COMMUNICATIONS IN NUMERICAL METHODS IN ENGINEERING
Commun. Numer. Meth. Engng 2008; **24**:573–584
Published online 12 January 2007 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/cnm.978

An interpolation-based local differential quadrature method to solve partial differential equations using irregularly distributed nodes

Hang Ma[1] and Qing-Hua Qin[2,*,†]

[1]*Department of Mechanics, College of Sciences, Shanghai University, Shanghai 200444, People's Republic of China*
[2]*Department of Engineering, Australian National University, ACT 0200, Australia*

SUMMARY

To circumvent the constraint in application of the conventional differential quadrature (DQ) method that the solution domain has to be a regular region, an interpolation-based local differential quadrature (LDQ) method is proposed in this paper. Instead of using regular nodes placed on mesh lines in the DQ method (DQM), irregularly distributed nodes are employed in the LDQ method. That is, any spatial derivative at a nodal point is approximated by a linear weighted sum of the functional values of irregularly distributed nodes in the local physical domain. The feature of the new approach lies in the fact that the weighting coefficients are determined by the quadrature rule over the irregularly distributed local supporting nodes with the aid of nodal interpolation techniques developed in the paper. Because of this distinctive feature, the LDQ method can be consistently applied to linear and nonlinear problems and is really a mesh-free method without the limitation in the solution domain of the conventional DQM. The effectiveness and efficiency of the method are validated by two simple numerical examples by solving boundary-value problems of a linear and a nonlinear partial differential equation. Copyright © 2007 John Wiley & Sons, Ltd.

Received 21 July 2006; Revised 5 November 2006; Accepted 24 November 2006

KEY WORDS: differential quadrature method; irregular nodes; interpolation; mesh free; partial differential equation

1. INTRODUCTION

The differential quadrature (DQ) method was introduced by Bellman and his co-workers in the early 1970s [1], following the concept of integral quadrature. It has been shown by many researchers that the DQ method (DQM) is an attractive numerical tool with high efficiency and accuracy. A comprehensive review of the work related to the method can be found in the paper of Bert and

*Correspondence to: Qing-Hua Qin, Department of Engineering, Australian National University, ACT 0200, Australia.
†E-mail: qhqin@tju.edu.cn

Malik [2]. Since then, the DQM has been recognized as a numerically accurate and computationally efficient numerical technique, and has been used successfully to deal with a large number of problems of physical and engineering science [3–6]. The basic idea of the DQM is that any derivative at a mesh point can be approximated by a weighted linear sum of all the functional values along a mesh line. The key procedure in the DQM is the determination of weighting coefficients. As shown by Shu and Richards [7], when the solution of a partial differential equation (PDE) is approximated by a high-order polynomial, the weighting coefficients can be computed by a simple algebraic formulation or by a recurrence relationship. However, as pointed out by Bert and Malik [2], the DQM is limited to applications for domains having boundaries that are aligned with the co-ordinate axes, such as rectangular and circular domains. For problems with complex geometry, the DQM cannot be directly applied. One has to rely on the co-ordinate transformation [8–10], the quadrature element method [11, 12], or the DQM based on the boundary integral equations [13]. Liew *et al*. proposed a moving least-squares DQM to solve plate problems where scattered nodes can be used [14, 15]. Although with these techniques very good results can be obtained for problems with complex geometry, one has to admit that the process is somewhat complex, and the approach is not as flexible as the finite element method.

There is a practical demand to develop a more efficient and flexible method for solving complex problems like the local differential quadrature (LDQ) method. In the LDQ method, a spatial derivative at a nodal point is approximated by the linear weighted sum of the functional values of nodes in the local physical domain only. The LDQ method was first proposed by Sun and Zhu for solving problems of incompressible viscous flow using regularly distributed nodal points [16]. Using the LDQ method, Zong and Lam solved the two-dimensional wave equation [17]. Recently, in a series of papers, Shu *et al*. successfully developed a radial basis function (RBF)-based DQM, which can be considered as a mesh-free method, using scattered nodes [18–21]. It is obvious that the application of the RBF-based DQM has no limitation in dimensions and complex geometries. However, the optimum shape parameter in the RBF must be selected numerically and incorporated with the special nodal distributions. The condition number of the system matrix would become large if the number of nodes increased in the case of the global RBF-DQ method [20]. In addition to the use of scattered nodes inside the solution domain, however, some regularly distributed nodes must be employed near the boundary in the case of the local RBF-DQ method [19, 21].

In the present work, an interpolation-based LDQ method is developed. The partial derivative of a function with respect to space variables at a reference point is approximated by a weighted linear summation of the functional values at a set of irregularly distributed local supporting nodes. The weighting coefficients are determined with the aid of the techniques of nodal interpolation. Two simple numerical examples are presented to validate the effectiveness and efficiency of the method by solving boundary-value problems of a linear and a nonlinear PDE.

