

LNCS 16196

Vladimir Voevodin
Alexander Antonov
Dmitry Nikitenko (Eds.)

Supercomputing

11th Russian Supercomputing Days, RuSCDays 2025
Moscow, Russia, September 29–30, 2025
Revised Selected Papers



Springer

Lecture Notes in Computer Science

16196

Founding Editors

Gerhard Goos

Juris Hartmanis

Editorial Board Members

Elisa Bertino , *Purdue University, West Lafayette, IN, USA*

Wen Gao, *Peking University, Beijing, China*

Bernhard Steffen , *TU Dortmund University, Dortmund, Germany*

Moti Yung , *Columbia University, New York, NY, USA*

The series Lecture Notes in Computer Science (LNCS), including its subseries Lecture Notes in Artificial Intelligence (LNAI) and Lecture Notes in Bioinformatics (LNBI), has established itself as a medium for the publication of new developments in computer science and information technology research, teaching, and education.

LNCS enjoys close cooperation with the computer science R & D community, the series counts many renowned academics among its volume editors and paper authors, and collaborates with prestigious societies. Its mission is to serve this international community by providing an invaluable service, mainly focused on the publication of conference and workshop proceedings and postproceedings. LNCS commenced publication in 1973.

Vladimir Voevodin · Alexander Antonov ·
Dmitry Nikitenko
Editors

Supercomputing

11th Russian Supercomputing Days, RuSCDays 2025
Moscow, Russia, September 29–30, 2025
Revised Selected Papers

Editors

Vladimir Voevodin 
Moscow State University
Moscow, Russia

Alexander Antonov 
Moscow State University
Moscow, Russia

Dmitry Nikitenko 
Moscow State University
Moscow, Russia

ISSN 0302-9743

ISSN 1611-3349 (electronic)

Lecture Notes in Computer Science

ISBN 978-3-032-13126-3

ISBN 978-3-032-13127-0 (eBook)

<https://doi.org/10.1007/978-3-032-13127-0>

© The Editor(s) (if applicable) and The Author(s), under exclusive license
to Springer Nature Switzerland AG 2026

This work is subject to copyright. All rights are solely and exclusively licensed by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

If disposing of this product, please recycle the paper.

Preface

The 11th Russian Supercomputing Days Conference (RuSCDays 2025) was held during September 29–30, 2025. The conference was organized by the Lomonosov Moscow State University, the Supercomputing Consortium of Russian Universities, the Russian Academy of Sciences, and the Moscow Center of Fundamental and Applied Mathematics. The conference organization coordinator was the Moscow State University Research Computing Center. The conference was supported by platinum sponsor ELPITECH, gold sponsor RSC, and silver sponsors BAIKAL Electronics, Intelligent Archive, Megapolis-Telecom, SAVT, and Tantor.

RuSCDays was born in 2015 as a union of several supercomputing event series in Russia and quickly became one of the most notable Russian supercomputing international meetings. The conference caters to the interests of a wide range of representatives from science, industry, business, education, government, and academia – anyone connected to the development or the use of supercomputing technologies. The conference topics cover all aspects of supercomputing technologies: software and hardware design, solving large tasks, application of supercomputing technologies in industry, exaflops-scale computing issues, supercomputing co-design technologies, supercomputing education, and others.

All 94 papers submitted to the conference were reviewed by three referees in the first review round. During single-blind peer reviewing, the papers were evaluated according to their relevance to the conference topics, scientific contribution, presentation, approbation, and related-works description. After notification of conditional acceptance, a second review round was arranged, which aimed at the final polishing of papers and also at the evaluation of the authors' work following revision based on the referees' comments. After the conference, the best 37 papers were carefully selected to be included in this volume.

The proceedings editors would like to thank all the conference committee members, especially the Organizing and Program Committee members, as well as the referees and reviewers for their contributions. We also thank Springer for producing these high-quality proceedings of RuSCDays 2025.

October 2025

Vladimir Voevodin
Alexander Antonov
Dmitry Nikitenko



Experimental Study of Macrogrid Domain Decomposition Methods with Approximate Block Bisection

Alexey Gurin¹ , Valery Il'in², and Ruslan Kardash^{2,3} 

¹ Lavrentyev Institute of Hydrodynamics of the Siberian Branch of the RAS,
Novosibirsk, Russia

² Institute of Computational Mathematics and Mathematical Geophysics SB RAS,
Novosibirsk, Russia

`ilin@sscc.ru`

³ Novosibirsk State University, Novosibirsk, Russia

`r.kardash@g.nsu.ru`

Abstract. This paper develops and experimentally investigates macrogrid domain decomposition methods for solving large systems of linear algebraic equations (SLAEs) with sparse symmetric matrices obtained from grid approximations of multidimensional boundary value problems. The proposed algorithms are based on constructing two-layer macrogrids and special ordering of nodes according to their belonging to different topological primitives of the macro-grid: macro-nodes, macro-edges, macro-faces, and subdomains. With consistent numbering of vector components, the SLAE matrices in the three-dimensional case take a block-tridiagonal form of fourth order. For its solution, an incomplete factorization algorithm is used, based on block bisection of the original matrix and application of parallel direct or preconditioned iterative algorithms in subdomains. The justification of the proposed methods is given for symmetric positive definite (s.p.d.) matrices.

