Johann Radon Institute for
Computational and Applied Mathematics
Austrian Academy of Sciences (ÖAW)

# Preconditioners for linear systems arising from the finite volume discretization of the incompressible Navier-Stokes equations

S. Engleder

RICAM-Report 2016-10

# Preconditioners for linear systems arising from the finite volume discretization of the incompressible Navier-Stokes equations

Sarah Engleder*†

Tuesday 15th March, 2016

**Abstract**

This article deals with the numerical solution of the incompressible Navier-Stokes equations using the finite volume method. A coupled approach for the discrete problem is used, which requires the solution of a block system in every non-linear iteration. The unknowns are velocity and pressure. We analyze the properties of the linear block system. Two different preconditioner variants, which are used in combination with a Krylov subspace method, are proposed. One preconditioner is based on an approximate block factorization of the linear system. The second preconditioner belongs to the class of algebraic multigrid methods (AMG). We introduce a special AMG version, which is suited for this type of application. The two preconditioner variants are compared for a number of benchmark problems.

## 1 Introduction

Nowadays many engineering development processes rely on the use of simulation software. In automotive industry, in many branches such as engine development, CFD (computational fluid dynamics) software is extensively used. Usually every CFD simulation requires the numerical solution of the Navier-Stokes equations. For the discretization of this system of partial differential equations in commercial codes the co-located finite volume method is frequently employed. This method leads to a non-linear set of equations, the unknowns are velocity and pressure. For the solution of the non-linear problem a fixed-point method is applied. The linearized problem, which has to be solved in every fixed-point iteration, is handled mainly with segregated approaches such as the SIMPLE algorithm (see e.g. [16, 10]). SIMPLE has turned out to be very powerful for real engineering applications, which are very complex and may involve complicated geometries. An advantage of the SIMPLE method is that the resulting linear systems possess properties, which can be exploited by numerical solution algorithms. For example the system matrix of the pressure correction equation is positive definite and symmetric in the incompressible case. Consequently algebraic multigrid methods, which were developed for this type of systems, can be used (see e.g. [7]). Segregated approaches are also preferred due to the low memory requirements compared to a coupled approach.

*Johann Radon Institute for Computational and Applied Mathematics (RICAM), Altenbergerstrasse 69, 4040 Linz, Austria

†MathConsult GmbH, Altenbergerstrasse 69, 4040 Linz, Austria

A drawback of the SIMPLE method is that it converges slowly and the convergence depends strongly on the choice of under-relaxation factors. Choosing too small relaxation parameters may slow down the convergence, choosing the relaxation parameters too large may have the effect that the non-linear iteration does not converge or even diverge. Choosing these factors involves a lot of experience and know-how of the user.

As nowadays memory is no longer a limiting factor, coupled approaches are more often considered (see e.g. [5, 3]). Using a coupled formulation means that the linearized problem is solved for velocity and pressure simultaneously. The aim is to reduce the number of non-linear iterations and thus to reduce the computation time and to achieve a more accurate solution. Moreover a method, which needs less non-linear iterations but leads to larger linear systems, makes it possible to achieve a better parallel performance. Also GPU acceleration of the linear solver is more efficient in this case, see [9]. A difficulty is that the coupled linear system is usually harder to solve than the linear systems in the SIMPLE iteration and the matrix vector multiplication is more expensive. This means that the number of non-linear iterations needed by the coupled solver has to be significantly smaller than the number of SIMPLE iterations. However, one has to bear in mind that normally also other physical quantities are involved in the non-linear iteration process such as energy, turbulence quantities, etc. If one of these other quantities is converging slowly this slows down the whole iteration process. Also bad mesh quality can slow down the whole convergence or produce non-physical oscillations in the solution. In this case the benefit due to solving the coupled system might not be so big or the simulation using the coupled solver might even take longer than with the segregated one.

In an industrial application a CFD simulation comprises a large number of degrees of freedom. Therefore using Krylov subspace methods in combination with a suitable preconditioner for the solution of linear systems is very common. For preconditioning we have several methods at hand. There are preconditioners, which are based on incomplete factorization methods, see e.g. [1]. Another important class of preconditioners are so-called block preconditioners. Their derivation is based on an approximate block factorization of the matrix and suitable approximations of the Schur complement. Finally Algebraic Multigrid methods represent another relevant class of preconditioners. In this paper we focus on the latter two classes of preconditioners. We introduce a block preconditioner, which is suited for our type of matrix. The advantage of the block preconditioner is the easy implementation, a drawback is that a good approximation of the Schur complement is needed. Algebraic multigrid methods were mainly developed for symmetric positive systems. Transferring the AMG approch to indefinite block systems requires some special considerations. Such methods for block systems, which originate from the discretization of PDE-systems, have been developed in [4]. AMG methods for finite element discretizations of the Stokes problem have been analyzed in [22, 14]. For general investigations of stability of AMG methods ('classical' interpolation and smoothed aggregation) for the Stokes problem we refer to [23, 24]. We work with 'unsmoothed' aggregation methods, such as 'pairwise aggregation' ([13, 15]), which lead to a constant interpolation. This technique is favorable since the AMG setup is fast. Further in combination with an ILU(0) smoother this method shows good convergence and serves as a robust preconditioner for Krylov subspace methods.

