Applied Mathematical Modelling 39 (2015) 2602–2616

Contents lists available at ScienceDirect

Applied Mathematical Modelling

journal homepage: www.elsevier.com/locate/apm

Performance of geometric multigrid method for coupled two-dimensional systems in CFD

C.D. Santiago [a], C.H. Marchi [b,*], L.F. Souza [c]

[a] Federal Technology University of Paraná (UTFPR), Zip Code: 86812-460, Apucarana, Paraná, Brazil
[b] Laboratory of Numerical Experimentation (LENA), Department of Mechanical Engineering (DEMEC), Federal University of Paraná (UFPR), Caixa postal 19040, CEP 81531-980, Curitiba, Paraná, Brazil
[c] University of São Paulo, Department of Applied Mathematics and Statistics, Campus de São Carlos, Caixa postal 668, Zip Code: 13560-220, São Carlos, SP, Brazil

ARTICLE INFO

Article history:
Received 6 October 2013
Received in revised form 8 July 2014
Accepted 29 October 2014
Available online 22 November 2014

Keywords:
Finite difference
Multigrid
Solvers
Burgers
Laplace
Navier–Stokes

ABSTRACT

The performance of a geometric multigrid method is analyzed for two-dimensional Laplace, Navier, Burgers and two formulations of Navier–Stokes (streamfunction–vorticity and streamfunction–velocity) equations. These equations are discretized with the Finite Difference Method on uniform grids with numerical approximations of first- and second-orders of accuracy. The systems of equations are solved with a Modified Strongly Implicit (MSI) and a Successive Over Relaxation (SOR) solver associated with the multigrid method with a V-cycle and a Correction Scheme (CS) and a Full Approximation Scheme (FAS). The effect of the number of inner iterations of the solver, the number of grid levels problems with grid sizes of 1025×1025 points, the influence of differential equations numbers and Reynolds number up to 1000 on Central Processing Unit (CPU) time are investigated. The results show that (1) a solution of two coupled equations (Navier or Burgers) is obtained with the same efficiency multigrid textbook that occurs in the solution of only one equation (Laplace), (2) the efficiency of the multigrid method in the solution of two coupled equations (Navier–Stokes streamfunction–vorticity formulation) or only one equation (the Navier–Stokes streamfunction–velocity formulation) decreases with increasing Reynolds numbers, and (3) the poor performance of the multigrid method for solving the Navier–Stokes seems to be related to the physics of the problem and not to the type of formulation or coupling between the equations.

1. Introduction

The discretization of problems in Computational Fluid Dynamics (CFD) with the Finite Difference Method (FDM) [1] results in systems of algebraic equations of type

$$A^h \phi^h = f^h, \tag{1}$$

where A^h is a discrete *operator* that depends on the discretization scheme, ϕ^h is an approximation of the solution, and f^h is the vector of independent terms. In this method, h represents the distance between two consecutive points in the discretized field. In practice, the linear system provided by Eq. (1) may involve millions of points, making the use of direct methods

* Corresponding author. Tel.: +55 41 3361 3126; fax: +55 41 3361 3701.
E-mail addresses: cosmo@utfpr.edu.br (C.D. Santiago), machi@pq.cnpq.br (C.H. Marchi), lefraso@icmc.usp.br (L.F. Souza).

http://dx.doi.org/10.1016/j.apm.2014.10.067
0307-904X/

prohibitive in terms of CPU time and computer memory [2,3]. Moreover, iterative methods such as Gauss–Seidel, Jacobi, SOR (Successive Over Relaxation), and MSI (Modified Strongly Implicit) [4] are more effective at solving linear systems such as Eq. (1) when involving many points, although the convergence rate in fluid flow problems is almost always dependent on parameters such as the Reynolds number and the size of h.

In recent years, several studies have been conducted with the aim of improving numerical methods to solve the system shown in Eq. (1). These studies resulted in techniques that are more stable, efficient, and require less CPU time and/or computational memory to obtain a solution [5,6]. Among the most successful approaches is a multigrid method (MG) [7–10] that uses a sequence of coarser grids to accelerate the convergence of the iterative procedure, with a computational effort of order $O(N)$, in elliptic problems. For problems dominated by diffusion, the multigrid method theoretically has a convergence rate that is independent of h [12,12]; however, the same does not occur with problems dominated by advection. A more detailed description of the multigrid method can be seen in [5,8,9].

Due to its importance in actual applications, several authors [14–16] have devoted efforts to improving the convergence rate of multigrid Navier–Stokes equations, but their performance in refined grids is still a challenge for numerical analysts. Possible explanations (that motivate this work) may be related to the variation of parameters, the coupling of equations and the effect of Reynolds numbers. A detailed study on some parameters of the multigrid method and its performance in two-dimensional problems with linear and nonlinear coupling (including Navier–Stokes equations with Reynolds numbers up to 5000) can be found in [18]; preliminary results can be found in [19–21].

The aforementioned studies highlight efforts in improving the performance of the multigrid method for flow problems. However, more detailed studies of the method are required to improve its performance, primarily when involving more complex problems and high Reynolds numbers.

More recently, a new paradigm for solving Navier–Stokes equations using a streamfunction–velocity formulation was proposed in [23]. They discretized the governing equation (which is a fourth-order derivative) with a compact finite difference scheme of second-order accuracy using a stencil of nine points. The numerical solutions were obtained with a bi-conjugate gradient method in a square cavity test problem (without a multigrid) in grids up to 161×161 points for Reynolds numbers up to 10,000. They concluded that it is possible with compact discretization to obtain numerical solutions with high accuracy and efficiency. With this same formulation, Tian and Yu [24] used a new compact finite difference scheme of second-order accuracy with the multigrid method, using a five-points approach aimed at reducing the computational complexity. They obtained numerical solutions for Reynolds numbers up to 7500 adopting grids with maximum 129×129 points. No systematic studies have been conducted on the performance of a multigrid method; however, according to the authors the new scheme also provides solutions to second-order accuracy and computational efficiency. Other authors [25–26] also provided contributions to the analysis of numerical methods using Navier–Stokes equations with a streamfunction–velocity formulation, but without adopting the multigrid method. This formulation has been used because it avoids the difficulties inherent in the calculation of the boundary condition and the pressure that arises in conventional formulations: streamfunction–vorticity and primitive variable formulations [23].

The primary objective of this paper is to investigate the performance of a geometric multigrid method for five different two-dimensional problems. The specific aim is to analyze the effect of some parameters on the CPU time caused by (i) the inner iterations number of the solver, (ii) the number of grid levels, (iii) the number of grid points, (iv) the coupling of the equations and (v) the Reynolds number in the Navier–Stokes equations. Furthermore, this paper presents significant contributions in demonstrating the behavior of a multigrid in refined grids for linear and non-linear couplings equations, as well as the alternative formulations for Navier–Stokes equations that do not address the pressure–velocity coupling. It is also analyzed the performance of a multigrid for a streamfunction–velocity formulation with a second order compact finite difference scheme and a nine-points approximation scheme in a lid-driven cavity problem with refined grids. The discretization scheme presented by [24] that adopt the same formulation with a second order compact finite difference scheme with five-points discretization scheme is also analyzed in the present paper.

To achieve the primary objective, a preliminary study is performed with the 2D Laplace equation to obtain the optimal parameters of a multigrid method and use these as a reference for analyzing the results of the other equations. It is known from existing literature [1,12] that a multigrid method in a Laplace equation exhibits great performance. The analysis of the parameters is extended to two problems: Navier (linear) and Burgers (nonlinear) equations. Finally, an analysis is made of the Navier–Stokes equations in two formulations: streamfunction–vorticity [2,26] and streamfunction–velocity [23]. To our knowledge, no studies in existing literature have used a multigrid method on refined grids for a streamfunction–velocity formulation.

Section 2 presents the five problems in a Dirichlet boundary condition. In Section 3, the numerical models are presented with the discretization schemes and general aspects of the multigrid method. Section 4 presents the results and offers discussion, and Section 5 concludes the paper.

2. Mathematical models

2.1. Laplace equation

The two-dimensional Laplace equation (2D) [1], for the variable $T(x, y)$, with constant properties, is given by

$$\frac{\partial^2 T}{\partial x^2}+\frac{\partial^2 T}{\partial y^2}=0, \tag{2}$$

where x and y are the spatial coordinates. The Dirichlet boundary condition types are $T(x,1)=x$, $T(0,y)=T(x,0)=0$ and $T(1,y)=y$. The analytical solution is given by $T=xy$.

2.2. Navier equations

The governing equations of the two-dimensional linear thermoelasticity permanent problem (also called Navier equations) can be reduced to two partial differential equations written in terms of displacements [28]:

$$C_\lambda \frac{\partial}{\partial x}\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}\right)+\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}=2C_\lambda \alpha \frac{\partial T}{\partial x}+S^u, \tag{3}$$

$$C_\lambda \frac{\partial}{\partial y}\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}\right)+\frac{\partial^2 v}{\partial x^2}+\frac{\partial^2 v}{\partial y^2}=2C_\lambda \alpha \frac{\partial T}{\partial y}+S^v, \tag{4}$$

where $C_\lambda=(1+\lambda)/(1-\lambda)$, λ is the Poisson ratio, α the thermal expansion coefficient and u and v are the displacements in the coordinate directions x and y, respectively. The parameters λ and α are constants. The temperature field T is given by an analytical solution of a two-dimensional diffusion problem. The algebraic expressions for the portions of source terms S^u and S^v can be seen in detail in [18].

