


Multiscale Flow Solver for Unstructured Networks

Master Thesis**Author(s):**

Epp, Robert 

Publication date:

2016

Permanent link:

<https://doi.org/10.3929/ethz-a-010877645>

Rights / license:

In Copyright - Non-Commercial Use Permitted

Multiscale Flow Solver for Unstructured Networks

Robert Epp

Master's degree program Mechanical Engineering

Master's Thesis FS 2016

Institut für Fluidodynamik
ETH Zürich

Supervisor: Dr. Karim Khayrat

Professor: Prof. Dr. Patrick Jenny

Abstract

The simulation of multiphase flow problems in highly heterogeneous porous media is computationally demanding due to the different length and time scales involved. By using the concepts of pore network modelling and multiscale flow solvers, an approximate solution is found which represents the most important macroscopic flow structures. In this work, the multiscale restriction-smoothed basis (MsRSB) method for the solution of the pressure distribution in porous media is extended for unstructured pore networks. A new method that creates fully connected support regions without isolated pore bodies is introduced. The implementation is purely algebraic and the geometric locations of the pores are not required for the computation. Consequently, the multiscale method can be used as a black-box solver for many different Poisson-type problems.

It was observed that the method produces results of good quality for numerous test cases. However, unphysical coarse systems with negative transmissibilities can occur for certain scenarios, especially if large transmissibility contrasts on the fine-scale are involved, e.g. due to channels. This can result in approximate pressure solutions that violate the maximum principle and, if an iterative multiscale formulation is used, the solver may converge slowly or even diverge. The performance is substantially improved when either the restriction or the prolongation operator are adapted to the underlying transmissibility field.

Contents

1. Introduction	1
1.1. Pore Network Modelling	1
1.2. Multiscale Solver	3
1.3. Thesis Outline	5
2. Numerical Modelling	6
2.1. Multiscale Formulation	6
2.2. Algebraic Grids of Fine and Coarse Systems	7
2.3. Support Regions	11
2.4. Prolongation Operator	17
2.5. Restriction Operator	20
2.6. Treatment of Boundaries	20
2.7. Iterative Multiscale Formulation	24
2.8. Adapted Grid Partitioning for Networks with Channels and Barriers	25
3. Numerical Results	27
3.1. Heterogeneous Transmissibility Field	27
3.2. Channels	33
3.3. Flow Barriers	36
3.4. Three-Dimensional Pore Networks	38
3.5. Iterative Solver	40
4. Conclusion and Outlook	42
Bibliography	44
A. Direct Numerical Solution of the Prolongation Operator	46
A.1. Numerical Modelling	46
A.2. Numerical Results	48
B. Multilevel MsRSB	52
B.1. Numerical Modelling	52
B.2. Numerical Results	53

1. Introduction

The solution of multiphase flow problems in highly heterogeneous porous media as observed in soils or rock formations is computationally demanding due to the numerous length and temporal scales involved. Usually, the problem cannot be efficiently solved on a high resolution grid by a standard CFD method that either uses a Finite Volume or Finite Element discretisation of the Navier Stokes Equations, or by a Lattice Boltzmann Method. Consequently, the concepts of pore network modelling and multiscale flow solvers are applied to find an approximate solution which represents the most important flow structures.

1.1. Pore Network Modelling

In the following the modelling equations for incompressible two-phase flow in pore networks are briefly reviewed based on [1] to obtain a linear system of equations for the pressure distribution. In pore network modelling, the geometry of the porous media is idealised such that the most important features of the flow are adequately represented on the macroscopic scale. An example of an unstructured two-dimensional pore network is shown in Figure 1.1 (left). Pore bodies are identified as computational nodes with constant pressure and saturation in each time step. The pores are linked by pore throats that are characterised by a hydraulic transmissibility k_{ij} , which corresponds to the reciprocal value of a hydraulic resistance. Note that k_{ij} may vary in large orders of magnitude over short distances.

A visualization of two pore bodies i and j that are connected by a pore throat ij is given in Figure 1.1 (right). The number of adjacent pores connected to each pore i is denoted as coordination number $n_{c,i}$. It is assumed that each fluid in a pore has its own pressure. If source terms are neglected, the flow balance for every pore body reads

$$V_i \frac{\partial S_i^\alpha}{\partial t} + \sum_{j=1}^{n_{c,i}} v_{ij}^\alpha = 0, \quad \alpha = n, w, \quad (1.1)$$

where V_i is the pore body volume, S_i the saturation in pore i and v_{ij} the flow rate in the throat connecting the pores i and j . The superscript index α represents the fluid phases, namely the *wetting* (w) and the *non-wetting* (n) fluid phases. The fluxes are computed from the difference of the pressures p in two neighbouring pores and k_{ij} by using Darcy's law, i.e.,

$$v_{ij}^\alpha = -k_{ij}^\alpha (p_i^\alpha - p_j^\alpha). \quad (1.2)$$

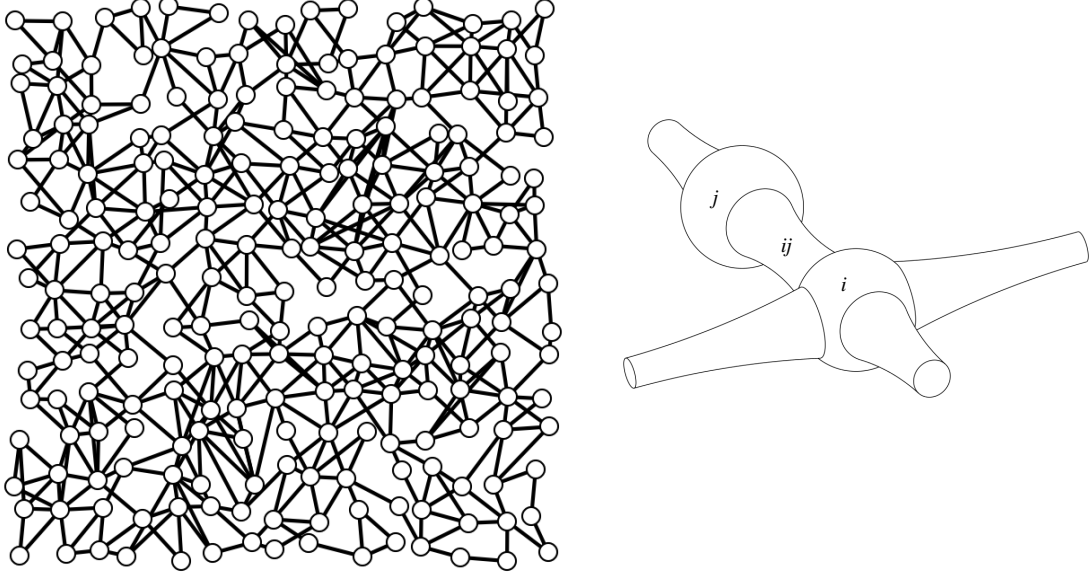


Figure 1.1.: Left: Irregular unstructured two-dimensional pore network. Right: Visualization of two neighbouring pore bodies i and j , and of the connecting pore throat ij .

The hydraulic transmissibility is related to the capillary pressures p^c , i.e. $k_{ij}^\alpha = f(p_i^c, p_j^c)$, which are defined as the pressure differences at the interface between the two fluid phases, i.e.

$$p_i^c = p_i^n - p_i^w. \quad (1.3)$$

This equation is only valid for equilibrium conditions and has to be modified to account for moving interfaces [1]. For a fluid-fluid interface, the capillary pressure is determined by the Young–Laplace equation, i.e.

$$p_i^c = \gamma \left(\frac{1}{r_{min}} + \frac{1}{r_{max}} \right), \quad (1.4)$$

where γ is the surface tension between the two phases and r_{min} and r_{max} are the minimum and maximum radii of curvatures at the interface. However, in the modelling of pore networks p_i^c is usually directly related to S_i by the so-called capillary pressure-saturation relationship, i.e. $p_i^c = f(S_i^w)$ [2]. One last condition is that all pores are completely filled with fluid, i.e.

$$S_i^w + S_i^n = 1. \quad (1.5)$$

Usually, the equations introduced above are reorganised to first solve for a pressure solution based on a saturation field given from the previous time step. The saturations and the transmissibilities

are then updated based on the pressure solution and the procedure is repeated for the subsequent time steps.

To keep things simple, an incompressible single-phase flow is considered in the following. Hence, the flow balance Equation (1.1) reduces to

$$\sum_{j=1}^{n_{c,i}} v_{ij} = q_i, \quad (1.6)$$

where q_i denotes a source term. The flux between two neighbouring pores is

$$v_{ij} = -k_{ij}(p_i - p_j). \quad (1.7)$$

For single phase flow, k_{ij} represents the structure and spatial properties of the network and can vary over several orders of magnitudes. By combining (1.6) and (1.7) a linear system of equations for the pressure is obtained, i.e.

$$\mathbf{A}\mathbf{p} = \mathbf{q}. \quad (1.8)$$

Since the flow is balanced in each pore, \mathbf{A} is a weakly diagonally dominant and symmetrical matrix with a zero row-sum, i.e. $\sum_j A_{ij} = 0$, for all cells except boundaries. A similar elliptic system of equations could be derived for multiphase flow by rearranging the previously introduced equations, or for continuous fluid domains by discretising the variable-coefficient Poisson equation. However, since the subsequent solution steps are independent of the specific physical system and valid for general Poisson-type equations, only the case of a single-phase flow in pore networks is considered in this work.

1.2. Multiscale Solver

Currently, it is unfeasible to solve the system of Equation (1.8) with a direct numerical method for realistic flow scenarios due to computational constraints. Using an iterative relaxation method such as Jacobi, Gauss-Seidel or Successive Over Relaxation (SOR) results in slow convergence of the pressure solution. More advanced methods, such as the Krylov Subspace Methods [3], converge faster. However, the computational cost is still high for large systems. A much higher convergence speed is usually achieved by using an Algebraic Multi Grid (AMG) method. AMG ensures a fast reduction of the short and long-range error components by using a coarsening process of the fine system to obtain algebraic grids on different coarse levels [4]. The fluxes determined from pressure solutions obtained by an iterative method are not necessarily conservative, depending on the level of convergence. However, a conservative velocity field is essential when transport phenomena are relevant and if saturation values are computed in a multiphase simulation.

Different methods have been developed to deal with the challenges arising in the flow simulation of porous media. In upscaling methods, the fine scale system is approximated by a coarser system that represents the large scale features of the solution and can be solved with less computational effort [5]. However, these methods may produce unsatisfying results when the length scales are not clearly separated and in addition, no information regarding the fine scale flow solution is obtained. These limitations are resolved by multiscale methods that allow the reconstruction of an approximate fine scale solution from a result obtained on a coarser scale. Different multiscale modelling approaches exist, such as the multiscale finite element (MsFE) [6], the multiscale finite volume (MsFV) [7] and the multiscale restriction-smoothed basis (MsRSB) [8] methods. The general idea of all these methods is that localised flow problems are solved to obtain basis functions which are used to map quantities between the fine and the coarse scales. Since only the MsFV and the MsRSB methods provide fluxes that are conservative over the coarse cell boundaries, which are used to reconstruct a conservative field for the whole domain, the MsFE method is not further discussed here.

In the MsFV method two overlapping coarse grids are required to locally compute two sets of basis functions that are used to construct the coarse transmissibilities and to reconstruct the fine-scale velocity field from the coarse solution. The basis functions are usually obtained by using a direct numerical method. Although MsFV can be formulated as an algebraic solver [9], the method is primarily suited for structured systems since the fine cells have to be reordered based on the dual-partitioning into interior, face, edge and vertex cells in the implementation. The MsFV was developed further to deal with challenging test cases involving high permeability contrasts by iteratively reducing the errors resulting from the localisation assumption by using global information [10], [11], and to improve convergence speed and robustness in an iterative implementation [12].

The MsRSB method uses an iterative smoothing process to obtain basis functions that are consistent with the local properties of the fine-scale system. The smoothing process is a concept that is related to AMG solvers using smoothed aggregation [13]. In [8] a simple damped Jacobi smoother is used for the computation of the basis function of each block which is restricted to a support region defined by a local triangulation using the centroids and shared face centres of the neighbouring coarse cells. The formulation of MsRSB is algebraic and the method can be applied equally for structured and unstructured grids. In a transient multiphase simulation with changing transmissibilities, only a few iterations on the result from the previous time step are necessary to update the basis functions. In a recent publication, it was shown how the MsRSB can be extended for fractured porous media [14].

1.3. Thesis Outline

The main goal of this Master's thesis is to extend the MsRSB method for unstructured pore networks that are able to represent real porous media. Unlike the implementation described in [8], the method here should be able to cope with unknown pore locations and therefore operate as a black-box solver for the system of equations (1.8).

The thesis is structured as follows. Chapter 2 summarises the general concepts of MsRSB and describes how the method is extended for unstructured pore networks. In Chapter 3, the numerical results of different test cases are presented. Finally, Chapter 4 gives a conclusion of this work and recommendations for future research.

2. Numerical Modelling

This chapter describes the multiscale restriction-smoothed basis (MsRSB) method for unstructured pore networks. The numerical modelling is based on [8], however, the geometric locations of the pores are not required here and the method can operate as a black-box solver for an approximate solution of the pressure distribution.

2.1. Multiscale Formulation

The solution of Equation (1.8) is represented on a fine-scale grid $\{\Omega_i^F\}_{i=1}^n$. The multiscale method starts with the partitioning of the fine grid into multiple non-overlapping blocks that form a coarse grid $\{\Omega_j^C\}_{j=1}^m$. Quantities associated with the coarse grid are interpolated to the fine grid with a prolongation operator, i.e. $\mathbf{P} : \Omega^C \rightarrow \Omega^F$. Analogously, a restriction operator is used for the mapping from the fine grid to the coarse grid, i.e. $\mathbf{R} : \Omega^F \rightarrow \Omega^C$.

The basic principle of every multiscale method is to first solve for a coarse solution \mathbf{p}^c on Ω^C and then to prolongate the coarse solution back to the fine scale to get an approximate solution \mathbf{p}^{approx} of \mathbf{p} , i.e.

$$\mathbf{p}^{approx} = \mathbf{P}\mathbf{p}^c. \quad (2.1)$$

A linear system of equations for the computation of \mathbf{p}^c is derived by inserting \mathbf{p}^{approx} into Equation $\mathbf{A}\mathbf{p} = \mathbf{q}$ (1.8) and applying the restriction operator to map the system to the coarse scale,

$$\mathbf{R}\mathbf{A}(\mathbf{P}\mathbf{p}^c) = \mathbf{R}\mathbf{q}. \quad (2.2)$$

This system can be rewritten to

$$\mathbf{A}^c \mathbf{p}^c = \mathbf{q}^c, \quad (2.3)$$

with $\mathbf{A}^c = \mathbf{R}\mathbf{A}\mathbf{P}$ and $\mathbf{q}^c = \mathbf{R}\mathbf{q}$ representing the coarse-scale system matrix and the coarse source terms, respectively. In general, this system is much smaller than the original fine-scale system and can usually be solved with a direct numerical method.

2.2. Algebraic Grids of Fine and Coarse Systems

After combining Equations (2.1) and (2.3) the multiscale method to compute the approximate solution of \mathbf{p} reads

$$\mathbf{p} \approx \mathbf{p}^{approx} = \mathbf{P}(\mathbf{A}^c)^{-1} \mathbf{R} \mathbf{q}. \quad (2.4)$$

Generally, the approximate solution \mathbf{p}^{approx} can significantly deviate from the exact solution \mathbf{p} . Therefore \mathbf{A}^c is usually used as a coarse correction step in an iterative multiscale formulation to solve for \mathbf{p} . The iterative multiscale formulation is presented in Section 2.7.

2.2. Algebraic Grids of Fine and Coarse Systems

In the following, it is shown how Ω^F and Ω^C are directly created from \mathbf{A} . A useful approach to get the topological characterisation of a system is its graph representation. A graph $\mathcal{G}^F(V^F, E^F)$ with the vertices V^F and the edges E^F of the fine-scale system is obtained by interpreting \mathbf{A} as the adjacency matrix of the graph. Every non-zero matrix entry $k_{ij} \neq 0$ of \mathbf{A} defines an edge between the vertices with indices i and j [15]. For certain applications it is advantageous to consider a weighted graph representation. Therefore the edge weights are set to the values of the corresponding transmissibilities k_{ij} . For symmetrical \mathbf{A} , \mathcal{G}^F is an undirected graph with $k_{ij} = k_{ji}$ and only the strictly upper triangular matrix of \mathbf{A} is used for the creation of \mathcal{G}^F . A simple illustration of how a graph is created from a system matrix is given in Figure 2.1.

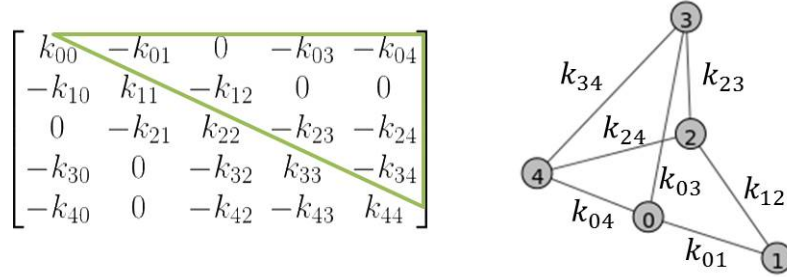


Figure 2.1.: System matrix \mathbf{A} and its corresponding graph representation.

The vertices of \mathcal{G}^F are now associated with the grid points Ω_i^F and the edges with the connections between the different grid cells of the fine-scale system. Hence, Ω^F is completely defined by \mathcal{G}^F , and therefore by \mathbf{A} .

Similarly, the coarse-scale grid Ω^C corresponds to a coarse graph $\mathcal{G}^C(V^C, E^C)$. To determine \mathcal{G}^C , \mathcal{G}^F is first partitioned into multiple coarse blocks. For the partitioning process, a variety of graph based algorithms can be applied. The partitioner should produce approximately equally sized blocks and reduce the number of edges connecting the different partitions. Here, algorithms from the partitioning software Metis were used to efficiently produce blocks of good quality [16]. After the partitioning each vertex with index i belongs to a unique coarse block, i.e. $i \in C_j$, with C_j denoting the set of fine scale vertices that belong to coarse block j . The coarse graph \mathcal{G}^C can

2.2. Algebraic Grids of Fine and Coarse Systems

now be created by merging the fine-scale vertices $V_j = \{i \mid i \in C_j\}$ to a unique coarse vertex for each block j and by the subsequent removal of multiple edges between the new coarse vertices. Figure 2.2 shows \mathcal{G}^F and \mathcal{G}^C for a structured two-dimensional system with 81 solution unknowns and 9 coarse cells with every vertex being connected to 8 other vertices except at the boundaries. Here, the different colours and indicator values designate partition membership.

It is observed that the multipoint stencils of \mathcal{G}^C and \mathcal{G}^F are identical for a regular structured grid. The connections between vertices on the coarse scale are later used to define the support regions of the basis functions. Therefore, all blocks that are in the close neighbourhood of each other should be connected in \mathcal{G}^C . For a nine-point stencil discretisation of the fine-scale system all neighbouring coarse blocks are automatically correctly connected in \mathcal{G}^C .

