Integrated Solver for Numerical Simulations and Mathematical Modeling in Engineering Physics

MERGU SRAVANTHI[1]| P RAMESH[2]| Kavitha[3]| PASHAM.SRAVANI[4]
1, 2 & 3 Assistant Professor, Physics department, Kasireddy Narayanreddy College of Engineering And Research, Hyderabad, TS.
4 UG SCHOLAR, Physics department, Kasireddy Narayanreddy College of Engineering And Research, Hyderabad, TS.

ABSTRACT:

his paper introduces a two-grid methodology aimed at creating a black-box iterative solver suitable for a diverse range of practical applications within continuum mechanics, including heat and mass transfer, fluid dynamics, elasticity, and electromagnetism. The primary design goals for this (non-)linear black-box solver include: (1) robustness, by minimizing the reliance on problem-specific components, (2) efficiency, by achieving algorithmic complexity close to optimal levels, and (3) parallelism, ensuring that a parallel robust algorithm outperforms the best sequential alternatives. The core concept revolves around utilizing an auxiliary structured grid to handle more computationally demanding tasks, where (non-)linear problems can be more easily solved and parallelized. This approach effectively merges the benefits of unstructured and structured grids: facilitating simpler generation for complex domain geometries and enabling the solution of (non-)linear (initial-)boundary value problems through the Robust Multigrid Technique. This article covers an in-depth description of the two-grid algorithm, along with assessments of its robustness, convergence, algorithmic complexity, and parallelism. The ongoing advancement of modern software for addressing real-life challenges underscores the significance of this research. The proposed two-grid algorithm can be integrated into black-box parallel software systems, leading to reduced execution times for solving (initial-)boundary value problems.

Keywords: mathematical modeling, parallel computing, high-performance computing, multigrid methods, black-box software, multiphysics simulation, practical applications

INTRODUCTION

Mathematical modeling of physical and chemical processes is a fundamental endeavor in science and engineering. However, the intricacies of mathematical models, numerical algorithms, parallel computing technologies, and supercomputer architectures often surpass the comprehension of many scientists and engineers. This challenge highlights the need for black-box software solutions. Across several industries, as well as in engineering and consulting firms globally, there is a reliance on commercially available general-purpose computational fluid dynamics (CFD) codes for simulating fluid flow, heat and mass transfer, and combustion in aerospace applications, including tools like Fluent, Star-CCM+, COMSOL's CFD Module, and Altair's AcuSolve. Additionally, numerous universities and research institutes utilize these commercial codes alongside custom-

developed solutions. Open-source alternatives, such as OpenFOAM, have also emerged, providing accessible options.

Key considerations in this domain include accurately representing complex geometries and generating appropriate computational grids. However, effectively deploying these codes and interpreting the results necessitates a solid understanding of the foundational concepts behind computational methods. A promising trend in numerical simulation and scientific computing is the development of a single code capable of addressing a multitude of previously solved problems. Typically, the mathematical modeling process involves the following stages:

1. Formulating the mathematical model for the relevant physical and chemical processes in the form $N(u) = f$;
2. Approximating the space-time continuum through the generation of a computational grid G;
3. Approximating the differential problems (1) on grid G to derive a discrete analogue of the mathematical model $N_h(u_h) = f_h$;
4. Obtaining a numerical solution for the (non-)linear discrete equations $u_h = N_h^{-1} f_h$ on either sequential or parallel computers;
5. Visualizing and analyzing the computational results.

In this context, $N(u) = f$ represents a system of (non-)linear partial differential equations (PDEs) and (initial-)boundary conditions, while $N_h(u_h) = f_h$ denotes the resulting system of (non-)linear algebraic equations, serving as the discrete analogue of the mathematical model. The solution $u_h = N_h^{-1} f_h$ represents the numerical result. Unfortunately, each stage of mathematical modeling presents complex challenges that remain inadequately addressed. The most time-consuming phase is the numerical solution of the (non-)linear discrete equations.

The term "black-box solver" has evolved to describe algorithms that resolve a system of linear algebraic equations based solely on the matrix formulation $Ax = b$—requiring only the coefficient matrix A, the right-hand side vector b, and an initial guess $x^{(0)}$ for the solution $A^{-1}b$. This concept can also extend to solving linear problems or globally linearized nonlinear problems without requiring geometric input. Monolithic methods applied across the entire system have demonstrated robust convergence for saddle point problems and multiphysics simulations. A particularly elegant approach to constructing monolithic algorithms leverages finite volume methods with careful ordering of unknowns based on geometric data from the computational grid.

However, creating an efficient monolithic approach for navigating systems of strongly coupled nonlinear PDEs solely through local linearization presents considerable difficulties.

We define software as a black-box if it necessitates minimal input from the user, requiring only a specification of the physical problem, including domain geometry, boundary and initial conditions, source terms, and the identification of the equations to be solved (such as heat conduction, Navier–Stokes, Maxwell equations, etc.). The user need not possess knowledge of numerical methods or the intricacies of high-performance parallel computing. The aim of robust algorithms is to select components that are problem-independent, thereby maximizing the solver's applicability across a broad range of issues.

Despite having a multitude of mathematical models for various physical and chemical processes for multiphysics simulation, along with various methods for generating adaptive unstructured or (block-)structured grids, and numerous iterative methods for parallel segregated/coupled solutions, combining these elements into a unified black-box solver for a wide range of real-life challenges remains a significant task. Notably, execution time heavily relies on the computational algorithm employed for solving real-world problems in parallel.

The ongoing development of classical solvers often seeks to enhance either robustness, convergence rate, or the efficiency of parallel algorithms. We propose that developing a black-box solver should aim to simultaneously improve all three aspects—robustness, convergence rate, and efficiency—despite these requirements traditionally being mutually exclusive.

This paper delves into the mathematical foundations necessary for the development of such a black-box solver. We begin by defining the challenges associated with creating this type of solver, followed by a presentation of a two-grid algorithm integrating both original and auxiliary grids for addressing nonlinear (initial-)boundary value problems. The Robust Multigrid Technique (RMT) for resolving nonlinear (initial-)boundary value problems on the auxiliary grid is subsequently outlined. Our goal is to devise a black-box computational methodology capable of paralleling the solution of a diverse array of practical problems, ranging from the Poisson equation to systems of nonlinear strongly coupled partial differential equations—particularly in complex geometric domains.

The article's structure is as follows: Section 1 highlights key concerns related to the advancement of black-box software for scientific and technical calculations. Section 2

delineates the general requirements for black-box solvers, encompassing robustness, complexity, and parallelism. Section 3 explains the two-grid algorithm with an auxiliary structured grid, aimed at simplifying the coupled solutions of nonlinear (initial-)boundary value problems. Section 4 details the application of Robust Multigrid Technique on the auxiliary grid. A discussion on multigrid methods is featured in Section 5. Finally, Section 6 summarizes the advantages of the two-grid algorithm.