The boundary conditions are $u(1,y)=\beta_1\mathbf{sin}(\pi)[(e^y-1)/(e-1)]$, $v(1,y)=\beta_2 y^2$, $u(x,1)=\beta_1\mathbf{sin}(\pi)[(e^{2x}-1)/(e^2-1)]$, $u(0,y)=v(0,y)=u(x,0)=v(x,0)=0$ and $v(x,1)=\beta_2 x$. The analytical solutions (based on [18]) are $u=\beta_1\mathbf{sin}(\pi x)[(e^{2x}-1)(e^y-1)]/[(e^2-1)(e-1)]$ and $v=\beta_2 xy^2$, where $\beta_1=0.01$ and $\beta_2=1$ are parameters.

2.3. Burgers equations

The two-dimensional Navier–Stokes equations (with some simplifications) are reduced to Burgers equations (or simply to the equations of motion) as given by

$$\frac{\partial u^2}{\partial x}+\frac{\partial (vu)}{\partial y}=-\frac{\partial p}{\partial x}+\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}, \tag{5}$$

$$\frac{\partial (uv)}{\partial x}+\frac{\partial v^2}{\partial y}=-\frac{\partial p}{\partial y}+\frac{\partial^2 v}{\partial x^2}+\frac{\partial^2 v}{\partial y^2}+B(x,y,\mathrm{Re}=1), \tag{6}$$

where p is the pressure (known analytically), u and v are the velocity components in the coordinate directions x and y, respectively, and B is a source term. The variables p and B are provided in [29]. The boundary conditions are $u(x,1)=16(x^4-2x^3+x^2)$, $u(0,y)=u(1,y)=u(x,0)=0$ and $v(0,y)=v(1,y)=v(x,0)=v(x,1)=0$. The analytical solutions (based on [29]) are $u=8(x^4-2x^3+x^2)(4y^3-2y)$ and $v=-8(4x^3-6x^2+2x)(y^4-y^2)$.

Due to their nonlinearity, the Burgers equations constitute an important problem-test for numerical methods. A literature review of research on this problem is used as a benchmark and can be found in [30].

2.4. Navier–Stokes – streamfunction–vorticity formulation

The two-dimensional Navier–Stokes equations for an incompressible viscous flow, written in terms of the streamfunction–vorticity (ψ,ω) [31], are given by

$$\frac{\partial^2 \psi}{\partial x^2}+\frac{\partial^2 \psi}{\partial y^2}=-\omega, \tag{7}$$

$$\frac{\partial^2 \omega}{\partial x^2}+\frac{\partial^2 \omega}{\partial y^2}=\mathbf{Re}\left(\frac{\partial \psi}{\partial y}\frac{\partial \omega}{\partial x}-\frac{\partial \psi}{\partial x}\frac{\partial \omega}{\partial y}\right), \tag{8}$$

where ω is the vorticity given by $\omega=\partial v/\partial x-\partial u/\partial y$, ψ the streamfunction and Re is the Reynolds number. Eqs. (7) and (8) are Poisson-like equations and are coupled by the variables ψ and ω.

The velocity components are defined in terms of streamfunction as follows:

$$u=\frac{\partial \psi}{\partial y} \quad \text{and} \quad v=-\frac{\partial \psi}{\partial x}. \tag{9}$$

The boundary conditions of Eqs. (7) and (8) are $\psi=0$ and ω as obtained by a numerical approximation of Eq. (7) at the boundaries.

2.5. Navier–Stokes – streamfunction–velocity formulation

The Navier–Stokes equations for an incompressible flow and a Newtonian fluid (written in the form of Eqs. (7) and (8)) can be reduced to a partial differential fourth-order equation in terms of the streamfunction ψ and the velocities u and v, called the streamfunction–velocity formulation (ψ, v) [23], and given by

$$\frac{\partial^4\psi}{\partial x^4} + 2\frac{\partial^4\psi}{\partial x^2\partial y^2} + \frac{\partial^4\psi}{\partial y^4} = Re\left[u\left(\frac{\partial^3\psi}{\partial x^3} + \frac{\partial^3\psi}{\partial x\partial y^2}\right) + v\left(\frac{\partial^3\psi}{\partial x^2\partial y} + \frac{\partial^3\psi}{\partial y^3}\right)\right], \tag{10}$$

where u and v are calculated by Eq. (9) and Re is the Reynolds number. The primary advantage of this formulation is that the boundary conditions of the streamfunction–velocity are generally known and easily implemented computationally [24].

In the lid-driven cavity problem, the boundary conditions are $u = 1$ and $v = \psi = 0$ in the upper boundary and $u = v = \psi = 0$ in other boundaries.

3. Numerical models

This section discusses schemes used to discretize the governing equations, the multigrid method and the resolution of the system of equations.

The domain was discretized into a uniform structured grid. The governing equations were approximated by the Finite Difference Method [1]. The calculations were considered in a cartesian coordinate system partitioned into a number of nodes (or points) given by $N = N_xN_y$, where N_x and N_y are the number of points in the directions x and y, respectively, including the contours.

Each point is defined in the computational grid as $(x_i, y_j) = ((i-1)h, (j-1)h)$, where $i = 1, \ldots, N_x$, $j = 1, \ldots, N_y$ and $h = 1/(N_x - 1) = 1/(N_y - 1)$; h is either the size of the grid element in the x and y directions or the distance between two consecutive points, as shown in Fig. 1.

The basic idea of the multigrid method is to smooth the error in a coarse grid and approximate the solution in a fine grid by using a grid hierarchy. For linear problems, Brandt [7] recommends a CS scheme (Correction Scheme); for nonlinear problems, the FAS scheme (Full Approximation Scheme) is indicated. The two schemes can be associated with V-, W- and F-cycle methodologies and a Full Multigrid (FMG). Details can be found in [8,9].

A multigrid V-cycle is a computational process that goes from the finest grid down to the coarsest grid to be used and back from the coarsest up to the finest. Pre-and post-smoothing is normally performed at each grid level of a V-cycle.

Smoothing is equivalent to a few iterations of an iterative technique (e.g., Gauss–Seidel and Jacobi methods, which are called smoothers in multigrid literature) that removes high-frequency error components faster than low-frequency components. With transfer operators called restriction and prolongation operators, the information is transferred between the grids.

In the CS scheme, the system of equations is approximated on only the finest grid with a zero initial estimate. The residue is then transferred to the subsequent coarser grid, where the residual equation is solved with few iterations (pre-smoothing). This procedure is repeated until reaching the coarsest specified grid, when the error is then interpolated to correct the approach (post-smoothing) in subsequently finer grids [7,8].

In an FAS scheme, the procedure is analogous, but the solution approach is also transferred to the subsequent coarser grid in addition to the residue. In the coarser grid, the correction is interpolated and used to approximate the solution in

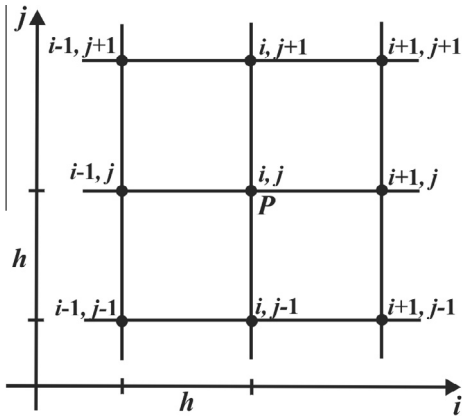

Fig. 1. Computational stencil.

subsequent finer grid. The main difference between the two schemes is that the FAS scheme is not solved to the equation residual, but to equations with approximations to the solution in the coarse grid.

An inner iteration is performed by the solver at each level of grid (as a pre- or post-smoothing) and will be denoted by η; the outer iteration is a full multigrid V-cycle. This research adopts the geometric multigrid method with V-cycle, CS and FAS schemes, with a coarsening ratio of 2.

The CS scheme has been implemented to the two linear problems (Laplace and Navier equations), while the FAS scheme has been implemented for all five problems. Multigrid algorithms for one and two equations can be seen in [18].

3.1. Laplace, Navier and Burgers equations

The second derivatives and cross-derivatives were approximated with a second-order Central Differencing Scheme (CDS). The first derivatives of Eqs. (5) and (6) were approximated with a first-order Upwind Differencing Scheme (UDS). This discretization scheme with a five-point approximation results in a system of algebraic equations that can be generically represented by

$$a_{P,p}\phi_{i,j} = a_{P,n}\phi_{i,j+1} + a_{P,s}\phi_{i,j-1} + a_{P,w}\phi_{i-1,j} + a_{P,e}\phi_{i+1,j} + b_P^{\phi}, \tag{11}$$

where the coefficient matrix has five diagonals and is valid for points within the discretized domain. The variable ϕ represents the physical property involved in the numerical approximation. The term $a_{P,p}$ is the coefficient of the center point, $a_{P,n}$, $a_{P,s}$, $a_{P,e}$ and $a_{P,w}$ are the coefficients north, south, east and west that connect with the point P, and b_P^{ϕ} is the independent term or right–hand side (RHS) term associated with the point P. For Eqs. (2)–(6), the coefficients for the boundaries are given by $a_{P,e} = a_{P,w} = a_{P,n} = a_{P,s} = 0$, $a_{P,p} = 1$ and $b_C^{\phi} = \phi_C$, where ϕ_C represents the known value of the property at each contour point. The expressions for the internal coefficients can be seen in [18]. The discretization of Eqs. (3) and (4) results in two systems of coupled equations whose general form is analogous to Eq. (11).

