MULTILEVEL PRECONDITIONERS AND THEIR APPLICATIONS IN GEOSCIENCE

A Dissertation

Presented to

the Faculty of the Department of Mathematics

University of Houston

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

By

Andrey Prokopenko

May 2011

MULTILEVEL PRECONDITIONERS AND THEIR APPLICATIONS IN GEOSCIENCE

Andrey Prokopenko

APPROVED:

Dr. Yuri Kuznetsov, Chairman

Dr. Jiwen He,

Dr. Tsorng-Whay Pan,

Dr. Serguei Maliassov
ExxonMobil URC

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my advisor, Professor Yuri Kuznetsov. He generously invited me to the University of Houston, and for years guided my research activities. Without his encouragement and enthusiasm this dissertation would not have been completed.

My special thanks go to Dr. Yuri Vassilevsky for sparking my interest in the field of numerical methods for partial differential equations and his guidance during my first years in the field.

I would like to thank the Exxon Mobil Corporation for its generous support of the research project which formed a substantial part of this dissertation, and, in particular, I am grateful to Dr. Serguei Maliassov for a number of valuable remarks and suggestions.

I would also like to thank the members of my committee, Drs. Jiwen He and Tsorng-Way Pan, who found time to read the manuscript and made many important comments.

During my first years at the University of Houston, my colleagues and friends Oleg Boiarkine and Nikolay Yavich were a big help to me. I am also thankful to my friend Eugene Kikinzon for his reading this manuscript and helping improve its quality.

Finally, I want to thank my parents and my sister, who inspired and encouraged me during this long project.

MULTILEVEL PRECONDITIONERS AND THEIR APPLICATIONS IN GEOSCIENCE

An Abstract of a Dissertation

Presented to

the Faculty of the Department of Mathematics

University of Houston

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

By

Andrey Prokopenko

May 2011

ABSTRACT

In this dissertation, we develop new approaches to the solution of problems stemming from discretizations of the diffusion-type equations with anisotropic, discontinuous coefficients. Both new preconditioners, and new discretization technique, are developed.

For the diffusion equation on geometrically simple meshes, discretized by the finite volume method, we construct a preconditioner based on a presentation of the graph of the system matrix in an assembling form. This form is used to design the coarsening procedure. The multilevel framework is based on the inner Chebyshev iterations. We show the performance of the preconditioner for a model problem and compare it with that of another algebraic multigrid preconditioner. The theoretical statement, that the condition number of the preconditioned system does not depend on the heterogeneity of the problem, is supported by the numerical results.

For more complex meshes, we design a discretization technique based on piecewise constant approximation to the fluxes inside each mesh cell. This method is oriented on typical meshes for reservoir simulation, including such features as pinchouts and faults. It is shown to be sufficiently accurate for desired problems. It also allows construction of an efficient preconditioner for arising algebraic systems.

Finally, we discuss a new type of preconditioner for unsymmetric M-matrices. It is based on the theory of weak regular splittings and nested iterations. The numerical results show its robustness and good performance.

Contents

List of Figures

Chapter 1

Introduction

Designing efficient solvers for elliptic partial differential equations (PDE) is a challenging task, and has been a major research topic since 1950s. Any significant achievements in this area are vital to a wide set of real-world applications, such as radiation transport, reservoir and underground water flow simulation, and many others.

For example, solving diffusion equation is a core part of giving accurate description of the multiphase flow through a porous medium. While being a difficult problem to solve, it becomes even harder for typical media, which is usually highly heterogeneous (see *e.g* model [61] with 10^{-7} to 10^{5} range of coefficients). The resulting algebraic systems from most discretization methods are extremely ill-conditioned.

The purpose of this dissertation is two-faced. First, it is to design, analyze, and test new efficient preconditioners for algebraic problems arising from the conventional discretization (for instance, finite volume method) of a diffusion type equation with anisotropic discontinuous coefficients. Second, it is to develop a new discretization method so that a good preconditioner for a resulting system is constructed naturally.

1.1 Review of solution methods

Every discretization scheme (finite elements, finite differences, finite volumes) of the convection-diffusion equation leads to an algebraic system with a sparse matrix. In many cases, this matrix is symmetric and positive definite, or positive semi-definite. Often, it is an M-matrix. In each case, producing a solution efficiently on a fine mesh is a challenging task.

The demands of the users of the engineering applications result in systems with tens or hundreds of millions of unknowns. Standard direct methods, while being efficient for small systems, are usually considered as inappropriately slow for these systems. On the other hand, the convergence of unpreconditioned iterative solvers is also slow, due to large condition numbers of matrices, arising from coefficient heterogeneity, coefficient anisotropy, or mesh anisotropy.

Discovery of preconditioners lead to significant improvements in convergence rates. Classical preconditioners, such as Jacobi, Gauss-Seidel, SOR, and SSOR (see *e.g.*, [68]) are effective for a number of simple problems. However, they do not have the efficiency required by current applications. Their main drawback is not being numerically scalable, i.e. the computational work is not linear with respect to the number of unknowns. A combination of these methods with nested iterations was disscussed in [51].

The development of multigrid methods [31, 32, 6, 5, 15, 16] in the 1960s provided a solution to this problem, as such methods, under some restrictions, are numerically scalable. Historically, their development was associated with the tight connection to the model geometry; specifically, to mesh grid. Geometric multigrids operated on a hierarchy of meshes, constructed *a priori* by coarsening of the discretization grid. The increase in complexity of the problem grids slowed down the development of such methods, and lead to a different approach: algebraic multi-

grids.

The algebraic multigrid (AMG) methods use coefficient matrix, instead of the discretization grid, to guide the coarsening procedure. The introductory articles of 1980s [62, 35, 4] lead to an enormous research field. An important feature of many such methods is that they can be used as a black-box algorithm, i.e. the only input for the coarsening procedures is the coefficient matrix. One such preconditioner was proposed by K.Stüben and his collaborators [62, 56, 20, 63]. One of its versions, `amg1r5` [56], was available to public. It can be used for any symmetric positive semi-definite system, and sometime it converges even for unsymmetric systems. However, a few drawbacks of this code should be mentioned. The code requires huge memory allocations. The setup time for large 3D problems usually increases significantly. The code may stagnate on geometrically anisotropic problems [20]. Later versions of the algorithm, `RAMG05` and `SAMG`, are free of these drawbacks [63].

A different algebraic multigrid was proposed by Kuznetsov [35, 37]. The main features of this preconditioner is its spectral equivalence to the system matrix, and the linear increase of the computational work with respect to the number of unknowns. However, this preconditioner uses the knowledge of the mesh grid. In Chapter 2, we extend its multilevel framework to general systems with symmetric M-matrices with strict diagonal domination; particularly, for matrices arising from the discretizations of the diffusion equation with heterogeneous coefficients.

A quite similar approach (referred as algebraic multilevel iteration, AMLI) with an inner Chebyshev iterative procedure was developed by Axelsson and Vassilevski [4] and then extended to anisotropic problems [53]. The latter approach is headed at the P_1 finite element discretization on uniform triangular meshes. In contrast to our approach (see Chapter 2), these approaches do not consider non-uniform meshes.

There are very few preconditioners for discretized diffusion equation on meshes

with faults. An example of such preconditioner is discussed in [49].

1.2 Review of approximation methods for the diffusion equation

There are many discretization methods developed for second order diffusion equation. The list includes Finite Differences (FD), Mimetic Finite Differences, Finite Volume (FV), Finite Element (FE), Mixed Finite Element (MFE), and Mixed Hybrid Finite Element (MHFE).

The idea behind the FD method is to approximate all derivatives with finite differences. Usually, it is used for uniform rectangular grids. The main advantage of the method is its simplicity. However, it has many drawbacks, such as restriction to simple geometries and implementation of the boundary conditions, especially for domains with curved boundaries. For the complete presentation of the FD method we refer to [59].

The Finite Volume (FV) method is an approximation method leading to locally conservative schemes. It is a Petrov-Galerkin type method (the solution space is different from the test space, as the test space is defined on a dual, so called Voronoi, mesh). For further information we refer to [26].

Mimetic Finite Differences method is based on the support operator approach, see [30, 54]. The discrete operators are constructed to preserve main physical properties of the original differential operator, such as conservation law, solution symmetries and so on. For the linear diffusion problem it preserves symmetry between the discrete gradient and divergence operators. It also preserves the null spaces of the above operators and guaranties the stability of the discretization.

FE methods are currently among the most popular methods in modern numerical mathematics. The paper by Courant [24] is considered to be the one of the first

on the subject. The term "finite element method" was proposed by R.W. Clough in [23].

The FE method is based on the concept of a weak solution of a PDE, i.e. the solution of the variational problem. The existence and uniqueness of the solution is proved by using the properties of certain Hilbert spaces. The main advantage of the FE method over many others is that it is geometry free and can be applied to complex shape domains. For instance, the application of the FE method to domains with curved boundaries is investigated in [7, 10].

We use the term "Mixed Method" when we solve a problem with two or more physical variables. The mixed form for the second order diffusion-reaction equation can be formulated as

$$\begin{aligned} K^{-1}\boldsymbol{u} &+ \nabla p = 0 \quad \text{in} \quad \Omega, \\ \nabla \cdot \boldsymbol{u} &+ cp = f \quad \text{in} \quad \Omega. \end{aligned} \tag{1.1}$$

Here, the flux vector function $\boldsymbol{u}$ is introduced. The mixed formulation is used to compute flux $\boldsymbol{u}$ and pressure p simultaneously. The variational formulation for this problem involves a pair of Hilbert spaces, the space $\boldsymbol{V}$ for fluxes and the space Q for pressures. A finite element solution $(\boldsymbol{u}, p)$ belongs to the space $\boldsymbol{V}_h \times Q_h$, where $\boldsymbol{V}_h$ and Q_h are finite dimensional subspaces of $\boldsymbol{V}$ and Q, respectively.

In classical literature, spaces $\boldsymbol{V}_h$ were constructed for a number of "simple" cells, such as triangles and rectangles in 2D, and tetrahedra, prisms and rectangular parallelepipeds in 3D. The Raviart-Thomas spaces RT_m, spaces BDM_m are introduced and investigated in [11, 57].

Yu. Kuznetsov and S. Repin introduced a new approach to define the space of fluxes $\boldsymbol{V}_h$ in [47, 48] on general shape polygonal (2D) and polyhedral (3D) meshes. This method requires partitioning of a general polyhedral cell into "simpler shaped" cells. In [44], the method was extended to the mimetic finite difference method.

1.3 Dissertation outline

The dissertation is organized as follows. In Chapter 2, we present a multilevel preconditioner. In Section 2.1, we briefly introduce the PDE we study, describe the finite volume discretization of the mesh, and analyze the properties of the resulting matrices. Section 2.2 presents a multilevel framework based on inner Chebyshev iterations. The framework makes use of spectrum estimates of a two-level preconditioner, described in Section 2.3, and provides an estimate for the condition number of the preconditioned system. Section 2.4 presents a set of numerical experiments. We compare our preconditioner with the Stüben's algebraic multigrid (AMG) preconditioner. Obtained numerical results comply with theoretical statements. Specifically, the condition number of the preconditioned system receives no impact from a diffusion tensor anisotropy.

Chapter 3 is devoted to a new discretization method and its natural preconditioner. In Section 3.1, we discuss the problem formulation and introduce the meshes used in typical applications. We show that the presence of pinchouts and faults leads to a number of unusual mesh cells, such as fault cells. In Section 3.2, we provide a new discretization method for the described meshes. Significant part of that Section is devoted to a discretization of a fault cell. The new method is based on piecewise constant functions. Section 3.3 is devoted to a preconditioner for such discretizations and its properties. Finally, in Section 3.4, we introduce a technique for a specific class of meshes, leading to cell-centered discretization.

In Chapter 4, we discuss a multilevel preconditioner for unsymmetric M-matrices. It is based on the theory of weak regular splittings and nested iterations, presented in Section 4.2 in the multilevel setting. The design of a two-level preconditioner is shown in Section 4.3. Section 4.4 provides numerical results for a few generated test cases.

Chapter 2

Preconditioner for symmetric M-matrices

In this Chapter, we present a new multilevel preconditioner for the iterative solution of algebraic problems arising in the reservoir simulation.

More precisely, we study 3D diffusion problem with heterogeneous anisotropic coefficients.

Essential features of the preconditioner are as follows:

- The condition number of the preconditioned algebraic problem is $O(1)$, i.e. the number of iterations performed by an iterative solver is independent of the mesh step size and the parameters of the differential problem.
- The computational cost to invert the proposed preconditioner is linear with respect to the problem size.
- It can be used as a black-box solver.

2.1 Problem formulation

2.1.1 Differential formulation

Let us consider the homogeneous Neumann boundary value problem for the diffusion equation

$$\begin{aligned} -\nabla \cdot (K \nabla p) + cp &= f \qquad \text{in} \quad \Omega, \\ (K \nabla p) \cdot \boldsymbol{n} &= 0 \qquad \text{on} \quad \partial\Omega, \end{aligned} \tag{2.1}$$

where p is an unknown scalar function (pressure), $K = K(x) \in \mathbb{R}^{3\times 3}$ is a diffusion tensor, c is a positive function, f is a source function, Ω is a domain in $\mathbb{R}^3$, $\partial\Omega$ is the boundary of Ω, and $\boldsymbol{n}$ is the unit outward normal to $\partial\Omega$.

We assume that Ω is presented as a union of parallelepipedal cells e_i, i.e. we define the mesh domain Ω_h by

$$\Omega_h = \bigcup_{i=1}^{n} e_i.$$

We also assume that K is a piecewise constant diagonal tensor, and c is a piecewise constant function in Ω, i.e.

$$K = K_i \equiv \begin{pmatrix} K_{x,i} & & \\ & K_{y,i} & \\ & & K_{z,i} \end{pmatrix} \quad \text{in} \quad e_i, \quad i = \overline{1,n}.$$

and

$$c = c_i \equiv \text{const}_i \quad \text{in} \quad e_i, \quad i = \overline{1,n}.$$

2.1.2 Finite volume discretization

We replace (2.1) by an equivalent first order system

$$\begin{aligned} K^{-1}\boldsymbol{u} + \nabla p &= 0 \quad \text{in} \quad \Omega, \\ \nabla \cdot \boldsymbol{u} + cp &= f \quad \text{in} \quad \Omega, \\ \boldsymbol{u} \cdot \boldsymbol{n} &= 0 \quad \text{on} \quad \partial\Omega\ . \end{aligned} \tag{2.2}$$

This formulation is called the mixed formulation of (2.1). The first equation represents the Darcy flow law, and the second equation is the mass conservation law.

We integrate the mass conservation law over cell e_i, $1 \leq i \leq n$, and get

$$\sum_{j \in Adj(i)} \int_{\Gamma_{ij}} \boldsymbol{u} \cdot \boldsymbol{n} \, d\sigma + c_i \int_{e_i} p \, dx = \int_{e_i} f dx, \tag{2.3}$$

where $Adj(i)$ denotes the set of indexes of the mesh cells adjacent to e_i and Γ_{ij} denotes the interface between e_i and a neighboring cell e_j.

Let us introduce the following notations:

$$\begin{aligned} u_{ij} &= \frac{1}{|\Gamma_{ij}|} \int_{\Gamma_{ij}} \boldsymbol{u} \cdot \boldsymbol{n} \, d\sigma, \\ p_i &= \frac{1}{|e_i|} \int_{e_i} p \, dx, \\ f_i &= \frac{1}{|e_i|} \int_{e_i} f \, dx. \end{aligned} \tag{2.4}$$

Here, $|\Gamma_{ij}|$ denotes the area of the face Γ_{ij}, and $|e_i|$ is a volume of the cell.

Then, the discrete form of equation (2.3) (the discrete conservation law) can be written as

$$\sum_{j \in Adj(i)} u_{ij} |\Gamma_{ij}| \; + \; c_i p_i |e_i| = f_i |e_i|. \tag{2.5}$$

In the finite volume method the degrees of freedom u_{ij} for the flux variable $\boldsymbol{u}$ are approximated using the Darcy law in the following way:

$$u_{ij} = - \left(p_j - p_i \right) \left[\int_{C_i}^{C_j} \frac{d\sigma}{K_d} \right]^{-1}, \tag{2.6}$$

where C_i and C_j are centers of the cells e_i and e_j, respectively, and K_d is one of K_x, K_y or K_z depending on whether C_i and C_j lie on the same x, y, or z line.

Since, by our assumptions, diffusion tensor K is constant in each mesh cell, we evaluate the integral in (2.6) explicitly:

$$u_{ij} = -2\,(p_j - p_i)\left[\frac{h_{d,i}}{K_{d,i}} + \frac{h_{d,j}}{K_{d,j}}\right]^{-1}. \tag{2.7}$$

Substituting the fluxes in (2.5) with (2.7) we get the equation

$$\sum_{j \in Adj(i)} a_{ij}(p_i - p_j) + c_i p_i |e_i| = f_i |e_i|, \tag{2.8}$$

where

$$a_{ij} = 2\left[\frac{h_{d,i}}{K_{d,i}} + \frac{h_{d,j}}{K_{d,j}}\right]^{-1} |\Gamma_{ij}|. \tag{2.9}$$

Combining (2.8) for all cells e_i, $i = \overline{1,\ n}$, we obtain the finite volume system of linear equations

$$A\bar{p} = \bar{f}. \tag{2.10}$$

2.1.3 Matrix description

We decompose the system matrix A of (2.10) as

$$A = D + M, \tag{2.11}$$

where D is a diagonal matrix with positive elements $d_i = c_i|e_i|$ on the diagonal (this matrix corresponds to the reaction term cp in (2.1)), and $M = A - D$ is the remaining part. Using the equation (2.8) and the symmetricity of matrix A ($a_{ij} = a_{ji}$ due to (2.9)), we conclude that for any two cells e_i and e_j with the common interface Γ_{ij} there exists a corresponding component in M having the form $N_{ij}\, M_{ij}\, N_{ij}^T$, where

$$M_{ij} = a_{ij} \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}, \tag{2.12}$$

and N_{ij} is the corresponding assembling matrix.

Thus, we decompose M as

$$M = \sum_{k=1}^{N} N_k a_k \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} N_k^T \equiv \sum_{i,j} N_{ij} M_{ij} N_{ij}^T, \tag{2.13}$$

where a_k is a positive number, and N_k is assembling matrix corresponding to the link with index k, $k = \overline{1,\ N}$. Here, N is the total number of links between centers of the mesh cells.

Remark. The decomposition (2.13) is unique up to the order of components.

One can easily show that the matrix $M = \left(m_{ij}\right)$ of (2.13) has the following properties:

- M is symmetric;
- $m_{ii} > 0$ for any i;
- $m_{ij} \leq 0$ for any $j \neq i$;
- $\sum_j m_{ij} = 0$ for any i;
- M is positive semi-definite;
- $ker(M) = \text{span}\Big\{(1,\ \dots\ , 1)^T\Big\}$.

Therefore, matrix A is a Stieltjes matrix. As we will show later, the preconditioner is designed for matrices of this class, with finite volume discretization, described in Section 2.1.2, providing one such example.

2.2 Multilevel preconditioner

Let $t \geq 1$. Assume that we constructed a sequence of matrices

$$A \equiv A_0 \to B_0 \to A_1 \to B_1 \to \cdots \to B_{t-1} \to A_t \to B_t, \tag{2.14}$$

satisfying the following properties:

- $A_k = A_k^T > 0, \quad k = \overline{0,\ t}$;
- $B_k = B_k^T > 0, \quad k = \overline{0,\ t}$;
- B_k, $k = \overline{0,\ t-1}$, can be presented in the 2×2 block form

$$B_k = \begin{pmatrix} B_{k,11} & B_{k,12} \\ B_{k,21} & B_{k,22} \end{pmatrix} \tag{2.15}$$

  such that

$$B_{k,11} = A_{k+1} + B_{k,12} B_{k,22}^{-1} B_{k,21}.$$

  This block representation is equivalent to the multiplicative form

$$B_k = F_k^T \begin{pmatrix} A_{k+1} & 0 \\ 0 & B_{k,22} \end{pmatrix} F_k, \tag{2.16}$$

  with

$$F_k = \begin{pmatrix} I & 0 \\ B_{k,22}^{-1} B_{k,21} & I \end{pmatrix}.$$

- For each k, $k = \overline{0,\ t}$, we know the values of the positive constants α_k and β_k,

$$0 < \alpha_k \leq \beta_k,$$

  such that

$$\alpha_k B_k \leq A_k \leq \beta_k B_k. \tag{2.17}$$

Using this sequence of matrices we construct our preconditioner $\widehat{B}$ as follows. We start from the coarsest level $k = t$ and define

$$\widehat{B}_t = B_t.$$

From (2.17) we have

$$a_t \widehat{B}_t \le A_t \le b_t \widehat{B}_t$$

with $a_t \equiv \alpha_t$ and $b_t \equiv \beta_t$.

