Acta Scientiarum

http://periodicos.uem.br/ojs
ISSN on-line: 1807-8664
Doi: 10.4025/actascitechnol.v44i1.57398

Measuring the effectiveness of extrapolation techniques associated with the multigrid method applied to the Navier-Stokes equations

Bruno Benato Rutyna[1], Marcio Augusto Villela Pinto[2*], Reverton Luis Antunes Neundorf[1], Marcio Alexandro Maciel de Anunciação[1] and Marcio André Martins[3]

[1]Programa de Pós Graduação em Métodos Numéricos em Engenharia, Universidade Federal do Paraná, Curitiba, Paraná, Brasil. [2]Departamento de Engenharia Mecânica, Universidade Federal do Paraná, Av. Cel. Francisco Heráclito dos Santos, 230, 81530-000, Centro Politécnico, Bloco IV, Curitiba, Paraná, Brasil. [3]Departamento de Matemática, Universidade Estadual do Centro-Oeste, Guarapuava, Paraná, Brasil. *Author for correspondence. E-mail: marcio_villela@ufpr.br

ABSTRACT. In this work, we applied different extrapolation techniques in association with the multigrid method to discover which one is the most effective in reducing the iteration error and the processing time (CPU time), as well as in improving the convergence factors. The mathematical model studied refers to the two-dimensional laminar flow of an isothermal time-dependent incompressible fluid modeled by the Navier-Stokes equations, with $Re = 1$, solved iteratively with the projection method and the Finite Volume Method. The extrapolation methods used were: Aitken, Empiric, Mitin, scalar Epsilon, scalar Rho, topological Epsilon, and topological Rho. A two-step application was performed: first, extrapolators methods were applied individually after the use of the multigrid method. Then, the best-performing extrapolation techniques were used in the second step, where they were applied between the cycles of the multigrid method. The methods that presented the best convergence properties in the first stage were topological and scalar Epsilon. In the second stage, both methods maintained their performance, however, the topological Epsilon method presented more significant convergence rates than the scalar Epsilon. The other parameters analyzed were: the storage memory peak, the dimensionless norm of the residual based on the initial estimate, and the error norms of iteration. Thus, it was possible to state which extrapolation technique performed best and to compare it with the multigrid method with no extrapolation, which in this study was the topological Epsilon method.

Keywords: Finite volume method; projection method; convergence acceleration; topological Epsilon.

Received on January 13, 2021.
Accepted on October 13, 2021.

Introduction

In Computational Fluid Dynamics (CFD), determining the solution for large systems is an important task, since these problems require increasingly faster and more efficient methods for finding solutions.

In this paper, we solved the problem of a two-dimensional laminar flow in a transient and isothermal regime of an incompressible fluid modeled by the Navier-Stokes equations with Dirichlet boundary conditions numerically. To overcome the dependence between pressure and velocity in this problem, several numerical schemes were developed. The Projection method (Chorin, 1968), which splits the solution into parts, stands out for solving a local process at each time step.

To solve a problem numerically, it is required to transform the continuous into a discrete domain. After this, the equations are represented by Finite Volume Method (FVM) (Versteeg & Malalasekera, 2007). A feature of the discretization of Partial Differential Equations (PDEs) through the FVM is that a linear system of the type

$$A\mathbf{u} = \mathbf{f} \tag{1}$$

is obtained, in which A is the coefficient matrix, **f** is a vector of independent terms and **u** is a vector of unknowns. Generally, these systems generate sparse and large matrices. To achieve the solution of such systems, iterative methods are widely applied (Burden, Faires, & Burden, 2016). These approaches may result in difficulties related to the slow convergence of the iterative process applied.

The multigrid method has been very efficient in improving the convergence of the iterative methods (Trottenberg, Oosterlee, & Schuller, 2001). Its philosophy is based on the use of a set of several grids with several degrees of refinement in which the iterations are performed during the process. This happens from the level with the highest refinement, i.e., the finest grid (which is the original discretization of the problem) to the less refined levels, i.e., coarser grid. The relationship between two grids with different levels of refinement is typically called coarsening ratio.

To improve the convergence of the multigrid method, we can also use extrapolation techniques or convergence acceleration associated with iterative methods (Brezinski & Zaglia, 2013), thus obtaining numerical solutions with a lower computational cost.

In the literature, many other techniques enhance the multigrid method. For example, multigrid-based preconditioners using Incomplete LU decomposition (ILU) solvers are used in conjunction with the Conjugate Gradient methods in Anunciação, Pinto, and Neundorf (2020). Another technique to reduce the CPU time is parallelization using Graphics Processing Unit (GPU) in the solution, as shown in Liu, Yang, and Cheng (2015). Moreover, the use of several sweeps for temporal discretization for parallelization purposes is shown in Franco, Gaspar, Pinto, and Rodrigo (2018) and Franco, Rodrigo, Gaspar, and Pinto (2018). In Zhang, Zhang, and Xi (2010), the Pseudospectral Chebyshev method is used along with the multigrid method to solve the Navier-Stokes equations in primitive variables. Repeated Richardson Extrapolations (RRE) is used to reduce the discretization error in solving CFD equations (Marchi et al., 2013; Marchi et al., 2016).

We emphasize that the focus of this work is the use of extrapolation techniques together with the multigrid method to accelerate the convergence of the resolution of systems generated from projection methods applied to the Navier-Stokes equations.

Material and methods

Mathematical model

The Navier-Stokes equations can be written in many forms depending on the properties of the fluid and the flow (Versteeg & Malalasekera, 2007). In this study, they are treated as a two-dimensional laminar flow of an isothermal time-dependent incompressible fluid modeled by

$$\nabla \cdot \mathbf{u} = 0, \tag{2}$$

$$\frac{\partial u}{\partial t} + \mathbf{u} \cdot \nabla u = -\frac{\partial p}{\partial x} + \frac{1}{Re}\nabla^2 u, \tag{3}$$

$$\frac{\partial v}{\partial t} + \mathbf{u} \cdot \nabla v = -\frac{\partial p}{\partial y} + \frac{1}{Re}\nabla^2 v, \tag{4}$$

being x and y the spatial coordinates; t the temporal coordinate; u and v the components of the velocity vector $\mathbf{u}$ in the directions x and y, respectively; p the fluid pressure; Re the Reynolds number; $\nabla \cdot \mathbf{u}$ the divergence of $\mathbf{u}$; ∇ the gradient and ∇^2, the Laplacian operators. To complete, there is

$$Re = \frac{\rho\, v_m D}{\mu}, \tag{5}$$

where ρ is the specific mass of the fluid; μ its dynamic viscosity; v_m its average speed and D, the tube diameter. Equation (2) is known as the mass conservation (or continuity) equation, and Equations (3) and (4) are known as the conservation equations of the linear momentum in the x and y directions, respectively.