1. GENERAL REQUIREMENTS ON BLACK-BOX SOLVER

To establish a clear understanding of black-box solvers, it is essential to examine various computational aspects of real-life problems and highlight the most significant features we consider important:

1. Grid Generation and Adaptive Refinement in Complex Domains:Creating unstructured automatic meshes is often simpler than generating (block-)structured grids for complicated domains. However, structured grids facilitate the construction of more efficient solvers. Adaptivity and parallelism are vital numerical principles that, while important, can conflict with each other.

2. Multiphysics Simulation: This involves the numerical analysis of multiple simultaneous physical and chemical phenomena—such as heat transfer, fluid flow, deformation, electromagnetics, acoustics, and mass transport. The nonlinear partial differential equations (PDEs) that describe these phenomena can be solved in either a coupled or decoupled manner. The differences in computational workload between coupled and decoupled iterations cannot always be predetermined and are typically identified during the iterative solution process.

3. Stationary vs. Non-Stationary Solutions: The initial-boundary value problems associated with time-dependent PDEs pose significant scientific and practical challenges. Reasons for this include:

- Certain physical processes, like turbulence, are inherently non-stationary and three-dimensional.

- Initial-boundary value problems are particularly beneficial for solutions exhibiting unsteady behavior, which cannot be forecasted. Steady-state solutions may be approached using pseudo-time marching techniques. Semi-implicit or fully implicit discretizations allow for larger, adaptable time steps, making parallel processing viable across both space and time.

- Systems of strongly coupled nonlinear PDEs often arise in multiphysics simulations, where time steps can also function as under-relaxation factors to help control the convergence of nonlinear iterations.

Given these considerations, a time-dependent formulation of PDEs is generally more suitable for black-box implementations. The computational algorithm must remain efficient for varying grid aspect ratios in time and space.

From our perspective, a true black-box algorithm should possess key properties such as robustness and optimal computational efficiency. Robustness is defined as the algorithm's resilience, characterized by the minimal number of problem-dependent components across algorithms within the same category. Optimal computational work refers to the capability to solve a variety of problems within truncation error using \(CN \) arithmetic operations, where \(N \) represents the number of unknowns.

Developing black-box solvers for multidisciplinary applications, while simultaneously addressing the "robustness–efficiency–parallelism" challenge, presents a novel obstacle within scientific computing. Unfortunately, a perfect black-box solver is unattainable since these requirements can inherently conflict. Therefore, adjustments to the "robustness–efficiency–parallelism" demands are necessary to move toward a close-to-black-box solver.

Algorithmic Complexity (W) serves as a metric for comparing the efficiency of computational algorithms, typically measured by the number of arithmetic operations required to resolve a problem. We will begin our exploration with linear PDEs due to their relative simplicity. Let \(N \) represent a linear elliptic operator (e.g., a Laplace operator) defined over a \(d \)-dimensional unit cube \(\Omega \) (where \(d = 2, 3 \)). A uniform computational grid \(G \) is created by dividing each edge of the cube into \(n \) subintervals, leading to the discretization parameter \(n \) and mesh size \(h = 1/n \).

Some approximation of the mathematical problem leads to a discrete analogue. Using an appropriate ordering of unknowns allows us to rewrite the linear discrete problem in a matrix representation:

$$A\mathbf{u} = \mathbf{b}, \tag{4}$$

where A is a coefficient matrix, u is a vector of unknowns, and b is a right-hand side vector. General linear iterations for solving the system (4) can be represented in the form

$$W\left(\mathbf{u}^{(s+1)} - \mathbf{u}^{(s)}\right) = \mathbf{b} - A\mathbf{u}^{(s)}, \quad s = 0, 1, 2, \ldots, \tag{5}$$

where the splitting matrix W defines a basic linear iterative algorithm

$$\mathbf{u}^{(s+1)} = \left(I - W^{-1}A\right)\mathbf{u}^{(s)} + W^{-1}\mathbf{b},$$

where I is the unity matrix, and s is the iteration counter. In the following, we will assume that the system (4) has an unique solution u = A −1b and iterations (5) converge to this solution: u (s) → u = A −1b for s → +∞. The basic linear algorithm (5) for iterative solving (4) has three problem-dependent components: the ordering of unknowns, the splitting matrix W and a stopping criterion for these iterations. For estimating the algorithmic complexity, we assume that a block ordering of the unknowns is used, i.e., the number of unknowns becomes

$$N = n^d = n_b N_b,$$

where nb is the number of blocks, Nb is the number of unknowns forming each block, N = n d , d = 2, 3 is the number of unknowns, and n is the discretization parameter (h = 1/n). The computational cost of each Vanka-type iteration (block Gauss–Seidel method used for the coupled numerical solution of systems of PDEs including the saddle-point problems [16,17]) is

$$\mathcal{W}_1 = Cn_b N_b^3 = Cn_h^{-2} N^3$$

arithmetic operations (ao), where C is some constant. The number of iterations (5) can be estimated as

$$\Theta = n^{\varkappa} = N^{\varkappa/d},$$

where the parameter κ depends on the condition number of the coefficient matrix A and the block size Nb , d = 2, 3. Then the algorithmic complexity of the block iterative method becomes

$$\mathcal{W} = \Theta \mathcal{W}_1 = Cn_b^{-2} N^{3+\varkappa/d} \text{ ao.} \qquad (6)$$

Use of the uniform grid in the above-mentioned linear analysis makes it possible to obtain the expression (6) for estimating the computational work. If nb = 1, then the block iteration coincides with the Gaussian elimination

$$\varkappa = 0 \Rightarrow \mathcal{W} = CN^3 \text{ ao},$$

i.e., the complexity W is κ-independent and

$$n_b \to 1 \Rightarrow \varkappa \to 0.$$

The Gaussian elimination with partial/complete pivoting is implemented without the problem-dependent components, but large complexity (W = O(N3) ao) allows this direct method to be used for solving small systems of linear algebraic Equations (SLAEs). The point ordering of an unknown corresponds to nb = N. In this case, the algorithmic complexity becomes

$$\mathcal{W} = CN^{1+\varkappa/d} \text{ ao.}$$

It is expected that the point iterative method (5) will be faster than the Gaussian elimination for sufficiently large N, i.e., $0 \le \kappa < 2d$. The parameter κ depends on the coefficient matrix A: $\kappa \to 0$ for well-conditioned problems and the point iterative method has almost optimal algorithmic complexity

$$\mathcal{W} \to CN \text{ ao for } \varkappa \to 0.$$

As a rule, it is not useful to accelerate a highly efficient solver. The extra effort does not pay off.