Then, for each k, $k = t-1, ..., 0$, we choose an integer $s_k \ge 1$, and construct matrix $\widehat{B}_k$ as

$$\widehat{B}_k = F_k^T \begin{pmatrix} \widehat{A}_{k+1} & 0 \\ 0 & B_{k,22} \end{pmatrix} F_k,$$

where

$$\widehat{A}_{k+1} = \left(\left[I \; - \; \prod_{j=1}^{s_k} (I \; - \; \tau_{k+1,j} \widehat{B}_{k+1}^{-1} A_{k+1}) \right] A_{k+1}^{-1} \right)^{-1}$$

is a symmetric positive definite matrix. The parameters $\tau_{k+1,j}$, $j = 1, \ldots, s_k$, are chosen in such a way that

$$p_{k+1}(z) = \prod_{j=1}^{s_k} \left(1 - \tau_{k+1,j} z\right)$$

is the least deviating from zero polynomial on the segment $[a_{k+1}; b_{k+1}]$. The solution to this problem is given in terms of Chebyshev polynomials (see [68]). The corresponding procedure is called a Chebyshev one. It is obvious that for given $\bar{\xi}$ the vector $\bar{\eta} = \widehat{A}_{k+1}^{-1} \bar{\xi}$ can be computed by the iterative procedure

$$\begin{aligned} \bar{\eta}_0 &= 0, \\ \bar{\eta}_j &= \bar{\eta}_{j-1} - \tau_{k+1,j} \widehat{B}_{k+1}^{-1} (A_{k+1} \bar{\eta}_{j-1} - \bar{\xi}), \quad j = 1, \ldots, s_k, \\ \bar{\eta} &= \bar{\eta}_{s_k}. \end{aligned} \tag{2.18}$$

The theory of Chebyshev methods implies (see [68]) that the eigenvalues of the matrix $\widehat{A}_{k+1}^{-1} A_{k+1}$ belong to the segment

$$\left[\frac{(1 - q_{k+1}^{s_k})^2}{1 + q_{k+1}^{2s_k}} \; ; \; \frac{(1 + q_{k+1}^{s_k})^2}{1 + q_{k+1}^{2s_k}} \right], \tag{2.19}$$

where

$$q_{k+1} = (\sqrt{\nu_{k+1}} - 1)/(\sqrt{\nu_{k+1}} + 1)$$

and

$$\nu_{k+1} = b_{k+1}/a_{k+1}.$$

Lemma. The eigenvalues of the matrix $\widehat{B}_k^{-1} A_k$ belong to the segment

$$[a_k;\ b_k] \equiv \left[\frac{(1 - q_{k+1}^{s_k})^2}{1 + q_{k+1}^{2s_k}} \alpha_k \ ;\ \frac{(1 + q_{k+1}^{s_k})^2}{1 + q_{k+1}^{2s_k}} \beta_k \right]. \tag{2.20}$$

We define $\widehat{B} = \widehat{B}_0$ to be our multilevel preconditioner for matrix A.

The construction of the sequence (2.14) is based on the fact that for a given matrix A, satisfying all the properties in Section 2.1.3, we are able to construct a symmetric positive definite matrix B, which has block 2×2 form of (2.15), such that corresponding matrix

$$A_1 = B_{11} - B_{12} B_{22}^{-1} B_{21}$$

has exactly the same properties as the original matrix A. We call B two-level preconditioner for matrix A, and A_1 coarse grid matrix. The exact algorithm for construction of B is described in the next Section.

2.3 Two-level preconditioner

As was pointed out in Section 2.2, in order to construct a sequence of matrices (2.14), it is sufficient to construct a two-level preconditioner for matrix A. We decompose matrix D from (2.11) using the same assembling matrices N_k as in decomposition (2.13)

$$D \ = \ \sum_{k=1}^{N} N_k \begin{pmatrix} d_{1k} & 0 \\ 0 & d_{2k} \end{pmatrix} N_k^T, \tag{2.21}$$

where d_{1k} are d_{2k} are some nonnegative constants. In contrast to the decomposition of M in (2.13), the decomposition (2.21) is not unique and we have some freedom to define coefficients d_{1k} and d_{2k}. The choice of d_{1k} and d_{2k} will be discussed later.

Using (2.21), the matrix A is presented in the assembling form

$$A = \sum_{k=1}^{N} N_k \left[\begin{pmatrix} d_{1k} & 0 \\ 0 & d_{2k} \end{pmatrix} + a_k \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} \right] N_k^T. \tag{2.22}$$

Our algorithm will use decomposition (2.22) for the construction of the preconditioner. The following lemma provides the necessary tool to achieve that.

Lemma. Let matrices A and B be defined by

$$A = \begin{pmatrix} d_1 & 0 \\ 0 & d_2 \end{pmatrix} + a \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$$

and

$$B = \begin{pmatrix} d_1 & 0 \\ 0 & d_2 \end{pmatrix} + \beta a \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix},$$

where β is some constant, $0 \le \beta \le 1$. Then, the following estimates hold:

$$\begin{array}{lcll} B & \le & A & \le \left(1 + \dfrac{d_1 + d_2}{d_1 d_2} a\right) B, \qquad \text{if} \quad \beta = 0, \quad d_1, d_2 > 0, \\ B & \le & A & \le \qquad \dfrac{1}{\beta} B, \qquad\qquad \text{otherwise.} \end{array} \tag{2.23}$$

For each index k in decomposition (2.22) we choose a coefficient $\beta_k \ge 0$ and define B by

$$B = \sum_{k=1}^{N} N_k \left[\begin{pmatrix} d_{1k} & 0 \\ 0 & d_{2k} \end{pmatrix} + \beta_k a_k \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} \right] N_k^T. \tag{2.24}$$

The matrix B has exactly the same properties as the original matrix A, i.e. it is symmetric and positive definite, and it can be assembled in the form similar to (2.11).

Now, let $\sigma \geq 1$ be some parameter. We choose $\beta_k = 0$ in (2.24) only for those submatrices when $d_{1k}, d_{2k} > 0$ and

$$1 + \frac{d_{1k} + d_{2k}}{d_{1k} d_{2k}} a_k \ \leq \ \sigma \ . \tag{2.25}$$

For the rest values of the index k we choose $\beta_k = 1/\sigma$. It can be proved that

$$B \ \leq \ A \ \leq \ \sigma B,$$

which provides parameters $\alpha = 1$ and $\beta = \sigma$ for the estimate (2.17).

We can look at choosing $\beta_k = 0$ as at the procedure of removing links from the graph of matrix A. After we choose the value β_k for all the links it may happen that for many nodes on the finite volume mesh all links with other nodes would be gone, i.e. the node would be isolated. It means that some rows in matrix B would contain only one nonzero element, namely, the element on the diagonal. Therefore, we partition the set of all nodes into two groups: in the first group, we put the nodes which have at least one link left with neighboring nodes, and in the second group we put the nodes without any remaining links. With respect to this partition, matrix B has the block 2×2 form

$$B = \begin{pmatrix} B_{11} & 0 \\ 0 & B_{22} \end{pmatrix}, \tag{2.26}$$

where the block B_{22} is a diagonal matrix. Matrix B in (2.26) has the form (2.15) with zero blocks B_{12} and B_{21}. Therefore, following Section 2.2 we define

$$A_1 = B_{11}.$$

Usually, the goal of multigrid methods is to quickly reduce the dimension of the matrix. The construction of our preconditioner is different from this approach in the way that the reduction of the matrix dimension is not as important as the reduction of the number of nonzero elements of the matrix. Our method depends

on Chebyshev iterations between levels. Therefore, the main computational cost of solving the system with the preconditioner is due to the multiplication of the level matrices by vectors. This cost is linear with respect to the the number of nonzero entries in the matrices A_k. We consider the following two algorithms.

Static algorithm

In the first approach, the coefficients $d_{1,k}$ and $d_{2,k}$ in the decomposition (2.21) are independent of the non-diagonal entries in A. In other words, the decomposition (2.21) is defined prior to the execution of the algorithm. We call this variant the static decomposition.

The implementation of this algorithm is as follows. Suppose that the center of each cell e_i has links with n_i centers of neighboring cells. Let k be the index corresponding to the link between cells e_i and e_j in assembling (2.22). Then, we take $d_{1k} = d_i/n_i$ and $d_{2k} = d_j/n_j$. For example, for the original matrix A we could take $n_i = 6$ for each cell and therefore $d_1 = d_i/6$ and $d_2 = d_j/6$. So, we choose $\beta_k = 0$ if

$$1 + \frac{6(c_i + c_j)}{c_i c_j} a_{ij} \leq \sigma.$$

Otherwise, we choose $\beta_k = 1/\sigma$.

The main advantage of this algorithm is in its simplicity, but its performance is worse than that of the algorithm described in the next section.

Dynamic algorithm

We call the second approach the dynamic algorithm. The name comes from the fact that it is impossible to determine coefficients d_{1k} and d_{2k} *a priori*. The determination of these coefficients is one of the procedures in the construction of the preconditioner. This decomposition depends on the coefficients of the matrix and desired properties of the preconditioner.

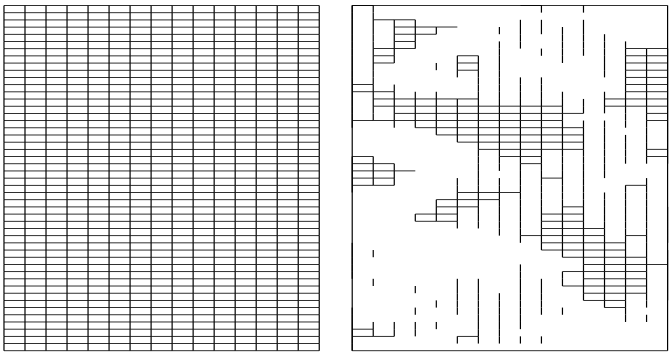

Figure 2.1: Graphs of the diagonal blocks of the matrices A (left) and A_1 (right) for a "horizontal" mesh plane

Let us discuss the implementation of this algorithm in brief. Suppose that we arrange all coefficients a_k in (2.13) in the increasing order. Then, during the implementation procedure starting with the lowest value of a_k we are trying to find the lowest values of d_{1k} and d_{2k} which satisfy inequality (2.25). If such values are found, we choose $\beta_k = 0$. Otherwise, we choose $\beta_k = 1/\sigma$.

Schur complement elimination of three-point links

The arithmetical complexity of the preconditioner highly depends on n, the dimension of A, and nnz, the number of nonzero entries of A. The procedure below allows significant decrease of both numbers.

Fig.2.1 shows the graphs of diagonal blocks of the matrices A_0 and A_1 which correspond to a "horizontal" mesh plane $z = const$. We notice that there are many nodes being connected to only two of its neighbors forming tridiagonal sequences. There are sequences of two types. For the first type, a sequence is bounded at both ends by a cluster formation. For the second type, a sequence starts with a node which has only one link. Examples of the tridiagonal sequences are shown in Figures 2.2 and 2.3.

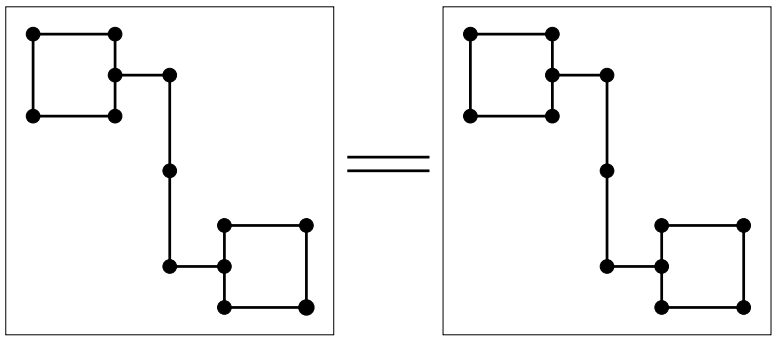

Figure 2.2: Tridiagonal sequence of the first type. No elimination

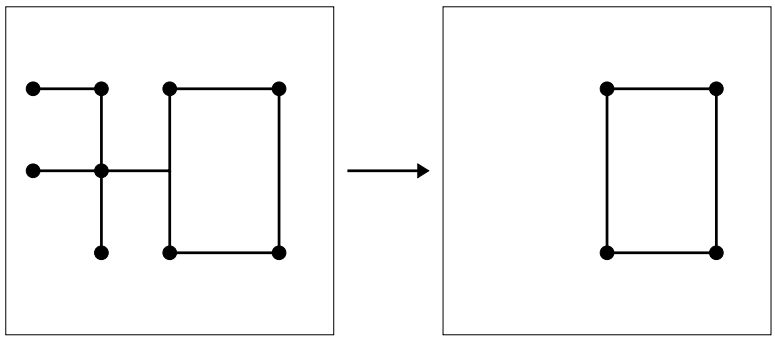

Figure 2.3: Tridiagonal sequence of the second type. Elimination procedure

Based on this graphical information, we can make an assumption that this is a regular phenomenon in the stencil of matrix B. The nodes in the second type of tridiagonal sequences can be eliminated by the Schur complement procedure. The computational cost of this elimination is linear with respect to the number of nodes in the sequence. We do not include the eliminated nodes into matrix B_{11}, which drastically reduces the numbers n and nnz. Therefore, the size of matrix A_1 can be drastically reduced by using another block partitioning. Namely, let matrix B be presented as

$$B = \begin{pmatrix} B_{11} & 0 \\ 0 & B_{22} \end{pmatrix} \equiv \begin{pmatrix} D_{11} & D_{12} & 0 \\ D_{21} & D_{22} & 0 \\ 0 & 0 & B_{22} \end{pmatrix},$$

where D_{22} is an easily invertible matrix corresponding to the tridiagonal sequences of the second type.

Then, we get the new block partitioning of matrix B by presenting it in another form

$$B = \begin{pmatrix} B_{\text{new},11} & B_{\text{new},12} \\ B_{\text{new},21} & B_{\text{new}22} \end{pmatrix}$$

with the blocks

$$B_{\text{new},11} = D_{11},$$

$$B_{\text{new},12} = \begin{pmatrix} D_{12} & 0 \end{pmatrix}, \quad B_{\text{new},21} = \begin{pmatrix} D_{21} \\ 0 \end{pmatrix},$$

and

$$B_{\text{new},22} = \begin{pmatrix} D_{22} & 0 \\ 0 & B_{\text{old},22} \end{pmatrix}.$$

The new matrix A_1 is the Schur complement for the matrix $B_{\text{new},22}$:

$$A_1 = D_{11} - D_{12} D_{22}^{-1} D_{21}.$$

Fig.2.4 shows the graphs of diagonal blocks of the matrices A_0, A_1 and A_2 which correspond to a "horizontal" mesh plane $z = const$ after the described procedure.

Remark. Due to the nature of this algorithm, the values of α and β do not change.

Comparison of approaches

In the Table 2.3, we present the dimension of matrices A_k (columns n) and the number of nonzero elements in A_k (columns nnz) for static, dynamic, and dynamic with Schur complement variants.

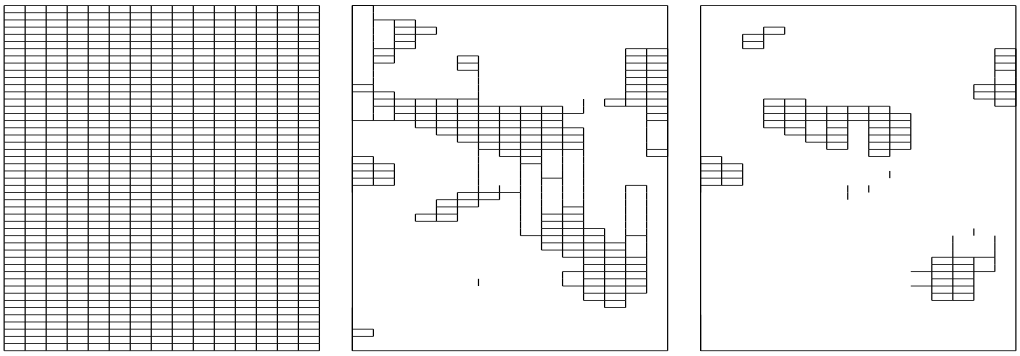

Figure 2.4: Graphs of the diagonal blocks of the matrices A_0 (left), A_1 (center) and A_2 (right) for a "horizontal" mesh plane

The dynamic variant performs much better than the static one, resulting not only in a drastic reduction in both values of n and nnz, but also in the reduction of the number of levels. Introducing additional Schur complement elimination of tridiagonal matrices further boosts the performance.

Table 2.1: The dimensions (n) and number of nonzero elements (nnz) of matrices A_k

Level	Static		Dynamic		Dynamic with Schur	
	n	nnz	n	nnz	n	nnz
0	1122000	7780000	1122000	7780000	1122000	7780000
1	604788	2680056	511104	1957400	338060	1476852
2	504061	2010591	360423	1205069	155890	653148
3	378493	1314309	212847	633211	54869	224197
4	235663	719407	98855	270315	15385	62391
5	116613	322269	36421	94081	3839	14947
6	46008	118872	10241	25091	376	1280
7	13496	32884	2169	4561	-	-
8	2953	6289	89	179	-	-
9	596	1216	-	-	-	-

2.4 Numerical results

In this Section, we present the numerical results for the SPE 10 model 2 benchmark [61]. We compare our method with the well known AMG preconditioner [63].

The key components of the data are:

- Ω is a rectangular parallelepiped of size $1200 \times 2200 \times 170$ (ft);
- Fine mesh is the uniform Cartesian $60 \times 220 \times 85$ grid;
- The medium is strongly heterogeneous, the diffusion coefficients vary in the interval $[10^{-7},\ 10^{5}]$;
- $K_x = K_y \equiv K_{xy}$.

The Figure 2.5 shows the distribution of the components K_{xy} and K_z of the diffusion tensor K.

Let us assume that the diffusion equation (2.1) comes from the discretization of the unsteady diffusion equation by the implicit finite difference method in time variable with the time step τ_{imp}. To this end, we assume that the coefficient c is a positive constant defined by the formula

$$c = \frac{1}{\tau_{imp}}. \tag{2.27}$$

For numerical experiments we choose

$$\tau_{imp} = \gamma\sqrt{\tau_{exp}}, \tag{2.28}$$

where γ is a positive factor ($\gamma \geq 1$), and

$$\tau_{exp} = \frac{1}{\|E^{-1}(A-D)\|_\infty} \tag{2.29}$$

with $E = diag\{|e_1|, ..., |e_n|\}$.

To solve the system (2.10), we use the PCG method with the preconditioner B, designed in Section 2.2. We use the typical stopping criteria for the PCG method:

$$\frac{\|A\bar{p}^k - f\|_2}{\|A\bar{p}^0 - f\|_2} \leq \epsilon,$$

where ϵ is a given accuracy. In the numerical experiments, we take $\epsilon = 10^{-6}$.

It is important for us to introduce a parameter which would show the effectiveness a preconditioner in terms of computational work. We choose this parameter to be the time of calculation of the residual vector. In our case, this time is equal to 0.05 seconds. It means, for instance, that if the time to invert our preconditioner once is 0.20 seconds, then its computation is equivalent to four evaluations of the residual vector for the original problem which is considered to be sufficiently cheap.

All experiments were performed with two Chebyshev iterations per level and $\sigma = 3$. In Tables 2.2 and 2.3 we give the CPU time and the number of iterations

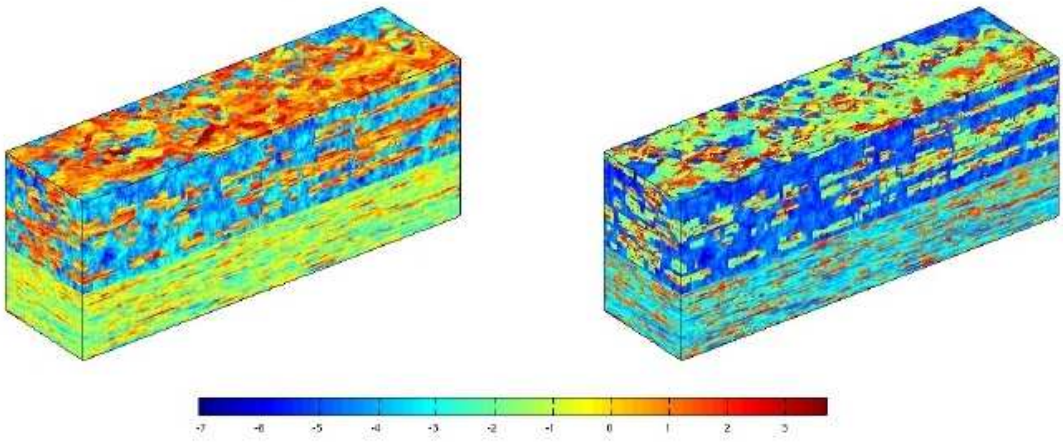

Figure 2.5: Distribution of the components of the diffusion tensor: K_{xy} (left) and K_z (right). Logarithmic scale

for the PCG method. We see that for sufficiently large values of the reaction term c our method performs faster than AMG because of cheap initialization time and quick inversion of the preconditioner, though it loses in number of PCG iterations. When we decrease the value of c, the number of links which can be removed also decreases, and so is the performance of the preconditioner.

Due to our knowledge of explicit spectral boundaries of our preconditioner in (2.20), it is also possible to use Chebyshev iterations as an outer iterative procedure. In Table 2.4 we compare the number of PCG and Chebyshev iterations. We see that they differ insignificantly. As Chebyshev method lack computation of inner products, this leads to a better parallel implementation.

Table 2.2: Numerical results for sufficiently large c

	$c = 1$ ($\gamma = 100$)		$c = 2$ ($\gamma = 50$)		$c = 4$ ($\gamma = 25$)	
	Our	AMG	Our	AMG	Our	AMG
Construction time (s)	3.17	13.22	2.83	12.39	2.46	11.65
B^{-1} time (s)	0.23	0.97	0.15	0.95	0.09	0.92
# of PCG iterations	16	7	15	7	15	7
Solution time (s)	5.25	7.45	3.72	7.30	2.94	7.10
Total time (s)	**8.42**	**20.67**	**6.55**	**19.69**	**5.40**	**18.75**

Table 2.3: Numerical results for small c

	$c = 0.05$ ($\gamma = 2000$)		$c = 0.1$ ($\gamma = 1000$)	
	Our	AMG	Our	AMG
Construction time (s)	5.29	14.60	4.71	14.65
B^{-1} time (s)	1.73	0.97	1.09	0.98
# of PCG iterations	17	8	17	8
Solution time (s)	30.73	8.45	20.11	8.54
Total time (s)	**36.02**	**23.05**	**24.82**	**23.19**

Table 2.4: Comparison of PCG and Chebyshev as external iterative method

	$c = 0.1$		$c = 1$		$c = 2$		$c = 4$	
	PCG	Cheb	PCG	Cheb	PCG	Cheb	PCG	Cheb
# of iterations	16	17	15	17	14	16	14	16

Chapter 3

Piecewise constant approximation

In this Section, we introduce an extension of the discretization method with piecewise constant (PWC) approximations for fluxes, introduced in [41], for the numerical solution of the diffusion equation on polyhedral meshes with possible pinchouts and faults. The research was motivated by applications in geoscience, namely, the reservoir simulation and basin modeling.

There are several possible methods of discretization for such meshes, including Kuznetsov-Repin ([47]) and mimetic ([44]) finite elements. While being accurate and robust, such methods usually have a significant drawback of high computational cost of the construction of the algebraic system. Additionally, these methods may provide insufficient accuracy for fluxes in strongly anisotropic mesh cells, which are typical for the described applications.

The proposed method has low cost of generation of the system while maintaining the accuracy similar to that of Kuznetsov-Repin elements.