This work solves the Taylor-Green vortex problem (Anunciação et al., 2020), whose domain is given by $\{(x, y) \in \mathbb{R}^2 : -\pi \leq x \leq \pi \text{ and } -\pi \leq y \leq \pi\}$. The analytical solutions for $Re = 1$ are:

$$u = -(\cos x . \sin y)e^{-2t}, \tag{6}$$

$$v = (\sin x . \cos y)e^{-2t}, \tag{7}$$

$$p = -\frac{1}{4}(\cos 2x + \sin 2y)e^{-4t}, \tag{8}$$

where $t = nh_t$, with n being the time step where the solution is considered, and h_t, the temporal refinement. The analytical solutions generate the initial and boundary conditions of the problem.

Projection methods

Projection methods are often divided into three classes: pressure-correction schemes, velocity-correction schemes, and consistent splitting schemes. The simplest pressure-correction scheme was proposed by Chorin (1968). It uses the Euler method in the time discretization and creates an auxiliary or provisional velocity field.

The projection methods have the property of solving at each time step a sequence of elliptic and decoupled equations for the velocities and pressure variables. The method used in this work, which is a variation of Chorin's (1968), is shown below:

First step:

$$\frac{(3\mathbf{u}^t - 4\mathbf{u}^n + \mathbf{u}^{n-1})}{2h_t} = \beta_1 g_1(\mathbf{u}^n) + \beta_0 g_0(\mathbf{u}^{n-1}) + \omega\nabla^2\mathbf{u}^t + \nabla p^n, \tag{9}$$

$$\mathbf{u}^t|_{\partial\Omega} = \mathbf{b}^n, \tag{10}$$

where $\mathbf{u}^t$ is the auxiliary velocity field, $\mathbf{u}^n$ is the velocity field, p^n is the pressure, $\mathbf{b}^n$ are the boundary conditions, all cases in the time step n; β_0 and β_1 are constants, and g_0 and g_1 are the convective terms $\mathbf{u} \cdot \nabla\mathbf{u}$. The choices of β_0 and β_1 imply whether the convective terms will be treated explicitly or implicitly. In this work, we use the second-order-explicit Adams-Bashforth scheme for the convective terms, that is $\beta_0 = 1/2$ and $\beta_1 = -3/2$ (Burden et al., 2016; Anunciação et al., 2020).

Second step:

$$\nabla^2\phi^{n+1} = \frac{3}{2h_t}\nabla \cdot \mathbf{u}^t, \tag{11}$$

$$\mathbf{u}^{n+1} = \mathbf{u}^t - \frac{2h_t}{3}\nabla\phi^{n+1}, \tag{12}$$

$$p^{n+1} = p^n + \phi^{n+1} - \omega\nabla \cdot \mathbf{u}^t, \tag{13}$$

where ϕ is the pressure correction term and $\omega = 1/Re$. This version of the projection method has Neumann boundary conditions in the pressure given by

$$\partial_n p^{n+1}|_{\partial\Omega} = -\omega(\nabla \times \nabla \times u^{n+1}) \cdot \mathbf{n}|_{\partial\Omega}, \tag{14}$$

where n being the unit normal vector external to the boundary of Ω, while × represents the cross product. These consistent boundary conditions for the pressure (Guermond, Miney, & Shen, 2006) show that this version used in this work, also called incremental version in rotational form, is of second order for velocities in the L_2, L_1, and L_∞ norms. For the pressure, the scheme is of second order in the L_2 and L_1 norms, and of a 3/2 order in the L_∞ norm. Several other projection methods are improved versions of Chorin's (1968) but do not reach the same convergence orders used in this paper.

The use of the projection method has advantages over others such as the SIMPLE method. The resulting equations avoid coupling in the velocities variables and are treated separately, reducing the complexity of the algorithm (Griffith, 2009). For a generalization of projection methods see Guermond et al. (2006).

Discretization

This article uses FVM with staggered grids. The velocities are placed on the faces and the pressure in the centers of the control volumes, thus avoiding pressure instabilities (Versteeg & Malalasekera, 2007).

As shown in the Figure (1), the inner volumes (solid lines) are the real volumes, whose numerical methodology – as described in Versteeg and Malalasekera (2007) – is normally applied, and in this paper, we applied with lexicographic ordering. The volumes with dashed lines represent the ghost volumes, i.e., volumes that do not belong to the physical domain of the problem created only as a numerical strategy to impose boundary conditions. However, they facilitate the computational implementation since they are treated similarly to those in the interior. The boundary conditions for the variable u are prescribed in the east and west contours. In the north and south boundaries, linear extrapolation is required to compute u. Analogously, the variable v is prescribed the north and south contours, and linear extrapolation is required in the other two directions. Details on the grids used and spatial discretization by FVM to this problem can be found in Anunciação et al. (2020).

