Compact computations based on a stream-function–velocity formulation of two-dimensional steady laminar natural convection in a square cavity

Pei Xiang Yu* and Zhen F. Tian†
Department of Mechanics and Engineering Science, Fudan University, Shanghai 200433, P.R. China
(Received 31 December 2010; published 13 March 2012)

A class of compact second-order finite difference algorithms is proposed for solving steady-state laminar natural convection in a square cavity using the stream-function–velocity (ψ-u) form of Navier-Stokes equations. The stream-function–velocity equation and the energy equation are all solved as a coupled system of equations for the four field variables consisting of stream function, two velocities, and temperature. Two strategies are considered for the discretizaton of the temperature equation, which are a second-order five-point compact scheme and a fourth-order nine-point compact scheme, respectively. The numerical capability of the presented algorithm is demonstrated by the application to natural convection in a square enclosure for a wide range of Rayleigh numbers (from 10^3 to 10^8) and compared with some of the accurate results available in the literature. The presented schemes not only show second-order accurate, but also prove effective. For larger Rayleigh numbers, the algorithm combining the second-order compact scheme for the stream-function–velocity equation with the fourth-order compact scheme for the temperature equation performs more stably and effectively.

DOI: 10.1103/PhysRevE.85.036703 PACS number(s): 47.11.−j, 44.25.+f

I. INTRODUCTION

Over the last few decades, the problem of a buoyancy-driven square cavity with adiabatic horizontal walls and differentially heated vertical walls has been studied widely because of its fundamental importance in the understanding of buoyancy-driven flows and its relevance to a wide range of engineering applications [1–3]. Natural convection heat transfer is relevant to large-scale natural phenomena in the fields of geophysics, atmospheric, astrophysics sciences, and a wide range of engineering applications such as the solar energy collector, cooling of electronic equipment, nuclear reactors, growing crystals, and solidification processes. Since the velocity and the temperature equations are coupled due to the buoyancy force, the study of natural convection is very complex.

Many numerical methods, including finite difference, finite element, finite volume, and lattice Bhatnagar-Gross-Krook model methods, have been employed to investigate the steady natural convection in a square cavity. Some numerical experiments have come to be known as the benchmark solutions [4–6], which would be used for investigating the performance of numerical methodologies solving the incompressible Navier-Stokes equations and for validating a computer code. De Vahl Davis [5] presented the benchmark solutions for $\mathrm{Ra} = 10^3$ to 10^6 using a second-order finite difference scheme and Richardson extrapolation; Hortmann *et al.* [6] proposed the benchmark solutions for Rayleigh numbers up to $\mathrm{Ra} = 10^6$ using the multigrid finite-volume multigrid method on a 640×640 nonuniform high-resolution grid; Le Quéré [4] used a second-order Chebychev polynomial approach to increase the spatial resolution and presented the benchmark solutions for $\mathrm{Ra} = 10^6$ to 10^8; Dennis and Hudson [7] proposed the compact h^4 difference solutions for Rayleigh numbers up to $\mathrm{Ra} = 10^5$; Choo and Schultz [8] developed the stable fourth-order difference solutions for $\mathrm{Ra} = 10^3$ to 10^6; Ramaswamy *et al.* [9] used a second-order finite-element method; Syrjälä [10] presented a higher-order Penalty-Galerkin finite-element approach for $\mathrm{Ra} = 10^4$ to 10^7; Ho and Lin [11] developed a pseudo-vorticity-velocity formulation for $\mathrm{Ra} = 10^3$ to 10^7; Nonino and Croce [12] presented an equal-order velocity-pressure finite-element algorithm for Rayleigh numbers up to $\mathrm{Ra} = 10^8$; Guo *et al.* [13] proposed a thermal lattice Boltzmann equation for $\mathrm{Ra} = 10^3$ to 10^6; and Kalita *et al.* [14] and Tian and Ge [15] established their fourth-order compact schemes for the stream-function–vorticity formulation, respectively. Kalita *et al.* [14] used conjugate gradient and hybrid biconjugate gradient-stabilized algorithms and computed the natural convection in a square cavity up to $\mathrm{Ra} = 10^7$. Tian and Ge [15] obtained highly accurate numerical solutions for both large Ra (up to 10^7) and small Pr (Pr is the nondimensional Prandtl number) using point-successive overrelaxation or point-successive underrelaxation iteration technique. Very recently Arpino *et al.* [16] presented the fully matrix-inversion-free artificial compressibility characteristic based split (AC-CBS) algorithm for the finite element method and obtained the new benchmark solutions for higher Rayleigh numbers (10^7 and 10^8) with a 450×450 grid. However, note that most methods are available to solve this problem in moderate Rayleigh numbers until 10^6, and therefore it is still necessary to develop accurate and efficient numerical methods suitable for a wide range of Rayleigh numbers. Due to the convenience of implementation on machines, difference methods are popular for natural convection in a square cavity [5,7,8,11,14,15].

Most of the numerical methods mentioned above are based on the stream-function–vorticity form of the two-dimensional (2D) incompressible Navier-Stokes equations, which have been used by a large number of researchers over the last several decades to test new methods for the numerical solutions of a variety of fluid flow and heat transfer problems [5,7,8,14,15]. However, the typical difficulty of the stream-function–vorticity

*ypxiang@gmail.com
†Corresponding author: zftian@fudan.edu.cn and z.f.tian@126.com

formulation is the lack of the simple physical boundary conditions for the vorticity field at the no-slip boundaries. In order to avoid the pitfalls associated with the vorticity values at the boundary, unlike the previous attempts, the stream-function–velocity or the stream-function formulation-based methodology for the solution of the 2D incompressible fluid flows, which eliminates the need to calculate the vorticity as a part of the computational process, has been utilized by some authors for solving Navier-Stokes equations [17–20]. Gupta and Kalita [19] proposed a new second-order nine-point scheme for solving the steady Navier-Stokes equations with the BiCGStab acceleration method, which obtained accuracy solutions with little additional cost for a couple of fluid flow problems. Ben-Artzi *et al.* [17] also proposed a very similar algorithm that can deal with the unsteady-state Navier-Stokes equations. The stream-function formulation includes the stream function and its first derivatives (i.e., velocities) resulting in a fourth-order differential equation in stream function. The boundary conditions for stream function and velocities are generally known and are easy to implement computationally; thus the computational schemes are found to be very efficient [17,19,20]. For the nature convection problem, some wide schemes based on stream-function–velocity formulation has been proposed in the last ten years. Bubnovich *et al.* [21] introduced an alternative direction implicit method to solve the transient natural convection problem. In their approach, upwind schemes were employed, and hence the accuracy cannot reach second order. Recently Hou and Wetton [22] presented a fourth-order wide scheme for the stream function formulation of the Navier-Stokes and Boussinesq equations. In their work, the numerical boundary condition is also applied for a stream function, which would not show the advantage of the stream-function formulation. Overall, no compact schemes have been found to solve the nature convection problem based on the stream-function–velocity formulation yet.

The present work proposes an idea to solve two-dimensional Navier-Stokes equations governing the fluid flow and heat transfer by efficiently exploiting the advantages of both the 2D Navier-Stokes equation in terms of the stream function and its first derivatives (velocities) as variables and the compact finite difference method. This work establishes an efficient second-order compact difference scheme to approximate the stream-function–velocity (ψ-u) formulation and develops two optional schemes including the treatments of the temperature Neumann boundary condition to approximate the energy equation. The proposed numerical difference algorithms are applied to determine the stream function, velocity, and temperature variables for the nature convection problem in a differentially heated square cavity, which is considered to be the best example problem to test the numerical capability of the presented coupled algorithms, for a Rayleigh number range from 10^3 to 10^8.

The remainder of this paper is organized as follows. Section II gives the nondimensional stream-function–velocity formulation of the governing equation for the natural convection problem. Section III deals with the discretization of the stream-function–velocity formulation and the energy equation with related issues. Section IV describes the solution of algebraic systems associated with the proposed finite difference approximations. Numerical experiments for a test problem is performed to validate the accuracy, and the results for the natural convection problem are discussed in Sec. V. Conclusions are included in Sec. VI.

