Algebraic Multigrid (AMG) for Saddle Point Systems

Dissertation

zur

Erlangung des Doktorgrades (Dr. rer. nat.)

der

Mathematisch–Naturwissenschaftlichen Fakultät

der

Rheinischen Friedrich–Wilhelms–Universität Bonn

vorgelegt von

Bram Metsch

aus

Haarlem

Bonn 2013

Angefertigt mit Genehmigung der Mathematisch–Naturwissenschaftlichen Fakultät der Rheinischen Friedrich–Wilhelms–Universität Bonn

1. Gutachter: Prof. Dr. Michael Griebel

2. Gutachter: Prof. Dr. Carsten Burstedde

Tag der Promotion: 03.07.2013

Erscheinungsjahr: 2013

Abstract

We introduce an algebraic multigrid method for the solution of matrices with saddle point structure. Such matrices e.g. arise after discretization of a second order partial differential equation (PDE) subject to linear constraints.
Algebraic multigrid (AMG) methods provide optimal linear solvers for many applications in science, engineering or economics. The strength of AMG is the automatic construction of a multigrid hierarchy adapted to the linear system to be solved. However, the scope of AMG is mainly limited to symmetric positive definite matrices. An essential feature of these matrices is that they define an inner product and a norm. In AMG, matrix-dependent norms play an important role to investigate the action of the smoother, to verify approximation properties for the interpolation operator and to show convergence for the overall multigrid cycle. Furthermore, the non-singularity of all coarse grid operators in a AMG hierarchy is ensured by the positive definiteness of the initial fine level matrix.
Saddle point matrices have positive and negative eigenvalues and hence are indefinite. In consequence, if conventional AMG is applied to these matrices, the method will not always converge or may even break down if a singular coarse grid operator is computed. In this thesis, we describe how to circumvent these difficulties and to build a stable saddle point AMG hierarchy. We restrict ourselves to the class of Stokes-like problems, i.e. saddle point matrices which contain a symmetric positive definite submatrix that arises from the discretization of a second order PDE.
Our approach is purely algebraic, i.e. it does not require any information not contained in the matrix itself. We identify the variables associated to the positive definite submatrix block (the so-called velocity components) and compute an inexact symmetric positive Schur complement matrix for the remaining degrees of freedom (in the following called pressure components). Then, we employ classical AMG methods for these definite operators individually and obtain an interpolation operator for the velocity components and an interpolation operator for the pressure matrix.
The key idea of our method is to not just merge these interpolation matrices into a single prolongation operator for the overall system, but to introduce additional couplings between velocity and pressure. The coarse level operator is computed using this "stabilized" interpolation operator. We present three different interpolation stabilization techniques, for which we show that they resulting coarse grid operator is non-singular. For one of these methods, we can prove two-grid convergence. The numerical results obtained from finite difference and finite element discretizations of saddle point PDEs demonstrate the practical applicability of our approach.

Contents

1. Introduction

Many scientific, engineering or economic processes can be described using a partial differential equation (PDE). In general form, a second order linear partial differential equation can be written as [Bra97]

$$-\sum_{i,j=1}^{d} \mathsf{a}_{ij}(\mathsf{x}) \frac{\partial}{\partial \mathsf{x}_i} \frac{\partial}{\partial \mathsf{x}_j} \mathsf{u}(\mathsf{x}) + \sum_{i=1} \mathsf{b}_i(\mathsf{x}) \frac{\partial}{\partial \mathsf{x}_i} \mathsf{u}(\mathsf{x}) + \mathsf{c}(\mathsf{x})\mathsf{u}(\mathsf{x}) = \mathsf{f}(\mathsf{x}) \tag{1.1}$$

where the coefficient functions a_{ij} $(i, j = 1, \ldots, d)$, b_i $(i = 1, \ldots, d)$, c, the solution u and the right hand side f map from the domain $\Omega \subset \mathbb{R}^d$ into the set of real numbers $\mathbb{R}$. A well-known example for a PDE is Poisson's equation on a domain $\Omega \subset \mathbb{R}^d$,

$$-\Delta \mathsf{u} = \mathsf{f} \text{ in } \Omega, \quad \mathsf{u} = \mathsf{g} \text{ on } \partial\Omega. \tag{1.2}$$

This PDE is e.g. used in electrostatics to find an electric potential for a given charge distribution. It is often considered as the prototype elliptic partial differential equation. (A PDE is called elliptic at x if the matrix $\mathsf{A}(\mathsf{x}) = (\mathsf{a}_{ij}(\mathsf{x}))_{i,j=1}^d$ is positive definite.)
So far, we have considered scalar unknowns u. In many cases, the unknown is a vector-valued function, i.e. $\mathsf{u} : \Omega \to \mathbb{R}^d$. Then, we are no longer dealing with a single partial differential equation of the form (1.1), but rather with a *system of partial differential equations.* A prominent example is given by the incompressible Navier–Stokes equations, that describe the velocity $\mathsf{u} : (0, T] \times \Omega \to \mathbb{R}^d$ and the pressure $\mathsf{p} : (0, T] \times \Omega \to \mathbb{R}^d$ of a fluid in a domain $\Omega \subset \mathbb{R}^d$ $(d = 2, 3)$ subject to an external force f at the time $t \in (0, T]$,

$$\begin{aligned} \frac{\partial}{\partial t}\mathsf{u} - \nu\nabla \cdot (\nabla \mathsf{u}) + (\mathsf{u} \cdot \nabla)\,\mathsf{u} + \nabla \mathsf{p} &= \mathsf{f}, & (1.3) \\ -\nabla \cdot \mathsf{u} &= 0. & (1.4) \end{aligned}$$

Here (1.3) already contains a system of d scalar (non-linear) partial differential equations (one for each spatial component of the velocity vector field), while (1.4) is a scalar equation.
A simpler, linear form of (1.3)–(1.4) can be used if the fluid has high viscosity, i.e. $\nu \to \infty$. Then, the non-linear term $(\mathsf{u} \cdot \nabla)\,\mathsf{u}$ can be neglected. If we assume that u and p are stationary, i.e. do not depend on the time t, we obtain *Stokes' equations*,

$$\begin{aligned} -\nabla \cdot (\nabla \mathsf{u}) + \nabla \mathsf{p} &= \mathsf{f}, & (1.5) \\ -\nabla \cdot \mathsf{u} &= 0. & (1.6) \end{aligned}$$

Stokes' equations can be considered as the prototype of a *saddle point system.* The solution (u, p) of (1.5)–(1.6) is not the minimizer of a quadratic functional (as in the case of elliptic partial differential equations), but a saddle point.

The analytical solution of a partial differential equation is usually not available except in very rare cases. Instead, we aim to approximately solve the partial differential equation(s) numerically. To this end, we *discretize* the partial differential equation. We construct a mesh Ω_h for the domain Ω depending on a mesh width h and represent the values of the continuous unknown u and right hand side f by vectors u and f, where the entries of these vectors represent the value of the corresponding function at a certain point of the mesh as in finite difference discretization or a coefficient inside a finite element ansatz. In the case of a linear PDE, the differential operator is replaced by a matrix K whose entries reflect the coefficients a_{ij}, b_i, and c. Hence we have translated the linear partial differential equation into a linear system of equations $Ku = f$.
To obtain an approximate solution of the PDE, we have to solve a linear system of equations. Direct linear solvers like LU or QR decomposition are not the first choice in this area. Letting N denote the length of the vectors u and f, a direct method requires $O(N^3)$ floating point operations for the solution. In other words, if we double our problem size (e.g. if we request a finer resolution of the discretization) the compute time grows by a factor of 8. In addition, direct methods need to store the full $N \times N$ matrix, but the discretization matrices of partial differential equations are usually *sparse*, i.e. they only contain a small (fixed) number of non-zero entries per row independent of N due to the locality of differential operators. Sparse matrix storage formats allow us to exploit this structure and to efficiently use the memory available in our computer. The usage of a direct solver would impose a much more severe limit on the mesh width than the problem discretization itself.
From these arguments it is clear that we need a linear solver that only requires $O(N)$ compute time and storage. Iterative solvers that employ one or few applications of sparse matrix-vector computations per iteration can be carried out using $O(N)$ operations per step, but are not always able to reduce the error by a fixed factor independent of the problem size. For example, for the well-known Jacobi and Gauss-Seidel iterations the convergence speed depends on the condition of the matrix, which in turn depends on the PDE coefficients and the discretization width h. However, for symmetric positive definite matrices K these methods are good *smoothers*, i.e. they very quickly remove oscillating parts from the error and leave a smooth component.
The key idea of *multigrid methods* (MG) is that after smoothing, the low-frequency error can be represented using less degrees of freedom, $N_C < N$. To this end, after a few smoothing steps the current residual $r^{it} = f - Ku^{it}$ is transferred to a *coarse mesh* Ω_H, $f_H \leftarrow Rr^{it}$ using a linear restriction operator R, and a coarse linear system $K_H e_H = f_H$ is solved. The solution is then used to update the fine grid iterate, $u^{it} \leftarrow u^{it} + Pe_C$, where P denotes a linear prolongation operator. The coarse solution e_C itself can be computed recursively, i.e. after a few smoothing steps the resulting residual is transferred to an even coarser level and so on, until the system is so small that it can be directly solved.
The question remains how the coarse mesh Ω_H and the matrices K_H, P_H and R_H are constructed. In geometric multigrid, Ω_H can e.g. be obtained by doubling the mesh width, $H = 2h$. The *coarse grid matrix* K_H is then computed as discretization of the

PDE on this coarse mesh, while for the *restriction* operator R and the *interpolation or prolongation* matrix P we can e.g. use linear interpolation or injection.
A drawback of geometric multigrid is that the construction of the coarse mesh and the coarse matrix require the knowledge of the underlying PDE as well as the geometry of Ω. Furthermore, a simple isotropic coarsening of the mesh might not well reflect the structure of the smoothed error, especially in the case of anisotropic or jumping PDE coefficients a_{ij} (where a “smooth error” may still contain certain oscillations). To circumvent these difficulties, *algebraic multigrid methods (AMG)* have been developed.
The key ingredients to AMG are operator-dependent interpolation operators P together with a Galerkin coarse grid operator $K_H := P^T K P$ [Den82] and automatic operator-dependent coarsening [BMR82, BMR84]. These components allow us to construct robust solvers for a wide class of problems, especially if K is a positive definite M-matrix. Such matrices often arise in the discretization of second order partial differential equations.
All information needed to build the multigrid hierarchy is obtained from the matrix on the finest level $K_1 = K$. In order to apply the multigrid cycle, we need to employ a *setup phase*. On each level l, we need to carry out the following steps depending on the matrix K_l:

- Construct the next coarser mesh Ω_{l+1}.
- Build a prolongation matrix P_l.
- Compute the coarse grid matrix $K_{l+1} \leftarrow P_l^T K_l P_l$.

We stop if the matrix K_{l+1} is small enough, i.e. it is feasible to be treated by a direct method (or few iterations of the smoother). With this AMG hierarchy available, we can now start the multigrid cycle.
In the past years, AMG has been further developed. One notable development is *smoothed aggregation* [VMB94, VMB96], where the coarse mesh is obtained from the fine mesh by aggregating fine nodes instead of picking coarse nodes as a subset of the fine nodes. The Bootstrap AMG approach [Bra01, FV04, BBKL11] tries to generalize AMG ideas to a wider class of problems. Many efforts have also been put in the parallelization of the AMG setup case, in particular the parallel coarse grid generation, see e.g. [MY06] for an overview.
AMG was at first developed for scalar systems, but already in [Rug86] it was demonstrated that it can also be used for the solution of systems of elliptic partial differential equations (here, a problem arising from linear elasticity). A further approach to AMG for linear elasticity, where the rigid body modes are preserved in the AMG hierarchy was given by [Oel01, GOS03]. A systematic introduction to AMG for systems of elliptic PDEs as well as applications in semiconductor industry is given in [Cle04].
All of these system AMG techniques rely on the fact that after discretization the matrix K is symmetric positive definite, as in the case of a single elliptic PDE, and thus defines an inner product and norm. This is an essential feature for AMG:

- For symmetric positive definite matrices, we have a smoothing property for the Jacobi Gauss–Seidel and SOR iterations,

- if in addition the matrix has M-matrix structure (which is the case in many PDE applications), we know how the smoothed error behaves w.r.t. the matrix entries,
- the coarse grid matrix $K_H = P^T K P$ computed by the Galerkin product is also symmetric positive definite and thus non-singular,
- we have a variational property for the coarse grid operator, which, together with the smoothing property, guarantees convergence.

A saddle point matrix K is indefinite and we cannot rely on the properties above. Instead, we must

- identify suitable smoothers for saddle point systems,
- construct restriction, interpolation and coarse grid operators such that the coarse grid matrix is invertible,
- show convergence in terms of a suitable norm.

Still, system AMG is applied to saddle point matrices even if all the points mentioned above are not clear. We refer e.g. to [LSAC08] for a CFD example or to [GKW11], where a smoothed aggregation AMG is extended to a full AMG solver for a fluid-structure interaction problem.
There are few approaches to AMG dedicated to saddle point problems. A smoothed aggregation method for contact problems is presented in [Ada04]. The method introduced in [LOS04] was used to solve a saddle point system arising from a particle method, where the boundary conditions cannot be easily eliminated and are included as constraints.
In [Wab03, Wab04, Wab06], an algebraic multigrid method for the Oseen equations is constructed. The Oseen equations arise from a linearization of the Navier-Stokes equations. This AMG method is used inside a finite element computational fluid dynamics (CFD) solver. The stability (invertibility) of the coarse systems is shown using geometrical properties.
So far, there exists no AMG for saddle point matrices that is truly algebraic, i.e. does not need geometric information for the hierarchy setup or the stability proofs and hence can be applied to a wide class of problems. The method introduced in this thesis fills this gap.

Contribution of this thesis We introduce an algebraic multigrid (AMG) method for Stokes-type saddle point matrices of the form

$$\mathcal{K} : \begin{pmatrix} \mathcal{V}, \\ \mathcal{W} \end{pmatrix} \to \begin{pmatrix} \mathcal{V} \\ \mathcal{W} \end{pmatrix}, \qquad \mathcal{K}x = y$$

where

$$\mathcal{K} = \begin{pmatrix} A & B^T \\ B & -C \end{pmatrix}, \; x = \begin{pmatrix} u \\ p \end{pmatrix} \text{ and } y = \begin{pmatrix} f \\ g \end{pmatrix}.$$

In the following, calligraphic letters denote operators defined on the space $\mathcal{V} \times \mathcal{W}$. Here, "Stokes-type" means that $A \in \mathbb{R}^{N \times N}$ is the discretization of an (elliptic) second order partial differential equation, while $B \in \mathbb{R}^{M \times N}$ is e.g. given by a discrete divergence operator. In other words, we are looking for the solution of a partial differential equation subject to a linear constraint equation. In consequence, we can assume that we know how to construct the algebraic multigrid components (coarse grid, interpolation, restriction and coarse grid operator) for the A matrix part.
In this thesis, we describe how to extend AMG to the whole coupled matrix $\mathcal{K}$. Our method will be purely algebraic, i.e. we do not require any information from the geometry or the discretization mesh to carry out the AMG setup procedure.
As with every AMG method, we first choose a suitable smoother. The first alternative is an inexact Uzawa method introduced in [SZ03]. Each relaxation step is given by

$$\begin{aligned} u^* &= u^{it} + \hat{A}^{-1}\left(f - Au^{it} - B^T p^{it}\right) \\ p^{it+1} &= p^{it} + \hat{S}^{-1}\left(Bu^* - Cp^{it} - g\right) \\ u^{it+1} &= u^{it} + \hat{A}^{-1}\left(f - Au^{it} - B^T p^{it+1}\right), \end{aligned}$$

where $\hat{A}$ is an easily invertible matrix such that $\hat{A} - A$ is symmetric positive definite. A common choice for $\hat{A}$ is the scaled (block) diagonal of A. The symmetric positive definite matrix $\hat{S}$ is chosen such that $\hat{S} - B\hat{A}^{-1}B^T - C$ is also symmetric positive definite.
The second option is to employ an algebraic Vanka smoother [Van86, SZ03]. Here, we solve, for $j = 1, \ldots, M$, small linear systems of the form

$$\begin{pmatrix} \hat{A}_j & B_j^T \\ B_j & \hat{C}_j \end{pmatrix} \begin{pmatrix} u_{(j)} \\ p_{(j)} \end{pmatrix} = \begin{pmatrix} f_{(j)} \\ g_{(j)} \end{pmatrix},$$

where $\hat{A}_j \in \mathbb{R}^{N_j \times N_j}$, $B_j \in \mathbb{R}^{M_j \times N_j}$, $\hat{C}_j \in \mathbb{R}^{M_j \times M_j}$ and $N_j \ll N$ as well as $M_j \ll M$. We use the solution vectors $u_{(j)}$ and $p_{(j)}$ to update the iterates u^{it} and p^{it}. The small right hand side vectors $f_{(j)}$ and $g_{(j)}$ are derived from the residuals $f - Au^{it} - B^T p^{it}$ and $g - Bu^{it} + Cp^{it}$, respectively. These residuals can either be computed after each subsystem solve (multiplicative application) or after all subsystems have been visited (additive application). For a special choice of $\hat{A}_j$, B_j and $\hat{C}_j$, the additive version coincides with the inexact Uzawa smoother.
The error propagation operator of the inexact Uzawa method motivates us to consider the block diagonal matrix

$$\begin{pmatrix} A & 0 \\ 0 & B\hat{A}^{-1}B^T + C \end{pmatrix}.$$

as starting point for the construction of the coarse grid and interpolation operators for the saddle point matrix $\mathcal{K}$. We hence compute a coarse grid and a tentative prolongation $R_{\mathcal{V}}^T$ for the operator A and a coarse grid and an interpolation matrix $R_{\mathcal{W}}^T$ for the symmetric positive semi-definite matrix $B\hat{A}^{-1}B^T + C$ to obtain a block diagonal interpolation operator

$$\mathcal{R}^T = \begin{pmatrix} R_{\mathcal{V}}^T & 0 \\ 0 & R_{\mathcal{W}}^T \end{pmatrix}.$$

We can however not simply use the Galerkin ansatz for the coarse grid operator,

$$\mathcal{K}^C \leftarrow \begin{pmatrix} R_{\mathcal{V}} & 0 \\ 0 & R_{\mathcal{W}} \end{pmatrix} \begin{pmatrix} A & B \\ B^T & -C \end{pmatrix} \begin{pmatrix} R_{\mathcal{V}}^T & 0 \\ 0 & R_{\mathcal{W}}^T \end{pmatrix}$$

as this can lead to a singular matrix.
For saddle point problems the stability (and thus the invertibility) of the discrete system is usually shown by means of a so-called *inf-sup condition*: Let there exist constants $c, d > 0$ such that

$$\sup_{0 \neq u \in \mathcal{V}} \frac{uB^T p}{\|u\|_A} \geq c\|p\|_{\mathcal{W}} - d \left(p^T C p\right)^{\frac{1}{2}} \text{ for all } p \in \mathcal{W}.$$

We use this inequality on the finest level $l = 1$ (i.e. for the original matrix $\mathcal{K} = \mathcal{K}_1$) to show an inf-sup condition for all levels $l = 2, \ldots, L$ of the AMG hierarchy by induction. To this end, we need to construct coupled interpolation operators $\mathcal{P}$. Here, we introduce three variants.

1. The first variant is to apply the inexact Uzawa smoother to the tentative block diagonal operator $\mathcal{R}^T$. We obtain

$$\mathcal{P} = \begin{pmatrix} I & -\hat{A}^{-1}B^T \\ 0 & I \end{pmatrix} \begin{pmatrix} I & 0 \\ \hat{S}^{-1}B & I \end{pmatrix} \begin{pmatrix} I - \hat{A}^{-1}A & 0 \\ 0 & I - \hat{S}^{-1}\left(B\hat{A}^{-1}B^T + C\right) \end{pmatrix} \begin{pmatrix} R_{\mathcal{V}}^T & 0 \\ 0 & R_{\mathcal{W}}^T \end{pmatrix}$$

   and a “Petrov-Galerkin” coarse grid operator

$$\mathcal{K}^C \leftarrow \mathcal{R}\mathcal{K}\mathcal{P}.$$

   In this case, we can show not only the stability of the coarse matrix $\mathcal{K}^C$, but also two-grid convergence. On the downside, this variant introduces many non-zero entries in the prolongation matrix $\mathcal{P}$ and hence also in the coarse matrix $\mathcal{K}^C$. To circumvent this problem, we introduce two sparser alternatives.

2. The second option is to compute

$$\mathcal{P} = \begin{pmatrix} I & -\hat{A}^{-1}B^T \\ 0 & I \end{pmatrix} \begin{pmatrix} R_{\mathcal{V}}^T & 0 \\ 0 & R_{\mathcal{W}}^T \end{pmatrix},$$

   and use a Galerkin coarse grid operator $\mathcal{K}^C \leftarrow \mathcal{P}^T\mathcal{K}\mathcal{P}$. The additional prolongation coupling term $-\hat{A}^{-1}B^T$ implies an additional stability term in the lower right block of the coarse matrix $\mathcal{K}^C$.

3. Finally, we can employ the decomposition of the grid for the velocity variables $u \in \mathbb{R}^N$ into coarse and fine variables to obtain an even sparser variant of the

last stabilization technique. To this end, letting $\hat{A}_{FF}$ and B_F denote the portion of $\hat{A}$ and B associated with the fine velocity variables and $R^T_{\mathcal{V},CF}$ denote the interpolation from coarse to fine points, we compute

$$\mathcal{P} = \begin{pmatrix} I_{FF} & 0 & -\hat{A}^{-1}_{FF}B^T_F \\ 0 & I_{CC} & 0 \\ 0 & 0 & I \end{pmatrix} \begin{pmatrix} R^T_{\mathcal{V},CF} & 0 \\ I_{CC} & 0 \\ 0 & R^T_{\mathcal{W}} \end{pmatrix}.$$

Again we use the Galerkin coarse grid operator $\mathcal{K}^C \leftarrow \mathcal{P}^T\mathcal{K}\mathcal{P}$.

An important ingredient for all of the stability proofs is the approximation property for the velocity variables, which is expressed by the inequality

$$\|u_F - R^T_{FC}u_C\|^2_{D,F} \leq \tau\|u\|^2_A. \tag{1.7}$$

Here, we assume again that u is decomposed into its coarse and fine parts u_C and u_F, while R^T_{FC} denotes the interpolation from the coarse to the fine variables. Letting D_F be the part of the diagonal of A associated with the fine grid points, we define $\|u_F\|^2_{D,F} := u_F^T D_F u_F$ and the energy norm $\|u\|^2_A := u^T A u$. Inequality 1.7 needs to be shown for the tentative velocity interpolation matrix

$$R^T_{\mathcal{V}} = \begin{pmatrix} R^T_{FC} \\ I_{CC} \end{pmatrix}.$$

In case of the direct and classical AMG interpolation techniques, the approximation property was already shown in [RS87, Stü99]. For more complex schemes (modified classical interpolation, extended(+i) interpolation, Jacobi interpolation), we prove the approximation property in this thesis.
To summarize, we are able to construct an algebraic multigrid hierarchy for saddle point problems based on AMG techniques applied to the matrices A and $B\hat{A}^{-1}B^T + C$, where the coarse grids and tentative transfer operators can be set up *independently* for each of these positive (semi-)definite matrices. The final interpolation operator introduces an additional coupling, which helps to ensure the stability of the coarse grid.

Outline of this thesis

- In Chapter 2 we give a systematic introduction to algebraic multigrid (AMG) methods. After a short recapitulation of iterative methods and, in particular, geometric multigrid, we focus on classical AMG by Ruge and Stüben [RS87]. We describe the algorithms of the AMG setup phase as well as their mathematical background. A special focus is placed on the derivation of approximation properties for the interpolation operators for modified classical, extended(+i) and Jacobi interpolation in Section 2.8.
  After this introduction, we describe the convergence theory for non-symmetric AMG by [Not10]. This convergence theory will form the basis for the two-level convergence proof of one of our saddle point AMG approaches.

For sake of completeness, we also shortly introduce three alternative AMG methods, namely AMGe, smoothed aggregation, and Bootstrap AMG.
We conclude this chapter with a summary of algebraic multigrid for systems of elliptic PDEs. Following [Cle04], we introduce unknown-based and point-based AMG and give smoothing and approximation properties for these techniques.

- In Chapter 3, we discuss the difficulties of parallel AMG, especially concerning the coarsening process. Various parallel coarsening algorithms are described together with their respective advantages and disadvantages. Then, we recapitulate our results from [Met04, GMOS06, GMS06], where we have introduced the parallel coarse grid classification (CGC) algorithm and give an extension for large numbers of processors first introduced in [GMS08].

- Chapter 4 forms the core of this thesis. After an introduction to saddle point systems (in particular, Stokes-like equations), we discuss the role of the inf-sup condition for both the continuous as well as the discrete equations. We present finite element and finite difference discretization techniques that satisfy an inf-sup condition and then give an overview of previous approaches to algebraic multigrid for saddle point systems.
We start the description of our saddle point AMG with an inexact Uzawa method and an algebraic Vanka smoother introduced by [SZ03]. Then, we introduce our algebraic stabilization techniques and show that the coarse grid operator remains stable. We also give the two-grid convergence proof for one of these methods.

- In Chapter 5 we show the numerical performance of our saddle point AMG. We conduct experiments with both finite difference and finite element discretizations of saddle point systems.

- We summarize our results in 6 and give an outlook for further research.

Acknowledgments At this point I would like to thank all people who have contributed to the completion of this thesis. First, my gratitude goes to my adviser Prof. Dr. Michael Griebel who introduced me to this theme and provided the excellent working conditions at the institute. Also, I want to thank Prof. Dr. Carsten Burstedde for providing the second opinion for this thesis as well as giving me access and introducing me to the Rhea mantle convection software.
A cordial "thank you" is directed to all colleagues at the Institute for Numerical Simulation, in particular to my former office mate Dr. Markus Holtz. I would like to thank Jutta Adelsberger, Markus Burkow, Margrit Klitz, Dr. Christian Rieger, Alexander Rüttgers, Ralph Thesen, and Peter Zaspel for proofreading parts of this thesis. A special thanks goes to Prof. Dr. Marc Alexander Schweitzer and Dr. Christian Rieger and for providing many insightful ideas and the numerous fruitful mathematical discussions.
A substantial part of the work described in this thesis was supported by the Collaborative Research Center (SFB) 611 at the University of Bonn, which is gratefully appreciated.
On a personal level, I will always be thankful to my wife Ute and my children Hendrik and Joris for their long patience and their understanding.

2. Algebraic Multigrid

In this chapter we introduce the construction of algebraic multigrid methods (AMG), where we focus on the *setup phase*, i.e. the construction of the multigrid hierarchy (coarse grids, coarse grid operators and transfer operators). We assume that a smoother is available and can be carried out on all levels using information from the respective system matrix only (e.g. simple relaxation schemes like Jacobi or Gauss-Seidel). We use the smoother and its error reduction behavior to construct the remaining AMG components. This is described in more detail in the following sections, see also [Stü99] and [Met04].

Before we proceed, we first introduce the notation used throughout this chapter. We denote the level index by $l = 1, \dots, L_{\max}$ where 1 is the finest and $L_{\max}$ the coarsest level. On each level l, we have a linear system of equations $A_l u_l = f_l$, or, in components,

$$\sum_{j \in \Omega^l} a^l_{ij} u^l_j = f^l_i \quad \left(i \in \Omega^l\right).$$

where Ω^l is the grid (index set) on level l and $N_l := |\Omega^l|$ denotes its cardinality. In the simplest case, Ω_l can be split disjointly into the set of *coarse grid points* C^l and *fine grid points* F^l, $\Omega^l = C^l \dot{\cup} F^l$. The set of coarse grid points defines the grid on the next coarser level: $\Omega^{l+1} := C^l$. The elements of Ω^{l+1} can also be formed by more advanced means, e.g. by forming unions of the members of Ω^l. To transfer information between two consecutive grids Ω^l and Ω^{l+1}, we introduce a *prolongation operator* $P_l : \Omega^{l+1} \to \Omega^l$ and a *restriction operator* $R_l : \Omega^l \to \Omega^{l+1}$. We omit the level index l if no mix-up is possible.

In many cases, the matrix A is *symmetric positive (semi-) definite*,

$$A = A^T \text{ and } x^T A x > (\geq)\ 0 \text{ for all } x \neq 0$$

We us the notation $A > 0$ for a symmetric positive definite A and $A \geq 0$ for a symmetric positive semi-definite operator. We write $A > B$ if $A - B$ is symmetric positive definite. $\lambda_{\max}(A)$ and $\lambda_{\min}(A)$ denote the largest and the smallest eigenvalue of a matrix A (if all eigenvalues of A are real). The smallest and largest singular value of A are denoted by $\sigma_{\min}(A)$ and $\sigma_{\max}(A)$.

We start this chapter with a short introduction to the multigrid concept and geometric multigrid methods. We first stress that the term “multigrid” does not refer to a single algorithm, but rather categorizes a whole class of methods that can be used to accelerate the speed of convergence of classical iterative solvers like Gauss-Seidel-iteration. Their key idea here lies in the observation that after a few steps of such a method, the error is “smooth” in the sense that it can be represented with less degrees of freedom than the original problem. Hence, this part of the error can be eliminated by solving a smaller linear system of equations. For a comprehensive introduction to multigrid methods, we refer to [TOS01].

2.1. Relaxation Solvers

Before we proceed to multigrid techniques, we first summarize some basic facts about Richardson-like iterative solution methods. In the simplest case, this iteration takes the form

$$u^{it+1} = u^{it} + \omega \left(f - Au^{it}\right)$$

for some damping parameter $\omega > 0$. In terms of the error $e^{it} = u - u^{it}$ the iteration can be rewritten as

$$e^{it+1} = Me^{it} = (I - \omega A)\, e^{it}.$$

From this equation it becomes clear that in the case of a symmetric, positive definite operator A, the iteration converges if and only if the spectral radius $\rho(M) = \rho(I - \omega A)$ (also called the *convergence factor* of M) is strictly smaller than one. This is the case if $\omega < \frac{2}{\lambda_{\max}(A)}$. To estimate the convergence speed, we also need to consider the (ℓ_2-) condition number $\kappa_2(A) = \frac{\lambda_{\max}(A)}{\lambda_{\min}(A)}$: The eigenvalues of the iteration are contained in the interval $[1 - \omega\lambda_{\max}(A), 1 - \omega\lambda_{\min}(A)]$. For large numbers of $\kappa_2(A)$, i.e. $\lambda_{\min}(A) \ll \lambda_{\max}(A)$, the largest eigenvalue of the iteration tends towards one and the convergence slows down. If however $\lambda_{\min}(A) \approx \lambda_{\max}(A)$, fast convergence is possible for a properly chosen relaxation parameter ω. In particular, for symmetric positive matrices A, the optimal relaxation parameter is given by ([Hac91], Theorem 4.4.3)

$$\omega^{opt} = \frac{2}{\lambda_{\max} + \lambda_{\min}} \tag{2.1}$$

which yields a convergence factor of

$$\rho(M_{\omega^{opt}}) = \frac{\kappa_2(A) - 1}{\kappa_2(A) + 1} = \frac{\lambda_{\max} - \lambda_{\min}}{\lambda_{\max} + \lambda_{\min}}. \tag{2.2}$$

This observation motivates to reformulate the problem to reduce the condition number $\kappa_2(A)$: Instead of solving the equation $Au = f$, we solve the equivalent equation

$$BAu = Bf,$$

where the non-singular matrix B is an approximation to A^{-1} that can be easily applied (or even explicitly constructed) such that the condition number $\kappa_2(BA) \ll \kappa_2(A)$. For example, we can take $B = D^{-1} = diag(A)^{-1}$ and obtain the *damped Jacobi relaxation*,

$$u^{it+1} = u^{it} + \omega D^{-1} \left(f - Au^{it}\right).$$

A sufficient criterion for convergence of the undamped ($\omega = 1$) iteration is given by ([Hac91], Theorem 4.4.11)

$$2D > A > 0.$$

If this condition is not fulfilled, a damping parameter $0 < \omega < 1$ is necessary such that $\frac{2}{\omega}D > A$ ([Hac91], Theorem 4.4.14). This is the case for most discretizations obtained from elliptical PDE problems.
A second commonly used choice for B is $B = (L + D)^{-1}$, where L is the strict lower triangular part of A,

$$L_{ij} = \begin{cases} a_{ij} \text{ if } j < i \\ 0 \text{ else.} \end{cases}$$

Inside the so-called *Gauss-Seidel iteration* $u^{it} \mapsto u^{it+1}$, the algorithm loops over the components u_i^{it} of u^{it} in the following way,

$$u_i^{it+1} = u_i^{it} + a_{ii}^{-1}\left(f_i - \sum_{j<i} a_{ij}u_j^{it+1} - \sum_{j\geq i} a_{ij}u_j^{it}\right). \tag{2.3}$$

Here, the new value for u_j is already used to determine u_i if $j < i$. Of course, instead of $(D + L)^{-1}$ one can also use $(D + U)^{-1}$ as preconditioner, where U is the strictly upper triangular part of A. Moreover, the decomposition of the non-diagonal part of A into L and $A - D - L$ can also be made depending on the underlying problem or the geometry of the discretized domain (for example, the iteration may run "checkerboard-wise" through the domain). The Gauss-Seidel iteration converges for any positive definite matrix A ([Hac91], Theorem 4.4.18). The damped variant, which is named *successive over-relaxation* (SOR), converges for any $0 < \omega < 2$ ([Hac91], Theorem 4.4.21). The convergence factor for any SOR scheme can be derived from the convergence factor of the Jacobi iteration applied to the same problem. This can be seen from Young's Theorem ([Hac91], Theorem 5.6.5), which not only applies to the point-wise Gauss-Seidel iteration described above but also to more general settings of the following form,

$$\begin{aligned} u^{it+1} &= M_\omega^{SOR}u^{it} + N_\omega^{SOR}f & (2.4)\\ A &= D - L - U,\ \hat{L} = D^{-1}L,\ \hat{U} = D^{-1}U & (2.5)\\ M_\omega^{SOR} &= (I - \omega\hat{L})^{-1}\left((1-\omega)I + \omega\hat{U}\right), N_\omega^{SOR} = \omega(I - \omega\hat{L})^{-1}D^{-1}. & (2.6) \end{aligned}$$

Theorem 2.1 *([Hac91], Theorem 5.6.5) Let the SOR iteration* (2.4) *satisfy the following conditions*

1. $0 < \omega < 2$
2. *The associated Jacobi iteration* $M^{JAC} = I - D^{-1}A$ *has only real eigenvalues.*
3. $\beta = \rho(M^{JAC}) < 1$
4. D *and* $I - \omega\hat{L}$ *are non-singular*
5. *The eigenvalues of* $z\hat{L} + \frac{1}{z}\hat{U}$, $z \in \mathbb{C} \setminus \{0\}$ *are independent of* z.

Then,

1. *The iteration* (2.4) *converges.*
2. *The convergence rate is given by*

$$\begin{aligned} \rho(M_\omega^{SOR}) &= \begin{cases} 1-\omega+\frac{\omega^2\beta^2}{2}+\omega\beta\sqrt{1-\omega+\frac{\omega^2\beta^2}{4}} \ \mathit{if}\ 0<\omega\le\omega_{opt} \\ w-1 \ \mathit{if}\ \omega_{opt}\le\omega<2 \end{cases} \\ \mathit{where}\ \omega_{opt} &= \frac{2}{1+\sqrt{1-\beta^2}}. \end{aligned}$$

3. *The convergence factor* $\rho(M_\omega^{SOR})$ *attains its minimum at* $\omega=\omega_{opt}$.
4. *For* $\omega\le\omega_{opt}$, $\rho(M_\omega^{SOR})$ *is an eigenvalue of* M_ω^{SOR}.
5. *If* $\omega\ge\omega_{opt}$, *the norm of all eigenvalues of* M_ω^{SOR} *is* $\omega-1$.

From this theorem we immediately see that in the case of undamped Gauss-Seidel iteration ($\omega=1$), we have $\rho(M_1^{SOR})=\beta^2$, i.e. the Gauss-Seidel method converges twice as fast as the Jacobi relaxation. For the optimal parameter ω_{opt} (which is always larger than one in nontrivial cases), we have

$$\rho(M_{\omega_{opt}}^{SOR})=\frac{1-\sqrt{1-\beta^2}}{1+\sqrt{1+\beta^2}}$$

In the following, we show the convergence behavior for the solution of a simple model problem. We consider Poisson's equation together with Dirichlet boundary conditions on a square,

$$-\Delta\mathsf{u} = \mathsf{f} \text{ in } \Omega=(0,1)^2, \tag{2.7}$$
$$\mathsf{u} = \mathsf{g} \text{ on } \partial\Omega, \tag{2.8}$$

where f and g are continuous functions and $\mathsf{u}\in C^2(\Omega)$ is the sought solution function. We discretize this equation on an equidistant grid Ω_h with mesh size h (see Figure 2.1(a) using a 5-point finite difference stencil,

$$\frac{1}{h^2}\begin{bmatrix} & -1 & \\ -1 & 4 & -1 \\ & -1 & \end{bmatrix}_h$$

and obtain a linear system of $N=(n-1)^2$ equations, where $n=1/h$, [1] which can be written in matrix-vector-form,

$$Au=f.$$

[1]Note that we have eliminated the equations on the discretized boundary

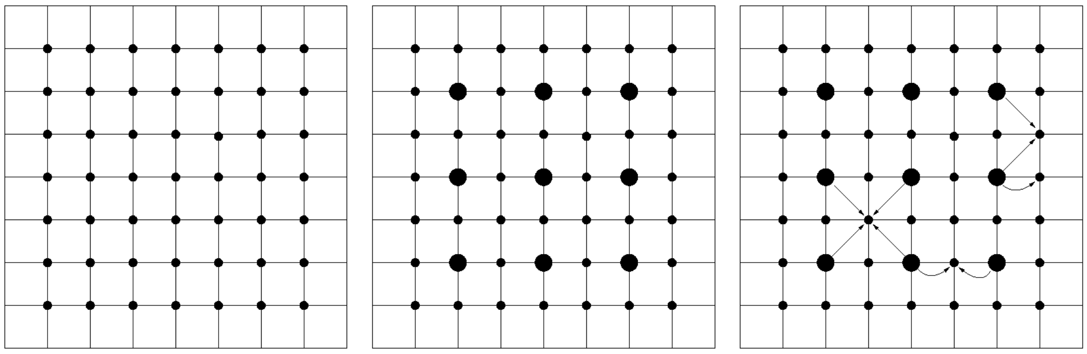


Figure 2.1.: Finite difference discretization grids for the model problem $\Delta u = f$ on the square $[0,1]^2$. Here, we have $h = 0.125$ and $n = 8$. The first figure shows the discretization on the finest level (the Dirichlet boundary conditions have already been eliminated), in the second figure we indicate the coarse grid points with the larger dots and the arrows in the third figure indicate how the interpolated values at the fine grid points are computed.

To estimate the convergence speed of the various iterations, we need to know the eigenvalues of the matrix A. These are given by

$$\lambda^{(k,l)} = \frac{4}{h^2}\left(\sin^2\left(\frac{k\pi h}{2}\right) + \sin^2\left(\frac{k\pi h}{2}\right)\right), \text{ for } k, l = 1, \ldots, n-1$$

with the respective eigenvectors

$$\phi_{ij}^{(k,l)} = \frac{h}{2}\sin(ik\pi h)\sin(jl\pi h).$$

From this we easily see that

$$\lambda_{\min}(A) = \lambda^{(1,1)} = \frac{8}{h^2}\sin^2(\frac{\pi h}{2}) \text{ and } \lambda_{\max}(A) = \lambda^{(n-1,n-1)} = \frac{8}{h^2}\cos^2(\frac{\pi h}{2}).$$

It is clear that A is symmetric positive definite, so we conclude that the Richardson iteration M converges for $\omega < \frac{h^2}{4\cos^2(\frac{\pi h}{2})}$ whereas the optimal relaxation parameter (2.1) is given by $\omega^{opt} = \frac{h^2}{4}$. In this case, the relaxation converges with

$$\rho(M^{JAC}) = 1 - 2\sin^2(\frac{\pi h}{2}) = \cos \pi h. \tag{2.9}$$

For this model problem, undamped Jacobi iteration $M^{JAC} = I - D^{-1}A$ for A just means Richardson relaxation for the re-scaled matrix $\frac{h^2}{4}A$. Hence, (2.9) also applies here.
The point-wise (lexicographic or checkerboard-wise) Gauss-Seidel method converges with the square rate of the Jacobi iteration, i.e. in this case

$$\rho(M^{GS}) = \rho(M^{JAC})^2 = \cos^2 \pi h = 1 - \sin^2 \pi h.$$

If we employ the SOR method with the optimal damping parameter $\frac{2}{1+\sin(\pi h)}$ (cf. Theorem 2.1), we obtain a convergence factor of

$$\rho(M^{SOR}_{\omega_{opt}}) = \frac{1-\sin(\pi h)}{1+\sin(\pi h)}.$$

For a specific problem discretized using a mesh width h, one is often interested in the convergence order, which is the smallest ξ such that $\rho(M)$ can be expanded as follows, ([Hac91], Remark 3.3.3)

$$\rho(M) = 1 - Ch^{\xi} + O(h^{2\xi})$$

for a constant C independent of h. For the Jacobi, the Gauss-Seidel and the optimally weighted SOR iteration, a Taylor expansion gives us the following convergence orders for the model problem,

$$\begin{aligned} \rho(M^{JAC}) &= 1 - \pi^2/2h^2 + O(h^4) \\ \rho(M^{GS}) &= 1 - \pi^2 h^2 + O(h^4) \\ \rho(M^{SOR}_{\omega_{opt}}) &= 1 - 2\pi h + O(h^2). \end{aligned}$$

Hence, with increasing accuracy (and thus diminishing mesh width h), the convergence factors deteriorate and more iterations are necessary to obtain a given error.
To determine the overall computational costs, we also need to estimate the floating point operations per iteration. In most relevant applications, the matrix A is sparsely populated, i.e. the number of non-zero entries per row A_i of A is small and independent on the size of the matrix,

$$nonzeros(A) < C_A N.$$

For the model problem, we have $C_A = 5$. It is easy to see that the Richardson, Jacobi, Gauss- Seidel and SOR methods require $O(C_A)$ scalar additions and multiplications to update a variable x_i, hence we need $O(N) = O(\frac{1}{h^2})$ operations for a single iteration.
In a nutshell, if we use these methods to solve a linear system with N unknown scalar variables and $O(N)$ non-zero entries in the matrix, the computational work grows superlinearly in terms of N. We cannot expect that a single iteration can be carried out with less than $O(N)$ operations (actually, the computation of the residual $f - Au$ already requires this amount of work), hence, to obtain a solver that only needs $O(N)$ overall computations, we must seek for a method that, in each iteration, reduces the error by a fixed factor $\rho < 1$ that does not depend on the number of unknowns N (or h). This can be accomplished by multigrid methods, which are explained in the next section.

2.2. Geometric Multigrid

Multigrid methods can be seen as an acceleration of simple iterative schemes. To describe this mechanism, we perform an iteration of the ω-damped Jacobi relaxation method to obtain iterate u^{it+1} from iterate u^{it},

$$u^{it+1} = u^{it} + \omega D^{-1}(f - Au^{it}).$$

Consequently, the error $e^{it} = u - u^{it}$ (here u denotes the exact solution) is propagated by

$$e^{it+1} = \left(I - \omega D^{-1} A\right) e^{it}.$$

We return to our Poisson problem example (2.7)–(2.8) from the previous section and decompose the error in terms of the eigenbasis of A. For the eigenvalues

$$\lambda_{ij} = \left(1 - \frac{1}{2}\cos(i\pi h) - \frac{1}{2}\cos(j\pi h)\right) \quad (i, j = 1, \dots n-1)$$

and eigenvectors

$$\phi_{i,j} = \sin(i\pi x)\sin(j\pi y), \qquad (x, y) \in \Omega_h$$

the error propagation reads as follows,

$$\phi^{k+1}_{i,j} = \left(1 - \omega\left(1 - \frac{1}{2}\cos(i\pi h) - \frac{1}{2}\cos(j\pi h)\right)\right)\phi^k$$

i. e. for small values of i and j we have $\phi^{k+1} \approx \phi^k$, while ϕ^k rapidly diminishes for large i or j. Now, we have a decomposition of e in the oscillating part of the error (the span of the eigenvectors ϕ_{ij} for large i and j) and the smooth part of the error (the span of ϕ_{ij} for small i and j). While it is generally not feasible to carry out this decomposition (actually, it would be at least as expensive as solving the system $Ax = b$), we still can employ this knowledge and conclude that the slow-to-converge error can be represented with less degrees of freedom than the dimension of the original problem N. Now the question arises how we can obtain a smaller basis that represents this error well. We study the point-wise error propagation of the Jacobi iteration at the (interior) mesh point i, j,

$$e^{k+1}_{i,j} = e^k_{i,j} - \frac{\omega}{4}\left(4e^k_{i,j} - e^k_{i,j-1} - e^k_{i-1,j} - e^k_{i+1,j} - e^k_{i,j+1}\right)$$

and see that letting $\omega = 1$ the new iterate $e^{k+1}_{i,j}$ is just the average of the old iterate $e^k_{i\pm1,j\pm1}$ at the neighboring grid points. (For other values of ω we obtain an interpolation between the old iterate and this average). Hence, we can conclude that during the Jacobi iteration, the error at each point tends to the average of the error at its neighbors. This motivates us to discretize the smooth error by forming averages of the fine grid error. As outlined above, we only need a quarter of degrees of freedom for the smooth error. Hence, we construct a coarse mesh Ω_{2h} by doubling the mesh size $h \to 2h$ in every spatial direction, see Figure 2.1(b) and obtain a grid with $N_C = \lfloor \frac{n}{2} \rfloor \times \lfloor \frac{n}{2} \rfloor$. To establish a mapping between the coarse and the fine mesh, we need two transfer operators $P : \mathbb{R}^{N_C} \to \mathbb{R}^N$ and $R : \mathbb{R}^N \to \mathbb{R}^{N_C}$. We first construct the *prolongation* or *interpolation* operator P. Its image has to represent the smooth error components well, so it is natural to choose the following interpolation scheme,

$$(Pe)_{k,l} = \begin{cases} e_{i,j} \text{ if } k = 2i \text{ and } l = 2j \\ \frac{e_{i,j}+e_{i+1,j}}{2} \text{ if } k = 2i-1 \text{ and } l = 2j \\ \frac{e_{i,j}+e_{i,j+1}}{2} \text{ if } k = 2i \text{ and } l = 2j-1 \\ \frac{e_{i,j}+e_{i,j+1}+e_{i+1,j}+e_{i+1,j+1}}{4} \text{ if } k = 2i-1 \text{ and } l = 2j-1 \end{cases}$$

i.e. at the coarse grid points we just employ an injection and at all other points we form averages of the errors at nearby coarse grid points, see Figure 2.1(c). For the restriction operator R, we have different options. First, we can only take the value at the coarse grid points, i.e.

$$(Re)_{i,j} = e_{2i,2j}.$$

Another possibility is to form averages during restriction. This can for example be done with the so-called *full weighting* operator,

$$\begin{aligned}(Re)_{i,j} = \frac{1}{16}(4e_{2i,2j} &+ 2(e_{2i,2j-1} + e_{2i-1,2j} + e_{2i+1,2j} + e_{2i,2j+1}) \\ &+ (e_{2i-1,2j-1} + e_{2i-1,2j+1} + e_{2i+1,2j-1} + e_{2i+1,2j+1}))\end{aligned}$$

which is up to a scaling factor the transpose of the prolongation operator. Other restriction operators are also possible.
Finally, we need a coarse linear system. We discretize the PDE on the coarse grid and obtain a coarse system matrix A_C of size $N/4 \times N/4$ (again, all Dirichlet boundary conditions are eliminated and A_C is regular). We can now solve a defect correction equation in the following way: We first restrict the defect to the coarse grid

$$f_C = Rr^{it} = R\left(f - Au^{it}\right).$$

Then, we solve a coarse defect equation,

$$A_C u_c = f_c.$$

Finally, we update the fine grid solution u^{it} using the coarse approximation to the error u_C,

$$u^{it+1} = u^{it} + Pu_C.$$

In terms of the error e^{it}, this correction can be written as

$$e^{it+1} = \left(I - PA_C^{-1}RA\right)e^{it},$$

i.e. we again have an error propagation of the form $(I - BA)$. Note however that in this case $B = PA_C^{-1}R$ is not invertible (though A_C is) and hence the spectral radius $\rho(I - PA_C^{-1}RA)$ cannot be expected to be strictly smaller than one. To obtain a convergent method, the coarse grid correction needs to be combined with one or more smoothing iterations.
In Algorithm 2.1 we give the two-grid cycle. In general, however, the application of a direct solver for the coarse system $A_C u_C = f_C$ is still too expensive. Instead, we compute an approximation to u_C using the two-grid algorithm with $A_C u_C = f_C$ as fine grid system. We recursively extend this procedure until the coarse system is small enough such that it can be efficiently solved using a direct solver or even by few steps of an iterative method. Now, we have constructed a multigrid algorithm 2.2. Note that the recursive application can be carried out once ($\mu = 1$, V-cycle) twice ($\mu = 2$, W-cycle) or combinations hereof depending on the level index l, see Figure 2.2.

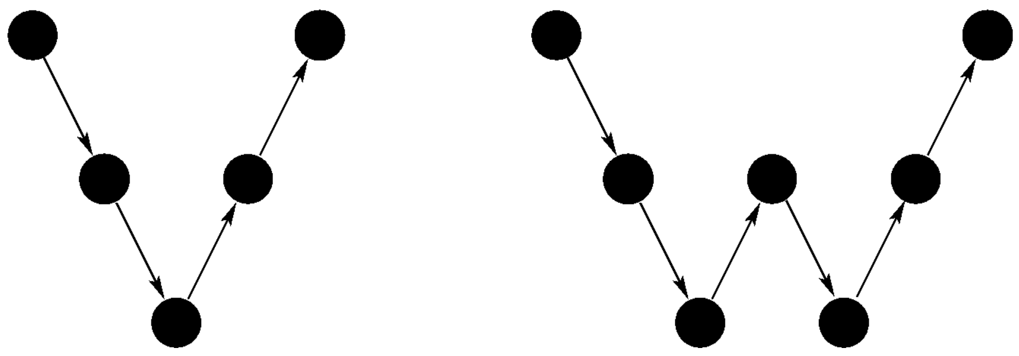


Figure 2.2.: V-cycle (left) and W-cycle (right)

algorithm 2.1 Two-grid cycle $\mathsf{MG}(A, f, u)$

```
begin
  for $\nu \leftarrow 1$ to $\nu_1$ do $u \leftarrow Su$;  od;          pre-smoothing
  $r \leftarrow f - Au$;                                                 residual
  $f_C \leftarrow Rr$;                                                   restriction
  $u_C \leftarrow (A_C)^{-1} f_C$;                                       coarse grid correction
  $u \leftarrow u + Pu_C$;                                               update solution
  for $\nu \leftarrow 1$ to $\nu_2$ do $u \leftarrow Su$;               post-smoothing
  end
```

algorithm 2.2 multigrid algorithm $\mathsf{MG}(A_l, f_l, u_l)$

```
begin
  for $\nu \leftarrow 1$ to $\nu_1$ do $u_l \leftarrow M_l u_l$;  od;   pre-smoothing
  $r_l \leftarrow f_l - A_l u_l$;                                        residual
  $f_{l+1} \leftarrow R_l r_l$;                                          restriction
                                                                         coarse grid correction
  if $l + 1 = l_{max}$
    then
         $u_{l+1} \leftarrow (A_{l+1})^{-1} f_{l+1}$;                    apply directly
     else
         for $\mu \leftarrow 1$ to $\mu$ do
             MG($A_{l+1}, f_{l+1}, u_{l+1}$));                            solve recursively
         od;
  fi;
  $u_l \leftarrow u_l + P_l u_{l+1}$;                                    update solution
  for $\nu \leftarrow 1$ to $\nu_2$ do $u_l \leftarrow M_l u_l$;  od;   post-smoothing
end
```

For sake of completeness, we now outline the classical multigrid convergence proof scheme as introduced by Wolfgang Hackbusch in the Chapters 6 and 7 of [Hac85]. First, we show the convergence of the two-grid cycle 2.1 and then we interpret the full multigrid cycle 2.2 as a disturbed two-grid cycle.
The two-grid cycle (on finer level l and coarser level $l+1$) with only pre-smoothing can be written in operator form,

$$\begin{aligned} e^{it+1} = T_l^{TGM}(\nu_1)e^{it} &= \left(I - P_l(A_{l+1})^{-1}R_lA_l\right)(M_l)^{\nu_1}e^{it} \\ &= \left((A_l)^{-1} - P_lA_{l+1}^{-1}R_l\right)(A_lM_l^{\nu_1})e^{it}, \end{aligned}$$

where M_l denotes the smoothing operator on level l. In the last step we have factored the iteration into two steps, a smoothing step and a coarse grid correction step. From the convergence properties of these two steps we derive the convergence of the two-grid method. To this end, let $A_l : U \to F$ and let the norms $\|\cdot\|_U$ and $\|\cdot\|_F$ as well as the induced operator norms $\|\cdot\|_{U\to F}$ and $\|\cdot\|_{F\to U}$ be available.

Definition 2.1 *([Hac85], Definition 6.1.3) Let $\|\cdot\|_U$ and$\|\cdot\|_F$ be given. M_l^ν is said to posses the* smoothing property *if there exists functions $\eta(\nu)$ and $\bar\nu(h)$ and a number κ such that*

$$\|A_lM_l^\nu\|_{U\to F} \le \eta(\nu)h_l^{-\kappa} \textit{ for all } 1 \le \nu < \bar\nu(h_l),\ l < l_{max}, \tag{2.10}$$

$$\lim_{\nu\to\infty} \eta(\nu) = 0, \tag{2.11}$$

$$\bar\nu(h) = \infty \textit{ or } \lim_{h\to 0} \bar\nu(h) = \infty. \tag{2.12}$$

The functions η and $\bar\nu$ are independent of l or h_l. The condition

$$\|A_lM_l^\nu\|_{U\to F} \le \eta(\nu)h_l^{-\kappa} \textit{ for all } \nu \ge 1, l < l_{max},$$

is described formally by $\bar\nu(h) = \infty$.

Note that an iteration that possesses the smoothing property is not necessarily convergent.

Definition 2.2 *([Hac85], Definition 6.1.6) The* approximation property *holds if there is some constant C such that*

$$\|A_l^{-1} - P_lA_{l+1}^{-1}R_l\|_{F\to U} \le Ch_l^\kappa \textit{ for all } l < l_{\max} \tag{2.13}$$

with κ from (2.10).

If our smoother and coarse-grid correction satisfy these properties, we obtain two-grid-convergence.

Theorem 2.2 *([Hac85], Theorem 6.1.7) Suppose the smoothing property 2.1 and the approximation property 2.2. Let $\rho > 0$ be a fixed number.*

1. *In the case of $\bar{\nu}(h) = \infty$ there is a number $\underline{\nu}$ such that the two-grid contraction number satisfies*
$$\|T_l^{TGM}(\nu)\|_{U\to U} \leq C\eta(\nu) \leq \rho \tag{2.14}$$
whenever $\nu \geq \underline{\nu}$ and $l < l_{max}$.

2. *In the remaining case of $\bar{\nu}(h) \to \infty$ there are $\bar{h} > 0$ and $\underline{\nu}$ such that inequality* (2.14) *holds for all $\underline{\nu} \leq \nu < \bar{\nu}(h_l)$ and all $h_l \leq \bar{h}$. For such l the interval $[\underline{\nu}, \bar{\nu}(h_l))$ is not empty.*

3. *Under condition 2 and for $\rho < 1$ one obtains convergence:*
$$u_l^{it} \to u_l = A_l^{-1} f_l.$$

Note that the right hand side of (2.14) is independent of l.
In the case or our model problem, we introduce the grid-depending scalar product

$$< u, v >= \sum_{x\in\Omega_h} h^d u(x)v(x)$$

where d is the spatial dimension of the domain. We need three vector norms

$$\begin{aligned}
|u|_0 &= \sqrt{< u, u >},\\
|u|_2 &= \left(\sum_{|\alpha|\leq 2} |\delta^\alpha \tilde{u}|_0^2\right)^{\frac{1}{2}} \quad \text{where } \tilde{u} = \begin{cases} u \text{ in } \Omega_h \\ 0 \text{ in } Q_h \setminus \Omega_h \end{cases}\\
|u|_{-2} &= \sup\{< u, v >: \ |v|_2 = 1\}
\end{aligned}$$

and the induced operator norms $|\cdot|_0$, $|\cdot|_{0\to 2}$, $|\cdot|_{2\to 0}$, $|\cdot|_{0\to -2}$, $|\cdot|_{-2\to 0}$, $|\cdot|_{-2\to -2}$. Here, Q_h denotes the infinite continuation of Ω_h and δ^α is the discrete difference of order α, where α is a multi-index,

$$\begin{aligned}
\delta_\alpha &:= \delta_1^{\alpha_1} \dots \delta_d^{\alpha_d},\\
\delta_j u(x_1, \dots, x_j, \dots, x_d) &:= \frac{u(x_1, \dots, x_j, \dots, x_d) - u(x_1, \dots, x_j - h, \dots, x_d)}{h}
\end{aligned}$$

The smoothing property is satisfied with the following functions $\eta(\nu)$.

- For the Richardson iteration, we have ([Hac85], Proposition 6.2.33)
$$\eta(\nu) = \frac{2C_A}{1+\sqrt{2}} \frac{1}{(\nu+1)^2},$$
where C_A satisfies $|h_l^2 A_l|_0 \leq C_A$.

- For the ω_l- damped Jacobi method, we have ([Hac85], Proposition 6.2.11)
$$\eta(\nu) = \frac{C_0 \nu^\nu}{(\nu+1)^{\nu+1}}$$
where C_0 is chosen such that $\omega_l \geq \frac{1}{C_0}$.

- For the lexicographical Gauss-Seidel relaxation one can show ([Hac85], Proposition 6.2.25)

$$\eta(\nu) = 4 \begin{cases} \frac{2}{3^\nu} & \text{if } \nu = 1, 2; \\ \frac{1}{2}\sqrt{\frac{(\nu-2)^{\nu-2}}{\nu^\nu}} & \text{else.} \end{cases}$$

To show the approximation property for this family of discretization, let $\delta_{l+1} = R_l A_l P_l - A_{l+1}$ be the difference between the Galerkin projection and the coarse grid operator. Furthermore, let $R^{'}$ denote the trivial injection from level l to level $l+1$.
One can show the following regularity statements ([Hac85], Section 6.3.2.2),

$$\begin{aligned} |\delta_{l+1}|_{-2\to 2} &\leq C_\delta h^2 \\ |A_l^{-1}|_{-2\to 0} &\leq C_R,\ |A_l^{-1}|_{0\to 2} \leq C_R \\ |A_l|_{-2\to 0} &\leq C_L,\ |A_l|_{0\to 2} \leq C_L \\ |I - P_l R^{'}|_{2\to 0} &\leq C_I h^2 \\ |R_l|_{-2\to -2} &\leq C_R,\ |R^{'}|_{2\to 2} \leq C_R^{'},\ |P_l|_{0\to 0} \leq C_P. \end{aligned}$$

We now split

$$\begin{aligned} A_l^{-1} - P_l A_{l+1}^{-1} R_l &= \left(I - P_l A_{l+1}^{-1} R_l A_l\right) A_l^{-1} \\ &= \left(I - P_l A_{l+1}^{-1} R_l A_l\right)\left(I - P_l R^{'}\right) A_l^{-1} - P_l A_{l+1}^{-1} \delta_{l+1} R^{'} A_l^{-1} \end{aligned}$$

We estimate the three components in the last line,

$$\begin{aligned} |I - P_l A_{l+1}^{-1} R_l A_l|_0 &\leq 1 + |P_l|_0 |A_{l+1}^{-1} R_l A_l|_0 \leq C_p C_R C_r C_A \\ |\left(I - P_l R^{'}\right) A_l^{-1}|_0 &\leq C_I h_l^2 C_R \\ |P_l|_0 |A_{l+1}^{-1} \delta_{l+1} R^{'} A_l^{-1}|_0 &\leq C_P C_R^2 C_\delta C_R^{'} \end{aligned}$$

and obtain the approximation property (2.14) with

$$C = (1 + C_p C_R C_r C_A) C_I C_R + C_P C_R^2 C_\delta C_R^{'}.$$

It remains to show the convergence of the whole multigrid cycle $T_l(\nu_1, \nu_2)$. This can recursively be written as

$$T_{l_{\max}-1}(\nu_1, \nu_2) = T_{l_{\max}-1}^{TGM}(\nu_1, \nu_2) \tag{2.15}$$

$$T_l(\nu_1, \nu_2) = T_l^{TGM}(\nu_1, \nu_2) + M_l^{\nu_2} P_l \left(M_{l+1}(\nu_1, \nu_2)\right)^\gamma A_{l+1}^{-1} R_l A_l M_l^{\nu_1}. \tag{2.16}$$

Hence, the multigrid iteration can be interpreted as a two-grid iteration plus a perturbation. To estimate this perturbation, we need the following additional assumptions on the smoother and the interpolation operators (which can be easily shown for our model problem),

$$\|S_l^\nu\|_{U\to U} \leq C_S \qquad \text{for all } l < l_{\max},\ 0 < \nu\overline{\nu}(h_1) \tag{2.17}$$

$$\underline{C}_P^{-1} \|u_{l+1}\|_{U\to U} \leq \|P u_{l+1}\|_{U\to U} \leq \overline{C}_P \|u_{l+1}\|_{U\to U}. \tag{2.18}$$

We consider the case of pre-smoothing only, i.e. $\nu_2 = 0$. By induction we already have $\|M_{l+1}(\nu_1, 0)\|_{U\to U}\zeta_{l+1}^{\gamma}$ for some $\zeta_{l+1} < 1$. By (2.18) we have

$$\begin{aligned}
\|A_{l+1}^{-1}R_l A_l M_l^{\nu_1}\|_{U\to U} &\leq \underline{C}_P\|P_l A_{l+1}^{-1}R_l A_l M_l^{\nu_1}\|_{U\to U} \\
&= \underline{C}_P\|M_l^{\nu_1} - \left(A_l^{-1} - P_l A_{l+1}^{-1}R_l\right) A_l M_l^{\nu_1}\|_{U\to U} \\
&= \underline{C}_P\|M_l^{\nu_1} - T_l^{TGM}(\nu_1, 0)\|_{U\to U}.
\end{aligned}$$

We can now estimate (2.16) by

$$\|T_l(\nu_1, 0)\|_{U\to U} \leq \|T_l^{TGM}\|_{U\to U} + C^*\|M_{l+1}(\nu_1, 0)\|_{U\to U}^{\gamma}.$$

where $C^* = \overline{C}_P\underline{C}_P(C_S + 1)$. Let now the norm of the two-grid method be bounded by $\|T_l^{TGM}\|_{U\to U} \leq \zeta$, we obtain the recursion ($l < l_{\max} - 1$)

$$\zeta_l \leq \zeta + C^*\zeta_{l+1}^{\gamma} \tag{2.19}$$

and $\zeta_{l_{\max}-1} \leq \zeta$.

Lemma 2.1 *([Hac85], Lemma 7.1.6) Assume $C^*\gamma > 1$. If*

$$\gamma \geq 2,\ \zeta \leq \zeta_{\max} = \frac{\gamma - 1}{\gamma}\left(\gamma C^*\right)^{-\frac{1}{\gamma-1}}, \tag{2.20}$$

any solution of (2.19) *is bounded by*

$$\zeta_l \leq \zeta^* < 1.$$

where ζ^* and ζ are related by

$$\zeta = \zeta^* - C^*(\zeta^*)^{\gamma},\ \zeta^* \leq \frac{\gamma}{\gamma - 1}\zeta.$$

In the case of the W-cycle ($\gamma = 2$) the requirement (2.20) *becomes*

$$\gamma = 2,\ \zeta \leq \zeta_{\max} = \frac{1}{4C^*}$$

and all ζ_l are bounded by

$$\zeta_l \leq \frac{2\zeta}{1 + \sqrt{1 - 4\zeta C^*}}.$$

The second part of (2.20) is not a very strong requirement, as we can always increase $\underline{C}_P$ such that $C^* < \frac{1}{\gamma}$. Now, we obtain the multigrid convergence theorem for the W-cycle.

Theorem 2.3 *([Hac85], Theorem 7.1.2) Suppose $\gamma \geq 2$,* (2.17), (2.18), *the smoothing property from Definition 2.1 and the approximation property from 2.2. Let $\zeta' \in (0, 1)$ be a fixed number.*

1. *In the case of $\bar{\nu}(h) = \infty$, there is a number $\underline{\nu}$ such that the multi-grid contraction number satisfies*

$$\|T_l(\nu_1, 0)\|_{U \to U} \leq \zeta' < 1, \|T_l(\nu_1, 0)\|_{U \to U} \leq \frac{\gamma}{\gamma - 1} C\eta(\nu), \tag{2.21}$$

   whenever $\nu \geq \underline{\nu}$, independently of $l < l_{\max}$.

2. *In the remaining case of $\lim_{h \to 0} \bar{\nu}(h) = \infty$ there exist $\bar{h} > 0$ and $\underline{\nu}$ such that Inequality (2.21) holds for all $\nu \in [\underline{\nu}, \bar{\nu}(h_{l_{\max}-1}))$, provided that $h_{\max-1} \leq \bar{h}$. For such $h_{\max-1}$ the interval $[\underline{\nu}, \bar{\nu}(h_{l_{\max}-1}))$ is not empty.*

One can show a dual result for the case of post-smoothing only ([Hac85], Theorem 7.1.7).
To show the convergence of the V-cycle, i.e. $\gamma = 1$, in a general setting we refer to [BD85].
In this section, we have seen how a geometric multigrid method for a simple model problem can be constructed. In addition, we have given an outline of the convergence proof. The coarsening process, the construction of the interpolation operator and the choice of the coarse grid operator however all required the knowledge of the underlying PDE as well as the discretization grid. Now the question arises whether we can *automate* this process and employ the fine level matrix A_1 only to obtain a complete multigrid hierarchy, i.e. a family of grids Ω_l, system matrices A_l, interpolation and restriction matrices P_l, R_l and smoothing operators M_l. The remainder of this chapter will deal with these issues.

2.3. AMG setup

algorithm 2.3 AmgSetup($\Omega, A, int, N_{min} L_{max}, L, \{A_l\}_{l=1}^{L}, \{P_l\}_{l=1}^{L-1} \{R_l\}_{l=1}^{L-1}$)

```
begin
  $\Omega^1 \leftarrow \Omega$;   $N_1 = |\Omega^1|$
  $A_1 \leftarrow A$;
  for $l \leftarrow 1$ to $L_{max} - 1$ do
      split $\Omega^l$ into $C^l \dot{\cup} F^l$;
      set $\Omega^{l+1} \leftarrow C^l$;   $N_{l+1} = |\Omega^{l+1}|$;
      build $P_l : \mathbb{R}^{N_{l+1}} \to \mathbb{R}^{N_l}$;
      set $R_l = \left(P^l\right)^T$;
      compute $A_{l+1} \leftarrow R_l A_l P_l$;
      if $|\Omega^{l+1}| \leq N_{min}$ then break;  fi;
  od;
  $L \leftarrow l + 1$;
end.
```

For any algebraic multigrid method, we first have to create the multigrid hierarchy before we can start the solving iteration. The latter part, called the *solve (or solution) phase* is identical to the geometrical multigrid cycle (see Algorithm 2.2). During the first part, called the *setup phase*, we create the sequence of grids $\{\Omega^l\}_{l=1}^{L}$, the inter-grid transfer operators $\{P_l\}_{l=1}^{L-1}$ and $\{R_l\}_{l=1}^{L-1}$ as well as the operators $\{A_l\}_{l=1}^{L}$. A brief sketch of the AMG setup process is outlined in Algorithm 2.3. In the following sections, we describe in detail each component of the setup process.

2.4. Smoothing

Before we can construct an adequate algebraic multigrid hierarchy for a given system matrix $A \in \mathbb{R}^{N\times N}$, we first have to understand the reduction of the error during the smoothing iterations. To this end, we need to establish a criterion that allows us to separate the smooth (or slow-to-converge) error components from the oscillating components. In the case of a symmetric positive definite operator A, where $D \in \mathbb{R}^{N\times N}$ denotes the diagonal part of A, we can define three inner products that help us to classify the error,

Definition 2.3 *([Stü99], Section 2.2)*

$$(u,v)_0 := (Du,v) \tag{2.22}$$
$$(u,v)_1 := (Au,v) \tag{2.23}$$
$$(u,v)_2 := (D^{-1}Au,Av) \tag{2.24}$$

and the respective discrete norms $\|u\|_0$, $\|u\|_1$ and $\|u\|_2$ for all $u,v \in \mathbb{R}^N$.

The equivalence factors between these norms can be estimated as follows,

Lemma 2.2 *([Stü99], Lemma 3.1) Let A be symmetric and positive definite, let D be the diagonal part of A. Then, for all $e \in \mathbb{R}^N$:*

$$\|e\|_1^2 \le \|e\|_0\|e\|_2, \quad \|e\|_2^2 \le \rho(D^{-1}A)\|e\|_1^2, \quad \|e\|_1^2 \le \rho(D^{-1}A)\|e\|_0^2. \tag{2.25}$$

The eigenvectors ϕ_i of $D^{-1}A$ and the corresponding eigenvalues λ_i satisfy

$$\|\phi_i\|_2^2 = \lambda_i\|\phi_i\|_1^2, \text{ and } \|\phi_i\|_1^2 = \lambda_i\|\phi_i\|_0^2. \tag{2.26}$$

We now formulate the smoothing property of a smoothing operator M in terms of these norms.

Definition 2.4 *([Stü99], Section 3.2) An operator M satisfies the* smoothing property *with respect to A, if we have a $\sigma > 0$ such that for all $e \in \mathbb{R}^N$ holds,*

$$\|Me\|_1^2 \le \|e\|_1^2 - \sigma\|e\|_2^2 \tag{2.27}$$

From this definition we see that error components e that satisfy $\|e\|_2 \ll \|e\|_1$ will not be reduced efficiently by M. In terms of the eigenvalues and eigenvectors of $D^{-1}A$, $\|\phi_i\|_2 \ll \|\phi_i\|_1$ implies $\lambda_i \approx 0$ by (2.26).
Before we further analyze the smooth components, we first note that damped Jacobi relaxation and Gauss-Seidel relaxation satisfy property (2.27).

Theorem 2.4 *([Stü99], Theorem 3.1)*
Let A be symmetric, positive definite and let $w \in \mathbb{R}^N$ be an arbitrary vector, $w_i > 0$ for all i. We define γ_- and γ_+,

$$\gamma_- := \max_i \{\frac{1}{w_i a_{ii}} \sum_{j<i} w_j |a_{ij}|\} \text{ and } \gamma_+ := \max_i \{\frac{1}{w_i a_{ii}} \sum_{j>i} w_j |a_{ij}|\}.$$

Then, the Gauss-Seidel iteration $M = I - (L+D)^{-1}A$ or $S = I - (U+D)^{-1}A$, where L (U) denotes the lower (upper) triangular part of A, satisfies the smoothing property (2.27) with $\sigma = \frac{1}{(1+\gamma_-)(1+\gamma_+)}$.

Theorem 2.5 *([Stü99], Theorem 3.2)*
Let A be symmetric positive definite and $\eta \geq \rho(D^{-1}A)$. Then, the damped Jacobi relaxation $M = I - \omega D^{-1}A$ satisfies the smoothing property (2.27) with $\sigma = \omega(2 - \omega\eta)$ for any $0 < \omega < \frac{2}{\eta}$. The optimal relaxation parameter is given by $\omega^* = \frac{1}{\eta}$, in this case also $\sigma = \frac{1}{\eta}$.

For two important classes of matrices we now give a more detailed characterization of the smooth error components.

2.4.1. M-Matrices

Definition 2.5 *A matrix $A \in R^{n\times n}$ is called* M matrix *if it satisfies the following properties,*

- *$a_{ii} > 0$ for all $i = 1, \dots, n$,*
- *$a_{ij} \leq 0$ for all $i, j = 1, \dots n,\ i \neq j$,*
- *A is non-singular,*
- *the entries of A^{-1} are non-negative.*

The *smooth error components* satisfy $\|e\|_2 \ll \|e\|_1$. Together with $\|e\|_1^2 \leq \|e\|_0 \|e\|_2$ (2.25) we get

$$\|e\|_1 \ll \|e\|_0 \text{ for } e \neq 0.$$

which can be expanded as follows

$$\begin{aligned} \|e\|_1^2 &= \sum_{i,j} a_{ij} e_i e_j = \frac{1}{2} \sum_{i,j} -a_{ij}(e_i - e_j)^2 + \sum_i s_i e_i^2, \\ \text{where } s_i &:= \sum_j a_{ij}. \end{aligned}$$

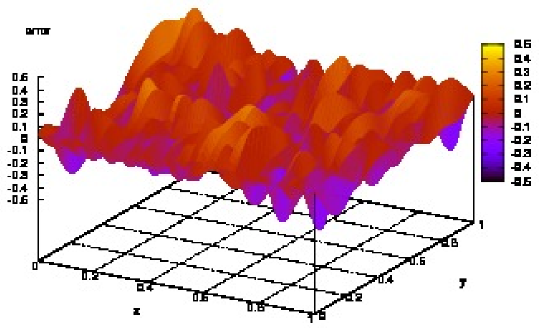


Figure 2.3.: Error after 10 Gauss-Seidel iterations for a finite difference discretization of $-u_{xx} - 0.001 \cdot u_{yy}$, randomly chosen starting vector.

Hence, we have on average for each $i = 1, \ldots, N$,

$$\sum_{j \neq i} - \frac{a_{ij}(e_i - e_j)^2}{a_{ii} e_i^2} \quad \ll \quad 1.$$

As all a_{ij}, $i \neq j$ have the same sign, we can conclude that an algebraically smooth error only varies slowly along the large couplings $-a_{ij} \gg 0$.

Example 2.1 We consider the anisotropic problem

$$\begin{aligned} -\epsilon \mathrm{u}_{xx} - \mathrm{u}_{yy} &= \mathrm{f} \text{ in } \Omega = (0,1)^2 \\ \mathrm{u} &= \mathrm{g} \text{ on } \partial\Omega \end{aligned}$$

for $\epsilon \ll 1$. From Figure 2.3 we see that after a few smoothing steps the error in y-direction only varies slowly between neighboring points, while we have huge oscillations in x-direction.

2.4.2. Essentially positive definite operators

In many applications, for example when discretizing mixed derivatives, we cannot expect that all off-diagonal entries of the matrix A are non-positive. If however the positive entries are relatively small, the results from the M-matrix case still can be applied.

Definition 2.6 *([Stü99], Section 3.3.2) A matrix $A \in \mathbb{R}^{N \times N}$ is called an* essentially positive matrix, *if there exists a constant $c > 0$, such that for all $e \in \mathbb{R}^N$*

$$\sum_{i,j} -a_{ij}(e_i - e_j)^2 \geq c \sum_{i,j} -a_{ij}^-(e_i - e_j)^2. \tag{2.28}$$

where

$$a_{ij}^- = \begin{cases} a_{ij} \text{ if } a_{ij} < 0 \\ 0 \text{ otherwise.} \end{cases}$$

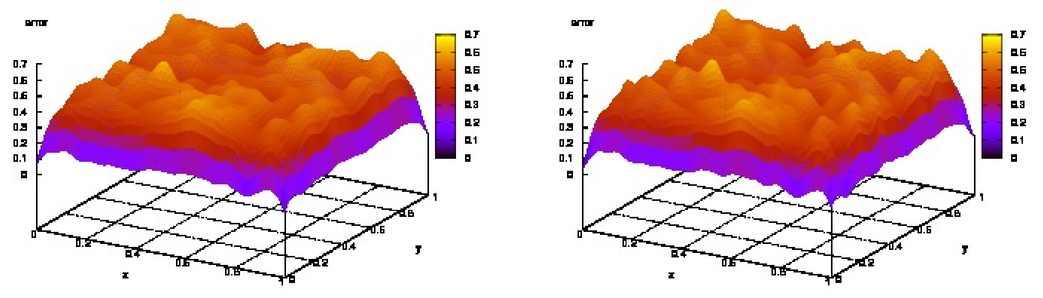


Figure 2.4.: Error after 5 Gauss-Seidel iterations on randomly chosen start vector for the Laplace operator (left) and the problem with mixed derivatives (right).

Remark 2.1 For an essentially positive matrix, each row containing off-diagonal non-zero entries has at least one negative off-diagonal entry.

Using the notation of the previous subsection, we get the following estimate for algebraically smooth error,

$$\frac{c}{2}\sum_{i,j} -a_{ij}^{-}(e_i - e_j)^2 + \sum_i s_i e_i^2 \ll \sum_i a_{ii} e_i^2$$

which again means this error component varies slowly in the direction of strong negative couplings $a_{ij}^{-} \ll 0$ and that the positive connections do not play an important role. This is also motivated by the following example.

Example 2.2 [Stü99] We consider a problem with mixed derivatives, $-\Delta u + u_{xy}$ discretized using a 9-point finite difference scheme on an uniform grid of mesh size h. We employ the stencil

$$\frac{1}{h^2}\begin{bmatrix} -0.25 & -1 & +0.25 \\ -1 & 4 & -1 \\ +0.25 & -1 & -0.25 \end{bmatrix}_h .$$

The resulting matrix fulfills condition (2.28) with $c = 0.5$. As can be seen from figure 2.4, the positive couplings resulting from the mixed derivatives do not have an influence on the error.

Remark 2.2 In the case of large positive connections (i.e. the matrix A does not fulfill property (2.28)), algebraically smooth errors can oscillate in the direction of these couplings. For example, when considering nearly weakly diagonally dominant matrices, we get the following estimate for algebraically smooth error ([Stü99], Section 3.3.3)

$$\sum_{j\neq i} -\frac{a_{ij}^{-}(e_i - e_j)^2}{a_{ii}e_i^2} + \sum_{j\neq i}\frac{a_{ij}^{+}(e_i + e_j)^2}{a_{ii}e_i^2} \ll 1,$$

which implies that $e_i \approx -e_j$ if a_{ij}^{+}/a_{ii} is relatively large.

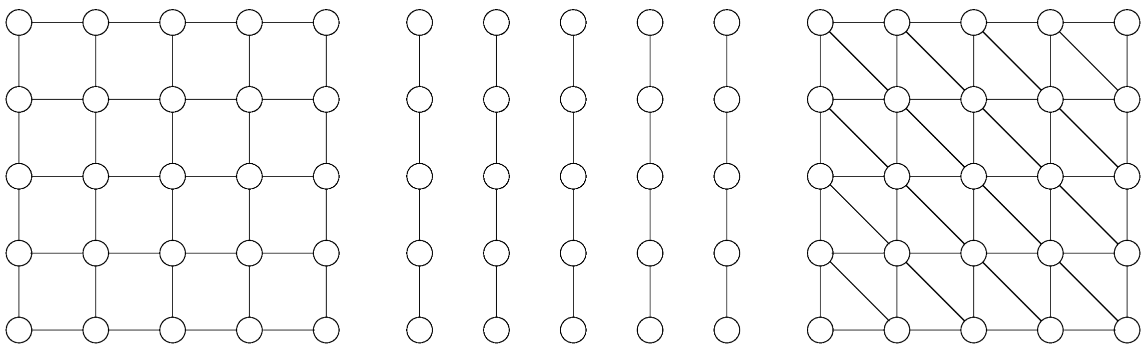


Figure 2.5.: Strong couplings for finite difference discretizations. The left picture shows the strong connections for an isotropic operator A. In case of the second image, we have strong connections only in y direction (see Example 2.1). The figure on the right hand side shows the strong connections for Example 2.2 if a threshold $\alpha \leq 0.25$ is used.

2.5. Strong Couplings

algorithm 2.4 AmgStrongCouplings(A, S, S^T)

```
begin
  for $i \leftarrow 1$ to $N$ do
    for $j \in N_i$ do
      if $-a_{ij} \geq \alpha \cdot \max_{k \neq i} -a_{ik}$
        then
            $S_i \leftarrow S_i \cup \{j\}$;
            $S_j^T \leftarrow S_j^T \cup \{i\}$;
      fi;
    od;
  od;
end
```

In the previous section, we have seen that for symmetric positive M-matrices and essentially positive matrices, the algebraically smooth error e varies slowly in the direction of the large negative off-diagonal matrix entries a_{ij}, i.e. $e_i \approx e_j$. We will use this observation for the construction of the interpolation (prolongation) operator and, in consequence, for the coarsening process. Therefore we introduce the concept of the *strong coupling*,

Definition 2.7 *([Stü99], Section 7.1) Let* $0 < \alpha < 1$ *be fixed.*

1. *A point* j *is* strongly coupled *to another point* $i \neq j$*, if*

$$-a_{ij} \geq \alpha \cdot \max_{k \neq i}\{-a_{ik}\}. \tag{2.29}$$

2. *We introduce the following notation*

$$\begin{aligned} S_i &:= \{j \in \Omega : \ -a_{ij} \geq \alpha \cdot \max_{k \neq i}\{-a_{ik}\}\} \textit{ the strong couplings } i, && (2.30)\\ S_i^T &:= \{j \in \Omega : i \in S_j\} \textit{ the set of all points strongly coupled to } i. && (2.31)\end{aligned}$$

To be more demonstrative, we can consider S_i as the dependencies of i, i.e. the smooth error e_i depends in particular on the error e_j at the strongly connected neighbors $j \in S_i$. On the other hand, S_i^T can be seen as the influence of i. The union of all S_i can be interpreted as a directed, unweighted graph where the vertices are given by the grid points $i \in \Omega$ and an edge is established between two points i and j if $j \in S_i$. We denote this graph and its connectivity matrix with S. S^T is defined similarly. Note that even for symmetric operators A, the connectivity matrix S is not necessarily symmetric. Consider for example the following very simple example ($\alpha = 0.25$),

$$A = \frac{1}{h^2}\begin{pmatrix} 2.1 & -1 & 0\\ -1 & 1.1 & -0.1\\ 0 & -0.1 & 0.2 \end{pmatrix}, \ S = \begin{pmatrix} 0 & 1 & 0\\ 1 & 0 & 0\\ 0 & 1 & 0 \end{pmatrix}.$$

In Algorithm 2.4 we give the algorithm for computing the strong connections. An example for two strength graphs is given in Figure 2.5.