Keywords: Iterative methods · Krylov subspaces · Macrogrid domain decomposition · Block bisection · Approximate factorization · Multilevel preconditioning

1 Introduction

Classical domain decomposition methods (DDM), which initially emerged as a means for theoretical analysis of multidimensional boundary value problems and later as a tool for parallel computing on multiprocessor systems (MPS), are based on reducing one complex problem to several simpler ones by decomposing a large computational domain into multiple intersecting or non-intersecting subdomains. Naturally, this implies solving SLAEs obtained from grid approximations of the original continuous problems. For each subdomain, auxiliary boundary value problems are formulated with additional (interface) boundary

conditions (typically, pairs of Dirichlet and Neumann boundary conditions are used on two contacting subdomains). The idea of parallelization lies in simultaneously solving problems in subdomains on different processors using some external iterative algorithm (for example, the block Jacobi method). Naturally, at each step, information is exchanged between neighboring subdomains. If the number of subdomains M_D increases, one can expect an increase in the speedup of parallel computations $S_p = T_1(A)/T_p(A)$, where $T_p(A)$ is the time to solve problem A on p processors. However, the number of outer iterations typically grows as well, since in one step the obtained perturbation of the approximate solution is transmitted from each subdomain only to its neighbors. Thus, the ultimate goal of DDM is to enhance computational performance when solving resource-intensive multidimensional grid boundary value problems with large SLAE dimension ($10^9 - 10^{10}$ and more) and sparse ill-conditioned matrices, by providing an algorithm where the number of outer iterations weakly depends on the number of subdomains M_D . This problem is mainly technological in nature and for practical problems with real data significantly depends on the types of unstructured grids used, the properties of the formed data structures, and the specifics of mapping parallelizable algorithms onto the architecture of multiprocessor computing systems (MCS) with distributed and hierarchical shared memory.

Due to its relevance, the DDM theme has an extensive specialized literature, including well-known monographs by O. Widlund [34], Yu.V. Vassilevski and M.A. Olshansky [35], A. Quarteroni [30], review papers by Yu.M. Laevsky with A.M. Matsokin and V.P. Il'in [14], as well as original research by Yu.A. Kuznetsov [24], S.V. Nepomnyashchy [26], E.D. Dyakonov [5], P.Bjorstad [32], F. Nataf, M. Gander [7], V.I. Agoshkov, V.I. Lebedev [1], V.D. Korneev [23]. About 30 international conferences have already been held on domain decomposition topics, and there is a vast internet resource DDM.Org.

This work continues previous research by the authors [8, 10], where a principle of decomposing grid computational domains into subdomains was proposed using a separating macrogrid consisting of macro-nodes, macro-edges, macro-faces, and macro-cells. The latter represent subdomains. All nodes of the original tri-dimensional grid are divided into four non-intersecting subsets according to their belonging to the topological primitives of the macrogrid. Note that the set of primitives is universal and does not depend on the grid type. Next, the nodes and corresponding vector components are reordered sequentially by the selected subsets, as a result of which the matrix of the solved SLAE takes a block-tridiagonal form of fourth order. For its solution, "classical" preconditioned iterative methods in Krylov subspaces are applied, see monographs by Y. Saad, M.A. Olshansky and E.E. Tyrtshnikov, V.P. Il'in [13, 28, 31], as well as more recent journal literature and numerous publications cited there [2, 14, 17, 20, 21, 33].

Thus, the proposed approaches consist of two parts. The first (geometric) stage includes procedures for forming block matrix structures, while the second is responsible for constructing computational algebra methods, taking into account the specifics of the problem.

Note that a similar construction of grid subsets by their belonging to topological primitives was proposed by the authors for multigrid order-optimal algorithms, but only at the micro-level rather than the macro-level, [3] and the literature cited there.

An important aspect of the considered approaches is the fact that the macrogrid can be single-layer or two-layer (there is at least a terminological analogy with single-layer and double-layer potentials in the theory of differential equations). A single-layer macrogrid implies that macro-nodes, macro-edges, and macro-faces are also objects of the original grid. In the opposite case, one can consider that subdomains are separated by virtual lines or surfaces passing through the midpoints of the steps of the original grid. This is the case that is investigated in the present work, see details below.

In the considered approaches, there is another significant algorithmic aspect. Traditionally, solving block-tridiagonal SLAEs is performed by sequential elimination, or reduction, of vector components. We, however, consider a block bisection method, where a block matrix of fourth order is reduced to second order, which allows efficient use of the property of finite conditionality of the macrogrid algebraic subsystem.

This work is structured as follows. Section 2 describes the algebraic-geometric structures of macrogrid domain decomposition methods. Section 3 is devoted to the presentation of preconditioned iterative methods in Krylov subspaces using a two-layer separating macrogrid and block bisection. Section 4 discusses the results of numerical experiments with algorithm parallelization. In conclusion, the prospects for developing the proposed methods for solving a wider class of problems are analyzed.

2 Algebraic-Geometric Structures of Macrogrid Domain Decomposition Methods

Consider a real algebraic system,

$$Au = f, \quad A = \{a_{t,s}\} \in \mathcal{R}^{N,N}, \quad u = \{u_t\}, \quad f = \{f_t\} \in \mathcal{R}^N, \quad (1)$$

which, as we assume, is obtained from some grid approximation (finite-difference, finite-volume, or finite-element) of a boundary or initial-boundary multidimensional problem for differential equations, see [15]. We consider the matrix A to be sparse, of high order ($N \approx 10^{10}$ and above) and with a large condition number ($\text{cond}(A) \approx 10^{13}$ and more), when the problem of efficient numerical solution of SLAEs is very resource-intensive and relevant.

