ELSEVIER

International Journal of Heat and Fluid Flow 19 (1998) 59–68

International Journal of HEAT AND FLUID FLOW

Comparison of two approaches for implementing stream function boundary conditions in DQ simulation of natural convection in a square cavity

C. Shu *, H. Xue

Department of Mechanical and Production Engineering, National University of Singapore, 10 Kent Ridge Crescent, Singapore 119260, Singapore

Received 14 March 1997; accepted 5 September 1997

Abstract

In this paper, the global method of generalized differential quadrature (GDQ) is applied to simulate the natural convection in a square cavity. The vorticity–stream function equation is taken as the governing equation. There are two boundary conditions (Dirichlet type and Neumann type) for the stream function at each boundary. Two approaches are introduced to implement these two boundary conditions. The first approach converts the two boundary conditions into a two-layer condition. For the second approach, the Neumann condition is built into the GDQ weighting coefficient matrices which are then used to discretize the stream function equation. Thus, the Neumann condition is exactly satisfied in the second approach. The performances of two approaches such as the accuracy and efficiency are comparatively studied in this work.

Keywords: Generalized differential quadrature; Vorticity–stream function; Incompressible; Natural convection; Boundary conditions; Weighting coefficients

Notation

$c_{ij}^{(n)}$	weighting coefficients of the nth order derivative with respect to x
$\bar{c}_{ij}^{(m)}$	weighting coefficients of the mth order derivative with respect to y
$f_x^{(n)}$	nth order derivative of function f with respect to x
$f_y^{(m)}$	mth order derivative of function f with respect to y
M	number of mesh points in y direction
N	number of mesh points in x direction
$\overline{\mathrm{Nu}}$	the average Nusselt number throughout the cavity
$\mathrm{Nu}_{1/2}$	the average Nusselt number on the vertical mid-plane of the cavity
Nu_0	the average Nusselt number on the vertical boundary of the cavity at $x=0$
$\mathrm{Nu}_{\max}$	the maximum value of the local Nusselt number on the boundary at $x=0$
$\mathrm{Nu}_{\min}$	the minimum value of the local Nusselt number on the boundary at $x=0$
Pr	Prandtl number
Ra	Rayleigh number
x_i	x coordinate
y_j	y coordinate
T	temperature
u, v	velocity components in x, y directions
$u_{\max}$	the maximum horizontal velocity on the vertical mid-plane of the cavity
$v_{\max}$	the maximum vertical velocity on the horizontal mid-plane of the cavity
ω	vorticity
ψ	stream function
$\lvert\psi_{\mathrm{mid}}\rvert$	the stream function at the mid-point of the cavity
$\lvert\psi\rvert_{\max}$	the maximum absolute value of the stream function

1. Introduction

Most numerical simulations of engineering problems can be currently carried out by conventional low order finite differences and finite elements using a large number of grid points. However, in some practical applications, the numerical solutions of partial differential equations are required at only a few specified points in the physical domain. For acceptable accuracy, the conventional low order methods still need to use a large number of grid points to obtain accurate solutions at these specified points. In seeking a more efficient method using just a few grid points to obtain accurate numerical results, the technique of differential quadrature (DQ) was proposed by Bellman et al. [1]. The DQ method follows the concept of classical integral quadrature. DQ approximates a spatial derivative of a function with respect to a coordinate at a discrete point as a weighted linear sum of all the functional values in the whole domain of that coordinate direction. The key to DQ is to determine the weighting coefficients for any order derivative

* Corresponding author. E-mail: mpeshuc@leonis.nus.sg.

0142-727X/98/$19.00
PII: S0142-727X(97)10010-8

discretization. Bellman et al. [1] suggested two methods to determine the weighting coefficients of the first order derivative. The first method solves an algebraic equation system. The second uses a simple algebraic formulation, but with the coordinates of grid points chosen as the roots of the shifted Legendre polynomial. Most previous applications of DQ in engineering [1–5] used Bellman's first method to obtain the weighting coefficients because it lets the coordinates of grid points be chosen arbitrarily. Unfortunately, when the order of algebraic equation system is large, its matrix is ill-conditioned. Thus, it is very difficult to obtain the weighting coefficients for a large number of grid points using this method. To overcome the drawbacks of DQ method in computing the weighting coefficients, Shu [6] presented the generalized differential quadrature (GDQ), in which all the current methods for determination of weighting coefficients are generalized under the analysis of a high order polynomial approximation and the analysis of a linear vector space. GDQ uses two sets instead of one set of base polynomials in a polynomial vector space. As a result, it computes the weighting coefficients of the first order derivative by a simple algebraic formulation without any restriction on choice of grid points, and the weighting coefficients of the second and higher order derivatives by a recurrence relationship. The major advantage of GDQ over DQ is its ease for the computation of weighting coefficients without any restriction on the choice of grid points. Currently, the DQ-type methods have been increasingly applied to solve incompressible flow problems [6–13] and structural and vibration problems [14–28].

Like some other numerical methods, the GDQ method discretizes the spatial derivatives and, therefore, reduces the partial differential equations into a set of algebraic equations. To solve these equations, the boundary conditions have to be implemented appropriately. For the case where there is only one boundary condition at each boundary, the implementation is very simple and can be done in a straightforward way. One just needs to replace the discretized governing equations by the boundary conditions at all the boundary points. However, in some cases, there are more than one boundary conditions at each boundary, which could result in difficulties in the numerical implementation of the boundary conditions. One example is the solution of two-dimensional incompressible Navier Stokes equations. Although the governing equations are second order partial differential equations, there are two boundary conditions for the stream function at each boundary. These two boundary conditions are derived from the boundary condition for two velocity components. Among them, one is of a Dirichlet type and the other is a Neumann type. To implement these two boundary conditions accurately, Shu et al. [8] proposed an approach which converts the two boundary conditions into two-layer numerical boundary conditions. Very accurate numerical solutions have been obtained by this approach. Another example is the flexural vibration analysis of a thin beam or a plate. For this case, the governing equation is a fourth order differential equation, which requires two boundary conditions at each boundary. To apply the DQ method to obtain accurate numerical solution of this problem, Wang and Bert [15] proposed an efficient approach to implement two boundary conditions at each boundary. In this approach, the derivative boundary conditions are built into the weighting coefficient matrices in the DQ discretization. Then, only Dirichlet boundary conditions are implemented. This approach has been successfully applied to solve some beam and plate problems with very good accuracy. However, as indicated by Wang and Bert [15], there are some limitations to the application of this approach. One limitation is in the implementation of the clamped–clamped (C–C) type boundary conditions. It was found that the implementation of the C–C type boundary conditions by this approach may lead to some wrong numerical results.

Although the performance of the approach proposed by Wang and Bert [15] is well studied in the vibration analysis, its performance in the numerical computation of incompressible Navier–Stokes equations is still unknown. In this paper, a comparative study of this approach and the approach proposed by Shu et al. [8] will be made through their application to simulate the natural convection in a square cavity.