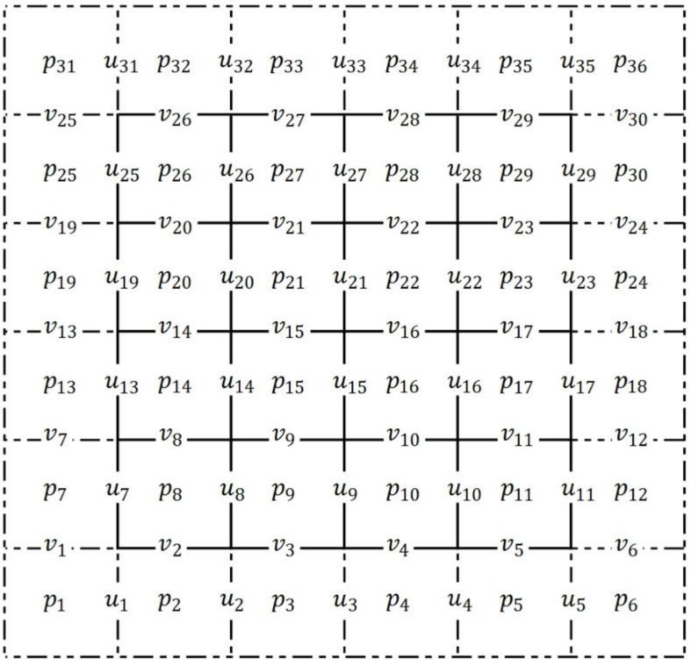
p_{31} u_{31} p_{32} u_{32} p_{33} u_{33} p_{34} u_{34} p_{35} u_{35} p_{36}
v_{25} v_{26} v_{27} v_{28} v_{29} v_{30}
p_{25} u_{25} p_{26} u_{26} p_{27} u_{27} p_{28} u_{28} p_{29} u_{29} p_{30}
v_{19} v_{20} v_{21} v_{22} v_{23} v_{24}
p_{19} u_{19} p_{20} u_{20} p_{21} u_{21} p_{22} u_{22} p_{23} u_{23} p_{24}
v_{13} v_{14} v_{15} v_{16} v_{17} v_{18}
p_{13} u_{13} p_{14} u_{14} p_{15} u_{15} p_{16} u_{16} p_{17} u_{17} p_{18}
v_7 v_8 v_9 v_{10} v_{11} v_{12}
p_7 u_7 p_8 u_8 p_9 u_9 p_{10} u_{10} p_{11} u_{11} p_{12}
v_1 v_2 v_3 v_4 v_5 v_6
p_1 u_1 p_2 u_2 p_3 u_3 p_4 u_4 p_5 u_5 p_6

Figure1. An example of a staggered grid.

The temporal discretization is done semi-implicitly by the Semi-Backward Difference Formula (SBDF) (Guermond et al., 2006). Anunciação et al. (2020) found satisfactory results when employing it in association with the multigrid method. The SBDF has three steps and requires parameter tuning, depending on the type of problem it is being applied to. In the first, the solutions are obtained from the initial and contour conditions of the problem. In the latter, the Euler Method is used to solve the equations related to the first advance in time, and only then the SBDF is normally used. Thus, we managed to obtain second-order convergence in the temporal discretization of the variables u and v in the Navier-Stokes equations. Based on how the projection method solves the pressure in the contours, there is a numerical boundary layer that degenerates the convergence order to approximately 1.5, as discussed in remark 3.2 in Guermond et al. (2006).

To guarantee the stability of the solutions, the time step h_t must respect the following criterion (Versteeg & Malalasekera, 2007):

$$h_t < \frac{1}{2Re}\frac{h_x^2 h_y^2}{h_x^2 + h_y^2}, \tag{15}$$

or, when $h = h_x = h_y$:

$$h_t < \frac{h^2}{4Re}. \tag{16}$$

In general, projection methods cause the boundary conditions on the pressure to be of the Neumann type. To guarantee the existence and uniqueness of the solution, they should have (Trottenberg et al., 2001):

$$\int_{\Omega} \nabla^2 \phi^{n+1} = \int_{\Omega} \frac{3}{2h_t} \nabla \cdot \mathbf{u}^t = 0. \tag{17}$$

More details on projection methods can be found in Guermond et al. (2006).

Multigrid method

The multigrid method is applied to systems of equations of type Equation (1), with a discretized domain Ω^h (where h is the size of the control volume). To begin, simply apply an iterative solver, also called smoother (Oliveira, Pinto, Gonçalves, & Rutz, 2018), such as weighted Jacobi or Gauss-Seidel, but only a reduced number of iterations. After this, the residual can be computed with:

$$\mathbf{r}^h = \mathbf{f}^h - A^h \mathbf{v}^h \tag{18}$$

in which $\mathbf{v}^h$ is the approximation of $\mathbf{u}$ obtained in the mesh Ω^h.

After the above procedure, the residual is transferred to the subsequently coarser grid using a restriction operator. The coarsening ratio of the mesh will be equal to 2 (Trottenberg et al., 2001). Thus, the following system is obtained:

$$A^{2h}\mathbf{e}^{2h} = \mathbf{r}^{2h}\ , in\ \Omega^{2h}, \tag{19}$$

where $\mathbf{e}^{2h}$ is the error to be estimated.

After obtaining an error approximation in the grid Ω^{2h}, the process of correction of the approximation $\mathbf{v}^h$ begins. First, we use an operator that transfers the error to the finer grid by using a prolongation operator, then we perform the following operation:

$$\mathbf{v}^h \leftarrow \mathbf{v}^h + \mathbf{e}^{2h}. \tag{20}$$

Finally, we once again apply the chosen smoother in the approximation $\mathbf{v}^h$ obtained. More details about restriction and prolongation operators are found in Trottenberg et al. (2001).

This procedure can be performed using coarser grids. Each time we return to the finest grid with the correction, there is what it is called a V-cycle, as shown in Figure 2, in which R represents the restriction operator, P represents the prolongation operator and S, the smoothing.

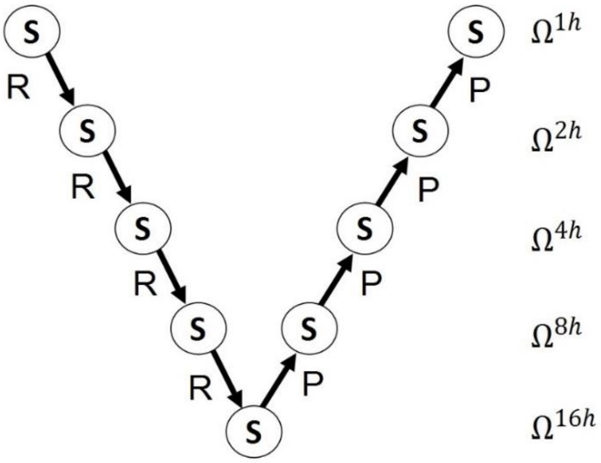

Figure 2. V-cycle applied in 5 grids. S = smoothing, R = restriction and P = prolongation.

The number of times to perform the solver iteration in each level of the V-cycle must be set by a fixed quantity defined by user or by parameters that involve the convergence of the solution. They are called number of pre- and post-smoothing in the restriction and prolongation, respectively.