2. THE LOCAL DQ METHOD

Since in the DQM, the derivatives of a function at a node point are approximated by a weighted linear sum of the functional values along a mesh line parallel to the co-ordinates, irregularly distributed nodes must be organized in some way before deriving the quadrature rule of the LDQ method. However, the basic idea of the organization is very simple, see below. As shown in Figure 1(a), a local co-ordinate system $\mathbf{I}-\mathbf{xy}$ can be established about the reference node $\mathbf{I}$ in the local area of the domain, surrounded by several supporting nodes. These nodes are chosen, in

DOI: 10.1002/cnm

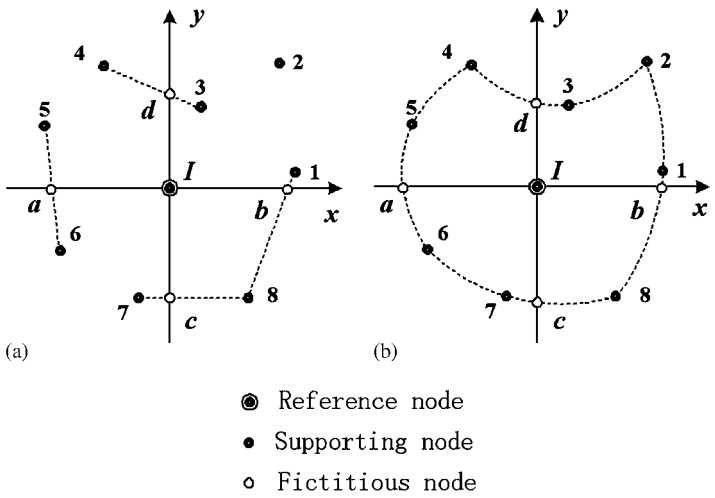

Figure 1. Schematics of deriving the quadrature rule in a local supporting area: (a) linear interpolation; and (b) quadratic interpolation.

general, to have the smallest distances from the node **I**. There are 8 of these nodes in Figure 1, for example, and they have been locally arranged and numbered in a counterclockwise manner. The purpose of the organization of the local supporting nodes is for ease of computer programming.

Now take the derivation of the quadrature rule in **x** direction as an example, by performing a 'nodal loop' (the name coming from computer programming). If the local **y** co-ordinates of the two adjacent nodes change the sign, then these two nodes can be organized as an 'element' (the linear element 5–6), which intersects with the **x**-axis at a point (the fictitious point **a** in Figure 1(a)). Then by continuing the nodal loop, if the local **y** co-ordinates of next two adjacent nodes change the sign again, then another linear element 8–1 and a cross point, the fictitious point **b**, can be obtained.

The local quadrature rule can be written, for the kth derivatives in **x** direction about the reference node **I**, as

$$\frac{\partial^k u_I}{\partial x^k} = A_{Ia}^{(k)} u_a + A_{II}^{(k)} u_I + A_{Ib}^{(k)} u_b \tag{1}$$

where u_i denotes the function values at node i, $A_{Ii}^{(k)}$ the weighting coefficients of the quadrature rule (see Appendix A at the end of this paper or refer to [2, 7, 9, 16] for the details of $A_{Ii}^{(k)}$). It needs to be pointed out that in the global DQM, any derivative at a mesh point can be approximated by a weighted linear sum of all the functional values along a mesh line. But in the local DQM, the derivative at a mesh point is approximated only by a weighted linear sum of the functional values on local nodes. The difference lies in the nodal number used in the quadrature rules between the two. However, the form of formulations for computing the weighting coefficients is exactly the same in both the global and local DQMs since it can be proved easily that the weighting coefficients are in reality the derivatives of the coefficients of Lagrange interpolation along that straight line.

DOI: 10.1002/cnm

As the function values at the fictitious points **a** and **b** are not the nodal values, they should be computed by Lagrange interpolation. Therefore, the local quadrature rule becomes

$$\frac{\partial^k u_I}{\partial x^k} = A_{Ia}^{(k)}(\phi_1^a u_5 + \phi_2^a u_6) + A_{II}^{(k)} u_I + A_{Ib}^{(k)}(\phi_1^b u_8 + \phi_2^b u_1) \tag{2}$$

where ϕ_j^a and ϕ_j^b $(j = 1, 2)$ denote the shape functions of the corresponding linear elements, respectively. The local quadrature rule in **y** direction can be derived (Figure 1(a)) in the same way, for the kth derivatives about the reference node **I**, as