3.1 Problem formulation

Similar to Section 2.1, let us consider the diffusion equation written in the mixed form,

$$\begin{aligned} K^{-1}\,\boldsymbol{u} \;+\; \nabla\, p \;&=\; 0, \\ \nabla\cdot\boldsymbol{u} \;+\; cp \;&=\; f, \end{aligned} \tag{3.1}$$

in a bounded domain $\Omega \subset \mathbb{R}^3$. We complement this equation with the boundary conditions

$$p \;=\; g_D \quad \text{on} \quad \Gamma_D \quad \text{and} \quad \boldsymbol{u}\cdot\boldsymbol{n} \;=\; g_N \quad \text{on} \quad \Gamma_N. \tag{3.2}$$

Here, Γ_D and Γ_N are parts of the boundary $\partial\Omega$ of Ω such that

$$\Gamma_D \;=\overline{\Gamma}_D, \quad \Gamma_D \bigcap \Gamma_N \;=\; \emptyset, \quad \Gamma_D \bigcup \overline{\Gamma}_N \;=\; \partial\Omega. \tag{3.3}$$

In the case of the Neumann problem with $c = 0$ and $\overline{\Gamma}_N = \partial\Omega$, in addition, we assume that the compatibility condition

$$\int\limits_{\partial\Omega} g_N \,\mathrm{ds} \;=\; \int\limits_{\Omega} f \,\mathrm{dx} \tag{3.4}$$

holds for functions g_N and f.

We assume that Ω is the union of several subdomains, i.e.

$$\overline{\Omega} \;=\; \bigcup_{s=1}^{S} \overline{\Omega}_s \tag{3.5}$$

and the reaction coefficient c is a piecewise constant function in Ω. Namely,

$$c \equiv c_s \quad \text{for all} \quad x \in \Omega_s, \tag{3.6}$$

where c_s is a nonnegative constant, $s = \overline{1,\ S}$.

3.1.1 Polyhedral meshes

In each subdomain Ω_s we construct a conforming polyhedral mesh $\Omega_{s,h}$

$$\Omega_{s,h} \;=\; \bigcup_{k=1}^{N_s} E_k^{(s)}, \tag{3.7}$$

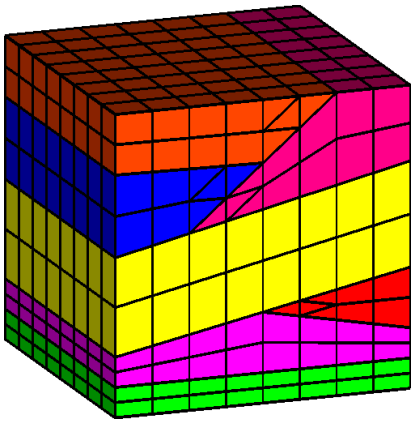

Figure 3.1: Distorted hexahedral mesh with pinchouts

consisting of (degenerated) hexahedrons. The resulting mesh Ω_h in Ω is the union of the subdomain meshes

$$\Omega_h = \bigcup_{s=1}^{S} \Omega_{s,h} = \bigcup_{s=1}^{S} \bigcup_{k=1}^{N_s} E_k^{(s)} \equiv \bigcup_{k=1}^{N} E_k. \tag{3.8}$$

We assume that Ω_h is a geometrically conforming mesh.

Figures 3.1.1 and 3.1.1 contain two examples of such meshes. In the first example, domain Ω is a unit cube divided into seven geological layers with two categories of mesh cells. Cells of first type are distorted hexahedrons, and the other cells are so called pinchout cells. Pinchouts happen when vertical edges of a hexahedron degenerate [52]. Different types of pinchout cells in Ω_h are given in the next section. In the second example, the domain Ω contains a fault. To construct a conforming mesh for such meshes, one performs an interface reconstruction algorithm. Such meshes contain hexahedrons and fault cells.

3.1.2 Description of mesh cells

We consider six types of cells: distorted hexahedron, distorted hexahedron with one degenerated edge (element H1), distorted hexahedron with two degenerated edges

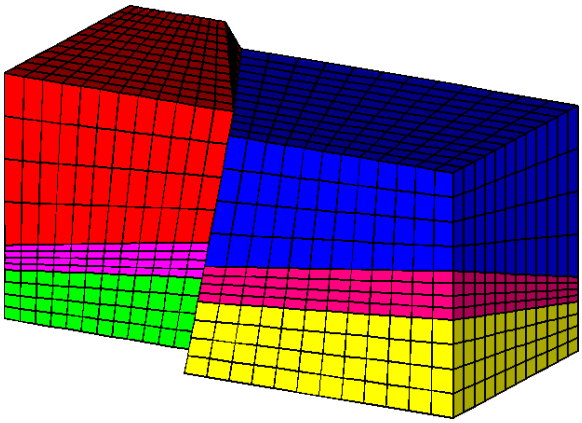

Figure 3.2: Distorted hexahedral mesh with a fault

(element H2 and horizontal prism), and a tent.

Distorted hexahedron.

Each distorted hexahedral cell is topologically equivalent to a cube as shown in Figure 3.3. It has eight vertices, six quadrilateral faces, and twelve edges.

Element H1

"Element H1" is obtained from a distorted hexahedron when one of its vertical edges degenerates. On Figure 3.4, element H1 is obtained from a regular hexahedron after the degeneration of the edge (0,4).

Element H1 has seven vertices, two triangular and four quadrilateral faces, and 11 edges.

Horizontal prism

Horizontal triangular prism is the second type of a pinchout cell. It occurs when two neighboring vertical edges degenerate. An example on Figure 3.5 shows a horizontal prism obtained from a regular hexahedron after the degeneration of

edges (0, 4) and (1, 5).

The horizontal prism has six vertices, and two triangular and three quadrilateral faces.

Element H2

"Element H2" appears when two opposite vertical edges of a regular hexahedron degenerate. The Figure 3.6 shows an element H2 obtained from a regular hexahedron with degenerated edges (1,5) and (3,7).

Element H2 has six vertices, and four triangular and two quadrilateral faces.

Tent

Tent is the last type of pinchout cell, which appears when three vertical edges degenerate. An example of such cell is shown on Figure 3.7.

Tent has five vertices, and two triangular and two quadrilateral faces.

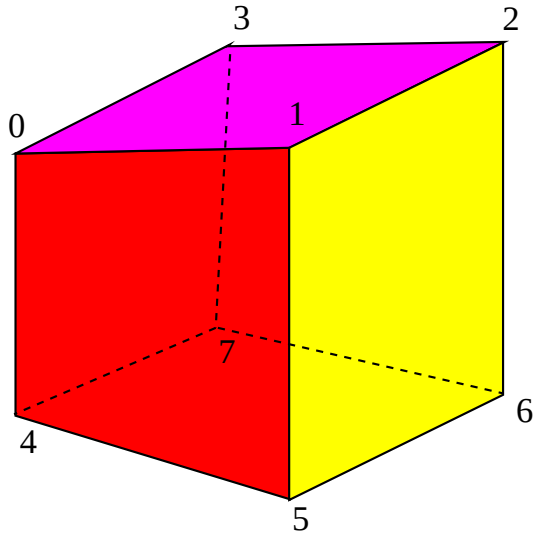

Figure 3.3: Distorted hexahedron

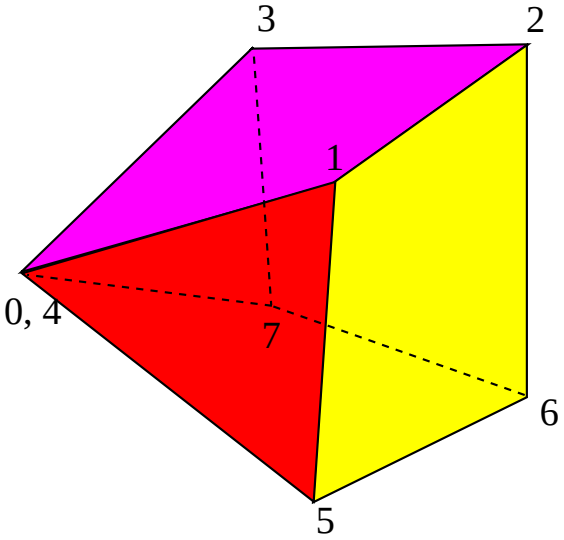

Figure 3.4: Element H1

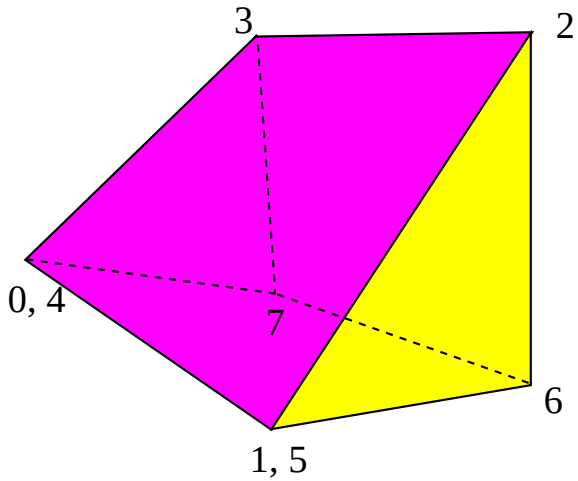

Figure 3.5: Horizontal triangular prism

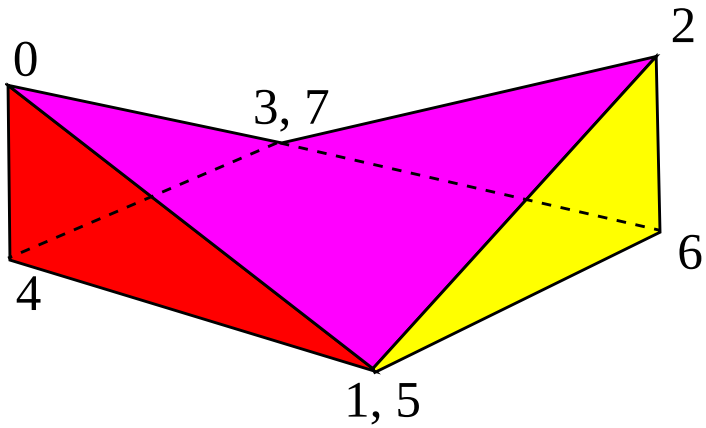

Figure 3.6: Element H2

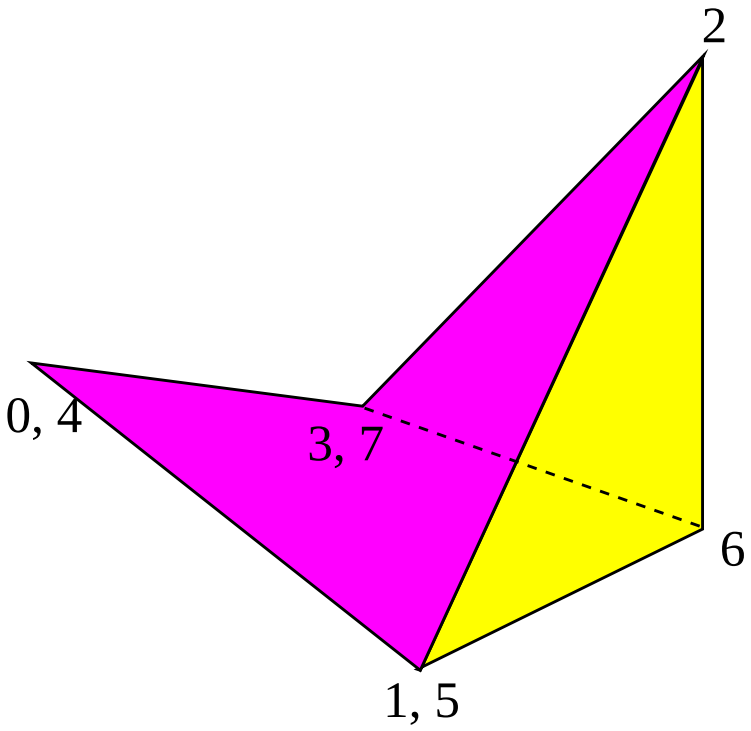

Figure 3.7: Tent

3.1.3 Discretization of the problem and algebraic system

In order to discretize the problem (3.1), (3.2), we apply the mixed FE method. Let $E \equiv E_k$ be a mesh cell with t faces/interfaces Γ_1, Γ_2, ..., Γ_t. We introduce meanvalues of the normal flux and the solution function by

$$u_s = \frac{1}{|\Gamma_s|} \int\limits_{\Gamma_s} \boldsymbol{u} \cdot \boldsymbol{n} \, \mathrm{ds}, \quad s = \overline{1,\ t}, \tag{3.9}$$

and

$$p_E \ = \ \frac{1}{|E|} \int\limits_{E} p \, \mathrm{dx}, \tag{3.10}$$

respectively.

We also introduce the meanvalue of the source function by

$$f_E \ = \ \frac{1}{|E|} \int\limits_{E} f \, \mathrm{dx}. \tag{3.11}$$

Then, the discrete form of the conservation law

$$\nabla \cdot \boldsymbol{u} \ = \ f \tag{3.12}$$

is

$$\int\limits_{E} \nabla \cdot \boldsymbol{u} \, \mathrm{dx} \ \equiv \ \sum_{s=1}^{t} u_s \, |\Gamma_s| \ = \ f_E \cdot |E| \ . \tag{3.13}$$

We replace the differential form of the Darcy law

$$K^{-1} \, \boldsymbol{u} \ + \ \nabla p \ = \ 0 \tag{3.14}$$

with the equivalent weak formulation

$$\int\limits_{E} K^{-1} \, \boldsymbol{u} \cdot \boldsymbol{v} \, \mathrm{dx} \ - \ \int\limits_{E} p \, (\nabla \cdot \boldsymbol{v}) \ \mathrm{dx} \ + \ \int\limits_{\partial E} p \ (\boldsymbol{v} \cdot \boldsymbol{n}) \ \mathrm{ds} \ = \ 0, \tag{3.15}$$

where $\boldsymbol{v} = \boldsymbol{v}(x)$ is a test vector-function.

Let $\boldsymbol{v}$ satisfy the following two conditions:

- $\boldsymbol{v} \cdot \boldsymbol{n} \;=\; v_s \;\equiv\; \mathrm{const}_s \quad \text{on} \quad \Gamma_s, \quad s = \overline{1,\ t};$

- $\nabla \cdot \boldsymbol{v} \;=\; \mathrm{const}_E \quad \text{in} \quad E.$

Then,

$$\int\limits_E p\,(\nabla \cdot \boldsymbol{v})\ \mathrm{dx} \;=\; p_E \cdot \sum_{s=1}^{t} v_s \cdot |\Gamma_s| \tag{3.16}$$

and

$$\int \partial E\, p\,(\boldsymbol{v} \cdot \boldsymbol{n})\ \mathrm{ds} \;=\; \sum_{s=1}^{t} \lambda_s \cdot v_s \cdot |\Gamma_s|\ . \tag{3.17}$$

Here,

$$\lambda_s \;=\; \frac{1}{|\Gamma_s|} \int\limits_{\Gamma_s} p\,\mathrm{ds}, \quad s = \overline{1,\ t}, \tag{3.18}$$

are meanvalues of the solution function on the interfaces.

Thus, the weak formulation of the Darcy law in cell E becomes:

$$\int\limits_E \left(K^{-1}\boldsymbol{u}\right) \cdot \boldsymbol{v}\,\mathrm{dx} \;-\; p_E \cdot \sum_{s=1}^{t} v_s \cdot |\Gamma_s| \;+\; \sum_{s=1}^{t} \lambda_s \cdot v_s \cdot |\Gamma_s| \;=\; 0\ . \tag{3.19}$$

Discretization of the integral term is the transition

$$\int_E \left(K^{-1}\boldsymbol{u}\right) \cdot \boldsymbol{v}\,\mathrm{dx} \quad \Rightarrow \quad (M_E \bar{u},\ \bar{v}) \tag{3.20}$$

where $\bar{u}$ and $\bar{v}$ are vectors in $\mathbb{R}^t$, and M_E is a symmetric positive definite matrix. The construction of this matrix is described in the next Section. As the result, when we discretize the Darcy law and the mass conservation law on each mesh cell $E \in \Omega_h$, and complement these equations by weak continuity equations on each interface between mesh cells, we obtain the system of linear algebraic equations in the form

$$A_{u,p,\lambda} \begin{pmatrix} \bar{u} \\ \bar{p} \\ \bar{\lambda} \end{pmatrix} \;=\; \begin{pmatrix} \bar{g}_D \\ \bar{f} \\ \bar{g}_N \end{pmatrix}, \tag{3.21}$$

with system matrix $A_{u,p,\lambda}$ having a three-by-three block structure

$$A \;=\; \begin{pmatrix} M & B^T & C^T \\ B & -\Sigma & 0 \\ C & 0 & 0 \end{pmatrix}, \tag{3.22}$$

where the mass matrix

$$M \;=\; \mathrm{diag}\Big\{ M_{E_1},\; \ldots\;,\; M_{E_N} \Big\} \tag{3.23}$$

is a block diagonal matrix with the blocks M_E.

We reduce the system by eliminating flux degrees of freedom from the consideration, resulting in

$$S \begin{pmatrix} \bar{p} \\ \bar{\lambda} \end{pmatrix} \;=\; \begin{pmatrix} \bar{f}_p \\ \bar{f}_\lambda \end{pmatrix}, \tag{3.24}$$

where the system matrix S is given by

$$S \;=\; \begin{pmatrix} \Sigma & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} B \\ C \end{pmatrix} M^{-1} \begin{pmatrix} B^T & C^T \end{pmatrix}. \tag{3.25}$$

The matrix S is symmetric. In the case of Neumann boundary conditions on the whole boundary and $c \equiv 0$, S is positive semidefinite, and its kernel is

$$\ker S \;=\; \mathrm{span}\big\{ \bar{e} \big\}, \quad \bar{e} \;=\; \begin{pmatrix} 1 & 1 & \ldots & 1 \end{pmatrix}^T. \tag{3.26}$$

Otherwise, S is positive definite.

Next, we apply a preconditioned iterative solver to solve system (3.24) in order to obtain a solution pair $(\bar{p},\; \bar{\lambda})$.

Finally, we express the DOF for fluxes by

$$\bar{u} \;=\; M^{-1} \Big(\bar{g}_D \;-\; B^T \bar{p} \;-\; C^T \bar{\lambda} \Big). \tag{3.27}$$

Remark. An alternative method, applicable for specific meshes, is presented in the Section 3.4.

3.2 Piecewise constant (PWC) flux approximation

In this Section, we present an approach for the approximation of the flux vector function. We construct mass matrices in the space of fluxes using piecewise constant vector fields. We call this approach PWC which is an acronym for "PieceWise Constant".

3.2.1 General algorithm for mass matrix construction

Let E be a polyhedral cell. Let us assume that there exists a decomposition

$$E = \bigcup_{l=1}^{N_E} e_l \tag{3.28}$$

into polyhedrons (possibly overlapping) such that:

- Each face Γ of e_l is either an inner face with respect to E, or it is a face of E;
- For each cell e_l there exists its vertex A of E such that there are exactly three faces (Γ_1, Γ_2 and Γ_3) of e_l adjacent to it, which are also the faces of E.

We will show later that for all types of cells described in Section 4.1 we can construct partition satisfying the above properties.

Now let e be one of the the cells e_l from partition (3.28). Let $\bar{v} \in \mathbb{R}^3$ be a vector of three degrees of freedom. We say that $\boldsymbol{v}_h \in \boldsymbol{V}_e^{(PWC)}$ if and only if the following two conditions hold:

- $\boldsymbol{v}_h \equiv \mathbf{const} \in \mathbb{R}^3$ in e;
- $\frac{1}{|\Gamma_i|} \int_{\Gamma_i} \boldsymbol{v}_h \cdot \boldsymbol{n} \, \mathrm{ds} \;=\; v_i, \quad i = \overline{1,\ 3}.$

Remark: The DOF v_i represents the average normal component $\boldsymbol{v}_h \cdot \boldsymbol{n}$ of $\boldsymbol{v}_h$ on face Γ_i, $i = \overline{1,\ 3}$.

Explicit formulas. Let

$$\Gamma_i \ = \ \bigcup_{j=1}^{m_i} \gamma_{ij} \tag{3.29}$$

be a triangulation of the face Γ_i. We denote by $\boldsymbol{n}_{ij}$ the unit outward normal vector to ∂e on triangle γ_{ij}. We define by

$$\boldsymbol{n}_i \ = \ \sum_{j=1}^{m_i} \frac{|\gamma_{ij}|}{|\Gamma_i|} \boldsymbol{n}_{ij} \tag{3.30}$$

the "effective outward normal vector" to e on Γ_i.

Remark. If Γ_i is planar then $\boldsymbol{n}_i$ is the outward unit normal vector to e on Γ_i. Otherwise, $\|\boldsymbol{n}_i\| < 1$.

Direct calculations show that these "effective normal vectors" uniquely determine a constant vector field $\boldsymbol{v}_h$ in e. Namely, the constant value of $\boldsymbol{v}_h$ in e is the vector

$$\boldsymbol{v} = N^{-T} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix}, \tag{3.31}$$

where

$$N = \begin{pmatrix} \boldsymbol{n}_1 \ \boldsymbol{n}_2 \ \boldsymbol{n}_3 \end{pmatrix} \tag{3.32}$$

is a three-by-three matrix which columns are the corresponding "effective" normals to sides of e.

In order to construct a mass matrix M, we first introduce a non-overlapping partitioning of E into polyhedral cells

$$E = \bigcup_{k=1}^{N_p} P_k, \tag{3.33}$$

such that each subcell e_l from the partitioning (3.28) is a union of several cells P_k.