The nonlinear terms of Eqs. (5) and (6) were linearized at the point (x_i, y_j) as in [13]. For example, in the x-direction, $(\partial\phi^2/\partial x)_{i,j} = (\phi'_{i,j}\phi_{i,j} - \phi'_{i-1,j}\phi_{i-1,j})/h$ if $\phi_{i,j} > 0$, and $(\partial\phi^2/\partial x)_{i,j} = (\phi'_{i+1,j}\phi_{i+1,j} - \phi'_{i,j}\phi_{i,j})/h$ if $\phi_{i,j} < 0$; the superscript "′" indicates that the values are known from the previous iteration. The expressions can be similarly obtained in the y-direction. In this case, the discretization results in two systems (also coupled by velocity) in which each equation has the form of Eq. (11). The term sources are expressed as a function of the pressure gradient [29]; in the case of the v component, the term also appears as B. The resulting equations are $b_P^u = -\partial p/\partial x$ and $b_P^v = -(\partial p/\partial y + B)$.

The systems of algebraic equations obtained for the Laplace, Navier and Burgers equations of the form of Eq. (11) were solved with the geometric multigrid method associated with the MSI solver and a zero initial estimate. In the multigrid V-cycle, the residues and solution approximations were transferred by direct injection into the restriction and by bi-linear interpolation in prolongation [9].

The iterative process was terminated when the maximum average norm [13] at the residue of each variable ϕ, calculated at the end of each V-cycle, was lower than 10^{-12}.

3.2. Navier–Stokes equations

The streamfunction–vorticity formulation (ψ, ω) was also approximated with the CDS, resulting in equations analogous to Eq. (11). The coefficients of Eq. (11) for ψ and ω are the same and are given by $a_{P,e} = a_{P,w} = a_{P,s} = a_{P,n} = 1$ and $a_{P,p} = 4$ for the internal points. The conditions are analogous at the boundaries.

Approximations of the source terms of Eqs. (7) and (8) are given, respectively, by

$$b_P^{\psi} = h^2\omega_P \quad \text{and} \quad b_P^{\omega} = -0.25\mathrm{Re}[(\psi_{i,j+1} - \psi_{i,j-1})(\omega_{i+1,j} - \omega_{i-1,j}) - (\psi_{i+1,j} - \psi_{i-1,j})(\omega_{i,j+1} - \omega_{i,j-1})]. \tag{12}$$

Similar to the values obtained for the Burgers equations, the discretization of the streamfunction–vorticity formulation results in two coupled systems of pentadiagonal algebraic equations that are solved to ψ and ω in the internal points in the domain.

The systems of equations were solved with a geometric multigrid method using an FAS scheme associated with an SOR solver and a zero initial estimate for all Reynolds numbers. In the V-cycle, the residues and approximate solutions were transferred by direct injection in restricting and by bi-linear interpolation in prolongation.

In each inner iteration, the vorticity was sub-relaxed with a parameter $\lambda \in (0, 1)$. The approximate solution of the system of equations in the vorticity was used in RHS to obtain an approximation of the streamfunction during the iterative process

The vorticity was calculated in the boundaries in only the finest grid with approximations of Eq. (9) by using Jensen's formula [32], where $\omega_c = (7\psi_c - 8\psi_1 + \psi_2)/2h^2 - 3\mathbf{u_c}/h$; subscript "$c$" refers to the points on contour, 1 refers to the points adjacent to the contour, 2 refers to the second line of points adjacent to the contour, $\mathbf{u_c}$ refers to the velocity of the contour, with its value being equal to 1 on the moving contour and 0 on the stationary contour. The velocity components were calculated in post-processing with second-order approximations by using Eq. (9), as in [23].

The iterative process was terminated in the finest grid when $\mathbf{max}(\|R_{\psi}^k\|_{\infty}/\|R_{\psi}^1\|_{\infty}, \|R_{\omega}^k\|_{\infty}/\|R_{\omega}^1\|_{\infty}) \leqslant \mathbf{10^{-7}}$, where R^k is the residual at iteration k, R^1 is the residue in the first iteration and the subscript indicates the variable.

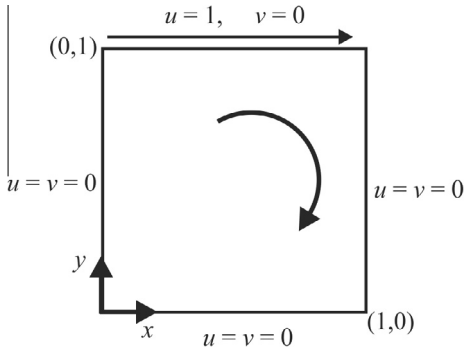

Fig. 2. Boundary conditions of the velocities in lid-driven cavity for the Navier–Stokes equations.

Equation (10) was discretized with a second-order compact finite difference scheme [23] involving nine points. In this scheme, the point P is connected to its eight nearest neighbors, resulting in a coefficient matrix with nine diagonals whose system of equations is written in terms of streamfunction as follows:

$$a_{P,sw}\psi_{i-1,j-1} + a_{P,nw}\psi_{i-1,j+1} + a_{P,s}\psi_{i,j-1} + a_{P,w}\psi_{i-1,j} + a_{P,p}\psi_{i,j} + a_{P,e}\psi_{i+1,j} + a_{P,n}\psi_{i,j+1} + a_{P,se}\psi_{i+1,j-1} + a_{P,ne}\psi_{i+1,j+1} = b_P, \quad (13)$$

where $a_{P,e} = a_{P,w} = a_{P,n} = a_{P,s} = -8$, $a_{P,ne} = a_{P,se} = a_{P,nw} = a_{P,sw} = 1$ and $a_{P,p} = 28$. The coefficients on the contours are analogous to previous cases.

The coefficients $a_{P,ne}$, $a_{P,se}$, $a_{P,nw}$ and $a_{P,sw}$ connect the point P to its neighbors northwest, northeast, southeast and southwest, respectively, as shown in Fig. 1. The resulting expression for the RHS is

$$b_P = 0.5\mathrm{Re}h^2\left\{v_{i,j}(u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1}) - u_{i,j}(v_{i+1,j} + v_{i-1,j} + v_{i,j+1} + v_{i,j-1})\right\} + 3h(u_{i,j-1} + u_{i,j+1} + v_{i+1,j} + v_{i-1,j}). \quad (14)$$

It can be verified that Eq. (14) depends on the term $\mathrm{Re}h^2$ and the product of the velocity components. This indicates that the system of algebraic equations that arises in this formulation is highly nonlinear.

In this formulation, a system of equations was only solved with the multigrid method where the calculations began with $\psi = 0$. In each inner iteration, the SOR solver was applied with a relaxation parameter of $\lambda \in (0, 1.4)$. In this formulation, the residue and the approximated solution were transferred by full weighting in restriction.

The work [24] shows a discretization scheme for the stream-function in Eq. (11) where the coefficients are $a_{P,e} = a_{P,w} = a_{P,n} = a_{P,s} = 12$ and $a_{P,p} = 48$. The resulting scheme for the RHS term is:

$$\begin{aligned} b_P = {} & 6h(v_{i+1,j} - v_{i-1,j} - u_{i,j+1} + u_{i,j-1}) + 0.5h\big[(v_{i+1,j+1} - v_{i-1,j+1} - v_{i-1,j-1} + v_{i+1,j-1} - 2(v_{i+1,j} - v_{i-1,j})) \\ & - (u_{i+1,j+1} + u_{i-1,j+1} - u_{i-1,j-1} - u_{i+1,j-1} - 2(u_{i,j+1} - u_{i,j-1}))\big] + \mathrm{Re}h^2\left\{v_{i,j}(u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1})\right. \\ & \left. - u_{i,j}(v_{i+1,j} + v_{i-1,j} + v_{i,j+1} + v_{i,j-1})\right\}. \end{aligned} \quad (15)$$

The resulting system obtained with the present approximations was also solved by a SOR solver, with $\lambda \in (0, 2)$. The residue and the approximated solution in each multigrid level were transferred by restriction and bilinear interpolation, respectively.

The velocity components in Eq. (9) were evaluated using finite difference approximations of fourth-order accuracy [23]. With this methodology, Eq. (9) yields two triangular linear systems that were solved with the TDMA method (Tridiagonal Matrix Algorithm) [1] at the end of each V-cycle, after the stream-function ψ had been calculated with Eq. (13). The convergence with the five and nine-points approximation schemes was similarly verified to the previous case using the streamfunction.

To analyze the parameters of the multigrid method in the Navier–Stokes equations, a classic square cavity test problem was used in which a recirculating incompressible flow was induced by sliding the upper contour ($y = 1$) left to right [2,3,26], as shown in Fig. 2.

This problem is extensively used for testing and verification of numerical methods of computer codes for the Navier–Stokes equations [32] due to the simplicity of its configuration and geometry, even with discontinuities on two corners. The results for high Reynolds numbers (Re) can be seen in [27,32]. For moderate Reynolds numbers, many results are available in existing literature, obtained with various types of numerical procedures.