For simplicity in describing the general idea of our proposed algorithms, we limit ourselves to problems obtained from the approximation of second-order elliptic-type equations on a cubic grid

$$\Omega^h : x_i = ih, \quad i = 1, \dots, N_x, \quad y_j = jh, \quad j = 1, \dots, N_y, \quad z_k = kh, \quad k = 1, \dots, N_z. \quad (2)$$

More specifically, we consider seven-point SLAEs, which in multi-index notation have the following form:

$$(Au)_{i,j,k} = a_{i,j,k}^{(0)}u_{i,j,k} - a_{i,j,k}^{(1)}u_{i-1,j,k} - a_{i,j,k}^{(2)}u_{i,j-1,k} - a_{i,j,k}^{(3)}u_{i+1,j,k} - a_{i,j,k}^{(4)}u_{i,j+1,k} - a_{i,j,k}^{(5)}u_{i,j,k-1} - a_{i,j,k}^{(6)}u_{i,j,k+1} = f_{i,j,k}. \quad (3)$$

We assume that A is a symmetric positive definite (s.p.d.) matrix, as well as regular (as $h \rightarrow 0$ its elements remain of the same order) and Stieltjes, i.e., the conditions of irreducibility, diagonal dominance, positive diagonal elements, and non-positive off-diagonal elements are satisfied, see more details in [13]. The possibilities of generalizing the classes of SLAEs and the considered methods will be presented below.

In this work, we consider algebraic domain decomposition methods (ADDM) based on separating subdomains using two-layer macro-grids of the following form:

$$\Omega^H : x_{i_1}, x_{i_1+1}, \dots, x_{M_x}, x_{M_x+1}; \quad (4)$$

$$y_{j_1}, y_{j_1+1}, \dots, y_{M_y}, y_{M_y+1}; \quad z_{k_{i_1}}, z_{k_1+1}, \dots, z_{M_z}, z_{M_z+1}.$$

We divide all nodes of the original grid into 4 subsets: $\Omega^h = \Omega_1 \cup \Omega_2 \cup \Omega_3 \cup \Omega_4$, according to their belonging to different topological primitives: macro-nodes Ω_1 , macro-edges Ω_2 , macro-faces Ω_3 , and macro-cells or subdomains Ω_4 . Here $1 < i_1, i_{M_x+1} < N_x, 1 < j_1, j_{M_y+1} < N_y, 1 < k_1, k_{M_z+1} < N_z$ present the numbers of coordinate macrolines and macro-planes which divide the original grid computational domain into $M = M_x M_y M_z$ subdomains. Next, we number the nodes from Ω^h and the corresponding vector components of the original SLAE: first those belonging to the first macro-node, then the second and other macro-nodes, then similarly we number all macro-edge nodes, followed by macro-face nodes and nodes from subdomains. As a result, the vectors u, f from (1) are split into 4 subvectors u_k, f_k which correspond to Ω_k , and the SLAE itself takes the following form:

$$Au = \begin{bmatrix} A_{1,1} & A_{1,2} & 0 & 0 \\ A_{2,1} & A_{2,2} & A_{2,3} & 0 \\ 0 & A_{2,3} & A_{3,3} & A_{3,4} \\ 0 & 0 & A_{4,3} & A_{4,4} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix}. \quad (5)$$

where the matrix A has a block-tridiagonal form of fourth order. A more detailed analysis shows that all diagonal blocks in (5) are block-diagonal matrices, and the block orders of each of them are equal to the number of macro-nodes, macro-edges, macro-faces, and subdomains, respectively. These four types of macrogrid objects present a generalization of the objects of classical grid: node, edge, face and cell. Note that when solving SLAE (3) using a two-layer macro-grid of the form (4), a macro-node consists of eight nodes, a macro-edge has four nodes in its cross-section, and a macro-face represents a fragment of a double grid

plane, which separate two neighbour subdomains. Each macro-edge separates four neighbour macro-faces and each macro-node separates six neighbour macro-edges. For the more detailed description of these topological objects see the recent paper [16].

3 Preconditioned Methods in Krylov Subspaces with Approximate Block Bisection

For solving SLAEs of type (1) with SPD matrices, we consider preconditioned conjugate direction methods based on macrogrid domain decomposition, which defines an algebraic system with a block structure of the form (5). More specifically, if B is some easily invertible (preconditioning) SPD matrix, then with its help, preconditioned iterative processes of Krylov type are formulated in the following general form:

$$r^0 = f - Au^0, \quad p^0 = B^{-1}r^n, \quad n = 0, 1, \dots, \quad (6)$$

$$u^{n+1} = u^n + \alpha_n p^n, \quad r^{n+1} = r^n - \alpha_n A p^n,$$

where p^n are direction vectors, r^0 is the initial residual, and α_n, β_n are iteration parameters to be determined. Under the conditions of A^γ -orthogonality of direction vectors

$$(A^\gamma p^n, p^k) = \rho_k^{(\gamma)} \delta_{n,k}, \quad \rho_k^{(\gamma)} = (A^\gamma p^k, p^k) = \|p^k\|_\gamma^2, \quad (7)$$

where $\gamma = 0, 1, 2$, and $\delta_{n,k}$ is the Kronecker symbol, relations (4) define the residual functional

$$\Phi_\gamma(r^n) = (A^{\gamma-2} r^n, r^n) = (r^0, r^0)_{\gamma-2} - \sum_{k=0}^{n-1} [2\alpha_k (r^0, A^{\gamma-1} p^k) - \alpha_k^2 \rho_k^{(\gamma)}], \quad (8)$$

which, when the relations $\alpha_k = (r^0, A^{\gamma-1} p^k) / \rho_k^{(\gamma)}$ are satisfied, reaches its minimum value

$$\Phi_\gamma(r^n) = \|r^0\|_{\gamma-2}^2 - \sum_{k=0}^{n-1} (r^0, p^k)_{\gamma-1}^2 / \rho_k^{(\gamma)}. \quad (9)$$

In this case, to satisfy the condition $\|r^n\| = (r^n, r^n)^{1/2} \leq \varepsilon \|f\|$ for a given value $\varepsilon \ll 1$, the required number of iterations is estimated by the quantity

$$n(\varepsilon) \leq \sqrt{\kappa} [(\log(2\varepsilon^{-1}))]/2, \quad (10)$$

where κ is the spectral condition number of the matrix $B^{-1}A$. In this family of preconditioned conjugate direction (PCD) methods, the value $\gamma = 0$ corresponds to the minimal error algorithm, while for $\gamma = 1$ and $\gamma = 2$ we have conjugate gradient and conjugate residual algorithms, respectively.