Thus, the simplest problem of constructing a robust iterative algorithm for numerical solving linear (initial-)boundary value problems on a uniform grid can be formulated as follows:

(1) If A in (4) is a well-conditioned coefficient matrix ($\kappa \to 0$), then the robust iterative algorithm must coincide with the basic linear algorithm (5);

(2) If A in (4) is an ill-conditioned coefficient matrix ($0 < \kappa < 2d$), then it is necessary to add the lowest number of problem-dependent components to the basic linear algorithm (5) to:

(a) Reduce the algorithmic complexity (6) down to a close-to-optimal value

$$W = Cn_b^{-2}N^3 \log N \text{ ao}, \qquad 1 \ll n_b \leq N \qquad (7)$$

in sequential implementation;

(b) Ensure that a parallel algorithm should be faster than the fastest sequential solver. The above considerations imply that it is necessary to coupled consider the two requirements of close-to-optimal complexity (7) and parallelism. For the given purpose, the execution time of a parallel close-to-black-box algorithm should be compared with the execution time of the fastest (optimal) sequential algorithm. Let

$$W_o = C_o n_b^{-2} N^3 \text{ ao} \qquad (8)$$

be the algorithmic complexity of optimal solver and

$$W_p = C_p \frac{1}{p} n_b^{-2} N^3 \log N \text{ ao}$$

be the algorithmic complexity of fully parallel close-to-black-box solver (7). Here, p is the number of independent computing units in parallel implementation and Co and Cp are some constant. Assuming that the execution time is proportional to the complexity in the algorithm considered ($T \sim W$), we have

$$S = \frac{T_o}{T_p} \approx \frac{C_o}{C_p} \frac{p}{\log N},$$

where S is the speed-up of the parallel solver over the optimal one, To is the execution time of the sequential optimal algorithm, and Tp is the execution time of the parallel close-toblack-box algorithm, N and p are the number of unknowns and independent computing units, respectively. If $Co \approx Cp$ then

$$S > 1 \Rightarrow p > \log N.$$

From the above results and considerations, one can conclude that parallel implementation of the close-to-black-box algorithm (7) will not lead to impressive reduction in the execution time as compared with the optimal sequential one (8) with N large. Constructing the iterative algorithm for numerically solving nonlinear (initial-boundary value problems remains the same:

(1) If the sequential Newton-type iterations converge slowly, then convergence should be accelerated up to close-to-optimal value (7) using the least number of extra problemdependent components;

(2) The parallel nonlinear algorithm should be faster than the fastest sequential one.

In general, development of the robust algorithm is more difficult than that of solving linear problems on a uniform grid. We summarize these considerations as follows:

(1) As a rule, systems of nonlinear strongly coupled PDEs in complex domains (multiphysics simulation) are needed to solve in a (de)coupled manner for industrial applications, so theoretical analysis of algorithmic complexity such as (6) becomes more difficult;

(2) Simplicity of Gauss–Seidel iterations makes this algorithm attractive for smoothing in low-memory sequential or parallel multigrid. For real-life applications, it is far from trivial to choose optimal robust algorithm components uniformly for a large class of problems. In many cases, the Krylov subspace methods may have advantages. Therefore, each iterative algorithm for the numerical solution of nonlinear (initial-)boundary value problems has at least three problem-dependent components: the ordering of unknowns, (de)coupled iterations for a locally/globally linearised discrete problems and a stopping criterion for this iterative process. As a result, a black-box solver requires black-box optimization (i.e., the optimal choice of the problem-dependent components of the robust algorithm for the given problem without user control).

The question remains: Is it possible to construct a close-to-optimal black-box solver (the least number of problem-dependent components, close-to-optimal complexity (7), the parallel algorithm should be faster than the fastest sequential one) instead of the true black-box one (absence of the problem-dependent components, optimal complexity (8), and full parallelism)? Yes! First, a general computational approach for combining the advantages of unstructured and structured grids (two-grid algorithm) will be presented. Second, a robust method for solving the discrete initial-boundary value problems on the auxiliary structured grid will be analysed.

2. TWO-GRID ALGORITHM

Close-to-black-box solver can be constructed by combining the advantages of unstructured and structured grids: simplicity of automatic generation in complex domain geometry and opportunity to solve nonlinear (initial-)boundary value problems by very efficient geometric multigrid methods in parallel (de-)coupled manner. The Auxiliary Space Method is a (non-)nested two-level preconditioning technique based on a simple relaxation scheme (smoother) and an auxiliary space (here a structured grid is the auxiliary space). The basic idea of the Auxiliary Space Method is to use an auxiliary (non-)linear problem in the auxiliary space, where it is simpler to solve [18,19]. The solution of auxiliary problem (auxiliary grid correction) is then transferred back to the original space. The mismatch between auxiliary space and the original space is corrected by a few smoothing iterations. For reason of simplicity, we consider a linear boundary value problem

$$\mathcal{L}(u) = f \qquad (9a)$$

on domain $\Omega \in Rd$, together with a set of appropriate boundary conditions

$$\mathcal{L}_{\partial\Omega}(u) = g \qquad (9b)$$

at the domain boundary $\partial\Omega$. Here, L is a linear elliptic operators, f is a known functions and u is the desired solution. Let $u^$ be an approximation to the solution u and $c = u - u^$ is a correction, i.e., difference between the solution and the approximation to it. The representation of the solution

$$u = \hat{u} + c, \qquad (10)$$

is called Σ-modification of the solution [15,20]. Substitution of (10) into (9) leads to Σmodified form of this problem

$$\mathcal{L}(c) = f - \mathcal{L}(\hat{u}), \quad (11a)$$
$$\mathcal{L}_{\partial\Omega}(c) = g - \mathcal{L}_{\partial\Omega}(\hat{u}). \quad (11b)$$

Σ-modification can be used for solving some nonlinear problems (for example, the Navier–Stokes equations), but Π-modification u = uˆ · c can be preferable for another nonlinear problems. The general approach for solving the nonlinear problems solution is Full Approximation Storage scheme [3].

Let an original (un-)structured grid Go and an auxiliary structured grids Ga be generated in the domain Ω. Figures 1 and 2 represents example of such computational grids. Approximation of (11) on these grids Go and Ga leads to the discrete problems written in the matrix form (with the eliminated boundary conditions):

(a) the original grid G_o

$$A_o \mathbf{c}_o^h = \mathbf{f}_o^h - A_o \hat{\mathbf{u}}_o^h, \quad (12)$$

(b) the auxiliary grids G_a

$$A_a \mathbf{c}_a^h = \mathbf{f}_a^h - A_a \hat{\mathbf{u}}_a^h. \quad (13)$$

It should be emphasized that the problem (11) is discretized on these grids Go and Ga independently. So, the systems (12) and (13) are independent from each other. It simplifies the coupled iterative solution of systems of PDEs. For interface between (12) and (13), a restriction operator Ro→a transferring the residual f h o − Ao ˆu h o from the grid Go onto the grid Ga

$$\mathbf{f}_a^h - A_a \hat{\mathbf{u}}_a^h = \mathcal{R}_{o\to a}\left(\mathbf{f}_o^h - A_o \hat{\mathbf{u}}_o^h\right), \quad (14)$$

and a prolongation operator Pa→o transferring the correction c h a from the grid Ga onto the grid G

$$\mathbf{c}_o^h = \mathcal{P}_{a\to o}\mathbf{c}_a^h \quad (15)$$

should be defined. Figure 3 demonstrates an example of the transfer operators Ro→a and Pa→o . Using (14) and (15), the correction c h o can be computed as

$$\mathbf{c}_o^h = \mathcal{P}_{a\to o} A_a^{-1} \mathcal{R}_{o\to a}\left(\mathbf{f}_o^h - A_o \hat{\mathbf{u}}_o^h\right).$$