II. THE PROBLEM

The problem being solved in this paper is a 2D square cavity with the width L filled with an incompressible flow of a Boussinesq fluid of $\mathrm{Pr} = 0.71$. Figure 1 shows the geometrical features of the cavity used. Both of the vertical walls are isothermal, and the left wall at temperature T_h is hotter than the right wall at temperature T_c, while the two horizontal walls are thermally insulated. The governing equations of the problem are written by

$$u_x + v_y = 0, \tag{1}$$

$$uu_x + vu_y = -\frac{1}{\rho}p_x + \nu(u_{xx} + u_{yy}), \tag{2}$$

$$uv_x + vv_y = -\frac{1}{\rho}p_y + \nu(v_{xx} + v_{yy}) + g\beta(T - T_0), \tag{3}$$

$$uT_x + vT_y = a(T_{xx} + T_{yy}), \tag{4}$$

where p, u, v, and T are the pressure, velocity components in x and y directions, and temperature, respectively. The kinematic viscosity ν, thermal diffusivity a, and coefficient of thermal expansion of fluid β are determined by the physical properties of the fluid at the reference density ρ and the reference temperature T_0, and g is the gravity acceleration. The above six parameters are treated as known and constant during the calculation. The subscripts x and y stand for the partial derivatives in the x and y direction, respectively. The boundary conditions are described as

$$\begin{aligned}
&u = v = 0, \quad T = T_h \quad \text{at} \quad x = 0 \ \ (0 \leqslant y \leqslant L),\\
&u = v = 0, \quad T = T_c \quad \text{at} \quad x = L \ \ (0 \leqslant y \leqslant L),\\
&u = v = 0, \quad T_y = 0 \quad \text{at} \quad y = 0 \ \ (0 \leqslant x \leqslant L),\\
&u = v = 0, \quad T_y = 0 \quad \text{at} \quad y = L \ \ (0 \leqslant x \leqslant L).
\end{aligned} \tag{5}$$

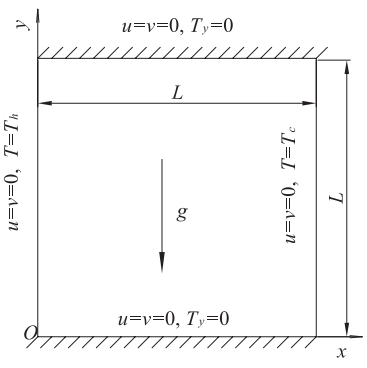

FIG. 1. Schematic view of the cavity problem for natural convection flow.

In order to make the above system dimensionless, the following nondimensional variables are defined:

$$x^* = \frac{x}{L}, \quad y^* = \frac{y}{L}, \quad u^* = \frac{uL}{a}, \quad v^* = \frac{vL}{a}, \quad T^* = \frac{T - T_0}{T_h - T_c}, \quad p^* = \frac{pH^2}{\rho a^2}. \tag{6}$$

Thus, the dimensionless form of Eqs. (1)–(4) on dropping the asterisks becomes

$$u_x + v_y = 0, \tag{7}$$

$$uu_x + vu_y = -p_x + \mathrm{Pr}(u_{xx} + u_{yy}), \tag{8}$$

$$uv_x + vv_y = -p_y + \mathrm{RaPr}T + \mathrm{Pr}(v_{xx} + v_{yy}), \tag{9}$$

$$uT_x + vT_y = T_{xx} + T_{yy}, \tag{10}$$

where $\mathrm{Ra} = (g\beta(T_h - T_c)L^3)/\nu a$ and $\mathrm{Pr} = \nu/a$ are the two dimensionless parameters. To eliminate the pressure p, we introduce the vorticity ω and the stream function ψ, which are defined by

$$\omega = u_y - v_x, \tag{11}$$

$$u = \psi_y, \quad v = -\psi_x. \tag{12}$$

Thus, Eqs. (7)–(9) can be written as

$$\psi_{xx} + \psi_{yy} = -\omega, \tag{13}$$

$$u\omega_x + v\omega_y = \mathrm{Pr}(\omega_{xx} + \omega_{yy} + \mathrm{Ra}T_x). \tag{14}$$

Combining the above two equations again, we can eliminate the vorticity ω as well. Finally, the stream function–velocity $(\psi - u)$ formulation of the full governing equations including the source term f and the temperature equation can be written as

$$\psi_{xxxx} + 2\psi_{xxyy} + \psi_{yyyy} = -\frac{1}{\mathrm{Pr}}[u(v_{xx} + v_{yy}) - v(u_{xx} + u_{yy})] + \mathrm{Ra}T_x - f, \tag{15}$$

$$u = \psi_y, \quad v = -\psi_x, \tag{16}$$

$$T_{xx} + T_{yy} = uT_x + vT_y. \tag{17}$$

When the reference temperature T_0 is taken as being equal to T_c, the dimensionless boundary conditions become

$$\begin{aligned} &\psi = u = v = 0, \quad T = 1 \quad \text{at} \quad x = 0 \quad (0 \leqslant y \leqslant 1),\\ &\psi = u = v = 0, \quad T = 0 \quad \text{at} \quad x = 1 \quad (0 \leqslant y \leqslant 1),\\ &\psi = u = v = 0, \quad T_y = 0 \quad \text{at} \quad y = 0 \quad (0 \leqslant x \leqslant 1),\\ &\psi = u = v = 0, \quad T_y = 0 \quad \text{at} \quad y = 1. \quad (0 \leqslant x \leqslant 1). \end{aligned} \tag{18}$$

The Nusselt number often describes the heat transfer characteristics across the cavity. In this problem we mainly pay attention to the Nusselt number on the hot wall, which is computed by

$$\mathrm{Nu} = -(T_x)_{0,j}, \tag{19}$$

and the average Nusselt number on that wall is computed by

$$\mathrm{Nu}_0 = -\int_0^1 (T_x)_{0,j}\, dy. \tag{20}$$

In this paper the composite Simpson integral formula is used to calculate Eq. (20).

III. SECOND-ORDER COMPACT (SOC) DIFFERENCE APPROXIMATION

A. Numerical scheme for the stream-function equation

For the sake of convenience, some standard finite difference (FD) operators at the grid point (x,y) are defined as follows:

$$\begin{aligned} \delta_x^2\delta_y\phi &= \frac{\phi_5 + \phi_6 - \phi_7 - \phi_8 - 2(\phi_2 - \phi_4)}{2h^3},\\ \delta_x\delta_y^2\phi &= \frac{\phi_5 - \phi_6 - \phi_7 + \phi_8 - 2(\phi_1 - \phi_3)}{2h^3},\\ \delta_x\delta_y\phi &= \frac{\phi_5 - \phi_6 + \phi_7 - \phi_8}{4h^2},\\ \delta_x^2\phi &= \frac{\phi_1 - 2\phi_0 + \phi_3}{h^2}, \quad \delta_x\phi = \frac{\phi_1 - \phi_3}{2h},\\ \delta_y^2\phi &= \frac{\phi_2 - 2\phi_0 + \phi_4}{h^2}, \quad \delta_y\phi = \frac{\phi_2 - \phi_4}{2h}. \end{aligned} \tag{21}$$

Here we use h to represent the space mesh size in x and y directions, and we number the nine mesh points (x,y), $(x+h,y)$, $(x,y+h)$, $(x-h,y)$, $(x,y-h)$, $(x+h,y+h)$, $(x-h,y+h)$, $(x-h,y-h)$, and $(x+h,y-h)$ as 0, 1, 2, 3, 4, 5, 6, 7, and 8, respectively (see Fig. 2).