This paper is organized as follows: First we introduce a finite volume discretization for the incompressible Navier-Stokes equations. The discretization gives rise to a non-linear system of equations with a block structure. The linearized problem is derived and the properties of the

linear system are investigated. In Section 3 we introduce different strategies for the solution of the linear system. A block preconditioner and a 'monolithic' AMG approach is proposed. In the last section the different preconditioners are compared on a number of benchmark problems. The simulations were done with the CFD software AVL FIRE®[1], where the presented methods have been implemented.

# 2 Finite Volume Methods

## 2.1 Mathematical Model

The governing equations for laminar flow for Newtonian fluids are the continuity equation (mass conservation)

$$\frac{\partial}{\partial t}\rho(t,x) + \mathrm{div}(\rho(t,x)\mathbf{u}(t,x)) = 0 \tag{1}$$

and the momentum equation (conservation of momentum)

$$\frac{\partial}{\partial t}(\rho u^k(t,x)) + \mathrm{div}(\rho u^k(t,x)\mathbf{u}(t,x)) - \mathrm{div}(\mu\nabla u^k(t,x)) + \frac{\partial}{\partial x_k}p(t,x) = f^k(t,x) \tag{2}$$

for $k = 1, 2, 3$. For incompressible stationary flow the equations reduce to

$$\rho\mathrm{div}(u^k(x)\mathbf{u}(x)) - \mathrm{div}(\mu(x)\nabla u^k(x)) + \frac{\partial}{\partial x_k}p(x) = f^k(x), \quad k = 1,\dots 3 \tag{3}$$

$$\rho\mathrm{div}(\mathbf{u}(x)) = 0. \tag{4}$$

Since most real-life applications cannot be simulated as laminar flow by direct numerical simulation, we need to take into account turbulence models. Often the $k-\varepsilon$ model (see [12]) is used to describe turbulent flow. For this model two additional transport equations for the turbulent kinetic energy $k$ and the turbulent dissipation $\varepsilon$ have to be solved. Using these two quantities the turbulent viscosity can be computed by $\mu_t = C_\mu\rho\frac{k^2}{\varepsilon}$.
The choice of boundary conditions depends on the physical problem, which we want to solve. In CFD codes usually a variety of boundary conditions is available, they can be in general classified as inlet, outlet and wall boundary conditions. As describing all types of boundary conditions would go beyond the content of this paper we only present a few of them. We introduce a decomposition of the boundary of the computational domain into three parts $\overline{\partial\Omega} = \overline{\Gamma_{\mathrm{Inlet}}} \cup \overline{\Gamma_{\mathrm{Wall}}} \cup \overline{\Gamma_{\mathrm{Outlet}}}$. If the velocity profile is known at the inlet, one can set

$$\mathbf{u}(x) = \mathbf{u}_b, \quad x \in \Gamma_{\mathrm{Inlet}}. \tag{5}$$

For the outlet typically 'static pressure' boundary conditions are assumed:

$$p(x) = p_{\mathrm{static}}, \quad x \in \Gamma_{\mathrm{Outlet}}. \tag{6}$$

[1] http://www.avl.com

To bound fluid regions, wall boundary conditions have to be set, for a no-slip boundary condition we have the condition (5) with $\mathbf{u}_b = 0$.

## 2.2 Finite Volume Discretization

Let us consider the computational domain $\Omega$ and introduce a decomposition into a finite number of control volumes

$$\Omega = \bigcup_{P=1}^{N} \Omega_P.$$

The control volumes are typically tetrahedra or hexahedra. Every control volume is bounded by a finite number of plane faces $\partial\Omega_P = \bigcup_{j=1}^{n_f(P)} A_j$, where $n_f(P)$ denotes the number of neighboring faces.[2]
If $A_j \cap \partial\Omega = \emptyset$, the face $A_j$ is called internal face, faces with $A_j \subset \partial\Omega$ are named boundary face. A cell, which has at least one boundary face, is called boundary cell, a cell, which is surrounded only by internal faces, is called internal cell. The total number of all internal faces is denoted by $M$. We denote the vector joining two cell centers $P$ and $P_j$ by $\mathbf{d}_j$. The face surface vector of an inner face is denoted by $\mathbf{A}_j$, the surface vector of a boundary face is denoted by $\mathbf{A}_b$. Note that the face surface vector is not normalized i.e. the norm of the face surface vector corresponds to the face area $\|\mathbf{A}_j\|_2 = \Delta_j$. The face based normal vector $\mathbf{A}_j$ and the vector $\mathbf{d}_j$ have a fixed direction, i.e. to every face $j$ a pair of cells ($P$,$P_j$) is associated, $\mathbf{A}_j$ then points outside of cell $P$ and $\mathbf{d}_j$ points from $P$ to cell $P_j$. Further we introduce face based flux values $\hat{f}_j \approx \mathbf{u}(\hat{x}_j) \cdot \mathbf{A}_j$, where $\hat{x}_j$ denotes the center of face $A_j$. As a co-located method is used the degrees of freedom are identified with the cell centers, a link between the face values $\hat{\varphi}_j$ and the cell center values has to be made. This is done by applying the interpolation

$$\hat{f}_j = \omega_j f_P + (1-\omega_j) f_{P_j}, \quad \omega_j = \frac{|\hat{x}_j - x_{P_j}|}{|\hat{x}_j - x_P| + |\hat{x}_j - x_{P_j}|}. \tag{7}$$