For the most economical of them, the conjugate gradient method, the direction vectors and coefficients are determined by the following formulas:

$$p^{n+1} = B^{-1}r^{n+1} + \beta_n p^n, \quad \alpha_n = \sigma_n / \rho_n, \quad (11)$$

$$\beta_n = \sigma_{n+1} / \sigma_n, \quad \sigma_n = (r^n, B^{-1}r^n), \quad \rho_n = (p^n, A^\gamma).$$

For the case of SLAE (5) of block-fourth order, we consider a block bisection algorithm, which consists of reducing A to block-second order:

$$Au = \begin{bmatrix} A_{a,a} & A_{a,4} \\ A_{4,a} & A_{4,4} \end{bmatrix} \begin{bmatrix} u_a \\ u_4 \end{bmatrix} = \begin{bmatrix} f_a \\ f_4 \end{bmatrix}, \quad u_a = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix}, \quad f_a = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix}, \quad (12)$$

$$A_{a,a} = \begin{bmatrix} A_{1,1} & A_{1,2} & 0 \\ A_{2,1} & A_{2,2} & A_{2,3} \\ 0 & A_{2,3} & A_{3,3} \end{bmatrix}, \quad A_{a,4} = \begin{bmatrix} 0 \\ 0 \\ A_{3,4} \end{bmatrix}, \quad A_{4,a} = [0 \ 0 \ A_{4,3}].$$

Regarding the properties of the matrix block $A_{a,a}$, the following statement can be made.

Theorem 1. If A is a regular Stieltjes matrix, then the spectral condition number of block $A_{a,a}$ is finite, i.e., as $h \rightarrow 0$ $\text{cond}(A_{a,a}) = \max_p \{|\lambda_{a,a}^{(p)}|\} / \min_p \{|\lambda_{a,a}^{(p)}|\} = O(1)$.

The proof of this fact can be easily carried out using estimates of Gershgorin circle radii for the diagonal blocks of the block upper triangular, diagonal, lower triangular factorization of the matrix $A_{a,a}$.

To construct conjugate direction iterative methods of type (6)–(9), we choose as preconditioner B an approximation of the original matrix of the following form:

$$A \approx B = \begin{bmatrix} \bar{A}_{a,a} & A_{a,4} \\ A_{4,a} & A_{4,4} \end{bmatrix}, \quad (13)$$

where $\bar{A}_{a,a}$ denotes some approximation to $A_{a,a}$.

The standard factorization of such a matrix is written in the form

$$B = \begin{bmatrix} \bar{A}_{a,a} & 0 \\ A_{4,a} & S \end{bmatrix} \begin{bmatrix} \bar{A}_{a,a}^{-1} & 0 \\ 0 & S \end{bmatrix} \begin{bmatrix} \bar{A}_{a,a} & A_{a,4} \\ 0 & A_{4,4} \end{bmatrix}, \quad S = A_{4,4} - A_{4,a} \bar{A}_{a,a}^{-1} A_{a,4}, \quad (14)$$

which we transform somewhat for the sake of computational efficiency:

$$B = LDU, \quad L = \begin{bmatrix} \bar{A}_{a,a} & 0 \\ A_{4,a} & I \end{bmatrix}, \quad D = \begin{bmatrix} I & 0 \\ 0 & S \end{bmatrix}, \quad U = \begin{bmatrix} I & \bar{A}_{a,a}^{-1} A_{a,4} \\ 0 & I \end{bmatrix}. \quad (15)$$

At each step of any of the Krylov iterative processes we consider, it is necessary to solve an auxiliary SLAE of the following form:

$$Bq = LDU \begin{bmatrix} q_a \\ q_4 \end{bmatrix} = \begin{bmatrix} r_a \\ r_4 \end{bmatrix} = r. \quad (16)$$

In block-component form, this is accomplished using the relations

$$D\bar{q} = \begin{bmatrix} \bar{q}_a \\ S\bar{q}_4 \end{bmatrix} = \bar{r} = L^{-1}r = \begin{bmatrix} \bar{r}_a = \bar{A}_{a,a}^{-1}r_a \\ \bar{r}_4 = r_4 - A_{4,a}r_a \end{bmatrix}, \quad (17)$$

$$\bar{q} = Uq, \quad q = U^{-1}\bar{q} = \begin{bmatrix} q_a = \bar{q}_a - \bar{A}_{a,a}^{-1}A_{a,4}\bar{q}_4 \\ q_4 = \bar{q}_4 \end{bmatrix}.$$

As follows from this, here it is necessary to solve by some iterative method the SLAE with a dense matrix (Schur complement S):

$$S\bar{q}_4 = (A_{4,4} - A_{4,a}\bar{A}_{a,a}^{-1}A_{a,4})\bar{q}_4 = r_4. \quad (18)$$

At the same time, as follows from (17), it is first necessary to compute the subvector $\bar{A}_{a,a}^{-1}r_a$, and after finding the solution, to restore the desired subvector $q_a = \bar{q}_a - \bar{A}_{a,a}^{-1}A_{a,4}\bar{q}_4$. In fact, these operations do not require finding an approximation of the inverse matrix $\bar{A}_{a,a}^{-1}$, but rather computing with the required accuracy the solutions of the corresponding algebraic subsystems for macrogrid components

$$A_{a,a}v_a = g_a, \quad (19)$$

which, by virtue of Theorem 1, using some preconditioned conjugate direction iterative method (PCD) or Chebyshev acceleration (see [13]), is accomplished in a finite number of steps, asymptotically as $h \rightarrow 0$ independent of the grid size.