Let us denote by n_k, $k = 1, ..., N_p$, the number of cells e_l containing P_k. We introduce the functions $\alpha_l(x)$, $l = 1, ..., N_E$, in the following form

$$\alpha_l(x) = \begin{cases} \dfrac{1}{n_k}, & \text{if } x \in P_k \bigcap e_l, \\ \\ 0, & \text{otherwise.} \end{cases} \tag{3.34}$$

Remark. Functions α_l form a unity partition on E, i.e.

$$\sum_{l=1}^{N_E} \alpha_l(x) \equiv 1. \tag{3.35}$$

Let $\bar{u}$ and $\bar{v}$ be vectors of degrees of freedom for E. We construct N_E piecewise constant vector fields $\boldsymbol{u}_h$ and $\boldsymbol{v}_h$ for each e_l according to the above procedure. Then we define the mass matrix M as

$$(Mu, v) = \sum_{l=1}^{N_l} \int_E \alpha_l(x)(a_E^{-1} u_h^l) \cdot v_h^l dx. \tag{3.36}$$

Direct calculations show that

$$M = \sum_{l=1}^{N_E} \mathcal{N}_l \left(\sum_{k: P_k \in e_l} \frac{|P_k|}{n_k} \right) N_l^{-1} K_E^{-1} N_l^{-T}, \mathcal{N}_l^T \tag{3.37}$$

where $\mathcal{N}_l$ are assembling matrices.

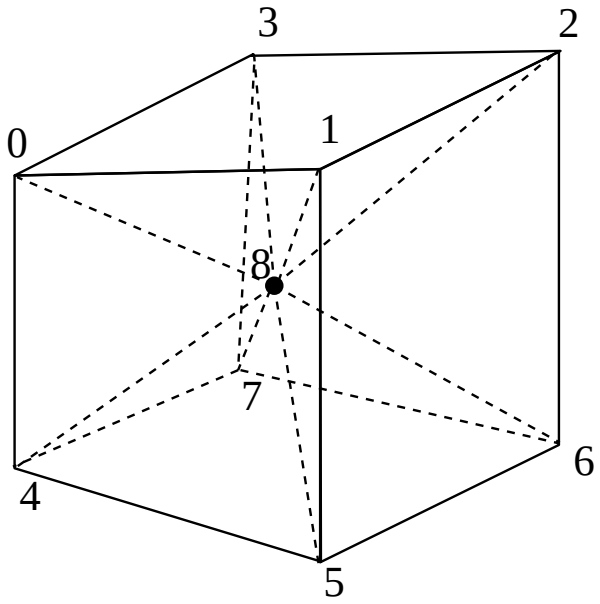

Figure 3.8: Partition of a distorted hexahedron into six pyramids

3.2.2 Mass matrices for regular and pinchout cells

Distorted hexahedron

Let E be a distorted hexahedron as shown in Figure 3.8. We denote by $\{\Gamma_i\}$, $i = \overline{1,\ 6}$, the set of quadrilateral faces of E. Without loss of generality, we may assume that the vertex 4 of E is a common vertex of the faces Γ_1, Γ_2, and Γ_3. Therefore, the vertex 2 is the common vertex of the faces Γ_4, Γ_5, and Γ_6.

Let vertex 8 be the center of E. We set P_i to be the pyramid with the base Γ_i, $i = \overline{1,\ 6}$, and vertex 8 as an apex, and define

$$e_1 = P_1 \bigcup P_2 \bigcup P_3$$

and

$$e_2 = P_4 \bigcup P_5 \bigcup P_6,$$

as shown on Figure 3.9.

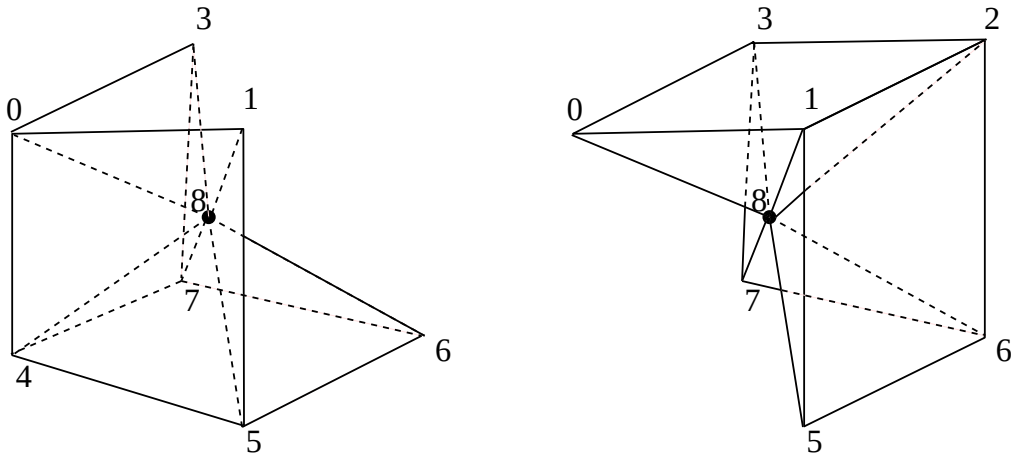

Figure 3.9: Subcells e_1 (left) and e_2 (right) for a hexahedron

The calculations show that the resulting mass matrix from (3.37) would have a block-diagonal form

$$M = \operatorname{diag}\left\{|e_1| N_1^{-1} a_E^{-1} N_1^{-T}, |e_2| N_2^{-1} a_E^{-1} N_2^{-T}\right\} \tag{3.38}$$

We would also like to note that in the case of a scalar tensor this mass matrix is easily invertible,

$$M^{-1} = a_E \operatorname{diag}\left\{\frac{1}{|e_1|} N_1^T N_1, \frac{1}{|e_2|} N_2^T N_2\right\}, \tag{3.39}$$

and is cheap to compute.

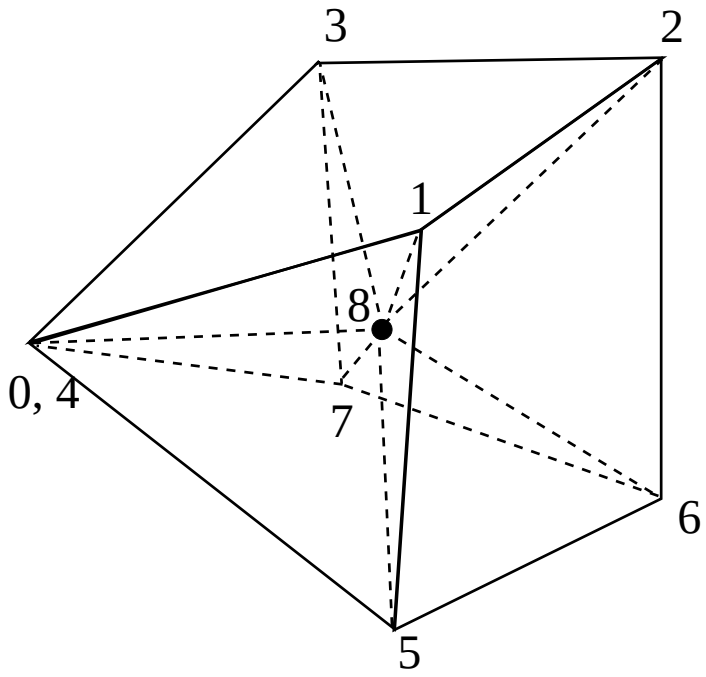

Figure 3.10: Partition of a H1 element into four pyramids and two tetrahedrons

Element H1

Let E be a H1 element as shown in Figure 3.10. We denote by $\{\Gamma_i\}$, $i = \overline{1,\ 6}$, the set of faces of E so that Γ_1, Γ_2, Γ_3, and Γ_4 are quadrilateral faces, and Γ_5 and Γ_6 are triangular faces. Without loss of generality, we may assume that the vertex 5 of E is a common vertex of the faces Γ_1, Γ_2, and Γ_5, and the vertex 3 is the common vertex of the faces Γ_3, Γ_4, and Γ_6.

Let vertex 8 be the center of E. We set P_i to be the pyramid with the base Γ_i, $i = \overline{1,\ 6}$, and vertex 8 as an apex, and define

$$e_1 = P_1 \bigcup P_2 \bigcup P_5$$

and

$$e_2 = P_3 \bigcup P_4 \bigcup P_6,$$

as shown on Figure 3.11. We notice that e_1 and e_2 do not overlap.

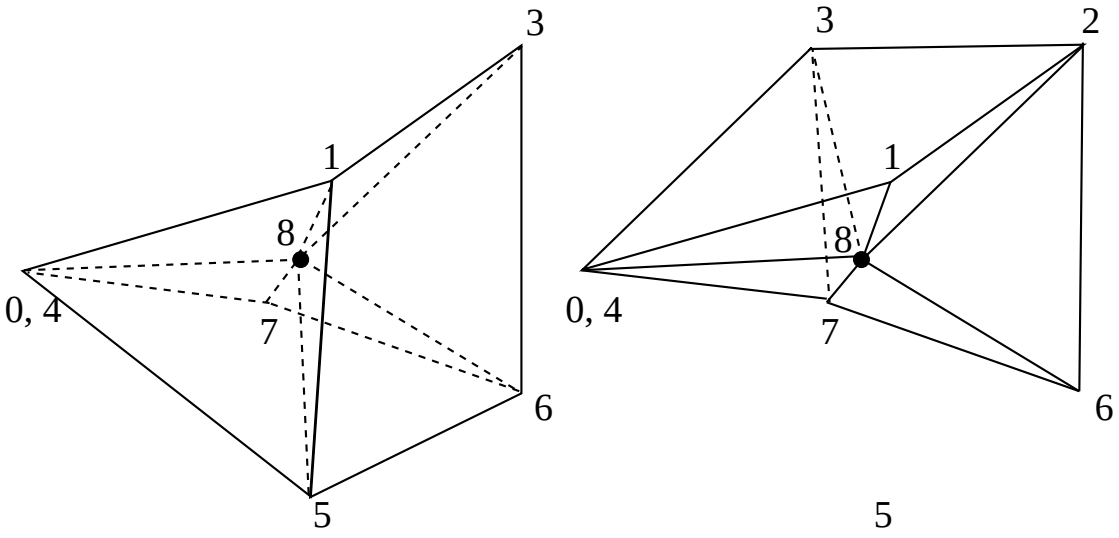

Figure 3.11: Subcells e_1 (left) and e_2 (right) for a H1 element

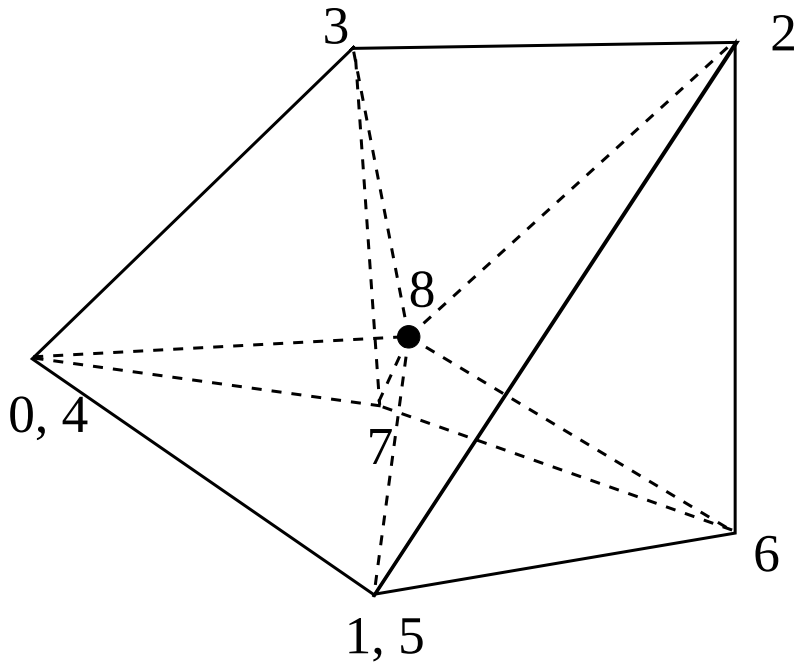

Figure 3.12: Partition of a prism into three pyramids and two tetrahedrons

Horizontal prism

Let E be a prism as shown in Figure 3.12. We denote by $\{\Gamma_i\}$, $i = \overline{1,\ 5}$, the set of faces of E so that Γ_1, Γ_2, and Γ_3 are quadrilateral faces, and Γ_4 and Γ_5 are triangular faces. Without loss of generality, we may assume that the vertex 6 of E is a common vertex of the faces Γ_1, Γ_2, and Γ_4, and the vertex 3 is the common vertex of the faces Γ_2, Γ_3, and Γ_5.

Let vertex 8 be the center of E. We set P_i to be the pyramid with the base Γ_i, $i = \overline{1,\ 5}$, and vertex 8 as an apex, and define

$$e_1 = P_1 \bigcup P_2 \bigcup P_4$$

and

$$e_2 = P_2 \bigcup P_3 \bigcup P_5,$$

as shown on Figure 3.13. We notice that in this case e_1 and e_2 overlap and have common pyramid P_2.

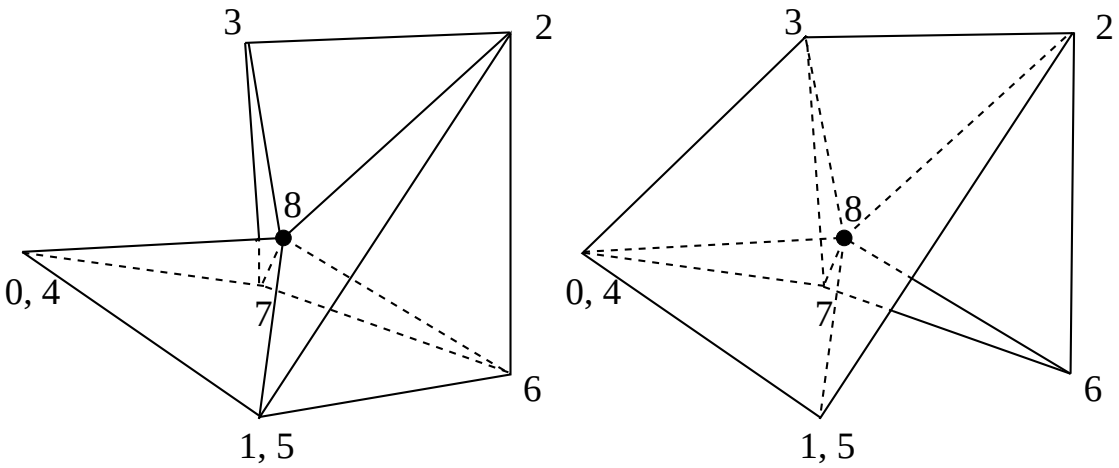

Figure 3.13: Subcells e_1 (left) and e_2 (right) for a prism

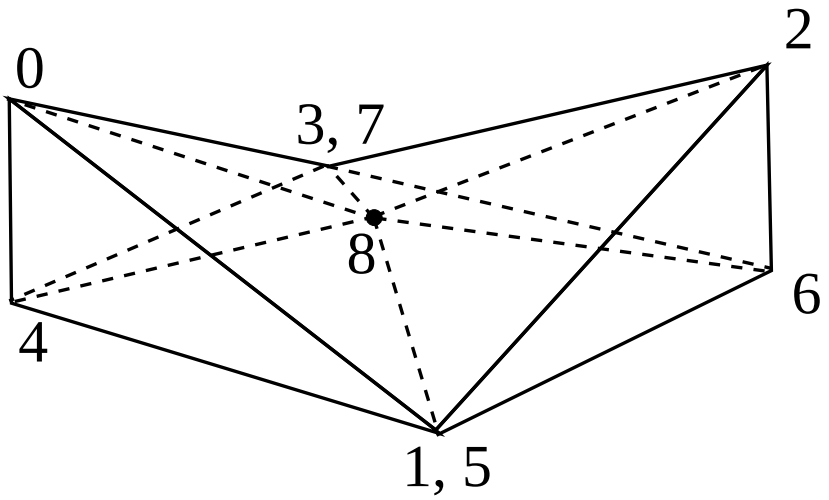

Figure 3.14: Partition of an element H2 into two pyramids and four tetrahedrons

Element H2

Let E be an element H2 as shown in Figure 3.14. We denote by $\{\Gamma_i\}$, $i = \overline{1,\ 5}$, the set of faces of E so that Γ_1 is the bottom quadrilateral face, Γ_2 is the top quadrilateral face, and Γ_i, $i = \overline{3,\ 6}$, are the triangular faces. Without loss of generality, we may assume that the vertex 4 of E is a common vertex of the faces Γ_1, Γ_3, and Γ_4, and the vertex 2 is a common vertex of the faces Γ_2, Γ_5, and Γ_6.

We construct the vertex 8 the following way. Let Q_1 be the center of the face Γ_1 and Q_2 be the center of the face Γ_2. Then we define vertex 8 to be the center of the segment Q_1Q_2. We set P_i to be the pyramid with the base Γ_i, $i = \overline{1,\ 6}$, and vertex 8 as an apex, and define

$$e_1 = P_1 \bigcup P_3 \bigcup P_4$$

and

$$e_2 = P_2 \bigcup P_5 \bigcup P_6,$$

as shown on Figure 3.15. We notice that in this case e_1 and e_2 do not overlap.

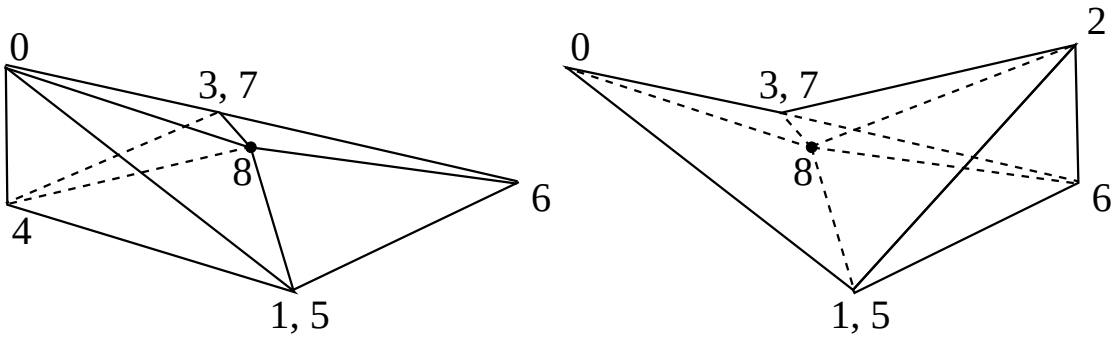

Figure 3.15: Subcells e_1 (left) and e_2 (right) for a H2 element

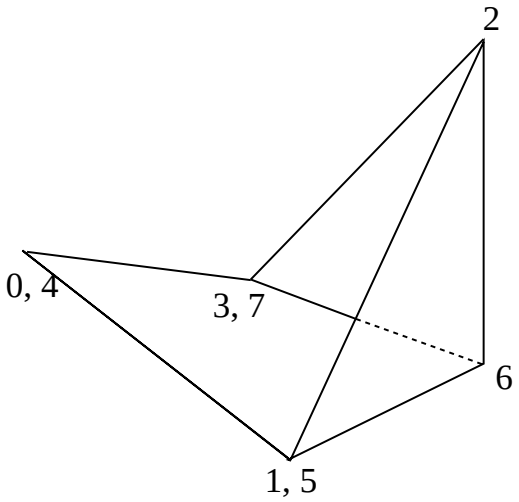

Figure 3.16: Tent

Tent

Let E be a tent as shown in Figure 3.16. We denote by $\{\Gamma_i\}$, $i = \overline{1,\ 4}$, the set of faces of E so that Γ_1 is the bottom quadrilateral face, Γ_2 is the top quadrilateral face and Γ_3 and Γ_4 are triangular faces. This way vertex 6 is the common vertex of the faces Γ_1, Γ_3, and Γ_4 and vertex 2 is the common vertex of faces Γ_2, Γ_3, and Γ_4.

We choose

$$e_1 = e_2 = E.$$

3.2.3 Mass matrices for fault cells

In this Section we describe the discretization technique for the mesh cells with a fault interface. In general, the mesh on different sides of the fault is non-matching, hence the fault reconstruction algorithm is used for the cells on both sides, which results in hexahedrons having one of their faces split into a union of skew polygons. Depending on the mesh geometry, these polygons might have from three to six sides, i.e. they range from triangles to skew hexagons.

An example of a hexahedral cell E with a fault interface, which is given by two

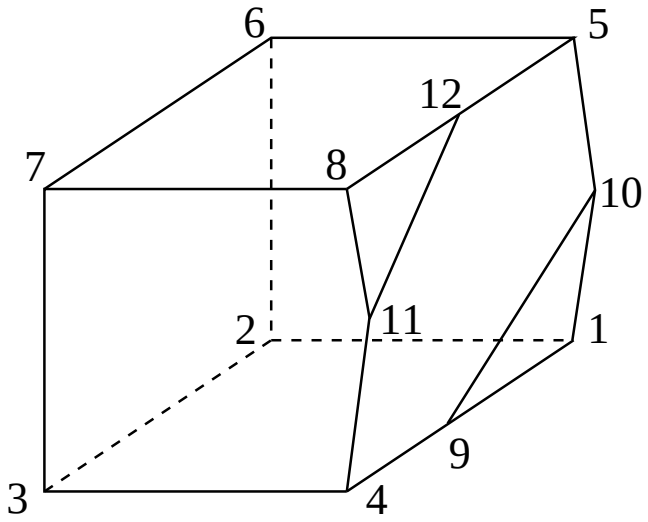

Figure 3.17: An example of a fault hexahedral cell E

triangles and one hexagon, is shown on Figure 3.17.

Let E be a fault hexahedral cell. We denote by V_i, $i = \overline{1,\ 8}$, the vertices of the hexahedron, and by Γ_k, $k = \overline{1,\ 5}$, its regular, non-split faces. The way we enumerate vertices is shown on Figure 3.17. Assuming that the fault face is on the right as in the picture, the ordering of other faces is as follows: bottom, top, back, left, and front.

Skew polygons of the fault face are denoted by Γ_k, $k = \overline{6,\ N}$, and enumerated from the one at the bottom corner to the one at the top corner, as shown on Figure 3.18. Here N stands for the total number of both the regular faces and subfaces of the fault face. The vertices of those polygons are denoted by V_i, $i = \overline{9,\ M}$, where M is the total number of both the original vertices of the hexahedron and the ones resulting from splitting. Additional vertices are indexed from left to right, bottom to top, as can be seen on Figure 3.17.

The discretization procedure requires several steps, which are described below.