Extrapolation methods

The iterative processes used in solving systems of equations may present slow and undesired convergence in the applications. One way to get around this is through extrapolation techniques, which are based on numerical sequence transformations. That is, if one sequence converges slowly, it is possible to transform it into another, as long as the original limit remains, and this new sequence may converge faster than the first one. These sequences must satisfy certain conditions, as seen in Brezinski and Zaglia (2013).

Suppose that a real or complex sequence S_n is converging to s. A sequence transformation (T_n) converging to s, according to Brezinski and Zaglia (2013), where T_n is the newly generated sequence, is also called extrapolator.

This paper uses seven well-known extrapolators, being five of the scalar type and two of the vector type. When $\mathrm{C}_1, \mathrm{C}_2, \ldots, \mathrm{C}_\mathrm{k}, \ldots, \mathrm{C}_\infty$ are obtained through an iterative process and $\phi_1 = \mathrm{C}_1(\mathrm{k})$, $\phi_2 = \mathrm{C}_2(\mathrm{k})$, $\phi_i = \mathrm{C}_\mathrm{i}(\mathrm{k})$ are the scalars for all k-th components of the vectors C_i, then the Aitken extrapolator (Burden et al., 2016) is defined by:

$$\phi_\infty^{Aitken} = \frac{\phi_1\phi_3 - \phi_2^2}{\phi_3 - 2\phi_2 + \phi_1}. \tag{21}$$

There are also the Empirical Estimator proposed by Martins and Marchi (2008),

$$\phi_{\infty}^{Empirical} = \phi_3 + \frac{(\phi_3 - \phi_2)^2}{2\phi_2 - \phi_3 - \phi_1}, \tag{22}$$

and

$$\phi_{\infty}^{Mitin} = \frac{\phi_1\phi_5 - \phi_3^2}{\phi_5 - 2\phi_3 + \phi_1}, \tag{23}$$

which is the formula used in Mitin extrapolation (Mitin, 1985).

The two other extrapolators used in this work, Epsilon and Rho, have a formal similarity between them but differ in terms of convergence capabilities. The Epsilon extrapolator is a powerful accelerator of iterative methods that solve discrete differential equations, however, it fails when applied to sequence that converge logarithmically (Delahaye, 2012). Conversely, the Rho extrapolator does not accelerate sequences with linear convergence but is very efficient for logarithmically convergent sequences (Gao, Jiang, Liao, & Song, 2010).

For a sequence (S_n), the scalar Epsilon extrapolator is defined as found in Brezinski and Zaglia (2013) by:

$$\epsilon_{-1}^{(n)} = 0,\ \ \epsilon_0^{(n)} = S_n,\ \ n = 0,1,.. \tag{24}$$

$$\epsilon_{k+1}^{(n)} = \epsilon_{k-1}^{(n+1)} + \frac{1}{\epsilon_k^{(n+1)} - \epsilon_k^{(n)}},\ \ n,k = 0,1,.. \tag{25}$$

for denominators other than zero. The superscript of ϵ represents the element, and the subscript, the iteration.

Remark 1: The expressions given by Equations (24) and (25) only make sense because Brezinski and Zaglia (2013) assumed infinite sequences. In our case, we always admitted a finite sequence, which depends on the number of iterations.

Remark 2: In Equation (25), the structure $n, k = 0,1,..$ is understood, for a recursive procedure, as For $n = 0,1,\ldots$ (For $k = 0,1,..$).

Similar, the scalar Rho extrapolator is given as found in Brezinski and Zaglia (2013) by:

$$\rho_{-1}^{(n)} = 0,\ \ \rho_0^{(n)} = S_n,\ \ n = 0,1,\ldots, \tag{26}$$

$$\rho_{k+1}^{(n)} = \rho_{k-1}^{(n+1)} + \frac{k+1}{\rho_k^{(n+1)} - \rho_k^{(n)}},\ \ n,k = 0,1,\ldots. \tag{27}$$

Both Epsilon and Rho extrapolation methods have scalar and topological variations. While the former preserves individual characteristics of each component of the vector, the latter considers the properties of the vector as a whole. These two variations are covered in this work. The Topological Epsilon extrapolator is given by the equations:

$$\tilde{\varepsilon}_{-1}^{(n)} = 0,\ \ n = 0,1,\ldots, \tag{28}$$

$$\tilde{\varepsilon}_{0}^{(n)} = S_n,\ \ n = 0,1,\ldots, \tag{29}$$

$$\tilde{\varepsilon}_{2k+1}^{(n)} = \tilde{\varepsilon}_{2k-1}^{(n+1)} + \frac{y}{< y, \tilde{\varepsilon}_{2k}^{(n+1)} - \tilde{\varepsilon}_{2k}^{(n)} >},\ \ k,n = 0,1,\ldots, \tag{30}$$

$$\tilde{\varepsilon}_{2k+2}^{(n)} = \tilde{\varepsilon}_{2k}^{(n+1)} + \frac{\tilde{\varepsilon}_{2k}^{(n+1)} - \tilde{\varepsilon}_{2k}^{(n)}}{< \tilde{\varepsilon}_{2k+1}^{(n+1)} - \tilde{\varepsilon}_{2k+1}^{(n)}, \tilde{\varepsilon}_{2k}^{(n+1)} - \tilde{\varepsilon}_{2k}^{(n)} >},\ \ k,n = 0,1,\ldots, \tag{31}$$

where S_n represents elements of vectorial space E, y is an arbitrary vector such that $y \in E^*$ (the algebric dual space of E) (Brezinski & Redivo-Zaglia, 2014), and the operator $<.,.>$ represents an inner product. Morover, the topological Rho extrapolator is given bellow as seen in Brezinski and Zaglia (2013):

$$\widetilde{\rho}_{-1}^{(n)} = 0,\ \ n = 0,1,\ldots, \tag{32}$$

$$\widetilde{\rho}_{0}^{(n)} = S_n,\ \ n = 0,1,\ldots, \tag{33}$$

$$\tilde{\rho}_{2k+1}^{(n)} = \tilde{\rho}_{2k-1}^{(n+1)} + \frac{(2k+1)y}{< y, \tilde{\rho}_{2k}^{(n+1)} - \tilde{\rho}_{2k}^{(n)} >}, \quad k, n = 0,1, \ldots, \tag{34}$$

$$\tilde{\rho}_{2k+2}^{(n)} = \tilde{\rho}_{2k}^{(n+1)} + \frac{(2k+2)\,\tilde{\rho}_{2k}^{(n+2)} - \tilde{\rho}_{2k}^{(n+1)}}{< \tilde{\rho}_{2k}^{(n+2)} - \tilde{\rho}_{2k}^{(n+1)}, \tilde{\rho}_{2k+1}^{(n+1)} - \tilde{\rho}_{2k+1}^{(n)} >}, \quad k, n = 0,1, \ldots. \tag{35}$$

More details of Epsilon and Rho extrapolation methods can be found in Brezinski and Zaglia (2013). Applications of the methods can be seen in Gao et al. (2010).