2. Generalized differential quadrature

The GDQ approach was developed by Shu [6] based on the DQ technique [1]. It approximates the spatial derivative of a function with respect to a space coordinate at a given grid point as a weighted linear sum of all the functional values at all grid points in the whole domain of that space coordinate. The computation of weighting coefficients by GDQ is based on the analysis of a high order polynomial approximation and the analysis of a linear vector space. The weighting coefficients of the first order derivative are calculated by a simple algebraic formulation, and the weighting coefficients of the second and higher order derivatives are given by a recurrence relationship. The details of GDQ method can be found in [6]. Some two-dimensional results are described as follows. For a smooth function $f(x, y)$, GDQ discretizes its nth order derivative with respect to x, and the mth order derivative with respect to y, at the grid point (x_i, y_j) as

$$f_x^{(n)}(x_i, y_j) = \sum_{k=1}^{N} c_{ik}^{(n)} f(x_k, y_j), \quad n = 1, 2, \ldots, N-1. \tag{1a}$$

$$f_y^{(m)}(x_i, y_j) = \sum_{k=1}^{M} \bar{c}_{jk}^{(m)} f(x_i, y_k), \quad m = 1, 2, \ldots, M-1 \tag{1b}$$

for $i = 1, 2, \ldots, N$; $j = 1, 2, \ldots, M$.

where N, M are the number of grid points in the x and y direction respectively, $c_{ik}^{(n)}, \bar{c}_{jk}^{(m)}$ are the weighting coefficients to be determined as follows,

Weighting coefficients for the first order derivative

$$c_{ij}^{(1)} = \frac{A^{(1)}(x_i)}{(x_i - x_j)A^{(1)}(x_j)}, \quad i, j = 1, 2, \ldots, N, \text{ but } j \neq i, \tag{2a}$$

$$\bar{c}_{ij}^{(1)} = \frac{B^{(1)}(y_i)}{(y_i - y_j)B^{(1)}(y_j)}, \quad i, j = 1, 2, \ldots, M, \text{ but } j \neq i, \tag{2b}$$

where

$$A^{(1)}(x_i) = \prod_{j=1, j\neq i}^{N} (x_i - x_j), \qquad B^{(1)}(y_i) = \prod_{j=1, j\neq i}^{M} (y_i - y_j).$$

Weighting coefficients for the second and higher order derivatives

$$c_{ij}^{(n)} = n\left(c_{ii}^{(n-1)} c_{ij}^{(1)} - \frac{c_{ij}^{(n-1)}}{x_i - x_j}\right) \tag{3a}$$

for $i, j = 1, 2, \ldots, N$, but $j \neq i$, $n = 2, 3, \ldots, N-1$.

$$\bar{c}_{ij}^{(m)} = m\left(\bar{c}_{ii}^{(m-1)} \bar{c}_{ij}^{(1)} - \frac{\bar{c}_{ij}^{(m-1)}}{y_i - y_j}\right) \tag{3b}$$

for $i, j = 1, 2, \ldots, M$, but $j \neq i$, $m = 2, 3, \ldots, M-1$.

When $j = i$, the weighting coefficients are given by

$$c_{ii}^{(n)} = -\sum_{j=1, j\neq i}^{N} c_{ij}^{(n)}, \quad i = 1, 2, \ldots, N, \quad n = 1, 2, \ldots, N-1. \tag{4a}$$

$$\bar{c}_{ii}^{(m)} = -\sum_{j=1, j\neq i}^{M} \bar{c}_{ij}^{(m)}, \quad i = 1, 2, \ldots, M, \quad m = 1, 2, \ldots, M-1. \tag{4b}$$

It is obvious from the above equations that the weighting coefficients of the second and higher order derivatives can be completely determined from those of the first order derivatives. When the coordinates of grid points are known, the weighting coefficients for the discretization of derivatives can be easily calculated from Eqs. (2a), (2b), (3a), (3b), (4a), (4b). Then using Eqs. (1a) and (1b), all the spatial derivatives can be discretized using a similar form. The difference for the respective derivatives is to use different weighting coefficients, which are usually computed in advance. This avails as an easily implementable scheme on the computer with greatly simplified code-editing features.

When the functional values at all grid points are obtained, it is easy to calculate the functional values in the whole computational domain with high order of accuracy in terms of the polynomial approximation, i.e.

$$f(x, y_j) = \sum_{i=1}^{N} f(x_i, y_j) r_i(x). \tag{5a}$$

$$f(x_i, y) = \sum_{j=1}^{M} f(x_i, y_j) s_j(y). \tag{5b}$$

$$f(x, y) = \sum_{i=1}^{N} \sum_{j=1}^{M} f(x_i, y_j) r_i(x) s_j(y), \tag{5c}$$

where $r_i(x)$, $s_j(y)$ are the Lagrange interpolation polynomials in the x and y direction, respectively. Eqs. (5a)–(5c) will be used to calculate the flow parameters at any specific point without losing accuracy since the GDQ approach is also based on the above approximation.

3. Governing equations and numerical discretization

The buoyancy driven flow in a square cavity with vertical sides which are differentially heated is a suitable vehicle for testing and validating numerical approaches used for a wide variety of practical problems. This problem has been extensively studied by many researchers such as illustrated in the paper of Davis and Jones [29] which outlined numerous contributed results reported at the second Conference on Numerical Methods in Thermal Problems. The problem being considered here is that of the two-dimensional flow of a Boussinesq fluid of Prandtl number 0.71 in an upright square cavity described in non-dimensional terms by $0 \leqslant x \leqslant 1$, $0 \leqslant y \leqslant 1$ with y vertically upwards. The problem definition and the boundary conditions are displayed in Fig. 1. The non-dimensional vorticity–stream function formulation and the energy transport equation are used to compute this problem, which can be written as

$$\frac{\partial \omega}{\partial t} + u\frac{\partial \omega}{\partial x} + v\frac{\partial \omega}{\partial y} = \mathrm{Pr}\left(\frac{\partial^2 \omega}{\partial x^2} + \frac{\partial^2 \omega}{\partial y^2}\right) + \mathrm{Ra}\,\mathrm{Pr}\frac{\partial T}{\partial x}, \tag{6a}$$

$$\frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} = \omega. \tag{6b}$$

$$\frac{\partial T}{\partial t} + u\frac{\partial T}{\partial x} + v\frac{\partial T}{\partial y} = \frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2}, \tag{6c}$$

where ω, ψ, T, Pr and Ra are the vorticity, stream function, temperature, Prandtl number and Rayleigh number, u, v are the components of velocity in the x and y direction, which can be calculated from the stream function

$$u = \frac{\partial \psi}{\partial y}, \quad v = -\frac{\partial \psi}{\partial x}. \tag{7}$$

Eqs. (6a)–(6c) is subjected to initial conditions

$$\omega = \psi = T = u = v = 0, \quad \text{when } t = 0 \tag{8}$$

and boundary conditions for $t > 0$,

$$\psi = \frac{\partial \psi}{\partial x} = 0, \quad T = 1, \quad \text{at } x = 0, \ 0 \leqslant y \leqslant 1, \tag{9a}$$

$$\psi = \frac{\partial \psi}{\partial x} = 0, \quad T = 0, \quad \text{at } x = 1, \ 0 \leqslant y \leqslant 1, \tag{9b}$$

$$\psi = \frac{\partial \psi}{\partial y} = \frac{\partial T}{\partial y} = 0, \quad \text{at } y = 0, 1, \ 0 < x < 1. \tag{9c}$$

It is noted that at each boundary, Eqs. (9a)–(9c) give one boundary condition for temperature T, and two boundary conditions for stream function ψ although the governing equation for ψ is just second order.