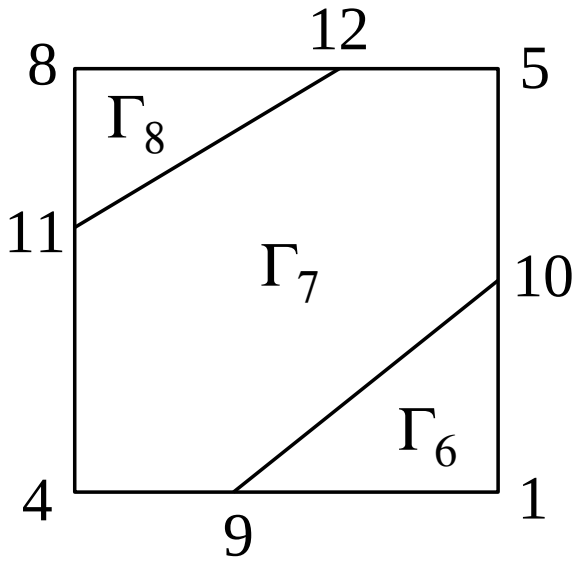

Figure 3.18: An example of a fault face

Step 1: Triangulation of the fault face.

As was mentioned earlier, the subfaces of the fault face are skew polygons ranging from triangles to skew hexagons. On this step, we perform further splitting by triangulating all the polygons with more than three sides. The simplest way to do so is to choose one of their vertices and connect it with all the others.

For all subfaces Γ_k, $k = \overline{6,\ N}$, we do the following. Let M_k be the number of vertices of the polygon Γ_k, which are locally denoted by $V_{k,i}$, $i = \overline{1,\ M_k}$, and are indexed either clockwise or counter-clockwise.

- If $M_k = 3$, we denote Γ_k by Γ_k^1 and set $t_k = 1$;
- If $M_k > 3$, we connect the vertex $V_{k,1}$ with vertices $V_{k,i}$, $i = \overline{3,\ M_k - 1}$. This results in $t_k = M_k - 2$ triangular subfaces, denoted by Γ_k^s, $s = \overline{1,\ t_k}$, and indexed from left to right or from bottom to top.

Performing this procedure we obtain a set of subfaces $\{\Gamma_k^s\}$, $k = \overline{6,\ N}$, $s = \overline{1,\ t_k}$, which is the triangulation of the fault face. The total number of triangular subfaces

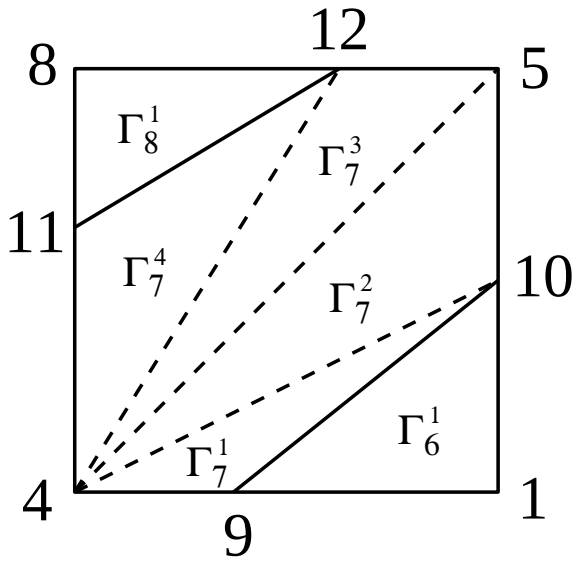

Figure 3.19: An example of the triangulation of a fault face

is denoted by N_T,

$$N_T \;=\; \sum_{k=6}^{N} t_k \;. \tag{3.40}$$

An example of such triangulation for the cell E is given on Figure 3.19. The resulting set of subfaces is $\{\Gamma_6^1,\ \Gamma_7^1,\ \Gamma_7^2,\ \Gamma_7^3,\ \Gamma_7^4,\ \Gamma_8^1\}$, i.e. the skew hexagon was split into four triangles, which makes N_T, the total number of triangular subfaces, equal to 6.

Step 2: Splitting the hexahedron into triangular prisms.

Our next step is representing the hexahedron E as a union of N_T prisms, where every prism is triangular and has one of Γ_k^s, the subfaces of the fault face, as its base. We do it by going through all the regular faces which have at least one extra vertex V_i, $i \in \overline{9,\ M}$, on one of their edges, and projecting these vertices onto the opposite side of the considered skew quadrilateral. Then, we connect every projected point with its original. If two original vertices are connected, we connect their projections as well.

The following procedure is repeated for all vertices V_i, $i = \overline{9,\ M}$. Denote the edge containing V_i by $(V_{i_1}V_{i_2})$, where $i_1, i_2 \in \overline{1,\ 8}$, i.e. V_{i_j} are the regular vertices of the hexahedron. Then the regular face which shares the edge $(V_{i_1}V_{i_2})$ with the fault face can be denoted by $(V_{i_1}V_{i_2}V'_{i_2}V'_{i_1})$. The vertices here are listed either clockwise or counter-clockwise, i.e. the edge $(V'_{i_1}V'_{i_2})$ is the opposite one for the edge $(V_{i_1}V_{i_2})$.

With the notations introduced, the projection V'_i of the vertex V_i is defined as follows:

- $V'_i \in (V'_{i_1}V'_{i_2})$, i.e. it's a point on the opposite edge;
- The ratio of distances to the edge's endpoints is the same for V_i and V'_i, i.e.

$$\frac{\|V'_i - V'_{i_1}\|}{\|V'_i - V'_{i_2}\|} = \frac{\|V_i - V_{i_1}\|}{\|V_i - V_{i_2}\|}, \tag{3.41}$$

  where $\|\cdot\|$ is the standard Euclidean norm.

Once all the projected vertices V'_i, $i = \overline{9,\ M}$, are obtained, we construct the edges of the triangular prisms.

For all the introduced vertices V'_i, $i = \overline{9,\ M}$, we perform the following:

- The projected vertex V'_i is connected with the original vertex V_i, forming the edge (V'_iV_i);
- For all the vertices V_j, $j = \overline{9,\ M}$, $j \neq i$, we check if the points V_i and V_j are connected, i.e. (V_iV_j) is the edge of some subface Γ^s_k, $k = \overline{6,\ N}$, $s = \overline{1,\ t_k}$. If they are, we connect their projections V'_i and V'_j, obtaining the edge $(V'_iV'_j)$;
- Assume that V_{i_1} is a hexahedron's vertex belonging to the fault face, and denote the hexahedron's vertex on the opposite end of the edge not contained in the fault face by $V_{i'_1}$. This is the vertex on the face opposite to the fault face. Starting from V_{i_1} and $V_{i'_1}$ correspondingly, and going clockwise, we

denote the fault face by $(V_{i_1}V_{i_2}V_{i_3}V_{i_4})$, and the opposite face by $(V_{i'_1}V_{i'_2}V_{i'_3}V_{i'_4})$, where $i_k, i'_k \in \overline{1,\ 8}$, i.e. we describe both faces in terms of the opposite vertices of the hexahedron.

Then, for all the vertices V_{i_k}, $k = \overline{1,\ 4}$, we check if the points V_i and V_{i_k} are connected, i.e. $(V_iV_{i_k})$ is the edge of some subface Γ^s_k, $k = \overline{6,\ N}$, $s = \overline{1,\ t_k}$. If they are, we connect the vertices V'_i and $V_{i'_k}$ on the opposite face, obtaining the edge $(V'_iV_{i'_k})$.

The described procedure results in the fault hexahedron E being split into a union of N_T triangular prisms. Each such prism has a subface Γ^s_k, $k \in \overline{6,\ N}$, $s \in \overline{1,\ t_k}$, as its base, and is denoted by e^s_k. Describing the subface Γ^s_k by its three vertices, say, $(V_{k_{s_1}}V_{k_{s_2}}V_{k_{s_3}})$, $k_{s_i} \in \overline{1,\ N}$, we can write $\Gamma^{s\prime}_k$, the other base of the prism e^s_k, as $(V'_{k_{s_1}}V'_{k_{s_2}}V'_{k_{s_3}})$, where $V'_{k_{s_i}}$ is the corresponding vertex for the vertex $V_{k_{s_i}}$, $i = \overline{1,\ 3}$, on the face opposite to the fault face.

An example of such splitting for the cell E is given on Figure 3.20. If we look at, say, the prism e^1_8, we can see that one of its bases is the subface Γ^1_8, which can be written as $(V_8V_{11}V_{12})$, and the other one is $\Gamma^{1\prime}_8 = (V_7V'_{11}V'_{12})$. Here, V'_{11} and V'_{12} are the projections of the fault-specific vertices V_{11} and V_{12}, while V_7 and V_8 are the corresponding vertices of the hexahedron on the fault face and the face opposite to it.

Step 3: Local discretization of the triangular prisms.

At the current stage, the fault hexahedron E can be considered as the union or a cluster of N_T subcells e^s_k, each of which is a triangular prism. Our next step is to apply PWC discretization to every prism e^s_k, $k = \overline{6,\ N}$, $s = \overline{1,\ t_k}$, and obtain N_T local algebraic systems in terms of DOF's $p_{e^s_k}$ and $\bar{\lambda}_{e^s_k}$.

The detailed description of PWC discretization for triangular prisms has been

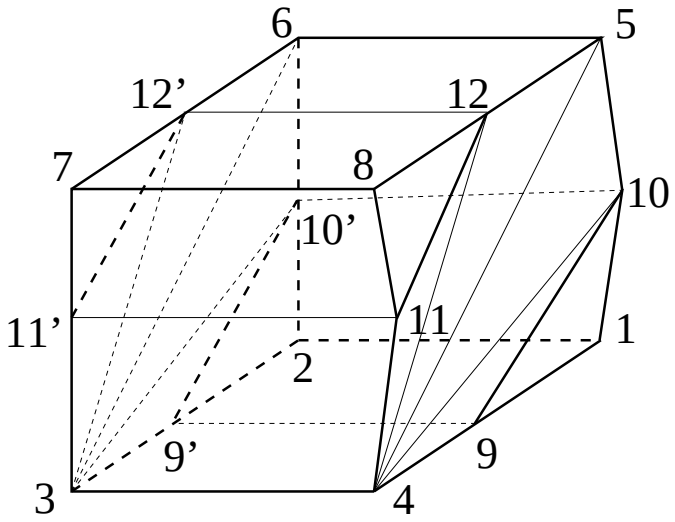

Figure 3.20: An example of splitting the hexahedron into triangular prisms

given earlier. The local system matrices obtained are of the form

$$S_{p,\lambda}^{e_k^s} = \begin{pmatrix} \Sigma_E & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} B_{e_k^s} \\ C_{e_k^s} \end{pmatrix} M_{e_k^s}^{-1} \begin{pmatrix} B_{e_k^s}^T & C_{e_k^s}^T \end{pmatrix}, \quad k = \overline{6,\ N},\ s = \overline{1,\ t_k}\ . \tag{3.42}$$

Note that $p_{e_k^s} \in \mathbb{R}$, $\bar{\lambda}_{e_k^s} \in \mathbb{R}^5$, and $S_{p,\lambda}^{e_k^s} \in \mathbb{R}^{6\times 6}$.

Step 4: Assembling the cluster matrix.

Let us recall that the fault hexahedron E has a total of N interfaces with other cells, where a single DOF $\lambda_{E,k}$ is associated with each such interface Γ_k. Therefore, the dimension of the vector $\overline{\lambda_E}$ in the local system for the cell E should be equal to N, i.e. $\overline{\lambda_E} \in \mathbb{R}^N$. But, when performing the discretization of the triangular prisms e_k^s which the fault hexahedron E was previously split into, we actually operated a greater number of interfaces and corresponding DOF's. To resolve this matter, let us first divide the interfaces of the split fault hexahedron into two groups.

- We define an outer subface $\Gamma_{k,s,n}$ to be a face of one and only one subcell e_k^s, $k \in \overline{6,\ N}$, $s \in \overline{1,\ t_k}$. Such subface $\Gamma_{k,s,n}$ is either contained in a certain interface Γ_n, $n \in \overline{1,\ N}$, or is the interface Γ_n itself.

The described splitting of the fault hexahedron into triangular prisms results in a total of $(2N_T + M - 4)$ outer subfaces.

If we consider the cell E shown on Figure 3.20, then for, say, subcell e_8^1 one example of an outer subface would be $\Gamma_{8,1,5} = (V_7V_8V_{11}V_{11'})$, which is contained in the front interface Γ_5, and another one is the subface $\Gamma_{8,1,8} = (V_8V_{11}V_{12})$, which is in fact the fault interface Γ_8 itself. Note that the opposite outer subface $\Gamma_{8,1,4} = (V_7V_{11'}V_{12'})$ is only a part of the left interface Γ_4.

- We say that an interface $\Gamma_{k,s,k',s'}$ is the inner interface if there are two subcells e_k^s and $e_{k'}^{s'}$, $k, k' \in \overline{6,\ N}$, $s, s' \in \overline{1,\ t_k}$, such that $\Gamma_{k,s,k',s'}$ is the face of both of them, i.e. the interface between these subcells. Such interface never shares more than an edge with any of the interfaces Γ_n, $n = \overline{1,\ N}$.

  The described splitting of the fault hexahedron into triangular prisms results in a total of $(N_T - 1)$ inner interfaces.

  For the cell E on Figure 3.20, an example of the interior interface would be $\Gamma_{7,2,7,3} = (V_3V_4V_5V_6)$, which is shared by subcells e_7^2 and e_7^3.

The cluster system matrix is then defined as such an assembling of the local system matrices for subcells e_k^s, $k = \overline{6,\ N}$, $s = \overline{1,\ t_k}$, that the entries corresponding to the outer subfaces $\Gamma_{k,s,n}$, $k = \overline{6,\ N}$, $s = \overline{1,\ t_k}$, are lumped together, i.e. we lump all the outer subfaces contained in the same interface Γ_n. This matrix is denoted by $\widetilde{S}^E_{p,\lambda}$,

$$\widetilde{S}^E_{p,\lambda} \ = \ \sum_{k,s} \ \widetilde{\mathcal{N}}_{e_k^s} \ S^{e_k^s}_{p,\lambda} \ \widetilde{\mathcal{N}}^T_{e_k^s} \ , \tag{3.43}$$

where the assembling matrices $\widetilde{\mathcal{N}}_{e_k^s}$, $k = \overline{6,\ N}$, $s = \overline{1,\ t_k}$, map all the entries corresponding to the DOF's on the outer subfaces $\Gamma_{k,s,n}$ to a single entry corresponding to a DOF on the interface Γ_n.

Therefore, the cluster system is given in terms of vectors $p_E \in \mathbb{R}^{N_T}$ and $\lambda_{\widetilde{E}} \in \mathbb{R}^{N+N_T-1}$. The interface DOF's are indexed in such a way that the vector $\lambda_{\widetilde{E}}$ can be represented as

$$\lambda_{\widetilde{E}} = \begin{pmatrix} \overline{\lambda_E} \\ \widetilde{\overline{\lambda_E}} \end{pmatrix}, \tag{3.44}$$

where $\overline{\lambda_E} \in \mathbb{R}^N$ is the vector of DOF's corresponding to the interfaces Γ_n, $n = \overline{1,\ N}$, and $\widetilde{\overline{\lambda_E}} \in \mathbb{R}^{N_T-1}$ is associated with the interior interfaces.

The cluster system matrix can then be written in a 3×3 block form,

$$\widetilde{S}^E_{p,\lambda} = \begin{pmatrix} \widetilde{S}_{p_E} & \widetilde{S}_{p_E,\lambda_E} & \widetilde{S}_{p_E,\widetilde{\lambda}_E} \\ \widetilde{S}^T_{p_E,\lambda_E} & \widetilde{S}_{\lambda_E} & \widetilde{S}_{\lambda_E,\widetilde{\lambda}_E} \\ \widetilde{S}^T_{p_E,\widetilde{\lambda}_E} & \widetilde{S}^T_{\lambda_E,\widetilde{\lambda}_E} & \widetilde{S}_{\widetilde{\lambda}_E} \end{pmatrix}, \tag{3.45}$$

where the entries corresponding to the interior interfaces have been grouped together.

Step 5: Elimination of DOF's corresponding to the the interior interfaces.

The cluster system obtained on the previous step still contains additional DOF's associated with the interior interfaces. Therefore, in order to obtain a local system matrix for the fault hexahedron E, we have to eliminate them by taking the Schur complement:

$$S^E_{p,\lambda} = \begin{pmatrix} \widetilde{S}_{p_E} & \widetilde{S}_{p_E,\lambda_E} \\ \widetilde{S}_{\lambda_E,p_E} & \widetilde{S}_{\lambda_E} \end{pmatrix} - \begin{pmatrix} \widetilde{S}_{p_E,\widetilde{\lambda}_E} \\ \widetilde{S}_{\lambda_E,\widetilde{\lambda}_E} \end{pmatrix} \widetilde{S}^{-1}_{\widetilde{\lambda}_E} \begin{pmatrix} \widetilde{S}^T_{p_E,\widetilde{\lambda}_E} & \widetilde{S}^T_{\lambda_E,\widetilde{\lambda}_E} \end{pmatrix}. \tag{3.46}$$

The reduced system is given in terms of vectors $p_E \in \mathbb{R}^{N_T}$ and $\overline{\lambda_E} \in \mathbb{R}^N$, where a component $\lambda_{E,k}$ corresponds to a DOF on the interface Γ_k, $k = \overline{1,\ N}$, of the fault hexahedron E.

The system matrix for the fault hexahedron E can therefore be written in a

2×2 block form,

$$S^E_{p,\lambda} = \begin{pmatrix} S_{p_E} & S_{p_E,\lambda_E} \\ S^T_{p_E,\lambda_E} & S_{\lambda_E} \end{pmatrix} \tag{3.47}$$

with $S_{p_E} \in \mathbb{R}^{N_T \times N_T}$ and $S_{\lambda_E} \in \mathbb{R}^{N \times N}$. This matrix is then assembled into a global system matrix S to find solution vectors p and λ.

3.2.4 Accuracy of approximation

In this Section, let us compare the errors of the PWC method with those of Kuznetsov-Repin elements ([47]).

Consider problem (3.1) with pure Neumann boundary conditions, i.e. $\Gamma_N = \partial\Omega$. Let the diffusion tensor K be the identity matrix in Ω, and $c \equiv 0$. Let p^* be the reference solution,

$$p^*(x, y, z) = \sin(2x)\cos(3y) + xyz^2 - \text{const},$$

and $\boldsymbol{u}^* = -\nabla p^*$ be the reference flux. We define the right-hand side f to be $f = \nabla \cdot \boldsymbol{u}^*$, and the boundary function g_N to be $g_N = \boldsymbol{u}^* \cdot \mathbf{n}$.

Let us introduce the following notations for relative errors:

$$\begin{aligned} \mathcal{E}(p, L_2) &= \|p_h - p^*\|_{L_2} \Big/ \|p^*\|_{L_2}, \\ \mathcal{E}(\boldsymbol{u}, \boldsymbol{L_2}) &= \|\boldsymbol{u}_h - \boldsymbol{u}^*\|_{\boldsymbol{L_2}} \Big/ \|\boldsymbol{u}^*\|_{\boldsymbol{L_2}}, \\ \mathcal{E}(\boldsymbol{u}, H_{\text{div}}) &= \|\boldsymbol{u}_h - \boldsymbol{u}^*\|_{H_{\text{div}}} \Big/ \|\boldsymbol{u}^*\|_{H_{\text{div}}}. \end{aligned} \tag{3.48}$$

We perform experiments for two types of meshes. The first type is a mesh with pinchouts, shown on Figure 3.2.4, with different degrees of refinement. The number of cells for each refinement and relative errors for such meshes are shown in Table 3.1. The second type of mesh is a mesh with a fault, shown on Figure 3.2.4. The number of cells for each refinement and relative errors for such meshes are shown in Table 3.2.

We see that for both types of meshes the PWC discretization is almost as good as KR discretization, and has a significantly smaller construction cost.

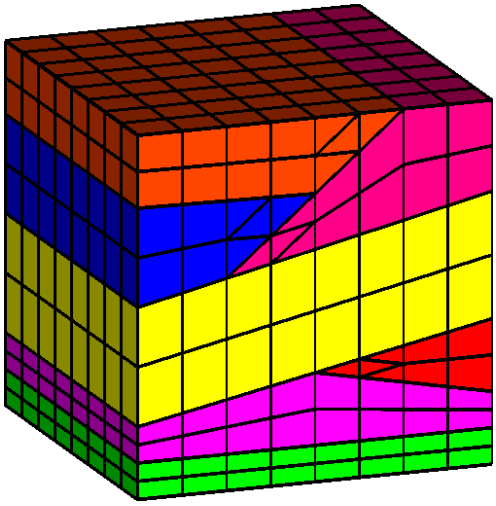

Figure 3.21: An example of a mesh with pinchouts

Table 3.1: Relative errors for a mesh with pinchouts

# of cells	$\mathcal{E}(p, L_2)$		$\mathcal{E}(\boldsymbol{u}, \boldsymbol{L_2})$		$\mathcal{E}(\boldsymbol{u}, H_{\text{div}})$	
	KR	PWC	KR	PWC	KR	PWC
704	13.16	13.17	8.85	8.93	12.56	12.56
5632	6.59	6.59	4.41	4.42	6.29	6.29
45056	3.29	3.29	2.20	2.20	3.15	3.15

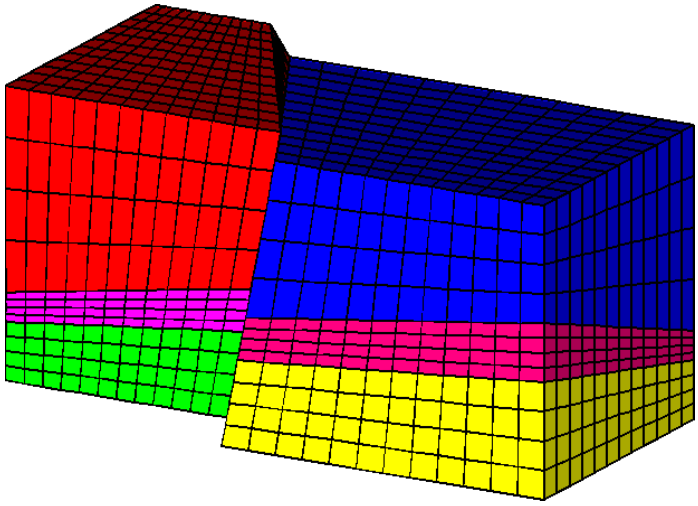

Figure 3.22: An example of a mesh with a fault

Table 3.2: Relative errors for a mesh with a fault

# of cells	$\mathcal{E}(p, L_2)$		$\mathcal{E}(\boldsymbol{u}, \boldsymbol{L_2})$		$\mathcal{E}(\boldsymbol{u}, H_{\text{div}})$	
	KR	PWC	KR	PWC	KR	PWC
3456	10.81	10.82	7.51	7.40	9.12	9.11
27648	5.41	5.41	3.75	3.69	4.57	4.42
221184	2.71	2.71	1.87	1.84	2.28	2.28

3.3 Preconditioner for the PWC system

The preconditioned Conjugate Gradient (PCG) method is one of the most efficient algorithms for solving systems with symmetric positive definite matrices. The goal of this Section is to design a symmetric positive definite matrix $\widehat{S}$, $\widehat{S} = \widehat{S}^T > 0$, which can be used as a reliable and sufficiently cheap preconditioner for the system matrix S from (3.24).