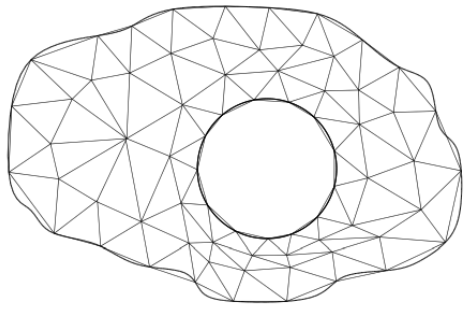

Figure 1. Example of the original (un-)structured grid Go

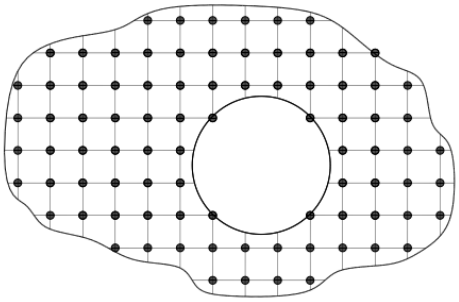

Figure 2. Example of the auxiliary structured grids Ga

Figure 4 represent the linear two-grid algorithm

1. Transfer of the residual f h o − Ao ˆu (q) o to the auxiliary grid Ga , where q is a counter of the intergrid iteration
2. 2. Solution of the auxiliary system by some numerical method

$$A_a \mathbf{c}_a = \mathcal{R}_{o\to a}\left(\mathbf{f}_o^h - A_o \hat{\mathbf{u}}_o^{(q)}\right) \Rightarrow c_a;$$

3. Prolongation of the correction ca to the original grid G

$$\mathbf{c}_o = \mathcal{P}_{a\to o}\mathbf{c}_a,$$

where Pa→o is a prolongation operator transferring the correction to Go ;

4. Computation of the starting guess for the smoothing iterations on G

$$\hat{\mathbf{u}}_o^{(0)} = \mathbf{c}_o + \hat{\mathbf{u}}_o^{(q)};$$

5. Smoothing iterations on the original grid G

$$W_o\left(\hat{\mathbf{u}}_o^{(s+1)} - \hat{\mathbf{u}}_o^{(s)}\right) = \mathbf{f}_o^h - A_o \hat{\mathbf{u}}_o^{(s)}, \quad s = 0, 1, 2, \ldots,$$

where s is the smoothing iteration counter;

6. Updating the approximation to the solution

$$\hat{\mathbf{u}}_o^{(q+1)} = \hat{\mathbf{u}}_o^{(s+1)},$$

where q is the intergrid iteration counter;

7. check convergence, repeat if necessary

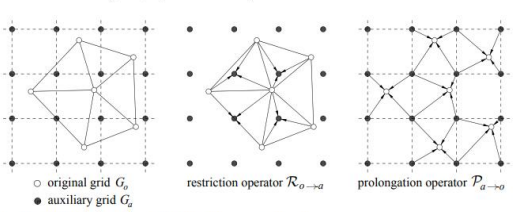

Figure 3. Illustration of the restriction and prolongation operators.

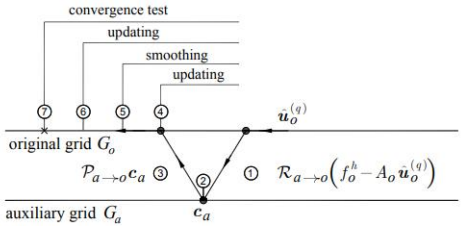

Figure 4. Linear two-grid algorithm

The linear two-grid algorithm can be rewritten in the matrix form

$$\mathbf{f}_o^h - A_o\hat{\mathbf{u}}_o^{(q+1)} = M(\mathbf{f}_o^h - A_o\hat{\mathbf{u}}_o^{(q)}), \qquad (16)$$

where the iteration matrix of the linear two-grid algorithm is

$$M = A_o S_o^{\nu}\left(A_o^{-1} - \mathcal{P}_{a\to o}A_a^{-1}\mathcal{R}_{o\to a}\right), \qquad (17)$$

and So = I − W−1 o Ao is a smoothing iteration matrix and ν is the smoothing iterations counter (kSok < 1).

The convergence properties of the linear two-grid algorithm can be easily analysed by considering factor ρq of the averaged reduction in the residual r (q) o = f h o − Ao ˆu (q)

$$\rho_q = \left(\frac{\|\mathbf{f}_o^h - A_o\hat{\mathbf{u}}_o^{(q)}\|}{\|\mathbf{f}_o^h - A_o\hat{\mathbf{u}}_o^{(0)}\|}\right)^{1/q}, \qquad (18)$$

which shows the averaged reduction in the residual over q integrid iterations [21]. The classical multigrid theory is based on the approximation and smoothing property as introduced by W. Hackbusch [22]:

1. Smoothing property: a monotonically decreasing function η(ν) : R+ → R+ exist such that η(ν) → 0 for ν → ∞ an

$$\|A_o S_o^{\nu}\| \leqslant \eta(\nu)\|A_o\|, \qquad (19)$$

$$\eta(\nu) \sim \begin{cases} 1/\nu, & \text{for symmetric problems} \\ 1/\sqrt{\nu}, & \text{for non-symmetric problems} \end{cases}.$$

2. Approximation property: there exists a constant CA > 0 such that

$$\|A_o^{-1} - \mathcal{P}_{a\to o}A_a^{-1}\mathcal{R}_{o\to a}\| \leqslant C_A\|A_o\|^{-1}. \qquad (20)$$

The smoothing property states, in principle, that the smoother reduces the highfrequency components of the error (without amplifying the low-frequency components). The approximation property requires the coarse grid correction to be reasonable [22]. The basis of this theory is a splitting (factorization) of the two-grid iteration matrix M (17).

Theorem 1.

Assuming that the smoothing property (19) and approximation property (20) hold, then the N-independent convergence of the intergrid iterations (16) follows immediately for ν that is large enough.

Proof of Theorem 1. The intergrid iterations (16) can be rewritten as

$$\mathbf{f}_o^h - A_o\hat{\mathbf{u}}_o^{(q)} = M^q(\mathbf{f}_o^h - A_o\hat{\mathbf{u}}_o^{(0)}).$$

This leads to the following estimation

$$\|\mathbf{f}_o^h - A_o\hat{\mathbf{u}}_o^{(q)}\| \le \|M\|^q \|\mathbf{f}_o^h - A_o\hat{\mathbf{u}}_o^{(0)}\| \Rightarrow \rho_q \le \|M\|.$$

The splitting of the two-grid iteration matrix M (17), the smoothing property (19), and approximation property (20) lead to the estimation

$$\rho_q \le \|M\| \le \|A_o S_o^v\| \cdot \|A_o^{-1} - \mathcal{P}_{a\to o} A_a^{-1} \mathcal{R}_{o\to a}\| \le \eta(v)\|A_o\|C_A\|A_o\|^{-1} = C_A\eta(v).$$

Since η(ν) is the monotonically decreasing function, it should be noted that

$$\rho_q \le C_A\eta(v) < 1$$

r a sufficiently large v.