We assume that the mesh belongs to a class of meshes with certain regularity properties, i.e. there exist constants $l, \varepsilon > 0$ such that for all faces $j$ it holds

$$\frac{\mathbf{A}_j \cdot \mathbf{d}_j}{\|\mathbf{A}_j\|\|\mathbf{d}_j\|} > c \quad \text{and} \quad |\frac{1}{2} - \omega_j| \leq \varepsilon. \tag{8}$$

The first condition is related to the mesh-orthogonality. The second condition reflects the 'uniformity' of the mesh and can be associated to the aspect ratio.
The key idea of the finite volume method is to consider the integral form of the conservation laws on every control volume separately and approximate the integrals by discrete values. The

[2]For ease of notation and if it is clear from the context we will just write $n_f$ instead of $n_f(P)$ for the number of neighboring faces for the cell $P$.

integral form of the momentum conservation (3) on a control volume $\Omega_P$ is

$$\int_{\Omega_P} \frac{\partial}{\partial t}(\rho u^k(x))dx + \int_{\Omega_P} \mathrm{div}(\rho u^k(x)\mathbf{u}(x))dx - \int_{\Omega_P} \mathrm{div}(\mu \nabla u^k(x))dx + \int_{\Omega_P} \frac{\partial}{\partial x_k} p(x)dx = \int_{\Omega_P} f^k(x)dx. \tag{9}$$

Integrating the continuity equation (4) for incompressible stationary flow yields

$$\int_{\Omega_P} \frac{\partial}{\partial t}\rho(x)dx + \int_{\Omega_P} \mathrm{div}(\rho \mathbf{u}(x))dx = 0. \tag{10}$$

The co-located finite volume method identifies the discrete solution for all physical unknowns with cell based values. Hence the integrals of the continuous variables have to be approximated by expressions involving its discrete counterpart. In the next sections we will shortly summarize the basic approximation schemes used for the discretization of (9) and (10).

### 2.2.1 Gradient Discretization

For the gradient discretization we apply Gauß theorem and approximate the gradient by interpolated cell values:

$$\begin{aligned}\int_{\Omega_P} \frac{\partial}{\partial x_i}\varphi(x)dx = \sum_{j=1}^{n_f} \int_{A_j} \varphi(x)\mathbf{e}_i \cdot d\mathbf{A}_j &\approx \sum_{j=1}^{n_{f,i}} \hat{\varphi}_j A_j^i + \sum_{j=1}^{n_b} \hat{\varphi}_b A_b^i \\ &\approx \sum_{j=1}^{n_{f,i}} \left(\omega_j \varphi_P + (1-\omega_j)\varphi_{P_j}\right) A_j^i + \sum_{b=1}^{n_b} \hat{\varphi}_b A_b^i,\end{aligned}$$

where $A_j^i = \mathbf{e}_i \cdot \mathbf{A}_j$ and $n_{f,i}$ denotes the number of internal faces and $n_b$ the number of boundary faces. The discrete version of the integral over divergence and gradient can be expressed as a matrix-vector multiplication

$$\int_{\Omega_P} \mathrm{div}\mathbf{u}(x)dx \approx [\mathbf{C}\mathbf{u}]_P, \quad \int_{\Omega_P} \nabla p(x)dx \approx [\mathbf{M}p]_P \tag{11}$$

with $\mathbf{M} = [M_1^T, M_2^T, M_3^T]^T \in \mathbb{R}^{3N\times N}$ and $\mathbf{C} = [C_1, C_2, C_3] \in \mathbb{R}^{N\times 3N}$. If $P$ is an inner cell we can write

$$M^i(P,P_j) = \omega_j A_j^i, \quad M^i(P_j,P) = -(1-\omega_j)A_j^i, \quad M^i(P,P) = \sum_{j=1}^{n_f} \omega_j A_j^i. \tag{12}$$

and we have $C^i(P,P_j) = M^i(P,P_j)$, $C^i(P_j,P) = M^i(P_j,P)$ and $C^i(P,P) = M^i(P,P)$. For a boundary cell we have to distinguish two cases, if either the velocity or the pressure is known on the boundary face. For the boundary condition (5) we know $\hat{u}_b A_b^i$ and put it on the right hand side of the discrete momentum equations. For the matrix entry $C^i(P,P)$ only the internal faces

are considered then. For the pressure gradient matrix we set

$$M^i(P,P) = \sum_{b=1}^{n_b} A_b^i + \sum_{j=1}^{n_{f,i}} \omega_j A_j^i.$$

Similarly for the pressure boundary condition (6) we put the known boundary pressure $\hat{p}_b$ on the right hand side and consider only the internal faces for the calculation of $M^i(P,P)$. For the velocity divergence we have

$$C^i(P,P) = \sum_{b=1}^{n_b} A_b^i + \sum_{j=1}^{n_{f,i}} \omega_j A_j^i.$$

We see that depending on the boundary conditions we could have $\text{diag}(C^i) \neq \text{diag}(M^i)$. The considerations above can also be used for evaluating the cell based values gradient values $\varphi_P$ by using the approximation $\nabla\varphi(x_P) \approx \frac{1}{|\Omega_P|}\int_{\Omega_P} \nabla\varphi(x)dx$. Another common method for cell based gradient calculation is based on the minimization of the Taylor expansion

$$\min \sum_{j=1}^{n_f} \|\varphi_P - \varphi_{P_j} - \mathbf{g}_P \cdot \mathbf{d}_j\|_2^2,$$

where $\mathbf{g}_P$ is computed via a least squares approximation.

