Chapter 4

Numerical methods for hybridized problem

In this chapter we consider a new approach to solving saddle point problem (3.1) proposed recently by Yu.Kuznetsov [46]. Applying an algebraic hybridization to the diffusion block, we reformulate the coupled problem in terms of displacement u and Lagrange multiplier λ (we no longer use the Lamé coefficients). The resulting problem is solved by an iterative method with a block diagonal preconditioner with as many blocks as there are physical variables. The elastic block is preconditioned by the matrix B_u described in Section 3.2. Therefore, in this chapter we focus on preconditioners for a Lagrange multiplier block. In Section 4.2 we formulate a new separable preconditioner for this block. The preconditioner can be applied to the case of anisotropic meshes and anisotropic diffusion tensors. However, its construction is restricted to topologically cubic meshes. In Section 4.3 we describe a multilevel preconditioner which works for quite general meshes but is sensitive to anisotropy of both the mesh and diffusion tensor.

To test the preconditioners numerically, we use a prismatic partition $\mathcal{T}_h$ based on a cubic mesh with mesh-step size h. The partition $\mathcal{T}_h$ is obtained by splitting each cubic cell into 2 prisms as shown on Fig. 3.1. Topologically prismatic meshes are very important for reservoir simulations. They are not so complicated as unstructured meshes and are more suitable for complex-shaped domains than topologically cubic meshes.

It is pertinent to note that the preconditioners for the Lagrange multiplier block can be used beyond the framework of block diagonal preconditioners for the coupled problem.

4.1 Hybridization procedure

Let us apply a hybridization procedure to the coupled problem (3.1). Without loss of generality, we assume that the stiffness matrix $\mathcal{A}$ can be written in the form

$$\mathcal{A} = \sum_{i=1}^{m} \mathcal{N}_i \mathcal{A}_i \mathcal{N}_i^T$$

where $\mathcal{N}_i$ is an assembling matrix, and $\mathcal{A}_i$ is an elemental matrix given by

$$\mathcal{A}_i = \begin{pmatrix} A_{i,\,uu} & 0 & A_{i,\,up}^T \\ 0 & A_{i,\,ff} & A_{i,\,pf}^T \\ A_{i,\,up} & A_{i,\,pf} & -c_{pp} A_{i,\,pp} \end{pmatrix}. \tag{4.1}$$

Let us consider the mesh element $e_i \in \mathcal{T}_h$ and define a new flux vector $\hat{f}_i = (f_{i,j_1}, \ldots, f_{i,j_{s_i}})^T$ where s_i is the number of faces of e_i except faces on $\partial\Omega \setminus \Gamma$ where the flux is zero, and j_k is the number of either a neighboring mesh element having a common face γ_{i,j_k} with e_i or a boundary mesh face. Let u_i and p_i be restrictions of displacement u and pressure p, respectively, to the mesh element e_i (for the case of tetrahedral meshes, $u_i \in \Re^{12}$ and $p_i \in \Re^1$). Furthermore, let $b_p = (b_1, \ldots, b_m)^T$. Obviously, problem (3.1) is equivalent to the following constraint problem: Find $u \in \Re^{n_u}$ and pairs $(\hat{f}_i, p_i)$, $i = 1, \ldots, m$, such that:

$$\begin{pmatrix} A_{i,uu} & 0 & A_{i,up}^T \\ 0 & A_{i,ff} & A_{i,pf}^T \\ A_{i,up} & A_{i,pf} & -c_{pp}A_{i,pp} \end{pmatrix} \begin{pmatrix} u_i \\ \hat{f}_i \\ p_i \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ b_i \end{pmatrix} \tag{4.2}$$

subject to constraints

$$|\gamma_{ij}| f_{i,j} = |\gamma_{ij}| f_{j,i} \qquad \forall \gamma_{ij} \notin \partial\Omega. \tag{4.3}$$

The constrained problem (4.2), (4.3) can be written in an algebraic form:

$$\begin{pmatrix} A_{uu} & 0 & A_{pu}^T & 0 \\ 0 & \hat{A}_{ff} & \hat{A}_{pf}^T & C_\lambda^T \\ A_{pu} & \hat{A}_{pf} & -c_{pp}A_{pp} & 0 \\ 0 & C_\lambda & 0 & 0 \end{pmatrix} \begin{pmatrix} u \\ \hat{f} \\ p \\ \lambda \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ b_p \\ 0 \end{pmatrix},$$

where λ is a vector of Lagrange multipliers, $\lambda \in \Re^{n_\lambda}$, and entries of matrix C_λ are areas $|\gamma_{ij}|$ of mesh faces. One of the advantages of the hybridization procedure is that

$$\hat{A}_{f+p} = \begin{pmatrix} \hat{A}_{ff} & \hat{A}_{pf}^T \\ \hat{A}_{pf} & -c_{pp}A_{pp} \end{pmatrix}$$

is a block diagonal matrix with as many blocks as there are mesh elements. Therefore, unknowns $\hat{f}$ and p may be explicitly eliminated resulting in a reduced problem:

$$\hat{\mathcal{A}} \begin{pmatrix} u \\ \lambda \end{pmatrix} = \begin{pmatrix} \hat{A}_{uu} & A_{\lambda u}^T \\ A_{\lambda u} & -A_{\lambda\lambda} \end{pmatrix} \begin{pmatrix} u \\ \lambda \end{pmatrix} = \begin{pmatrix} b_u \\ b_\lambda \end{pmatrix}, \tag{4.4}$$

where

$$\begin{aligned} A_{\lambda\lambda} &= C_\lambda \hat{A}_{ff}^{-1} C_\lambda^T - C_\lambda \hat{A}_{ff}^{-1} \hat{A}_{pf}^T S_{pp}^{-1} \hat{A}_{pf} \hat{A}_{ff}^{-1} C_\lambda^T, \\ \hat{A}_{uu} &= A_{uu} + A_{pu}^T S_{pp}^{-1} A_{pu}, \qquad S_{pp} = c_{pp}A_{pp} + \hat{A}_{pf}\hat{A}_{ff}^{-1}\hat{A}_{pf}^T, \\ A_{\lambda u} &= C_\lambda \hat{A}_{ff}^{-1} \hat{A}_{pf}^T S_{pp}^{-1} A_{pu}, \end{aligned}$$

and

$$(b_u,\ b_\lambda)^T = \hat{A}_{f+p}^{-1}\ (0,\ b_p)^T.$$

We shall search for a preconditioner $\hat{\mathcal{B}}$ in a class of block diagonal positive definite matrices:

$$\hat{\mathcal{B}} = \begin{pmatrix} B_u & 0 \\ 0 & B_\lambda \end{pmatrix}. \tag{4.5}$$

Theorem 4.1 *Let B_u be the matrix described in Section 3.2 and B_λ be a matrix spectrally equivalent to $A_{\lambda\lambda}$ ($B_\lambda \sim A_{\lambda\lambda}$ shortly). Furthermore, let $\mathcal{T}_h$ be a quasi-uniform partition. Then $\hat{\mathcal{B}} \sim \hat{\mathcal{A}}$.*

Proof. We begin with proving the spectral equivalence of matrices B_u and $\hat{A}_{uu}$. The block diagonal structure of matrix $\hat{A}_{ff}$ implies that S_{pp} is a diagonal positive definite matrix. Let h be a discretization parameter of the quasi-uniform partition $\mathcal{T}_h$. Applying the lumping procedure to the mass matrix $\hat{A}_{ff}$, we can easily show that $S_{pp} \sim hI_{n_p}$. Recall that I_s denotes the identity matrix of size s. Since A_{pu} is a sparse matrix with non-zero entries of order h^2, the Gershgorin theorem implies:

$$(A_{pu}^T S_{pp}^{-1} A_{pu} v,\ v) \le ch^3(v,\ v) \qquad \forall v \in \Re^{n_u}.$$

The Friedrich inequality for $\boldsymbol{v}^h \in V_h$ and (3.8) imply that

$$h^3(v,\ v) \le c\|\boldsymbol{v}^h\|^2 \le c\|\nabla \boldsymbol{v}^h\|^2 = c(B_u v,\ v) \le cc_1^{-1}(A_{uu} v,\ v)$$

where v is a discrete counterpart of $\boldsymbol{v}^h$. Combining the last inequality with (3.8), we get

$$c_1(B_u v,\ v) \le (\hat{A}_{uu} v,\ v) \le c(B_u v,\ v) \qquad \forall v \in \Re^{n_u}. \tag{4.6}$$

In addition, matrices $\hat{A}_{uu}$ and A_{uu} are spectrally equivalent.

Note that the formulae for computing $A_{\lambda\lambda}$ do not involve matrices associated with the displacement vector u. Therefore, the theory of mixed-hybrid finite element discretizations [3, 17] may be applied here. It states that $A_{\lambda\lambda}$ is a positive definite matrix. More precisely, the matrix $A_{\lambda\lambda}$ comes from a mixed-hybrid approximation of an elliptic equation

$$-\mathrm{div}(\boldsymbol{K}\nabla p) + c_{pp}p = S_f \tag{4.7}$$

subject to homogeneous boundary conditions $p = 0$ on Γ and $\boldsymbol{f} \cdot \boldsymbol{n} = 0$ on $\partial\Omega \setminus \Gamma$.

We proceed by proving that matrices $A_{\lambda\lambda}$ and $S_\lambda \equiv A_{\lambda\lambda} + A_{\lambda u}\hat{A}_{uu}^{-1}A_{\lambda u}^T$ are spectrally equivalent. Therefore, it is sufficient to evaluate the contribution of the second term. Lemma 3.2 and $\hat{A}_{uu} \sim A_{uu}$ imply that $A_{pp} + A_{pu}^{-1}\hat{A}_{uu}^{-1}A_{pu}^T \sim A_{pp}$. Since entries of C_λ are of order h^2, $A_{pp} \sim h^3 I_{n_p}$, $S_{pp} \sim hI_{n_p}$, and $\hat{A}_{ff}^{-1}$ is a sparse matrix, the following chain of inequalities holds:

$$\begin{aligned}
(A_{\lambda u}\hat{A}_{uu}^{-1}A_{\lambda u}^T\lambda,\, \lambda) &\leq c\,(C_\lambda \hat{A}_{ff}^{-1}\hat{A}_{pf}^T S_{pp}^{-1} A_{pp} S_{pp}^{-1}\hat{A}_{pf}\hat{A}_{ff}^{-1}C_\lambda^T\,\lambda,\, \lambda) \\
&\leq ch\,(C_\lambda \hat{A}_{ff}^{-1}\hat{A}_{pf}^T\hat{A}_{pf}\hat{A}_{ff}^{-1}C_\lambda^T\,\lambda,\, \lambda) \\
&\leq ch^3(\lambda,\, \lambda).
\end{aligned}$$

The discrete Friedrich inequality in the space of Lagrange multipliers gives $h^3(\lambda,\, \lambda) \leq c(A_{\lambda\lambda}\lambda, \lambda)$. Summarizing the results above, we get

$$(A_{\lambda\lambda}\lambda,\, \lambda) \leq (S_{\lambda\lambda}\lambda,\, \lambda) \leq c(A_{\lambda\lambda}\lambda,\, \lambda) \qquad \forall\lambda \in \Re^{n_\lambda}. \tag{4.8}$$

The assertion of the theorem follows from (4.6), (4.8) and Lemma 3.1. □

It is pertinent to note that after solving (4.4), we may easily recover the flux $\hat{f}$ and pressure p by solving a problem with the block diagonal matrix $\hat{A}_{f+p}$.

Numerical results in Chapter 3 have shown that the convergence of Lanczos iterations is driven by preconditioning properties of matrix B_p. In other words, a good preconditioner for the diffusion block results in a good block diagonal preconditioner for the coupled problem. Theorem 4.1 allows us to extrapolate this conclusion to the hybridized problem (4.4). Therefore, the last two sections of this chapter are devoted to the study of preconditioners for matrix $A_{\lambda\lambda}$.