4. Results and discussion

The computational codes were written in FORTRAN 95 with double precision using the Compaq Visual Fortran 6.6 compiler. The simulations were performed on a computer with an Intel Core 2 Duo 2.66 GHz processor with 4 GB of RAM.

Approximately 1000 simulations were performed to evaluate the influence on the CPU time, the parameters of the multigrid method, the Reynolds numbers, the number of equations and the parameters of the relaxation solver.

The results and their numerical solutions were analyzed according to the characteristics of each parameter. The accuracy of the solutions were checked and compared with solutions available in existing literature and can be viewed in detail in [18], including results for Re = 5000. This text presents the most relevant results for parameters of the multigrid method.

The optimal value of a parameter was used when the solution of the problem was obtained in less CPU time for the fixed values of the other parameters. It is denoted by η_{opt} and L_{opt} (the optimal number of inner iterations in the solver and the optimal number of grid levels), respectively.

4.1. Inner iterations (η)

For the Laplace, Navier and Burgers problems, the effect of the parameter η was examined in the grids $N = 257 \times 257$, 513×513 and 1025×1025 with the maximum number of levels. Fig. 3 illustrates the influence of the number of inner iterations (η) on the CPU time for the CS and FAS schemes.

Each curve indicates the value of η with the "star" symbol that results in less CPU time. Fig. 3(a) shows the results for the Laplace and Navier problems with the CS scheme; it should be noted that the minor CPU time was obtained with two inner iterations, namely $\eta_{opt} = 2$, at any grid.

Fig. 3(b) shows the results with the FAS scheme, where the results are also observed for the Burgers equations. Note that the optimal smoothing number is kept at 2 for the Navier equations. For the Laplace equation, however, it was obtained in three grids $\eta_{opt} = 8$. For the Burgers equations, it was obtained at $\eta_{opt} = 5$ for three different grid sizes. In all examined cases, it should be noted that when the value of η decreases or increases compared to η_{opt}, the CPU time increases. This increase can be significant, depending on the value used for η.

This analysis also found that the increase in CPU time observed for the problem with one equation (Laplace) is equal or very similar to that observed in problems with two equations (Burgers and Navier), as shown in Fig. 3.

A CS scheme was used in [1] in a two-dimensional Laplace equation with 129×129 grid points and Gauss–Seidel smoothing. They found $\eta_{opt} = 3$ or 4 as parameters that minimized the amount of CPU time. In [34], tests were performed with three solvers in the same equation (for the MSI solver used in this work); they found that $\eta_{opt} = 1$ or 2 for the CS scheme and $\eta_{opt} = 4$ in the case of the FAS scheme. These results show that the coupling equations, compared with the Laplace equation, do not influence the optimum number of inner iterations.

The analysis of the number of inner iterations in the two formulations of the Navier–Stokes equations was performed on grids $N = 65 \times 65$, 129×129, 257×257, 513×513 and 1025×1025 points, each of which with Re = 100, 400 and 1000. Fig. 4 illustrates the effect of the parameter η on the CPU time in the two formulations.

With the velocity–stream-function formulation, it was found that the best results in terms of CPU time was obtained with an iteration in the finest grid of $\eta = 1$ and $\eta = 20$ in the coarsest grid specified; therefore, the parameter η was analyzed for the intermediate grids. It can be seen in Fig. 4 that η_{opt} is not the same in all grids; it increases with the number of points for any Reynolds number. For $\eta < \eta_{opt}$ (see Fig. 4(a)), the problem did not converge. The CPU time curves for Re = 100 exhibited a growth pattern similar to those observed for the coupled Navier and Laplace equations (see Fig. 3). In this analysis, it was also noted that the CPU time was very sensitive to the Reynolds number and the number of grid points, especially when Re > 100.

With the velocity–stream-function formulation adopting the five-points discretization scheme, the CPU time reaches great values for all grid sizes and Reynolds numbers simulations. The increase in CPU time comparing with the nine-points discretization scheme can be explained by the stronger interaction of the terms that appears in the RHS of Eq. (15). Many simulations were performed with the Gauss–Seidel, SOR and MSI solvers, however, the best performance was obtained with the SOR solver with $0 < \lambda < 2$, and some results are discussed in this paper. The analysis of the number of internal iterations

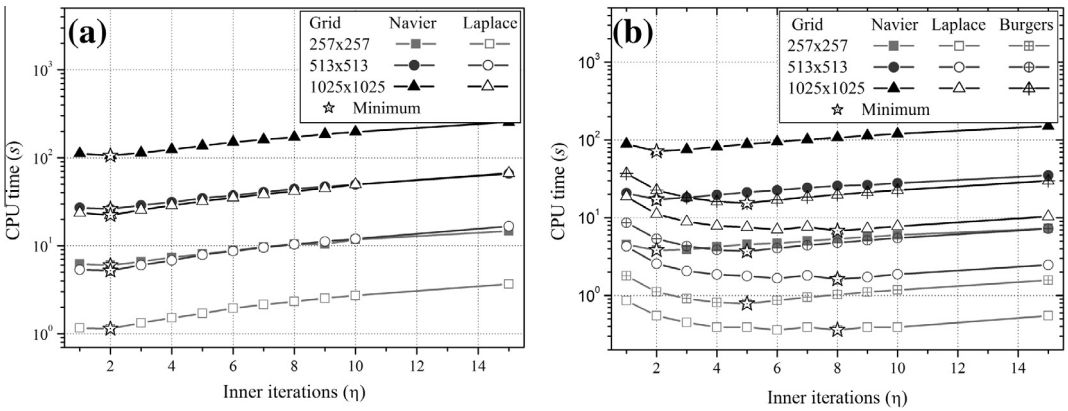

Fig. 3. Effect of the number of inner iterations (η) on the CPU time in different grids: (a) CS scheme; (b) FAS scheme.

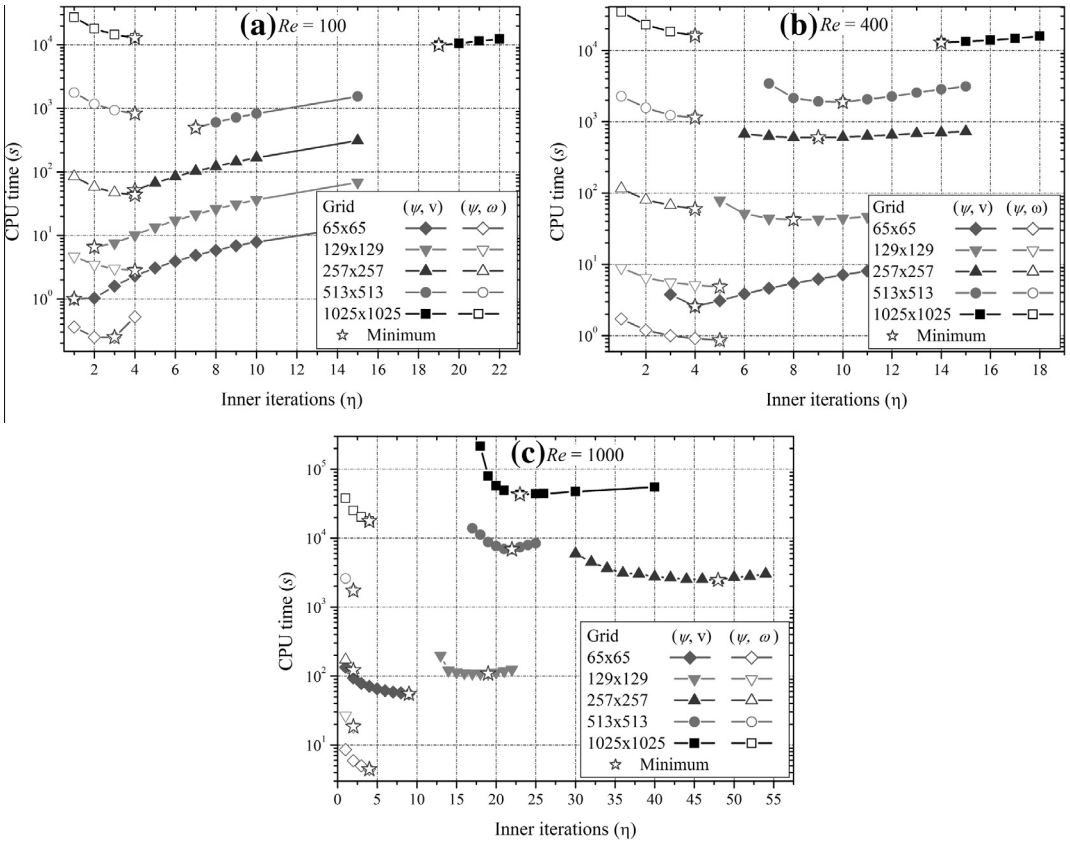

Fig. 4. Effect of the number of inner iterations (η) on the CPU time for the Navier–Stokes equations in two formulations for different grids.

it was verified that the larger the Reynolds number the smaller the parameter and the relaxation solver and the $\eta_{opt} \to 1$. This fact was also observed in nine-points discretization scheme with the vorticity–stream-function formulation. For example, with the grid 129×129 for Re = 100 it was obtained $\eta_{opt} = 5$ with $\lambda = 1.9$; in the same grid changing the value of Reynolds number to Re = 400 or 1000, it was obtained $\eta_{opt} = 1$ with $\lambda = 1.7$ and 0.9, respectively. For Reynolds number Re = 1000, if $\eta > \eta_{opt}$ or $\lambda > 0.9$ the solver does not converge to the solution. Tian and Yu [24] also found the dependence of the relaxation parameter with the Reynolds number.