For solving SLAE (18), a two-level iterative method has to be applied, where the preconditioner for the Schur complement matrix is easily determined using diagonal compensation:

$$S \approx B_S = A_{4,4} - \theta C = \bar{A}_{4,4}, \quad Ce = A_{4,a}\bar{A}_{a,a}^{-1}A_{a,4}e, \quad (20)$$

where $\theta \in [0, 1]$ is a compensating parameter, $e = \{1\}$, C is a diagonal matrix, which is computed once before iterations, and matrix $\bar{A}_{4,4}$ has the same block-diagonal structure as $A_{4,4}$ (their elements differ only for near-boundary nodes of subdomains). At each iteration, multiplication by matrix S is required, which is accompanied by solving SLAE (19), as well as parallel solution of systems in subdomains, which is formally written as

$$\bar{A}_{4,4}w = g. \quad (21)$$

When solving (21), it seems natural to apply an order-optimal multigrid method (MG) to (21), but options with other preconditioned iterative algorithms are also possible. There are also extreme alternatives with using single-level iterations for (18): either exact solution of (21) using LU decomposition, or, conversely, a sufficiently rough approximation of the matrix $\bar{A}_{4,4}^{-1}$ based on incomplete factorization. In the latter two approaches, one can naturally expect a reduction in the number of iterations with their simultaneous "increase in cost" or vice versa.

Since solving the original SLAE (1) also requires performing outer iterations with the preconditioning matrix B from (13) - (15), in the general case we have a three-level process, and instead of "classical" RCD methods, their

"flexible" variants with long recursions must be applied. For example, instead of the preconditioned conjugate gradient method – FPCG (Flexible Preconditioned Conjugate Gradient) [27].

Regarding outer iterations, the following remark can be made: since the matrix $A_{a,a}$ has finite conditionality, by setting a sufficiently small iteration termination criterion ϵ when solving system (19), we can assume that we obtain an almost exact solution v_a , and limit ourselves to one outer iteration. The dependence of the resulting error on the value of ϵ can be estimated either theoretically or experimentally.

Note that using the properties of the block-tridiagonal structure of matrix $A_{a,a}$ from (12) allows constructing economical iterative and even direct methods based on the algorithm of block elimination for macro-edge variables [21]. However, these promising approaches require special research, and we will not dwell on them.

Regarding the considered two-level conjugate direction methods, when ϵ for SLAE (19) is sufficiently small and up to the used machine arithmetic we can consider $B = A$ in (13), and for $\bar{A}_{4,4}$ LU or ILU decompositions are used, the following statement is valid.

Theorem 2. When solving SLAE (1) with s.p.d. matrix A , the number of iterations of two-level PCD methods with preconditioners of the form (13) – (21), required to satisfy the condition $(r^n, r^n) \leq \epsilon^2(f, f)$, is estimated by the quantity $n(\epsilon)$ from (10), where $\varkappa$ is the condition number of the matrix product $B_S^{-1}S$.

Let us now consider the features of applying the block SSOR method to solving SLAEs of the form (12). In this case, the preconditioning matrix is represented by the formulas (we use subindex "b" instead of "4" in (12)):

$$B_G = \begin{bmatrix} G_a & 0 \\ A_{b,a} & G_b \end{bmatrix} \begin{bmatrix} G_a^{-1} & 0 \\ 0 & G_b^{-1} \end{bmatrix} \begin{bmatrix} G_a & A_{a,b} \\ 0 & G_b \end{bmatrix}, \quad G_a = \omega^{-1}A_{a,a}, \quad G_b = \omega^{-1}A_{b,b}, \quad (22)$$

where $\omega \in (0, 2)$ is a relaxation parameter. In this case, at each iteration, the auxiliary SLAE $B_G q_G = r$ is solved using factorization similar to (15):

$$B_G q_G = L_G D_G U_G q_G = r, \quad L_G = \begin{bmatrix} G_a & 0 \\ A_{b,a} & I \end{bmatrix}, \quad D_G = \begin{bmatrix} I & 0 \\ 0 & G_b \end{bmatrix}, \quad U_G = \begin{bmatrix} I & G_a^{-1}A_{a,b} \\ 0 & I \end{bmatrix}. \quad (23)$$

From this we obtain the relations

$$D_G \bar{q}_G = \bar{r}, \quad \bar{r} = L_G^{-1}r, \quad \bar{q} = U_G q_G, \quad q_G = U_G^{-1} \bar{q}_G, \quad (24)$$

which assume before each iteration and after its completion the solution of SLAE with matrix G_a , by analogy with (23). Formulas (22)-(24) have an advantage over (17), (18), that they do not require double iterations for solving the system with block-diagonal matrix D_G . From the theory of SSOR method

[13], it follows that Theorem 2 is valid in this case also, but we need change the preconditioner B_S of the form (20) to the preconditioner B_G of the form (22)-(24).

However, since this requires iterative computation of solutions in subblocks, we actually, when using preconditioned conjugate direction methods (*PCG* for example), have a variable matrix B_G . Strictly speaking, in this case, outer iterations should apply "flexible", or dynamic, preconditioning of type *FPCG* from [27], with long recursions. One can assume, however, that if sufficiently small parameter ϵ is used on inner iterations, then for outer iterations the "classical" *PCG* will provide good convergence.