4.2 Separable preconditioner for $A_{\lambda\lambda}$

Frequently, in numerical experiments we have to solve systems of algebraic equations with a *separable* matrix $L \in \Re^{\tilde{n}\times\tilde{n}}$:

$$L = L_1 \otimes M_2 \otimes M_3 + M_1 \otimes L_2 \otimes M_3 + M_1 \otimes M_2 \otimes L_3 \tag{4.9}$$

where symbol $\otimes$ stands for the Kronecker (tensor) product of two matrices and L_i, $M_i \in \Re^{\tilde{n}_i\times\tilde{n}_i}$, $i = 1, 2, 3$. The separable matrices are usually related to regular geometric structures such as cubic meshes. The classical example here is a finite difference approximation of the Dirichlet boundary value problem for the Poisson equation on a cubic mesh. In this example L_i and M_i are symmetric positive definite tridiagonal and diagonal matrices, respectively.

In a more general case, we assume that L_i, $i = 2, 3$, are symmetric matrices, and M_i, $i = 2, 3$, are symmetric positive definite matrices. Then an algebraic problem with matrix L can be solved by a method of separation of variables. If L_i and M_i, $i = 1, 2, 3$, are band matrices with the band width k, fast solution algorithms based on separation of variables require $C(k)\,\tilde{n}\log^2\tilde{n}$ arithmetical operations where $C(k)$ is a positive constant independent of $\tilde{n}$ [50].

In this section we extend the concept of separable matrices to a class of matrices appearing either from a mixed-hybrid finite element discretization of an elliptic equation on cubic meshes [18, 62] or from an equivalent algebraic hybridization described in the previous section. Since the solution entries are associated with faces of an orthogonal mesh, the matrix $A_{\lambda\lambda}$ can not be written in a tensor form (4.9). In fact, the sparsity structure of the $A_{\lambda\lambda}$ is given by an 11-point stencil describing connections between faces belonging to the same mesh cell. A similar sparsity structure appears in finite difference methods such as nodal integration methods in reactor physics (see, e.g., [60] and references therein) and mimetic discretizations [68] (after a hybridization procedure).

We shall show that the solution of $A_{\lambda\lambda}\lambda = b$ requires $O(n_\lambda \log^2 n_\lambda)$ arithmetical operations. Thus, the method has the same quasi-optimal complexity [20, 41, 74] as the well known fast direct methods for the matrix L.

In the case of a general topologically cubic mesh, the direct solution of the associated algebraic problem is impossible. However, the fast direct method can be used as a preconditioner.

In order to simplify notations, we shall write A instead of $A_{\lambda\lambda}$.

4.2.1 Fast direct method for 2D problems

We begin with analysis of a 2D problem (4.7) associated with a quadrilateral mesh. For the sake of simplicity we assume that $\boldsymbol{K} = \boldsymbol{I}$, $c_{pp} = 0$, $\Gamma = \partial\Omega$, and computational mesh is rectangular with square cells. A few important generalizations of the method to more complicated problems and meshes will be discussed later. The sparsity structure of matrix A (for a model problem) is given by the 7-point stencil shown in Fig. 4.1. The stencil is modified for near-boundary edges by removing connections to boundary edges.

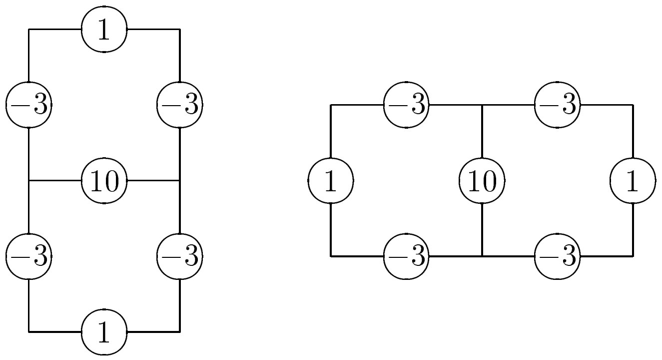

Figure 4.1: A stencil for a 2D model problem.

The rectangular mesh $\mathcal{T}_h$ is uniquely determined by a set of mesh points. Let us define one-dimensional meshes:

$$\begin{aligned} \mathcal{T}_x &= \{[x_i,\, x_{i+1}]:\ x_1 < \ldots < x_{n_x+1}\}, \\ \mathcal{T}_y &= \{[y_j,\, y_{j+1}]:\ y_1 < \ldots < y_{n_y+1}\}. \end{aligned} \tag{4.10}$$

Then

$$\mathcal{T}_h = \{e_{ij}:\ 1 \le i \le n_x,\ 1 \le j \le n_y\}$$

where e_{ij} is a rectangle with vertices $(x_i,\, y_j)$, $(x_{i+1},\, y_j)$, $(x_i,\, y_{j+1})$ and $(x_{i+1},\, y_{j+1})$. Let $h_{xi} = x_{i+1} - x_i$, $i = 1, \ldots, n_x$, and $h_{yj} = y_{j+1} - y_j$, $j = 1, \ldots, n_y$, be mesh steps in x and y directions, respectively. The one-dimensional mesh is called *uniform* if the mesh steps are equal. It is called *quasi-uniform* if the ratio of the maximal mesh step to the minimal one is bounded by a constant independent of the mesh steps.

Without loss of generality we assume that $x_i = i$ and $y_j = j$. Let $\boldsymbol{x}_{i,\,j+\frac{1}{2}} = (i,\, j+\frac{1}{2})$ and $\boldsymbol{x}_{i+\frac{1}{2},\,j} = (i+\frac{1}{2},\, j)$ be the middle edge points associated with entries of the solution vector λ. We shall call them the "solution points". Straightforward calculations show that the only non-zero entries of matrix A are 10α, -3α and α, where $\alpha = 1/2$ (see Fig. 4.1).

Let us divide components of vector λ into $k_x = 2n_x - 1$ separate groups as follows. Each group consists of components corresponding to mesh points with the same x-coordinate. The partition is illustrated in Fig. 4.2, where different groups are marked by different symbols. Recall that the Dirichlet boundary conditions imply that the solution entries are associated only with the inner mesh edges.

Let I be the generic notation for the identity matrix. Hereafter, we propose the new direct method for solving

$$A\lambda = b, \qquad b \in \Re^n. \tag{4.11}$$

Based on this method, we formulate a fast direct algorithm with the quasi-optimal arithmetical complexity. Let us rewrite system (4.11) in a block form associated with

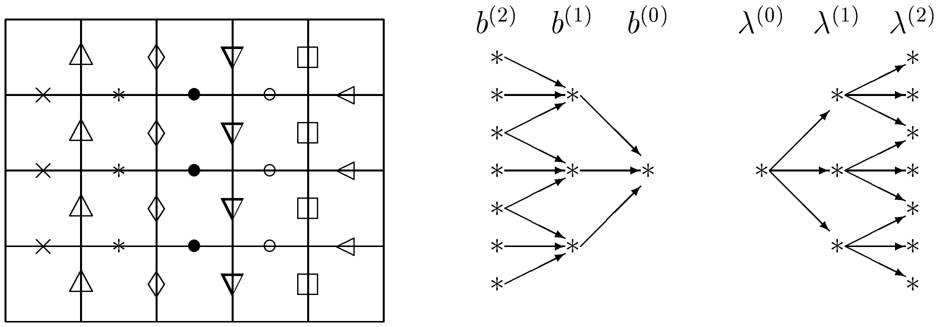

Figure 4.2: Partition of the solution components and the order of computations.

the partition above of the solution components:

$$
\begin{pmatrix}
A_1 & C_1 & & & & & & \\
C_1^T & D_2 & C_2^T & E_3 & & & & \\
& C_2 & A_3 & C_3 & & & & \\
& E_3^T & C_3^T & D_4 & C_4^T & E_5 & & \\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
& & & E_{k_x-2}^T & C_{k_x-2}^T & D_{k_x-1} & C_{k_x-1}^T & \\
& & & & & C_{k_x-1} & A_{k_x} &
\end{pmatrix}
\begin{pmatrix}
\lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_4 \\ \cdots \\ \lambda_{k_x-1} \\ \lambda_{k_x}
\end{pmatrix}
=
\begin{pmatrix}
b_1 \\ b_2 \\ b_3 \\ b_4 \\ \cdots \\ b_{k_x-1} \\ b_{k_x}
\end{pmatrix}
\tag{4.12}
$$

where $E_i = E_y = \alpha I$, $D_i = D_y = 10\alpha I$, $C_i = C_y \in \Re^{(n_y-1)\times n_y}$ ia a two-diagonal matrix, $A_i = A_y \in \Re^{(n_y-1)\times(n_y-1)}$ is a tridiagonal symmetric matrix, $1 \le i \le k_x$,

$$
C_y = \alpha \begin{pmatrix}
-3 & -3 & & \\
& \ddots & \ddots & \\
& & -3 & -3
\end{pmatrix}, \quad \text{and} \quad A_y = \alpha \begin{pmatrix}
10 & 1 & & \\
1 & 10 & \ddots & \\
& \ddots & \ddots & 1 \\
& & 1 & 10
\end{pmatrix}.
$$

A more general case of system (4.12) will be studied later. Obviously,

$$
A_y = 8\alpha I + \frac{1}{9\alpha} C_y C_y^T.
$$

The crucial point of our algorithm is the following eigenvalue problem:

$$\begin{pmatrix} 0 & C_y^T \\ C_y & 0 \end{pmatrix} \begin{pmatrix} \varphi_j \\ \psi_j \end{pmatrix} = \nu_j \begin{pmatrix} \varphi_j \\ \psi_j \end{pmatrix}, \qquad j = 1, \ldots, k_y, \tag{4.13}$$

where $k_y = 2n_y - 1$. The problem (4.13) can be reduced to a smaller eigenvalue problem for either a positive definite matrix $C_y C_y^T$ or a positive semi-definite matrix $C_y^T C_y$. Since both matrices are tridiagonal, the reduced eigenvalue problems can be easily solved. To be more specific, we consider the second problem:

$$C_y^T C_y \varphi_j = \nu_j^2 \varphi_j, \qquad j = 1, \ldots, n_y. \tag{4.14}$$

Let the eigenvalues ν_j be enumerated in the following way:

$$\nu_1 > \nu_2 > \ldots > \nu_{n_y} = 0 > \nu_{n_y+1} > \ldots > \nu_{2n_y-1}.$$

It is easy to check that $\varphi_{n_y} = (1, -1, 1, -1, \ldots, 1)^T$ and $\psi_{n_y} = (0, \ldots, 0)^T$ are eigenvectors corresponding to the zero eigenvalue. The eigenvectors ψ_j are given by

$$\psi_j = \frac{1}{\nu_j} C_y \varphi_j, \qquad j = 1, \ldots, n_y - 1, n_y + 1, \ldots, 2n_y - 1.$$

The system of eigenvectors φ_j, $j = 1, \ldots, n_y$, forms an orthogonal basis in $\Re^{n_y}$. Without loss of generality we may assume that the vectors are orthonormal. Similarly, the system of eigenvectors ψ_j, $j = 1, \ldots, n_y - 1$, forms an orthonormal basis in $\Re^{n_y - 1}$. Indeed,

$$\psi_i^T \psi_j = \frac{1}{\nu_i \nu_j} \varphi_i^T C_y^T C_y \varphi_j = \frac{\nu_j}{\nu_i} \varphi_i^T \varphi_j = \delta_{ij},$$

where δ_{ij} is the Kronecker symbol. Thus, we have the Fourier expansions

$$\lambda_{2i} = \sum_{j=1}^{n_y} \lambda_{2i}^j \varphi_j \qquad \text{and} \qquad \lambda_{2i-1} = \sum_{j=1}^{n_y-1} \lambda_{2i-1}^j \psi_j \tag{4.15}$$

for the vectors with even and odd indices, respectively. Similar expansions hold for the vectors b_i in (4.12), $i = 1, \ldots, k_x$. Let b_i^j be the corresponding Fourier coefficients. The orthogonality of basis vectors imply that

$$b_{2i}^j = \varphi_j^T b_{2i} \qquad \text{and} \qquad b_{2i-1}^j = \psi_j^T b_{2i-1}$$

for the vectors with even and odd indices, respectively. Then, substituting all expansions in (4.12) and using orthogonality of the basis vectors, we get n_y independent five-diagonal systems:

$$\begin{pmatrix} \sigma_j & \nu_j & & & & & \\ \nu_j & 10\alpha & \nu_j & \alpha & & & \\ & \nu_j & \sigma_j & \nu_j & & & \\ & \alpha & \nu_j & 10\alpha & \nu_j & \alpha & \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ & & & \alpha & \nu_j & 10\alpha & \nu_j \\ & & & & & \nu_j & \sigma_j \end{pmatrix} \begin{pmatrix} \lambda_1^j \\ \lambda_2^j \\ \lambda_3^j \\ \lambda_4^j \\ \cdots \\ \lambda_{k_x-1}^j \\ \lambda_{k_x}^j \end{pmatrix} = \begin{pmatrix} b_1^j \\ b_2^j \\ b_3^j \\ b_4^j \\ \cdots \\ b_{k_x-1}^j \\ b_{k_x}^j \end{pmatrix} \tag{4.16}$$

where $\sigma_j = 4 + 2\nu_j^2/9$ and $j = 1, \ldots, n_y$. After solving systems (4.16), the solution vector λ is computed by formulae (4.15). The simple calculations give us the following estimate.