Tian and Yu [24] analyzed the accuracy of numerical solutions using grids with up to 129×129 points, and the information about the CPU time has been reported for only few cases. They used a Samsung r23 plus computer with 2 GB of memory in all simulations. The results obtained with the five and nine-points compact approximation schemes of the present study were carried out on a Intel Core i7, 2.8 GHz processor and 4 GB of RAM computer. Although the comparison is not ideal, some results in Table 1 with the values and CPU time obtained in this work, with the five and nine-points compact approximation schemes and the CPU time of [24].

Table 1 shows that the CPU time increases with the number of grid points with the five-points approximation scheme faster than with the nine-points approximation scheme of Tian and Yu. For Reynolds number $Re \leqslant 400$, with the 65×65

Table 1
CPU time and η_{opt} for velocity–stream-function formulation with five and nine-points approximation schemes and CPU time with the Tian and Yu [24] scheme.

Grid	Re	(ψ, v)				[24]
		Nine-points		Five-points		
		η_{opt}	CPU(s)	η_{opt}	CPU(s)	CPU(s)
65×65	100	1	1.00	1	8.12	97.44
	400	4	2.59	1	26.05	89.20
	1000	9	55.25	1	565.12	138.78
129×129	100	2	6.55	5	436.64	1687.73
	400	8	42.25	1	878.33	–
	1000	19	108.55	1	4630.39	–

points, the CPU time obtained with the present scheme is much smaller than the one obtained with the method of [24], this is also true for the 129 × 129 points with Re = 100. The nine-points discretization scheme showed significant gain in CPU time for all cases tested here. Due to the high computational cost in refined grid with high Reynolds number grids (e.g., CPU time in the 257 × 257 points simulation with multigrid scheme and Re = 100 was more than 4 h), the remaining analyzes of parameters are not given here. More detailed analyzes of the compact five-points approximation scheme for the streamfunction–velocity formulation, and other high-order approximation schemes, including the high-order exponential scheme [35] is in development by the present authors and will be addressed in a future work.

With the streamfunction–vorticity formulation, $\eta_{opt} = 4$ was obtained for Re = 100, with the exception of the 65 × 65 grid with $\eta_{opt} = 3$. For Re = 400, $\eta_{opt} = 4$ was obtained on grids 257 × 257, 513 × 513 and 1025 × 1025 points. For Re = 400, $\eta_{opt} = 5$ on grids 65 × 65 and 129 × 129 points; in this case, the difference in CPU time for $\eta = 4$ was approximately 5%. For Re = 1000, $\eta_{opt} = 4$ was obtained on grids 65 × 65 and 1025 × 1025 points, and $\eta_{opt} = 2$ on grids 129 × 129, 257 × 257 and 513 × 513 points. In all grids and Reynolds numbers, the CPU time decreased to η_{opt}. However, it should be noted that for $\eta > \eta_{opt}$ the problem diverged (except in the 65 × 65 grid with Re = 100), as shown in Fig. 4(a).

4.2. Number of grid levels (L)

The number of grid levels (L) was considered the optimal number of inner iterations found in the previous section because the intention was to optimize the CPU time for a given number of levels.

For each grid, the problem was solved with a number of levels of grid L such that $1 \leqslant L \leqslant L_{max}$, where $L = 1$ is the singlegrid (SG) case and $L = L_{max}$ is the maximum possible number of grids in a V-cycle, with the coarser grid having only one internal point. For example, if $N = 257 \times 257$, the grids are 257 × 257, 129 × 129, 65 × 65, 33 × 33, 17 × 17, 9 × 9, 5 × 5 and 3 × 3 points; in this case, $L_{max} = 8$.

The objective of this analysis is to show the performance of the multigrid in relation to the number of levels for the five problems. Fig. 5 shows the influence of L on the CPU time for three problems with CS and FAS schemes. Fig. 5(a) shows curves for the Navier and Laplace equations to the CS scheme. Fig. 5(b) shows the curves of the FAS scheme and includes the Burgers equations. The symbol "star" indicates the L that resulted in a lower CPU time on each curve. It should be noted that the behavior of each curve and L_{opt} resembles both the Navier equations with CS and FAS schemes and the Burgers equations (FAS) in any grid size.

For both schemes (CS and FAS) in the three grids, the optimal value of the number of levels may be regarded as $L_{opt} = L_{max}$. In some cases, the $L < L_{opt}$ CPU time increases significantly, but for $L > L_{opt}$ it is practically the same. For example, in the finest grid with the FAS scheme and Navier equations, the difference in CPU time between L_{max} and L_{opt} is less than 0.3%; in the case of the Burgers equations, this difference is 0.1%.

The results obtained as to the number of levels in the Laplace equation are consistent with existing literature: [34] recommended the use of $L = L_{max}$; [1] found that the computational effort using $L = 4$ or 5 is practically the same as $L = L_{max}$ in a grid with $N = 129 \times 129$ points; [36] suggested no less than four levels in a two-dimensional advection–diffusion problem. These analyses show that the parameter number of levels (L) is not modified in problems with two equations using FAS and CS schemes.

In the Navier–Stokes equations, the analyses were made with grids of $N = 129 \times 129$, 257 × 257 and 513 × 513, with Re = 100, 400 and 1,000 for each problem size. The effect of the number of levels (L) on the CPU time is shown in Fig. 6. In the streamfunction–velocity formulation, the CPU singlegrid time was too high for some grids and was not presented.

In the streamfunction–velocity formulation for Re = 100, Fig. 6(a) shows that in an analysis of η for any N, the curves demonstrate similar behavior to the previous problems in Fig. 5. The optimal number of levels was obtained with the multigrid

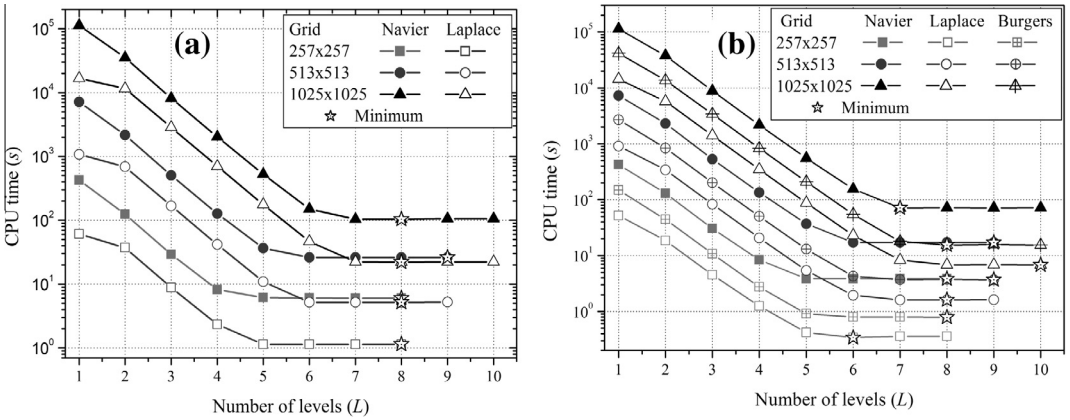

Fig. 5. Effect of number of levels (L) on the CPU time: (a) CS scheme; (b) FAS scheme.

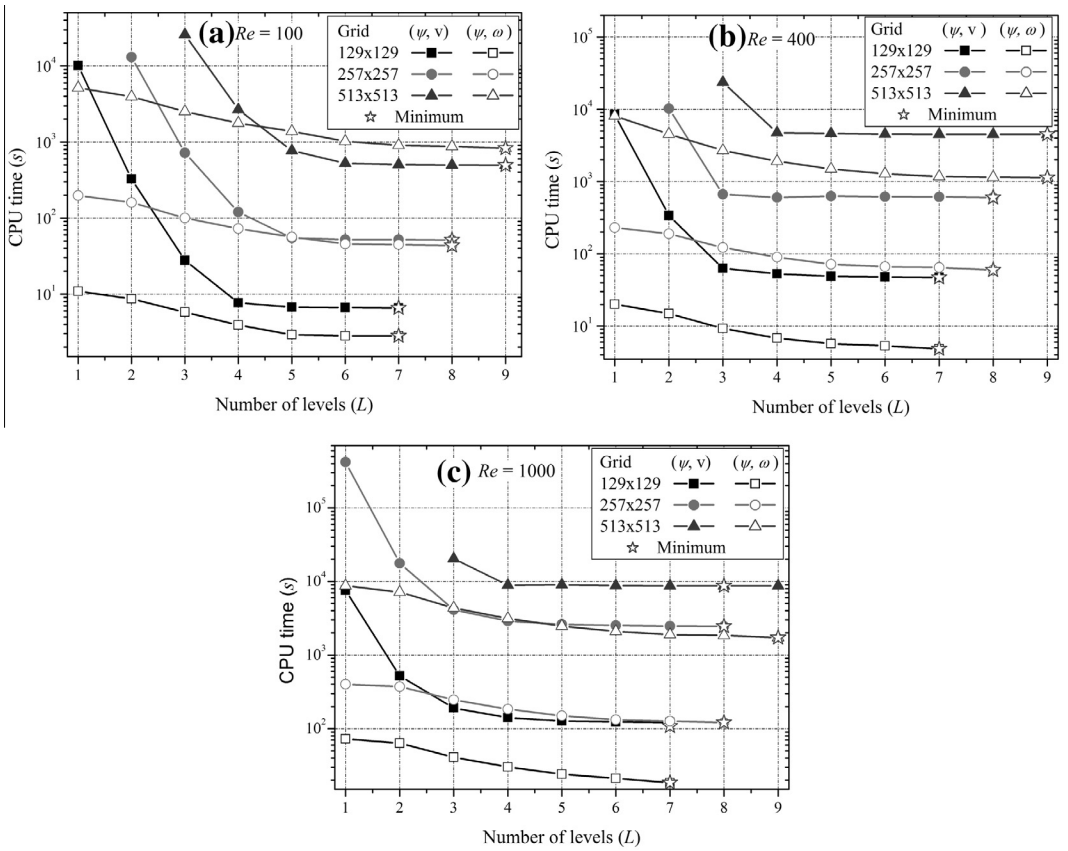

Fig. 6. Effect of number of levels on the CPU time for the Navier–Stokes equations.