Remark 2.3 A slightly different way to define the strong couplings is to use the inequality

$$|a_{ij}| \geq \alpha \cdot \max_{k \neq i} |a_{ik}| \tag{2.32}$$

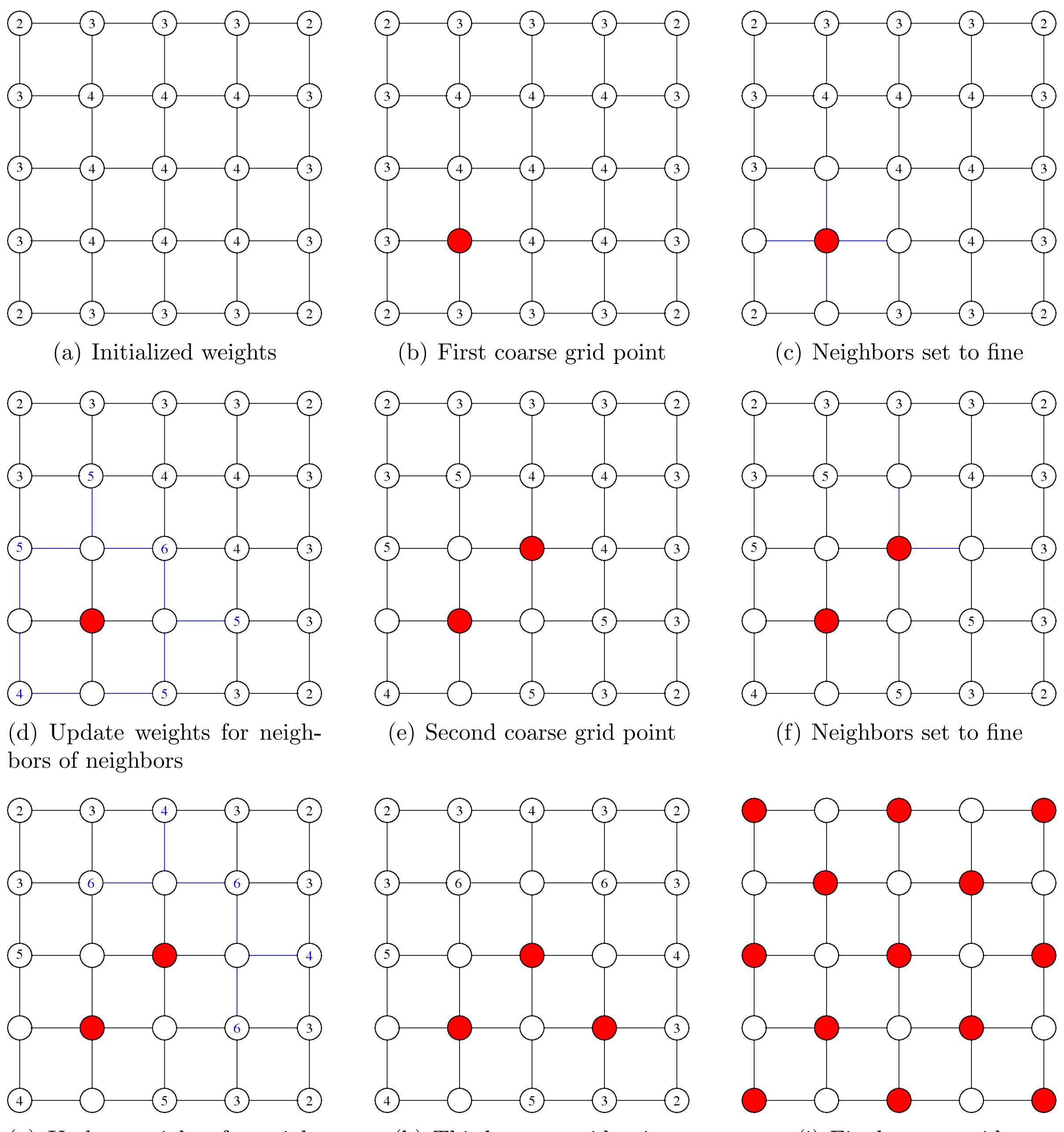
instead of (2.29). In this case all couplings beyond a certain threshold are considered coarse, even the positive ones. This may especially be useful in the case of essentially positive matrices, where for every large positive connection $i \leftarrow j$ there is also a path of strong negative connections $i \leftarrow k \leftarrow j$. So instead of the more expensive detection of these paths, we simply take the large positive connection. It is also possible to use different thresholds α^- and α^+ for the positive and negative connections.

2.6. Coarsening according to Ruge and Stüben

As in the geometric case, the coarse grid $\Omega^{l+1} = C^l$ should meet two demands:

1. The smooth error components e should be well represented on Ω^{l+1}.
2. The coarse grid should be a constant (independent of $|\Omega^l|$) factor smaller than the fine grid.

Now, algebraic multigrid is often used if geometric multigrid fails, i.e. a geometrically obtained coarsening like mesh width doubling does not reflect the behavior of smooth error components very well. This can for example be due to anisotropies in the operator



(a) Initialized weights

(b) First coarse grid point

(c) Neighbors set to fine

(d) Update weights for neighbors of neighbors

(e) Second coarse grid point

(f) Neighbors set to fine

(g) Update weights for neighbors of neighbors

(h) Third coarse grid point

(i) Final coarse grid

Figure 2.6.: Coarsening process for a 5-point stencil. The numbers denote the weights of still undecided points, the red points belong to the coarse grid and the white points belong to the fine grid. Updated weights are given in blue. The blue edges indicate which connections are considered within the graph at the current step. Note that the strength matrix is symmetric here.

(see e.g. Example 2.1) or different PDE coefficients in different parts of the domain. Furthermore, the geometry of the discretization domain may not allow a straightforward coarsening.
Instead of relying on geometrical information, the key to the construction of an algebraic coarse grid is to use the information that we know about the structure of the smooth error. In the previous section, we have already seen that for relevant PDE problems the smooth error varies slowly along the strong connections. In other words, the value e_i at a point $i \in \Omega$ can easily be approximated from the values of strongly connected points $j \in S_i$. The coarsening demands mentioned above can now be expressed in terms of S.

C1 For any fine grid point $i \in F$ and every strongly connected point $j \in S_i$ let either j be a coarse grid point, or j itself depend on a coarse grid point $k \in S_i \cap C$.

C2 Let C be a maximal set so that two points in C are not strongly connected to each other.

The first condition ensures that for any fine grid point $i \in F$ all strong connections are reflected in the coarsening process, not only those that directly lead to a coarse grid point $j \in C$. The reason for this is the that we will construct the interpolation operators (see Section 2.8) to interpolate along the strong connections, i.e. the value at a fine grid point $i \in F$ will be interpolated from the values e_j, where j is contained in the set of strongly connected coarse neighbors $S_i \cap C$. On the other hand, the value e_k at any strongly connected coarse grid point $k \in S_i$ should be represented in the interpolation scheme. Now, as long as $e_k, k \in S_i \cap F$, also depends on and interpolates from $e_j, j \in S_i \cap C$, e_k cannot differ "too much" from these e_j and hence from e_i.
If C fulfills Condition C2, it is called a *maximal independent set (MIS)*. An independent set $I \subset V$ in a graph (V, E) is a set of vertices such that two vertices $i, j \in I$ are not directly connected to each other, i.e. there is no edge (i, j) in E. A maximal independent set is an independent set I such that no additional vertex i can be added to I without loosing the independent set property. This should not be confused with a *maximum independent set*, which is an independent set with a maximum number of vertices. The task of constructing a coarse grid hence turns into the task of finding a maximal independent set, for which many greedy algorithms are available, e.g. [Lub86]. In this particular case, however, the set of coarse grid points has to satisfy condition C1 to ensure the robustness of the interpolation. Therefore, the construction of coarse grids is split into two phases. In the first phase, an independent set is created. The second phase then enforces condition C1, thereby sacrificing the independent nature of the set C.

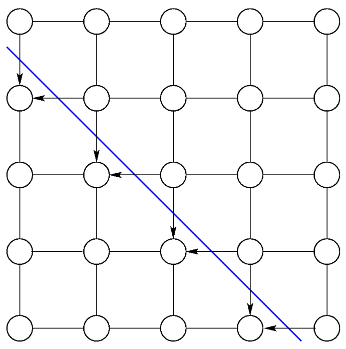
In Algorithm 2.5 we give the first phase of the classical coarsening algorithm. Most important here is that we assign a weight λ_i to each point i, which measures the "usefulness" of i as coarse grid point. Before starting the iteration, we initialize λ_i to the number of points j that depend on i, i.e. $|S_i^T|$ (Figure 2.6(a)). A point with maximal weight is chosen and added to the set of coarse grid points C (Figure 2.6(b)). Then, all neighbors j that depend on i ($j \in S_i^T$) can interpolate from i and hence we assign them to the set of fine grid points (Figure 2.6(c)). On the other hand, interpolation should

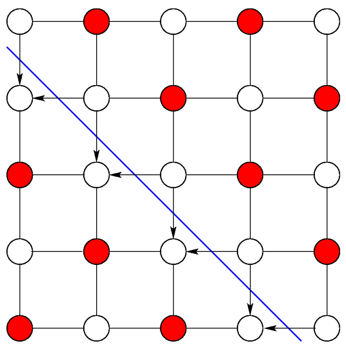
algorithm 2.5 AmgPhaseI(Ω, S, S^T, C, F) ([RS87], algorithm A2)

```
begin
  $U \leftarrow \Omega$;
  $C \leftarrow \emptyset$;
  $F \leftarrow \emptyset$;
  for $i \in \Omega$ do $\lambda_i \leftarrow |S_i^T|$; od;
  while $\max_{i \in U} \lambda_i \neq 0$ do
      $i \leftarrow \arg\max_{j \in U} \lambda_j$;                          pick coarse grid point
      $C \leftarrow C \cup \{i\}$; $U \leftarrow U \setminus \{i\}$;
      for $j \in S_i^T \cap U$ do
          $F \leftarrow F \cup \{j\}$; $U \leftarrow U \setminus \{j\}$;            add neighbors to fine grid
          for $k \in S_j \cap U$ do
              $\lambda_k \leftarrow \lambda_k + 1$;          increase weights of neighbors of neighbors
          od;
      od;
      for $j \in S_i \cap U$ do
          $\lambda_j \leftarrow \lambda_j - 1$;             decrease weights of point that influence i
      od;
  od;
  $F \leftarrow F \cup U$;
end
```

be from a large portion of connection of j, not only one strong coupling (see Section 2.8 for an intensive discussion). This motivates us to increase, for all newly assigned fine grid points j, the weight of all $k \in S_j$ such that they are more likely to become coarse in future iterations (Figure 2.6(d)).
Furthermore, as i is now coarse, the strong connections of i, S_i are not needed for interpolation anymore. Hence, we can reduce the weights λ_j for all $j \in S_i$ if j is still undecided. Then, we restart the iteration and again pick a point with maximal weight λ_i (Figure 2.6(e)) until all points are assigned either fine or coarse (Figure 2.6(i)). It is easy to see that the iteration needs at most $O(N \cdot nzr^2)$ steps, where N is the number of unknowns and nzr the maximal number of nonzeros per row, which is assumed to be small and independent of N for relevant PDE problems.

The second phase algorithm 2.6, which is derived from Algorithm A3 in [RS87], takes as input the C/F-splitting determined by the first pass. We loop over all fine grid points $i \in F$ and their strongly coupled fine neighbors $j \in S_i \cap F$. Then we check whether these points share a common C-point, i.e. a point k such that both $k \in S_i$ and $k \in S_j$. If such a k does not exist, either i or j is added to the set of coarse grid points, where i is given precedence if two or more additional coarse points would be needed to ensure stability for interpolating e_i. This algorithm also takes $O(n \cdot nzr^2)$ operations (outer iteration $O(n)$, inner iteration and $S_j \cap S_i \cap C = \emptyset$ check each $O(nzr)$).


(a) Strong Couplings


(b) After first coarsening phase (red points are coarse)

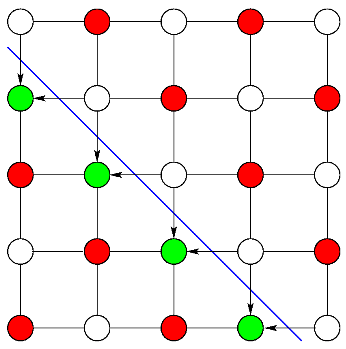

(c) After second coarsening phase (red and green points are coarse)

Figure 2.7.: Finite difference discretization of a PDE with a jumping diffusion coefficient (Example 2.3). The blue line gives the phase boundary; the arrows denote a strong coupling only in the indicated direction.

algorithm 2.6 AmgPhaseII(Ω, S, C, F)

```
begin
  for $i \in F$ do
     $\tilde{C} \leftarrow \emptyset$;
     for $j \in S_i \cap F$ do
        if $S_j \cap S_i \cap C = \emptyset$
          then
               if $\tilde{C} \neq \emptyset$
                 then          change $i$, because otherwise 2 points need to be changed
                     $C \leftarrow C \cup \{i\}$;
                     $F \leftarrow F \setminus \{i\}$;
                     GOTO NEXTi;
                  else                        $j$ could be an interpolation point for $i$
                     $\tilde{C} \leftarrow \{j\}$;
               fi;
        fi;
     od;
     $C \leftarrow C \cup \tilde{C}$;
     $F \leftarrow F \setminus \tilde{C}$;
     NEXTi :
  od;
end
```

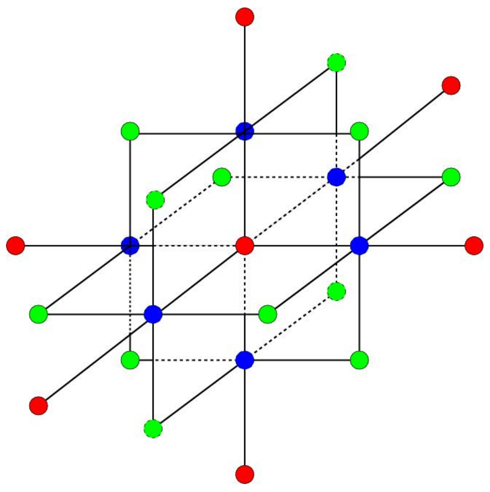


Figure 2.8.: Standard coarsening for a three-dimensional Laplacian. The blue points are fine grid points, the red and green points belong to the coarse grid. On the coarse level, the point in the center has connections to 18 other points. After the application of aggressive coarsening using the sets $\hat{S}_i^{(2,2)}$, all green points are also fine grid points and the original 7-point stencil is recovered on the coarse level.

Example 2.3 We consider the PDE

$$-\nabla \cdot \eta(x,y)\nabla \mathsf{u}(x,y) = \mathsf{f}(x,y)$$

on a square domain with Dirichlet boundary conditions, and

$$\eta(x,y) = \begin{cases} 1 \text{ if } x+y > 0.9 \\ 1000 \text{ else.} \end{cases}$$

In Figure 2.7(a) we show the strong couplings ($\alpha = 0.25$) for a finite difference discretization. The blue line indicates the phase boundary for η. We see that the strength matrix is not symmetric (the arrows denote the strong couplings that are only present in one direction). Figure 2.7(b) shows the coarse grid (red points) after phase I (Algorithm 2.5) of the coarsening scheme. We see that the fine grid points at the upper right side to the phase boundary are strongly dependent on the points at the lower left side of the boundary, these points however are also fine grid points that do not depend on the same set of coarse grid points, thus violating condition C1. Algorithm 2.6 repairs this problem by inserting the green points in Figure 2.7(c).

2.7. Aggressive Coarsening

In geometric multigrid settings, the coarse grid operator is usually just the discretization of the underlying problem on a smaller mesh, hence the number of nonzeros per matrix

row will be roughly the same on all levels. In consequence, the memory requirements per level decrease proportional to the mesh size.
In AMG (but also in the case of semi-algebraical approaches [Den82]), the coarse grid operator is computed from the transfer operators and the fine grid operator,

$$A_{l+1} = R_l A_l P_l.$$

This approach establishes many connections, i.e. matrix entries, between points that were previously far away from each other. The matrix A_{l+1} may have the same number or even more nonzeros than A_l. This not only increases the storage requirements, but also the compute time for a matrix-vector multiplication (which is proportional to the overall number of nonzeros in a matrix), the computation of A_{l+1} itself and the setup on the next coarser level.

Example 2.4 We consider the simple case of a 7 point isotropic finite difference stencil in three spatial dimensions, e.g. the discretization of the Laplacian, see Figure 2.8. Using classical Ruge-Stüben coarsening, we obtain a coarse grid with $N/2$ vertices, where N is the size of the fine grid (We ignore any boundary conditions for these considerations). We assume that interpolation directly follows the strong connections. Now, on the next coarser level, each matrix row contains 19 non-zero values. In consequence, the matrix on level 2 has $O(9.5 \cdot N)$ non-trivial entries , comparing to only $O(7N)$ on the finest level.

A way around this is to allow smaller coarse grids, i.e. with less coarse grid points. To this end, however, we have to give up coarsening principle C1 and replace it by a weaker condition [SMYH06],

C1' Let any fine grid point $i \in F$ be strongly connected to at least one coarse grid point $j \in C$.

As we have already pointed out in the previous section, a stable AMG interpolation scheme requires that *all* strong connections of a fine grid point $i \in F$ are reflected in the prolongation operator. Now, Condition C1' does not guarantee us that the strongly connected fine grid points $k \in S_i \cap F$ are strongly connected to the set $S_i \cap C$, i.e. the coarse grid points directly connected to i. We can however at least be sure that k has a strong connection to at least one coarse grid point $v \in S_k \cap C$. The idea is then to interpolate the value at $i \in F$ not only from the directly coupled coarse grid points $S_i \cap C$, but also from the set of indirectly coupled points

$$\bigcup_{k \in S_i \cap F} (S_k \cap C)\,.$$

In Sections 2.8.5 - 2.8.8 we will give various approaches to long-range interpolation.
In the remainder of this section, we describe the *aggressive coarsening* approach ([Stü99], Section 7.1.2). This algorithm requires an extended concept of strength.

Definition 2.8 *The point i is* strongly n-connected *to another point j along a path of length m if there exists a sequence of variables $i_0, i_1, i_2, \ldots, i_m$ such that $i_{k+1} \in S_{i_k}$ for all $k = 0, \ldots, m-1$ and $i_0 = i,\ i_m = j$.*
For $q \geq 1,\ m \geq 1$, i is strongly n-connected to $j \neq i$ w.r.t. (q, m) is there exists at least q paths of length $\leq m$ such that i is strongly n-connected to j along each of these paths. Analogously to (2.30) *we then define the sets $S_i^{(q,m)}$*

$$S_i^{(q,m)} = \{j \in \Omega :\ i \textit{ strongly n-connected to } j \textit{ w.r.t } (q, m)\}.$$

It is clear that $S_i = S_i^{(1,1)}$. For aggressive coarsening, usually the sets $S_i^{(1,2)}$ or $S_i^{(2,2)}$ are used. However, the computation of $S_i^{(q,m)}$ (and its transpose $(S_i^{(q,m)})^T$ may be quite expensive. Instead a more practical approach is to employ the standard coarsening algorithm 2.5 twice. First, the coarsening process is applied using the sets S_i as usual. Then, we only regard the coarse points $i \in C$ and the strong connection paths between them,

$$\hat{S}_i^{(q,m)} = \{j \in C :\ i \text{ strongly n-connected to } j \text{ w.r.t } (q, m)\}.$$

These sets $\hat{S}_i^{(q,m)}$ are then fed into a second run of the coarsening scheme 2.5 and we obtain the final set of coarse grid points. Returning to our 3D Laplacian example (Figure 2.8), we compute the sets $S_i^{(2,2)}$ for the green points and see that they are strongly n-coupled to the red points w.r.t $(2, 2)$. Hence, an additional coarsening pass only leaves the red points on the coarse grid.
In Section 3.3 we will introduce additional coarsening schemes that only satisfy condition C1'. The advantage of these algorithms is that they can easily be parallelized.

2.8. Interpolation

After the coarse grid points and hence the coarse space have been determined, we need to construct the interpolation operator $P : \mathbb{R}^{N_{l+1}} \to \mathbb{R}^{N_l}$ and the restriction operator $R : \mathbb{R}^{N_l} \to \mathbb{R}^{N_{l+1}}$. Throughout this section, we will use the following notation,

$E_i = \{j \neq i :\ a_{ij} \neq 0\}$	set of neighbors of i.
$C_i = S_i \cap C$	set of strongly connected coarse grid points of i.
$F_i^s = S_i \cap F$	set of strongly connected fine grid points of i.
$E_i^w = E_i \setminus S_i$	set of weakly connected neighbors of i.
I_i	set of interpolatory points for i.

In the following, without loss of generality we assume that each fine grid point $i \in F$ has at least one strong connection a_{ij}, $j \in S_i$. If this is not the case, the error at this point is assumed to be reduced efficiently by smoothing only.

2.8.1. Limit case: Direct solver

Before we proceed to practically useful interpolation schemes, we first show that for a special choice of the interpolation and smoothing operators ([Stü99], Section 2.3), the two-grid method with one post-smoothing step

$$e^{it+1} = M\left(I - PA_C^{-1}RA\right)e^{it}$$

as well as the two-grid method with one pre-smoothing step

$$e^{it+1} = \left(I - PA_C^{-1}RA\right)Me^{it}$$

turn into direct solvers, i.e. we have

$$\left(I - PA_C^{-1}RA\right)M = M\left(I - PA_C^{-1}RA\right) = 0$$

We assume that we have been given a C/F-splitting of $\Omega = C\dot{\cup}F$. Then, we re-arrange the coarse and fine components of u, f and A into coarse and fine components,

$$Au = \begin{pmatrix} A_{FF} & A_{FC} \\ A_{CF} & A_{CC} \end{pmatrix} \begin{pmatrix} u_F \\ u_C \end{pmatrix} = \begin{pmatrix} f_F \\ f_C \end{pmatrix} = f.$$

In the following we only assume that A and A_{FF} are invertible. (This is always the case if A is symmetric positive definite). We define a special smoother $\hat{M}$ that only acts on the fine grid points,

$$\begin{aligned} u_F^{it+1} &= A_{FF}^{-1}\left(f_F - A_{FC}u_C^{it}\right) \\ u_C^{it+1} &= u_C^{it}. \end{aligned}$$

For the error propagation we have

$$\begin{aligned} e_F^{it+1} &= -A_{FF}^{-1}\left(A_{FC}e_C^{it}\right) \\ e_C^{it+1} &= e_C^{it}. \end{aligned}$$

Furthermore, letting I_{CC} be the identity matrix on the coarse variables, define the following restriction and interpolation operators, by

$$\hat{P} = \begin{pmatrix} -A_{FF}^{-1}A_{FC} \\ I_{CC} \end{pmatrix}, \quad \hat{R} = \begin{pmatrix} I_{CC} & -A_{FF}^{-1}A_{CF} \end{pmatrix} \tag{2.33}$$

Now, for any restriction operator operator of the form

$$R = \begin{pmatrix} R_{CF} & I_{CC} \end{pmatrix}$$

a straightforward calculation shows that the coarse grid operator equals the Schur complement,

$$A_C = RA\hat{P} = A_{CC} - A_{CF}A_{FF}^{-1}A_{FC}$$

which is invertible as A_{FF} and A are. Furthermore, the two-grid cycle with one post-smoothing step reduces the error to zero,

$$\left(I - \tilde{P}A_C^{-1}RA\right)\tilde{M} = 0,$$

i.e. we have constructed a direct solver. The same also holds for the cycle

$$\tilde{M}\left(I - PA_C^{-1}\tilde{R}A\right), \quad A_C = \hat{R}AP = A_{CC} - A_{CF}A_{FF}^{-1}A_{FC}$$

where P is an arbitrary interpolation operator of the form

$$P = \begin{pmatrix} P_{CF} & I_{CC} \end{pmatrix}.$$

These smoothing and interpolation operators are not practically useful (we would need to compute the inverse of A_{FF}). except in some rare cases. (For example, consider a finite difference discretization of Poisson equation in one spatial dimension yielding a tri-diagonal matrix. Here A_{FF} becomes diagonal if every other point belongs to the coarse grid.) However, these results indicate that we should approximate $-(A_{FF})^{-1}A_{FC}$ to obtain an efficient interpolation. Furthermore, we have a justification to use the Galerkin ansatz $A_C = RAP$ for the coarse grid operator. In section 2.9 we will further discuss the consequences of this choice.

2.8.2. Approximation properties

Like in the geometric case (see Section 2.2, especially Definition 2.2) we need, in addition to a smoothing property, an approximation property to show convergence of the multigrid cycle. Again, we formulate this property in terms of the norms $\|u\|_1^2 = u^T Au$ and $\|u\|_2^2 = u^T AD^{-1}Au$.
In the remainder of this section, we restrict ourselves again to the class of M-matrices (Definition 2.5) and essentially positive matrices (Definition 2.6). First, we consider a two-grid method with one post-smoothing step,

$$e^{it+1} = MTe^{it} = M\left(I - PA_C^{-1}RA\right)e^{it}.$$

We assume that the two-grid correction operator $T = \left(I - PA_C^{-1}RA\right)$ satisfies

$$\begin{aligned} \|Te\|_1^2 &\leq \|e\|_1^2 && (2.34)\\ \|Te\|_1^2 &\leq \tau\|Te\|_2^2 && (2.35)\end{aligned}$$

for some $\tau > 0$ independent of e. Furthermore, we assume that the smoother M satisfies the smoothing property (2.27),

$$\|Me\|_1^2 \leq \|e\|_1^2 - \sigma\|e\|_2^2$$

where $\sigma > 0$ is again independent of e. Then we obtain (see also [Stü99], Theorem 4.1)

$$\|MTe\|_1^2 \leq \|Te\|_1^2 - \sigma\|Te\|_2^2 \leq \left(1 - \frac{\sigma}{\tau}\right)\|Te\|_1^2 \leq \|e\|_1^2. \qquad (2.36)$$

In the following, we will formulate conditions on the interpolation operator such that (2.35) is satisfied. For a more extensive discussion, we refer to [Stü99], Chapter 4. Inequality (2.34), on the other hand, can be shown for any full-rank interpolation operator. This will be done in Section 2.9.
We now translate (2.35) into a condition on the interpolation operator. Again, assuming a splitting of Ω into C and F is given, we re-arrange all relevant operators into coarse and fine blocks,

$$P = \begin{pmatrix} P_{FF} & P_{FC} \\ P_{CF} & P_{CC} \end{pmatrix}, \quad A = \begin{pmatrix} A_{FF} & A_{FC} \\ A_{CF} & A_{CC} \end{pmatrix}, \quad D = \begin{pmatrix} D_{FF} & 0 \\ 0 & D_{CC} \end{pmatrix}, \quad e = \begin{pmatrix} e_F \\ e_C \end{pmatrix}$$

and define the 0-inner product and norm restricted to the F-variables,

$$(x, y)_{0,F} = x^T D_{FF} y, \ \|x\|_{0.F} = \sqrt{(x, x)_{0,F}}.$$

Theorem 2.6 *[Stü99], Theorem 4.2*
If the C/F-splitting and the interpolation P_{FC} are such that, for all e,

$$\|e_F - P_{FC} e_C\|_{0,F}^2 \leq \tau \|e\|_1^2 \tag{2.37}$$

with τ being independent of e, then (2.35) *is satisfied.*

In terms of the eigenvalues λ and eigenvectors ϕ of $D^{-1}A$, (2.37) becomes

$$\|\phi_F - P_{FC}\phi_C\|_{0,F}^2 \leq \tau\lambda\|\phi\|_0^2$$

(see (2.26)). We conclude that especially for the small eigenvalues, the corresponding eigenvectors should be approximated exactly by the interpolation. These eigenmodes correspond to the smooth error components that cannot be handled efficiently by the smoother, see (2.27) and the discussion thereafter. Moreover, in the limit case of zero row sum matrices, the smallest eigenvalue equals zero and the corresponding eigenvector is the constant vector throughout the domain. This eigenvector then must be approximated exactly. In practical examples, the matrices often have zero row sums except these rows that involve a boundary condition. For these linear systems, it is also important that locally constant vectors are interpolated exactly, a property that we always will enforce for the interpolation operators introduced in the next subsections.
In the remainder of this section, we restrict ourselves again to the class of M-matrices (Definition 2.5) and essentially positive matrices (Definition 2.6). As we have seen in Sections 2.4 - 2.5, for this type of matrices algebraically smooth errors vary slowly along the strong couplings,

$$\frac{c}{2}\sum_{i,j} -a_{ij}^{-}(e_i - e_j)^2 + \sum_i s_i e_i^2 \ll \sum_i a_{ii} e_i^2.$$

This motivates us to approximate the error e_i at a fine grid point i by a weighted average of its strongly connected coarse neighbors $j \in C_i$,

$$e_i = \begin{cases} e_i \text{ if } i \in C \\ \sum_{j \in S_i \cap C} w_{ij} e_j \text{ else.} \end{cases} \tag{2.38}$$

To give an indication on how the weights w_{ij} should be chosen, we again look at the smooth error characterization in term of the norms $\|\cdot\|_1$ and $\|\cdot\|_2$,

$$\begin{aligned} e^T A D^{-1} A e = \|e\|_2^2 &\ll \|e\|_1^2 = e^T A e \\ \Leftrightarrow r^T D^{-1} r &\ll e^T r, \end{aligned}$$

or, on average for each i, we have that the scaled residual is much smaller than the error,

$$\frac{|r_i|^2}{a_{ii}} \ll |e_i| \cdot |r_i| \quad \rightsquigarrow \quad \frac{|r_i|}{a_{ii}} \ll |e_i|.$$

To obtain the actual interpolation weights w_{ij}, we hence approximate

$$a_{ii} e_i + \sum_{j \in E_i} a_{ij} e_j = r_i = 0. \tag{2.39}$$

by

$$e_i + \sum_{j \in E_i} w_{ij} e_j = 0. \tag{2.40}$$

Now, we formulate a sufficient condition for (2.37) in terms of the interpolation weights w_{ij}.

Theorem 2.7 *([RS87], Theorem 5.3) Let A be symmetric positive definite and assume, for any given set C of C-points, that P is of the form (2.38) with $w_{ij} \geq 0$ and $\sum_j w_{ij} \leq 1$. Then property (2.37) is satisfied if the following two inequalities hold with $\tau > 0$ independently of e,*

$$\sum_{i \in F} \sum_{j \in C} a_{ii} w_{ij} (e_i - e_j)^2 \leq \frac{\tau}{2} \sum_{i,k} (-a_{ik})(e_i - e_k)^2, \tag{2.41}$$

$$\sum_{i \in F} a_{ii} (1 - \sum_k w_{ik}) e_i^2 \leq \tau \sum_i \left(\sum_k a_{ik} \right) e_i^2. \tag{2.42}$$

Note that Condition (2.42) implies that for zero row sum matrices ($\sum_k a_{ik} = 0$), the sum of all interpolation weights $\sum_k w_{ik}$ for an $i \in F$ should be one. In other words, a constant vector should be interpolated to a constant vector.

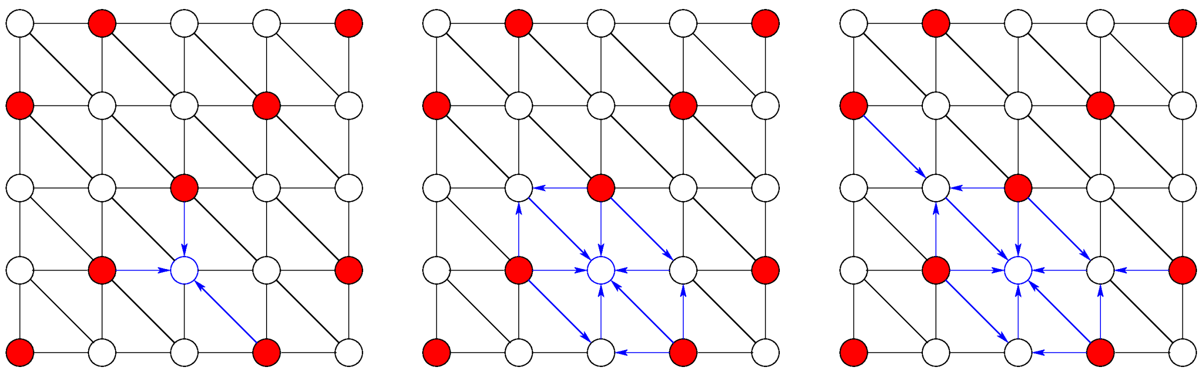

(a) Direct interpolation (b) (Modified) classical interpolation (c) Standard and extended interpolation

Figure 2.9.: Comparison of interpolation methods for the mixed problem from Example 2.2. The red dots denote the coarse grid points, the blue circle is the fine grid point whose value is to be interpolated. The blue arrows show how the values are interpolated directly and indirectly from the coarse to the fine grid points.

2.8.3. Direct Interpolation

The first and simplest ansatz to obtain an interpolated value for e_i, $i \in F$, is just to set $I_i = C_i$, i.e. only to interpolate directly from strongly connected coarse grid points and compute, for all $j \in C_i$,

$$w_{ij} = -\frac{1}{a_{ii}} \frac{\sum_{k \in E_i} a_{ik}}{\sum_{k \in C_i} a_{ik}} a_{ij}. \tag{2.43}$$

Here, the correction factor $\frac{\sum_{k \in E_i} a_{ik}}{\sum_{k \in C_i} a_{ik}}$ is needed to ensure that in the case of zero row sum matrices (where $a_{ii} = -\sum_{k \in E_i} a_{ik}$), the sum over all interpolation weights equals one,

$$\sum_{j \in C_i} w_{ij} = -\frac{1}{a_{ii}} \frac{\sum_{k \in E_i} a_{ik}}{\sum_{k \in C_i} a_{ik}} \sum_{j \in C_i} a_{ij} = 1$$

which implies that a constant vector on the coarse grid will be interpolated to a constant vector on the fine grid. We have pointed out in the last subsection that this is crucial for an efficient AMG interpolation operator. In the next theorem, we show how the distribution of coarse interpolatory points around a fine grid point i influences the constant τ. Note that if we have rows with negative row sum, τ becomes dependent on the smallest row sum as well as the smallest eigenvalue of A. On the other hand, for the class of weakly diagonally dominant matrices, (2.37) can be satisfied with uniform τ if the sets of interpolatory points are large enough.

Theorem 2.8 *([Stü99], Theorem 4.3 - 4.4)*
Let the symmetric M-matrix A satisfy $\sum_j a_{ij} \geq -c$ for some $c \geq 0$ and assume $e^T A e \geq$

$\epsilon e^T e$ for all e with some $\epsilon > 0$. With fixed $\tau \geq 1$, select a C/F-splitting so that, for each $i \in F$, there is a set of interpolatory points $I_i \subset C \cap \{j \in E_i : \; a_{ij} \neq 0\}$ satisfying

$$\sum_{j \in I_i} |a_{ij}| \geq \frac{1}{\tau} \sum_{j \in E_i} |a_{ij}|. \tag{2.44}$$

Then the interpolation (2.38) *with weights* (2.43) *satisfies* (2.37) *with τ replaced by some $\tilde{\tau} = \tilde{\tau}(\epsilon, c, \tau)$. As a function of ϵ and c, we have $\tilde{\tau} \to \infty$ if either $c \to \infty$ or $\epsilon \to 0$. If $\sum_j a_{ij} \geq 0$, then $\tilde{\tau} = \tau$.*

If we have strong negative and positive connections, the denominator $\sum_{k \in C_i} a_{ik}$ may become (close to) zero. To avoid this instability we separate the positive and negative connections. To this end, we split the set of strongly connected coarse grid points into two sets of interpolatory points,

$$\begin{aligned} C_i^- &= \{j \in C_i : \; a_{ij} < 0\} \\ C_i^+ &= \{j \in C_i : \; a_{ij} > 0\} \end{aligned}$$

We rewrite (2.40),

$$e_i + \sum_j w_{ij}^- e_j + \sum_j w_{ij}^+ e_j = 0,$$

where

$$w_{ij}^- = -\frac{1}{a_{ii}} \frac{\sum_{k \in E_i} a_{ik}^-}{\sum_{k \in C_i^-} a_{ik}^-} \, a_{ij} \text{ if } a_{ij} < 0 \tag{2.45}$$

$$w_{ij}^+ = -\frac{1}{a_{ii}} \frac{\sum_{k \in E_i} a_{ik}^+}{\sum_{k \in C_i^+} a_{ik}^+} \, a_{ij} \text{ if } a_{ij} > 0. \tag{2.46}$$

If a matrix row i only contains negative (positive) off-diagonal entries, then $C_i^+ = \emptyset$ ($C_i^- = \emptyset$) and we set the corresponding correction factor to zero. In the case of M-matrices, this definition of direct interpolation is identical to the construction above. The approximation property is also a direct extension of the previous result.

Theorem 2.9 *([Stü99], Theorem 4.6) Let $A > 0$ and $t_i = a_{ii} - \sum_{j \in E_i} |a_{ij}| \geq 0$. With fixed $\tau \geq 1$, select a C/F-splitting such that the following holds for each $i \in F$: If $\{j \in E_i : \; a_{ij} < 0\} \neq \emptyset$, there is a set $C_i^- \subset C \cap \{j \in E_i : \; a_{ij} < 0\}$ satisfying*

$$\sum_{j \in C_i^-} |a_{ij}| \geq \frac{1}{\tau} \sum_{j \in E_i} |a_{ij}^-|,$$

and, if $\{j \in E_i : \; a_{ij} > 0\} \neq \emptyset$, there is a set $C_i^+ \subset C \cap \{j \in E_i : \; a_{ij} > 0\}$ satisfying

$$\sum_{j \in C_i^+} a_{ij} \geq \frac{1}{\tau} \sum_{j \in E_i} a_{ij}^+.$$

Then the interpolation (2.38) *with weights (2.45-2.46) satisfies* (2.37).

The largest drawback of the direct interpolation scheme is the fact that only the strong connections to directly coupled coarse grid points are used, $a_{ij}: \ j \in C_i$. The smooth error e_i at $i \in F$ however also depends on the smooth error e_k at another $k \in F_i^s$ if i is strongly coupled to k. For example, in Figure 2.9(a) we give the interpolation directions for the second order with mixed derivatives of Example 2.2. We see that for the interpolation of the blue fine grid points, only three of the six strong couplings are used. More precisely, in (2.44), we need $\tau \geq \frac{20}{9}$.
In the following sections, we construct interpolation operators that employ all strong couplings a_{jk}.

2.8.4. Classical and Modified classical interpolation

In this section, we describe an interpolation technique that, for each fine grid point $i \in F$, takes into account all strong connections a_{ik}, $k \in S_i$ regardless whether k is coarse or fine. Still, we only interpolate from the directly connected coarse grid points $j \in C_i$. To determine the weights, however, we also follow the paths

$$i \overset{a_{ik}}{\leftarrow} k \overset{a_{kj}}{\leftarrow} j.$$

if $j \in S_k$ and $k \in S_i$. Remember from the coarsening criterion C1 that, for each fine $k \in F_i^s$, there must be a coarse $j \in C$ such that $j \in S_i$ as well as $j \in S_k$, i.e. there must be a common coarse point j for the strongly connected fine grid points points i and k.
Now, we again approximate (2.39),

$$a_{ii}e_i = -\sum_{j \in E_i} a_{ij}e_j = -\sum_{j \in C_i} a_{ij}e_j - \sum_{k \in F_i^s} a_{ik}e_k - \sum_{l \in E_i^w} a_{il}e_l.$$

First, we replace $\sum_{l \in E_i^w} a_{il}e_l$ by $\sum_{l \in E_i^w} a_{il}e_i$ (though the assumption $e_l \approx e_i$ may seem strange for weakly coupled points l, we need this to ensure that constants are interpolated exactly). Then, we replace each e_k, $k \in F_i^s$, by

$$e_k \leftarrow \frac{1}{\sum_{m \in C_i} a_{km}} \sum_{j \in C_i} a_{kj}e_j. \tag{2.47}$$

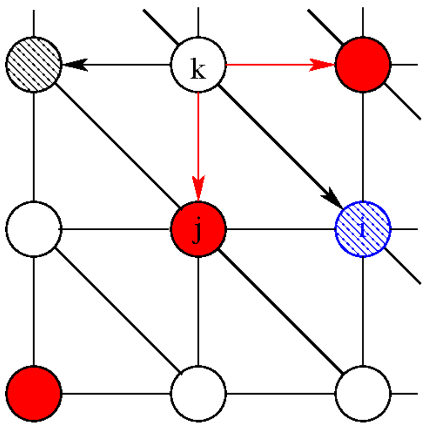
We obtain the interpolation weights [HMY02]

$$w_{ij} = -\frac{1}{a_{ii} + \sum_{l \in E_i^w} a_{il}} \left(a_{ij} + \sum_{k \in F_i^s} \frac{a_{ik}a_{kj}}{\sum_{m \in C_i} a_{km}} \right). \tag{2.48}$$

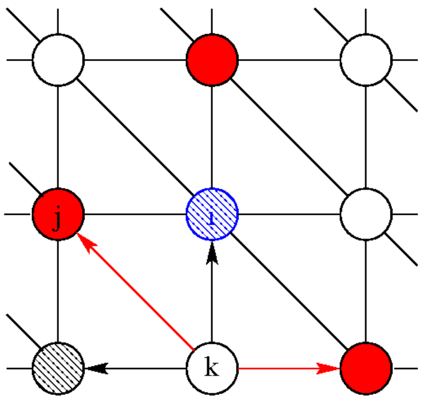
For symmetric positive M-matrices, we can show (2.37) using the following variant of Theorem 5.5 from [RS87],

Theorem 2.10 *Let $\zeta > 0$ be a fixed number. Assume, for any symmetric, weakly diagonally dominant M-matrix A, that the C-points and $S_i \cap C$ ($i \in F$) are picked such that, for each k and $j \in C_i$ with $i \in F_{kj}$, we have*

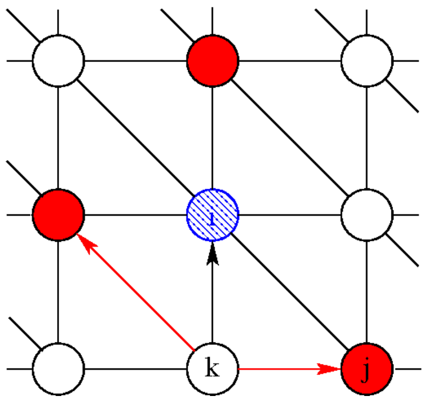
$$\zeta \sum_{l \in C_i} -a_{kl} \geq \sum_{v \in F_{kj}} -a_{kv} \tag{2.49}$$



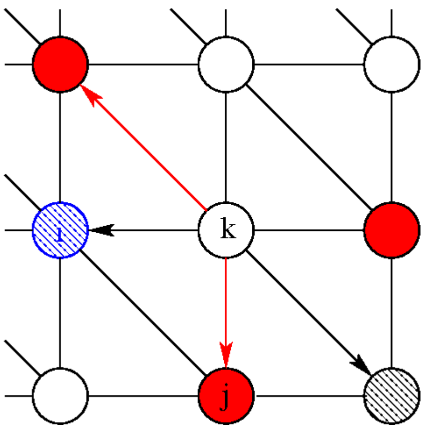
(a) $\sum_{l \in C_i} -a_{kl} = 2, \sum_{v \in F_{kj}} -a_{kv} = 1.25$



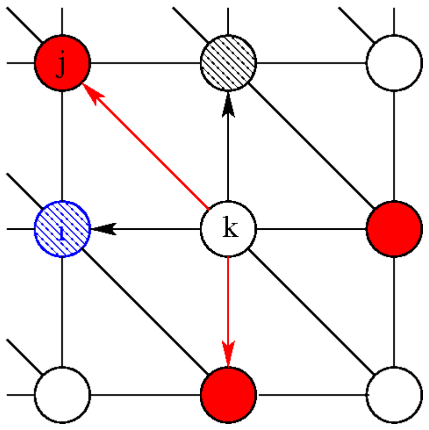
(b) $\sum_{l \in C_i} -a_{kl} = 1.25, \sum_{v \in F_{kj}} -a_{kv} = 2$



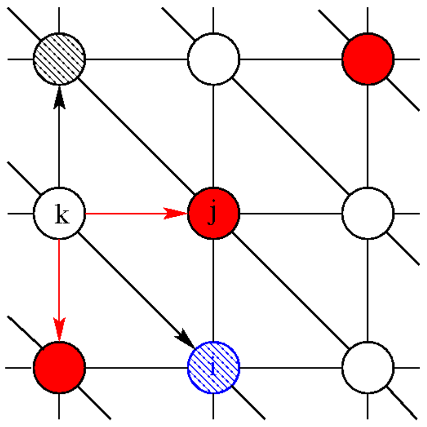
(c) $\sum_{l \in C_i} -a_{kl} = 1.25, \sum_{v \in F_{kj}} -a_{kv} = 1$



(d) $\sum_{l \in C_i} -a_{kl} = 1.25, \sum_{v \in F_{kj}} -a_{kv} =$ 1.25



(e) $\sum_{l \in C_i} -a_{kl} = 1.25, \sum_{v \in F_{kj}} -a_{kv} = 2$



(f) $\sum_{l \in C_i} -a_{kl} = 2, \sum_{v \in F_{kj}} -a_{kv} = 1.25$

Figure 2.10.: (Modified) classical interpolation for the mixed problem from Example 2.2. We want to interpolate the value at the blue point i. The red dots denote the sets C_i, the ruled nodes show the sets F_{kj} for different constellations of k and j. The red arrows indicate the weights a_{kl}, $l \in C_i$, while the black arrows correspond to the weights $a_{k\nu}$, $\nu \in F_{kj}$.

where $F_{kj} = \{v \in F : \; j \in S_v \cap C, \; k \in S_v \cap F\}$. Second, let the strength threshold α be chosen such that for a fixed $\tau \geq 1$ we have, for each $i \in F$,

$$\sum_{i \in S_i} -a_{ik} \geq \frac{1}{\tau} \sum_{j \in E_i} -a_{ij} \tag{2.50}$$

Then, the interpolation (2.38) *with weights* (2.48) *satisfies* (2.37) *with some $\tilde{\tau} = \tilde{\tau}(\zeta, \tau)$ that does not depend on A.*

Before we proceed to the proof, we first explain the role of F_{kj}. This set contains, for a given pair $j \in C$, $k \in F$, the fine grid points v that have strong connections to both j and k. Now, condition (2.49) demands that, for a fine grid point i that is strongly coupled to another fine grid point k, the total amount of coupling from this point k to the set of directly coupled coarse grid points for i, C_i, is (up to the factor ζ) larger than the sum of the couplings to other fine grid points v which have strong couplings to the same j and k. In Figure 2.9(b), we give the sets F_{kj} for six different constellations of k and j as well as the quantities of (2.49).
The second condition (2.50) ensures that for each fine grid point i the sum of the strong connections dominates the sum of the weak connections. In contrast to (2.44), we do not require that these strong connections all lead to a coarse interpolatory point.
Proof: As A is diagonally dominant, we have $a_{ii} = -\sum_{j \in E_i} a_{ij} + b_i$, where $b_i \geq 0$. Using (2.50) we estimate the denominator of (2.48),

$$a_{ii} + \sum_{k \in E_i^w} a_{ik} = b_i - \sum_{k \in S_i} a_{ik} \geq b_i + \frac{1}{\tau} \sum_{j \in E_i} -a_{ij} \geq \frac{1}{\tau} a_{ii}$$

Hence, we have

$$\begin{aligned} \sum_{i \in F} a_{ii} \left(1 - \sum_{j \in C_i} w_{ij} \right) e_i^2 &= \sum_{i \in F} \frac{a_{ii}}{a_{ii} + \sum_{k \in E_i^w} a_{ik}} \left(a_{ii} + \sum_{k \in E_i^w} a_{ik} + \sum_{j \in C_i} a_{ij} + \sum_{k \in F_i^s} a_{ik} \right) e_i^2 \\ &\leq \tau \sum_i \left(\sum_j a_{ij} \right) e_i^2 \end{aligned}$$

which proves (2.42). To show (2.41), first note that w_{ij} is always positive. We again use $a_{ii} + \sum_{k \in E_i^w} a_{ik} \geq \frac{1}{\tau} a_{ii}$ and obtain, by $(e_i - e_j)^2 \leq 2\left[(e_i - e_k)^2 + (e_k - e_j)^2\right]$,

$$a_{ii} w_{ij} (e_i - e_j)^2 \leq -\tau a_{ij} (e_i - e_j)^2 - 2\tau \sum_{k \in F_i^s} \frac{a_{ik} a_{kj}}{\sum_{m \in C_i} a_{km}} \left[(e_i - e_k)^2 + (e_k - e_j)^2 \right].$$

We take the sum over $i \in F$, $j \in C_i$ and obtain

$$-\sum_{i \in F} \sum_{j \in C_i} \sum_{k \in F_i^s} \frac{a_{ik} a_{kj}}{\sum_{m \in C_i} a_{km}} (e_i - e_k)^2 = \sum_{i \in F} \sum_{k \in F_i^s} (-a_{ik})(e_i - e_k)^2.$$

Regarding the remaining term, use the definition of F_{kj} to rewrite

$$\begin{aligned}
-\sum_{i\in F}\sum_{j\in C_i}\sum_{k\in F_i^s}\frac{a_{ik}a_{kj}}{\sum_{m\in C_i}a_{km}}(e_k-e_j)^2 &= \sum_{k\in F}\sum_{j\in C}\sum_{i\in F_{kj}}\frac{a_{ik}}{\sum_{m\in C_i}a_{km}}(-a_{kj})(e_k-e_j)^2\\
&\le \zeta\sum_{k\in F}\sum_{j\in C}(-a_{kj})(e_k-e_j)^2
\end{aligned}$$

where we have used (2.49) in the last inequality. Combining the last three estimates and expanding the sum over all $i\in F$ and $j\in C$, we obtain (2.41),

$$\sum_{i\in F}\sum_{j\in C}a_{ii}w_{ij}(e_i-e_j)^2\le(3\tau+2\tau\zeta)\sum_{i,j}(-a_{ij})\,(e_i-e_j)^2.$$

In the case of positive and negative off-diagonal connections, the sum $\sum_{m\in C_i}a_{km}$ may approach zero. To circumvent this potential instability, we modify the interpolation formula as follows [HMY02],

$$w_{ij}=-\frac{1}{a_{ii}+\sum_{l\in E_i^w}a_{il}}\left(a_{ij}+\sum_{k\in F_i^s}\frac{a_{ik}\bar{a}_{kj}}{\sum_{m\in C_i}\bar{a}_{km}}\right).\tag{2.51}$$

where

$$\bar{a}_{kj}=\begin{cases}a_{kj}\text{ if } sign(a_{kj})=-sign(a_{kk})\\ 0\text{ else.}\end{cases}$$

With this modification, we can show stability for essentially positive and weakly diagonally dominant matrices A. Note that in this case we always have $\bar{a}_{kj}=a_{kj}^-$.

Theorem 2.11 *Let $\zeta>0$ be a fixed number. Assume, for any symmetric, weakly diagonally dominant positive matrix A, that the C-points and C_i $(i\in F)$ are picked such that, for each k and $j\in C_i$ with $i\in F_{kj}$, we have*

$$\tilde{\zeta}\sum_{l\in C_i}-a_{kl}^-\ge\sum_{v\in F_{kj}}-a_{kv}^-\tag{2.52}$$

where $F_{kj}=\{v\in F:\ j\in S_v\cap C,\ k\in S_v\cap F\}$ and $\bar{a}_{kl}$ is defined as above. Second, let the strength threshold α be chosen such that for a fixed $\tau\ge1$ we have, for each $i\in F$,

$$\sum_{i\in S_i}-a_{ik}\ge\frac{1}{\tau}\sum_{j\in E_i}-a_{ij}$$

Then, the interpolation (2.38) with weights (2.51) satisfies (2.37) with some $\tilde{\tau}=\tilde{\tau}(\zeta,\tau)$ that does not depend on A.

Proof: Inequality (2.42) is proven as in the previous case. Regarding (2.41), we start with

$$\begin{aligned} a_{ii}w_{ij}(e_i-e_j)^2 &\le -\tau a_{ij}(e_i-e_j)^2 - 2\tau \sum_{k\in F_i^s} \frac{a_{ik}\bar{a}_{kj}}{\sum_{m\in C_i}\bar{a}_{km}}\left[(e_i-e_k)^2+(e_k-e_j)^2\right] \\ &\le -\tau a_{ij}^-(e_i-e_j)^2 - 2\tau \sum_{k\in F_i^s} \left(\frac{a_{ik}a_{kj}^-}{\sum_{m\in C_i}a_{km}^-}\right)^- \left[(e_i-e_k)^2+(e_k-e_j)^2\right] \end{aligned}$$

It is easy to see that $\left(\frac{a_{ik}a_{kj}^-}{\sum_{m\in C_i}a_{km}^-}\right)^- = \frac{a_{ik}^-a_{kj}^-}{\sum_{m\in C_i}a_{km}^-}$. Hence, we can proceed as in the M-matrix case and obtain

$$\sum_{i\in F}\sum_{j\in C} a_{ii}w_{ij}(e_i-e_j)^2 \le (3\tau+2\tau\tilde{\zeta})\sum_{i,j}\left(-a_{ij}^-\right)(e_i-e_j)^2.$$

Now, we use the fact that A is essentially positive (2.28),

$$c\sum_{i,j}(-a_{ij}^-)(e_i-e_j)^2 \le \sum_{i,j}(-a_{ij})(e_i-e_j)^2$$

and obtain (2.41),

$$\sum_{i\in F}\sum_{j\in C} a_{ii}w_{ij}(e_i-e_j)^2 \le \frac{3\tau+2\tau\tilde{\zeta}}{c}\sum_{i,j}\left(-a_{ij}\right)(e_i-e_j)^2.$$

2.8.5. Standard Interpolation

The standard interpolation scheme ([Stü99], Section 7.2.1) also takes into account, for any fine grid point $i \in F$, all strong connections a_{ij}, $j \in S_i$. In contrast to the direct and classical interpolation methods described above, the set of interpolatory points for i is not necessarily a subset of S_i but may also include points in S_v, $v \in F_i^s$. In other words, the interpolation stencil for i reaches to points that are only connected via an intermediate point v,

$$i \overset{a_{iv}}{\leftarrow} v \overset{a_{vj}}{\leftarrow} j.$$

We call this kind of interpolation *distance-two interpolation*.
To derive an interpolation rule for a fine grid point $i \in F$, we first consider all its strongly connected fine neighbors $v \in F_i^s$ and replace,

$$e_v \leftarrow -\frac{1}{a_{vv}}\sum_{k\in E_v} a_{vk}e_k.$$

We obtain a modification of (2.39),

$$\hat{a}_{ii}e_i + \sum_{j\in\hat{E}_i}\hat{a}_{ij}e_j \approx 0. \tag{2.53}$$

where

$$\begin{aligned}
\hat{a}_{ik} &= \begin{cases} a_{ik} - \sum\limits_{v\in F_i^s} \frac{a_{iv}a_{vk}}{a_{vv}} \text{ if } k \notin F_i^s \\ -\sum\limits_{k\neq v\in F_i^s} \frac{a_{iv}a_{vk}}{a_{vv}} \text{ if } k \in F_i^s. \end{cases} \\
\hat{E}_i &= E_i \cup \bigcup_{v\in F_i^s} E_v \\
\hat{C}_i &= C_i \cup \bigcup_{v\in F_i^s} C_v \\
\hat{E}_i^- &= \{j \in E_i :\ \hat{a}_{ij} < 0\},\ \hat{E}_i^+ = \{j \in E_i :\ \hat{a}_{ij} > 0\} \\
\hat{C}_i^- &= \{j \in C_i :\ \hat{a}_{ij} < 0\},\ \hat{C}_i^+ = \{j \in C_i :\ \hat{a}_{ij} > 0\}
\end{aligned}$$

and apply the direct interpolation rules to this approximation (2.53),

$$w_{ij}^- = -\frac{1}{\hat{a}_{ii}} \frac{\sum_{k\in\hat{E}_i^+} \hat{a}_{ik}^-}{\sum_{k\in\hat{C}_i^-} \hat{a}_{ik}^-} \hat{a}_{ij} \text{ if } j \in \hat{C}_i^- \tag{2.54}$$

$$w_{ij}^+ = -\frac{1}{\hat{a}_{ii}} \frac{\sum_{k\in\hat{E}_i^-} \hat{a}_{ik}^+}{\sum_{k\in\hat{C}_i^-} \hat{a}_{ik}^+} \hat{a}_{ij} \text{ if } j \in \hat{C}^+. \tag{2.55}$$

Again, if one of the sets $\hat{C}_i^-$ or $\hat{C}_i^+$ is empty, then we set $w_{ij}^- = 0$ ($w_{ij}^+ = 0$).

2.8.6. Extended Interpolation

Extended interpolation [SFNMY08] is a distance-two interpolation scheme based on the modified classical distance-one interpolation (2.51). The only difference is that interpolation, for any fine grid point i, is not only from the directly connected coarse grid points $j \in C_i$ but also from all $j \in C_k$ if $k \in F_i^s$, i.e., we use the same interpolation set $\hat{C}_i$ as in the case of standard interpolation,

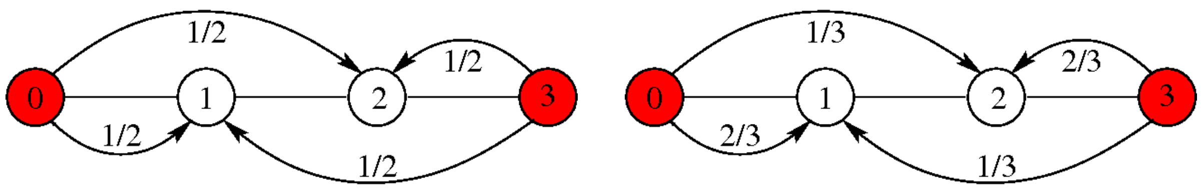
$$\hat{C}_i = C_i \cup \bigcup_{k\in F_i^s} C_k.$$

We obtain the interpolation weights (cf.(2.51))

$$w_{ij} = -\frac{1}{a_{ii} + \sum_{l\in E_i^w} a_{il}} \left(a_{ij} + \sum_{k\in F_i^s} \frac{a_{ik}\bar{a}_{kj}}{\sum_{m\in\hat{C}_i} \bar{a}_{km}} \right) \tag{2.56}$$

for all $j \in \hat{C}_i$, where again

$$\bar{a}_{kj} = \begin{cases} a_{kj} \text{ if } sign(a_{kj}) = -sign(a_{kk}) \\ 0 \text{ else.} \end{cases}$$



(a) Extended interpolation (b) Extended+i interpolation

Figure 2.11.: Extended and extended+i interpolation weights for the finite difference discretization of a 1D Laplace. The red points denote the coarse grid points.

The interpolation property for these weights (2.56) can be seen from the following straightforward extension of Theorem 2.11 (note the definition of $\hat{F}_i^s$).

Theorem 2.12 *Let $\zeta > 0$ be a fixed number. Assume, for any symmetric, weakly diagonally dominant positive matrix A, that the C-points and C_i $(i \in F)$ are picked such that, for each k and $j \in F_i^s$ with $i \in F_{kj}$, we have*

$$\tilde{\zeta} \sum_{l \in \hat{C}_i} -a_{kl}^- \geq \sum_{v \in \hat{F}_{kj}} -a_{kv}^- \tag{2.57}$$

where $\hat{F}_{kj} = \{v \in F : \ j \in \hat{C}_v, \ k \in F_v^s\}$ and $\bar{a}_{kl}$ is defined as above. Second, let the strength threshold α be chosen such that for a fixed $\tau \geq 1$ we have, for each $i \in F$,

$$\sum_{i \in S_i} -a_{ik} \geq \frac{1}{\tau} \sum_{j \in E_i} -a_{ij}$$

Then, the interpolation (2.38) *with weights* (2.56) *satisfies* (2.37) *with some $\tilde{\tau} = \tilde{\tau}(\zeta, \tau)$ that does not depend on A.*

As illustrated by the following example from this interpolation technique has however a minor approximation deficiency that already arises in very simple situations.

Example 2.5 [SFNMY08] Consider a finite difference discretization of Poisson's equation in one spatial dimension and let the coarsening be as depicted in Figure 2.11. In this case, formula (2.56) gives us the interpolation weights (Figure 2.11(a))

$$w_{10} = 1/2, \ w_{13} = 1/2, \ w_{20} = 1/2 \ w_{23} = 1/2,$$

where we would expect

$$w_{10} = 2/3, \ w_{13} = 1/3, \ w_{20} = 1/3, \ w_{23} = 2/3 \tag{2.58}$$

which would be given if standard interpolation (2.54) were used.

To circumvent this difficulty, we use a different approximation for the error at strongly connected fine grid points $k \in F_i^s$. Instead of (2.47) we substitute

$$e_k \leftarrow \frac{1}{\sum_{m \in C_i \cup \{i\}} a_{km}} \sum_{j \in C_i \cup \{i\}} a_{kj} e_j,$$

i.e. we also include the connections a_{ki} that lead from k back to i here. Then we apply the extended interpolation procedure as described above and obtain the following weights,

$$w_{ij} = -\frac{1}{\tilde{a}_{ii}} \left(a_{ij} + \sum_{k \in F_i^s} \frac{a_{ik} \bar{a}_{kj}}{\sum_{m \in \hat{C}_i \cup \{i\}} \bar{a}_{km}} \right) \tag{2.59}$$

for all $j \in \hat{C}_i$, where

$$\tilde{a}_{ii} = a_{ii} + \sum_{l \in E_i^w} a_{il} + \sum_{k \in F_i^s} \frac{a_{ik} \bar{a}_{ki}}{\sum_{l \in \hat{C}_i \cup \{i\}} \bar{a}_{kl}}.$$

This is called *extended+i interpolation* [SFNMY08]. In case of Example 2.5, we obtain the correct weights (2.58), see Figure 2.11(b).
To show the approximation property of extended+i interpolation, we conclude this section with the following result.

Theorem 2.13 *Let $\zeta > 0$ be a fixed number. Assume, for any symmetric, weakly diagonally dominant essentially positive matrix A, that the C-points and C_i ($i \in F$) are picked such that, for each k and $j \in F_i^s$ with $i \in F_{kj}$, we have*

$$\zeta \sum_{l \in \hat{C}_i} -a_{kl}^- \geq \sum_{v \in F_{kj}} -a_{kv}^- \tag{2.60}$$

where $\hat{F}_{kj} = \{v \in F : \; j \in \hat{C}_v, \; k \in F_v^s\}$. Furthermore, let the strength threshold α be chosen such that for a fixed $\tau \geq 1$ we have, for each $i \in F$,

$$\sum_{k \in S_i} -a_{ik} \geq \frac{1}{\tau} \sum_{j \in E_i} -a_{ij}$$

and, for each $i \in F$ and $k \in F_i^s$ such that $a_{ik} < 0$,

$$\sum_{j \in \hat{C}_i} -a_{kj} \geq \frac{1}{\tau^*} \sum_{j \in E_i} -a_{ij}$$

Then, the interpolation (2.38) with weights (2.59) satisfies (2.37) with some $\tilde{\tau} = \tilde{\tau}(\zeta, \tau)$ that does not depend on A.

Proof: We first consider the denominator $\tilde{a}_{ii}$ (let $0 \le b_i = a_{ii} + \sum_{k\in E_i} a_{ik}$ as before and note that $\bar{a}_{ik} = a_{ik}^-$)),

$$\begin{aligned}
\tilde{a}_{ii} &= a_{ii} + \sum_{k\in E_i^w} a_{ik} + \sum_{k\in F_i^s} a_{ik} \frac{\bar{a}_{ki}}{\sum_{l\in\hat{C}_i\cup\{i\}} \bar{a}_{kl}} \\
&= b_i - \sum_{k\in S_i} a_{ik} + \sum_{k\in F_i^s} a_{ik}^- \frac{a_{ki}^-}{\sum_{l\in\hat{C}_i\cup\{i\}} a_{kl}^-} \\
&= b_i - \sum_{k\in C_i} a_{ik} - \sum_{k\in F_i^s} a_{ik}^+ - \sum_{k\in F_i^s} a_{ik}^- \left(1 - \frac{a_{ki}^-}{\sum_{l\in\hat{C}_i\cup\{i\}} a_{kl}^-}\right)
\end{aligned}$$

We then estimate, using (2.60) and $a_{ki}^- \ge \sum_{v\in\hat{F}_{kj}} a_{kv}^- \ge \zeta \sum_{l\in\hat{C}_i} a_{kl}^-$,

$$1 - \frac{a_{ki}^-}{\sum_{l\in\hat{C}_i\cup\{i\}} a_{kl}^-} = \frac{\sum_{l\in\hat{C}_i} a_{kl}^-}{\sum_{l\in\hat{C}_i\cup\{i\}} a_{kl}^-} \ge \frac{\sum_{l\in\hat{C}_i} a_{kl}^-}{\sum_{l\in\hat{C}_i} a_{kl}^- + \sum_{v\in\hat{F}_{kj}} a_{kv}^-} \ge \frac{1}{1+\zeta}.$$

We can now continue,

$$\begin{aligned}
& b_i - \sum_{k\in C_i} a_{ik} - \sum_{k\in F_i^s} a_{ik}^+ - \sum_{k\in F_i^s} a_{ik}^- \left(1 - \frac{a_{ki}^-}{\sum_{l\in\hat{C}_i\cup\{i\}} a_{kl}^-}\right) \\
\ge\ & b_i - \sum_{k\in C_i} a_{ik} - \sum_{k\in F_i^s} a_{ik}^+ - \frac{1}{1+\zeta} \sum_{k\in F_i^s} a_{ik}^- \\
\overset{b_i - \sum_{k\in C_i} a_{ik} - \sum_{k\in F_i^s} a_{ik}^+ \ge 0}{\ge}\ & \frac{1}{1+\zeta}\left(b_i - \sum_{k\in S_i} a_{ik}\right) \ge \frac{1}{\tau(1+\zeta)}\left(b_i - \sum_{k\in E_i} a_{ik}\right) \\
=\ & \frac{1}{\tau(1+\zeta)} a_{ii}.
\end{aligned}$$

Hence, we can prove (2.42) as before,

$$\begin{aligned}
\sum_{i\in F} a_{ii}\left(1 - \sum_{j\in\hat{C}_i} w_{ij}\right) e_i^2 &= \sum_{i\in F} \frac{a_{ii}}{\tilde{a}_{ii}} \left(a_{ii} + \sum_{k\in E_i^w} a_{ik} + \sum_{j\in C_i} a_{ij} + \sum_{k\in F_i^s} a_{ik}\right) e_i^2 \\
&\le \tau(1+\zeta) \sum_i \left(\sum_j a_{ij}\right) e_i^2.
\end{aligned}$$

Inequality (2.41) can be shown using the same techniques as for Theorems 2.10 and 2.11.

2.8.7. Multi-Pass Interpolation

We now consider an interpolation scheme that is especially used in conjunction with aggressive coarsening, i.e. in situations where not every fine grid point $i \in F$ is strongly

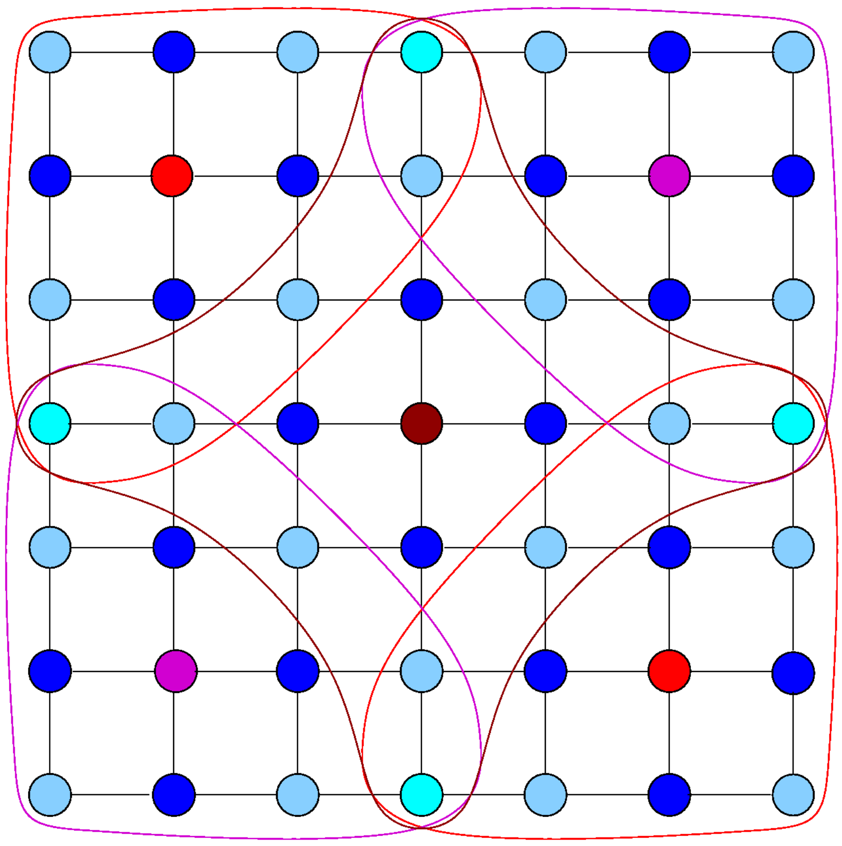


Figure 2.12.: Multi-pass interpolation for a 5-point stencil. The red, magenta and brown points denote the coarse grid points, the blue points denote the fine grid points. The value at the dark blue points is interpolated directly, all lighter blue points are interpolated indirectly. The red, magenta and brown lines indicate the interpolation influence of each coarse grid point.

algorithm 2.7 Multi-pass interpolation ([Stü99], Section 7.2.2)

1. For all $i \in F$ with $C_i \neq \emptyset$, apply direct interpolation (2.45-2.46) to obtain the interpolation weights w_{ij}, $j \in I_i := C_i$ and define F^* as the set of all these points i. If $F^* = F$, then stop.

2. For all $i \in F \setminus F^*$ for which $S_i \cap F^*$ is not empty, identify all $k \in F^*$ that belong to S_i. Then, for each such k, replace in (2.39),

$$e_k \leftarrow \sum_{j \in I_k} w_{kj} e_j$$

and set $I_i = \bigcup_{k \in S_i \cap F^*} I_k$. Then, we construct an interpolation formula for these i similar to the case of standard interpolation. After the interpolation has been constructed for all $i \in F \subset F^*$, $S_i \cap F^* \neq \emptyset$, we add these points to F^*.

3. If $F^* = F$, we stop. Otherwise we continue the iteration with the previous step.

coupled to a coarse grid point $j \in C$. In this case, we first use direct interpolation for all points $k \in F$ that have a strong connection to a coarse grid point $j \in C$. Then, such a point k can be used to derive interpolation formulas for fine grid points $i \in F$ such that $k \in S_i$. This process is continued until we have constructed the interpolation for each fine grid point, see Algorithm 2.7.
This interpolation technique ensures that we have a prolongation for each fine grid point as long as there is a (arbitrarily long) strong path from each fine grid point to a coarse grid point. However, depending on the layout of the coarse grid the interpolation for some fine grid points i may only be piecewise constant. For example, consider the situation in Figure 2.12.

2.8.8. Jacobi Interpolation

In contrast to the previously described interpolation methods, the *Jacobi interpolation* scheme [Stü99] does not build a prolongation operator from scratch, but can be used to improve an existing one. To this end, one or more (partial) Jacobi steps are applied to the interpolation matrix.
Let $\mu \geq 1$. Given an interpolation operator $P^{\mu-1}$ with interpolation weights $w_{ik}^{\mu-1}$ we obtain P^{μ} by relaxing the rows that belong to a fine grid point i. The modified interpolation weights w_{ij}^{μ} $(i \in F, j \in C)$ read as follows,

$$w_{ij}^{\mu} = w_{ij}^{\mu-1} - a_{ii}^{-1} \left(\sum_{k \in F} w_{kj}^{\mu-1} + a_{ij} \right),$$

or, written in terms of the block matrices as defined in Subsection 2.8.1,

$$P^\mu = \begin{pmatrix} I_{CC} \\ P^\mu_{FC} \end{pmatrix}, \; P^\mu_{FC} = P^{\mu-1}_{FC} - D^{-1}_{FF}\left(A_{FF}P^{\mu-1}_{FC} + A_{FC}\right). \tag{2.61}$$

It is easy to see that if $P^{\mu-1}$ satisfies the interpolation property (2.37), then so does P^μ.

Theorem 2.14 *Let A be a positive definite, weakly diagonally dominant matrix and let $P^{\mu-1}$ satisfy the interpolation approximation property* (2.37) *with constant $\tau = \tilde{\tau}$. Then the Jacobi interpolation* (2.61) *satisfies* (2.37) *with $\tau \leq 7\tilde{\tau} + 4\sqrt{2\tilde{\tau}} + 2$.*

Proof: We consider the left hand side of (2.37) and use (2.61),

$$\begin{aligned}\|e_F - P^\mu_{FC}e_C\|^2_{0,F} =& \left(e_F - P^{\mu-1}_{FC}e_C\right)^T D_{FF}\left(e_F - P^{\mu-1}_{FC}e_C\right) \\ &+ \left(A_{FF}P^{\mu-1}_{FC}e_C + A_{FC}e_C\right)^T D^{-1}_{FF}\left(A_{FF}P^{\mu-1}_{FC}e_C + A_{FC}e_C\right) \\ &+ 2\left(e_F - P^{\mu-1}_{FC}e_C\right)^T\left(A_{FF}P^{\mu-1}_{FC}e_C + A_{FC}e_C\right) \qquad (2.62)\\ \leq& \|e_F - P^{\mu-1}_{FC}e_C\|^2_{0,F} + \|A_{FF}P^{\mu-1}_{FC}e_C + A_{FC}e_C\|^2_{D^{-1}_{FF}} \\ &+ 2\|e_F - P^{\mu-1}_{FC}e_C\|_{0,F}\|A_{FF}P^{\mu-1}_{FC}e_C + A_{FC}e_C\|_{D^{-1}_{FF}} \qquad (2.63)\end{aligned}$$

By induction we have (2.37) for all $\mu \geq 1$,

$$\|e_F - P^{\mu-1}_{FC}e_C\|^2_{0,F} \leq \tilde{\tau}\|e\|^2_1.$$

On the other hand, we can split

$$\|A_{FF}P^{\mu-1}_{FC}e_C + A_{FC}e_C\|_{D^{-1}_{FF}} \leq \|A_{FF}\left(P^{\mu-1}_{FC}e_C - e_F\right)\|_{D^{-1}_{FF}} + \|A_{FF}e_F + A_{FC}e_C\|_{D^{-1}_{FF}}. \tag{2.64}$$

We have (see also (2.25)),

$$u_F^T A_{FF}D^{-1}_{FF}A_{FF}u \leq \rho(D^{-1}_{FF}A_{FF})^2 u^T D_{FF}u,$$

hence we can estimate the first summand of (2.64),

$$\|A_{FF}(P^{\mu-1}_{FC}e_C - e_F)\|_{D^{-1}_{FF}} \leq \rho(D^{-1}_{FF}A_{FF})\|e_F - P^{\mu-1}_{FC}e_C\|_{0,F} \leq \sqrt{\tilde{\tau}}\rho(D^{-1}_{FF}A_{FF})\|e\|_1.$$

Regarding the second summand, we note that

$$\begin{aligned}\|A_{FF}e_F + A_{FC}e_C\|^2_{D^{-1}_{FF}} &= \begin{pmatrix} A_{FF}e_F + A_{FC}e_C \\ A_{CF}e_F + A_{CC}e_C \end{pmatrix}^T \begin{pmatrix} D^{-1}_{FF} & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} A_{FF}e_F + A_{FC}e_C \\ A_{CF}e_F + A_{CC}e_C \end{pmatrix} \\ &\leq \begin{pmatrix} A_{FF}e_F + A_{FC}e_C \\ A_{CF}e_F + A_{CC}e_C \end{pmatrix}^T \begin{pmatrix} D^{-1}_{FF} & 0 \\ 0 & D^{-1}_{CC} \end{pmatrix} \begin{pmatrix} A_{FF}e_F + A_{FC}e_C \\ A_{CF}e_F + A_{CC}e_C \end{pmatrix} \\ &= \|e\|^2_2 \\ &\leq \rho(D^{-1}A)\|e\|^2_1.\end{aligned}$$

For weakly diagonally dominant matrices, we have $\rho(D_{FF}^{-1}A_{FF}) \le \rho(D^{-1}A) \le 2$. Hence we obtain for (2.64),

$$\|A_{FF}P_{FC}^{\mu-1}e_C + A_{FC}e_C\|_{D_{FF}^{-1}} \le (2\sqrt{\tilde{\tau}} + \sqrt{2})\|e\|_1. \tag{2.65}$$

We combine (2.63) and (2.65) to obtain the result,

$$\|e_F - P_{FC}^{\mu}e_C\|_{0,F}^2 \le \left(\tilde{\tau} + (2\sqrt{\tilde{\tau}} + \sqrt{2})^2 + 2\sqrt{\tilde{\tau}}(2\sqrt{\tilde{\tau}} + \sqrt{2})\right)\|e\|_1^2.$$

2.8.9. Truncation of Interpolation

The long range interpolation methods described in this section share one major disadvantage: They can substantially enlarge the set of interpolatory points I_i $(i \in F)$ and hence the number of non-zero elements per matrix row for the interpolatory matrix P. This also affects the coarse grid operator A^C, which is computed as $A^C = P^TAP$. In consequence, the matrices A_l become less and less sparse as the level index l becomes larger.
However, interpolation weights w_{ij} that lead to points j "far away" from i (i.e. there is no direct connection $a_{ij} \neq 0$) may be much smaller than the weights for nearby points k. This is especially true if there is only a single path $i \leftarrow v \leftarrow j$, $a_{iv} \neq 0$, $a_{vk} \neq 0$. In this case, the influence of the error e_j on e_i is of less relevance than the influence of e_k where w_{ik} is large. Hence, we can drop these small interpolation weights, however, we need to take care that the row sums of all interpolatory weights remain unchanged ([Stü99], Section 7.2.4).
To compute the new interpolatory sets $\hat{I}_i = \hat{I}_i^+ \cup \hat{I}_i^-$ and weights $\hat{w}_{ij}$, we have again to distinguish between positive and negative weights to prevent divisions by (near-) zero,

$$\hat{I}_i^+ = \{j \in I_i^+ : \ w_{ij} \ge \epsilon \max_{k \in I_i^+}, w_{ik}\} \ \hat{I}_i^- = \{j \in I_i^- : \ w_{ij} \le \epsilon \min_{k \in I_i^-} w_{ik}\}$$

$$\hat{w}_{ij} = \begin{cases} \frac{\sum_{k \in I^+} w_{ik}}{\sum_{k \in \hat{I}^+} w_{ik}} w_{ij} \text{ if } w_{ij} \ge \epsilon \max_{k \in I_i^+} w_{ik} \\ \frac{\sum_{k \in I^-} w_{ik}}{\sum_{k \in \hat{I}^-} w_{ik}} w_{ij} \text{ if } w_{ij} \le \epsilon \min_{k \in I_i^-} w_{ik} \\ 0 \text{ else.} \end{cases}$$

where ϵ is an user-defined parameter, typically $\epsilon = 0.2$.

Remark 2.4 Another approach is to limit the number of non-zero elements per interpolation matrix row and only to take the largest (absolute) values.

2.9. The coarse grid operator

The remaining task in the setup phase on level l is to construct the operator A_{l+1} for the next level $l+1$. Unlike the geometric case, as discussed in section 2.2, we cannot obtain A_{l+1} from a discretization on the underlying grid Ω^{l+1}, as the latter is just an

index set without any geometric information attached to it.
Instead, as already indicated in section 2.8.1, we use the *Galerkin ansatz* for the coarse grid operator,

$$A_{l+1} = R_l A_l P_l, \tag{2.66}$$

where the restriction R_l is obtained as the transpose of the interpolation,

$$R_l = P_l^T. \tag{2.67}$$

In Section 2.8.1 we employed a special smoother and a special construction of one of the transfer operators to obtain a direct solver with just two levels. This solver required the computation of A_{FF}^{-1}, i.e. the part of A_l that corresponds to the fine grid points $i \in F^l$. In general, this is not feasible. We can however show that for *any* full-rank interpolation operator P_l, the Galerkin product (2.66) is the "optimal" choice for the coarse grid operator in the sense that the two-grid correction

$$e^{new} = T_l e^{old} = \left[I - P_l \left(A_{l+1}\right)^{-1} R_l A_l\right] e^{old}$$

minimizes the error within the range of P_l. To show this, first note that T_l is an orthogonal projector w.r.t the inner product $(\cdot,\cdot)_1$, i.e.

$$T_l^2 = T_l \text{ and } (T_l\cdot,\cdot)_1 = (\cdot,\cdot T_l)_1$$

which can be easily seen from a straightforward calculation. The following theorem then gives us the desired result.