Results and discussion

Linear systems for the variables u and v must be solved at each time step of the projection method. The multigrid method combined with different extrapolation techniques was used to solve these systems.

The experiment was divided into two stages. In the first one, the extrapolation was performed after the cycles of the multigrid method whereas in the second one, it was applied between cycles. The simulations were performed in different grid sizes and using different stopping criteria. We expected the use of the extrapolation in the Navier-Stokes equations to be as advantageous as the Richardson-Lucy's algorithm, as shown in Gao et al. (2010).

It is noteworthy that the implemented algorithms were validated in other applications, such as in Anunciação et al. (2020), who applied the extrapolators seen in this work in Poisson and Burgers equations.

Implementation data

The multigrid method used was implemented with the correction scheme (CS) and V-cycle, grid coarsening ratio $\tau = 2$ (standard coarsening), and it was solved until the coarsest grid possible was reached (Trottenberg et al., 2001). The maximum number of coarsening levels, called L_{max}, was used. Gauss-Seidel Red-Black smoothing, with pre- and post-smoothing numbers equal to 3, that is, $\nu = \nu_1 = \nu_2 = 3$ was applied (Oliveira, Pinto, Marchi, & Araki, 2012). Full weighting restriction and prolongation using bilinear interpolation was performed. $Re = 1$ was adopted.

At each time step of the projection method, Equation (9), which is a reaction-diffusion equation, is solved in variable u, and Equation (11), which is a Poisson equation, is solved for variable ϕ. This means that each of these variables is solved separately, that is, without a general stopping criterion based on the Navier-Stokes solutions. In this paper, we used the Euclidean-norm of the residual dimensionless by the initial estimate as a stopping criterion in the iterative method (Trottenberg et al., 2001):

$$\|\boldsymbol{r}\|_2 = \frac{\|\boldsymbol{r}^{it}\|_2}{\|\boldsymbol{r}^0\|_2}, \tag{36}$$

where $\mathbf{r}^{it}$ and $\mathbf{r}^0$ represent the residual of the current iteration and initial estimate, respectively. At the end of each V-cycle, Equation (36) is calculated, and when smaller values than the tolerance ε are reached, the process is terminated.

The algorithms were implemented in the Fortran 2003 language with double precision. The tests were performed in a computer with 2.50 GHz x 4 Intel processor, 4GB of RAM, and 64-bit Windows operating system.

Use of post-multigrid extrapolation

During this procedure, we executed the multigrid method until the established tolerance ε was reached. Then, one of these two paths were taken:

- an additional cycle of multigrid was performed - denoted as "MG + 1 ITE"; or
- one extrapolation method was applied - denoted as "MG + Extrapolator".

Table 1 shows the number of approximations used for each extrapolation method.

Table 1. Number of approximations per extrapolation method.

Extrapolator	Number of solutions
MG + Aitken	3
MG + Empirical	3
MG + Mitin	5
MG + Scalar Epsilon	5
MG + Scalar Rho	5
MG + Topological Epsilon	5
MG + Topological Rho	5

The results were measured with the following parameters: processing time or CPU time (t_{cpu}), in seconds [s]; memory peak (M) reached during the execution of the algorithm, in Megabytes [MB]; the dimensionless residual based on the initial estimate ($\|\boldsymbol{r}\|_2$), given by Equation (36); the mean empirical convergence factor ($\hat{q}^{(k)}$), defined by Trottenberg et al. (2001):

$$\hat{q}^{(k)} = \sqrt[k]{\frac{\|\boldsymbol{r}^{it}\|_2}{\|\boldsymbol{r}^0\|_2}}, \qquad (37)$$

where k represents the k-th iteration. Values of $\hat{q}^{(k)}$ considered good are close to zero (Burden et al., 2016). Finally, the Euclidian-norm $\|\boldsymbol{e}\|_2$, is also measured of variables u, v and p, with $\boldsymbol{e}$ being the numerical error in the n-th time step, that is, the difference obtained between the approximation in the n-th time step and the analytical solution.

For the simulations to be validated in different sets, variations were made regarding grid size and the chosen stopping criteria. The experiments are classified according to Table 2. Every scenario uses $n = 10$ as the number of time step, which is interpreted as $10 \times h_t$, where h_t is calculated differently for each grid used, respecting the criterion of Equation (16).

Table 2. Classification of the post-multigrid applications.

Identification	Grid size	Stopping criterion (ϵ)
A1	512×512	10^{-10}
B1	1024×1024	10^{-6}

To highlight the performance of the different extrapolation methods studied, we measured $\hat{q}^{(k)}$ and $\|\mathbf{r}\|_2$ for the variable v. The measurements were obtained at each instant of time t at the resolution step of the linear system to calculate the correction of velocity v, which can be seen in Figure 3. Based on this evaluation, the topological Epsilon extrapolator had the best performance, followed by the Empirical and scalar Epsilon extrapolators.