covering all grids, i.e., $L_{opt} = L_{max}$. However, the difference between L_{opt} and $L_{max} - (1a3)$ is on average 1.5% for any N. For example, with the 257×257 grid, the shortest CPU time was obtained with the maximum number of levels, i.e., $L_{opt} = L_{max}$ with 51.45 s, and the CPU time for $L_{max} - 3$ was 52.11—a variation of 1.3%. When $L < L_{max} - 3$, the CPU time increases significantly.

For Re = 400 and 1000 (Fig. 6(b) and (c)), the shortest CPU time was obtained with the multigrid covering all grids, i.e., $L_{opt} = L_{max}$. In this case, there was also a small variation as it reduced the number of levels to a certain L, being practically stable when $L > L_{max} - (1a4)$. The only exception was observed when Re = 1000 in the 513×513 grid, where $L_{opt} = L_{max} - 1$ was obtained. For $N = 129 \times 129$, it is apparent that the shortest CPU time occurred with the maximum number of levels, i.e., $L_{opt} = L_{max}$.

With the streamfunction–vorticity formulation the results presented in Fig. 6 show similar behavior in the three grid sizes, independent of the Reynolds number used. The smallest CPU time was obtained with the multigrid visiting all levels grid, i.e., $L_{opt} = L_{max}$ in any case. For $N = 513 \times 513$ with Re = 100 and Re = 1000, the variation of CPU time between $L_{max} - (1a3)$ and L_{opt} is relatively small at 9.5% and 10%, respectively.

This analysis qualitatively shows that the parameter number of levels (L) in the Navier–Stokes equations, when written in terms of streamfunction–velocity with Re = 100, exhibit excellent agreement when compared with the results shown in Fig. 5(b). These results also agree with other authors [1,33,36]. However, for other Reynolds numbers and the streamfunction–vorticity formulation, there is a slight difference in the curves shown in Fig. 5.

4.3. Size of problem (N)

In analyzing the influence of the size of the problem on the CPU time, the optimal number of inner iterations (η_{opt}) and the optimal number of grid levels (L_{opt}) are considered to be those obtained in the previous sections.

For the Laplace, Navier and Burgers equations, this analysis considers a grid size of $N = 5 \times 5$ to be the highest supported by the computer physical memory used ($N = 2049 \times 2049$). For comparison, Fig. 7 shows the results obtained with the singlegrid method (SG) with an MSI solution to the three problems.

Coarse grids ($N < 33 \times 33$ points) obtained CPU times very close to zero in both the multigrid and singlegrid methods. In this case, a methodology was adopted based on the average time to obtain the CPU time to eliminate the maximum possible

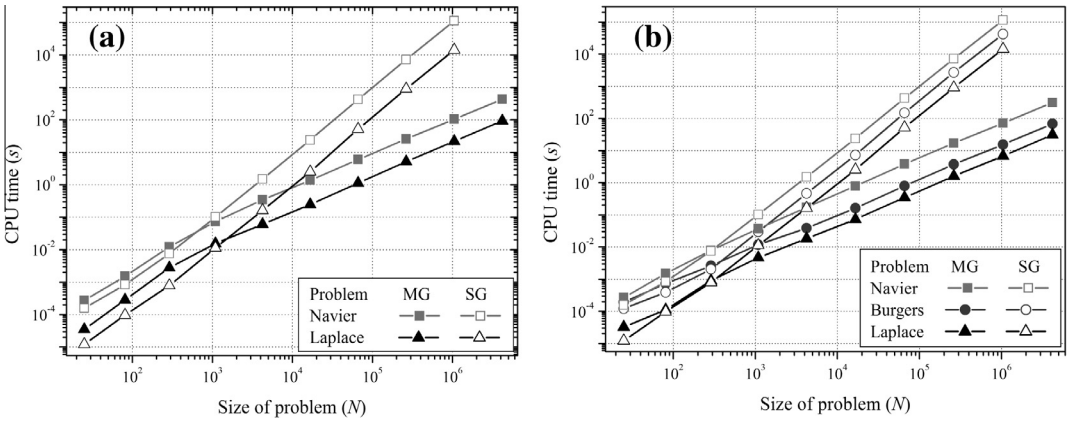

Fig. 7. Effect of problem size on CPU time for Laplace, Navier and Burgers equations: (a) CS scheme; (b) FAS scheme.

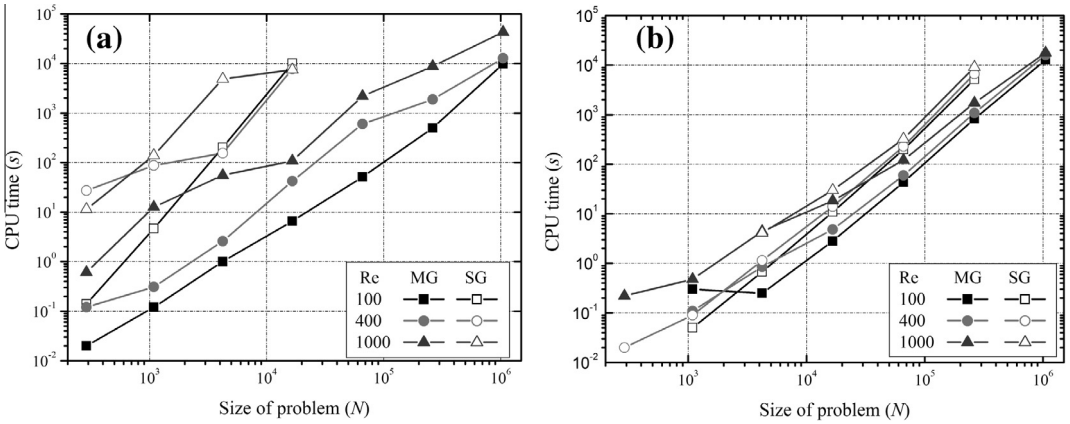

Fig. 8. Multigrid vs. singlegrid performance for both formulations: (a) streamfunction–velocity formulation; (b) streamfunction–vorticity formulation.

error due to the measurement uncertainty function TIMEF. Thus, for a given grid with a simulated CPU time of less than 10 s, an outermost cycle was adopted for the program to perform the number of simulations required to achieve 10 s or more. The simulation time for this grid is the average time spent in all simulations in this grid. For example, in the Navier problem with a CS scheme for a 5×5 grid (simulated with the multigrid method), 38,175 simulations were required to achieve 10.02 s, resulting in an average time of 0.000262 s for a simulation.

In the present analysis, the multigrid method was very efficient; the greater the number of points in the grid, the greater the advantage of the multigrid method over the singlegrid method—this was also observed in [13]. This property can also be seen in Fig. 7, where the CPU time of the multigrid method is shorter than that of the singlegrid method beginning from $N = 33 \times 33$ in the CS scheme and from $N = 17 \times 17$ in the FAS scheme. A linear increase in CPU time with N is also evident from these grids. It is noteworthy that in the $N = 513 \times 513$ grid with the FAS scheme, the multigrid method is approximately 1600 times faster than the singlegrid for the Navier equations and 2702 times faster for the Burgers equations.

For the problems with two equations (Burgers and Navier), Fig. 7(a) and (b) show that the multigrid CPU time exhibits linear growth, similar to the problem with only one equation (Laplace). The CPU time observed for the Navier problem in the finest grid was about four times greater than the time observed for the Laplace problem in the same grid. This increase in CPU time is related to the complexity of the mathematical model.

Based on the results obtained in this analysis, it should be noted that the number of equations, their complexity, the coupling of variables, and their (non)linearity have very little influence on the performance of the multigrid method.

For the Navier–Stokes equations in very coarse grids, it was not possible in some cases to obtain solutions to Re = 400 or 1000. In simulations without the multigrid method, the CPU time is too great when N was large; in this case, the present analysis was limited to problems of sizes $N = 17 \times 17$ up to $N = 129 \times 129$ in the streamfunction–velocity formulation and up to $N = 513 \times 513$ in the streamfunction–vorticity formulation. Fig. 8 presents the curves of the CPU time for both the multigrid and singlegrid methods.

The greatest advantage of the multigrid method (with respect to singlegrid) is most evident in the streamfunction–velocity formulation when the nine-points approximation scheme is adopted. For example, with Re = 100 and 400 in the 65×65

Table 2
Exponent (p) of Eq. (15) for Laplace, Navier and Burgers equations with MSI solver and Navier–Stokes with SOR solver.