### 2.2.2 Diffusive Term

The diffusive term in (9) can be reformulated by Gauß theorem

$$-\int_{\Omega_P} \text{div}(\mu\nabla u^k)(x)dx = -\sum_{j=1}^{n_f} \int_{A_j} \mu\nabla u^k(x) \cdot d\mathbf{A}_j. \tag{13}$$

Let us first consider internal cells $P$. As we need an approximation of the velocity gradient in the direction of the face normal $\nabla u^k \cdot \mathbf{A}_j$, we introduce the decomposition

$$\mathbf{A}_j = \frac{\|\mathbf{A}_j\|^2}{\mathbf{A}_j \cdot \mathbf{d}_j}\mathbf{d}_j + \mathbf{t}, \quad \text{with} \quad \mathbf{t} = \mathbf{A}_j - \frac{\|\mathbf{A}_j\|^2}{\mathbf{A}_j \cdot \mathbf{d}_j}\mathbf{d}_j.$$

$\mathbf{t}$ lies in the plane of face $j$ and is perpendicular to the normal vector $\mathbf{A}_j$. With this we can rewrite the face normal derivative

$$\nabla u^k(x) \cdot \mathbf{A}_j = \frac{\|\mathbf{A}_j\|^2}{\mathbf{A}_j \cdot \mathbf{d}_j}\nabla u^k(x) \cdot \mathbf{d}_j + \nabla u^k(x) \cdot \mathbf{t}.$$

The first part of the sum can be approximated by a differencing scheme:

$$\nabla u^k(x_j) \cdot \mathbf{d}_j \approx u^k(x_{P_j}) - u^k(x_P)$$

For the tangential part we use the approximation

$$\nabla u^k(x)|_{\hat{x}_j} \cdot \mathbf{t} \approx \frac{1}{2}(\mathbf{g}_P^k + \mathbf{g}_{P_j}^k) \cdot \mathbf{t},$$

where $\mathbf{g}_P^k$ and $\mathbf{g}_{P_j}^k$ are approximations of $\nabla u^k(x_P)$ and $\nabla u^k(x_{P_j})$. Summarizing this we get

$$\nabla u^k(x)|_{\hat{x}_j} \cdot \mathbf{A}_j \approx \frac{1}{2}(\mathbf{g}_P^k + \mathbf{g}_{P_j}^k) \cdot \mathbf{A}_j + \frac{\|\mathbf{A}_j\|^2}{\mathbf{A}_j \cdot \mathbf{d}_j} \left( (u_{P_j}^k - u_P^k) - \frac{1}{2}(\mathbf{g}_P^k + \mathbf{g}_{P_j}^k) \cdot \mathbf{d}_j \right). \tag{14}$$

With this approximation we get a discretization of the term (13) into $-\mu[A^{\text{dif}}u^k](P)+\mu[A^{\text{crodif}}u^k](P)$. As the cross-diffusion term is treated implicitly an explicit representation of the matrix $A^{\text{crodif}}$ is not needed. The diffusion matrix entries can be computed by

$$A^{\text{dif}}(P,P_j) = A^{\text{dif}}(P_j,P) = -\frac{\|\mathbf{A}_j\|^2}{\mathbf{A}_j \cdot \mathbf{d}_j}, \quad A^{\text{dif}}(P,P) = -\sum_{j=1}^{n_f} A^{\text{dif}}(P,P_j)$$

For a cell, which is bounded by a boundary face with center $\hat{x}_b$, we use the approximation $\nabla u^k(x)|_{\hat{x}_b} \approx \mathbf{g}_P$ and get

$$\nabla u^k(x)|_{\hat{x}_b} \cdot \mathbf{A}_b \approx \mathbf{g}_P \cdot \mathbf{A}_b + \frac{\|\mathbf{A}_b\|^2}{\mathbf{A}_b \cdot \mathbf{d}_b} \left( (u_b^k - u_P^k) - \mathbf{g}_P^k \cdot \mathbf{d}_b \right).$$

If $\mathbf{u}_b$ is known, i.e. we have a boundary condition like (5) we have to add $\frac{\|\mathbf{A}_b\|^2}{\mathbf{A}_b \cdot \mathbf{d}_j}$ to the diagonal entry $A^{\text{dif}}(P,P)$, where $P$ is the corresponding boundary cell. The cross diffusion part is again moved to the right hand side.
The entries of the matrix $A^{\text{dif}}$ depend only on geometric properties of the mesh. Due to the assumptions (8) on the mesh regularity, it can be easily seen that the matrix is diagonal dominant and irreducible, hence it can be concluded that $A^{\text{dif}}$ is positive semi-definite. If $u^k$ is fixed on a part of the boundary, e.g. we have the boundary condition (5), then $A^{\text{dif}}$ is positive definite.