According to the above notations and Taylor series, the following FD formulas at an interior grid point (x,y) yield

$$\begin{aligned} \psi_{xxyy} &= \tfrac{1}{2}\left(\delta_x\delta_y^2\psi_x + \delta_x^2\delta_y\psi_y\right) + O(h^2)\\ &= \tfrac{1}{2}\left(-\delta_x\delta_y^2 v + \delta_x^2\delta_y u\right) + O(h^2), \end{aligned} \tag{22}$$

$$u_{xx} = \delta_x^2 u + O(h^2), \tag{23}$$

$$u_{yy} = \delta_y^2 u + O(h^2), \tag{24}$$

$$v_{xx} = \delta_x^2 v + O(h^2), \tag{25}$$

$$v_{yy} = \delta_y^2 v + O(h^2). \tag{26}$$

For a sufficiently smooth solution ψ, the following Stephenson discrete operators (introduced in Ref. [23]) are used to

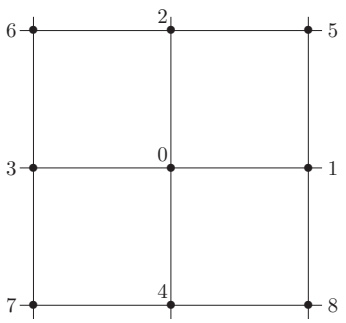

FIG. 2. Computational stencil.

approximate ψ_{xxxx} and ψ_{yyyy}:

$$\psi_{xxxx} = \frac{12}{h^2}\left(-\delta_x^2\psi + \delta_x\psi_x\right) + O(h^4)$$
$$= \frac{12}{h^2}\left(-\delta_x^2\psi - \delta_x v\right) + O(h^4), \quad (27)$$
$$\psi_{yyyy} = \frac{12}{h^2}\left(-\delta_y^2\psi + \delta_y\psi_y\right) + O(h^4)$$
$$= \frac{12}{h^2}\left(-\delta_y^2\psi + \delta_y u\right) + O(h^4), \quad (28)$$

with the fourth-order Páde scheme for ψ_x and ψ_y. The detail truncation error analysis can be seen in Ref. [24].

Substituting Eqs. (26)–(27) into (15), we obtain

$$48\psi_0 - 12\sum_{j=1}^{4}\psi_j = 6h(v_1 - v_3 - u_2 + u_4) + h^4\left(\delta_x\delta_y^2 v - \delta_x^2\delta_y u\right) + \frac{1}{\mathrm{Pr}}h^2\left(v_0\sum_{j=1}^{4}u_j - u_0\sum_{j=1}^{4}v_j\right) + h^4(\mathrm{Ra}T_x - f)_0 + O(h^6), \quad (29)$$

where T_{x0} and f_0 stand for the values of T_x and f at the mesh point (x,y), respectively. The fourth-order Páde scheme is used to compute the velocities u and v.

Hence, we can obtain the following second-order compact scheme for solving the stream function formulation of steady-state Navier-Stokes equations (15) with Eq. (16):

$$48\psi_0 - 12\sum_{j=1}^{4}\psi_j = 6h(v_1 - v_3 - u_2 + u_4) + h^4\left(\delta_x\delta_y^2 v - \delta_x^2\delta_y u\right) + \frac{1}{\mathrm{Pr}}h^2\left(v_0\sum_{j=1}^{4}u_j - u_0\sum_{j=1}^{4}v_j\right) + h^4(\mathrm{Ra}T_x - f)_0, \quad (30)$$

$$\frac{1}{6}v_1 + \frac{4}{6}v_0 + \frac{1}{6}v_3 = \frac{\psi_3 - \psi_1}{2h}, \quad (31)$$

$$\frac{1}{6}u_2 + \frac{4}{6}u_0 + \frac{1}{6}u_4 = \frac{\psi_2 - \psi_4}{2h}. \quad (32)$$

The above second-order compact approximations (30) for the stream-function formulation ψ at grid point (x,y) utilize only the values of ψ at four nearest-neighbor grid points (x,y) in the compact stencil. The algebraic systems arising from the newly proposed FD approximations (30) are linear and diagonally dominant. We also note that the associated matrices are symmetric and positive definite, which allows algorithms such as Jacobi, Gauss-Seidel, successive overrelaxation (SOR), and conjugate gradient to be used.

In addition, the first-order derivative of temperature T_x is included in the source term of Eq. (30), which at a point (x,y) may be approximated as follows:

$$\frac{1}{6}T_{x_1} + \frac{4}{6}T_{x_0} + \frac{1}{6}T_{x_3} = \frac{T_1 - T_3}{2h}. \quad (33)$$

In this problem the temperature on the hot (left) and the cold (right) walls are known. There is still need to calculate T_x on these boundaries when we try to use the discretization (33) mentioned above. Here we use the four-point fourth-order semi-implicit difference scheme, which on the left and right walls is given, respectively, by

$$T_{x_{1,j}} + 3T_{x_{2,j}} = \frac{1}{6h}(-17T_{1,j} + 9T_{2,j} + 9T_{3,j} - T_{4,j}), \quad (34)$$

$$T_{x_{N,j}} + 3T_{x_{N-1,j}} = \frac{1}{6h}(17T_{N,j} - 9T_{N-1,j} - 9T_{N-2,j} + T_{N-3,j}). \quad (35)$$

It can be seen that the full difference equations for T_x including Eqs. (34), (35), and (33) yield the tridiagonal linear systems, which can be directly solved. And it should be mentioned that the above scheme uses only the values in the x direction, so that it is convenient to calculate T_x of the corner points.

B. Numerical schemes for the temperature equation

1. *Second-order compact (SOC) scheme*

It is easy to obtain the following scheme for Eq. (17) by the operators defined in Eq. (21):

$$\delta_x^2 T + \delta_y^2 T = u_0\delta_x T + v_0\delta_y T. \quad (36)$$

Rewriting the above equation, a five-point second-order compact scheme is given by

$$8T_0 + (hu_0 - 2)T_1 + (hv_0 - 2)T_2 - (hu_0 + 2)T_3 - (hv_0 + 2)T_4 = 0. \quad (37)$$

2. *Fourth-order compact (FOC) scheme*

As mentioned in Ref. [25], the second-order central difference approximations often suffer from computational instablility, while the high-order compact schemes are computationally efficient and stable. So we also develop a fourth-order compact scheme for the temperature equation. Equation (17) can be treated as a Poisson-type equation, in which the source terms are described as $uT_x + vT_y$. Based on the method described in Ref. [15] for Poisson-type equation, a fourth-order semidiscretization for Eq. (17) can be obtained as follows:

$$\frac{1}{6h^2}\left(4\sum_{j=1}^{4}T_j + \sum_{j=5}^{8}T_j - 20T_0\right) = (uT_x + vT_y)_0 + \frac{h^2}{12}[(uT_x + vT_y)_{xx} + (uT_x + vT_y)_{yy}]_0. \quad (38)$$

Straightforwardly calculating and using Eq. (17), we have

$$(uT_x + vT_y)_{xx} + (uT_x + vT_y)_{yy} = T_x(u_{xx} + u_{yy}) + T_y(v_{xx} + v_{yy}) + 2(u_xT_{xx} + v_xT_{xy} + u_yT_{xy} + v_yT_{yy}) + u(T_{xx} + T_{yy})_x + v(T_{xx} + T_{yy})_y$$

$$
\begin{aligned}
&= T_x(u_{xx}+u_{yy}) + T_y(v_{xx}+v_{yy}) \\
&\quad + 2(u_xT_{xx} + v_xT_{xy} + u_yT_{xy} + v_yT_{yy}) \\
&\quad + u(u_xT_x + uT_{xx} + v_xT_y + vT_{xy}) \\
&\quad + v(u_yT_x + uT_{xy} + v_yT_y + vT_{yy}). \qquad (39)
\end{aligned}
$$

All of the first- and second-order derivatives of u, v, and T in Eq. (39) are approximated by the second-order operators defined in Eq. (21). That is, $[(uT_x + vT_y)_{xx} + (uT_x + vT_y)_{yy}]_0$ can give a truncation error of $O(h^2)$.