This theorem predicts that the number of intergrid iterations of the two-grid algorithm does not depend on the number of unknowns N, i.e., ρq 6= ρq (N) (18). The main features of the two-grid algorithm can be summarized as follows:

(1) The number of extra problem-dependent components as compared with the basic algorithm (5) are:

(a) A non-nested case: let Go and Ga be the unstructured and structured grid, respectively. The two-grid algorithm has two extra problem-dependent components: transfer operators Ro→a and Pa→o (Figure 3);

(b) A nested case: let Go and Ga be the block-structured grids [9]. In this case, we suppose Ga = Go ⇒ Ro→a = Pa→o = I in absence of smoothing on the original grid (So = I) and the two-grid algorithm has extra problem-dependent components (interblock interpolation);

(c) A nested case: let Go and Ga be the structured grids. In this case, we suppose Ga = Go ⇒ Ro→a = Pa→o = I in absence of smoothing on the original grid (So = I) and the two-grid algorithm has no extra problem-dependent components.

(2) The nonlinear two-grid algorithm based on Full Approximation Scheme approach is given in [15].

(3) The nonlinear two-grid algorithm offers a general possibility to employ low order schemes and obtain high order accuracy (the high order defect correction iteration [3]). Remember that mathematical modelling in continuum mechanics is a chain of approximations: (a) The difference schemes approximate the governing differential Equation (1). (A difference scheme is a finite system of algebraic equations replacing some differential problem);

(b) The differential Equation (1) approximate the fundamental conservation laws of continuum mechanics. (In fact, the hypothesis of continuity prohibits the limit leading to the differential equations);

(c) A continuous medium approximates a real one. Since any chemical reaction is the result of intermolecular interactions, modelling chemical processes in continuum mechanics is only possible by using empirical hypotheses and experimental data to approximate the quantum nature of these intermolecular interactions.

As a rule, the mathematical description errors of physical and chemical processes in real-life problems has a physical nature (inaccurate the initial and\or the boundary conditions, equation state errors, approximate description of the turbulent transport and the chemical reactions, etc.) and they exceed the discretization errors of the governing (integro-)differential Equations [23]. In many cases, the second-order accurate finite volume discretization does not damage the discrete solution accuracy of the mathematical model equations required for practical applications. However, advanced software can use the high-order discretization without significant changes in the computational algorithm. For reasons of robustness, the finite volume method of the second order discretization will be used on the auxiliary grid Ga , but high order discretization approaches can be used on the original grid Go .

(4) The basic ingredients of the two-grid algorithm are computation of correction ca on the auxiliary grid Ga and smoothing iterations on the original grid Ga . The most timeconsuming component of the solver is numerical inversion of the coefficient matrix Aa , i.e., the matrix A −1 a in (17).

(5) The differential problem is approximated on the grids Go and Ga separately in order to simplify coupled iterative solution of systems of PDEs on the auxiliary structured grid Ga (for example, the Vanka-type iterations or the volume-coupled approach used in monolithic algebraic multigrid methods [5]).

The two-grid algorithm puts more computational work on the auxiliary (structured) grid Ga , where the (non-)linear problems are simpler to solve and parallelize. The final effort is the construction of an efficient iterative algorithm for solving the (non-)linear (initial-)boundary value problems on the auxiliary grid Ga .

3. Robust Multigrid Technique

An epochal event in world computational mathematics was publication of R.P. Fedorenko's paper (Keldysh Institute of Applied Mathematics of Russian Academy of Sciences, Moscow, USSR/Russia) in 1961 [24], where the author formulated a new iterative method for solving discrete boundary value problems (BVPs) on structured grids (https://team.kiam.ru/botchev/fedorenko/, accessed on 1 August 2023). Thoughtful conclusions on the basis of elementary analysis were far ahead of their time, and after many years this paper was called the "first true multigrid publication" in the scientific literature. The first theoretical results were reported in the pioneering papers of N.S. Bakhvalov and G.P. Astrakhantsev. At the end of the 1970s and at the beginning of the 1980s research on the multigrid methods increased. Very interesting multigrid approaches were proposed and developed in the Theoretical Division of the Los Alamos National Laboratory, USA. In papers [25,26], P.O. Frederickson and O.A. McBryan studied the efficiency of the Parallel Superconvergent Multigrid Method (PSMG). The basic idea behind the PSMG is the observation that for each fine grid there are two natural coarse grids—the even and odd points of the fine grid (Figure 5). The authors tried to develop a optimized multigrid algorithm by combining these coarse grid solutions for more accurate correction.

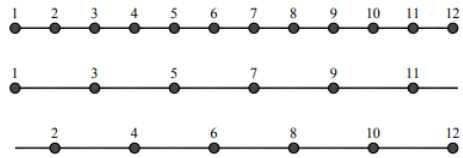

Figure 5. Multiple coarsening strategy used in PSMG.

Although P.O. Frederickson and O.A. McBryan restrict themselves to a theoretical analysis of the PSMG, they demonstrate that besides numerical efficiency, the algorithm is also highly parallel. The PSMG and related ideas essentially refer to massively parallel computing. However, combinations or the extrapolation of the coarse grid corrections is a very efficient approach only for the simplest BVPs. Also in 1990, a similar multiple coarse grid correction strategy had been proposed for the development of a robust multigrid method for black-box software in Baranov Central Institute of Aviation Motors, Moscow, USSR/Russia. A developed solver is called Robust Multigrid Technique (RMT). The RMT uses a multiple coarsening strategy coupled with the finite volume discretization in order to obtain the problem-independent transfer operators and coarse grid operator (i.e., the matrix Aa in (13)), high parallel efficiency, and to make the smoother's task the least demanding. The history of RMT is given in [20], and for the theoretical description of RMT and corresponding parallel analysis, we refer interested readers to [6,15,20,27].

The uniform finest grid G 0 1 consists of two sets of points G v (0;1) and G f (0;1):

$$G^{\mathrm{v}}(0;1) = \{x_i^{\mathrm{v}} \mid x_i^{\mathrm{v}} = h(i-1), \quad i = 1,2,\ldots,n^0+1, \quad h = 1/n_1^0\},$$
$$G^{\mathrm{f}}(0;1) = \{x_i^{\mathrm{f}} \mid x_i^{\mathrm{f}} = (x_i^{\mathrm{v}} + x_{i+1}^{\mathrm{v}})/2, \quad i = 1,2,\ldots,n_1^0\},$$

where the discretization parameter n 0 1 defines the finest grid G 0 1 . Figure 6 represents the finest uniform grid G 0 1 = G v (0;1) ∪ G f (0;1) generated with n 0 1 = 8 or mesh size h = 1/8 in the unit segment: x v 1 = 0, x v 9 = 1.

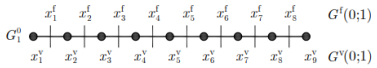

Figure 6. Uniform finest grid $G_1^0 = G^{\mathrm{v}}(0;1) \cup G^{\mathrm{f}}(0;1)$ for the finite volume discretization ($n_1^0 = 8$).