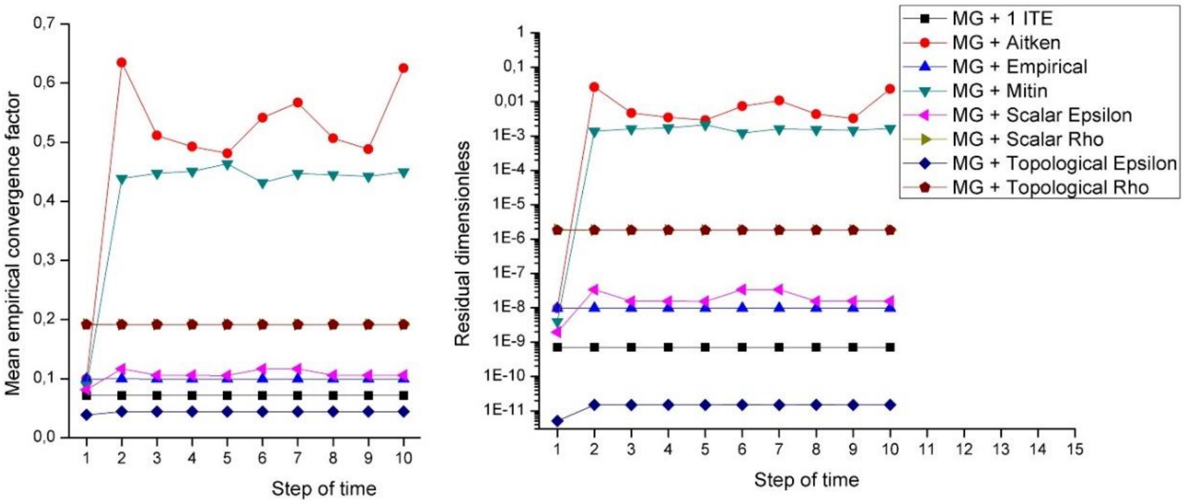

Figure3. $\hat{q}^{(k)}$ and $\|\mathbf{r}\|_2$ at each instant of time for the variable v.

It is worth emphasizing that both parameters $\hat{q}^{(k)}$ and $\|\mathbf{r}\|_2$, presented very similar results when measured for variable u, as seen in Figure 3.

Table 3, which refers to application A1, shows that Empirical, scalar and topological Epsilon had the best performance, with a slight advantage of scalar and topological Epsilon. Note that their computational time is smaller than that of the multigrid with no extrapolation, which hereinafter we will call pure multigrid, and the other parameters presented similar numbers, except for the memory. The memory usage when performing the multigrid method coupled with an extrapolation method is always greater than without it due to the storage of solutions necessary for the method to operate.

Table 3. A1 application results.

Methodology	t_{cpu} [s]	M(MB)	$\|\boldsymbol{e}\|_2(u)$	$\|\boldsymbol{e}\|_2(v)$	$\|\boldsymbol{e}\|_2(p)$
MG + 1 ITE	78.5	101.784	7.385E-08	7.385E-08	1.055E-04
MG + Aitken	66.3	116.204	2.339E-02	2.100E-02	7.368E+00
MG + Empirical	75.4	116.148	8.476E-08	8.476E-08	1.055E-04
MG + Mitin	69.2	116.196	4.625E-04	4.567E-04	3.148E-01
MG + Scalar Epsilon	75.8	116.164	7.463E-08	7.463E-08	1.055E-04
MG + Scalar Rho	72.0	116.196	2.082E-06	2.082E-06	1.122E-04
MG + Topological Epsilon	74.6	134.644	7.459E-08	7.459E-08	1.055E-04
MG + Topological Rho	74.3	134.684	2.083E-06	2.083E-06	1.121E-04

Table 4. shows the results of the post-multigrid application B1 – with a more refined grid.

Table 4. B1 application results.

Methodology	t_{cpu} [s]	M(MB)	$\|\boldsymbol{e}\|_2(u)$	$\|\boldsymbol{e}\|_2(v)$	$\|\boldsymbol{e}\|_2(p)$
MG + 1 ITE	197.5	378.868	4.214E-06	4.214E-06	1.408E-04
MG + Aitken	181.6	424.944	6.870E-05	6.670E-05	1.142E-02
MG + Empirical	194.6	430.34	5.724E-05	5.724E-05	1.598E-03
MG + Mitin	181.5	430.364	5.830E-07	5.838E-07	5.385E-04
MG + Scalar Epsilon	188.7	424.952	2.763E-08	2.763E-08	5.390E-05
MG + Scalar Rho	190.8	430.384	1.061E-02	1.061E-02	2.563E-01
MG + Topological Epsilon	196.0	504.052	1.078E-08	1.077E-08	5.371E-05
MG + Topological Rho	192.9	504.028	1.061E-02	1.061E-02	2.551E-01

The results depicted in Table 4 are the most promising, with highlights to the scalar and topological Epsilon extrapolators. They outperform pure multigrid in every parameter, except for memory usage, which will have a more thorough analysis towards the end of this paper. Noticeably, the decrease of the $\|\boldsymbol{e}\|_2(p)$ was equivalent to one order of magnitude over pure multigrid in both methods. These two methods were selected based on their performance, as shown in this section, and are analyzed in the second stage of results.

Use of extrapolation between cycles

In this section, scalar and topological Epsilon extrapolators were applied between the multigrid cycles. This occurs as follows: given the initial estimate for the problem, five cycles of the multigrid are then performed (see Table 1). Subsequently, five more cycles and a new extrapolation are performed, and so on, until the stopping criterion is reached.

The simulations were organized according to the grid and stopping criteria used (Table 5). The number of time steps n = 10 was used in all cases analyzed.

Table 5. Classification of applications in-between multigrid cycles.

Identification	Grid size	Stopping criterion (ϵ)
C1	512 × 512	10^{-10}
D1	1024 × 1024	10^{-8}
D2	1024 × 1024	10^{-10}

Table 6 shows the application of the scalar and topological Epsilon extrapolation between the multigrid cycles for application C1. Noticeably, the use of the extrapolation produced equal or slightly better than without extrapolation, except for memory usage, as expected. Moreover, the scalar Epsilon presented better memory usage and CPU time results than the topological version.

Table 6. C1 application results.

Methodology	t_{cpu} [s]	M(MB)	$\|\boldsymbol{e}\|_2(u)$	$\|\boldsymbol{e}\|_2(v)$	$\|\boldsymbol{e}\|_2(p)$
MG	65.1	97.568	8.476E-08	8.476E-08	1.055E-04
MG + Topological Epsilon	57.5	130.72	7.492E-08	7.492E-08	1.055E-04
MG + Scalar Epsilon	53.3	112.264	7.585E-08	7.585E-08	1.055E-04

Table 7 shows the data for application D1 application – case with the most refined grid. The most successful extrapolator was the scalar Epsilon, which had better results than pure multigrid and topological Epsilon in all the parameters, except for memory and norm of the error of p in former, and a small difference in t_{CPU}, in latter. In turn, the lowest computational time was reached with the topological Epsilon, which was slightly better than pure multigrid concerning the norms $\|e\|_2(u)$ and $\|e\|_2(v)$.