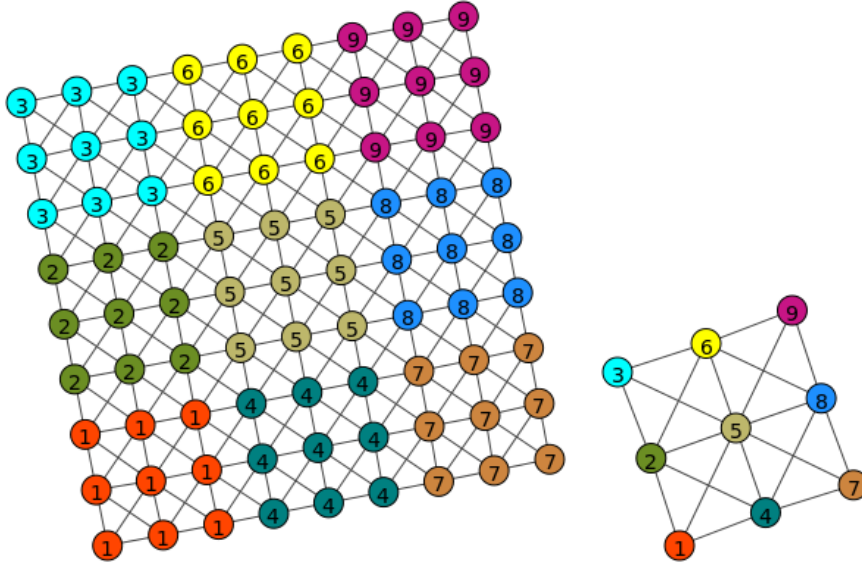


Figure 2.2.: Graph representation of the fine and coarse systems \mathcal{G}^F and \mathcal{G}^C for a structured nine-point stencil discretisation.

An example of a structured fine-scale system with a five-point stencil discretisation is shown in Figure 2.3. Following the same procedure to obtain \mathcal{G}^C as described above, not all blocks are connected on the coarse scale although they are in the close vicinity of each other. For example, the coarse vertices with indices $j = 3$ and $j = 5$ are not directly connected in \mathcal{G}^C , since there are no connections between the corresponding fine-scale vertices in \mathcal{G}^F . It is advantageous for the multiscale method to also connect these vertices in \mathcal{G}^C . Therefore, additional edges denoted as *weak links* are subsequently added to the coarse graph as indicated with dashed lines in Figure 2.3. The weak links are added to \mathcal{G}^C based on a neighbourhood search on the fine-scale that identifies other blocks in the vicinity of each block. First, the fine-scale vertices within a topological distance R from each coarse partition are identified. A weak link is then added to \mathcal{G}^C if the so found fine-scale vertices belong to a coarse block that is not already connected in \mathcal{G}^C . The parameter R depends on the size of the system and should be chosen appropriately. For the

2.2. Algebraic Grids of Fine and Coarse Systems

example shown in Figure 2.3 a value of $R = 2$ was chosen. For $R = 1$, no additional weak links would have been added to the system.

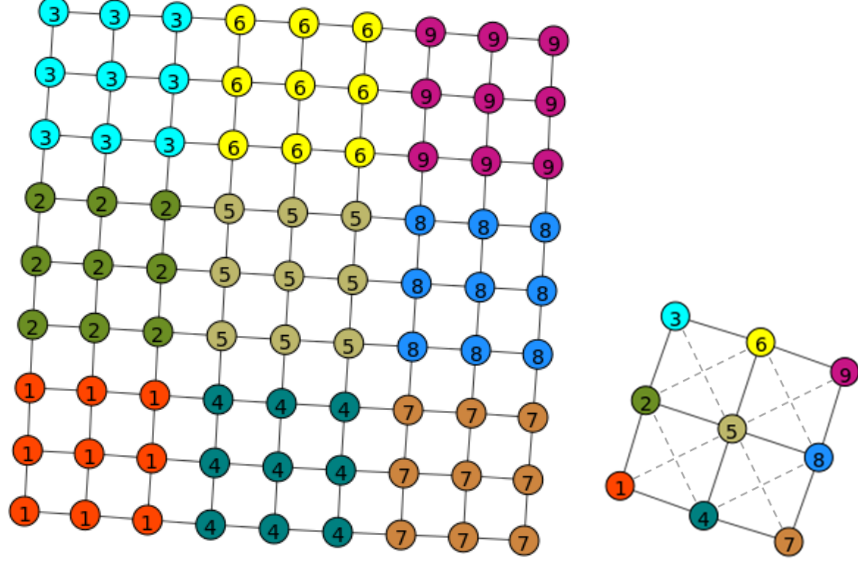


Figure 2.3.: Graph representation of the fine and coarse systems \mathcal{G}^F and \mathcal{G}^C for a structured five-point stencil discretisation. For \mathcal{G}^C , the subsequently added weak links are illustrated with dashed lines.

An example of the neighbourhood search for each coarse block j is presented in Algorithm 1. Here, v and v_0 denote a set of fine-scale vertices, i.e. $v, v_0 \subseteq V^F$, and N_j is the set of indices of the neighbouring coarse blocks of j . The function $neighbours(v)$ returns the adjacent fine scale vertices of v .

The above described procedure to get \mathcal{G}^F and \mathcal{G}^C is valid for general unstructured and three-dimensional systems. An example for an unstructured two-dimensional system with 81 solution unknowns and 9 coarse cells is shown in Figure 2.4. Here, a value $R = 2$ was chosen to obtain \mathcal{G}^C .

Algorithm 1 Adding weak links to \mathcal{G}^C

```

1: for each coarse block  $j$  do
2:    $v_0 = \{i \mid i \in C_j\}$ 
3:    $v = \{v_0 \mid \text{neighbours}(v_0) \notin C_j\}$ 
4:    $r = 0$ 
5:   while  $r < R$  do
6:      $v \leftarrow v \cup \text{neighbours}(v)$ 
7:      $v = \{v \mid v \notin C_j\}$ 
8:      $r = r + 1$ 
9:    $N_j = \{k \mid i \in C_k \wedge i \in v\}$ 
10:  for all  $k \in N_j$  do
11:    if Coarse vertices with indices  $j$  and  $k$  are not yet connected then
12:      Add weak link between coarse vertices with indices  $j$  and  $k$ 

```

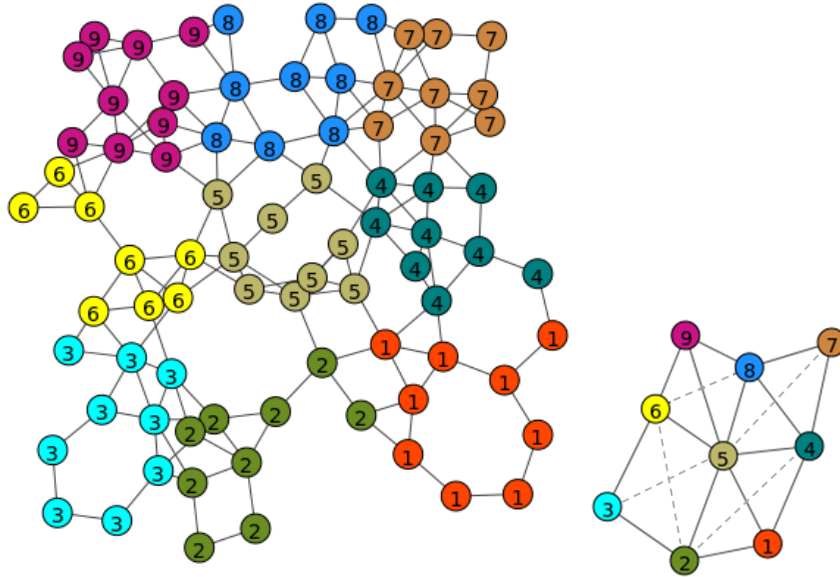


Figure 2.4.: Graph representation of the fine and coarse systems \mathcal{G}^F and \mathcal{G}^C for an unstructured two-dimensional system. The weak links are illustrated with dashed lines.

2.3. Support Regions

The prolongation operator \mathbf{P} is formed by piecing together a set of basis functions Φ_j which are specified for each coarse cell Ω_j^C , i.e.,

$$P_{ij} = \Phi_j(\Omega_i^F), \quad (2.5)$$

with Ω_i^F being the fine grid cell with index i . In the MsRSB method *support regions* I_j are defined to determine the support of Φ_j . Here, I_j denotes the fine cells that are contained in the support region of coarse cell Ω_j^C . Basis functions have non-zero values only in the support region,

$$\Phi_j(\Omega_i^F) > 0, \quad \Omega_i^F \in I_j. \quad (2.6)$$

The support region I_j can be constructed by creating a local triangulation of the block centroids and shared-face centres of all neighbouring blocks of Ω_j^C [8]. However, in pore networks geometrical neighbouring pores are not necessarily connected by pore throats, as can be easily observed in Figure 1.1 (left). Therefore, a triangulation would frequently produce disconnected support regions with isolated vertices. Furthermore, if the locations of the fine cells are unknown the triangulation cannot be created and therefore this method is not suited for a black-box implementation. Another problem is that only convex support regions can be easily created with a Delaunay triangulation. A new method to create I_j directly from the graphs \mathcal{G}^F and \mathcal{G}^C will be introduced later in this section. For convenience, the *support boundary* B_j , the *global support boundary* G and the *coarse cell centres* M_j are defined first.

B_j contains the fine cells that are topological neighbours to the outermost cells of I_j so that $B_j \cap I_j = \emptyset$, and G is the union of all cells that are in B_j of at least one block, i.e.

$$G = \bigcup_{j=1}^m B_j. \quad (2.7)$$

The centres M_j are determined by using the graph representation \mathcal{G}^F of the system matrix \mathbf{A} . For each coarse partition j a fine-scale subgraph $\mathcal{G}_j^{sub}(V_j^{sub}, E_j^{sub})$ with the subgraph vertices V_j^{sub} , i.e.

$$V_j^{sub} = \{i \mid i \in C_j \wedge i \in V^F\}, \quad (2.8)$$

and edges E_j^{sub} , i.e.

$$E_j^{sub} = \{(i, k) \mid i \in C_j \wedge k \in C_j \wedge (i, k) \in E^F\}, \quad (2.9)$$

2.3. Support Regions

is formed and based on that, the relative closeness centralities of the fine cells are computed. The relative closeness centrality of a vertex is a measure to quantify how easily it can be reached from the other vertices and is defined as $RC_i = (n-1)/\sum_{k=1}^n d_{ik}$, with d_{ik} being the unweighted shortest path length from vertex i to vertex k and n being the number of vertices in the graph [17]. The coarse cell centre M_j of each block with n_j fine cells is then specified to be the fine cell with the largest closeness centrality of \mathcal{G}_j^{sub} , i.e.

$$M_j = \arg \max_{i \in C_j} \left\{ \frac{(n_j - 1)}{\sum_{k \in C_j} d_{ik}} \right\}. \quad (2.10)$$

If \mathcal{G}_j^{sub} has disconnected vertices, it is necessary to ensure that M_j belongs to the connected part of the subgraph. This can be achieved by setting d_{ik} of a disconnected vertex pair i and k to a value that is higher than the maximum shortest path occurring in \mathcal{G}_j^{sub} , e.g. by setting it to n_j . However, it should be ensured that the partitions do not have too many disconnected vertices in the first place. The method could be extended to determine M_j based on a weighted closeness centrality which was not considered here.

In the following, the method to find I_j and B_j that only requires the graph representations \mathcal{G}^F and \mathcal{G}^C of \mathbf{A} is introduced. In a first step, a temporary support region s_j is created that initially includes all fine scale vertices of coarse cell Ω_j^C , i.e.

$$s_j = \{i \mid i \in C_j\}. \quad (2.11)$$

The temporary support s_j is then grown into the neighbouring coarse blocks of Ω_j^C , which are directly identified by the coarse graph \mathcal{G}^C . Here, coarse vertices that are connected through edges, including the ones connected through *weak links*, are considered to be neighbouring blocks. The growing is done by repeatedly adding the neighbouring fine vertices to s_j , i.e.

$$s_j \leftarrow s_j \cup \text{neighbours}(s_j), \quad (2.12)$$

until a fine scale centre cell M_k , $k \neq j$ in the coarse neighbour block Ω_k^C is reached and the growing in the corresponding block Ω_k^C is stopped. The growing then continues only in the other blocks, until all centres of the coarse neighbourhood are reached. Finally the outermost vertices of s_j are assigned to B_j and all others to I_j . In a subsequent step it should be ensured that the neighbouring coarse centres M_k , $k \neq j$ are always in B_j and not in I_j , as it may occur in complex unstructured and three-dimensional systems. The so determined support regions are fully connected and have no isolated vertices which is an advantage compared to the support regions obtained by local triangulation in [8].

An example algorithm for the creation of the support regions is presented in Algorithm 2. Here, N is the index set of neighbouring coarse vertices, $v_{centres}$ the set of neighbouring (fine scale) block centre vertices, s_0 the vertices from the initial temporary support region, s the temporary support region and s_{new} the vertices which are currently being added to s . In the implementation

2.3. Support Regions

the growing of each support region is done on a local subgraph that only includes vertices from the current coarse block and from its coarse neighbours. Therefore, this part of the code can be easily parallelised in the future.

Algorithm 2 Creating the support regions

```

1: for each coarse partition  $j$  do
2:    $N \leftarrow \text{coarse neighbours}(j)$ 
3:    $v_{centres} = \{M_k \mid k \in N\}$ 
4:    $s_0 = \{i \mid i \in C_j\}$ 
5:    $s \leftarrow s_0$ 
6:    $s_{new} = \{s \mid \text{neighbours}(s) \notin C_j\}$ 
7:   while  $\text{len}(N) \neq 0$  do
8:     if  $v_{centres,k} \in s$  then
9:        $N \leftarrow N \setminus N_k$ 
10:     $s_{new} \leftarrow \text{neighbours}(s_{new})$ 
11:     $s_{new} \leftarrow s_{new} \in N$ 
12:     $s \leftarrow s \cup s_{new}$ 
13:     $B_j = \{s \mid \text{neighbours}(s) \notin s\}$ 
14:     $I_j = s \setminus B_j$ 

```

In the following, the method is illustrated based on two examples which are introduced in Figure 2.5. The examples are both two-dimensional and the vertices are represented on an equidistant grid. In the first example the edges form a nine-point stencil and the grid is partitioned into nine structured coarse cells (left), whereas in the second example an unstructured network and partitioning is used (right). In the figure the coarse cells are indicated with different colours and the coarse centres are marked with a large red dot. For the structured example, the stepwise growth of s_j is visualized in Figure 2.7 for the centre block. In the top left subfigure s_j only includes the vertices from the centre block. The other subfigures then show the iterative growth of s_j into the neighbouring coarse partitions, until all coarse centres are reached. In the last figure the resulting I_j (blue) and B_j (yellow) are shown. The same is shown for the unstructured system in Figure 2.8. After creating I_j and B_j , the global support region G is determined. In Figure 2.6 the vertices $G \cap B_j$ are highlighted in green.

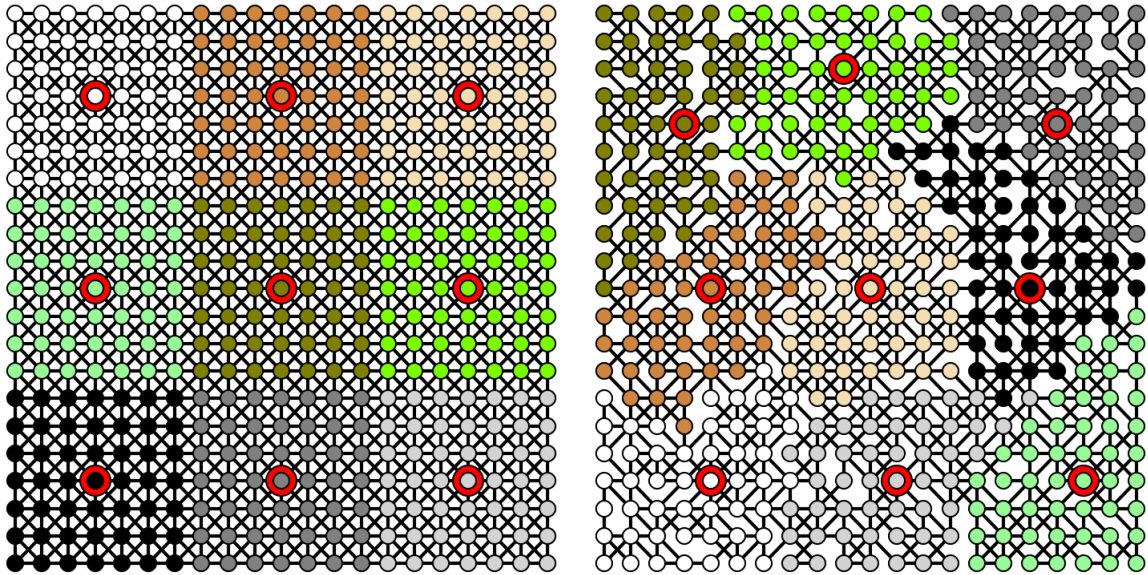


Figure 2.5.: Two examples of pore networks represented on equidistant grids which are both partitioned into nine coarse blocks. The coarse cell centres are highlighted with a large red dot.

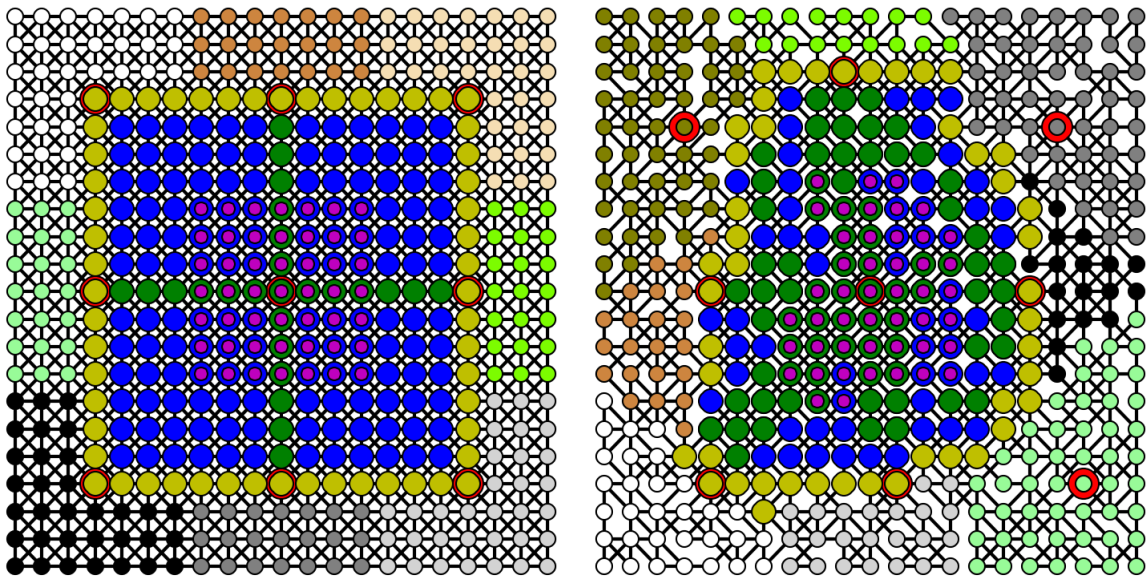


Figure 2.6.: The resulting support regions I_j (blue), support boundaries B_j (yellow) and the global support boundary vertices $G \cap I_j$ (green) illustrated for the centre blocks of the above introduced structured and unstructured networks.