$$\frac{\partial^k u_I}{\partial y^k} = B_{Ic}^{(k)}(\phi_1^c u_7 + \phi_2^c u_8) + B_{II}^{(k)} u_I + B_{Id}^{(k)}(\phi_1^d u_3 + \phi_2^d u_4) \tag{3}$$

where $B_{Ii}^{(k)}$ also represents the weighting coefficients. Equations (2) and (3) construct the LDQ rules with the aid of linear interpolation. Similarly, if three nodes are organized consecutively as a quadratic 'element' in performing the 'nodal loop' as shown in Figure 1(b), the LDQ rules with the aid of quadratic interpolation can be constructed as follows:

$$\frac{\partial^k u_I}{\partial x^k} = A_{Ia}^{(k)}(\phi_1^c u_4 + \phi_2^c u_5 + \phi_3^c u_6) + A_{II}^{(k)} u_I + A_{Ib}^{(k)}(\phi_1^d u_8 + \phi_2^d u_1 + \phi_3^d u_2) \tag{4}$$

$$\frac{\partial^k u_I}{\partial y^k} = B_{Ic}^{(k)}(\phi_1^a u_6 + \phi_2^a u_7 + \phi_3^a u_8) + B_{II}^{(k)} u_I + B_{Id}^{(k)}(\phi_1^b u_2 + \phi_2^b u_3 + \phi_3^b u_4) \tag{5}$$

where ϕ_j^a, ϕ_j^b, ϕ_j^c and ϕ_j^d $(j = 1, 2, 3)$ denote the shape functions of the corresponding quadratic elements, respectively. The local quadrature rule for the boundary nodes can be derived in much the same way as shown in Figure 2, the local co-ordinate system $\mathbf{B} - \mathbf{nt}$ is established along the normal and the tangential direction of the boundary Γ, respectively, about the reference point **B**. After selecting local nodes and organizing them as quadratic elements, the local quadrature rule for the outward normal can be written as

$$\begin{aligned}\frac{\partial u_B}{\partial n} &= N_{Be}^{(1)} u_e + N_{BB}^{(1)} u_B + N_{Bh}^{(1)} u_h \\ &= N_{Be}^{(1)}(\phi_1^e u_1 + \phi_2^e u_2 + \phi_3^e u_3) + N_{BB}^{(1)} u_B + N_{Bh}^{(1)}(\phi_1^h u_4 + \phi_2^h u_5 + \phi_3^h u_6)\end{aligned} \tag{6}$$

where $N_{Bi}^{(1)}$ denotes the weighting coefficients of the quadrature rule along the outward normal direction at the reference node **B**. This paper is confined to the case where the governing equation is of second-order derivatives, so that only first-order derivatives are considered in the expression of the local quadrature rule for the boundary nodes.

It is obvious that the construction of the local quadrature rule in the present work relies on the suitable arrangement and organization of the local supporting nodes, which can be performed automatically in the computer program (let the computer do it!). The local quadrature rule was tested and found to be very robust, because there is no drawback of obtaining an ill-posed set of local nodes, as possibly encountered in the generalized finite difference method using scattered nodes [22]. Using whole derivative formulas, Liu and Li [23] constructed a mesh-free method, which is in fact equivalent to the LDQ rules, with the aid of linear interpolation, Equations (2)

DOI: 10.1002/cnm

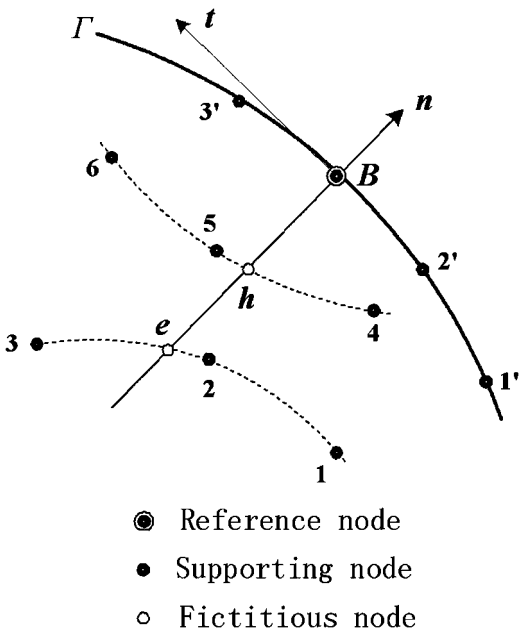

Figure 2. Schematics of deriving the quadrature rule on the boundary.

and (3). However, the derivation of the present method is much more concise and versatile, and it can easily be extended not only to cases of the quadrature rule with quadratic interpolation (Equations (4)–(5)), but also to the derivation of higher-order quadrature formulas. For example, the local nodes 1–8 in Figure 1(b) form a loop (the first loop), which can be considered the first layer. Obviously, one can form the second layer (the second loop) by another set of local nodes with slightly greater distances from the reference node, to construct a higher-order local quadrature formula. But in the present work, only the low-order local quadrature formula is considered.