4 Discussion of Numerical Experiments

In this section, we consider the results of preliminary studies on solving methodological three-dimensional grid boundary value problems with the aim of analyzing such properties of the proposed algorithms as iteration convergence rate, computational performance during parallelization, and stability. As test examples, solutions of seven-point grid equations on a cubic grid in a cubic computational domain with Dirichlet conditions were taken. As a model solution and initial guess, the simplest functions $u = 1$ and $u = 0$ were used, respectively (with other functions, the calculation results change insignificantly).

The software implementation of the developed algorithms was performed on the INMOST [22] platform using HYPRE [12] and PETSc [29] libraries. Calculations were performed on the a100serv computing node of the Gazpromneft-NSU Research and Education Center - 72 cores Intel Xeon Platinum 8354H 3.1 GHz, 3 TB RAM, 10 Gb/s Ethernet network.

Experiments were conducted on grids with the number of nodes $N = 32^3, 64^3, 128^3, 256^3$ and with macro-grids of size $M = 2^3, 4^3$, while algorithm parallelization was performed on $P = M$ processors. Iterations in all variants were executed with their stopping criterion $\epsilon = 10^{-7}$. The solution of auxiliary SLAEs with matrix $A_{a,a}$ was implemented in two variants: by conjugate gradient methods (CG) and Chebyshev acceleration (CH) with experimentally selected approximate spectrum boundaries $\lambda_{min} = 1$, $h_{max} = \|A_{a,a}\|_{\infty}$. The calculation results according to our proposed algorithms are compared with tools from PETSc and HYPRE libraries implementing a parallel order-optimal algebraic multigrid method.

Table 1 presents numerical results for solving the original SLAEs (1) by the BoomerAMG algorithm. Hypre BoomerAMG was used with the following parameters.

```
-pc_hypre_boomeramg_agg_n1 1
-pc_hypre_boomeramg_strong_threshold 0.7
-pc_hypre_boomeramg_coarsen_type HMIS
-pc_hypre_boomeramg_interp_type ext+i
-pc_hypre_boomeramg_truncfactor 0.2
```

Table 1. Calculation results for different numbers of grid points N , and number of MPI processes P ; BoomerAMG.

N	32 ³		64 ³		128 ³		256 ³	
P	8	64	8	64	8	64	8	64
n	11	12	12	13	14	15	16	17
$\Delta_n (\times 10^{-7})$	3.18	3.17	6.14	6.28	4.29	6.27	9.93	8.08
t	0.0759	0.402	0.259	0.672	2.18	1.35	24	5.6
$\delta_n (\times 10^{-8})$	6.46	6.96	8.52	10.5	3.54	6.6	5.54	6.22

Table contains six rows: size of grid, the number of MPI processes, the number of iterations n and total computation time t in seconds, the absolute error of the iterative solution $\delta_n = \|u - u^n\|_\infty$ and the corresponding relative residual $\Delta_n = \frac{\|f - Au^n\|_2}{\|f\|_2}$. Table 2 contains the results of an experiment with the application of the above-described preconditioned conjugate gradient methods DDM, see (6)-(21), when using the multigrid method BoomerAMG in subdomains (this means solving SLAE (21) with matrix $\bar{A}_{4,4}$, which is a preconditioner for the Schur complement from (19)-(20), and the conjugate gradient method without preconditioning for solving the algebraic subsystem (19) on the macrogrid. Rows of this table contain seven numbers: the numbers of iterations n_1 for macrogrid SLAE (19) solver and n_2 for outer SLAE (18) solver, as well as the absolute error δ_n and relative residual Δ_n , the times t_1 , t_2 for solving systems (19),(18) respectively and the total calculation time t of the variant.

Table 2. Computation time for different numbers of grid points N , number of MPI processes P .

N	32 ³		64 ³		128 ³		256 ³	
P	8	64	8	64	8	64	8	64
n_1	3	3	3	3	3	3	3	3
t_1	0.045	0.832	0.085	0.231	0.291	0.339	2.122	1.016
n_2	18	33	28	33	38	50	56	74
t_2	0.231	1.904	0.730	1.904	4.958	4.940	59.320	27.915
$\delta_n (\times 10^{-6})$	0.518	1.34	1.09	1.42	2.60	3.85	6.61	9.02
t	0.290	2.320	0.838	2.196	5.323	5.372	61.866	29.185
$\Delta_n (\times 10^{-7})$	1.10	2.06	1.11	1.82	1.50	1.86	1.45	1.75

Table 3 presents results similar to Table 2. The only difference is that for solving the macrogrid SLAE (19), the more economical Chebyshev acceleration method is used instead of CG (without preconditioning).

Analysis of the presented data shows, firstly, that the Chebyshev acceleration method outperforms the CG algorithm in terms of performance, although it

Table 3. Calculation results using the DDM algorithm with Chebyshev acceleration for solving the macrogrid SLAE (19).

N	32 ³		64 ³		128 ³		256 ³	
P	8	64	8	64	8	64	8	64
n_1	34	47	31	35	30	34	29	31
t_1	0.005	0.018	0.006	0.017	0.012	0.054	0.059	0.067
n_2	18	33	28	33	38	50	56	74
t_2	0.111	0.698	0.411	0.698	3.830	2.155	52.340	21.800
$\delta_n (\times 10^{-5})$	0.553	0.695	0.904	1.19	1.73	2.20	3.25	3.35
t	0.124	0.474	0.429	0.750	3.887	2.246	52.736	22.060
$\Delta_n (\times 10^{-7})$	1.56	2.07	1.57	2.59	2.05	2.50	1.90	2.70

is inferior in the number of iterations, which was to be expected. Secondly, both considered DDM implementations are characterized by a slight slowdown of the overall iterative process both when decreasing the grid step h and when increasing the number of macrogrid edges, as a result of which they are about one and a half times inferior to the multigrid BoomerAMG method, where the number of iterations practically does not increase with the growing number of grid nodes. In all experiments, the iterative methods provide good accuracy and stability, in terms of the values of absolute error δ_n and relative residual Δ_n for the different number of unknowns.