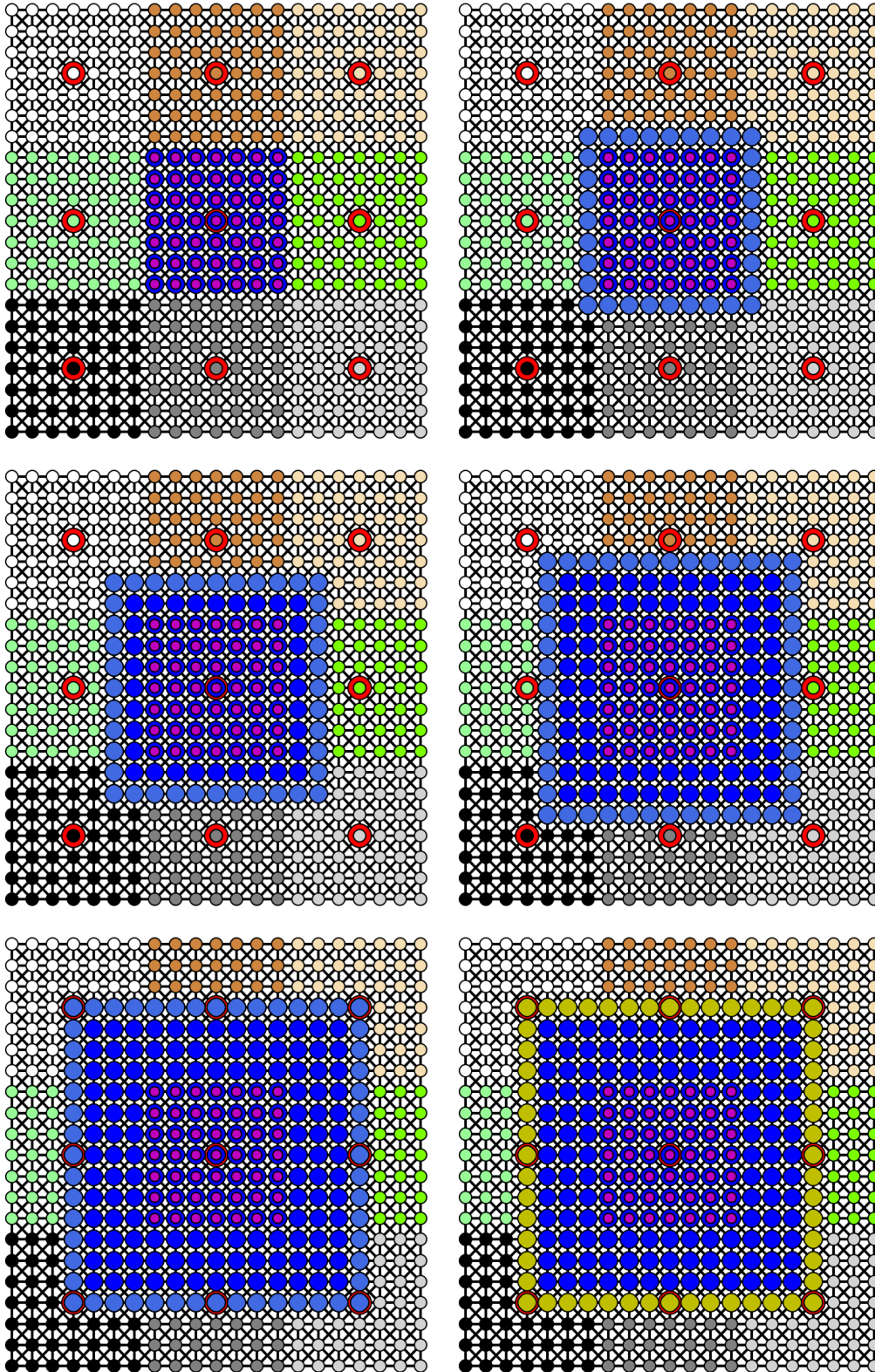


Figure 2.7.: Stepwise growth of the temporary support region for the centre block of a structured network. The initial support region is highlighted in magenta, the current support region in blue and the vertices that are added in every step in light blue. The bottom right figure shows the resulting support region (blue) and support boundary (yellow).

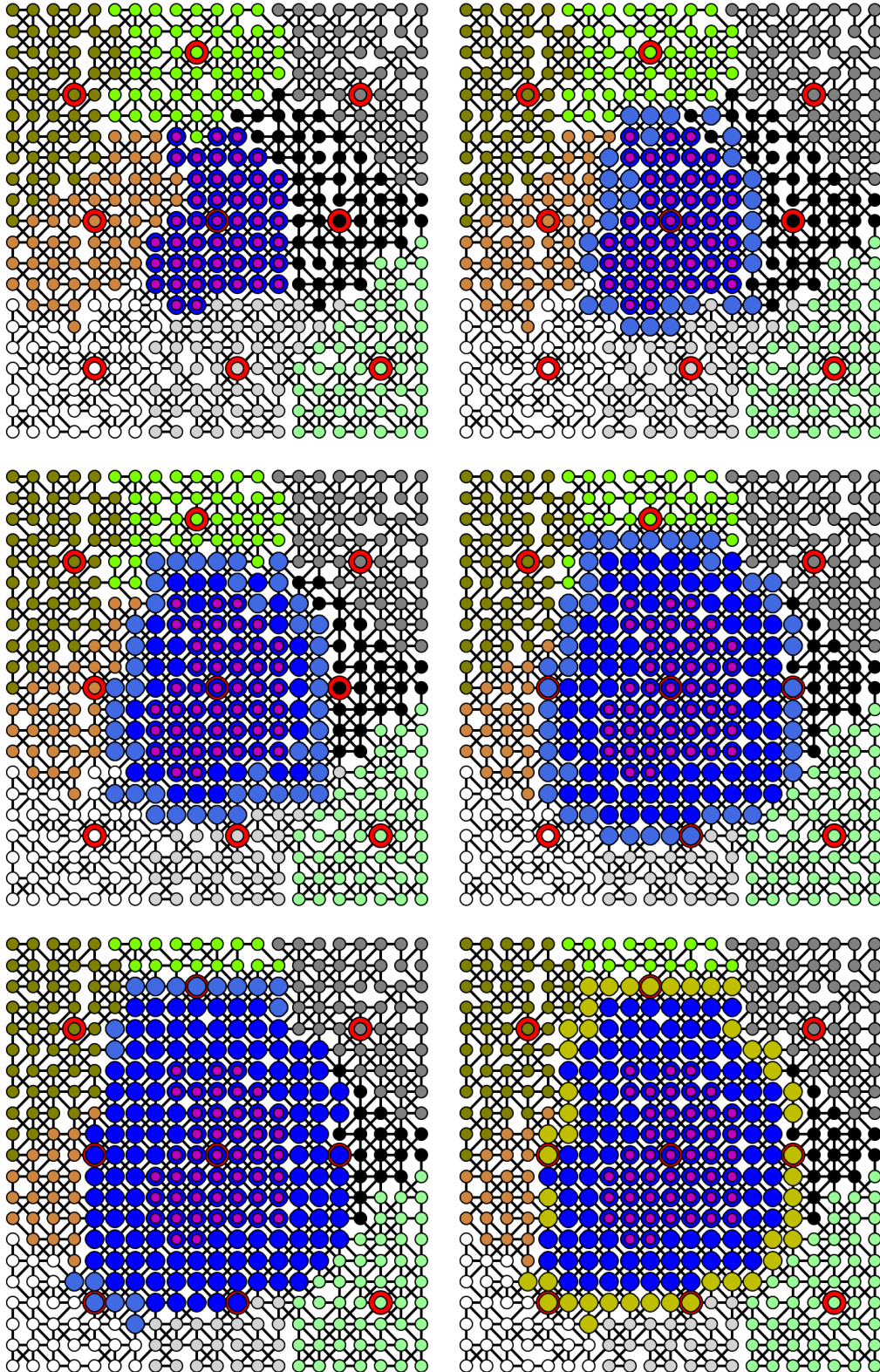


Figure 2.8.: Stepwise growth of the temporary support region for a central block of an unstructured network. The initial support region is highlighted in magenta, the current support region in blue and the vertices that are added in every step in light blue. The bottom right figure shows the resulting support region (blue) and support boundary (yellow).

2.4. Prolongation Operator

The MsRSB method uses an iterative process to compute the basis functions that form the prolongation operator \mathbf{P} [8]. In the following, the process to compute \mathbf{P} is briefly summarised. It is assumed that the system matrix \mathbf{A} has zero row sum, i.e.

$$\sum_j A_{ij} = 0, \quad (2.13)$$

which is usually valid everywhere except at cells adjacent to Dirichlet boundaries, to obtain a prolongation operator that has partition-of-unity, i.e.

$$\sum_j P_{ij} = 1. \quad (2.14)$$

If Equation (2.13) is violated, the diagonal entries of \mathbf{A} are adjusted to ensure zero row sum for the computation of \mathbf{P} [8].

The basis functions are initialised by setting them to a value of one inside the corresponding block and to zero otherwise, i.e.

$$P_{ij}^{(0)} = \begin{cases} 1, & \text{if } i \in C_j \\ 0, & \text{otherwise.} \end{cases} \quad (2.15)$$

Then, a smoothing step is applied to get the iterative increments of the basis functions, i.e.

$$\hat{\mathbf{d}} = -\omega \mathbf{D}^{-1} \mathbf{A} \mathbf{P}^{(n)}, \quad (2.16)$$

where \mathbf{D} is the diagonal matrix of \mathbf{A} and ω a relaxation parameter. Throughout this work, ω is set to 2/3, similar to [8]. In every iteration, the cell values of the basis functions are modified based on topological neighbouring cells. Hence, the basis functions will eventually have non-zeros values in the entire network. To enforce the basis functions to have support in the corresponding support regions only, $\hat{\mathbf{d}}$ is modified in an additional step. For convenience, a sum of the increments $\hat{\mathbf{d}}$ is first computed including all fine cells in G , i.e.

$$\hat{d}_{sums,i} = \sum_j \hat{d}_{ij}, \quad i \in I_j, i \in G. \quad (2.17)$$

2.4. Prolongation Operator

Note that the increment values of cells in B_j are not included in the computation of the sum. The increment is then modified by setting \hat{d}_{ij} outside I_j to zero and scaling the values on G to preserve partition-of-unity in the whole network, i.e.

$$d_{ij} = \begin{cases} \frac{\hat{d}_{ij} - P_{ij}^{(n)} \hat{d}_{sums,i}}{1 + \hat{d}_{sums,i}} & i \in I_j, i \in G, \\ \hat{d}_{ij}, & i \in I_j, i \notin G, \\ 0, & i \notin I_j. \end{cases} \quad (2.18)$$

The prolongation operator is then updated, i.e.

$$\mathbf{P}^{(n+1)} = \mathbf{P}^{(n)} + \mathbf{d}. \quad (2.19)$$

The convergence of \mathbf{P} is measured by computing the maximum value of d_{ij} outside G , i.e.

$$d_{max} = \max(|d_{ij}|), \quad i \notin G. \quad (2.20)$$

The prolongation operator has converged if $d_{max} < d_{crit}$. If not, $\mathbf{P}^{(n)}$ is set to $\mathbf{P}^{(n+1)}$ and the iteration steps (2.16) - (2.20) are repeated. The converged basis functions of the centre block for the networks from Figure 2.5 with constant transmissibilities are shown in Figure 2.9, represented on a structured grid.

In a transient multiphase flow simulation the transmissibilities of the flow network may change in every time step, and therefore the system matrix and the prolongation operator need to be updated continuously. The computation of $\mathbf{P}(t)$ at time step t can be done by using the result of the previous time step $\mathbf{P}(t-1)$ as an initial guess for $\mathbf{P}^{(0)}(t)$, i.e. $\mathbf{P}^{(0)}(t) = \mathbf{P}^{converged}(t-1)$. The updating procedure usually converges after a few iteration steps.

In the implementation, the prolongation operator \mathbf{P} is a large sparse matrix with zero entries everywhere except for the cells on the support region. Since the values outside I_j are always set to zero, the smoothing step (2.16) only affects the basis functions on the corresponding support regions and support boundaries. To reduce the computational costs the steps (2.16), (2.18) and (2.19) can be performed separately for every basis function $\Phi_j = P_j$ on local subsystems \mathbf{A}_j of \mathbf{A} that involve the cells $\{\Omega_i^F \mid i \in I_j \cup B_j\}$. However, the sums from step (2.17) depend on values from different basis functions and have to be computed globally. Therefore, the benefits of parallelising the computation are limited due to the required processor communication in every iteration step.

The iterative method is computationally expensive especially if \mathbf{P} is computed from scratch and if the tolerance criteria to stop the iteration is low, as usually required for highly heterogeneous and anisotropic flow networks, e.g. for domains involving channels. The smoothing step (2.16) resembles a Jacobi iterative solver which has slow convergence properties. Switching to an iterative method with faster convergence, e.g. a Gauss-Seidel solver, or using multiple smoothing

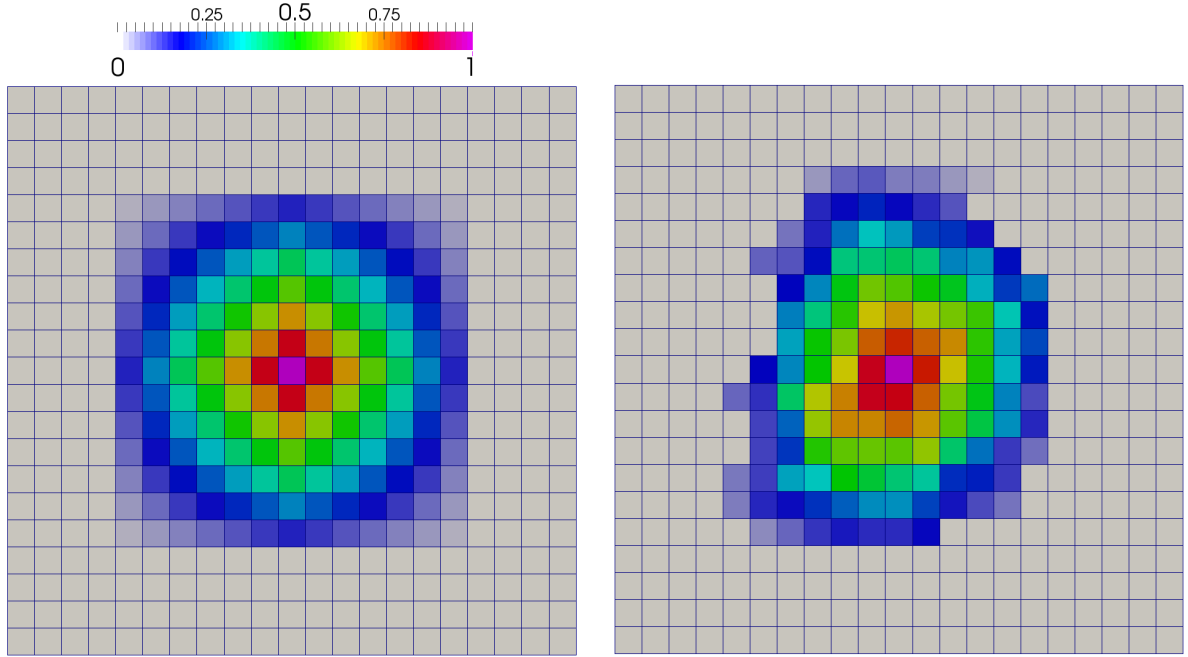


Figure 2.9.: Basis functions for the centre blocks of the above introduced structured and unstructured pore networks.

cycles before applying the correction step (2.18), is not straightforward, since the computation of the update depends solely on the previous iteration step to maintain partition-of-unity.

Alternatively, one could think of using a direct method to compute the basis functions Φ_j , which is discussed in Appendix A. Although reasonable results are obtained for homogeneous systems, the direct method produces unusable prolongation operators for more complex flow scenarios with high transmissibility contrasts due to a normalization step which is necessary to ensure partition-of-unity.

2.5. Restriction Operator

The restriction operator \mathbf{R} is obtained by either using a control volume summation [8], i.e.

$$R_{ji,CV} = \begin{cases} 1, & \text{if } i \in C_j \\ 0, & \text{otherwise,} \end{cases} \quad (2.21)$$

or a Galerkin operator formulation [8], i.e.

$$\mathbf{R}_G = \mathbf{P}^T, \quad (2.22)$$

where \mathbf{P}^T is the transposed matrix of \mathbf{P} .

Usually, \mathbf{R}_{CV} is used to ensure that the fluxes across the border of each coarse cell are conservative. However, using \mathbf{R}_G in the implementation of an iterative multiscale formulation (Section 2.7) improves the convergence speed of the solver. Consequently, \mathbf{R}_{CV} is only applied in the last iteration step to obtain a conservative result on the coarse scale.

2.6. Treatment of Boundaries

In a linear system of equations Dirichlet boundaries are usually defined by locally replacing \mathbf{A} by the identity matrix and specifying the boundary values at the right-hand-side of the system. However, adjusting the diagonal elements of the resulting matrix to get zero row sum (2.13) leads to a singular matrix which cannot be used in the iterative computation of \mathbf{P} . Therefore, Dirichlet boundaries are assigned to virtual cells outside the grid which are connected to the simulation domain through an interface. Hence, the system matrix can easily be modified to ensure zero row sum and be used as an iteration matrix in Equation (2.16). Note that it is best to set the transmissibilities of the interfaces to values that are high enough to not cause a significant pressure drop. For Neumann boundaries, no special treatment of \mathbf{A} is necessary, since the boundaries only affect the right-hand-side of the equations.

On the left of Figures 2.10 and 2.11 the support region and the basis function for a coarse block which is adjacent to the boundary at the east of a structured flow network are shown. If the coarse centre M_j of a boundary block is specified to be the fine cell with the largest closeness centrality of the corresponding block according to (2.10), the resulting basis function remains constant east of the coarse centre towards the boundary to ensure partition-of-unity of the basis functions, which leads to errors in the approximate pressure solution. The quality of the prolongation operator is significantly increased if M_j is moved to a fine cell adjacent to the boundary [8]. Two different cases were considered to specify M_j of a boundary block. For connected boundary fine

cells, Equation (2.10) is modified to only include the closeness centralities of the boundary cells, i.e.

$$M_j = \arg \max_{i \in (C_j \cap BC)} \left\{ \frac{(n_j - 1)}{\sum_{k \in (C_j \cap BC)} d_{ik}} \right\}. \quad (2.23)$$

Here, BC corresponds to the index set of fine cells adjacent to boundaries. Alternatively, if the boundary cells are disconnected, all cells are considered in the computation similarly to (2.10), but M_j is set to be the cell at the boundary with the highest closeness centrality with respect to the whole block, i.e.

$$M_j = \arg \max_{i \in (C_j \cap BC)} \left\{ \frac{(n_j - 1)}{\sum_{k \in C_j} d_{ik}} \right\}. \quad (2.24)$$

The resulting support region and the basis function are shown on the right of Figures 2.10 and 2.11 for the structured system. Moving the coarse centre to the very east of the domain results in a basis function that decays smoothly.