To reach a fully fourth-order FD, we also need to approximate $(uT_x + vT_y)_0$ with $O(h^4)$ accuracy. According to Taylor series expansion:

$$
\begin{aligned}
uT_x + vT_y = {} & \frac{1}{2h}[u(T_1 - T_3) + v(T_2 - T_4)] \\
& - \frac{h^2}{6}(uT_{xxx} + vT_{yyy}) + O(h^4). \qquad (40)
\end{aligned}
$$

From Eq. (17),

$$
\begin{aligned}
&uT_{xxx} + vT_{yyy} \\
&\quad = u(T_{xx} + T_{yy})_x + v(T_{xx} + T_{yy})_y - uT_{xyy} - vT_{xxy} \\
&\quad = u(u_xT_x + uT_{xx} + v_xT_y + vT_{xy}) \\
&\qquad + v(u_yT_x + uT_{xy} + v_yT_y + vT_{yy}) \\
&\qquad - uT_{xyy} - vT_{xxy}. \qquad (41)
\end{aligned}
$$

Combining Eqs. (38)–(41) and rearranging the coefficients, a nine-point fourth-order compact scheme is given by

$$
\sum_{j=0}^{8} A_j T_j = 0, \qquad (42)
$$

where

$$
\begin{aligned}
A_0 &= -160 - 8h(-u_1 + u_3 - v_2 + v_4) \\
&\quad - 8h^2\left(u_0^2 + v_0^2\right), \\
A_1 &= 32 - 2h(4u_0 + 3u_1 + u_2 - u_3 + u_4) \\
&\quad + h^2\left(4u_0^2 + u_0u_1 - u_0u_3 + u_2v_0 - u_4v_0\right), \\
A_2 &= 32 - 2h(4v_0 + v_1 + 3v_2 + v_3 - v_4) \\
&\quad + h^2\left(4v_0^2 + u_0v_1 + v_0v_2 - u_0v_3 - v_0v_4\right), \\
A_3 &= 32 + 2h(4u_0 - u_1 + u_2 + 3u_3 + u_4) \\
&\quad + h^2\left(4u_0^2 - u_0u_1 + u_0u_3 - u_2v_0 + u_4v_0\right), \\
A_4 &= 32 + 2h(4v_0 + v_1 - v_2 + v_3 + 3v_4) \\
&\quad + h^2\left(4v_0^2 - u_0v_1 - v_0v_2 + u_0v_3 + v_0v_4\right), \\
A_5 &= 8 + h(-4u_0 - u_2 + u_4 - 4v_0 - v_1 + v_3) \\
&\quad + 2h^2u_0v_0, \\
A_6 &= 8 + h(4u_0 + u_2 - u_4 - 4v_0 + v_1 - v_3) \\
&\quad - 2h^2u_0v_0, \\
A_7 &= 8 + h(4u_0 - u_2 + u_4 + 4v_0 - v_1 + v_3) \\
&\quad + 2h^2u_0v_0, \\
A_8 &= 8 + h(-4u_0 + u_2 - u_4 + 4v_0 + v_1 - v_3) \\
&\quad - 2h^2u_0v_0.
\end{aligned}
$$

C. Treatments of the boundary conditions

1. Third-order discretization for the Neumann boundary

In this problem, the temperature variables on the top and bottom walls are unknown, so we have to construct the other schemes to calculate the temperature variables on Neumann boundary conditions.

First, we consider the difference scheme of T_y on the bottom boundary. Using the forward difference operator, we get

$$
T_y = \delta_y^+ T - \frac{h}{2}T_{yy} + O(h^2), \qquad (43)
$$

where $\delta_y^+ T = (T_{i,j+1} - T_{i,j})/(2h)$.

Substituting Eq. (17) into Eq. (43) and noticing that the velocities u and v on the walls boundary are zeros, the above equation can be deduced by

$$
\begin{aligned}
T_y &= \delta_y^+ T + \frac{h}{2}T_{xx} + O(h^2) \\
&= \delta_y^+ T + \frac{h}{2}\delta_x^2 T + O(h^2). \qquad (44)
\end{aligned}
$$

Due to the Neumann boundary condition $T_y = 0$, the discretization with $O(h^3)$ truncation error on the bottom wall yields

$$
T_0 = \tfrac{1}{2}T_2 + \tfrac{1}{4}(T_1 + T_3). \qquad (45)
$$

Similarly, the discretization on the top wall can also be derived as follows:

$$
T_0 = \tfrac{1}{2}T_4 + \tfrac{1}{4}(T_1 + T_3). \qquad (46)
$$

When we combine the above third-order compact approximations on Neumann boundaries with the SOC scheme (37) for the inner points, the full schemes for the temperature equation are still compact and utilize only the values of T at four nearest-neighbor grid points (x, y).

2. Fourth-order discretization for the Neumann boundary

Similar to the above method, we can also derive the fourth-order accurate discretization on the insulated walls.

From Taylor series expansion,

$$
T_y = \delta_y^+ T - \frac{h}{2}T_{yy} - \frac{h^2}{6}T_{yyy} + O(h^3). \qquad (47)
$$

In the above subsection, we have applied Eq. (17) on the wall to approximate T_{yy} [i.e., $T_{yy} = -\delta_x^2 T + O(h^2)$], which indicates that the second item on the right-hand of Eq. (47) has already reached $O(h^3)$. Thus, we should just discretize T_{yyy} to $O(h)$ accuracy. Applying Eq. (17) on the wall again, we get

$$
\begin{aligned}
T_{yyy} &= (uT_x + vT_y - T_{xx})_y \\
&= u_yT_x + v_yT_y - T_{xxy} \\
&= \delta_y^+ u\delta_x T + \delta_y^+ v\delta_y^+ T - \delta_x^2\delta_y^+ T + O(h). \qquad (48)
\end{aligned}
$$

Here $u = v = 0$ has been considered on the wall.

Combining Eqs. (47)–(48) with $T_{yy} = -\delta_x^2 T + O(h^2)$, we have

$$T_y = \delta_y^+ T + \frac{h}{2}\delta_x^2 T - \frac{h^2}{6}\left(\delta_y^+ u \delta_x T + \delta_y^+ v \delta_y^+ T - \delta_x^2 \delta_y^+ T\right) + O(h^3). \tag{49}$$

Then the fourth-order compact schemes on the bottom and the top walls are given by

$$[20 - 2h(v_2 - v_0)]T_0 = [8 - 2h(v_2 - v_0)]T_2 + [4 - h(u_2 - u_0)]T_1 + [4 + h(u_2 - u_0)]T_3 + 2(T_5 + T_6), \tag{50}$$

and

$$[20 - 2h(v_0 - v_4)]T_0 = [8 - 2h(v_0 - v_4)]T_4 + [4 + h(u_0 - u_4)]T_1 + [4 - h(u_0 - u_4)]T_3 + 2(T_7 + T_8), \tag{51}$$

respectively. It should be mentioned that the full fourth-order compact schemes of the temperature equation including Eqs. (50)–(51) for boundaries and Eq. (42) for the inner points are still compact.

D. Summary

We have developed a second-order compact approximation [Eqs. (30)–(32)] for the stream-function equations (15). For the temperature equation, a second-order and a fourth-order compact schemes are also presented. The second-order scheme includes Eq. (37) for the inner region and Eqs. (45)–(46) for Neumann boundary conditions on the bottom and the top walls. The fourth-order scheme contains Eq. (42) for the inner region and Eqs. (50)–(51) for the bottom and the top boundaries. Thus, two compact difference methods can be obtained for solving the full governing equations [(15)–(17)]. One is combined with the second-order compact scheme for the stream function and the second-order compact scheme for the temperature (SOC-SOC), and another is combined with the second-order compact scheme for the stream function and the fourth-order compact scheme for the temperature (SOC-FOC). Because the approximation (30) for the stream function is only second-order accurate, the full accuracy of the two present methods is $O(h^2)$ as well.

IV. SOLUTION OF ALGEBRAIC SYSTEMS

It is necessary to accelerate the computation of the algebraic systems arising from the proposed approximations. In the present work, the inner-outer iteration technique [19,26] is used to solve the flow problems.

We notice that the solution procedure for the stream-function equation costs much time when the conventional iteration methods such as Jacobi, Gauss-Seidel, and SOR are employed. To improve the convergence, the multigrid method is hence employed to solve the sparse linear systems arising from the approximations of the stream-function equation (30). By a conventional iterative technique (e.g., Gauss-Seidel), the high-frequency error components are generally removed faster than low-frequency components. However, by the multigrid technique, low-frequency components can also be removed fast in a coarser grid because they would be treated as high-frequency ones. In general, the multigrid algorithm is made up of the presmoothing, coarse-grid correction, and postsmoothing steps. A multigrid V cycle is the computational process that goes from the finest grid down to the coarsest grid and back from the coarsest up to the finest. A V(m_1, m_2) cycle stands for a multigrid V-cycle algorithm that performs m_1 presmoothing sweeps in the finer grid and performs m_2 postsmoothing sweeps in the coarser grid, respectively. The detailed description of the multigrid method can be found in Refs. [27–30] and references therein.