Theorem 2.15 *([Stü99], Theorem 2.2) Let $(\cdot,\cdot)$ be any inner product with corresponding norm $\|\cdot\|$and let the matrix T be symmetric w.r.t. $(\cdot,\cdot)$ and $T^2 = T$. Then we have*

- *$rng(T) \perp rng(I - T)$*
- *For $u \in rng(T)$ and $v \in rng(I - T)$ we have $\|u + v\|^2 = \|u\|^2 + \|v\|^2$*
- *$\|T\| = 1$*
- *For all u: $\|Tu\| = \min_{v \in rng(I-T)} \|u - v\|$.*

We note that $rng(I-T_l) = rng(P_l)$ and obtain that, in our setting, the last line translates to

$$\|T_l e_l\|_1 = \min_{\bar{e}_{l+1}} \|e_l - P_l \bar{e}_{l+1}\|,$$

i.e. the coarse grid correction minimizes the energy norm of the error over all possible corrections within the range of the interpolation operator P_l. This is called the *variational principle.*
The next lemma helps us to extend the convergence proof from the two-grid setting to complete V-cycles.

Lemma 2.3 *([Stü99], Lemma 2.2) Let the exact coarse-grid correction*

$$e_{l+1} = A_{l+1}^{-1} R^l A_l e_l$$

be replaced by any approximate coarse-grid correction $\tilde{e}_{l+1}$ satisfying

$$\|e_{l+1} - \tilde{e}_{l+1}\|_1 \leq \|e_{l+1}\|_1$$

where $\|\cdot\|_1$ is taken w.r.t A_{l+1}. Then the approximate two-grid correction $\tilde{T}^l$ still satisfies

$$\|\tilde{T}_l\|_1 \leq 1.$$

With this lemma, we see that a V-cycle never diverges as long as the smoothers on all levels l satisfies $\|M_l\|_1 \leq 1$. This is always the case if M_l satisfies the smoothing property (2.27),

$$\|M_l e\|_1^2 \leq \|e\|_1^2 - \sigma \|e\|_2^2$$

which is satisfied for most relevant applications if we use a damped Jacobi or Gauss-Seidel smoother, as previously shown in Section 2.4.

Remark 2.5 To show the variational principle, the only condition on the interpolation is that it has full (column) rank. While it is clear that a truncation of the interpolation operator as described in section 2.8.9 does not change the rank and hence we still obtain a convergent method, this is not guaranteed if we truncate the Galerkin operator. However, recent developments (e.g. *collocation coarse approximation* [WY09]) indicate that it still may be useful to sparsify the coarse grid matrix, i.e. to use a non-Galerkin coarse grid operator. There are two main reasons for this. First, the number of nonzero entries per matrix row tends to grow as l increases, i.e. more and more direct connections between previously distant nodes are introduced. Hence, the matrices on the coarser levels cannot be considered "sparse" any more, while they are still too large to be efficiently treated by direct solvers and a single step of an iterative solver requires more than a linear amount of computations. This issue is even more severe on parallel computers, where a certain part of the non-zero matrix entries introduces couplings to off-processor nodes. In consequence, to carry out a matrix-vector (or matrix-matrix) multiplication, a data transfer is needed, which is relatively expensive compared to on-processor computations.

2.10. Two-grid convergence: The non-symmetric case

In this section, we demonstrate how to show the convergence of a two-grid algebraic multilevel hierarchy in the case of a non-symmetric fine grid operator A. In this case, it is not required that the restriction operator R equals the transpose of the interpolation operator P. For example, it might be useful to first derive an interpolation operator $\hat{P}$ based on the transpose of A, A^T and then set the restriction as the transpose of $\hat{P}$, i.e. $R = \hat{P}^T$. For details on the construction of AMG hierarchies for non-symmetric systems, we refer to [MO08].

The convergence proof described here is not only useful for non-symmetric systems. As we will show in Section 4.8, it also forms the core of the two-grid convergence proof for one variant of (symmetric) saddle point system AMG.
The convergence theory described in this section has been introduced in [Not10]. It is based on the symmetric two-grid AMG convergence theory form [FV04]. In this section, we cite the main result as well as a corollary that will be important for our saddle point AMG.

Theorem 2.16 *([Not10], Theorem 2.1) Let A be a non-singular $N \times N$ matrix. Let P be a $N \times N_C$ matrix of rank N_C and let R be a $N_C \times N$ matrix of rank N_C, such that $A_C = RAP$ is non-singular. Let M_1, M_2 be $N \times N$ matrices and ν_1, ν_2 be non-negative integers, such that $(I - M_1 A)^{\nu_1} (I - M_2 A)^{\nu_2} - I$ is non-singular. Let Q be the matrix such that*

$$I - Q^{-1}A = (I - M_1 A)^{\nu_1} (I - M_2 A)^{\nu_2}$$

and assume that

$$Q_C = RQP$$

is non-singular. The matrix

$$A^{-1} Q \left(I - \pi_Q\right) \tag{2.68}$$

where

$$\pi_Q = P Q_C^{-1} R Q$$

has eigenvalue 0 with multiplicity N_C and $N - N_C$ nonzero eigenvalues. Letting $\mu_1, \ldots, \mu_{N-N_C}$ be these nonzero eigenvalues, the following propositions hold:

1. *The iteration matrix*

$$T = (I - M_2 A)^{\nu_2} \left(I - P A_C^{-1} R A\right) (I - M_1 A)^{\nu_1} \tag{2.69}$$

   has eigenvalues $1 - \nu_1^{-1}, \ldots, 1 - \nu_{N-N_C}^{-1}$, plus N_C times the eigenvalue 0.

2. *For any $(N - N_C) \times N$ matrix Z and any $N \times (N - N_C)$ matrix S such that the matrices*

$$\begin{pmatrix} R \\ Z \end{pmatrix} \text{ and } \begin{pmatrix} P & S \end{pmatrix}$$

   are non-singular, $(ZAS) - (ZAP) A_C^{-1} (RAS)$ is non-singular and the matrix

$$\left((ZAS) - (ZAP) A_C^{-1} (RAS)\right)^{-1} ZX \left(I - \pi_Q\right) S \tag{2.70}$$

   has eigenvalues $\mu_1, \ldots, \mu_{N-N_C}$.

3. *For any $N \times N_C$ matrix $\tilde{P}$ and for any $N_X \times N$ matrix $\tilde{R}$. the matrix*

$$A^{-1} \left(I - \tilde{P} R\right) Q \left(I - \pi_Q\right) \left(I - P \tilde{R}\right) \tag{2.71}$$

   has the same eigenvalues as (2.68), that is, $\mu_1, \ldots, \mu_{N-N_C}$, plus N_C times the eigenvalue 0.

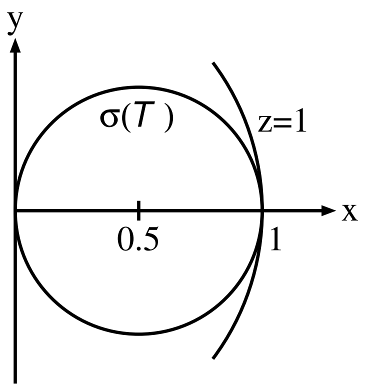


Figure 2.13.: Eigenvalue estimate of the two-grid error propagation matrix T as defined in Corollary 2.1.

4. *The matrices*

$$(I - \pi_A)\, Q^{-1} A \tag{2.72}$$

$$Q^{-1} A\, (I - \pi_A) \tag{2.73}$$

$$(I - \pi_A)\, Q^{-1} A\, (I - \pi_A) \tag{2.74}$$

where

$$\pi_A = P A_C^{-1} R A$$

have eigenvalues $\mu_1^{-1}, \ldots, \mu_{N-N_C}^{-1}$, *plus* N_C *times the eigenvalue* 0.

5. *For all* μ_i, $i = 1, \ldots, N_{N-N_C}$, *there exist some* $z_i \in \mathbb{C}^N$ *such that* $R z_i = 0$ *and*

$$V^H A^{-1} z_i = \mu_i v^H Q^{-1} z_i \text{ for all } v \in \mathbb{C}^N : \; P^H V = 0. \tag{2.75}$$

6. *If, in addition,* $R = P^H$, *and there is no* $v \in \mathbb{C}^N$ *such that* $P^H v = 0$ *and* $v^H A^{-1} v = v^H Q^{-1} v = 0$, *then, for* $1 = 1, \ldots, N_{N-N_C}$

$$\mu_i \in \left\{ \frac{v^H A^{-1} v}{v^H Q^{-1} v} : \; v \in \mathbb{Q}^N \setminus \{0\}, \; P^H v = 0, v^H A^{-1} v \neq 0 \text{ and } v^H Q^{-1} v \neq 0 \right\}. \tag{2.76}$$

We now look at the special case $R = P^H$, i.e. the classical choice for the restriction operator as the (conjugate) transpose of the interpolation operator. In this case, we can derive more specific bounds on the eigenvalues of the error propagation T.

Corollary 2.1 *([Not10], Corollary 2.1) If, in addition to the assumptions of Theorem 2.16,* $R = P^H$ *and there is no* $v \in \mathbb{C}^N$ *such that* $v^H A^{-1} v = v^H Q^{-1} v = 0$, *then,*

$$\sigma(T) \subset \left\{ 1 - \frac{v^H Q^{-1} v}{v^H A^{-1} v} : \; v \in \mathbb{C}^N \setminus \{0\}, v^H A^{-1} v \neq 0 \text{ and } v^H Q^{-1} v \neq 0 \right\} \cup \{0\}. \tag{2.77}$$

In particular, if A, Q are Hermitian positive definite, then the μ_i are real positive

$$\sigma(T) \subset \left[1 - \lambda_{\max}\left(Q^{-1}A\right), 1 - \lambda_{\min}\left(Q^{-1}A\right)\right] \cup \{0\} \tag{2.78}$$

and

$$\rho(T) \leq \max\left(\lambda_{\max}\left(Q^{-1}A\right) - 1, 1 - \frac{1}{\max_i \mu_i}\right). \tag{2.79}$$

On the other hand, if Q is Hermitian positive and if

$$\|\alpha I - Q^{-1}A\|_Q \leq \alpha \tag{2.80}$$

for some positive α, then

$$\lambda \in \sigma(T) \Rightarrow |1 - \alpha - \lambda| \leq \alpha. \tag{2.81}$$

From statement (2.81) we see that the two-grid method T converges if $\alpha \leq \frac{1}{2}$. In this case, all eigenvalues of T lie within a disc of radius $\frac{1}{2}$ around the point $\left(\frac{1}{2}, 0\right) \in \mathbb{C}$, see Figure 2.13, where $\lambda = 1$ is not possible as this would imply $v^H Q^{-1} v = 0$, see (2.77). Note that inequality (2.80) is equivalent to

$$\|I - \frac{1}{\alpha} Q^{-1}A\|_Q \leq 1$$

i.e. to obtain $\alpha = \frac{1}{2}$ we require that the "over-relaxed" smoother $I - 2Q^{-1}A$ still converges,

$$\|I - 2Q^{-1}A\|_Q \leq 1.$$

In other cases, an appropriate damping of the smoother would be required.

2.11. AMG: Not just point coarsening

We now summarize three AMG variants that do not follow the classical Ruge-Stüben setup outlined throughout this chapter. First, we introduce element *agglomeration AMG (AMGe)*, a geometric-algebraic approach for finite element discretizations that employs information from the mesh in addition to the matrix entries. *Smoothed aggregation* groups together adjacent fine grid points into one coarse node (i.e. we do not partition into "coarse" and "fine" grid points here). Then, from a tentative interpolation operator (e.g. a constant value per group), the final interpolation matrix is derived by an additional smoothing process applied to the tentative prolongation. Finally, the *Bootstrap AMG* approach aims to identify the smooth (i.e. slow-to-converge) error components by analyzing the smoothing action and then construct the interpolation operator to be well-fitted to these errors.

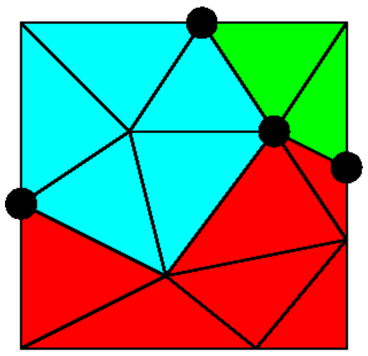


Figure 2.14.: Agglomerated finite element mesh. Each color indicates an aggregate. The coarse vertices are indicated by the black dots.

algorithm 2.8 Element agglomeration AMGe ([JV01], Algorithm 4.1)

```
• initiate. Set $w(\gamma) \leftarrow 0$ for all faces $\gamma$.
• global search. Find a face $\gamma$ with maximal $w(\gamma)$; set $E \leftarrow \emptyset$;
    1. Set $E \leftarrow E \cup e_1 \cup e_2$, where $\gamma = e_1 \cup e_2$, and set $w_{\max} \leftarrow w(\gamma)$, $w(\gamma) \leftarrow -1$;
    2. Increment $w(\gamma_1) \leftarrow w(\gamma_1) + 1$ for all faces $\gamma_1$ such that $w(\gamma_1) \neq -1$ and $\gamma_1$ is a neighbor of $\gamma$;
    3. Increment $w(\gamma_2) \leftarrow w(\gamma_2) + 1$ for all faces $\gamma_2$ such that $w(\gamma_2) \neq -1$, $\gamma_2$ is a neighbor of $\gamma$, and $\gamma_2$ and $\gamma$ are the faces of a common element;
    4. From the neighbors of $\gamma$, choose a face $\gamma^*$ with a maximal $w(\gamma^*)$; if $w(\gamma^*) \geq w_{\max}$, set $\gamma = \gamma^*$, and go to step 1;
    5. If all neighbors of $\gamma$ have smaller weight than $w_{\max}$, the agglomerated element $E$ is complete; set $w(\gamma) = -1$ for all faces of the elements $e$ contained in $E$; go to global search;
```

2.11.1. AMGe

Element-based AMG (AMGe) is an approach to AMG especially for operators A that arise from finite element discretizations but do not have M-matrix properties. The convergence theory as well as the construction of the interpolation operators was introduced by Brezina et. al. in [BCF+00], while the coarsening algorithm (here called *element agglomeration*) was described in [JV01].
In the following, we assume that, in addition to the fine grid matrix A, we are given

- a set of fine grid degrees of freedom (dofs) $\mathcal{D}_F = \{i\}$,
- a set of fine grid elements $\mathcal{E}_F = \{e\}$, where each element is given as a set of dofs $e = \{i_j\}_{j=1}^{n_e}$.
- the element stiffness matrices $\{A_e\}_{e \in \mathcal{E}_f}$.

In Algorithm 2.8 we give the agglomeration process for the two-dimensional case. Note that this procedure only depends on the finite element geometry and does not take into account the matrix A. Hence, it is not possible to reflect anisotropies or coefficient jumps inside the operator. To overcome this problem, an additional measure $\alpha(\gamma)$ can be assigned to each face. Then, a threshold can be defined to forbid agglomerating two elements if the weight of their common face is too high, see [JV01], Section 4 for details.
After the agglomeration process, we need to identify the coarse faces and the coarse vertices [JV01].

- Any maximal intersection $E_i \cap E_j$ for different coarse elements $E_i \neq E_j$ is called a coarse face F. In addition, the coarse boundary faces are given by the (maximal) intersections $E_i \cap \Gamma$. The set of all faces is denoted by $\mathcal{F}_C$.
- Consider each coarse face $F \in \mathcal{F}$ as a set of dofs $\{i\}$. For each i, compute the intersection $\bigcap_{i \in \mathcal{F}_C} F$. The minimal nonempty intersections form the coarse vertices $\mathcal{V} = \{V\}$.

In the simplest case (scalar PDE), each coarse vertex consists of just a single degree of freedom.
As in the Ruge-Stüben case, we let the interpolation at the coarse vertices $V \in \mathcal{V}$ be given by the identity, such that the interpolation operator takes the form

$$P = \begin{pmatrix} P_{FC} \\ I_{CC} \end{pmatrix}.$$

The entries of P_{FC} are computed recursively. To this end, define for each fine grid vertex i,

- a neighborhood $\Omega(i) = \bigcup_{i \in E} E$ of all agglomerated elements that contain i;
- a minimal set $N(i) = \bigcap_{i \in E} E$ of all agglomerated elements that contain i.

The sets $N(i)$ are, by definition, non-empty. In addition, we define the boundary of each $\partial N(i)$,

- if $N(i)$ is an element E, then $\partial N(i)$ is the union of all faces of E;
- if $N(i)$ is a face, then $\partial N(i)$ consists of all vertices that belong to more than one face;
- in all other cases $\partial N(i)$ consists of all vertices in $N(i)$.

Now, for any minimal set $N(i)$ for which we know the interpolated values at its boundary $\partial N(i)$, we can compute the prolongation for all dofs $j \in N(i)$ (e.g. for a face F we compute the interpolation for its dofs from the interpolation weights at its boundary vertices). We consider the corresponding element stiffness matrices of all (fine mesh) elements $e \subset \Omega(i)$ contained in the neighborhood $\Omega(i)$ and assemble the matrix $A_{\Omega(i)}$,

$$A_{\Omega(i)} = \sum_{e \subset \Omega(i)} A_e = \begin{pmatrix} A_{ii} & A_{ib} \\ A_{bi} & A_{bb} \end{pmatrix} \begin{array}{l} \}\Omega(i) \setminus \partial N(i) \\ \}\partial N(i) \end{array} ,$$

where we have rearranged $A_{\Omega(i)}$ with respect to the boundary $\partial N(i)$ and the remaining points $\Omega(i) \setminus \partial N(i)$. Now, assume that the interpolation $P_{\partial N(i)}$ at the boundary $\partial N(i)$ is already computed, we construct the interpolation for all $j \in N(i)$ from

$$A_{ii} e_j + A_{ib} P_{\partial N(i)} e_c = 0$$

and obtain, for all $j \in N(i) \setminus \partial N(i)$, $k \in \partial N(i)$) the weights (cf. (2.33)) [JV01]

$$w_{jk} = -\left(A_{ii}^{-1} A_{ib} P_{\partial N(i)}\right)\Big|_j .$$

To this end, A_{ii} must be invertible. This is always the case if the basis vectors for the null-space of $A_{\Omega(i)}$ are linear independent if restricted to $\partial N(i)$ ([JV01], Lemma 2.2). In other words, the boundary $\partial N(i)$ must be "large enough" to contain the kernel components of $A_{\Omega(i)}$. We can also interpret this restriction as the requirement that (near-) zero energy components must be contained within the range of the prolongation. For a detailed analysis of energy minimization properties for smoothed aggregation interpolation operators, we refer to [BCF+00] and [JV01].
A generalization of AMGe interpolation that does not require access to the element stiffness matrices is called *element-free AMGe* [HV01].

2.11.2. Smoothed Aggregation

While in classical Ruge-Stüben AMG the coarse mesh is constructed by selecting a subset $\Omega^{l+1} = C^l \subset \Omega^l$, in *smoothed aggregation (SA)* multigrid [VMB94] [VMB96] there are no designated "coarse" or "fine" points. Instead, we group neighboring vertices $i \in \Omega^l$ into N_C disjoint aggregates $\Omega^l = \Omega^l_1 \dot\cup \ldots \dot\cup \Omega^l_j$, where the size of each subset Ω^l_k is much smaller than the grid Ω^l_l, $|\Omega^l_k| \ll |\Omega^l|$. Now, each aggregate Ω^l_k is represented by a single

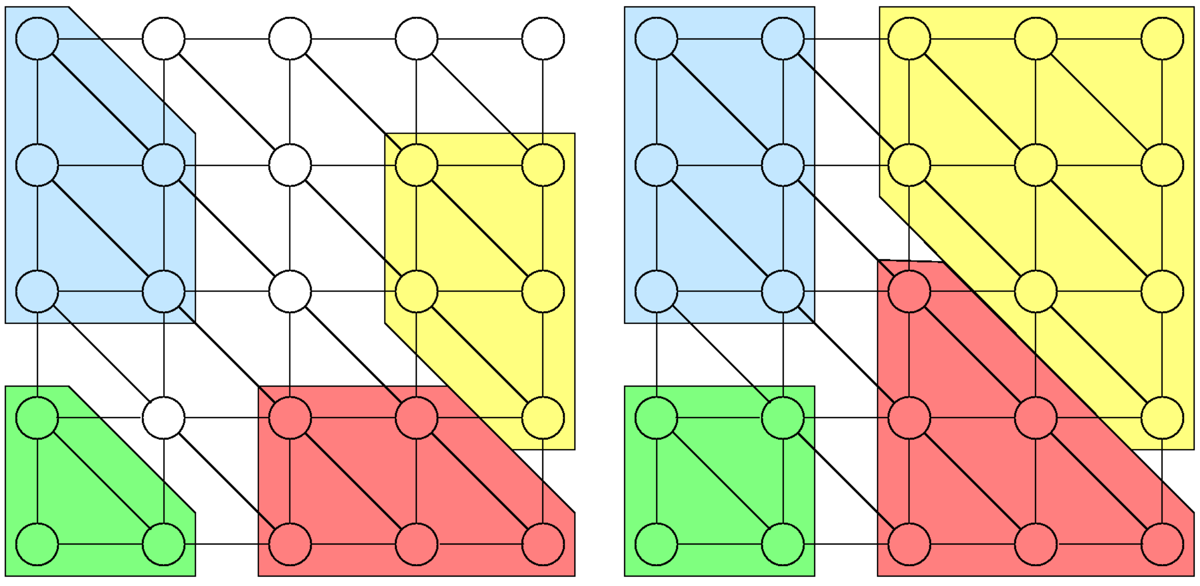

(a) First pass (b) Second pass

Figure 2.15.: Aggregation procedure for the finite difference mesh from Example 2.2. Each colored area shows an aggregate.

algorithm 2.9 Aggregation algorithm ([VMB94], Algorithm 2)

```
begin
  initialize $U \leftarrow \Omega$; $j \leftarrow 0$;
  while $\exists\ S_i \subset U$ do                                   find disjoint aggregates
        $j \leftarrow j+1$;
        $C_j \leftarrow S_i$;
        $U \leftarrow U \setminus C_j$;
  od
  for $k = 1, \dots, j$ do $\tilde{C}_k \leftarrow C_k$;
     while $\exists\ i \in U$ and $1 \leq k \leq j$ such that $S_i \cap \tilde{C}_k \neq \emptyset$ do     add points to aggregates
           $C_k \leftarrow C_k \cup \{i\}$;
           $U \leftarrow U \setminus \{i\}$;
     od
  od
  while $U \neq \emptyset$ do                                 parition remaining points into aggregates
        $j \leftarrow j+1$;
        $C_j \leftarrow S_i \cap U$;
        $U \leftarrow U \setminus C_j$;
  od
end
```

algorithm 2.10 Tentative interpolation algorithm ([VBM01], Algorithm 4.1)

For the given system of aggregates $\{\Omega_j^l\}_{j=1}^{N_l}$ and the $n_l \times r$ matrix B_l satisfying $\tilde{P}_l^1 B_l = B_1$, we create a prolongator $\tilde{P}_l$, a matrix B_{l+1} satisfying (2.83) and supernodes on level $l+1$ as follows,

1. Let n_j^l denote the number of degrees of freedom associated with aggregate Ω_j^l. Partition the $n^l \times r$ matrix B_l into blocks B_j^l of size $n_j^l \times r$, $j = 1, \dots, N^l$, each corresponding to the set of (fine) degrees of freedom on an aggregate Ω_j^l.
2. Decompose $B_j^l = Q_j^l R_j^l$, where $Q_j^l \in \mathbb{R}^{n_j^l \times r}$ is an orthogonal matrix, and $R_j^l \in \mathbb{R}^{r \times r}$ is an upper triangular matrix.
3. Create the tentative prolongator $\tilde{P}_l$ and the coarse "zero energy modes" B_{l+1} by

$$\tilde{P}_l = \begin{pmatrix} Q_1^l & 0 & \dots & 0 \\ 0 & Q_2^l & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & Q_{N^l}^l \end{pmatrix}, \quad B_{l+1} = \begin{pmatrix} R_1^l \\ R_2^l \\ \vdots \\ R_{N^l}^l \end{pmatrix}.$$

4. For each aggregate Ω_j^l, the coarsening gives rise to r degrees of freedom on the coarse level (the j-th block column of $\tilde{P}_l$). These degrees of freedom define the j-th coarse level supernode.

vertex $k \in \Omega^{l+1}$ on the next coarser level $l+1$.
The aggregation procedure is given in Algorithm 2.9. It builds aggregates of strongly connected points, where, in contrast to (2.29) we use a slightly different concept of strength,

$$S_i = \left\{ j : \; |a_{ij}| \geq \alpha \sqrt{a_{ii} a_{jj}} \right\} \cup \{i\}. \tag{2.82}$$

Note that here the strength matrix is symmetric if A is symmetric and that each set S_i also contains the point i itself. The algorithm itself is divided in three steps: First, we identify sets of strongly coupled points S_i that are completely undecided, i.e. no member of S_i is already part of a patch. Such a S_i then forms a new patch C_j. If no such set can be created, we proceed with the second step: We attach still undecided points $i \in U$ to a strongly connected patch C_k, if there exists one. During the third pass, any remaining points are grouped into additional aggregates. In Figure 2.15, we show the first and second pass of the aggregation algorithm applied to the finite difference discretization for the mixed problem from Example 2.2. Here, we are finished after the second pass and no undecided points remain.
As in the case of Ruge-Stüben AMG, the range of the interpolation should contain the low-energy (smooth) error components e, i.e. the $\|e\|_1 \approx 1$, see the discussion in Section 2.4. A particular property of smoothed aggregation is that these error components can be explicitly prescribed. To this end, we provide a matrix $B \in \mathbb{R}^{N_1 \times r}$. The r columns of B form a basis of the error space that we require to be exactly interpolated on all levels, i.e.

$$rngB \subset rng\tilde{P}_l^1 \text{ for all} l = 1, \ldots, L-1 \text{ where } \tilde{P}_l^1 = \tilde{P}_1 \cdot \tilde{P}_2 \cdot \ldots \cdot \tilde{P}_l.$$

$\tilde{P}_l^1$ is the concatenated tentative interpolation operator from level $l+1$ to level 1. We do not need to explicitly compute this matrix, instead we recursively define a version B_l on all levels l, (letting $B_1 = B$),

$$\tilde{P}_l B_{l+1} = B_l. \tag{2.83}$$

We construct the columns of B_{l+1} from B_l together with the tentative interpolation operator $\tilde{P}_l$, see Algorithm 2.10.
In the most simple case, which is for example applicable if A is the discretization of a second-order partial differential equation, we can choose $r = 1$ and a constant vector B,

$$B = \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}.$$

Hence we enforce that on each patch Ω_j^l constant vectors are interpolated exactly. However, sharp jumps can occur at the interface of two patches, such that the range of $\tilde{P}_l$ is not completely contained within the space of the low-energy modes of A_l.
We overcome this issue by an additional smoothing of the interpolation operator and define [VBM01],

$$P_l = \left(I - \frac{4}{3} \frac{1}{\lambda_l} M_l^{-1} A_l \right) \tilde{P}_l.$$

where $\lambda_l \geq \rho(M_l^{-1} A_l)$ and

$$M_l = \left(\tilde{P}_l\right)^T \tilde{P}_l,$$

Here, M_l ensures that the smoothed interpolation P^l only depends on the range of $\tilde{P}_l$, while the specific choice of the latter operator is irrelevant. The scaling factor $\frac{4}{3\lambda_l}$ minimizes the spectral radius $\rho(M_{l+1}^{-1} A_{l+1})$ for the next level $l+1$, see [VBM01] for a theoretical analysis.

2.11.3. Bootstrap AMG

algorithm 2.11 Compatible Relaxation [Liv04], [FV04]

```
initialize$U \leftarrow \Omega$;  $C \leftarrow \emptyset$;  $F \leftarrow \Omega$;
while $U \neq \emptyset$ do
    for $i \in C$ do $e_i^0 \leftarrow 0$;  od;
    for $i \in F$ do $e_i^0 \leftarrow rand(0,1)$;  od;
    for $j = 1$ to $\nu$ do
        $e^j \leftarrow M e^{j-1}$                                apply smoother
        for $i \in C$ do $e_i^\nu \leftarrow 0$;          leave coarse variables unchanged
        od;
        $U \leftarrow \{i \in F : \frac{|e_i^\nu|}{|e_i^{\nu-1}|} > \theta\}$;
        $C \leftarrow C \cup \{$independent set of $U\}$;  $F \leftarrow \Omega \setminus C$;
    od;
```

All previously described AMG techniques rely on certain assumptions on the system to solve, in particular that the relaxation scheme used in the multigrid cycle leaves behind smooth error components that vary slowly along the strong negative couplings. In practice, this applies to a wide class of second-order elliptic partial differential equations but may not hold for other interesting problems, e.g. the bi-harmonic operator Δ^2 or the gauge Laplacian, which is a component of the lattice Dirac operator used in quantum chromodynamics (QCD) computations. We refer to [Kah09] and [BBKL11] for a detailed discussion and only give a brief description. The gauge Laplacian is given by the stencil

$$A(\mathcal{U}) = \begin{pmatrix} 0 & -\overline{U_y^z} & 0 \\ -U_x^{z-e_x} & 4+m & \overline{U_y^z} \\ 0 & U_y^{z-e_y} & 0 \end{pmatrix},$$

where $\mathcal{U} = \{U_\mu^z \in U(1), \mu = x, y,\ z \in \Omega\}$, e_x and e_y denote the unit vectors in x and y direction, respectively, and m is a constant chosen such that the resulting matrix is positive definite. If $U = 1$ across the computational domain and $m = 0$, the gauge Laplacian equals a scaled discrete Laplacian, while for a stochastic distribution of U, the gauge variables U_μ^z are not necessarily correlated and the support of low energy eigenmodes of A becomes small, i.e., it is not obvious how to approximate these eigenvectors on a

coarser mesh.
To generalize AMG for a wider class of problems, several approaches have been suggested to both improve the coarsening process as well as the construction of the interpolation operator. The key idea to the construction of the coarse grid is, given a C/F-splitting, to apply the smoother to the homogeneous equation $Ae = 0$ while keeping the variables $e_i = 0$ at the coarse grid points $i \in C$ constant. Then, if this smoothing process does not converge at acceptable speed, a subset of the fine grid points $I \subset F$ is transferred to the coarse grid. We repeat this process until we obtain a desired smoothing rate. This process is called *compatible relaxation*, see Algorithm 2.11 for details. The smoothing factor obtained from the compatible relaxation process gives an indication for the two-grid AMG convergence rate and can be controlled by the parameter θ, see [Liv04] and [FV04] for details.
A general form of interpolation can be derived from the minimization of a least-squares functional. This was already suggested in [Bra01] and further developed in [Kah09] and [BBKL11]. Given a set of linear independent smooth vectors $e^1, \dots, e^K$ (these can for example be derived from the compatible relaxation process), the goal is to obtain an accurate interpolation for these vectors in a least-square sense, i.e., to minimize, for all fine grid points $i \in F$, the functional

$$\sum_{k=1}^{K} \omega_k \left(e_i^k - \sum_{j \in I_i} w_{ij} e_j^k \right)^2$$

with respect to the interpolatory weights w_{ij}. Here $I_i \subset C$ denotes the set of interpolatory points for $i \in F$. The weights $\omega_k > 0$ can e.g. be chosen such that the energy (e^k, Ae^k) of the vectors e^k is reflected within the least squares functional.
It is even possible to let the multigrid solver improve itself. To this end, a tentative AMG hierarchy is used within a multilevel eigensolver to identify low-energy eigenmodes. Then, if needed, the interpolation operators are adapted (by the least-squares method described above) to better approximate these vectors. For details of this bootstrap AMG method, we refer to [Kah09] and [BBKL11].
Finally, it should be mentioned that compatible relaxation is not restricted to point-wise C/F coarsening. More generally, one can define the coarse space in terms of some restriction operator $R : \mathbb{R}^{N_l} \to \mathbb{R}^{N_{l+1}}$. Additionally, let $F : \mathbb{R}^{N_{l+1} - N_l} \to \mathbb{R}^{N_l}$ be such that $RF = 0$, i.e. the range of F defines the “fine grid space”, for which the error must be damped efficiently by smoothing. A compatible relaxation step (that leaves the coarse vectors unchanged) for some smoother $I - Q^{-1}A$ then reads

$$e^{it+1} = \left(I - F \left(F^T Q F \right)^{-1} F^T A \right) e^{it}.$$

A detailed description and error convergence estimates are given in [FV04].

2.12. AMG for systems of elliptic PDEs

In this section, we describe various approaches to the construction of AMG hierarchies for operators $\mathcal{A}$ that arise from the discretization of *systems* of partial differential equations,

$$\mathsf{L}(\mathsf{u}_1, \dots, \mathsf{u}_\mathsf{M}) = \mathsf{f}$$

where $\mathsf{u}_\mathsf{i}, \mathsf{i} = 1, \dots, \mathsf{M}$ are scalar functions, $\mathsf{u}_\mathsf{i} : \Omega \to \mathbb{R}$, $\Omega \subset \mathbb{R}^d$. The right hand side f is a vector-valued function, $\mathsf{f} : \Omega \to \mathbb{R}^M$.
We will use the term *physical unknown* for the u_i, $\mathsf{i} = 1, \dots, \mathsf{M}$, as in most applications, each u_i is the discrete variant of a physical quantity. For example, in a fluid dynamics simulation, we might have the quadruple

$$(\mathsf{u}_1, \mathsf{u}_2, \mathsf{u}_3, \mathsf{p})$$

where u_1, u_2 and u_3 describe the velocity components in the x, y and z spatial direction, respectively, while p describes the pressure.
We now give an example of an elliptic system of partial differential equations.

Example 2.6 (Linear Elasticity) [Bra97, Oel01, GOS03] We consider a solid $\Omega \subset \mathbb{R}^3$ with boundary $\partial\Omega = \Gamma_0 \dot{\cup} \Gamma_1$, which is fixed at the boundary Γ_0. We seek for the vector u that describes the displacement of this solid subject to an external force f and a surface force g on Γ_1. To this end, we define the strain tensor $\epsilon_{ij} = \frac{1}{2}(\partial_i \mathsf{u}_\mathsf{j} + \partial_j \mathsf{u}_\mathsf{i}) =: \mathsf{Du}$, which describes the deformation of the solid, and the stress tensor σ, which in this case is linearly related to the strain tensor, $\epsilon = \frac{1+\nu}{E}\sigma - \frac{\nu}{E} \operatorname{tr} \sigma I$. The (material-dependent) Poisson ration $\nu \in [0, 0.5)$ is determined as the ratio between transversal and radial deformation. Young's modulus E describes the elasticity of the material. Now, the deformation u is the minimizer of the energy functional

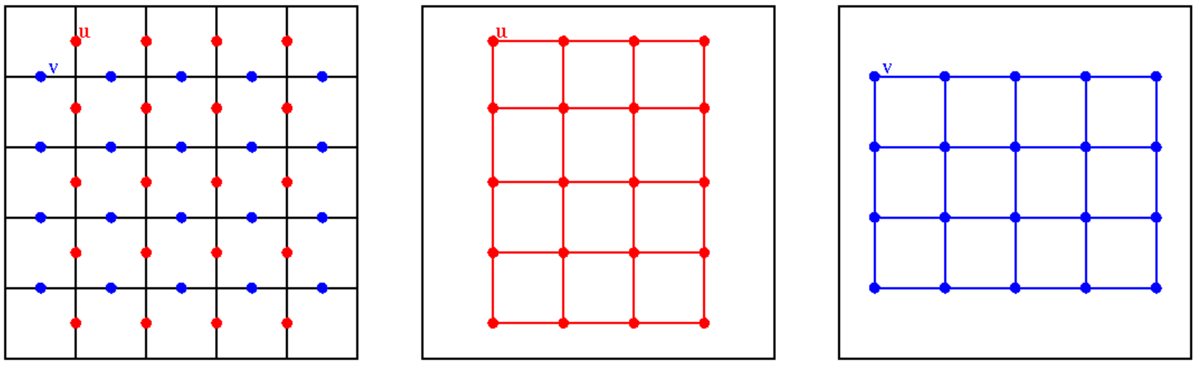
$$\int_\Omega \left(\frac{1}{2}\epsilon : \sigma - \mathsf{f} \cdot \mathsf{u} \right) d\Omega + \int_{\Gamma_1} \mathsf{g} \cdot \mathsf{u} d\Gamma_1$$

where $\epsilon : \sigma = \sum_{i,j=1}^3 \epsilon_{ij}\sigma_{ij}$. We eliminate the stress σ and rewrite $\epsilon = \mathsf{Du}$ to obtain a weak formulation of the problem: Find $\mathsf{u} \in H^1_{\Gamma_0} = \{\mathsf{v} \in H^1(\Omega)^3 : \mathsf{v}|_{\Gamma_0} = 0\}$ such that

$$\int_\Omega \mathsf{Du} : C\mathsf{Dv} d\Omega = \int_\Omega \mathsf{f} \cdot \mathsf{v} d\Omega - \int_{\Gamma_1} \mathsf{g} \cdot \mathsf{v} d\Gamma_1 \text{ for all } \mathsf{v} \in H^1_{\Gamma_0}.$$

where C depends on ν and E.
The solution u is not scalar, but has three physical unknowns, namely the displacement in x- y- and z- direction, respectively.

Throughout this section, we assume that the discretization of L leads to a symmetric positive definite matrix A. In the following, we describe three approaches to AMG for systems of equations. Each one of them has its own advantages and disadvantages, and the choice on which particular method to use for a specific problem has to be made with some knowledge of the underlying problem (or the structure of the matrix A) in mind. For a detailed introduction to AMG for elliptic systems of equations, we refer to [Cle04].



(a) Staggered discretization mesh (b) Discretization of the u component (c) Discretization of the v component

Figure 2.16.: A finite difference staggered grid with two physical unknowns. On the left, we give the mesh cells as well as the discretization positions for each physical unknown. The middle and the right picture show the connections of a discrete operator (e.g. the Laplacian) for each of the unknowns.

2.12.1. Variable-based AMG (VAMG)

The first approach is just to ignore the fact that A arises from the discretization of a system of partial differential equations. Instead, we apply the algebraic multigrid setup phase as described in the previous sections "as-is" to the matrix A. This so-called *variable-based AMG* ([Cle04], Section 3.2) will only work well if A is a M-matrix or an essentially positive matrix, see Sections 2.4.1 and 2.4.2. In practice, regarding systems of PDEs this requires that the couplings between the different physical unknowns are very weak.

2.12.2. Unknown-based AMG (UAMG)

In *unknown-based AMG* ([Cle04], Section 3.3), scalar AMG algorithms are applied to each physical unknown u_i separately. To apply this method, a variable-to-unknown mapping (*VU mapping*) has to be provided. Having a discrete vector $u \in \mathbb{R}^N$, this mapping identifies, for each entry u_i, the physical unknown $\mathsf{j} \in \{1, \ldots, \mathsf{M}\}$ that u_i belongs to. In other words, we disjointly divide the index set $\Omega = \Omega_{[1]} \dot{\cup} \ldots \dot{\cup} \Omega_{[\mathsf{M}]}$, where each $\Omega_{[i]}$ contains the indices corresponding to discretization of the physical unknown u_i. Now, we reorder the matrix such that it is sorted by the physical unknowns. We obtain a block structure,

$$A = \begin{pmatrix} A_{[1,1]} & A_{[1,2]} & \ldots & A_{[1,\mathsf{M}]} \\ A_{[2,1]} & A_{[2,2]} & \ldots & A_{[2,\mathsf{M}]} \\ \vdots & \vdots & \ddots & \vdots \\ A_{[\mathsf{M},1]} & A_{[\mathsf{M},2]} & \ldots & A_{[\mathsf{M},\mathsf{M}]} \end{pmatrix}, \tag{2.84}$$

where each matrix block $A_{[i,j]}$, $i, j = 1, \ldots, \mathsf{M}$ describes the couplings between the physical unknowns i and j. For example, in Figure 2.16, we have two physical unknowns that are discretized at different positions. The connectivity structure of $A_{[1,1]}$ is depicted in red and the connectivity structure of $A_{[2,2]}$ is shown in blue.
We note that the matrix blocks $A_{[i,j]}, i \neq j$ are not necessarily square, as the discretization meshes $\Omega_{[i]}$ for different physical unknowns may not have the same size.
Each diagonal block $A_{[i,i]}$ can be viewed as the discretization of a scalar equation for the i-th unknown. In unknown-based AMG, we build the AMG around these scalar blocks. More precisely, for each $i = 1, \ldots, \mathsf{M}$, we carry out the following steps,

1. extract a strength matrix $S_{[i]}$ from the matrix entries of $A_{[i,i]}$,
2. construct a C/F-splitting $C_{[i]} \cup F_{[i]} \Omega_{[i]}$ based on $S_{[i]}$,
3. build an interpolation operator $P_{[i]} : \mathbb{R}^{|C_{[i]}|} \to \mathbb{R}^{|\Omega_{[i]}|}$, and a restriction operator $R_{[i]} = P_{[i]}^T$.

Now, we can merge the per-unknown coarse grids into a global coarse grid,

$$C = \bigcup_{i=1}^{\mathsf{M}} C_{[i]},$$

and we assemble the global interpolation (the so-called *multiple-unknown (MU)* interpolation) operator and the restriction operator,

$$P = \begin{pmatrix} P_{[1]} & 0 & \ldots 0 & \\ 0 & P_{[2]} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \ldots & 0 & P_{[\mathsf{M}]} \end{pmatrix}, \quad R = \begin{pmatrix} R_{[1]} & 0 & \ldots 0 & \\ 0 & R_{[2]} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \ldots & 0 & R_{[\mathsf{M}]} \end{pmatrix}. \tag{2.85}$$

Finally, we compute the coarse grid operator by the Galerkin product (*full Galerkin*),

$$A^C = RAP.$$

A different way to define the coarse grid operator is to employ the diagonal block entries $A_{[i,i]}$ only (*block Galerkin*), i.e.

$$\tilde{A^C} = \begin{pmatrix} R_{[1]} & 0 & \ldots 0 & \\ 0 & R_{[2]} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \ldots & 0 & R_{[\mathsf{M}]} \end{pmatrix} \begin{pmatrix} A_{[1,1]} & 0 & \ldots 0 & \\ 0 & A_{[2,2]} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \ldots & 0 & A_{[\mathsf{M},\mathsf{M}]} \end{pmatrix} \begin{pmatrix} P_{[1]} & 0 & \ldots 0 & \\ 0 & P_{[2]} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \ldots & 0 & P_{[\mathsf{M}]} \end{pmatrix}.$$

The resulting coarse grid matrix $\tilde{A^C}$ is block diagonal and thus sparser than the full Galerkin product. Hence, the computational cost on the coarser levels can be reduced, but the variational principle (Theorem 2.15) is violated.

As we have pointed out, the unknown-AMG approach treats each physical unknown separate. Hence, the coarse grids, transfer operators and coarse grid operators can be adapted to the specific properties (e.g. anisotropies, coefficient jumps, ...) inside each physical property, as these are reflected within the diagonal matrix blocks $A_{[i,i]}$. Also, it is not required that the different physical unknowns are discretized on a common mesh. On the downside, information *between* different unknowns (the entries inside the off-diagonal block matrices $A_{[i,j]}$, $i \neq j$) is completely ignored. If these entries are large, the resulting AMG hierarchy may loose its efficiency as not all relevant information is reflected.
To investigate the two-grid convergence of the unknown AMG approach, we introduce the block diagonal matrix A_u,

$$A_u = \begin{pmatrix} A_{[1,1]} & 0 & \dots 0 & \\ 0 & A_{[2,2]} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & A_{[\mathsf{M},\mathsf{M}]}. \end{pmatrix} \tag{2.86}$$

A_u is symmetric positive definite for symmetric positive definite A and hence defines a scalar product,

$$(u, v)_{u,1} := u^T A_u v.$$

The associated norm is denoted with $\|\cdot\|_{u,1}$. Now, we can formulate the approximation property for unknown-wise AMG, which is just a slight deviation from the scalar approximation property needed in Theorem (2.6).

Lemma 2.4 *([Cle04], Lemma 3.6)*
Let $A > 0$ and and a VU mapping be given. If the C/F-splitting and interpolation P_{FC} are such that the τ-condition of MU-interpolation (2.12.2),

$$\|e_F - P_{FC}e_C\|_{0,F}^2 \leq \tau_u \|e\|_{u,1}^2 \tag{2.87}$$

is fulfilled with τ_u being independent of e, then (2.35),

$$\|Te\|_1^2 \leq \tau \|Te\|_2^2, \tag{2.88}$$

is satisfied with $\tau = \tau_u \rho\left(A^{-1}A_u\right)$. ($T = \left(I - PA_C^{-1}RA\right)$ denotes the two-grid correction operator.)

Together with the smoothing property (2.27),

$$\|Me\|_1^2 \leq \|e\|_1^2 - \sigma \|e\|_2^2,$$

we obtain the following result for the two-grid cycle with one post-smoothing step.

Lemma 2.5 *([Cle04], Lemma 3.7)*
Let $A > 0$ and a VU mapping be given. Let M satisfy the smoothing property (2.27).

Furthermore, assume the C/F-splitting and interpolation to be such that the condition (2.87) *is fulfilled with some τ_u being independent of e. Then*

$$\|MT\|_1 \leq \sqrt{1 - \frac{\sigma}{\tau}}$$

is satisfied with $\tau = \tau_u \rho(A^{-1}A_u) \geq \sigma$.

The smoothing property (2.27) does not reflect the decomposition of A into the matrix blocks $A_{[i,j]}$. However, the AMG hierarchy is build solely using information from the diagonal blocks $A_{[i,i]}$. More precisely, we implicitly use heuristics that rely on the algebraic smoothness of the error with respect to each $A_{[i,i]}$ separately. Hence, to obtain an efficient interplay between smoothing and coarse grid correction, we should formulate the smoothing property in this sense.

Definition 2.9 *[Cle04]*
An operator M satisfies the unknown-smoothing property *with respect to $A > 0$ and a given VU mapping, if we have a $\sigma_u > 0$ such that for all e holds,*

$$\|Me\|_1^2 \leq \|e\|_{1,u}^2 - \sigma_u \|e\|_{2,u}^2. \tag{2.89}$$

Here, the scalar product $(u, v)_{u,2} := u^T A_u D^{-1} A_u v$ and the associated norm $\| \cdot \|_{2,u}$ are defined w.r.t A_u. Now, we can estimate the two-grid correction operator with one post-smoothing step MT in terms of $\| \cdot \|_{u,1}$ instead of $\| \cdot \|_1$.

Theorem 2.17 *([Cle04], Theorem 3.9) Let $A > 0$ and M satisfy the unknown-smoothing property* (2.89)*. Furthermore, assume the C/F-splitting and interpolation be such that the τ_u-condition* (2.87) *of MU-interpolation is fulfilled with τ_u being independent of e. Then*

$$\|MT\|_{u,1} \leq \sqrt{\rho(A^{-1}A_u)\rho(A_u^{-1}A)} \sqrt{1 - \frac{\sigma_u}{\tilde{\tau}}} \tag{2.90}$$

with $\tilde{\tau} = \tau_u \rho(A^{-1}A_u)^2 \rho\left((A_u^{-1}A)^2\right)$.

The factor

$$\rho_u := \rho(A^{-1}A_u)\rho(A_u^{-1}A)$$

can be interpreted as an indication of the strength of unknown cross-couplings, i.e. how well the spectrum of A is captured by A_u. If ρ_u is large, the overall convergence may deteriorate. Note, however, that the bound (2.90) is not sharp.

Remark 2.6 A variant on unknown-based AMG method for linear elasticity problems is described in [BKMY10]. Here, first a multiple-unknown interpolation (2.12.2) is computed, which is then augmented by couplings between different physical unknowns to capture the rigid body modes, which form the kernel of the matrix.

2.12.3. Point-Based AMG

The other main approach to the construction of AMG for systems of partial differential equations is to arrange the variables by points instead of physical unknowns. Then, the coarse grid, the interpolation and the coarse grid operator are obtained from the coarsening of these points. This approach is called *point-based AMG* ([Cle04], Section 3.4)
In this case, we first need a variable-to-point mapping (*VP mapping*) that assigns a point $j \in \tilde{\Omega} := \{1, \ldots, m\}$ to each entry u_i of a vector $u \in \mathbb{R}^N$. Again, we obtain a disjoint decomposition of $\Omega = \Omega_{(1)} \dot{\cup} \ldots \dot{\cup} \Omega_{(m)}$ and a block structure of the matrix A,

$$A = \begin{pmatrix} A_{(1,1)} & A_{(1,2)} & \dots & A_{(1,m)} \\ A_{(2,1)} & A_{(2,2)} & \dots & A_{(2,m)} \\ \vdots & \vdots & \ddots & \vdots \\ A_{(m,1)} & A_{(m,2)} & \dots & A_{(m,m)} \end{pmatrix}, \tag{2.91}$$

where each $A_{(i,j)}$, $i, j = 1, \ldots, m$ describes the connections between all physical quantities that are discretized at the points i and j. Note that not all physical unknowns need to exist at each point, so the off-diagonal matrices $A_{(i,j)}$, $i \neq j$, might be non-square.

Block smoothing We employ the block structure of the matrix to define block relaxation schemes, which are straightforward extensions of the point-wise Jacobi and Gauss-Seidel iterations. Define the block diagonal matrix

$$D_P := \begin{pmatrix} A_{(1,1)} & 0 & \dots & 0 \\ 0 & A_{(2,2)} & \ddots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & A_{(m,m)} \end{pmatrix} \tag{2.92}$$

and the block lower triangular matrix

$$L_P := \begin{pmatrix} 0 & 0 & \dots & 0 \\ A_{(2,1)} & 0 & \ddots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ A_{(m,1)} & \dots & A_{(m,m-1)} & 0 \end{pmatrix}$$

Then, in terms of the error propagation matrix, the *damped block Jacobi smoother* is given by

$$M = I - \omega D_P^{-1} A$$

and the *block Gauss-Seidel iteration* by

$$M = I - (D_P + L_P)^{-1} A.$$

To formulate the smoothing property for the block smoothers, we introduce the inner products ([Cle04], Section 2.4.5)

$$(u,v)_{P,0} \;:=\; (D_P u, v) \tag{2.93}$$
$$(u,v)_{P,2} \;:=\; (D_P^{-1} Au, Av) \tag{2.94}$$

and the respective discrete norms $\|u\|_{P,0}$, and $\|u\|_{P,2}$, cf. Definition 2.3. Note that we do not need to define an inner product $(u,v)_{P,1}$. As in (2.25) we have

$$\|e\|_1^2 \le \|e\|_{P,0}\|e\|_{P,2}, \quad \|e\|_{P,2}^2 \le \rho(D^{-1}A)\|e\|_1^2, \quad \|e\|_1^2 \le \rho(D^{-1}A)\|e\|_{P,0}^2. \tag{2.95}$$

We are now ready to define a smoothing property in terms of these norms.

Definition 2.10 *([Cle04], Section 3.4.1.1) An operator M satisfies the* point-smoothing property *with respect to $A > 0$, if we have a $\sigma > 0$ such that for all e holds,*

$$\|Me\|_1^2 \le \|e\|_1^2 - \sigma\|e\|_{P,2}^2. \tag{2.96}$$

The smoothing property for Gauss-Seidel and damped Jacobi iteration can be shown analogous to the scalar case.

Theorem 2.18 *([Oel01], Theorem 3.1)*
Let the block matrix A be symmetric, positive definite and all $A_{(i,j)} \in \mathbb{R}^{\mathsf{M}\times\mathsf{M}}$. Furthermore, let $M = I - (L_P + D_P)^{-1}A$ or $M = I - (U_P + D_P)^{-1}A$, where L_P (U_P) denotes the lower (upper) block triangular part of A, $\|\cdot\|$ an operator norm induced by a fixed vector norm for $\mathbb{R}^{\mathsf{M}}$, let $w \in \mathbb{R}^m$ be an arbitrary vector, $w_i > 0$ for all i. Then, the block Gauss-Seidel iteration M satisfies the smoothing property (2.96) *with $\sigma = \frac{1}{\gamma_-\gamma_+}$. Here, γ_- and γ_+ are defined by*

$$\gamma_- := \max_{1\le i\le m}\left\{\frac{1}{w_i}\sum_{j\le i} w_j \|A_{(i,i)}^{-1}A_{(i,j)}\|\right\} \text{ and } \gamma_+ := \max_{1\le i\le m}\left\{\frac{1}{w_i}\sum_{j\ge i} w_j \|A_{(i,i)}^{-1}A_{(i,j)}\|\right\}.$$

Theorem 2.19 *(cf. Theorem 2.5) Let the block matrix A be symmetric positive definite and $\eta \ge \rho(D_P^{-1}A)$. Then, the damped Jacobi relaxation $S = I - \omega D_P^{-1}A$ satisfies the smoothing property* (2.96) *with $\sigma = \omega(2-\omega\eta)$ for any $0 < \omega < \frac{2}{\eta}$. The optimal relaxation parameter is given by $\omega^* = \frac{1}{\eta}$, in in this case also $\sigma = \frac{1}{\eta}$.*

Point Coarsening As mentioned above, the coarsening is carried out on the set of points $\tilde{\Omega}$, i.e. $\tilde{\Omega} = \tilde{C}\dot{\cup}\tilde{F}$. The classical Ruge-Stüben coarsening algorithm 2.5, however, requires a strength matrix $\tilde{S} \in \mathbb{R}^{m\times m}$ defined on the point set $\tilde{\Omega}$. To compute the strength matrix, we first need a so-called *primary matrix* $\tilde{A} \in \mathbb{R}^{m\times m}$ from which $\tilde{S}$ can be computed as described in Section 2.5.
We now give the most common approaches to the construction of $\tilde{A}$.

1. The entries $\tilde{a}_{ij}$, $i, j = 1, \dots, m$ of $\tilde{A}$ are computed using the block structure (2.91). A widely used ansatz is to compute the norms of the blocks,

$$\tilde{a}_{ij} := -\|A_{(i,j)}\| \text{ for } i \neq j. \tag{2.97}$$

   For the diagonal entry, we have the possibility of either computing the norm of the diagonal, i.e. $\tilde{a}_{ii} := \|A_{(i,i)}\|$ or using the negative sum of the off-diagonal entries,

$$\tilde{A}_{i,i} := \begin{cases} -\sum_{j\neq i} \tilde{a}_{ij} \text{ if } \sum_{j\neq i} \tilde{a}_{ij} \neq 0 \\ 1 \text{ else.} \end{cases}$$

   The latter has the advantage that $\tilde{A}$ is guaranteed to be a symmetric positive definite M-matrix.
   Note that all off-diagonal entries are negative. One could also take into account the definiteness of the sub matrices, i.e.

$$\tilde{a}_{ij} := \begin{cases} \|A_{(i,j)}\| \text{ if } A_{i,j} \geq 0. \\ -\|A_{(i,j)}\| \text{ else} \end{cases} \quad \text{for } i \neq j, \tag{2.98}$$

   and compute $\tilde{a}_{ii}$ either as $\|A_{i,i}\|$ or $\tilde{a}_{i,i} := \sum_{j\neq i} |\tilde{a}_{ij}|$. This would make some theoretical considerations easier, but only if all off-diagonal blocks $A_{i,j}$ are symmetric and at least semi-definite, see [Cle04], Section 3.4.4. However, the computation of (2.98) is significantly more expensive than (2.97).
   The question remains which norm to choose in (2.97). For theoretical considerations, the Euclidean norm $\|A\|_E = \sqrt{\lambda_{\max}(A^T A)}$ is the easiest to handle. In practice, the Frobenius norm or the row sum norm are often used as they are easy to compute. It is also possible to use quantities like the largest (absolute) element in each block or the sum of all elements.

2. We can designate one of the physical unknowns $\mathsf{u}_{[\mathsf{k}]}$ to lead the coarsening, i.e. $\tilde{A} = A_{[k,k]}$, where $A_{[k,k]}$ is the matrix block corresponding to unknown k, cf. (2.84). In this case it is desirable that the unknown k is “representative” for the whole system, i.e. that k is present at each point i and its sparsity pattern resembles the block sparsity pattern of 2.91, that is, $a_{ij}^{[k,k]} \neq 0$ if the matrix block $A_{(i,j)}$ is not empty. We refer to [Cle04], Section 3.4.2.1 and the references therein for a more detailed explanation.

3. Another approach, related to the previous one, is to provide an external scalar matrix $\tilde{A}$ which is not computed or extracted from A. In this case, we also need to build a hierarchy for $\tilde{A}$ analogously to the hierarchy of A to obtain this information on the coarser levels. A common choice is to employ a discretization of the Laplacian on the same mesh. This approach is for example being used to build an AMG hierarchy inside a CFD solver [LSAC08].

4. If geometric information is available, the primary matrix can also be obtained from the coordinate vectors x_i of the discretization points,

$$\tilde{a}_{ij} = -\frac{1}{|x_i - y_j|} \text{ if } i \neq j \text{ and } A_{(i,j)} \neq 0. \tag{2.99}$$

   Again, the diagonal entry is computed as the negative sum over all off-diagonal entries, $\tilde{a}_{ii} = -\sum_{j \neq i} \tilde{a}_{ij}$. Note that the block sparsity pattern of A is copied to $\tilde{A}$. More sophisticated approaches that also reflect the positions of the points to each other are possible, see [Cle04], Section 3.4.2.4.

We now apply the strength algorithm 2.4 to the primary matrix $\tilde{A}$ and obtain a strength matrix $\tilde{S} \in \mathbb{R}^{m \times m}$. Here, as in (2.32), we consider the absolute values of $\tilde{A}$, i.e. we set $s_{ij} = 1$ if and only if

$$|\tilde{a}_{ij}| \geq \alpha \cdot \max_{k \neq i} |\tilde{a}_{ik}|.$$

As mentioned above, $\tilde{A}$ only has non-positive off-diagonal entries if it is constructed as a norm- or distance based primary matrix.

Interpolation for point-based AMG It remains to construct the interpolation operator. Again, there are different approaches available. First, we introduce some notation (as in Section 2.8),

$\tilde{E}_i = \{j \neq i : \ A_{(i,j)} \neq 0\}$	set of neighbors of i.
$\tilde{C}_i = \tilde{S}_i \cap \tilde{C}$	set of strongly connected coarse grid points of i.
$\tilde{F}_i^s = \tilde{S}_i \cap \tilde{F}$	set of strongly connected coarse grid points of i.
$\tilde{E}_i^w = \tilde{E}_i \setminus \tilde{S}_i$	set of weakly connected neighbors of i.
$\tilde{I}_i$	set of interpolatory points for i.

Block interpolation The first option is to extend the scalar interpolation formulas to the block structure (*block interpolation*, [Cle04], Section 3.4.3.1). This is the most "natural" choice if we use a block smoother and the matrix blocks are also used to compute the strength matrix. Analogously to the construction of scalar interpolation operators as described in 2.8, we approximate

$$A_{(i,i)} e_{(i)} + \sum_{j \in \tilde{E}_i} A_{(i,j)} e_{(j)} = r_{(i)} = 0. \tag{2.100}$$

by the sum

$$e_{(i)} + \sum_{j \in \tilde{E}_i} W_{(i,j)} e_{(j)} = 0. \tag{2.101}$$

Note that we now deal with blocks instead of scalars. $e_{(i)}$ and $r_{(i)}$ describe the parts of e and r that are associated with the point i, i.e. they are vectors of size M_i, the number of physical unknowns present at the point i. $A_{(i,j)}$ and $W_{(i,j)}$ are matrices of size $\mathsf{M}_i \times \mathsf{M}_j$. The general form of a block interpolation operator is given by (cf. (2.38))

$$e_{(i)} = \begin{cases} e_{(i)} \text{ if } i \in \tilde{C} \\ \sum_{j \in \tilde{S}_i \cap \tilde{C}} W_{(i,j)} e_{(j)} \text{ else.} \end{cases} \tag{2.102}$$

In the following, we assume that all physical unknowns $\mathsf{k} \in 1, \ldots, \mathsf{M}$ are present at all points $i = 1, \ldots, m$, which, in consequence, means that all matrix blocks $A_{(i,j)}$ are square. We first introduce the block version of direct interpolation (2.43) [Oel01, GOS03]. The weights then read

$$W_{(i,j)} = -A_{(i,i)}^{-1} \left(\sum_{k \in \tilde{E}_i} A_{(i,k)} \right) \left(\sum_{k \in \tilde{C}_i} A_{(i,k)} \right)^{-1} A_{(i,j)}. \tag{2.103}$$

Note that this construction is only possible if $\left(\sum_{k \in \tilde{C}_i} A_{(i,k)}\right)$ is invertible. ($A_{(i,i)}$ is always non-singular due to the symmetric positive definiteness of A.) To circumvent this restraint, we can replace the normalization matrices $\left(\sum_{k \in \tilde{E}_i} A_{(i,k)}\right)$ and $\left(\sum_{k \in \tilde{C}_i} A_{(i,k)}\right)^{-1}$ by diagonal $\mathsf{m} \times \mathsf{m}$-matrices that have the same row sums (if not equal to zero), i.e. for each block row i we define ([Cle04], Section 3.4.3.1)

$$\tilde{R}^{E_i}_{kk} := \begin{cases} \tilde{r}^{E_i}_{kk} \text{ if } \tilde{r}_{kk} \neq 0 \\ 1 \text{ else.} \end{cases}, \quad \tilde{R}^{P_i}_{kk} := \begin{cases} \tilde{r}^{P_i}_{kk} \text{ if } \tilde{r}_{kk} \neq 0 \\ 1 \text{ else.} \end{cases},$$

where

$$\tilde{r}^{E_i}_{kk} := \sum_{j \in \tilde{E}_i} \sum_{\mathsf{l}=1}^{\mathsf{M}} a^{(i,j)}_{\mathsf{kl}}, \quad \tilde{r}^{I_i}_{kk} := \sum_{j \in \tilde{I}_i} \sum_{\mathsf{l}=1}^{\mathsf{M}} a^{(i,j)}_{\mathsf{kl}},$$

and $A_{(i,j)} = \left(a^{(i,j)}_{\mathsf{kl}}\right)_{\mathsf{k,l}=1}^{\mathsf{M}}$. The formula for $W_{i,j}$ then reads

$$W_{(i,j)} = -A_{(i,i)}^{-1} \, \tilde{R}^{E_i}_{kk} \, \left(\tilde{R}^{P_i}_{kk}\right)^{-1} \, A_{i,j)}.$$

This variant is cheaper to compute than (2.103) and can also easily be applied to the case of non-square $A_{(i,j)}$. The sum of the interpolation weights $\sum_j W_{(i,j)}$ is however not necessarily the identity, hence constant vectors may not be interpolated exactly (cf. (2.42)).

Theoretically, the block direct interpolation can also be carried out with separate renormalization scalings for positive and non-positive semi-definite weights, cf. (2.45)-(2.46) To this end, we define

$$\begin{aligned} \tilde{E}_i^- &:= \{j \in \tilde{E}_i : \; A_{(i,j)} \leq 0\} \\ \tilde{E}_i^+ &:= \{j \in \tilde{E}_i : \; A_{(i,j)} \geq 0\} \\ \tilde{C}_i^- &:= \{j \in \tilde{C}_i : \; A_{(i,j)} \leq 0\} \\ \tilde{C}_i^+ &:= \{j \in \tilde{C}_i : \; A_{(i,j)} \geq 0\} \end{aligned}$$

and we compute the interpolatory weights as follows,

$$W_{ij} = \quad -A_{(i,i)}^{-1}\left(\sum_{k\in E_i^+} A_{(i,k)}\right)\left(\sum_{k\in C_i^+} A_{(i,k)}\right)^{-1} A_{(i,j)} \text{ if } A_{(i,j)} \geq 0. \tag{2.104}$$

$$W_{ij} = \quad -A_{(i,i)}^{-1}\left(\sum_{k\in E_i\setminus E_i^+} A_{(i,k)}\right)\left(\sum_{k\in C\setminus C_i^+} A_{(i,k)}\right)^{-1} A_{(i,j)} \text{ else.} \tag{2.105}$$

Note that the indefinite submatrices are treated together with the negative (semi-) definite ones. This requires us however to determine the definiteness of each submatrix $A_{(i,j)}$, $i \neq j$, see (2.98) and the discussion thereafter.
For sake of completeness, we give the approximation property for the block direct interpolation schemes. To this end, we first define

$$\begin{aligned} T_{(i)} &:= A_{(i,i)} + \sum_{k\in E_i\setminus E_i^+} A_{(i,k)} - \sum_{k\in E_i^+} A_{(i,k)}. \\ \mu_i &:= \sum_{k\in I_i} \|W_{(i,k)}\|_E + \|I - \sum_{k\in I_i^-} W_{(i,k)} + \sum_{k\in I_i^+} W_{(i,k)}\|_E \end{aligned}$$

$T_{(i)}$ can be seen as the block-version counterpart of $t_i = a_{ii} - \sum_{j\in E_i} |a_{ij}|$ from Theorem 2.9. We now give the straightforward analogs of (2.36) and (2.37) for block AMG. First, let us look at the convergence of the two-grid cycle with one post-smoothing step. To this end, let $T = I - PA_C^{-1}RA$ be the exact coarse grid correction step, where the coarse grid operator is given by the Galerkin product $A_C = RAP$.

Theorem 2.20 *([Cle04], Theorem 3.11)*
Let $A > 0$ and M satisfy the point-smoothing property (2.96). Furthermore, assume the C/F-splitting and the interpolation be such that for all e

$$\|Te\|_1^2 \leq \tau \|e\|_{P,2}^2 \tag{2.106}$$

with some $\tau > 0$ being independent of e. Then $\tau \geq \sigma$ and

$$\|MT\|_1 \leq \sqrt{1 - \frac{\sigma}{\tau}}.$$

Theorem 2.21 *([Cle04], Theorem 3.12)*
If $A > 0$ and M and the C/F-splitting and interpolation $P_{(FC)}$ are such that for all e

$$\|e_F - P_{FC}e_C\|_{P,0,F}^2 \leq \tau \|e\|_1^2 \tag{2.107}$$

with some $\tau > 0$ being independent of e, then (2.106) is satisfied.

The following theorem gives sufficient criteria for (2.107) in the case of block direct interpolation schemes.

Theorem 2.22 *([Cle04], Corollary 3.1)*
Let $A > 0$, $E_i = E_i^+ \cup E_i^-$ and $T_{(i)} \geq 0$ for all $i \in \tilde{\Omega}$. Select a C/F-splitting and set a I_i for each $i \in F$.

1. *If P_{FC} can be defined by* (2.105)-(2.104), *then*

$$\mu_i = \sum_{j \in I_i} \|W_{(i,j)}\|_E + \|A_{(i,i)}^{-1} T_{(i)}\|_E$$

with $T_{(i)}$ defined by (2.106). *If for all $i \in F$ the inequalities*

$$\begin{aligned} \tau \lambda_{\min}(T_{(i)}) &\geq \mu_i \kappa_E(A_{(i,i)}) \|T_{(i)}\|_E \\ \text{and } \tau \lambda_{\min}(-A_{(i,j)}) &\geq \mu_i \kappa_E(A_{(i,i)})\ \rho \left(\left[\sum_{k \in E_i^-} A_{(i,j)} \right] \left[\sum_{k \in P_i^-} A_{(i,j)} \right]^{-1} \right) \quad \text{for all } j \in I_i^-, \\ \text{and } \tau \lambda_{\min}(A_{(i,j)}) &\geq \mu_i \kappa_E(A_{(i,i)})\ \rho \left(\left[\sum_{k \in E_i^+} A_{(i,j)} \right] \left[\sum_{k \in P_i^+} A_{(i,j)} \right]^{-1} \right) \quad \text{for all } j \in I_i^+, \end{aligned}$$

hold with a $\tau \geq 1$ not depending on i and j, the τ-condition (2.107) *is fulfilled.*

2. *If P_{FC} can be defined by* (2.103), *then*

$$\mu_i = \sum_{j \in I_i} \|W_{(i,j)}\|_E + \left\| I + A_{(i,i)}^{-1} \left(\sum_{j \in E_i} A_{(i,j)} \right) \Phi_{(i)} \right\|_E$$

with

$$\Phi_{(i)} := \left(\sum_{k \in E_i} A_{(i,k)} \right)^{-1} \left(\sum_{k \in E_i^-} A_{(i,k)} - \sum_{k \in E_i^+} A_{(i,k)} \right).$$

If for all $i \in F$ the inequalities

$$\begin{aligned} \tau \lambda_{\min}(T_{(i)}) &\geq \mu_i \kappa_E(A_{(i,i)}) \|A_{(i,i)} + \left(\sum_{j \in E_i} A_{(i,j)} \right) \Phi_{(i)} \|_E \\ \text{and } \tau \lambda_{\min}(-A_{(i,j)}) &\geq \mu_i \kappa_E(A_{(i,i)}) \left\| \left[\sum_{k \in E_i} A_{(i,j)} \right] \left[\sum_{k \in P_i} A_{(i,j)} \right]^{-1} \right\|_E \quad \text{for all } j \in I_i^-, \\ \text{and } \tau \lambda_{\min}(A_{(i,j)}) &\geq \mu_i \kappa_E(A_{(i,i)}) \left\| \left[\sum_{k \in E_i} A_{(i,j)} \right] \left[\sum_{k \in P_i} A_{(i,j)} \right]^{-1} \right\|_E \quad \text{for all } j \in I_i^+, \end{aligned}$$

hold with a $\tau \geq 1$ not depending on i and j, the τ-condition (2.107) *is fulfilled.*

The first part of this theorem can only be applied if all off-diagonal matrices $A_{(i,j)}$ are symmetric and either positive or negative semi-definite. If all $A_{(i,j)} \leq 0$, $i \neq 0$, then $\Phi_{(i)} = I$ and both statements of this theorem coincide.
If $\lambda_{\min}(T_{(i)}) = 0$, e.g. one of the equations in block row i has a zero row sum, the conditions in Theorem 2.22 require that also $\mu_i = 0$. This essentially means that $\sum_{k\in I_i^-} W_{(i,k)} - \sum_{k\in I_i^+} W_{(i,k)} = I$, i.e. constant vectors are interpolated exactly.

Remark 2.7 If, in Theorem 2.22, we replace all $A_{(i.i)}$ with the diagonal matrices $D_{(i,i)} = diag(A_{(i,i)})$, we obtain that the respective interpolation operators satisfy the same kind of approximation property as in the scalar case (2.37),

$$\|e_F - P_{FC}e_C\|_{0,F}^2 \leq \tau \|e\|_1^2.$$

Remark 2.8 The block direct interpolation scheme (2.103) can also be used inside an AMG methods for solving the equations arising in linear elasticity computations. Within this context, it is important that the interpolation preserves the so-called rigid body modes, i.e. the translations and rotations, as these vectors form the kernel of the operator considered. It can be shown [Oel01, GOS03] that (2.103) exactly interpolates the rigid body modes if used in conjunction with a suitable coarsening scheme.

Of course, it is also possible to derive block interpolation counterparts for the more sophisticated interpolation schemes describes in Section 2.8. For example, the block version of the classical interpolation scheme employs the following weights [hyp],

$$W_{(i,j)} = -\left(A_{(i,i)} + \sum_{l\in E_i^w} A_{(i,l)}\right)^{-1} \left(A_{(i,j)} + \sum_{k\in F_i^s} \left(\sum_{m\in C_i} A_{(k,m)}\right)^{-1} A_{(i,k)}A_{(k,j)}\right). \tag{2.108}$$

Note that the denominators may become singular. To prevent this, we have to make two modifications.

1. If $\sum_{m\in C_i} A_{(k,m)}$ is singular for a $k \in F_i^s$, then the strong coupling S_{ik} is treated as a weak coupling, i.e. we add $A_{(i,k)}$ to $A_{(i,i)} + \sum_{l\in E_i^w} A_{(i,l)}$ instead of computing the indirect weight $\left(\sum_{m\in C_i} A_{(k,m)}\right)^{-1} A_{(i,k)}A_{(k,j)}$ in the rightmost term of (2.108).
2. If $A_{(i,i)} + \sum_{l\in E_i^w} A_{(i,l)}$ is singular (after the possible modification described above), it is replaced by the identity. Another option is to only use $A_{(i,i)}^{-1}$ here, as this can always be carried out.

To save computational costs, one can also interpolate within each physical unknown. To this end, let $D_{(i,j)} = diag(A_{(i,j)})$ be the diagonal of the block $A_{(i,j)}$ (not to be confused with D_P!). Then the interpolation formula reads

$$W_{(i,j)} = -\left(D_{(i,i)} + \sum_{l\in E_i^w} D_{(i,l)}\right)^{-1} \left(D_{(i,j)} + \sum_{k\in F_i^s} \left(\sum_{m\in C_i} D_{(k,m)}\right)^{-1} D_{(i,k)}D_{(k,j)}\right). \tag{2.109}$$

The same modifications as above need to be carried out to prevent inversions of singular blocks. A hybrid version of this two approaches was already described in a very early paper [Rug86],

$$W_{(i,j)} = -\left(A_{(i,i)} + \sum_{l\in E_i\setminus C} A_{(i,l)}B_{(l,i)}\right)^{-1}\left(A_{(i,j)} + \sum_{k\in E_i\setminus C} A_{(i,k)}B_{(k,j)}\right). \tag{2.110}$$

where the $B_{(k,j)}$, $k \notin C_i$, $j \in C_i \cup \{i\}$ are diagonal matrices defined by

$$b^{(k,j)}_{\nu\mu} = \begin{cases} \frac{b^{(k,j)}_{\nu\nu}}{\sum_{l\in C_i\cup\{i\}} b^{(k,l)}_{\nu\nu}} & \text{if } \mu = \nu \\ 0 \text{ else.} \end{cases}$$

Compared to (2.108), this saves some computations as we only need to compute a single full matrix block inversion per block row. Note that we do not distinguish between weak and strong connections between fine grid points here.

Multiple-unknown interpolation The second approach to the construction of the interpolation is to compute a *multiple-unknown* interpolation as described in Section 2.12.2 ([Cle04], Section 3.4.3.2). That is, after computing the C/F-splitting of the points, $\tilde\Omega = \tilde C \dot\cup \tilde F$, we extend these sets as well as the strength matrix $\tilde S$ to the set of all vertices Ω,

$$\begin{aligned} C &= \{k \in \Omega \text{ where } k \in \tilde\Omega_{(i)} \text{ and } i \in \tilde C\} \\ F &= \{k \in \Omega \text{ where } k \in \tilde\Omega_{(i)} \text{ and } i \in \tilde F\} \\ S_k &= \{l \in \Omega \text{ where } k \in \tilde\Omega_{(i)},\ l \in \tilde\Omega_{(j)} \text{ and } j \in \tilde S_{(i)}\}. \end{aligned}$$

Then, we can compute the interpolation unknown-wise using one of the scalar interpolation schemes described in Section 2.8. Note that we now may have strong couplings $j \in S_i$ where the corresponding matrix entry $a_{ij} = 0$, or, more generally, the set of strong couplings S_i for an $i \in \Omega$ may not reflect the large off-diagonal matrix entries of the row $(a_{ij})_{j=1}^N$ at all. Hence, the interpolation may be inaccurate in these cases.

Single-unknown interpolation Finally, it is also possible to derive the interpolation by means of the primary matrix $\tilde A$. We apply a scalar interpolation routine from Section 2.8 to $\tilde A$ (together with $\tilde S$, $\tilde C$, $\tilde F$) and obtain a scalar interpolation operator $\tilde P$. This operator and the C/F splitting are then extended to the whole domain Ω,

$$\begin{aligned} C &:= \{k \in \Omega \text{ where } k \in \tilde\Omega_{(i)} \text{ and } i \in \tilde C\}, \\ F &:= \{k \in \Omega \text{ where } k \in \tilde\Omega_{(i)} \text{ and } i \in \tilde F\}, \\ p_{kl} &:= \tilde p_{ij} \text{ where } k \in \tilde\Omega_{(i)},\ l \in \tilde\Omega_{(j)}, \end{aligned}$$

where $P = (p_{kl})_{k\in\Omega,\ l\in C}$ and $\tilde P = (\tilde p_{ij})_{i\in\tilde\Omega,\ j\in\tilde C}$. We obtain interpolation weights that are identical for all physical unknowns at the same point i (*single-unknown interpolation*

[Cle04], Section 3.4.3.3). Again, the interpolation is carried out unknown-wise, but modifications are possible (and may be required) if not all unknowns are present at each grid point, see [Cle04], Section 4.3.2.2 for details.
We conclude this paragraph with some theoretical approximation properties of single-unknown direct interpolation in the case of a primary matrix defined by Euclidean norms.

Theorem 2.23 *([Cle04], Theorem 3.14 and Corollary 3.3) Let* $A > 0$*,* $E_i = E_i^+ \cup E_i^-$ *and* $T_{(i)} \geq 0$ (2.106) *for all* $i \in \tilde{\Omega}$*. Select a C/F-splitting and set a* I_i *for each* $i \in F$*.*

1. *Let* $\tilde{A}$ *be defined by* (2.98)*,*

$$\tilde{a}_{ij} := \begin{cases} \|A_{(i,j)}\|_E \text{ if } A_{i,j} \geq 0. \\ -\|A_{(i,j)}\|_E \text{ else} \end{cases} \quad \text{for } i \neq j, \quad \tilde{a}_{ii} := \|A_{(i,i)}\|_E$$

and the interpolation weights be derived from $\tilde{A}$ *by direct interpolation with weight separation* (2.45) - (2.46)*,*

$$\begin{aligned} \tilde{w}_{ij}^- &= -\frac{1}{\tilde{a}_{ii}} \frac{\sum_{k \in \tilde{E}_i} \tilde{a}_{ik}^-}{\sum_{k \in \tilde{C}_i^-} \tilde{a}_{ik}^-} \tilde{a}_{ij} \text{ if } \tilde{a}_{ij} < 0, && (2.111) \\ \tilde{w}_{ij}^+ &= -\frac{1}{\tilde{a}_{ii}} \frac{\sum_{k \in \tilde{E}_i} \tilde{a}_{ik}^+}{\sum_{k \in \tilde{C}_i^-} \tilde{a}_{ik}^+} \tilde{a}_{ij} \text{if } \tilde{a}_{ij} > 0. && (2.112) \end{aligned}$$

If for all $i \in \tilde{F}$

$$t_i := \|A_{(i,i)}\|_E - \sum_{k \in \tilde{E}_i} \|A_{(i,k)}\|_E \geq 0, \tag{2.113}$$

as well as the inequalities

$$\begin{aligned} \tau \lambda_{\min}(T_{(i)}) &\geq t_i \\ \text{and } \tau \lambda_{\min}(-A_{(i,j)}) &\geq \|A_{(i,j)}\|_E \frac{\sum_{k \in E_i^-} \|A_{(i,k)}\|_E}{\sum_{k \in C_i^-} \|A_{(i,k)}\|_E} \text{ for all } j \in \tilde{I}_i^-, \\ \text{and } \tau \lambda_{\min}(A_{(i,j)}) &\geq \|A_{(i,j)}\|_E \frac{\sum_{k \in E_i^+} \|A_{(i,k)}\|_E}{\sum_{k \in C_i^+} \|A_{(i,k)}\|_E} \text{ for all } j \in \tilde{I}_i^+, \end{aligned}$$

hold with a $\tau \geq 1$ *not depending on* i *and* j*, the* τ*-condition* (2.107)*,*

$$\|e_F - P_{FC} e_C\|_{P,0,F}^2 \leq \tau \|e\|_1^2$$

is fulfilled.