### 2.2.3 Convective Term

The choice of the discretization scheme for the non-linear convective part of the momentum conservation has a major influence on the result. There is a variety of discretization techniques available. The simplest is the upwind discretization, which approximates the face value $\hat{\mathbf{u}}_j$ by the values of one of the neighboring cells depending on the direction of the flux:

$$\int_{\Omega_P} \frac{\partial}{\partial x_i}(\rho u^k u_i) dx = \sum_{j=1}^{n_f} \int_{A_j} \rho u^k \mathbf{u} \cdot d\mathbf{A}_j \approx \sum_{j=1}^{n_f} \rho \hat{f}_j \hat{u}_j^k. \tag{15}$$

Depending on the sign of the flux we set

$$\hat{u}_j^{k,\text{upw}} = \begin{cases} u_P^k & \text{for } \hat{f}_j \geq 0, \\ u_{P_j}^k & \text{for } \hat{f}_j < 0. \end{cases}$$

Defining the positive and negative flux

$$\hat{f}_j^+ = \begin{cases} \hat{f}_j & \text{if } \hat{f}_j \geq 0 \\ 0 & \text{if } \hat{f}_j < 0 \end{cases}, \qquad \hat{f}_j^- = \begin{cases} 0 & \text{if } \hat{f}_j \geq 0 \\ \hat{f}_j & \text{if } \hat{f}_j < 0 \end{cases}$$

we can write discrete version of (15) as matrix vector multiplication, with the matrix

$$A^{\text{con}}(P, P_j) = \rho \hat{f}_j^-, \quad A^{\text{con}}(P_j, P) = -\rho \hat{f}_j^+, \quad A^{\text{con}}(P, P) = \sum_{j=1}^{n_f} \rho \hat{f}_j^+.$$

If $P$ is a boundary cell and the velocity on the boundary is given (e.g. (5)), we have according to the upwind scheme

$$A^{\text{con}}(P, P) = -\rho \hat{f}_b^- + \sum_{j=1}^{n_f} \rho \hat{f}_j^+,$$

further we have to add $-\hat{f}_b^- \mathbf{u}_b$ on the right hand side.
The matrix $A^{\text{con}}$ is in general non-symmetric. It can be easily seen that the off-diagonal entries of the convection matrix $A^{\text{con}}$ are always negative, the diagonal entries are always positive.

### 2.2.4 Face Interpolation

Summarizing the previous results the finite volume discretization of the incompressible steady-state Navier-Stokes equations leads to a non-linear system of equations

$$\begin{pmatrix} \mathbf{A}(\mathbf{u}) & \mathbf{M} \\ \mathbf{C} & 0 \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ p \end{pmatrix} = \begin{pmatrix} \mathbf{f}(\mathbf{u}, p) \\ g(\mathbf{u}) \end{pmatrix}, \tag{16}$$

where we have set $\mathbf{A}(\mathbf{u}) = \mu \mathbf{A}^{\text{dif}} + \mathbf{A}^{\text{con}}(\mathbf{u}) \in \mathbb{R}^{3N \times 3N}$, which is a block diagonal matrix, with the diagonal blocks $\mu A^{\text{dif}} + A^{\text{con}}(\mathbf{u})$. The right hand side contains cross diffusion terms, sources and terms, which stem from boundary conditions. Also if other discretization schemes than the upwind method for the convective terms are used, the 'non-upwind'-part is then treated implicitly and is contained in $\mathbf{f}$. It is well known that for (16) it cannot be assumed in general that it is uniquely solvable, e.g. if there exists a $p^*$ with $\mathbf{M}p^* = 0$. For a uniform mesh of a cube such a $p^*$ can be easily constructed. Then we have $\omega_j = \frac{1}{2}$, which leads to

$$M^i \underline{p}^* = \frac{1}{2} \frac{1}{V} \sum_{j=1}^{n_f} (p_P^* + p_{P_j}^*) A_j^i = 0.$$

For a $p^*$ with a checkerboard pattern this condition is fulfilled. To overcome this problem a special interpolation for the flux has been introduced by Rhie and Chow [18]:

$$\hat{f}_j^* = (\omega_j \mathbf{u}_P + (1 - \omega_j) \mathbf{u}_{P_j}) \cdot \mathbf{A}_j - \frac{1}{2} \left( \frac{|\Omega_P|}{a_P} + \frac{|\Omega_{P_j}|}{a_{P_j}} \right) \frac{\|\mathbf{A}_j\|_2^2}{\mathbf{A}_j \cdot \mathbf{d}_j} \left( (p_{P_j} - p_P) - \frac{1}{2} (\mathbf{g}_P + \mathbf{g}_{P_j}) \cdot \mathbf{d}_j \right) \tag{17}$$

where $\mathbf{g}_P$ and $\mathbf{g}_{P_j}$ are approximations for the gradient in the cell center and $a_P = A(P,P)$. Using the Rhie-Chow representation we get the following discrete version of the continuity equation:

$$\mathbf{C}\mathbf{u} + A_p p - \tilde{A}_p p = g(\mathbf{u}) \tag{18}$$

with

$$A_p(P,P_j) = A_p(P_j,P) = -\frac{1}{2}\left(\frac{|\Omega_P|}{a_P} + \frac{|\Omega_{P_j}|}{a_{P_j}}\right)\frac{\|\mathbf{A}_j\|_2^2}{\mathbf{A}_j \cdot \mathbf{d}_j}, \quad A_p(P,P) = -\sum_{j=1}^{n_f} A_p(P,P_j). \tag{19}$$

The term $\tilde{A}_p p$ is treated implicitly, hence no matrix representation of $\tilde{A}_p$ is needed. Inserting this into equation (16) we get

$$\begin{pmatrix} \mathbf{A}(\mathbf{u}) & \mathbf{M} \\ \mathbf{C} & A_p \end{pmatrix}\begin{pmatrix} \mathbf{u} \\ p \end{pmatrix} = \begin{pmatrix} \mathbf{f}(\mathbf{u},p) \\ g(\mathbf{u}) + \tilde{A}_p p \end{pmatrix} = \begin{pmatrix} \mathbf{f}(\mathbf{u},p) \\ \tilde{g}(\mathbf{u},p) \end{pmatrix}. \tag{20}$$