It is well known that the Schur complement matrix S from (3.24) can be written in the assembling form

$$S = \sum N_k S_k N_k^T, \tag{3.49}$$

where

$$S_k = \begin{pmatrix} \Sigma_k & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} B_k \\ C_k \end{pmatrix} M_k^{-1} \begin{pmatrix} B_k & C_k \end{pmatrix}, \tag{3.50}$$

for a regular or pinchout cell, and is given by (3.47) for a fault cell. Using (3.49), we construct our preconditioner in two stages.

Stage 1.

For each mass matrix M_k let $\widetilde{M_k}$ be its diagonal, i.e.

$$\widetilde{M_k} = diag(M_k). \tag{3.51}$$

Then, we introduce matrices

$$S_k = \begin{pmatrix} \Sigma_k & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} B_k \\ C_k \end{pmatrix} \widetilde{M_k}^{-1} \begin{pmatrix} B_k & C_k \end{pmatrix}, \tag{3.52}$$

for regular and pinchout cells. In the case of a fault cell, $\widetilde{S}_k$ is obtained according to the procedure described in Section 3.2.3 with matrices M_k replaced by matrices $\widetilde{M_k}$ in (3.42).

We define matrix $\widetilde{S}$ as

$$\widetilde{S} = \sum N_k \widetilde{S}_k N_k^T. \tag{3.53}$$

Similar to S, $\widetilde{S}$ is a 2×2 block matrix

$$\widetilde{S} = \begin{pmatrix} \widetilde{S}_{11} & \widetilde{S}_{12} \\ \widetilde{S}_{21} & \widetilde{S}_{22} \end{pmatrix}, \tag{3.54}$$

with

$$\widetilde{S}_{22} \;=\; \sum_{k=1}^{n} \mathcal{N}_{k,22}\; \widetilde{S}_{k,22}\; \mathcal{N}_{k,22}^{T} \tag{3.55}$$

where

$$\widetilde{S}_{k,22} \;=\; C_k \widetilde{M}_k^{-1} C_k^T \tag{3.56}$$

for a regular or pinchout cell, and the block S_{λ_E} from (3.47) for a fault cell.

The matrix $\widetilde{M}_k^{-1}$ is diagonal, so it follows that the matrices $C_k \widetilde{M}_k^{-1} C_k^T$, $k = \overline{1,n}$, are also diagonal with positive diagonal entries. In the case of a fault cell, the block corresponding to $\lambda_{\widetilde{E}}$ in (3.45),

$$\widetilde{S}_{\lambda_{\widetilde{E}}} \;=\; \begin{pmatrix} \widetilde{S}_{\lambda_E} & \widetilde{S}_{\lambda_E,\widetilde{\lambda}_E} \\ \\ \widetilde{S}^{T}_{\lambda_E,\widetilde{\lambda}_E} & \widetilde{S}_{\widetilde{\lambda}_E} \end{pmatrix}, \tag{3.57}$$

is the assembling of the diagonal matrices $C_k \widetilde{M}_k^{-1} C_k^T$, which come from the local prismatic subcells, and is therefore also diagonal. Hence, the block $\widetilde{S}_{k,22}$ in the local fault cell matrix $\widetilde{S}_k$ remains diagonal as well.

Therefore, we have that the matrix $\widetilde{S}_{22}$ is diagonal with positive entries.

Now, consider a system

$$\widetilde{S} \begin{pmatrix} \bar{\eta}_1 \\ \\ \bar{\eta}_2 \end{pmatrix} \;=\; \begin{pmatrix} \bar{\xi}_1 \\ \\ \bar{\xi}_2 \end{pmatrix}. \tag{3.58}$$

The block Gauss elimination method for this system can be implemented in the following way. First, we eliminate by substitution the subvector $\bar{\eta}_2$ from the first block equation:

$$\bar{\eta}_2 \;=\; \widetilde{S}_{22}^{-1}\,\left(\bar{\xi}_2 \;-\; \widetilde{S}_{21}\bar{\xi}_1\right)\,, \tag{3.59}$$

where the diagonal matrix $\widetilde{S}_{22}$ is easy to invert. Then, we get the system

$$A_{11}\bar{\eta}_1 \;=\; \bar{z}_1\,, \tag{3.60}$$

where $\bar{z}_1 = \bar{\xi}_1 - \widetilde{S}_{12}\;\widetilde{S}_{22}^{-1}\;\bar{\xi}_2$, and

$$A_{11} \;=\; \widetilde{S}_{11} \;-\; \widetilde{S}_{12}\;\widetilde{S}_{22}^{-1}\;\widetilde{S}_{21}\,. \tag{3.61}$$

After solving system (3.60), we can find the remaining solution vector $\bar{\eta}_2$ from (3.59).

Stage 2.

It can be shown that A_{11} is a Stieltjes matrix. Let $B_{11} \in \mathbb{R}^{n\times n}$ be a symmetric and positive definite matrix which we consider to be a suitable preconditioner for the matrix A_{11} in (3.60). Then, we define a preconditioner $\widehat{S}$ for the matrix S by

$$\widehat{S} \;=\; \begin{pmatrix} B_{11} + \widetilde{S}_{11}\;\widetilde{S}_{11}^{-1}\;\widetilde{S}_{21} & \widetilde{S}_{12} \\ \\ \widetilde{S}_{21} & \widetilde{S}_{22} \end{pmatrix}\,. \tag{3.62}$$

We restrict ourselves with two possible choices of B_{11}. One is the well-known AMG preconditioner, and another is the diagonal preconditioner.

3.3.1 Numerical experiments

We consider the following test example. The domain Ω, shown on Fig. 3.23, is a parallelepiped with the dimensions $1.0 \times 1.0 \times 0.25$. It consists of five subdomains, called geological layers, with layers 2 and 4 being "thin". The layers are enumerated from bottom to top. The mesh used is conforming, uniform in the xy-plane, and is

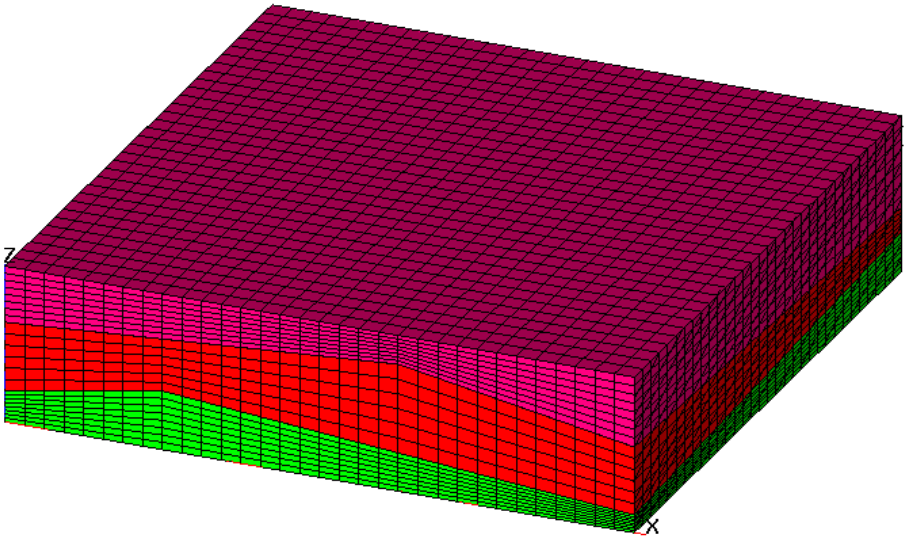

Figure 3.23: Domain with five oblique "bended" geological layers

uniform along z-direction inside each geological layer. Table 3.3 contains the mesh step sizes in each direction for each layer.

The domain is a parallelepiped with the dimensions $1.0 \times 1.0 \times 0.25$. The mesh is uniform for x and y coordinates with the step $h_{xy} = 0.3125$, i.e. we have a grid of 32×32 square cells, resulting in 1024 bases for the hexahedrons in each horizontal mesh layer.

The diffusion tensor is diagonal and piecewise constant, i.e. K_s, the diffusion tensor in the s-th layer, is as follows:

$$K_s = \begin{pmatrix} K_{s,xy} & 0 & 0 \\ 0 & K_{s,xy} & 0 \\ 0 & 0 & K_{s,z} \end{pmatrix}, \tag{3.63}$$

where $K_{s,xy}$ and $K_{s,z}$ are given constants, presented in Table 3.4.

The fault is assumed to happen on the right boundary of the domain, and is

Table 3.3: Geometrical parameters of the mesh cells

	h_{xy}	h_z	h_{xy}/h_z
Layer #1	0.3125	[0.003558, 0.016125]	[1.9379, 8.7826]
Layer #2	0.3125	0.000005	6250
Layer #3	0.3125	[0.002258, 0.023737]	[1.3165, 13.839]
Layer #4	0.3125	0.00005	625
Layer #5	0.3125	[0.005775, 0.018405]	[1.6979, 5.4118]

Table 3.4: Diffusion tensor parameters for the domain with five layers

	Layer #1	Layer #2	Layer #3	Layer #4	Layer #5
$K_{s,xy}$	5	10000	10	1000	10
$K_{s,z}$	1	1000	5	500	1

emulated by splitting the corresponding faces using a set of parallel planes

$$z - \alpha y = \beta_k, \tag{3.64}$$

where α is the slope of the plane which depends on the domain height, and β_k is a set of coefficients such that

$$\beta_{k+1} - \beta_k = const.$$

An illustration of the geometry, leading to the fault hexahedrons described above, is given on Figure 3.24. There, you can see that the geological layer on the right side of the fault has been shifted in vertical direction and rotated around the x-axis, which resulted in the "fault" faces of the cells on both sides being split into unions of polygons by the planes containing "horizontal" sides of the cells across the fault.

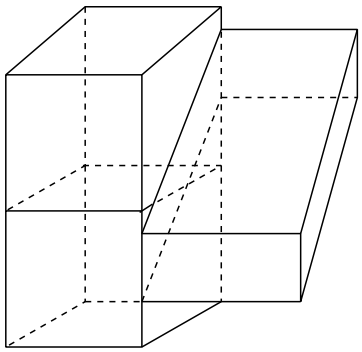

Figure 3.24: The cells on two sides of the geological fault

We compare the performance of the diagonal preconditioner (DIAG), the well-known AMG preconditioner (AMG), and the preconditioner proposed in this Chapter. In the case of our new preconditioner $H = \widehat{S}^{-1}$, we consider two possible choices for the internal substitution of the matrix A_{11} by its preconditioner B_{11} as shown in (3.62). Using AMG preconditioner for that purpose gives us the first variant, Schur Complement AMG (SCAMG). The alternative is to use the diagonal preconditioner, which results in Schur Complement Diagonal preconditioner (SCDiag).

We use PCG as an outer iterative procedure with a typical stopping criteria

$$\frac{\|\vec{r}^k\|_2}{\|\vec{r}^0\|_2} < 10^{-6}. \tag{3.65}$$

The numerical results are presented in Table 3.5 with the following notations used:

- t_{const} - the construction time of a preconditioner;
- t_{prec} - the inversion time of a preconditioner;
- $\#iter$ - number of PCG iterations;
- t_{sol} - the solution time of the system;
- t_{total} - the total time to solve the system.

Table 3.5: Comparison of preconditioners for the domain with five layers

Prec		t_{const}	t_{prec}	# iter	t_{sol}	t_{total}
$c = 0$	AMG	0.813	28	0.022	0.731	**1.544**
	SCAMG	0.360	13	0.010	0.179	**0.515**
$c = 100$	AMG	0.872	11	0.022	0.301	**1.173**
	Diag	0.001	4622	0.001	14.700	**14.701**
	SCAMG	0.361	11	0.010	0.156	**0.517**
	SCDiag	0.025	2291	0.002	11.076	**11.101**
$c = 10000$	AMG	0.694	6	0.021	0.167	**0.862**
	Diag	0.001	602	0.001	1.947	**1.942**
	SCAMG	0.294	10	0.009	0.138	**0.432**
	SCDiag	0.025	348	0.002	1.708	**1.732**

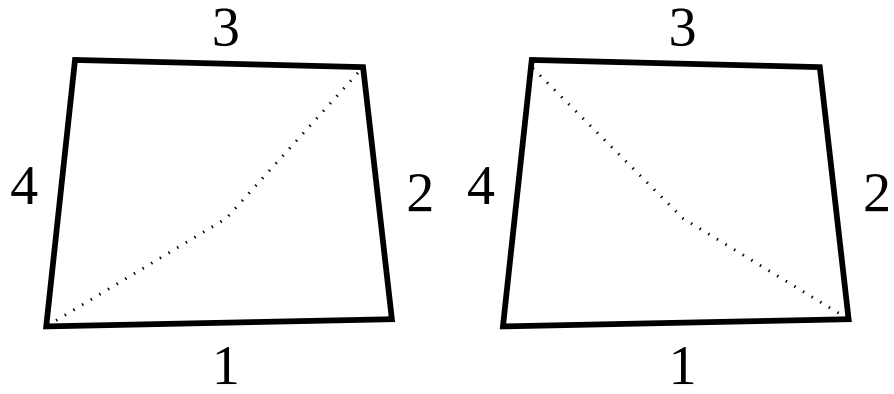

Figure 3.25: "Splitting One" (left) and "Splitting Two" (right) of a quadrilateral cell

3.4 Cell-centered scheme

In this Section, we are going to describe a way to construct a matrix in terms of only variables p from the standard mixed hybrid matrix $A_{u,p,\lambda}$ in terms of u, p and λ defined in (3.21). In order to do that, we will assume that our mesh consists of only hexahedral (3D) cells, and that we use the piecewise constant approximation, described in Section 3.2. We will first describe the idea in 2D and then expand it to the 3D space.

3.4.1 2D: quadrilateral mesh cells

Let E be a quadrilateral cell. In order to make use of the PWC approximation, one should be able to split E into two subcells E_1 and E_2. It can be done in two different ways, shown on Figure 3.25. Let us call these splittings "Splitting One" and "Splitting Two".

The stencil for the local mass matrix will play an important role in the following discussion. It depends on the splitting. Let us denote by "x" the positions of

nonzero elements in the matrix. Then, for the "Splitting One", we would have

$$M = \begin{pmatrix} \times & \times & & \\ \times & \times & & \\ & & \times & \times \\ & & \times & \times \end{pmatrix}, \tag{3.66}$$

and for the "Splitting Two" we have

$$M = \begin{pmatrix} \times & & & \times \\ & \times & \times & \\ & \times & \times & \\ \times & & & \times \end{pmatrix}. \tag{3.67}$$

Matrix M of (3.21) is a block-diagonal matrix

$$M = \begin{pmatrix} M_1 & & & \\ & M_2 & & \\ & & \ddots & \\ & & & M_N \end{pmatrix}, \tag{3.68}$$

with each M_k having one of the two patterns, (3.66) or (3.67).

The matrix $A_{p,\lambda}$ in terms of only p and λ can be derived from the matrix $A_{u,p,\lambda}$ by assembling the local Schur complement matrices:

$$A_{p,\lambda} = \sum_k N_k S_k N_k^T = \begin{pmatrix} \widetilde{A}_p & \widetilde{A}_{p,\lambda} \\ \widetilde{A}_{\lambda,p} & \widetilde{A}_\lambda \end{pmatrix}, \tag{3.69}$$

where

$$S_k = \begin{pmatrix} \Sigma_k & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} B_k \\ C_k \end{pmatrix} M_k^{-1} \begin{pmatrix} B_k^T & C_k^T \end{pmatrix}. \tag{3.70}$$

The matrix S_k can be written in a block 2×2 form

$$S_k = \begin{pmatrix} S_{k,p} & S_{k,p\lambda} \\ S_{k,\lambda p} & S_{k,\lambda} \end{pmatrix}. \tag{3.71}$$

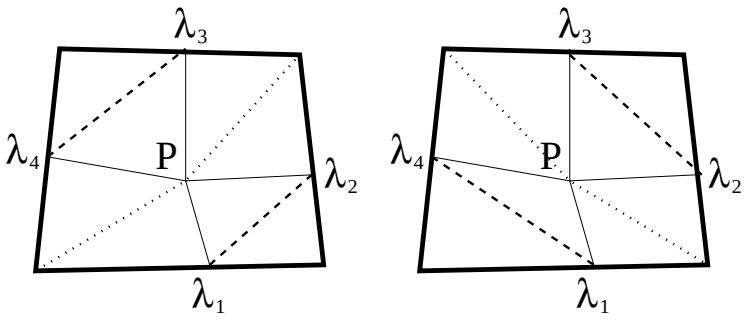

Figure 3.26: Connections $p-p$ and $p-\lambda$ for the "Splitting One" (left) and "Splitting Two" (right)

Depending on the splitting type, S_k has one of the following two nonzero patterns:

$$\begin{pmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & & \\ \times & \times & \times & & \\ \times & & & \times & \times \\ \times & & & \times & \times \end{pmatrix} \quad \text{or} \quad \begin{pmatrix} \times & \times & \times & \times & \times \\ \times & \times & & & \times \\ \times & & \times & \times & \\ \times & & \times & \times & \\ \times & \times & & & \times \end{pmatrix}. \tag{3.72}$$

Figure 3.26 shows the connections between degrees of freedom λ and p inside mesh cells. Connections $\lambda - \lambda$ are shown as dotted lines, and connections $p - \lambda$ are shown as thin lines.

Now let $\Omega = \bigcup_k E_k$ where each E_k is a quadrilateral cell. Let us choose the splitting type of each cell such that each neighbor of a cell with splitting type 1 has splitting type 2, and vice versa. We will call it a chess splitting. An example of pattern of $A_{p,\lambda}$ for this kind of mesh is shown on Figure 3.27.

We see these $\lambda - \lambda$ connections form clusters, consisting of one, two, or four connections, and clusters are isolated. In terms of matrix $A_{p,\lambda}$, it means that there exists an ordering of λ such that $\widetilde{A}_\lambda$ is block-diagonal. Therefore, we can easily construct matrix A_p as a Schur complement. The stencil for the resulting matrix is shown on Figure 3.28.

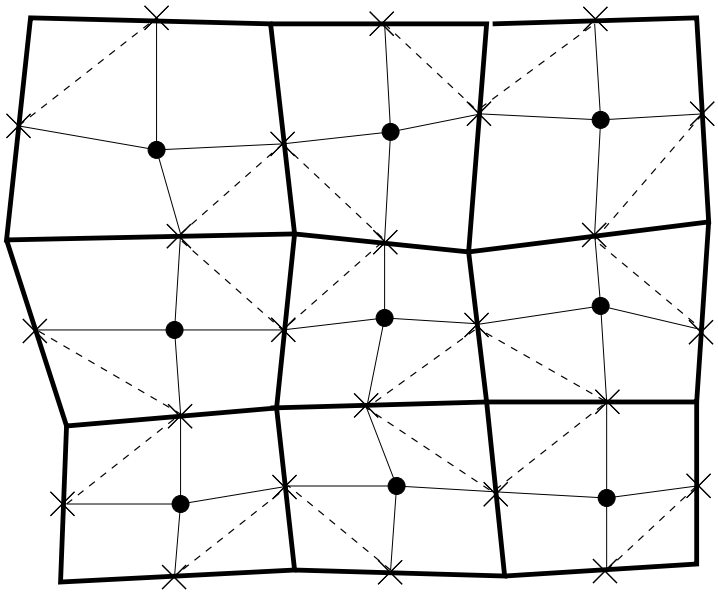

Figure 3.27: $p - p$ and $p - \lambda$ connections for the matrix $A_{p,\lambda}$

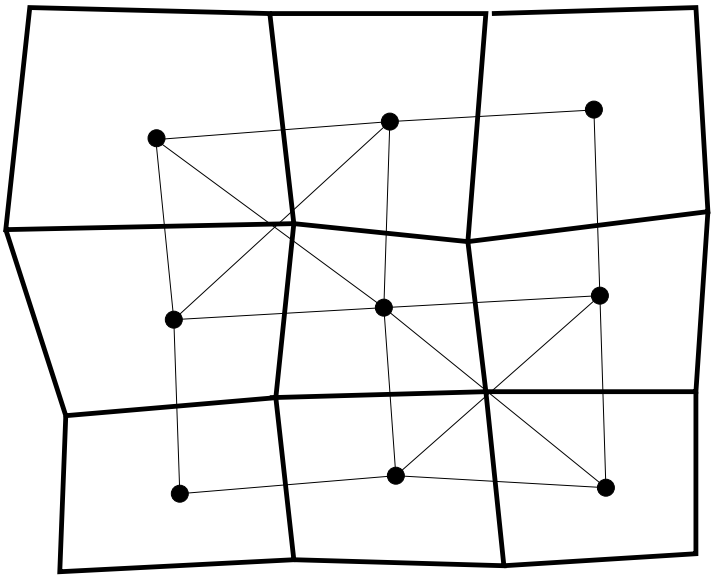

Figure 3.28: Stencil for the matrix A_p

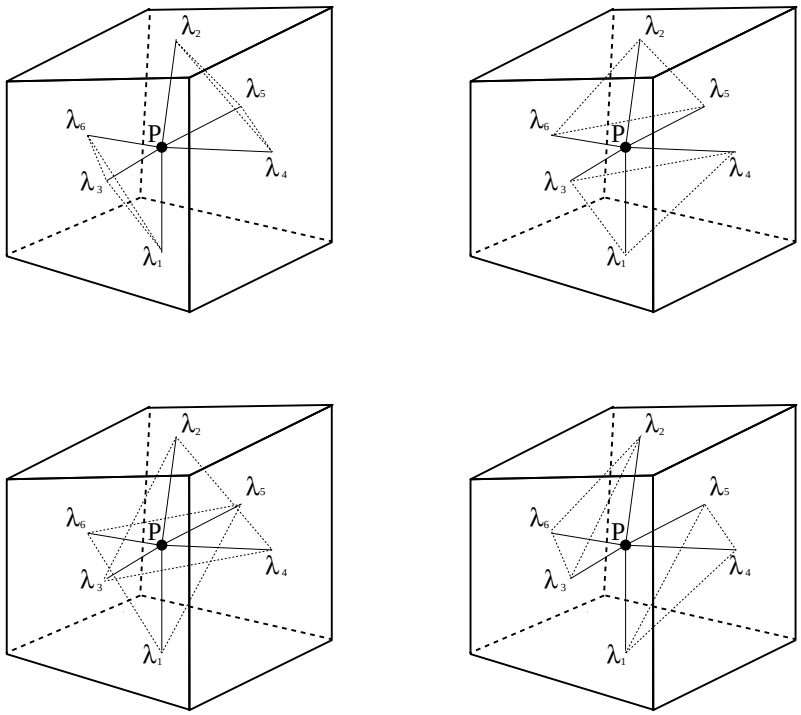

Figure 3.29: Connections $p-p$ and $p-\lambda$ for different splittings of a hexahedral cell

3.4.2 3D: hexahedral mesh cells

For a hexahedral cell we have four different splittings into two subcells. The resulting stencils of the matrix S_k are shown on Figure 3.29.