2. *Let* $\tilde{A}$ *be defined by* (2.97)*,*

$$\tilde{a}_{ij} := -\|A_{(i,j)}\|_E \text{ for } i \neq j, \quad \tilde{a}_{ii} := \|A_{(i,i)}\|_E.$$

and the interpolation weights be derived from $\tilde{A}$ *by direct interpolation* (2.43)*,*

$$\tilde{w}_{ij} = -\frac{1}{\tilde{a}_{ii}} \frac{\sum_{k\in\tilde{E}_i} \tilde{a}_{ik}}{\sum_{k\in\tilde{C}_i} \tilde{a}_{ik}}\, \tilde{a}_{ij} \tag{2.114}$$

If for all $i \in \tilde{F}$

$$\|A_{(i,i)}\|_E - \phi_i \sum_{k\in\tilde{E}_i} \|A_{(i,k)}\|_E \geq 0,$$

where

$$\phi_i := \frac{\sum_{k\in C_i^-} \|A_{(i,k)}\|_E - \sum_{k\in C_i^+} \|A_{(i,k)}\|_E}{\sum_{k\in C_i} \|A_{(i,k)}\|_E} \tag{2.115}$$

as well as the inequalities

$$\begin{aligned} \tau\lambda_{\min}(T_{(i)}) &\geq \|A_{(i,i)}\|_E - \phi_i \sum_{k\in\tilde{E}_i} \|A_{(i,k)}\|_E \\ \text{and } \tau\lambda_{\min}(-A_{(i,j)}) &\geq \|A_{(i,j)}\|_E \frac{\sum_{k\in E_i} \|A_{(i,k)}\|_E}{\sum_{k\in C_i} \|A_{(i,k)}\|_E} \text{ for all } j \in \tilde{I}_i^-, \\ \text{and } \tau\lambda_{\min}(A_{(i,j)}) &\geq \|A_{(i,j)}\|_E \frac{\sum_{k\in E_i} \|A_{(i,k)}\|_E}{\sum_{k\in C_i} \|A_{(i,k)}\|_E} \text{ for all } j \in \tilde{I}_i^+, \end{aligned}$$

hold with a $\tau \geq 1$ *not depending on* i *and* j*, the* τ*-condition* (2.107) *is fulfilled.*

3. *Let* $\tilde{A}$ *be defined by* (2.98)*,*

$$\tilde{a}_{ij} := \begin{cases} \|A_{(i,j)}\|_E \text{ if } A_{i,j} \geq 0. \\ -\|A_{(i,j)}\|_E \text{ else} \end{cases} \quad \text{for } i \neq j, \quad \tilde{a}_{ii} := \sum_{j\neq i} -\tilde{a}_{ij},$$

and the interpolation weights be derived from $\tilde{A}$ *by direct interpolation with weight separation* (2.111) - (2.112)*, If for all* $i \in \tilde{F}$ *the inequalities*

$$\begin{aligned} \text{and } \tau\lambda_{\min}(-A_{(i,j)}) &\geq \frac{\|A_{(i,j)}\|_E \|A_{(i,i)}\|_E}{\sum_{k\in E_i} \|A_{(i,k)}\|_E} \frac{\sum_{k\in E_i^-} \|A_{(i,k)}\|_E}{\sum_{k\in C_i^-} \|A_{(i,k)}\|_E} \text{ for all } j \in \tilde{I}_i^-, \\ \text{and } \tau\lambda_{\min}(A_{(i,j)}) &\geq \frac{\|A_{(i,j)}\|_E \|A_{(i,i)}\|_E}{\sum_{k\in E_i} \|A_{(i,k)}\|_E} \frac{\sum_{k\in E_i^+} \|A_{(i,k)}\|_E}{\sum_{k\in C_i^+} \|A_{(i,k)}\|_E} \text{ for all } j \in \tilde{I}_i^+, \end{aligned}$$

hold with a $\tau \geq 1$ *not depending on* i *and* j*, the* τ*-condition* (2.107) *is fulfilled. (Note that* $t_i = 0$ (2.113) *here.)*

4. *Let* $\tilde{A}$ *be defined by* (2.97)*,*

$$\tilde{a}_{ij} := -\|A_{(i,j)}\|_E \text{ for } i \neq j, \quad \tilde{a}_{ii} := \sum_{j\neq i} -\tilde{a}_{ij}$$

and the interpolation weights be derived from $\tilde{A}$ by direct interpolation (2.114)*. If for all $i \in \tilde{F}$*

$$1 - \phi_i = 0$$

where ϕ_i is defined by (2.115)*, as well as the inequalities*

$$\begin{aligned} \tau\lambda_{\min}(T_{(i)}) &\geq (1-\phi_i)\|A_{(i,i)}\|_E \\ \text{and } \tau\lambda_{\min}(-A_{(i,j)}) &\geq \frac{\|A_{(i,j)}\|_E\|A_{(i,i)}\|_E}{\sum_{k\in C_i}\|A_{(i,k)}\|_E} \text{ for all } j \in \tilde{I}_i^-, \\ \text{and } \tau\lambda_{\min}(A_{(i,j)}) &\geq \frac{\|A_{(i,j)}\|_E\|A_{(i,i)}\|_E}{\sum_{k\in C_i}\|A_{(i,k)}\|_E} \text{ for all } j \in \tilde{I}_i^+, \end{aligned}$$

hold with a $\tau \geq 1$ not depending on i and j, the τ-condition (2.107) *is fulfilled.*

Remark 2.9 If we replace $A_{(i,i)}$ with its diagonal $D_{(i,i)}$ in all of the statements above, we obtain the classical approximation property (2.37),

$$\|e_F - P_{FC}e_C\|_{0,F}^2 \leq \tau\|e\|_1^2.$$

instead of(2.107).

Finally, regardless of the coarsening and interpolation strategy chosen, the coarse grid operator is computed by the Galerkin ansatz,

$$A_C = RAP.$$

As already mentioned, we also might need to transfer the primary matrix to the coarse level (especially if we employ an externally-defined primary matrix that cannot be extracted from A, but have a single-unknown transfer matrix $\tilde{P}$ available),

$$\tilde{A}_C = \tilde{R}\tilde{A}\tilde{P}.$$

3. Parallel AMG

In the last chapter, especially in Sections 2.3–2.9, we introduced the AMG setup procedure according to Ruge and Stüben. Recall from Algorithm 2.3 that on each level l we have to

1. determine the set of strong couplings S_i for all $i \in \Omega^l$;
2. based on the sets S_i, disjointly divide Ω^l into the set of coarse grid points C^l and fine grid points F^l and set $\Omega^{l+1} = C^l$;
3. construct the prolongation matrix P_l and set the restriction $R_l := P_l^T$;
4. compute the coarse grid operator $A_{l+1} := R_l A_l P_l$.

Now, the question arises how these steps can be parallelized on a *multiple instruction, multiple data (MIMD)* parallel computer, i.e. on a machine where each processor has exclusive access to its own memory and operates independently of the others, while communication between different processors is performed by *message passing* (e.g. implemented in MPI [MPI]). This class of parallel computers ranges from multi-core desktop PCs to recent petascale supercomputers.
Let us assume that on the finest level we have a non-overlapping partitioning

$$\Omega^1 = \Omega^1_1 \dot\cup \ldots \dot\cup \Omega^1_{np} \tag{3.1}$$

where np denotes the number of processes. This partitioning can e.g. be generated by a mesh partitioning method like Metis [KK99], Parmetis [KK97] or Zoltan [DBH⁺02].

For each Ω^l_p, we define the set of boundary points $\partial\Omega^l_p$, and the set of inner points $\overset{\circ}{\Omega}{}^l_p$ by

$$\begin{aligned} \partial\Omega^l_p &= \left\{ i \in \Omega^l_p : \exists j \in \Omega^l_q,\ q \neq p \text{ such that } j \in S_i \right\}, \\ \overset{\circ}{\Omega}{}^l_p &= \Omega^l_p \setminus \partial\Omega^l_p. \end{aligned}$$

Note that we only consider strong couplings across boundaries here, i.e. a point $i \in \Omega^l_p$, which has a non-zero connection a_{ij} to a point $j \notin \Omega^l_p$, is still considered as an interior point if all of its strongly connected neighbors S_i are contained within Ω^l_p.
In Figure 3.1 we give a simple example of a finite difference discretization mesh distributed among four processors.
We assume that the matrix A is distributed row-wise, i.e. on all processors $p = 1, \ldots, np$ and for all $i \in \Omega_p$ the row $\mathbf{a}_i := (a_{ij})_{j=1}^n$ is stored on processor p. This storage is mostly

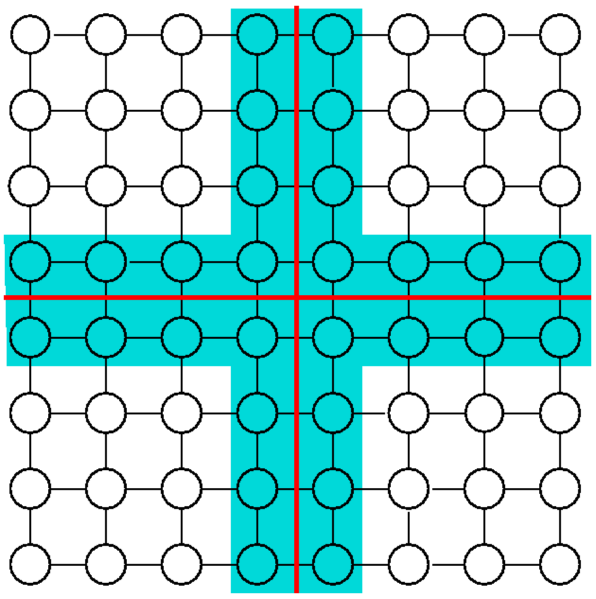


Figure 3.1.: Distribution of a 5 point finite difference grid among four processors. The red lines denote the processor boundaries, the blue area indicates the processor boundary points $\partial\Omega_p^l$.

done by means of a parallel *compressed sparse row (CSR)* matrix format as it is e.g. used in PETSc [BGMS97], *hypre* [CCF98] or Trilinos [HBH⁺05]. In these formats, the part of A that resides on processor p, here denoted by $A_{(p)}$, is organized in two blocks,

$$A_{(p)} = \begin{pmatrix} A_{diag} & A_{offd} \end{pmatrix} ,$$

where A_{diag} contains all local couplings a_{ij}, $i, j \in \Omega_p$, and A_{offd} contains all a_{ij}, $i \in \Omega_p$ and $j \notin \Omega_p$ and each of the A_{diag} and A_{offd} is a CSR matrix. Regarding the vectors u and f needed in the solution phase, each entry u_i and f_i is stored on the processor p with $i \in \Omega_p^l$.
For most components of the setup phase, it is obvious how a parallel computation can be carried out. To obtain the strong couplings for $i \in \Omega_p^l$, we only need access to the row $\mathbf{a}_i$. The same holds for the computation of the direct interpolation weights w_{ij}, $i \in F$, $j \in C$ (see Section 2.8.3). For other interpolation operators, we need access to the rows $\mathbf{a}_k$ corresponding to strongly coupled fine grid points $k \in F_i^s$ even if k resides on another processor. We obtain these rows in a single communication step before we start computing the interpolation weights, see Algorithm 3.1.

For multi-pass interpolation (Section 2.8.7) and Jacobi interpolation (Section 2.8.8), a communication of the updated weights is required between two iterations of the respective algorithm.

The computation of the coarse grid operator as it is carried out in *hypre* is outlined in Algorithm 3.2. Here, we do not transpose the prolongation P_l in parallel to obtain R_l, but instead exploit the fact that the blocks P_{diag}^l and P_{offd}^l are available. We take the transpose of each of these blocks individually, obtaining R_{diag}^l and R_{offd}^l. In addition,

algorithm 3.1 Parallel computation of the interpolation operator

1. On each processor $p = 1, \dots, np$, identify the fine grid points $F^{l,p} := F \cap \Omega_p^l$ and the strongly coupled remote fine grid points
$$F^s_{(p)} \leftarrow \bigcup_{i \in F^{l,p}} F_i^s \setminus \Omega_p^l.$$
2. For all $k \in F^s_{(p)}$, obtain the row $\mathbf{a}_k$.
3. Compute the interpolation weights for all $i \in F^{l,p}$ according to the chosen interpolation method (see Sections 2.8.4–2.8.6).

algorithm 3.2 AmgParallelRAP($A^l_{(p)}, P^l_{(p)}, A^{l+1}_{(p)}$) [hyp]

```
begin
    transpose $R^l_{diag} \leftarrow \left(P^l_{diag}\right)^T, R_{offd} \leftarrow \left(P^l_{offd}\right)^T$;
    communication: obtain $P^l_{ext} = \left(P^l_k\right)_k$ for all nonzero columns $k$ of $A^l_{offd}$;
    re-organize $P^l_{ext} = \begin{pmatrix} P^l_{ext,diag} & P^l_{ext,offd} \end{pmatrix}$;
    compute $RAP^l_{remote} \leftarrow R^l_{offd} \cdot A^l_{diag} \cdot P^l_{diag} + R^l_{offd} \cdot A^l_{diag} \cdot P^l_{offd}$
        $+R^l_{offd} \cdot A^l_{offd} \cdot P^l_{ext,diag} + R^l_{offd} \cdot A^l_{offd} \cdot P^l_{ext,offd}$;
    communication: distribute $RAP^l_{remote}$ among owning processors, receive $RAP^l_{ext}$;
    re-organize $RAP^l_{ext} = \begin{pmatrix} RAP^l_{ext,diag} & RAP^l_{ext,offd} \end{pmatrix}$;
    compute $A^{l+1}_{diag} \leftarrow R^l_{diag} \cdot A^l_{diag} \cdot P^l_{diag} + R^l_{diag} \cdot A^l_{offd} \cdot P^l_{ext,diag} + RAP^l_{ext,diag}$;
    compute $A^{l+1}_{diag} \leftarrow R^l_{diag} \cdot A^l_{diag} \cdot P^l_{offd} + R^l_{diag} \cdot A^l_{offd} \cdot P^l_{ext,diag} + RAP^l_{ext,offd}$;
    $A^{l+1}_{(p)} \leftarrow \begin{pmatrix} A^{l+1}_{diag} & A^{l+1}_{offd} \end{pmatrix}$;
end
```

for all non-zero columns k in A^l_{offd}, we need to communicate the respective row $\mathbf{p}^l_k$. We denote the sub-matrix consisting of these rows by P^l_{ext} and split it into $P^l_{ext,diag}$ and $P^l_{ext,offd}$ according to the column distribution of $P^l_{(p)}$ into P^l_{diag} and P^l_{offd}. Then, the triple matrix product on each processor can be written as a block triple matrix product,

$$A_{l+1} = \begin{pmatrix} R^l_{diag} \\ R^l_{offd} \end{pmatrix} \cdot \begin{pmatrix} A^l_{diag} & A^l_{offd} \end{pmatrix} \cdot \begin{pmatrix} P^l_{diag} & P^l_{offd} \\ P^l_{ext,diag} & P^l_{ext,offd} \end{pmatrix}.$$

We see that we also partially compute remote rows, i.e. parts of A^C that will not be stored on the local processor. These products (whose leftmost factor is R^l_{offd}) need to be distributed among the processors owning the respective rows and then need to be added to the locally computed rows there.
All these steps can be easily parallelized, though some communication is needed. The same is true for the multigrid cycle (Algorithm 2.1), where all of the communication is contained in the parallel matrix-vector and parallel norm computations. The only obstacle here is the Gauss-Seidel smoother (2.3), which is replaced by a block Jacobi smoothing scheme

$$M^l = I - \begin{pmatrix} Q^l_{(1)} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & Q^l_{(np)} \end{pmatrix}^{-1} \begin{pmatrix} A^l_{(1)} \\ \vdots \\ A^l_{(np)} \end{pmatrix}. \tag{3.2}$$

Letting $A_{diag} = L_{diag} + D_{diag} + U_{diag}$ be the decomposition of the diagonal part of $A_{(p)}$ into the lower triangular, diagonal and upper triangular part , we set $Q^l_{(p)} = L_{diag} + D_{diag}$. In terms of the scalar values u_i, we obtain the following relaxation scheme,

$$u_i^{it+1} = u_i^{it} + a_{ii}^{-1}\left(f_i - \sum_{j\notin\Omega^l_p} a_{ij}u_j^{it} - \sum_{j\in\Omega^l_p,\ j<i} a_{ij}u_j^{it+1} - \sum_{j\in\Omega^l_p, j\geq i} a_{ij}u_j^{it}\right). \tag{3.3}$$

This means that if for an $i \in \Omega^l_p$, all connections to remote points are zero, i.e. $a_{ij} = 0$ for all $j \notin \Omega^l_p$, then the value u_i is relaxed Gauss-Seidel like. The other points (at the processor boundaries) are updated rather additively. A communication step is required after each sweep is complete. It is, of course, also possible to use backward or symmetric sweeps inside each processor domain. For a more detailed discussion of parallel smoothers, we refer to [MY04, MY06].
The parallel creation of suitable coarse grids, however, is a major challenge in the process of constructing the AMG hierarchy. The coarse grid selection algorithm 2.5 cannot be carried out in parallel, as the following actions need to be performed within each iteration:

1. select a suitable undecided point i with maximal weight λ_i and add it to the coarse grid C,
2. assign all undecided neighbors j that strongly depend on i, $j \in S^T_i$ to the fine grid,

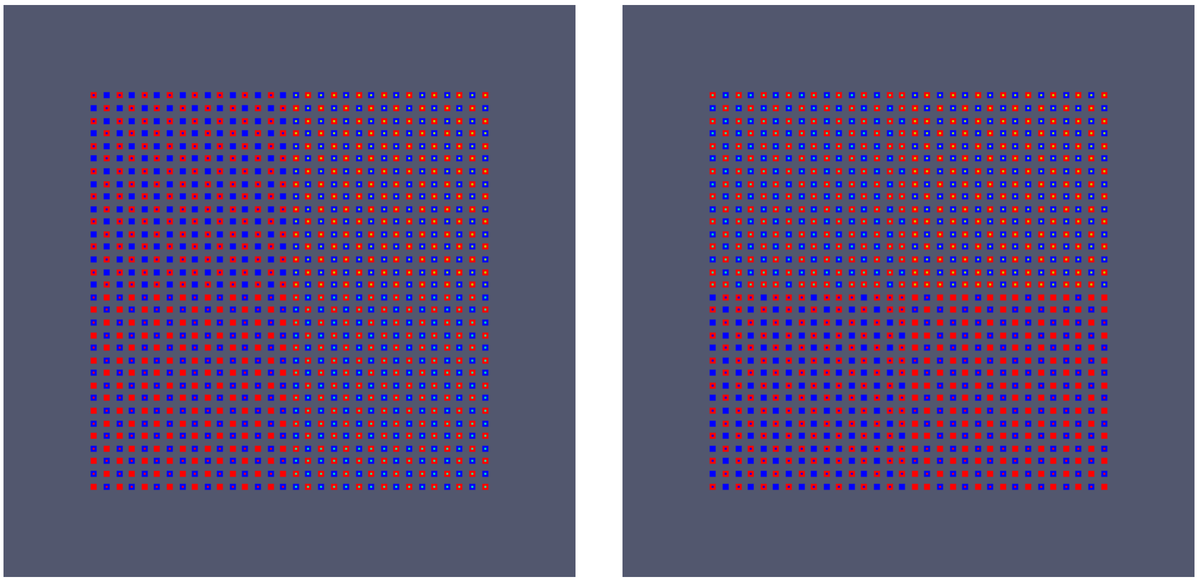

(a) Ruge-Stüben coarsening applied to each processor subdomain only (b) Third pass coarsening

Figure 3.2.: Ruge-Stüben coarsening and third pass coarsening applied to a 5-point discretization of the Laplace operator, distributed among 4 processors. Depicted are the C/F-splittings on the finest level, where the blue squares indicate the fine grid points $i \in F$, and the red squares indicate the coarse grid points $j \in C$.

3. for all undecided neighbors $k \in S_j$ of these newly created fine grid points, increase their weights λ_j,
4. for all strongly coupled undecided neighbors $j \in S_i$, decrease the weights λ_j.

We see that we need to update the weights λ_k of all undecided points two layers away from the newly chosen coarse grid point i, see also Figures 2.6(b)–2.6(d). This update (as well as the update of the status of i and the status of all newly created fine grid points $j \in S_i^T$) needs to be performed after *each* iteration. Even if this was feasible, we still had to deal with race hazards: e.g. process p would set point j to fine while process q would concurrently assign j to the coarse grid.
If we just apply Algorithm 2.5 on each processor subdomain Ω_p^l and do not change the status or weight of remote points $j \in \Omega_q^l$, $q \neq p$, we can obtain undesirable coarse grids, where condition C1 is not satisfied. For example, in Figure 3.2(a) we see that we have many strong couplings between fine grid points across the processor boundary, while these points do not share a common coarse grid point to interpolate from. Several approaches to parallel coarsening have been developed in the past. A first idea is to keep the classical Ruge-Stüben algorithm on each processor and to fix the inconsistencies described above. Two methods that follow this approach are described in Section 3.1. Other parallel coarsening schemes use parallel independent set algorithms to find a coarse

grid. We discuss them in Sections 3.2–3.3. We introduce our parallel AMG approach in Section 3.4. Our approach is based on Ruge-Stüben coarsening, but instead of fixing inconsistencies at the subdomain boundaries, our method (mostly) avoids unwanted situations. An extension of our method for very large supercomputers is outlined in Section 3.5. We also refer to [MY06] for an overview over parallel algebraic multigrid. As in Chapter 2, we omit the level index l if it is not needed.

Remark 3.1 As mentioned above, the parallel AMG algorithms presented here are designed for MIMD parallel computers. In the past years, the power of graphics processor units (GPUs) has increasingly been employed to accelerate computations on workstations as well as on computer clusters. These kind of processors however follow a *single instruction, multiple data (SIMD)* design, i.e. a single instruction can efficiently be applied to a large amount of data synchronously. Any efficient parallel algorithm for GPU computation needs to exploit this setup, which leads to different programming techniques compared to classical parallel AMG. A detailed description is beyond the scope of this chapter, we instead refer to [BDO12] for an approach to AMG on GPUs.

3.1. Minimum Subdomain Blocking, Third pass coarsening

We first address a class of parallel coarsening schemes that, based on the classical Ruge-Stüben coarsening algorithm, ensure a stable interpolation for all fine grid points by means of an additional coarsening step at the subdomain interfaces. While the *Minimum Subdomain Blocking (MSB)* scheme coarsens the boundary first and then proceeds to the interior, *Third Pass Coarsening (RS3)* first coarsens the interior and then fixes the boundaries.

Minimum Subdomain Blocking The Minimum Subdomain Blocking scheme [KS99] allows parallel coarsening with minimal communication between the processes. First, each processor p applies the classical Ruge-Stüben algorithm (first and optionally second pass) to the set of boundary points $\partial\Omega_p$, see Algorithm 3.3. Then, the interior of each subdomain is coarsened. Hence, each point of the boundary is either a coarse point or is strongly coupled to a coarse boundary point. All strong couplings which cross a

algorithm 3.3 AmgMSB($\Omega_p, S_{(p)}, S^T_{(p)}, C_p, F_p$)

```
begin
    AmgPhaseI($\partial\Omega_p, S, S^T, C_p, F_p$);
    AmgPhaseII($\partial\Omega_p, S, C_p, F_p$);
    AmgPhaseI($\Omega_p, S, S^T, C_p, F_p$);          where $C$ and $F$ are not newly initialized
    AmgPhaseII($\Omega_p, S, C_p, F_p$);
end
```

subdomain boundary are however ignored in this approach. On the one hand, this can lead to a unphysical coarse grid structure near a boundary if the direction of strength is orthogonal to the boundary (in this case, the MSB algorithm needs to assign every boundary point to the coarse grid as all connections within the boundary are weak). On the other hand, and even more severe, there is no possibility to check whether two strongly connected fine grid points $i \in \Omega_p$ and $j \in S_i \cap \Omega_q$, $q \neq p$ share a common strongly coupled coarse grid point. The resulting interpolation operator may not be stable even if an interpolation formula for all fine grid points can be found.

Third pass Coarsening The Third Pass (RS3) coarsening scheme [HMY02] is a straightforward parallel extension of the classical Ruge-Stüben coarsening scheme. In contrast to the subdomain blocking described before, a processor subdomain boundary treatment is applied after the coarsening of the interior is done.
First, each processor p applies both phases of the Ruge-Stüben algorithm to its subdomain Ω_p, including all of its boundary $\partial\Omega_p$. We note that we initialize the weights λ_i for the first phase to the global values, i.e. $\lambda_i := |S_i^T|$ even if S_i^T contains non-local points. For each processor p, we obtain a splitting into coarse grid points C_p and fine grid points F_p. While every $i \in F_p$ has at least one strong connection to a $j \in C_p$, there still may exist strong fine–to–fine $(F - F)$ couplings across the boundary for which the occurrence of a common C-point is not yet checked.
To remove these inconsistencies at the subdomain interfaces, first every processor p gathers the coarse/fine–splitting of its adjacent ghost points $\bigcup_{i\in\Omega_p} S_i \setminus \Omega_p$. Then, in a local step, processor p now applies the second pass (Algorithm 2.6) restricted to its boundary as well as to the ghost points,

$$\overline{\partial\Omega_p} := \partial\Omega_p \cup \left(\cup_{i\in\Omega_p} S_i \setminus \Omega_p\right).$$

We denote the set of coarse grid points generated in this third pass with $\tilde{C}_p$. Note that for a common interface $\overline{\partial\Omega_p} \cap \overline{\partial\Omega_q}$ the coarse grid points designated by processes p and q do not necessarily match. Hence after a second communication step, the following strategy decides on the final splitting: Every processor keeps the coarse grid points created by itself (i.e. C_p and $\tilde{C}_p$) and adds all coarse grid points $\tilde{C}_q \cap \Omega_p$ designated by lower rank processors $q < p$. Hence, the final set of coarse grid points for processor p is

$$C_p \cup \{\bigcup_{q \leq p} \tilde{C}_q \cap \Omega_p\}.$$

While a slight load imbalance is possible, this approach ensures a stable interpolation formula for all fine grid points throughout the domain. We sketch the RS3 approach in Algorithm 3.4. In Figure 3.2(b) we show an example of the RS3 coarsening algorithm applied to a 5-point discretization of the Laplacian. The additional points inserted after the third pass can be seen at the processor interfaces.

algorithm 3.4 AmgRS3(Ω_p, S, C_p, F_p)

```
begin
  $\tilde{C}_p = C_p$;
  AmgPhaseII($\overline{\partial\Omega}_p, S, \tilde{C}_p, F_p$);
  for $q > p$ do
      send $\tilde{C}_p \cap \Omega_q$ to processor $q$;                communication
  od
  for $q < p$ do
      receive $\tilde{C}_q \cap \Omega_p$ from processor $q$;           communication
      $C_p \leftarrow \tilde{C}_p \cup \tilde{C}_q \cap \Omega_p$;
  od;
end
```

3.2. CLJP coarsening schemes

In contrast to the previously described methods, the CLJP coarsening scheme [HMY02] is an a priori parallel coarsening algorithm. The main idea is to form the coarse grid as the union of multiple, in parallel constructed, independent sets $D \subset \Omega$. To describe this algorithm, we first write the strong couplings as an influence matrix $S = (S_{ij})_{i,j=1}^{N}$, where

$$S_{ij} := \begin{cases} 1 & \text{if } j \in S_i, \\ 0 & \text{otherwise.} \end{cases} \tag{3.4}$$

This matrix defines a directed graph on the set of points Ω. As in the Ruge-Stüben case, we assign a weight w_i to each node $i \in \Omega$ and initialize it such that it measures the "usefulness" of this point as a coarse point, i.e. $w_i = |S_i^T|$. The CLJP algorithm, however, requires adjacent nodes to have different weights. To this end, we add a random component $\sigma_i \in [0, 1)$ and obtain $w_i = |S_i^T| + \sigma_i$.
We now construct the independent set D. A vertex i of the graph is added to D if its weight w_i is larger than the weights w_j of all adjacent neighbors $j \in S_i \cup S_i^T$. Then, the weights of all vertices $j \notin D$ are modified using the following heuristics,

H1 D-points, i.e. newly created coarse grid points, do not require interpolation. If j influences a D-point i, we reduce the weight w_j by one and remove the edge S_{ij} from the graph.

H2 If both k and j depend on a D-point i ($i \in S_j \cap S_k$) and k depends on j ($j \in S_k$), we reduce w_j by one and remove S_{kj} from the graph, as the value at k can be interpolated from i directly.

If the weight of a vertex is less than one, it is not needed for interpolation. We remove this vertex from the graph and add the corresponding point to the set of fine grid points F. After the heuristics have been applied to all undecided nodes j, we add D to C and remove all vertices in D as well as all incident edges from the graph. Now, a

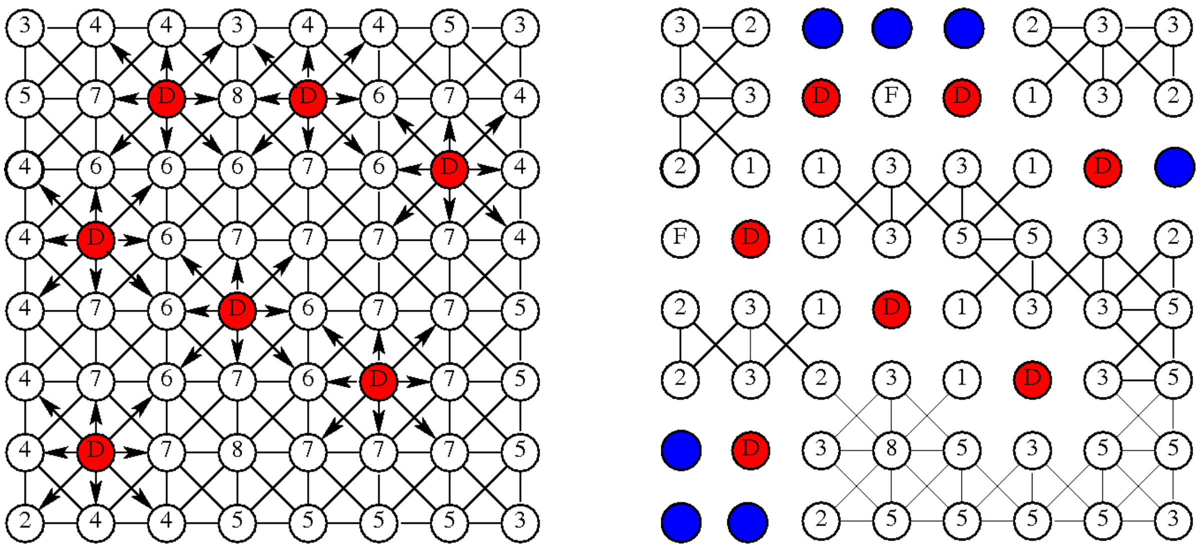


Figure 3.3.: Discretization grid of a 9-point stencil after application of the first (left) and second (right) heuristics of the CLJP algorithm. The numbers indicate the weights w_i (without the added random number). Arrows indicate the direction of influence, lines without arrays indicate bidirectional influence. Fine grid points are blue, D-points red.

communication step is needed to update the weights and the current C/F distribution. Then, we continue choosing a new set D until the whole domain is split into C and F.

In Figure 3.3 we show the meshes of a 9-point stencil after the application of both heuristics, respectively. We give the overall CLJP program in Algorithm 3.5. The CLJP algorithm runs entirely in parallel. Moreover, if the random number assignment does not depend on the domain decomposition, the C/F-splitting will also be independent of the parallel layout. Furthermore, depending on the fine grid matrix (direction of anisotropy/diffusion coefficients etc.) the CLJP process can coarsen up to a single point without the need for some subdomain agglomeration on coarser levels. As a neighbor of a coarse grid point is, however, not forced to become a fine grid point, this method produces denser coarse grids than the Ruge-Stüben algorithm. This can e.g. be seen from Figure 3.4(a).

CLJP-c coarsening The CLJP-in-color (CLJP-c) [Alb06, Alb07] approach is a modification of the CLJP algorithm which aims at overcoming the issues described above. First, we observe that the results of the CLJP algorithm depend heavily on the random numbers $\sigma(i)$. These random numbers are needed for tie-breaking only, so the question arises whether one could utilize another tie-breaking mechanism that employs information about the graph.

Here, the key idea is to first color the graph, i.e. to assign a color to each node such that two adjacent nodes do not have the same color. Now, we assign a weight $c_w = (c-1)/|c_l| \in [0,1)$ to each color c and add it to the vertex weight, i.e. in Algorithm

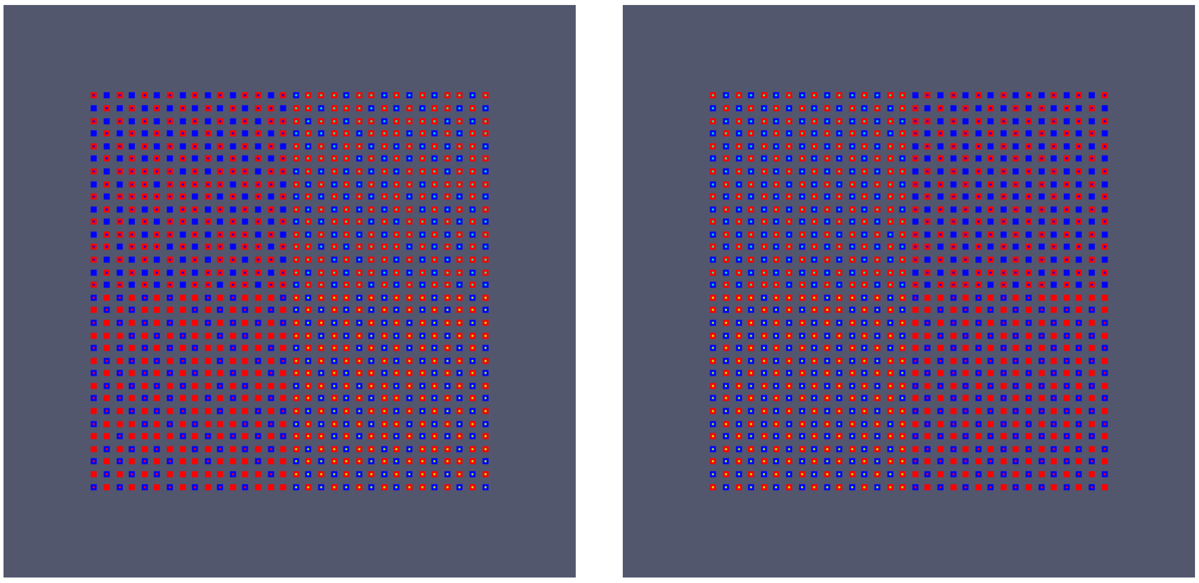

(a) CLJP coarsening (b) Falgout coarsening

Figure 3.4.: CLJP and Falgout coarsening schemes applied to a 5-point discretization of the Laplace operator, distributed among 4 processors. Depicted are the C/F-splittings on the finest level, where the blue squares indicate the fine grid points $i \in F$ and the red squares the coarse grid points $j \in C$.

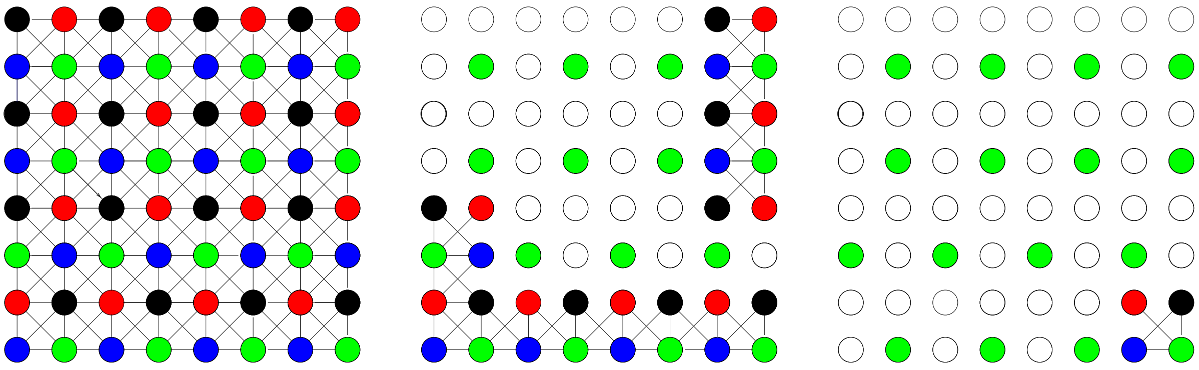


Figure 3.5.: CLJP-c coarsening for a 9-point stencil. We assume that the green color corresponds to the highest augmentation weight (3.5)

algorithm 3.5 CLJP algorithm $\mathsf{CLJP}(\Omega, S, S^T, C, F, D)$

```
begin
  $C \leftarrow \emptyset$;
  $F \leftarrow \emptyset$;
  for $i \in \Omega$ do
      $w(i) \leftarrow |S_i^T| + \sigma(i)$;
  od;
  while $C \cup F \neq \Omega$ do
        if $D = \emptyset$
          then
                for $i \in \Omega$ do
                    if $w(i) > w(j)$ for all $j \in S_i \cup S_i^T$
                      then
                            $D \leftarrow D \cup \{i\}$;
                    fi;
                od;
        fi;
        for $i \in D$ do
            for $j \in S_i$ do                                        heuristics H1
                $w(j) \leftarrow w(j) - 1$;
                $S_i \leftarrow S_i \backslash \{j\}$;
            od;
            for $j \in S_i^T$ do                                      heuristics H2
                for $k \in S_j^T$ do
                    if $k \in S_i^T$
                      then
                            $w(j) \leftarrow w(j) - 1$;
                            $S_j^T \leftarrow S_j^T \backslash \{k\}$;
                    fi;
                od;
                $S_i^T \leftarrow S_i^T \backslash \{j\}$;
            od;
        od;
        for $i \in \Omega$ do
            if $w(i) < 1$ then $F \leftarrow F \cup \{i\}$ fi;
        od;
        $C \leftarrow C \cup D$;
        $D \leftarrow \emptyset$;
  od;
end;
```

3.5 we replace the step

$$w(i) = |S_i^T| + \sigma(i)$$

by

$$w(i) = |S_i^T| + c_w(color(i)). \tag{3.5}$$

Here, c_l denotes the set of all colors (represented by integers) in the graph. Figure 3.5 shows the coloring of a nine-point stencil. We assume that the green nodes have obtained the highest augmentation and see that after the first CLJP iteration, most of the interior is already partitioned into coarse and fine points.

algorithm 3.6 AmgFalgout(Ω, S, S^T, C, F)

```
begin
    AmgPhaseI($\Omega_p, S, S^T, C_p, F_P$);        where the $\lambda_i$ are initialized globally
    AmgPhaseII($\Omega_p, S, C_p, F_p$);
    $D_p \leftarrow C_p \cap \overset{\circ}{\Omega}_p$;
    $D = \bigcup_{p=1}^{np} D_p$;        no communication, each processor just manages its own part of $D$
    $C \leftarrow \emptyset$;
    $F \leftarrow \emptyset$;
    AmgCLJP($\Omega, S, S^T, C, F, D$);        parallel algorithm
end
```

Falgout's coarsening scheme Falgout's coarsening process [HMY02] is a hybrid coarsening scheme that comprises the classical Ruge-Stüben algorithm as well as the CLJP algorithm.
Again, like the RS3 algorithm, every processor applies the first and optionally second pass of the Ruge-Stüben algorithm to its local subdomain (with global weight initialization). The resulting coarse grid points in the interior $\overset{\circ}{\Omega}_p = \Omega_p \setminus \partial\Omega_p$ then form the first independent set for the CLJP algorithm, i.e. $D = C \cap \cup_p \overset{\circ}{\Omega}_p$. This special initialization eliminates most inner edges of the graph restricted to $\overset{\circ}{\Omega}_p$ in the first CLJP iteration. Thus, only few additional coarse grid points will be added to the interior during further CLJP steps. Hence, the coarse grid structure there is very similar to a Ruge-Stüben one. Near the boundaries $\partial\Omega_p$, however, the coarse grid is completely constructed by the CLJP algorithm.
On coarser levels, if the ratio of boundary to interior points increases, the coarsening becomes more and more CLJP-like. Again, it is possible to coarsen up to a single point for appropriate operators. We give the pseudo-code of Falgout's method in Algorithm 3.6. In Figure 3.4(b), we show an example of the Falgout algorithm applied to a five-point stencil. We clearly see a denser coarse grid near the boundaries, while the interior of each subdomain is coarsening Ruge-Stüben like.

3.3. PMIS and HMIS coarsening schemes

We continue with another coarsening method based on parallel maximal independent set techniques. The PMIS coarsening scheme is a simplification of the CLJP algorithm, while the HMIS algorithm is its respective counterpart of Falgout's coarsening method [SMYH06].
The aim of the algorithms is to produce smaller coarse grids than the CLJP counterparts. As in the case of Ruge-Stüben based aggressive coarsening (Section 2.7), we relax condition C1 and only require that condition C1' holds:

C1' Let $i \in F$. Then there exists at least one $j \in S_i \cap C$.

Again, we define the influence matrix $S = (S_{ij})_{i,j=1}^{N}$, where

$$S_{ij} := \begin{cases} 1 & \text{if } j \in S_i, \\ 0 & \text{otherwise,} \end{cases}$$

and construct the *undirected* graph $G = (V, E)$ where

$$V = \Omega \text{ and } E = \{(i,j) \in V \times V \mid S_{ij} = 1 \text{ or } S_{ji} = 1\}.$$

As in the CLJP method (Algorithm 3.5), we introduce vertex weights $w_i = |S_i^T| + \sigma(i)$, where $\sigma(i) \in [0,1)$ is a uniformly distributed random number. First, we eliminate all isolated points from the graph, i.e. all $i \in V$ that satisfy $|S_i^T| = 0$. Then, we proceed to the main PMIS iteration: We choose an independent set I from G, where $i \in I$ if $w(i) > w(j)$ for all $j : (i,j) \in E$. I is then added to the set of coarse grid points C, while all strongly dependent points $\{j \in V \subset I \mid \exists i \in I : (j,i) \in E\}$ join the set of fine grid points F. The newly created coarse and fine grid points are removed from the graph and the iteration starts over, see Algorithm 3.7.

The main difference to the CLJP algorithm is that strongly influenced neighbors of newly created coarse grid points are immediately assigned to the set of fine grid points. While this is also done in the classical Ruge-Stüben scheme, here no weights are updated and hence strongly connected —yet undecided— points of these fine grid points are not more likely to become coarse grid points. Moreover, the PMIS algorithm does not include a second pass that checks whether two strongly connected fine grid points share a common C-point to interpolate from. In consequence, interpolation may suffer from a loss of stability and either long-range interpolation schemes (like extended interpolation, see Section 2.8.6) must be used or the resulting AMG hierarchy can only be used as preconditioner for a Krylov subspace iteration. On the other hand, the PMIS algorithm produces sparse coarse grids (see e.g. Figure 3.6(a)) and is a truely parallel algorithm that can construct a coarse grid regardless of the domain decomposition.

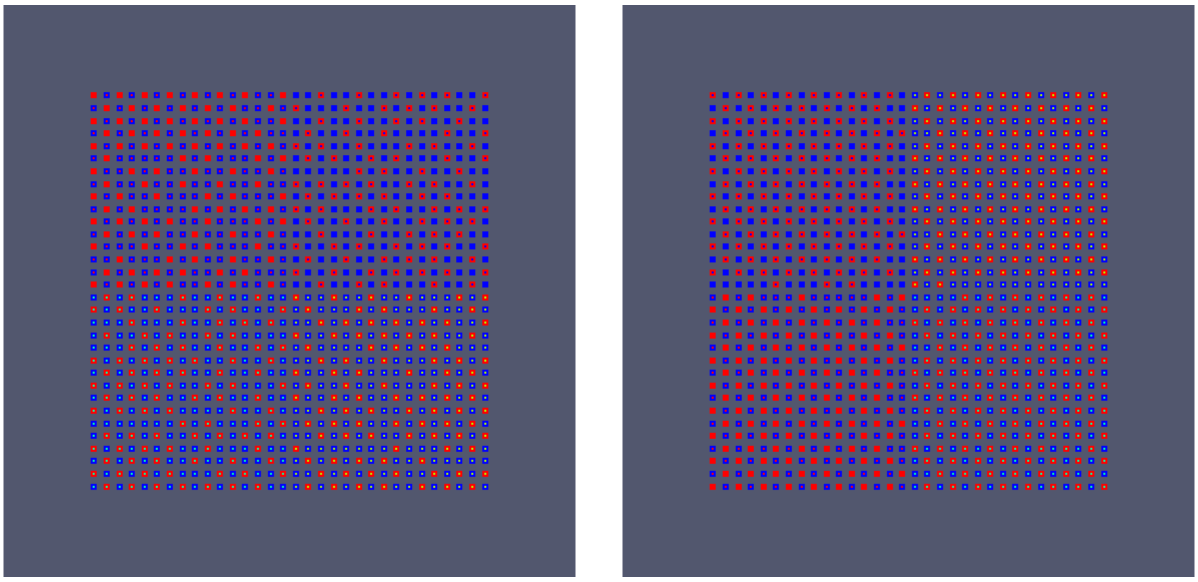
Analogous to Falgout's coarsening method, the HMIS coarsening scheme is a combination of classical Ruge-Stüben coarsening and the PMIS algorithm. In this case, only the first pass of the Ruge-Stüben algorithm is carried out to obtain the first independent set for the PMIS algorithm. The second pass does not make sense here, as the

algorithm 3.7 PMIS algorithm AmgPMIS$(\Omega, S, S^T, C, F, I)$

```
begin
  $C \leftarrow \emptyset$;
  $F \leftarrow \{j \in \Omega : \; S_j^T = \emptyset\}$;
  $V = \Omega \setminus F$;
  $E = \{(i,j) \in V \times V : \; i \in S_j$ or $j \in S_j\}$;
  for $i \in V$ do
      $w(i) = |S_i^T| + \sigma(i)$;
  od;
  while $C \cup F \neq \Omega$ do
        if $I == \emptyset$
          then
               for $i \in V$ do
                   if $w(i) > w(j)$ for all $j \in V : \; (i,j) \in E$
                     then
                          $I \leftarrow I \cup \{i\}$;
                   fi;
               od;
        fi;
        for $i \in D$ do
            for $j \in S_i^T \cap V \subset I$ do
                $F \leftarrow F \cup \{j\}$;
                $V \leftarrow V \setminus \{j\}$;
            od;
        od;
        $C \leftarrow C \cup I$;
        $V \leftarrow V \setminus I$;
        $I \leftarrow \emptyset$;
  od;
end;
```

algorithm 3.8 AmgHMIS(Ω, S, S^T, C, F)

```
begin
  AmgPhaseI$(\Omega_p, S, S^T, C_p, F_P)$;          where the $\lambda_i$ are initialized globally
  $I_p \leftarrow C_p \cap \overset{\circ}{\Omega}_p$;
  $I = \bigcup_{p=1}^{np} I_p$;        no communication, each processor just manages its own part of $I$
  $C \leftarrow \emptyset$;
  $F \leftarrow \emptyset$;
  AmgPMIS$(\Omega, S, S^T, C, F, I)$;                                     parallel algorithm
end
```



(a) PMIS coarsening (b) HMIS coarsening

Figure 3.6.: PMIS and HMIS coarsening schemes applied to a 5-point discretization of the Laplace operator, distributed among 4 processors. Depicted are the C/F-splittings on the finest level, where the blue squares indicate the fine grid points $i \in F$ and the red squares the coarse grid points $j \in C$.

PMIS scheme only aims to enforce condition C1', not C1. Again, we initialize the first independent set $I = C \cap \cup_p \overset{\circ}{\Omega}_p$ using the interior coarse grid points constructed by the Ruge-Stüben algorithm and proceed with the PMIS iteration. The HMIS method is outlined in Algorithm 3.8, an example is shown in Figure 3.6(b). We clearly see that the coarsening at the processor boundaries is sparser than in the interior (in contrast to Falgout's coarsening algorithm, where the opposite is true, cf. Figure 3.4(b)).

Remark 3.2 Aggressive coarsening variants of the PMIS and HMIS algorithms are described in [MY10]. As in the case of Ruge-Stüben aggressive coarsening, these algorithms employ the concept of strong n-connections (Definition 2.8). The *strength matrix* S^m is then defined as follows,

$$S_{ij}^m := \begin{cases} q & \text{if } i \text{ depends on } j \text{ w.r.t. } (q,m), \\ 0 & \text{otherwise.} \end{cases}$$

In practice, usually the cases $l = 2$ and $p = 1$ or $p = 2$ are used. For $p = 1$, the coarse grid can be constructed by two subsequent applications of PMIS/HMIS to the strength matrix S (3.4). The grid obtained by the first run (denoted by $\tilde{C}$) serves as input for the second application, where the vertex weights are now derived from the matrix $S^2 \cap \tilde{C}$. Note that in the second stage, isolated points may not be removed first, as they could serve as the only interpolation point for a whole region within the domain. Like in

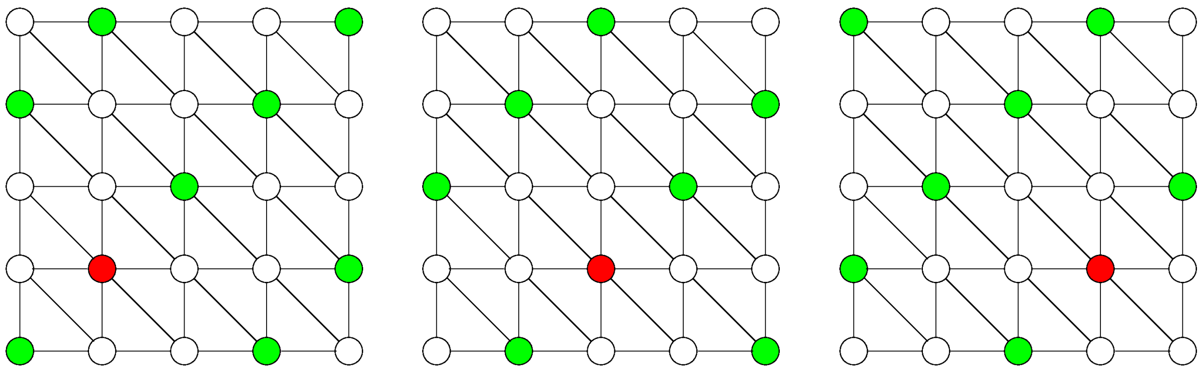


Figure 3.7.: Resulting coarse grids for the problem with mixed derivatives from Example 2.2 constructed by three different initial choices. The green points indicate the respective coarse grid points, the red point indicates the first coarse grid point chosen.

the sequential case, aggressive coarsening requires long-range interpolation techniques to ensure an accurate coarse grid correction.

Remark 3.3 We note that, like in the CLJP-c case, it is also possible to use a graph coloring instead of random numbers for tie-breaking [Alb07].

Remark 3.4 Parallel maximal independent set algorithms are also used within parallel smoothed aggregation algorithms (Section 2.11.2), see [TT00].

3.4. CGC coarsening

In this section, we describe the coarse grid classification (CGC) algorithm introduced in [Met04, GMOS06, GMS06]. As with the RS3 and MSB methods, we employ the classical Ruge-Stüben coarsening algorithm. Here, instead of repairing the coarse grid structure at the processor interfaces, we create coarse grids that (mostly) match at these boundaries. The remaining inconsistencies can be removed by a simplified version of the RS3 boundary fixing mechanism.

In our parallel coarsening method, we exploit a characteristic of the Ruge-Stüben algorithm that is closely related to its sequential nature. We recall that this classical algorithm determines the coarse grid points in dependence of the previously selected points. Hence, we can guide the coarsening process by changing the initial choice for the first coarse grid point. In Figure 3.7 we give an example of three different coarse grids resulting from different initial choices. Note that in the sequential case, the quality of the resulting coarse grids with respect to multigrid convergence and memory requirements is very similar. Hence, there is no special advantage of using either one of these coarse grids in a sequential computation. On the other hand, we have an additional degree of freedom in our coarse grid selection process. We can use this freedom to compose a

global coarse grid from the coarse grids we have constructed for each processor subdomain Ω_p individually.
More precisely our approach works as follows: First, we independently construct multiple coarse grids on each processor domain by running the classical algorithm multiple times with different initial coarse grid points. Note that this procedure is computationally efficient, since the classical Ruge–Stüben algorithm requires only a very small amount of compute time compared with the construction of the transfer and coarse grid operators, while a well-constructed grid can save a large amount of time during the operator construction and in the multigrid cycle.
After the construction of these coarse grids on all processors, we need to select exactly one grid for each processor domain such that the union of these coarse grids forms a suitable coarse grid for the whole domain. We achieve this by defining a weighted graph whose *vertices* represent the *grids* constructed by the multiple coarsening runs. *Edges* are defined between vertices which represent grids on neighboring processor domains. Each edge weight measures the quality of the boundary constellation if these two grids are chosen to be part of the composed grid. Finally, we use this graph to choose one coarse grid for each processor subdomain which automatically matches with most of its neighbors.

In our implementation, see Algorithm 3.9, each processor p first determines the maxi-

algorithm 3.9 CGC algorithm $CGC(S, S^T, ng, \{C_i\}_{i=1}^{ng}, \{F_i\}_{i=1}^{ng})$

```
for $j \leftarrow 1$ to $|\Omega|$ do $\lambda_j \leftarrow |S_j^T|$; od;
$C_0 \leftarrow \emptyset$;  $\lambda_{\max} \leftarrow \arg\max_{k\in\Omega} \lambda_k$;
do
    $U \leftarrow \Omega \setminus \bigcup_{i\le it} C_i$;
    if $\max_{k\in U} \lambda_k < \lambda_{\max}$ then break; fi;
    $it \leftarrow it + 1$;  $F_{it} \leftarrow \emptyset$;  $C_{it} \leftarrow \emptyset$;
    do
        $j \leftarrow \arg\max_{k\in U} \lambda_k$;
        if $\lambda_j = 0$ then break; fi;
        $C_{it} \leftarrow C_{it} \cup \{j\}$;  $\lambda_j \leftarrow 0$;
        for $k \in S_j^T \cap U$ do
            $F_{it} \leftarrow F_{it} \cup \{k\}$;  $\lambda_k \leftarrow 0$;
            for $l \in S_k \cap U$ do $\lambda_l \leftarrow \lambda_l + 1$; od;
        od;
        for $k \in S_j \cap U$ do $\lambda_k \leftarrow \lambda_k - 1$; od;
    od;
od
$ng \leftarrow it$;
```

mal weight $\lambda_{\max}$ of all points $i \in \Omega_p$. As mentioned earlier, every point with this weight can be chosen as an initial point for the classical coarsening algorithm. We choose one particular point $\tilde{i}$ and construct a coarse grid $C_{(p),1}$. We now re-initialize the weights

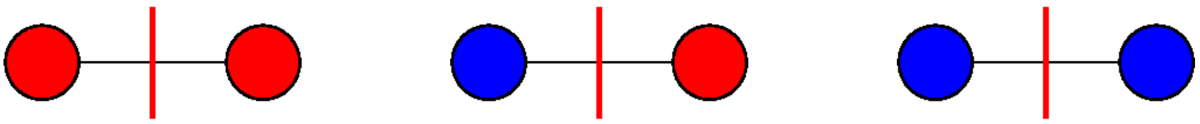


Figure 3.8.: Three possible C/F-constellations at a processor's domain boundary. The red points belong to C, the blue points belong to F.

$\lambda_i := |S_i^T|$ of all remaining points $i \in \Omega \setminus C_{(p),1}$ to their original values. From these points, we select another point $\tilde{j}$ with weight $\lambda_{\max}$ and construct a second coarse grid $C_{(p),2}$ starting with this point. Only points not contained in $C_{(p),1}$ may be inserted into $C_{(p),2}$, i.e. we construct disjoint coarse grids. We repeat these steps as long as there is a point with weight $\lambda_{\max}$ that is not already a member of a coarse grid $C_{(p),it}$. Note that the number of iterations is bounded by the maximal number of strong couplings $|S_i|$ over all points $i \in \Omega_p$, which in turn is bounded by the maximal stencil width. Hence, the number of constructed grids ng_p per processor p is independent of the number of unknowns N and the number of processors np. Note that the coarse grids that are constructed later may be of inferior quality compared to the first ones but the selection mechanism described in the following will avoid these grids.[1]
Now, we have obtained ng_p valid coarse grids $\{C_{(p),i}\}_{i=1}^{ng_p}$ on each processor p. To determine which grid to choose on each processor, we construct a directed, weighted graph $G = (V, E)$ whose vertices represent the created coarse grids,

$$V_p := \{C_{(p),i}\}_{i=1,\dots,ng_p}, \quad V := \bigcup_{p=1}^{np} V_p.$$

The set of edges E consists of all pairs (v, u), $v \in V_p$, $u \in V_q$ such that $q \in \mathcal{S}_p$ is a neighboring processor of p,

$$E_p := \{\bigcup_{q \in \mathcal{S}_p} \bigcup_{v \in V_p,\ u \in V_q} (v, u)\}, \quad E := \bigcup_{p=1}^{np} E_p,$$

where $\mathcal{S}_p$ is defined as the set of processors q with points j which strongly influence points i on processor p, i.e.

$$\mathcal{S}_p := \{q \neq p :\ \exists i \in \Omega_p,\ j \in \Omega_q :\ j \in S_i\}.$$

To determine the weight $\gamma(e)$ of the edge $e = (v, u)$, we consider the nodes $v \in V_p$, $u \in V_q$. Each of these particular nodes represents a local C/F-splitting (C_p, F_p) for Ω_p and (C_q, F_q) for Ω_q, respectively. Together they form a C/F-splitting for the domain $\Omega_p \cup \Omega_q$. At the processor subdomain boundary, three grid configurations can occur,

[1]The initially chosen point $\tilde{i}$ still has some influence on the constructed coarse grids due to our disjoint construction. This may lead to some matching problems in applications where the coarse grids have special structure, e.g. strong anisotropies.

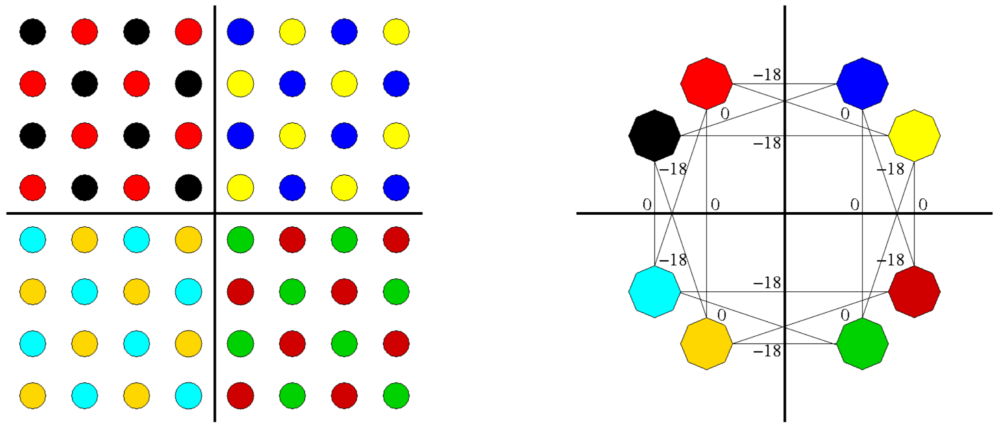


Figure 3.9.: Application of the CGC algorithm to a 5-point stencil, distributed among 4 processors. The figure on the left shows the assignment of the points to the coarse grids. The figure on the right shows the weighted graph.

see Figure 3.8. We denote by $c_{C,C}$ the number of strong $C-C$-couplings (left), by $c_{C,F}$ the number of strong $C-F$-couplings (center) and by $c_{F,F}$ the number of strong $F-F$-couplings (right). Based on these classes of couplings, we define the edge weight

$$\gamma(e) := c_{C,C}\gamma_{C,C} + (c_{C,F} + c_{F,C})\gamma_{C,F} + c_{F,F}\gamma_{F,F}$$

with $\gamma_{C,C}$, $\gamma_{C,F}$, $\gamma_{F,F} \in \mathbb{R}$ defined as follows: The most important case is the $F-F$-coupling case. Here, two fine grid points $i \in F_p$ and $j \in F_q$ are strongly coupled, which can lead to two problems: These two points may not have a common C-point to interpolate from, which violates condition (C1). On the other hand, even if (C1) is satisfied, we have to transfer the matrix rows $\mathbf{a}_i$ and $\mathbf{a}_j$ to construct a stable interpolation operator. Therefore, this situation must be avoided, which motivates us to penalize strong $F-F$-couplings with a large negative weight $\gamma_{F,F} := -8$.
The strong $C-C$-couplings should also be avoided because they can increase the operator and the grid complexity. We therefore set $\gamma_{C,C} := -1$. In the remaining case, which can be considered as the (optimal) sequential coarsening scenario, we do not add an additional weight, i.e. $\gamma_{C,F} := 0$.

Figure 3.9 shows the graph $G = (V, E)$ obtained by our CGC algorithm for a 5-point discretization of the Laplacian. We can observe that constellations with $C-C$–couplings and $F-F$–couplings are heavily penalized, while constellations with $C-F$–couplings are weighted by zero only.
Since we now have constructed the graph G of admissible local grids, we can use it to choose a particular coarse grid for each processor such that the union of these local grids automatically matches at subdomain boundaries. Observe that the number of vertices is related to the number of processors np only; i.e., it is much smaller than the number of unknowns N. Furthermore, the cardinality of E is small compared to N since edges are only constructed between neighboring processors. Thus, we can transfer the whole graph onto a single processor without large communication costs.

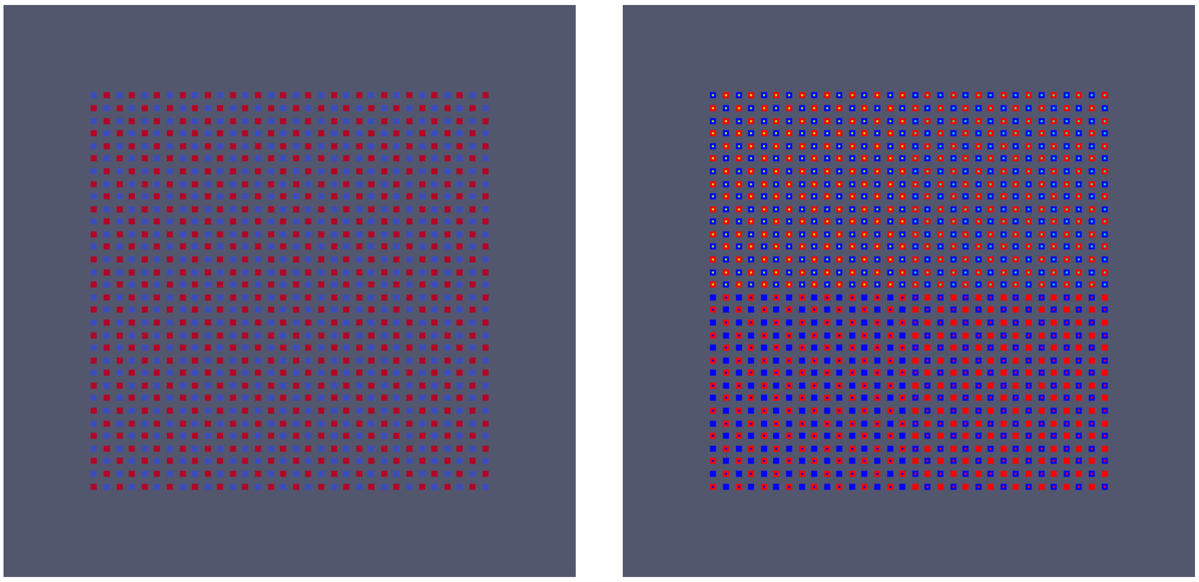

(a) Sequential coarsening on one processor (b) CGC coarsening

Figure 3.10.: CGC coarsening applied to a 5-point discretization of the Laplace operator, distributed among 4 processors (right), compared to sequential coarsening on one processor (left). Depicted are the C/F-splittings on the finest level, where the blue squares indicate the fine grid points $i \in F$ and the red squares the coarse grid points $j \in C$.

On this processor, we choose exactly one node v_p from each subset $V_p \subset V$ with the following scheme: We denote by $\mathcal{C}$ the set of the selected local coarse grids.

1. First, we define *heavy edges or couplings* H_v between the nodes v of the graph, where p denotes the processor which v belongs to (i.e. $v \in V_p$),

   $$H_v := \cup_{q \in \mathcal{S}_p} \{w \mid \gamma(v,w) = \max_{u \in V_q} \gamma(v,u)\} \text{ and } H_v^T := \{w \mid v \in H_w\}.$$

   The heavy edges indicate which coarse grid on processor q can be fitted best to the coarse grid represented by $v \in V_p$. We assign a weight λ_v to each node v, where $\lambda_v = |H_v| + |H_v^T|$. This weight indicates how many coarse grids on other processors can be fitted to the coarse grid represented by v.

2. For some processors p, all nodes $v \in V_p$ might have weight $\lambda_v = 0$. As this means that there are no strong couplings across the subdomain boundary, *any* grid constructed on this processor can be chosen. Here, we choose one arbitrary $v \in V_p$ and remove V_p from the graph.

3. We choose the node $v \in V_p$ with maximal weight, put it into $\mathcal{C}$ and remove the subset V_p from the graph, as a coarse grid for domain Ω_p is now determined. We then increase the weight of each node $w \in H_v \cup H_v^T$ to the maximal weight of all remaining nodes in the graph plus one (so that one of these will be chosen in the next step), and repeat this step as long as the graph is not empty.

This procedure takes up to *np* steps, one for each processor domain, see Algorithm 3.10 for details. After running the algorithm, we transfer the choice $v_p \in \mathcal{C} \cap V_p$ back to processor p. Now, the union of all elements in $\mathcal{C}$ defines the global consistent grid for the complete domain, see Figure 3.10(b). Here, this grid does not differ from the coarse mesh produced by the sequential Ruge-Stüben coarsening (Algorithm 2.5) applied to the whole domain.
Recall that after applying the first phase of the Ruge–Stüben coarsening, there may exist a few fine grid points with strong connections to other fine grid points only. These strong couplings are however very rare. In the sequential phase, this is corrected by the second pass of the classical coarsening scheme. To correct these very few couplings across processor boundaries, we employ a more straightforward method: We check whether a fine grid point is strongly coupled to other fine grid points only and insert this point into the coarse grid if that is the case. Hence, the CGC algorithm (essentially) employs no special boundary treatment.

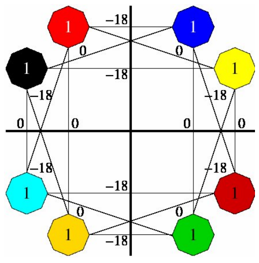
3.5. Outlook: CGC-ML coarsening

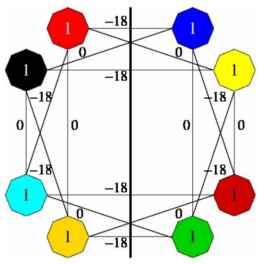
The contents of the following section have been previously published in [GMS08].
The main advantage of CGC over other parallel AMG coarsening schemes is that the constructed coarse grids are very close to those produced by a sequential AMG. On the other hand, the original CGC has one major drawback: The graph representing the

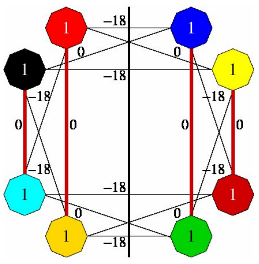
algorithm 3.10 AmgCGCChoose$(V, H, \mathcal{C})$

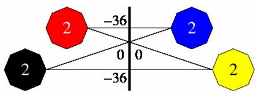
```
begin
  $\mathcal{C} \leftarrow \emptyset$;
  $\mathcal{U} \leftarrow V$;
  for $v \in \mathcal{U}$ do $\lambda_v \leftarrow |H_v| + |H_v^T|$;  od;
  for $p \leftarrow \{1, \dots, np\}$ do
     if $\lambda_v = 0$ for all $v \in V_p$
        then
           $\mathcal{C} \leftarrow \{v\}$;                                   arbitrary $v \in V_p$
           $\mathcal{U} \leftarrow \mathcal{U} \setminus V_p$;
     fi;
  od;
  while $\mathcal{U} \neq \emptyset$
        do
        $v \leftarrow \arg\max_{w \in \mathcal{U}} \lambda_w$;
        $\mathcal{C} \leftarrow \mathcal{C} \cup \{v\}$;
        $\mathcal{U} \leftarrow \mathcal{U} \setminus V_p$ such that $v \in V_p$;
        $\lambda_{\max} \leftarrow \max_{w \in \mathcal{U}} \lambda_w$;
        for $w \in \left(H_v \cup H_v^T\right) \cap \mathcal{U}$ do $\lambda_w \leftarrow \lambda_{\max} + 1$;  od;
  od;
end
```

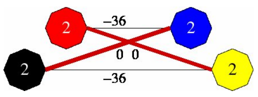
candidate coarse grids needs to be transferred to a single processor. For large numbers of processors ($np \gtrapprox 1000$) this leads to large communication costs as well as a significant run-time for the coarse grid selection Algorithm 3.10.
In this section, we outline an alternative which bypasses this costly communication step. To this end, we reduce the size of the graph by a coarsening process that collapses vertices which already represent well-matching grids, i.e. we construct a multilevel hierarchy of graphs. This coarsening process can be carried out in parallel and only requires communication inside small subsets of all involved processors. On the coarse levels of this hierarchy, the vertices now represent candidate coarse grids on unions of the processor subdomains. As before, the edge weights penalize connections between non-matching grids.
As the graph becomes coarser, we agglomerate it on fewer and fewer processors. Finally, we can choose a single vertex in the coarsest graph that represents a candidate coarse grid on a large part (or even all) of the computational domain. By this choice, all candidate coarse grids which were recursively collapsed into this vertex are designated to be a part of the global coarse grid.
In the following, we will describe this algorithm in more detail. We construct the graph $G = (V, E)$ as described in the last section. In addition, we assign a weight to each vertex which denotes the number of processor subdomains covered by the coarse grid represented by this vertex. Naturally, this weight is initialized with 1.
We do *not* transfer the whole graph onto a single processor. Instead, we proceed as


(a) original graph


(b) after agglomeration


(c) heavy edge matching


(d) collapsing vertices and edges


(e) second agglomeration step

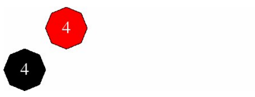
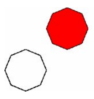
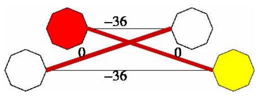
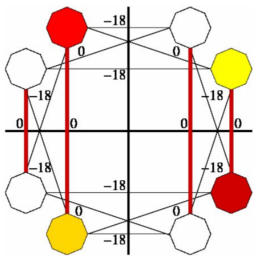

(f) final collapsed graph

Figure 3.11.: Graph clustering process. The graph is constructed by the CGC algorithm to a 5-point finite-difference discretization of the Laplace operator distributed among four processors (cf. Figure 3.9). The numbers in the vertices denote the number of subdomains covered by the coarse grid, which is represented by the respective vertex. The number at each edge denotes the edge weight.


(a) Initial choice


(b) First refinement step


(c) Second refinement step

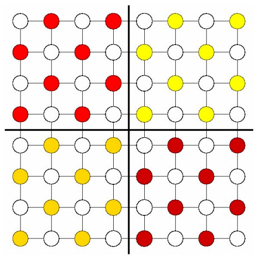

(d) Composed coarse grid

Figure 3.12.: CGC-ML refinement process.

follows:

1 We agglomerate the graph on a subset of the processors, see Figure 3.11(b). Hence, a part of the edges (in this case, the vertical edges) does not cross the processor boundaries any more.

2 Now, we can employ an edge matching on the inner edges of each processor subdomain, see Figure 3.11(c). We match each vertex u to a vertex v that is not already matched and for which the edge is least penalized, i.e. $\omega(u, v) = \min_{w:\ (u,w)\in W} \omega(u, w)$.

3 We collapse the matched vertices and merge the edge sets, see Figure 3.11(d). Each vertex now represents a candidate coarse grid on a union of processor subdomains. Accordingly, we update the vertex weights, i,e, the number of subdomains covered. The edge set of each vertex u is the union of the edge sets of the vertices v, w that were collapsed into u: $E_u \leftarrow E_v \cup E_w$. Note that we never create an edge between two vertices which represent candidate coarse grids on the same processor subdomain. If two edges are collapsed into the same edge, we add their edge weights.

4 We repeat the previous step until no further matching is possible. Then, we again agglomerate the graph on a smaller subset of processors. If we are already on a single processor and cannot match further, we stop, see Figure 3.11(e) – 3.11(f).

We have obtained a small set of vertices on a single processor. Now, we choose one vertex u such that it covers a maximal number of processor subdomains and mark it, see Figure 3.12(a). Then, we mark the vertices v and w that were collapsed into u. We recursively proceed to refine this choice until we have reached the original graph, see 3.12(b) – 3.12(c). Now on each processor subdomain, the candidate coarse grid represented by the marked vertex is selected as a coarse grid for this processor subdomain and we obtain a coarse grid for the global discretization domain as depicted in Figure 3.12(d). During this refinement process, we must ensure that one vertex is marked per processor subdomain on each level of the graph hierarchy. In consequence, this will guarantee that after finishing the refinement, we have selected one candidate coarse grid on each processor subdomain. In our implementation, we proceed as follows: At each step in the refinement process where more processors are involved as in the previous step (i.e. a processor agglomeration was performed in the matching phase), we determine if a vertex is marked on *each* processor. If this is not the case, we mark the vertex that is most heavily coupled to the marked vertices on neighboring processors.

Numerical Results We conclude this section with an example computed on the JUBL supercomputer at the Forschungszentrum Jülich. This IBM BlueGene/L cluster, which was in service between 2005 and 2008, consisted of 8192 compute nodes which were connected by a three-dimensional torus for local communication and a tree-shaped network for collective communication. JUBL performed 36.49 Tera-flop/s in the LINPACK

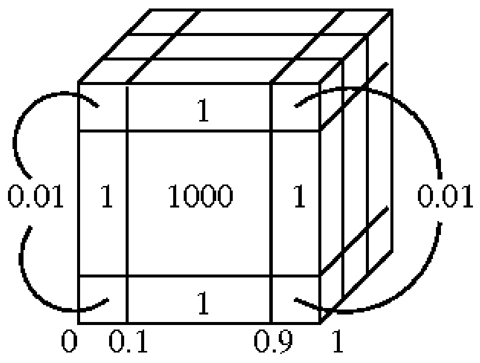


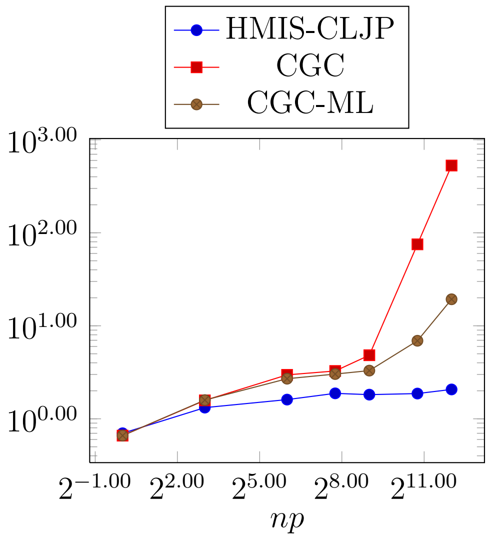
Figure 3.13.: Distribution of the diffusion coefficient for problem (3.6).

benchmark and held the 8th position on the Top 500 list in June 2006.
Each compute node contained two Power 440 processors running at a clock speed of 700MHz and provided 512MB of memory. This small amount of memory necessitates algorithms with little memory overhead, which can be a strong limitation in the case of parallel AMG especially for three-dimensional computations.
We consider an elliptic PDE on a unit cube,

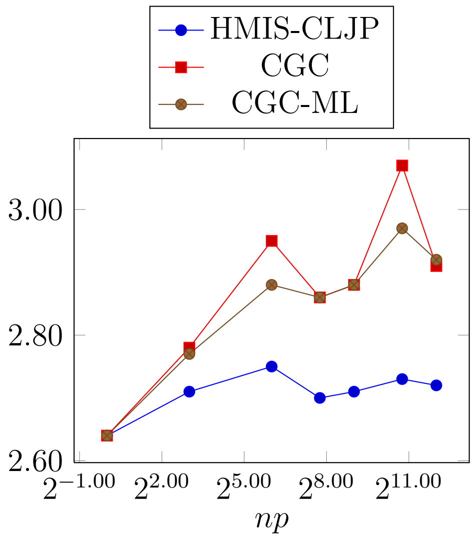
$$-\nabla \cdot a(x, y, z)\nabla \mathsf{u} = \mathsf{f} \tag{3.6}$$

with Dirichlet boundary conditions. The value of the coefficient $a(x, y, z)$ is depicted in Figure 3.5. We discretize the PDE on a finite difference mesh with $31 \times 31 \times 31$ points per processor subdomain. We compare the CGC-ML algorithm with the original CGC algorithm as well as the HMIS parallel coarsening algorithm (Algorithm 3.8). In the latter case, we use a hybrid coarsening process where the CLJP algorithm is employed on the coarsest levels of the AMG hierarchy. For the CGC-ML algorithm, we merged eight processor subdomains per agglomeration step. To limit the memory overhead, we do not include the second stage of the Ruge-Stüben coarsening process in the CGC and CGC-ML cases, and we use modified classical interpolation (see Section 2.8.4). A (more stable) long-range interpolation scheme was not feasible due to the memory limitations – even if we truncated the interpolation operator. As strength threshold in (2.29), we set $\alpha = 0.25$. On each level of the multigrid hierarchy, we employ a hybrid Gauss-Seidel/Jacobi smoother.
In this example, we use AMG as a preconditioner for the conjugate gradient method. We start the iterations with a zero initial vector u_0 and stop if the residual $r_{it} = f - Au_{it}$ drops below 10^{-8} measured in the l^2-norm.

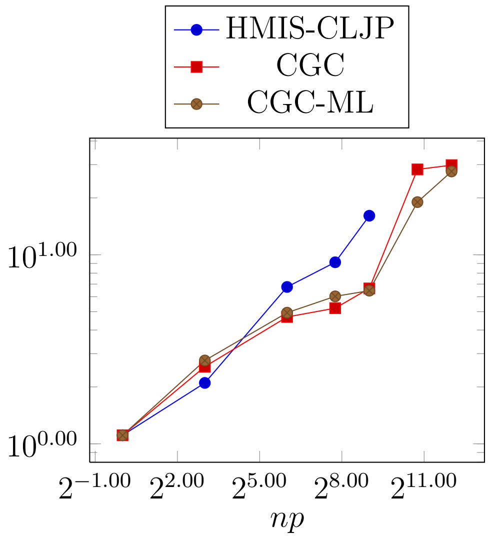
As can be seen from Table 3.1, the CGC-ML algorithm yields a significant run-time benefit compared to the original CGC algorithm. The HMIS coarsening scheme is even faster and produces less memory overhead than the CGC variants (Table 3.2). However, the grids produced by this method are too sparse to provide an accurate interpolation of the smooth error. In consequence, the iterations needed to reduce the residual below $1e - 8$ increase for a larger problem size as can be seen from Table 3.4. For 1728 and 4096 processors ($372 \times 372 \times 372$ and $496 \times 496 \times 496$ unknowns respectively), 1000



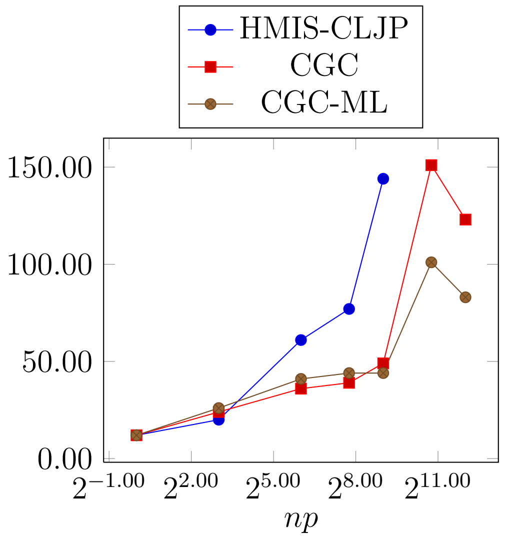
(a) Setup time in seconds.



(b) Operator complexity.



(c) Solution time in seconds.



(d) Iterations.

Figure 3.14.: Numerical Results for problem (3.6)

np	HMIS-CLJP	CGC	CGC-ML
1	0.70	0.66	0.66
8	1.32	1.58	1.59
64	1.61	2.97	2.70
216	1.88	3.27	3.03
512	1.82	4.83	3.30
1,728	1.87	75.20	6.91
4,096	2.07	529.00	19.30

Table 3.1.: Setup time in seconds

np	HMIS-CLJP	CGC	CGC-ML
1	2.64	2.64	2.64
8	2.71	2.78	2.77
64	2.75	2.95	2.88
216	2.70	2.86	2.86
512	2.71	2.88	2.88
1,728	2.73	3.07	2.97
4,096	2.72	2.91	2.92

Table 3.2.: Operator Complexity

np	HMIS-CLJP	CGC	CGC-ML
1	1.11	1.11	1.11
8	2.10	2.56	2.76
64	6.76	4.69	4.94
216	9.13	5.22	6.03
512	16.10	6.64	6.46
1,728	–	28.30	19.00
4,096	–	29.80	27.60

Table 3.3.: Solution time in seconds

np	HMIS-CLJP	CGC	CGC-ML
1	12	12	12
8	20	24	26
64	61	36	41
216	77	39	44
512	144	49	44
1,728	–	151	101
4,096	–	123	83

Table 3.4.: Iterations

Table 3.5.: Numerical Results for problem (3.6).

iterations were not sufficient to reach this threshold. The CGC variants also need more iterations here, but the increase is less severe. In turn, the solution phase CPU timings for the CGC variants are lower than those of the HMIS-coarsened AMG hierarchy, see Table 3.3.

4. AMG for Saddle Point Systems

In the following, we consider saddle point systems of the shape

$$\mathcal{K} : \begin{pmatrix} \mathcal{V} \\ \mathcal{W} \end{pmatrix} \to \begin{pmatrix} \mathcal{V} \\ \mathcal{W} \end{pmatrix}, \qquad \mathcal{K}x = y \tag{4.1}$$

where

$$\mathcal{K} = \begin{pmatrix} A & B^T \\ B & -C \end{pmatrix}, \; x = \begin{pmatrix} u \\ p \end{pmatrix} \text{ and } y = \begin{pmatrix} f \\ g \end{pmatrix} \tag{4.2}$$

Here, A is a symmetric positive semi-definite $N \times N$-matrix, C is a symmetric positive semi-definite $M \times M$ matrix and $B \in \mathbb{R}^{M \times N}$. As an important class of saddle point systems arises within the context of fluid dynamics, we will refer to $u \in \mathcal{V}$ as the *velocity component(s)* and $p \in \mathcal{W}$ as the *pressure component*.
This chapter is organized as follows. First, in Section 4.1 we introduce Stokes' equations as a model saddle point problem and discuss conditions for the existence and uniqueness of a solution. We proceed with finite difference and finite element discretizations for saddle point problems in Section 4.2. Classical iterative solution methods for saddle point problems are described in Section 4.3. In Section 4.4 we address several previous approaches to the construction of AMG for saddle point problems, before we start introducing our saddle point AMG. We first give a detailed description of our smoother in Sections 4.5–4.6. Afterwards, we discuss the construction of stable (invertible) prolongation and coarse grid operators in Sections 4.7–4.10. We conclude this chapter with the overall setup procedure for saddle point AMG, see Section 4.11.