Each d-dimensional computational grid used in RMT can be represented as product of d one-dimensional grids, so the one-dimensional grid G 0 1 = G v (0;1) ∪ G f (0;1) will be considered in detail. Figure 7 represents triple coarsening of RMT. This triple coarsening is independent of the assignment of grid functions to points x v i (x v i are the vertices, x f i are the finite volume faces) or to points x f i (x f i are the vertices, x v i are the finite volume faces). This triple coarsening, which does not depend on the configuration of finite volumes, gives a straightforward generalization to multidimensional (un-)staggered discretization of the (initial-)boundary value problems. Later, the triple coarsening was proposed in the Theoretical Division of the Los Alamos National Laboratory [28].

PSMG and RMT are the single-grid algorithms based on the essential multigrid principle of iterations with a basic iterative method on the fine grid [21], but representation of these single-grid algorithms as the multigrid solvers allows analyse their convergence and complexity by elementary methods. The essential multigrid principle is to approximate the smooth (long wavelength) part of the error on coarser grids. The non-smooth or rough part is reduced with a small number (independent of mesh size) h.

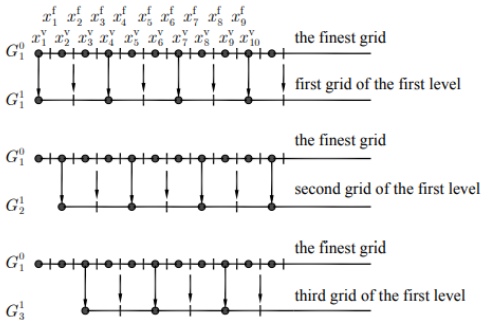

Figure 7. The triple coarsening of RMT.

Figure 7 illustrates the main properties of the coarse grids of RMT: Property 1: all coarse grids G 1 1 , G 1 2 and G 1 3 have no common grid points:

$$G_n^1 \cap G_m^1 = \varnothing, \quad n \neq m.$$

It result in the massive parallelization of computations. Property 2: the fine grid G 0 1 is represented as a union of the coarse grids G 1 1 , G 1 2 and G 1 3 :

$$G_1^0 = \bigcup_{k=1}^{3} G_k^1.$$

It result in the problem-independent prolongation operator P. Property 3: all grids are geometrically similar, but the mesh size of the coarse grids G 1 1 , G 1 2 and G 1 3 is three times larger than the mesh size of the finest grid G 0 1 . It result in the unified finite volume discretization of the modified (initial-)boundary value problems on the multigrid structures, i.e., to the problem-independent construction of the coarse grid operator Aa in (17). Property 4: independent of the grid functions assignment, each finite volume on the coarse grids G 1 1 , G 1 2 and G 1 3 is a union of three finite volumes on the fine grid G 0 1 . It result in the problem-independent restriction operator R based on the additive interval property to evaluate integrals in the finite volume discretization.

The finest grid G 0 1 forms zero grid level, but the coarse grids G 1 1 , G 1 2 and G 1 3 form the first grid level. The following coarsening is carried out recursively: each computational grid G l i , i = 1, . . . , 3l of level l is considered to be the finest grid for three coarse grids of level l + 1. Nine coarser grids obtained from three coarse grids of the first level form a second level as shown in Figure 8. The coarsening stops when the coarse grids will have a few points x v and x f , and further coarsening cannot be performed. The coarsest level will be denoted by L + 3 . The total number of levels (coarse levels L + 3 and the finest grid (zero level)) is L + 3 + 1. The grid hierarchy G l m, m = 1, . . . , 3l , l = 0, . . . , L + 3 will be called a multigrid structure (MS) generated by the grid G 0 1

$$\mathcal{MS}(G_1^0) = \{G_m^l \mid m = 1,2,\ldots,3^l,\ l = 0,1,\ldots,L_3^+\}.$$

Here the coarse grids are used the only for obviousness of the technique description, since RMT is a single-grid algorithm, the index mapping gives the multigrid illusion [6,15,20,27].

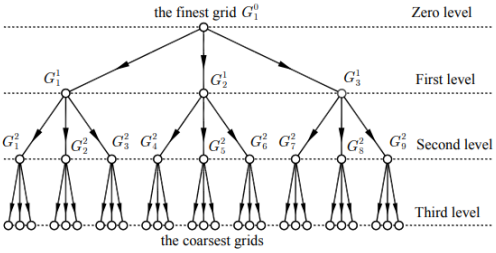

Figure 8. Multigrid structure generated by the finest grid G_1^0.

Figure 9 represents the finest grid G 0 1 (n 0 1 = 30, h = 1/n 0 1 = 1/30) and the coarse grids of the first and second levels. The number of coarse levels L + 3 + 1 can be computed before the generation of a multigrid structure. Assume that many of the coarsest grids of level L + 3 have three points x v or x f . Then the number of points on the finest grid G 0 1 is n 0 1 + 1 ≈ 3 L + 3 +1 or

$$n_1^0 + 1 \approx 3^{L_3^+ + 1} \Rightarrow L_3^+ \approx \left[\log_3 (n_1^0 + 1) - 1\right] \approx \left[\log_3 \frac{n_1^0}{3}\right], \qquad (21)$$

where the square brackets indicate an integer part. The procedure of the fast approximation of integrals on the multigrid structures uses the ghost points of each grid [20,27].

The multigrid schedule of RMT is the V-cycle with no pre-smoothing (the so-called sawtooth cycle [21]). The sawtooth cycle is a special case of the V-cycle, in which smoothing before the coarse grid correction (pre-smoothing) is deleted. The computational cost of each multigrid iteration of RMT can be estimated as

$$\mathcal{W}_q = \mathcal{W}_0(L_3^+ + 1)\ \text{ao},$$

where W0 = CN ao is cost of the finest grid smoothing, N is the number of unknowns, C is some constant, and L + 3 + 1 is the number of levels. Since

$$L_3^+ + 1 \approx \left[\log_3 (n_1^0 + 1)\right] \approx \frac{1}{d}\left[\log_3 N\right]$$

(21) and all grids of the some level have the same number of points, the algorithmic complexity of RMT can be estimated as

$$\mathcal{W}_{\text{RMT}} = q\mathcal{W}_q = q\mathcal{W}_0(L_3^+ + 1) = C\frac{q}{d}N\left[\log_3 N\right] \le C\frac{q}{d}N\log_3 N = O(N\log N)\ \text{ao},$$

i.e., RMT has the required close-to-optimal algorithmic complexity (7). Theoretical analysis predicts that the single-grid RMT has the most attractive property of classic multigrid methods, namely h-independent convergence in general situations [6,15,20,27].

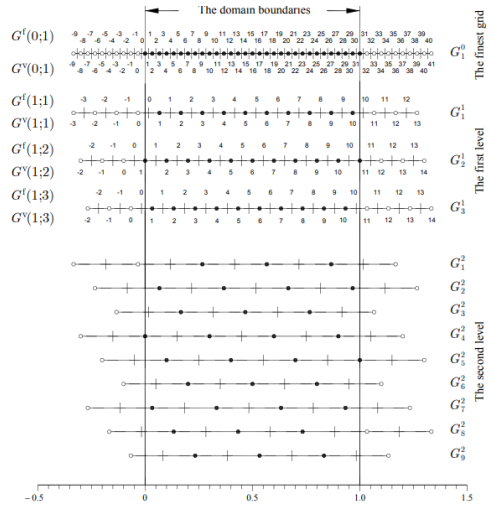

Figure 9. Multigrid structure generated by the finest grid G_1^0 ($n_1^0 = 30$, $h = 1/n_1^0 = 1/30$).