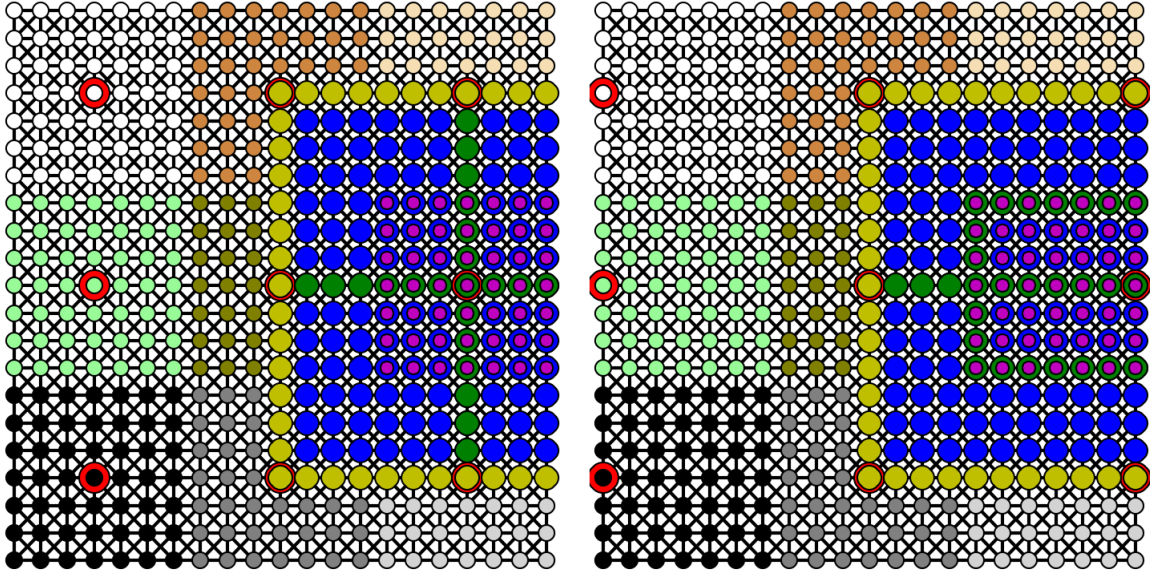


Figure 2.10.: Coarse centre cells M_j (red), support regions I_j (blue), support boundaries B_j (yellow) and the global support boundary vertices $G \cap I_j$ (green) for the boundary blocks of a structured network. In the right image M_j is moved to a cell adjacent to the boundary.

For the sake of completeness, the effect of moving the coarse centres of the above introduced unstructured network to a fine cell adjacent to the boundary is illustrated in Figures 2.12 and

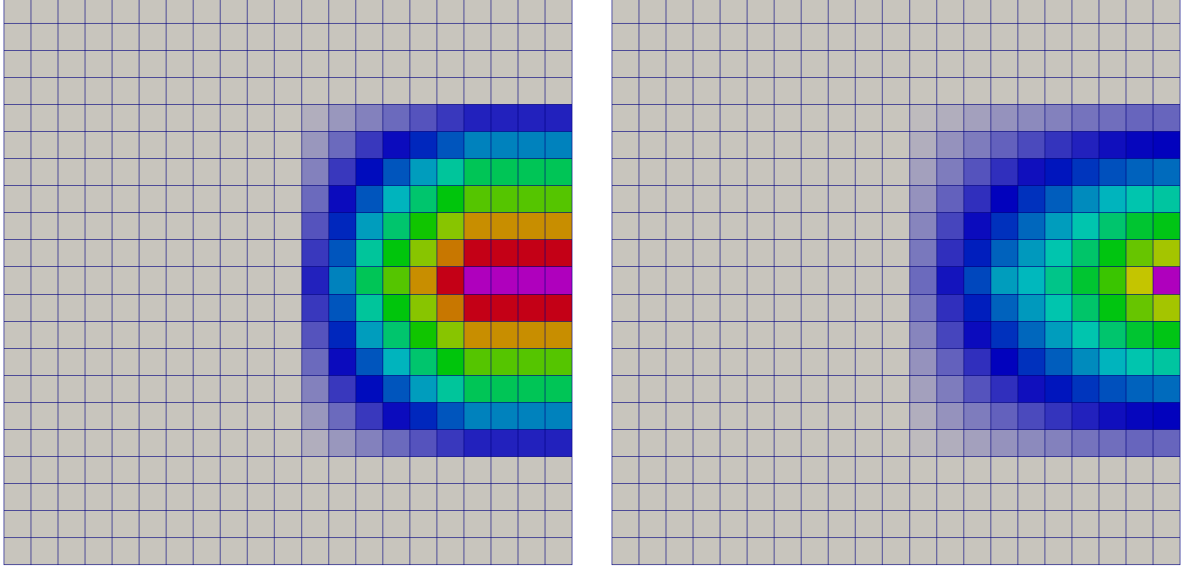


Figure 2.11.: Basis functions of a block that is adjacent to the east boundary of a structured network. In the right image M_j is moved to the boundary. The colour scheme is identical to the one of Figure 2.9.

2.13. Similarly to the structured case, the basis function with the modified M_j decays smoother out from the boundary.

As mentioned above, the here presented multiscale method should also work if only the matrix \mathbf{A} and the right hand side \mathbf{q} of the system of equations (1.8) are given and when the locations of the boundary cells are unknown. General boundaries BC are extracted directly from \mathbf{q} by identifying the non-zero entries, i.e.

$$BC = \{\Omega_i^F, \quad i \mid q_i \neq 0\}. \quad (2.25)$$

To specifically obtain the Dirichlet boundary cells, the fine cells with a non-zero row sum in \mathbf{A} , i.e.

$$BC = \{\Omega_i^F, \quad i \mid \sum_j A_{ij} \neq 0\}, \quad (2.26)$$

are determined. Note that no-flow boundaries cannot be extracted by using such an approach.

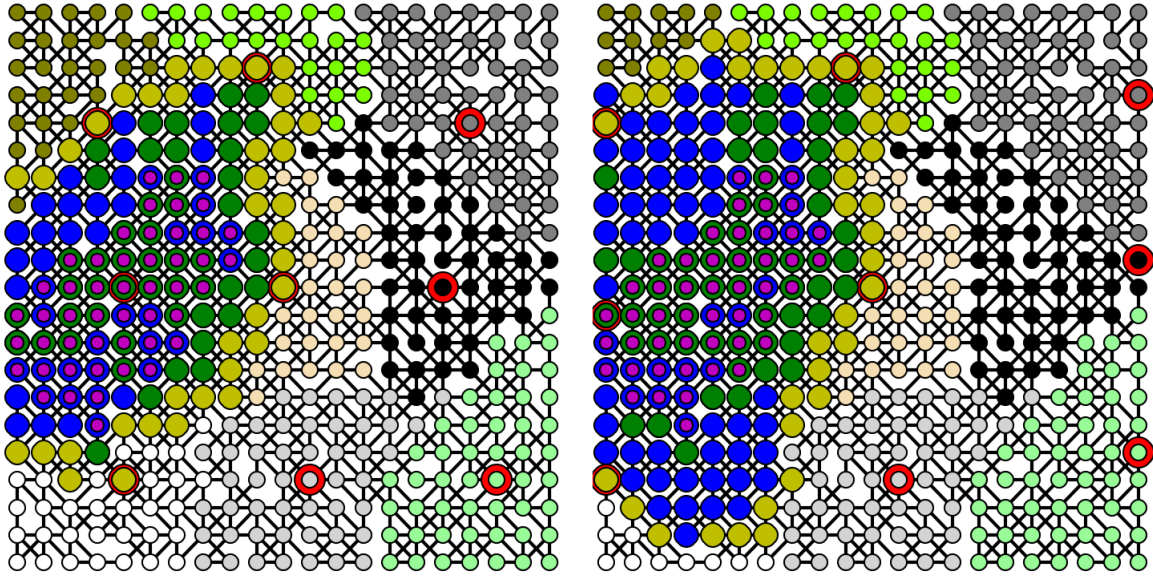


Figure 2.12.: Coarse centre cells M_j (red), support regions I_j (blue), support boundaries B_j (yellow) and the global support boundary vertices $G \cap I_j$ (green) for a boundary block of an unstructured network. In the right image M_j is moved to a cell adjacent to the boundary.

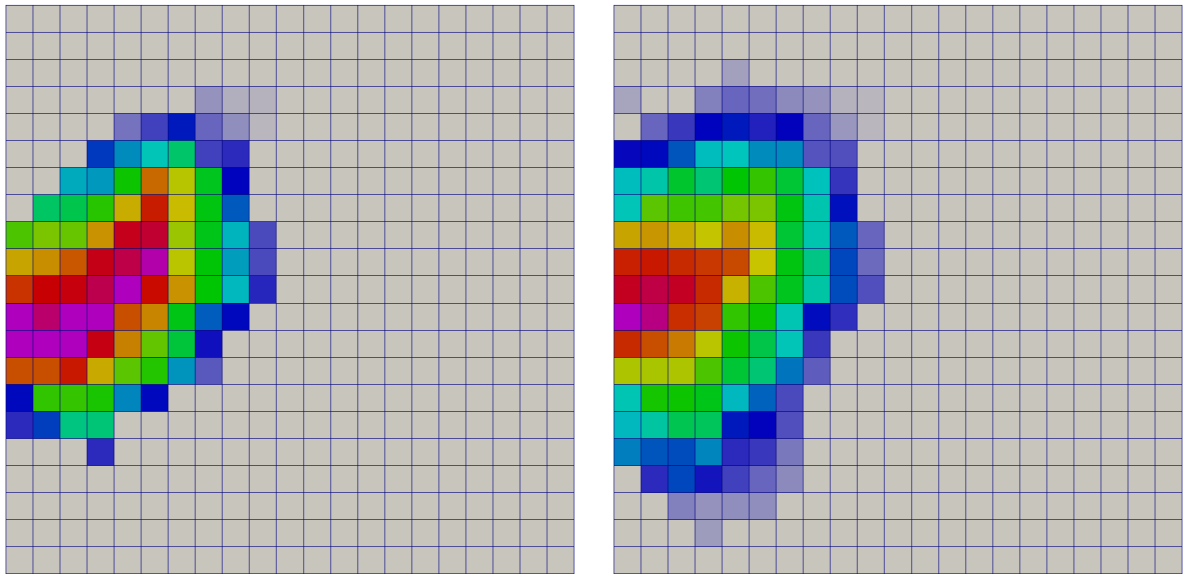


Figure 2.13.: Basis functions of a block which is adjacent to the west boundary of an unstructured network. In the right image M_j is moved to the boundary. The colour scheme is identical to the one of Figure 2.9.

2.7. Iterative Multiscale Formulation

Usually, (2.1) gives only an approximation for the solution of (1.8) and its quality depends, among other factors, on the size of the coarse system, the level of convergence of \mathbf{P} and the specific test case. The accuracy can be improved with an iterative multiscale formulation of the MsRSB method which uses a smoothing step to reduce the short-wavelength errors. In the following, an iterative multiscale formulation based on [8] is briefly summarised.

To compute the pressure solution $\mathbf{p}^{(n+1)}$ at iteration step $(n+1)$, the residual \mathbf{r} of the fine system at the previous step (n) is first determined, i.e.

$$\mathbf{r}^{(n)} = \mathbf{q} - \mathbf{A}\mathbf{p}^{(n)}. \quad (2.27)$$

A smoothing step \mathcal{S} is then applied to the residual to get $\mathbf{y}^{(n)} = \mathcal{S}(\mathbf{r}^{(n)})$. In the implementation, a few Jacobi iterations with the initial result set to zero or alternatively an incomplete LU-factorization (ILU) are performed on the fine scale system

$$\mathbf{A}\mathbf{y}^{(n)} = \mathbf{r}^{(n)}. \quad (2.28)$$

The coarse error $\mathbf{e}^{c(n)}$ is computed with a direct method, i.e.

$$\mathbf{e}^{c(n)} = (\mathbf{A}^c)^{-1} \mathbf{R}(\mathbf{r}^{(n)} - \mathbf{A}\mathbf{y}^{(n)}), \quad (2.29)$$

and the multiscale cycle is completed with the update of the solution, i.e.

$$\mathbf{p}^{(n+1)} = \mathbf{p}^{(n)} + \mathbf{P}\mathbf{e}^{c(n)} + \mathbf{y}^{(n)}. \quad (2.30)$$

The steps (2.27) to (2.30) are repeated until the maximum value of the residual r_i is below a certain tolerance and the pressure is assumed to be converged. The initial result $\mathbf{p}^{(0)}$ is usually set to $\mathbf{0}$ or to the result of the previous time step in a transient flow simulation.

It is common in multigrid methods to have more than one coarse level to increase convergence speed in an iterative multiscale formulation [18]. In the here presented black-box form of the MsRSB method, a second set of coarse prolongation and restriction operators could be determined based on \mathbf{A}^c to obtain a system of equations for the pressure on the next coarser level. One possible formulation of an iterative multilevel method is presented in Appendix B. However, coarse-scale transmissibilities of \mathbf{A}^c can be negative for certain fine-scale systems (see Section 2.8) and the so computed coarse-scale basis functions would have negative values. Therefore, the MsRSB method is not straightforwardly extendable to a multilevel multiscale method for general pore networks.

2.8. Adapted Grid Partitioning for Networks with Channels and Barriers

Although the here presented method produces results of good quality for a wide range of heterogeneous test cases, flow scenarios involving large transmissibility contrasts are in general more challenging. It was observed that the errors of the approximate solution computed with the MsRSB method are quite large in the vicinity of channels and barriers. In some cases, the multiscale method produces non-physical coarse systems with negative coarse transmissibilities that may lead to approximate solutions that violate the maximum principle. Therefore, a high number of smoothing steps per coarse step is usually required for reasonable convergence rates in an iterative formulation and in some extreme cases the iterative procedure may fail entirely. The performance is significantly increased if the restriction operator, and consequently the grid partitioning, are adapted to the structure of the basis functions [8]. This is done by finding the basis function with the largest value for every fine cell i and assigning the cell to the corresponding block, i.e.

$$C_k^{adapted} = \{i \mid k = \underset{j}{\operatorname{argmax}} P_{ij}\}. \quad (2.31)$$

In the following, a two-dimensional homogeneous test case with 200×200 fine cells and a diagonal channel with a much higher transmissibility than the surrounding pore network, e.g. $k_{channel}/k = 10^6$, is considered. Figure 2.14 shows the test case for the initial structured partitioning into 5×5 coarse blocks (left) and for the adapted partitioning (centre). The updated partitioning is now well aligned with the prolongation operator, as can be observed in Figure 2.14 (right). The yellow line in the figure represents the support boundary of the block considered.

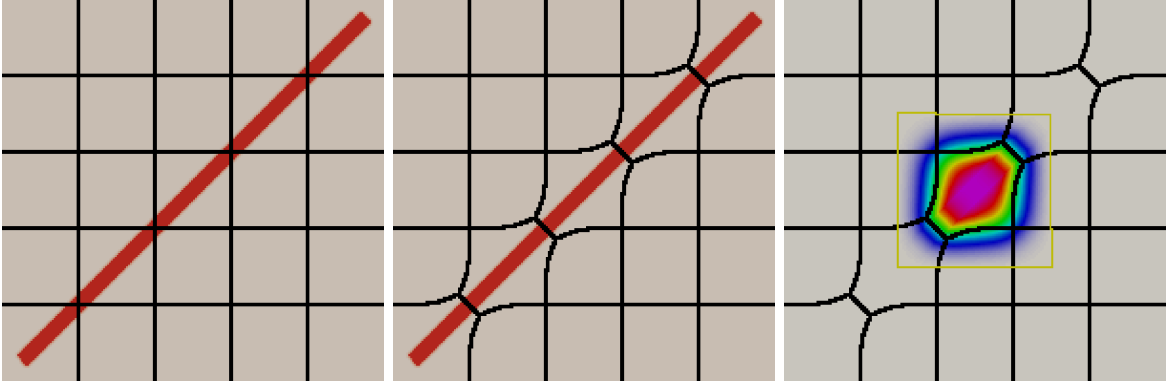


Figure 2.14.: Structured system with a diagonal channel. Left: Initial structured partitioning, Centre: Adapted partitioning, Right: Basis function for a block that includes the channel (The colour scheme is identical to the one of Figure 2.9).

The same is shown in Figure 2.15 for a test case where an unstructured initial partitioning into 25 blocks was used. It was reported in [8] and also observed in the here performed simulations, that adapting the restriction operator generally leads to better results, also for test cases involving

multiple channels. Flow scenarios including a large number of narrow channels in fractured media were not considered here and are preferably treated as described in [14].

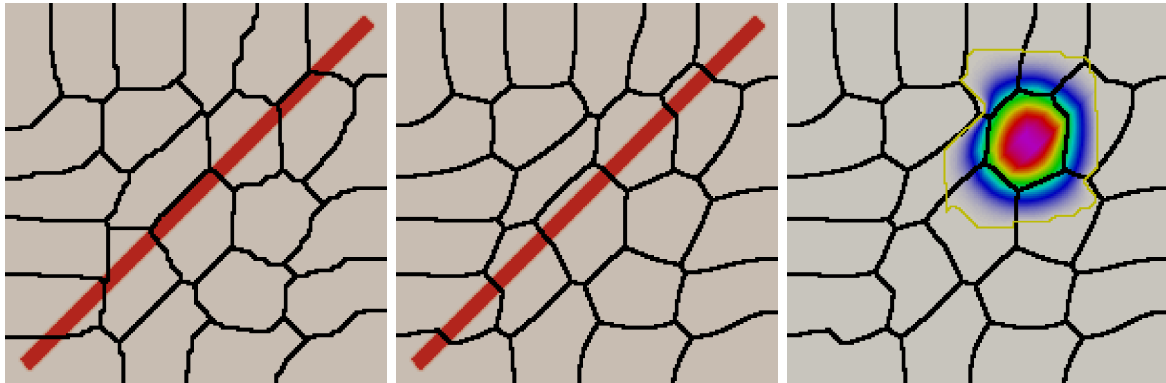


Figure 2.15.: Structured system with a diagonal channel. Left: Unstructured initial partitioning done with Metis, Centre: Adapted partitioning, Right: Basis function for a block that includes the channel (The colour scheme is identical to the one of Figure 2.9).

Figure 2.16 shows the initial structured partitioning, the adapted partitioning and two different basis functions for a test case with a diagonal flow barrier that has a much lower transmissibility than the surrounding geometry, e.g. $k_{\text{barrier}}/k = 10^{-6}$. In this case, modifying the restriction operator does not significantly increase the approximate solution and, in general, better results are obtained if the partitioning is fitted to the barrier before the prolongation operator is computed [8].

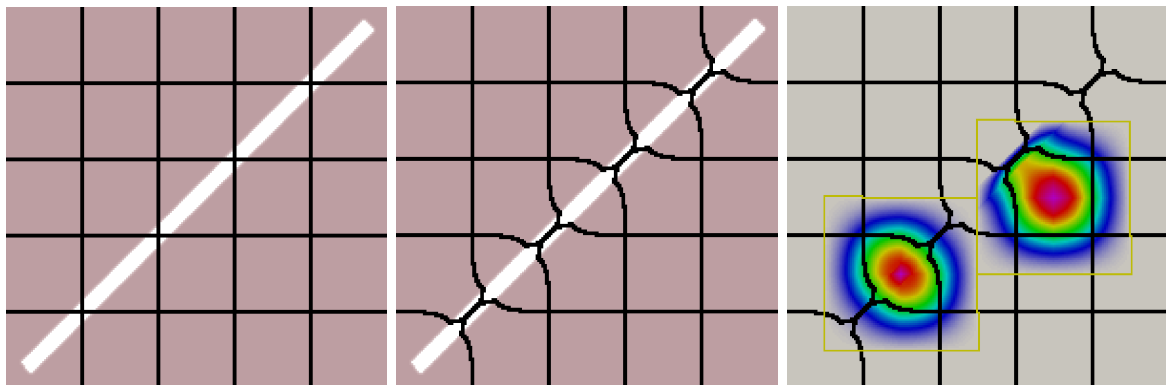


Figure 2.16.: Structured system with a diagonal barrier. Left: Initial structured partitioning, Centre: Adapted partitioning, Right: Basis functions for two blocks that include the barrier (The colour scheme is identical to the one of Figure 2.9).

3. Numerical Results

The multiscale method described in Chapter 2 was implemented in Python using the libraries NumPy [19], SciPy [20], python-igraph [21] and PyMetis [22]. In its black-box form, the resulting code only requires the system matrix \mathbf{A} and the right-hand-side vector \mathbf{q} of the linear system of equations (1.8) along with the number of coarse partitions as input parameters. Therefore the implementation works equally for structured and unstructured, two- and three dimensional model problems.