3. NUMERICAL EXAMPLES

Two simple numerical examples of boundary-value problems are presented to demonstrate the validity and the potential of the LDQ method developed in this paper. The problem area is an elliptical domain, Ω, with a quarter elliptical cut as shown in Figure 3, with the long axis 1 and the short axis 0.6, discretized by irregularly distributed nodes. As reported earlier [2], the DQM can yield highly accurate results with quite a small number of sampling points. However, to describe the irregular geometry correctly, the total node numbers cannot be so small as those used for describing regular geometry such as rectangular. Four total node numbers, $n = 55$, 72, 106 and 186, were used in the present work.

3.1. Convection–diffusion equation

A convection–diffusion equation with varying parameters was numerically tested using the developed LDQ method. The controlling differential equation and the boundary conditions are

DOI: 10.1002/cnm

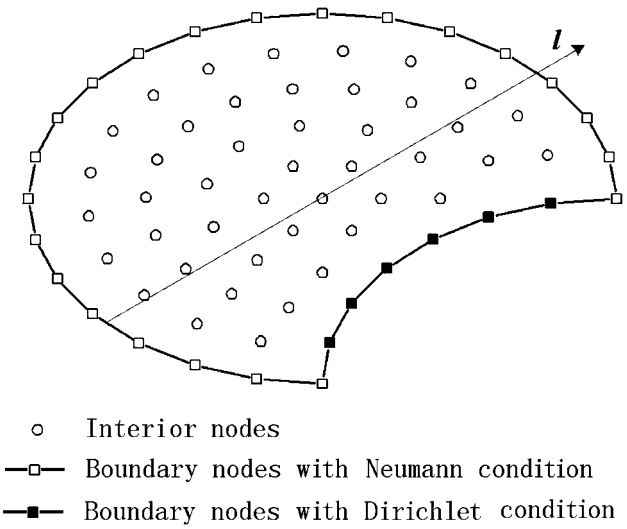

Figure 3. Discretization of the problem domain (total node number $n = 72$).

as follows:

$$\nabla^2 u + yu_{,x} + xu_{,y} = (1 - y)\exp(-x) + (1 - x)\exp(-y) \tag{7}$$

$$u(\bar{x}) = \bar{u} \quad (\bar{x} \in \Gamma_u), \qquad \frac{\partial u(\bar{x})}{\partial n} = \bar{q} \quad (\bar{x} \in \Gamma_q) \tag{8}$$

where Γ_u and Γ_q represent Dirichlet and Neumann boundaries, and $\bar{u}$ and $\bar{q}$ are prescribed values at the corresponding boundaries, respectively. The problem has an analytical solution:

$$u = \exp(-x) + \exp(-y) \tag{9}$$

By substituting the quadrature formulas (4) and (5) with quadratic interpolation, for example, into Equation (7), the controlling equation in discrete form about the reference node **I** (Figure 1(b)) can be written as

$$\begin{aligned}
&(A_{Ia}^{(2)} + y_I A_{Ia}^{(1)})(\phi_1^c u_4 + \phi_2^c u_5 + \phi_3^c u_6) + (A_{II}^{(2)} + y_I A_{II}^{(1)})u_I \\
&\quad + (A_{Ib}^{(2)} + y_I A_{Ib}^{(1)})(\phi_1^d u_8 + \phi_2^d u_1 + \phi_3^d u_2) \\
&\quad + (B_{Ic}^{(2)} + x_I B_{Ic}^{(1)})(\phi_1^a u_6 + \phi_2^a u_7 + \phi_3^a u_8) + (B_{II}^{(2)} + x_I B_{II}^{(1)})u_I \\
&\quad + (B_{Id}^{(2)} + x_I B_{Id}^{(1)})(\phi_1^b u_2 + \phi_2^b u_3 + \phi_3^b u_4) \\
&= (1 - y_I)\exp(-x_I) + (1 - x_I)\exp(-y_I)
\end{aligned} \tag{10}$$

Similarly, the Neumann boundary condition about the node **B** (Figure 2) in discrete form can be written as

$$\bar{q}_B = N_{Be}^{(1)}(\phi_1^e u_1 + \phi_2^e u_2 + \phi_3^e u_3) + N_{BB}^{(1)} u_B + N_{Bh}^{(1)}(\phi_1^h u_4 + \phi_2^h u_5 + \phi_3^h u_6) \tag{11}$$