4.1. Stokes equations

The Stokes problem is one of the most-known systems of partial differential equations (PDEs) with saddle point structure. It models the velocity u and the pressure p of a viscous flow in a domain $\Omega \subset \mathbb{R}^d$ ($d = 2$ or $d = 3$) subject to an external force f,

$$\begin{aligned} -\nabla \cdot (\nabla \mathsf{u}) + \nabla \mathsf{p} &= \mathsf{f} & (4.3)\\ -\nabla \cdot \mathsf{u} &= 0 & (4.4) \end{aligned}$$

subject to to the boundary conditions

$$\begin{aligned} \mathsf{u}(x) &= \mathsf{r}(x) \text{ for all } x \in \Gamma_D & (4.5)\\ \frac{\partial \mathsf{u}}{\partial n}(x) - n\mathsf{p}(x) &= \mathsf{s}(x) \text{ for all } x \in \Gamma_N & (4.6) \end{aligned}$$

where $\partial\Omega = \Gamma_D \dot{\cup} \Gamma_N$ and n denotes the outer normal vector on Γ_N. Note that the velocity u is a vector-valued function of dimension d, while p has scalar values.
The equations (4.3)–(4.4) have a unique solution (u, p) only if both the Dirichlet boundary Γ_D and the Neumann boundary Γ_N are non-trivial. However, in many applications we have $\partial\Omega = \Gamma_D$, i.e. the velocity of the fluid is prescribed on the whole boundary. In this case, while there is still a unique solution for the velocity, the pressure is unique up to a constant c, i.e. for any solution $(\mathsf{u}^*, \mathsf{p}^*)$ of (4.3)–(4.4), $(\mathsf{u}^*, \mathsf{p}^* + \mathsf{c})$ is also a solution for any constant function $\mathsf{c} \in C(\Omega)$.
The Stokes problem belongs to the class of *saddle point problems*. The solution of (4.3)–(4.4) is not a minimizer of a quadratic functional, but represents a saddle point. In the following, we give a general formulation of this kind of problems.
We consider the Hilbert spaces V and W as well as the corresponding dual spaces V' and W'. Let $< \cdot, \cdot >_{\mathsf{V}\times\mathsf{V}'}$ and $< \cdot, \cdot >_{\mathsf{W}\times\mathsf{W}'}$ denote the bi-linear forms given by

$$< \mathsf{u}, \mathsf{f} >_{\mathsf{V}\times\mathsf{V}'} = \mathsf{f}(\mathsf{u}) \text{ and } < \mathsf{p}, \mathsf{g} >_{\mathsf{W}\times\mathsf{W}'} = \mathsf{g}(\mathsf{p}),$$

for all $\mathsf{u} \in \mathsf{V}$, $\mathsf{f} \in \mathsf{V}'$, $\mathsf{p} \in \mathsf{W}$, and $\mathsf{g} \in \mathsf{W}'$.
Given a linear operator $\mathsf{A} : \mathsf{V} \to \mathsf{V}'$ and $\mathsf{f} \in \mathsf{V}'$, we seek the minimizer of

$$\frac{1}{2} \langle \mathsf{u}, \mathsf{A}\mathsf{u} - \mathsf{f} \rangle_{\mathsf{V}\times\mathsf{V}'}, \tag{4.7}$$

but, in contrast to the elliptic case, we need to satisfy a constraint

$$\langle \mathsf{w}, \mathsf{B}\mathsf{u} - \mathsf{g} \rangle_{\mathsf{W}\times\mathsf{W}'} = 0 \text{ for all } \mathsf{w} \in \mathsf{W}, \tag{4.8}$$

where $\mathsf{g} \in \mathsf{W}'$ and $\mathsf{B} : \mathsf{V} \to \mathsf{W}'$ is a linear operator. The adjoint operator $\mathsf{B}' : \mathsf{W} \to \mathsf{V}'$ is defined by $< \mathsf{v}, \mathsf{B}'\mathsf{w} >_{\mathsf{V}\times\mathsf{V}'} = < \mathsf{w}, \mathsf{B}\mathsf{v} >_{\mathsf{W}\times\mathsf{W}'}$ for all $\mathsf{v} \in \mathsf{V}$, $\mathsf{w} \in \mathsf{W}$.
We introduce the Lagrange multiplier $\mathsf{p} \in \mathsf{W}$ to combine (4.7) and (4.8) into a single Lagrange functional

$$\mathsf{L}(\mathsf{u}, \mathsf{p}) = \frac{1}{2} \langle \mathsf{u}, \mathsf{A}\mathsf{u} - \mathsf{f} \rangle_{\mathsf{V}\times\mathsf{V}'} + \langle \mathsf{p}, \mathsf{B}\mathsf{u} - \mathsf{g} \rangle_{\mathsf{W}\times\mathsf{W}'}. \tag{4.9}$$

It is easy to see that $\mathsf{L}(\mathsf{u}, \mathsf{p})$ is not bounded. Hence, a solution $(\mathsf{u}^*, \mathsf{p}^*)$ of

$$\begin{aligned} \langle \mathsf{v}, \mathsf{A}\mathsf{u} + \mathsf{B}'\mathsf{p} - \mathsf{f} \rangle_{\mathsf{V}\times\mathsf{V}'} &= 0 \text{ for all } \mathsf{v} \in \mathsf{V} \\ \langle \mathsf{w}, \mathsf{B}\mathsf{u} - \mathsf{g} \rangle_{\mathsf{W}\times\mathsf{W}'} &= 0 \text{ for all } \mathsf{w} \in \mathsf{W} \end{aligned}$$

is only a *saddle point* (not a minimizer) of $\mathsf{L}(\mathsf{u}, \mathsf{p})$, i.e. we have

$$\mathsf{L}(\mathsf{u}^*, \mathsf{p}) \leq \mathsf{L}(\mathsf{u}^*, \mathsf{p}^*) \leq \mathsf{L}(\mathsf{u}, \mathsf{p}^*)$$

for all $\mathsf{u} \in \mathsf{V}$, $\mathsf{p} \in \mathsf{W}$, see [Bra97] or [Hac86] for details.

Existence and uniqueness of a solution As in the case of elliptic partial differential equations, we formulate necessary and sufficient conditions for the existence and uniqueness of a solution to (4.7)–(4.8)
Let the spaces V and W be equipped with norms $\|\cdot\|_{\mathsf{V}}$ and $\|\cdot\|_{\mathsf{W}}$. We consider the system of equations

$$\begin{pmatrix} \mathsf{A} & \mathsf{B}' \\ \mathsf{B} & 0 \end{pmatrix} \begin{pmatrix} \mathsf{u} \\ \mathsf{p} \end{pmatrix} = \begin{pmatrix} \mathsf{f} \\ \mathsf{g} \end{pmatrix}. \tag{4.10}$$

Furthermore, we define the nullspace of B,

$$\mathsf{V}_0 = \{\mathsf{v} \in \mathsf{V} : \ \mathsf{Bv} = 0\}. \tag{4.11}$$

As V_0 is a closed set, we can decompose $\mathsf{V} = \mathsf{V}_0 \oplus \mathsf{V}_\perp$, where $V_\perp = V_0^\perp$. For the dual space V' we have $\mathsf{V}' = \mathsf{V}_0' \oplus \mathsf{V}_\perp'$, where

$$\begin{aligned} \mathsf{V}_0' &= \{\mathsf{v}' \in \mathsf{V}' : \ \langle \mathsf{v}', \mathsf{v}\rangle = 0 \text{ for all } \mathsf{v} \in \mathsf{V}_\perp\}, \\ \mathsf{V}_\perp' &= \{\mathsf{v}' \in \mathsf{V}' : \ \langle \mathsf{v}', \mathsf{v}\rangle = 0 \text{ for all } \mathsf{v} \in \mathsf{V}_0\}. \end{aligned}$$

Using these subspaces, we can decompose A,

$$\mathsf{A} = \begin{pmatrix} \mathsf{A}_{00} & \mathsf{A}_{0\perp} \\ \mathsf{A}_{\perp 0} & \mathsf{A}_{\perp\perp} \end{pmatrix}$$

where

$$\mathsf{A}_{00} : \mathsf{V}_0 \to \mathsf{V}_0', \quad \mathsf{A}_{0\perp} : \mathsf{V}_\perp \to \mathsf{V}_0', \quad \mathsf{A}_{\perp 0} : \mathsf{V}_0 \to \mathsf{V}_\perp', \quad \mathsf{A}_{\perp\perp} : \mathsf{V}_\perp \to \mathsf{V}_\perp'.$$

Theorem 4.1 *([Hac86], Theorem 12.2.7) The saddle point system* (4.10) *has a unique solution for all $f \in V'$ if and only if the inverses*

$$\mathsf{A}_{00}^{-1} : \mathsf{V}_0' \to \mathsf{V}_0 \ \textit{and} \ (\mathsf{B}')^{-1} : \mathsf{V}_\perp' \to \mathsf{W} \tag{4.12}$$

exist.

In terms of the norms $\|\cdot\|_{\mathsf{V}}, \ \|\cdot\|_{\mathsf{W}}$, (4.12) can be expressed as [Hac86]

$$\inf_{0 \neq \mathsf{u}_0 \in \mathsf{V}_0} \sup_{0 \neq \mathsf{v} \in \mathsf{V}} \frac{\langle \mathsf{Au}_0, \mathsf{v}\rangle}{\|\mathsf{u}_0\|_{\mathsf{V}} \|\mathsf{v}\|_{\mathsf{V}}} \ \geq \ a > 0, \tag{4.13}$$

$$\inf_{0 \neq \mathsf{p} \in \mathsf{W}} \sup_{0 \neq \mathsf{v} \in \mathsf{V}} \frac{\langle \mathsf{B}'\mathsf{w}, \mathsf{v}\rangle}{\|\mathsf{p}\|_{\mathsf{W}} \|\mathsf{v}\|_{\mathsf{V}}} \ \geq \ c > 0. \tag{4.14}$$

The first inequality is fulfilled if the bilinear form $\langle \mathsf{A}\cdot, \cdot\rangle$ is elliptic w.r.t. V or V_0, see [Hac86], Corollary 12.2.10. The latter inequality is called *Ladyzenskaya-Babuška-Brezzi (LBB)* condition or *inf-sup-condition* [Bre74]. In the following sections, we will see that this condition is not only required in the continuous case, but also plays an important role to show the stability of a discretization as well as the stability of coarse grid systems within a saddle point multigrid hierarchy.

Weak formulation In this paragraph, we shortly outline the weak formulation of the Stokes problem. As ansatz and test spaces for the velocity we use

$$H^1_E(\Omega)^d := \{\mathsf{v} \in H^1(\Omega)^d : \mathsf{v}_{|\Gamma_D} = \mathsf{r}\} \tag{4.15}$$

$$H^1_{E_0}(\Omega)^d := \{\mathsf{v} \in H^1(\Omega)^d : \mathsf{v}_{|\Gamma_D} = 0\}. \tag{4.16}$$

while for the pressure we use $L^2(\Omega)$. Now, the weak form of (4.3)–(4.4) together with the boundary conditions (4.5)–(4.6) reads as follows.
Find $\mathsf{u} \in H^1_E(\Omega)^d$ and $\mathsf{p} \in L^2(\Omega)$ such that

$$\int_\Omega \nabla \mathsf{u} : \nabla \mathsf{v} - \int_\Omega \mathsf{p} \nabla \cdot \mathsf{v} = \int_\Omega \mathsf{f} \cdot \mathsf{v} + \int_{\Gamma_N} \mathsf{s} \cdot \mathsf{v} \text{ for all } \mathsf{v} \in H^1_{E_0}(\Omega)^d, \tag{4.17}$$

$$\int_\Omega \mathsf{r} \nabla \cdot \mathsf{u} = 0 \text{ for all } \mathsf{r} \in L^2(\Omega), \tag{4.18}$$

where $\nabla \mathsf{u} : \nabla \mathsf{v} = \sum_{i=1}^d \nabla \mathsf{u}_i \cdot \nabla \mathsf{v}_i$ is the component-wise scalar product.
We now formulate the invertibility conditions for (4.17)–(4.18). To this end, we restrict ourselves to the case of zero boundary conditions on Γ_D. We use $\mathsf{V} := H^1_{E_0}(\Omega)^d$ as ansatz space and note that $H^1_{E_0}(\Omega)^d \subset \mathsf{V}'$. From the theory of elliptic partial differential equations, we know that

$$\mathsf{a} : H^1_{E_0}(\Omega)^d \times H^1_{E_0}(\Omega)^d \to \mathbb{R}, \qquad \mathsf{a}(\mathsf{u}, \mathsf{v}) := \int_\Omega \nabla \mathsf{u} : \nabla \mathsf{v}$$

is continuous and coercive over $H^1_{E_0}(\Omega)^d \times H^1_{E_0}(\Omega)^d$, hence it follows that (4.13) holds. It remains to formulate the inf-sup condition (4.14). Here, we must distinguish between two cases depending on the boundary conditions (4.5)–(4.6).
In the case of $\partial\Omega = \Gamma_D$, the pressure solution is unique only up to a constant. This is reflected by the definition of W,

$$\mathsf{W} := \left\{ p \in L^2(\Omega) : \int_\Omega p = 0 \right\},$$

equipped with the usual L^2-norm. By this definition, the only constant function in W is identical to zero. Otherwise, if $\Gamma_N \neq \emptyset$, then we set $\mathsf{W} = L^2(\Omega)$. The inf-sup condition reads

$$\inf_{0 \neq \mathsf{p} \in \mathsf{W}} \sup_{0 \neq \mathsf{v} \in \mathsf{H}^1_{\mathsf{E}_0}(\Omega)^\mathsf{d}} \frac{|\mathsf{b}(\mathsf{v}, \mathsf{p})|}{\|\mathsf{p}\|_{\mathsf{L}^2(\Omega)} \|\mathsf{v}\|_{\mathsf{H}^1_{\mathsf{E}_0}(\Omega)^\mathsf{d}}} \geq c > 0 \tag{4.19}$$

where the inner product $\mathsf{b}(\mathsf{v}, \mathsf{p}) : H^1_{E_0}(\Omega)^d \times L^2(\Omega) \to \mathbb{R}$ is defined by

$$\mathsf{b}(\mathsf{v}, \mathsf{p}) := \int_\Omega p \nabla \cdot v.$$

We refer to [Hac86] and [Bra97] for a more detailed description as well as for sufficient criteria to establish the LBB condition (4.19).

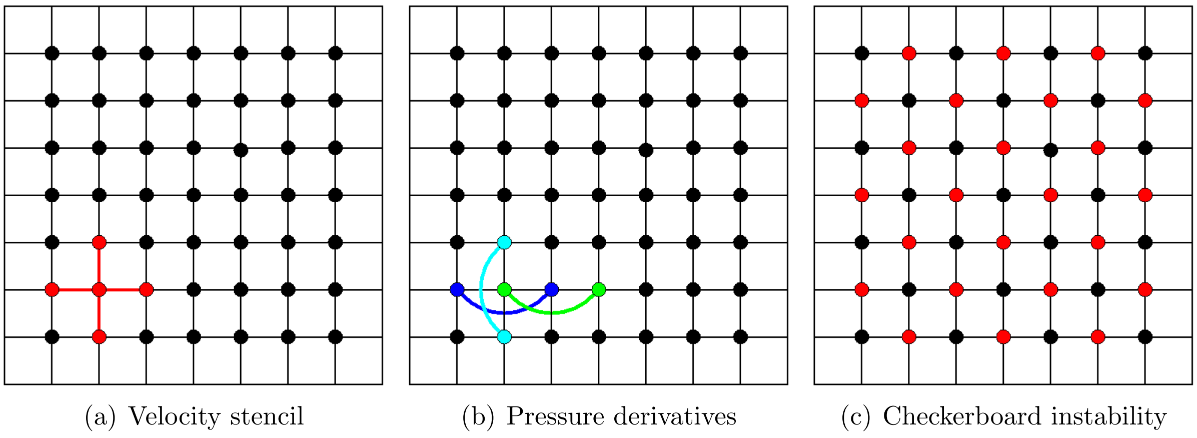

(a) Velocity stencil (b) Pressure derivatives (c) Checkerboard instability

Figure 4.1.: Finite difference discretization grids for the Stokes problem $\Delta u = f$ on the square $[0,1]^2$.

4.2. Discretization of Saddle Point PDEs

In this section, we outline various finite difference (FD) and finite element (FE) discretization schemes for saddle point problems, especially for the Stokes' equation. As in the continuous case, it is important that the discrete operators A and B satisfy an inf-sup condition (4.14). Moreover, in the case of a non-trivial Neumann boundary Γ_N, the matrix B^T should have full rank, while in the case of Dirichlet boundary conditions everywhere the kernel of B^T should just represent the constant pressure, which in most cases means that the kernel of B^T is spanned by a constant vector. Unfortunately, the most obvious discretization schemes are not suitable.

4.2.1. Finite Difference Discretization

We start with finite difference discretizations. For sake of simplicity, let us assume $\Omega = (0,1)^d$. We choose a mesh width $h < 1$ and cover $\overline{\Omega}$ with an equidistant grid with $n+1$ points per spatial dimension (where $n = \frac{1}{h}$),

$$\Omega_h := \left\{ x = (x_1, \ldots, x_d)^T \in \Omega : \; x_i = j_i \cdot h, \; j_i \in \{0, \ldots, n\} \right\}. \tag{4.20}$$

In the following, we assume $\partial\Omega = \Gamma_D$, i.e. we only have Dirichlet boundary conditions. A first approach is to discretize the quantities u and p at the grid points in Ω_h, i.e. in the case of $d = 2$ we set

$$u_{i,j} := \mathsf{u}_1(ih, jh), \qquad v_{i,j} := \mathsf{u}_2(ih, jh), \qquad p_{i,j} := \mathsf{p}(ih, jh),$$
$$f_{i,j} := \mathsf{f}_1(ih, jh), \qquad g_{i,j} := \mathsf{f}_2(ih, jh)$$

To achieve an appropriate accuracy, we employ a central difference scheme for the pressure and the usual 5 point (two dimensions) or 7 point (three dimensions) FD stencil for the velocity. For example, in two spatial dimension we obtain the following algebraic equations for each $i, j = 1, \dots, n-1$ (see Figure 4.1(a)–4.1(b)),

$$\frac{1}{h^2}\left(4u_{i,j} - u_{i,j-1} - u_{i-1,j} - u_{i+1,j} - u_{i,j+1}\right) \quad + \frac{1}{2h}\left(p_{i+1,j} - p_{i-1,j}\right) \quad = f_{i,j}, \tag{4.21}$$
$$\frac{1}{h^2}\left(4v_{i,j} - v_{i,j-1} - v_{i-1,j} - v_{i+1,j} - v_{i,j+1}\right) \quad + \frac{1}{2h}\left(p_{i,j+1} - p_{i,j-1}\right) \quad = g_{i,j}, \tag{4.22}$$
$$\frac{1}{2h}\left(u_{i+1,j} - u_{i-1,j} + v_{i,j+1} - v_{i,j-1}\right) \quad = 0. \tag{4.23}$$

Boundary entries (i.e. $u_{0,j}$, $u_{n,j}$, $u_{i,0,}$, $u_{i,n)}$, $v_{0,j}$, $v_{n,j}$, $v_{i,0,}$, $v_{1,n}$) are replaced by their respective values r and transferred to the right hand side, such that these are no longer contained in the vectors u and v.
Note that the discretization of the derivatives $\frac{\partial \mathsf{p}}{\partial x}$ and $\frac{\partial \mathsf{p}}{\partial y}$ at (x_i, y_j) does not involve the discrete value $p_{i,j}$, but these derivatives are computed from the neighbors $p_{i+1,j}$, $p_{i-1,j}$, $p_{i,j+1}$, $p_{i,j-1}$. Likewise, the discrete divergence (4.23) only employs the values $u_{i+1,j}$, $u_{i-1,j}$, $v_{i+1,j}$, and $v_{i-1,j}$. In matrix-vector notation, we obtain a linear system

$$\begin{pmatrix} A_1 & 0 & B_1^T \\ 0 & A_2 & B_2^T \\ B_1 & B_2 & 0 \end{pmatrix} \begin{pmatrix} u \\ v \\ p \end{pmatrix} = \begin{pmatrix} f \\ g \\ 0 \end{pmatrix} \tag{4.24}$$

where the matrix blocks A_1 and A_2 are obtained from the discretization stencil

$$\frac{1}{h^2}\begin{bmatrix} & -1 & \\ -1 & 4 & -1 \\ & -1 & \end{bmatrix}$$

while the B_1^T and B_2^T parts are originated from the stencils

$$\frac{1}{2h}\begin{bmatrix} -1 & 0 & 1 \end{bmatrix} \text{ and } \frac{1}{2h}\begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix},$$

respectively.
We now look at the equations at the point $(i+1, j)$. They take the form

$$\frac{1}{h^2}\left(4u_{i+1,j} - u_{i+1,j-1} - u_{i,j} - u_{i+2,j} - u_{i+1,j+1}\right) \quad + \frac{1}{2h}\left(p_{i+2,j} - p_{i,j}\right) \quad = f_{i+1,j},$$
$$\frac{1}{h^2}\left(4v_{i+1,j} - v_{i+1,j-1} - v_{i,j} - v_{i+2,j} - v_{i+1,j+1}\right) \quad + \frac{1}{2h}\left(p_{i+1,j+1} - p_{i+1,j-1}\right) \quad = g_{i+1,j},$$
$$\frac{1}{2h}\left(u_{i+2,j} - u_{i,j} + v_{i+2,j} - v_{i,j}\right) \quad = 0.$$

We see that the derivative $\frac{\partial \mathsf{p}}{\partial x}$ is discretized using the values at i, j and $i+2, j$ and that the divergence term involves $u_{i+2,j}$, $u_{i,j}$, $v_{i+2,j}$, and $v_{i,j}$, i.e. in comparison to (4.21)–(4.23), a completely disjoint set of discretization points (and hence discrete values u, v, p) is used. The same phenomenon can be observed throughout the whole discrete domain Ω_h. More precisely, any discrete pressure value $p_{i,j}$ is only coupled to pressure values "two points away", i.e. $p_{i,j-2}$, $p_{i-2,j}$, $p_{i-2,j}$, and $p_{i,j+2}$, see Figure 4.1(b). In consequence, the matrix B can be decomposed into two decoupled parts B_{odd} and B_{even} such that (after re-arrangement of rows and columns),

$$B = \begin{pmatrix} B_{odd} & 0 \\ 0 & B_{even} \end{pmatrix}.$$

Simple observations (consider e.g. (4.21)–(4.23)) show that the kernels of both B_{odd}^T and B_{even}^T contain the constant value. In consequence, the matrix B^T has a two-dimensional nullspace,

$$\begin{aligned} \ker B^T &= \operatorname{span}(p_{odd}, p_{even}) \\ \text{where } (p_{odd})_{i,j} &= \begin{cases} 1 \text{ if } i+j \text{ odd} \\ 0 \text{ if } i+j \text{ even} \end{cases} \\ \text{where } (p_{even})_{i,j} &= \begin{cases} 1 \text{ if } i+j \text{ even} \\ 0 \text{ if } i+j \text{ odd.} \end{cases} \end{aligned}$$

This space forms a checkerboard-wise pattern across the domain, see Figure 4.1(c). Hence, this phenomena is called *checkerboard instability.*

A remedy against this problem is to use a *staggered mesh*, i.e. to discretize the velocity components and the pressure variables at different locations. We describe this technique in the two-dimensional case here, but it is also applicable to three spatial dimensions (see e.g. [GDN98]).

As before, we use the mesh Ω_h as defined in (4.20), but we now consider it rather as a union of cells

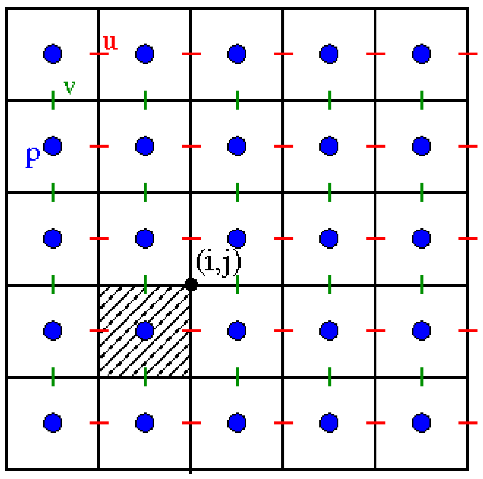
$$\Omega_{i,j} \;:=\; \{(x,y) \in \Omega : \; (i-1)\cdot h \le x \le i \cdot h; \;\; (j-1)\cdot h \le x \le j \cdot h; \; \},$$

where $i, j = 1, \ldots, n$, see Figure 4.2(a). We discretize the pressure variables at the centers of these cells,

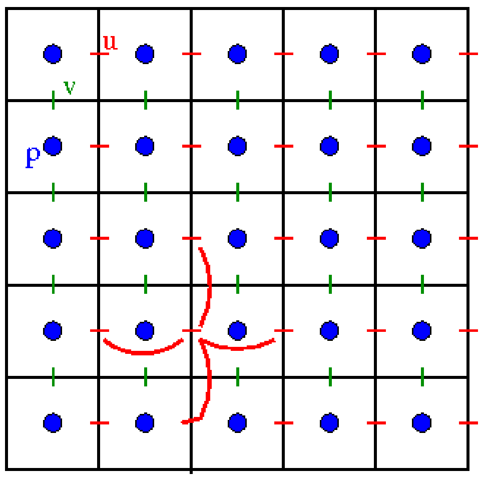
$$\Omega_p \;:=\; \left\{ \begin{pmatrix} \left(i - \frac{1}{2}\right) h \\ \left(j - \frac{1}{2}\right) h \end{pmatrix}, \quad i, j = 1, \ldots, n \right\},$$

and let $p_{ij} = \mathsf{p}\left(\left(i - \frac{1}{2}\right) h, \left(j - \frac{1}{2}\right) h\right)$. The velocity component u (x-direction) and the first component of the right hand side are discretized at the centers of the interfaces between two adjacent cells in x-direction,

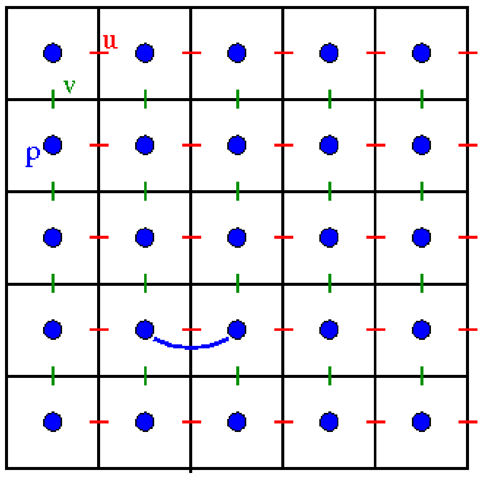
$$\begin{aligned} \Omega_u &:= \left\{ \begin{pmatrix} ih \\ (j - \frac{1}{2})h \end{pmatrix}, \quad i = 0, \ldots, n; \;\; j = 1, \ldots, n \right\} \\ u_{ij} &= \mathsf{u}\left(ih, \left(j - \frac{1}{2}\right) h \right), \\ f_{ij} &= \mathsf{f}_1\left(ih, \left(j - \frac{1}{2}\right) h \right). \end{aligned}$$



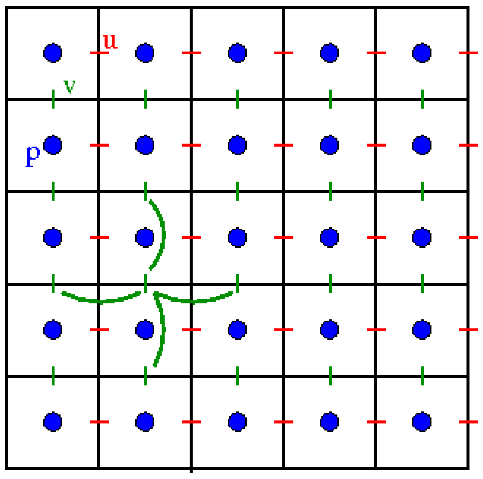
(a) Staggered grid with point (i, j) and cell $\Omega_{(i,j)}$



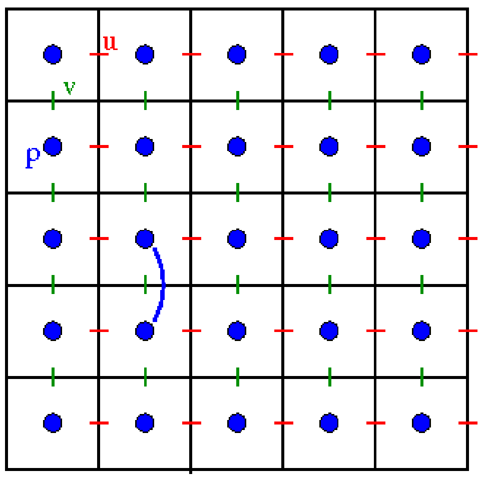
(b) Velocity stencil for $\Delta \mathsf{u}\left(ih, j - \frac{1}{2}h\right)$



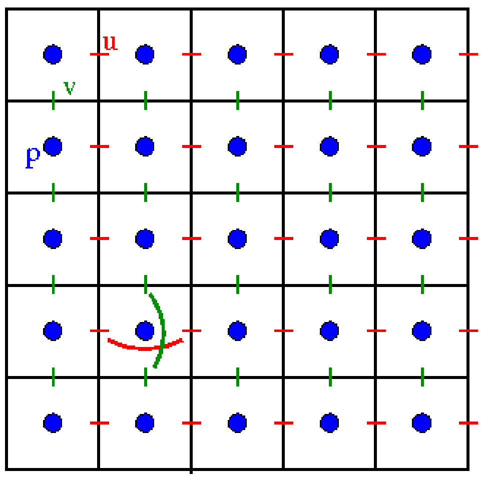
(c) Pressure difference for $\frac{\partial \mathsf{p}}{\partial x}\left(ih, j - \frac{1}{2}h\right)$



(d) Velocity stencil for $\Delta \mathsf{v}\left(i - \frac{1}{2}h, j\right)$



(e) Pressure difference for $\frac{\partial \mathsf{p}}{\partial y}\left(i - \frac{1}{2}h, j\right)$



(f) Discrete divergence for the cell $\Omega_{(i,j)}$

Figure 4.2.: Staggered grid discretization for the Stokes problem $\Delta u = f$ on the square $[0, 1]^2$.

Likewise, the velocity component v and the right hand side component f_2 are discretized at the interfaces between two adjacent cells in y-direction,

$$\begin{aligned}\Omega_v &:= \left\{\begin{pmatrix}(i-\frac{1}{2})h)\\ jh\end{pmatrix}, \quad i=1,\dots,n;\ \ j=0,\dots,n\right\}\\ v_{ij} &= \mathsf{v}\left(\left(i-\frac{1}{2}\right)h, jh\right),\\ g_{ij} &= \mathsf{f}_2\left(\left(i-\frac{1}{2}\right)h, jh\right).\end{aligned}$$

For each of these sets, we define the interior by

$$\Omega_u^\circ := \Omega_u \cap (0,1)^2, \quad \Omega_u^\circ := \Omega_v \cap (0,1)^2,$$

i.e. we take away the first and the last points of Ω_u in x-direction, and the first and last points in y-direction of Ω_v .
We now have exactly one pressure value per cell $\Omega_{i,j}$. The pressure derivatives $\frac{\partial \mathsf{p}}{\partial x}$ and $\frac{\partial \mathsf{p}}{\partial y}$ are now approximated by the differences between two adjacent cells (Figures 4.2(c) and 4.2(e)),

$$\begin{aligned}\frac{\partial \mathsf{p}}{x} &\approx \frac{p_{i+1,j}-p_{i,j}}{h} \qquad \text{for } i=1,\dots,n-1,\ j=1,\dots,n,\\ \frac{\partial \mathsf{p}}{y} &\approx \frac{p_{i,j+1}-p_{i,j}}{h} \qquad \text{for } i=1,\dots,n,\ j=1,\dots,n-1,\end{aligned}$$

which shows that these finite difference terms can be seen as central differences for the pressure derivative at the points in Ω_u° and Ω_v°, respectively.
We can now discretize the impulse equations. We obtain one scalar equation for all $i=1,\dots,n-1$, $j=1,\dots,n$ (Figure 4.2(b)),

$$\frac{1}{h^2}\left(4u_{i,j}-u_{i,j-1}-u_{i-1,j}-u_{i+1,j}-u_{i,j+1}\right)+\frac{1}{h}\left(p_{i+1,j}-p_{i,j}\right)=f_{i,j}, \qquad (4.25)$$

and one equation for each $i=1,\dots,n$, $j=1,\dots,n-1$ (Figure 4.2(d)),

$$\frac{1}{h^2}\left(4v_{i,j}-v_{i,j-1}-v_{i-1,j}-v_{i+1,j}-v_{i,j+1}\right)+\frac{1}{h}\left(p_{i,j+1}-p_{i,j}\right)=g_{i,j}, \qquad (4.26)$$

Finally, the continuity term is discretized per cell ($i,j=1,\dots,n$) (Figure 4.2(f)),

$$\frac{1}{2h}\left(u_{i,j}-u_{i-1,j}+v_{i,j}-v_{i,j-1}\right)=0.$$

The latter equation can be seen as an incompressibility constraint for each cell: It requires that the sum of all ingoing and outgoing fluid matter is equal to zero.
It remains to eliminate the boundary entries from the equations above. The treatment of Dirichlet boundary conditions is straightforward for u if $x=0$ or $x=1$, and for v if

$y = 0$ or $y = 1$. In other cases, some extrapolation is needed. For example, the Dirichlet boundary value $\mathsf{r}(x, 0)$ for u at $y = 0$ is used to replace $u_{i,0}$ by linear extrapolation,

$$u_{i,0} = 2\mathsf{r}(ih, 0) - u_{i,1}.$$

Substituting into (4.25) we obtain

$$\frac{1}{h^2}\left(5u_{i,1} - u_{i-1,1} - u_{i+1,1} - u_{i,2}\right) + \frac{1}{h}\left(p_{i+1,1} - p_{i,1}\right) = f_{i,1}.$$

Regarding Neumann boundary conditions, let us assume that these are imposed for $x = 1$. We discretize (4.6) by

$$\begin{aligned} \frac{1}{2h}\left(u_{n+1,j} - u_{n-1,j}\right) - \frac{1}{2}\left(p_{n+1,j} + p_{n,j}\right) &= \mathsf{s}_1(1, j), \\ \frac{1}{2h}\left(v_{n,j} - v_{n-1,j}\right) &= \mathsf{s}_2(1, jh), \end{aligned}$$

and insert these into (4.25)–(4.26) (here, (4.25) is also needed for $i = n$). We obtain for all $j = 1, \ldots, n-1$,

$$\begin{aligned} \frac{1}{h^2}\left(4u_{n,j} - u_{n,j-1} - 2u_{n-1,j} - u_{n,j+1}\right) - \frac{2}{h}p_{n,j} &= f_{n,j} + \frac{2}{h}\mathsf{s}_1(1, j), \\ \frac{1}{h^2}\left(3v_{n,j} - v_{n,j-1} - v_{n,j} - v_{n,j+1}\right) + \frac{1}{h}\left(p_{n,j+1} - p_{n,j}\right) &= g_{n,j} + \frac{2}{h}\mathsf{s}_2(1, jh). \end{aligned}$$

Note that we can rescale the first equation by 0.5 to obtain a symmetric operator A (if all of the boundary at $x = 1$ is included in Γ_N).
We again obtain a linear system of the form (4.24)

$$\begin{pmatrix} A_1 & 0 & B_1^T \\ 0 & A_2 & B_2^T \\ B_1 & B_2 & 0 \end{pmatrix} \begin{pmatrix} u \\ v \\ p \end{pmatrix} = \begin{pmatrix} f \\ g \\ 0 \end{pmatrix},$$

but the stencils for the B_1^T and B_2^T blocks are different. They now read

$$\frac{1}{h}\begin{bmatrix} -1 & 1 \end{bmatrix} \text{ and } \frac{1}{h}\begin{bmatrix} 1 \\ -1 \end{bmatrix}.$$

To formulate the inf-sup condition for the two-dimensional case, we restrict ourselves again to the case of zero Dirichlet boundary conditions on the whole of $\partial\Omega$. We define three mesh-dependent inner products [SS97],

$$\begin{aligned} (p, q)_{0,\Omega_p} &:= h^2 \sum_{i,j=1}^{n} p_{i,j} \cdot q_{i,j}, \\ (u, u')_{1,\Omega_u} &:= \sum_{i,j=1}^{n} (u_{i,j} - u_{i-1,j}) \cdot (u'_{i,j} - u'_{i-1,j}) + (u_{i,j} - u_{i,j-1}) \cdot (u'_{i,j} - u'_{i,j-1}), \\ (v, v')_{1,\Omega_v} &:= \sum_{i,j=1}^{n} (v_{i,j} - v_{i-1,j}) \cdot (v'_{i,j} - v'_{i-1,j}) + (v_{i,j} - v_{i,j-1}) \cdot (v'_{i,j} - v'_{i,j-1}). \end{aligned}$$

These bilinear forms and their induced norms $\|\cdot\|_{0,\Omega_p}$, $\|\cdot\|_{1,\Omega_u}$, $\|\cdot\|_{1,\Omega_v}$, can be interpreted as mesh-dependent counterparts of the L^2 and H^1 inner products and norms. We define $L^2(\Omega_p)$ as the space of all discrete functions over Ω_p,

$$L^2(\Omega_p) := \{p : \Omega_p \to \mathbb{R}\}$$

Furthermore, we define the space of all functions with zero mean over Ω_p,

$$L_0^2(\Omega_p) = \{p \in L^2(\Omega_p) : \ (p,1)_0 = 0\}$$

and the spaces $H_0^1(\Omega_u)$ and $H_0^1(\Omega_v)$,

$$\begin{aligned} H_0^1(\Omega_u) &:= \{u : \Omega_u \to \mathbb{R}, \ u|_\Gamma = 0\}, \\ H_0^1(\Omega_v) &:= \{v : \Omega_v \to \mathbb{R}, \ v|_\Gamma = 0\}. \end{aligned}$$

Theorem 4.2 *([SS97],Theorem 4) There exists a positive constant c, which is independent of h, such that*

$$\sup_{u \in H_0^1(\Omega_u),\ v \in H_0^1(\Omega_v)} \frac{(B_1 u + B_2 v, p)_{0,\Omega_p}^2}{\|u\|_{1,\Omega_u}^2 + \|v\|_{1,\Omega_v}^2} \geq c\|p\|_{0,\Omega_p}^2 \text{ for all } p \in L_0^2(\Omega_p). \tag{4.27}$$

Straightforward calculations show that $(B_1 u + B_2 v, p)_{0,\Omega_p}^2 = h^4 \left(u^T B_1^T + v^T B_2^T\right) p$ and $\|u\|_{1,\Omega_u}^2 + \|v\|_{1,\Omega_v}^2 = h^2 \left(u^T A_1 u + v^T A_2 v\right)$. Hence we obtain, in terms of the discrete algebraic vectors u, v, and p and operators (matrices) $A = \begin{pmatrix} A_1 & 0 \\ 0 & A_2 \end{pmatrix}$ and $B = \begin{pmatrix} B_1 & B_2 \end{pmatrix}$,

$$\sup_{u \in H_0^1(\Omega_u),\ v \in H_0^1(\Omega_v)} \frac{\left((u^T, v^T) B^T p\right)^2}{(u^T, v^T) A \begin{pmatrix} u \\ v \end{pmatrix}} \geq c\|p\|_E^2 \text{ for all } p \in L_0^2(\Omega_p). \tag{4.28}$$

where $\|\cdot\|_E$ denotes the Euclidean norm.

4.2.2. Finite Elements for Saddle Point Systems

We now turn ourselves to finite element (FE) methods for saddle point problems. As in the case of finite differences, a straightforward extension of scalar discretizations is not sufficient to achieve stability and thus invertibility of the algebraic system.
First, we define discrete (finite-dimensional) spaces $\mathcal{V}_h \subset \mathsf{V}$ and $\mathcal{W}_h \subset \mathsf{W}$ and the corresponding bases,

$$\begin{aligned} \mathcal{V}_h &:= \operatorname{span}\{\phi_i : i = 1, \ldots, N\} && (4.29) \\ \mathcal{W}_h &:= \operatorname{span}\{\psi_i : i = 1, \ldots, M\}. && (4.30) \end{aligned}$$

The variables u and p can be represented as

$$\mathsf{u}_1 = \sum_{i=1}^{N} u_i \phi_i, \quad \mathsf{p} = \sum_{i=1}^{M} w_i \psi_i.$$

From the weak formulation (see Section 4.1) we obtain a linear system of equations,

$$\sum_{j=1}^{N} u_j \mathsf{a}(\phi_i, \phi_j) + \sum_{j=1}^{M} p_j \mathsf{b}(\phi_i, \psi_j) \qquad = \langle \mathsf{f}, \phi_i \rangle \text{ for all } i = 1, \ldots, N, \tag{4.31}$$

$$\sum_{j=1}^{N} u_j \mathsf{b}(\phi_j, \psi_i) \qquad = \langle \mathsf{g}, \psi_i \rangle \text{ for all } i = 1, \ldots, M, \tag{4.32}$$

or, in matrix-vector notation,

$$\begin{pmatrix} A & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} f \\ g \end{pmatrix}$$

where the entries of $A = (a_{ij}) \in \mathbb{R}^{N \times N}$ and $B = (b_{ij}) \in \mathbb{R}^{M \times N}$ are given by

$$\begin{aligned} a_{ij} =& \mathsf{a}(\phi_i, \phi_j) = \int_\Omega \nabla \phi_i : \nabla \phi_j && \text{for all } i, j = 1, \ldots, M, \\ b_{ij} =& \mathsf{b}(\phi_j, \psi_i) = \int_\Omega \psi_i \nabla \cdot \phi_j && \text{for all } i = 1, \ldots, M,\ j = 1, \ldots, N, \end{aligned}$$

and $u = (u_j)_{j=1}^N$, $p = (p_j)_{j=1}^M$, $f = (f_i)_{i=1}^N$, $g = (g_i)_{i=1}^M$, where

$$f_i = \int_\Omega \phi_i \mathsf{f}, \quad g_i = \int_\Omega \psi_i \mathsf{g}.$$

Note that the ϕ_i are vector-valued functions, i.e. they map into a d-dimensional space. In the most common cases however we choose to have distinguished basis functions per dimension, i.e.

$$\phi_i = \begin{pmatrix} 0 \\ \vdots \\ \varphi_i \\ \vdots \\ 0 \end{pmatrix}$$

where φ_i is a scalar-valued function. In consequence, in case of the Stokes equation the matrix A has no non-zero couplings between different velocity components.

Example 4.1 Let a finite element mesh for the domain $\Omega = (0, 1)^2$ be constructed as in Figure 4.3. We use a lexicographical ordering of the nodes (x_j, y_j) and construct a nodal basis $\{\varphi_j\}_{j=1}^n$ (i.e. $\varphi_i(x_j, y_j) = \delta_{ij}$ for all $i, j = 1, \ldots, n$) where all φ_j are piecewise

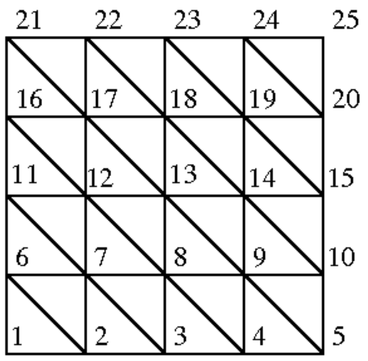


Figure 4.3.: Finite element mesh on the domain $\Omega = [0,1]^2$ with lexicographical node numbering.

linear functions on the triangles T such that $(x_i, y_i) \in \bar{T}$ and zero on all other triangles. Now, we extend these φ_i to a basis for discrete velocity and pressure space,

$$\begin{aligned}
\phi_i &= \begin{pmatrix} \varphi_i \\ 0 \end{pmatrix} && \text{for } i = 1, \ldots, 25, \\
\phi_i &= \begin{pmatrix} 0 \\ \varphi_{i-25} \end{pmatrix} && \text{for } i = 26, \ldots, 50, \\
\psi_i &= \varphi_i && \text{for } i = 1, \ldots, 25,
\end{aligned}$$

Furthermore, we assume that we have imposed Dirichlet boundary conditions on the whole boundary $\partial\Omega$. In consequence, all degrees of freedom for the velocity variables u and v vanish on the boundary, hence the corresponding ansatz and test functions can be removed from the discrete spaces. We obtain

$$\begin{aligned}
\mathcal{V}_h &:= \operatorname{span}\{\phi_i : i \in \{7,8,9,12,13,14,17,18,19,32,33,34,37,38,39,42,43,44\}\} \\
\mathcal{W}_h &:= \operatorname{span}\{\psi_i : i = 1, \ldots, 25\} \cap \{\int_{[0,1]^2} \sum_{i=1}^{25} w_i \psi_i = 0\}.
\end{aligned}$$

Now , the discrete pressure space $\mathcal{W}_h$ has dimension 24 (25 degrees of freedom at the nodes minus one for the constant function), while the velocity space $\mathcal{V}_h$ only has 18 degrees of freedom (9 for each component, as boundary values are already eliminated). Hence, the matrix B^T cannot be injective and the matrix

$$\begin{pmatrix} A & B^T \\ B & 0 \end{pmatrix}$$

is singular.

To ensure a stable (invertible) system, we introduce discrete counterparts of (4.13)–(4.14), i.e. we require that these conditions hold for discrete spaces $\mathcal{V}_h$ and $\mathcal{W}_h$ and discrete operators A and B,

$$\inf_{0\neq u_0\in V_{h,0}} \sup_{0\neq v\in\mathcal{V}_h} \frac{\langle Au_0, v\rangle}{\|u_0\|_{\mathcal{V}_h}\|v\|_{\mathcal{V}_h}} \geq a_h > 0, \tag{4.33}$$

$$\inf_{0\neq p\in\mathcal{W}_h} \sup_{0\neq v\in\mathcal{V}_h} \frac{\langle B'w, v\rangle}{\|p\|_{\mathcal{W}_h}\|v\|_{\mathcal{V}_h}} \geq b_h > 0. \tag{4.34}$$

Here $V_{h,0}$ is defined analogously to (4.11),

$$V_{h,0} := \{v \in \mathcal{V}_h : Bv = 0\}.$$

Theorem 4.3 *([Hac86], Theorem 12.3.6) Let* $\mathsf{a}(\cdot,\cdot) : \mathsf{V}\times\mathsf{V} \to \mathbb{R}$ *and* $\mathsf{b}(\cdot,\cdot) : \mathsf{W}\times\mathsf{V} \to \mathbb{R}$ *be continuous,*

$$\mathsf{a}(\mathsf{u},\mathsf{v}) \leq C_a\|\mathsf{u}\|_{\mathsf{V}}\|\mathsf{v}\|_{\mathsf{V}}, \quad \mathsf{b}(\mathsf{p},\mathsf{v}) \leq C_b\|\mathsf{p}\|_{\mathsf{W}}\|\mathsf{v}\|_{\mathsf{V}}$$

$\mathsf{f} \in \mathsf{V}'$ *and* $\mathsf{g} \in \mathsf{W}'$*, and* $\dim \mathcal{V}_h < \infty$*. Brezzi's conditions* (4.33)–(4.34) *are sufficient and necessary for the existence of a unique solution for the discrete problem* (4.31)–(4.32)*. The discrete solution* (u, p) *satisfies*

$$\left(\|u\|_{\mathsf{V}}^2 + \|p\|_{\mathsf{W}}^2\right)^{\frac{1}{2}} \leq C_h \left(\|f\|_{\mathsf{V}'}^2 + \|g\|_{\mathsf{W}'}^2\right)^{\frac{1}{2}},$$

where C_h *depends on* a_h *and* c_h *as well as* C_a *and* C_b*. If for a sequence of discretizations we have*

$$a_h \geq a > 0, \quad b_h \geq b > 0$$

for all parameters h*, then the discretization is called* stable *and the* C_h *are bounded,* $C_h < C$ *for all* h*.*

In the case of Stokes' equation with Dirichlet boundary conditions, we need $\mathcal{V}_h \subset (H_0^1(\Omega))^2$ and $\mathcal{W}_h \subset L_0^2(\Omega)$. For bounded Ω, (4.33) is satisfied. The inf-sup condition (4.34) translates into

$$\inf_{0\neq p\in\mathcal{W}_h} \sup_{0\neq v\in\mathcal{V}_h} \frac{|\mathsf{b}(p,v)|}{\|p\|_{L^2(\Omega)}\|v\|_{H^1(\Omega)}} \geq b_h > 0. \tag{4.35}$$

Alternatively, we can use a higher order ansatz for the pressure, $\mathcal{W}_h \subset H^1(\Omega)$. Then, if Ω is a Lipschitz bounded domain, the Poisson problem has H^2-regularity and if we had an approximation property of the form

$$\inf\{\|u - u_h\|_{H^1(\Omega)} : u_h \in \mathcal{V}_h\} \leq C_A h\|u\|_{H^2(\Omega)} \text{ for all } u \in H^2(\Omega)\cap H_0^1(\Omega)$$

as well as an inverse inequality

$$\|u_h\|_{H^1(\Omega)} \leq C_I h^{-1}\|u_h\|_{L^2(\Omega)} \text{ for all } u_h \in \mathcal{V}_h,$$

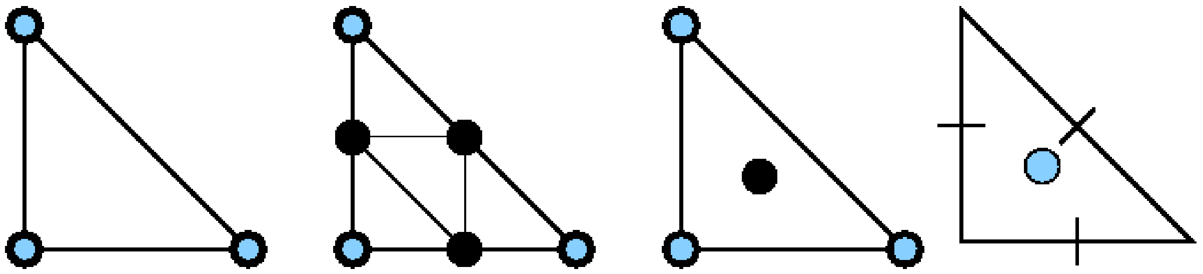

(a) $P_1 - P_1$ element (b) $P_1isoP2_2 - P1$ element (c) Bubble element (d) Crouzeix-Raviart element

Figure 4.4.: Triangular reference elements for saddle point problems. The black dots denote degrees of freedom for velocity, the blue dots denote pressure degrees of freedom.

then the following condition is sufficient for (4.35), see [Hac86], Theorem 12.3.8,

$$\inf_{0 \neq p \in \mathcal{W}_h} \sup_{0 \neq v \in \mathcal{V}_h} \frac{|\mathsf{b}(p,v)|}{\|p\|_{H^1(\Omega)} \|v\|_{L^2(\Omega)}} \geq b_h > 0. \tag{4.36}$$

In the following, we introduce some finite element methods that fulfill one of these inf-sup conditions. One idea is to increase the number of degrees of freedom for the velocity components, while another approach is to introduce an additional stability term. Beside the techniques described in the following, there are many more approaches to create stable finite element methods for saddle point systems. We refer to [GS98] and [ESW05] for a detailed introduction. We use the notation from [Wab03].

1 The first idea is to replace the piecewise linear ansatz functions for the velocity by piecewise quadratic bases. We obtain

$$\begin{aligned} \mathcal{V}_h &: \text{ piecewise } d\text{-dimensional quadratic elements for the triangulation } \mathcal{T}_h, \\ \mathcal{W}_h &: \text{ piecewise linear elements for the triangulation } \mathcal{T}_h. \end{aligned} \tag{4.37}$$

This approach is called *Taylor-Hood element*, or $P_2 - P_1$ element, i.e. quadratic ansatz functions for velocity and linear ansatz functions for pressure.

2 An alternative version of the Taylor-Hood element is denoted by $P_1isoP_2 - P_1$. We uniformly refine the triangulation $\mathcal{T}_h$ to obtain the triangulation $\mathcal{T}_{\frac{h}{2}}$ (new nodes are created at the midpoints of each edge , see Figure 4.4(b)) and use this refined triangulation for the piecewise linear discretization of the velocity only.

$$\begin{aligned} \mathcal{V}_h &: \text{ piecewise } d\text{-dimensional linear elements for the triangulation } \mathcal{T}_{\frac{h}{2}}, \\ \mathcal{W}_h &: \text{ piecewise linear elements for the triangulation } \mathcal{T}_h. \end{aligned} \tag{4.38}$$

3 Another approach is to enrich the space $\mathcal{V}_h$ by one *bubble* function per triangle $T \in \mathcal{T}$. For example, in two spatial dimensions we define,

$$\varphi_{bubble}(x,y) := xy(1-x-y)$$

on the reference element $\{x, y \geq 0,\ x + y \leq 1\}$ and $\varphi_{bubble} = 0$ else. The discrete spaces $\mathcal{V}_h$ and $\mathcal{W}_h$ for this so-called MINI element (Figure 4.4(c)) are then defined by

$$\begin{aligned} V_h^1 :\ & \text{linear combinations of piecewise scalar linear elements} \\ & \text{and bubble functions for the triangulation } \mathcal{T}_h, \\ \mathcal{V}_h :=\ & \left(V_h^1\right)^d, \\ \mathcal{W}_h :\ & \text{piecewise linear elements for the triangulation } \mathcal{T}_h. \end{aligned} \tag{4.39}$$

For each triangle T, the scalar products $\int \varphi_{bubble,T}\varphi_i$ are nonzero only for those φ_i that correspond to the nodes of the triangle T. Hence, it is easy to eliminate the degrees of freedom that belong to the bubble function and to obtain a linear system for the nodal functions only. This introduces an additional nonzero matrix block $-C$ in the lower right part of $\mathcal{K}$.

4 The Crouzeix-Raviart element or $P_1^{nc} - P_0$ element employs a non-conforming ansatz for the velocity, i.e. $\mathcal{V}_h \not\subset H^1(\Omega)^d$.

$$\begin{aligned} V_h^1 :\ & \{v_h \in L^2(\Omega)^d : v_h \text{ is piecewise linear per triangle } \tau \in \mathcal{T}_h, \\ & v_h \text{ is continuous at the midpoint of all element edges (faces) } \} \\ \mathcal{V}_h :=\ & \left(V_h^1\right)^d \\ \mathcal{W}_h :\ & \text{piecewise constant functions per triangle } \tau \in \mathcal{T}_h. \end{aligned} \tag{4.40}$$

In contrast to the elements described above, the degrees of freedom for the velocity are determined at the midpoints of the edges (faces) of the element. The pressure is determined at the center of the triangle, see Figure 4.4(d). In an alternative definition of this elements, the velocity basis is constructed such that all $v_h \in \mathcal{V}_h$ are divergence-free per element $\tau \in \mathcal{T}_h$ and the pressure component is eliminated, see [Bra97].

Theorem 4.4 *([Hac86], Theorem 12.3.11 and Theorem 12.3.12) Let $\mathcal{T}_h$ be a quasi-uniform triangulation for the polygonally bounded domain Ω and let $\mathcal{V}_h$ and $\mathcal{W}_h$ be defined by either* (4.37), (4.38), *or* (4.39). *Then, the stability criterion* (4.36) *is satisfied. If, in addition, Ω is Lipschitz bounded and the Poisson problem is H^2-regular, then also* (4.35) *holds.*

Instead of adding degrees of freedom to the velocity space, we can introduce an additional stability term into the lower right component of the matrix, i.e.

$$\mathcal{K} = \begin{pmatrix} A & B^T \\ B & -C \end{pmatrix},$$

where $C \in \mathbb{R}^{M \times M}$ is a symmetric positive semi-definite matrix. In the case of $\partial\Omega = \Gamma_D$, the (discretization of the) constant vector must lie within the nullspace of C. For a uniform grid, it is possible to use a discrete Laplacian (with Neumann boundary conditions), i.e. instead of (4.18) we discretize

$$-\sum_{j=1}^{N}\int_\Omega \psi_i \nabla\cdot\phi_j - \beta h^2 \sum_{j=1}^{M}\int_\Omega \nabla\psi_i\nabla\psi_j = 0 \text{ for all } i=1,\dots,M. \tag{4.41}$$

This stabilization technique however involves a parameter β that needs to be carefully chosen. Moreover, in the case of non-uniform meshes the geometry of the mesh must be reflected in the Laplacian (by introducing some anisotropy). A parameter-free and more robust alternative is given by [BDG06],

$$C_{ij} := \int_\Omega (\psi_i - \Pi\psi_i, \psi_j - \Pi\psi_j), \tag{4.42}$$

where Π is the projection on the space of functions that are constant on each element. In other words, C is a mass matrix for the basis $\{\psi_j\}_{=1}^{M}$ minus an averaging operator. The stabilization techniques (4.41) and (4.42) can be applied not only to triangular (2D) or tetrahedral (3D) elements ($P_1 - P_1$ elements), but also to rectangular and brick elements ($Q_1 - Q_1$-elements).

Lemma 4.1 *([BDG06], Corollary 2.4) Let $\mathcal{V}_h$ and $\mathcal{W}_h$ given by*

$$\begin{aligned}\mathcal{V}_h &:= \left(\left\{v\in C^0(\Omega):\ v|_T \in P_1(T) \text{ for all } T\subset\mathcal{T}\right\}\cap H_0^1(\Omega)\right)^d\\ \mathcal{W}_h &:= \left\{v\in C^0(\Omega):\ v|_T \in P_1(T) \text{ for all } T\subset\mathcal{T}\right\}\cap L_0^2(\Omega),\end{aligned}$$

in the case of simplicial elements, or,

$$\begin{aligned}\mathcal{V}_h &:= \left(\left\{v\in C^0(\Omega):\ v|_T = \hat{v}\circ F^{-1};\ \ \hat{v}\in Q_1(T_{ref})\right\}\cap H_0^1(\Omega)\right)^d\\ \mathcal{W}_h &:= \left\{v\in C^0(\Omega):\ v|_T = \hat{v}\circ F^{-1};\ \ \hat{v}\in Q_1(T_{ref})\right\}\cap L_0^2(\Omega)\end{aligned}$$

in the case of quadrilateral and hexahedral elements. Here, $P_1(T)$ denotes the space of all linear polynomials on T and $Q_1(T_{ref})$ denotes the space of all polynomials on the reference element T_{ref} whose degree does not exceed 1 in each coordinate direction, and $F: T_{ref} \to T$ is a bilinear or trilinear mapping. Furthermore, let the corresponding Poisson problem be H^2-regular. Then, there exist constants c_1 and c_2 independent on h such that the weak inf-sup condition

$$\sup_{v\in V}\frac{\int_\Omega p\nabla\cdot v}{\|v\|_{H^1(\Omega)^d}} \geq c_1\|p\|_{L^2(\Omega)} - c_2\|(I-\Pi)p\|_{L^2(\Omega)} \text{ for all } p\in\mathcal{W}_h \tag{4.43}$$

holds.

We see that the stability matrix C appears on the right hand side of the modified inf-sup condition (4.43). It remains to show that this condition is also sufficient for the solution of the discrete problem $\mathcal{K}x = y$.

Theorem 4.5 *([BDG06], Theorem 4.1)*

1 *If* $\Pi : L^2(\Omega) \to L^2(\Omega)$ *is continuous (i.e.* $\|\Pi p\|_{L^2(\Omega)} \leq c_\Pi \|p\|_{L^2(\Omega}$ *for all* $p \in L^2(\Omega)$*, we have*

$$(u,p)^T \mathcal{K}(v,q) \leq c_{\mathcal{K}} \left(\|u\|_{H^1(\Omega)^d} + \|p\|_{L^2(\Omega)}\right)\left(\|v\|_{H^1(\Omega)^d} + \|q\|_{L^2(\Omega)}\right) \tag{4.44}$$

for all (u,p) *and* (v,q) *in* $\mathcal{V}_h \times \mathcal{W}_h$*.*

2 *Under the assumptions of Lemma 4.1, there exists some constant* $\zeta > 0$ *such that*

$$\sup_{0\neq v\in\mathcal{V}_h,\ 0\neq q\in\mathcal{W}_h} \frac{(u,p)^T\mathcal{K}(v,q)}{\|v\|_{H^1(\Omega)^d} + \|q\|_{L^2(\Omega)}} \geq \zeta\left(\|u\|_{H^1(\Omega)^d} + \|p\|_{L^2(\Omega)}\right) \tag{4.45}$$

for all $(u,p) \in \mathcal{V}_h \times \mathcal{W}_h$*.*

The message of this theorem is that the following variational problem is well-posed. Seek $(u,p) \in V_h \times W_h$ such that

$$u^TAv + p^TBv + u^TB^Tq - p^TCq = v^Tf + q^Tg \ \text{ for all } (v,q) \in V_h \times W_h.$$

We have re-gained stability by adding an additional term. This idea will also play an important role to ensure the stability of the coarse level systems for our saddle point algebraic multigrid method.

4.3. Iterative solvers for Saddle Point Systems

We now discuss several iterative solution methods for saddle point systems. First, we introduce the *generalized minimal residual (GMRES)* algorithm, which, in exact arithmetic, converges within m iterations for any non-singular matrix $K \in \mathbb{R}^{m\times m}$. Then, we describe some iterative solvers and smoothers aimed at saddle point matrices.

4.3.1. GMRES

The *generalized minimal residual (GMRES)* method [SS86] is a generalization of the *minimal residual (MINRES)* method introduced in [PS75]. While the latter requires that $\mathcal{K}$ is symmetric and an optional preconditioner $\mathcal{Q}$ must be symmetric positive definite, the GMRES method is not restricted to this case.
Both methods belong to the class of *Krylov subspace* methods. A Krylov subspace is spanned by powers of K applied to the right hand side y,

$$\mathfrak{K}^{it}(\mathcal{K}, y) := \operatorname{span}\{y, \mathcal{K}y, \dots, \mathcal{K}^{it-1}y\}.$$

After each iteration it, the current approximation x^{it} minimizes the residual over the Krylov space,

$$\min_{x^{it}\in\mathfrak{K}^{it}(K,y)} \|y - \mathcal{K}x^{it}\|_E.$$

To this end, the GMRES algorithm constructs an orthonormal basis of $\mathcal{K}^{it}(\mathcal{K}, y)$. As the number of iterations grows, this requires $O(it \cdot m)$ memory to store all of these vectors. Hence, in practice one often restarts GMRES, i.e. after a certain amount of steps we start the iteration with the current approximation x^{it} as the initial guess and abandon the previously created orthonormal vectors. In Algorithm 4.1 we give the preconditioned restarted GMRES algorithm.

4.3.2. Uzawa methods

The *Uzawa method* is a well-known iterative solver for saddle point systems of the form (4.1). In each iteration, the following equations are solved ([Bra97], algorithm 5.1),

$$Au^{it+1} = f - B^T p^{it} \tag{4.46}$$
$$p^{it+1} = p^{it} + \sigma \left(Bu^{it} - Cp^{it} - g \right). \tag{4.47}$$

Note that the previous velocity iterate u^{it} is not used at all, instead u^{it+1} is computed to solve the momentum equation $Au^{it+1} + B^T p^{it} = f$ given the pressure iterate p^{it}. Hence, for convergence considerations, we only need to monitor the pressure p. We eliminate u^{it} from (4.47) and obtain

$$p^{it+1} = p^{it} + \sigma \left(BA^{-1}B^T + C \right) \left(p - p^{it} \right),$$

where p denotes the pressure part of the exact solution for (4.1). We have convergence for $\sigma < \frac{2}{\|BA^{-1}B^T + C\|}$, see [Bra97].
A drawback of this method is that we must solve $Au^{it+1} = f - B^T p^{it}$ within each iteration, i.e. we need a direct or iterative solver, e.g. geometric or algebraic multigrid, for A.
To circumvent this difficulty, one can choose to solve (4.46) only *inexactly*, i.e. to replace (4.46) by

$$\hat{A} \left(u^{it+1} - u^{it} \right) = f - Au^{it} - B^T p^{it},$$

where $\hat{A}$ is an easily invertible preconditioner for A, e.g. $\hat{A} = \alpha I$ for some number α (Arrow-Hurwicz method [AHU58]).
A symmetric version of the inexact Uzawa method is also possible. The iteration then reads [Wab03, Zul02]

$$u^* = u^{it} + \hat{A}^{-1} \left(f - Au^{it} - B^T p^{it} \right) \tag{4.48}$$
$$p^{it+1} = p^{it} + \hat{S}^{-1} \left(Bu^* - Cp^{it} - g \right) \tag{4.49}$$
$$u^{it+1} = u^* - \hat{A}^{-1} B^T p^{it+1} \left(p^{it+1} - p^{it} \right), \tag{4.50}$$

i.e. we first compute a predictor velocity u^* and then use it to determine u^{it+1} and p^{it+1}. Note that (4.50) can be rewritten as

$$u^{it+1} = u^{it} + \hat{A}^{-1} \left(f - Au^{it} - B^T p^{it+1} \right). \tag{4.51}$$

We will further discuss the inexact symmetric Uzawa method in Section 4.5. The additive version of our saddle point AMG smoother also belongs to this class.

algorithm 4.1 GMRES(k,$\mathcal{K}$, $\mathcal{Q}$,y,x,tol) ([Wab03], algorithm 2.11)

```
begin
  choose initial guess $x^0$;
  $q^1 \leftarrow \mathcal{Q}^{-1}(y - \mathcal{K}x^0)$;                          apply preconditioner
  $z_1 \leftarrow |q^1|$;
  $q^1 \leftarrow \frac{1}{z_1} q^1$;
  while $|z_1| > tol$ do
      for $j \leftarrow 1$ to $k$ do                                    form orthonormal basis
          $q^{j+1} \leftarrow \mathcal{Q}^{-1}\mathcal{K}q^j$;
          for $i \leftarrow 1$ to $it$ do
              $h_{i,j} \leftarrow q^i \cdot q^{j+1}$;
              $q^{j+1} \leftarrow q^{j+1} - h_{i,j}q^i$;
          od;
          $h_{j+1,j} \leftarrow |q^{j+1}|$;
          $q^1 \leftarrow \frac{1}{|q^{j+1}|} q^{j+1}$;
      od
      for $j \leftarrow 1$ to $k$ do                                         seek minimizer
          $d \leftarrow \sqrt{h_{j,j}^2 + h_{j+1,j}^2}$;
          $c \leftarrow \frac{h_{j,j}}{d}$;
          $s \leftarrow \frac{h_{j+1,j}}{d}$;
          $h_{j,j} \leftarrow d$;
          for $i \leftarrow j + 1$ to $k$ do
              $h_{j,i} \leftarrow c \cdot h_{j,i} + s \cdot h_{j+1,i}$;
              $h_{j+1,i} \leftarrow s \cdot h_{j,i} - c \cdot h_{j+1,i}$;
          od;
          $z_j \leftarrow c \cdot z_j$;
          $z_{j+1} \leftarrow s \cdot z_j$;
      od;
      $\alpha_k \leftarrow \frac{z_k}{h_{k,k}}$;
      for $i \leftarrow k$ to $1$ do
          $\alpha_i \leftarrow \frac{1}{h_{i,i}} \left( z_i - \sum_{j=i+1}^{k} h_{i,j}\alpha_j \right)$;
      od;
      $x^k \leftarrow x^0 + \sum_{i=1}^{k} \alpha_i q^i$;                               update iterate
      $r^k \leftarrow \mathcal{Q}^{-1}(y - \mathcal{K}x^k)$;                       preconditioned residual
      $x^0 \leftarrow x^k$; $r^0 \leftarrow r^k$;                                             restart
      $z_1 \leftarrow |r^0|$;
      $q^1 \leftarrow \frac{1}{z_1} r^0$;
  od;
end
```

4.3.3. SIMPLE

The *(semi-implicit method for pressure–linked equations (SIMPLE)* method [PS72, Pat80] is another approach to circumvent the exact solution of (4.46). We determine an auxiliary vector u^* that satisfies

$$\tilde{A}u^* = f - B^T p^{it},$$

where $\tilde{A}$ is some approximation to A. To obtain the next velocity iterate u^{it+1}, we first need to compute a pressure correction q^*,

$$q^* = -\alpha \hat{S}^{-1}\left(Bu^{it} - Cp^{it} - g\right), \tag{4.52}$$

Here, $\hat{S}$ denotes an approximation to the Schur complement $BD^{-1}B^T + C$, where D is the diagonal of A. The damping parameter α must be chosen such that no overshooting occurs. Finally, we obtain the new iterates for velocity and pressure by

$$u^{it+1} = u^* - \alpha D^{-1}B^T q^*, \quad p^{it+1} = p^{it} + q^*. \tag{4.53}$$

It remains to define $\hat{A}^{-1}$ and $\hat{S}^{-1}$, which describe the action of a linear solver. Depending on the size and the shape of A and $BD^{-1}B^T + C$, we might choose direct methods, relaxation processes like Jacobi or Gauss–Seidel, or even (algebraic) multigrid methods. Several variants of the SIMPLE method have been developed (SIMPLER, SIMPLEV, SIMPLEC [VDR84]). Note that, as in the exact Uzawa iteration, the previous velocity iterate u^{it} is not used to obtain u^{it+1} and p^{it}.

4.3.4. Transforming Smoothers

The idea of transforming smoothers [Wit89, Wit90] is to multiply the matrix $\mathcal{K}$ from the left and the right by some non-singular matrices $\mathcal{K}_L$ and $\mathcal{K}_R$ and to apply a standard smoothing method to the transformed system,

$$z^{it+1} = z^{it} + \hat{\mathcal{K}}^{-1}\left(\mathcal{K}_L y - \mathcal{K}_L \mathcal{K}\mathcal{K}_R z^{it}\right),$$

In terms of the original unknown x, we use $x = \mathcal{K}_R^{-1} z$ and re-write the iteration,

$$x^{it+1} = x^{it} + \mathcal{K}_R \hat{\mathcal{K}}^{-1}\mathcal{K}_L\left(y - \mathcal{K}x^{it}\right).$$

A possible choice is ([Wit89], Section 3.1)

$$\mathcal{K}_L = I, \quad \mathcal{K}_R = \begin{pmatrix} I & A^{-1}B^T\left(BA^{-1}B^T + C\right)F \\ 0 & -\left(BA^{-1}B^T + C\right)^{-1}F \end{pmatrix}$$

with some positive definite matrix F. for example, one can take $F = \left(BA^{-1}B^T + C\right)$, or a discretization of the Laplacian on the pressure grid. The transformed system takes the form

$$\mathcal{K}\mathcal{K}_R = \begin{pmatrix} A & 0 \\ B & F \end{pmatrix}.$$

A suitable smoother for the transformed system only needs to posses the smoothing property for the diagonal blocks A and F [Wit90], i.e. we can set

$$\hat{\mathcal{K}} = \begin{pmatrix} K_U & 0 \\ 0 & K_P \end{pmatrix}$$

where K_U and K_P can be chosen e.g. to be the scaled diagonals (Jacobi), the diagonals and the lower right blocks (Gauss–Seidel) or an incomplete LU discretization of A and F, respectively. Under certain assumptions on the discretization, the smoothing ratio for the transforming smoothers applied to a Stokes problem can be shown to range from $O(\frac{1}{\sqrt{it}})$ (damped Jacobi) to $O(\frac{\ln it}{\sqrt{it}})$ (ILU), where it denotes the number of iterations ([Wit90], Theorems 3.1.4, 3.1.7, 3.2.1).

4.3.5. Braess–Sarazin smoother

In contrast to the exact Uzawa and the SIMPLE methods, the *Braess–Sarazin smoother* [BS97] computes the iterates u^{it+1} and p^{it+1} from the old velocity iterate u^{it}. To this end, let $\hat{A} = \alpha\ diag(A)$, where α is chosen such that $\hat{A} > A$. Furthermore, let $\hat{S}$ be a preconditioner or approximate solver for the Schur complement $B\hat{A}^{-1}B^T + C$. Then we can compute the iterates u^{it+1} and p^{it+1} as follows,

$$u^* = u^{it} + \hat{A}^{-1}\left(f - Au^{it}\right), \tag{4.54}$$

$$p^{it+1} = -\hat{S}^{-1}\left[g - Bu^*\right], \tag{4.55}$$

$$u^{it+1} = u^* - \hat{A}^{-1}B^T p^{it+1}. \tag{4.56}$$

It can be shown that the Braess-Sarazin smoother possesses a smoothing ratio of $O(\frac{1}{it})$ ([BS97], Lemma 3.2 and Theorem 5.1).

4.3.6. Vanka Smoothers

The smoother introduced in [Van86] treats velocity and pressure updates simultaneously. This method was first introduced within the context of a multigrid method for staggered mesh discretizations of the Navier–Stokes equations. Here, a global symmetric block Gauss–Seidel iteration processes all discretization cells. On each such cell, a small saddle point system is solved to obtain an update for the pressure unknown and velocity unknowns belonging to this cell.
The multiplicative variant of our smoother for saddle point AMG also belongs to this class. We postpone a further discussion to Section 4.6.

4.4. Towards AMG

We now summarize some previous approaches to the algebraic construction of a multigrid hierarchy for saddle point systems.

We start with an AMG method for constrained systems, especially contact problems, [Ada04] of the form

$$\begin{pmatrix} A & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} u \\ \lambda \end{pmatrix} = \begin{pmatrix} f \\ g \end{pmatrix}$$

where A is obtained from the discretization of a PDE and B comprises the constraint equations (contacts). An AMG method for the A part is assumed to be available, that is, we have a hierarchy of interpolation operators $\{P_l\}_{l=1}^{L-1}$, coarse grid operators $\{A_l\}_{l=1}^{L}$ and smoothers denoted by M_l, $l = 1, \dots, L$. It remains to construct interpolation operators $\{\bar{P}_l\}_{l=1}^{L}$ for the Lagrangian multipliers. Then, the overall coarse operator can be constructed as

$$\begin{pmatrix} P_l^T & 0 \\ 0 & \bar{P}_l^T \end{pmatrix} \begin{pmatrix} A_l & B_l^T \\ B_l & 0 \end{pmatrix} \begin{pmatrix} P_l & 0 \\ 0 & \bar{P}_l \end{pmatrix}.$$

The main idea is to compute a symmetric auxiliary matrix G_l for the Lagrangian multiplier space and to derive the coarse grid and the interpolation operator by using plain unsmoothed aggregation AMG techniques (see Section 2.11.2) applied to G_l. Within the context of contact problems, the ansatz

$$G_l := B_l P_l P_l^T B_l^T$$

is preferred, as this allows to maximize the angles between the coarse level constraints $B_{l+1} = \bar{P}_l^T B_l P_l$, see [Ada04] for details. The constraint interpolation $\bar{P}_l$ is constant per aggregate and scaled such that $\bar{P}_l^T \bar{P}_l$ is the identity.
Smoothing is performed by either symmetric inexact Uzawa smoothing 4.3.2 or a Schwarz subdomain smoother, where each subdomain consists of a subset of the constraints as well as the primal variables involved.
A special case is the saddle point AMG proposed in [LOS04]. Here the constraints describe the Dirichlet boundary conditions for a reproducing kernel particle method (RKPM) applied to an elliptic partial differential equation. More precisely, the entries of B are computed by

$$b_{ij} := \int_{\Gamma_D} \tilde{\psi}_i \operatorname{tr} \psi_j$$

where the ansatz functions ψ_j are defined throughout the domain Ω and are used for the discretization of the PDE, while the basis functions $\tilde{\psi}_i$ are defined on the boundary $\Gamma_D \subset \partial\Omega$. Here, tr denotes the trace operator on the boundary Γ_D.
The coarsening process for the constraints employs a matrix C defined by

$$C_{ij} := \int_{\Gamma_D} \tilde{\psi}_i \tilde{\psi}_j.$$

Then, a smoothed aggregation algorithm 2.11.2 is applied to both A and C to obtain interpolation operators P and $\bar{P}$ for the ansatz and the constraint space, respectively. the overall interpolation operator is assembled as

$$\mathcal{P} = \begin{pmatrix} P & 0 \\ 0 & \bar{P} \end{pmatrix}$$

and the coarse grid operator is computed by the Galerkin product $\mathcal{P}^T\mathcal{K}\mathcal{P}$.
The smoother employed here takes the form of a Braess-Sarazin method (4.54)–(4.56),

$$\begin{aligned}\lambda^{it+1} &\leftarrow \left[B\omega D^{-1}B^T\right]^{-1}\left[B\left(u^{it}+\omega D^{-1}(f-Au^{it})\right)-g\right]\\ u^{it+1} &\leftarrow u^{it}+\omega D^{-1}(f-Au^{it})-\omega D^{-1}B^T\lambda^{it+1},\end{aligned}$$

that is a ω-damped Jacobi iteration for the ansatz space, while a direct solver is used to compute $\left[B\omega D^{-1}B^T\right]^{-1}$.
In [Wab03], [Wab04], [Wab06], a monolithic semi-algebraic AMG approach to the solution of the Navier-Stokes equations is described. To this end, the incompressible Navier-Stokes equations

$$\frac{\partial}{\partial t}\mathsf{u}-\nu\nabla\cdot(\nabla\mathsf{u})+(\mathsf{u}\cdot\nabla)\,\mathsf{u}+\nabla\mathsf{p} = f \tag{4.57}$$

$$-\nabla\cdot\mathsf{u} = 0 \tag{4.58}$$

given on a domain $\Omega\subset\mathbb{R}^d$, $d=2,3$, are linearized to obtain the *Oseen equations*,

$$\frac{\partial}{\partial t}\mathsf{u}-\nu\nabla\cdot(\nabla\mathsf{u})+(\mathsf{w}\cdot\nabla)\,\mathsf{u}+\nabla\mathsf{p} = f \tag{4.59}$$

$$-\nabla\cdot\mathsf{u} = 0. \tag{4.60}$$

Here, (assuming that the fluid is Newtonian, i.e. has a constant shear to stress ratio,) the kinematic velocity $\nu := \frac{\mu}{\rho}$ is defined as the quotient of the dynamic viscosity μ and the density ρ, which both depend on the material. As in the case of Stokes' equations, we additionally introduce boundary conditions on $\partial\Omega = \Gamma_D\dot\cup\Gamma_N$,

$$\mathsf{u}(x) = \mathsf{r}(x) \text{ for all } x\in\Gamma_D, \tag{4.61}$$

$$\nu\frac{\partial\mathsf{u}}{\partial n}(x)-n\mathsf{p}(x) = \mathsf{s}(x) \text{ for all } x\in\Gamma_N, \tag{4.62}$$

and (as we consider an instationary problem), initial conditions for $t=0$,

$$\mathsf{u}|_{t=0}=\mathsf{u}_0,\quad \mathsf{p}|_{t=0}=\mathsf{p}_0. \tag{4.63}$$

For a detailed introduction to these equations, we refer to [GS98] and [ESW05].
The linear Oseen equations (4.59)–(4.60) can be used inside a fixed point iteration for (4.57)–(4.58), where the newly introduced vector field w is set to the previous approximation of u. Now, using a finite element discretization for the spatial variables and a time-stepping scheme, we end up with a linear system of equations of the form

$$\mathcal{K}(w)x=\begin{pmatrix}A(w) & B^T\\ B & -C\end{pmatrix}\begin{pmatrix}u\\ p\end{pmatrix}=\begin{pmatrix}f\\ g\end{pmatrix}.$$

Here,

$$A(w)=c_1M+A_D+c_2A_C(w)+c_3A_S(w)+c_4A_R(w) \tag{4.64}$$

consists of a mass matrix M which stems from the time discretization, a diffusion matrix (Laplacian) A_D, non-symmetric convection and reaction matrices $A_C(w)$ and $A_R(w)$ and a symmetric positive definite convection stabilization matrix $A_S(w)$. Depending on the discretization, C is zero or a pressure stabilization matrix, see Section 4.2.2.
Depending on the element chosen, the coarsening is either performed element-wise (i.e. AMGe, cf. Section 2.11.1) for the Crouzeix-Raviart element (4.40), or point-wise (classical Ruge-Stüben) by coarsening the nodes of the finite element mesh. In case of a $P_1 - P_1$-stab discretization (4.41), the same coarse mesh is used for all physical unknowns u and p, while in case of the $P_1 iso P_2 - P1$-element (4.38), the pressure mesh is coarsened first and the velocity mesh at each level $l \geq 2$ is identical to the pressure mesh at level $l - 1$[1]. In all cases, the interpolation is unknown-wise, i.e.

$$\mathcal{P} = \begin{pmatrix} P_{\mathcal{V}_{(1)}} & 0 & 0 \\ 0 & P_{\mathcal{V}_{(2)}} & 0 \\ 0 & 0 & P_{\mathcal{W}} \end{pmatrix} \text{ or } \mathcal{P} = \begin{pmatrix} P_{\mathcal{V}_{(1)}} & 0 & 0 & 0 \\ 0 & P_{\mathcal{V}_{(2)}} & 0 & 0 \\ 0 & 0 & P_{\mathcal{V}_{(3)}} & 0 \\ 0 & 0 & 0 & P_{\mathcal{W}} \end{pmatrix}$$

where each $P_{\mathcal{V}_{(i)}}$, $i = 1, \ldots, d$ is a scalar interpolation scheme. The coarse level system is computed by the Galerkin product with two slight modifications: the coarse convection stabilization term $A_{S,l+1}$ part needs to be rescaled,

$$A_{S,l+1} := \sqrt[d]{\frac{n_l}{n_{l+1}}} P_{\mathcal{V}^l}^T A_{S,l} P_{\mathcal{V}^l}$$

to prevent oscillations, and, in the case of the P1-P1-stab element, the matrix C_{l+1} needs to be computed as

$$C_{l+1} := \frac{\lambda_{\max}(D_l^{-1} M_l)}{h^2} P_{\mathcal{W}^l}^T C_l P_{\mathcal{W}^l},$$

where $M_l := P_{\mathcal{W}^l}^T \cdots P_{\mathcal{W}_1}^T M P_{\mathcal{W}_1} \cdots P_{\mathcal{W}^l}$ is the Galerkin projection of the fine level (pressure) mass matrix M to level l, D_l is the diagonal of one of the component blocks of $A_{D,l}$, and h is the mesh width on the finest level.
The stability of the coarse level systems, i.e. the existence of a discrete inf-sup condition on all levels can only be shown with additional geometric information. In the case of P1isoP2-elements, a rigorous analysis is not known and stability can only be motivated heuristically. We refer to [Wab03], [Wab04], and [Wab06] for details. We now give the stability lemma for the P1-P1-stab element, which will be the basis of our more general stability results presented in Sections 4.8–4.10.