3D analogue of a "chess" ordering results in clusters of degrees of freedom of λ of one of the sizes 3, 6, 8, 12.

3.4.3 Numerical results

For our experiments, we construct a distorted hexahedron mesh (an example of a $4 \times 6 \times 24$ mesh is shown on Figure 3.30). We generate the mesh by taking a regular Cartesian grid and shifting each node randomly in its neighborhood, so that in each

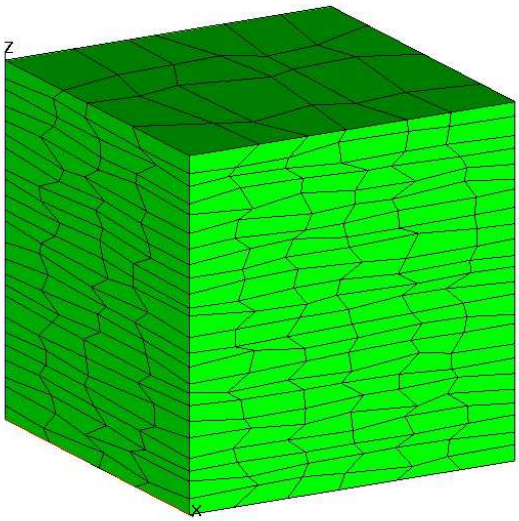

Figure 3.30: Example of a hexahedral mesh ($4 \times 6 \times 24$)

dimension it is shifted by no more than 30% of the mesh step in that dimension.

In Tables 3.6 and 3.7 we compare two ways of solving the original system. The first way is to solve $A_{p,\lambda}$ system by using PCG method with the AMG preconditioner. The results of this approach are shown in the second column of both tables. The second way (shown in column 3) is to construct the system with the matrix A_p and to solve it by using PCG method with AMG preconditioner, and then restore λ degrees of freedom. The restriction and restoration steps time, denoted by Schur time, takes a significant part of the total time. However, the AMG construction time for the smaller system is about 30% less than for of the full system, and each iteration is much cheaper, resulting in a faster solution of the system.

Table 3.6: Comparison of two methods for a $20 \times 30 \times 120$ mesh

	AMG	AMG w Schur
Schur time (s)	-	1.29
Construction time (s)	3.41	2.48
# of PCG iterations	14	15
Solution time (s)	4.18	2.71
Total time (s)	**7.59**	**6.48**

Table 3.7: Comparison of two methods for a $80 \times 80 \times 80$ mesh

	AMG	AMG w Schur
Schur time (s)	-	9.04
Construction time (s)	54.85	35.53
# of PCG iterations	8	8
Solution time (s)	16.73	7.23
Total time (s)	**71.58**	**51.80**

Chapter 4

Preconditioner for unsymmetric M-matrices

Finite volume discretizations of convection-diffusion equations result in large scale systems of algebraic equations with nonsymmetric diagonally dominant M-matrices. By the definition, all off-diagonal entries of M-matrices are non-positive.

In this Chapter, we propose and investigate multilevel iterative solvers for M-matrices with strong diagonal dominance, which are relevant to applications in the reservoir simulation.

4.1 Problem formulation

In this Chapter, we consider an algebraic system

$$A\bar{x} = \bar{b}, \tag{4.1}$$

assuming that the system matrix A is an M-matrix. By definition [68], $n \times n$ matrix A with entries a_{ij}, $i, j = \overline{1, n}$, is said to be an M-matrix if $a_{ij} \leq 0$ for all $i \neq j$, A is non-singular, and $A^{-1} \geq \Theta$, i.e. all entries in A^{-1} are non-negative (Θ is the null

matrix). We additionally assume that A is strictly diagonally dominant matrix, i.e.

$$\sum_{j=1}^{n} a_{ij} > 0 \quad \text{for all } i. \tag{4.2}$$

Let A be an M-matrix. A splitting

$$A = B - C, \tag{4.3}$$

where $B^{-1} \geq \Theta$ and $C \geq \Theta$ are given matrices, is called a *regular splitting of the M-matrix A.*

Let A be a block 2×2 M-matrix

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \tag{4.4}$$

with square diagonal blocks A_{11} and A_{22}. Then matrices A_{11} and A_{22} are also M-matrices.

Let us represent the matrix A from (4.4) in the following equivalent form

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & S_{22} + A_{21}A_{11}^{-1}A_{12} \end{pmatrix} , \tag{4.5}$$

where the matrix

$$S_{22} = A_{22} - A_{21}A_{11}^{-1}A_{12} \tag{4.6}$$

is said to be the Schur complement for the submatrix A_{11}. It is well known that S_{22} is also an M-matrix.

The splitting $A = B - C$ is called *weak regular* if $B^{-1} \geq \Theta$ and $CB^{-1} \geq \Theta$. Obviously, a regular splitting is weak regular, but the converse is not always true.

The following theorem plays a fundamental role in the theory of M-matrices.

Theorem 4.1. *If $A = B - C$ is a regular splitting of the matrix A, and $A^{-1} \geq \Theta$, then*

$$\rho(B^{-1}C) = \frac{\rho(B^{-1}C)}{1 + \rho(A^{-1}C)} < 1. \tag{4.7}$$

Thus, the matrix $B^{-1}C$ is convergent, and the iterative method

$$Bx^{k+1} = Cx^k + b \tag{4.8}$$

converges for any initial vector x^0.

The iterative method (4.8) can be written in the alternative form

$$Bx^{k+1} = Bx^k + (b - Ax^k),$$

or

$$x^{k+1} = x^k + B^{-1}(b - Ax^k). \tag{4.9}$$

Therefore, one can say that matrix B is a preconditioner for the matrix A in a general iterative scheme

$$M(x^{k+1} - x^k) + Ax^k = b. \tag{4.10}$$

It can be proved that for a weak regular splitting $A = B - C$ of an M-matrix A, iterative method (4.8) converges for any initial vector x^0 (see [45, 51]).

4.2 Multilevel preconditioner

Let A_0 be an M-matrix, and assume that the splitting

$$A_0 = B_0 - C_0 \tag{4.11}$$

is regular with B_0 being an M-matrix. Assume now that B_0 is written in a block 2×2 form

$$B_0 = \begin{pmatrix} B_{0,11} & B_{0,12} \\ B_{0,21} & B_{0,22} \end{pmatrix}. \tag{4.12}$$

Define $A_1 = B_{0,11} - B_{0,12}B_{0,22}^{-1}B_{0,21}$.

Lemma 4.1. *A_1 is an M-matrix.*

Proof. As $B_{0,12}B_{0,22}^{-1}B_{0,21} \geq 0$, all off-diagonal entries of A_1 are non-positive. Also,

$$B_0^{-1} = \begin{pmatrix} A_1^{-1} & -A_1^{-1}B_{0,12}B_{0,22}^{-1} \\ -B_{0,22}^{-1}B_{0,21}A_1^{-1} & B_{0,22}^{-1} + B_{0,22}^{-1}B_{0,21}A_1^{-1}B_{0,12}B_{0,22}^{-1} \end{pmatrix} \geq 0.$$

Therefore $A_1^{-1} \geq 0$ and A_1 is an M-matrix. □

Let us define matrices F_1 and F_2 as follows:

$$F_1 = \begin{pmatrix} I & B_{0,12}B_{0,22}^{-1} \\ 0 & I \end{pmatrix}, \qquad F_2 = \begin{pmatrix} I & 0 \\ B_{0,22}^{-1}B_{0,21} & I \end{pmatrix}. \tag{4.13}$$

Then B_0 can be written in the following form:

$$B_0 = F_1 \begin{pmatrix} A_1 & \\ & B_{0,22} \end{pmatrix} F_2.$$

Lemma 4.2. *Let $A_1 = B_1 - C_1$ be a weak regular splitting. Let $s \geq 1$ be an integer. Define $\widehat{A}_1$ as*

$$\widehat{A}_1^{-1} = B_1^{-1}\left[I + (C_1B_1^{-1}) + ... + (C_1B_1^{-1})^{s-1}\right]. \tag{4.14}$$

Then $A_0 = \widehat{B}_0 - \widehat{C}_0$, where

$$\widehat{B}_0 = F_1 \begin{pmatrix} \widehat{A}_1 & 0 \\ 0 & B_{0,22} \end{pmatrix} F_2, \tag{4.15}$$

is a weak regular splitting.

Proof. Let us first prove that $\widehat{B}_0^{-1} \geq 0$. As the splitting of A_1 is weak regular, we have $B_1^{-1} \geq 0$ and $C_1B_1^{-1} \geq 0$. Therefore, $\widehat{A}_1^{-1} \geq 0$. One can easily show that $F_1^{-1} \geq 0$ and $F_2^{-1} \geq 0$. As $B_{0,22}^{-1} \geq 0$, we can conclude that the matrix

$$\widehat{B}_0^{-1} = F_2^{-1} \begin{pmatrix} \widehat{A}_1^{-1} & 0 \\ 0 & B_{0,22}^{-1} \end{pmatrix} F_1^{-1}$$

is non-negative.

It remains to prove that $\widehat{C}_0\widehat{B}_0^{-1} \geq 0$,

$$\widehat{C}_0\widehat{B}_0^{-1} = (\widehat{B}_0 - A_0)\widehat{B}_0^{-1} = I - (B_0 - C_0)\widehat{B}_0^{-1} = (I - B_0\widehat{B}_0^{-1}) + C_0\widehat{B}_0^{-1}.$$

As both matrices C_0 and $\widehat{B}_0^{-1}$ are non-negative, it is sufficient to prove that the matrix $I - B_0\widehat{B}_0^{-1}$ is non-negative:

$$\begin{aligned} I - B_0\widehat{B}_0^{-1} &= I - F_1 \begin{pmatrix} A_1 & \\ & B_{0,22} \end{pmatrix} F_2 F_2^{-1} \begin{pmatrix} \widehat{A}_1^{-1} & \\ & B_{0,22}^{-1} \end{pmatrix} F_1^{-1} \\ &= F_1 \left[I - \begin{pmatrix} A_1\widehat{A}_1^{-1} & \\ & I \end{pmatrix} \right] F_1^{-1} \\ &= F_1 \begin{pmatrix} I - A_1\widehat{A}_1^{-1} & 0 \\ 0 & 0 \end{pmatrix} F_1^{-1} \\ &= \begin{pmatrix} I - A_1\widehat{A}_1^{-1} & 0 \\ 0 & 0 \end{pmatrix}. \end{aligned}$$

Let us prove that $I - A_1\widehat{A}_1^{-1} \geq 0$:

$$\begin{aligned} I - A_1\widehat{A}_1^{-1} &= I - (B_1 - C_1)B_1^{-1}(I + (C_1B_1^{-1}) + ... + (C_1B_1^{-1})^{s-1}) \\ &= I - (I - C_1B_1^{-1})(I + (C_1B_1^{-1}) + ... + (C_1B_1^{-1})^{s-1}) \\ &= I - (I - (C_1B_1^{-1})^s) = (C_1B_1^{-1})^s \geq 0. \end{aligned}$$

□

Let $t \geq 1$. Assume that we constructed a sequence of matrices

$$A \equiv A_0 \to B_0 \to A_1 \to B_1 \to \cdots \to B_{t-1} \to A_t \to B_t, \tag{4.16}$$

such that for any $k \geq 1$

- A_k is an M-matrix;

- $A_k = B_k - C_k$ is a regular splitting with B_k being an M-matrix;

- B_k can be presented in a 2×2 block form

$$B_k = \begin{pmatrix} B_{k,11} & B_{k,12} \\ B_{k,21} & B_{k,22} \end{pmatrix},$$

such that $B_{k,11} = A_{k+1} + B_{k,12}B_{k,22}^{-1}B_{k,21}$;

This block representation is equivalent to the multiplicative form

$$B_k = F_{k,1} \begin{pmatrix} A_{k+1} & 0 \\ 0 & B_{k,22} \end{pmatrix} F_{k,2}, \tag{4.17}$$

with

$$F_{k,1} = \begin{pmatrix} I & B_{k,12}B_{k,22}^{-1} \\ 0 & I \end{pmatrix}, \qquad F_{k,2} = \begin{pmatrix} I & 0 \\ B_{k,22}^{-1}B_{k,21} & I \end{pmatrix}. \tag{4.18}$$

Using this sequence of matrices, we construct our preconditioner $\widehat{B}$ in a following way. We start from the coarsest level $k = t$ and define $\widehat{B}_t = B_t$. Then, for each k, $k = t-1, ..., 0$, we choose an integer s_k and construct the matrix $\widehat{B}_k$ as

$$\widehat{B}_k = F_{k,1} \begin{pmatrix} \widehat{A}_{k+1} & \\ & B_{k,22} \end{pmatrix} F_{k,2},$$

where

$$\widehat{A}_{k+1} = \widehat{B}_{k+1}^{-1} \left[I + (C_{k+1}B_{k+1}^{-1}) + ... + (C_{k+1}B_{k+1}^{-1})^{s_k - 1} \right].$$

It can be shown that for given ξ, the vector $\eta = \widehat{A}_{k+1}^{-1}\xi$ can be computed by the iterative procedure

$$\begin{aligned} \eta_0 &= 0, \\ \eta_j &= \eta_{j-1} + \widehat{B}_{k+1}(\xi - A_{k+1}\eta_{j-1}), \\ \eta &= \eta_{s_k}. \end{aligned}$$

From Lemma 4.2 it follows that the resulting splitting $A = \widehat{B}_0 - \widehat{C}_0$ is weak regular, and therefore the iterative method

$$x^{k+1} = x^k + \widehat{B}_0^{-1}(b - Ax^k) \tag{4.19}$$

is convergent.

4.3 Two-level preconditioner

The goal of this Section is to construct a regular splitting of $A = B - C$ such that

$$\rho(B^{-1}C) \leq q < 1, \tag{4.20}$$

where q is a given parameter.

While there are many splittings satisfying (4.20) (for instance, $B = A$, $C = O$ results in $\rho(B^{-1}C) = 0$), the properties of our algorithm significantly depend on whether or not the following criteria are met

1. Matrix B should have a sparser structure than that of A;

2. Matrix B should be easier to invert than matrix A.

Let us replace property (2) with an easier one

2′. Matrix B should have greater diagonal dominance than A.

Let i be an index of a row in A. We define a_{ij} as

$$a_{ij} = \begin{cases} (A)_{ii}, & j = i, \\ -(A)_{ij}, & j \neq i. \end{cases} \tag{4.21}$$

Let us denote $J_i = \{j \neq i \mid a_{ij} \neq 0\}$. As A is strictly diagonally dominant by our assumptions, we can write

$$a_{ii} = c_i + \sum_{j \in J_i} a_{ij}, \tag{4.22}$$

where $c_i > 0$.

We define matrix $B = (B_{ij})$ as

$$B_{ij} = \begin{cases} c_i^{(1)} + \sum_{j \in J_i} \alpha_{ij} a_{ij}, & j = i, \\ -\alpha_{ij} a_{ij}, & j \neq i, j \in J_i, \\ 0, & \text{otherwise}, \end{cases} \tag{4.23}$$

where α_{ij} are some coefficients from the segment $[0, 1]$, and $c_i^{(1)} > 0$. It is clear that B is an M-matrix with strict diagonal domination. One can also show that the matrix $C = B - A = (C_{ij})$ has the form

$$C_{ij} = \begin{cases} c_i^{(2)}, & j = i, \\ (1 - \alpha_{ij}) a_{ij}, & j \neq i, \end{cases} \tag{4.24}$$

with

$$c_i^{(2)} = c_i^{(1)} - c_i - \sum_{j \in J_i} (1 - \alpha_{ij}) a_{ij}. \tag{4.25}$$

Our goal is to construct a regular splitting, therefore matrix C must be non-negative, i.e. $c_i^{(2)} \geq 0$, which is equivalent to

$$c_i^{(1)} \geq c_i + \sum_{j \in J_i} (1 - \alpha_{ij}) a_{ij}. \tag{4.26}$$

Now consider the following generalized eigenvalue problem

$$Cw = \lambda B w. \tag{4.27}$$

As matrix $B^{-1}C$ is non-negative, the Perron-Frobenius theorem states that there exists an eigenvector $w \geq 0$ such that the corresponding eigenvalue is equal to $\rho(B^{-1}C)$. Let i be an index such that

$$w_i = \max_j w_j > 0.$$

Then the i-th equation of (4.27),

$$c_i^{(2)} w_i + \sum_{j \in J_i} (1 - \alpha_{ij}) a_{ij} w_j = \lambda \left[\left(c_i^{(1)} + \sum_{j \in J_i} \alpha_{ij} a_{ij} \right) w_i - \sum_{j \in J_i} \alpha_{ij} a_{ij} w_j \right] ,$$

results in the inequality

$$(c_i^{(2)} + \sum_{j \in J_i} (1 - \alpha_{ij}) a_{ij}) w_i \geq \lambda c_i^{(1)} w_i,$$

or

$$\lambda \leq \frac{c_i^{(2)} + \sum_{j \in J_i} (1 - \alpha_{ij}) a_{ij}}{c_i^{(1)}}. \tag{4.28}$$

Substituting $c_i^{(2)}$ in (4.28) with (4.25), we get

$$\lambda \leq \frac{c_i^{(1)} - c_i}{c_i^{(1)}}. \tag{4.29}$$

In order to satisfy (4.20), we require

$$\frac{c_i^{(1)} - c_i}{c_i^{(1)}} \leq q. \tag{4.30}$$

Coupled with (4.26), it gives us the following range for $c_i^{(1)}$

$$c_i + \sum_{j \in J_i} (1 - \alpha_{ij}) a_{ij} \leq c_i^{(1)} \leq \frac{1}{1 - q} c_i. \tag{4.31}$$

The following condition is required for the unemptiness of the segment:

$$c_i + \sum_{j \in J_i} (1 - \alpha_{ij}) a_{ij} \leq \frac{1}{1 - q} c_i, \tag{4.32}$$

or, in a simplier form,

$$\sum_{j \in J_i} (1 - \alpha_{ij}) a_{ij} \leq \frac{q}{1 - q} c_i. \tag{4.33}$$

As i may be an arbitrary index, this inequality must be satisfied for all rows.

Let us split the set J_i into three subsets, $J_i = J_{i0} \cup J_{i1} \cup J_{i2}$, where

$$\begin{array}{rcll} J_{i0} & = & \{j \in J_i \mid \alpha_{ij} = 0\} & = \{j \in J_i \mid (i,j) \text{ is nonzero only in } C\}, \\ J_{i1} & = & \{j \in J_i \mid \alpha_{ij} \in (0,1)\} & = \{j \in J_i \mid (i,j) \text{ is nonzero in both } B \text{ and } C\}, \\ J_{i2} & = & \{j \in J_i \mid \alpha_{ij} = 1\}, & = \{j \in J_i \mid (i,j) \text{ is nonzero only in } B\}. \end{array} \tag{4.34}$$

Then we can rewrite (4.33) as

$$\sum_{j\in J_{i0}} a_{ij} + \sum_{j\in J_{i1}} (1-\alpha_{ij})a_{ij} \le \frac{q}{1-q} c_i. \tag{4.35}$$

Remark 1. The construction of the matrix B clearly shows that the sparse pattern of the matrix B is at least as sparse as that of A. If any of α_{ij} is set to 0, then it is actually sparser.