Applying the GDQ method to discretize the spatial derivatives in Eqs. (6a)–(6c) gives

$$\frac{\mathrm{d}\omega_{ij}}{\mathrm{d}t} + u_{ij}\sum_{k=1}^{N} c_{ik}^{(1)}\omega_{kj} + v_{ij}\sum_{k=1}^{M} \bar{c}_{jk}^{(1)}\omega_{ik} = \mathrm{Pr}\left[\sum_{k=1}^{N} c_{ik}^{(2)}\omega_{kj} + \sum_{k=1}^{M} \bar{c}_{jk}^{(2)}\omega_{ik}\right] + \mathrm{Ra}\,\mathrm{Pr}\sum_{k=1}^{N} c_{ik}^{(1)} T_{kj}, \tag{10a}$$

$$\sum_{k=1}^{N} c_{ik}^{(2)}\psi_{kj} + \sum_{k=1}^{M} \bar{c}_{jk}^{(2)}\psi_{ik} = \omega_{ij}. \tag{10b}$$

$$\frac{\mathrm{d}T_{ij}}{\mathrm{d}t} + u_{ij}\sum_{k=1}^{N} c_{ik}^{(1)} T_{kj} + v_{ij}\sum_{k=1}^{M} \bar{c}_{jk}^{(1)} T_{ik} = \sum_{k=1}^{N} c_{ik}^{(2)} T_{kj} + \sum_{k=1}^{M} \bar{c}_{jk}^{(2)} T_{ik} \tag{10c}$$

$$\text{for } i = 1, 2, \ldots, N, \quad j = 1, 2, \ldots, M,$$

where N, M are the number of grid points in the x and y direction respectively, $c_{ij}^{(n)}$ are the weighting coefficients of the nth order derivative of a function with respect to x, and $\bar{c}_{ij}^{(m)}$ are

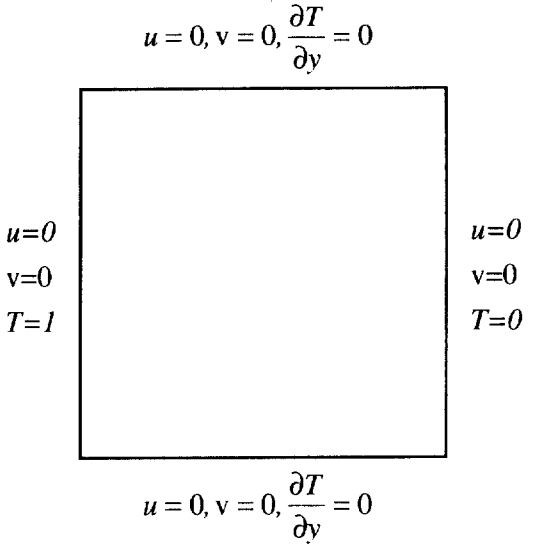


Fig. 1. Configuration of natural convection in a square cavity.

the weighting coefficients of the mth order derivative of a function with respect to y. Similarly, the derivatives in Eqs. (9a)–(9c) can be discretized by GDQ method. In numerical simulation, the boundary conditions for ω are obtained from Eq. (10b). Using Eqs. (9a)–(9c) and (10b) gives the boundary conditions for ω as

$$\omega_{1,j} = \sum_{k=1}^{N} c_{1,k}^{(2)} \psi_{k,j} \quad \text{for } j = 1, 2, \ldots, M, \tag{11a}$$

$$\omega_{N,j} = \sum_{k=1}^{N} c_{N,k}^{(2)} \psi_{k,j} \quad \text{for } j = 1, 2, \ldots, M, \tag{11b}$$

$$\omega_{i,1} = \sum_{k=1}^{M} \bar{c}_{1,k}^{(2)} \psi_{k,j} \quad \text{for } i = 2, 3, \ldots, N-1, \tag{11c}$$

$$\omega_{i,M} = \sum_{k=1}^{M} \bar{c}_{M,k}^{(2)} \psi_{k,j} \quad \text{for } i = 2, 3, \ldots, N-1. \tag{11d}$$

The boundary condition for T at $x = 0$ and $x = 1$ can be easily implemented by

$$T_{1,j} = 1, \quad T_{N,j} = 0 \quad \text{for } j = 1, 2, \ldots, M. \tag{12a}$$

The derivative conditions for T at $y = 0$ and $y = 1$ are discretized by the GDQ method and then combined to give

$$T_{i,1} = \frac{1}{AYT}\left[\sum_{k=2}^{M-1}\left(\bar{c}_{1,k}^{(1)}\bar{c}_{M,M}^{(1)} - \bar{c}_{M,k}^{(1)}\bar{c}_{1,M}^{(1)}\right) T_{i,k}\right], \tag{12b}$$

$$T_{i,M} = \frac{1}{AYT}\left[\sum_{k=2}^{M-1}\left(\bar{c}_{M,k}^{(1)}\bar{c}_{1,1}^{(1)} - \bar{c}_{1,k}^{(1)}\bar{c}_{M,1}^{(1)}\right) T_{i,k}\right] \tag{12c}$$

for $i = 2, 3, \ldots, N-1$,

where

$$AYT = \bar{c}_{M,1}^{(1)}\bar{c}_{1,M}^{(1)} - \bar{c}_{1,1}^{(1)}\bar{c}_{M,M}^{(1)}.$$

The implementation of boundary conditions for stream function will be discussed in the following section.

4. Implementation of boundary conditions for the stream function

In this section, two approaches will be used to implement the boundary condition for the stream function. As shown in Eqs. (9a)–(9c), there are two boundary conditions for ψ at each boundary. Numerically, the Dirichlet condition can be easily implemented by

$$\psi_{1,j} = 0, \quad \psi_{N,j} = 0, \quad \psi_{i,1} = 0, \quad \psi_{i,M} = 0 \tag{13}$$

for $i = 1, 2, \ldots, N$; $j = 2, 3, \ldots, M-1$.

Eq. (13) will be applied by both approaches exactly at the boundary point. The Neumann condition will be treated differently by two approaches which are shown as follows.

4.1. Two-layer condition from GDQ discretization of Neumann condition

For this approach, Eq. (13) gives the first layer condition at all boundary points, and the Neumann condition in Eqs. (9a)–(9c) is discretized by the GDQ method and then combined to provide the second layer condition. Applying the GDQ method to discretize the derivative in Neumann condition gives

$$c_{1,1}^{(1)}\psi_{1,j} + c_{1,2}^{(1)}\psi_{2,j} + \sum_{k=3}^{N-2} c_{1,k}^{(1)}\psi_{k,j} + c_{1,N-1}^{(1)}\psi_{N-1,j} + c_{1,N}^{(1)}\psi_{N,j} = 0 \tag{14a}$$

for the boundary of $x = 0$,

$$c_{N,1}^{(1)}\psi_{1,j} + c_{N,2}^{(1)}\psi_{2,j} + \sum_{k=3}^{N-2} c_{N,k}^{(1)}\psi_{k,j} + c_{N,N-1}^{(1)}\psi_{N-1,j} + c_{N,N}^{(1)}\psi_{N,j} = 0 \tag{14b}$$

for the boundary of $x = 1$,

$$\bar{c}_{1,1}^{(1)}\psi_{i,1} + c_{1,2}^{(1)}\psi_{i,2} + \sum_{k=3}^{M-2} \bar{c}_{1,k}^{(1)}\psi_{i,k} + \bar{c}_{1,M-1}^{(1)}\psi_{i,M-1} + \bar{c}_{1,M}^{(1)}\psi_{i,M} = 0 \tag{14c}$$

for the boundary of $y = 0$, and

$$\bar{c}_{M,1}^{(1)}\psi_{i,1} + \bar{c}_{M,2}^{(1)}\psi_{i,2} + \sum_{k=3}^{M-2} \bar{c}_{M,k}^{(1)}\psi_{i,k} + \bar{c}_{M,M-1}^{(1)}\psi_{i,M-1} + \bar{c}_{M,M}^{(1)}\psi_{i,M} = 0 \tag{14d}$$

for the boundary of $y = 1$.