Lemma 4.1 *The algorithm (4.15)–(4.16) requires $O(n_x n_y^2)$ arithmetical operations.*

When $n_y > n_x$, we can obtain a better arithmetical complexity by switching x and y directions. In other words, making the Fourier transformation in the x direction, we get n_x five diagonal systems of order k_y. Then, the arithmetical complexity of the algorithm will be $O(n_x^2 n_y)$.

Let us assume that the right hand side b in (4.11) has $O(n_x)$ non-zero components and only $O(n_x)$ components of the solution vector λ have to be computed. Such problems were studied in [49] and called there the *partial problems*. Their arithmetical complexity is obviously less than that of the whole problem. The following statement follows from the main results of [49].

Lemma 4.2 (Parallel partial problem) *Let $b_i \neq 0$ for a few indices $i_1, \ldots, i_s$ where $s = O(1)$. Furthermore, assume only solution vectors $\lambda_{\tilde{i}}$ with indices $\tilde{i}_1, \ldots, \tilde{i}_l$*

where $l = O(1)$, have to be computed. Then, algorithm (4.15)–(4.16) requires $O(n_x n_y)$ arithmetical operations.

The partial problems play the important role in the fast direct method (FDM) formulated and analyzed below. We begin with a hierarchical one-dimensional partitioning of the computational domain Ω. For simplicity, we assume that each new level of the partition is made by splitting all subdomains in two approximately equal parts by lines passing through mesh points $\boldsymbol{x}_{1,j}$ and $\boldsymbol{x}_{n_x,j}$ for some j. We denote these subdomains by Ω_s^l, where l is the refinement level, $l \le L$, and $s = 1, \ldots, 2^l$. Set $\Omega_1^0 \equiv \Omega$ and $b^{(L)} \equiv b$. Obviously that $L \approx \log_2 n_y$.

Let $P_s^l \colon \Re^n \to \Re^n$ be a projector associated with an interior of subdomain Ω_s^l. In other words, the non-zero entries of P_s^l correspond to "solution points" from $int(\Omega_s^l)$. Define

$$P^l = \sum_{s=1}^{2^l} P_s^l \qquad \text{and} \qquad A^l = P^l A P^l, \qquad l = 0, \ldots, L. \tag{4.17}$$

We assume that matrices A^l, $l = 0, \ldots, L$, are nonsingular in $im(P^l)$.

Algorithm 4.1 (Fast Direct Method)

1. *The solution vector λ is shifted by the vectors $P^l v_l$ such that each new right hand side vector has fewer non-zero entries:*

$$\begin{aligned} A^l v^{(l)} &= P^l b^{(l)}, \\ b^{(l-1)} &= b^{(l)} - A P^l v^{(l)}, \qquad l = L, \ldots, 1. \end{aligned} \tag{4.18}$$

2. *The partial solutions $(I - P^l)\lambda^{(l)}$ are computed recursively:*

$$\begin{aligned} & A^l \lambda^{(l)} = P^l b^{(l)} - P^l A (I - P^l) \lambda^{(l-1)}, \\ & (I - P^l)\lambda^{(l)} = (I - P^l)\lambda^{(l-1)}, \qquad l = 0, \ldots, L, \end{aligned} \tag{4.19}$$

   where $\lambda^{(-1)} = (0, \ldots, 0)^T$. Then $\lambda = \lambda^{(L)} = A^{-1} b$.

We refer to [41, 50, 74] for the proofs of Algorithm 4.1. Its parallel implementation has been studied in [65]. Arithmetical complexity of the algorithm depends on the arithmetical complexity of partial problems in (4.18) and (4.19). In a particular case, when a stiffness matrix is given by the Kronecker product, the structure of the partial problems has been discussed in [41, 50, 65]. Lemma 4.2 is useful for studying the partial problems in the case of mixed-hybrid discretizations.

Algorithm 4.1 is illustrated by Fig. 4.2. The order of computations is schematically shown on the right picture, where each star means a horizontal row of "solution points" from the left figure. In this example $L = 2$. The non-zero components of $b^{(1)}$ (resp., $b^{(2)}$) correspond to "solution points" with y-coordinate 1, 2 and 3 (resp., 2). The partial solution $\lambda^{(0)}$ (resp., $\lambda^{(1)}$) is computed at "solution points" with y-coordinate 2 (resp., 1 and 3). It follows from (4.19) that the non-zero components of $(I - P^l)\lambda^{(l-1)}$ are already components of λ [41, 74].

Lemma 4.3 *Efficient implementation of Algorithm 4.1 requires $O(n_y n_x \log_2 n_y)$ arithmetical operations.*

<u>Proof</u>. Consider the first step of the algorithm. Note that

$$\begin{aligned} b^{(l-1)} &= b^{(l)} - AP^l v^{(l)} \\ &= b^{(l)} - (I - P^l)AP^l v^{(l)} - A^l v^{(l)} = (I - P^l)b^{(l-1)}, \qquad l = L, \ldots, 1. \end{aligned}$$

Thus, the non-zero entries in $b^{(l-1)}$, $1 \leq l \leq L$, correspond to mesh edges living on interfaces $\partial\Omega_s^{l-1} \cap \partial\Omega_{s+1}^{l-1}$, $s = 1, \ldots, 2^{l-1} - 1$. The sparsity structure of A implies that a computation of $b^{(l-1)}$ involves the entries of $P^l v^{(l)}$ associated with nearby mesh edges. Therefore, the computation of vector $b^{(l)}$ requires $O(n_x 2^l)$ operations. Summing over l and recalling that $2^L \approx n_y$, we conclude that all right-hand side vectors can be computed for $O(n_x n_y)$ operations.

The definition of the projector P^l implies that matrix A^l is block diagonal. Thus, the solution of (4.18) is reduced to 2^l independent problems associated with subdomains Ω_s^l, $s = 1, \ldots, 2^l$. Moreover, each of the smaller problems is the partial problem in a sense that the right hand side has n_x non-zero entries and $O(n_x)$ non-zero entries of the solution have to be found. By Lemma 4.2, each of the smaller problems requires $O(n_x 2^{L-l})$ operations. Therefore, the total arithmetical complexity of (4.18) is $O(n_x n_y \log_2 n_y)$.

Let us consider the second step in the algorithm. The computation of the right-hand side in (4.19) involves the partial solution $(I - P^l)\lambda^{(l-1)}$ which has $n_x(2^l - 1)$ non-zero entries. Thus, the right-hand side in (4.19) may be computed for $O(n_x 2^l)$ operations. The sparsity structure of matrix A and $b^{(l-1)} = (I - P^l)b^{(l-1)}$ imply that the number of non-zero components in the right-hand side is $O(n_x 2^l)$. Since A^l is the block diagonal matrix, the solution of (4.19) is reduced to 2^l independent partial problems with arithmetical complexity $O(n_x 2^{L-l})$. Summing over l, we prove the lemma. $\square$

Note that the same arithmetical complexity has been obtained in [41, 74] for algebraic problems with separable matrices.

4.2.2 Applications of 2D algorithm

The convergence of Algorithm 4.1 can be shown under very general assumptions about the projectors P^l, $l = 1, \ldots, L$ (see. e.g., [65]). On the contrary, its effective implementation presumes existence of inexpensive partial solvers based on a separation of variables. The latter imposes strong restrictions on the mesh topology and the problem coefficients. Fortunately, similar restrictions are used by engineers for modeling complicated diffusive processes, propagation of electromagnetic and acoustic waves, etc. However, the FDM is mainly applied for accelerating convergence of iterative methods.

In this subsection we discuss some generalizations and applications of Algorithm 4.1. In particular, we apply the algorithm for an iterative solution of diffusion equations on topologically rectangular meshes.

The binary (radix-2) hierarchy of subdomains Ω_s^l, $s = 1, \ldots, 2^l$, $l = 0, \ldots, L$, can be replaced by a more general partition. The radix-q variant of the algorithm has been carefully studied in [65] for the case of separable matrices. The authors have shown that the arithmetical complexity of the algorithm is minimized when $4 \le q \le 8$.

For the future analysis it will be useful to rewrite the stiffness matrix A in an assembling form

$$A = \sum_{i=1}^{n_x} \sum_{j=1}^{n_y} \mathcal{N}_{ij} A_{ij} \mathcal{N}_{ij}^T, \tag{4.20}$$

where $A_{ij} \in \Re^{s\times s}$ is an elemental matrix, s is the number of edges of e_{ij}, expect edges on $\partial\Omega$, and $\mathcal{N}_{ij} \in \Re^{n\times s}$ is an assembling matrix. Assume that the edges of each quadrilateral are ordered counter-clockwise starting with the bottom edge.

Theorem 4.2 *Let $\mathcal{T}_x$ and $\mathcal{T}_y$ from (4.10) be arbitrary and uniform meshes, respectively. In addition, let $\boldsymbol{K}$ and c_{pp} be constant in each mesh cell and satisfy one of the following:*

$$\boldsymbol{K}(\boldsymbol{x}) = \boldsymbol{K}_x(x)\, K_y(y), \qquad c_{pp}(\boldsymbol{x}) = 0, \tag{4.21a}$$

$$\boldsymbol{K}(\boldsymbol{x}) = \boldsymbol{K}_x(x), \qquad c_{pp}(\boldsymbol{x}) = const \tag{4.21b}$$

where $\boldsymbol{K}_x(x)$ is a diagonal tensor and $K_y(y)$ is a scalar function. Then, the problem (4.11) can be solved by the method of separation of variables.

Proof. We shall prove this theorem only under assumption (4.21a). The second part can be proved by a similar technique.