Multiple numerical studies were conducted to investigate the performance of the method and to verify the implementation. The accuracy is evaluated by computing a relative error ϵ_i for every fine cell Ω_i^F of the multiscale result \mathbf{p}^{approx} compared to a reference pressure \mathbf{p}^{ref} which is obtained by a direct numerical method, i.e.

$$\epsilon_i = \left| \frac{p_i^{approx} - p_i^{ref}}{p_i^{ref}} \right|. \quad (3.1)$$

To quantify the overall accuracy of one specific simulation, the maximum and root mean square errors are used, i.e.

$$\epsilon_{MAX} = \max_i \{\epsilon_i\}, \quad \epsilon_{RMS} = \sqrt{\frac{1}{n} \sum_{i=1}^n \epsilon_i^2}. \quad (3.2)$$

In the following the results of several numerical studies are presented. The analysis only includes steady state and incompressible single phase test cases. Furthermore, only the pressure distribution in pore networks is discussed and no fine-scale fluxes are reconstructed from the conservative coarse boundary fluxes. Note that transmissibilities and pressures used in the computation are dimensionless numbers with usually arbitrarily chosen values that have no relation to real porous media.

3.1. Heterogeneous Transmissibility Field

First, the test case of an unstructured pore network with 51'200 pores and a highly heterogeneous transmissibility distribution is investigated. To visualize the results, the pores are represented on a regular two-dimensional grid with 320 x 160 cells as shown in Figure 3.1 (only the lower left part of the domain is visualized).

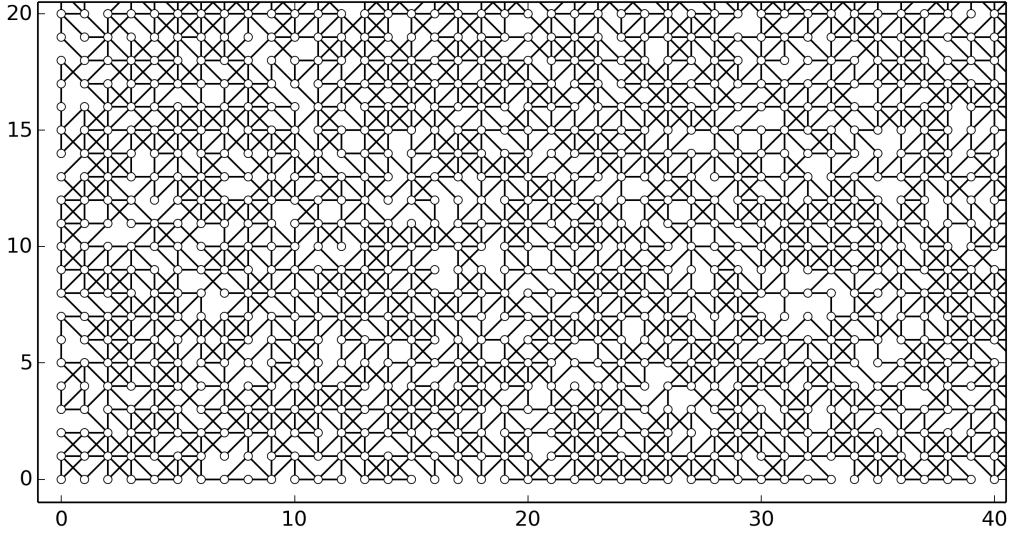


Figure 3.1.: Lower left part of an unstructured pore network represented on a regular grid.

The average coordination number of the network is $n_{c,avg} = 6.5$ and transmissibilities with values from the range $k_{max}/k_{min} = 10^6$ are randomly assigned to the pore throats. An average transmissibility \bar{k}_i is determined for every pore based on the values from its neighbouring pore throats, i.e.

$$\bar{k}_i = \frac{1}{n_{c,i}} \sum_{j=1}^{n_{c,i}} k_{ij}, \quad (3.3)$$

and the resulting field is visualized in Figure 3.2 along with the partitioning of the fine scale system into 50 coarse blocks.

The basis functions of three blocks are visualized in Figure 3.3 for three different levels of convergence of the prolongation operator with $d_{crit} = 10^{-3}$ (top), $d_{crit} = 10^{-4}$ (middle) and $d_{crit} = 10^{-5}$ (bottom). The yellow lines in the figure represent the support boundaries of the corresponding blocks. In general, low d_{crit} values are required for highly heterogeneous networks to obtain prolongation operators of good quality. However, more than 10'000 iteration steps on the prolongation operator are here required for the lowest tolerance criteria which is computationally expensive.

The one step pressure solution p^{approx} obtained from Equation (2.4) and the corresponding relative error are shown in Figure 3.4 for a test case with fixed pressure values of 2 and 1 at the left and right, no-flow boundaries at the top and bottom and a prolongation operator converged to $d_{crit} = 10^{-5}$. It is observed that an accurate pressure field with a low error results in the whole domain.

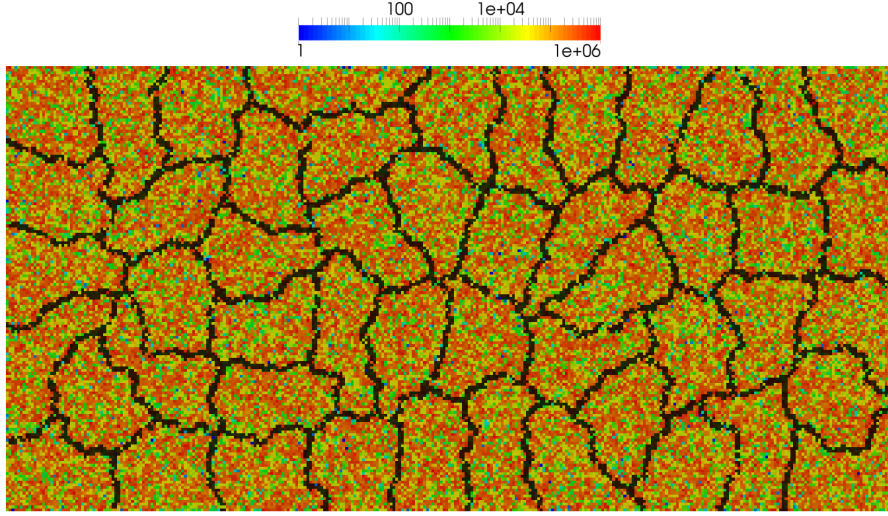


Figure 3.2.: Transmissibility field \bar{k} of a highly heterogeneous network and grid partitioning into 50 coarse blocks.

The level of convergence has a direct influence on the accuracy of the one step pressure approximation. The errors ϵ_{RMS} and ϵ_{MAX} for different d_{crit} are listed in Table 3.1. As expected, results of better quality are obtained for lower d_{crit} . However, this comes with a higher computational cost for the calculation of the prolongation operator. It is observed that in this specific test case, ϵ_{MAX} is higher for $d_{crit} = 10^{-6}$ than for $d_{crit} = 10^{-5}$. Note that the fine scale fluxes over the coarse scale grid borders are always conservative if a control volume restriction operator is used, independent of the convergence level of the prolongation operator.

Table 3.1.: ϵ_{MAX} and ϵ_{RMS} of p^{approx} for different convergence levels

	ϵ_{RMS}	ϵ_{MAX}
$d_{crit} = 10^{-3}$	0.0288	0.1025
$d_{crit} = 10^{-4}$	0.0243	0.0860
$d_{crit} = 10^{-5}$	0.0146	0.0551
$d_{crit} = 10^{-6}$	0.0126	0.0624

The overall accuracy of the one step approximation is improved if the number of coarse blocks m is increased. The results for $m = 25$ (top) and $m = 105$ (bottom) are shown in Figure 3.5 and the error norms are summarised in Table 3.2. Here, the values n/m denote the average number of fine cells per block. Note that the number of 105 partitions is chosen since the grid partitioner Metis produces disconnected blocks for the initially chosen number of 100 blocks. It is observed that ϵ_{MAX} is higher for $m = 105$ than for $m = 50$. However, although all results were obtained by using the same d_{crit} for the prolongation operator, comparison between the three results is difficult, since the optimum d_{crit} may vary depending on the number and size of the partitions.

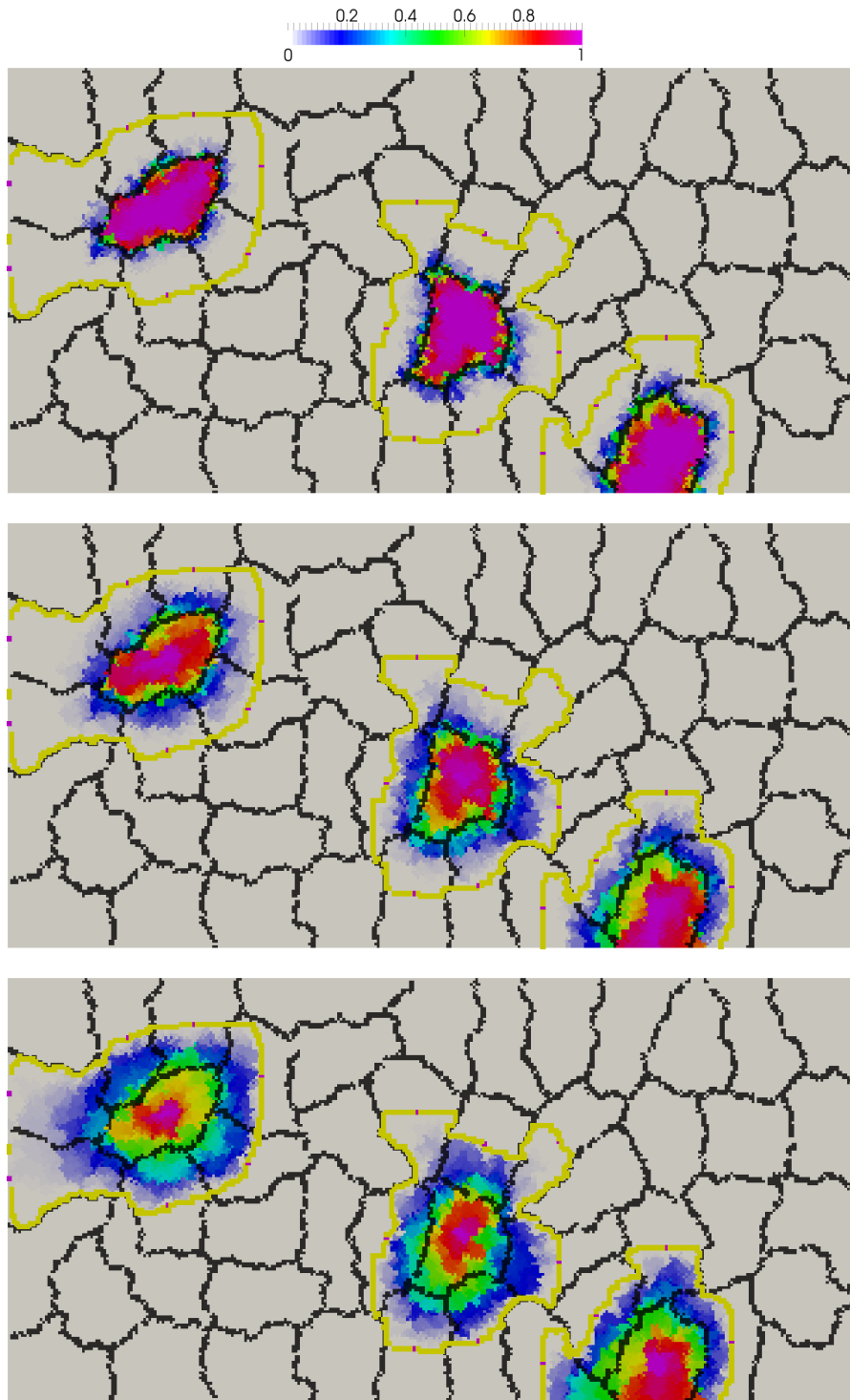


Figure 3.3.: Basis functions for different levels of convergence. Top: $d_{crit} = 10^{-3}$, Middle: $d_{crit} = 10^{-4}$, Bottom: $d_{crit} = 10^{-5}$.

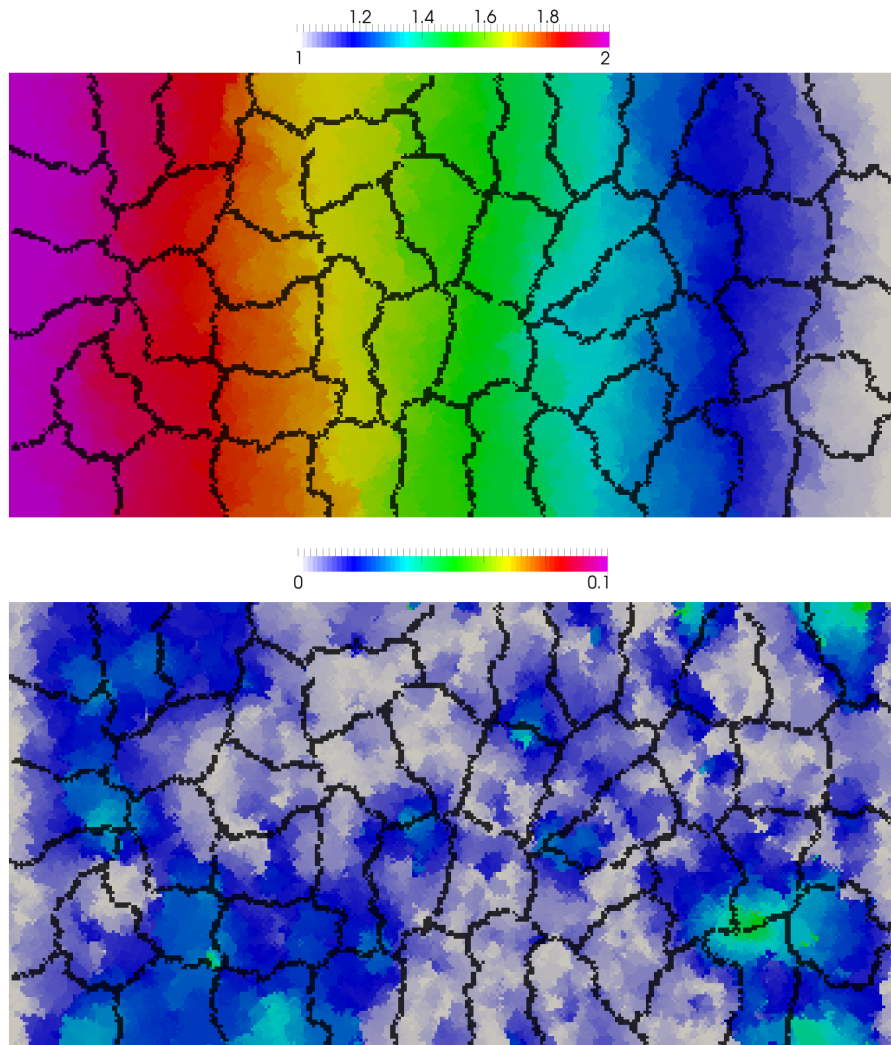


Figure 3.4.: One step pressure approximation p^{approx} (top) and the corresponding error ϵ (bottom).

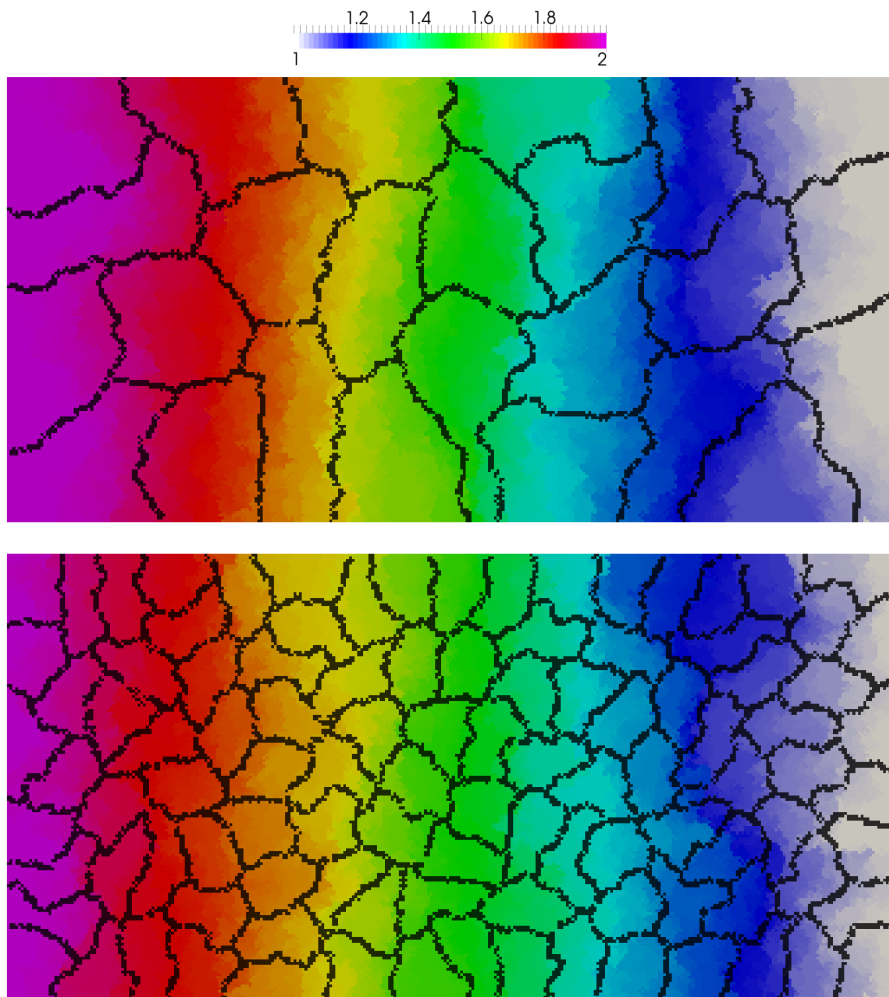


Figure 3.5.: One step approximation p^{approx} for 25 (top) and 105 (bottom) coarse blocks.

Table 3.2.: ϵ_{MAX} and ϵ_{RMS} of p^{approx} for different numbers of coarse partitions

m	n/m	ϵ_{RMS}	ϵ_{MAX}
25	2048	0.0307	0.0963
50	1024	0.0146	0.0551
104	488	0.0139	0.0642

3.2. Channels

Pore networks with high permeability contrasts are in general challenging to simulate with a multiscale method. In the following, a test case involving five highly permeable channels with $k_{channel}/k = 10^6$, where k is the transmissibility outside the channel, is investigated. The simulation setup is identical to the one introduced in the previous section, except that the transmissibilities of the pore throats are modified to represent the channels. Figure 3.6 shows the transmissibility field \bar{k}_i with the initial partitioning into 50 coarse blocks (top) and the adapted partitioning (bottom) according to Section 2.8. Note that for the adapted case, coarse blocks with different number of fine cells result, which may be a disadvantage for the multiscale method regarding accuracy and iterative performance.

Examples of basis functions converged to $d_{crit} = 10^{-5}$ are visualized in Figure 3.7 (top) together with the resulting one step pressure approximation (middle) and the relative error (bottom) for the case of an adapted restriction operator. It is observed that the errors are in general larger than for the heterogeneous test case, especially at the right boundary.

Using an adapted restriction operator significantly decreases the error norms if the same convergence level of the prolongation operator is used, as shown in Table 3.3.