Table 7. D1 application results.

Methodology	t_{cpu} [s]	M(MB)	$\|e\|_2(u)$	$\|e\|_2(v)$	$\|e\|_2(p)$
MG	197.0	374.788	2.999E-07	2.999E-07	5.384E-05
MG + Topological Epsilon	183.4	499.984	2.305E-07	2.305E-07	8.969E-05
MG + Scalar Epsilon	184.8	426.428	6.380E-08	6.308E-08	5.529E-05

To conclude the analysis of the behavior of the scalar and topological Epsilon extrapolation between the multigrid cycles, the Table 8 brings the results of application D2 – case with the most refined grid and the most rigorous stopping criterion.

Table 8. D2 application results.

Methodology	t_{cpu} [s]	M(MB)	$\|e\|_2(u)$	$\|e\|_2(v)$	$\|e\|_2(p)$
MG	225.8	378.816	3.171E-08	3.171E-08	5.376E-05
MG + Topological Epsilon	226.0	504.124	9.674E-09	9.675E-09	5.370E-05
MG + Scalar Epsilon	230.5	426.428	1.146E-08	1.146E-08	5.371E-05

Although the computational times of multigrid associated with extrapolation methods were slightly higher than pure multigrid, both techniques had better results concerning all norms (except for $\|e\|_\infty(p)$, obtained with the scalar Epsilon). Due to the outstanding results obtained in application D2 and presented in Table 8, only the topological Epsilon extrapolator will be analyzed in the next figures (Figures 4-5).

Figure 4 shows the measurements $\|\mathbf{r}\|_2$ for the variable *v* at each iteration, with the number of time steps *n* = 10 while displaying data from application D2. The variable *u* presents similar results. Note that by applying extrapolation between multigrid cycles, the L_2-norm of residual (in logarithmic scale) has a greater decrease (about 100 times smaller) than pure multigrid.

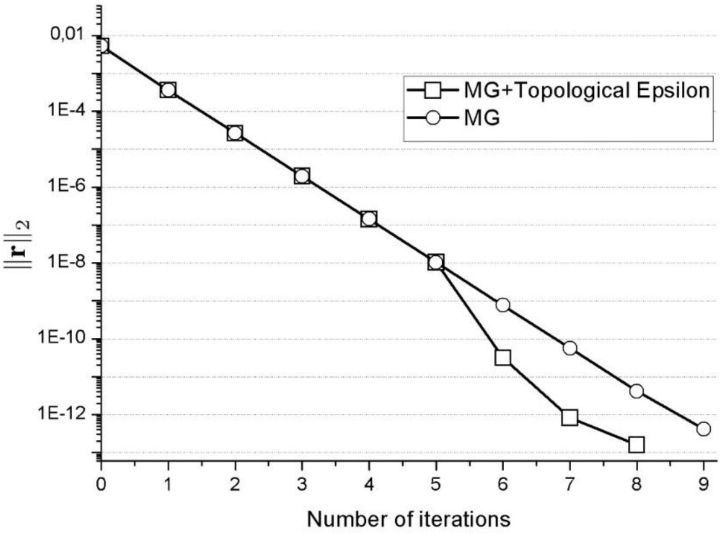

Figure4. L_2-norm of residual for the variable *v* at each iteration and *n* = 10.

To obtain clearer results regarding CPU time and memory usage, we performed additional tests. The first one relates to CPU time. For this, simulations with N ranging from 64 × 64 to 1024 × 1024, tolerance of 10^{-8}, and the number of time steps *n* = 1000 for the proposed problem were made.

Then, we calculated the speed-up parameter, given by Trottenberg et al. (2001)

$$S_p = \frac{t_{CPU}(algorithmA)}{t_{CPU}(algorithmB)}. \quad (38)$$

In this case, the algorithm A represents the multigrid method without the extrapolation and, algorithm B, the multigrid method coupled with the topological Epsilon extrapolation. Values of S_p greater than one indicate that the multigrid combined with extrapolation is faster than without it. This result can be seen in Figure 5(a).

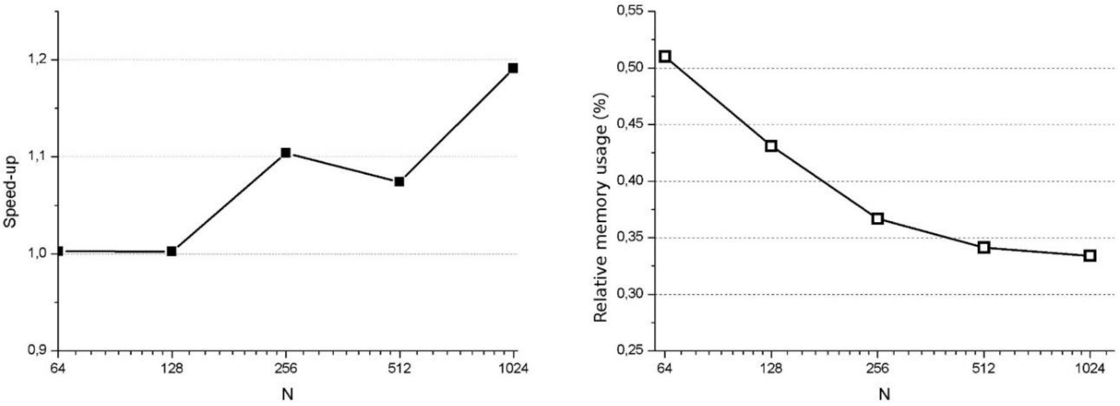

Figure 5. Additional tests: (left) speed-up value and (right) memory RAM usage concerning pure multigrid.

To analyze the effect on RAM memory usage, we compared the memory percentage of the method with and without extrapolation as we increased the number of points in the grid. The results are summarized in Figure 5(b).

Figure 5(b), shows that in addition to decreasing, the relative memory usage has an asymptotic stability behavior with the refinement of the grids. This means that there is no exponential growth when increasing the number of volumes in the domain, making extrapolation techniques feasible.

Conclusion

We solved the Navier-Stokes equations using projection methods, FVM, and the multigrid combined with different extrapolation techniques. The multigrid method was set with Gauss-Seidel red-black smoother and V-cycle.