For convenience, we describe only the procedure of the present SOC-SOC algorithm as follows. Suppose ψ^n, u^n, v^n, and T^n are known; the solution of the discrete Navier-Stokes equations [(30)–(32) and (37)] is obtained by the following iterative procedure:

(1) In the inner region $[h,1-h]\times[h,1-h]$, solve the stream-function equation (30) with multigrid V(3, 3) cycle algorithm:

$$48\psi_0^{n+1} - 12\sum_{j=1}^{4}\psi_j^{n+1} = 6h\left(v_1^n - v_3^n - u_2^n + u_4^n\right) + h^4\left(\delta_x\delta_y^2 v^n - \delta_x^2\delta_y u^n\right) + \frac{1}{\mathrm{Pr}}h^2\left(v_0^n\sum_{j=1}^{4}u_j^n - u_0^n\sum_{j=1}^{4}v_j^n\right) + h^4\left(\mathrm{Ra}T_x^n - f^n\right). \tag{52}$$

The above process constitutes inner iterations to calculate ψ^{n+1}, and we repeat the step five times.

(2) Determine the velocities u^{n+1} and v^{n+1} in the same region from Eqs. (31) and (32):

$$\frac{1}{6}v_1^{n+1} + \frac{4}{6}v_0^{n+1} + \frac{1}{6}v_3^{n+1} = \frac{\psi_3^{n+1} - \psi_1^{n+1}}{2h}, \tag{53}$$

$$\frac{1}{6}u_2^{n+1} + \frac{4}{6}u_0^{n+1} + \frac{1}{6}u_4^{n+1} = \frac{\psi_2^{n+1} - \psi_4^{n+1}}{2h}. \tag{54}$$

(3) Solve the temperature equation:

(a) Firstly, solve the temperature on Neumann boundaries. Equations (45) and (46) are used for the bottom and the top walls, respectively:

$$T_0^{n+1} = \frac{1}{2}T_2^{n+1} + \frac{1}{4}\left(T_1^{n+1} + T_3^{n+1}\right), \tag{55}$$

$$T_0^{n+1} = \frac{1}{2}T_4^{n+1} + \frac{1}{4}\left(T_1^{n+1} + T_3^{n+1}\right). \tag{56}$$

(b) In the region $[h,1-h]\times[h,1-h]$, solve the temperature equation (37) by

$$8T_0^{n+1} + \left(hu_0^{n+1} - 2\right)T_1^{n+1} + \left(hv_0^{n+1} - 2\right)T_2^{n+1} - \left(hu_0^{n+1} + 2\right)T_3^{n+1} - \left(hv_0^{n+1} + 2\right)T_4^{n+1} = 0. \tag{57}$$

TABLE I. The L^2 errors and the convergence rates of the stream function ψ, temperature T, and velocities u and v in $\mathrm{Ra} = 10^4$ and $\mathrm{Pr} = 1$.

Gird	ψ error	Rate	T error	Rate	u error	Rate	v error	Rate
				SOC-SOC				
17×17	1.3164(−6)[a]		1.0523(−7)		4.4699(−6)		4.3052(−6)	
33×33	3.3712(−7)	1.965	2.6776(−8)	1.975	1.1535(−6)	1.954	1.1282(−6)	1.932
65×65	8.5277(−8)	1.983	6.7623(−9)	1.985	2.9222(−7)	1.981	2.8216(−7)	1.999
129×129	2.1294(−8)	2.003	1.6929(−9)	1.998	7.2153(−8)	2.018	7.0563(−8)	2.000
				SOC-FOC				
17×17	1.3255(−6)		1.0464(−7)		4.4991(−6)		4.3352(−6)	
33×33	3.3770(−7)	1.973	2.6738(−8)	1.968	1.1553(−6)	1.961	1.1147(−6)	1.959
65×65	8.5320(−8)	1.985	6.7605(−9)	1.984	2.9235(−7)	1.982	2.8230(−7)	1.981
129×129	2.1346(−8)	1.999	1.6969(−9)	1.994	7.2341(−7)	2.015	7.0744(−8)	1.997

[a] $1.3164(-6) = 1.3164 \times 10^{-6}$, etc.

(c) The steps (a) and (b) constitute the inner iteration to calculate the temperature T^{n+1} in the whole region. Here we repeat this computation ten times.

(4) Determine the gradients of temperature T_x^{n+1} [(34)–(35)]:

(a) On the left and the right boundaries, Eqs. (34) and (35) are used, respectively, by

$$T_{x_{1,j}}^{n+1} + 3T_{x_{2,j}}^{n+1} = \frac{1}{6h}\left(-17T_{1,j}^{n+1} + 9T_{2,j}^{n+1} + 9T_{3,j}^{n+1} - T_{4,j}^{n+1}\right), \quad (58)$$

$$T_{x_{N,j}}^{n+1} + 3T_{x_{N-1,j}}^{n+1} = \frac{1}{6h}\left(17T_{N,j}^{n+1} - 9T_{N-1,j}^{n+1} - 9T_{N-2,j}^{n+1} + T_{N-3,j}^{n+1}\right). \quad (59)$$

(b) In the inner field, T_x is computed by

$$\frac{1}{6}T_{x_1}^{n+1} + \frac{4}{6}T_{x_0}^{n+1} + \frac{1}{6}T_{x_3}^{n+1} = \frac{T_1^{n+1} - T_3^{n+1}}{2h}. \quad (60)$$

Both of the two parts are combined as the tridiagonal matrix, which can be solved by the Thomas algorithm.

TABLE II. Numerical performances for the present two methods with different Ra and $\mathrm{Pr} = 0.71$.

		SOC-SOC		SOC-FOC		Rate[a]	
Grid size	λ	CPU	Iter.[b]	CPU	Iter.	CPU	Iter.
			$\mathrm{Ra} = 10^3$				
33×33	1.7	Div.[c]	Div.	5.984	2249		
	1.6	4.484	2391	6.406	2391	1.43	1.00
	1.5	4.875	2558	6.797	2559	1.39	1.00
	1.2	6.125	3264	8.562	3266	1.40	1.00
	1.0	7.562	4049	10.547	4051	1.39	1.00
65×65	1.6	Div.	Div.	113.078	9757		
	1.5	85.203	10568	121.906	10572	1.43	1.00
	1.2	114.422	14185	164.531	14191	1.44	1.00
	1.0	153.703	18340	215.937	18347	1.40	1.00
			$\mathrm{Ra} = 10^5$				
33×33	1.6	Div.	Div.	2.140	797		
	1.5	1.609	852	2.334	849	1.45	1.00
	1.2	2.000	1065	2.812	1062	1.41	1.00
	1.0	2.484	1293	3.421	1289	1.38	1.00
65×65	1.6	Div.	Div.	33.875	3080		
	1.5	27.047	3345	36.140	3309	1.34	0.99
	1.2	34.766	4344	47.109	4300	1.36	0.99
	1.0	44.281	5470	59.128	5416	1.34	0.99
			$\mathrm{Ra} = 10^7$				
33×33	0.2	Div.	Div.	6.922	2566		
65×65	0.4	Div.	Div.	52.953	4788		
129×129	0.9	Div.	Div.	483.125	7322		
	0.5	787.610	16707	1140.296	16764	1.45	1.00

[a] The rate is calculated by the value of the SOC-FOC scheme over the value of the SOC-SOC scheme.
[b] Iter. = Outer iterative number.
[c] Div. = Divergence.

TABLE III. Comparison of the results by different numerical methods with different Ra and Pr = 0.71.