Substituting Eq. (13) into Eqs. (14a) and (14b), we can obtain

$$\psi_{2,j} = \frac{1}{AXN}\left[\sum_{k=3}^{N-2}\left(c_{1,k}^{(1)}c_{N,N-1}^{(1)} - c_{N,k}^{(1)}c_{1,N-1}^{(1)}\right)\psi_{k,j}\right], \tag{15a}$$

$$\psi_{N-1,j} = \frac{1}{AXN}\left[\sum_{k=3}^{N-2}\left(c_{N,k}^{(1)}c_{1,2}^{(1)} - c_{1,k}^{(1)}c_{N,2}^{(1)}\right)\psi_{k,j}\right] \tag{15b}$$

for $j = 1, 2, \ldots, M$,

where

$$AXN = c_{N,2}^{(1)}c_{1,N-1}^{(1)} - c_{1,2}^{(1)}c_{N,N-1}^{(1)}.$$

Similarly, substituting Eq. (13) into Eqs. (14c) and (14d) gives

$$\psi_{i,2} = \frac{1}{AYM}\left[\sum_{k=3}^{M-2}\left(\bar{c}_{1,k}^{(1)}\bar{c}_{M,M-1}^{(1)} - \bar{c}_{M,k}^{(1)}\bar{c}_{1,M-1}^{(1)}\right)\psi_{i,k}\right], \tag{16a}$$

$$\psi_{i,M-1} = \frac{1}{AYM}\left[\sum_{k=3}^{M-2}\left(\bar{c}_{M,k}^{(1)}\bar{c}_{1,2}^{(1)} - \bar{c}_{1,k}^{(1)}\bar{c}_{M,2}^{(1)}\right)\psi_{i,k}\right] \tag{16b}$$

for $i = 2, 3, \ldots, N-1$,

where

$$AYM = \bar{c}_{M,2}^{(1)}\bar{c}_{1,M-1}^{(1)} - \bar{c}_{1,2}^{(1)}\bar{c}_{M,M-1}^{(1)}.$$

Eqs. (15a) and (15b) and Eqs. (16a) and (16b) provide the solution at a layer adjacent to the boundary. It is noted that when this approach is used, Eq. (10b) can only be applied for the interior points $3 \leqslant i \leqslant N-2$, $3 \leqslant j \leqslant M-2$. For the convenience of following discussion, this approach is termed the "two-layer approach".

4.2. One-layer condition from modified weighting coefficient matrices

For this approach, only the Dirichlet condition (Eq. (13)) is implemented at the boundary point. The Neumann condition is built into the GDQ weighting coefficient matrices, which will then be applied to discretize the stream function equation.

Consider the GDQ analog of the derivative $\partial\psi/\partial x$ at the grid points on a line y_j parallel to the x-axis

$$\begin{Bmatrix} (\partial\psi/\partial x)_{1,j} \\ (\partial\psi/\partial x)_{2,j} \\ \cdot \\ \cdot \\ \cdot \\ (\partial\psi/\partial x)_{N-1,j} \\ (\partial\psi/\partial x)_{N,j} \end{Bmatrix} = \begin{bmatrix} c_{1,1}^{(1)} & c_{1,2}^{(1)} & \cdots & c_{1,N-1}^{(1)} & c_{1,N}^{(1)} \\ c_{2,1}^{(1)} & c_{2,2}^{(1)} & \cdots & c_{2,N-1}^{(1)} & c_{2,N}^{(1)} \\ \cdot & \cdot & \cdots & \cdot & \cdot \\ \cdot & \cdot & \cdots & \cdot & \cdot \\ \cdot & \cdot & \cdots & \cdot & \cdot \\ c_{N-1,1}^{(1)} & c_{N-1,2}^{(1)} & \cdots & c_{N-1,N-1}^{(1)} & c_{N-1,N}^{(1)} \\ c_{N,1}^{(1)} & c_{N,2}^{(1)} & \cdots & c_{N,N-1}^{(1)} & c_{N,N}^{(1)} \end{bmatrix} \begin{Bmatrix} \psi_{1,j} \\ \psi_{2,j} \\ \cdot \\ \cdot \\ \cdot \\ \psi_{N-1,j} \\ \psi_{N,j} \end{Bmatrix}. \tag{17}$$

To satisfy the derivative condition of ψ (Eqs. (9a) and (9b)), Eq. (17) can be modified by zeroing the first and last rows of the matrix

$$\begin{Bmatrix} (\partial\psi/\partial x)_{1,j} \\ (\partial\psi/\partial x)_{2,j} \\ \cdot \\ \cdot \\ \cdot \\ (\partial\psi/\partial x)_{N-1,j} \\ (\partial\psi/\partial x)_{N,j} \end{Bmatrix} = \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ c_{2,1}^{(1)} & c_{2,2}^{(1)} & \cdots & c_{2,N-1}^{(1)} & c_{2,N}^{(1)} \\ \cdot & \cdot & \cdots & \cdot & \cdot \\ \cdot & \cdot & \cdots & \cdot & \cdot \\ \cdot & \cdot & \cdots & \cdot & \cdot \\ c_{N-1,1}^{(1)} & c_{N-1,2}^{(1)} & \cdots & c_{N-1,N-1}^{(1)} & c_{N-1,N}^{(1)} \\ 0 & 0 & \cdots & 0 & 0 \end{bmatrix} \begin{Bmatrix} \psi_{1,j} \\ \psi_{2,j} \\ \cdot \\ \cdot \\ \cdot \\ \psi_{N-1,j} \\ \psi_{N,j} \end{Bmatrix}. \tag{18}$$

The modified matrix in Eq. (18) is noted as $[\tilde{A}^1]$ while the original matrix in Eq. (17) is represented by $[A^1]$. The matrix form of Eq. (18) can be written as

$$\{\partial\psi/\partial x\}_j = [\tilde{A}^1]\{\psi\}_j. \tag{19}$$

Similarly, the matrix form of GDQ analog of the derivative $\partial^2\psi/\partial x^2$ at the grid points on a line y_j parallel to the x-axis can be written as

$$\{\partial^2\psi/\partial x^2\}_j = [\tilde{A}^2]\{\psi\}_j, \tag{20}$$

where $[\tilde{A}^2]$ is the modified GDQ weighting coefficient matrix for the second order derivative $\partial^2\psi/\partial x^2$. On the other hand, we note that the second order derivative $\partial^2\psi/\partial x^2$ can be obtained by differentiating the first order derivative $\partial\psi/\partial x$. Thus, the matrix $[\tilde{A}^2]$ can be computed by

$$[\tilde{A}^2] = [A^1][\tilde{A}^1]. \tag{21}$$

The above process can also be applied to modify the GDQ weighting coefficient matrices in the y direction. Let $[B^1]$, $[B^2]$ be the original GDQ weighting coefficient matrices related to $\partial\psi/\partial y$ and $\partial^2\psi/\partial x^2$, $[\tilde{B}^1]$, $[\tilde{B}^2]$ be the corresponding modified weighting coefficient matrices. Similar to Eq. (21), we have

$$[\tilde{B}^2] = [B^1][\tilde{B}^1], \tag{22}$$

where

$$[B^1] = \begin{bmatrix} \overline{c}_{1,1}^{(1)} & \overline{c}_{1,2}^{(1)} & \cdots & \overline{c}_{1,M-1}^{(1)} & \overline{c}_{1,M}^{(1)} \\ \overline{c}_{2,1}^{(1)} & \overline{c}_{2,2}^{(1)} & \cdots & \overline{c}_{2,M-1}^{(1)} & \overline{c}_{2,M}^{(1)} \\ \cdot & \cdot & \cdots & \cdot & \cdot \\ \cdot & \cdot & \cdots & \cdot & \cdot \\ \cdot & \cdot & \cdots & \cdot & \cdot \\ \overline{c}_{M-1,1}^{(1)} & \overline{c}_{M-1,2}^{(1)} & \cdots & \overline{c}_{M-1,M-1}^{(1)} & \overline{c}_{M-1,M}^{(1)} \\ \overline{c}_{M,1}^{(1)} & \overline{c}_{M,2}^{(1)} & \cdots & \overline{c}_{M,M-1}^{(1)} & \overline{c}_{M,M}^{(1)} \end{bmatrix},$$