Equations	Re		Singlegrid	Multigrid	
				FAS	CS
Laplace			2.06	1.08	1.06
Navier			2.01	1.15	1.05
Burgers			1.92	1.06	
Navier–Stokes	(ψ, v)	100	2.8	1.6	
		400	1.3	1.5	
		1000	2.3	1.3	
	(ψ, ω)	100	2.2	2.0	
		400	2.0	1.7	
		1000	1.9	1.5	

grid, it was found that the multigrid method was 203 and 60 times faster than the singlegrid, respectively, while these values were 1543 and 185 in the 129×129 grid, respectively. With Re = 1000 in the 129×129 grid, the multigrid was 69 times faster than the singlegrid. Fig. 8 shows that the CPU time curves on the multigrid tend to be much closer when the grid is refined. It was also found that the performance decreased for higher Reynolds numbers. These results are consistent with results of other authors [13,15], who also observed a reduction in multigrid method performance (in relation to singlegrid) with increasing Reynolds numbers. It can be seen in Fig. 8(a) that the computational effort is proportional to the size of the problem. When it doubles, the number of grid elements in each direction (disregarding the contours, i.e., of 128×128 for 256×256) quadruples the number of elements; for Re = 100, the CPU time is 8 times greater. When the number of grid elements changes from 256×256 to 512×512, the CPU time is 9.6 times greater. At Re = 400, the factors are 14 and 7.5, and for Re = 1000, 22.7 and 3.6, respectively.

In the streamfunction–vorticity formulation, Fig. 8(b) shows that the performance of the multigrid method was not significantly better than the singlegrid. It can be seen that for $N > 65 \times 65$, the curves are close to each other and have almost the same inclination. The advantage of the multigrid in relation to the singlegrid is 4 times faster in the 129×129 grid with Re = 100; in the 513×513 grid, the efficiency rises to 6 times faster. For Re = 1000 in the 513×513 grid, the multigrid method is 5 times faster than the singlegrid. These results are very close to those presented in [2], who used the same formulation; however, the results in the present study were obtained with very refined grids.

In comparing the two formulations, the results show that the streamfunction–velocity formulation CPU time multigrid method is superior to the streamfunction–vorticity formulation. This occurs because the reference for each formulation is its own singlegrid solution; note that the streamfunction–velocity formulation CPU time singlegrid solution is much higher than that of the streamfunction–vorticity formulation.

Fig. 7 shows that the multigrid method exhibits superior performance when it is used to solve the two-dimensional Laplace, Navier and Burgers equations. It is also known [13] that the convergence rate of the multigrid method is not proportional to the number of grid points when used to solve two-dimensional Navier–Stokes equations in primitive variables. The results observed in Fig. 8 show that the multigrid method also exhibits poor performance when it is used to solve two-dimensional Navier–Stokes equations with alternative formulations, thus agreeing with the results provided in [2].

4.4. Computational effort

To determine the order of the solver and the behavior of the curve with the CPU time, a curve fitting uses geometric least squares [38] whose function is given by

$$t_{CPU}(N) = cN^p, \tag{16}$$

where p represents the order of the solver associated with the method employed (or the inclination of each curve of Figs. 7 and 8), and c is a coefficient that depends on the adopted method and solver. The closer the unit is to the value of p, the better the performance of the algorithm used. Theoretically, the multigrid method is ideal if $p = 1$; this means that the CPU time increases linearly with the size of the grid.

Table 2 presents the exponents p of Eq. (16) obtained for the five problems and the CS and FAS schemes. The results confirmed that the CPU time multigrid method and MSI solver grows almost linearly with increasing N to the Laplace, Navier and Burgers equations. In this case, the value of p is not significantly affected by the number of equations, the complexity (linear or nonlinear), or the type of scheme (CS or FAS); the relevant effect is the use of the multigrid. Values of p near 2 for the singlegrid method agree with the theoretical values [38], as well as values near unity obtained for the multigrid method. However, the same results were not observed for the Navier–Stokes equations.

For the Navier–Stokes equations, the value of p was obtained for $N \geqslant 17 \times 17$ points. The results indicated that the CPU time of the multigrid method with the SOR solver does not grow proportionally with N. It can be observed in Table 2 that the

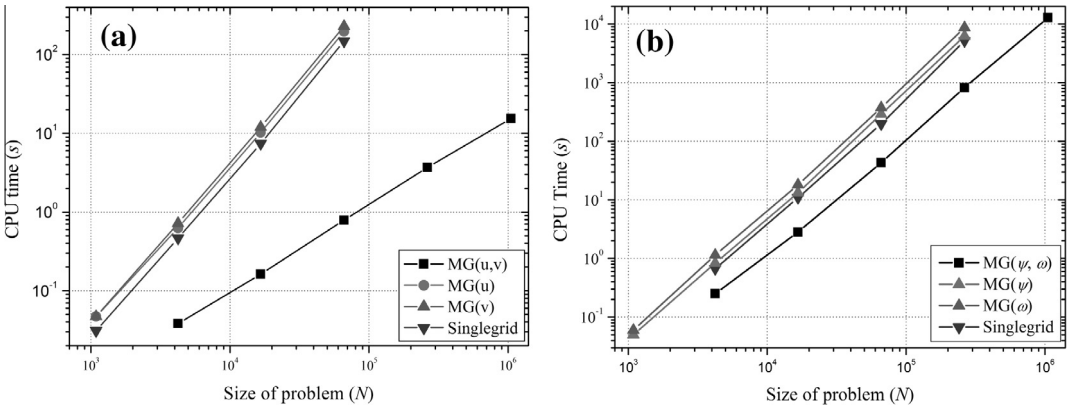

Fig. 9. Performance of the multigrid method when applied to only one equation in problems with two equations: (a) Burgers equations; (b) Navier–Stokes equations in streamfunction–vorticity formulation.

order p of the solver in the multigrid method decreases with increasing Reynolds numbers in both formulations but still falls short from the ideal value (i.e., p is close to the expected value 1 when Re = 1000); for smaller Reynolds numbers, the values are not good.

In the case of the singlegrid in the streamfunction–velocity formulation, there is a significant oscillation when the Reynolds number varies. In the streamfunction–vorticity formulation, the values of p showed more stable behavior (close to 2), which is the theoretical order of the adopted solver; it is noted, however, that the same does not occur in the multigrid.

The degeneration of the performance of the multigrid method for the solution of the Navier–Stokes equation seems to be primarily due to the physics of the problem; other factors appear to play a secondary role on the performance of the multigrid method, including the mathematical formulation used and the specific components of the multigrid method employed.

Although the results for the multigrid method have not reached a desirable linear convergence, it should be emphasized that there was a considerable reduction in CPU time compared to the singlegrid method. Even with higher Reynolds numbers, there were significant gains; however, the results are still important, as a greater number of points were used in the computational grid, as seen in Fig. 8.

4.5. Analysis of problems with two equations and multigrid in one

This section presents the performance analysis of the multigrid method applied to only one equation when it solves problems with two coupled equations. Some authors only use the multigrid method in an equation when solving coupled problems, primarily those involving a Poisson equation. One explanation is that a Poisson equation requires more CPU time [36].

The present analysis adopted Burgers equations that involve the coupling of equations that conserved the momentum in the coordinate directions, and Navier–Stokes equations in streamfunction–vorticity formulation that involved the coupling of streamfunction–vorticity with Re = 100. The simulations were performed with $N \geqslant 33 \times 33$, except in the streamfunction–vorticity formulation, whose CPU time with this grid is close to zero in both singlegrid and multigrid methods. The computer codes were specifically written for each case. The procedure consists of updating one of the variables (u or v (Burgers equations); ψ or ω (streamfunction–vorticity formulation)) in only the finest grid, while the other variable visits all grids in the V-cycle.

Fig. 9 shows the evolution of the CPU time with the grid refinement obtained for both tested problems. Fig. 9(a) shows the results for the Burgers equations, in which MG(u, v) solved the problem with the multigrid in two variables, MG(u)—only u and MG(v)—only v. The same nomenclature was adopted for Fig. 9(b).

For both analyzed problems, the curves of Fig. 9 have shown that the convergence of the multigrid degenerated when it was used in only one equation. The CPU time curves of the singlegrid and of only one equation in the multigrid are very close and grow with the same inclination, independent of the variable that is solved with the multigrid. The small increase in CPU time observed in the curves of Fig. 9, for MG(u), MG(v), MG(ψ) and MG(ω) may be due to the increased complexity of the operations of restriction and prolongation in the multigrid method.

5. Conclusions

This paper analyzed the influence of some parameters of the geometric multigrid method in resolving five problems, as well as the effect of the number of differential equations and Reynolds numbers on the CPU time for the Navier–Stokes equations in two formulations.

The primary conclusions are as follows:

Number of inner iterations (η): (a) For a CS scheme and Laplace and Navier equations, an $\eta_{opt} = 2$ was obtained. For an FAS scheme, an $\eta_{opt} = 2$ was obtained for Navier equations; an $\eta_{opt} = 5$ for Burgers equations, and an $\eta_{opt} = 8$ for Laplace equation. When the value of η is different from η_{opt}, the CPU time increases, which can be significant depending on the value used for η. (b) In the Navier–Stokes equations using a streamfunction–velocity formulation, with the five or nine-points approximation schemes, the parameter η_{opt} is very sensitive to the Reynolds number and grid size; when using a streamfunction–vorticity formulation, the η_{opt} is less sensitive to these two parameters. (c) The simplification with the five-points approximation scheme did not reduce the CPU time as expected.