The matrix $A_p$ arising from the Rhie-Chow interpolation is symmetric and positive semi-definite due to (8). Clearly we have $\mathrm{Ker}(A_p) = \underline{1}$ if no pressure boundary condition is given, hence $p$ is only defined up to a constant. We can always get a positive definite matrix $A_p$ by either fixing the pressure in a cell or by introducing a stabilization:

$$A_{p,s} = A_p + a\underline{1} \cdot \underline{1}^T, \quad a > 0. \tag{21}$$

### 2.2.5 Solution of the non-linear system of equations

In the steady case the solution of the Navies-Stokes equations boils down to solving a non-linear system of equations. In the time dependent case we have an additional time stepping loop. The discrete system is the same as for the steady case, only the time derivative term adds additional positive entries on the diagonal of $\mathbf{A}$. More precisely we have $\mathbf{A} = \mu\mathbf{A}^{\mathrm{dif}} + \mathbf{A}^{\mathrm{con}}(\mathbf{u}) + \mathbf{A}_t$. $\mathbf{A}_t$ is a block diagonal matrix, we denote the diagonal blocks with $A_t$, where

$$A_t(P,P_j) = A_t(P_j,P) = 0, \quad A_t(P,P) = \rho\frac{|\Omega_P|}{\Delta t}. \tag{22}$$

Let us now assume that we have either a steady case or one time step, then we have to solve a non-linear system of equations:

$$\underbrace{\begin{pmatrix} \mathbf{A}(\mathbf{u}) & \mathbf{M} \\ \mathbf{C} & A_p \end{pmatrix}}_{\mathcal{A}(\mathbf{u})}\begin{pmatrix} \mathbf{u} \\ p \end{pmatrix} = \begin{pmatrix} \mathbf{f}(\mathbf{u},p) \\ \tilde{g}(\mathbf{u},p) \end{pmatrix}. \tag{23}$$

This system is usually solved using a fixed-point iteration. For this method start values $\mathbf{u}_0, p_0$ have to be chosen. One iteration then requires the solution of the linear system

$$\begin{pmatrix} \mathbf{A}(\mathbf{u}_n) & \mathbf{M} \\ \mathbf{C} & A_p \end{pmatrix}\begin{pmatrix} \mathbf{u}_{n+1} \\ p_{n+1} \end{pmatrix} = \begin{pmatrix} \mathbf{f}(\mathbf{u}_n,p_n) \\ \tilde{g}(\mathbf{u}_n,p_n) \end{pmatrix}. \tag{24}$$

Solving the full block system (24) is time consuming and requires a lot of memory. For this reason segregated approaches like the SIMPLE method ([16, 10]) have been developed. The SIMPLE method can be derived by applying the concept of a preconditioned Richardson iteration to (23)

$$\begin{pmatrix}\mathbf{u}_{n+1}\\ p_{n+1}\end{pmatrix} = \begin{pmatrix}\mathbf{u}_n\\ p_n\end{pmatrix} + \mathcal{P}^{-1}\left(\begin{pmatrix}\mathbf{f}(\mathbf{u}_n, p_n)\\ \tilde{g}(\mathbf{u}_n, p_n)\end{pmatrix} - \mathcal{A}(\mathbf{u}_n)\begin{pmatrix}\mathbf{u}_n\\ p_n\end{pmatrix}\right).$$

For the SIMPLE iteration we choose

$$\mathcal{P}_{\text{SIMPLE}} = \begin{pmatrix}\mathbf{A}(\mathbf{u}_n) & 0\\ \mathbf{C} & A_p\end{pmatrix}\begin{pmatrix}I & \mathbf{A}_D(\mathbf{u}_n)^{-1}\mathbf{M}\\ 0 & I\end{pmatrix}. \tag{25}$$

$\mathcal{P}$ is an approximation to the exact block-decomposition of $\mathcal{A}$:

$$\mathcal{A} = \begin{pmatrix}\mathbf{A}(\mathbf{u}_n) & 0\\ \mathbf{C} & S\end{pmatrix}\begin{pmatrix}I & \mathbf{A}(\mathbf{u}_n)^{-1}\mathbf{M}\\ 0 & I\end{pmatrix} \tag{26}$$

with the Schur-Complement $S = A_p - \mathbf{M}\mathbf{A}^{-1}\mathbf{C}$. Thus one Richardson iteration requires the approximate solution of the linear system

$$\begin{pmatrix}\mathbf{A}(\mathbf{u}_n) & 0\\ \mathbf{C} & A_p\end{pmatrix}\begin{pmatrix}I & \mathbf{A}_D(\mathbf{u}_n)^{-1}\mathbf{M}\\ 0 & I\end{pmatrix}\begin{pmatrix}\mathbf{u}'\\ p'\end{pmatrix} = \left(\begin{pmatrix}\mathbf{f}(\mathbf{u}_n, p_n)\\ \tilde{g}(\mathbf{u}_n, p_n)\end{pmatrix} - \mathcal{A}(\mathbf{u}_n)\begin{pmatrix}\mathbf{u}_n\\ p_n\end{pmatrix}\right). \tag{27}$$

