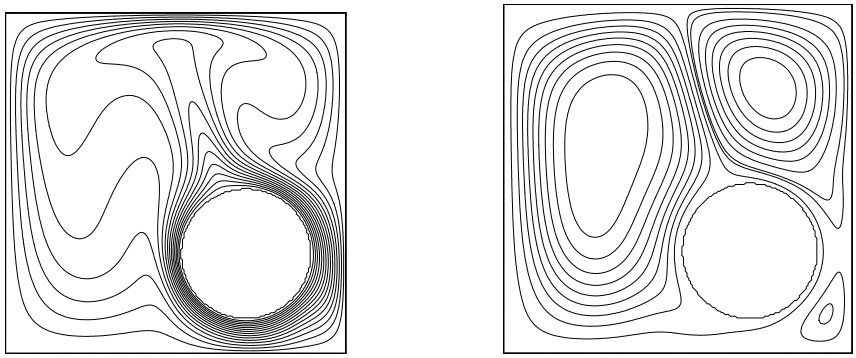
ε =0.25



ε =0.50



ε =0.75



ε =0.95

Figure 3.5 Streamlines and isotherms for $Ra = 3 \times 10^5$, $rr = 2.6$, and $\varphi_0 = 45^0$

References

R.L. Hardy, Multiquadric equations of topography and other irregular surfaces, Journal of Geophysical Research, **76**, pp1905-1915 (1971).

R. Franke, "Scattered data interpolation: tests of some methods", Math. Comp., **38**, pp. 181-199, (1982).

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**, No (6-8) pp127-145 (1990).

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**, No (6-8) pp147-161 (1990).

R. E. Bellman, B. G. Kashef, and J. Casti, "Differential quadrature: A technique for the rapid solution of nonlinear partial differential equations," J. Comput. Phys. **10**, 40-52 (1972).

C. Shu, Differential quadrature and its application in engineering, Springer-Verlag, London, 2000.

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).

C. Shu and Y. T. Chew, "Fourier expansion-based differential quadrature and its application to Helmholtz eigenvalue problems," Commun. Numer. Methods Eng. **13**, 643-653 (1997).

C. Shu, H. Ding, K. S. Yeo, 'Local Radial Basis Function-based Differential Quadrature Method and Its Application to Solve Two-dimensional Incompressible Navier-Stokes Equations', *Computer Methods in Applied Mechanics and Engineering*, Vol. **192**, 941-954 (2003).