DOI: 10.1002/cnm

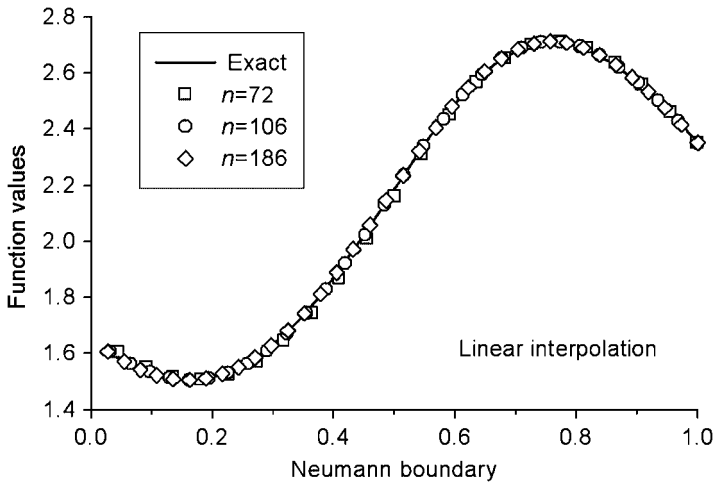

Figure 4. Comparison of nodal function values along the Neumann boundary.

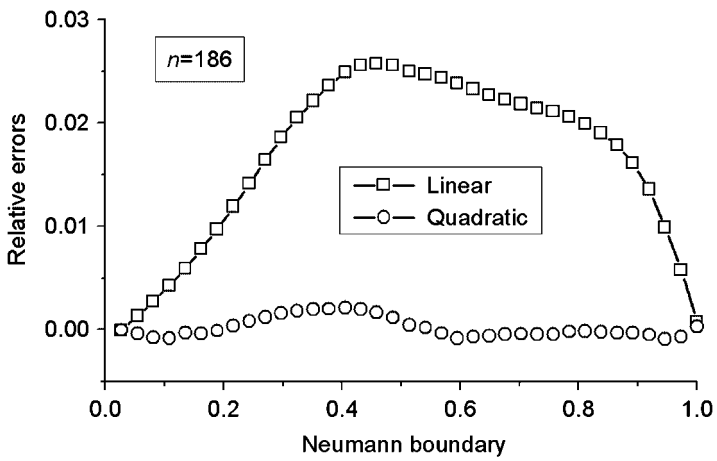

Figure 5. Comparison of nodal errors along the Neumann boundary.

By collecting globally all the discrete equations for the domain and boundary nodes with Neumann condition, incorporated with the Dirichlet boundary condition, the system equations in matrix form can be obtained:

$$\mathbf{A}\mathbf{x} = \mathbf{b} \tag{12}$$

The problem was solved by the quadrature rules with the aid of both linear and quadratic interpolations. The numerical solutions using different total node numbers with linear interpolation are compared with the exact solution in Figure 4 along the Neumann boundary. The relative errors of the two interpolations using the total node number $n = 186$ are shown in Figure 5. The maximum and the root mean square errors for all the internal and Neumann boundary nodes are summarized

DOI: 10.1002/cnm

Table I. Comparison of relative errors between linear and quadratic interpolation for the convection–diffusion equation.

	Maximum errors				Root mean square errors			
Total nodal number	55	72	106	186	55	72	106	186
Linear interpolation	0.0609	0.0296	0.0273	0.0257	0.0411	0.0199	0.0188	0.0177
Quadratic interpolation	0.0134	0.0096	0.0072	0.0072	0.0091	0.0055	0.0042	0.0008

in Table I. The numerical results of the convection–diffusion equation show that the proposed method is robust, accurate and efficient. The accuracy with quadratic interpolation is consistently better than that with linear interpolation, which shows almost no change with the increase in total node numbers. It can be seen that the LDQ method retains mesh-free features and has the potential to become an efficient approach for solving PDEs.

3.2. A nonlinear partial differential equation

The second example considered is a nonlinear differential equation with the Dirichlet boundary condition as follows:

$$L(u) = 4u\Delta u - 10u_{,x}\,u_{,y} + 5 = 0, \quad u(\bar{x}) = \bar{u} \quad (\bar{x} \in \Gamma_u) \tag{13}$$

with an exact solution $u = \cos(x + 0.5y)$. The domain in consideration is the same as the previous problem as shown in Figure 3. With the proposed LDQ method, this nonlinear problem can be solved by applying the quadrature rules (2) and (3) or (4) and (5) without much difficulty. However, the numerical solution of (13) should follow some iterative procedure. For this purpose, we adopt Newton's approach wherein, beginning with an assumed field $u^{(0)}$ consistent with the boundary condition, the successively refined solutions can be obtained through the following iterative scheme:

$$u^{(j+1)} = u^{(j)} + \theta^{(j)} \tag{14}$$

where $\theta = \theta(x, y)$ is the refinement of u and j is the iteration count. The refinement θ is determined by the solution of the following equation written in the operator form as

$$\theta L'(u) + L(u) = 0 \tag{15}$$

where L' represents the Frechet derivative defined as [2]

$$\theta L'(u) = \frac{\partial}{\partial \varepsilon} L(u + \varepsilon\theta)|_{\varepsilon=0} \tag{16}$$