Let $\boldsymbol{K}_{ij}$ be the value of $\boldsymbol{K}(\boldsymbol{x})$ over mesh cell e_{ij}. Then, $\boldsymbol{K}_{ij} = \boldsymbol{K}_{xi}\, K_{yj}$ by assumption (4.21a). The symmetry of the problem implies that elemental matrices A_{ij} in (4.20) have the following structure:

$$A_{ij} = K_{yj} \begin{pmatrix} d_i^{(1)} & b_i & c_i^{(1)} & b_i \\ b_i & d_i^{(2)} & b_i & c_i^{(2)} \\ c_i^{(1)} & b_i & d_i^{(1)} & b_i \\ b_i & c_i^{(2)} & b_i & d_i^{(2)} \end{pmatrix}. \tag{4.22}$$

Note that the elemental matrix for a near boundary mesh cell is a diagonal block of the matrix in (4.22). Let $K_{xi,s}$, $s = 1, 2$, be the diagonal entries of tensor $\boldsymbol{K}_{xi}$. Then, tedious but straightforward computations result in

$$
\begin{aligned}
b_i &= -3\alpha_i K_{xi,1} K_{xi,2}, \\
c_i^{(1)} &= \alpha_i (2K_{xi,1}K_{xi,2} - \beta_i^2 K_{xi,2}^2), \qquad & c_i^{(2)} &= \alpha_i (2K_{xi,1}K_{xi,2} - \beta_i^{-2} K_{xi,1}^2), \\
d_i^{(1)} &= \alpha_i (4K_{xi,1}K_{xi,2} + \beta_i^2 K_{xi,2}^2), \qquad & d_i^{(2)} &= \alpha_i (4K_{xi,1}K_{xi,2} + \beta_i^{-2} K_{xi,1}^2),
\end{aligned}
$$

where

$$
\beta_i = \frac{h_{xi}}{h_y} \qquad \text{and} \qquad \alpha_i = \frac{\beta_i}{K_{xi,2}\beta_i^2 + K_{xi,1}}, \qquad i = 1, \ldots, n_x.
$$

Let us revise formulae (4.12)–(4.16). Let D_y and $\hat{D}_y$ be diagonal matrices given by

$$
D_y = \operatorname{diag}\{K_{y1}, \ldots, K_{yn_y}\} \quad \text{and} \quad \hat{D}_y = \operatorname{diag}\{K_{y1} + K_{y2}, \ldots, K_{yn_y-1} + K_{yn_y}\},
$$

respectively. Then formula (4.22) implies that

$$
\begin{aligned}
D_{2i} &= (d_i^{(2)} + d_{i+1}^{(2)}) D_y, \\
E_{2i-1} &= c_i^{(2)} D_y,
\end{aligned}
\qquad
C_{2i-1} = C_{2i} = \begin{pmatrix} b_i & b_i & & \\ & \ddots & \ddots & \\ & & b_i & b_i \end{pmatrix} D_y,
$$

and

$$
A_{2i-1} = d_i^{(1)} \hat{D}_y + c_i^{(1)} \begin{pmatrix} 0 & K_{y2} & & \\ K_{y2} & 0 & \ddots & \\ & \ddots & \ddots & K_{yn_y-1} \\ & & K_{yn_y-1} & 0 \end{pmatrix}
$$

for $1 \le i \le n_x$. Straightforward calculations show that

$$
A_{2i-1} = \left(d_i^{(1)} - c_i^{(1)}\right) \hat{D}_y + \frac{c_i^{(1)}}{b_i^2} C_{2i-1} D_y^{-1} C_{2i-1}^T .
$$

To prove the theorem, it is sufficient to show that the following conditions are satisfied. First, the generalized eigenvalue problems

$$
\begin{pmatrix} 0 & C_i^T \\ C_i & 0 \end{pmatrix} \begin{pmatrix} \varphi \\ \psi \end{pmatrix} = \nu \begin{pmatrix} D_y & 0 \\ 0 & \hat{D}_y \end{pmatrix} \begin{pmatrix} \varphi \\ \psi \end{pmatrix}, \qquad 1 \le i \le k_x, \tag{4.23}
$$

have the same set of eigenvectors φ_j and ψ_j, $j = 1, \ldots, k_y$. Second, the eigenvectors ψ_j are also eigenvectors of matrices A_{2i-1}. Then, the rest of the method of separation of variables is given by formulae similar to (4.15)–(4.16). The generalized eigenvalue problem (4.23) implies:

$$C_i\, D_y^{-1}\, C_i^T \psi = \nu^2 \hat{D}_y \psi, \qquad 1 \le i \le k_x.$$

It immediately follows from the form of matrices C_i and A_i, that both conditions hold. $\square$

The FDM can be generalized for problems with different types of boundary conditions: Neumann, Dirichlet and Robin. The only requirement here is to have the same type of the boundary condition for each of four sides of rectangle Ω. For the Robin boundary condition, we additionally require that the corresponding coefficients are constant.

In rest of this subsection we study iterative methods for a general diffusion equation. We assume that the problem coefficients are functions of the space variable $\boldsymbol{x}$, and $\mathcal{T}_h$ is a topologically rectangular mesh (e.g., mesh on Fig. 4.3).

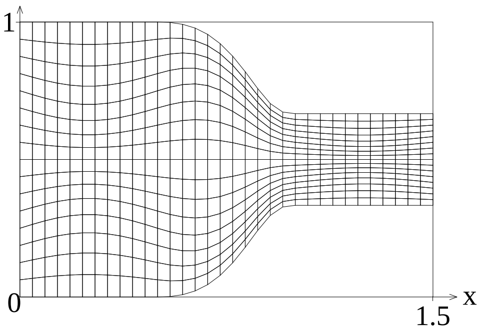

Figure 4.3: An example of a topologically rectangular mesh.

In a general case, the FDM may not be applicable. However, the problem (4.11) may be solved iteratively with a symmetric preconditioner B given by

$$B = \sum_{i=1}^{n_x} \sum_{j=1}^{n_y} \mathcal{N}_{ij} B_{ij} \mathcal{N}_{ij}^T, \tag{4.24}$$

where B_{ij} is a preconditioner for A_{ij}. Then, the following classical algebraic result holds [41].

Lemma 4.4 *Let the matrices B_{ij}, $i = 1, \ldots, n_x$, $j = 1, \ldots, n_y$, satisfy*

$$\frac{1}{c}(B_{ij}v,\, v) \leq (A_{ij}v,\, v) \leq c(B_{ij}v,\, v) \qquad \forall v \in \Re^s, \quad s = \dim(B_{ij}),$$

where c is a positive constants independent of i and j. Then

$$\frac{1}{c}(Bv,\, v) \leq (Av,\, v) \leq c(Bv,\, v) \qquad \forall v \in \Re^n. \tag{4.25}$$

One of the possible ways for choosing matrices B_{ij} in the form (4.22) is to average the problem coefficients in vertical mesh strips $\bigcup_{j=1}^{n_y} e_{ij}$, $i = 1, \ldots, n_x$. In some applications, like simulation of flows in porous media, these strips are adapted to media with different properties which are changed slightly inside the strips. Assume that it is true for our problem. In addition, assume that the aspect ratio of mesh cells is changed slightly in each of the strips. However, we may allow the average aspect ratio to be different for different strips. Under these two assumptions, it can be easily shown that the constant c in (4.25) is independent of jumps in coefficients and mesh topology.

We illustrate the speculations above by a few numerical experiments. Let Ω be the domain shown on Fig. 4.3, $\Gamma = \partial\Omega$, and $c_{pp} = 0$.

The condition number of $B^{-1}A$ on a sequence of uniformly refined meshes is shown in Table 4.1 for different diffusion tensors $\boldsymbol{K}$. The computational meshes were obtained through a continuous mapping of rectangular meshes preserving convexity of mesh cells.

In the first experiment we take a scalar tensor $\boldsymbol{K} = \boldsymbol{I}$. In the second experiment we replace it by a diagonal tensor $\boldsymbol{K} = \mathrm{diag}\{1,\, 10\}$. In the third experiment we take a piecewise constant diffusion tensor $\boldsymbol{K}(\boldsymbol{x}) = \boldsymbol{K}_x(x)K_y(y)$ where $\boldsymbol{K}_x(x)$ and $K_y(y)$ are given by

$$\boldsymbol{K}_x(x) = \begin{cases} \mathrm{diag}\{1,\, 1\}, & x < 0.5, \\ \mathrm{diag}\{1,\, 10\}, & x \geq 0.5 \end{cases} \quad \text{and} \quad K_y(y) = \begin{cases} 1, & y < 0.5, \\ 10^3, & y \geq 0.5, \end{cases} \tag{4.26}$$

respectively. In the fourth experiment the last tensor is rotated clockwise by 30°.

In order to construct matrices B_{ij} we map each vertical strip $\cup_{j=1}^{n_y} e_{ij}$ to a closest $1 \times n_y$ rectangular grid and use formula (4.22) for the main diagonal of tensor $\boldsymbol{K}_{ij} = \boldsymbol{K}|_{e_{ij}}$. Note that both A_{ij} and B_{ij} are approximations of the elliptic operator. The classical finite element theory states that condition number of $B^{-1}A$ depends on $C^{1,\infty}$-norm of this mapping [23]. When it is small, the condition number is just slightly deteriorated by the mesh topology (see first column in Table 4.1).

Grid	Experiment 1	Experiment 2	Experiment 3	Experiment 4
15x8	4.49	2.70	2.74	17.2
30x16	6.98	3.22	3.18	25.9
60x32	8.62	3.28	3.35	30.5
120x64	8.52	3.26	3.39	28.0
240x128	8.18	3.26	3.40	28.3
480x256	8.10	3.26	3.40	28.5

Table 4.1: $\mathrm{Cond}(B^{-1}A)$ on a sequence of uniformly refined 2D grids.

Comparing the third and fourth columns in Table 4.1 we observe that the condition number is independent of jumps in coefficients. Moreover, one order of anisotropy in the diffusion tensor is effectively suppressed by the preconditioner. However, it is only true when the anisotropy directions are aligned with the orientation of mesh edges. The results from the fifth column tell us that the rotation of the diffusion tensor deteriorates the preconditioner quality by about 8 times.

4.2.3 Fast direct method for 3D problems

We proceed with the analysis of a 3D problem (4.7) associated with a cubic grid $\mathcal{T}_h$. For the sake of simplicity we assume that $\boldsymbol{K} = \boldsymbol{I}$, $c_{pp} = 0$, and $\Gamma = \partial\Omega$. A few important generalizations will be discussed later. The cubic mesh is uniquely determined by a set of mesh points. In addition to one-dimensional meshes (4.10), we define

$$\mathcal{T}_z = \{[z_k,\, z_{k+1}]:\ z_1 < \ldots < z_{n_k+1}\}.$$

Then

$$\mathcal{T}_h = \{e_{ijk}:\ 1 \leq i \leq n_x,\ 1 \leq j \leq n_y,\ 1 \leq k \leq n_z\}$$

where e_{ijk} is a parallelepiped with a left-front-bottom vertex $(x_i,\, y_j,\, z_k)$ and right-back-top vertex $(x_{i+1},\, y_{j+1},\, z_{k+1})$. We denote the mesh points by $\boldsymbol{x}_{ijk} = (x_i,\, y_j,\, z_k)$, $i = 1, \ldots, n_x + 1$, $j = 1, \ldots, n_y + 1$ and $k = 1 \ldots, n_z + 1$. Without loss of generality we assume that $x_i = i$, $y_j = j$ and $z_k = k$. In addition, let $\boldsymbol{x}_{i,j+\frac{1}{2},k+\frac{1}{2}}$, $\boldsymbol{x}_{i+\frac{1}{2},j,k+\frac{1}{2}}$ and $\boldsymbol{x}_{i+\frac{1}{2},j+\frac{1}{2},k}$ be the middle face points which are associated with components of the solution vector λ. We shall call them the "solution points". Then, straightforward calculations show that the only non-zero entries of the matrix A are 6 (diagonal entries), 1 (connections between "solution points" on parallel neighboring faces), and -1 (connections between other "solution points" on faces belonging to the same mesh cell).

The solution vector λ can be again found by Algorithm 4.1. However, its effective implementation depends on the careful choice of projectors P^l. We begin with a hierarchical one-dimensional partitioning of the computational domain Ω along the z-axis. For the sake of simplicity we assume that each new level of the partition is made by splitting all subdomains in two approximately equal parts by planes orthogonal to the z-axis and passing through "solution points" $\boldsymbol{x}_{\frac{1}{2},\frac{1}{2},k}$ for some k. We denote these subdomains by Ω_s^l, where l is the refinement level, $l \leq L$, and $s = 1, \ldots, 2^l$. Set $\Omega_1^0 \equiv \Omega$ and $b^{(L)} \equiv b$. Obviously, $L \approx \log_2 n_z$.

Let $P_s^l \colon \Re^n \to \Re^n$ be a projector associated with the interior of the subdomain Ω_s^l. In other words, the non-zero entries of P_k^s correspond to "solution points" from $int(\Omega_s^l)$. Let P^l and A^l be given by (4.17).

Lemma 4.5 *Efficient implementation of Algorithm 4.1 requires $O(n_y n_x n_z \log_2 n_y \log_2 n_z)$ arithmetical operations.*

Proof. The proof repeats the main steps from Lemma 4.3. In particular, we may immediately conclude that implementation of (4.18)–(4.19) requires

$$O\left(n_x n_y n_z + \sum_{l=1}^{L} W(l)\right)$$

operations. Here $W(l)$ is the arithmetical complexity of the partial problem $A^l v = g$, where the right-hand side vector g has $O(n_x n_y 2^l)$ non-zero components and $O(n_x n_y 2^l)$ components of the solution vector v have to be computed. Note that these components correspond to "solution points" living on a few planes orthogonal to the z-axis.