In principle, decomposition approaches are more promising for parallelizing very large problems using hybrid programming tools, when solving auxiliary SLAEs for different corresponding subdomains is performed as synchronously as possible using MPI on multiprocessor computational nodes with distributed memory, while "internal" parallelization of each subsystem is implemented using multithreaded technologies on shared hierarchical memory with capabilities for vectorizing AVX procedures, see [17].

As for the rather complex algebraic data structures and vector-matrix multilevel iterative schemes in the proposed macrogrid approaches with block bisection, there are still significant reserves for improving both mathematical efficiency and parallel computing performance. Among the possible acceleration techniques are the following:

- When solving SLAE (18) using two-level iterations, it is necessary to repeatedly solve algebraic systems with the same matrix A but with sequentially determined different right-hand sides. In this case, the process can be significantly accelerated by applying deflation algorithms based on expanding the Krylov basis taking into account previous external iterations [9].
- We solve algebraic systems of the form (19) using the simplest iterative methods without preconditioning. However, the special block structure of the matrix $A_{a,a}$ allows for effective use of fine-block variants of sweeping

methods [18] and other approaches that significantly improve computational performance.

- When constructing a preconditioner for the Schur matrix S in (20), we use traditional diagonal compensation. Further improvement of the method can be achieved by computing an approximate sparse inverse matrix (SPAI) [4] or using the generalized principle of row sum criteria [22] for the inverse matrix $A_{a,a}^{-1}$.

The considered macrogrid domain decomposition methods can be naturally applied to other types of grids, including for solving more general types of boundary value problems, for example, for the Stokes system of equations.

5 Conclusion

Two-level macrogrid domain decomposition methods for solving multidimensional grid boundary value problems have been developed and experimentally investigated. A comparison of the performance of parallel algorithms with the multigrid approach of the BoomerAMG program has been made, and issues of improving the efficiency of the proposed methods have been considered. Also, we have plan to increase the performance of our code by using parallelization on the distributed and shared memory, as well as to generalize the algorithms for solving three-dimensional Stokes problem.

Acknowledgments. The paper was financially supported by the Russian Science Foundation (Project No. 24-21-00402).

References

1. Agoshkov, V.I., Lebedev, V.I.: Variational algorithms of the domain decomposition method. *Soviet J. Num. Anal. Math. Model.* **5**(1), 27–46 (1990). <https://doi.org/10.1515/rnam.1990.5.1.27>
2. Bank, R., Falgout, R., Jones, T., Manteuffel, T.A., McCormick, S.F., Ruge, J.W.: Algebraic multigrid domain and range decomposition (amg-dd/amg-rd). *SIAM J. Sci. Comput.*, **37**(5), 113–136 (2015). <https://doi.org/10.1137/140974717>
3. Maxim, B., Il’In, V.: Multigrid incomplete factorization methods in krylov subspaces on unstructured grids. In: Sokolinsky, L., Zymbler, M., Voevodin, V., Dongarra, J., eds, *Parallel Computational Technologies*, pp. 163–176, Cham. Springer Nature Switzerland (2024). ISBN 978-3-031-73372-7
4. Benzi, M., Meyer, C.D., Tũma, M.: A sparse approximate inverse preconditioner for the conjugate gradient method. *SIAM J. Sci. Comput.* **17**(5), 1135–1149 (1996). <https://doi.org/10.1137/S1064827594271421>. The special character in Tũma is correctly represented using ũ for LaTeX
5. D’yakovon, E.G.: On some direct and iterative methods based on matrix bordering. In: *Numerical Methods in Mathematical Physics*, vol. 4 of *Proceedings of the Seminar “Methods of Computational and Applied Mathematics”*, pp. 45–68 (1979)