Reference	$\lvert\psi_{\text{mid}}\rvert$	$\lvert\psi\rvert_{\max}$	$u_{\max}$	$v_{\max}$	Nu_0	$\text{Nu}_{\max}$	$\text{Nu}_{\min}$
			$\text{Ra} = 10^3$				
SOC-SOC	1.1748	1.1748	3.6499	3.6978	1.1178	1.5062	0.6913
SOC-FOC	1.1748	1.1748	3.6499	3.6978	1.1178	1.5064	0.6912
Tian [15]	1.1741	1.1741	3.6481	3.6958	1.1176	1.5058	0.6912
Dennis [7]	1.1747	n.a.[a]	3.6497	3.6977	1.1176	1.5058	0.6913
Kalita [14]	1.175	n.a.	3.650	3.697	1.118	1.505	0.692
De Vahl Davis [5]	1.174	n.a.	3.649	3.697	1.117	1.505	0.692
			$\text{Ra} = 10^4$				
SOC-SOC	5.0734	5.0734	16.1796	19.6268	2.2445	3.5304	0.5853
SOC-FOC	5.0741	5.0741	16.1817	19.6284	2.2449	3.5313	0.5850
Tian [15]	5.0738	5.0738	16.1837	19.6282	2.2441	3.5295	0.5847
Dennis [7]	5.0735	n.a.	16.1829	19.6293	2.2396	3.5193	0.5851
Kalita [14]	5.080	n.a.	16.203	19.613	2.243	3.526	0.586
De Vahl Davis [5]	5.071	n.a.	16.178	19.617	2.238	3.528	0.586
			$\text{Ra} = 10^5$				
SOC-SOC	9.1135	9.6128	34.7068	68.5018	4.5199	7.7152	0.7290
SOC-FOC	9.1194	9.6202	34.7363	68.5385	4.5214	7.7216	0.7280
Tian [15]	9.1161	9.6173	34.7417	68.6383	4.5195	7.7121	0.7275
Dennis [7]	9.1126	n.a.	34.716	68.637	4.4959	7.6830	0.7279
Kalita [14]	9.123	n.a.	34.825	68.606	4.512	7.670	0.733
De Vahl Davis [5]	9.111	9.612	34.73	68.59	4.509	7.717	0.729
Nonino [12]	n.a.	9.618	34.749	68.646	4.521	7.729	0.723
			$\text{Ra} = 10^6$				
SOC-SOC	16.3667	16.7885	64.6038	220.0961	8.8041	17.4143	0.9836
SOC-FOC	16.4183	16.8411	64.8066	220.6730	8.8091	17.4752	0.9798
Tian [15]	16.3863	16.8107	64.8308	220.5675	8.8216	17.5087	0.9787
Le Quéré [4]	16.3864	16.8111	64.8344	220.559	8.8252	17.5360	0.9795
Kalita [14]	16.420	n.a.	65.332	221.658	8.763	17.018	1.007
De Vahl Davis [5]	16.320	16.750	64.63	219.36	8.817	17.925	0.989
Nonino [12]	n.a.	16.817	64.827	220.63	8.825	17.558	0.983
			$\text{Ra} = 10^7$				
SOC-SOC	29.2046	29.9963	145.1701	690.9627	16.2975	38.7885	1.3869
SOC-FOC	29.5445	30.3506	148.2162	697.8232	16.3024	39.2433	1.3724
Tian [15]	29.3562	30.1553	148.5695	699.2991	16.5106	39.2540	1.3655
Kalita [14]	29.382	n.a.	155.82	696.238	16.075	34.925	1.509
Le Quéré [4]	29.361	30.165	148.59	699.17	16.523	39.3947	1.3664
Nonino [12]	n.a.	30.165	148.59	699.67	16.522	39.472	1.377
Arpino [16]	n.a.	*n.a.*	148.6	699.2	16.52	39.35	1.366
			$\text{Ra} = 10^8$				
SOC-FOC	52.5875	54.0957	314.8838	2227.43	29.9137	88.3745	1.9222
Le Quéré [4]	52.3223	53.8475	321.875	2222.39	30.255	87.2335	1.9191
Nonino [12]	n.a.	53.885	320.67	2223.2	30.225	87.911	1.949
Arpino [16]	n.a.	n.a.	322.7	2223	30.31	88.18	1.923

[a]n.a. = Not available.

(5) Repeat steps 1 to 4(which constitute the inner iteration) from $n = 0, 1, 2,...$until the maximum ψ error between two successive outer iteration steps is smaller than a certain convergence criterion.

Here a same relaxation parameter λ is employed to slightly overrelax to solve ψ^{n+1} and T^{n+1} in the outer iterations. Similarly, the detail procedure for the SOC-FOC scheme can be made like the above steps. In this article all of the numerical computations are run on a SΛMSUNG R23$^{\text{plus}}$ machine with 2 Gbytes of memory using double precision arithmetic.

V. RESULTS AND DISCUSSION

A. Verification of the discretization

According to the analysis from the above section, the two approaches we developed are second-order accurate for the full system no matter what scheme we chose to calculate the temperature equation because the stream-function equation can reach only h^2 accuracy. In the next subsection we will show the advantage of the fourth-order scheme for solving the temperature equation.

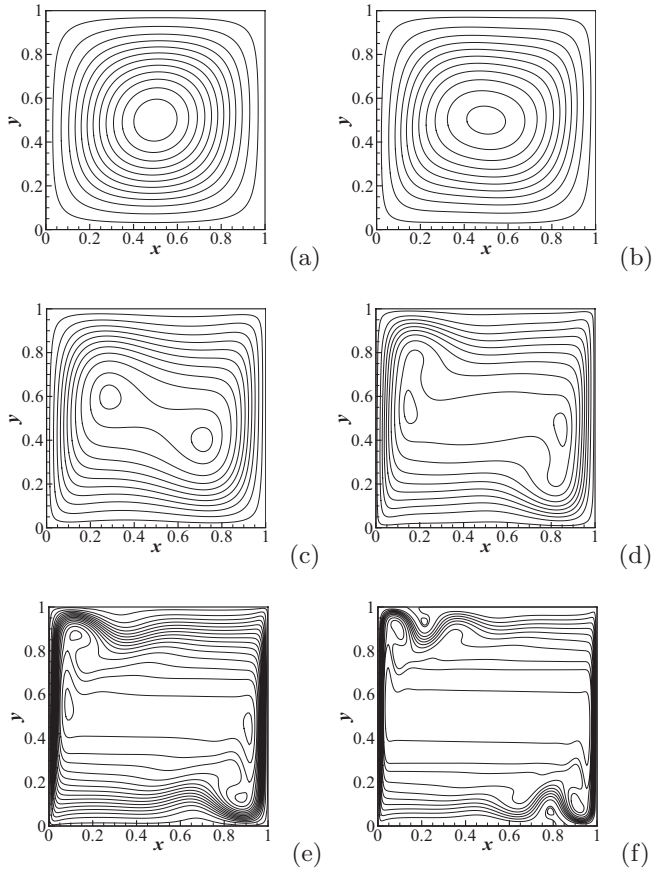

FIG. 3. Stream-function contours computed with 129 × 129 grid (Ra = 10^3–10^7) or 257 × 257 grid (Ra = 10^8): (a) Ra = 10^3, (b) Ra = 10^4, (c) Ra = 10^5, (d) Ra = 10^6, (e) Ra = 10^7, and (f) Ra = 10^8.

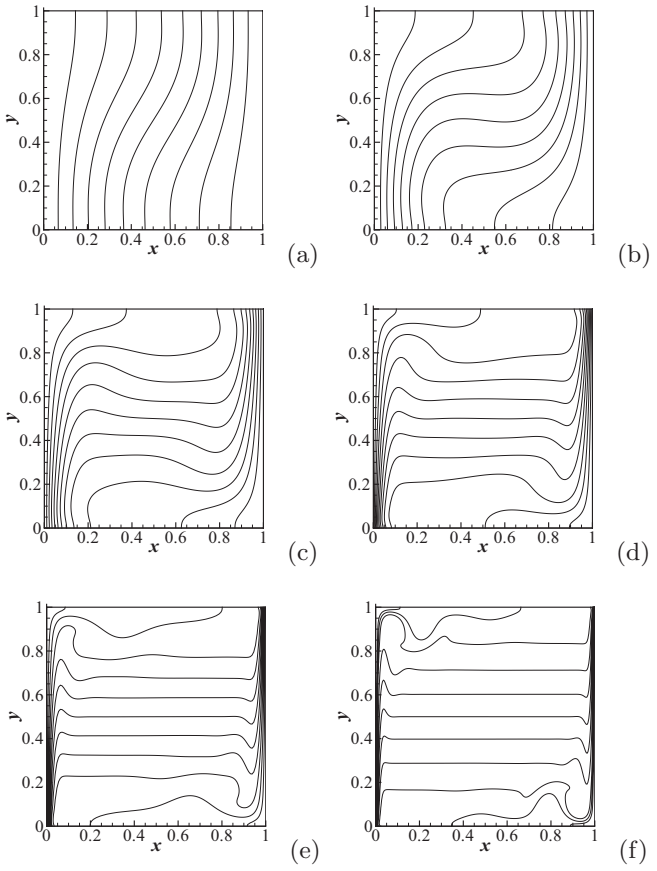

FIG. 4. Isothermal contours computed with 129 × 129 grid (Ra = 10^3–10^7) or 257 × 257 grid (Ra = 10^8): (a) Ra = 10^3, (b) Ra = 10^4, (c) Ra = 10^5, (d) Ra = 10^6, (e) Ra = 10^7, and (f) Ra = 10^8.