Table 3.3.: ϵ_{MAX} and ϵ_{RMS} of p^{approx} for different restriction operators R

	ϵ_{RMS}	ϵ_{MAX}
R not adapted	0.1270	0.2551
R adapted	0.0216	0.1670

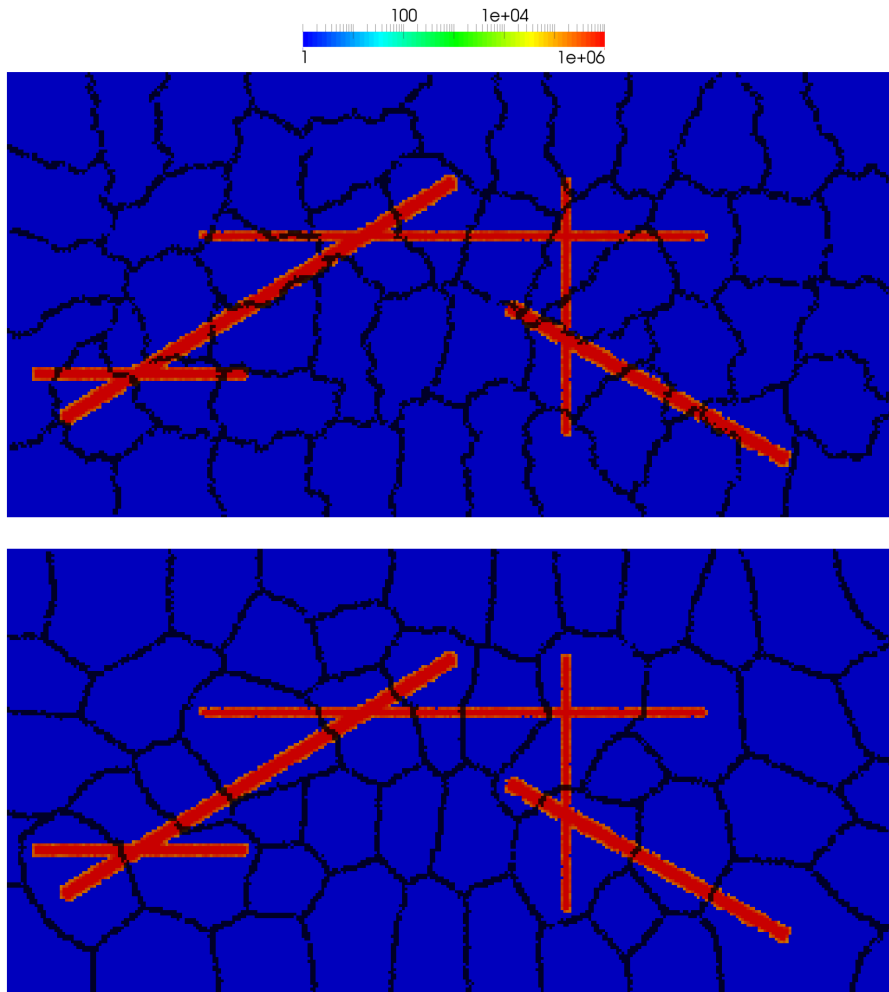


Figure 3.6.: Transmissibility field \bar{k} with initial (top) and adapted (bottom) partitioning

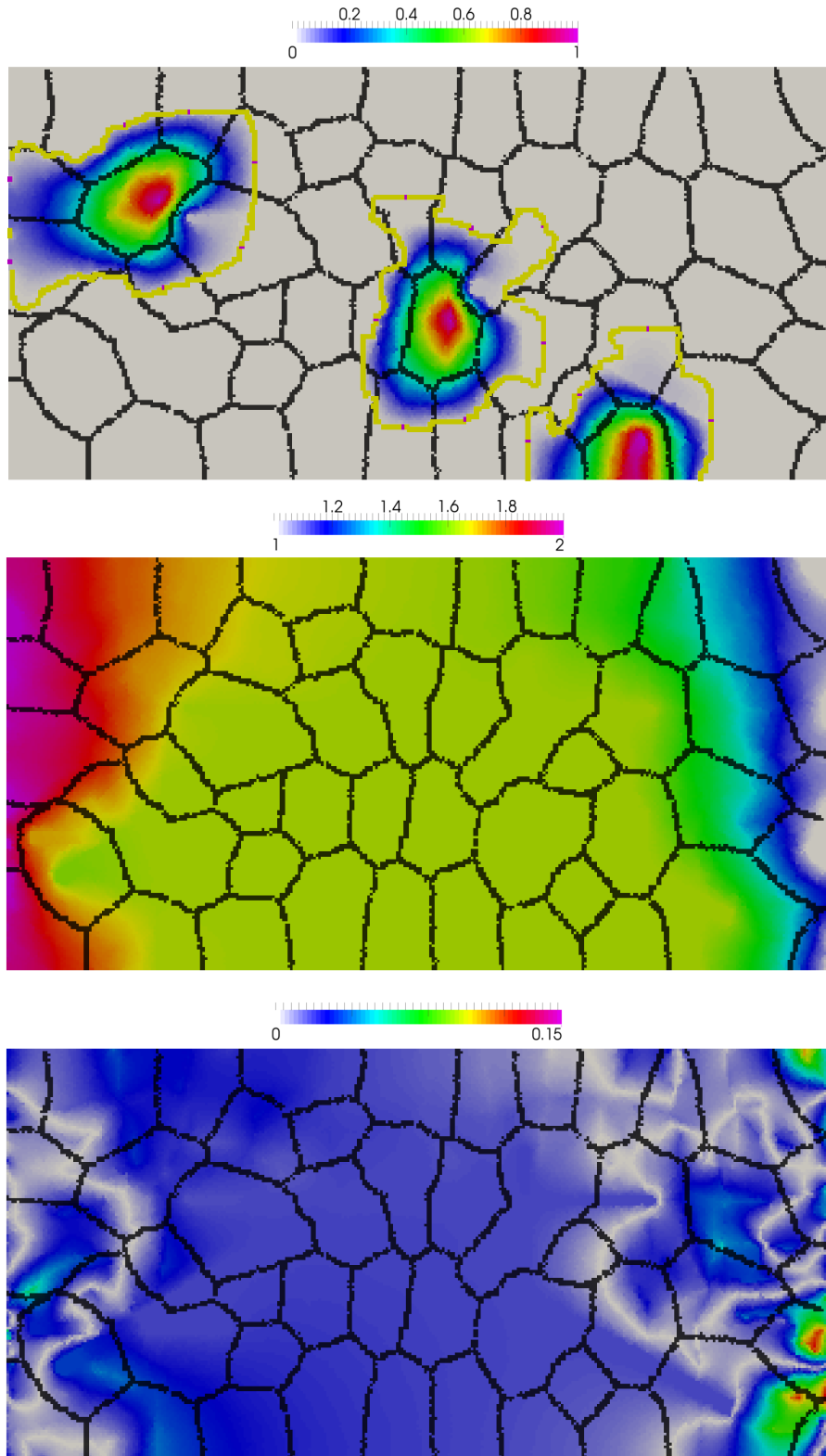


Figure 3.7.: Basis functions (top), one step pressure approximation p^{approx} (middle) and relative error ϵ (bottom) for a pore network with channels.

3.3. Flow Barriers

Next, a test case involving five flow barriers is discussed. The barriers are represented by modifying the transmissibilities of the channels from the previous section to much lower values compared to the surrounding geometry, i.e. $k_{\text{barrier}}/k = 10^{-6}$. The transmissibility distribution is shown in Figure 3.8 together with the partitioning into 50 blocks using Metis. Table 3.4 summarises the error norms for different coarse partitions used. In contrast to the channel test case, using a grid partitioning and restriction operator adapted to the prolongation operator does not improve the multiscale result. However, the overall result is significantly improved if the transmissibility-values are used as edge weights in the initial Metis partitioning.

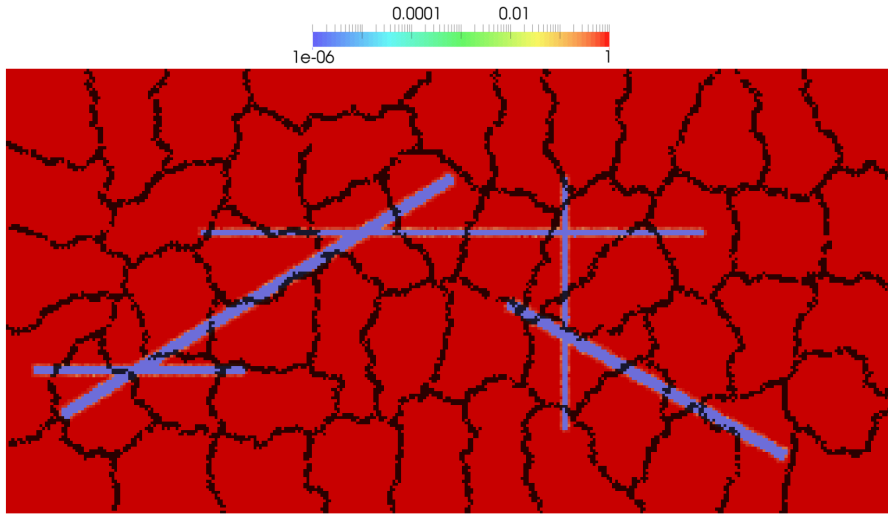


Figure 3.8.: Transmissibility field \bar{k} with an unweighted partitioning into 50 blocks.

Table 3.4.: ϵ_{MAX} and ϵ_{RMS} of p^{approx} for different restriction operators and partitioning

	ϵ_{RMS}	ϵ_{MAX}
Unweighted partitioning, R not adapted	0.0441	0.2123
Unweighted partitioning, R adapted	0.0507	0.2119
Weighted partitioning, R not adapted	0.0216	0.3528

The approximate pressure solution (top) and its relative error (bottom) are shown in Figure 3.9 for the test case with a weighted partitioning. The maximum error, which corresponds to $\epsilon_{MAX} = 0.3528$ in Table 3.4, is located in the vertical barrier.

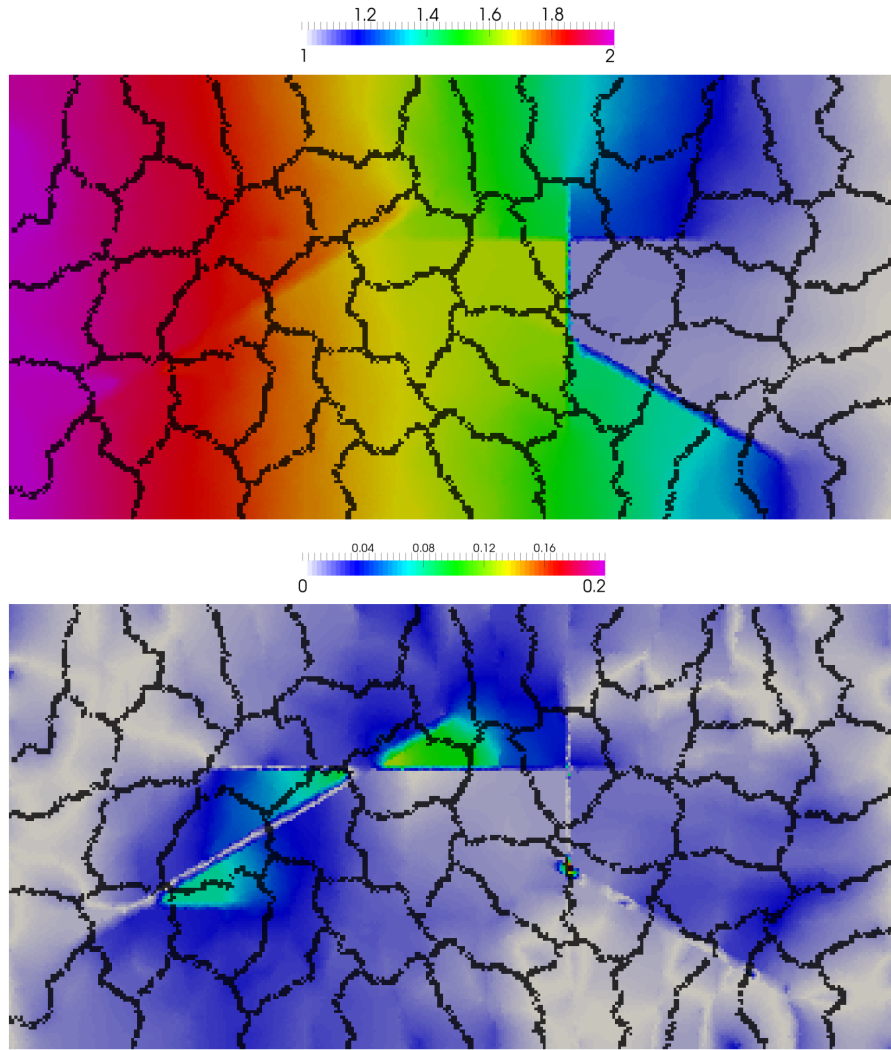


Figure 3.9.: One step approximation p^{approx} (top) and relative error ϵ (bottom) for a pore network with flow barriers.

3.4. Three-Dimensional Pore Networks

For the sake of completeness, a three-dimensional test case with 54'705 pores and a heterogeneous transmissibility distribution with $k_{max}/k_{min} = 1.4 \cdot 10^6$ as visualized in Figure 3.10 (top) is briefly discussed. In the figure, the spheres represent the pores, however, the sizes of the spheres are arbitrarily chosen and have no relation to actual length scales. The average coordination number is $n_{c,avg} = 10.3$ and the grid is partitioned into 50 blocks. The resulting approximate pressure solution for a prolongation operator converged to $d_{crit} = 10^{-5}$ is shown in Figure 3.10 (bottom). Here, Dirichlet boundaries on two opposing sides of the pore network were used with the pressures fixed to 2 and 1, respectively. Table 3.5 summaries the error norms for different levels of convergence of the prolongation operator. Generally, the errors are higher than for the two-dimensional case presented in Table 3.1.

Table 3.5.: ϵ_{MAX} and ϵ_{RMS} of p^{approx} for different convergence levels.

	ϵ_{RMS}	ϵ_{MAX}
$d_{crit} = 10^{-3}$	0.0364	0.1357
$d_{crit} = 10^{-4}$	0.0270	0.1346
$d_{crit} = 10^{-5}$	0.0261	0.1343
$d_{crit} = 10^{-6}$	0.0260	0.1342

The overall computation time to obtain the basis functions is significantly higher for three-dimensional than for two-dimensional test cases. This is due to the generally larger support regions resulting from the higher number of neighbouring coarse cells per block. At the same time, the number of global support boundary vertices can be very large for three dimensional pore networks. Therefore a fast implementation of the partition-of-unity scaling steps (2.17) and (2.18) is crucial for the overall performance of the method.

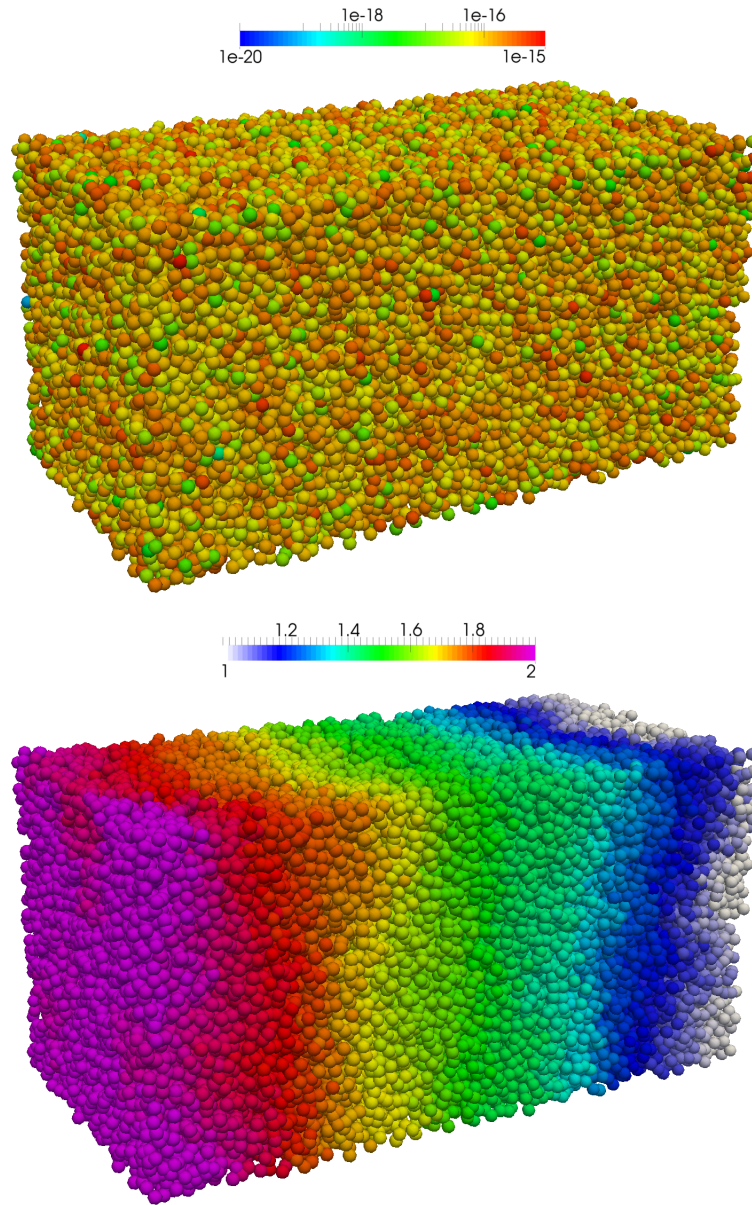


Figure 3.10.: Transmissibility field \bar{k} (top) and approximate pressure solution p^{approx} (bottom) of a three-dimensional heterogeneous network.

3.5. Iterative Solver

By using an iterative implementation of the multiscale method the error of the pressure solution is reduced below a predefined tolerance as discussed in Section 2.7. In the following, the test case involving five highly-permeable channels as introduced in Section 3.2 with the prolongation operator converged to $d_{crit} = 10^{-5}$ is considered. The iterative performance depends on the choice of the restriction operator, which can be obtained by either using a control-volume or a Galerkin formulation. The solution diverges if a control-volume operator unadapted to the prolongation operator is used. This is due to the relatively large positive off-diagonal values in the coarse matrix \mathbf{A}^c which represent negative and therefore unphysical coarse scale transmissibilities. For the channel test case, normalized positive coarse off-diagonals, i.e. a_{ij}^c/a_{ii}^c , with values up to 0.98 were observed. By adapting the control-volume operator to the basis functions as discussed in Section 2.8 the maximum normalized coarse off-diagonal is reduced to 0.46 and the solution converges. Figure 3.11 compares the convergence histories of ϵ_{RMS} (left) and ϵ_{MAX} (right) of the iterative solvers that either use a Galerkin or an adapted control-volume (CV) implementation of the restriction operator. For the fine-scale smoother, 10 Jacobi iterations per multiscale cycles were used. As expected, faster convergence is observed by using a Galerkin approach. However, the advantage of a control-volume formulation is that the fluxes over the coarse borders are conservative after each multiscale cycle. Therefore it is recommended to first use the Galerkin approach to ensure fast convergence and then switch to the control-volume formulation in the last iteration step to obtain a solution that is conservative on the coarse scale.