We summarize some well-known facts about sequential RMT here:

(1) RMT is a single-grid algorithm having close-to-optimal algorithmic complexity and h-independent convergence;

(2) RMT uses a multiple coarsening strategy coupled with the finite volume discretization in order to obtain the problem-independent transfer operators and coarse grid operator, high parallel efficiency, and to make the smoother's task the least demanding. The history of RMT is given in [6], for the theoretical description of RMT and corresponding examples and parallel analysis, we refer to [6,15,20,27];

(3) RMT has extra problem-dependent component (the number of smoothing iterations on the coarse levels);

(4) All problem-dependent components of RMT can be optimized on the multigrid structure in the black-box manner. The basic idea of black-box optimization is the experimental study of the iteration convergence rate on a multigrid structure starting from the same initial guess. For example, a discrete problem can be solved using different problem-dependent components on several grids of the same level starting from the same initial guess. Analysis of reduction in the residual norm during the smoothing iterations makes it possible to choose close-to-optimal problem-dependent components of the algorithm. Figure 9 illustrates that the similarity of all grids of the same level leads to almost the same problem-dependent components of the algorithm at this level. It should be emphasized that this black-box optimization does not require any theoretical input or a priori information on problem to be solved. The extra effort for this black-box optimization is negligible compared to the effort for smoothing

Finally, analysis of the parallel RMT completes analysis of the two-grid algorithm. Assuming that the finest grid is deleted, we can solve the discrete problems on 3 d , d = 2, 3 independent coarse grids of the first level. This geometric parallelism of RMT is based on non-overlapping the finest grid partition for distribution of 3 d , d = 2, 3 independent tasks over p = 3 κ , κ = 1, 2, . . . , d computing units. In the following, the coarse grids, which are considered the finest grids in the solution process, will be called dynamic finest grids. The difference between this starting guess and the finest grid numerical solution does not exceed ` significant digits for the second order finite volume discretization, where ` is the serial number of the dynamic level.

To illustrate the parallel RMT, we consider the following example:

$$\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}+\frac{\partial^2 u}{\partial z^2}=-f(x,y,z) \qquad (22)$$

in the unit cube Ω = (0, 1) 3 . If the exact solution is given by

$$u_a(x,y,z)=\exp(x+y+z), \qquad (23)$$

then substitution of (23) into (22) gives the right-hand side function

$$f(x,y,z)=-3\exp(x+y+z)$$

and the Dirichlet boundary conditions. Standard seven-point finite volume discretization of (22) on the uniform grid G 0 1 is abbreviated as

$$\Delta^h u^h=-f^h,$$

where Δ h , u h , and f h are the discrete analogues of the Laplace operator, the solution u, and the right-hand side function f , respectively.

This boundary value problem is solved by RMT with (hx = hy = hz = 1/100) using the stopping criterion

$$\max_{ijk}\left|\Delta^h u^h+f^h\right|<10^{-6}.$$

The error of the numerical solution is defined by comparison of the exact and approximated solutions

$$\|\mathbf{e}\|_\infty=\max_{ijk}\left|u_a(x_i^{\mathrm{v}},y_j^{\mathrm{v}},z_k^{\mathrm{v}})-u_{ijk}^h\right|,$$

where ua and u h are the exact (23) and the numerical solutions, respectively. Figure 10 represents the error of the numerical solutions kek∞ obtained on the multigrid structures MS(G 0 1) and MS(G 1 k), k = 1, . . . , 3d starting with the iterant zero on 101 × 101 × 101 uniform finest grid G 0 1 (n 0 1 = 100, h = 1/100). Taking into account the stop-ping criterion, the iterative solution of the model BVP on the finest grid G 0 1 is a reduction in the error of the zero starting guess (ke (0)k∞ ≈ e 3 ≈ 20) down to kek∞ ≈ 10−6 . Figure 10 illustrates the accuracy of the starting guess to the finest grid solution assembled from the solutions obtained in parallel on the coarse grids of the first level (dynamic finest grids). This geometric parallelism does not require parallelization of the iterative/direct solvers.

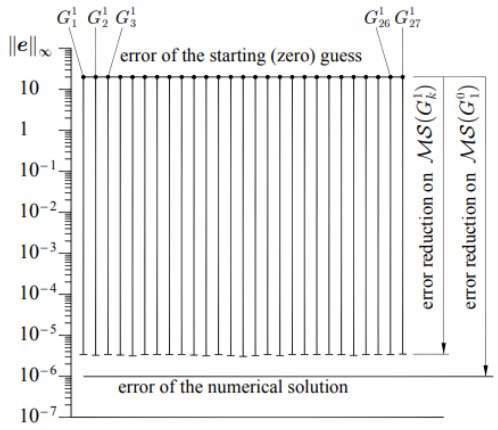

Figure 10. Error of the numerical solutions obtained on the multigrid structures MS(G 0 1) and MS(G 1 k), k = 1, . . . , 27.

The algebraic parallelism of RMT is based on the multicolour orderings of unknowns (or block of unknowns) to parallelize the smoothing iterations on the finer levels where the number of grids is less than the number of computing units. The small-scale granular algebraic parallelism is grid-independent. The solution of the modified boundary value problem starts on the auxiliary grid. 3 d multigrid structures generated by the dynamic grids of the first level makes it possible to obtain accurate approximation to the solution in parallel by handling 3 d independent discrete problems. In addition to black-box optimization of the problem-dependent components, the dynamic grid refinements are carried out during the solution process on the multigrid structures generated by the dynamic grids, controlled by some appropriate adaptation criteria. Figure 11 demonstrates an example of the adaptive grid refinement. Stopping of iterative computation of the auxiliary grid correction on the finest grid means that the sufficiently accurate approximation to the solution on the original grid obtained and subdomains for the grid refinement are determined. The next step is generation of an original (un)structured grid taking into account the subdomains for the auxiliary grid refinement. The auxiliary grid correction is prolonged to the original grid and it is corrected by a few smoothing iterations. The iterative process is continued with restriction of residual to the auxiliary grid if the required accuracy is not yet achieved. Otherwise, the two-grid algorithm stops. Subsequent intergrid iterations are performed without dynamic grids, the black-box optimization and the auxiliary grid refinement. The initial-boundary value problems are solved in the same parallel manner: parallel in space as that for the boundary value problems and parallel in time (waveform relaxation) [15,29,30].

Theoretically, the execution time of the parallel RMT implemented over nine computing units is approximately equal to the execution time of the sequential V-cycle [15].