Setting $\mathbf{u}^* = \mathbf{u}' - \mathbf{A}_D(\mathbf{u}_n)^{-1}\mathbf{M}p'$ it boils down to solving a block triangular system:

$$\begin{pmatrix}\mathbf{A}(\mathbf{u}_n) & 0\\ \mathbf{C} & A_p\end{pmatrix}\begin{pmatrix}\mathbf{u}^*\\ p'\end{pmatrix} = \left(\begin{pmatrix}\mathbf{f}(\mathbf{u}_n, p_n)\\ \tilde{g}(\mathbf{u}_n, p_n)\end{pmatrix} - \mathcal{A}(\mathbf{u}_n)\begin{pmatrix}\mathbf{u}_n\\ p_n\end{pmatrix}\right). \tag{28}$$

In most cases the SIMPLE iteration doesn't converge or even diverges without relaxation. Mostly two relaxation methods are applied: In the discrete momentum equations the diagonal entries are multiplied with the relaxation factor $\frac{1}{\alpha_u}$, where $0 < \alpha_u < 1$. For the pressure updates damping is applied $p_{n+1} = p_n + \alpha_p p'$, with $0 < \alpha_p < 1$. The major part of the computation time in the SIMPLE algorithm is taken by the solution of the four linear systems. The solution of the 'predictor systems' for $\mathbf{u}^*$ is usually less time consuming since the system matrix is well conditioned and diagonal dominant due to under-relaxation. Solving the pressure correction equation is in general the most expensive part regarding computational cost.
A drawback of the SIMPLE method is that it converges slowly. A reason is that the SIMPLE approximation for the solution of the linear system appearing in the fixed-point iteration might be inaccurate. The idea is now to speed up convergence by solving the coupled system in one fixed-point iteration. For this approach we set $\mathcal{P} = \mathcal{A}$, the new iterate is

$$\begin{pmatrix}\mathbf{u}_{n+1}\\ p_{n+1}\end{pmatrix} = \begin{pmatrix}\mathbf{u}_n\\ p_n\end{pmatrix} + \mathcal{A}^{-1}(\mathbf{u}_n)\left(\begin{pmatrix}\mathbf{f}(\mathbf{u}_n, p_n)\\ \tilde{g}(\mathbf{u}_n, p_n)\end{pmatrix} - \mathcal{A}(\mathbf{u}_n)\begin{pmatrix}\mathbf{u}_n\\ p_n\end{pmatrix}\right). \tag{29}$$

In practice $\mathcal{A}$ is not inverted exactly but the linear system

$$\mathcal{A}(\mathbf{u}_n)\begin{pmatrix}\mathbf{u}'\\ p'\end{pmatrix} = \begin{pmatrix}\mathbf{f}(\mathbf{u}_n, p_n)\\ \tilde{g}(\mathbf{u}_n, p_n)\end{pmatrix} - \mathcal{A}(\mathbf{u}_n)\begin{pmatrix}\mathbf{u}_n\\ p_n\end{pmatrix}$$

is solved up to a certain accuracy. For steady convection dominated flows the non-linear iteration (29) may not converge, therefore we add the stabilization term $\alpha\mathbf{D}$:

$$\underbrace{\begin{pmatrix}\mathbf{A}(\mathbf{u}_n)+\alpha\mathbf{D} & \mathbf{M}\\ \mathbf{C} & A_p\end{pmatrix}}_{\mathcal{A}_\alpha(\mathbf{u}_n)}\begin{pmatrix}\mathbf{u}'\\ p'\end{pmatrix} = \begin{pmatrix}\mathbf{f}(\mathbf{u}_n,p_n)\\ \tilde{g}(\mathbf{u}_n,p_n)\end{pmatrix} - \mathcal{A}(\mathbf{u}_n)\begin{pmatrix}\mathbf{u}_n\\ p_n\end{pmatrix} \tag{30}$$

where $\mathbf{D}$ is a diagonal matrix and $\alpha > 0$ a relaxation parameter. We set $\mathbf{A}_\alpha = \mathbf{A} + \alpha\mathbf{D}$. A possible choice could be $\mathbf{D} = \mathrm{diag}(\mathbf{A})$. Another possibility would be to add a pseudo time term. For non-steady problems the stabilization is not necessary since the diagonal matrix $A_t$ coming from the time discretization already acts as a stabilization. The positive effect of this stabilization is that the diagonal dominance of $\mathbf{A}$ is improved and the linear system becomes easier to solve, the drawback is that the bigger $\alpha$ is chosen the slower the convergence will be. Damping of the pressure updates is not necessary in the coupled approach.