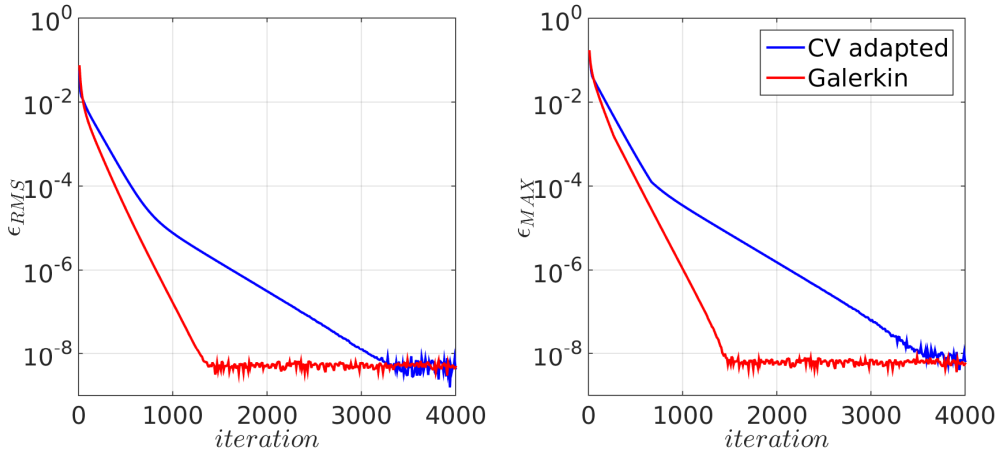


Figure 3.11.: Convergence histories of ϵ_{RMS} (left) and ϵ_{MAX} (right) for an iterative MsRSB formulation with 10 Jacobi steps per multiscale cycle.

The convergence speed is significantly increased if 10 ILU steps instead of the Jacobi iterations are used as a smoother, as can be seen in Figure 3.12. It is observed that the implementation with the control-volume restriction operator initially converges faster than the Galerkin approach in this case. After ≈ 50 steps, the convergence speed of the control-volume implementation

3.5. Iterative Solver

decreases and the two curves are congruent after ≈ 150 steps. No further studies were conducted to investigate the reasons for this behaviour.

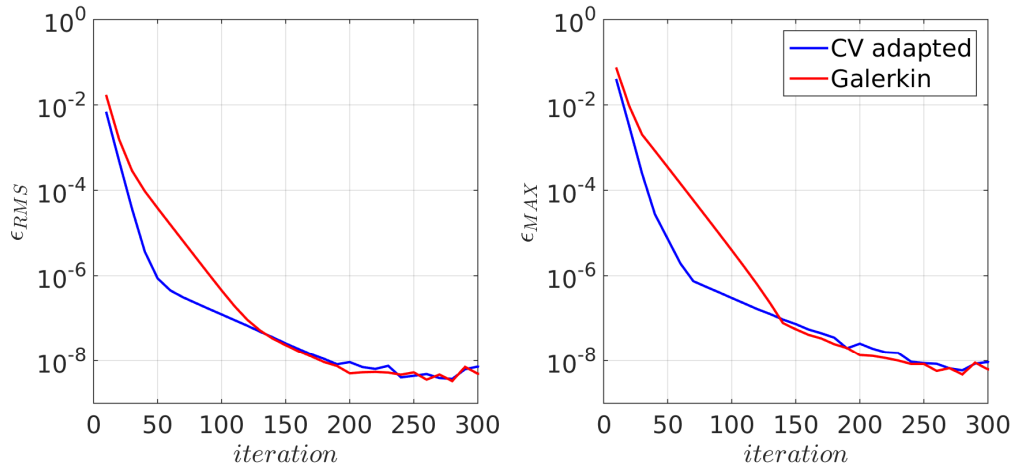


Figure 3.12.: Convergence histories of ϵ_{RMS} (left) and ϵ_{MAX} (right) for an iterative MsRSB formulation with 10 ILU steps per multiscale cycle.

4. Conclusion and Outlook

In this work, the MsRSB method was extended for unstructured pore networks. A graph representation of the fine scale system is used to determine the block centres of every coarse partition. Support regions are then obtained for each partition by repeatedly performing a neighbourhood search on the fine scale graph until the centres of the neighbouring coarse blocks are reached. This method produces support regions that are fully connected and does not require knowledge of the geometrical pore locations. Subsequently, the basis functions are computed by using an iterative smoothing process that is restricted to the corresponding support regions. In combination with an automatic partitioning algorithm such as Metis, the method can be used as a black-box solver for many different Poisson-type problems.

It was observed that the MsRSB method produces results of good quality for different test cases involving highly heterogeneous and unstructured pore networks. However, unphysical coarse systems with negative transmissibilities can occur for certain scenarios, especially if large transmissibility contrasts on the fine-scale are involved, e.g. due to channels or flow barriers. This can result in approximate pressure solutions that violate the maximum principle and, if an iterative multiscale formulation is used, the solver may converge slowly or even diverge. The performance is improved when either the restriction or the prolongation operator is adapted to the underlying transmissibility field. For this adaption step knowledge about the specific test case is necessary and hence the method cannot fully operate as a black-box solver for these scenarios.

The thesis is closed with suggestions for further research:

- In order to further investigate the accuracy and the iterative performance of the MsRSB method for unstructured pore networks, a systematic study on various test cases with different boundary conditions should be conducted. This study should include multiphase flow scenarios and simulation geometries resembling realistic porous media.
- The influence of negative coarse transmissibilities on accuracy, stability and iterative performance of the multiscale method should be investigated. This is relevant especially for test cases involving channels and flow-barriers.
- In a parametric study, the influence of several model parameters such as weak link radius and tolerance of the prolongation operator should be analysed. Furthermore, it should be investigated whether it is better to use a weighted or unweighted graph representation to define the partitioning and to find the coarse block centres.
- The current implementation should be improved regarding computational efficiency in order to obtain a code that is able to simulate large three-dimensional networks in a reasonable time. The main potential is seen in the normalisation steps required to ensure partition of unity in the computation of the prolongation operator. Currently, Python dictionaries

are created and accessed frequently, which is computationally inefficient and therefore a bottleneck in the implementation.

Bibliography

- [1] V. JOEKAR-NIASAR AND S. M. HASSANIZADEH *Analysis of Fundamentals of Two-Phase Flow in Porous Media Using Dynamic Pore-Network Models: A Review*, 2012, Critical Reviews in Environmental Science and Technology 42, 1895-1976.
- [2] V. JOEKAR-NIASAR, S. M. HASSANIZADEH AND H. K. DAHLE *Non-equilibrium effects in capillarity and interfacial area in two-phase flow: dynamic pore modelling*, 2010, Journal of Fluid Mechanics 655, 38-71.
- [3] M. A. OLSHANSKII AND E. E. TYRTYSHNIKOV *Iterative methods for linear systems : theory and applications* , 2014, SIAM Society for Industrial and Applied Mathematics, Chapter 1, Krylov Subspace Methods.
- [4] K. STÜBEN *A review of algebraic multigrid*, 1999, Journal of Computational and Applied Mathematics 128, 281-309.
- [5] C. L. FARMER *Upscaling: a review*, 2002, International journal for numerical methods in fluids 40, 63-78.
- [6] T. Y. HOU AND X. H. WO *A Multiscale Finite Element Method for Elliptic Problems in Composite Materials and Porous Media*, 1996, Journal of Computational Physics 134, 169-189.
- [7] P. JENNY, S.H. LEE AND H.A. TCHELEPI *Multi-scale finite-volume method for elliptic problems in subsurface flow simulation*, 2003, Journal of Computational Physics 187, 47-67.
- [8] O. MØYNER AND K.-A. LIE *A multiscale restriction-smoothed basis method for high contrast porous media represented on unstructured grids*, 2016, Journal of Computational Physics 304, 46-71.
- [9] Y. WANG, H. HAJIBEYGI AND H.A. TCHELEPI *Algebraic multiscale solver for flow in heterogeneous porous media*, 2014, Journal of Computational Physics 259, 284-303.
- [10] H. HAJIBEYGI, G. BONFIGLI, M. A. HESSE AND P. JENNY *Iterative multiscale finite-volume method*, 2008, Journal of Computational Physics 227, 8604-8621.
- [11] H. HAJIBEYGI AND P. JENNY *Adaptive iterative multiscale finite volume method*, 2011, Journal of Computational Physics 230, 628-643.
- [12] D. CORTINOVIS AND P. JENNY *Iterative Galerkin-enriched multiscale finite volume method*, 2014, Journal of Computational Physics 277, 248-267.
- [13] P. VANĚK, J. MANDEL AND M. BREZINA *Algebraic Multigrid by Smoothed Aggregation for Second and Forth Order Elliptic Problems*, 1995, Computing (Springer-Verlag) 56, 179-196.

- [14] S. SHAH, O. MØYNER, M. TENE, K.-A. LIE AND H. HAJIBEYGI *The multiscale restriction smoothed basis method for fractured porous media (F-MsRSB)*, 2016, Journal of Computational Physics 318, 1-22.
- [15] W.L. BRIGGS, V.E. HENSON AND S.F. MCCORMICK *A Multigrid Tutorial, second edition*, 2000, SIAM Society for Industrial and Applied Mathematics, Chapter 8, Algebraic Multigrid (AMG).
- [16] G. KARYPIS *Metis. A Software Package for Partitioning Unstructured Graphs, Partitioning Meshes, and Computing Fill-Reducing Orderings of Sparse Matrices.*, 2013, <http://glaros.dtc.umn.edu/gkhome/fetch/sw/metis/manual.pdf>.
- [17] M. A. BEAUCHAMP *An improved index of centrality*, 1965, Behavioral Science 10 (2), 161-163.
- [18] W.L. BRIGGS, V.E. HENSON AND S.F. MCCORMICK *A Multigrid Tutorial, second edition*, 2000, SIAM Society for Industrial and Applied Mathematics, Chapter 3, Elements of Multigrid.
- [19] S. VAN DER WALT, S. C. COLBERT AND G. VAROQUAUX *The NumPy Array: A Structure for Efficient Numerical Computation*, 2011, Computing in Science and Engineering 13, 22-30.
- [20] E. JONES, E. OLIPHANT, P. PETERSON, ET AL. *SciPy: Open Source Scientific Tools for Python*, 2001-, <http://www.scipy.org/> [Online; accessed 2016-10-25].
- [21] PYTHON-IGRAPH <http://igraph.org/python>, Version 0.6.5
- [22] PYMETIS <https://mathematician.de/software/pymetis>, Version 2016.1

A. Direct Numerical Solution of the Prolongation Operator

The iterative method to compute the prolongation operator described in Section 2.4 is computationally expensive especially if \mathbf{P} is computed from scratch and if the tolerance criteria d_{crit} to stop the iteration is low. Alternatively, one could think of using a direct method to compute the basis functions Φ_j .

A.1. Numerical Modelling

A direct procedure to compute \mathbf{P} consists of first solving a linear system of equations to get an initial result $\hat{\Phi}_j$ for each coarse cell Ω_j^C which is independent of other basis functions, i.e.,

$$\mathbf{A}_j \hat{\Phi}_j = \mathbf{b}_j. \quad (\text{A.1})$$

Here, \mathbf{b}_j is zero everywhere except at the coarse block centre cell M_j , i.e.,

$$b_{ij} = \begin{cases} 1, & \text{if } i = M_j \\ 0, & \text{otherwise,} \end{cases} \quad (\text{A.2})$$

and \mathbf{A}_j is the system matrix \mathbf{A} that is locally replaced by the identity matrix at all cells outside I_j and at the centre cell M_j . The initial basis functions are then scaled to ensure partition-of-unity, i.e.

$$\Phi_{ij} = \frac{\hat{\Phi}_{ij}}{\sum_j \hat{\Phi}_{ij}}. \quad (\text{A.3})$$

It was observed that basis functions obtained with iterative and direct methods coincide for one-dimensional test cases. However, large differences can occur for multidimensional networks. In the following, the basis functions resulting from the iterative and direct methods are compared for a simple two-dimensional and structured network with homogeneous conductivities which is partitioned into three coarse blocks. The partitioning and the corresponding support regions and -boundaries for the centre block are visualized in Figure A.1.

The basis functions resulting from the iterative (left) and direct (right) approaches are shown in Figure A.2. In the case of the iterative method, the basis function decays linearly from the global support boundary vertices in the middle towards the eastern and western support boundaries and has constant values regarding to the vertical axis of the figure. Therefore, a linear pressure drop from left to right could be exactly reproduced in a flow network with constant transmissibilities

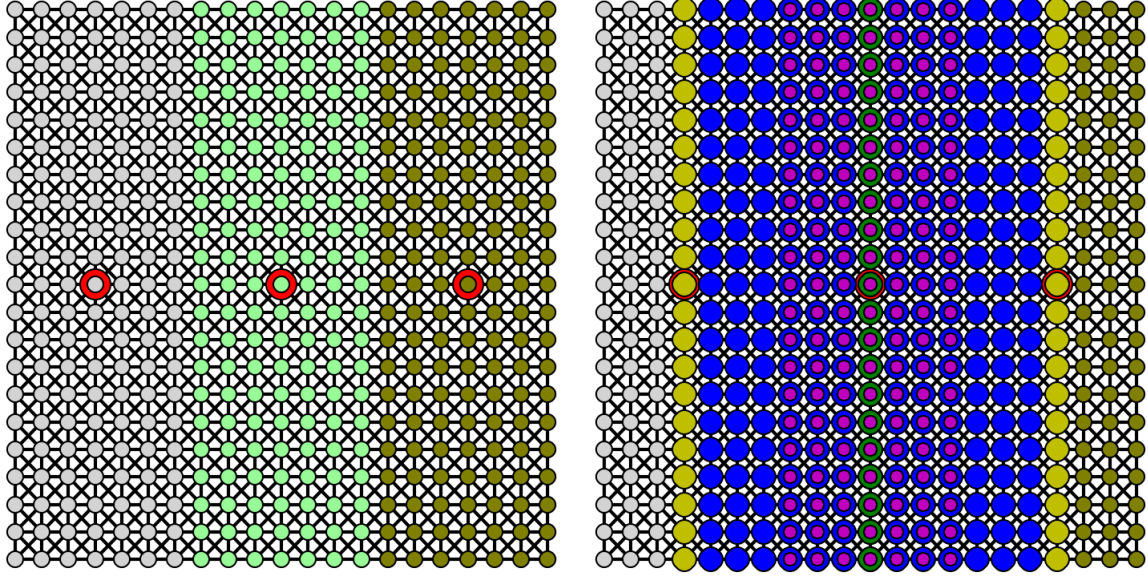


Figure A.1.: Left: Partitioning of a structured network into three coarse blocks. Right: Support regions, support boundaries and global support boundary vertices for the centre block. The colour scheme is identical to Figure 2.6.

for a fully converged prolongation operator. On the contrary, the direct method produces a basis function with values that increase towards the coarse block centre.

A better basis function is obtained if the result from (A.3) is corrected in a subsequent step: The values of Φ_{ij} on G are used as Dirichlet boundaries in a second linear system of equations to solve for the modified basis function $\hat{\Phi}_j^{mod}$, i.e.

$$\mathbf{A}_j^{mod} \hat{\Phi}_j^{mod} = \mathbf{b}_j^{mod}, \quad (\text{A.4})$$

with \mathbf{A}_j^{mod} and \mathbf{b}_j^{mod} being the system matrix adapted to the new boundaries and the modified right-hand-side, i.e.,

$$b_{ij}^{mod} = \begin{cases} \Phi_{ij}, & \text{if } i \in I_j \cap G \\ 0, & \text{otherwise,} \end{cases} \quad (\text{A.5})$$

respectively. The correction step is concluded by scaling the basis functions to again obtain partition of unity. Figure A.3 (left) shows the basis function for the system introduced in Figure A.1 obtained by the direct method with subsequent correction step which is now identical to the fully converged iterative solution.

For the unstructured network introduced in Figure 2.5 (right) basis functions obtained with the iterative and direct procedures are also quite similar, as can be observed by comparing Figures A.3

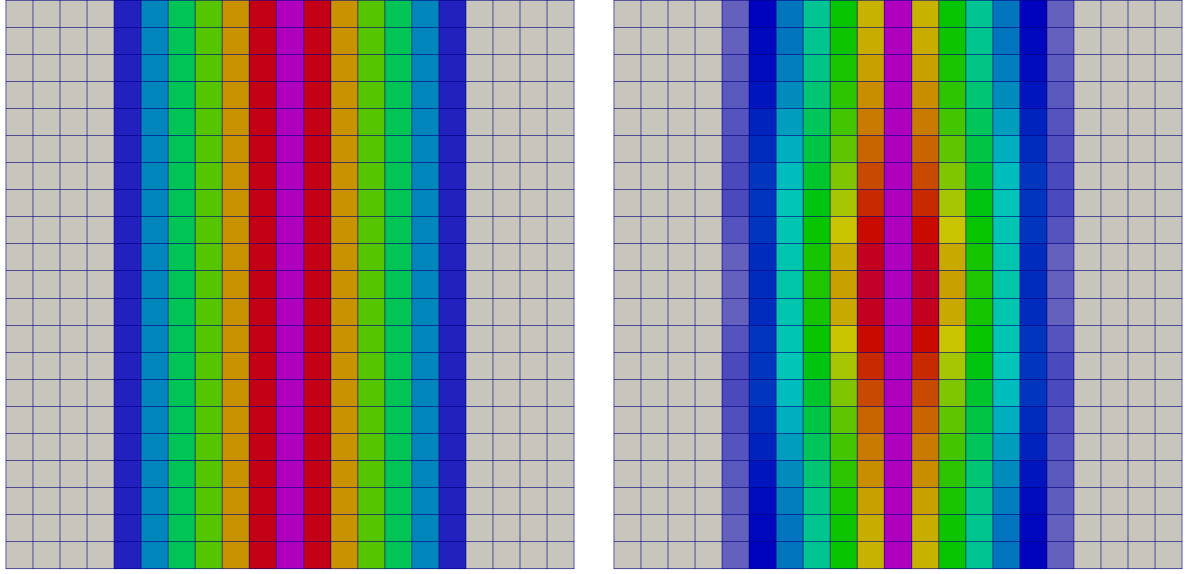


Figure A.2.: Basis functions obtained with an iterative (left) and direct (right) method for the centre block of a structured network with three coarse partitions. The colour scheme is identical to the one of Figure 2.9.

(right) and 2.9 (right). However, the direct procedure may produce bad results for more complex systems with large transmissibility contrasts due to the scaling step (A.3), as demonstrated in the next subsection.

A.2. Numerical Results

For the highly heterogeneous system introduced in Section 3.1, the basis functions (top), approximate pressure solution (middle) and relative error (bottom) are shown in Figure A.4 for a prolongation operator obtained with a direct method. Although the overall error is small, unphysical solutions that violate the maximum principle are observed at the lower right side of the pressure field.

The multiscale method with a direct calculation of the prolongation operator entirely fails for test cases involving high permeable channels as discussed in Section 3.2. Figure A.5 shows an unphysical behaviour of the approximate pressure solution with large errors in a large portion of the pore network.

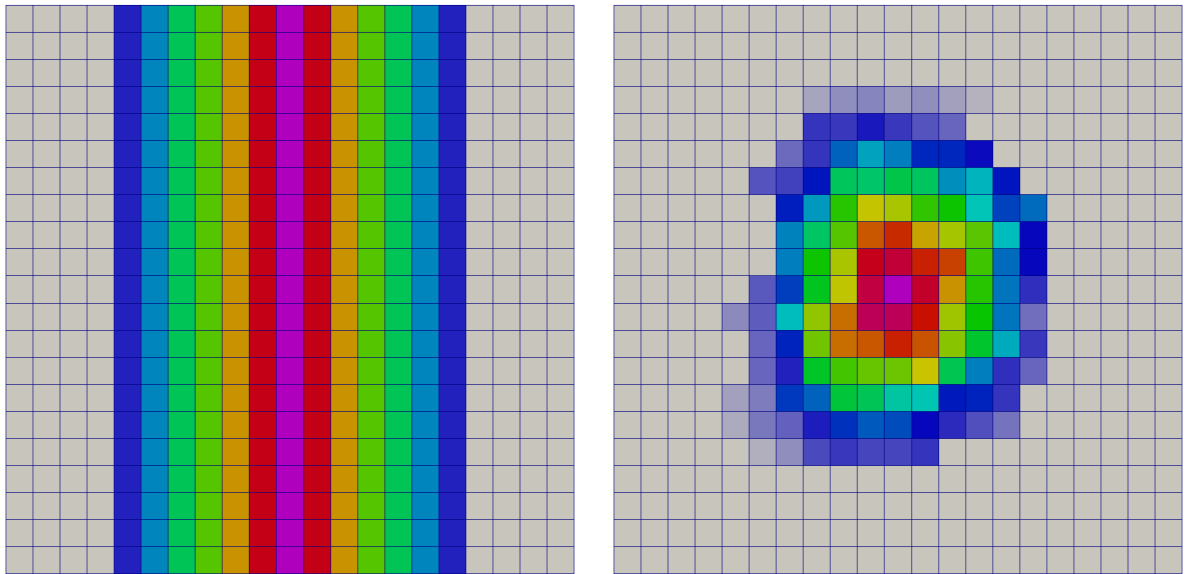


Figure A.3.: Basis functions obtained with a direct method and a subsequent correction step for a structured system with three coarse partitions (left) and for the unstructured network introduced in Figure 2.5 (right). The colour scheme is identical to the one of Figure 2.9.