$$[\tilde{B}^1] = \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ \overline{c}_{2,1}^{(1)} & \overline{c}_{2,2}^{(1)} & \cdots & \overline{c}_{2,M-1}^{(1)} & \overline{c}_{2,M}^{(1)} \\ \cdot & \cdot & \cdots & \cdot & \cdot \\ \cdot & \cdot & \cdots & \cdot & \cdot \\ \cdot & \cdot & \cdots & \cdot & \cdot \\ \overline{c}_{M-1,1}^{(1)} & \overline{c}_{M-1,2}^{(1)} & \cdots & \overline{c}_{M-1,M-1}^{(1)} & \overline{c}_{M-1,M}^{(1)} \\ 0 & 0 & \cdots & 0 & 0 \end{bmatrix}.$$

It is noted that the modified GDQ weighting coefficient matrices $[\tilde{A}^2]$ and $[\tilde{B}^2]$ are only applied to Eq. (10b). For other equations, the original GDQ weighting coefficient matrices should be employed. Similar to the finite element method, when this approach is applied, the derivative (Neumann) condition is automatically and exactly satisfied. And since only the Dirichlet condition is implemented at the boundary point, Eq. (10b) has to be applied for the interior points, $2 \leqslant i \leqslant N-1$, $2 \leqslant j \leqslant M-1$. For the convenience of following discussion, this approach is called the "one-layer approach".

5. Results and discussion

In this section, two approaches presented in the preceding section will be used to simulate the natural convection in a square cavity. As shown in Eqs. (11a)–(11d) and Eqs. (12a)–(12c), there is only one boundary condition at each boundary point for ω and T. Thus, Eqs. (10a) and (10b) should be applied at the interior points, $2 \leqslant i \leqslant N-1$, $2 \leqslant j \leqslant M-1$. In the present study, the set of $(N-2) \times (M-2)$ ordinary differential equations for ω and T are solved by the 4-stage Runge–Kutta scheme. Eq. (10b) gives an algebraic equation system which is solved in this work by a direct method of LU decomposition. It should be indicated that the matrix size of Eq. (10b) for the one-layer approach and two-layer approach is different. For the one-layer approach, the matrix size is $(N-2) \times (M-2)$ by $(N-2) \times (M-2)$ while for the two-layer approach, the matrix size is $(N-4) \times (M-4)$ by $(N-4) \times (M-4)$. It is noted that the Laplacian operator in Eq. (10b) is a linear operator. Thus, when LU decomposition is used to solve Eq. (10b), we only need to decompose the matrix of the equation system once and store the inverted matrix elements for all the following computations.

For the GDQ simulation, the coordinates of grid points are chosen as [8]

$$x_i = \frac{1}{2}\left[1 - \cos\left(\frac{i-1}{N-1}\pi\right)\right], \quad i = 1, 2, \ldots, N. \tag{23}$$

$$y_j = \frac{1}{2}\left[1 - \cos\left(\frac{j-1}{M-1}\pi\right)\right], \quad j = 1, 2, \ldots, M. \tag{24}$$

In order to compare the performance of two approaches for the implementation of boundary conditions for stream function, the following quantities are calculated respectively:

$\lvert\psi_{mid}\rvert$	the stream function at the mid-point of the cavity
$\lvert\psi_{max}\rvert$	the maximum absolute value of the stream function (together with its location)
u_{max}	the maximum horizontal velocity on the vertical mid-plane of the cavity (together with its location)
v_{max}	the maximum vertical velocity on the horizontal mid-plane of the cavity (together with its location)
$\overline{Nu}$	the average Nusselt number throughout the cavity
$Nu_{1/2}$	the average Nusselt number on the vertical mid-plane of the cavity
Nu_0	the average Nusselt number on the vertical boundary of the cavity at $x=0$
Nu_{max}	the maximum value of the local Nusselt number on the boundary at $x=0$ (together with its location)
Nu_{min}	the minimum value of the local Nusselt number on the boundary at $x=0$ (together with its location)

Since it is very hard to get an analytical expression for the solution, the values of above variables are usually determined from the numerical solution defined at the grid points. Obviously, to obtain these variables accurately, the mesh size should be very large. However, as experienced in our previous work [6–13], the main feature of the GDQ method is that it can obtain accurate numerical solution using a very coarse mesh. Although the GDQ solution at the coarse mesh point is very accurate, it cannot give the accurate values of above variables, especially for determination of maximum and minimum values. To improve this, an interpolation process is introduced in this study. When the converged solution on a coarse mesh is obtained, the functional values on a fine mesh of 101×101 are given from Eqs. (5a)–(5c). It is noted that the use of Eqs. (5a)–(5c) will not affect the accuracy of numerical solution since the GDQ discretization is also based on the Lagrange interpolation polynomials. In the following, all the results shown in the tables are based on the interpolated values.

Tables 1–4 list the numerical results of two approaches for Rayleigh numbers of 10^3, 10^4, 10^5, 10^6, respectively. Also included in these tables is the bench mark solution given by Davis [30]. It is indicated that the bench mark solution of Davis [30] is obtained by using an extrapolation strategy from finite difference results of different mesh sizes. Thus it can be expected that the bench mark solution is more accurate than the single-mesh finite difference results. It is noted that for each case of computation, four mesh sizes are used to obtain GDQ results. It can be seen from the tables that the convergence trend of both the one-layer results and the two-layer results is very apparent. As the mesh size increases, the accuracy of both numerical results is improved. It was found that, as compared to the bench mark solution, accurate numerical results of all the cases can be obtained by two approaches using a relatively small number of mesh size. This can be attributed to the feature of the GDQ method. Since the GDQ method is a global method, the order of truncation error in its discretization is very high. That is, the number k in the truncation error expression of $O[(\Delta s)^k]$ is very large. Thus, to limit the truncation error to a given criterion, the spacing of mesh points Δs can be relatively large since k is large. In other words, a small mesh size can provide a high order of accuracy for numerical solution.

It was also found that the one-layer approach performs very well in the numerical simulation of natural convection problems. Unlike the case of vibration analysis with clamped–clamped boundary conditions, no spurious solution was found in this study when the one-layer approach is applied. Actually, it can be observed from Tables 1–4 that for all the cases, although the one-layer and two-layer results agree very well with the bench mark solution, the one-layer approach gives better accuracy than the two-layer approach. This phenomenon can be analyzed as follows. The error of numerical results to the true solution of a partial differential equation is due to the truncated error arising from the numerical discretization of derivatives in the partial differential equation and the boundary conditions. In the one-layer approach, the derivative conditions are built into the weighting coefficient matrices, which are exactly satisfied without any numerical discretization while in the two-layer approach, the derivatives in both the partial differential equation and the boundary conditions are discretized by GDQ with high order of accuracy. So, we can see that the numerical errors in the one-layer approach are only contributed by the discretization of the derivatives in the partial differential equation while the numerical errors in the two-layer approach are contributed by the discretization of derivatives in both the partial differential equation and the boundary conditions. It can be expected that the one-layer approach provides less numerical errors than the two-layer approach. That is probably the reason that the one-layer approach gives more accurate numerical results. However, the improvement of accuracy by the one-layer approach cannot be seen in the flow patterns near the boundary. Both approaches give the same flow patterns. The isotherms and streamlines of $Ra = 10^3, 10^4, 10^5, 10^6$ are shown in Figs. 2 and 3.