In this subsection a test problem with an analytical solution is considered. Following Ref. [15], one solution is given by

$$T = x + y, \quad \psi = \frac{1}{\mathrm{Pr}} e^{x+y} \tag{61}$$

on the unit square. Thus the corresponding velocities and the forcing function can be calculated as

$$u = v = \frac{1}{\mathrm{Pr}} e^{x+y}, \quad f = \mathrm{Ra} - \frac{4}{\mathrm{Pr}} e^{x+y}. \tag{62}$$

The above solution is smooth but does not satisfy the Neumann boundary condition, and thereby we solve the test problem with Dirichlet boundary conditions; i.e., the boundary values of T, ψ, u, and v are given. Here we choose Pr = 1.0 and Ra = 10^4 with the convergence criterion of 10^{-12}. Table I shows L^2 errors of the stream function, temperature, and velocities with various grid sizes, respectively, and the convergence rates calculated by the neighbor coarser and finer grid sizes.

It is observed that the convergence rates of both the SOC-SOC and the SOC-FOC schemes are very close to 2, and the results calculated by two methods are almost same. This confirms that both the SOC-SOC and the SOC-FOC schemes are of second-order spatial accuracy.

B. Results of the natural convection problem

In this section the classical natural convection problems are solved by our compact second-order accurate algorithms. In order to compare the results with already existing results, the Prandtl number is set as 0.71 with Ra ranging from 10^3 to 10^8. Here the convergence criterion is 10^{-10}.

First, the numerical performance of our two second-order schemes with different discretization of the temperature equation is compared. Table II shows CPU times and the numbers of outer iterations by different relaxation parameters λ in different Rayleigh numbers including 10^3, 10^5, and 10^7 with different grids including 33 × 33, 65 × 65, and 129 × 129.

From the table, we notice that for the same Ra and the same grid, if both of the discretization can converge to the same criterion, the numbers of outer iteration are approximately the same. However, differences exist for the two methods. On the one hand, the SOC-SOC scheme takes much less time for the same relaxation parameters because the scheme for the temperature equation is much simpler. For example, when we choose the same parameters including Ra, λ, and

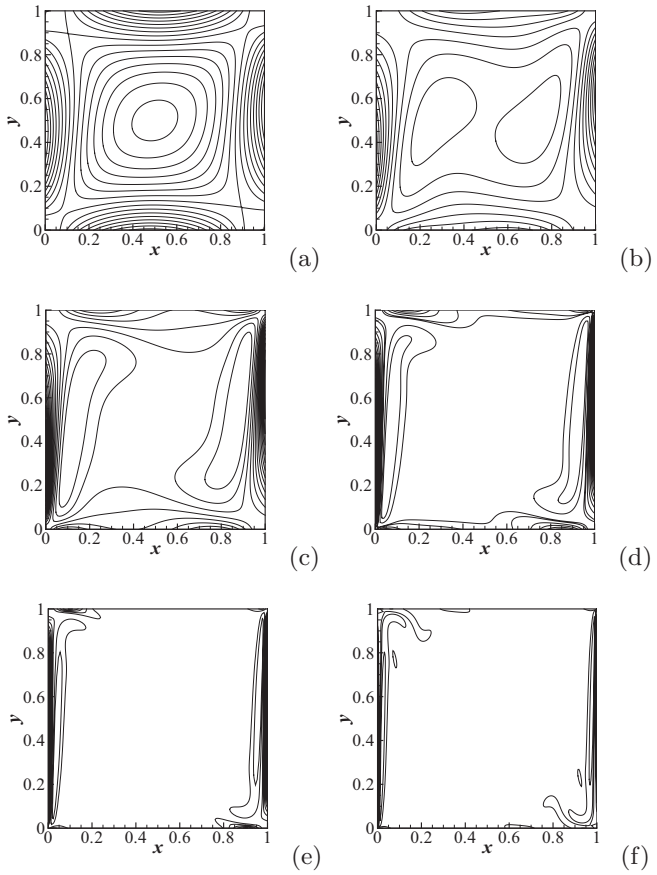

FIG. 5. Vorticity contours computed with 129×129 grid ($\mathrm{Ra} = 10^3$–10^7) or 257×257 grid ($\mathrm{Ra} = 10^8$): (a) $\mathrm{Ra} = 10^3$, (b) $\mathrm{Ra} = 10^4$, (c) $\mathrm{Ra} = 10^5$, (d) $\mathrm{Ra} = 10^6$, (e) $\mathrm{Ra} = 10^7$, and (f) $\mathrm{Ra} = 10^8$.

grid, the CPU times of the SOC-FOC scheme are about 1.4 times over the ones of the SOC-SOC scheme. On the other hand, the SOC-FOC scheme performs more stably. It can be found that the SOC-FOC scheme can use the larger relaxation parameters λ, which will accelerate the convergence. Although this advantage is not obvious in small Ra, it plays a great role in the convergence solution for $\mathrm{Ra} = 10^7$. For the SOC-SOC scheme, computations reach divergence with 33×33 and 65×65 grids even when smaller λ is chosen, while the convergence results by the SOC-FOC scheme can be obtained. When a 129×129 grid is used, the necessary λ are 0.5 and 0.9 for the SOC-SOC and SOC-FOC schemes, respectively. CPU times to reach the convergence solution are 787.610 and 483.125 s, respectively, which means the SOC-FOC scheme can save about 40% CPU time. For these results, it can be concluded that the higher-order compact scheme can make the full system run more stably and perform more efficiently.

Second, we compare our solutions with the other existing numerical results. The quantities presented here are the absolute stream function $|\psi_{\mathrm{mid}}|$ at the midpoint of the cavity, the maximum absolute value of stream function $|\psi|_{\mathrm{max}}$, the maximum horizontal velocity u_{max} on the vertical midplane, the maximum vertical velocity v_{max} on the horizontal midplane, the average Nusselt number Nu_0 on the hot wall, and the maximum and minimum values $\mathrm{Nu}_{\mathrm{max}}$ and $\mathrm{Nu}_{\mathrm{min}}$ of the local Nusselt number on the hot wall.

In Table III the above quantities computed by our two methods with 129×129 grid size ($\mathrm{Ra} = 10^3 - 10^7$) and 257×257 grid size ($\mathrm{Ra} = 10^8$) and some well-established results for this problem including benchmark solutions proposed by De Vahl Davis ($\mathrm{Ra} = 10^3 - 10^6$) [5] and by Le Quéré ($\mathrm{Ra} = 10^6 - 10^8$) [4] are tabulated. We notice that, for the range of Rayleigh numbers considered, the numerical results calculated by the present second-order scheme (SOC-FOC) are in excellent agreement with all the available data from the literature [4,14–16]. It is worth pointing out that the compared results listed in Table III were obtained by higher-order accurate schemes [4,14,15] or a nonuniform grid [12] or finer grids [16].

Figures 3–5 contain the contours for the stream, vorticity, and temperature functions. For the range of Rayleigh numbers, the distributions of the stream, temperature, and vorticity are symmetric with respect to the center, which is also agreement with the property for the flow. It can be seen that a central elliptic vortex in the center of the cavity when the Rayleigh numbers are small ($\mathrm{Ra} = 10^3$ and 10^4). As Ra increases, the vortex breaks into two vortices moving toward the vertical boundaries, and recirculation regions appear at the upper left and lower right corners at $\mathrm{Ra} = 10^7$. When Ra reaches 10^8, two small vortices attach on the top and the bottom walls, which agrees with the phenomenon presented in Ref. [4].