Lemma 4.2 *([Wab03], Lemma 4.3) Assume that for all elements $\tau \in \mathcal{T}_h$ the diameter h_τ fulfills*

$$\underline{\alpha} h \leq h_\tau \leq \overline{\alpha} h,$$

[1]For stability reasons, it might be necessary to use the pressure mesh at level $l - 2$ instead for $l \geq 3$ and let the velocity mesh at level 3 be identical to the finest mesh.

with positive constants $\underline{\alpha}$ and $\overline{\alpha}$ and the discretization parameter h, and assume further that $A_{D,l}$ is symmetric and of essentially positive type (2.28) *and that for all $v_l \in \mathcal{V}^l$ we can find $\Pi_l v_l \in V_{l+1}$ such that*

$$\|v_l - P_{\mathcal{V}_l}\Pi_l\|^2_{D_l} \leq \beta \|v_l\|^2_{A_{D,l}}, \tag{4.65}$$

with some constant β. Then for all levels $l \in \{1, \dots, L\}$ there exist positive c_l and d_l such that

$$\sup_{0 \neq v \in \mathcal{V}^l} \frac{vB_l^T p}{\|v\|_{A_{D,l}}} \geq c_l \|p\|_{M_l} - d_l \left(p^T C_l p\right)^{\frac{1}{2}} \text{ for all } p \in \mathcal{W}^l.$$

The approximation property 4.65 is fulfilled for many of the classical AMG interpolation operators, see Section 2.8.

In the remainder of this chapter, we describe the components of our algebraic multigrid approach to saddle point problems. Unlike the ansatzes introduced above, we are not restricted to specific discretizations or geometric information. We only need to know the decomposition of the saddle point matrix $\mathcal{K}$,

$$\mathcal{K} = \begin{pmatrix} A & B^T \\ B & -C \end{pmatrix}.$$

In many applications, the vector u describes discretization of two or three physical quantities (e.g. the velocity components in different spatial directions). In consequence, A itself is a discretization of a system of elliptic partial differential equations. We have already presented three approaches to system AMG methods in Section 2.12: First, one can ignore the decomposition into the physical unknowns and just apply AMG to A (VAMG, Section 2.12.1). The second possibility is to split A by the physical components (velocity directions) and to apply scalar AMG component-wise (UAMG, Section 2.12.2). The third ansatz is to utilize additional information to sort the matrix A by (discretization) points (PAMG, Section 2.12.3). All of them can be combined with the techniques described below to obtain an algebraic multigrid method for saddle point systems.

4.5. The smoother I: An inexact Uzawa scheme

In this section, we introduce a convergence theory for a class of inexact Uzawa relaxation schemes developed by [SZ03]. This theory can be applied to any predictor-corrector scheme of the form (4.48)–(4.50),

$$\begin{aligned} u^* &= u^{it} + \hat{A}^{-1}\left(f - Au^{it} - B^T p^{it}\right) \\ p^{it+1} &= p^{it} + \hat{S}^{-1}\left(Bu^* - Cp^{it} - g\right) \\ u^{it+1} &= u^{it} + \hat{A}^{-1}\left(f - Au^{it} - B^T p^{it+1}\right), \end{aligned}$$

where $\hat{A}$ and $\hat{S}$ are symmetric positive definite matrices that serve as preconditioners for A and $B\hat{A}^{-1}B^T + C$ respectively. We require that

$$\hat{A} > A \text{ and } \hat{S} > B\hat{A}^{-1}B^T + C \tag{4.66}$$

where $E > F$ again means that the matrix $E - F$ is positive definite.

Remark 4.1 A convergence theory is also possible for $\hat{A} < A$ and $\hat{S} > B\hat{A}^{-1}B^T + C$ or $\hat{A} > A$ in combination with $\hat{S} < B\hat{A}^{-1}B^T + C$. Then, we can even re-interpret the smoother as an iteration over a symmetric positive definite matrix $\mathcal{L}$,

$$\mathcal{M} = I - \mathcal{Q}^{-1}\mathcal{L}$$

where $\mathcal{L} = \mathcal{Q}\hat{\mathcal{K}}^{-1}\mathcal{K}$,

$$\mathcal{Q} = \pm \begin{pmatrix} A - \hat{A} & 0 \\ 0 & \hat{S} - B\hat{A}^{-1}B^T - C \end{pmatrix},$$

and the sign is chosen such that $\mathcal{Q}$ is positive definite. We refer to Appendix A as well as [Zul00] and [Zul02] for a discussion of these cases.

The most basic approach to the choice of $\hat{A}$ and $\hat{S}$ is to use scaled versions of the diagonals of A and $B\hat{A}^{-1}B^T + C$ such that (4.66) is satisfied. In this case, their inverse is still a diagonal matrix, which will be advantageous for our AMG setup.
Now, the indefinite matrix $\mathcal{K}$ does not define an inner product and norm. Hence, unlike the symmetric positive definite case, we cannot show an AMG convergence theory in terms of the norms $\|\cdot\|_0$, $\|\cdot\|_1$ and $\|\cdot\|_2$ induced by the scalar products (2.22) – (2.24),

$$(u, v)_0 = (Du, v), \quad (u, v)_1 = (Au, v), \quad (u, v)_2 = (D^{-1}Au, Av)$$

which can only be defined for symmetric positive definite A with diagonal D.
To overcome this sticking point, we will introduce a symmetric positive definite matrix $\mathcal{Q}$ that defines an inner product $(\cdot, \cdot)_{\mathcal{Q}}$ and a norm $\|\cdot\|_{\mathcal{Q}}$. Not only can we estimate the error propagation of the inexact Uzawa smoothers in terms of this norm, it also helps us to transfer the convergence theory from Section 2.10 to our setting.
First, we write the inexact Uzawa scheme in matrix form,

$$\begin{pmatrix} u \\ p \end{pmatrix}^{it+1} = \begin{pmatrix} u \\ p \end{pmatrix}^{it} + \hat{\mathcal{K}}^{-1}\left(\begin{pmatrix} f \\ g \end{pmatrix} - \mathcal{K}\begin{pmatrix} u \\ p \end{pmatrix}^{it}\right) \text{ where } \hat{\mathcal{K}} = \begin{pmatrix} \hat{A} & B^T \\ B & B^T\hat{A}^{-1}B^T - \hat{S} \end{pmatrix}$$

and introduce the corresponding error propagation matrix $\mathcal{M}$,

$$\mathcal{M} = \left(I - \hat{\mathcal{K}}^{-1}\mathcal{K}\right). \tag{4.67}$$

Lemma 4.3 *([SZ03], Lemma 1) Let $\hat{A}$ be a symmetric positive definite $N \times N$ matrix, and $\hat{S}$ a symmetric positive $M \times M$ matrix, satisfying,*

$$\hat{A} > A \text{ and } \hat{S} > C + B\hat{A}^{-1}B^T$$

Then we have:
The iteration matrix $\mathcal{M} = I - \hat{K}^{-1}K$ can be written in the form

$$\mathcal{M} = \mathcal{Q}^{-\frac{1}{2}}\overline{\mathcal{M}}\mathcal{Q}^{\frac{1}{2}} \tag{4.68}$$

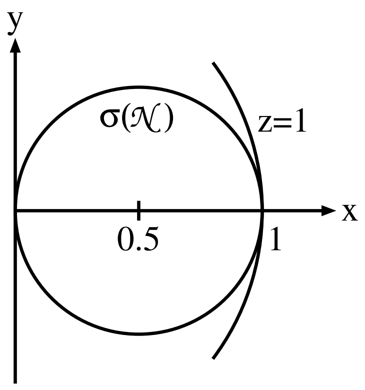


Figure 4.5.: Eigenvalues of the normal matrix $\mathcal{N}$ as defined in Lemma 4.3. All eigenvalues reside on a circle with radius $\frac{1}{2}$ around $(\frac{1}{2}, 0)$, hence their absolute value is less than one.

with the symmetric positive definite block diagonal matrix

$$\mathcal{Q} = \begin{pmatrix} \hat{A} - A & 0 \\ 0 & \hat{S} - C - B\hat{A}^{-1}B^T \end{pmatrix} \tag{4.69}$$

and

$$\overline{\mathcal{M}} = \mathcal{O}^T \mathcal{N} \mathcal{O},$$

where $\mathcal{N}$ *is a normal matrix and* $\mathcal{O}$ *satisfies the conditions* $\|\mathcal{O}\|_{l_2} \leq 1$.
Moreover, for the spectrum $\sigma(\mathcal{N})$ *we have:*

$$\sigma(\mathcal{N}) \subset \left\{ z \in \mathbb{C} : \left| z - \frac{1}{2} \right| = \frac{1}{2} \right\}.$$

In Figure 4.5 we give the location of the eigenvalues of $\mathcal{N}$. Together with $\|\mathcal{O}\|_{\ell_2} \leq 1$ we have $\|\overline{\mathcal{M}}\|_{\ell_2} < 1$. Now, from (4.68) we see that the matrix $\overline{\mathcal{M}}$ only differs from the smoother's iteration matrix $\mathcal{M}$ by a basis transform which is given by $\mathcal{Q}^{\frac{1}{2}}$, hence we can show convergence in terms of the norm induced by $\mathcal{Q}$. This is done more generally in the following theorem, where we also consider the damped iteration

$$\mathcal{M}_\omega = (1 - \omega) I + \omega \mathcal{M}.$$

Theorem 4.6 *([SZ03], Theorem 2) Let* $\hat{A}$ *be a symmetric and positive* $N \times N$ *matrix, and* $\hat{S}$ *a symmetric positive* $M \times M$ *matrix, satisfying* (4.66).
Then we have

$$\|(1 - \omega) I + \omega \mathcal{M}\|_{\mathcal{Q}} \leq 1$$

for all relaxation factors $\omega \in [0, 2]$, *and*

$$\|(1 - \omega) I + \omega \mathcal{M}\|_{\mathcal{Q}} < 1$$

for all relaxation factors $\omega \in (0,2)$. Here $\|\cdot\|_{\mathcal{Q}}$ denotes the matrix norm associated to the scalar product

$$((u,p),(v,q)) = \left(\left(\hat{A} - A\right) u, v\right) + \left(\left(\hat{S} - C - B\hat{A}^{-1}B^T\right) p, q\right). \tag{4.70}$$

The statement from Theorem 4.6 will serve as smoothing property in our convergence analysis, which is outlined in Section 4.8.
One can also provide a smoothing property which is very similar to the smoothing property for geometric multigrid methods as defined by Definition 2.1, especially (2.10). This is done in the following theorem.

Theorem 4.7 *([SZ03], Theorem 3) Let $\hat{A}$ by a symmetric and positive definite $N \times N$ matrix, and $\hat{S}$ a symmetric positive definite $M \times M$ matrix, satisfying*

$$\hat{A} \geq A \text{ and } \hat{S} \geq C + B\hat{A}^{-1}B^T.$$

Then

$$\|\mathcal{K}\mathcal{M}^\nu\|_{\ell_2} \leq \eta_0(\nu)\|\hat{\mathcal{K}} - \mathcal{K}\|_{\ell_2}$$

where

$$\eta_0(\nu) = \frac{1}{2^{\nu-1}}\binom{\nu-1}{[\nu]/2} \leq \begin{cases} \sqrt{\frac{2}{\pi(\nu-1)}} & \text{for even } \nu, \\ \sqrt{\frac{2}{\pi\nu}} & \text{for odd } \nu, \end{cases}$$

where $\binom{n}{k}$ denotes the binomial coefficient and $[x]$ denotes the largest integer smaller than or equal to $x \in \mathbb{R}$.

Before we proceed to the construction of the interpolation and the coarse grid operators as well as stability and convergence theory, we first will present a special implementation of an inexact Uzawa method that fits within the convergence theory of this section.

4.6. The smoother II: An algebraic Vanka-type smoother

In this section, we describe a box relaxation scheme introduced in [SZ03]. This smoother can be constructed without knowledge of the geometry or the discretization of the underlying PDE. We will see that the additive version of this smoother can be interpreted as an inexact Uzawa relaxation scheme, which allows us to apply the convergence theory of the previous section. The multiplicative variant, on the other hand, belongs to the class of Vanka smoothers [Van86].
First, we define an overlapping decomposition of the discrete velocity and pressure spaces $\mathcal{V}$ and $\mathcal{W}$,

$$\begin{pmatrix} \mathcal{V} \\ \mathcal{W} \end{pmatrix} = \bigcup_{j=1}^{M} \begin{pmatrix} \mathcal{V}^j \\ \mathcal{W}^j \end{pmatrix},$$

where $\mathcal{V}^j$ contains N_j velocity variables and $\mathcal{W}^j$ consists of M_j pressure variables. We need prolongation and restriction operators for each subspace (not to be confused with prolongation and restriction in multigrid!). We denote these interpolation operators by

$$V_j : \mathcal{V}^j \to \mathcal{V},\ V_j \in \mathbb{R}^{N \times N_j}$$
$$W_j : \mathcal{W}^j \to \mathcal{W},\ W_j \in \mathbb{R}^{M \times M_j}.$$

For restriction, we just take their transpose, i.e. V_j^T and W_j^T. V_j and W_j need to satisfy

$$\sum_{j=1}^{M} V_j (V_j)^T = I_{N\times N} \text{ and } \sum_{j=1}^{M} W_j (W_j)^T \in \mathbb{R}^{M\times M} \text{ non-singular} \tag{4.71}$$

which will be important to prove the convergence of the overall smoother. For each subspace $\mathcal{V}^j \times \mathcal{W}^j$, we solve local saddle point problems of the form

$$\begin{pmatrix} \hat{A}_j & B_j^T \\ B_j & B_j \hat{A}_j^{-1} B_j^T - \hat{S}_j \end{pmatrix} \begin{pmatrix} u_{(j)} \\ p_{(j)} \end{pmatrix} = \begin{pmatrix} V_j^T (f - Au^{it} - B^T p^{it}) \\ W_j^T (g - Bu^{it} + Cp^{it}). \end{pmatrix} \tag{4.72}$$

where $u_{(j)} \in \mathbb{R}^{N_j}$, $p_{(j)} \in \mathbb{R}^{M_j}$, $\hat{A}_j \in \mathbb{R}^{N_j \times N_j}$, $B_j \in \mathbb{R}^{M_j \times N_j}$, and

$$\hat{S}_j = \beta^{-1}(C_j + B_j \hat{A}_j^{-1} B_j^T),$$

where $C_j \in \mathbb{R}^{M_j \times M_j}$ and $\beta > 0$ is a scaling parameter. Then, we update the global solution according to

$$\begin{pmatrix} u^{it+1} \\ p^{it+1} \end{pmatrix} = \begin{pmatrix} u^{it} \\ p^{it} \end{pmatrix} + \sum_{j=1}^{M} \begin{pmatrix} V_j & 0 \\ 0 & W_j \end{pmatrix} \begin{pmatrix} u_{(j)} \\ p_{(j)} \end{pmatrix}. \tag{4.73}$$

In the following theorem, we give conditions on $\hat{A}_j$ and B_j under which the Schwarz smoother defined by (4.72) and (4.73) turns into an inexact global Uzawa smoother.

Theorem 4.8 *([SZ03], Theorem 1) Assume that* (4.71) *is satisfied, the matrices A_j, S_j are symmetric and positive definite, and that there is a symmetric positive definite $N \times N$ matrix $\hat{A}$ such that*

$$V_j^T \hat{A} = \hat{A}_j V_j^T \tag{4.74}$$

for all $j = 1, \ldots, M$. Furthermore, assume that the matrices B_j obey the condition

$$W_j^T B = B_j V_j^T \tag{4.75}$$

for all $j = 1, \ldots, M$.
Then we have

$$u^{it+1} = u^{it} + v^{it}, \quad p^{it+1} = p^{it} + w^{it}$$

where v^{it} and w^{it} satisfy the equation

$$\hat{\mathcal{K}} \begin{pmatrix} v^{it} \\ w^{it} \end{pmatrix} = \begin{pmatrix} f \\ g \end{pmatrix} - \mathcal{K} \begin{pmatrix} u^{it} \\ p^{it} \end{pmatrix}$$

with

$$\hat{\mathcal{K}} = \begin{pmatrix} \hat{A} & B^T \\ B & B\hat{A}^{-1}B^T - \hat{S} \end{pmatrix} \text{ and } \hat{S} = \left(\sum_{j=1}^{M} W_j \hat{S}_j^{-1} W_j^T \right)^{-1} \tag{4.76}$$

This theorem allows us to interpret the additive Vanka-type smoother (4.72)–(4.73) as an inexact symmetric Uzawa smoother for the *global* problem (4.1). Its iteration matrix $\mathcal{M}$ is given by

$$\mathcal{M} = I - \hat{\mathcal{K}}^{-1}\mathcal{K}$$

and convergence can be shown by Theorem 4.6.

We give the multiplicative iteration in Algorithm 4.2. In most cases, it is useful to carry

algorithm 4.2 VankaSmootherMult($\mathcal{K}, \{\hat{A}_j\}_{j=1}^M, \{B_j\}_{j=1}^M, \{\hat{S}_j\}_{j=1}^M, \{V_j\}_{j=1}^M, \{W_j\}_{j=1}^M, f, g, u^{it}, p^{it}$)

```
begin
  $u^{it,0} \leftarrow u^{it}$;
  $p^{it,0} \leftarrow p^{it}$;
  for $j = 1, \dots, M$ do
    $f_{(j)} = V_j^T \left(f - Au^{it,j-1} - Bp^{it,j-1}\right)$;
    $g_{(j)} = W_j^T \left(g - Bu^{it,j-1} + Cp^{it,j-1}\right)$;
    $\begin{pmatrix} u_{(j)} \\ p_{(j)} \end{pmatrix} \leftarrow \begin{pmatrix} \hat{A}_j & B_j^T \\ B_j & B_j\hat{A}_j^{-1}B_k^T - \hat{S}_j \end{pmatrix}^{-1} \begin{pmatrix} f_{(j)} \\ g_{(j)} \end{pmatrix}$;
    $u^{it,j} \leftarrow u^{it,j-1} + V_j u_{(j)}$;
    $p^{it,j} \leftarrow p^{it,j-1} + W_j p_{(j)}$;
  od;
  $u^{it+1} \leftarrow u^{it,M}$;
  $p^{it+1} \leftarrow p^{it,M}$;
end
```

out a symmetric iteration, i.e. after a single sweep of Algorithm 4.2 we carry out another iteration, but traverse the subspaces from $j = M$ down to $j = 1$. Unfortunately, a convergence theory is not known [personal communication with Walter Zulehner, 2012].

We now describe how an overlapping decomposition of the discrete domain Ω (and hence the discrete subspaces $\mathcal{V}^j$ and $\mathcal{W}^j$) can be constructed purely algebraically. We employ the non-zero structure of the B matrix part: For each row b_j of B, we identify the non-zero entries b_{ji} and define the subsets

$$\Omega^{(j)} = \begin{pmatrix} \Omega^j_{\mathcal{V}} \\ \Omega^j_{\mathcal{W}} \end{pmatrix}, \qquad \Omega^j_{\mathcal{V}} = \{i : \; b_{ji} \neq 0\}, \quad \Omega^j_{\mathcal{W}} = \{j\}. \tag{4.77}$$

For any velocity index i, there typically exists more than one j with $b_{ji} \neq 0$, hence these subsets overlap. The number of velocity components per subdomain N_j equals the number of non-zero entries in b_j. For example, in a staggered grid finite difference discretization of the Stokes or Navier-Stokes equations, each subdomain corresponds to a discretization cell, which consists of a pressure unknown at the cell center and the

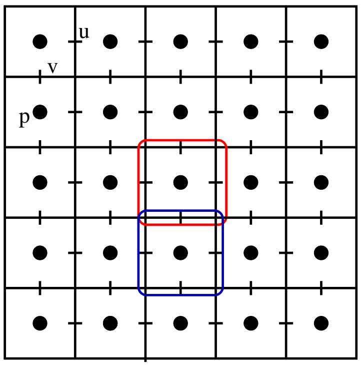


Figure 4.6.: Discretization mesh for the Stokes equation in two spatial dimensions. The pressure p is discretized at the centers of the cells, the velocities u and v are discretized at the midpoints of of the cell borders. The red and the blue box each represent a subdomain for the Vanka-type smoother.

velocity components at the cell borders, see Figure 4.6.
In this case, the interpolation operator for the pressure is given by

$$W_j : \mathbb{R} \rightarrow \mathbb{R}^M \tag{4.78}$$
$$W_j = I^{\mathcal{W}}_{\mathcal{W}^j}. \tag{4.79}$$

We define two interpolation operators for the velocity component,

$$V_j, \hat{V}_j : \mathbb{R}^{N_j} \rightarrow \mathbb{R}^N \tag{4.80}$$
$$V_j = diag\,(v_i)_{i=1,\dots,N} \cdot I^{\mathcal{V}}_{\mathcal{V}^j}. \tag{4.81}$$
$$\hat{V}_j = diag\left(\frac{1}{v_i}\right)_{i=1,\dots,N} \cdot I^{\mathcal{V}}_{\mathcal{V}^j}. \tag{4.82}$$

Here, $I^V_U : U \hookrightarrow V$ denotes the trivial injection. Each weight v_i depends on the number number of non-zero entries in the respective row of B^T, or equivalently, the number of subdomains $\Omega^j_{\mathcal{V}}$ that contain i,

$$v_i = \frac{1}{\sqrt{|\{j : b_{ji} \neq 0\}|}} = \frac{1}{\sqrt{\#\Omega^j_{\mathcal{V}} : i \in \Omega^j_{\mathcal{V}}}} \tag{4.83}$$

It is easy that V_j and W_j satisfy (4.71). Regarding $\hat{V}_j$, note that we have

$$\left(\hat{V}_j V_j^T\right)_{ik} = \begin{cases} 1 \text{ for } i = k \in \Omega^{(j)} \\ 0 \text{ elsewhere.} \end{cases}$$

To define the local matrices $\hat{A}_j \in \mathbb{R}^{N_j \times N_j}$, $\hat{S}_j \in \mathbb{R}$, $B_j \in \mathbb{R}^{1 \times N_j}$, we first need a global preconditioner $\hat{A}$ for A, which is assumed to be a diagonal matrix (e.g. the scaled diagonal of A such that $\hat{A} > A$). Then we set

$$\hat{A}_j = V_j^T \hat{A} \hat{V}_j, \tag{4.84}$$
$$B_j = W_j^T B \hat{V}_j, \tag{4.85}$$
$$\hat{S}_j = \beta^{-1}(c_{jj} + B_j \hat{A}_j^{-1} B_j^T). \tag{4.86}$$

where we choose $\beta > 0$ such that $\hat{S} > C + B\hat{A}^{-1}B^T$. Straightforward calculations then show that we have satisfied (4.71), (4.74), (4.75) and hence Theorem 4.8 allows us to apply the inexact Uzawa smoother's convergence theory as given by Theorems 4.6 and 4.7.

Remark 4.2 In the case of point-based AMG (Section 2.12.3), the construction is slightly different. In this case, let us assume for sake of simplicity that we have d velocity components and one pressure component that are discretized at every point $j = 1, \dots, M$. This yields a block structure of $\mathcal{K}$,

$$\mathcal{K} = \begin{pmatrix} \mathcal{K}_{(1,1)} & \mathcal{K}_{(1,2)} & \dots & \mathcal{K}_{(1,M)} \\ \mathcal{K}_{(2,1)} & \mathcal{K}_{(2,2)} & \dots & \mathcal{K}_{(2,M)} \\ \vdots & \vdots & \ddots & \vdots \\ \mathcal{K}_{(M,1)} & \mathcal{K}_{(M,2)} & \dots & \mathcal{K}_{(M,M)} \end{pmatrix},$$

where each $\mathcal{K}_{(i,j)}$ has the form

$$\mathcal{K}_{(i,j)} = \begin{pmatrix} A_{(i,j)} & B_{(i,j)}^T \\ B_{(i,j)} & C_{(i,j)} \end{pmatrix}$$

with block matrices $A_{(i,j)} \in \mathbb{R}^{d \times d}$, $B_{(i,j)} \in \mathbb{R}^{1 \times d}$, and $C_{(i,j)} \in \mathbb{R}^{1 \times 1}$.
To define the subdomains $\Omega^{(j)}$, we replace, in (4.77), the definition of $\Omega_{\mathcal{V}}^j$ by

$$\Omega_{\mathcal{V}}^j = \{i : \; B_{(j,i)} \neq 0\}$$

i.e. each subset $\Omega_{\mathcal{V}}^j$ contains all points i for which at least one velocity variable has a connection to the discretized pressure value at point j. Analogously, we define the weights v_i by

$$v_i = \frac{1}{\sqrt{|\{j : B_{(j,i)} \neq 0\}|}} = \frac{1}{\sqrt{\#\Omega_{\mathcal{V}}^j : i \in \Omega_{\mathcal{V}}^j}}. \tag{4.87}$$

For $\hat{A}$, we can either take the block diagonal matrix

$$\begin{pmatrix} A_{(1,1)} & & 0 \\ & \ddots & \\ 0 & & A_{(M,M)} \end{pmatrix},$$

or just its diagonal. Then, we can define W_j, V_j, $\hat{V}_j$, $\hat{A}_j$, B_j and $\hat{S}_j$ as in (4.78)–(4.86) (Note that each $c_{jj} = C_{(j,j)}$ is just a single scalar). We will comment on which variant to choose in Section 4.8

algorithm 4.3 VankaSmootherSetup$(A, B, C, \{\hat{A}_j\}_{j=1}^M, \{B_j\}_{j=1}^M, \{\hat{S}_j\}_{j=1}^M, \{V_j\}_{j=1}^M, \{W_j\}_{j=1}^M)$

```
begin
  $\overline{\Omega_{\mathcal{V}_q}} \leftarrow \Omega_{\mathcal{V}_q} \cup \{i \notin \Omega_{\mathcal{V}_q} : \exists j \in \Omega_{\mathcal{W}_q} : b_{ji} \neq 0\}$        local and ghost velocity points
  $D \leftarrow diag(A)$;
  $\alpha \leftarrow \lambda_{\max}\left(D^{-\frac{1}{2}} A D^{-\frac{1}{2}}\right)$;        (parallel) power iteration
  $\hat{A} \leftarrow \alpha D$;
  for $i = 1, \ldots, N$ do
      compute weights $v_i$;  depending on additive, multiplicative, parallel multiplicative
  od
  if (parallel computation)
    obtain $v_i$ and $\hat{a}_{ii}$ for all $i \in \overline{\Omega_{\mathcal{V}_q}} \setminus \Omega_{\mathcal{V}_q}$;
    $\hat{A}_{(q)} \leftarrow diag(\hat{a}_{ii})_{i \in \overline{\Omega_{\mathcal{V}_q}}}$;
  fi
  for $j \in \Omega_{\mathcal{W}_q}$ do
      $V_j \leftarrow diag(v_i) \cdot I_{\mathcal{V}^j}^{\overline{\mathcal{V}_q}}$;
      $W_j \leftarrow I_{\mathcal{W}^j}^{\mathcal{W}_q}$;
      $\hat{A}_j \leftarrow V_j^T \hat{A}_{(q)} \hat{V}_j$;
      $B_j \leftarrow W_j^T B \hat{V}_j$;
      $\hat{S}_j \leftarrow (c_{jj} + B_j \hat{A}_j^{-1} B_j^T)$;
  od;
  $\hat{S} \leftarrow diag(\hat{S}_j)_{j=1}^M$;        each processor just keeps its own part
  $\beta \leftarrow \lambda_{\max}\left(\hat{S}^{-\frac{1}{2}}\left(B\hat{A}^{-1}B^T + C\right)\hat{S}^{-\frac{1}{2}}\right)$;        (parallel) power iteration
  for $j \in \Omega_{\mathcal{W}_q}$ do
      $\hat{S}_j \leftarrow \beta \hat{S}_j$;
  od;
end
```

In the multiplicative case, we are not restricted to the special scalings (4.83) or (4.87). Whether it is possible to use unscaled injection instead, i.e. $v_i = 1$ for all $i = 1, \dots, N$, must be determined numerically.
We conclude this section with the parallel implementation of the smoother. To this end, as in (3.1), let us assume that we have a non-overlapping decomposition of the velocity and the pressure spaces among the np processors,

$$\begin{aligned} \mathcal{V} &= \mathcal{V}_1 \dot{\cup} \dots \dot{\cup} \mathcal{V}_{np} \\ \mathcal{W} &= \mathcal{W}_1 \dot{\cup} \dots \dot{\cup} \mathcal{W}_{np} \end{aligned}$$

and a corresponding decomposition of the index set Ω, cf. (3.1)

$$\Omega = \Omega_1 \cup \dots \cup \Omega_{np} = \begin{pmatrix} \Omega_{\mathcal{V}_1} \\ \Omega_{\mathcal{W}_1} \end{pmatrix} \cup \dots \cup \begin{pmatrix} \Omega_{\mathcal{V}_{np}} \\ \Omega_{\mathcal{W}_{np}} \end{pmatrix}.$$

We store each submatrix

$$\begin{pmatrix} \hat{A}_j & B_j^T \\ B_j & B_j \hat{A}_j^{-1} B_j^T - \hat{S}_j \end{pmatrix} \tag{4.88}$$

on the processor q which contains the pressure variable $j \in \Omega_{\mathcal{W}_q}$. Note that the corresponding velocity subset $\mathcal{V}^j$ is not necessarily contained within the local variables. We define the set of ghost velocity points for processor q,

$$\begin{aligned} \Omega_{\mathcal{V}_q}^{ghost} &:= \{ i \notin \Omega_{\mathcal{V}_q} : \text{ there exists a } j \in \Omega_{\mathcal{W}_q} \text{ such that } b_{ji} \neq 0 \} \\ \overline{\Omega_{\mathcal{V}_q}} &:= \Omega_{\mathcal{V}_q} \cup \Omega_{\mathcal{V}_q}^{ghost} \end{aligned}$$

Hence, we need to obtain parts of $\hat{A}$ as well as the weights v_i for all velocity variables $i \in \Omega_{\mathcal{V}_q}^{ghost}$ Then, (4.84)–(4.86) can be computed locally and the velocity prolongation operators V_j maps into $\overline{\mathcal{V}_q} := \{(u_i)_{i \in \overline{\Omega_{\mathcal{V}_q}}}\}$ for all $j \in \mathcal{W}_q$, $V_j : \mathcal{V}^j \to \overline{\mathcal{V}_q}$.
In the additive case, the parallel application of the smoother is straightforward. The communication takes place within the computation of the residual,

$$\begin{pmatrix} f \\ g \end{pmatrix} - \begin{pmatrix} A & B^T \\ B & -C \end{pmatrix} \begin{pmatrix} u^{it} \\ p^{it} \end{pmatrix}.$$

Afterwards, each processor q needs to obtain the velocity residuals for all $i \in \Omega_{\mathcal{V}_q}^{ghost}$. Then, we can locally apply the restriction scalings and solve the systems (4.72) for all $j \in \Omega_{\mathcal{W}_q}$. Afterwards, we apply the prolongation scalings and transfer the velocity updates back to the owning processors.

As with the Gauss-Seidel iteration, the multiplicative smoother cannot be efficiently parallelized. Inspired by block Jacobi smoothing (3.2), we propose a hybrid additive/multiplicative smoother that only updates velocities between processors after a

algorithm 4.4
VankaSmootherPar$(\mathcal{K}, \{\hat{A}_j\}_{j\in\Omega_{\mathcal{W}_q}}, \{B_j\}_{j\in\Omega_{\mathcal{W}_q}}, \{\hat{S}_j\}_{j\in\Omega_{\mathcal{W}_q}}, \{V_j\}_{j\in\Omega_{\mathcal{W}_q}}, \{W_j\}_{j\in\Omega_{\mathcal{W}_q}}, f, g, u^{it}, p^{it})$

```
begin
  $\begin{pmatrix} f^{it} \\ g^{it} \end{pmatrix} \leftarrow \begin{pmatrix} f \\ g \end{pmatrix} - \begin{pmatrix} A & B^T \\ B & -C \end{pmatrix} \begin{pmatrix} u^{it} \\ p^{it} \end{pmatrix}$;          parallel residual computation
  communication: receive ghost values of $\overline{f_{(q)}}$ and $\overline{u^{it}_{(q)}}$;
  $\overline{u^{it,0}_{(q)}} \leftarrow \overline{u^{it}_{(q)}}$;
  $p^{it,0}_{(q)} \leftarrow p^{it}_{(q)}$;
  $\overline{f^{it,0}_{(q)}} \leftarrow \overline{f^{it}_{(q)}}$;
  $g^{it,0}_{(q)} \leftarrow g^{it}_{(q)}$;
  for $j \in \Omega_{\mathcal{W}_q}$ do
    $f_{(j)} = V_j^T \overline{f^{it,j-1}_{(q)}}$;
    $g_{(j)} = W_j^T g^{it,j-1}_{(q)}$;
    $\begin{pmatrix} u_{(j)} \\ p_{(j)} \end{pmatrix} \leftarrow \begin{pmatrix} \hat{A}_j & B_j^T \\ B_j & B_j \hat{A}_j^{-1} B_k^T - \hat{S}_j \end{pmatrix}^{-1} \begin{pmatrix} f_{(j)} \\ g_{(j)} \end{pmatrix}$;
    $\overline{u^{it,j}_{(q)}} \leftarrow \overline{u^{it,j-1}_{(q)}} + V_j u_{(j)}$;          local solution update
    $p^{it,j}_{(q)} \leftarrow p^{it,j-1}_{(q)} + W_j p_{(j)}$;
    $\overline{f^{it,j}_{(q)}} \leftarrow \overline{f^{it,j-1}_{(q)}} - \overline{A_{(q)}} V_j u_{(j)} - \overline{B^T_{(q)}} W_j p_{(j)}$;          residual update
    $g^{it,j}_{(q)} \leftarrow g^{it,j-1}_{(q)} - B_{(q)} V_j u_{(j)} + C_{(q)} W_j p_{(j)}$;
  od;
  communication: transfer ghost values of $\overline{u^{it,|\Omega_{\mathcal{W}_q}|}}$ to owning processors;
  $u^{it+1}_{(q)} \leftarrow u^{it,|\Omega_{\mathcal{W}_q}|}_{(q)} +$ received values;
  $p^{it+1}_{(q)} \leftarrow p^{it,|\Omega_{\mathcal{W}_q}|}_{(q)}$;
end
```

sweep is completed. Let

$$\begin{aligned}
A_{(q)} &:= (a_{ik})_{i\in\Omega_{\mathcal{V}_q},\ k=1,\dots,N},\\
B^T_{(q)} &:= (b^T_{ij})_{i\in\Omega_{\mathcal{V}_q},\ j=1,\dots,M},\\
B_{(q)} &:= (b_{jk})_{j\in\Omega_{\mathcal{W}_q},\ k=1,\dots,N},\\
C_{(q)} &:= (c_{jr})_{j\in\Omega_{\mathcal{W}_q},\ r=1,\dots,M},
\end{aligned}$$

denote the rows of A, B^T, B, and C that reside on processor q, respectively (see Chapter 3 for a description of the parallel matrix format we use). Note that $B^T_{(q)}$ is not the transpose of $B_{(q)}$. We further define

$$\begin{aligned}
\text{the local and ghost part of } A, \overline{A_{(q)}} &:= (a_{ik})_{i\in\overline{\Omega_{\mathcal{V}_q}},\ k=1,\dots,N},\\
\text{the local and ghost part of } B^T, \overline{B^T_{(q)}} &:= (b^T_{ij})_{i\in\overline{\Omega_{\mathcal{V}_q}},\ j=1,\dots,M},\\
\text{the local part of } u^{it},\ u^{it}_{(q)} &:= (u^{it}_i)_{i\in\Omega_{\mathcal{V}_q}},\\
\text{the local and ghost part of } u^{it}, \overline{u^{it}_{(q)}} &:= (u^{it}_i)_{i\in\overline{\Omega_{\mathcal{V}_q}}},\\
\text{the local part of } p^{it}, p^{it}_{(q)} &:= (p^{it}_j)_{j\in\Omega_{\mathcal{W}_q}},\\
\text{the local part of } f^{it},\ f^{it}_{(q)} &:= (f^{it}_i)_{i\in\Omega_{\mathcal{V}_q}},\\
\text{the local and ghost part of } f^{it}, \overline{f^{it}_{(q)}} &:= (f^{it}_i)_{i\in\overline{\Omega_{\mathcal{V}_q}}},\\
\text{the local part of } g^{it}, g^{it}_{(q)} &:= (g^{it}_j)_{j\in\Omega_{\mathcal{W}_q}}.
\end{aligned}$$

We give a single parallel smoothing sweep in Algorithm 4.4. Note that inside the iteration, we update the residuals not only for the local velocity indices $i \in \Omega_{\mathcal{V}_q}$, but also for the ghost points $i \in \Omega^{ghost}_{\mathcal{V}_q}$. After the iteration is complete, we transfer all ghost velocity values back to the processors that own them. On the owning processor, we denote the respective velocity indices with

$$\partial\Omega_{\mathcal{V}_q} := \{i \in \Omega_{\mathcal{V}_q} : \exists j \notin \Omega_{\mathcal{W}_q} : b_{ji} \neq 0\}.$$

For all $i \in \partial\Omega_{\mathcal{V}_q}$, we receive one or more velocity updates from another processor and add it to u_i.
The prolongation weights v_i need to reflect the mixed additive/multiplicative updates for these points. We suggest two options:

1 As in the additive case, let $v_i = \frac{1}{\sqrt{|\{j : b_{ji}\neq 0\}|}}$ for all $i \in \partial\Omega_{\mathcal{V}_q}$.

2 Use different weights for the same index i depending on whether it is a local or ghost point. More precisely, assuming that $i \in \partial\Omega_{\mathcal{V}_q}$, we set

$$v_i = \frac{\sqrt{|\{j \in \Omega_{\mathcal{W}_q} : b_{ji} \neq 0\}|}}{\sqrt{|\{j \in \Omega_{\mathcal{W}} : b_{ji} \neq 0\}|}} \tag{4.89}$$

on processor q and

$$v_i = \frac{1}{\sqrt{|\{j \in \Omega_{\mathcal{W}} : b_{ji} \neq 0\}|}} \tag{4.90}$$

on all processors q' for which $i \in \Omega^{ghost}_{\mathcal{V}_{q'}}$.

For all interior points $i \in \Omega_{\mathcal{V}_q} \setminus \partial\Omega_{\mathcal{V}_q}$ we set $v_i = 1$. Which option leads to the best result must be determined experimentally. It also may occur that we must set $v_i = \frac{1}{\sqrt{|\{j : b_{ji} \neq 0\}|}}$ for all $i \in \Omega_{\mathcal{V}_q}$, as in the additive case.

4.7. Interpolation and Coarse Grid Correction: General Remarks

We now turn our attention to the coarse grid correction, the other main component of a multigrid solver. As we have already pointed out in Chapter 2, the coarse grid correction should be well-adapted to the smooth error components, i.e. the error components e that are not reduced efficiently by the smoother,

$$\mathcal{M}e \approx e.$$

For a well-fitted coarse grid correction, we need to construct three components:

1 A *coarse grid* Ω^{l+1} whose cardinality is less than the size of the fine grid Ω^l and is well-suited to represent the smooth error components from level l,

2 *restriction* and *prolongation* operators

$$\mathcal{R}_l \in \mathbb{R}^{|\Omega^l| \times |\Omega^{l+1}|}, \quad \mathcal{P}_l \in \mathbb{R}^{|\Omega^{l+1}| \times |\Omega^l|}$$

which transfer the error resp. the error correction between the fine discrete space $\mathcal{V}^l \times \mathcal{W}^l$ and the coarse space $\mathcal{V}^{l+1} \times \mathcal{W}^{l+1}$,

3 a *coarse grid operator*

$$\mathcal{K}_{l+1} \in \mathbb{R}^{|\Omega^{l+1}| \times |\Omega^{l+1}|}$$

that describes the underlying problem on the coarse scale, i.e. in terms of the smooth error components.

With these components, we can formulate the two-grid correction operator

$$I - \mathcal{P}^l \left(\mathcal{K}^{l+1}\right)^{-1} \mathcal{R}^l \mathcal{K}^l,$$

and the multi-level correction

$$I - \mathcal{P}_l \left(\tilde{\mathcal{K}}_{l+1}\right)^{-1} \mathcal{R}_l \mathcal{K}_l,$$

where $\left(\tilde{\mathcal{K}}_{l+1}\right)^{-1}$ denotes the approximation to $\left(\mathcal{K}_{l+1}\right)^{-1}$ by recursive application of the multi-grid cycle.
We take again a look at the smoother $\mathcal{M}_l$,

$$\mathcal{M}_l = I - \hat{\mathcal{K}_l}^{-1}\mathcal{K}_l = \begin{pmatrix} I & -\hat{A}_l^{-1}B_l^T \\ 0 & I \end{pmatrix} \begin{pmatrix} I & 0 \\ \hat{S}_l^{-1}B_l & I \end{pmatrix} \begin{pmatrix} I - \hat{A}_l^{-1}A_l & 0 \\ 0 & I - \hat{S}_l^{-1}\left(B_l\hat{A}_l^{-1}B_l^T + C_l\right) \end{pmatrix}. \tag{4.91}$$

The rightmost factor essentially performs a "Jacobi-like" sweep over the matrix

$$\begin{pmatrix} A_l & 0 \\ 0 & B_l\hat{A}_l^{-1}B_l^T + C_l \end{pmatrix}. \tag{4.92}$$

This suggests to build a first guess for interpolation and coarse grid correction based on the symmetric positive definite block matrix (4.92), i.e.

1 build a (block) strength matrix $\mathcal{S}_l$ based on the matrix blocks A_l and $B_l\hat{A}_lB_l^T + C_l$,

$$\mathcal{S}_l = \begin{pmatrix} S_{\mathcal{V}^l} & 0 \\ 0 & S_{\mathcal{W}^l} \end{pmatrix},$$

2 construct a block interpolation operator

$$\mathcal{P}_l = \begin{pmatrix} P_{\mathcal{V}^l} & 0 \\ 0 & P_{\mathcal{W}^l} \end{pmatrix}$$

where $P_{\mathcal{V}^l}$ and $P_{\mathcal{W}^l}$ are obtained by applying one of the well-known interpolation schemes for the symmetric positive matrices A_L and $B_l\hat{A}_l^{-1}B_l^T + C_l$ from Section 2.8,

3 compute the coarse grid operator

$$\mathcal{K}_{l+1} = \mathcal{R}_l\mathcal{K}_l\mathcal{P}_l$$

where $\mathcal{R}_l$ denotes the restriction operator.

While this approach seems straightforward, there are some issues that we need to deal with. First, unlike the symmetric positive definite case, it is not clear whether $\mathcal{K}_{l+1}$ is invertible at all even if we set $\mathcal{R}_l = \mathcal{P}_l^T$.

Example 4.2 [Wab03] Let the Stokes equations on $[0,1]^2$ with Dirichlet boundary conditions be discretized using $P_1isoP_2 - P_1$ elements on an uniform mesh, (4.38), see Figure 4.7. We see that after one coarsening step on the velocity mesh (for both velocity components) we obtain an instable situation as in Example 4.1 if all pressure nodes are taken into the coarse mesh.

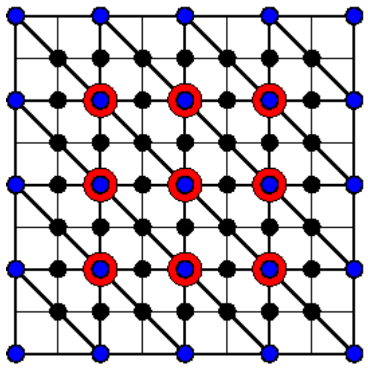


Figure 4.7.: Finite element mesh for a $P_1 iso P_2 - P_1$ discretization of the Stokes' equations on the domain $\Omega = [0,1]^2$ (we assume Dirichlet boundary conditions on $\partial\Omega$). The black and the red dots denote the velocity nodes on the finest and the first coarse mesh, the blue dots denote pressure nodes on both levels.

This example might seem exceptional, but it illustrates that the stability of the coarse system is not automatically ensured. The second obstacle is that we have no variational principle (see Theorem 2.15) here, as $\mathcal{K}$ does not define an inner product. In consequence, we need a different approach to show two-grid convergence.

Remark 4.3 In Remark 4.1 we stated that under certain assumptions it is possible to re-interpret the smoother $\mathcal{M}$ as an iteration scheme over a symmetric positive definite matrix $\mathcal{L}$,

$$\mathcal{M} = I - \mathcal{Q}^{-1}\mathcal{L}$$

where $\mathcal{Q}$ is a block diagonal symmetric positive definite matrix. Using this formulation, it is possible to apply the classical AMG convergence theory in terms of inner products and norms defined by

$$(x,y)_0 := x\mathcal{Q}y, \quad (x,y)_1 := x\mathcal{L}y, \quad (x,y)_2 := x\mathcal{L}\mathcal{Q}^{-1}\mathcal{L}y,$$

cf. (2.22)–(2.24).
On the downside, to apply this theory we need either

$$\begin{array}{lll} \hat{A} < A & \text{and} & \hat{S} > B\hat{A}^{-1}B^T + C, \text{ or,} \\ \hat{A} > A & \text{and} & \hat{S} < B\hat{A}^{-1}B^T + C, \end{array}$$

which can become compute-intensive as we need to apply an eigensolver to determine the correct scaling for $\hat{A}$ of $\hat{S}$. (In contrast, to obtain both $\hat{A} > A$ and $\hat{S} > B\hat{A}^{-1}B^T + C$ a few cheap power iteration cycles or the application of Gershgorin's circle theorem are sufficient.) Furthermore, the variational principle shown in the norm defined by $\mathcal{L}$ cannot be applied directly to the coarse grid operator obtained from $\mathcal{P}^T\mathcal{K}\mathcal{P}$. We refer to Appendix A for a detailed discussion.

In the following sections, we will show how the invertibility of the coarse grid operator can be ensured. To this end, we first assume that an inf-sup condition on the finest level

$l = 1$ is available:
Let there exist constants $c_1, d_1 > 0$ such that, for all $p \in \mathcal{W}^1$ such that

$$\sup_{0 \neq u \in \mathcal{V}_1} \frac{u B_1^T p}{\|u\|_{A_1}} \geq c_1 \|p\|_{\mathcal{W}^1} - d_1 \left(p^T C_1 p\right)^{\frac{1}{2}} \text{ for all } p \in \mathcal{W}^1. \tag{4.93}$$

From this inf-sup condition we will derive inf-sup conditions on all levels $l > 1$ by induction,

$$\sup_{0 \neq u \in \mathcal{V}^l} \frac{u B_l^T p}{\|u\|_{A_l}} \geq c_l \|p\|_{\mathcal{W}^l} - d_l \left(p^T C_l p\right)^{\frac{1}{2}} \text{ for all } p \in \mathcal{W}^l. \tag{4.94}$$

for all $p \in \mathcal{W}^l$ and $c_l, d_l > 0$ independent of p.
Here, as in the remainder of this chapter, we define for a symmetric positive definite matrix A an inner product and a norm,

$$\langle x, x \rangle_A := x^T A x, \quad \|x\|_A := \sqrt{\langle x, x \rangle_A}.$$

The inf-sup-condition (4.93) on the finest level usually only can be obtained with knowledge of the underlying problem. First, we need a norm $\|\cdot\|_{\mathcal{W}^1}$ for the pressure space on the finest level $\mathcal{W}^1$. Usually, this norm is derived from a scalar product

$$(x, y)_{\mathrm{M}_1} := x^T \mathrm{M}_1 y$$

where M_1 is a "mass matrix" for the discrete pressure space such that $\|\cdot\|_{\mathcal{W}^1} := \|\cdot\|_{\mathrm{M}_1}$ can be interpreted as the discrete counterpart of a L^2-norm. In Section 4.2, we have introduced several stable discretization schemes as well as the corresponding inf-sup conditions, which can be written in the form (4.93). The coarse mass matrices M_l, $l > 1$, are obtained from M_1 by recursive application of the Galerkin product

$$\mathrm{M}_{l+1} = P_{\mathcal{W}^l}^T \mathrm{M}_l P_{\mathcal{W}^l}.$$

From (4.94) we can, for each level l, prove the stability result for $\mathcal{K}_l$ on all levels, which is a generalization of Theorem 4.4 in [Wab03] (see also Theorem 4.5). For sake of completeness, we also give the proof.

Theorem 4.9 *Suppose that* (4.94) *holds. Then*

$$\sup_{0 \neq v \in \mathcal{V}^l,\ 0 \neq q \in \mathcal{W}^l} \frac{(u, p)^T \mathcal{K}_l (v, q)}{\|v\|_{A_l} + \|q\|_{\mathrm{M}_l}} \geq \zeta_l \left(\|u\|_{A_l} + \|p\|_{\mathrm{M}_l}\right) \textit{ for all } (u, p) \in \mathcal{V}^l \times \mathcal{W}^l, \tag{4.95}$$

for some $\zeta_l > 0$ *depending on* l.

Proof: ([Wab03],pp. 56–57) For $p \in \mathcal{W}^l$, choose an $u^* \in \mathcal{V}^l$ such that the supremum in (4.94) is attained. Note that this supremum is invariant under scaling of u^*, so we are free to scale u^* such that

$$\|u^*\|_{A_l} = \|p\|_{\mathrm{M}_l}.$$

Let $u \in \mathcal{V}^l$. We apply (4.94),

$$\begin{aligned}
\begin{pmatrix} u & p \end{pmatrix} \mathcal{K}_l \begin{pmatrix} u^* \\ 0 \end{pmatrix} &= u^T A_l u^* + p^T B_l u^* \\
&\geq -\|u\|_{A_l}\|u^*\|_{A_l} + c_l\|u^*\|_{A_l}\|p\|_{\mathrm{M}_l} - d_l \left(p^T C_l p\right)^{1/2} \|p\|_{\mathrm{M}_l} \\
&= -\|u\|_{A_l}\|p\|_{\mathrm{M}_l} + c_l\|p\|^2_{\mathrm{M}_l} - d_l \left(p^T C_l p\right)^{1/2} \|p\|_{\mathrm{M}_l} \\
&\geq -\frac{1}{2\epsilon}\|u\|^2_{A_l} - \frac{\epsilon}{2}\|p\|^2_{\mathrm{M}_l} + c_l\|p\|^2_{\mathrm{M}_l} - \frac{d_l}{2\epsilon} p^T C_l p - \frac{d_l \epsilon}{2}\|p\|^2_{\mathrm{M}_l}
\end{aligned}$$

where the last step can be seen from $xy \leq \frac{x^2/\epsilon + \epsilon y^2}{2}$. Now, we introduce the constants

$$\theta_1 = \frac{1}{2\epsilon}, \quad \theta_2 = c_l - \frac{\epsilon}{2}(1 + d_l), \quad \theta_3 = \frac{d_l}{2\epsilon}$$

which are strictly positive as long as $0 < \epsilon < \frac{2c_l}{1+d_l}$. We have

$$\begin{pmatrix} u & p \end{pmatrix} \mathcal{K}_l \begin{pmatrix} u^* \\ 0 \end{pmatrix} \geq -\theta_1\|u\|^2_{A_l} + \theta_2\|p\|^2_{\mathrm{M}_l} - \theta_3 p^T C_l p.$$

We now consider a (v^*, q^*) of the form $(v^*, q^*) = (u + \vartheta u^*, -p)$ with a parameter ϑ,

$$\begin{aligned}
\begin{pmatrix} u & p \end{pmatrix} \mathcal{K}_l \begin{pmatrix} v^* \\ q^* \end{pmatrix} &= \begin{pmatrix} u & p \end{pmatrix} \mathcal{K}_l \begin{pmatrix} u \\ -p \end{pmatrix} + \vartheta \begin{pmatrix} u & p \end{pmatrix} \mathcal{K}_l \begin{pmatrix} u^* \\ 0 \end{pmatrix} \\
&\geq \|u\|^2_{A_l} + p^T C_l p - \vartheta\theta_1\|u\|^2_{A_l} + \vartheta\theta_2\|p\|^2_{\mathrm{M}_l} - \vartheta\theta_3 p^T C_l p.
\end{aligned}$$

Now, choose $0 < \vartheta < \min\left(\frac{1}{\theta_1}, \frac{1}{\theta_3}\right)$,

$$\begin{pmatrix} u & p \end{pmatrix} \mathcal{K}_l \begin{pmatrix} v^* \\ q^* \end{pmatrix} \geq \theta_4 \left(\|u\|^2_{A_l} + \|p\|^2_{\mathrm{M}_l}\right) \tag{4.96}$$

where $\theta_4 = \min(1 - \vartheta\theta_1, \vartheta\theta_2)$.
For our special choice of (v^*, q^*) we have

$$\|v^*\|_{A_l} + \|q^*\|_{\mathrm{M}_l} = \|u + \vartheta u^*\|_{A_l} + \|p\|_{\mathrm{M}_l} \leq \|u\|_{A_l} + \vartheta\|u^*\|_{A_l} + \|p\|_{\mathrm{M}_l} \leq (1+\vartheta)\left(\|u\|_{A_l} + \|p\|_{\mathrm{M}_l}\right). \tag{4.97}$$

We combine (4.96) and (4.97),

$$\frac{(u,p)^T \mathcal{K}_l (v^*, q^*)}{\|v^*\|_{A_l} + \|q^*\|_{\mathrm{M}_l}} \geq \frac{\theta_4}{1+\vartheta} \frac{\|u\|_{A_l} + \|p\|_{\mathrm{M}_l}}{\|u\|_{A_l} + \|p\|_{\mathrm{M}_l}} \geq \frac{\theta_4}{2 + 2\vartheta}\left(\|u\|_{A_l} + \|p\|_{\mathrm{M}_l}\right).$$

Now, taking the supremum over (v, q) completes the proof,

$$\sup_{0 \neq v \in \mathcal{V}^l,\ 0 \neq q \in \mathcal{W}^l} \frac{(u,p)^T \mathcal{K}_l (v, q)}{\|v\|_{A_l} + \|q\|_{\mathrm{M}_l}} \geq \frac{(u,p)^T \mathcal{K}_l (v^*, q^*)}{\|v^*\|_{A_l} + \|q^*\|_{\mathrm{M}_l}} \geq \zeta\left(\|u\|_{A_l} + \|p\|_{\mathrm{M}_l}\right)$$

where $\zeta = \frac{\theta_4}{2+2\vartheta}$. □

Remark 4.4 If the matrix A_l is singular, we replace the term $\|\cdot\|_{A_l}$ in (4.93), (4.94), (4.95) and the proof thereafter by a norm $\|\cdot\|_{A_{D_l}}$, where the matrix A_{D_l} is non-singular and $A_{D_l} > A_l$. Such an operator can always be constructed as A_l is continuous.

In the following sections, we will give three different approaches to construct the interpolation operator as well as the coarse grid matrix. For one of them, we can also show two-grid convergence.

4.8. The coarse grid operator I: Auto-stabilizing coarsening and Two-level convergence

Our first approach to a stable coarse system is to formulate a "Petrov-Galerkin" type coarse grid operator, which however still leads to a symmetric matrix. For this formulation we can show two-grid convergence using the two-grid convergence theory for non-symmetric matrices as outlined in Section 2.10.
We start with the operator $\mathcal{L}_l = \mathcal{Q}_l\hat{\mathcal{K}}_l^{-1}\mathcal{K}_l$, where the symmetric positive definite matrix $\mathcal{Q}_l$ is defined as in (4.69),

$$\mathcal{Q}_l = \begin{pmatrix} I_{\mathcal{V}^l} & 0 \\ 0 & -I_{\mathcal{W}^l} \end{pmatrix} \left(\hat{\mathcal{K}}_l^{-1} - \mathcal{K}_l\right) = \begin{pmatrix} \hat{A}_l - A_l & 0 \\ 0 & \hat{S}_l - B_l\hat{A}_l^{-1}B_l^T - C_l \end{pmatrix}.$$

The matrix $\mathcal{L}_l$ can also be written in terms of $\mathcal{K}_l$ and the inexact Uzawa smoother $\mathcal{M}_l$ (4.67),

$$\mathcal{L}_l = \mathcal{Q}_l\hat{\mathcal{K}}_l^{-1}\mathcal{K}_l = \begin{pmatrix} I_{\mathcal{V}^l} & 0 \\ 0 & -I_{\mathcal{W}^l} \end{pmatrix} \left(\mathcal{K}_l - \mathcal{K}_l\hat{\mathcal{K}}_l^{-1}\mathcal{K}_l\right) = \begin{pmatrix} I_{\mathcal{V}^l} & 0 \\ 0 & -I_{\mathcal{W}^l} \end{pmatrix} \mathcal{K}_l\mathcal{M}_l.$$

Now, assume that we have a prototype restriction $\mathcal{R}_l$ available, which establishes no couplings between the velocity and the pressure part, i.e.,

$$\mathcal{R}_l = \begin{pmatrix} R_{\mathcal{V}^l} & 0 \\ 0 & R_{\mathcal{W}^l} \end{pmatrix}.$$

The error propagation operator $\mathcal{T}_l$ of the two-grid cycle for $\mathcal{L}_l$ with one post-smoothing step can now be reformulated,

$$\begin{aligned} \mathcal{T}_l =& \mathcal{M}_l(I - \mathcal{R}_l^T\mathcal{L}_{l+1}^{-1}\mathcal{R}_l\mathcal{L}_l) \\ =& \mathcal{M}_l - \mathcal{M}_l\mathcal{R}_l^T\left[\mathcal{R}_l\begin{pmatrix} I_{\mathcal{V}^l} & 0 \\ 0 & -I_{\mathcal{W}^l} \end{pmatrix}\mathcal{K}_l\mathcal{M}_l\mathcal{R}_l^T\right]^{-1}\mathcal{R}_l\begin{pmatrix} I_{\mathcal{V}^l} & 0 \\ 0 & -I_{\mathcal{W}^l} \end{pmatrix}\mathcal{K}_l\mathcal{M}_l \\ =& \left(I - \mathcal{P}_l\left[\mathcal{R}_l\mathcal{K}_l\mathcal{P}_l\right]^{-1}\mathcal{R}_l\mathcal{K}_l\right)\mathcal{M}_l. \end{aligned} \tag{4.98}$$

Here, we have set

$$\mathcal{P}_l = \mathcal{M}_l\mathcal{R}_l^T \tag{4.99}$$

which can be seen as a smoothed version of the prototype interpolation $\mathcal{R}_l^T$. We now have obtained a two-grid method for the matrix $\mathcal{K}_l$ with one pre-smoothing step which can be interpreted as a two-grid method for the operator $\mathcal{L}_l$, where the smoothing operator

$$\mathcal{M}_l = I - \hat{\mathcal{K}_l}^{-1}\mathcal{K}_l = I - \mathcal{Q}_l^{-1}\mathcal{L}_l$$

can be written using a symmetric positive definite operator $\mathcal{Q}_l$. From Theorem 4.6 we know that

$$\|I - 2\mathcal{Q}_l^{-1}\mathcal{L}_l\|_{\mathcal{Q}_l} \leq 1.$$

Hence, Corollary 2.1 implies that all eigenvalues λ of the two-grid error propagation operator $\mathcal{T}_l$ satisfy $|\lambda - \frac{1}{2}| \leq \frac{1}{2}$. In addition, the case $\lambda = 1$ is impossible due to the positive definiteness of $\mathcal{Q}_l$, see Corollary 2.1 and the discussion thereafter.
It remains to show that the coarse operator

$$\mathcal{K}_{l+1} = \mathcal{R}_l\mathcal{K}_l\mathcal{P}_l = \begin{pmatrix} A_{l+1} & B_{l+1}^T \\ B_{l+1} & -C_{l+1} \end{pmatrix},$$

where

$$A_{l+1} = R_{\mathcal{V}^l}\left(A_l(I - \hat{A}_l^{-1}A_l) + (I - A_l\hat{A}_l^{-1})B_l^T\hat{S}_l^{-1}B_l(I - \hat{A}_l^{-1}A_l)\right)R_{\mathcal{V}^l}^T, \tag{4.100}$$

$$B_{l+1} = R_{\mathcal{W}^l}\left(I - (B_l\hat{A}_l^{-1}B_l^T + C_l)\hat{S}_l^{-1}\right)B_l(I - \hat{A}_l^{-1}A_l)R_{\mathcal{V}^l}^T, \tag{4.101}$$

$$C_{l+1} = R_{\mathcal{W}^l}(B_l\hat{A}_l^{-1}B_l^T + C_l)\left(I - \hat{S}_l^{-1}(B_l\hat{A}_l^{-1}B_l^T + C_l)\right)R_{\mathcal{W}^l}^T, \tag{4.102}$$

is non-singular. We will show that, in contrast to the (semi-)algebraic methods introduced in Section 4.4, the regularity of the coarse system does not depend on geometric properties or "matching" coarse grids for the velocity and pressure components. Instead, the term C_{l+1} stabilizes the coarse system.
Let the projection operator $\Pi_{\mathcal{V}^l} : \mathcal{V}^l \to \mathcal{V}^{l+1}$ be such that $\Pi_{\mathcal{V}^l}R_{\mathcal{V}^l}^T v_{l+1} = v_{l+1}$ for all $v_{l+1} \in \mathcal{V}^{l+1}$. Such an operator can always be constructed if the tentative prolongation operator $R_{\mathcal{V}^l}^T$ has full column rank.

Lemma 4.4 *Let A_l be symmetric positive definite. Assume that for all $v_l \in \mathcal{V}^l$ we have*

$$\|v_l - (I - \hat{A}_l^{-1}A_l)R_{\mathcal{V}^l}^T\Pi_{\mathcal{V}^l}v_l\|_{\hat{A}_l}^2 \leq \beta_1\|v_l\|_{A_l}^2 \tag{4.103}$$

$$\|R_{\mathcal{V}^l}^T\Pi_{\mathcal{V}^l}v_l\|_{A_l}^2 \leq \beta_2\|v_l\|_{A_l}^2 \tag{4.104}$$

$$\sigma_1 v_l^T B_l^T\hat{S}_l^{-1}B_l v_l \leq v_l^T A_l v_l \tag{4.105}$$

for some constants $\sigma_1, \beta_1 > 0$. Then, the inf-sup-condition on the fine level,

$$\sup_{0 \neq v_l \in \mathcal{V}_1} \frac{v_l B_l^T p_l}{\|v_l\|_{A_l}} \geq c_l\|p_l\|_{\mathrm{M}_l} - d_l\left(p_l^T C_l p_l\right)^{\frac{1}{2}} \text{ for all } p_l \in \mathcal{W}^l \tag{4.106}$$

implies an inf-sup-condition on the coarse level,

$$\sup_{0\neq v_{l+1}\in\mathcal{V}^{l+1}} \frac{v_{l+1}B_{l+1}^T p_{l+1}}{\|v_{l+1}\|_{A_{l+1}}} \geq c_{l+1}\|p_{l+1}\|_{M_{l+1}} - d_{l+1}\left(p_{l+1}^T C_{l+1} p_{l+1}\right)^{\frac{1}{2}} \text{ for all } p_{l+1}\in\mathcal{W}^{l+1}. \tag{4.107}$$

where

$$c_{l+1} = \frac{1}{\sqrt{\beta_2\cdot(1+\frac{1}{\sigma_1})}}c_l \text{ and } d_{l+1} = \frac{1}{\sqrt{\beta_2\cdot(1+\frac{1}{\sigma_1})}}\left(\sqrt{\beta_1}+\max(d_l,\frac{1}{\sqrt{\sigma_1}})\right).$$

and $\mathrm{M}_{l+1} = R_{\mathcal{W}^l}\mathrm{M}_l R_{\mathcal{W}^l}^T$.

Proof: Our aim is to find a lower bound for the supremum

$$\sup_{0\neq v_{l+1}\in\mathcal{V}^{l+1}} \frac{v_{l+1}^T B_{l+1}^T p_{l+1}}{\|v_{l+1}\|_{A_{l+1}}}.$$

First, we need to estimate $\|v_{l+1}\|_{A_{l+1}}$. To this end, we use (4.100) re-write

$$\|v_{l+1}\|_{A_{l+1}}^2 = v_{l+1}^T R_{\mathcal{V}^l} A_l R_{\mathcal{V}^l}^T v_{l+1} \tag{4.108}$$
$$- v_{l+1}^T R_{\mathcal{V}^l}(A_l - A_l\hat{A}_l^{-1}A_l)R_{\mathcal{V}^l}^T v_{l+1} \tag{4.109}$$
$$+ v_{l+1}^T R_{\mathcal{V}^l}(I - A_l\hat{A}_l^{-1})B_l^T\hat{S}_l^{-1}B_l(I-\hat{A}_l^{-1}A_l)R_{\mathcal{V}^l}^T v_{l+1}. \tag{4.110}$$

From $\hat{A}_l > A_l$ it is immediately clear that $-v_{l+1}^T R_{\mathcal{V}^l}(A_l - A_l\hat{A}_l^{-1}A_l)R_{\mathcal{V}^l}^T v_{l+1} < 0$ and the term (4.109) can be omitted. To estimate (4.110), we employ (4.105),

$$\begin{aligned} &v_{l+1}^T R_{\mathcal{V}^l}(I - A_l\hat{A}_l^{-1})B_l^T\hat{S}_l^{-1}B_l(I-\hat{A}_l^{-1}A_l)R_{\mathcal{V}^l}^T v_{l+1} \\ &\leq \frac{1}{\sigma}v_{l+1}^T R_{\mathcal{V}^l}(I - A_l\hat{A}_l^{-1})A_l(I-\hat{A}_l^{-1}A_l)R_{\mathcal{V}^l}^T v_{l+1} \\ &\overset{\hat{A}_l>A_l}{\leq} \frac{1}{\sigma}v_{l+1}^T R_{\mathcal{V}^l}A_l R_{\mathcal{V}^l}^T v_{l+1}. \end{aligned}$$

We can hence estimate $\|v_{l+1}\|_{A_{l+1}} \leq \sqrt{1+\frac{1}{\sigma_1}}\|R_{\mathcal{V}^l}^T v_{l+1}\|_{A_l}$ and we obtain

$$\sup_{0\neq v_{l+1}\in\mathcal{V}^{l+1}} \frac{v_{l+1}^T B_{l+1}^T p_{l+1}}{\|v_{l+1}\|_{A_{l+1}}} \geq \sup_{0\neq v_{l+1}\in\mathcal{V}^{l+1}} \frac{v_{l+1}^T B_{l+1}^T p_{l+1}}{\sqrt{1+\frac{1}{\sigma_1}}\|R_{\mathcal{V}^l}^T v_{l+1}\|_{A_l}}$$

for all $p_{l+1}\in\mathcal{W}^{l+1}$. From the definition of $\Pi_{\mathcal{V}^l}$ it is clear that for any $v_{l+1}\in\mathcal{V}^{l+1}$ we have a (not necessarily unique) $v_l\in\mathcal{V}^l$ such that $v_{l+1}=\Pi_{\mathcal{V}^l}v_l$. This allows us to take the supremum over v_l instead of v_{l+1},

$$\sup_{0\neq v_{l+1}\in\mathcal{V}^{l+1}} \frac{v_{l+1}^T B_{l+1}^T p_{l+1}}{\sqrt{1+\frac{1}{\sigma_1}}\|R_{\mathcal{V}^l}^T v_{l+1}\|_{A_l}} = \sup_{0\neq v_l\in\mathcal{V}^l} \frac{v_l^T\Pi_{\mathcal{V}^l}^T B_{l+1}^T p_{l+1}}{\sqrt{1+\frac{1}{\sigma_1}}\|R_{\mathcal{V}^l}^T\Pi_{\mathcal{V}^l}v_l\|_{A_l}} \geq \sup_{0\neq v_l\in\mathcal{V}^l} \frac{v_l^T\Pi_{\mathcal{V}^l}^T B_{l+1}^T p_{l+1}}{\sqrt{\beta_2\cdot(1+\frac{1}{\sigma_1})}\|v_l\|_{A_l}}$$

where the last inequality is obtained from (4.104). Now, we consider the nominator

$$v_l^T \Pi_{\mathcal{V}^l}^T B_{l+1}^T p_{l+1} = v_l^T \Pi_{\mathcal{V}^l}^T R_{\mathcal{V}^l} (I - A_l \hat{A}_l^{-1}) B_l^T \left(I - \hat{S}_l^{-1} (B_l \hat{A}_l^{-1} B_l^T + C_l) \right) R_{\mathcal{W}^l}^T p_{l+1},$$

where we insert

$$\pm v_l^T B_l^T \left[I - \hat{S}_l^{-1} \left(B_l \hat{A}_l^{-1} B_l^T + C_l \right) \right] R_{\mathcal{W}^l}^T p_{l+1}$$

to obtain

$$v_l^T \Pi_{\mathcal{V}^l}^T B_{l+1}^T p_{l+1} = v_l^T B_l^T \left[I - \hat{S}_l^{-1} \left(B_l \hat{A}_l^{-1} B_l^T + C_l \right) \right] R_{\mathcal{W}^l}^T p_{l+1} \tag{4.111}$$

$$- \left[v_l - \left(I - \hat{A}_l^{-1} A \right) R_{\mathcal{V}^l}^T \Pi_{\mathcal{V}^l} v_l \right]^T B_l^T \left[I - \hat{S}_l^{-1} \left(B_l \hat{A}_l^{-1} B_l^T + C_l \right) \right] R_{\mathcal{W}^l}^T p_{l+1}. \tag{4.112}$$

We first regard (4.112),

$$\begin{aligned}
&\left[v_l - \left(I - \hat{A}_l^{-1} A \right) R_{\mathcal{V}^l}^T \Pi_{\mathcal{V}^l} v_l \right]^T \hat{A}_l^{1/2} \hat{A}_l^{-1/2} B_l^T \left[I - \hat{S}_l^{-1} \left(B_l \hat{A}_l^{-1} B_l^T + C_l \right) \right] R_{\mathcal{W}^l}^T p_{l+1} \\
\leq &\left\{ v_l^T \left[I - \left(I - \hat{A}_l^{-1} A \right) R_{\mathcal{V}^l}^T \Pi_{\mathcal{V}^l} \right]^T \hat{A}_l \left[I - \left(I - \hat{A}_l^{-1} A \right) R_{\mathcal{V}^l}^T \Pi_{\mathcal{V}^l} \right] v_l \right\}^{1/2} \\
\cdot &\left\{ p_{l+1}^T R_{\mathcal{W}^l} \left[I - \hat{S}_l^{-1} \left(B_l \hat{A}_l^{-1} B_l^T + C_l \right) \right]^T B_l \hat{A}_l^{-1} B_l^T \left[I - \hat{S}_l^{-1} \left(B_l \hat{A}_l^{-1} B_l^T + C_l \right) \right] R_{\mathcal{W}^l}^T p_{l+1} \right\}^{1/2}
\end{aligned}$$

We estimate the first factor,

$$\begin{aligned}
&\left\{ v_l^T \left[I - \left(I - \hat{A}_l^{-1} A \right) R_{\mathcal{V}^l}^T \Pi_{\mathcal{V}^l} \right]^T \hat{A}_l \left[I - \left(I - \hat{A}_l^{-1} A \right) R_{\mathcal{V}^l}^T \Pi_{\mathcal{V}^l} \right] v_l \right\}^{1/2} \\
&= \left\| \left[I - \left(I - \hat{A}_l^{-1} A \right) R_{\mathcal{V}^l}^T \Pi_{\mathcal{V}^l} \right] v_l \right\|_{\hat{A}_l} \\
&\overset{(4.103)}{\leq} \sqrt{\beta_1} \left\| v_l \right\|_{A_l}.
\end{aligned} \tag{4.113}$$

For the second factor, we employ $\rho \left(\hat{S}_l^{-1/2} \left(B \hat{A}_l^{-1} B_l^T + C_l \right) \hat{S}_l^{-1/2} \right) < 1$,

$$\begin{aligned}
&p_{l+1}^T R_{\mathcal{W}^l} \left[I - \hat{S}_l^{-1} \left(B_l \hat{A}_l^{-1} B_l^T + C_l \right) \right]^T B_l \hat{A}_l^{-1} B_l^T \left[I - \hat{S}_l^{-1} \left(B \hat{A}_l^{-1} B_l^T + C_l \right) \right] R_{\mathcal{W}^l}^T p_{l+1} \\
&\leq p_{l+1}^T R_{\mathcal{W}^l} \Big\{ B_l \hat{A}_l^{-1} B_l^T + C_l - 2 \left(B_l \hat{A}_l^{-1} B_l^T + C_l \right) \hat{S}_l^{-1} \left(B_l \hat{A}_l^{-1} B_l^T + C_l \right) \\
&\quad + \left(B_l \hat{A}_l^{-1} B_l^T + C_l \right) \hat{S}_l^{-1} \left(B_l \hat{A}_l^{-1} B_l^T + C_l \right) \hat{S}_l^{-1} \left(B_l \hat{A}_l^{-1} B_l^T + C_l \right) \Big\} R_{\mathcal{W}^l}^T p_{l+1} \\
&\leq p_{l+1}^T R_{\mathcal{W}^l} \Big\{ B_l \hat{A}_l^{-1} B_l^T + C_l - \left(B_l \hat{A}_l^{-1} B_l^T + C_l \right) \hat{S}_l^{-1} \left(B_l \hat{A}_l^{-1} B_l^T + C_l \right) \Big\} R_{\mathcal{W}^l}^T p_{l+1} \\
&= p_{l+1}^T C_{l+1} p_{l+1},
\end{aligned} \tag{4.114}$$

We combine (4.113) and (4.114) to obtain a lower limit for (4.112),

$$\begin{aligned} &- \left[v_l - \left(I - \hat{A}_l^{-1} A\right) R_{\mathcal{V}^l}^T \Pi_{\mathcal{V}^l} v_l\right]^T B_l^T \left[I - \hat{S}_l^{-1}\left(B_l \hat{A}_l^{-1} B_l^T + C_l\right)\right] R_{\mathcal{W}^l}^T p_{l+1} \\ &\qquad \geq -\sqrt{\beta_1}\, \|v_l\|_{A_l} \left(p_{l+1}^T C_{l+1} p_{l+1}\right)^{1/2} \end{aligned} \tag{4.115}$$

We now consider (4.111),

$$\begin{aligned} &v_l^T B_l^T \left[I - \hat{S}_l^{-1}\left(B_l \hat{A}_l^{-1} B_l^T + C_l\right)\right] R_{\mathcal{W}^l}^T p_{l+1} \\ &\qquad = v_l^T B_l^T R_{\mathcal{W}^l}^T p_{l+1} - v_l^T B_l^T \hat{S}_l^{-1}\left(B_l \hat{A}_l^{-1} B_l^T + C_l\right) R_{\mathcal{W}^l}^T p_{l+1}, \end{aligned} \tag{4.116}$$

and estimate the subtrahend of (4.116),

$$\begin{aligned} &v_l^T B_l^T \hat{S}_l^{-1}\left(B_l \hat{A}_l^{-1} B_l^T + C_l\right) R_{\mathcal{W}^l}^T p_{l+1} \\ &\quad = v_l^T B_l^T \hat{S}_l^{-1/2} \hat{S}_l^{-1/2}\left(B_l \hat{A}_l^{-1} B_l^T + C_l\right) R_{\mathcal{W}^l}^T p_{l+1} \\ &\quad \leq \left(v_l^T B_l^T \hat{S}_l^{-1} B_l v_l\right)^{1/2} \cdot \left(p_{l+1}^T R_{\mathcal{W}^l}\left(B_l \hat{A}_l^{-1} B_l^T + C_l\right) \hat{S}_l^{-1}\left(B_l \hat{A}_l^{-1} B_l^T + C_l\right) R_{\mathcal{W}^l}^T p_{l+1}\right)^{1/2} \\ &\quad \overset{(4.105)}{\leq} \frac{1}{\sqrt{\sigma_1}} \|v_l\|_{A_l} \cdot \left(p_{l+1}^T R_{\mathcal{W}^l}\left(B_l \hat{A}_l^{-1} B_l^T + C_l\right) \hat{S}_l^{-1}\left(B_l \hat{A}_l^{-1} B_l^T + C_l\right) R_{\mathcal{W}^l}^T p_{l+1}\right)^{1/2}, \end{aligned} \tag{4.117}$$

hence we can estimate (4.116),

$$\begin{aligned} &v_l^T B_l^T \left[I - \hat{S}_l^{-1}\left(B_l \hat{A}_l^{-1} B_l^T + C_l\right)\right] R_{\mathcal{W}^l}^T p_{l+1} \\ &\quad \geq v_l^T B_l^T R_{\mathcal{W}^l}^T p_{l+1} \\ &\qquad - \frac{1}{\sqrt{\sigma_1}} \|v_l\|_{A_l} \cdot \left(p_{l+1}^T R_{\mathcal{W}^l}\left(B_l \hat{A}_l^{-1} B_l^T + C_l\right) \hat{S}_l^{-1}\left(B_l \hat{A}_l^{-1} B_l^T + C_l\right) R_{\mathcal{W}^l}^T p_{l+1}\right)^{1/2} \end{aligned} \tag{4.118}$$

We are now ready to plug in the inf-sup-condition (4.106) for B_l^T and perform the induction,

$$\begin{aligned}
&\sup_{0\neq v_{l+1}\in\mathcal{V}^{l+1}} \frac{v_{l+1}^T B_{l+1}^T p_{l+1}}{\|v_{l+1}\|_{A_{l+1}}} \\
&\geq \sup_{0\neq v_l\in\mathcal{V}_l} \frac{v_l^T \Pi_{\mathcal{V}^l}^T B_{l+1}^T p_{l+1}}{\sqrt{\beta_2\cdot(1+\frac{1}{\sigma_1})}\|v_l\|_{A_l}} \\
&\overset{(4.115),(4.118)}{\geq} \frac{1}{\sqrt{\beta_2\cdot(1+\frac{1}{\sigma_1})}}\Big[\sup_{0\neq v_l\in\mathcal{V}_l}\frac{v_l^T B_l^T R_{\mathcal{W}^l}^T p_{l+1}}{\|v_l\|_{A_l}} - \sqrt{\beta_1}\left(p_{l+1}C_{l+1}p_{l+1}\right)^{1/2} \\
&\qquad - \frac{1}{\sqrt{\sigma_1}}\left(p_{l+1}^T R_{\mathcal{W}^l}\left(B_l\hat{A}_l^{-1}B_l^T + C_l\right)\hat{S}_l^{-1}\left(B_l\hat{A}_l^{-1}B_l^T + C_l\right)R_{\mathcal{W}^l}^T p_{l+1}\right)^{1/2}\Big] \\
&\overset{(4.106)}{\geq} \frac{1}{\sqrt{\beta_2\cdot(1+\frac{1}{\sigma_1})}}\Big[c_l\|p_{l+1}\|_{\mathrm{M}_{l+1}} - d_l\left(p_{l+1}^T R_{\mathcal{W}^l}C_l R_{\mathcal{W}^l}^T p_{l+1}\right)^{1/2} - \sqrt{\beta_1}\left(p_{l+1}C_{l+1}p_{l+1}\right)^{1/2} \\
&\qquad - \frac{1}{\sqrt{\sigma_1}}\left(p_{l+1}^T R_{\mathcal{W}^l}\left(B_l\hat{A}_l^{-1}B_l^T + C_l\right)\hat{S}_l^{-1}\left(B_l\hat{A}_l^{-1}B_l^T + C_l\right)R_{\mathcal{W}^l}^T p_{l+1}\right)^{1/2}\Big].
\end{aligned}$$

It remains to show that, for all $p_{l+1}\in\mathcal{W}^{l+1}$,

$$\begin{aligned}
&d_l\left(p_{l+1}^T R_{\mathcal{W}^l}C_l R_{\mathcal{W}^l}^T p_{l+1}\right)^{1/2} \\
&\quad + \frac{1}{\sqrt{\sigma_1}}\left(p_{l+1}^T R_{\mathcal{W}^l}\left(B_l\hat{A}_l^{-1}B_l^T + C_l\right)\hat{S}_l^{-1}\left(B_l\hat{A}_l^{-1}B_l^T + C_l\right)R_{\mathcal{W}^l}^T p_{l+1}\right)^{1/2} \\
&\leq \frac{\sqrt{\sigma_2}+1}{\sqrt{\sigma_2}-1}\cdot\max(d_l,\frac{1}{\sqrt{\sigma_1}})\left(p_{l+1}^T C_{l+1}p_{l+1}\right)^{1/2}.
\end{aligned}$$

This can be seen from

$$\frac{(\sqrt{x}+\sqrt{y})^2}{x-y} = \frac{1+\sqrt{\frac{2y}{x}}+\frac{y}{x}}{1-\frac{y}{x}} < 1$$

for positive numbers $x > y > 0$. Here we have

$$\begin{aligned}
x &= p_{l+1}^T R_{\mathcal{W}^l}\left(B_l\hat{A}_l^{-1}B_l^T + C_l\right)R_{\mathcal{W}^l}^T p_{l+1}, \\
y &= p_{l+1}^T R_{\mathcal{W}^l}\left(B_l\hat{A}_l^{-1}B_l^T + C_l\right)\hat{S}_l^{-1}\left(B_l\hat{A}_l^{-1}B_l^T + C_l\right)R_{\mathcal{W}^l}^T p_{l+1}, \\
\text{and } x-y &= p_{l+1}^T R_{\mathcal{W}^l}C_{l+1}R_{\mathcal{W}^l}^T p_{l+1}.
\end{aligned}$$

where we have used $\hat{S}_l > \left(B_l\hat{A}_l^{-1}B_l^T + C_l\right)$. (If $p_{l+1}^T R_{\mathcal{W}^l}\left(B_l\hat{A}_l^{-1}B_l^T + C_l\right)R_{\mathcal{W}^l}^T p_{l+1} = 0$, for a $p_{l+1}^T \neq 0$, this implies $B_l^T R_{\mathcal{W}^l}^T p_{l+1} = C_l R_{\mathcal{W}^l}^T p_{l+1} = 0$, in contradiction to (4.106).) We now have shown the inf-sup-condition for the coarse system with the constants

$$c_{l+1} = \frac{1}{\sqrt{\beta_2\cdot(1+\frac{1}{\sigma_1})}}c_l \text{ and } d_{l+1} = \frac{1}{\sqrt{\beta_2\cdot(1+\frac{1}{\sigma_1})}}\left(\sqrt{\beta_1}+\max(d_l,\frac{1}{\sqrt{\sigma_1}})\right).$$

□

In the remainder of the section, we discuss how the assumptions of Lemma 4.4 can be fulfilled. First, we address condition (4.104), which can easily be satisfied if A_l is a M-matrix or of essentially positive type and we employ point-wise coarsening for the velocity (i.e Π_l^T is the trivial injection from level $l+1$ into level l). Letting D_l be the diagonal of A_l, we assume that the tentative interpolation operator $R_{\mathcal{V}_l}^T$ is such that we have an approximation property (cf. Theorem 2.6),

$$\|v_F - R_{FC}^T v_C\|_{D_l,F}^2 \leq \tau \|v_l\|_{A_l}^2 \tag{4.119}$$

where v_C and v_F respectively denote the vector v_l at the coarse grid and the fine grid points, R_{FC}^T denotes the block of $R_{\mathcal{V}^l}^T$ that contains the interpolation weights for the fine grid variables and $\|v_l\|_{D_l,F}$ is the norm induced by D_l restricted to the fine grid variables. We then have

$$\|v_l - R_{\mathcal{V}^l}^T \Pi_{\mathcal{V}^l} v_l\|_{D_l}^2 \leq \tau \|v_l\|_{A_l}^2. \tag{4.120}$$

In addition, for an essentially positive type matrix we have

$$\frac{2}{c} v_l^T D_l v_l \geq v_l^T A_l v_l \tag{4.121}$$

for some constant c independent of v_l (2.28). Combining these inequalities, we get ([Wab03], p. 54)

$$\begin{aligned} \|R_{\mathcal{V}^l}^T \Pi_{\mathcal{V}^l} v_l\|_{A_l} - \|v_l\|_{A_l} &\leq \|v_l - R_{\mathcal{V}^l}^T \Pi_{\mathcal{V}^l} v_l\|_{A_l} \\ &\leq \sqrt{\frac{2}{c}} \|v_l - P_{\mathcal{V}^l} \Pi_{\mathcal{V}^l} v_l\|_{D_l} \\ &\leq \sqrt{\frac{2\tau}{c}} \|v_l\|_{A_l} \end{aligned}$$

from which we conclude that $\|P_{\mathcal{V}} \Pi_{\mathcal{V}^l} v_l\|_{A_l}^2 \leq \beta_2 \|v_l\|_{A_l}^2$ (4.104) holds with $\beta_2 := \left(1 + \sqrt{\frac{2\tau}{c}}\right)^2$. To see (4.103), we first note that $\hat{A}_l = \alpha D_l$ for some $\alpha > 1$. Hence we can re-write the left hand side of (4.103) as

$$\sqrt{\alpha} \left\| v_l - (I - \hat{A}_l^{-1} A_l) R_{\mathcal{V}^l}^T \Pi_{\mathcal{V}^l} v_l \right\|_{D_l} \leq \sqrt{\alpha} \left[\left\| v_l - R_{\mathcal{V}^l}^T \Pi_{\mathcal{V}^l} v_l \right\|_{D_l} + \left\| R_{\mathcal{V}^l}^T \Pi_{\mathcal{V}^l} v_l \right\|_{A_l D_l^{-1} A_l} \right].$$

We again use (4.120) to estimate the first summand. For the second summand, we employ (2.25) and (4.104),

$$\left\| R_{\mathcal{V}^l}^T \Pi_{\mathcal{V}^l} v_l \right\|_{A_l D_l^{-1} A_l} \leq \sqrt{\rho(D_l^{-1} A_l)} \|R_{\mathcal{V}^l}^T \Pi_{\mathcal{V}^l} v_l\|_{A_l} \leq \sqrt{\beta_2 \rho(D_l^{-1} A_l)} \|v_l\|_{A_l}^2.$$

Combining these estimates, we obtain (4.103) with $\beta_1 = \alpha \left(\sqrt{\tau} + \sqrt{\beta_2 \rho(D_l^{-1} A_l)} \right)^2$. Note that a smaller (more accurately computed) value of α leads to a smaller value of

β_1, and, in consequence, a smaller value of d_{l+1}.
For diagonally dominant matrices A_l, we know that $\rho(D_l^{-1}A_l) \leq 2$. This spectral radius may even be replaced by a much smaller constant if the tentative interpolation operator $R_{\mathcal{V}^l}^T$ only maps into the space of smooth error components (w.r.t. A_l), as for these e_l we have $\|e\|_{A_l D_l^{-1} A_l} \ll \|e\|_{A_l}$, see Section 2.4.
Regarding (4.105), the value of σ_1 depends on the interplay between $B_l\hat{S}_l^{-1}B_l^T$ and A_l, i.e. requires that low-energy modes of A_l also correspond to low-energy-modes of $B_l\hat{S}_l^{-1}B_l^T$ (if not, σ_1 must be chosen smaller or $\hat{S}_l$ must be multiplied with a larger constant, at the cost of a less efficient smoother). Apart from this, the stability proof does not depend on the interplay between the velocity and the pressure spaces, in particular we do not require anything on the interplay between the images of $R_{\mathcal{V}^l}^T$ and $R_{\mathcal{W}^l}^T$. This allows us to set up the coarse grids for the velocity and the pressure part separately, while the term

$$C_{l+1} = R_{\mathcal{W}^l}(B_l\hat{A}_l^{-1}B_l^T + C_l)\left(I - \hat{S}_l^{-1}(B_l\hat{A}_l^{-1}B_l^T + C_l)\right)R_{\mathcal{W}^l}^T \tag{4.122}$$

ensures the stability.