Although the one-layer approach gives better accuracy than the two-layer approach, it does come with a price. In the present study, the unsteady Navier Stokes equations are solved

Table 1
Comparison of numerical results for $Ra = 10^3$

	Two-layer approach				One-layer approach				Davis [30]
Mesh	7×7	9×9	11×11	13×13	7×7	9×9	11×11	13×13	
$\lvert\psi_{mid}\rvert$	1.180	1.173	1.173	1.175	1.176	1.174	1.174	1.175	1.174
u_{max}	3.669	3.660	3.642	3.649	3.652	3.650	3.646	3.649	3.649
y	0.810	0.815	0.815	0.815	0.815	0.815	0.815	0.815	0.813
v_{max}	3.710	3.690	3.690	3.698	3.700	3.694	3.694	3.698	3.697
x	0.180	0.180	0.180	0.180	0.180	0.180	0.180	0.180	0.178
$\overline{Nu}$	1.123	1.118	0.118	0.118	1.123	0.118	0.118	0.118	0.118
$Nu_{1/2}$	1.121	1.117	0.117	0.118	1.120	0.117	0.117	0.118	0.118
Nu_0	1.100	1.117	0.117	0.118	1.100	0.117	0.117	0.118	0.117
Nu_{max}	1.495	1.505	1.505	1.506	1.515	1.507	1.506	1.506	1.505
y	0.085	0.085	0.085	0.090	0.090	0.090	0.090	0.090	0.092
Nu_{min}	0.683	0.690	0.690	0.691	0.684	0.690	0.690	0.691	0.692
y	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000

Table 2
Comparison of numerical results for $Ra = 10^4$

	Two-layer approach				One-layer approach				Davis [30]
Mesh	9 × 9	11 × 11	13 × 13	15 × 15	9 × 9	11 × 11	13 × 13	15 × 15	
$\lvert\psi_{mid}\rvert$	5.022	5.055	5.077	5.075	5.042	5.064	5.075	5.075	5.071
u_{max}	15.964	16.130	16.189	16.190	16.130	16.155	16.181	16.190	16.178
y	0.820	0.825	0.825	0.825	0.825	0.825	0.825	0.825	0.823
v_{max}	18.947	19.462	19.668	19.638	19.462	19.530	19.626	19.627	19.617
x	0.115	0.120	0.120	0.120	0.120	0.120	0.120	0.120	0.119
$\overline{Nu}$	2.216	2.236	2.249	2.245	2.210	2.239	2.248	2.245	2.243
$Nu_{1/2}$	2.207	2.246	2.250	2.245	2.119	2.248	2.249	2.244	2.243
Nu_0	2.075	2.205	2.262	2.248	2.052	2.206	2.261	2.248	2.238
Nu_{max}	3.185	3.449	3.571	3.543	3.287	3.423	3.571	3.543	3.528
y	0.095	0.155	0.150	0.145	0.100	0.160	0.150	0.145	0.143
Nu_{min}	0.652	0.567	0.575	0.586	0.673	0.570	0.575	0.586	0.586
y	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000

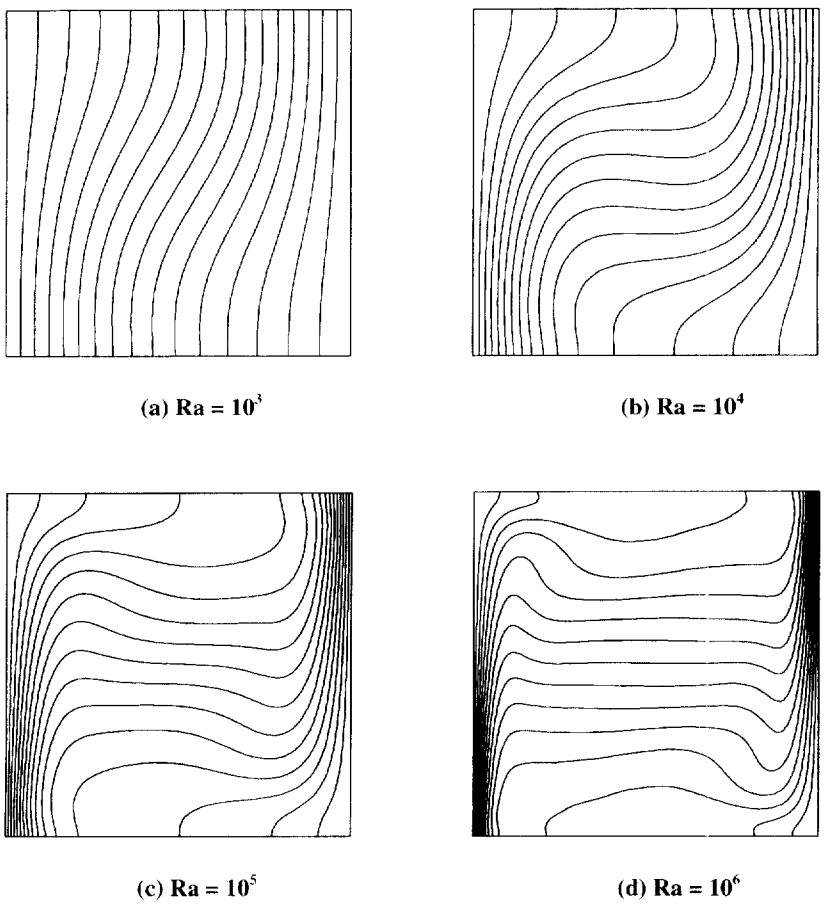
Table 3
Comparison of numerical results for $Ra = 10^5$

	Two-layer approach				One-layer approach				Davis [30]
Mesh	17 × 17	19 × 17	21 × 17	21 × 19	17 × 17	19 × 17	21 × 17	21 × 19	
$\lvert\psi_{mid}\rvert$	9.114	9.123	9.116	9.117	9.113	9.119	9.115	9.116	9.111
$\lvert\psi_{max}\rvert$	9.618	9.626	9.617	9.618	9.614	9.621	9.616	9.617	9.612
x	0.285	0.285	0.285	0.285	0.285	0.285	0.285	0.285	0.285
y	0.600	0.600	0.600	0.600	0.600	0.600	0.600	0.600	0.600
u_{max}	34.701	34.754	34.730	34.736	34.689	34.727	34.710	34.730	34.730
y	0.855	0.855	0.855	0.855	0.855	0.855	0.855	0.855	0.855
v_{max}	68.956	68.756	68.640	68.640	68.670	68.670	68.640	68.640	68.590
x	0.065	0.065	0.065	0.065	0.065	0.065	0.065	0.065	0.066
$\overline{Nu}$	4.538	4.529	4.527	4.523	4.533	4.529	4.526	4.523	4.519
$Nu_{1/2}$	4.537	4.535	4.530	4.524	4.533	4.532	4.530	4.524	4.519
Nu_0	4.566	4.553	4.528	4.527	4.562	4.551	4.527	4.527	4.509
Nu_{max}	7.761	7.859	7.790	7.788	7.760	7.865	7.790	7.789	7.717
y	0.095	0.085	0.080	0.080	0.090	0.090	0.080	0.080	0.081
Nu_{min}	0.685	0.705	0.715	0.725	0.685	0.704	0.715	0.725	0.729
y	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000