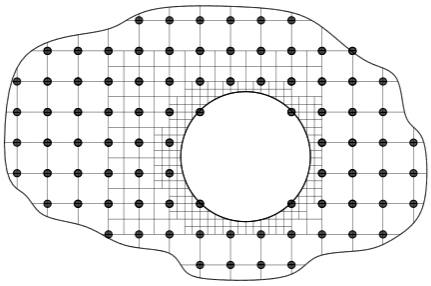

Figure 11. Adaptation of the auxiliary grid shown in Figure 2.

4. Discussion

In our perspective, the design of a black-box algorithm for tackling real-life problems should adhere to several key principles:

1. Dominance of Physical Errors: In many instances, second-order accurate finite volume discretizations do not significantly compromise the accuracy of discrete solutions for real-world scenarios. Advanced software has the potential to implement high-order discretizations without necessitating substantial changes to the underlying algorithm.

2. Formalization of Computations: The black-box algorithm is intended to address nonlinear initial-boundary value problems in a unified manner, building upon established methods for known issues rather than innovating new methods for unprecedented problems. This emphasizes the importance of consolidating existing solutions.

3.Robustness, Efficiency, and Parallelism Requirements: The interplay between robustness, efficiency, and parallelism forms the core of a viable black-box algorithm. Attempts to prioritize one of these requirements at the expense of the others are generally unproductive.

4.(De)coupled Solutions for Complex Problems:The most challenging cases for a black-box algorithm involve strongly coupled systems of nonlinear partial (integro-)differential equations, such as the steady Navier–Stokes equations. A suitable solver must be capable of addressing these systems not only in a segregated manner but also in a coupled manner across both structured and unstructured grids.

While it is evident that a true black-box solver—characterized by entirely problem-independent components, optimal convergence rates, and complete parallelism—remains unattainable, the aforementioned two-grid algorithm serves as a strong approximation of the desired solver for black-box applications. This method minimizes the number of

problem-dependent components while achieving close-to-optimal convergence rates and maintaining high unified parallelism through the Robust Multigrid Technique (RMT) for initial-boundary value problems.

From a scientific standpoint, pursuing the development of additional computational techniques capable of solving a wide range of nonlinear (initial-)boundary value problems in a decoupled or coupled manner provides a compelling area for future exploration, particularly if these new methods can parallel the robustness, efficiency, and parallelism demonstrated by the two-grid algorithm.

CONCLUSIONS

In summary, the two-grid algorithm presents several advantages that align with the established conditions for effective black-box solvers:

1.Robustness: The algorithm features a minimal number of problem-dependent components, making it robust.

2.Efficiency:It achieves close-to-optimal algorithmic complexity, ensuring efficient performance in various computational scenarios.

3. Parallelism: The parallel implementation of the algorithm is designed to outperform the fastest sequential solvers, enhancing computational speed.

When comparing the two-grid algorithm to the basic algorithm, the additional problem-dependent components are as follows:

- Non-Nested Case: For unstructured grid \(G_o \) and structured grid \(G_a \), the two-grid algorithm incorporates three extra components: the transfer operators (\(R_{o \to a} \) and \(P_{a \to o} \)) and the number of smoothing iterations on the auxiliary grid.
- Nested Case with Block-Structured Grids: Assuming \(G_a = G_o \) and \(R_{o \to a} = P_{a \to o} = I \) (with no smoothing on the original grid, \(S_o = I \)), the algorithm has two additional components: interblock interpolation and the number of smoothing iterations on the auxiliary grid.
- Nested Case with Structured Grids: Similarly, when both grids are structured (\(G_a = G_o \) and \(R_{o \to a} = P_{a \to o} = I \)), the algorithm again includes extra components related to the number of smoothing iterations on the auxiliary grid.

Furthermore, the two-grid algorithm allows for elegant large-scale granular geometric parallelization of the Robust Multigrid Technique (RMT) by distributing the \(3d \) independent discrete tasks across \(3d \) computing units. This type of parallelization is smooth and independent of the grid. For small-scale granular algebraic parallelism, implementation occurs at finer levels, which remains grid-independent.

Theoretical analysis suggests that parallel RMT executed over nine computing units achieves performance roughly equivalent to that of a sequential V-cycle in terms of execution time. Initial-boundary value problems can be addressed in a similar parallel fashion to boundary value problems, incorporating parallelism in time as well.

Numerical experiments illustrating the robustness and efficiency of the RMT are documented in the literature, while both sequential and parallel software implementations are presented in relevant sources.

REFERENCES

1. Sedov, L.I. A Course in Continuum Mechanics; Groningen: Wolters-Noordhoff, The Netherlands, 1971; Volume 1.

2. Dendy, J.E. Black box multigrid. J. Comput. Phys. 1982, 48, 366–386. [CrossRef]

3. Trottenberg, U.; Oosterlee, C.W.; Schüller, A. Multigrid; Academic Press: London, UK, 2001.

4. Luo, P.; Rodrigo, C.; Gaspar, F.J.; Oosterlee, C.W. Monolithic multigrid method for the coupled Stokes flow and deformable porous medium system. J. Comput. Phys. 2018, 353, 148–168.

5. Ohm, P.; Wiesner, T.A.; Cyr, E.C.; Hu, J.J.; Shadid, J.N.; Tuminaro, R.S. A monolithic algebraic multigrid framework for multiphysics applications with examples from resistive MHD. Electron. Trans. Numer. Anal. 2022, 55, 365–390. [CrossRef]

6. Martynenko, S.I. The Robust Multigrid Technique: For Black-Box Software; De Gruyter: Berlin, Germany, 2017.

7. Kuzenov, V.V.; Ryzhkov, S.V.; Varaksin, A.Y. Numerical Modeling of Individual Plasma Dynamic Characteristics of a Light-Erosion MPC Discharge in Gases. Appl. Sci. 2022, 12, 3610. [CrossRef]

8. Brown, J.; He, Y.; MacLachlan, S.P.; Menickelly, M.; Wild, S. Tuning multigrid methods with Robust optimization and local Fourier analysis. SIAM J. Sci. Comput. 2021, 43, 109–138. [CrossRef]

9. Kuzenov, V.V.; Ryzhkov, S.V.; Varaksin, A.Y. The Adaptive Composite Block-Structured Grid Calculation of the Gas-Dynamic Characteristics of an Aircraft Moving in a Gas Environment. Mathematics 2022, 10, 2130. [CrossRef]

10. Berg, J.; Nyström, K. A unified deep artificial neural network approach to partial differential equations in complex geometries. Neurocomputing 2018, 317, 28–41. [CrossRef]

11. He, J.; Xu, J. Mgnet: A unified framework of multigrid and convolutional neural network. Sci. China Math. 2019, 62, 1331–1354. [CrossRef]

12. Katrutsa, A.; Daulbaev, T.; Oseledets, I. Deep multigrid: Learning prolongation and restriction matrices. arXiv 2018, arXiv:1711.03825.

13. Frey, P.; George, P.L. Mesh Generation; Wiley: New York, NY, USA, 2010.

14. George, P.L. Automatic Mesh Generation; Wiley: New York, NY, USA, 1991.

15. Martynenko, S.I. Numerical Methods for Black-Box Software in Computational Continuum Mechanics; De Gruyter: Berlin, Germany, in print.