Remark 4.5 In the case of a singular A_l, we construct a matrix $A_{D_l} > A_l$ and replace, in (4.103), (4.104) and (4.120) the norm $\|\cdot\|_{A_l}$ by a norm $\|\cdot\|_{A_{D_l}}$. In addition, in (4.105), we substitute A_l by A_{D_l}. On the other hand, in (4.103) the term $I - \hat{A}_l^{-1}A_l$ is not modified. If A_l is a weakly diagonally dominant matrix, such an A_{D_l} can e.g. be obtained by multiplying the diagonal of A_l by $1+\epsilon$ for an $\epsilon > 0$. Hence, it is still easy to see (4.121) using a slightly smaller constant c and to show the prerequisites of Lemma 4.4 for essentially positive matrices.

Remark 4.6 As we usually have more than one velocity unknown, the A part itself is the discretization of a system of elliptic partial differential equations. For these kind of matrices, we have described various AMG approaches in Section 2.12. For the considerations in this section (and in the next sections), we assumed that the tentative interpolation operator $R_{\mathcal{V}_l}^T$ was constructed using a variable-based AMG (Section 2.12.1) approach for A, i.e. we just ignored its decomposition according to physical unknowns. If we employ unknown-based AMG (Section 2.12.2), the approximation property for interpolation reads (2.87)

$$\|v_F - P_{FC}v_C\|_{0,F}^2 \leq \tau_u \|v_l\|_{u,1}^2$$

where $\|\cdot\|_{u,1}$ is induced by the block diagonal matrix A_u (2.86) that only contains the couplings within each physical unknown. Inequality (4.119) then follows with $\tau = \rho(A^{-1}A_u)\tau_u$.
In the case of point-based AMG (Section 2.12.3), where the matrix A is organized in blocks $A_{(i,j)}$ corresponding to (discretization) points, we must distinguish between two cases.
First, if the interpolation operator involves the full inverses of the diagonal block matrices $A_{(i,i)}$, we have an approximation property (2.107)

$$\|v_F - P_{FC}v_C\|_{P,0,F}^2 \leq \tau \|v\|_1^2$$

where $\| \cdot \|_{P,0,F}$ is the norm induced by the block diagonal matrix $D_P = diag(A_{(i,i)})_i$ (2.93), restricted to the fine grid points. In this case, we need to define $\hat{A}_l = \alpha D_P$ with an appropriate scaling factor α to obtain the stability results in this and the following sections. In consequence, the smoother should also be built around the block diagonal matrix $\hat{A}_l$, see Remark 4.2. In the other case, where just the diagonal of A_l is inverted during prolongation setup, we have the classical approximation property 4.119, see Remarks 2.7 and 2.9. Of course, we just use the diagonal in the smoother here.

4.9. The coarse grid operator II: A sparser stable coarse operator

In the last section we have introduced a stable coarse grid operator for the saddle point problem. The stability was obtained by applying the smoothing operator (4.67)

$$\mathcal{M}_l = \left(I - \hat{\mathcal{K}_l}^{-1} \mathcal{K}_l\right)$$

to a tentative block-wise prolongator

$$\mathcal{R}_l^T = \begin{pmatrix} R_{\mathcal{V}^l}^T & 0 \\ 0 & R_{\mathcal{W}^l}^T \end{pmatrix}.$$

While we were able to show stability and two-grid convergence for this approach, we have two more or less severe drawbacks. First, the application of the smoother

$$\mathcal{M} = \begin{pmatrix} I & -\hat{A}_l^{-1} B_l^T \\ 0 & I \end{pmatrix} \begin{pmatrix} I & 0 \\ \hat{S}_l^{-1} B_l & I \end{pmatrix} \begin{pmatrix} I - \hat{A}_l^{-1} A_l & 0 \\ 0 & I - \hat{S}_l^{-1}\left(B_l \hat{A}_l^{-1} B_l^T + C_l\right) \end{pmatrix} \tag{4.123}$$

to the tentative interpolation operator $\mathcal{R}_l^T$ heavily enlarges the number of non-zero entries of the final interpolation operator $\mathcal{P}_l$:
First, a Jacobi-like iteration increases the radius of interpolation by the "stencil width" of the operators A_l and $B_l \hat{A}_l^{-1} B_l^T + C_l$ and then two additional coupling operators introduce not only connections between the velocity and pressure parts, but together also perform a further smoothing sweep over the velocity variables were the operator is given by $B^T \hat{S}^{-1} B$.
Second, no interpolation truncation can be applied *after* this stabilization process, as the restriction operator remains $\mathcal{R}_l$ and a truncation of $\mathcal{P}_l$ would destroy the symmetry of $\mathcal{K}_{l+1} = \mathcal{R}_l \mathcal{K}_l \mathcal{P}_l$.
In this section we introduce a different stabilization operator, which is applied to both the prolongation and the restriction. Recall that in the last section, the prolongation smoothing resulted in a stabilization matrix C_{l+1} that contains terms of the kind $B_l \hat{A}_l^{-1} B_l^T$, see (4.122). This motivates us to choose the simple ansatz

$$\mathcal{P}_l = \begin{pmatrix} I & -\hat{A}_l^{-1} B_l^T \\ 0 & I \end{pmatrix} \mathcal{R}_l^T \tag{4.124}$$

to obtain the stabilized interpolation $\mathcal{P}_l$. A simple calculation shows that the coarse matrix then reads

$$\mathcal{K}_{l+1} = \mathcal{P}_l^T \mathcal{K}_l \mathcal{P}_l = \mathcal{R}_l \begin{pmatrix} A_l & \left(I - A_l \hat{A}_l^{-1}\right) B_l^T \\ B_l \left(I - \hat{A}_l^{-1} A_l\right) & B_l \hat{A}_l^{-1} A \hat{A}_l^{-1} B_l^T - 2 B_l \hat{A}^{-1} B_l^T - C_l \end{pmatrix} \mathcal{R}_l^T \tag{4.125}$$

The stability of the coarse system $\mathcal{K}_{l+1}$ is considered in the following lemma. Let again $\Pi_{\mathcal{V}^l} : \mathcal{V}^l \to \mathcal{V}^{l+1}$ be such that $\Pi_{\mathcal{V}^l} R_{\mathcal{V}^l}^T v_{l+1} = v_{l+1}$ for all $v_{l+1} \in \mathcal{V}^{l+1}$.

Lemma 4.5 *Let A_l be symmetric positive definite. Assume that for all $v_l \in \mathcal{V}^l$ we have*

$$\begin{aligned} \|v_l - (I - \hat{A}^{-1} A) R_{\mathcal{V}^l}^T \Pi_{\mathcal{V}^l} v_l\|_{\hat{A}_l}^2 &\le \beta_3 \|v_l\|_{A_l}^2 && (4.126) \\ \|R_{\mathcal{V}^l}^T \Pi_{\mathcal{V}^l} v_l\|_{A_l}^2 &\le \beta_4 \|v_l\|_{A_l}^2 && (4.127) \end{aligned}$$

for some constants $\beta_3, \beta_4 > 0$. Then, the inf-sup-condition on the fine level,

$$\sup_{0 \neq v_l \in \mathcal{V}_1} \frac{v_l B_l^T p_l}{\|v_l\|_{A_l}} \ge c_l \|p_l\|_{\mathrm{M}_l} - d_l \left(p_l^T C_l p_l\right)^{\frac{1}{2}} \text{ for all } p_l \in \mathcal{W}^l \tag{4.128}$$

implies an inf-sup-condition on the coarse level,

$$\sup_{0 \neq v_{l+1} \in \mathcal{V}^{l+1}} \frac{v_{l+1} B_{l+1}^T p_{l+1}}{\|v_{l+1}\|_{A_{l+1}}} \ge c_{l+1} \|p_{l+1}\|_{\mathrm{M}_{l+1}} - d_{l+1} \left(p_{l+1}^T C_{l+1} p_{l+1}\right)^{\frac{1}{2}} \text{ for all } p_{l+1} \in \mathcal{W}^{l+1}. \tag{4.129}$$

where

$$c_{l+1} = \frac{1}{\sqrt{\beta_4}} c_l \text{ and } d_{l+1} = \frac{1}{\sqrt{\beta_4}} \max(d_l, \sqrt{\beta_3}).$$

and $\mathrm{M}_{l+1} = R_{\mathcal{W}^l} \mathrm{M}_l R_{\mathcal{W}^l}^T$.

Proof: In this case, we have $\|v_{l+1}\|_{A_{l+1}} = \|R_{\mathcal{V}^l}^T v_{l+1}\|_{A_l}$. Again, the definition of $\Pi_{\mathcal{V}^l}$ allows us to choose a not necessarily unique $v_l \in \mathcal{V}^l$ such that $v_{l+1} = \Pi_{\mathcal{V}^l} v_l$. Hence, we can take the supremum over v_l instead of v_{l+1},

$$\sup_{0 \neq v_{l+1} \in \mathcal{V}^{l+1}} \frac{v_{l+1}^T B_{l+1}^T p_{l+1}}{\|R_{\mathcal{V}^l}^T v_{l+1}\|_{A_l}} = \sup_{0 \neq v_l \in \mathcal{V}^l} \frac{v_l^T \Pi_{\mathcal{V}^l}^T B_{l+1}^T p_{l+1}}{\|R_{\mathcal{V}^l}^T \Pi_{\mathcal{V}^l} v_l\|_{A_l}} \overset{(4.127)}{\ge} \sup_{0 \neq v_l \in \mathcal{V}^l} \frac{v_l^T \Pi_{\mathcal{V}^l}^T B_{l+1}^T p_{l+1}}{\sqrt{\beta_4} \|v_l\|_{A_l}} \tag{4.130}$$

We re-write $B_{l+1}^T = \left(I - A_l \hat{A}_l^{-1}\right) B_l^T$ as in (4.125) and insert $\pm v_l B_l^T R_{\mathcal{W}^l}^T p_{l+1}$ into the nominator of the rightmost term of (4.130),

$$\begin{aligned} v_l^T \Pi_{\mathcal{V}^l}^T B_{l+1}^T p_{l+1} &= v_l^T B_l R_{\mathcal{W}^l}^T p_{l+1} && (4.131) \\ &- \left[v_l - \left(I - \hat{A}_l^{-1} A\right) R_{\mathcal{V}^l}^T \Pi_{\mathcal{V}^l} v_l\right]^T B_l^T R_{\mathcal{W}^l}^T p_{l+1}. && (4.132) \end{aligned}$$

We estimate (4.132),

$$\begin{aligned}&\left[v_l - \left(I - \hat{A}_l^{-1}A\right) R_{\mathcal{V}^l}^T \Pi_{\mathcal{V}^l} v_l\right]^T \hat{A}_l^{1/2}\hat{A}_l^{-1/2} B_l^T R_{\mathcal{W}^l}^T p_{l+1}\\ &\quad\le \left[v_l - \left(I - \hat{A}_l^{-1}A\right) R_{\mathcal{V}^l}^T \Pi_{\mathcal{V}^l} v_l\right]^T \hat{A}_l \left[v_l - \left(I - \hat{A}_l^{-1}A\right) R_{\mathcal{V}^l}^T \Pi_{\mathcal{V}^l} v_l\right]\\ &\qquad\cdot p_{l+1}^T R_{\mathcal{W}^l} B_l \hat{A}_l^{-1} B_l^T R_{\mathcal{W}^l}^T p_{l+1}.\end{aligned}$$

From $\rho(\hat{A}_l^{-1/2} A_l \hat{A}_l^{-1/2}) < 1$ we obtain $B_l \hat{A}_l^{-1} A_l \hat{A}_l^{-1} B_l^T < B_l \hat{A}_l^{-1} B_l^T$ and hence, cf. (4.125),

$$R_{\mathcal{W}^l} B_l \hat{A}_l^{-1} B_l^T R_{\mathcal{W}^l}^T < R_{\mathcal{W}^l}\left(C_l + 2B_l\hat{A}_l^{-1}B_l^T - B_l\hat{A}_l^{-1}A_l\hat{A}_l^{-1}B_l^T\right)R_{\mathcal{W}^l}^T = C_{l+1}. \tag{4.133}$$

The first factor can be estimated using (4.126),

$$\begin{aligned}&\left[v_l - \left(I - \hat{A}_l^{-1}A\right) R_{\mathcal{V}^l}^T \Pi_{\mathcal{V}^l} v_l\right]^T \hat{A}_l \left[v_l - \left(I - \hat{A}_l^{-1}A\right) R_{\mathcal{V}^l}^T \Pi_{\mathcal{V}^l} v_l\right]\\ &\quad= \left\|v_l - \left(I - \hat{A}_l^{-1}A\right) R_{\mathcal{V}^l}^T \Pi_{\mathcal{V}^l} v_l\right\|_{\hat{A}_l}\\ &\quad\le \sqrt{\beta_3}\|v_l\|_{A_l},\end{aligned} \tag{4.134}$$

and combining (4.133) and (4.134) we obtain

$$\left[v_l - \left(I - \hat{A}_l^{-1}A\right) R_{\mathcal{V}^l}^T \Pi_{\mathcal{V}^l} v_l\right]^T B_l^T R_{\mathcal{W}^l}^T p_{l+1} \le \sqrt{\beta_3}\|v_l\|_{A_l} \cdot \left(p_{l+1}^T C_{l+1} p_{l+1}\right)^{1/2}. \tag{4.135}$$

Finally, we perform the induction,

$$\begin{aligned}&\sup_{0\ne v_{l+1}\in\mathcal{V}^{l+1}} \frac{v_{l+1}^T B_{l+1}^T p_{l+1}}{\|v_{l+1}\|_{A_{l+1}}}\\ \overset{(4.130)}{\ge}\ &\sup_{0\ne v_l\in\mathcal{V}_l} \frac{v_l^T \Pi_{\mathcal{V}^l}^T B_{l+1}^T p_{l+1}}{\sqrt{\beta_4}\|v_l\|_{A_l}}\\ \overset{(4.135)}{\ge}\ &\frac{1}{\sqrt{\beta_4}}\left[\sup_{0\ne v_l\in\mathcal{V}_l} \frac{v_l^T B_l^T R_{\mathcal{W}^l}^T p_{l+1}}{\|v_l\|_{A_l}} - \sqrt{\beta_3}\left(p_{l+1}^T C_{l+1} p_{l+1}\right)^{1/2}\right]\\ \overset{(4.128)}{\ge}\ &\frac{1}{\sqrt{\beta_4}}\left[c_l\|p_{l+1}\|_{\mathrm{M}_{l+1}} - d_l\left(p_{l+1}^T R_{\mathcal{W}^l} C_l R_{\mathcal{W}^l}^T p_{l+1}\right)^{1/2} - \sqrt{\beta_3}\left(p_{l+1}^T C_{l+1} p_{l+1}\right)^{1/2}\right]\\ \ge\ &\frac{1}{\sqrt{\beta_4}}\left[c_l\|p_{l+1}\|_{\mathrm{M}_{l+1}} - \tilde{d}\left(p_{l+1}^T C_{l+1} p_{l+1}\right)^{1/2}\right]\end{aligned}$$

where $\tilde{d} = \max(d_l, \sqrt{\beta_3})$.
We have shown the inf-sup-condition for the coarse system with the constants

$$C_{l+1} = \frac{1}{\sqrt{\beta_4}} c_l \text{ and } d_{l+1} = \frac{1}{\sqrt{\beta_4}} \max(d_l, \sqrt{\beta_3}). \qquad \square$$

For a discussion on the conditions (4.126) and (4.127), we refer to the previous section. Note that we have no condition on the interplay (4.105) between A_l and $B_l^T \hat{S}_l^{-1} B_l$ here.

4.10. The coarse grid operator III: F-stabilization

We again take a look at the coarse operator $\mathcal{K}_{l+1}$ (4.125) and note that the coarse coupling operator

$$B_{l+1} = R_{\mathcal{W}^l} B_l \left(I - \hat{A}_l^{-1} A_l \right) R_{\mathcal{V}^l}^T$$

actually contains a Jacobi-like smoothed version of the velocity prolongation $R_{\mathcal{V}^l}^T$. Note however, that in contrast to Jacobi interpolation as introduced in Section 2.8.8, here also the rows corresponding to the coarse variables $i \in C$ are relaxed. While in the case of of smoothed aggregation AMG methods (Section 2.11.2), where we do not have "coarse" and "fine" points but form aggregates of points to obtain the coarse level, this can be seen as just an additional smoothing of the tentative interpolation operator, in classical AMG we follow the principle that an error e_i at a coarse grid point $i \in C$ should be interpolated just by injection from its coarse counterpart, see Section 2.8. Now, the question arises whether it would be sufficient, in (4.124), to just introduce the coupling for the fine velocity rows $i \in F \cap \mathcal{V}^l$, i.e.

$$\mathcal{P}_l = \begin{pmatrix} I_{FF} & 0 & -\hat{A}_{FF}^{-1} B_F^T \\ 0 & I_{CC} & 0 \\ 0 & 0 & I \end{pmatrix} \begin{pmatrix} R_{\mathcal{V},CF}^T & 0 \\ I_{CC} & 0 \\ 0 & R_{\mathcal{W}^l}^T \end{pmatrix}. \tag{4.136}$$

Here, like in Section 2.8.8, we have re-arranged the coarse and the fine velocity variables v_l such that we can write the operators $\hat{A}_l$, $\mathcal{K}_l$ and $\mathcal{R}_l^T$ in block form,

$$v_l = \begin{pmatrix} v_F \\ v_C \end{pmatrix}, \quad \hat{A}_l = \begin{pmatrix} \hat{A}_{FF} & 0 \\ 0 & \hat{A}_{CC} \end{pmatrix}, \quad \mathcal{K}_l = \begin{pmatrix} A_{FF} & A_{FC} & B_F^T \\ A_{CF} & A_{CC} & B_C^T \\ B_F & B_C & -C_l \end{pmatrix}, \quad \mathcal{R}_l^T = \begin{pmatrix} R_{\mathcal{V},CF}^T & 0 \\ I_{CC} & 0 \\ 0 & R_{\mathcal{W}^l}^T \end{pmatrix}.$$

Here, B_C and B_F denote the parts of B that correspond to the coarse and fine velocity variables, respectively. (The further considerations do not require us to separate the pressure variables in our notation. Also note that for sake of readability we have omitted the level index l inside the matrix blocks.)
We now obtain the following coarse system,

$$\mathcal{K}_{l+1} = \mathcal{P}_l^T \mathcal{K}_l \mathcal{P}_l = \begin{pmatrix} A_{l+1} & B_{l+1}^T \\ B_{l+1} & -C_{l+1} \end{pmatrix} \tag{4.137}$$

where

$$A_{l+1} = R_{\mathcal{V},CF} A_{FF} R_{\mathcal{V},CF}^T + R_{\mathcal{V},CF} A_{FC} + A_{CF} R_{\mathcal{V},CF}^T + A_{CC}, \tag{4.138}$$

$$B_{l+1} = R_{\mathcal{W}^l} B_F \left(R_{\mathcal{V},CF}^T - \hat{A}_{FF}^{-1} \left(A_{FF} R_{\mathcal{V},CF}^T + A_{FC} \right) \right) + B_C, \tag{4.139}$$

$$C_{l+1} = R_{\mathcal{W}^l} \left[C_l + 2 B_F \hat{A}_{FF}^{-1} B_F^T + B_F \hat{A}_{FF}^{-1} A_{FF} \hat{A}_{FF}^{-1} B_F^T \right] R_{\mathcal{W}^l}^T. \tag{4.140}$$

To see the relation to Jacobi interpolation as introduced in (2.61) we rewrite the right hand side of (4.139)

$$B_{l+1} = R_{\mathcal{W}^l} \begin{pmatrix} B_F & B_C \end{pmatrix} \begin{pmatrix} R_{\mathcal{V},CF}^T - \hat{A}_{FF}^{-1} \left(A_{FF} R_{\mathcal{V},CF}^T + A_{FC} \right) \\ I_{CC} \end{pmatrix} \tag{4.141}$$

and conclude that, up to the choice of $\hat{A}_{FF}$, the rightmost matrix gives us a Jacobi interpolation operator for the velocity. Now, using the approximation result from Theorem 2.14, we can show the stability for this coarse system.

Lemma 4.6 *Let A_l be a symmetric positive definite matrix and let the projection on the coarse level $\Pi_{\mathcal{V}^l}$ be defined by*

$$\Pi_{\mathcal{V}^l} = \begin{pmatrix} 0 & I_{CC} \end{pmatrix}. \tag{4.142}$$

Assume that for all $v_l \in \mathcal{V}^l$ we have

$$\begin{aligned} \left\| v_F - \left[R^T_{\mathcal{V},CF} - \hat{A}^{-1}_{FF} \left(A_{FF} R^T_{\mathcal{V},CF} + A_{FC} \right) \right] v_C \right\|^2_{\hat{A}_{FF}} &\leq \beta_5 \|v_l\|^2_{A_l} && (4.143) \\ \left\| R^T_{\mathcal{V}^l} \Pi_{\mathcal{V}^l} v_l \right\|^2_{A_l} &\leq \beta_6 \|v_l\|^2_{A_l} && (4.144) \end{aligned}$$

for some constants $\beta_5, \beta_6 > 0$. Then, the inf-sup-condition on the fine level,

$$\sup_{0 \neq v_l \in \mathcal{V}_1} \frac{v_l B_l^T p_l}{\|v_l\|_{A_l}} \geq c_l \|p_l\|_{\mathrm{M}_l} - d_l \left(p_l^T C_l p_l \right)^{\frac{1}{2}} \text{ for all } p_l \in \mathcal{W}^l \tag{4.145}$$

implies an inf-sup-condition on the coarse level,

$$\sup_{0 \neq v_{l+1} \in \mathcal{V}^{l+1}} \frac{v_{l+1} B_{l+1}^T p_{l+1}}{\|v_{l+1}\|_{A_{l+1}}} \geq c_{l+1} \|p_{l+1}\|_{\mathrm{M}_{l+1}} - d_{l+1} \left(p_{l+1}^T C_{l+1} p_{l+1} \right)^{\frac{1}{2}} \text{ for all } p_{l+1} \in \mathcal{W}^{l+1}. \tag{4.146}$$

where

$$c_{l+1} = \frac{1}{\sqrt{\beta_6}} c_l \text{ and } d_{l+1} = \frac{1}{\sqrt{\beta_6}} \max(d_l, \sqrt{\beta_5}).$$

and $\mathrm{M}_{l+1} = R_{\mathcal{W}^l} \mathrm{M}_l R^T_{\mathcal{W}^l}$.

Proof: The proof is very similar to the proof of Lemma 4.5.
We have $\|v_{l+1}\|_{A_{l+1}} = \|R^T_{\mathcal{V}^l} v_{l+1}\|_{A_l}$. From the definition of $\Pi_{\mathcal{V}^l}$ (4.142) it is clear that we can rewrite any $v_{l+1} \in \mathcal{V}^{l+1}$ as $v_{l+1} = \Pi_{\mathcal{V}^l} v_l$, and, in the leftmost term of (4.146), take the supremum over v_l instead of v_{l+1},

$$\sup_{0 \neq v_{l+1} \in \mathcal{V}^{l+1}} \frac{v_{l+1}^T B_{l+1}^T p_{l+1}}{\|R^T_{\mathcal{V}^l} v_{l+1}\|_{A_l}} = \sup_{0 \neq v_l \in \mathcal{V}^l} \frac{v_l^T \Pi^T_{\mathcal{V}^l} B^T_{l+1} p_{l+1}}{\|R^T_{\mathcal{V}^l} \Pi_{\mathcal{V}^l} v_l\|_{A_l}} \geq \sup_{0 \neq v_l \in \mathcal{V}^l} \frac{v_l^T \Pi^T_{\mathcal{V}^l} B^T_{l+1} p_{l+1}}{\sqrt{\beta_6} \|v_l\|_{A_l}} \tag{4.147}$$

where we have used (4.144) in the last inequality. We insert $\pm v_l B_l^T R^T_{\mathcal{W}^l} p_{l+1}$ into the nominator of (4.147),

$$v_l^T \Pi^T_{\mathcal{V}^l} B^T_{l+1} p_{l+1} = v_l^T B_l^T R^T_{\mathcal{W}^l} p_{l+1} - \left\{ v_l^T \left[B_l^T R^T_{\mathcal{W}^l} - \Pi^T_{\mathcal{V}^l} B^T_{l+1} \right] p_{l+1} \right\}. \tag{4.148}$$

We now use the definition of B_{l+1} (4.141),

$$B^T_{l+1} = \begin{pmatrix} R^T_{\mathcal{V},CF} - \hat{A}^{-1}_{FF} \left(A_{FF} R^T_{\mathcal{V},CF} + A_{FC} \right) \\ I_{CC} \end{pmatrix}^T \begin{pmatrix} B_F^T \\ B_C^T \end{pmatrix} R_{\mathcal{W}^l}$$

and replace B_{l+1}^T in the rightmost term of (4.148),

$$\begin{aligned}
&v_l^T \left[B_l^T R_{\mathcal{W}^l}^T - \Pi_{\mathcal{V}^l}^T B_{l+1}^T\right] p_{l+1} \\
&= v_l^T \left[\begin{pmatrix} I_{FF} & 0 \\ 0 & I_{CC}\end{pmatrix} - \begin{pmatrix} R_{\mathcal{V},CF}^T - \hat{A}_{FF}^{-1}\left(A_{FF}R_{\mathcal{V},CF}^T + A_{FC}\right) \\ I_{CC}\end{pmatrix} \Pi_{\mathcal{V}^l}\right]^T \begin{pmatrix} B_F^T \\ B_C^T\end{pmatrix} R_{\mathcal{W}^l}^T p_{l+1} \\
&\overset{(4.142)}{=} v_l^T \left[\begin{pmatrix} I_{FF} & 0 \\ 0 & I_{CC}\end{pmatrix} - \begin{pmatrix} R_{\mathcal{V},CF}^T - \hat{A}_{FF}^{-1} A_{FF} R_{\mathcal{V},CF}^T - \hat{A}_{FF}^{-1} A_{FC} \\ I_{CC}\end{pmatrix} \begin{pmatrix} 0 & I_{CC}\end{pmatrix}\right]^T \begin{pmatrix} B_F^T \\ B_C^T\end{pmatrix} R_{\mathcal{W}^l}^T p_{l+1} \\
&= v_l^T \begin{pmatrix} I_{FF} & \hat{A}_{FF}^{-1}\left(A_{FF}R_{FC}^T + A_{FC}\right) - R_{\mathcal{V},CF}^T \\ 0 & 0\end{pmatrix}^T \begin{pmatrix} \hat{A}_{FF}^{1/2} & 0 \\ 0 & 0\end{pmatrix} \begin{pmatrix} \hat{A}_{FF}^{-1/2} & 0 \\ 0 & 0\end{pmatrix} \begin{pmatrix} B_F^T \\ B_C^T\end{pmatrix} R_{\mathcal{W}^l}^T p_{l+1} \\
&\leq \left\| v_F - \left[R_{\mathcal{V},CF}^T - \hat{A}_{FF}^{-1}\left(A_{FF}R_{\mathcal{V},CF}^T + A_{FC}\right)\right] v_c \right\|_{\hat{A}_{FF}} \cdot \left(R_{\mathcal{W}^l} B_F \hat{A}_{FF}^{-1} B_F^T R_{\mathcal{W}^l}^T\right)^{1/2}
\end{aligned}$$

Together with (4.143) we have

$$\left\| v_F - \left[R_{\mathcal{V},CF}^T - \hat{A}_{FF}^{-1}\left(A_{FF}R_{\mathcal{V},CF}^T + A_{FC}\right)\right] v_C \right\|_{\hat{A}_{FF}} \leq \sqrt{\beta_5}\|v_l\|_{A_l}.$$

From $\rho(\hat{A}_l^{-1/2} A_l \hat{A}_l^{-1/2}) < 1$ we also have $\rho(\hat{A}_{FF}^{-1/2} A_{FF} \hat{A}_{FF}^{-1/2}) < 1$. Hence we can conclude

$$R_{\mathcal{W}^l} B_F \hat{A}_{FF}^{-1} B_F^T R_{\mathcal{W}^l}^T < R_{\mathcal{W}^l}\left(C_l + 2B_F\hat{A}_{FF}^{-1}B_F^T - B_F\hat{A}_{FF}^{-1}A_{FF}\hat{A}_{FF}^{-1}B_F^T\right) R_{\mathcal{W}^l}^T = C_{l+1}$$

so that we can estimate the nominator of the rightmost term in (4.148),

$$v_l^T \left[B_l^T R_{\mathcal{W}^l}^T - \Pi_{\mathcal{V}^l}^T B_{l+1}^T\right] p_{l+1} < \sqrt{\beta_5}\|v_l\|_{A_l}\left(p_{l+1}^T C_{l+1} p_{l+1}\right)^{1/2}. \tag{4.149}$$

Finally, we perform the induction as in the proof of the previous lemma,

$$\begin{aligned}
&\sup_{0\neq v_{l+1}\in\mathcal{V}^{l+1}} \frac{v_{l+1}^T B_{l+1}^T p_{l+1}}{\|v_{l+1}\|_{A_{l+1}}} \\
\overset{(4.147)}{\geq}\ & \sup_{0\neq v_l\in\mathcal{V}_l} \frac{v_l^T \Pi_{\mathcal{V}^l}^T B_{l+1}^T p_{l+1}}{\sqrt{\beta_6}\|v_l\|_{A_l}} \\
\overset{(4.149)}{\geq}\ & \frac{1}{\sqrt{\beta_6}}\left[\sup_{0\neq v_l\in\mathcal{V}_l} \frac{v_l^T B_l^T R_{\mathcal{W}^l}^T p_{l+1}}{\|v_l\|_{A_l}} - \sqrt{\beta_5}\left(p_{l+1}^T C_{l+1} p_{l+1}\right)^{1/2}\right] \\
\overset{(4.145)}{\geq}\ & \frac{1}{\sqrt{\beta_6}}\left[c_l\|p_{l+1}\|_{\mathrm{M}_{l+1}} - d_l\left(p_{l+1}^T R_{\mathcal{W}^l} C_l R_{\mathcal{W}^l}^T p_{l+1}\right)^{1/2} - \sqrt{\beta_5}\left(p_{l+1}^T C_{l+1} p_{l+1}\right)^{1/2}\right] \\
\geq\ & \frac{1}{\sqrt{\beta_6}}\left[c_l\|p_{l+1}\|_{\mathrm{M}_{l+1}} - \tilde{d}\left(p_{l+1}^T C_{l+1} p_{l+1}\right)^{1/2}\right]
\end{aligned}$$

where $\tilde{d} = \max(d_l, \sqrt{\beta_5})$.
From the last line, we obtain the constants

$$c_{l+1} = \frac{1}{\sqrt{\beta_6}} c_l \text{ and } d_{l+1} = \frac{1}{\sqrt{\beta_6}} \max(d_l, \sqrt{\beta_5}). \qquad \square$$

To show inequality (4.143), note that in our implementation we have $\hat{A}_{FF} = \alpha D_{FF}$, where D_{FF} denotes the part of the diagonal of A_l corresponding to the fine grid variables and $\alpha > 1$ is chosen such that $\hat{A}_l > A_l$. Let us assume that A_l is weakly diagonally dominant and that our tentative interpolation operator $R^T_{\mathcal{V},CF}$ satisfies (2.37),

$$\|v_F - R^T_{\mathcal{V},FC} v_C\|^2_{D_{FF}} \le \tau \|v_l\|^2_{A_l} \tag{4.150}$$

where $\tau > 0$ is independent of V_l.
We follow the proof of Theorem 2.14. First, we replace $\hat{A}_{FF} = \alpha D_{FF}$,

$$\begin{aligned}
&\left\| v_F - \left[R^T_{\mathcal{V},CF} - \hat{A}^{-1}_{FF} \left(A_{FF} R^T_{\mathcal{V},CF} + A_{FC} \right) \right] v_C \right\|^2_{\hat{A}_{FF}} \\
&= \left\| \left(v_F - R^T_{\mathcal{V},CF} v_C \right) + \frac{1}{\alpha} D^{-1}_{FF} \left(A_{FF} R^T_{\mathcal{V},CF} + A_{FC} \right) v_C \right\|^2_{\alpha D_{FF}} \\
&= \left(v_F - R^T_{\mathcal{V},CF} v_C \right)^T \alpha D_{FF} \left(v_F - R^T_{\mathcal{V}l} v_C \right) \\
&\quad + \left(A_{FF} R^T_{\mathcal{V},CF} v_C + A_{FC} v_C \right)^T \frac{1}{\alpha} D^{-1}_{FF} \left(A_{FF} R^T_{\mathcal{V},CF} v_C + A_{FC} v_C \right) \\
&\quad + 2 \left(v_F - R^T_{\mathcal{V},CF} v_C \right)^T D_{FF} D^{-1}_{FF} \left(A_{FF} R^T_{\mathcal{V},CF} v_C + A_{FC} v_C \right) \\
&\overset{\alpha>1}{\le} \alpha \left\| v_F - R^T_{\mathcal{V},CF} v_C \right\|^2_{D_{FF}} + \left\| A_{FF} R^T_{\mathcal{V},CF} v_C + A_{FC} v_C \right\|^2_{D^{-1}_{FF}} \\
&\quad + 2 \left\| v_F - R^T_{\mathcal{V},CF} v_C \right\|_{D_{FF}} \left\| A_{FF} R^T_{\mathcal{V},CF} v_C + A_{FC} v_C \right\|_{D^{-1}_{FF}} .
\end{aligned} \tag{4.151}$$

The second summand and the second factor in the last line of (4.151) are further estimated, (cf. (2.64))

$$\|A_{FF} R^T_{\mathcal{V},CF} v_C + A_{FC} v_C\|_{D^{-1}_{FF}} \le \|A_{FF} \left(R^T_{\mathcal{V},CF} v_C - v_F \right)\|_{D^{-1}_{FF}} + \|A_{FF} v_F + A_{FC} v_C\|_{D^{-1}_{FF}} . \tag{4.152}$$

Regarding the first summand of (4.152), we have (cf. (2.25))

$$\|A_{FF}(R^T_{\mathcal{V},CF} v_C - v_F)\|_{D^{-1}_{FF}} \le \rho(D^{-1}_{FF} A_{FF}) \|v_F - R^T_{\mathcal{V},CF} v_C\|_{D_{FF}} \le \sqrt{\tau} \rho(D^{-1}_{FF} A_{FF}) \|v_l\|_{A_l}.$$

The second summand in (4.152) is estimated by

$$\begin{aligned}
\|A_{FF} v_F + A_{FC} v_C\|^2_{D^{-1}_{FF}} &= \begin{pmatrix} A_{FF} v_F + A_{FC} v_C \\ A_{CF} v_F + A_{CC} v_C \end{pmatrix}^T \begin{pmatrix} D^{-1}_{FF} & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} A_{FF} v_F + A_{FC} v_C \\ A_{CF} v_F + A_{CC} v_C \end{pmatrix} \\
&\le \begin{pmatrix} A_{FF} v_F + A_{FC} v_C \\ A_{CF} v_F + A_{CC} v_C \end{pmatrix}^T \begin{pmatrix} D^{-1}_{FF} & 0 \\ 0 & D^{-1}_{CC} \end{pmatrix} \begin{pmatrix} A_{FF} v_F + A_{FC} v_C \\ A_{CF} v_F + A_{CC} v_C \end{pmatrix} \\
&= \|v_l\|^2_{A_l D_l^{-1} A_l} \\
&\le \rho(D^{-1} A) \|v_l\|^2_{A_l}.
\end{aligned}$$

We use $\rho(D^{-1}_{FF} A_{FF}) \le \rho(D_l^{-1} A_l) \le 2$ for weakly diagonally dominant matrices A_l and obtain (cf. (2.65))

$$\|A_{FF} R^T_{\mathcal{V},CF} v_C + A_{FC} v_C\|_{D^{-1}_{FF}} \le (2\sqrt{\tau} + \sqrt{2}) \|v\|_{A_l}.$$

We insert this result into (4.151) and conclude that we have (4.143), where

$$\beta_5 = \alpha\tau + (2\sqrt{\tau} + \sqrt{2})^2 + 2\sqrt{\tau}(2\sqrt{\tau} + \sqrt{2}).$$

In the case of point-based AMG with $\hat{A}_l = D_P$ (2.92) (see Remark 4.6), we replace D_{FF} by the part of D_P belonging to the fine grid points and $\| \cdot \|_{0,F}$ by $\| \cdot \|_{P,0,F}$ and employ

$$\|v_F - P_{FC}v_C\|^2_{P,0,F} \leq \tau\|v\|^2_1,$$

instead of (4.150).
Note that the constant β_5 depends on α so a smaller (more accurately computed) value of α allows a smaller β_5, and, in consequence, a smaller d_{l+1}. Likewise, an accurate interpolation operator $R^T_{\mathcal{V}l}$ (where τ in (4.150) is small) also leads to a small d_{l+1}.
Regarding inequality (4.144), we refer to the discussion in Section 4.8. Again, if A_l is singular, we replace the norm $\|\cdot\|_{A_l}$ by a suitable norm $\|\cdot\|_{A_{D_l}}$, where A_{D_l} is non-singular and $A_{D_l} > A_l$.

4.11. Setup of the AMG hierarchy for saddle point systems

Now, we have all components ready to set up the AMG hierarchy for saddle point problems. In algorithm 4.5 we sketch the setup algorithm for saddle point AMG. First, if the decomposition of Ω into velocity and pressure unknowns is not given by the user, we automatically detect these subsets according to the respective diagonal entry of $\mathcal{K}$. Then, on each level we first set up the smoother (Sections 4.5 and 4.6) and compute the Schur complement $T = B\hat{A}^{-1}B^T + C$. After this is done, we carry out the classical AMG coarsening and interpolation algorithms as in the case of scalar AMG, cf. Algorithm 2.3, this time applied to both the velocity and pressure components. Finally, we compute the stabilized interpolation by (4.99), (4.124), or (4.136) and the coarse grid operator $\mathcal{K}_{l+1} \leftarrow \mathcal{R}^T_l\mathcal{K}_l\mathcal{P}_l$.
Note that in most cases the velocity space $\mathcal{V}$ itself consists of multiple physical unknowns. Hence, for the coarsening and interpolation for the matrix block A we have to choose whether we employ variable-based AMG (Section 2.12.1), unknown-based AMG (Section 2.12.2), or point-based AMG (Section 2.12.3). In the latter case, we also have the option to use the same coarse mesh for both velocity and pressure variables if the initial discretization on level 1 also employs a common mesh (e.g. P_1-P_1-stab finite elements).

algorithm 4.5 AmgSaddleSetup$(\Omega, \mathcal{K} = (\mathbf{k}_{ij})_{i,j}, N_{min}, L_{max}, L, \{\mathcal{K}_l\}_{l=1}^{L}, \{P_l\}_{l=1}^{L-1}, \{R_l\}_{l=1}^{L-1})$

```
begin
  $\Omega_u^1 \leftarrow \{j \in \Omega : \mathbf{k}_{jj} > 0\}$;  $\Omega_p^1 \leftarrow \Omega \setminus \Omega_u^1$;        or partitioning can be supplied by user
  $N_1 \leftarrow |\Omega_u^1|$;  $M_1 \leftarrow |\Omega_p^1|$;
  $\mathcal{K}_1 \leftarrow \mathcal{K}$;
  for $l \leftarrow 1$ to $L_{\max} - 1$ do
      $A \leftarrow (\mathbf{k}_{i,j}^l)_{i,j \in \Omega_u^l}$;  $B \leftarrow (\mathbf{k}_{i,j}^l)_{i \in \Omega_p^l, j \in \Omega_u^l}$;  $C \leftarrow (\mathbf{k}_{i,j}^l)_{i,j \in \Omega_p^l}$;
      VankaSmootherSetup$(A, B, C, \{\hat{A}_j\}_{j=1}^{M_l}, \{B_j\}_{j=1}^{M_l}, \{\hat{S}_j\}_{j=1}^{M_l}, \{V_j\}_{j=1}^{M_l}, \{W_j\}_{j=1}^{M_l})$
      compute $T \leftarrow B\hat{A}^{-1}B^T + C$;
      AmgStrongCouplings$(A, S_u, S_u^T)$;
      AmgStrongCouplings$(T, S_p, S_p^T)$;
      split $\Omega_u^l$ into $C_u^l \dot{\cup} F_u^l$ using$S_u$;                    see Section 2.6 or Chapter 3
      split $\Omega_p^l$ into $C_p^l \dot{\cup} F_p^l$ using$S_p$;
      set $\Omega_u^{l+1} \leftarrow C_u^l$;  $N_{l+1} = |\Omega_u^{l+1}|$;
      set $\Omega_p^{l+1} \leftarrow C_p^l$;  $M_{l+1} = |\Omega_p^{l+1}|$;
      build velocity interpolation $R_{\mathcal{V}^l}^T : \mathbb{R}^{N_{l+1}} \to \mathbb{R}^{N_l}$ using $A$;        see Section 2.8
      build pressure interpolation $R_{\mathcal{W}^l}^T : \mathbb{R}^{M_{l+1}} \to \mathbb{R}^{M_l}$ using $T$;
      compute $\mathcal{P}_l$ according to (4.99), (4.124), or (4.136);
      if used (4.99)
        then
            $\mathcal{R}_l \leftarrow \begin{pmatrix} R_{\mathcal{V}^l} & 0 \\ 0 & R_{\mathcal{W}^l} \end{pmatrix}$;
        else
            $\mathcal{R}_l \leftarrow \mathcal{P}_l^T$;
      fi;
      compute $\mathcal{K}_{l+1} \leftarrow \mathcal{R}_l^T \mathcal{K}_l \mathcal{P}_l$;
      if $|\Omega_u^{l+1} \cup \Omega_p^{l+1}| \leq N_{min}$ then break;  fi;
  od;
  $L \leftarrow l + 1$;
end.
```

5. Numerical Results

In this chapter, we demonstrate the practicability of our saddle point AMG. We investigate the performance of our AMG using two finite difference model problems and one finite element example.
We have implemented our saddle point AMG using the *hypre* software package [hyp] [CCF98]. An important component of this parallel linear solver suite is the *Boomer*AMG algebraic multigrid solver and preconditioner for positive definite matrices. The ingredients of *Boomer*AMG include smoothers (Jacobi, Gauss–Seidel, SOR, polynomial), parallel coarse grid generation techniques (third pass coarsening, CLJP, Falgout's scheme, PMIS, HMIS, CGC(-E), compatible relaxation, ...), interpolation setup routines (direct, modified classical, extended(+i), Jacobi, and may more). Furthermore, for systems of elliptic PDEs both the unknown-based (UAMG) and the point-based (PAMG) approach are supported. For the latter, block smoothers and block interpolation routines can be chosen.
For the numerical experiments presented in this chapter, we have set the following parameters, if not stated otherwise:

- The strength threshold (2.29) is $\alpha = 0.25$.
- We perform both phases of Ruge-Stüben coarsening (Algorithms 2.5 and 2.6).
- As tentative interpolation for both velocity and pressure, we employ modified classical interpolation (2.51) per physical unknown. The interpolation is not truncated.
- We stop the coarsening if the number of degrees of freedom is less than 1000. A direct solver is used on the coarsest level.

In addition to the wall clock time consumed for the setup (Algorithm 2.3) and solve (Algorithm 2.2) routines, we give two further quantities:

- The *operator complexity*
$$C_A := \frac{\sum_{l=1}^{L_{max}} \textsf{non-zeros}\,(\mathcal{K}_l)}{\textsf{non-zeros}\,(\mathcal{K}_1)}$$
gives an indication of the "memory overhead" required for the AMG hierarchy compared to the original linear system.
- The *convergence factor* or *convergence rate*
$$\rho = \left(\frac{r^{it}}{r^0}\right)^{\frac{1}{it}},$$

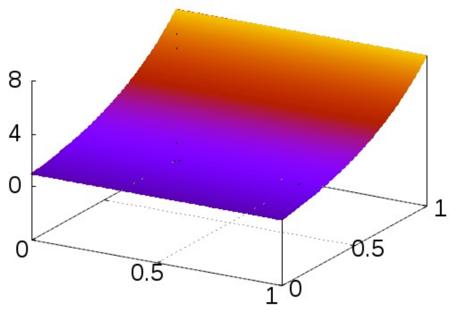


Figure 5.1.: Diffusion coefficient for the SOLKY problem (5.1)–(5.2)

where $r^{it} := |y - \mathcal{K}x^{it}|$ is the l_2-norm of the residual.

If the right hand side y is zero, we initialize x^0 with random values and scale it such that $|x_0| = 1$. We stop the iteration if the residual norm r^{it} is less than 10^{-8}. Otherwise, we use a zero start vector x^0 and terminate if $r^{it} < 10^{-8} r^0$.

5.1. Finite difference examples

We start with two geodynamic benchmark examples in two spatial dimensions, the SOLKY and the SINKER problem. In both cases, we use a staggered grid (Section 4.2.1) to discretize the respective PDE on a square domain $\Omega = [0,1]^2$. We impose a Neumann boundary condition (free outflow) for $x = 1$, while on all other boundary we set zero Dirichlet conditions.

Example 5.1 SOLKY problem The first example is a variant of the SOLKY problem [MM08]. We solve the equations

$$-\nabla \cdot \nu \nabla \mathsf{u} + \nabla \mathsf{p} = 0 \tag{5.1}$$

$$\nabla \cdot \mathsf{u} = 0 \tag{5.2}$$

in $\Omega = (0,1)^2$, where ν is given by

$$\nu(x,y) = \exp(2y),$$

see Figure 5.1.

Example 5.2 SINKER problem Here, we consider a problem with a jumping diffusion coefficient $\nu(x,y)$. The equations are as in (5.1)–(5.2), but $\nu(x,y)$ is no longer continuous. Instead, for $0.5 \le x \le 0.75$ and $0.5 \le y \le 0.75$, we have $\nu(x,y) = \nu_1$, while for the remainder of the domain we set $\nu(x,y) = \nu_0 = 1$, see Figure 5.2. In our

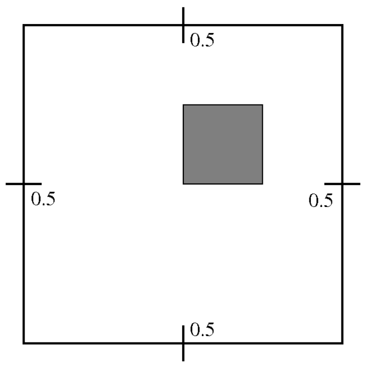


Figure 5.2.: Computational domain for the SINKER problem. The gray square indicates the area where the diffusion coefficient ν_1, while the in the remainder of the domain the diffusion coefficient ν_0 equals 1.

experiments, we let $\nu_1 = 10^{-6}$, 10^{-3}, 1, 10^3, or 10^6. In the case of $\nu_1 = 1$, we obtain a Stokes problem.

All finite difference experiments were carried out on dual processor Intel Xeon 3.20 GHz machines with 6GB RAM.

Two-level experiments In Sections 4.8–4.10, we have introduced three different stabilization techniques for the coarse saddle point system,

- a full smoothing of the prolongation matrix $\mathcal{P} := \mathcal{M}\mathcal{R}^T$ and a Petrov-Galerkin approach to the coarse grid operator, $\mathcal{K}^C := \mathcal{R}\mathcal{K}\mathcal{P}$ (4.99);
- an application of the pressure–to–velocity coupling operator $\begin{pmatrix} I & -\hat{A}_l^{-1}B_l^T \\ 0 & I \end{pmatrix}$ (4.124) to the tentative prolongation and a Galerkin coarse grid ansatz,
- "F-stabilization": as in the previous case, but restricted to the fine grid velocity variables only (4.136)

$$\mathcal{P}_l = \begin{pmatrix} I_{FF} & 0 & -\hat{A}_{FF}^{-1}B_F^T \\ 0 & I_{CC} & 0 \\ 0 & 0 & I \end{pmatrix} \begin{pmatrix} R_{\mathcal{V},CF}^T & 0 \\ I_{CC} & 0 \\ 0 & R_{\mathcal{W}}^T \end{pmatrix}.$$

For the first variant, we were able to prove two-grid convergence if used in combination with a single pre-smoothing step with the inexact Uzawa method 4.67 (see Section 4.8), but this involves a substantial memory usage. The question arises whether the other approaches, especially F-stabilization, can also provide a converging method. To this end, we numerically compare the three stabilization techniques as well as a unstabilized coarse grid operator (where we just employ the block interpolation operators $\begin{pmatrix} R_{\mathcal{V}}^T & 0 \\ 0 & R_{\mathcal{W}}^T \end{pmatrix}$). We consider three different mesh refinements: A 32×32 mesh (3,040 degrees of freedom),

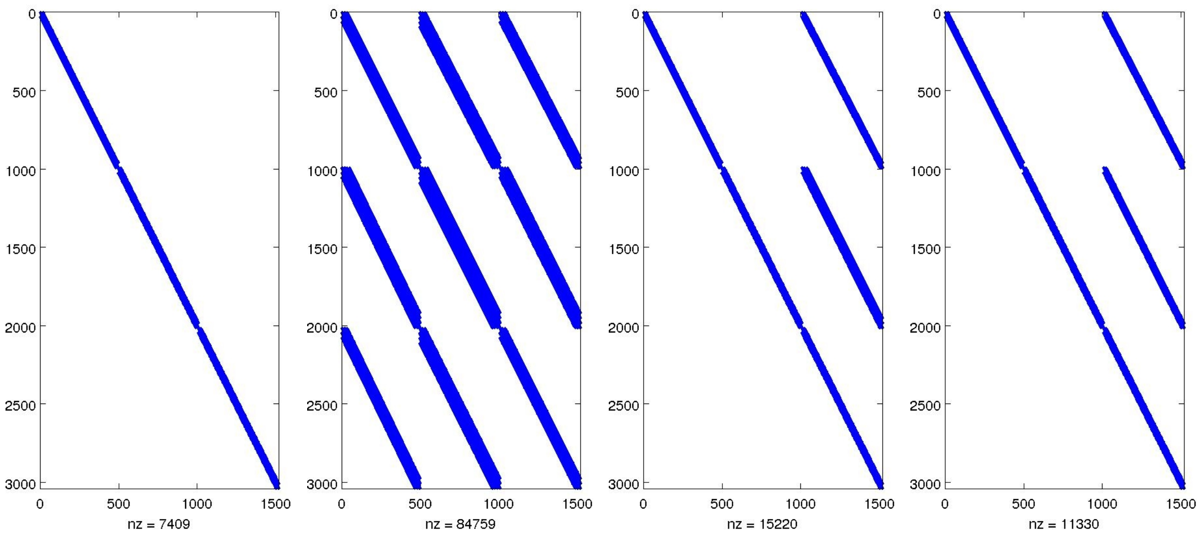


Figure 5.3.: Sparsity patterns of the tentative and stabilized interpolation operators on the finest level for a Stokes problem discretized on a staggered finite difference mesh with 32×32 cells. From left to right: tentative interpolation operator, fully smoothed prolongation (4.99), pressure–to–velocity stabilization (4.124), F-stabilization (4.136)

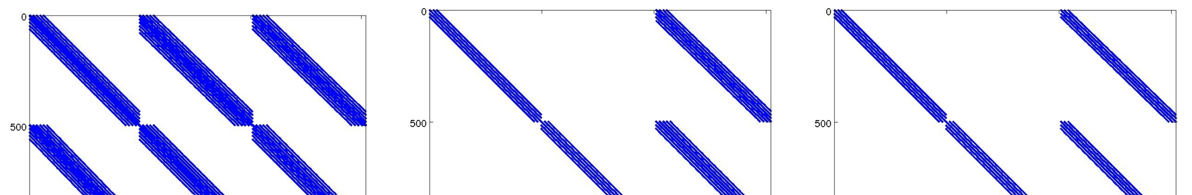


Figure 5.4.: Sparsity patterns of the coarse grid operator on the first coarse level for a Stokes problem discretized on a staggered finite difference mesh with 32×32 cells. From left to right: fully smoothed prolongation (4.99), pressure–to–velocity stabilization (4.124), F-stabilization (4.136)

		none		full		P2V		F	
Mesh	dof	C_A	ρ	C_A	ρ	C_A	ρ	C_A	ρ
32×32	3,040	2.48	0.40	7.00	0.41	3.80	0.41	2.69	0.42
64×64	12,224	2.50	0.41	7.26	0.42	3.89	0.42	2.72	0.43
128×128	49,024	2.51	0.42	7.39	0.42	3.93	0.42	2.74	0.43

Table 5.1.: Operator complexity C_A and convergence factor ρ for the two-grid iteration applied to the SOLKY problem. The different stabilization techniques are denoted by none (no stabilization), full (full prolongation smoothing and Petrov-Galerkin coarse grid operator (4.99)), P2V (pressure–to–velocity coupling (4.124)) and F (F-stabilization (4.136)).

a 64×64 mesh (12, 224 degrees of freedom), and a 128×128 mesh (49, 024 degrees of freedom).
In Figure 5.3 we show the sparsity patterns and numbers of non-zero matrix entries for a tentative interpolation operator $\mathcal{R}_1$ on the finest level (Stokes problem discretized on 32×32 cells) as well as the stabilized interpolation operator $\mathcal{P}_1$ computed according to (4.99), (4.124), and (4.136), respectively. We see that the full stabilization technique produces a significantly denser interpolation operator than the other variants. Also, this matrix includes couplings between the different velocity components that are not present in the tentative prolongation and the sparser stabilization variants. All these drawbacks carry over to the first coarse grid operator, see Figure 5.4. In Table 5.1 we give the operator complexity as well as the convergence factor for the SOLKY problem. We see that all convergence rates are nearly equal, while the operator complexity differs significantly depending on the stabilization technique chosen.
While for the SOLKY problem the two-level method without stabilization still converged, this is not longer true for the SINKER problem if the diffusion coefficient has a jump, see Table 5.2 (right). We see however that the robustness of the stabilized methods does not depend on the diffusion coefficients. Again, the the F-stabilization approach involves an operator complexity that is only about 10 percent higher than the unstabilized two-grid hierarchy, in contrast to the other stabilization techniques that have a significantly higher memory overhead.

Multilevel experiments In the following, we turn our attention from the two-grid iteration with one additive pre-smoothing step 4.67 to full V-cycles. We compare $V(1,1)$, $V(3,3)$, $V(5,5)$ with both additive, multiplicative and symmetric multiplicative smoothing iterations. We first give the setup time and the operator complexity for the algebraic multigrid hierarchies (Table 5.3). The full stabilization method already runs out of memory for very small problem sizes. Not surprising, the setup without stabilization is faster than the other variants and also has the least memory overhead. However, without stabilization we only have convergence for the smallest problem size 32×32 and all other cycles diverge. In the following, we give the convergence figures for the pressure–to–velocity coupling and F stabilization techniques.

			none		full		P2V		F	
Mesh	dof	ν_1	C_A	ρ	C_A	ρ	C_A	ρ	C_A	ρ
		$1e^{-6}$	2.46	–	6.97	0.41	3.79	0.41	2.68	0.41
		$1e^{-3}$	2.46	–	6.97	0.41	3.79	0.41	2.68	0.41
32×32	3040	1	2.48	0.41	6.99	0.41	3.80	0.41	2.69	0.41
		$1e^{3}$	2.46	–	6.95	0.41	3.77	0.42	2.68	0.42
		$1e^{6}$	2.46	–	6.95	0.41	3.77	0.42	2.68	0.42
		$1e^{-6}$	2.50	–	7.25	0.42	3.88	0.42	2.72	0.42
		$1e^{-3}$	2.50	–	7.25	0.42	3.88	0.42	2.72	0.42
64×64	12224	1	2.50	0.42	7.26	0.42	3.89	0.42	2.72	0.42
		$1e^{3}$	2.49	–	7.24	0.42	3.87	0.42	2.72	0.42
		$1e^{6}$	2.49	–	7.24	0.42	3.87	0.42	2.72	0.42
		$1e^{-6}$	2.51	–	7.39	0.42	3.93	0.42	2.73	0.42
		$1e^{-3}$	2.51	–	7.39	0.42	3.93	0.42	2.73	0.42
128×128	49024	1	2.51	0.42	7.39	0.42	3.93	0.42	2.74	0.42
		$1e^{3}$	2.51	–	7.39	0.42	3.92	0.42	2.73	0.42
		$1e^{6}$	2.51	–	7.39	0.42	3.92	0.42	2.73	0.42

Table 5.2.: Operator complexity C_A and convergence factor ρ for the two-grid iteration applied to the SINKER problem. A dash denotes that the method did not converge within 1000 iterations.

		none		full		P2V		F	
Mesh	dof	t_{setup}	C_A	t_{setup}	C_A	t_{setup}	C_A	t_{setup}	C_A
32×32	3,040	$4 \cdot 10^{-2}$	2.95	0.86	21.94	$6 \cdot 10^{-2}$	4.67	$4 \cdot 10^{-2}$	3.33
64×64	12,224	0.12	3.16	118.47	44.59	0.35	5.13	0.20	3.61
128×128	49,024	0.57	3.28	–	–	1.72	5.29	0.96	3.77
256×256	196,352	2.75	3.40	–	–	8.98	5.44	5.54	3.92
512×512	785,920	10.82	3.48	–	–	50.86	5.56	24.90	4.00
1024×1024	3,144,704	65.67	3.55	–	–	269.09	5.65	141.31	4.08

Table 5.3.: Setup time t_{setup} and operator complexity C_A for the AMG hierarchy computed for the SOLKY problem.

	additive			multiplicative			symmetric multiplicative		
Mesh	$V(1,1)$	$V(3,3)$	$V(5,5)$	$V(1,1)$	$V(3,3)$	$V(5,5)$	$V(1,1)$	$V(3,3)$	$V(5,5)$
32×32	0.27	0.15	0.16	0.47	0.24	0.32	0.34	0.37	0.50
64×64	–	3.00	1.38	–	4.03	2.26	–	2.23	3.27
128×128	–	24.13	9.76	–	34.38	13.57	–	12.60	18.79
256×256	–	–	45.54	–	–	75.56	–	87.36	100.51
512×512	–	–	252.74	–	–	715.22	–	534.19	695.39
1024×1024	–	–	1,009.49	–	–	5,269.60	–	2,155.71	2,958.88

Table 5.4.: Solution time for the V-cycle iteration applied to the SOLKY problem (Pressure–to–velocity coupling).

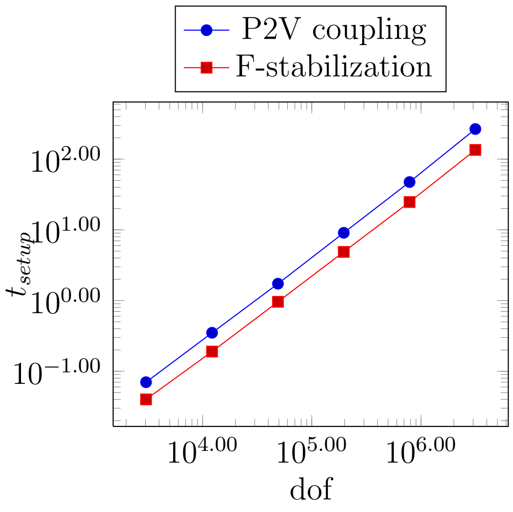
	additive			multiplicative			symmetric multiplicative		
Mesh	$V(1,1)$	$V(3,3)$	$V(5,5)$	$V(1,1)$	$V(3,3)$	$V(5,5)$	$V(1,1)$	$V(3,3)$	$V(5,5)$
32×32	0.56	0.06	0.03	0.57	0.06	0.03	0.28	0.03	0.02
64×64	–	0.39	0.04	–	0.32	0.04	–	0.04	0.03
128×128	–	0.57	0.07	–	0.51	0.07	–	0.04	0.03
256×256	–	–	0.11	–	–	0.09	–	0.10	0.03
512×512	–	–	0.18	–	–	0.24	–	0.14	0.06
1024×1024	–	–	0.31	–	–	0.45	–	0.10	0.08

Table 5.5.: Convergence factor for the V-cycle iteration applied to the SOLKY problem (Pressure–to–velocity coupling).

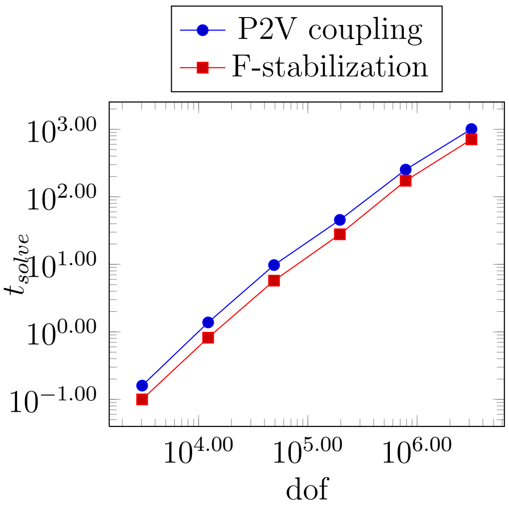
From Tables 5.4–5.7 we learn that the $V(1,1)$ is not sufficient to solve the problem except for small problem sizes. Furthermore, the (non-symmetric) multiplicative smoother has no advantage over the additive smoother. To obtain more robust convergence, we have to employ symmetric multiplicative smoothing, (see Tables 5.5 and 5.7). On the other hand, despite the fact that its convergence factors are not completely independent of the problem size, the $V(5,5)$ cycle with additive smoothing provides the fastest in nearly all cases, see Tables 5.4 and 5.6. We mention here that for the multiplicative and symmetric multiplicative smoothing methods we still had to employ the same scaling factors as in the case of additive smoothing, cf. the discussion in Section 4.6.

	additive			multiplicative			symmetric multiplicative		
Mesh	$V(1,1)$	$V(3,3)$	$V(5,5)$	$V(1,1)$	$V(3,3)$	$V(5,5)$	$V(1,1)$	$V(3,3)$	$V(5,5)$
32×32	0.36	$9 \cdot 10^{-2}$	0.10	0.29	0.15	0.19	0.13	0.19	0.27
64×64	–	0.75	0.82	–	0.94	1.31	1.35	1.26	1.86
128×128	–	4.38	5.74	–	5.15	7.61	9.28	7.04	9.82
256×256	–	49.87	27.87	–	53.12	39.90	–	34.66	60.34
512×512	–	1,748.63	173.15	–	–	233.39	–	204.31	271.25
1024×1024	–	–	710.60	–	–	1,506.32	–	1,462.72	1,786.08

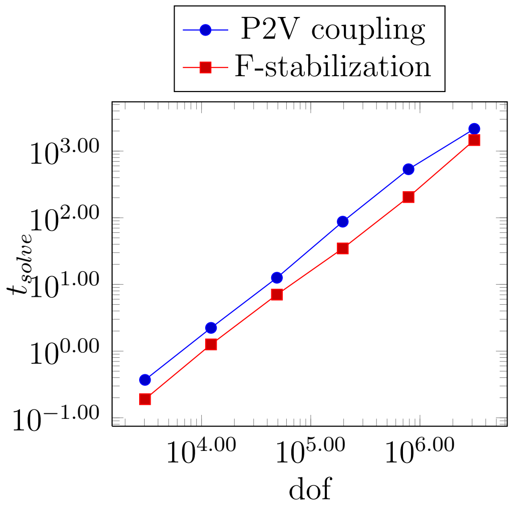
Table 5.6.: Solution time for the V-cycle iteration applied to the SOLKY problem (F-stabilization).



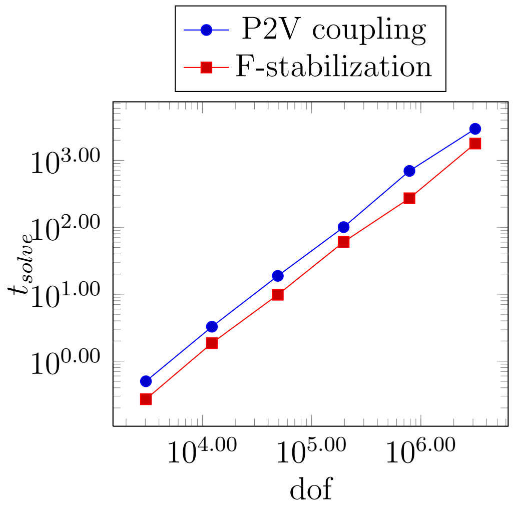
(a) Setup time t_{setup}



(b) Solution time t_{solve} for the $V(5,5)$ cycle with additive smoothing.



(c) Solution time t_{solve} for the $V(3,3)$ cycle with symmetric multiplicative smoothing.



(d) Solution time t_{solve} for the $V(5,5)$ cycle with symmetric multiplicative smoothing.

Figure 5.5.: Numerical Results for the SOLKY problem

	additive			multiplicative			symmetric multiplicative		
Mesh	$V(1,1)$	$V(3,3)$	$V(5,5)$	$V(1,1)$	$V(3,3)$	$V(5,5)$	$V(1,1)$	$V(3,3)$	$V(5,5)$
32×32	0.74	0.06	0.03	0.57	0.06	0.03	0.14	0.03	0.02
64×64	–	0.06	0.05	–	0.06	0.04	0.34	0.03	0.02
128×128	–	0.08	0.07	–	0.07	0.04	0.42	0.03	0.02
256×256	–	0.38	0.05	–	0.30	0.04	–	0.04	0.03
512×512	–	0.89	0.11	–	–	0.08	–	0.05	0.03
1024×1024	–	–	0.27	–	–	0.17	–	0.11	0.04

Table 5.7.: Convergence factor for the V-cycle iteration applied to the SOLKY problem (F-stabilization).

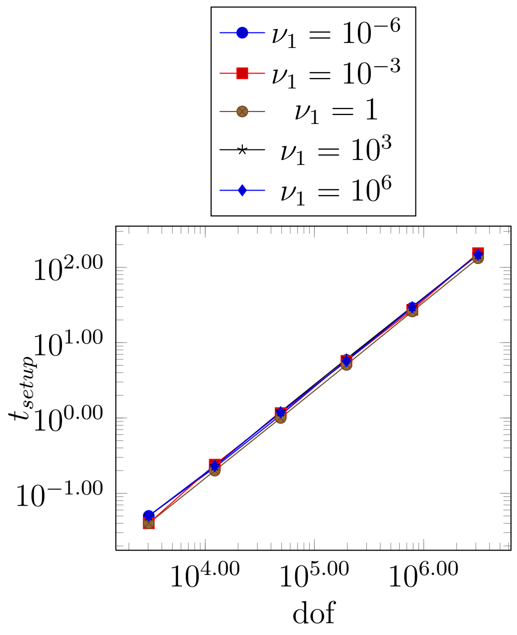
We already mentioned in Section 4.10 that in contrast to pressure-to–velocity coupling, the F-stabilization technique better fits into the context of classical AMG, with designated "coarse" and "fine" points. Comparing the numerical results in this section for both variants (Tables 5.3–5.7, Figure 5.1) we obtain a justification for this hypothesis: F-stabilization performs better than pressure–to–velocity coupling both in terms of memory overhead (operator complexity) as well as setup and solve timings. In the remainder of this chapter, we will only us F-stabilization.

In Table 5.8 and Figure 5.6 we give the numerical results of the F-stabilization technique applied within an AMG hierarchy computed for the SINKER problem. In contrast to the two-level method, we now see that all relevant figures (setup time, operator complexity, solution time, convergence factor) now depend on the coefficient ν_1. Not surprising, we obtain the best results for the plain Stokes problem ($\nu_1 = 1$). It is also clear that there is no "ideal" smoothing parameter set: In most cases, a $V(5,5)$ cycle with additive smoothing yields the fastest solution (even if the convergence factors for symmetric multiplicative smoothing are lower), but for the 1024×1024 mesh and $\nu_1 = \pm 10^6$, it is not sufficient any more and symmetric multiplicative smoothing is required.

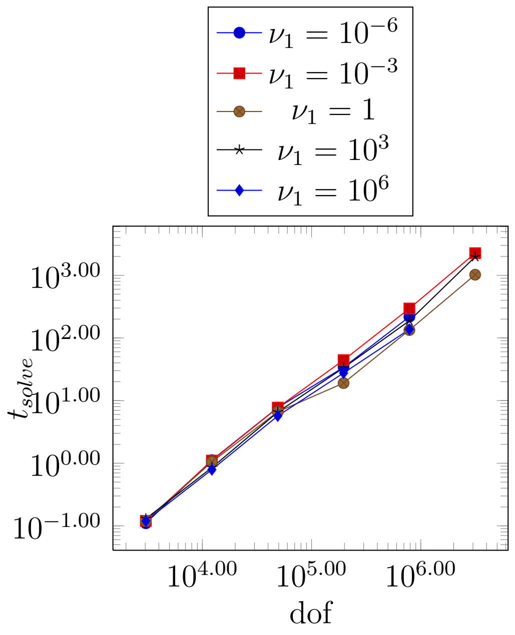
We omitted the $V(3,3)$ cycle with additive smoothing from the table, as this iteration did not converge for any mesh size larger than 128×128 if $\nu_1 \neq 1$.

					additive		symmetric multiplicative			
					$V(5,5)$		$V(3,3)$		$V(5,5)$	
Mesh	dof	ν_1	t_{setup}	C_A	t_{solve}	ρ	t_{solve}	ρ	t_{solve}	ρ
32×32	3040	10^{-6}	0.05	3.62	0.11	0.05	0.19	0.04	0.27	0.03
		10^{-3}	0.04	3.62	0.12	0.05	0.19	0.04	0.27	0.03
		1	0.04	3.30	0.12	0.05	0.19	0.04	0.30	0.03
		10^{3}	0.05	3.62	0.13	0.05	0.21	0.05	0.30	0.03
		10^{6}	0.05	3.62	0.12	0.05	0.21	0.05	0.31	0.03
64×64	12224	10^{-6}	0.22	3.89	1.12	0.05	1.41	0.05	2.03	0.03
		10^{-3}	0.24	3.89	1.10	0.05	1.44	0.05	2.10	0.03
		1	0.20	3.61	1.05	0.06	1.27	0.05	1.86	0.03
		10^{3}	0.23	4.02	0.85	0.06	1.62	0.06	2.38	0.05
		10^{6}	0.23	4.02	0.79	0.06	1.61	0.06	2.37	0.05
128×128	49024	10^{-6}	1.09	4.03	7.73	0.10	8.62	0.06	12.02	0.03
		10^{-3}	1.16	4.03	7.72	0.10	7.73	0.05	11.94	0.03
		1	1.00	3.78	6.61	0.07	7.14	0.05	10.77	0.03
		10^{3}	1.21	4.16	6.55	0.06	9.32	0.06	15.37	0.06
		10^{6}	1.19	4.14	5.69	0.07	9.35	0.06	12.75	0.04
256×256	196352	10^{-6}	5.88	4.12	33.96	0.16	55.29	0.11	70.42	0.06
		10^{-3}	5.73	4.10	44.24	0.27	60.60	0.15	68.75	0.06
		1	5.07	3.92	19.01	0.07	33.31	0.04	51.52	0.03
		10^{3}	6.09	4.17	33.34	0.09	51.53	0.09	89.82	0.08
		10^{6}	5.68	4.16	27.57	0.10	48.72	0.08	73.88	0.06
512×512	785920	10^{-6}	28.89	4.17	217.88	0.30	386.00	0.21	475.49	0.08
		10^{-3}	27.26	4.17	294.90	0.32	531.14	0.28	494.96	0.13
		1	25.95	4.01	133.49	0.14	211.38	0.08	320.88	0.03
		10^{3}	30.04	4.15	187.91	0.23	263.96	0.07	411.32	0.06
		10^{6}	29.43	4.15	137.05	0.15	260.47	0.08	399.46	0.07
1024×1024	3144704	10^{-6}	152.69	4.24	–	–	7,667.65	0.66	4,937.48	0.34
		10^{-3}	154.05	4.25	2,254.40	0.60	4,036.61	0.45	3,548.96	0.23
		1	132.01	4.09	1,020.69	0.42	2,001.59	0.23	1,659.03	0.05
		10^{3}	147.36	4.19	1,969.40	0.59	2,050.04	0.21	2,257.60	0.09
		10^{6}	147.20	4.20	–	–	3,594.67	0.41	2,694.19	0.13

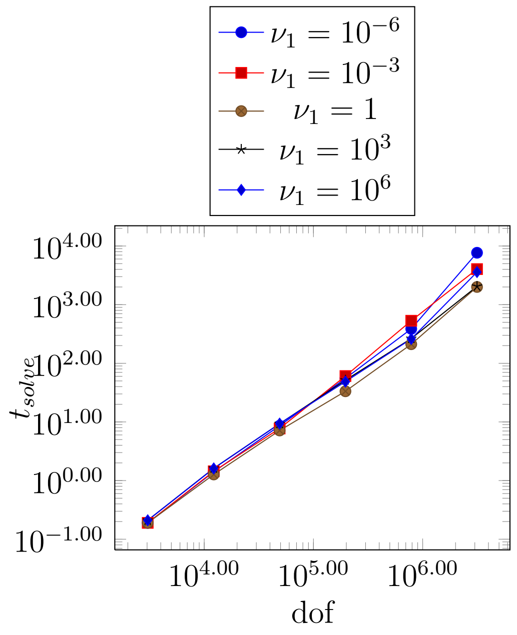
Table 5.8.: Numerical results for the SINKER problem. We show the mesh dimension, the degrees of freedom, the AMG setup time t_{setup}, the operator complexity C_A and the AMG solve time t_{solve} as well as the convergence factor ρ for $V(3,3)$ and $V(5,5)$ cycles with additive and symmetric multiplicative smoothing.