Number of levels (L): (a) For the two schemes (CS and FAS), the three grids and the three problems in Fig. 5, the CPU time when $L = L_{max}$, $L_{max} - 1$, $L_{max} - 2$ and $L_{max} - 3$ does not significantly change close to the optimal value; it may be considered that $L_{opt} = L_{max}$. For $L \ll L_{opt}$, the CPU time increases significantly. (b) For the Navier–Stokes equations in two formulations, the expected performance was obtained for the number of levels as in the case of Laplace, Navier and Burgers equations. However, the results are qualitatively similar. The optimal number of levels is $L_{opt} = L_{max}$, with the exception of the streamfunction–velocity formulation with Re = 1000 in the 513×513 grid, which is $L_{opt} = L_{max} - 1$.

Size of problem (N): (a) The theoretical performance of the multigrid method for Laplace, Navier and Burgers equations was confirmed, i.e., $p \approx 1$, according to Table 2. (b) In the streamfunction–velocity formulation, the parameters L, N and Re strongly influenced the CPU time. However, in the streamfunction–vorticity formulation, the influence of these parameters is less significant. By varying of Reynolds number, the value of p also varies. These results indicate that the performance of the multigrid method for the solution of the Navier–Stokes equation seems to be primarily affected by the physics of the problem; other factors appear to play a secondary role on the performance of the multigrid method, including the mathematical formulation used and the parameters of the multigrid method.

Equations with couplings: (a) For the Navier and Burgers equations, the number of equations involved did not affect the order of p, confirming the theoretical performance of the multigrid method for coupled equations. (b) The excellent performance of the multigrid method verified with the coupled problems was not confirmed for the Navier–Stokes formulation written in the streamfunction–vorticity formulation. (c) Although the streamfunction–velocity formulation involves only a partial differential equation in the multigrid cycle (i.e., without coupling), it was insufficient to ensure the order of $p = 1$.

Problems with two equations and a multigrid in one: For both analyzed problems, the results showed that the convergence of the multigrid degenerated when it was used in only one equation.

Acknowledgments

The authors would like to acknowledge the financial support provided by CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico – Brazil), AEB (Agência Espacial Brasileira, by the Uniespaço Program), and CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brazil). The second author is supported by a CNPq scholarship.

References

[1] J.C. Tannehill, D.A. Anderson, R.H. Pletcher, Computational Fluid Mechanics and Heat Transfer, second ed., Washington, 1997.
[2] U. Ghia, N. Ghia, C. Shin, High-resolutions for incompressible flow using the Navier–Stokes equations and a multigrid method, J. Comput. Phys. 48 (1982) 387–411.
[3] J. Zhang, Numerical simulation of 2D square driven cavity using fourth-order compact finite difference schemes, Comput. Math. Appl. 45 (2003) 43–52.
[4] G.E. Schneider, M. Zedan, A modified strongly implicit procedure for the numerical solution of field problems, Numer. Heat Transfer 4 (1981) 1–19.
[5] A. Brandt, T. Diskin, J.L. Thomas, Recent advances in achieving text-book multigrid efficiency for computational simulations, NASA/CR ICASE Rep. 16 (2002).
[6] M.C. Thompson, J.H. Ferziger, An adaptive multigrid technique for the incompressible Navier–Stokes equations, J. Comput. Phys. 82 (1989) 94–121.
[7] A. Brandt, Multi-level adaptive solutions to boundary-value problems, Math. Comput. 31 (1977) 333–390.
[8] W. Briggs, V. Henson, S. McCormick, A Multigrid Tutorial, second ed., SIAM, Philadelphia, 2000.
[9] U. Trottenberg, C.W. Oosterlee, A. Schüller, Multigrid, first ed., Academic Press, San Diego, 2001.
[10] W. Hackbusch, A multigrid method applied to a boundary problem with variable coefficients in a rectangle, Institut Fur Angewandte Mathematik, Universitat Koln. 17 Report 77, 1977.
[11] C. Hirsch, Numerical Computational of Internal and External Flows, Wiley, 1988.
[12] J.H. Ferziger, M. Peric, Computational Methods for Fluids Dynamics, third ed., Springer, 2001.
[13] S.P. Vanka, Block-implicit multigrid solution of Navier–Stokes equations in primitive variables, J. Comput. Phys. 65 (1986) 138–158.
[14] P. Sockol, Multigrid solution of the Navier–Stokes equations on highly stretched grids, Int. J. Numer. Methods Fluids 17 (1993) 543–566.
[15] J. Yan, F. Thiele, L. Xue, A modified full multigrid algorithm for the Navier–Stokes equations, Comput. Fluids 36 (2007) 445–454.
[16] D. Kumar, K. Kumar, M. Das, A fine grid solution for a lid-driven cavity flow using multigrid method, Eng. Appl. Comput. Fluid Mech. 3 (2009) 336–354.
[17] C.D. Santiago, Study of the parameters of the multigrid method for systems of equations in CFD (Doctoral thesis), Federal University of Paraná – Curitiba, PR, Brazil, 2010 (in Portuguese).

[18] C.D. Santiago, C.H. Marchi, Optimum parameters of a geometric multigrid for a two-dimensional problem of two-equations, in: Proc. 19st International Congress of Mechanical Engineering, Brasília-Brazil, 2007, pp. 1–9.
[19] C.D. Santiago, C.H. Marchi, Optimum parameters of the geometric multigrid CS and FAS for two-dimensional problems of two-equations, in: Proc. XXIX Iberian–Latin–American Congress on Computational Methods in Engineering, Maceió-Brazil, 2008, pp. 1–17 (in Portuguese).
[20] C.D. Santiago, C.H. Marchi, L.F. Souza, Analysis of the performance of the geometric multigrid method with the streamfunction–velocity formulation, in: Proc. VI National Congress of Mechanical Engineering (CONEM), Campina Grande-Brazil, 2010, pp. 1–10 (in Portuguese).
[21] C.D. Santiago, C.H. Marchi, L.F. Souza, L.K. Araki, Performance of the multigrid method with alternative formulations for the Navier–Stokes equations, in: Proc. 21st International Congress of Mechanical Engineering, Natal-Brazil, 2011, pp. 1–10.
[22] M.M. Gupta, J.C. Kalita, A new paradigm for solving Navier–Stokes equations: streamfunction–velocity formulation, J. Comput. Phys. 207 (2005) 52–68.
[23] Z.F. Tian, P.X. Yu, An efficient compact difference scheme for solving the streamfunction formulation of the incompressible Navier–Stokes equations, J. Comput. Phys. 230 (2011) 6404–6419.
[24] S.K. Pandit, On the use of compact streamfunction–velocity formulation of steady Navier–Stokes equations on geometries beyond rectangular, J. Sci. Comput. 36 (2008) 219–242.
[25] K. Poochinapan, Numerical implementations for 2D lid-driven cavity flow in stream function formulation, ISRN Appl. Math. 2012 (2012). 17 pages.
[26] E. Erturk, T.C. Corke, C. Gokcol, Numerical solutions of 2D steady in-compressible driven cavity flow at high Reynolds numbers, Int. J. Numer. Methods Fluids 48 (2005) 747–774.
[27] J. Salençon, Handbook of Continuum Mechanics: General Concepts Thermoelasticity, Berlin, New York, 2001.
[28] T.M. Shih, C.H. Tan, B.C. Hwang, Effects of grid staggering on numerical scheme, Int. J. Numer. Methods Fluids 49 (1989) 193–212.
[29] M.O. Efe, Observer-based boundary control for 2D Burgers equations, Trans. Inst. Meas. Control 28 (2006) 177–185.
[30] R. Schreiber, H.B. Keller, Driven cavity flows by efficient numerical techniques, J. Comput. Phys. 49 (1983) 330–333.
[31] O.N. Shankar, M.D. Deshpande, Fluid mechanics in the driven cavity, Annu. Rev. Fluid Mech. 32 (2000) 93–136.
[32] E. Erturk, B. Dursun, Numerical solutions of 2D steady incompressible flow in a driven skewed cavity, ZAMM – J. Appl. Math. Mech. 87 (2009) 377–392.
[33] J.A. Rabi, M.J.S. De-Lemos, Optimization of convergence acceleration in multigrid numerical solutions of conductive–convective problems, Appl. Math. Comput. 124 (2001) 215–226.
[34] Z.F. Tian, S.Q. Dai, High-order compact exponential finite difference methods for convection–diffusion type problems, J. Comput. Phys. 220 (2007) 952–974.
[35] J. Larsson, F.S. Lien, E. Yee, Conditional semicoarsening multigrid algorithm for the Poisson equation on anisotropic grids, J. Comput. Phys. 208 (2005) 368–383.
[36] M.A.V. Pinto, C.H. Marchi, Effect of parameters of CS and FAS multigrid method on the CPU time for the two-dimensional Laplace equation, in: Proceedings of ENCIT, 2006, pp. 1–10 (in Portuguese).
[37] R.L. Burden, J.D. Faires, Análise Numérica, second ed., Pioneira Thomson Learning, São Paulo, 2003.