Since A^l is the block diagonal matrix, it is sufficient to analyze the partial problem only for the matrix $A^0 = A$. The key point of our analysis is a reduction of the 3D partial problem to a full 2D problem studied above.

We partition components of vector v into $k_{xy} = 3n_x n_y - n_x - n_y$ disjoint groups as follows. Each group consists of either n_z or $n_z - 1$ components corresponding to "solution points" with the same x and y coordinates. These points are located on vertical lines intersecting the xy-plane at points $(x_{i+\frac{1}{2}},\, y_j)$, $(x_i,\, y_{j+\frac{1}{2}})$ and $(x_{i+\frac{1}{2}},\, y_{j+\frac{1}{2}})$. We shall use notations $v_{i+\frac{1}{2},j}$, $v_{i,j+\frac{1}{2}}$ and $v_{i+\frac{1}{2},j+\frac{1}{2}}$, respectively, for the corresponding group of components. In addition, we assume that inside each group the vector components are ordered by the z-coordinates of the corresponding "solution points". Then, a Fourier transformation in z-direction will give us a two-dimensional structure shown on Figure 4.4 (compare with Fig. 4.2). Let us show that such a transformation is possible.

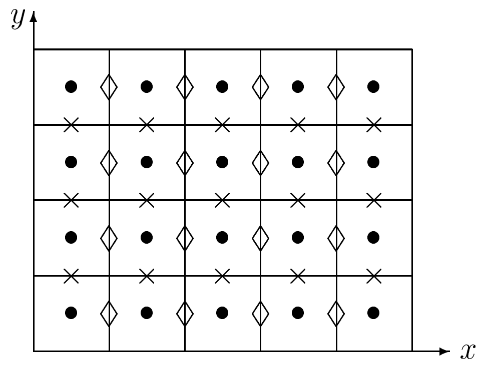

Figure 4.4: Schematic representation of a 3D partition with $k_{xy} = 51$.

Let $C_z \in \Re^{(n_z-1)\times n_z}$ be a two-diagonal rectangular matrix and $A_z \in \Re^{(n_z-1)\times(n_z-1)}$ be a tridiagonal symmetric matrix given by

$$C_z = \begin{pmatrix} -1 & -1 & & \\ & \ddots & \ddots & \\ & & -1 & -1 \end{pmatrix} \qquad \text{and} \qquad A_z = \begin{pmatrix} 6 & 1 & & \\ 1 & 6 & \ddots & \\ & \ddots & \ddots & 1 \\ & & 1 & 6 \end{pmatrix},$$

respectively. Then, the problem (4.11) can be rewritten in a block form associated with the partition above:

$$\begin{aligned}
6v_{i,j+\frac{1}{2}} &+ C_z^T(v_{i-\frac{1}{2},j+\frac{1}{2}} + v_{i+\frac{1}{2},j+\frac{1}{2}}) \\
&- \sum_{s=\pm 1} v_{i+s,j+\frac{1}{2}} - \sum_{s=\pm 1/2}\sum_{r=0,1} v_{i+s,j+r} = g_{i,j+\frac{1}{2}}, \\
6v_{i+\frac{1}{2},j} &+ C_z^T(v_{i+\frac{1}{2},j-\frac{1}{2}} + v_{i+\frac{1}{2},j+\frac{1}{2}}) \\
&- \sum_{r=\pm 1} v_{i+\frac{1}{2},j+r} - \sum_{s=0,1}\sum_{r=\pm 1/2} v_{i+s,j+r} = g_{i+\frac{1}{2},j}, \\
A_z v_{i+\frac{1}{2},j+\frac{1}{2}} &+ C_z(v_{i,j+\frac{1}{2}} + v_{i+1,j+\frac{1}{2}} \\
&\qquad + v_{i+\frac{1}{2},j} + v_{i+\frac{1}{2},j+1}) = g_{i+\frac{1}{2},j+\frac{1}{2}}
\end{aligned} \tag{4.27}$$

where vectors with indices beyond permissible intervals are zero vectors. Let ν_k and (φ_k, ψ_k), $k = 1, \ldots, 2n_z - 1$ be solutions of the eigenvalue problem (4.13) with the matrix C_z instead of C_y. Similarly to (4.15), we have the Fourier expansions:

$$v_{i+\frac{1}{2},j+\frac{1}{2}} = \sum_{k=1}^{n_z-1} v^k_{i+\frac{1}{2},j+\frac{1}{2}}\psi_k, \qquad v_{i,j+\frac{1}{2}} = \sum_{k=1}^{n_z} v^k_{i,j+\frac{1}{2}}\varphi_k, \qquad v_{i+\frac{1}{2},j} = \sum_{k=1}^{n_z} v^k_{i+\frac{1}{2},j}\varphi_k.$$

Similar expansions hold for the right-hand side vectors g^k in (4.27). Let $g^k_{i+\frac{1}{2},j+\frac{1}{2}}$, $g^k_{i,j+\frac{1}{2}}$ and $g^k_{i+\frac{1}{2},j}$ be the corresponding Fourier coefficients. The orthogonality of eigenvectors implies that

$$g^k_{i+\frac{1}{2},j+\frac{1}{2}} = g^T_{i+\frac{1}{2},j+\frac{1}{2}}\psi_k, \qquad g^k_{i,j+\frac{1}{2}} = g^T_{i,j+\frac{1}{2}}\varphi_k \quad \text{and} \quad g^k_{i+\frac{1}{2},j} = g^T_{i+\frac{1}{2},j}\varphi_k. \tag{4.28}$$

Substituting all expansions in (4.27) and using the orthogonality of the eigenvectors, we get n_z independent two-dimensional problems for the Fourier coefficients. The sparsity structure of these problems is schematically illustrated by Fig. 4.4. The unknowns may be connected only with neighboring unknowns sharing the same cell. Thus, unknowns marked by dots are connected with at most four unknowns and can be easily eliminated with $O(n_x n_y)$ operations. The resulting problem will have the same structure as the two-dimensional problem studied in Section 4.2.1. By Lemma 4.3 its solution requires $O(n_x n_y \log_2 n_y)$ operations.

By the previous assumptions, computation of Fourier coefficients (4.28) requires $O(n_x n_y n_z)$ operations. The same number of operations is required to compute the partial solution. Summing all expenses, we get that

$$W(l) = O(n_x n_y n_z \log_2 n_y)$$

which immediately proves the assertion of the lemma. $\square$

4.2.4 Applications of 3D algorithm

The FDM method described above is generalized in the same ways as its 2D analog (see Section 4.2.2). First, we formulate an analog of Theorem 4.2. We rewrite the stiffness matrix A in an assembling form

$$A = \sum_{i=1}^{n_x} \sum_{j=1}^{n_y} \sum_{k=1}^{n_z} \mathcal{N}_{ijk} A_{ijk} \mathcal{N}_{ijk}^T, \tag{4.29}$$

where $A_{ijk} \in \Re^{s \times s}$ is an elemental matrix, s is the number of faces of e_{ijk} except faces of $\partial\Omega$, and $\mathcal{N}_{ijk} \in \Re^{n \times s}$ is the corresponding assembling matrix.

Theorem 4.3 *Let $\mathcal{T}_y$ and $\mathcal{T}_z$ be uniform grids and $\mathcal{T}_x$ be an arbitrary grid. Furthermore, let $\boldsymbol{K}$ and c_{pp} be constant in each mesh cell and satisfy one of the following:*

$$\boldsymbol{K}(\boldsymbol{x}) = \boldsymbol{K}_x(x)\, K_y(y)\, K_z(z), \qquad c_{pp}(\boldsymbol{x}) = 0, \tag{4.30a}$$

$$\boldsymbol{K}(\boldsymbol{x}) = \boldsymbol{K}_x(x), \qquad c_{pp}(\boldsymbol{x}) = const \tag{4.30b}$$

where $\boldsymbol{K}_x(x)$ is a diagonal tensor, and $K_y(y)$ and $K_z(z)$ are scalar functions. Then, the problem (4.11) can be solved by the method of separation of variables.

The proof repeats the main steps in the proof of Theorem 4.2. Therefore, it is omitted here.

The method can be generalized for the problems with different types of boundary conditions: Neumann, Dirichlet and Robin. The only requirement here is to have the same type of boundary condition for each of six sides of the parallelepiped Ω. In the case of topologically cubic non-orthogonal meshes, Theorem 4.3 is used to construct a separable preconditioner B,

$$B = \sum_{i=1}^{n_x} \sum_{j=1}^{n_y} \sum_{k=1}^{n_z} \mathcal{N}_{ijk} B_{ijk} \mathcal{N}_{ijk}^T, \tag{4.31}$$

where B_{ijk} is a preconditioner for A_{ijk}. The quality of this preconditioner is analyzed in Lemma 4.4. We illustrate it by the following example.

Let Ω be a parallelepiped with edges of length $l_x = l_y = 20$ and $l_z = 1$. We impose the homogeneous Dirichlet boundary condition on the top side of $\partial\Omega$, the homogeneous Neumann boundary condition on the rest of $\partial\Omega$, and set $c_{pp} = 0.1$. In addition, we restrict ourself to the case of orthogonal meshes and study in more detail the dependence of $\mathrm{Cond}(B^{-1}A)$ on the diffusion tensor $\boldsymbol{K}$.

In the first experiment we take a scalar tensor $\boldsymbol{K} = \boldsymbol{I}$. In the second experiment we replace it by a diagonal tensor $\boldsymbol{K} = \mathrm{diag}\{1,\ 10,\ 10\}$. In the third experiment we take a piecewise constant tensor $\boldsymbol{K}(\boldsymbol{x}) = \boldsymbol{K}_x(0.05x)K_y(0.05y)K_y(z)$ with $\boldsymbol{K}(x)$ and $K_y(\cdot)$ from (4.26) and assume that $\boldsymbol{K}_x$ is isotropic in the yz-plane. In the fourth experiment, the last tensor is rotated by 30° around the x-axis and by 30° around the z-axis.

In order to construct matrices B_{ijk} we replace the diffusion tensor $\boldsymbol{K}$ by its main diagonal and take $c_{pp} = 0$. The modified elliptic problem will satisfy conditions (4.30a). Therefore, its mixed-hybrid discretization with the RT finite elements gives us a separable matrix B.

The results of numerical experiments are shown in Table 4.2. The condition number is independent of jumps in coefficients. It is close to one in the first three experiments because the reaction coefficient is small with respect to the minimal eigenvalue of the stiffness matrix. However, if the principle directions of the diffusion tensor are rotated, the condition number begins to depend on the level of anisotropy (see the last column in Table 4.2) .

Grid	Experiment 1	Experiment 2	Experiment 3	Experiment 4
4x4x4	1.01	1.01	1.01	3.27
8x8x8	1.01	1.01	1.01	4.21
16x16x16	1.01	1.01	1.01	4.90
32x32x32	1.01	1.01	1.01	4.96
64x64x64	1.01	1.01	1.01	5.00

Table 4.2: $\mathrm{Cond}(B^{-1}A)$ on a sequence of uniformly refined 3D meshes.

Let us revise the block diagonal preconditioner $\hat{\mathcal{B}}$ from (4.5). The choice $B_\lambda = B$ with B given by (4.31) results in the block diagonal preconditioner spectrally equivalent to the matrix $\hat{\mathcal{A}}$. The results of numerical experiments show that the preconditioner will have excellent spectral properties (in a sense of Definition 3.1).

4.3 Multilevel preconditioner for $A_{\lambda\lambda}$

In this section we describe a multilevel preconditioner B_λ for the matrix $A_{\lambda\lambda}$ from (4.4). The preconditioner has been proposed in [44] and may be classified as a geometric multigrid. We prove its optimal arithmetical complexity and study numerically its preconditioning properties on prismatic partitions $\mathcal{T}_h = \{e_1, \ldots, e_m\}$. The partition $\mathcal{T}_h$ is obtained by splitting each cell of a cubic grid into 2 prisms as shown on Fig. 3.1.