Substituting the Frechet derivative (16) and (13) into (15), a linear equation in terms of the refinement θ is obtained

$$4u\Delta\theta - 10u_{,y}\,\theta_{,x} - 10u_{,x}\,\theta_{,y} + 4\Delta u\theta = -(4u\Delta u - 10u_{,x}\,u_{,y} + 5) \tag{17}$$

DOI: 10.1002/cnm

By applying the quadrature rules (4) and (5), for example, into (17) at the reference nodal point **I** (Figure 1(b)), the discrete equation yields as follows:

$$\begin{aligned}&(4uA_{Ia}^{(2)} - 10u_{,y}\,A_{Ia}^{(1)})(\phi_1^c\theta_4 + \phi_2^c\theta_5 + \phi_3^c\theta_6) + (4uA_{II}^{(2)} - 10u_{,y}\,A_{II}^{(1)})\theta_I\\ &\quad + (4uA_{Ib}^{(2)} - 10u_{,y}\,A_{Ib}^{(1)})(\phi_1^d\theta_8 + \phi_2^d\theta_1 + \phi_3^d\theta_2)\\ &\quad + (4uB_{Ic}^{(2)} - 10u_{,x}\,B_{Ic}^{(1)})(\phi_1^a\theta_6 + \phi_2^a\theta_7 + \phi_3^a\theta_8) + (4uB_{II}^{(2)} - 10u_{,x}\,B_{II}^{(1)})\theta_I\\ &\quad + (4uB_{Id}^{(2)} - 10u_{,x}\,B_{Id}^{(1)})(\phi_1^b\theta_2 + \phi_2^b\theta_3 + \phi_3^b\theta_4) + 4\Delta u\theta_I = -(4u\Delta u - 10u_{,x}\,u_{,y} + 5)\end{aligned} \quad (18)$$

where the following boundary condition for the θ-variables has been included in (19):

$$\theta(x, y) = 0, \quad (x, y \in \Gamma_u) \quad (19)$$

In the iteration scheme of Newton's method, the following convergence criterion was employed:

$$\sum(\theta_k)^2 \leqslant 10^{-10} \quad (20)$$

The relative errors of the linear and the quadratic interpolations using the total node number $n = 186$ along the skew line, **l** ($y/x = \tan 30$, Figure 3), for the nonlinear equation are shown in

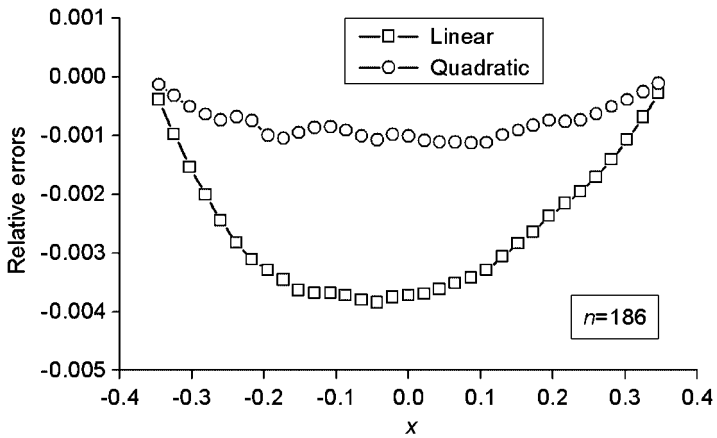

Figure 6. Comparison of errors along the skew line, **l**, for the nonlinear equation.

Table II. Comparison of relative errors between linear and quadratic interpolation for the nonlinear partial differential equation.

	Maximum errors				Root mean square errors			
Total nodal number	55	72	106	186	55	72	106	186
Linear interpolation	0.1074	0.0051	0.0043	0.0040	0.0796	0.0031	0.0026	0.0025
Quadratic interpolation	0.0066	0.0023	0.0018	0.0017	0.0023	0.0013	0.0006	0.0007

DOI: 10.1002/cnm

Figure 6. The maximum and the root mean square errors for all the internal nodes are compared in Table II. The numerical results of the nonlinear differential equation show again that the accuracy with quadratic interpolation is consistently better than that with linear interpolation. With its really mesh-free features, the LDQ method can be consistently applied successfully to linear and nonlinear problems without the limitation in solution domain of the conventional DQM.