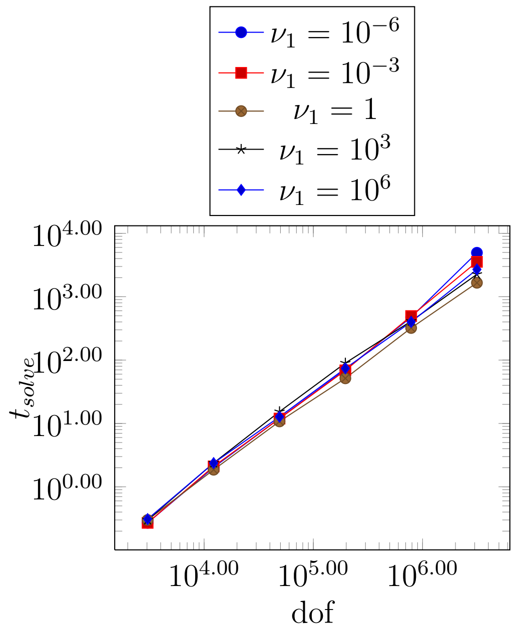
(a) Setup time t_{setup}



(b) Solve time t_{solve} for the $V(5,5)$ cycle with additive smoothing



(c) Solve time t_{solve} for the $V(3,3)$ cycle with symmetric multiplicative smoothing



(d) Solve time t_{solve} for the $V(5,5)$ cycle with symmetric multiplicative smoothing

Figure 5.6.: Numerical results for the SINKER problem

5.2. Mantle Convection

We conclude our numerical experiments with an example from earth mantle convection simulation. We use our saddle point AMG as linear solver inside the parallel adaptive-mesh mantle convection code Rhea [BGG+08, BSA+13]. In long timescales (millions of years) the earth mantle is assumed to behave as a viscous fluid, whose (dimension-less) velocity u, viscosity $\mu(\mathsf{T}, \mathsf{u})$ pressure p and temperature T are described by

$$\begin{aligned} -\nabla \left[\mu(\mathsf{T}, \mathsf{u}) \left(\nabla \mathsf{u} + \nabla \mathsf{u}^T\right)\right] + \nabla \mathsf{p} &= Ra \mathsf{T} e_r, && (5.3)\\ \nabla \cdot \mathsf{u} &= 0, && (5.4)\\ \frac{\partial \mathsf{T}}{\partial t} + \mathsf{u} \cdot \mathsf{T} - \nabla^2 \mathsf{T} &= \gamma. && (5.5) \end{aligned}$$

where e_r is the unit vector in radial direction, γ denotes the rate of internal heat generation, and the Rayleigh number is given by

$$Ra = \frac{\alpha \rho_0 g \Delta \mathsf{T} (D R_0)^3}{\kappa \mu_0}$$

where α, ρ_0, μ_0 and κ are the reference constants for thermal expansion, density, viscosity, and thermal diffusivity, respectively; the temperature difference across a mantle with relative thickness D is denoted by $\Delta \mathsf{T}$, and g is the gravitational acceleration. Here, we have $D = 0.45$ and the earth radius is given by $R_0 = 6371$km.
A streamline-upwind Petrov-Galerkin formulation is used to discretize (5.5), see [BSA+13], Section 3.4 for details. This leads to a system of ordinary differential equations, which is integrated in time using an explicit first order scheme. Inside each time step, a solution of the nonlinear Stokes system (5.3)–(5.4) must be computed. To this end, a Picard iteration is used to treat the non-linearity induced by the viscosity $\mu(\mathsf{T}, \mathsf{u})$. The linearized version of (5.3)–(5.4) then reads in weak form:
Find u and p such that

$$\begin{aligned} \int_\Omega \frac{\mu}{2} \left(\nabla \mathsf{u} + \nabla \mathsf{u}^T\right) : \left(\nabla \mathsf{v} + \nabla \mathsf{v}^T\right) - \int_\Omega (\nabla \cdot \mathsf{v})\, p & \\ + \int_{\partial\Omega} \left[\left(\mathsf{p} I - \mu \left(\nabla \mathsf{u} + \mathsf{u}^T\right)\right) n\right] \cdot v = \int_\Omega \mathsf{f} \cdot \mathsf{v} & \qquad \text{for all } \mathsf{v}, \qquad (5.6)\\ - \int_\Omega (\nabla \cdot \mathsf{u})\, q = 0 & \qquad \text{for all } \mathsf{q}, \qquad (5.7) \end{aligned}$$

where $\mathsf{f} = Ra \mathsf{T} e_r$. On the boundary $\partial\Omega$, free-slip boundary conditions are imposed, i.e.

$$\mathsf{u} \cdot n = 0, \qquad \mathsf{v} \cdot n = 0, \qquad s \cdot \left[\left(\mathsf{p} I - \mu \left(\nabla \mathsf{u} + \mathsf{u}^T\right)\right) n\right] = 0,$$

where n denotes an outer normal vector and s any tangential vector. Hence, inside (5.6) the boundary integral term $\int_{\partial\Omega} \left[\left(\mathsf{p} I - \mu \left(\nabla \mathsf{u} + \mathsf{u}^T\right)\right) n\right] \cdot v$ vanishes. Finally, we discretize (5.6)–(5.7) on a (deformed) hexahedral finite element mesh using trilinear

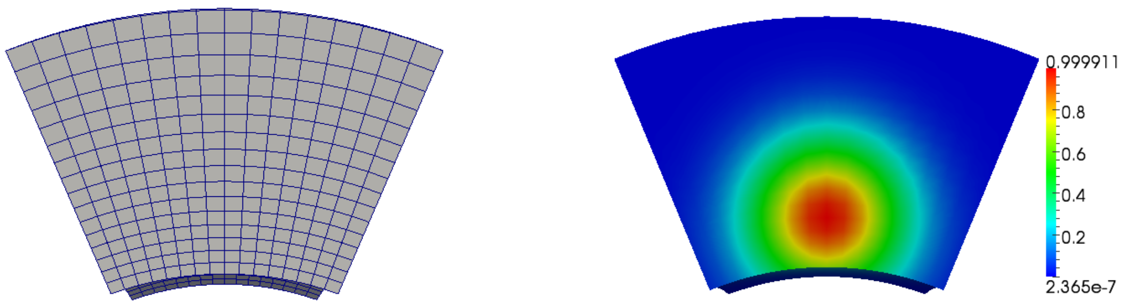


Figure 5.7.: Finite element mesh (left) and temperature distribution (right) on a $45° \times 45°$ section of the Earth mantle.

functions $\{\varphi_i\}_i$ for all velocity components as well as the pressure. We obtain a linear system of the form (4.1)

$$\begin{pmatrix} A & B^T \\ B & -C \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} f \\ 0 \end{pmatrix} \tag{5.8}$$

where the components of the matrix C are given by (4.42)

$$c_{ij} := \int_\Omega (\phi_i - \Pi\phi_i, \phi_j - \Pi\phi_j),$$

i.e. we have a $Q_1 - Q_1$1-stab discretization, for which the stability is shown in Theorem 4.5.
In the following, we focus on the solution of (5.8).

More precisely, we consider a $45° \times 45°$ section of a spherical shell, which is axially symmetrical arranged around the x-axis, see Figure 5.7. We have re-scaled the geometry such that the outer radius of the shell is given by $R_2 = 1$, the inner radius is $R_1 = 0.55$. We impose a temperature distribution given by

$$\begin{aligned} \mathsf{T} = \min(1, &\exp(-50((x-0.64)^2 + y^2 + z^2)) \\ &+ \exp(-50(x^2 + (y-0.64)^2 + z^2)) \\ &+ \exp(-50(x^2 + y^2 + (z-0.64)^2))). \end{aligned}$$

In other words, we have three “hot blobs” centered at the positions $(0.64, 0, 0)$, $(0, 0.64, 0)$, and $(0, 0, 0.64)$, where only the first one is located inside the domain. In Figure 5.7 (right) we show a slice through the computational domain to give an indication of the temperature distribution. The viscosity μ is derived from the temperature by

$$\mu = \exp(-E_0 \mathsf{T}).$$

			unknown		frobenius		abs sum		schur	
Level	dof	E_0	t_{setup}	C_A	t_{setup}	C_A	t_{setup}	C_A	t_{setup}	C_A
3	2916	0	0.50	1.21	0.46	1.13	0.48	1.14	0.57	1.23
		7.5	0.23	1.22	0.19	1.09	0.21	1.13	0.36	1.53
		15	0.31	1.34	0.20	1.13	0.19	1.11	0.46	1.74
4	19652	0	4.43	1.22	2.61	1.14	2.59	1.13	4.54	1.27
		7.5	5.48	1.27	2.91	1.15	2.81	1.14	5.63	1.33
		15	8.44	1.36	2.99	1.14	3.03	1.14	10.33	2.21
5	143748	0	22.49	1.27	19.12	1.21	16.47	1.16	29.72	1.37
		7.5	27.01	1.30	20.37	1.22	17.32	1.17	31.63	1.37
		15	35.14	1.35	22.72	1.23	19.94	1.19	41.93	1.49
6	1098500	0	170.44	1.28	165.66	1.23	136.09	1.17	282.45	1.39
		7.5	177.81	1.31	171.12	1.24	142.37	1.17	281.20	1.39
		15	181.77	1.32	168.48	1.25	136.40	1.18	278.20	1.41

Table 5.9.: Setup time t_{setup} and operator complexity C_A for the AMG hierarchy to the HOTBLOB problem. Depicted are the figures unknown-based coarsening (Section 2.12.2) and various point-based AMG coarsening methods (Section 2.12.3), where the primary matrix $\tilde{A}$ is given by the Frobenius norms of the diffusion matrix blocks, the sum of all absolute values per diffusion matrix block, or the approximate Schur complement $B\hat{A}^{-1}B^T + C$, respectively. In all cases, the interpolation is computed unknown-wise.

We will show simulations for $E_0 = 0$, $E_0 = 7.5$ and $E_0 = 15$, respectively. The Rayleigh number is given by $Ra = 1e4$.
All computations were carried out on the "SIEBENGEBIRGE" cluster at the Institute for Numerical Simulation at the University of Bonn. This cluster consists of five SMP nodes with four eight-core Intel Xeon X7560 2.226 GHz CPUs and 512GB RAM each, connected by QDR 4x Infiniband (40Gb/s).
The solution of (5.8) is only unique up to a constant pressure function. We employ an AMG-preconditioned GMRES iteration (Algorithm 4.1), restarted after 20 steps. To circumvent the singularity, inside each matrix-vector product inside the GMRES we carry out the following steps,

1 we compute the inner product $q = \bar{\text{M}} \cdot p$, where $\bar{\text{M}}$ is the lumped mass matrix and p is the discrete pressure vector,

2 we add $c_p q$ to the first entry p_0 of the pressure factor, where we use the penalty factor $c_p = 10^{-3} here$,

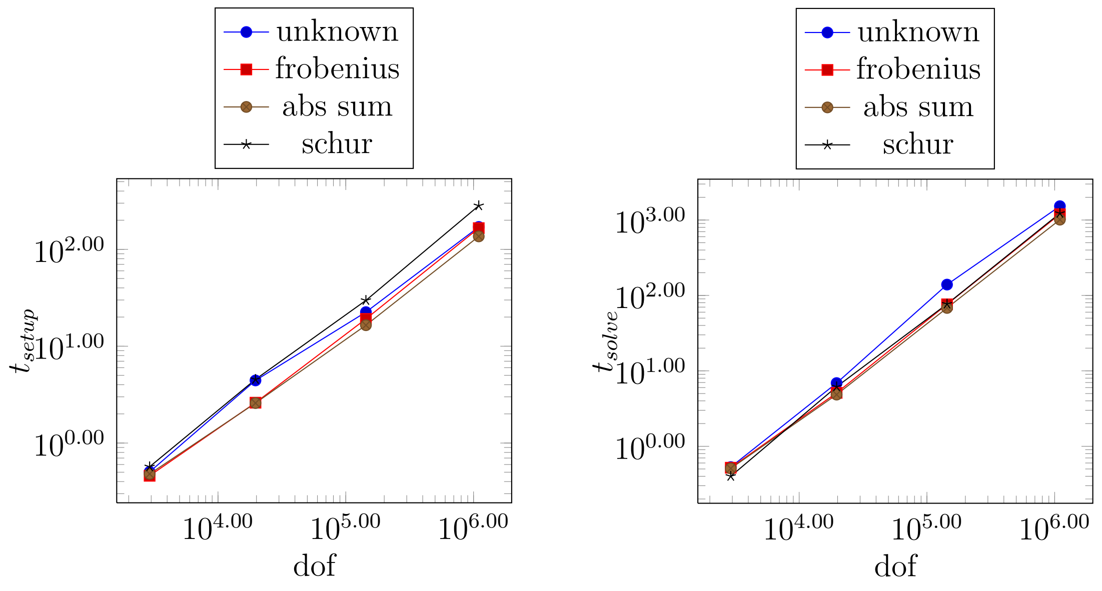
3 we add $c_p q$ to all entries in p (for symmetry reasons).

As AMG is used as preconditioner here, we employ the parallel PMIS coarsening scheme (Section 3.3). For the $Q_1 - Q_1$ finite element discretization, all unknowns are located at the same points. This allows us to employ point-based AMG approaches here.

We first compare unknown-based AMG (Section 2.12.2) with three different

			unknown		frobenius		abs sum		schur	
Level	dof	E_0	t_{solve}	it	t_{solve}	it	t_{solve}	it	t_{solve}	it
3	2916	0	0.53	14	0.52	13	0.51	13	0.40	11
		7.5	0.68	17	0.56	15	0.54	15	0.51	13
		15	1.26	31	1.02	26	1.07	27	1.27	31
4	19652	0	6.89	19	5.11	15	4.84	14	6.16	17
		7.5	7.54	20	5.83	16	5.73	16	6.96	18
		15	14.91	32	8.45	22	9.53	23	29.45	20
5	143748	0	139.15	33	76.21	20	68.22	19	76.85	16
		7.5	138.69	33	94.15	23	68.07	20	92.24	18
		15	188.39	43	130.73	35	118.45	33	133.25	20
6	1098500	0	1,525.75	41	1,193.41	36	1,009.71	32	1,227.56	25
		7.5	1,423.22	36	1,179.53	36	1,144.64	36	1,287.70	28
		15	1,450.09	40	1,574.80	49	1,316.97	47	1,411.75	31

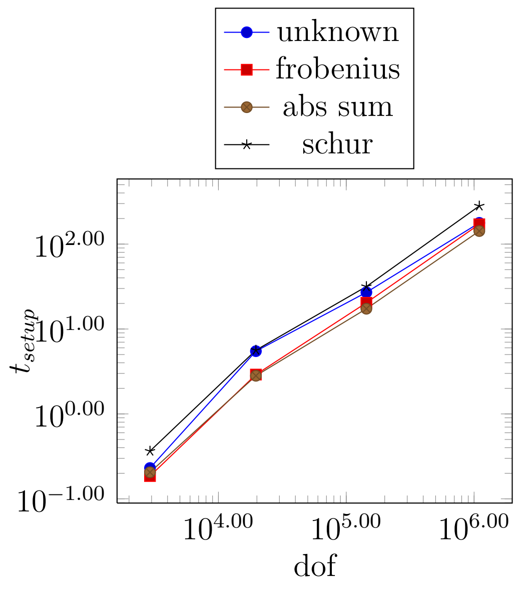
Table 5.10.: Solution time t_{solve} and number of iterations for AMG-preconditioned GMRES(20) applied to the HOTBLOB problem. Depicted are the figures unknown-based AMG (Section 2.12.2) and various point-based AMG (Section 2.12.3) methods, where the primary matrix $\tilde{A}$ is given by the Frobenius norms of the diffusion matrix blocks, the sum of all absolute values per diffusion matrix block, or the approximate Schur complement $B\hat{A}^{-1}B^T + C$, respectively.



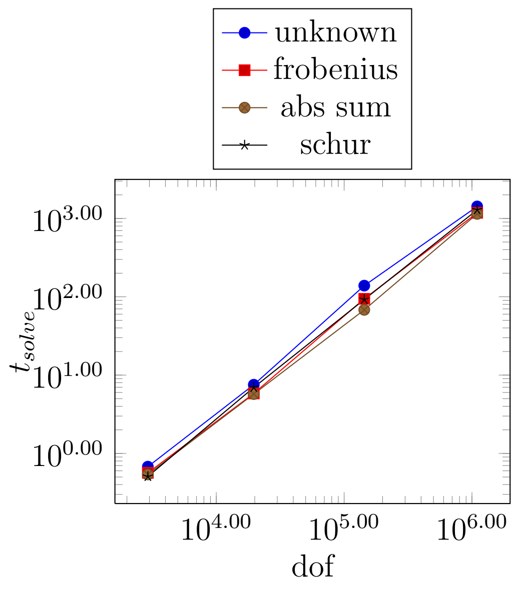
(a) Setup time t_{setup}

(b) Solution time t_{solve}

Figure 5.8.: Numerical Results for the HOTBLOB problem with $E_0 = 0$

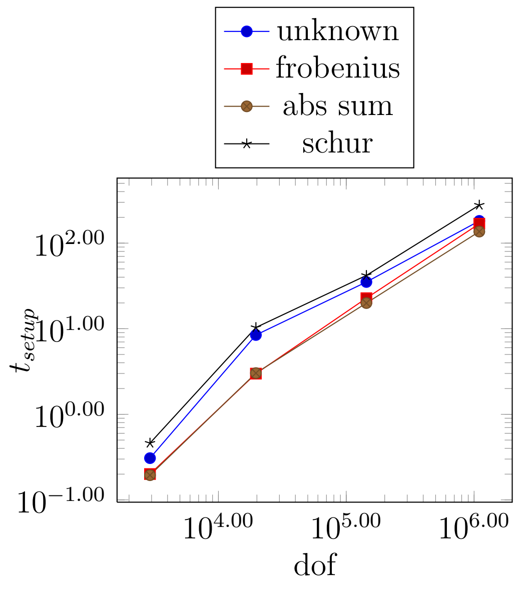


(a) Setup time t_{setup}

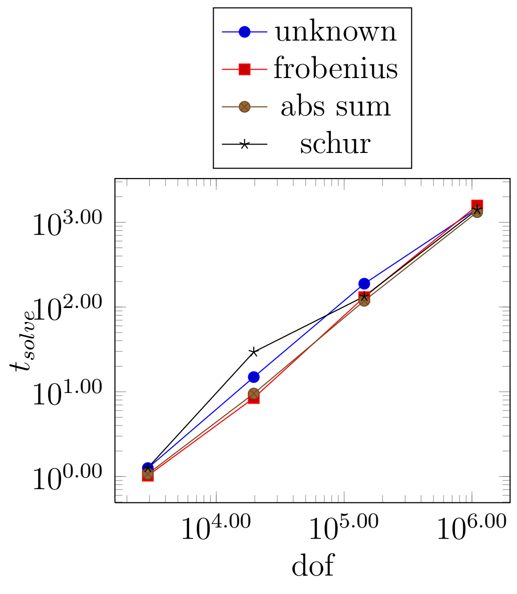


(b) Solution time t_{solve}

Figure 5.9.: Numerical Results for the HOTBLOB problem with $E_0 = 7.5$



(a) Setup time t_{setup}



(b) Solution time t_{solve}

Figure 5.10.: Numerical Results for the HOTBLOB problem with $E_0 = 15$

point-based AMG variants (Section 2.12.3). Remember that point-based AMG requires a *primary matrix* $\tilde{A}$. We employ the structure of the matrix $\mathcal{K}$,

$$\mathcal{K} = \begin{pmatrix} \mathcal{K}_{(1,1)} & \mathcal{K}_{(1,2)} & \dots & \mathcal{K}_{(1,M)} \\ \mathcal{K}_{(2,1)} & \mathcal{K}_{(2,2)} & \dots & \mathcal{K}_{(2,M)} \\ \vdots & \vdots & \ddots & \vdots \\ \mathcal{K}_{(M,1)} & \mathcal{K}_{(M,2)} & \dots & \mathcal{K}_{(M,M)} \end{pmatrix},$$

where each $\mathcal{K}_{(i,j)}$ has the form

$$\mathcal{K}_{(i,j)} = \begin{pmatrix} A_{(i,j)} & B^T_{(i,j)} \\ B_{(i,j)} & C_{(i,j)} \end{pmatrix}.$$

To define the primary matrix $\tilde{A}$, we propose the following variants:

1 Use the Frobenius norm of $A_{(i,j)}$, $\tilde{a}_{ij} := \|A_{(i,j)}\|_F$.

2 Define $\tilde{a}_{ij}$ as the sum over the absolute values of all entries in $A_{(i,j)}$.

3 Compute the inexact Schur complement $\tilde{A} := B\hat{A}^{-1}B^T + C$.

We use the primary matrix $\tilde{A}$ to obtain to coarsen the mesh, where we stop the coarsening if the number of points is less than 1000. Consequently, to obtain comparable results, for the unknown-based AMG variants we stop if there are less than 4000 degrees of freedom (we have three velocity and one pressure degree of freedom per point).
We employ multiple-unknown interpolation, i.e. the interpolation operator is computed unknown-wise. As can be seen from Table 5.9, the variants using the Frobenius norm and the sum over all absolute values per block allow a faster setup than unknown-based coarsening. Especially for the last one we see that the operator complexity only grows slightly with increasing problem size and is nearly independent of the viscosity $\mu = \exp(E_0 \mathsf{T})$. In consequence, we obtain scalable setup timings, i.e. the computational work per degree of freedom is nearly constant. The setup costs for the inexact Schur complement primary matrix variant are higher than for unknown-based AMG.
In Table 5.10 we give the solution timings and the number of iterations. Here, for all variants we have some dependence on the problem size as well as the viscosity. In nearly all cases the point-based approaches are faster than unknown-based AM, see also Figures 5.2–5.2.

We now investigate whether the performance can be improved if we use point-based interpolation instead of the multiple-unknown interpolation scheme. To this end, we fix the primary matrix $\tilde{A}$ to be computed using the sum of all absolute elements in each block (the fastest variant in tables 5.9–5.10). We compare

- multiple-unknown interpolation, i.e. modified classical interpolation computed per physical unknown,
- single-unknown interpolation, where $\tilde{A}$ is used to derive an interpolation operator that is extended to the whole system,

			multiple-unknown		single-unknown		block		diagonal block	
Level	dof	E_0	t_{setup}	C_A	t_{setup}	C_A	t_{setup}	C_A	t_{setup}	C_A
		0	0.48	1.14	0.46	1.14	0.46	1.25	0.46	1.25
3	2916	7.5	0.21	1.13	0.21	1.13	0.41	1.23	0.80	1.23
		15	0.19	1.11	0.18	1.11	0.36	1.20	0.36	1.20
		0	2.59	1.13	2.61	1.13	4.70	1.27	4.61	1.27
4	19652	7.5	2.81	1.14	2.80	1.14	5.13	1.30	4.91	1.30
		15	3.03	1.14	2.98	1.14	5.02	1.31	5.05	1.31
		0	16.47	1.16	16.11	1.16	75.86	1.51	57.66	1.42
5	143748	7.5	17.32	1.17	17.67	1.17	78.06	1.54	57.69	1.42
		15	19.94	1.19	18.98	1.19	89.31	1.64	71.99	1.50
		0	136.09	1.17	137.27	1.17	2,284.03	1.69	783.63	1.49
6	1098500	7.5	142.37	1.17	138.07	1.17	2,620.75	1.71	844.22	1.49
		15	136.40	1.18	142.31	1.17	–	1.79	1,015.58	1.53

Table 5.11.: Setup time t_{setup} and operator complexity C_A for the AMG hierarchy to the HOTBLOB problem. We give the numbers for multiple-unknown interpolation, single-unknown interpolation derived from the primary matrix $\tilde{A}$, block interpolation using the full blocks and block interpolation using the diagonal blocks. In all cases, the primary matrix $\tilde{A}$ is computed using the sum of the absolute values inside each diffusion matrix block.

			multiple-unknown		single-unknown		block		diagonal block	
Level	dof	E_0	t_{solve}	it	t_{solve}	it	t_{solve}	it	t_{solve}	it
		0	0.51	13	0.55	12	0.86	12	0.83	12
3	2916	7.5	0.54	15	0.68	14	0.95	14	1.00	15
		15	1.07	27	1.19	27	1.93	28	1.85	28
		0	4.84	14	4.73	13	7.88	14	7.78	14
4	19652	7.5	5.73	16	5.10	14	8.83	15	9.07	16
		15	9.53	23	8.18	22	12.45	21	15.36	26
		0	68.22	19	44.95	14	111.22	16	120.66	17
5	143748	7.5	68.07	20	54.49	15	127.56	18	148.68	21
		15	118.45	33	93.13	25	340.27	43	247.92	34
		0	1,009.71	32	438.38	15	5,776.09	36	2,385.97	30
6	1098500	7.5	1,144.64	36	486.76	16	5,063.50	30	2,553.29	32
		15	1,316.97	47	866.16	27	–	–	7,715.96	90

Table 5.12.: Solution time t_{solve} and number of iterations for AMG-preconditioned GMRES(20) applied to the HOTBLOB problem. We give the numbers for multiple-unknown interpolation, single-unknown interpolation derived from the primary matrix $\tilde{A}$, block interpolation using the full blocks and block interpolation using the diagonal blocks. In the case of (diagonal) block interpolation, also a (diagonal) block smoother is used. In all cases, the primary matrix $\tilde{A}$ is computed using the sum of the absolute values inside each diffusion matrix block.

Level	dof	E_0	1	8	16	32	64	128
5	143748	0	16.11	3.68	2.86	2.56	–	–
		7.5	17.67	3.76	3.22	2.68	–	–
		15	18.98	5.16	4.08	3.73	–	–
6	1098500	0	137.27	21.79	13.75	8.41	–	–
		7.5	138.07	22.66	14.12	8.81	–	–
		15	142.31	25.17	15.52	10.61	–	–
7	8586756	0	1,018.18	156.20	86.60	48.76	32.53	30.83
		7.5	1,045.64	159.02	87.98	49.18	33.79	29.81
		15	1,057.58	159.12	88.99	51.38	35.16	32.50
8	67898372	0	–	–	–	352.22	192.62	113.74
		7.5	–	–	–	424.77	195.32	117.27
		15	–	–	–	–	193.81	120.61

Table 5.13.: Setup time t_{setup} for the parallel AMG hierarchy to the HOTBLOB problem. In the top row we give the number of processors. The primary matrix $\tilde{A}$ is computed using the sum of the absolute values inside each diffusion matrix block and single unknown modified classical interpolation is employed.

- block classical interpolation (2.108) for the velocity,
- diagonal block classical interpolation (2.109) for the velocity,

where in the last two cases the pressure interpolation is computed separately using a modified classical interpolation scheme. For the (diagonal) block interpolation schemes, also a (diagonal) block Vanka smoother is used, see Remark 4.2.
From Table 5.11 we learn that the setup time using a block interpolation variants is significantly higher than point AMG with single- or multiple-unknown interpolation. The resulting AMG hierarchy is also larger, while the solution is not accelerated at all, see Table 5.12. Here, the additional costs for computing the block interpolation operators do not pay off, rather the opposite is true. On the other hand, if we employ single-unknown interpolation, we have the same setup timings and operator complexities as in the case of multiple-unknown interpolation, but obtain a preconditioned GMRES iteration that is very robust with respect to the problem size. We conclude this chapter with some results obtained from parallel experiments. We again use the sums of all absolute values inside each $A_{(i,j)}$ matrix block to obtain the primary matrix $\tilde{A}$ and employ single-unknown interpolation.
In Table 5.13 and we give the setup timings. While we have no perfect speed-up, we still significantly benefit from a larger number of processors. The same holds for the solution time (Table 5.14 and Figure). Regarding the number of iterations (Table 5.15) we see a significant increase if we go from sequential to parallel computation. Remember from Section 4.6 that our smoother is a hybrid additive/multiplicative method in this case: At the processor interfaces, we have a less efficient error reduction compared to the interior of each domain. The Vanka subdomain interpolation scalings are computed by (4.89) and (4.90), respectively.

Level	dof	E_0	1	8	16	32	64	128
		0	44.95	14.25	9.42	6.28	–	–
5	143748	7.5	54.49	14.82	9.44	6.70	–	–
		15	93.13	24.74	13.52	9.72	–	–
		0	438.38	137.51	96.33	55.61	–	–
6	1098500	7.5	486.76	146.52	90.07	57.06	–	–
		15	866.16	212.16	128.55	83.41	–	–
		0	4,896.55	1,073.85	745.90	480.81	294.03	182.69
7	8586756	7.5	7,275.45	1,105.01	599.86	459.91	303.35	157.96
		15	6,971.72	2,380.60	994.01	557.28	299.91	179.22
		0	–	–	–	5,234.90	2,586.21	1,228.07
8	67898372	7.5	–	–	–	5,319.72	2,044.21	1,276.77
		15	–	–	–	–	2,693.43	1,302.38

Table 5.14.: Solve time t_{solve} for parallel AMG-preconditioned GMRES(20) applied to the HOTBLOB problem. In the top row we give the number of processors. The primary matrix $\tilde{A}$ is computed using the sum of the absolute values inside each diffusion matrix block and single unknown modified classical interpolation is employed.

Level	dof	E_0	1	8	16	32	64	128
		0	14	28	35	41	–	–
5	143748	7.5	15	29	35	43	–	–
		15	25	43	49	57	–	–
		0	15	32	40	46	–	–
6	1098500	7.5	16	33	38	46	–	–
		15	27	50	53	69	–	–
		0	21	32	43	51	55	52
7	8586756	7.5	27	33	35	48	55	47
		15	27	75	55	58	53	53
		0	–	–	–	69	63	56
8	67898372	7.5	–	–	–	56	49	58
		15	–	–	–	–	65	60

Table 5.15.: Numbers of parallel AMG-preconditioned GMRES(20) iterations for the HOTBLOB problem. In the top row we give the number of processors. The primary matrix $\tilde{A}$ is computed using the sum of the absolute values inside each diffusion matrix block and single unknown modified classical interpolation is employed.

To summarize this chapter, we have demonstrated that our saddle point AMG, especially the F-stabilization method, can be applied in practice even if theoretical convergence questions remain open. We must however carefully choose the parameters (smoothing, interpolation unknown or point AMG etc.) to obtain an efficient method.

6. Conclusions and Outlook

In this thesis, we have constructed an algebraic multigrid (AMG) method for matrices with saddle point structure as they arise from the discretization of Stokes-like problems. In contrast to previous approaches to AMG for saddle point problems, our method is purely algebraic and does not need any additional geometric information.
The main idea is to employ a coupled interpolation operator, which leads to an additional stabilization term in the coarse grid matrix computed by the Galerkin product.
We have introduced three different approaches to construct such interpolation operators. In all of these cases, we start with a decoupled tentative interpolation matrix. The first ansatz is to apply a full symmetric inexact Uzawa smoothing step to the tentative prolongation to compute the final interpolation operator, while the restriction is just the transpose of the tentative prolongation. Consequently, the coarse grid operator is computed Petrov-Galerkin-wise and we were able to show two-grid convergence. The disadvantage of this method is the huge memory requirement of the smoothed prolongation, which in turn also leads to large (in terms of non-zero entries) coarse grid operators. This issue motivated us to abandon the full smoothing of the prolongation and only introduce a coupling between velocity and pressure components that is sufficient to establish a stabilization term on the next coarser level. This coupling term introduced additional interpolation weights for the velocity variables, which now also interpolate from coarse pressure variables. We have developed two variants of this approach. In the first case (pressure–to–velocity coupling), all velocity variables receive additional updates from the pressure space, while in the other case (F-stabilization) only the interpolation for the designated fine grid points is altered.
For all our stabilization techniques, we were able to prove the stability and thus the invertibility of the coarse level matrix by induction from the finest level, where we assume that an inf-sup condition is given. Moreover, the only additional preconditions in these stability proofs can easily be satisfied if the tentative velocity interpolation block is a well-fitted AMG interpolation for the upper left (symmetric positive definite) matrix block, i.e. satisfies a certain approximation property. While for the direct and classical AMG interpolation these theoretical results are well-known, so far they were missing for various elliptic AMG interpolation schemes that are commonly used, namely modified classical interpolation, extended interpolation, extended+i interpolation, and Jacobi interpolation. We have filled this gap and were able to verify an approximation property also for these methods.
Our saddle point AMG is not just a specific method, but rather an extension that can be wrapped around any existing AMG for symmetric positive definite matrices. Indeed, during the setup phase we just need to employ AMG coarsening and interpolation techniques to the upper left matrix block and an approximate (at least semi-definite) Schur

complement. Our prolongation stabilization then creates an overall stable coarse matrix. No geometric information or specific interplay between the coarse velocity and pressure spaces is needed.
We investigated the numerical performance of our saddle point AMG in several finite difference and finite element examples. As expected, the Petrov-Galerkin approach with the fully smoothed prolongation is not feasible due to the abundant memory requirements. On the other hand, we demonstrated the robustness of the F-stabilization method, especially if the AMG cycle is used as preconditioner within a restarted GMRES method.

There is still much room for further investigations. So far we have focused on Ruge-Stüben AMG, i.e. we have designated coarse and fine points and interpolated values from coarse to fine. A first extension would be to apply our method in the context of smoothed aggregation AMG [VMB94, VMB96], where the fine grid variables are grouped in patches that form the coarse grid variables. The pressure–to–velocity coupling stabilization technique seems to be especially fitted for this purpose. Second, we have experienced that the efficiency of our AMG can be very sensitive to the choice of parameters (additive or multiplicative smoothing, scaling within the multiplicative smoother, number of smoothing steps, unknown or nodal coarsening, etc). To obtain a real black-box solver, an automatic parameter fitting technique would be required. A related issue is the lack of an overall convergence theory, currently we just have a two-grid convergence result that holds in the case of the most memory-consuming stabilization approach and additive smoothing.
In summary, we have constructed an AMG method for Stokes-like problems. There are however other important saddle point systems that arise in the context of optimization problems subject to PDE constraints. The resulting matrices have a completely different nature and require their own techniques. While some geometric multigrid methods have been developed, an algebraic approach is still not known.

A. A positive definite reformulation of the Uzawa smoother

In this chapter, we describe under which conditions the inexact Uzawa smoother introduced in Section 4.5 can be rewritten as an iteration scheme over a symmetric positive definite matrix. We will also discuss to which extent the definite reformulation can be used to carry over AMG convergence theory for symmetric positive definite matrices to saddle point systems.
As in Chapter 4, let the saddle point matrix $\mathcal{K}$ be partitioned as follows (4.2),

$$\mathcal{K} = \begin{pmatrix} A & B^T \\ B & -C \end{pmatrix}, \; x = \begin{pmatrix} u \\ p \end{pmatrix} \text{ and } y = \begin{pmatrix} f \\ g \end{pmatrix}$$

where A and C are symmetric positive semi-definite matrices. The iteration matrix of the smoother is given by (4.67),

$$\mathcal{M} = I - \hat{\mathcal{K}}^{-1}\mathcal{K}$$

where

$$\hat{\mathcal{K}} = \begin{pmatrix} \hat{A} & B^T \\ B & B\hat{A}^{-1}B^T - \hat{S} \end{pmatrix}$$

In the following, we will discuss two different cases,

1 $\hat{A} < A$ and $\hat{S} > C + B\hat{A}^{-1}B^T$

2 $\hat{A} > A$ and $\hat{S} < C + B\hat{A}^{-1}B^T$.

For the case $C = 0$, the results presented in this chapter were first shown in [Zul02]. The case $C \neq 0$ has been addressed to in [Zul00], although with weaker bounds on the spectrum of $\mathcal{M}$. The proofs presented here mainly follow the ideas from [Zul02].

Case 1 In the first case, we assume that there are constants $\alpha_2 > 1, 0 < \beta_1 < 1$ such that

$$\begin{aligned} \hat{A} &< A \leq \alpha_2 \hat{A} && \text{(A.1)} \\ \beta_1 \hat{S} &\leq B\hat{A}^{-1}B^T + \beta_1 C && \text{(A.2)} \\ B\hat{A}^{-1}B^T + C &< \hat{S} && \text{(A.3)} \end{aligned}$$

We first define an inner product

$$(u, v)_{\mathcal{Q}} = u^T \mathcal{Q} v, \tag{A.4}$$

where

$$\mathcal{Q} = \mathcal{K} - \hat{\mathcal{K}} = \begin{pmatrix} A - \hat{A} & 0 \\ 0 & \hat{S} - B\hat{A}^{-1}B^T - C \end{pmatrix}.$$

The following theorem is an extension of Theorem 5.1 in [Zul02].

Theorem A.1 *Let the positive definite $A, \hat{A} \in \mathbb{R}^{N\times N}$, $\hat{S} \in \mathbb{R}^{M\times M}$, the positive semi-definite $C \in \mathbb{R}^{M\times M}$ and $B \in \mathbb{R}^{M,N}$ satisfy* (A.1) - (A.3). *Then,*

1 The iteration matrix $\mathcal{M} = I - \hat{\mathcal{K}}^{-1}\mathcal{K}$ (4.67) *is symmetric w.r.t.* (A.4)

2 $\sigma(\mathcal{M}) \subset [1-\alpha_2, \rho] \subset (-\infty, 1)$, where

$$\rho = \frac{(2-\alpha_2)(1-\beta_1)}{2} + \sqrt{\frac{(2-\alpha_2)^2(1-\beta_1)^2}{4} + (\alpha_2 - 1)(1-\beta_1)}$$

3 If $\alpha_2 < 2$, then $\rho(\mathcal{M}) = \|\mathcal{M}\|_{\mathcal{Q}} < 1$

4 $\hat{\mathcal{K}}^{-1}\mathcal{K}$ is spd w.r.t. (A.4) *and $\sigma(\hat{\mathcal{K}}^{-1}\mathcal{K}) \subset [1-\rho, \alpha_2] \subset (0, \infty)$*

The proof of this theorem heavily relies on the following lemma. For easier understanding, we also give its proof.

Lemma A.1 *([Zul02], Lemma 3.1) Let $\hat{A}$ and F be symmetric, positive definite $N \times N$-matrices, $B \in \mathbb{R}^{M\times N}$, $\hat{S}$ and G symmetric positive $M \times M$ matrices.*
Assume that there are real numbers $\rho_1 \le \rho_2 \le 0 < \rho_3 \le \rho_4 < \rho_5 \le \rho_6$ with

$$\varphi(\rho_1) \ge 0,\ \varphi(\rho_2) \le 0,\ \varphi(\rho_5) \ge 0,\ \varphi(\rho_6) \le 0 \tag{A.5}$$

where

$$\varphi(\mu) = \mu B(\mu\hat{A} - \hat{A}F^{-1}\hat{A})^{-1}B^T - \mu G - \hat{S} \tag{A.6}$$

and

$$\rho_3 F \le \hat{A} \le \rho_4 F \tag{A.7}$$

Then we have: If μ is an eigenvalue of the generalized eigenvalue problem

$$\begin{pmatrix} \hat{A} & B^T \\ B & B\hat{A}^{-1}B^T - \hat{S} \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix} = \mu \begin{pmatrix} Fu \\ Gp \end{pmatrix} \tag{A.8}$$

then $\mu \in [\rho_1, \rho_2] \cup [\rho_3, \rho_4] \cup [\rho_5, \rho_6]$.

Proof: Let $(u, p)^T \ne (0, 0)^T$ and λ be a solution of (A.8). Furthermore, let $\lambda \notin [\rho_3, \rho_4]$. Then, from (A.7), we have that $\lambda F - \hat{A}$ is either positive or negative definite and $p \ne 0$. In this case we obtain from the first line in (A.8)

$$u = \left(\lambda F - \hat{A}\right)^{-1} B^T p$$

and insert it into the right hand side of $u + \hat{A}^{-1}B^T p = \lambda \hat{A}^{-1}Fu$,

$$\begin{aligned} u + \hat{A}^{-1}B^T p &= \lambda \hat{A}^{-1}Fu, \\ &= \lambda \left(\lambda \hat{A} - \hat{A}F^{-1}\hat{A}\right)^{-1} B^T p. \end{aligned}$$

We use this term to substitute $u + \hat{A}^{-1}B^T p$ in the second line of (A.8) and obtain

$$\lambda B \left(\lambda \hat{A} - \hat{A}F^{-1}\hat{A}\right)^{-1} B^T p - \left(\lambda G + \hat{S}\right) p = 0.$$

We multiply from the left with p^T and obtain $p^T \varphi(\lambda) p = 0$. Furthermore, as

$$u^T \mu \left(\mu \hat{A} - \hat{A}F^{-1}\hat{A}\right)^{-1} u = u^T \left(\hat{A} - \mu^{-1}\hat{A}F^{-1}\hat{A}\right)^{-1} u$$

is monotone decreasing in μ, $p^T \varphi(\mu) p$ is strictly decreasing outside $[\rho_3, \rho_4]$. Now, (A.5) yields that $\lambda \in [\rho_1, \rho_2]$ or $\lambda \in [\rho_5, \rho_6]$. $\square$

Remark A.1 It is easy to see that, for all p, $p^T \varphi(0) p < 0$ and, for $\rho_0 > 0$ satisfying $\hat{S} \leq \rho_0 G$, we have

$$p^T \varphi(-\rho_0) p = p^T B \left(\hat{A} + \rho_0^{-1}\hat{A}F^{-1}\hat{A}\right)^{-1} B^T p + \rho_0 p^T G p - p^T \hat{S} p \geq 0.$$

This shows the existence of of ρ_1 and ρ_2. Also, in combination with the monocity of $p^T \varphi(\mu) p$, it follows that no positive eigenvalue of (A.8) can be smaller than ρ_3. Thus, even if ρ_5 and/or ρ_6 satisfying (A.5) cannot be found, we have an estimate from below for all positive eigenvalues.
In the following, we will derive estimates for ρ_2 and ρ_3 to provide estimates for the eigenvalues of the smoother.

Proof of Theorem A.1: From $\mathcal{Q} = \mathcal{K} - \hat{\mathcal{K}}$ one easily obtains that $\mathcal{Q}\mathcal{M} = \mathcal{Q}(I - \hat{\mathcal{K}}^{-1}\mathcal{K}) = -\mathcal{Q}\hat{\mathcal{K}}\mathcal{Q}$ is symmetric.
We consider the eigenvalue problem (A.8) with $F = A - \hat{A}$ and $G = \hat{S} - C - B\hat{A}^{-1}B^T$. Then the function $\varphi(\mu)$ of (A.6) has the form

$$\varphi(\mu) = \mu B \left[\mu \hat{A} - \hat{A}(A - \hat{A})^{-1}\hat{A}\right]^{-1} B^T + \mu C + \mu B\hat{A}^{-1}B^T - (\mu + 1)\hat{S}. \tag{A.9}$$

The iteration matrix $\mathcal{M}$ (4.67) of our smoother can be written as

$$\mathcal{M} = -\begin{pmatrix} \hat{A} & B^T \\ B & B\hat{A}^{-1}B^T - \hat{S} \end{pmatrix}^{-1} \begin{pmatrix} A - \hat{A} & 0 \\ 0 & \hat{S} - C - B\hat{A}^{-1}B^T \end{pmatrix}$$

and its eigenvalues ν correspond to the eigenvalues $\mu = -\frac{1}{\nu}$ of (A.8). Hence, using the notation of Lemma A.1, a sufficient criterion for the convergence of the smoother is given

by $\rho_2 < -1$ and $\rho_3 > 1$.
Assuming that $\mu \leq -1$ and multiplying (A.2) by $-\frac{\mu+1}{\beta_1}$ we obtain

$$-(\mu+1)\hat{S} \;\leq\; -\frac{\mu+1}{\beta_1} B\hat{A}^{-1}B^T - (\mu+1)C. \tag{A.10}$$

We insert (A.10) into (A.9) and obtain:

$$\varphi(\mu) \leq \mu B\left[\mu\hat{A} - \hat{A}(A-\hat{A})^{-1}\hat{A}\right]^{-1} B^T - C + \mu B\hat{A}^{-1}B^T - \frac{\mu+1}{\beta_1}B\hat{A}^{-1}B^T.$$

We see that $\varphi(\mu) \leq -C \leq 0$ if

$$\begin{aligned} \mu\left[\mu\hat{A} - \hat{A}(A-\hat{A})^{-1}\hat{A}\right]^{-1} + \mu\hat{A}^{-1} - \frac{\mu+1}{\beta_1}\hat{A}^{-1} &\leq 0 \\ \Leftrightarrow \mu\left[\mu\hat{A} - \hat{A}(A-\hat{A})^{-1}\hat{A}\right]^{-1} &\leq \frac{\mu+1-\mu\beta_1}{\beta_1}\hat{A}^{-1} \end{aligned}$$

which is equivalent to

$$\mu\beta_1\hat{A} \geq (\mu+1-\mu\beta_1)(\mu\hat{A} - \hat{A}(A-\hat{A})^{-1}\hat{A}) \tag{A.11}$$

(note that from $\mu \leq -1$ and $\hat{A} < A$ it follows that $\left[\mu\hat{A} - \hat{A}(A-\hat{A})^{-1}\hat{A}\right] < 0$). We multiply from both sides with $\hat{A}^{-1/2}$ to obtain

$$\mu\beta_1 I \geq (\mu+1-\mu\beta_1)(\mu I - (\bar{A}-I)^{-1}) \tag{A.12}$$

where $\bar{A} = \hat{A}^{-1/2}A\hat{A}^{-1/2}$. From the second inequality in (A.1) we derive $\rho(\bar{A}) \leq \alpha_2$ and $\rho(-(\bar{A}-I)^{-1}) \leq -\frac{1}{\alpha_2-1}$. In consequence, for $\mu \geq -\frac{1}{1-\beta_1}$ (i.e. $\mu+1-\mu\beta_1 > 0$), inequality (A.12) is satisfied if and only if

$$\mu\beta_1 \geq (\mu+1-\mu\beta_1)(\mu - \frac{1}{\alpha_2-1})$$

The negative root of the quadratic equation

$$\mu\beta_1 = (\mu+1-\mu\beta_1)(\mu - \frac{1}{\alpha_2-1})$$

is given by

$$\frac{1}{\mu^*} = -\frac{(2-\alpha_2)(1-\beta_1)}{2} - \sqrt{\frac{(2-\alpha_2)^2(1-\beta_1)^2}{4} + (\alpha_2-1)(1-\beta_1)} \in (-1, -(1-\beta_1)).$$

We have found an μ^* with $\varphi(\mu^*) \leq 0$. Hence, all negative eigenvalues of the generalized eigenvalue problem (A.8) satisfy $\mu \leq \mu^* < -1$ and all positive eigenvalues ν of the smoother fulfill $\nu \leq -\frac{1}{\mu^*} < 1$.

The positive eigenvalues of (A.8) satisfy, according to Lemma A.1, $\mu > \rho_3$. The lower bound is given by $\rho_3(A - \hat{A}) \leq \hat{A}$ respectively $\rho_3(\bar{A} - I) \leq I$. This is satisfied for

$$\rho_3 = \frac{1}{\alpha_2 - 1}$$

so we have $\nu \leq -\frac{1}{\rho_3} = 1 - \alpha_2$.
Statement 3 follows from Statement 2 and Statement 4 is a consequence of 1 and 2. □
We now consider the damped iterative method $\mathcal{M}_\omega = (1-\omega) + \omega\mathcal{M}$. This corresponds to the eigenvalue problem (A.8) with $\mu = -\frac{\omega}{\nu+\omega-1}$. Straightforward calculations show that a sufficient condition for $\nu > -1$ is given by $\omega < \frac{2}{\alpha_2}$ and $\nu > 0$ if $\omega < \frac{1}{\alpha_2}$. On the other hand, $\nu < 1$ for all positive values of ω. Hence we obtain

Corollary A.1 *The damped smoother $\mathcal{M}_\omega = (1-\omega)+\omega\mathcal{M}$ converges for all $\omega \in (0, \frac{2}{\alpha_2})$. If $\omega \in (0, \frac{1}{\alpha_2})$, then the spectrum of $\mathcal{M}_\omega$ is positive.*

Case 2 We assume that there are constants $\alpha_1 > 0, \beta_2 > 1$ such that

$$\begin{aligned} \alpha_1 \hat{A} &\leq A < \hat{A} && \text{(A.13)}\\ \hat{S} &< B\hat{A}^{-1}B^T + C && \text{(A.14)}\\ B\hat{A}^{-1}B^T + \beta_2 C &\leq \beta_2 \hat{S} && \text{(A.15)} \end{aligned}$$

As in the previous paragraph, we define an inner product

$$(x, y)_{\mathcal{Q}} = x^T \mathcal{Q} y, \tag{A.16}$$

where

$$\mathcal{Q} = \hat{\mathcal{K}} - \mathcal{K} = \begin{pmatrix} \hat{A} - A & 0 \\ 0 & B\hat{A}^{-1}B^T + C - \hat{S} \end{pmatrix}.$$

With these conditions and definitions, we are able to proof the following generalization of Theorem 5.2 from [Zul02].

Theorem A.2 *Let the symmetric positive definite $\hat{A} \in \mathbb{R}^{N\times N}, \hat{S} \in \mathbb{R}^{M\times M}$, the symmetric positive semi-definite, $A \in \mathbb{R}^{N\times N}, C \in \mathbb{R}^{M\times M}$ and $B \in \mathbb{R}^{M,N}$ satisfy* (A.13) - (A.15). *Then,*

1 The iteration matrix $\mathcal{M} = I - \hat{\mathcal{K}}^{-1}\mathcal{K}$ is symmetric w.r.t. (A.16)

2 $\sigma(\mathcal{M}) \subset [-\rho, 1-\alpha_1] \subset (-\infty, 1)$, where

$$\rho = \frac{(2-\alpha_1)(\beta_2 - 1)}{2} + \sqrt{\frac{(2-\alpha_1)^2(\beta_2-1)^2}{4} + (1-\alpha_1)(\beta_2 - 1)}$$

3 If $\beta_2 < 1 + 1/(3 - 2\alpha_1)$, then $\rho(\mathcal{M}) = \|\mathcal{M}\|_{\mathcal{Q}} < 1$

4 $\hat{\mathcal{K}}^{-1}\mathcal{K}$ *is symmetric positive definite w.r.t.* (A.16) *and* $\sigma(\hat{\mathcal{K}}^{-1}\mathcal{K}) \subset [\alpha_1, 1+\rho] \subset (0,\infty)$

Proof: Again, it is clear that $\mathcal{QM}$ is symmetric.
We now consider the eigenvalue problem (A.8) with $F = \hat{A}-A$ and $G = C+B\hat{A}^{-1}B^T-\hat{S}$. The function $\varphi(\mu)$ (A.6) takes the form

$$\varphi(\mu) = \mu B\left[\mu\hat{A} - \hat{A}(\hat{A}-A)^{-1}\hat{A}\right]^{-1} B^T - \mu C - \mu B\hat{A}^{-1}B^T + (\mu-1)\hat{S}$$

The smoother can be written as

$$\mathcal{M} = \begin{pmatrix} \hat{A} & B^T \\ B & B\hat{A}^{-1}B^T - \hat{S} \end{pmatrix}^{-1} \begin{pmatrix} \hat{A}-A & 0 \\ 0 & C + B\hat{A}^{-1}B^T - \hat{S} \end{pmatrix}$$

and its eigenvalues ν correspond to the eigenvalues $\mu = \frac{1}{\nu}$ of (A.8). As in the first case and again using the notation of Lemma A.1, a sufficient criterion for the convergence of the smoother is given by $\rho_2 < -1$ and $\rho_3 > 1$.
We assume that $\mu \le -1$ and reformulate (A.15) into

$$(\mu-1)\hat{S} \ \le\ \frac{\mu-1}{\beta_2} B\hat{A}^{-1}B^T + (\mu-1)C. \tag{A.17}$$

As before, we insert (A.17) into (A.9):

$$\varphi(\mu) = \mu B\left[\mu\hat{A} - \hat{A}(\hat{A}-A)^{-1}\hat{A}\right]^{-1} B^T - C - \mu B\hat{A}^{-1}B^T + \frac{\mu-1}{\beta_2} B\hat{A}^{-1}B^T.$$

We obtain $\varphi(\mu) \le -C \le 0$ for

$$\mu\left[\mu\hat{A} - \hat{A}(A-\hat{A})^{-1}\hat{A}\right]^{-1} \le \frac{\beta_2\mu - \mu + 1}{\beta_2}\hat{A}^{-1}$$

and completely on the lines of the first case we conclude that equality is obtained for

$$\frac{1}{\mu^*} = -\frac{(2-\alpha_1)(\beta_2-1)}{2} - \sqrt{\frac{(2-\alpha_1)^2(\beta_2-1)^2}{4} + (1-\alpha_1)(\beta_2-1)}.$$

We have $\mu \le \mu^* < -1$ for the negative eigenvalues of (A.8) and, in consequence, $\nu \ge \frac{1}{\mu^*} > -1$ for the negative eigenvalues of the smoother.
The remainder of the proof is is analogous to the proof of Theorem A.1. □

Corollary A.2 *The damped smoother* $\mathcal{M}_\omega = (1-\omega) + \omega\mathcal{M}$ *converges for all* $\omega \in (0, \frac{2}{\rho+1})$*. If in addition* $\omega < \frac{1}{\rho+1}$*, then the spectrum of* $\mathcal{M}_\omega$ *is positive.*

These theorems allow us to interpret the inexact Uzawa-type smoothers as relaxation schemes for the symmetric positive definite matrix $\mathcal{L} = \mathcal{Q}\hat{\mathcal{K}}^{-1}\mathcal{K}$. The iteration matrix can be rewritten as

$$\mathcal{M} = I - \hat{\mathcal{K}}^{-1}\mathcal{K} = I - \mathcal{Q}^{-1}\mathcal{L}$$

i.e. we employ $\mathcal{Q}$ as a "preconditioner" for $\mathcal{L}$. In the following, we derive an algebraic formulation of the smoothing property for Uzawa-type smoothers. As in the scalar case, we introduce three inner products,

$$(u,v)_0 := u^T\mathcal{Q}v, \quad (u,v)_1 := u^T\mathcal{L}v, \quad (u,v)_2 := u^T\mathcal{L}\mathcal{Q}^{-1}\mathcal{L}v \tag{A.18}$$

and the respective norms $\|e\|_0,\ \|e\|_1,\ \|e\|_2$. With these norms, we can now formulate the smoothing property as in the scalar case,

$$\|\mathcal{M}e\|_1^2 \le \|e\|_1^2 - \sigma\|e\|_2^2. \tag{A.19}$$

A straightforward calculation shows

$$\|\mathcal{M}e\|_1^2 = \|e - \mathcal{Q}^{-1}\mathcal{L}e\|_1^2 = \|e\|_1^2 - \left(\mathcal{Q}(2I - \hat{\mathcal{K}}^{-1}\mathcal{K})\hat{\mathcal{K}}^{-1}\mathcal{K}e, \hat{\mathcal{K}}^{-1}\mathcal{K}e\right)$$

Hence, the smoothing property is equivalent to

$$\sigma\mathcal{Q} \le \mathcal{Q}\left(2I - \hat{\mathcal{K}}^{-1}\mathcal{K}\right). \tag{A.20}$$

One immediately obtains inequality (A.20) from the Theorems A.1 or A.2 with $\sigma = 2-\alpha_2$ or $\sigma = 1-\rho$, respectively.
Now, the question arises whether the coarse grid operator can also be constructed as in the scalar case, i.e.

$$\mathcal{K}^C = \mathcal{P}^T\mathcal{K}\mathcal{P}. \tag{A.21}$$

The coarse grid correction is then obtained by

$$\tilde{e} \leftarrow \left(I - \mathcal{P}\left(\mathcal{K}^C\right)^{-1}\mathcal{P}^T\mathcal{K}\right)e. \tag{A.22}$$

To measure the error reduction of this coarse grid correction in the norms $\|e\|_0,\ \|e\|_1,\ \|e\|_2$ defined by the scalar products from (A.18), we need to reformulate (A.22) in terms of $\mathcal{Q}$ and $\mathcal{L}$. To this end, let again $\mathcal{L} = \mathcal{Q}\hat{\mathcal{K}}^{-1}\mathcal{K}$ and

$$\mathcal{T} = Q\hat{\mathcal{K}}^{-1} = \left(K\hat{\mathcal{K}}^{-1} - I\right) = -\mathcal{M}^T \text{ (Case 1)}$$

or

$$\mathcal{T} = Q\hat{\mathcal{K}}^{-1} = \left(I - K\hat{\mathcal{K}}^{-1}\right) = \mathcal{M}^T \text{ (Case 2).}$$

Furthermore, we assume that there exists an invertible $\tilde{\mathcal{T}}$ such that $\mathcal{P}^T\mathcal{T} = \tilde{\mathcal{T}}\mathcal{P}^T$. Then we have

$$\begin{aligned}
\mathcal{P}\left(\mathcal{P}^T\mathcal{L}\mathcal{P}\right)^{-1}\mathcal{P}^T\mathcal{L} &= \mathcal{P}\left(\mathcal{P}^T\mathcal{T}\mathcal{K}\mathcal{P}\right)^{-1}\mathcal{P}^T\mathcal{T}\mathcal{K} \\
&= \mathcal{P}\left(\tilde{\mathcal{T}}\mathcal{P}^T\mathcal{K}\mathcal{P}\right)^{-1}\tilde{\mathcal{T}}\mathcal{P}^T\mathcal{K} \\
&= \mathcal{P}\left(\mathcal{P}^T\mathcal{K}\mathcal{P}\right)^{-1}\mathcal{P}^T\mathcal{K}
\end{aligned}$$

i.e. the coarse grid correction for the operator $\mathcal{L}$ equals the coarse grid correction for $\mathcal{K}$. Now, we can apply Theorem 2.15 (for the two-grid case) and Lemma 2.3 (for the approximate correction) to obtain

$$\|\tilde{e}\|_1 \leq \|e\|_1$$

where $\tilde{e}$ is obtained from e by the exact or approximate coarse grid correction. Together with the smoothing property (A.19) we get V-cycle convergence, see e.g. (2.36) and [Stü99], Theorem 4.1,

$$\|M\tilde{e}\|_1^2 \leq \|\tilde{e}\|_1^2 - \sigma\|\tilde{e}\|_2^2 \leq \left(1 - \frac{\sigma}{\tau}\right)\|\tilde{e}\|_1^2 \leq \|e\|_1^2.$$

Now, we discuss the conditions under which $\tilde{\mathcal{T}}$ can be constructed. To this end, we take an arbitrary $x \in ker\mathcal{P}^T$. Then, from

$$0 = \tilde{\mathcal{T}}\mathcal{P}^T x = \mathcal{P}^T\mathcal{T}x$$

we see that also $\mathcal{T}x$ needs to be in the nullspace of $\mathcal{P}^T$ (remember that $\mathcal{T}$ is invertible). Vice versa, any x such that $\mathcal{T}x \in ker\mathcal{P}^T$ must satisfy $x \in ker\mathcal{P}^T$. In other words, the nullspace of the restriction operator $\mathcal{P}^T$ needs to be invariant under the action of $\mathcal{T} = \pm\mathcal{M}^T$. Hence, to construct the interpolation operator $\mathcal{P}$ we first need to exactly identify invariant subspaces under the smoothing operator $\mathcal{M}$. Then, $\mathcal{P}$ needs to be carefully set up such that its image (and the kernel of its transpose $\mathcal{P}^T$) are exactly aligned along these subspaces.
In general, a prolongation matrix $\mathcal{P}$ that is constructed this way is not sparse. On the other hand, it is not clear whether an invertible $\tilde{\mathcal{T}}$ such that $\mathcal{P}^T\mathcal{T} \approx \tilde{\mathcal{T}}\mathcal{P}^T$ is only approximate satisfied also gives a convergent method.
In the remainder of this chapter, we give two additional drawbacks of this approach to a saddle point AMG. First, we need to construct one of the preconditioner $\hat{A}$ or $\hat{S}$ such that

$$\begin{array}{rcl} \hat{A} & < & A \text{ (Case 1)} \\ \text{or } \hat{S} & < & B\hat{A}^{-1}B^T + C \text{ (Case 2)}. \end{array}$$

In practice, $\hat{A}$ and $\hat{S}$ are obtained from the scaled diagonals of A and $B\hat{A}^{-1}B^T + C$ respectively. Now, to determine the correct scaling factor, we need to know the smallest eigenvalue of

$$\hat{A}^{-1/2}A\hat{A}^{-1/2} \text{ or } \hat{S}^{-1/2}\left(B\hat{A}^{-1}B^T + C\right)\hat{S}^{-1/2},$$

i.e. we need to carry out an eigenvalue solver [BDD+00] which also may need a preconditioner itself to converge fast enough. We hence may need to employ a scalar AMG method for A or $B\hat{A}^{-1}B^T + C$ on each level just to obtain the correct scaling to set up the smoother.
If we do not require a definite formulation, i.e. if we proceed as described in Section 4.5, we only need upper bounds on the dominant eigenvalues of

$$\hat{A}^{-1/2}A\hat{A}^{-1/2} \text{ and } \hat{S}^{-1/2}\left(B\hat{A}^{-1}B^T + C\right)\hat{S}^{-1/2}.$$

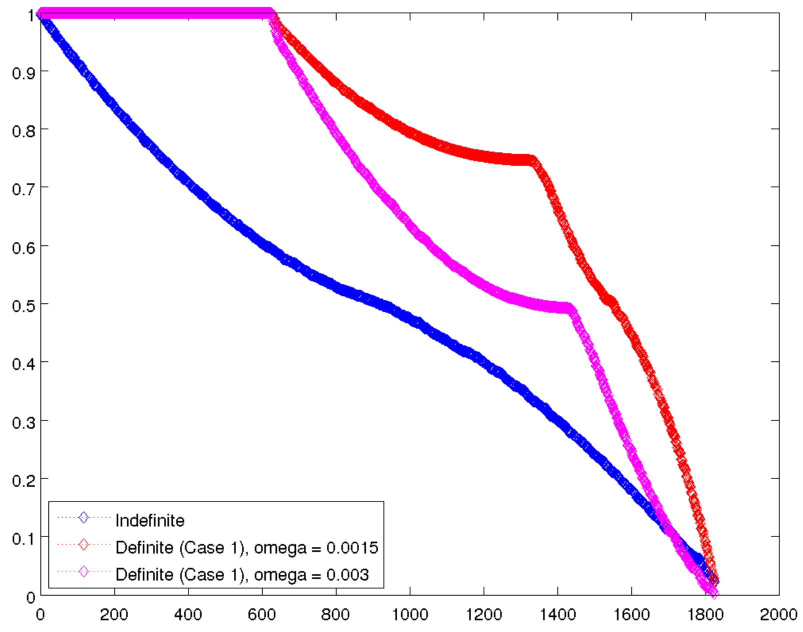


Figure A.1.: Absolute eigenvalues of the inexact Uzawa smoother in the indefinite formulation as well as the definite (Case 1) formulation with different damping parameters.

As these bounds do not need to be very accurate, they can easily be obtained by a few (unpreconditioned) power iterations or from Gershgorin's circle theorem.
Another disadvantage of the definite smoothers is the fact that they generally must be heavily damped to converge. In the first case, the damping parameter ω needs to be smaller that $\frac{2}{\alpha_2}$ (or $\frac{1}{\alpha_2}$ if one wants only positive eigenvalues). Now, according to (A.1), α_2 is chosen such that $\hat{A} < A < \alpha_2 \hat{A}$, which implies that α_2 is proportional to the condition number of $\hat{A}^{-1/2} A \hat{A}^{-1/2}$. In contrast, for the indefinite formulation we do not need damping at all. We illustrate the effects of the damping in the following example (For Case 2, similar considerations can be made).

Example A.1 On a square domain in two spatial dimensions, consider Stokes' equation

$$\begin{aligned} -\Delta u + \nabla p &= f \\ -\nabla \cdot u &= 0 \end{aligned}$$

with inflow/outflow boundary conditions on the western and eastern boundary and Neumann boundary conditions on the other boundaries. We employ a staggered grid discretization with 25 pressure cells in each spatial direction, yielding a total of 1825 scalar unknowns. In Figure A.1 we show the absolute eigenvalues of different formulations of the inexact Uzawa smoother (note that the indefinite formulation leads to non-real eigenvalues). For the definite case, we consider both a damping parameter yielding a positive spectrum (the red plot) as well as a damping parameter that is just small enough to ensure convergence (the magenta colored line). We see that in the definite case, about a third of the eigenvalues remains very near to 1 and the remaining eigenvalues are larger than those of the indefinite formulation.

Summarizing this chapter, we have seen that the inexact Uzawa smoother can be reformulated as an iteration method for a definite linear system $\mathcal{L}$. We are able to define inner norms $\|\cdot\|_i$, where $i = 0, 1, 2$, and show an algebraic smoothing property

$$\|\mathcal{M}e\|_1^2 \leq \|e\|_1^2 - \sigma\|e\|_2^2.$$

This does however not help us to completely carry over the AMG convergence theory from symmetric positive definite (M-) matrices, as the variational property can only be shown under a restrictive condition. Furthermore, the definite reformulations of the smoother require both a compute-intensive setup and are slower than their indefinite counterpart.

List of Algorithms

List of Figures

List of Tables

Bibliography

[Ada04] Mark F. Adams. Algebraic multigrid methods for constrained linear systems with applications to contact problems in solid mechanics. *Numerical Linear Algebra with Applications*, 11(2-3):141–153, 2004.

[AHU58] Kenneth J. Arrow, Leonid Hurwicz, and Hirofumi Uzawa. *Studies in Linear and Non-Linear Programming*. Stanford University Press, 1958.

[Alb06] David M. Alber. Modifying CLJP to select grid hierarchies with lower operator complexities and better performance. *Numerical Linear Algebra With Applications*, 13(2–3):87–104, 2006.

[Alb07] David M. Alber. *Efficient Setup Algorithms for Parallel Algebraic Multigrid*. PhD thesis, University of Illinois, 2007.

[BBKL11] Achi Brandt, James Brannick, Karsten Kahl, and Ira Livshits. Bootstrap AMG. *SIAM Journal on Scientific Computing*, 33(2):612–613, 2011.

[BCF$^+$00] Marian Brezina, Andrew J. Cleary, Robert D. Falgout, Van Emden Henson, Jim E. Jones, Thomas A. Manteuffel, Steve F. McCormick, and John W. Ruge. Algebraic Multigrid Based on Element Interpolation (AMGe). *SIAM Journal on Scientific Computing*, 22:1570–1592, 2000.

[BD85] Randolf E. Bank and Craig C. Douglas. Sharp estimates for multigrid rates of convergence with general smoothing and acceleration. *SIAM Journal on Numerical Analysis*, 22:617–633, 1985.

[BDD$^+$00] Zhaojun Bai, James Demmel, Jack Dongarra, Axel Ruhe, and Henk van der Vorst, editors. *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide*. SIAM, Philadelphia, 2000.

[BDG06] Pavel B. Bochev, Clark R. Dohrmann, and Max D. Gunzburger. Stabilization of low-order mixed finite elements for the stokes equations. *SIAM Journal on Numerical Analysis*, 44:82–101, 2006.

[BDO12] Nathan Bell, Steven Dalton, and Luke N. Olson. Exposing Fine-Grained Parallelism in Algebraic Multigrid Methods. *SIAM Journal on Scientific Computing*, 34(4):C123–C152, 2012.

[BGG$^+$08] Carsten Burstedde, Omar Ghattas, Michael Gurnis, Georg Stadler, Eh Tan, Tiankai Tu, Lucas C. Wilcox, and Shijie Zhong. Scalable adaptive mantle convection simulation on petascale supercomputers. In *Proceedings of the 2008 ACM/IEEE conference on Supercomputing*, SC '08, pages 62:1–62:15, Piscataway, NJ, USA, 2008. IEEE Press.

[BGMS97] Satish Balay, William D. Gropp, Lois Curfman McInnes, and Barry F. Smith. Efficient management of parallelism in object oriented numerical software libraries. In Erlend Arge, Are Magnus Bruaset, and Hans Petter Langtangen, editors, *Modern Software Tools in Scientific Computing*, pages 163–202. Birkhäuser Press, 1997.

[BKMY10] Allison H. Baker, Tzanio V. Kolev, and Ulrike Meier Yang. Improving algebraic multigrid interpolation operators for linear elasticity problems. *Numerical Linear Algebra With Applications*, 17(2–3):495517, 2010.

[BMR82] Achi Brandt, Steve McCormick, and John Ruge. Algebraic multigrid (amg) for automatic multigrid solution with application to geodetic computations. Technical report, Instute for Computational Studies, Colorado State University, Fort Collins, CO, 1982.

[BMR84] Achi Brandt, Steve McCormick, and John Ruge. Algebraic multigrid (amg) for sparse matrix equations. In D.J. Evans, editor, *Sparsity and Its Applications.* Cambridge Univsity Press, 1984.

[Bra97] Dietrich Braess. *Finite Elemente.* Springer Verlag, Berlin Heidelberg New York, 2 edition, 1997.

[Bra01] Achi Brandt. Multiscale scientific computation: Review 2001. In *Multiscale and Multiresolution Methods*, pages 1–96. Springer Verlag, 2001.

[Bre74] Franco Brezzi. On the existence, uniqueness and approximation of saddle-point problems arising from lagrangian multipliers. *Revue francaiçe d'automatique, informatique, recherche opérationelle. Analyse numérique*, 8:129–151, 1974.

[BS97] Dietrich Braess and Regina Sarazin. An efficient smoother for the Stokes problem. *Applied Numerical Mathematics*, 23:3–19, 1997.

[BSA$^+$13] Carsten Burstedde, Georg Stadler, Laura Alisic, Lucas C. Wilcox, Eh Tan, Michael Gurnis, and Omar Ghattas. Large-scale adaptive mantle convection simulation. *Geophysical Journal International*, 2013.

[CCF98] Edmond Chow, Andrew J. Cleary, and Robert D. Falgout. Design of the *hypre* Preconditioner Library. In *Proceedings of the SIAM Workshop on Object Oriented Methods for Inter-operable Scientific and Engineering Computing*, 1998.

[Cle04] Tanja Clees. *AMG Strategies for PDE Systems with Applications in Industrial Semiconductor Simulation.* Dissertation, Universität zu Köln, 2004.

[DBH$^+$02] Karen Devine, Erik Boman, Robert Heaphy, Bruce Hendrickson, and Courtenay Vaughan. Zoltan data management services for parallel dynamic applications. *Computing in Science and Engineering*, 4(2):90–97, 2002.

[Den82] Joel E. Dendy, Jr. Black box multigrid. *J. Comp. Phys.*, 48:366–386, 1982.

[ESW05] Howard Elman, David Silvester, and Andy Wathen. *Finite Elements and Fast Iterative Solvers.* Numerical Simulation and Scientific Computation. Oxford University Press, New York, 2005.

[FV04] Robert D. Falgout and Panayot S. Vassilevski. On generalizing the algebraic multigrid framework. *SIAM Journal on Numerical Analysis*, 42(4):1669–1693, 2004.

[GDN98] Michael Griebel, Thomas Dornseifer, and Tilman Neunhoeffer. *Numerical Simulation in Fluid Dynamics, a Practical Introduction.* SIAM, Philadelphia, 1998.

[GKW11] Michael W. Gee, Ulrich Küttler, and Wolfgang A. Wall. Truly monolithic algebraic multigrid for fluidstructure interaction. *International Journal for Numerical Methods in Engineering*, 85(8):987–1016, 2011.

[GMOS06] Michael Griebel, Bram Metsch, Daniel Oeltz, and Marc Alexander Schweitzer. Coarse grid classification: A parallel coarsening scheme for algebraic multigrid methods. *Numerical Linear Algebra with Applications*, 13(2–3):193–214, 2006. Also available as SFB 611 preprint No. 225, Universität Bonn, 2005.

[GMS06] Michael Griebel, Bram Metsch, and Marc Alexander Schweitzer. Coarse grid classification–Part II: Automatic coarse grid agglomeration for parallel AMG. Preprint 271, Sonderforschungsbereich 611, Universität Bonn, 2006.

[GMS08] Michael Griebel, Bram Metsch, and Marc Alexander Schweitzer. Coarse Grid Classification: AMG on Parallel Computers. In Gernot Münster, Dietrich Wolf, and Manfred Kremer, editors, *NIC Symposium 2008*, volume 39 of *NIC Series*, pages 299–306, February 2008. Also available as SFB 611 preprint No. 368, Universität Bonn, 2008.

[GOS03] Michael Griebel, Daniel Oeltz, and Marc Alexander Schweitzer. An Algebraic Multigrid Method for Linear Elasticity. *SIAM Journal on Scientific Computing*, 25(2):385–407, 2003.

[GS98] Philip M. Gresho and Robert L. Sani. *Incompressible Flow and the Finite Element Method.* John Wiley and Sons, Chichester, 1998.

[Hac85] Wolfgang Hackbusch. *Multi-Grid Methods and Applications*, volume 4 of *Springer Series in Computational Mathematics.* Springer-Verlag, Berlin, Heidelberg, 1985.

[Hac86] Wolfgang Hackbusch. *Theorie und Numerik elliptischer Differentialgleichungen.* Teubner Studienbücher: Mathematik. B. G. Teubner, Stuttgart, 1986.

[Hac91] Wolfgang Hackbusch. *Iterative Lösung großer schwachbesetzter Gleichungssysteme*, volume 69 of *Leitfäden der angewandten Mathematik und Mechanik LAMM.* B. G. Teubner, Stuttgart, 1991.

[HBH$^+$05] Michael A. Heroux, Roscoe A. Bartlett, Vicki E. Howle, Robert J. Hoekstra, Jonathan J. Hu, Tamara G. Kolda, Richard B. Lehoucq, Kevin R. Long, Roger P. Pawlowski, Eric T. Phipps, Andrew G. Salinger, Heidi K. Thornquist, Ray S. Tuminaro, James M. Willenbring, Alan Williams, and Kendall S. Stanley. An overview of the trilinos project. *ACM Transactions on Mathematical Software*, 31(3):397–423, 2005.

[HMY02] Van Emden Henson and Ulrike Meier Yang. BoomerAMG: a Parallel Algebraic Multigrid Solver and Preconditioner. *Applied Numerical Mathematics*, 41:155–177, 2002. Also available as technical report UCRL-JC-141495, Lawrence Livermore National Laboratory, March 2001.

[HV01] Van Emden Henson and Panayot S. Vassilevski. Element-Free AMGe: General Algorithms for Computing Interpolation Weights in AMG. *SIAM Journal on Scientific Computing*, 23:629–650, 2001.

[hyp] *hypre:* high performance preconditioners. `http://computation.llnl.gov/casc/linear_solvers/sls_hypre.html`.

[JV01] Jim E. Jones and Panayot S. Vassilevski. AMGe based on Element Agglomeration. *SIAM Journal on Scientific Computing*, 23:109–133, 2001.

[Kah09] Karsten Kahl. *Adaptive Algebraic Multigrid for Lattice QCD Computations.* Dissertation, Bergische Universität Wuppertal, 2009.

[KK97] George Karypis and Vipin Kumar. A Coarse-Grain Parallel Formulation of Multilevel k-way Graph Partitioning Algorithm. In *8th SIAM Conference on Parallel Processing for Scientific Computing*, 1997.

[KK99] George Karypis and Vipin Kumar. A Fast and High Quality Multilevel Scheme for Partitioning Irregular Graph. *SIAM Journal on Scientific Computing*, 20(1):359–392, 1999.

[KS99] Arnold Krechel and Klaus Stüben. Parallel Algebraic Multigrid Based on Subdomain Blocking. Technical Report REP-SCAI-1999-71, GMD, December 1999.

[Liv04] Oren E. Livne. Coarsening by compatible relaxation. *Numerical Linear Algebra with Applications*, 11:205–227, 2004.

[LOS04] Koung Hee Leem, Suely Oliveira, and David Stewart. Algebraic multigrid (AMG) for saddle point systems from meshfree discretizations. *Numerical Linear Algebra with Applications*, 11:293–308, 2004.

[LSAC08] Gregory Larson, Deryl Snyder, David Vanden Abeele, and Tanja Clees. Application of single-level, pointwise algebraic, and smoothed aggregation multigrid methods to direct numerical simulation of incompressible turbulent flows. *Computing and Visualization in Science*, 11:27–40, 2008.

[Lub86] Michael Luby. A simple parallel algorithm for the maximal independent set problem. *SIAM J. Comp.*, 15:1036–1053, 1986.

[Met04] Bram Metsch. Ein paralleles graphenbasiertes algebraisches Mehrgitterverfahren. Diplomarbeit, Institut für Numerische Simulation, Universität Bonn, 2004.

[MM08] Dave A. May and Louis Moresi. Preconditioned iterative methods for Stokes flow problems arising in computational geodynamics. *Physics of the Earth and Planetary Interiors*, 171:33–47, 2008.

[MO08] Scott P. MacLachlan and Cornelis W. Oosterlee. Algebraic Multigrid solvers for complex–valued matrices. *SIAM Journal on Scientific Computing*, 30(3):1548–1571, 2008.

[MPI] MPI Forum. Message Passing Interface (MPI) Forum Home Page. `http://www.mpi-forum.org/`.

[MY04] Ulrike Meier Yang. On the use of relaxation parameters in hybrid smoothers. *Numerical Linear Algebra with Applications*, 11(2–3):155–172, 2004.

[MY06] Ulrike Meier Yang. Parallel Algebraic Multigrid Methods - High Performance Preconditioners. In Are Magnus Bruaset and Alsal Tveito, editors, *Numerical Solution of Partial Differential Equations on Parallel Computers*, volume 51 of *Lecture Notes in Computational Science and Enigineering*, chapter 6, pages 209–236. Springer, 2006.

[MY10] Ulrike Meier Yang. On long-range interpolation operators for aggressive coarsening. *Numerical Linear Algebra with Applications*, 17(2–3):453–472, 2010.

[Not10] Yvan Notay. Algebraic analysis of two-grid methods: The nonsymmetric case. *Numerical Linear Algebra with Applications*, 17(1):7396, 2010.

[Oel01] Daniel Oeltz. Algebraische Mehrgittermethoden für Systeme partieller Differentialgleichungen. Diplomarbeit, Institut für Angewandte Mathematik, Universität Bonn, 2001.

[Pat80] Suhas V. Patankar. *Numerical Heat Transfer and Fluid Flow.* Taylor & Francis, New York, 1980.

[PS72] Suhas V. Patankar and D. Brian Spalding. A calculation procedure for heat and mass transfer in three-dimensional parabolic flows. *International Journal on Heat and Mass Transfer*, 15:1787–1806, 1972.

[PS75] Chris C. Paige and Michael A. Saunders. Solution of Sparse Indefinite Systems of Linear Equations. *SIAM Journal on Numerical Analysis*, 12:617–629, 1975.

[RS87] John Ruge and Klaus Stüben. Algebraic Multigrid. In Stephen F. McCormick, editor, *Multigrid Methods*, Frontiers in Applied Mathematics, chapter 4, pages 73–130. SIAM, Philadelphia, 1987.

[Rug86] John Ruge. AMG for Problems of Elasticity. *Applied Mathematics and Computation*, 19:293–309, 1986.

[SFNMY08] Hans De Sterck, Robert D. Falgout, Joshua W. Nolting, and Ulrike Meier Yang. Distance-two interpolation for parallel algebraic multigrid. *Numerical Linear Algebra With Applications*, 15(2–3):115–139, 2008.

[SMYH06] Hans De Sterck, Ulrike Meier Yang, and Jeffrey J. Heys. Reducing complexity in parallel algebraic multigrid preconditioners. *SIAM Journal on Matrix Analysis and Applications*, 27(4):1019–1039, 2006.

[SS86] Yousef Saad and Martin H. Schultz. GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal Scientific and Statistical Computing*, 7:856–869, 1986.

[SS97] Dongho Shin and John C. Strikwerda. Inf-sup conditions for finite-difference approximations of the stokes equations. *The ANZIAM Journal*, 39(01):121–134, 1997.

[Stü99] Klaus Stüben. Algebraic Multigrid (AMG): An Introduction with Applications. Technical report, GMD - Forschungszentrum Informationstechnik GmbH, March 1999. Also available as part of [TOS01].

[SZ03] Joachim Schöberl and Walter Zulehner. On Schwarz-type Smoothers for Saddle Point Problems. *Numerische Mathematik*, 95(2):377–399, 2003.

[TOS01] Ulrich Trottenberg, Cornelis Oosterlee, and Anton Schüller. *Multigrid.* Academic Press, London, 2001.

[TT00] Ray S. Tuminaro and Charles Tong. Parallel smoothed aggregation multigrid: Aggregation strategies on massively parallel machines. In Jed Donnelley, editor, *Proceedings of the 2000 ACM/IEEE conference on Supercomputing*, 2000.

[Van86] S. Pratap Vanka. Block-Implicit Multigrid Solution of Navier-Stokes Equations in Primitive Variables. *Journal of Computational Physics*, 65:138–158, 1986.

[VBM01] Peter Vanek, Marian Brezina, and Jan Mandel. Convergence Analysis of Algebraic Multigrid Based on Smoothed Aggregation. *Numerische Mathematik*, pages 559–579, 2001.

[VDR84] Jeffrey P. Van Doormaal and George D. Raithby. Enhancements of the SIMPLE method for prediciting incompressible fluid flows. *Numerical Heat Transfer: An International Journal of Computation and Methodology*, 7:147–163, 1984.

[VMB94] Peter Vanek, Jan Mandel, and Marian Brezina. Algebraic multigrid on unstructered meshes. Technical Report 34, UCD/CCM, 1994.

[VMB96] Peter Vanek, Jan Mandel, and Marian Brezina. Algebraic multigrid based on smoothed aggregation for second and fourth order elliptic problems. *Computing*, pages 179–196, 1996.

[Wab03] Markus Wabro. *Algebraic Multigrid Methods for the Numerical Simulation of the Incompressible Navier-Stokes Equations.* Dissertation, Institut für Numerische Mathematik, Johannes Kepler Universität, Linz, 2003.

[Wab04] Markus Wabro. Coupled algebraic multigrid methods for the Oseen problem. *Computing and Visualization in Science*, 7:141–151, 2004.

[Wab06] Markus Wabro. AMGe-Coarsening Strategies and Application to the Oseen Equations. *SIAM Journal on Scientific Computing*, 27:2077–2097, 2006.

[Wit89] Gabriel Wittum. Multi-Grid Methods for Stokes and Navier-Stokes Equations. *Numerische Mathematik*, 54:543–563, 1989.

[Wit90] Gabriel Wittum. On the Convergence of Multi-Grid Methods with Transforming Smoothers. *Numerische Mathematik*, 57:15–38, 1990.

[WY09] Roman Wienands and Irad Yavneh. Collocation coarse approximation in multigrid. *SIAM Journal on Scientific Computing*, 31(5):3643–3660, 2009.

[Zul00] Walter Zulehner. A Class of Smoothers for Saddle Point Problems. *Computing*, 65:227–246, 2000.

[Zul02] Walter Zulehner. Analys of Iterative Methods for Saddle Point Problems: A Unified Approach. *Mathematics of Computation*, 71(238):479–505, 2002.