Table 4
Comparison of numerical results for $Ra = 10^6$

	Two-layer approach				One-layer approach				Davis [30]
Mesh	17 × 17	19 × 17	21 × 17	21 × 19	17 × 17	19 × 17	21 × 17	21 × 19	
$\lvert\psi_{mid}\rvert$	15.960	16.070	16.240	16.270	15.920	16.080	16.240	16.280	16.320
$\lvert\psi_{max}\rvert$	16.356	16.478	16.667	16.714	16.344	16.492	16.663	16.714	16.750
x	0.150	0.150	0.150	0.150	0.150	0.150	0.150	0.150	0.151
y	0.545	0.545	0.545	0.550	0.545	0.545	0.545	0.550	0.547
u_{max}	64.465	63.900	64.150	64.775	64.250	64.060	64.378	64.891	64.630
y	0.850	0.845	0.845	0.850	0.850	0.850	0.845	0.850	0.850
v_{max}	208.92	216.40	220.62	220.64	213.13	216.63	219.19	219.20	219.36
x	0.040	0.035	0.035	0.035	0.040	0.035	0.035	0.035	0.038
$\overline{Nu}$	8.638	8.713	8.794	8.762	8.557	8.699	8.795	8.759	8.800
$Nu_{1/2}$	8.676	8.745	8.797	8.727	8.582	8.713	8.782	8.716	8.799
Nu_0	8.092	8.466	8.778	8.721	8.195	8.367	8.754	8.722	8.817
Nu_{max}	15.441	15.810	16.300	16.070	17.520	15.740	15.890	15.961	17.925
y	0.040	0.030	0.030	0.040	0.040	0.030	0.030	0.030	0.038
Nu_{min}	1.548	1.194	0.876	1.019	1.665	1.274	0.992	1.058	0.989
y	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000

and the numerical computation is marched in the time direction until a steady state resolution is obtained. Obviously, the larger the time step size, the faster the convergence rate. The allowable time step size for marching in the time direction should satisfy the stability condition. From Eqs. (10a) and (10b), it is very difficult to find a criterion from the stability



(a) Ra = 10^3 (b) Ra = 10^4

(c) Ra = 10^5 (d) Ra = 10^6

Fig. 2. Isotherms of Ra = 10^3, 10^4, 10^5, 10^6.

analysis for the choice of time step size. In this study, the optimum (maximum) time step size was found through a trial-and-error process. It is interesting to see that the optimum time step size for the one-layer approach is quite different from that for the two-layer approach. It was found that for all the cases, the allowable maximum time step size for the two-layer approach is almost twice as large as that for the one-layer approach. As a result, the iteration number required for a converged solution by the two-layer approach is about half of that by the one-layer approach. This is a very interesting phenomenon. Although the two approaches are only applied differently for implementing boundary conditions of stream function, they do have some effect on the stability condition of vorticity and temperature equations. Obviously, this effect is through the velocity field. Table 5 lists the allowable time step size, iteration number and the CPU time (s) on the DEC Alpha workstation for Ra = 10^4 with three mesh sizes. The iteration number is obtained when the following criterion is satisfied:

$$\overline{\mathrm{Res}} = \sqrt{\frac{1}{(N-2)(M-2)}\sum_{i=2}^{N-1}\sum_{j=2}^{M-1}\mathrm{Res}_{i,j}^2} < 10^{-7}. \tag{25}$$

where $\mathrm{Res}_{i,j}$ represent residuals of vorticity and temperature equation, respectively. It can be observed from Table 5 that for the same mesh size, $\Delta t_{\max}$ of the one-layer approach is much less than that of the two-layer approach. As a consequence, the iteration number of the one-layer approach is much larger than that of the two-layer approach. It can also be seen from Table 5 that for the same mesh size, the ratio of iteration number for the one-layer approach over that for the two-layer approach is less than 2 (around 1.8). However, the ratio of CPU time for the one-layer approach over that for the two-layer approach is larger than 2 (around 2.2). This indicates that the operation per iteration for the one-layer approach is larger than that for the two-layer approach. The difference of operation for two approaches may be attributed to the solution of stream function equation. In this study, the stream function equation is solved directly by LU decomposition. As discussed earlier on, the matrix size of Eq. (10b) for two approaches is different. The matrix size of the one-layer approach is $(N-2)\times(M-2)$ by $(N-2)\times(M-2)$ while the matrix size of two-layer approach is just $(N-4)\times(M-4)$ by $(N-4)\times(M-4)$. Clearly, the matrix size of one-layer approach is much bigger than that of two-layer approach. Therefore, the computational effort of solving Eq. (10b) for the one-layer approach is larger than that for the two-layer approach. For other Rayleigh numbers, it was also found that the two-layer approach is more efficient than the one-layer approach. This can be observed in Table 6, in which the allowable maximum time step sizes, iteration numbers, and the CPU time (s) for Ra = 10^3, 10^4, 10^5, 10^6 are listed. Obviously, when the mesh size is fixed, the CPU time required by the one-layer approach is much larger than that required by the two-layer approach.

6. Conclusions

In this paper, two approaches are introduced to implement the boundary conditions of stream function in GDQ simulation

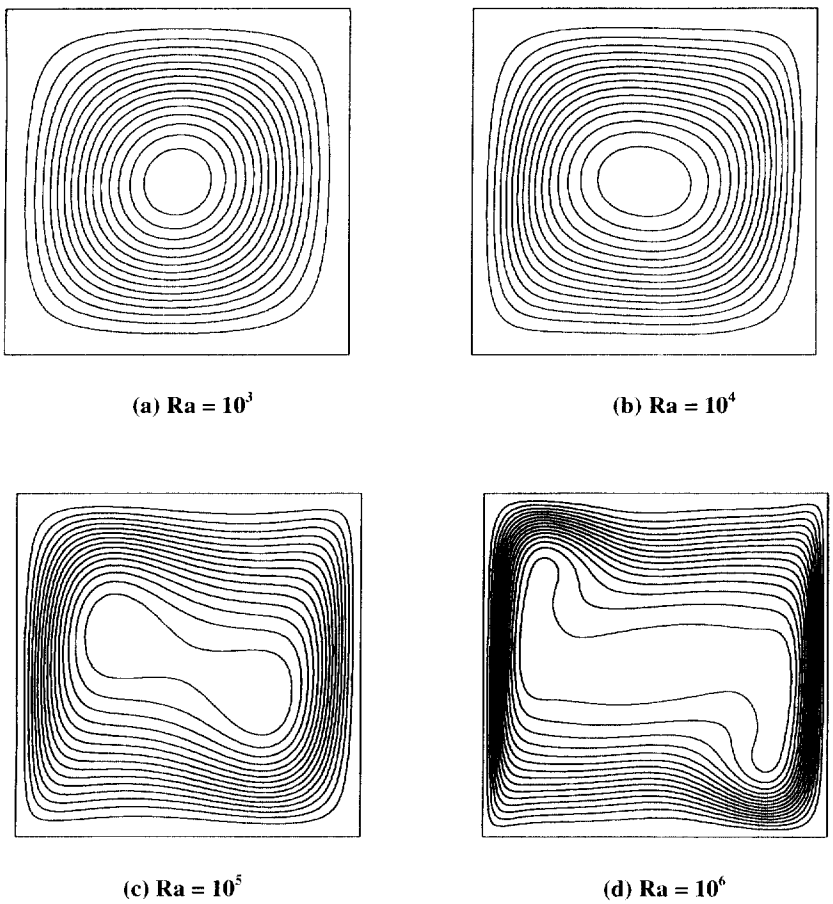


Fig. 3. Streamlines of $Ra = 10^3$, 10^4, 10^5, 10^6.