In order to simplify notation, we shall write A and B instead of $A_{\lambda\lambda}$ and B_λ, respectively.

4.3.1 Abstract form of multilevel algorithm

We describe the multilevel preconditioner B in an abstract form applicable for arbitrary meshes. However, we assume the existence of a hierarchical structure of mesh superelements. In the general case, a construction of such a structure is a separate problem which is beyond the scope of this thesis. When the computational mesh may be mapped onto a submesh of a topologically cubic/prismatic mesh, a structure of superelements can be easily constructed.

We begin by introducing additional notation. Let Ω be a polygonal domain, $n_0 \equiv m$, $e_{0,i} \equiv e_i$, and $\mathcal{T}_{0,h} = \mathcal{T}_h$. Then, the stiffness matrix A can be written in an assembling form

$$A = \sum_{i=1}^{n_0} \mathcal{N}_{0,i} A_i \mathcal{N}_{0,i}^T$$

where $A_i \in \Re^{s_i \times s_i}$ is an elemental matrix, s_i is the number of faces of $e_{0,i}$ except faces on Γ where the pressure (and hence, the Lagrange multiplier λ) is zero, and $\mathcal{N}_{0,i} \in \Re^{n_\lambda \times s_i}$ is an assembling matrix.

Let us consider the generalized eigenvalue problem

$$A_i w = \nu M_{0,i} w$$

with a symmetric positive definite matrix $M_{0,i}$. Furthermore, let $W_{0,i}$ be a matrix of $M_{0,i}$-orthogonal eigenvectors, $W_{0,i} = [w_{0,i}^{(1)}, \ldots, w_{0,i}^{(s_i)}]$, and Λ_i be a matrix of eigenvalues:

$$\Lambda_i = \mathrm{diag}\{\nu_{0,i}^{(1)}, \ldots, \nu_{0,i}^{(s_i)}\}, \qquad 0 \leq \nu_{0,i}^{(1)} < \nu_{0,i}^{(2)} \leq \ldots \leq \nu_{0,i}^{(s_i)}.$$

Note that $w_{0,i}^{(1)} = \alpha_{0,i}(1, 1, \ldots, 1)^T$ and the spectral decomposition of matrix A_i is

$$A_i = M_{0,i} W_{0,i} \Lambda_i W_{0,i}^T M_{0,i}. \tag{4.32}$$

Let us define a sequence of partitions

$$\mathcal{T}_{l,h} = \{e_{l,i} \colon\ 1 \leq i \leq n_l\}, \qquad \bar{\Omega} = \bigcup_{i=1}^{n_l} e_{l,i}, \qquad l = 0, \ldots, L.$$

We assume that the partitions are nested, i.e.,

$$e_{l,i} = \bigcup_{j \in \mathfrak{S}_l(i)} e_{l-1,j} \quad \text{for} \quad l \geq 1$$

where $\mathfrak{S}_l(i)$ is a collection of indexes of elements from level $l-1$ whose union is the element $e_{l,i}$ from level l (see Fig. 4.5). When $\mathcal{T}_h$ is a topologically cubic mesh, the partitions $\mathcal{T}_{l,h}$ may be produced by the algorithm described below.

Remark 4.1 *Let $\mathcal{T}_{0,h}$ be an orthogonal mesh with cubic mesh elements $e_{0,i}$, $i = 1, \ldots, n_0$. The mesh elements are the result of cutting domain Ω by planes orthogonal to coordinate axes and passing through the mesh vertices. Let $\mathcal{P}_{x,i}$, $\mathcal{P}_{y,j}$ and $\mathcal{P}_{z,k}$ be the planes orthogonal to x, y and z axes, respectively. In order to construct $\mathcal{T}_{1,h}$, we take approximately half as many planes in each direction, e.g., every other plane. A new set of planes will cut Ω into superelements $e_{1,i}$, $i = 1, \ldots, n_1$. By construction,*

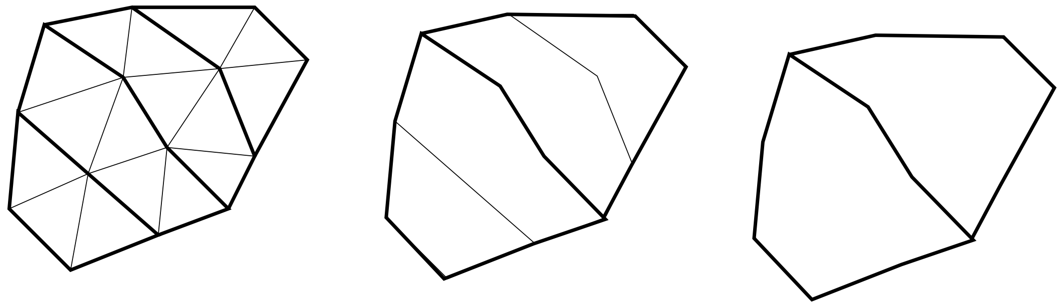

Figure 4.5: An example of a hierarchical structure of superelements.

$e_{1,i}$ is the union of mesh elements from $\mathcal{T}_{0,l}$. Repeating the procedure above recursively, we get a hierarchical sequence of partitions $\mathcal{T}_{l,h}$, $l = 0, \dots, L$.

Definition 4.1 *Let $M_{l,i}$ be diagonal matrices given by*

$$\begin{aligned} M_{0,i} &= \pi_{0,i}\,\mathrm{diag}(A_i), && i = 1, \dots, n_0, \\ M_{l,i} &= \sum_{j \in \mathfrak{S}_l(i)} \hat{\mathcal{N}}_{l-1,j} M_{l-1,j} \hat{\mathcal{N}}_{l-1,j}^T, && i = 1, \dots, n_l, \quad l = 0, \dots, L, \end{aligned} \tag{4.33}$$

where $\pi_{0,i}$ is a scalar factor such that $\nu_{0,i}^{(s_i)} = 1$, and $\hat{\mathcal{N}}_{l-1,i}$ is an assembling matrix for the superelement $e_{l,i}$.

Triples $\{A_i;\, M_{0,i};\, \mathcal{N}_{0,i}\}$ are substructures associated with the matrix A. Recall that we already studied substructures in Section 3.2.2.

With each superelement $e_{l,i}$, we associate a $M_{l,i}$-orthogonal projector $P_{l,i}M_{l,i}$ where $P_{l,i} = w_{l,i}w_{l,i}^T$, $w_{l,i} = \alpha_{l,i}(1, 1, \dots, 1)^T$, and $w_{l,i}^T M_{l,i} w_{l,i} = 1$. Let A_l be a matrix defined by

$$A_l = \sum_{i=1}^{n_l} \mathcal{N}_{l,i} A_{l,i} \mathcal{N}_{l,i}^T + \sum_{i=1}^{n_0} \nu_{0,i}^{(1)} \mathcal{N}_{0,i} M_{0,i} \mathcal{N}_{0,i}^T, \qquad A_{l,i} = M_{l,i} - M_{l,i} P_{l,i} M_{l,i}, \tag{4.34}$$

where $\mathcal{N}_{l,i}$ is an assembling matrix and $l = 0, \dots, L$. The first and the second sums in (4.34) correspond to the diffusion and reaction terms of (4.7), respectively.

Using Definition 4.1, we derive that

$$\begin{aligned} M_l \equiv \sum_{i=1}^{n_l} \mathcal{N}_{l,i} M_{l,i} \mathcal{N}_{l,i}^T &= \sum_{i=1}^{n_l} \sum_{j \in \mathfrak{S}_l(i)} \mathcal{N}_{l,i} \hat{\mathcal{N}}_{l-1,j} M_{l,i} \hat{\mathcal{N}}_{l-1,j}^T \mathcal{N}_{l,i}^T \\ &= \sum_{i=1}^{n_{l-1}} \mathcal{N}_{l-1,i} M_{l-1,i} \mathcal{N}_{l-1,i}^T \\ &= M_{l-1}. \end{aligned}$$

In order to simplify notations, we introduce diagonal matrices

$$\hat{M}_l = M_l + \sum_{i=1}^{n_0} \nu_{0,i}^{(1)} \mathcal{N}_{0,i} M_{0,i} \mathcal{N}_{0,i}^T, \qquad l = 0, \ldots, L.$$

Since $M_l = M_0$, we get that $\hat{M}_l = \hat{M}_0$ for $l = 1, \ldots, L$. At the moment, matrices A_l, $l = 0, \ldots, L$, have the same size and no multilevel structure is observed. However, matrices

$$C_l = \hat{M}_l - A_l = \sum_{i=1}^{n_l} \mathcal{N}_{l,i} M_{l,i} P_{l,i} M_{l,i} \mathcal{N}_{l,i}^T, \quad l = 0, \ldots, L,$$

have different ranks. It is obvious that $\text{Rank}(C_l) = n_l$, and the numbers n_l, $l = 0, \ldots, L$, form a decreasing geometric sequence. This feature is intensively exploited in the multilevel algorithm.

Let $V_{-1} \equiv \Re^{n_\lambda}$ be the original vector space for the Lagrange multipliers and

$$V_l = im(C_l), \quad l = 0, \ldots, L.$$

The definition of matrix C_l implies that

$$V_l = \{\lambda \in V_{-1}: \ \lambda = \sum_{i=1}^{n_l} a_i \mathcal{N}_{i,l} M_{l,i} w_{l,i}, \quad a_i \in \Re^1\}. \tag{4.35}$$

Since $w_{l,i} = \alpha_{l,i}(1, 1, \ldots, 1)^T$, Definition 4.1 implies that spaces V_l, $l = 0, \ldots, L$, are nested:

$$V_L \subset V_{L-1} \subset \ldots \subset V_0 \subset V_{-1}.$$

Lemma 4.6 *Let $\mathcal{T}_{l,h}$ be a sequence of cubic nested meshes from Remark 4.1. Then*

$$\frac{1}{c\, 2^l}(A_l \lambda, \lambda) \leq (A_{l-1}\lambda, \lambda) \leq c(A_l \lambda, \lambda) \qquad \forall \lambda \in V_{l-1}, \quad l = 0, \ldots, L. \tag{4.36}$$

where $A_{-1} \equiv A$ and c is independent of l and the discretization parameter h.

Proof. Let $l = 0$. The formula (4.34) yields

$$A_{0,i} = M_{0,i} - M_{0,i} w_{0,i} w_{0,i}^T M_{0,i} = M_{0,i} W_{0,i} \Lambda_{0,i} W_{0,i}^T M_{0,i}$$

where $\Lambda_{0,i} = \mathrm{diag}\{0, 1, 1, \ldots, 1\}$. Therefore, for any $\lambda \in V_{-1}$, we get

$$\frac{(A\lambda,\, \lambda)}{(A_0\lambda,\, \lambda)} \le \max_{1 \le i \le n_0} \frac{(A_i\, \mathcal{N}_{0,i}^T \lambda,\, \mathcal{N}_{0,i}^T \lambda)}{((A_{0,i} + \nu_{0,i}^{(1)} M_{0,i})\, \mathcal{N}_{0,i}^T \lambda,\, \mathcal{N}_{0,i}^T \lambda)} \le \max_{1 \le i \le n_0} \frac{\nu_{0,i}^{(s_i)}}{1 + \nu_{0,i}^{(1)}}.$$

Definition 4.1 implies that $\nu_{0,i}^{(s_i)} = 1$ and $\nu_{0,1}^{(1)} < 1$. Thus,

$$(A\lambda,\, \lambda) \le (A_0\lambda,\, \lambda) \qquad \forall \lambda \in V_{-1}.$$

The lower estimate is obtained by similar arguments.