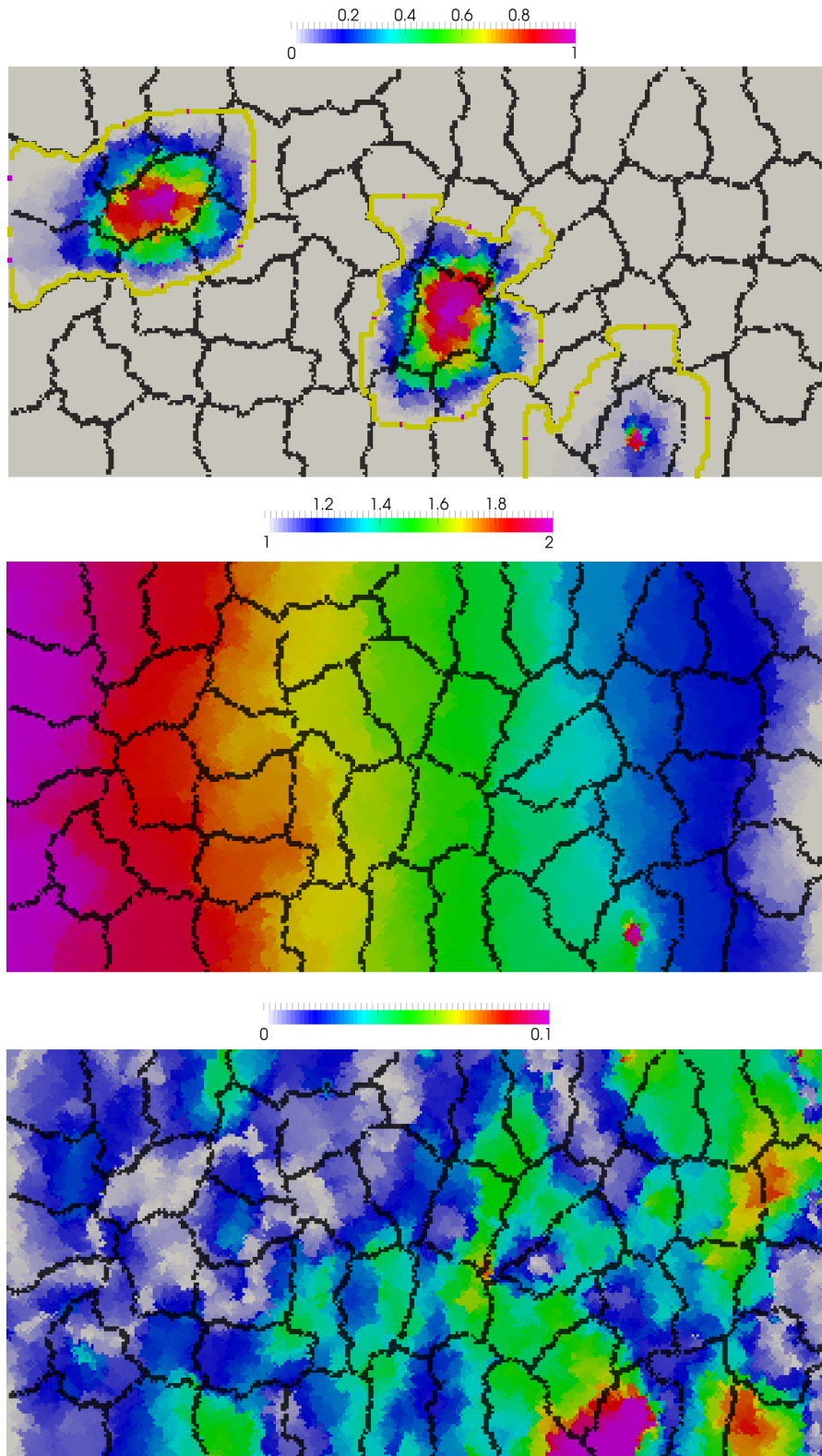


Figure A.4.: Basis functions (top), approximate pressure solution p^{approx} (middle) and relative error ϵ (bottom) resulting from a direct calculation of the prolongation operator for a heterogeneous test case.

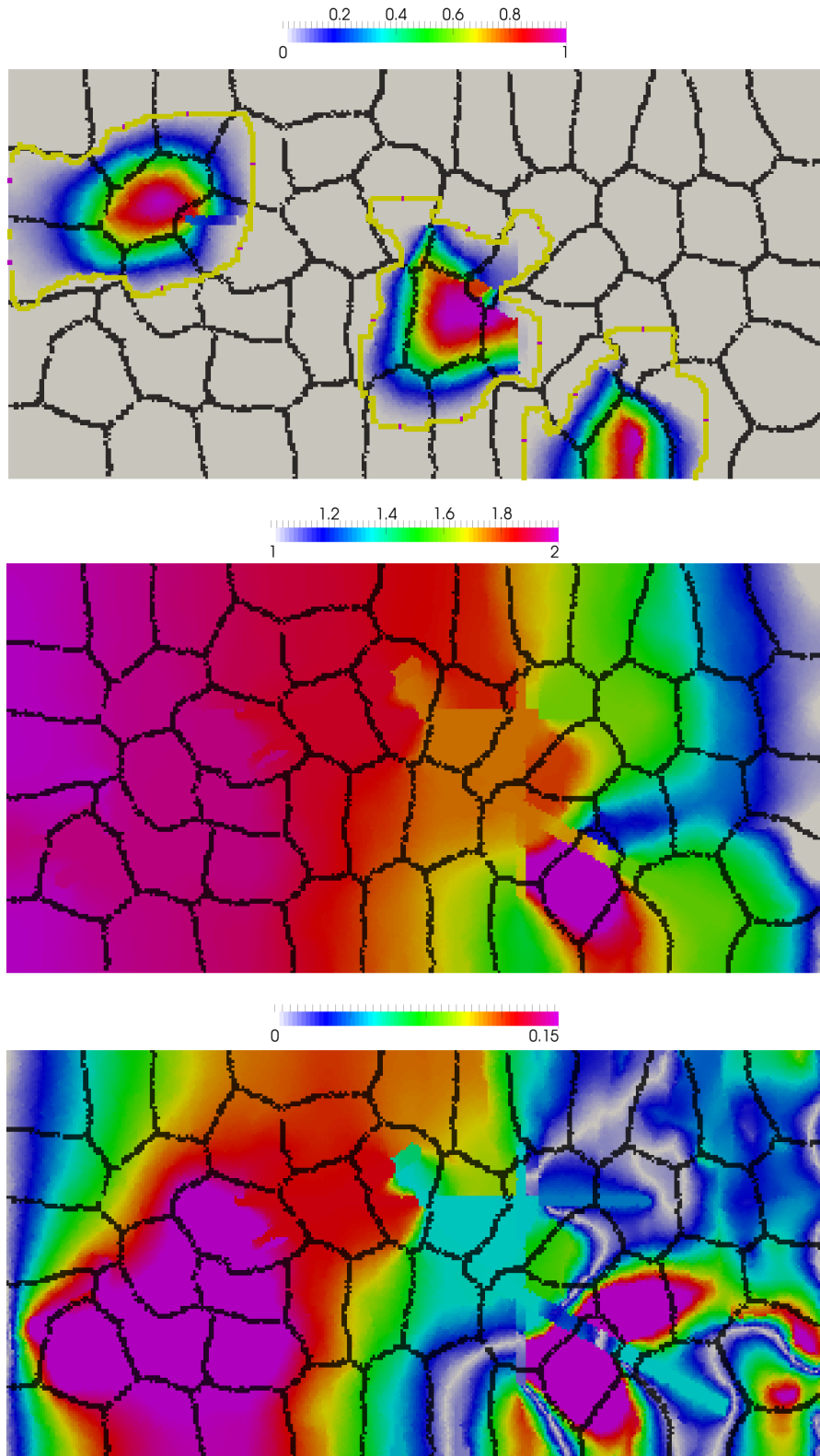


Figure A.5.: Basis functions (top), approximate pressure solution p^{approx} (middle) and relative error ϵ (bottom) resulting from a direct calculation of the prolongation operator for a channel test case.

B. Multilevel MsRSB

It is common in multigrid methods to use more than one coarse level to increase convergence speed in an iterative multiscale formulation.

B.1. Numerical Modelling

In the following it is shown how the black-box formulation of the MsRSB method introduced in Chapter 2 can be extended to a three-level multiscale method for certain systems. First, a symmetrical coarse system matrix \mathbf{A}_*^c is computed, i.e.

$$\mathbf{A}_*^c = \mathbf{P}^T \mathbf{A}_* \mathbf{P}, \quad (\text{B.1})$$

where \mathbf{A}_* is the fine scale system matrix with the diagonal elements adjusted to ensure zero row-sum and \mathbf{P}^T the transpose matrix of \mathbf{P} . Note that \mathbf{A}_*^c is different to \mathbf{A}^c introduced in Equation (2.3). By following the steps presented in Sections 2.2 to 2.5 a set of coarse scale prolongation and restriction operators \mathbf{P}_{II} and \mathbf{R}_{II} are then determined from \mathbf{A}_*^c and used to obtain a system matrix \mathbf{A}_{II}^c on the next coarser level, i.e.

$$\mathbf{A}_{II}^c = \mathbf{R}_{II} \mathbf{A}^c \mathbf{P}_{II} = \mathbf{R}_{II} \mathbf{R} \mathbf{A} \mathbf{P} \mathbf{P}_{II}. \quad (\text{B.2})$$

An approximate fine scale solution is computed analogously to (2.4) by subsequently applying \mathbf{P} and \mathbf{P}_{II} to a pressure solution obtained on the coarsest scale, i.e.

$$\mathbf{p} \approx \mathbf{p}^{approx} = \mathbf{P} \mathbf{P}_{II} (\mathbf{A}_{II}^c)^{-1} \mathbf{R}_{II} \mathbf{R} \mathbf{q}. \quad (\text{B.3})$$

For the case of an iterative three-level multiscale formulation, the calculation of the residuals and the smoothing step on the finest level are similar to (2.27) and (2.28). However, instead of directly solving for the coarse correction according to (2.29), the residuals $\mathbf{r}^{c(n)}$ on the coarse scale are computed, i.e.

$$\mathbf{r}^{c(n)} = \mathbf{R}(\mathbf{r}^{(n)} - \mathbf{A} \mathbf{y}^{(n)}) \quad (\text{B.4})$$

and smoothed, i.e.

$$\mathbf{y}^{c(n)} = \mathcal{S}(\mathbf{r}^{c(n)}), \quad (\text{B.5})$$

B.2. Numerical Results

e.g. by using a few Jacobi or ILU iterations analogously to (2.28). The coarse correction $\mathbf{e}_{II}^{c(n)}$ is then computed on the coarsest scale by a direct solve, i.e.

$$\mathbf{e}_{II}^{c(n)} = (\mathbf{A}_{II}^c)^{-1} \mathbf{R}_{II}(\mathbf{r}^{c(n)} - \mathbf{A}^c \mathbf{y}^{c(n)}). \quad (\text{B.6})$$

The multiscale cycle is completed with the update of the solution, i.e.

$$\mathbf{p}^{(n+1)} = \mathbf{p}^{(n)} + \mathbf{y}^{(n)} + \mathbf{P} \mathbf{y}^{c(n)} + \mathbf{P} \mathbf{P}_{II} \mathbf{e}_{II}^{c(n)}. \quad (\text{B.7})$$

The above presented formulation is applicable and extendable to even more coarse levels as long as the off-diagonals of the coarse level matrices, e.g. of \mathbf{A}_*^c for the second level, are ≤ 0 . Off-diagonals > 0 may result in negative basis functions that lead to non-physical solutions. Therefore the multilevel multiscale method cannot generally be used for arbitrary pore networks.

B.2. Numerical Results

In the following the results of a study conducted on a pore network with 2'000'000 pores arranged on a structured 4000 x 500 fine scale grid are presented. The mean transmissibility is $k = 1$ with a standard deviation $k_{std} = 0.3$ and additionally, four channels with either $k_{channel} = 100$ or $k_{channel} = 10$ are added to the domain as shown in Figure B.1 (top). The partitioning into 8005 intermediate and 41 coarse blocks is indicated in Figure B.1 (bottom) with grey and black lines, respectively, only for the very left part of the domain. Note that the number of partitions is chosen this way since the grid partitioner Metis produces disconnected blocks for the initially chosen numbers of 8000 and 40 coarse blocks.

The iterative performance of the three-level multiscale method is investigated by comparing the convergence histories of ϵ_{RMS} and ϵ_{MAX} to the ones obtained with a two-level implementation. Note that all prolongation operators are converged to $d_{crit} = 10^{-4}$ and that five Jacobi iterations per multiscale cycle were used for the intermediate and fine scale smoothers in the three-level case. For the two-level case, ten Jacobi steps were applied.

It is expected that a three-level method converges faster than a two-level solver if the number of blocks on the coarsest scale are identical for both cases, e.g. 41 in the here considered test case. This is confirmed by comparing the convergence histories ‘3-level’ and ‘2-level 41’ shown in Figure B.2 for the first 350 iteration steps. The iterations on the intermediate scale enable a much faster convergence than using prolongation and restriction operators that directly map between the finest and coarsest scales.

However, if the direct solve in the two-level case is done on the intermediate level, thus on the level with 8005 blocks, the three-level method is clearly outperformed, as can be seen by considering the convergence rate ‘2-level 8005’ in the Figure. Since the intermediate system can be solved with a direct method without difficulty, the two-level method is also faster regarding the computation time.

B.2. Numerical Results

It is expected that a three-level implementation is an option if much larger fine scale systems are used and if the coarse system cannot be solved directly, anymore. However, this case is not further investigated here and may be a subject of further studies.

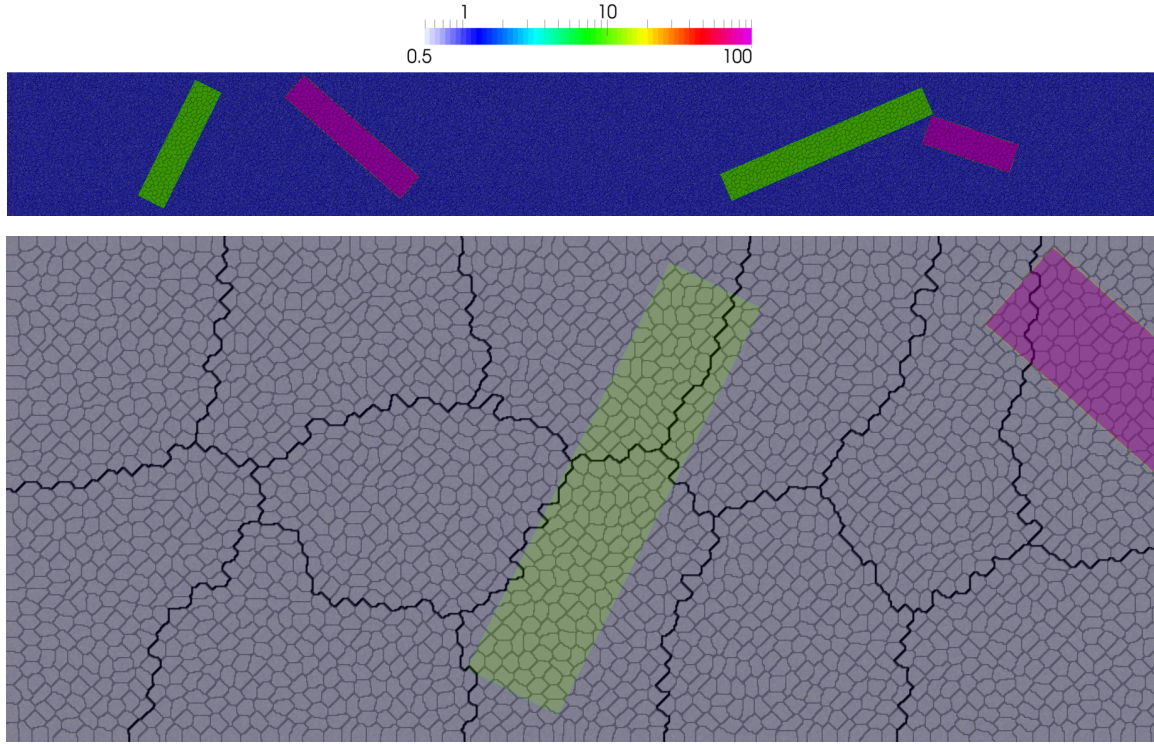


Figure B.1.: Transmissibility field (top) and partitioning into coarse and intermediate grids (bottom, shown only for the left part of the domain).

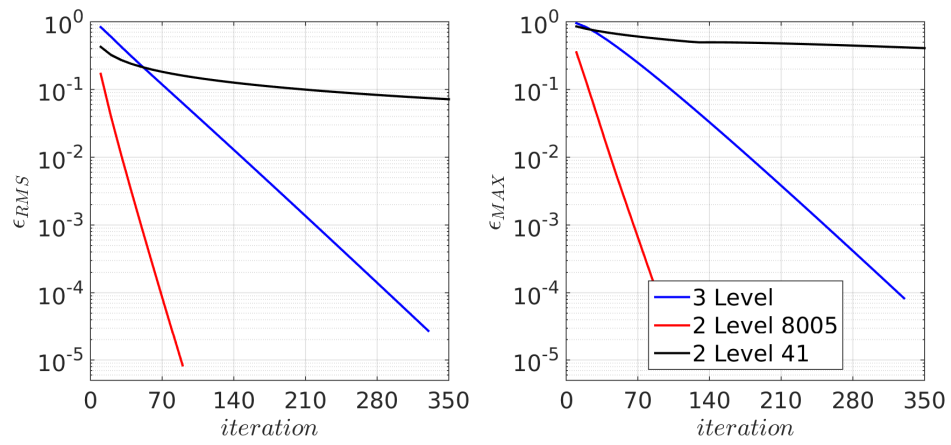


Figure B.2.: Convergence history of ϵ_{RMS} (left) and ϵ_{MAX} (right) for a 3-Level and 2-Level iterative multiscale formulation.



Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of originality

The signed declaration of originality is a component of every semester paper, Bachelor's thesis, Master's thesis and any other degree paper undertaken during the course of studies, including the respective electronic versions.

Lecturers may also require a declaration of originality for other written papers compiled for their courses.

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

Title of work (in block letters):

Multiscale Flow Solver for Unstructured Networks

Authored by (in block letters):

For papers written by groups the names of all authors are required.

Name(s):

Epp

First name(s):

Robert Meinrad

With my signature I confirm that

- I have committed none of the forms of plagiarism described in the '[Citation etiquette](#)' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

Place, date

Winterthur, 25.10.2016

Signature(s)

For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.