6. Dolean, V., Jolivet, P., Nataf, F.: An Introduction to domain decomposition methods: algorithms, theory and parallel implementation. Lecture (2015). <https://hal.science/cel-01100932>
7. Gander, M.J., Magoulès, F., Nataf, F.: Optimized schwarz methods without overlap for the helmholtz equation. *SIAM J. Sci. Comput.* **24**(1), 38–60 (2002). <https://doi.org/10.1137/S1064827501387012>
8. Gurieva, Y., Il'in, V., Kardash, R.: Incomplete factorization approach in algebraic domain decomposition methods. In: Voevodin, V., Antonov, A., Nikitenko, D., eds., *Supercomputing*, volume 15406 of *Lecture Notes in Computer Science*, page TBD (2025)
9. Gurieva, Y.L., Il'in, V.P.: Conjugate direction methods for multiple solution of slaes. *J. Math. Sci.* **255**(3), 231–241 (2021.) <https://doi.org/10.1007/s10958-021-05365-8>, <https://link.springer.com/content/pdf/10.1007/s10958-021-05365-8.pdf>. Translated from Russian, ISSN 1072-3374.
10. Gurieva, Y.L., Il'in, V.P., Kozlov, D.I.: Parallel domain decomposition methods with graph preconditioning. In: *Parallel Computing Technologies (PAVT'2023)*, XVII All-Russian Scientific Conference with International Participation, pp. 215–228, Chelyabinsk (2023)
11. Henson, V.E., Yang, U.M.: Boomeramg: a parallel algebraic multigrid solver and preconditioner. *Appl. Num. Math.* **41**(1), 155–177 (2002). ISSN 0168-9274. [https://doi.org/10.1016/S0168-9274\(01\)00115-5](https://doi.org/10.1016/S0168-9274(01)00115-5), <https://www.sciencedirect.com/science/article/pii/S0168927401001155>. Developments and Trends in Iterative Methods for Large Systems of Equations - in memorium Rudiger Weiss
12. HYPRE: Scalable Linear Solvers and Multigrid Methods. <https://computing.llnl.gov/projects/hypre-scalable-linear-solvers-multigrid-methods/>
13. Il'in, V.P.: Finite element methods and techniques. ICM&MG SB RAS, Novosibirsk (2007)
14. Il'in, V.P.: Iterative preconditioned methods in krylov spaces: trends of the 21st century. *Comput. Math. Math. Phys.* **61**(11) 1750–1775, 11 (2021). ISSN 1555-6666 <https://doi.org/10.1134/S0965542521110099>. URL <https://doi.org/10.1134/S0965542521110099>
15. Il'in, V.P.: Mathematical modeling part 1. continuous and discrete models. Publisher of the Siberian Branch of the Russian Academy of Sciences, Novosibirsk (2017). ISBN 978-5-7692-1553-7
16. Il'in, V.P.: Multigrid methods of macrogrid domain decomposition. *Comput. Math. Math. Phys.* **65**(6), 1218–1229 (2025). ISSN 1555-6662
17. Il'in, V.P.: Parallel intelligent computing in algebraic problems. In: Sokolinsky, L., Zymbler, M., eds., *Parallel Computational Technologies*, pp. 108–117, Cham. Springer International Publishing (2021). ISBN 978-3-030-81691-9
18. Il'in, V.P., Kuznetsov, Y.I.: Tridiagonal Matrices and Their Applications. Library of Computational Mathematics. Nauka, Moscow (1985). In Russian
19. Il'in, V.P., Yu. Laevsky, K.: Generalized compensation principle in incomplete factorization methods. *Russ. J. Numerical Anal. Math. Model.* **12**(5), 399–420 (1997). https://icmmg.nsc.ru/sites/default/files/pubs/ilin_laevsky_1997_rjnamm.pdf, ISSN 0927-6467
20. Il'in, V.P.: Multi-preconditioned domain decomposition methods in the krylov subspaces. In: *Lecture Notes in Computer Science*, vol. 10187, pp. 95–106. Springer (2017). https://doi.org/10.1007/978-3-319-58629-1_8
21. Tregub, I.V.: On the applicability of the random walk model with stable steps for forecasting the dynamics of prices of financial tools in the russian market. *J. Math. Sci.* **216**(5), 716–721 (2016). <https://doi.org/10.1007/s10958-016-2933-8>

22. INMOST: A Toolkit for Distributed Mathematical Modeling. <https://www.inmost.org/>
23. Korneev, V., Langer, U.: Domain decomposition methods and preconditioning. In: Encyclopedia of Computational Mechanics, pp. 617–647. John Wiley & Sons (2004). <https://doi.org/10.1002/0470091355.ecm019>, ISBN 978-0-470-84699-5.
24. Kuznetsov, Y.A.: Algebraic multigrid domain decomposition methods. Preprint 232, Academy of Sciences of the USSR, Department of Computational Mathematics, Moscow (1989)
25. Laevsky, Y.M., Matsokin, A.M.: Decomposition methods for solving elliptic and parabolic boundary value problems. Siberian J. Comput. Math., **2**, 361–372 (1999). ISSN 1029-3041
26. Nepomnyashchikh, S.V.: Decomposition and fictitious domains methods for elliptic boundary value problems. In: Fifth International Symposium on Domain Decomposition Methods for Partial Differential Equations, pp. 62–72, Philadelphia. SIAM (1992)
27. Notay, Y.: Flexible conjugate gradients. SIAM J. Sci. Comput. **22**(4), 1444–1460 (2000). <https://doi.org/10.1137/S1064827599362314>
28. Olshansky, M.A., Tyrtshnikov, E.E.: Iterative Methods for Linear Systems Theory and Applications. SIAM (2014)
29. PETSc: Portable Extensible Toolkit for Scientific Computation. <https://petsc.org/release/>
30. Quarteroni, A., Valli, A.: Domain decomposition methods for partial differential equations. Clarendon Press, Oxford (1999).
31. Saad, Y.: Iterative methods for sparse linear systems. Soc. Industr. Appl. Math. (2003). <https://doi.org/10.1137/1.9780898718003>, <https://epubs.siam.org/doi/abs/10.1137/1.9780898718003>
32. Smith, B.F., Bjørstad, P.E., Gropp, W.D.: Domain decomposition: parallel multi-level methods for elliptic partial differential equations. Cambridge University Press (1996). <https://doi.org/10.1017/CBO9780511573916>
33. Bern, M., Gilbert, J.R., Hendrickson, B., Nguyen, N., Toledo, S.: Support graph preconditioners. SIAM J. Matrix Anal. Appl. **27**(4), 930–951 (2006). <https://doi.org/10.1137/S0895479801384019>
34. Domain Decomposition Methods — Algorithms and Theory. SSCM, vol. 34. Springer, Heidelberg (2005). <https://doi.org/10.1007/b137868>
35. Vassilevski, Y.V., Olshansky, M.A.: Quick course on multigrid methods and domain decomposition methods: textbook. MAKSPress (2007) (in Russian)