We can state that:

1 For more refined grids and rigorous stopping criteria, topological Epsilon proved to be the most effective;

2 Regarding memory usage, the topological Epsilon presented an asymptotic stability behavior as the grid is refined;

3 The topological Epsilon reduced the mean empirical convergence factor, the residual norm, and the computational time for problems in refined grids and when many time steps are used.

Acknowledgements

The authors are thankful to the Graduate Program in Numerical Methods in Engineering of the Federal University of Paraná (UPFR). The authors also thank the UFPR and State University of Centro-Oeste (UNICENTRO) for the financial support.

References

Anunciação, M., Pinto, M. A. V., & Neundorf, R. (2020). Solution of the Navier-Stokes equations using projection method and preconditioned conjugated gradient with multigrid and ILU solver. *Revista Internacional de Métodos Numéricos para Cálculo y Diseño en Ingeniería, 36*(1), 1-17. DOI: http://doi.org/10.23967/j.rimni.2020.01.003

Brezinski, C., & Redivo-Zaglia, M. (2014). The simplified topological ε-algorithms for accelerating sequences in a vector space. *SIAM Journal on Scientific Computing, 36*(5), 2227–2247. DOI: http://doi.org/10.1137/140957044

Brezinski, C., & Zaglia, M. R. (2013). *Extrapolation methods: theory and practice* (Vol. 2). North Holand, NL: Elsevier.

Burden, R. L., & Faires, D. J., & Burden, A. M. (2016). *Numerical analysis* (10th ed.). Boston, MA: Cengage Learning.

Chorin, A. J. (1968). Numerical solution of the Navier-Stokes equations. *Mathematics of Computation, 22*, 745–762. DOI: http://doi.org/10.1090/S0025-5718-1968-0242392-2

Delahaye J.-P. (2012). *Sequence transformations* (Vol. 11). Berlim, GE: Springer Science & Business Media.

Franco, S. R., Gaspar, F. J., Pinto, M. A. V., & Rodrigo, C. (2018). Multigrid method based on a space-time approach with standard coarsening for parabolic problems. *Applied Mathematics and Computation, 317*, 25-34. DOI: http://doi.org/10.1016/j.amc.2017.08.043

Franco, S. R., Rodrigo, C., Gaspar, F. J., & Pinto, M. A. V. (2018). A multigrid waveform relaxation method for solving the poroelasticity equations. *Computational and Applied Mathematics, 37*(4), 4805–4820. DOI: http://doi.org/10.1007/s40314-018-0603-9

Gao, Q., Jiang, Z. F., Liao, T., & Song, K. (2010). Application of the vector ε and ρ extrapolation methods in the acceleration of the Richardson–Lucy algorithm. *Optics Communications, 283*(21), 4224–4229. DOI: http://doi.org/10.1016/J.OPTCOM.2010.06.030

Griffith, B. E. (2009). An accurate and efficient method for the incompressible Navier-Stokes equations using the projection method as a preconditioner. *Journal of Computational Physics, 228*(20), 7565–7595. DOI: http://doi.org/10.1016/j.jcp.2009.07.001

Guermond, J. L., Minev, P., & Shen, J. (2006). An overview of projection methods for incompressible flows. *Computer Methods in Applied Mechanics and Engineering, 195*(44), 6011–6045. DOI: http://doi.org/10.1016/j.cma.2005.10.010

Liu, H., Yang, B., & Chen, Z. (2015). Accelerating algebraic multigrid solvers on NVIDIA GPUs. *Computers & Mathematics with Applications, 70*(5), 1162–1181. DOI: http://doi.org/10.1016/j.camwa.2015.07.005

Marchi, C. H., Araki, L. K., Alves, A. C., Suero, R., Gonçalves, S. F. T., & Pinto, M. A. V. (2013). Repeated Richardson extrapolation applied to the two-dimensional Laplace equation using triangular and square grids. *Applied Mathematical Modelling, 37*(7), 4661- 4675. DOI: http://doi.org/10.1016/j.apm.2012.09.071

Marchi, C. H., Martins, M. A., Novak, L. A., Araki, L. K., Pinto, M. A. V., Gonçalves, S. d. F. T., ... Freitas, I. d. S. (2016). Polynomial interpolation with repeated Richardson extrapolation to reduce discretization error in CFD. *Applied Mathematical Modelling, 40*(21-22), 8872-8885. DOI: http://doi.org/10.1016/j.apm.2016.05.029

Martins, M. A., & Marchi, C. H. (2008). Estimate of iteration errors in computational fluid dynamics. *Numerical Heat Transfer, Part B: Fundamentals – An International Journal of Computation and Methodology, 53*(3), 234–245. DOI: http://doi.org/10.1080/10580530701790142

Mitin, A. V. (1985). Linear extrapolation in an iterative method for solving systems of equations. *USSR Computational Mathematics and Mathematical Physics, 25*(2), 1–6.

DOI: http://doi.org/10.1016/0041-5553(85)90097-7

Oliveira, F., Pinto, M. A. V., Marchi, C. H., & Araki, L. K. (2012). Optimized partial semicoarsening multigrid algorithm for heat diffusion problems and anisotropic grids. *Applied Mathematical Modelling, 36*(10), 4665–4676. DOI: http://doi.org/10.1016/j.apm.2011.11.084

Oliveira, M. L. d., Pinto, M. A. V., Gonçalves, S. d. F. T., & Rutz, G. V. (2018). On the robustness of the xy-Zebra-Gauss-Seidel smoother on an anisotropic diffusion problem. *CMES – Computer Modeling in Engineering & Sciences, 117*(2), 251–270. DOI: http://doi.org/10.31614/cmes.2018.04237

Trottenberg, U., Oosterlee, C., & Schuller, A. (2001). *Multigrid*. London, UK: Academic Press.

Versteeg, H. K., & Malalasekera, W. (2007). *An introduction to computational fluid dynamics: the finite volume method*. London, UK: Pearson Education.

Zhang, W., Zhang, C. H., & Xi, G. (2010). An explicit Chebyshev pseudospectral multigrid method for incompressible Navier-Stokes equations. *Computers & Fluids, 39*(1), 178–188. DOI: http://doi.org/10.1016/j.compfluid.2009.08.001