Let $l > 0$. The definition of matrices A_{l-1} and A_l imply that

$$\frac{(A_{l-1}\lambda,\, \lambda)}{(A_l\lambda,\, \lambda)} \le \max \left\{ 1,\ \max_{1 \le i \le n_l} \frac{\left(\sum_{j \in \mathfrak{S}_l(i)} A_{l-1,j}\, \mathcal{N}_{l-1,j}^T \lambda,\, \mathcal{N}_{l-1,j}^T \lambda \right)}{\left(A_{l,i}\, \mathcal{N}_{l,i}^T \lambda,\, \mathcal{N}_{l,i}^T \lambda \right)} \right\}.$$

The maximum contains 1 because $\nu_{0,i}^{(1)} > 0$ for matrices associated with mesh elements having a common interface with Γ where the Dirichlet boundary condition is imposed. The second term in the last estimate is bounded from above by the maximal eigenvalue of the generalized eigenvalue problem

$$\sum_{j \in \mathfrak{S}_l(i)} \hat{\mathcal{N}}_{l-1,j} A_{l-1,j} \hat{\mathcal{N}}_{l-1,j}^T\, v = \nu A_{l,i}\, v, \qquad v \in V_{l,i}, \quad v \notin \ker(A_{l,i}), \tag{4.37}$$

where $V_{l,i} = \mathcal{N}_{l,i}^T V_{l-1}$ and $\hat{\mathcal{N}}_{l-1,j}$ is the assembling matrix from (4.33). Let

$$\begin{aligned} \hat{A}_{l,i} &= \sum_{j \in \mathfrak{S}_l(i)} \hat{\mathcal{N}}_{l-1,j} A_{l-1,j} \hat{\mathcal{N}}_{l-1,j}^T \\ &= M_{l,i} - \sum_{j \in \mathfrak{S}_l(i)} \hat{\mathcal{N}}_{l-1,j} M_{l-1,j} P_{l-1,j} M_{l-1,j} \hat{\mathcal{N}}_{l-1,j}^T \equiv M_{l,i} - \hat{C}_{l,i}, \\ A_{l,i} &= M_{l,i} - M_{l,i} P_{l,i} M_{l,i} \equiv M_{l,i} - C_{l,i}. \end{aligned}$$

The definition of $P_{l,i}$ implies that $C_{l,i}$ is a rank-1 matrix and

$$im(C_{l,i}) = \ker(A_{l,i}) = \ker(\hat{A}_{l,i}).$$

Therefore, the maximal eigenvalue of (4.37) is bounded by 1. Let us show that this bound is attained on an eigenvector $v \in \ker(\hat{C}_{l,i}) \cap V_{l,i}$. First, we observe that

$$\begin{aligned} 0 = \hat{C}_{l,i}v &= \sum_{j \in \mathfrak{S}_l(i)} \hat{\mathcal{N}}_{l-1,j} M_{l-1,j} w_{l-1,j} w_{l-1,j}^T M_{l-1,j} \hat{\mathcal{N}}_{l-1,j}^T v \\ &= \sum_{j \in \mathfrak{S}_l(i)} \hat{\mathcal{N}}_{l-1,j} M_{l-1,j} w_{l-1,j} \hat{v}_j, \qquad \hat{v}_j = w_{l-1,j}^T M_{l-1,j} \hat{\mathcal{N}}_{l-1,j}^T v, \end{aligned}$$

implies $\hat{v}_j = 0$ for all $j \in \mathfrak{S}_l(i)$. Second, the characterization property (4.35) of V_{l-1} and the definitions of the scalar $\hat{v}_j$ and space $V_{l,i}$ yield

$$0 = \hat{v}_j = w_{l-1,j}^T M_{l-1,j} \mathcal{N}_{l-1,j}^T \sum_{k=1}^{n_{l-1}} a_k \mathcal{N}_{l-1,k} M_{l-1,k} w_{l-1,k} \qquad \forall j \in \mathfrak{S}_l(i), \tag{4.38}$$

for some scalars a_k. Since the number of equations is much less than the number of unknowns, (4.38) has a non-trivial solution $(a_1, \ldots, a_{n_{l-1}})$. The vector v corresponding to this solution satisfies

$$\hat{A}_{l,i}v = M_{l,i}v = A_{l,i}v, \qquad v \notin \ker(A_{l,i}).$$

Thus, we proved that

$$(A_{l-1}\lambda,\, \lambda) \leq (A_l\lambda, \lambda) \qquad \forall \lambda \in V_{l-1}.$$

The lower estimate in (4.36) is obtained by similar arguments. The problem is reduced to computing a minimal eigenvalue of (4.37). Moreover, we may additionally require $v \in im(\hat{C}_{l,i})$. Multiplying both sides of (4.37) by $w_{l-1,k}^T M_{l-1,k} \mathcal{N}_{l-1,k}^T M_{l,i}^{-1}$, $k \in \mathfrak{S}_l(i)$, we get an eigenvalue problem

$$\hat{S}\hat{v} = \nu\hat{v}, \qquad \hat{v} \notin \ker(\hat{S}), \tag{4.39}$$

where $\hat{S} \in \Re^{8\times 8}$ is a symmetric matrix and $\hat{v} = (\hat{v}_1, \ldots, \hat{v}_8)^T$. Without loss of generality, we assume that $\mathfrak{S}_l(i) = \{1, \ldots, 8\}$. Then, the entries of $\hat{S}$ and $\hat{v}$ are given by

$$\begin{aligned}
\hat{s}_{kj} &= \delta_{kj} - w_{l-1,k}^T M_{l-1,k} \mathcal{N}_{l-1,k}^T M_{l,k}^{-1} \mathcal{N}_{l-1,j} M_{l-1,j} w_{l-1,j}, \\
\hat{v}_k &= w_{l-1,k}^T M_{l-1,k} \mathcal{N}_{l-1,k}^T v, \qquad 1 \le j, k \le 8.
\end{aligned}$$

The non-zero entries $\hat{s}_{jk}$ are proportional to the ratio of the superelement surface area to its volume, i.e., $\hat{s}_{jk} \sim 2^{-l}$. Therefore, simple calculations give that the minimal positive eigenvalue of (4.39) is bounded from below by $c2^{-l}$. This proves the lower estimate in (4.36). □

Algorithm 4.2 (Multilevel Preconditioner)

Let $H_0 \equiv B^{-1}$. Then, for any $g \in V_{-1}$, a matrix-vector product $H_0\, g$ is given by the following recursive algorithm:

1. *for $g_{l-1} \in V_{l-1}$ compute $x_l^0 = \hat{M}_0^{-1} g_{l-1}$, $\xi_l^0 = A_l x_l^0 - g_{l-1}$ and set $y_l^0 = -\xi_l^0$;*

2. *iterate $y_l^i = y_l^{i-1} - \gamma_i\; C_l H_{l+1} \xi_l^{i-1}$,*

   $$\xi_l^i = \xi_l^{i-1} - \gamma_i\; A_l H_{l+1} \xi_l^{i-1}, \qquad i = 1, \ldots, \beta_l;$$

3. *set $H_l\, g_{l-1} \equiv \hat{M}_0^{-1}(y_l^{\beta_l} + g_{l-1})$;*

where $l = 0, \ldots, L-1$, $g_{-1} \equiv g$, γ_i are coefficients of a Chebyshev polynomial, and H_L is a coarse grid solver.

Algorithm 4.2 fits into the standard multigrid scheme. However, a coarse grid projection is replaced by the *subspace entering* (Step 1), and a fine grid interpolation is replaced by the *subspace leaving* (Step 3). Note that H_{l+1} is an approximation to A_{l+1}^{-1}, $l = 0, \ldots, L-1$. When L is small enough, Lemma 4.6 states that matrices H_{l+1} and A_{l+1}^{-1} are spectrally equivalent. The opposite case will be studied numerically in Section 4.3.3. Surprisingly enough, that Algorithm 4.2 is stable for the large number of multigrid levels. This feature of the algorithm is still an open question.

4.3.2 Algorithm implementation

In this subsection we analyze the arithmetical complexity of Algorithm 4.2. Let $Q_l \equiv \Re^{n_l}$, and $\mathcal{K}_l \colon V_l \to Q_l$ be a bijective mapping between spaces Q_l and V_l. We shall use small letters for vectors in space V_l (e.g., ξ, g, z, etc) and calligraphic capital letters for their counterparts in space Q_l (resp. $\mathcal{E}$, $\mathcal{G}$, $\mathcal{Z}$, etc). Then, the characterization property (4.35) implies:

$$g_l = \mathcal{K}_l^{-1}(\mathcal{G}_l) = \sum_{i=1}^{n_l} \mathcal{G}_{l,i}\mathcal{N}_{l,i}M_{l,i}w_{l,i}.$$

1. The first step in Algorithm 4.2 is called the subspace entering. Indeed, given $g_{l-1} \in V_{l-1}$ we have to compute the residual $\xi_l^0 = -C_l\hat{M}_0^{-1}g_{l-1}$ living in the space V_l of a smaller dimension. Let $\mathcal{E}_l^0 = \mathcal{K}_0(\xi_l^0) \in Q_0$ be the counterpart of ξ_l^0. We have to consider two separate cases: $l = 0$ and $l > 0$.

(a) Let $l = 0$. Then

$$\xi^0 = -\sum_{i=1}^{n_0} \mathcal{N}_{0,i}M_{0,i}w_{0,i}w_{0,i}^T M_{0,i}\mathcal{N}_{0,i}\hat{M}_0^{-1}g = \sum_{i=1}^{n_0} \mathcal{N}_{0,i}M_{0,i}w_{0,i}\mathcal{E}_{0,i}^0$$

where

$$\mathcal{E}_{0,i}^0 = -w_{0,i}^T M_{0,i}\mathcal{N}_{0,i}^T\hat{M}_0^{-1}g. \tag{4.40}$$

This step requires $O(n_0)$ arithmetical operations.

(b) Let $l > 0$. Note that $g_{l-1} \in V_{l-1}$ is numerically represented by a vector $\mathcal{G}_{l-1} = \mathcal{K}_{l-1}(g) \in Q_{l-1}$, i.e.,

$$g_{l-1} = \sum_{i=1}^{n_{l-1}} N_{l-1,i}M_{l-1,i}w_{l-1,i}\mathcal{G}_{l-1,i}.$$

Note that we can not use a straightforward approach as in (4.40) to compute the residual $\mathcal{E}_l^0$ because it is too expensive. In fact, we have

$$\begin{aligned}
\mathcal{E}_{l,i}^0 &= -w_{l,i}^T M_{l,i}\mathcal{N}_{l,i}^T\hat{M}_0^{-1}\sum_{j=1}^{n_{l-1}} \mathcal{N}_{l-1,j}M_{l-1,j}w_{l-1,j}\mathcal{G}_{l-1,j} \\
&= -w_{l,i}^T \sum_{k\in\mathfrak{S}_l(i)} \hat{\mathcal{N}}_{l-1,k}M_{l-1,k}\mathcal{N}_{l-1,k}^T\hat{M}_0^{-1}\sum_{j=1}^{n_{l-1}} \mathcal{N}_{l-1,j}M_{l-1,j}w_{l-1,j}\mathcal{G}_{l-1,j}.
\end{aligned}$$

Suppose that we have a recursive procedure for computing coefficients

$$\Upsilon_{kl}^{l-1} = w_{l-1,k}^T M_{l-1,k} \mathcal{N}_{l-1,k} \hat{M}_0^{-1} \mathcal{N}_{l-1,j} M_{l-1,j} w_{l-1,j}. \tag{4.41}$$

Then

$$\mathcal{E}_{l,i}^0 = \sum_{k \in \mathfrak{S}_l(i)} \sum_{j \in \mathfrak{O}_{l-1}(k)} \frac{\alpha_{l,i}}{\alpha_{l-1,k}} \Upsilon_{kj}^{l-1} \mathcal{G}_{l-1,j} \tag{4.42}$$

where $\mathfrak{O}_{l-1}(k)$ denotes a collection of indices of neighboring superelements for the superelement $e_{l-1,k}$ including the superelement itself. The arithmetical complexity of computing $\mathcal{E}_l^0$ is $O(n_l)$. Efficient algorithms for recursive computing coefficients $\alpha_{l,i}$ and Υ_{kj}^l are discussed at the end of this subsection.