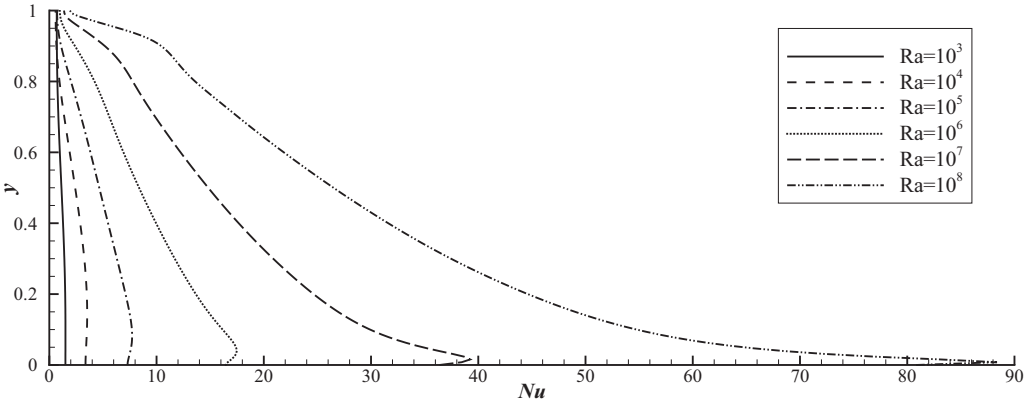

FIG. 6. The variation of the average Nusselt numbers across the hot wall for different Rayleigh numbers.

Figure 6 shows that the variation of Nusselt numbers across the hot wall, it can be found that the Nusselt number changes more drastically near the bottom wall as the Rayleigh number increases, which is in accordance with the results presented in Ref. [14].

VI. CONCLUSION

In this paper we focus on the researches of the compact schemes for the standard thermally driven square cavity problem with adiabatic horizontal walls and differentially heated vertical walls with different Rayleigh numbers varying from 10^3 to 10^8. The proposed algorithm is able to solve the 2D Navier-Stokes steady equation in terms of the stream function and its first derivatives (velocities) as variables. Although this method reaches only second-order accuracy, there are three advantages. The first one is that the method utilizes the ψ-u formulation, which can avoid dealing with the boundaries because the values on the boundaries are all known. The second one is that the present method is based on a compact scheme for the stream function–velocity formulation, which utilizes only the stream function values at (x,y), and its four nearest neighboring grid points and the coefficients are all constant. The associated matrices are symmetric and positive definite, which allows the conventional iterative methods to be used. Here, in order to accelerate the computation, a multigrid technique is employed to solve the stream function. The third one is that the third-order and the fourth-order schemes developed for Neumann boundary conditions of temperature at the insulated walls are also compact. To match the full algorithms, the second-order and the fourth-order compact schemes for the temperature equation are proposed, respectively.

A numerical experiment with a analytic solution is conducted to verify the accuracy of the present algorithms. After that, we calculate the classical natural convection in a closure square cavity with adiabatic horizontal walls and differentially heated vertical walls for values of Ra varying from 10^3 to 10^8 and $\mathrm{Pr} = 0.71$. Both of the experiments show that our schemes with the multigrid technique not only are of second-order accuracy, but also prove effective in the aspect of computational cost, which is reflected by the low demand on CPU time. Considering the numerical performances, it can be found that the higher-order discretization for the temperature (i.e., SOC-FOC) can make the full system computation more stable. As a result, in some cases, especially for larger Rayleigh numbers (i.e., $\mathrm{Ra} = 10^7$ and 10^8), the SOC-FOC scheme is more efficient than the SOC-SOC scheme. Because the stream-function–velocity formulation is convenient to handle with the boundary, the new proposed scheme seems to have good potential for efficient application to many problems of incompressible viscous flows.

Finally, it is worth pointing out that an extension of the present compact scheme to the stream-function formulation of unsteady incompressible viscous heat transfer is realizable. A fourth scheme for the pure-stream-function formulation of Navier-Stokes equations was proposed by Ben-Artzi *et al*. [31] very recently. So an investigation of the fourth-order scheme is underway as well.

ACKNOWLEDGMENTS

This work was supported in part by the National Natural Science Foundation of China under Grants Nos. 10972058 and 10662006, the Research Fund for the Doctoral Program of Higher Education of China (Grant No. 20100071110017), the China Postdoctoral Science Foundation, and the Teaching and Research Award Program for Outstanding Young Teachers in Higher Education Institutions of MOE, P.R. China.

[1] I. Catton, in *Int. Heat Transfer Conf. Proc. 6th*, Vol. 6 (Elsevier Pub. Co., Toronto, Canada, 1978), p. 13.
[2] S. Ostrach, J. Heat Transfer **110**, 1175 (1988).
[3] K. Yang, J. Heat Transfer **110**, 1191 (1988).
[4] P. L. Quéré, Comput. Fluid **20**, 29 (1991).
[5] G. De Vahl Davis, Int. J. Numer. Meth. Fluids **3**, 249 (1983).
[6] M. Hortmann, M. Peric, and G. Scheure, Int. J. Numer. Meth. Fluids **11**, 189 (1990).
[7] S. C. R. Dennis and J. D. Hudson, J. Comput. Phys. **85**, 390 (1989).
[8] J. Y. Choo and D. H. Schultz, Int. J. Numer. Meth. Fluids **15**, 1313 (1992).
[9] B. Ramaswamy, T. C. Jue, and J. E. Akin, AIAA J. **30**, 412 (1992).
[10] S. Syrjälä, Numer. Heat Transfer A **29**, 197 (1996).
[11] C. J. Ho and F. Lin, Numer. Heat Transfer A **31**, 881 (1997).
[12] C. Nonino and G. Croce, Numer. Heat Transfer B **32**, 17 (1997).
[13] Z. Guo, C. Zheng, B. Shi, and T. S. Zhao, Phys. Rev. E **75**, 036704 (2007).
[14] J. C. Kalita, D. C. Dalal, and A. K. Dass, Phys. Rev. E **64**, 066703 (2001).
[15] Z. F. Tian and Y. B. Ge, Int. J. Numer. Meth. Fluids **41**, 495 (2003).
[16] F. Arpino, N. Massarottib, and A. Maurob, Numer. Heat Transfer B **58**, 73 (2010).
[17] M. Ben-Artzi, J. P. Croisille, D. Fishelov, and S. Trachtenberg, J. Comput. Phys. **205**, 640 (2005).
[18] R. Kupferman, SIAM J. Sci. Comput. **23**, 1 (2001).
[19] M. M. Gupta and J. C. Kalita, J. Comput. Phys. **207**, 52 (2005).
[20] Z. F. Tian and P. X. Yu, J. Comput. Phys. **230**, 6404 (2011).
[21] V. Bubnovich, C. Rosas, R. Santander, and G. Cáceres, Numer. Heat Transfer A **42**, 401 (2002).
[22] T. Y. Hou and B. R. Wetton, J. Comput. Appl. Math. **27**, 441 (2009).
[23] J. W. Stephenson, J. Comput. Phys. **55**, 65 (1984).
[24] M. Ben-Artzi, J. P. Croisille, and D. Fishelov, SIAM J. Numer. Anal. **44**, 1997 (2006).
[25] M. Li, T. Tang, and B. Fornberg, Int. J. Numer. Meth. Fluids **20**, 1137 (1995).

[26] M. M. Gupta, J. Comput. Phys. **93**, 343 (1991).
[27] J. Zhang, Ph.D. dissertation, George Washington University, 1997.
[28] J. Zhang, Comput. Math. Appl. **45**, 43 (2005).
[29] P. Wesseling, *An Introduction to Multigrid Methods* (Wiley, Chichester, England, 1992).
[30] A. Brandt, Comput. Math. **31**, 333 (1977).
[31] M. Ben-Artzi, J. P. Croisille, and D. Fishelov, J. Sci. Comput. **42**, 216 (2010).