Remark 2. From (4.26) we see that $c_i^{(1)} \ge c_i$. Moreover, we should choose $c_i^{(1)}$ to be the largest possible value, i.e. the right boundary of the segment (4.31),

$$c_i^{(1)} = \frac{1}{1-q} c_i > c_i, \quad i = \overline{1,n}. \tag{4.36}$$

Remark 3. One of the possible algorithms of constructing the splitting is as follows:

for all $i,\ i = \overline{1,n}$ **do**
  Compute $c_i = a_{ii} - \sum_{j\neq i} a_{ij}$;
  Order a_{ij}, $j \neq i$, in increasing order;
  Set $s = \dfrac{q}{1-q} c_i$;
  for all $j \neq i$ **do**
    if $a_{ij} \le s$ **then**
      Set $C_{ij} = a_{ij}$;
      $s = s - a_{ij}$;
    else
      Set $\alpha_{ij} = 1 - s/a_{ij}$;
      Set $s = 0$;
      Set $B_{ij} = -\alpha_{ij} a_{ij}$;
      Set $C_{ij} = (1-\alpha_{ij}) a_{ij}$;
      Break;
    end if

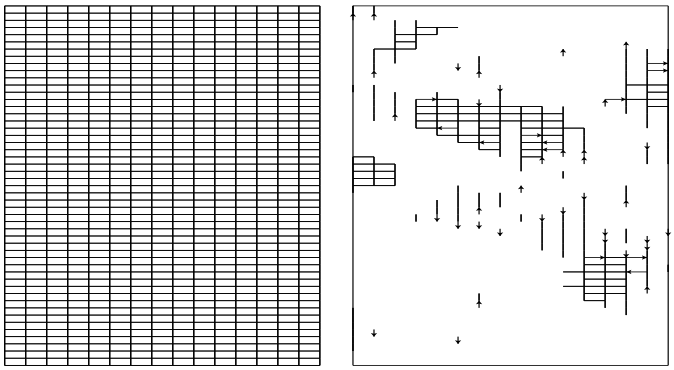

Figure 4.1: Graphs of the diagonal blocks of the matrices A (left) and A_1 (right) for a "horizontal" mesh plane

16: **end for**

17: Set $B_{ii} = \dfrac{1}{1-q} c_i - \sum_{j \neq i} (M)_{ij}$;

18: Set $B_{ii} = \dfrac{1}{1-q} c_i - c_i - \sum_{j \neq i} (N)_{ij}$;

19: **end for**

Remark 4. If the smallest off-diagonal value in a row is smaller than the $\frac{q}{1-q}c$, then we may or may not perform the splitting. Splitting increases the complexity of C and decreases that of B.

Similar to Section 2.3, we can enhance the performance of the preconditioner by eliminating nodes with few links using Schur elimination procedure.

Fig.4.1 shows the graphs of diagonal blocks of the matrices A_0 and A_1 which correspond to a "horizontal" mesh plane $z = const$. Fig.4.2 shows the graphs of diagonal blocks of the matrices A_0 and A_1 which correspond to a "horizontal" mesh plane $z = const$ after the described procedure.

In the Table 4.1, we show the dimensions of matrices A_k (columns n), and the number of nonzero elements in A_k (columns nnz) for algorithms with and without Schur complement variants for one of test cases.

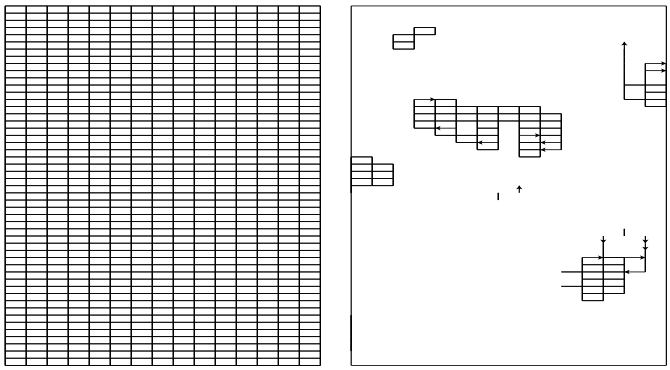

Figure 4.2: Graphs of the diagonal blocks of the matrices A_0 (left) and A_1 (center) for a "horizontal" mesh plane after Schur elimination

4.4 Numerical results

Let us construct matrix A in two steps.

The first step is to construct an intermediate symmetric M-matrix $\widehat{A}$ with strict diagonal domination. This is done by considering the homogeneous Neumann boundary value problem for the diffusion equation

$$\begin{aligned} -\nabla \cdot (K \nabla p) + cp &= f \qquad \text{in} \quad \Omega, \\ (K \nabla p) \cdot \boldsymbol{n} &= 0 \qquad \text{on} \quad \partial\Omega. \end{aligned} \tag{4.37}$$

Here, p is an unknown scalar function (pressure), $K = K(x) \in \mathbb{R}^{3\times 3}$ is a diffusion tensor, c is a positive function, f is a source function, Ω is a domain in $\mathbb{R}^3$, $\partial\Omega$ is the boundary of Ω, and $\boldsymbol{n}$ is the unit outward normal to $\partial\Omega$.

The domain Ω, the diffusion tensor K and reaction coefficient c are chosen the same way as in Section 2.4. We discretize the differential formulation (4.37) by using the classical finite volume method which is known to produce a Stieltjes matrix $\widehat{A}$. Due to the positivity of the reaction coefficient c, the resulting matrix has strict diagonal domination.

Table 4.1: The dimensions (n) and number of nonzero elements (nnz) of matrices A_k

Level	Without Schur		With Schur	
	n	nnz	n	nnz
0	1122000	7780000	1122000	7780000
1	346512	1065149	149319	573233
2	131852	342054	30901	109531
3	31298	71982	4321	13245
4	4068	7767	42	83
5	48	72	-	-

During the second step, we define matrix $A = (A_{ij})$ to be

$$A_{ij} = \begin{cases} (1 - \alpha_{ij})\widehat{A}_{ij}, & j \neq i, \\ \widehat{A}_{ii} + \sum_{j \neq i} \alpha_{ij}\widehat{A}_{ij}, & j = i, \end{cases} \tag{4.38}$$

where coefficients α_{ij} are randomly taken from the interval $[-\alpha, \alpha]$ with $\alpha < 1$. It is clear that using this method produces a matrix A with $A_{ij} \leq 0$ and strict diagonal domination

$$\sum_j A_{ij} = \sum_j \widehat{A}_{ij} > 0.$$

Therefore, the constructed matrix is a non-symmetric M-matrix.

In Table 4.4, we show the dimensions of matrices A_k (columns n) and the number of nonzero elements in A_k (columns nnz) for $q = 0.8$, $\alpha = 0.1$, and different values of c.

In our experiments we compare our preconditioner with a diagonal preconditioner and with the AMG preconditioner. It is important for us to introduce a parameter which would show how efficient a preconditioner is in terms of computa-

Table 4.2: The dimensions (n) and number of nonzero elements (nnz) of matrices A_k for $q = 0.8$ and $\alpha = 0.1$

Level	$c = 0.1$		$c = 1$		$c = 10$	
	n	nnz	n	nnz	n	nnz
0	1122000	7780000	1122000	7780000	1122000	7780000
1	474500	2115125	149319	573233	16993	57657
2	237426	939229	30901	109531	1400	3920
3	63487	230838	4321	13245	7	13
4	10809	36022	42	83	-	-
5	952	2366	-	-	-	-

tional work. We choose this parameter to be the time of calculation of the residual vector, which is equal to 0.023.

The numerical results are presented in Tables 4.3 and 4.4 for two different values of α. The following notations are used:

- t_{const} - the construction time of a preconditioner;
- t_{prec} - the inversion time of a preconditioner;
- $\#iter$ - number of PCG iterations;
- t_{sol} - the solution time of the system;
- t_{total} - the total time to solve the system.

We use notation "Msplit(a,b)" to denote that we construct our preconditioner with $q = a$ and perform b iterations per level.

As usual, we use the PCG method as an outer iterative procedure with the typical stopping criteria

$$\frac{\|A\bar{p}^k - f\|_2}{\|A\bar{p}^0 - f\|_2} \leq \epsilon,$$

where $\epsilon = 10^{-6}$.

Table 4.3: Results of experiments for $\alpha = 0.1$

Prec		t_{const}	t_{prec}	# iter	t_{sol}	t_{total}
$c = 0.1$ ($\gamma = 1000$)	AMG	7.58	0.400	9	3.92	**11.50**
	Diag	0.02	0.003	35829	1068.86	**1068.88**
	Msplit (0.8, 5)	0.53	0.367	128	50.58	**51.12**
	Msplit (0.7, 5)	0.68	2.032	47	94.21	**94.89**
$c = 1$ ($\gamma = 100$)	AMG	6.93	0.409	9	3.92	**10.85**
	Diag	0.02	0.003	6669	204.03	**204.05**
	Msplit (0.8, 5)	0.34	0.047	119	8.66	**9.00**
	Msplit (0.7, 5)	0.39	0.135	50	8.19	**8.58**
$c = 10$ ($\gamma = 10$)	AMG	5.78	0.365	9	3.61	**9.39**
	Diag	0.02	0.003	1176	35.55	**35.57**
	Msplit (0.8, 5)	0.22	0.010	99	3.58	**3.80**
	Msplit (0.7, 5)	0.24	0.014	48	1.95	**2.19**
$c = 100$ ($\gamma = 1$)	AMG	5.10	0.343	8	2.98	**8.08**
	Diag	0.02	0.003	169	5.01	**5.03**
	Msplit (0.8, 5)	0.19	0.006	68	2.21	**2.40**
	Msplit (0.7, 5)	0.19	0.006	40	1.33	**1.52**

Table 4.4: Results of experiments for $\alpha = 0.5$

Prec		t_{const}	t_{prec}	# iter	t_{sol}	t_{total}
	AMG	-	-	-	-	**-**
$c = 0.1$	Diag	0.02	0.003	34527	1067.88	**1067.90**
($\gamma = 1000$)	Msplit (0.8, 5)	0.55	0.407	130	56.42	**56.97**
	Msplit (0.7, 5)	0.68	2.204	47	104.92	**105.61**
	AMG	-	-	-	-	**-**
$c = 1$	Diag	0.02	0.003	6183	186.16	**186.18**
($\gamma = 100$)	Msplit (0.8, 5)	0.34	0.051	122	9.30	**9.64**
	Msplit (0.7, 5)	0.40	0.159	50	9.13	**9.53**
	AMG	5.85	0.371	11	4.53	**10.38**
$c = 10$	Diag	0.02	0.003	1110	33.46	**33.48**
($\gamma = 10$)	Msplit (0.8, 5)	0.22	0.011	103	3.95	**4.17**
	Msplit (0.7, 5)	0.24	0.016	49	2.09	**2.33**
	AMG	5.17	0.351	10	3.80	**8.97**
$c = 100$	Diag	0.02	0.003	167	5.04	**5.06**
($\gamma = 1$)	Msplit (0.8, 5)	0.19	0.006	69	2.35	**2.55**
	Msplit (0.7, 5)	0.19	0.007	40	1.33	**1.53**

Bibliography

[1] J. E. Aarnes, *Efficient domain decomposition methods for elliptic problems arising from flows in heterogeneous porous media*, Comput. Visual. Sci., 8(2) pp. 93–106, 2005.

[2] Y. Achdou and Yu. Kuznetsov, *Substructuring preconditioners for finite element methods on non-matching grids*, East-West J. Numer. Math., 3 pp. 1–28, 1995.

[3] O. Axelsson, *A survey of algebraic multilevel iteration methods*, BIT Numerical Mathematics, 43 pp. 863-879, 2003.

[4] O. Axelsson, *An algebraic multilevel preconditioning methods. I*, Numer. Math., 56 pp. 157–177, 1989.

[5] G. Astrachancev, *An iterative method for solving elliptic net problems*, USSR Computational Math. and Math. Phys., 11(2) pp. 171–182, 1971.

[6] N. S. Bachvalov, *On the convergence of relaxation method with natural constrains of the elliptic operator*, USSR Computational Math. and Math. Phys., 6(5) pp. 101–135, 1966.

[7] C. Bernardi, *Optimal finite-element interpolation on curved domains*, SIAM J. Numer. Anal., 5 pp. 1212–1240, 1989.

[8] C. Bernardi, Y. Maday, and A. Patera, *A new nonconforming approach to domain decomposition: the mortar element method*, Nonlinear Partial Differential Equations and Their Applications (Eds. H. Brézis, and J.-L. Lions), Pitman, Boston, pp. 269–286, 1994.

[9] C. Bernardi, Y. Maday, and G. Sacchi-Landriani, *Non-conforming matching conditions for coupling spectral finite element methods*, Appl. Numer. Math., 54 pp. 64–84, 1989.

[10] J. H. Bramble and M. Zlámal, *Triangular elements in the finite element method*, Math. Comp., 112 pp. 809–820, 1970.

[11] F. Brezzi, J. Douglas Jr., and L. D. Marini, *Two families of mixed finite elements for second order elliptic problems*, Calcolo, 26 pp. 135–148, 1989.

[12] O. Boyarkin, I. Kapyrin, Yu. Kuznetsov, and N. Yavich, *Numerical analysis of a two-level preconditioner for diffusion equations with anisotropic coefficients*, Russian Journal of Numerical Analysis and Mathematical Modeling, 22(4) pp. 377–392, 2007.

[13] D. Braess, *Towards algebraic multigrid for elliptic problems of second order*, Computing, 55 pp. 379–393, 1995.

[14] J. Bramble and X. Zhang, *Uniform convergence of the multigrid V-cycle for an anisotropic problem*, Mathematics of Computation, 70(234) pp. 453–470, 2000.

[15] A. Brandt, *Multilevel adaptive techniques for fast numerical solution to boundary value problems*, In: *Proceedings of the Third International Conference of Numerical Methods in Fluid Dynamics, 1972*, Lecture Notes in Physics, Springer-Verlag, 18 pp. 82–89, 1973.

[16] A. Brandt, *Multilevel adaptive solution to boundary value problems*, Math. Comput., 31 pp. 333–390, 1977.

[17] A. Brandt, *Algebraic multigrid theory: the symmetric case*, Appl. Math. Comput., 19 pp. 23–56, 1986.

[18] F. Brezzi and M. Fortin, *Mixed and Hybrid Finite Element Methods*, Springer-Verlag, 1991.

[19] T. F. Chan and T. P. Mathew, *Domain decomposition algorithms*, Acta Numerica, 3 pp. 61–143, 1994.

[20] A. Cleary, R. Falgout, V. Henson, J. Jones, T. Manteuffel, S. McCormick, G. Miranda, and J. Ruge, *Robustness and scalability of algebraic multigrid*, SIAM J. Sci. Comput, 21 pp. 1886–1908, 1998.

[21] Z. Chen, G. Huan, and Y. Ma, *Computational Methods for Multiphase Flows in Porous Media*, SIAM, Philadelphia, 2006.

[22] T. Clees and K. Stüben, *Algebraic multigrid for industrial semiconductor device simulation*, In: *Proceedings of the First International Conference on Challenges in Scientific Computing, Berlin, Germany, Oct 2-5, 2002*, Lecture Notes in Computational Science and Engineering, Springer, 35, pp. 110–130, 2003.

[23] R. W. Clough, *The finite element method in plane stress analysis*, In: *Proceedings Second ASCE Conference on Electronic Computation*, Pittsburg, PA, 1960.

[24] R. Courant, *Variational methods for the solution of problems of equilibrium and variations*, Bull. Amer. Math. Comp., 64, pp. 1–23, 1943.

[25] J. E. Dendy, *Black box multigrid*, J. Comp. Physics, 48 pp. 366–386, 1982.

[26] R. Eymard, Th. Gallouët, and R. Herbin, *Finite Volume Methods*, volume VII of *Handbook of Numerical Mathematics*, Elsevier, 2000.

[27] R. Glowinski, Q. V. Dinh, and J. Periaux, *Domain decomposition methods for nonlinear problems in fluid dynamics*, Comp. Meth. Appl. Mech. Engng., 40 pp. 27–109, 1983.

[28] W. Hackbusch, *Implementation of the multi-grid method for solving partial differential equations*, Technical Report RA 82, IBM T.J. Watson Research Center, 1976.

[29] W. Hackbusch, *The Frequency Decomposition Multi-Grid Method. Part I: Application to Anisotropic Equations*, Numer. Math., 56 pp 229–245, 1989.

[30] J. Hyman, J. Morel, M. Shashkov, and S. Steinberge, *Mimetic finite difference methods for diffusion equations*, Comp. Geosciences, 6(3-4) pp. 332–352, 2002.

[31] R. P. Fedorenko, *A relaxation method for elliptic difference equations*, USSR Computational Math. and Math. Phys., 1(5) pp. 1092–1096, 1961.

[32] R. P. Fedorenko, *The speed of convergence of one iterative process*, USSR Computational Math. and Math. Phys., 4(3) pp. 227–235, 1964.

[33] Yu. Kuznetsov, *Matrix iterative methods in subspaces*. In: *Proceedings of the International Congress of Mathematicians*, 1–2 pp. 1509–1521, Warsaw, 1984.

[34] Yu. Kuznetsov, *Iterative Methods in Subspaces*, Moscow, 1984 (in Russian).

[35] Yu. Kuznetsov, *Multigrid domain decomposition methods*, the Third International Symposium on Domain Decomposition Methods for Partial Differential Equations, SIAM, pp. 290–313, 1989.

[36] Yu. Kuznetsov, *Multigrid domain decomposition methods for elliptic problems*, Computer Methods in Applied Mechanics and Engineering, 75 pp. 185–193, 1989.

[37] Yu. Kuznetsov, *A new parallel algebraic preconditioner*, Journal of Numerical Linear Algebra with Applications, 1(2) pp. 215–225, 1992.

[38] Yu. Kuznetsov, *Efficient iterative solvers for elliptic finite element problems on nonmatching grids*, Russ. J. Numer. Anal. Math. Modelling, 10 pp. 187–211, 1995.

[39] Yu. Kuznetsov, *Two-level preconditioners with projectors for unstructured grids*, Russian J. Numer. Anal. Math. Modeling, 15(3-4) pp. 247–256, 2000.

[40] Yu. Kuznetsov, *Domain decomposition preconditioner for anisotropic diffusion*, Domain Decomposition Methods in Science and Engineering XVII, Springer Berlin Heidelberg, pp. 105–118, 2008.

[41] Yu. Kuznetsov and O. Boyarkine, *New discretizations of the diffusion equation on distorted hexahedral meshes with pinchouts and faults*, ExxonMobil, 2008.

[42] Yu. Kuznetsov and K. Lipnikov, *An efficient iterative solver for a simplified poroelasticity problem*, East-West J. Numer. Math., 8(3) pp. 207–221, 2000.

[43] Yu. Kuznetsov, K. Lipnikov, and S. Lyons, *Mathematical modeling and numerical algorithms for poroelastic problems*, Contemporary Mathematics, 329(6) pp. 191–202, 2003.

[44] Yu. Kuznetsov, K. Lipnikov, and M. Shashkov, *The mimetic finite difference method on polygonal meshes for diffusion-type problems*, Computational Geosciences, 8(4) pp. 301–324, 2004.

[45] Yu. Kuznetsov and G. Marchuk, *Iterative methods and quadratic functional functional*, Nauka, Novosibirsk, 1972.

[46] Yu. Kuznetsov and A. Prokopenko, *A new multilevel algebraic preconditioner for the diffusion equation in heterogeneous media*, Numerical Linear Algebra with Applications, 17 pp. 759–769, 2010.

[47] Yu. Kuznetsov and S. Repin, *New mixed finite element method on polygonal and polyhedral meshes*, Russian Journal of Numerical Analysis and Mathematical Modelling, 18(3) pp. 261–278, 2003.

[48] Yu. Kuznetsov and S. Repin, *Mixed finite element method on polygonal and polyhedral meshes*, In: *Proceedings of the 5th ENUMATH conference, Prague, 2003.*, Springer, pp. 615–622, 2004.

[49] Yu. Kuznetsov and M. F. Wheeler, *Optimal order substructuring preconditioners for mixed finite element methods on nonmatching grids*, East West Journal of Numerical Mathematics, 3 pp. 127–144, 1995.

[50] O. A. Ladyzhenskaia, *Boundary Value Problems of Mathematical Physics*, Springer, New York, 1985.

[51] P. J. Lanzkron, D. J. Rose, and D. B. Szyld, *Convergence of nested classical iterative methods for linear systems*, Numerische Mathematik, 58(1) pp. 685–702, 1990.

[52] S. L. Lyons, R. R. Parashkevov, and X. H. Wu, *A family of H^1-conforming finite element spaces for calculations on 3D grids with pinch-outs*, Numerical Linear Algebra with Applications, 13(9) pp. 789–799, 2006.

[53] S. Margenov and P. Vassilevski, *Algebraic multilevel preconditioning of anisotropic elliptic problems*, SIAM J. Sci. Comput., 15(5) pp. 1026–1037, 1994.

[54] L. Margolin, M. Shashkov, and P. Smolarkiewicz, *A discrete operator calculus for finite difference approximations*, Comput. Meth. Appl. Mech. Engrg., 187 pp. 365–383, 2000.

[55] T. P. A. Mathew, *Domain Decomposition Methods for the Numerical Solution of Partial Differential Equations*, Springer, Berlin, 2008.

[56] S. F. McCormick, *Multigrid Methods*, Philadelphia, SIAM, 1987.

[57] P. A. Raviart and J. M. Thomas, *A mixed finite element method for 2nd order elliptic problems*, In: *Mathematical Aspects of Finite Element Methods* (Eds. I. Galligani and E. Magenes), Springer-Verlag, New York-Berlin, pp. 292–315, 1977.

[58] Y. Saad, *Iterative Methods for Sparse Linear Systems*, SIAM, 2003.

[59] A. A. Samarskii, *The theory of difference schemes*, Marcel Dekker, Inc. New York-Basel, 2001. (Translation from Russian version Teoriya Raznostnych Schem, Moscow, Nauka, 1977).

[60] H. A. Schwarz, *Gesammelte Mathematische Abhandlungen*, Vierteljahrsschrift der Naturforschenden Gesselschaft in Zürich 15, pp. 272–286, 1870.

[61] *Tenth SPE Comparative Solution Project, model 2*, http://www.spe.org/web/csp//datasets/set02.htm

[62] K. Stüben, *Algebraic multigrid: experience and comparisons*. Applied Math. and Comp., 13 (3-4) pp. 419–451, 1983.

[63] K. Stüben, *A review of algebraic multigrid*, Journal of Computational and Applied Mathematics, 128(1-2) pp. 281–309, 2001.

[64] K. Stüben, *Solving Reservoir Simulation Equations*, Ninth International Forum on Reservoir Simulation, Abu Dhabi, United Arab Emirates, 2007.

[65] K. Stüben, P. Delaney, and S. Chmakov, *Algebraic Multigrid (AMG) for Ground Water Flow and Oil Reservoir Simulation*, presented at the Groundwater Modelling Conference "MODFLOW and MORE", Colorado School of Mines, Golden, Colorado, Sept 17-19, 2003.

[66] A. Toselli and O. Widlund, *Domain Decomposition Methods–Algorithms and Theory*, Springer, Berlin, 2005.

[67] U. Trottenberg, C. W. Oosterlee, and A. Schüller, *Multigrid*, Academic Press, 2001.

[68] R. S. Varga, *Matrix Iterative Analysis*, Prentice-Hall, 1962.