2. The second step in Algorithm 4.2 is called the *subspace iterations*. Since V_l is the image space of matrix C_l, we have

$$\xi_l^0 = -C_l \hat{M}_0^{-1} g_{l-1} \in V_l, \quad C_l H_{l+1} \xi_l^{i-1} \in V_l \quad \text{and} \quad y^i \in V_l, \qquad i = 0, \ldots, \beta_l.$$

From the third step in Algorithm 4.2, we derive that $\hat{M}_0 H_{l+1} \xi_l^{i-1} \in V_l$. Since $A_l = \hat{M}_l - C_l$, all residuals ξ_l^{i-1}, $i = 0, \ldots, \beta_l$, belong to V_l. Using the bijective mapping $\mathcal{K}_l$, we define

$$\mathcal{Z}_l^i = \mathcal{K}_l(C_l H_{l+1} \xi_l^i) \quad \text{and} \quad \mathcal{P}_l^i = \mathcal{K}_l(\hat{M}_0 H_{l+1} \xi_l^i).$$

Then, the iterative step in Algorithm 4.2 can be written as follows:

$$\begin{aligned} \mathcal{Y}_l^i &= \mathcal{Y}_l^{i-1} - \gamma_i \mathcal{Z}_l^{i-1}, \\ \mathcal{E}_l^i &= \mathcal{E}_l^{i-1} - \gamma_i (P_l^{i-1} - \mathcal{Z}_l^{i-1}) \end{aligned} \tag{4.43}$$

where $\mathcal{Y}_l^i = \mathcal{K}(y_l^i)$. The arithmetical complexity of (4.43) is $O(n_l)$.

3. The third step in Algorithm 4.2 is called the subspace leaving. Indeed, the solution vector $\hat{M}_0 H_l g_{l-1} \in V_{l-1}$ while iterations were performed in space V_l of a smaller dimension. Again, we have to consider two separate cases: $l = 0$ and $l > 0$.

(a) Let $l = 0$. Then

$$H_l g_{l-1} = \hat{M}_0^{-1}\left(\sum_{i=1}^{n_0} \mathcal{N}_{0,i} M_{0,i} w_{0,i} \mathcal{Y}_{0,i}^{\beta_0} + g_{l-1}\right).$$

The arithmetical complexity of this step is $O(n_0)$.

(b) Let $l > 0$. Given $y_l^{\beta_l} \in V_l$ and $g_{l-1} \in V_{l-1}$, we have to compute vectors $\mathcal{P}_{l-1} = \mathcal{K}_{l-1}(y_l^{\beta_l} + g_{l-1})$ and $\mathcal{Z}_{l-1} = \mathcal{K}_{l-1}(C_{l-1}\hat{M}_0^{-1}(y_l^{\beta_l} + g_{l-1}))$. Let $p_{l-1} = \mathcal{K}_{l-1}^{-1}(\mathcal{P}_{l-1})$ and $z_{l-1} = \mathcal{K}_{l-1}^{-1}(\mathcal{Z}_{l-1})$. Then

$$\begin{aligned}
p_{l-1} &= \sum_{i=1}^{n_{l-1}} \mathcal{N}_{l-1,i} M_{l-1,i} w_{l-1,i} \mathcal{G}_{l-1,i} + \sum_{i=1}^{n_l} \mathcal{N}_{l,i} M_{l,i} w_{l,i} \mathcal{Y}_{l,i}^{\beta_l} \\
&= \sum_{i=1}^{n_{l-1}} \mathcal{N}_{l-1,i} M_{l-1,i} w_{l-1,i} \mathcal{G}_{l-1,i} \\
&\quad + \sum_{i=1}^{n_l} \mathcal{N}_{l,i}\left(\sum_{j\in\mathfrak{S}_l(i)} \hat{\mathcal{N}}_{l-1,j} M_{l-1,j} \hat{\mathcal{N}}_{l-1,j}\right) w_{l,i} \mathcal{Y}_{l,i}^{\beta_l} \\
&= \sum_{i=1}^{n_{l-1}} \mathcal{N}_{l-1,i} M_{l-1,i} w_{l-1,i} \mathcal{G}_{l-1,i} + \sum_{i=1}^{n_{l-1}} \mathcal{N}_{l-1,i} M_{l-1,i} w_{l-1,i} \frac{\alpha_{l,j}}{\alpha_{l-1,i}} \mathcal{Y}_{l,j}^{\beta_l}.
\end{aligned}$$

The definition of the bijective mapping $\mathcal{K}_{l-1}$ gives

$$\mathcal{P}_{l-1,i} = \mathcal{G}_{l-1,i} + \frac{\alpha_{l,j}}{\alpha_{l-1,i}} \mathcal{Y}_{l,j}^{\beta_l}.$$

The computation of $\mathcal{Z}_{l-1}$ is associated with a matrix-vector multiplication. Indeed,

$$\begin{aligned}
z_{l-1} &= C_{l-1}\hat{M}_0^{-1} p_{l-1} \\
&= \sum_{i=1}^{n_{l-1}} \mathcal{N}_{l-1,i} M_{l-1,i} w_{l-1,i} w_{l-1,i}^T M_{l-1,i} \mathcal{N}_{l-1,i}^T \hat{M}_0^{-1} \\
&\qquad \sum_{j=1}^{n_{l-1}} \mathcal{N}_{l-1,j} M_{l-1,j} w_{l-1,j} \mathcal{P}_{l-1,j}
\end{aligned}$$

and

$$\mathcal{Z}_{l-1,i} = \sum_{j\in\mathfrak{O}_{l-1}(i)} \Upsilon_{ij}^{l-1} \mathcal{P}_{l-1,j}. \tag{4.44}$$

The arithmetical complexity of computing $\mathcal{P}_{l-1}$ and $\mathcal{Z}_{l-1}$ is obviously $O(n_{l-1})$.

4. Let us consider recursive algorithms for computing coefficients Υ^l_{ij} and $\alpha_{l,i}$. We recall that $w_{l,i} = \alpha_{l,i}\varepsilon_l$, $\varepsilon_l = (1,\dots,1)^T$, and $w^T_{l,i} M_{l,i} w_{l,i} = 1$. Then, Definition 4.1 leads to the following recursion:

$$\alpha_{l,i} = \frac{1}{\sqrt{\varepsilon_l^T M_{l,i}\varepsilon_l}} = \frac{1}{\sqrt{\varepsilon_l^T \left(\sum_{j\in\mathfrak{S}_l(i)} \hat{\mathcal{N}}_{l-1,j} M_{l-1,j} \hat{\mathcal{N}}^T_{l-1,j} \right) \varepsilon_l}} = \frac{1}{\sqrt{\sum_{j\in\mathfrak{S}_l(i)} \alpha^{-2}_{l-1,j}}}. \tag{4.45}$$

The coefficients Υ^0_{ij} are computed directly for $O(n_0)$ arithmetical operations. Combining Definition 4.1 with (4.41) we get a recursive procedure:

$$\begin{aligned}
\Upsilon^l_{kj} &= w^T_{l,k} M_{l,k} \mathcal{N}^T_{l,k} \hat{M}_0^{-1} \mathcal{N}_{l,j} M_{l,j} w_{l,j} \\
&= w^T_{l,k} \left[\sum_{r\in\mathfrak{S}_l(k)} \hat{\mathcal{N}}_{l-1,r} M_{l-1,r} \hat{\mathcal{N}}^T_{l-1,r} \right] \mathcal{N}^T_{l,k} \hat{M}_0^{-1} \mathcal{N}_{l,j} \left[\sum_{t\in\mathfrak{S}_l(j)} \hat{\mathcal{N}}_{l-1,t} M_{l-1,t} \hat{\mathcal{N}}^T_{l-1,t} \right] w_{l,j} \\
&= \left[\sum_{r\in\mathfrak{S}_l(k)} \frac{\alpha_{l,k}}{\alpha_{l-1,r}} w^T_{l-1,r} M_{l-1,r} \mathcal{N}^T_{l-1,r} \right] \hat{M}_0^{-1} \left[\sum_{t\in\mathfrak{S}_l(j)} \frac{\alpha_{l,j}}{\alpha_{l-1,t}} \mathcal{N}_{l-1,t} M_{l-1,t} w_{l-1,t} \right] \\
&= \sum_{r\in\mathfrak{S}_l(k)} \sum_{t\in\mathfrak{S}_l(j)} \frac{\alpha_{l,k}}{\alpha_{l-1,r}} \frac{\alpha_{l,j}}{\alpha_{l-1,t}} \Upsilon^{l-1}_{rt}.
\end{aligned} \tag{4.46}$$

The arithmetical complexity of algorithms (4.45) and (4.46) is $O(\sum_{l=1}^L n_l)$. Assuming that the numbers $n_1,\dots,n_L$ form a decreasing geometric series, we get an upper bound $O(n_1)$ for the arithmetical complexity. Summarizing the estimates above we get the following result.

Lemma 4.7 *The arithmetical complexity of $B^{-1}g$ is proportional to*

$$n_\lambda + \sum_{l=0}^{L-1} \left(\prod_{i=0}^{l} \beta_i \right) n_l + \left(\prod_{i=0}^{L-1} \beta_i \right) W(H_L)$$

where $W(H_L)$ is the arithmetical complexity of the coarse grid solver.

Remark 4.2 *Let the partitions $\mathcal{T}_{l,h}$ be as in Remark 4.1. Then, $n_l \approx 8n_{l-1}$ and any $\beta \le 7$ leads to a multilevel preconditioner with the optimal arithmetical complexity.*

4.3.3 Numerical experiments

In this section we study the multilevel preconditioner B for the case of cubic/prismatic meshes and strongly varying coefficients. A direct solver is used as the coarse grid solver H_L. We use a W-cycle with at most β inner Chebyshev iterations.

Let us consider problem (4.7). Let Ω be a parallelepiped with edges of length $l_x = 50$, $l_y = 100$, and $l_z = 5$. Furthermore, let $S_f = -1.43\text{e-}4$, $c_{pp} = 2.25\text{e-}7$, and $\boldsymbol{K} = \{K_{ij}\}_{i,j=1}^{3}$. We impose the Dirichlet boundary condition $p = 300$ on the top side of $\partial\Omega$ and the Neumann boundary condition $\boldsymbol{f} \cdot \boldsymbol{n} = f_N$ of the rest of $\partial\Omega$,

$$f_N = \begin{cases} -\dfrac{30}{K_{33}} \cos\left(\dfrac{8\pi x}{l_x}\right) \cos\left(\dfrac{6\pi y}{l_y}\right) & \text{when} \quad z = 0, \\ 0 & \text{otherwise.} \end{cases}$$

This problem simulates a temperature diffusion in a soil.

The numerical experiments have been done on a SGI workstation with R10000 / R10010 (195 MHz) processor. In all experiments the PCG iterations were stopped when the Euclidean norm of the initial residual has been decreased by a factor 10^6. The initial guess was always zero.

In the first experiment we take a scalar tensor $\boldsymbol{K} = 100\boldsymbol{I}$. In the second experiment we replace it by a diagonal tensor $\boldsymbol{K} = \mathrm{diag}\{100,\, 100,\, 10\}$. In the third experiment we take a piecewise constant diffusion tensor $\boldsymbol{K}(\boldsymbol{x}) = \mathrm{diag}\{100,\, 100,\, 10\}\psi(\boldsymbol{x})$ where $\psi(\boldsymbol{x})$ is a random factor between 1 and 10^3 constant in each mesh prism. In the forth experiment we change a direction of anisotropy by taking $\boldsymbol{K}(\boldsymbol{x}) = \mathrm{diag}\{10,\, 10,\, 100\}\psi(\boldsymbol{x})$. The computational results are shown in Table 4.3.

The numerical experiments show that $\mathrm{Cond}(B^{-1}A)$ is independent of random jumps in the diffusion tensor. Note that the effective anisotropy of the problem is a product of the mesh and tensor anisotropies. It is maximal in the fourth experiment where the number of PCG iterations is twice the number in the other experiments. Finally, the size of the coarse grid problem was the same in all experiments. However, we hardly observe any dependence on the number of multigrid levels. Therefore, the

Grid	Experiment 1	Experiment 2	Experiment 3	Experiment 4
8x8x4	18	19	21	36
16x16x8	19	20	21	38
32x32x16	22	20	22	39
64x64x32	22	20	22	40

Table 4.3: The number of PCG iterations for multilevel preconditioner B with $\beta = 3$.

multilevel preconditioner B is an excellent choice for the matrix B_λ in the block diagonal preconditioner (4.5).