4. CONCLUSION

The local differential quadrature (LDQ) method was successfully developed using irregularly distributed nodes, where any spatial derivative at a nodal point is approximated by a linear weighted sum of the functional values of nodes in the local physical domain, which retains mesh-free features. Numerical results show that the accuracy with quadratic interpolation is consistently better than that with linear interpolation. It is validated that the proposed method is robust, accurate and efficient and has the potential to become an efficient approach for solving partial differential equations (PDEs). The LDQ method can be consistently applied successfully to linear and nonlinear problems and is in fact a mesh-free method without the limitation in the solution domain of the conventional DQM.

APPENDIX A

Suppose $x_i (i = 1, \ldots, n)$ are distinct points along a straight line in parallel with one of the co-ordinates. We first write the coefficients of Lagrange interpolation within an interval which only includes three adjacent local points x_{i-1}, x_i and x_{i+1}:

$$A_{i-1}^{(0)}(x) = \frac{(x - x_i)(x - x_{i+1})}{(x_{i-1} - x_i)(x_{i-1} - x_{i+1})} \tag{A1}$$

$$A_i^{(0)}(x) = \frac{(x - x_{i-1})(x - x_{i+1})}{(x_i - x_{i-1})(x_i - x_{i+1})} \tag{A2}$$

$$A_{i+1}^{(0)}(x) = \frac{(x - x_{i-1})(x - x_i)}{(x_{i+1} - x_{i-1})(x_{i+1} - x_i)} \tag{A3}$$

The differentiation of $A_k^{(0)}(x)$ with respect to x yields

$$A_{i-1}^{(1)}(x) = \frac{2x - x_i - x_{i+1}}{(x_{i-1} - x_i)(x_{i-1} - x_{i+1})} \tag{A4}$$

$$A_i^{(1)}(x) = \frac{2x - x_{i-1} - x_{i+1}}{(x_i - x_{i-1})(x_i - x_{i+1})} \tag{A5}$$

$$A_{i+1}^{(1)}(x) = \frac{2x - x_{i-1} - x_i}{(x_{i+1} - x_{i-1})(x_{i+1} - x_i)} \tag{A6}$$

DOI: 10.1002/cnm

Then the weighting coefficients of the local quadrature rule corresponding to the first derivative can be obtained

$$A^{(1)}_{i-1,i-1} = A^{(1)}_{i-1}(x_{i-1}) = \frac{2x_{i-1} - x_i - x_{i+1}}{(x_{i-1} - x_i)(x_{i-1} - x_{i+1})} \tag{A7}$$

$$A^{(1)}_{i,i-1} = A^{(1)}_{i-1}(x_i) = \frac{x_i - x_{i+1}}{(x_{i-1} - x_i)(x_{i-1} - x_{i+1})} \tag{A8}$$

$$A^{(1)}_{i+1,i-1} = A^{(1)}_{i-1}(x_{i+1}) = \frac{x_{i+1} - x_i}{(x_{i-1} - x_i)(x_{i-1} - x_{i+1})} \tag{A9}$$

$$A^{(1)}_{i-1,i} = A^{(1)}_{i}(x_{i-1}) = \frac{x_{i-1} - x_{i+1}}{(x_i - x_{i-1})(x_i - x_{i+1})} \tag{A10}$$

$$A^{(1)}_{i,i} = A^{(1)}_{i}(x_i) = \frac{2x_i - x_{i-1} - x_{i+1}}{(x_i - x_{i-1})(x_i - x_{i+1})} \tag{A11}$$

$$A^{(1)}_{i+1,i} = A^{(1)}_{i}(x_{i-1}) = \frac{x_{i+1} - x_{i-1}}{(x_i - x_{i-1})(x_i - x_{i+1})} \tag{A12}$$

$$A^{(1)}_{i-1,i+1} = A^{(1)}_{i+1}(x_{i-1}) = \frac{x_{i-1} - x_i}{(x_{i+1} - x_{i-1})(x_{i+1} - x_i)} \tag{A13}$$

$$A^{(1)}_{i,i+1} = A^{(1)}_{i+1}(x_i) = \frac{x_i - x_{i-1}}{(x_{i+1} - x_{i-1})(x_{i+1} - x_i)} \tag{A14}$$

$$A^{(1)}_{i+1,i+1} = A^{(1)}_{i+1}(x_{i+1}) = \frac{2x_{i+1} - x_i - x_{i-1}}{(x_{i+1} - x_{i-1})(x_{i+1} - x_i)} \tag{A15}$$