Table 5
Comparison of CPU time for $Ra = 10^4$ with different mesh sizes

	One-layer approach			Two-layer approach		
Mesh sizes	13 × 13	15 × 15	17 × 17	13 × 13	15 × 15	17 × 17
Δt_{max}	1.9×10^{-3}	1.0×10^{-3}	5.7×10^{-4}	3.6×10^{-3}	1.8×10^{-3}	1.0×10^{-3}
Iteration numbers	1516	2150	3332	869	1207	1895
CPU time (s)	17.9	49.9	125.8	8.0	22.5	50.9

Table 6
Comparison of CPU time for different Rayleigh numbers

Ra	Approaches	Meshes	Δt_{max}	Iterations	CPU time (s)
10^3	One-layer	13 × 13	1.9×10^{-3}	1492	23.066
	Two-layer	13 × 13	3.6×10^{-3}	812	8.416
10^4	One-layer	15 × 15	1.0×10^{-3}	2150	49.863
	Two-layer	15 × 15	1.8×10^{-3}	1207	21.470
10^5	One-layer	21 × 17	2.8×10^{-4}	4808	298.023
	Two-layer	21 × 17	5.3×10^{-4}	2544	105.100
10^6	One-layer	21 × 17	2.9×10^{-4}	4764	284.197
	Two-layer	21 × 17	5.3×10^{-4}	2549	103.800

of natural convection in a square cavity. It was found in this study that the one-layer approach gives more accurate numerical results than the two-layer approach. This is because for the one-layer approach, the derivative condition is exactly satisfied in the GDQ discretization while for the two-layer approach, the derivative condition is approximated by the GDQ method with high order of accuracy. Although the one-layer approach can provide better accuracy of numerical results, it is less

efficient than the two-layer approach. It was found that for all the cases, when the mesh size is fixed, the iteration number required for a converged solution by the one-layer approach is much larger than that by the two-layer approach. In addition, the computational effort per iteration for the one-layer approach is bigger than that for the two-layer approach. Since the accuracy of GDQ results can be greatly improved by slightly increasing the number of grid points, it can be concluded that the two-layer approach is more efficient than the one-layer approach.

References

[1] R. Bellman, B.G. Kashef, J. Casti, Differential quadrature: a technique for the rapid solution of nonlinear partial differential equations, J. Comput. Phys. 10 (1972) 40–52.

[2] J.O. Mingle, The method of differential quadrature for transient nonlinear diffusion, J. Math. Anal. Appl. 60 (1977) 559–569.

[3] F. Civan, C.M. Sliepcevich, Application of differential quadrature to transport processes, J. Math. Anal. Appl. 93 (1983) 711–724.

[4] F. Civan, C.M. Sliepcevich, Differential quadrature for multi-dimensional problems, J. Math. Anal. Appl. 101 (1984) 423–443.

[5] S.K. Jang, C.W. Bert, A.G. Striz, Application of differential quadrature to static analysis of structural components, Int. J. Numer. Meth. Eng. 28 (1989) 561–577.

[6] C. Shu, Generalized differential–integral quadrature and application to the simulation of incompressible viscous flows including parallel computation, Ph.D. Thesis, University of Glasgow, 1991.

[7] C. Shu, B.E. Richards, High resolution of natural convection in a square cavity by generalized differential quadrature, in: Proceedings of third International Conference on Advances in Numerical Methods in Engineering: Theory and Applications, vol. 2, Swansea, UK, 1990, pp. 978–985.

[8] C. Shu, B.E. Richards, Application of generalized differential quadrature to solve two-dimensional incompressible Navier–Stokes equations, Int. J. Numer. Meth. Fluids 15 (1992) 791–798.

[9] C. Shu, B.E. Richards, Parallel simulation of incompressible viscous flows by generalized differential quadrature, Comput. Systems in Eng. 3 (1992) 271–281.

[10] C. Shu, B.C. Khoo, K.S. Yeo, Y.T. Chew, Application of GDQ scheme to simulate natural convection in a square cavity, Int. Commun. Heat and Mass Trans. 21 (1994) 809–817.

[11] C. Shu, B.C. Khoo, K.S. Yeo, Numerical solutions of incompressible Navier–Stokes equations by generalized differential quadrature, Finite Elements in Analysis and Design 18 (1994) 83–97.

[12] C. Shu, Y.T. Chew, B.C. Khoo, K.S. Yeo, Solutions of three-dimensional boundary layer equations by global methods of generalized differential-integral quadrature, Int. J. Numer. Meth. Heat Fluid Flow 6 (1995) 61–75.

[13] C. Shu, B.C. Khoo, Y.T. Chew, K.S. Yeo, Numerical studies of unsteady boundary layer flows past an impulsively started circular cylinder by GDQ and GIQ approaches, Comput. Meth. Appl. Mech. Eng. 135 (1996) 229–241.

[14] C.W. Bert, S.K. Jang, A.G. Striz, Two new approximate methods for analyzing free vibration of structural components, AIAA J. 26 (1988) 612–618.

[15] X. Wang, C.W. Bert, A new approach in applying differential quadrature to static and free vibrational analyses of beams and plates, J. Sound Vibr. 162 (1993) 566–572.

[16] X. Wang, A.G. Striz, C.W. Bert, Free vibration analysis of annular plates by the DQ method, J. Sound Vibr. 164 (1993) 173–175.

[17] C.W. Bert, X. Wang, A.G. Striz, Differential quadrature for static and free vibration analysis of anisotropic plates, Int. J. Solids Structures 30 (1993) 1737–1744.

[18] P.A.A. Laura, R.H. Gutierrez, Analysis of vibrating Timoshenko beams using the method of differential quadrature, Shock and Vibration 1 (1993) 89–93.

[19] X. Wang, A.G. Striz, C.W. Bert, Buckling and vibration analysis of skew plates by the differential quadrature method, AIAA J. 32 (1994) 886–888.

[20] C.W. Bert, X. Wang, A.G. Striz, Static and free vibrational analysis of beams and plates by differential quadrature method, Acta Mechanica 102 (1994) 11–24.

[21] H. Du, M.K. Lim, R.M. Lin, Application of Generalized differential quadrature method to structural problems, Int. J. Numer. Meth. Eng. 37 (1994) 1881–1896.

[22] H. Du, M.K. Lim, R.M. Lin, Application of differential quadrature to vibration analysis, J. Sound Vibr. 181 (1995) 279–293.

[23] H. Du, K.M. Liew, M.K. Lim, Generalized differential quadrature method for buckling analysis, Journal of Engineering Mechanics 122 (1996) 95–100.

[24] C.W. Bert, M. Malik, Free vibration analysis of tapered rectangular plates by differential quadrature method: a semi-analytical approach, J. Sound Vibr. 190 (1996) 41–63.

[25] C.W. Bert, M. Malik, The differential quadrature method for irregular domains and application to plate vibration, Int. J. Mech. Sci. 38 (1996) 589–606.

[26] A.R. Kukreti, J. Farsa, Differential quadrature and Rayleigh–Ritz methods to determine the fundamental frequencies of simply supported rectangular plates with linearly varying thickness, J. Sound Vibr. 189 (1996) 103–122.

[27] K.M. Liew, J.B. Han, Z.M. Xiao, H. Du, Differential quadrature method for Mindlin plates on Winkler foundations, Int. J. Mech. Sci. 38 (1996) 405–421.

[28] C. Shu, Free vibration analysis of composite laminated conical shells by generalized differential quadrature, J. Sound Vibr. 194 (1996) 587–604.

[29] G. de Vahl Davis, I.P. Jones, Natural convection in a square cavity: a comparison exercise, Int. J. Numer. Meth. Fluids 3 (1983) 227–248.

[30] G. de Vahl Davis, Natural convection of air in a square cavity: a benchmark numerical solution, Int. J. Numer. Meth. Fluids 3 (1983) 249–264.