The weighting coefficients of the local quadrature rule corresponding to the second derivative can be obtained in the same way. As only three local points are involved in the quadrature rule, the coefficients are all constants:

$$A^{(2)}_{i-1}(x) = \frac{2}{(x_{i-1} - x_i)(x_{i-1} - x_{i+1})} \tag{A16}$$

$$A^{(2)}_{i}(x) = \frac{2}{(x_i - x_{i-1})(x_i - x_{i+1})} \tag{A17}$$

$$A^{(2)}_{i+1}(x) = \frac{2}{(x_{i+1} - x_{i-1})(x_{i+1} - x_i)} \tag{A18}$$

Moreover, it is easy to prove that

$$\sum_{j=i-1}^{i+1} A^{(k)}_{j}(x) = \begin{cases} 1 & (k = 0) \\ 0 & (k \geqslant 1) \end{cases} \tag{A19}$$

DOI: 10.1002/cnm

REFERENCES

1. Bellman RE, Kashef BG, Casti J. Differential quadrature: a technique for rapid solution of nonlinear partial differential equations. *Journal of Computational Physics* 1972; **10**:40–52.
2. Bert CW, Malik M. Differential quadrature method in computational mechanics: a review. *Applied Mechanics Review* 1996; **49**:1–27.
3. Civan F. Rapid and accurate solution of reactor models by the quadrature method. *Computers in Chemical Engineering* 1994; **18**:1005–1009.
4. Tomasiello S. Differential quadrature method: application to initial-boundary-value problems. *Journal of Sound and Vibration* 1998; **218**:573–585.
5. Malik M, Civan F. Comparative study of differential quadrature and cubature methods vis-à-vis some conventional techniques in context of convection–diffusion–reaction problems. *Chemical Engineering Science* 1995; **50**: 531–547.
6. Jang SK, Bert CW, Striz AG. Application of differential quadrature to static analysis of structural components. *International Journal for Numerical Methods in Engineering* 1989; **28**:561–577.
7. Shu C, Richards BE. Application of generalized differential quadrature to solve two dimensional incompressible Navier–Stokes equations. *International Journal of Numerical Methods in Fluids* 1992; **15**:791–798.
8. Bert CW, Malik M. The differential quadrature method for irregular domains and application to plate vibration. *International Journal of Mechanical Sciences* 1996; **38**(6):589–606.
9. Han JB, Liew KM. An eight-node curvilinear differential quadrature formulation for Reissner/Mindlin plates. *Computer Methods in Applied Mechanics and Engineering* 1997; **141**:265–280.
10. Karami G, Malekzadeh P. An efficient differential quadrature methodology for free vibration analysis of arbitrary straight-sided quadrilateral thin plates. *Journal of Sound and Vibration* 2003; **263**:415–442.
11. Chen CN. The development of irregular elements for differential quadrature element method steady-state heat conduction analysis. *Computer Methods in Applied Mechanics and Engineering* 1999; **170**:1–14.
12. Chen WL, Striz AG, Bert CW. High-accuracy plane stress and plate elements in the quadrature element method. *International Journal of Solids and Structures* 2000; **37**:627–647.
13. Ma H, Qin QH. Boundary integral equation supported differential quadrature method to solve problems over irregular geometries. *Computational Mechanics* 2005; **36**(1):21–33.
14. Liew KM, Huang YQ, Reddy JN. Moving least squares differential quadrature method and its application to the analysis of shear deformable plates. *International Journal for Numerical Methods in Engineering* 2003; **56**:2331–2351.
15. Liew KM, Huang YQ, Reddy JN. Analysis of general shaped thin plates by the moving least-squares differential quadrature method. *Finite Elements in Analysis and Design* 2004; **40**:1453–1474.
16. Sun JA, Zhu ZY. Upwind local differential quadrature method for solving incompressible viscous flow. *Computer Methods in Applied Mechanics and Engineering* 2000; **188**:495–504.
17. Zong Z, Lam Y. A localized differential quadrature (LDQ) method and its application to the 2D wave equation. *Computational Mechanics* 2002; **29**:382–391.
18. Wu L, Shu C. Development of RBF-DQ method for approximation and its application to simulate natural convection in concentric annuli. *Computational Mechanics* 2002; **29**:477–485.
19. Shu C, Ding H, Yeo KS. 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* 2003; **192**:941–954.
20. Shu C, Ding H, Yeo KS. Solution of partial differential equations by a global radial basis function-based differential quadrature method. *Engineering Analysis with Boundary Elements* 2004; **28**:1217–1226.
21. Shu C, Ding H, Chen HQ, Wang TQ. An upwind local RBF-DQ method for simulation of inviscid compressible flows. *Computer Methods in Applied Mechanics and Engineering* 2005; **194**:2001–2017.
22. Gavete L, Gavete KL, Benito JJ. Improvements of generalized finite difference method and comparison with other meshless method. *Applied Mathematical Modelling* 2003; **27**:831–847.
23. Liu GL, Li XW. Mesh free method on local Cartesian frame. *Applied Mathematics and Mechanics* 2006; **27**(1):1–5.

DOI: 10.1002/cnm