# 3 The solution of the linear system

In this section we will investigate the stabilized block system (30) with a fixed $u_n$ and derive preconditioners. Let us first look at the properites of the single blocks: We know that the matrix $A_p$ is at least positive semi-definite and positive definite if the pressure is fixed on a part of the boundary or a rank-1 stabilization (21) is applied. For the matrix $\mathbf{A}_\alpha$ positive definiteness in the sense of $\langle \mathbf{A}_\alpha\mathbf{u},\mathbf{u}\rangle > 0$ for all $\mathbf{u} \in \mathbb{R}^{3N}\setminus\{0\}$ cannot be assumed in general. Since $\mathbf{A}_\alpha$ is non-symmetric, let us look at the symmetric part $\mathbf{H} = \frac{1}{2}(\mathbf{A}_\alpha + \mathbf{A}_\alpha^T)$ first. $\mathbf{H}$ is a block diagonal matrix, we denote the diagonal blocks with $H$. From the definition of the definition of $A^{\mathrm{dif}}$ and $A^{\mathrm{con}}(\mathbf{u})$ we get

$$H(P,P) = \alpha D(P,P) + \mu\sum_{j=1}^{n_f}\frac{\|\mathbf{A}_j\|^2}{\mathbf{A}_j\cdot\mathbf{d}_j} + \sum_{j=1}^{n_f}\rho\hat{f}_j^+ \tag{31}$$

and

$$\sum_{j=1}^{n_f}|H(P,P_j)| = \frac{1}{2}\sum_{j=1}^{n_f}\left|-2\mu\frac{\|\mathbf{A}_j\|^2}{\mathbf{A}_j\cdot\mathbf{d}_j} - \rho|\hat{f}_j|\right| = \sum_{j=1}^{n_f}\left(\mu\frac{\|\mathbf{A}_j\|^2}{\mathbf{A}_j\cdot\mathbf{d}_j} + \frac{\rho}{2}|\hat{f}_j|\right). \tag{32}$$

It directly follows that

$$|\mathbf{H}(P,P)| - \sum_{j=1}^{n_f}|\mathbf{H}(P,P_j)| = \alpha D(P,P) + \frac{1}{2}\sum_{j=1}^{n_f}\rho\hat{f}_j. \tag{33}$$

In general we cannot guarantee that $\sum_{j=1}^{n_f}\rho\hat{f}_j$ is greater or equal zero for all cells. However we can adjust the stabilization parameters $\alpha$ and $D$ such that

$$\alpha D(P,P) + \frac{1}{2}\sum_{j=1}^{n_f}\rho\hat{f}_j > 0 \tag{34}$$

is fulfilled. If condition (34) is satisfied the positive definiteness of $\mathbf{H}$ follows immediately from the diagonal dominance and the fact that all diagonal entries are positive and all off-diagonal entries negative. Thus from (34) we can conclude

$$\langle \mathbf{A}_\alpha \mathbf{u}, \mathbf{u} \rangle = \frac{1}{2}(\langle \mathbf{A}_\alpha \mathbf{u}, \mathbf{u} \rangle + \langle \mathbf{u}, \mathbf{A}_\alpha^T \mathbf{u} \rangle) = \langle \mathbf{H}\mathbf{u}, \mathbf{u} \rangle > 0 \tag{35}$$

for all $\mathbf{u} \in \mathbb{R}^{3N} \setminus \{0\}$.
The term $\sum_{j=1}^{n_f} \rho \hat{f}_j$ in (34) reflects the continuity equation on the discrete level. In the course of the non-linear iteration this term will become smaller so that stabilization can be weakened. In the beginning of the iteration, when the solution is far from the physical solution and mass conservation is not well satisfied $D$ and $\alpha$ have to be set to a bigger value. The condition (34) can be easily checked in a code in every non-linear iteration with very little extra computational effort.
Let us now assume that $\mathbf{A}_\alpha$ includes a stabilization which satisfies (34). For $x = (\mathbf{u}, p) \in \mathbb{R}^{4N}$ we get

$$\langle \mathcal{A}_\alpha x, x \rangle = \langle \mathbf{A}_\alpha u, u \rangle + \langle A_p p, p \rangle + \langle (\mathbf{M} + \mathbf{C}^T) p, \mathbf{u} \rangle.$$

From the positive definiteness of $A_p$ and assumption (34) it follows that $\langle \mathbf{A}_\alpha \mathbf{u}, \mathbf{u} \rangle + \langle A_p p, p \rangle > 0$. If we would have $\mathbf{C}^T = -\mathbf{M}$ then $\mathrm{Re}(\lambda) > 0$ for all $\lambda \in \sigma(\mathcal{A})$ would follow immediately (e.g. from theorem 3.6 in [2]). In contrast to most finite element discretizations this is not the case for the discretization introduced in this paper for general unstructured meshes. Let us look at the entries of $M^i + (C^i)^T$:

$$[M^i + (C^i)^T](P, P_j) = (2\omega_j - 1) A_j^i, \quad [M^i + (C^i)^T](P, P) = \sum_{b=1}^{n_b} A_b^i + 2 \sum_{j=1}^{n_{f,i}} \omega_j A_j^i$$

For an inner cell it is $n_b = 0$. We have $n_b > 0$ if $P$ is a boundary cell and either the boundary condition (5) or (6) is given. For a uniform mesh it can be easily seen that it holds $M^i + (C^i)^T = 0$, since we have $\omega_j = \frac{1}{2}$ and

$$\sum_{j=1}^{n_f} A_j^i = 0, \quad i = 1, 2, 3. \tag{36}$$

For general unstructured meshes the entries of the sum $M^i + (C^i)^T$ depend on the geometrical properties of the mesh. Using (36) and the assumption (8) on the mesh we can deduce the following estimates

$$\begin{aligned} |[M^i + (C^i)^T](P, P_j)| &= |(2\omega_j - 1) A_j^i| \leq 2\varepsilon |A_j^i|, \\ |[M^i + (C^i)^T](P, P)| &= \left| \sum_{j=1}^{n_{f,i}} (2\omega_j - 1) A_j^i \right| \leq 2\varepsilon \sum_{j=1}^{n_{f,i}} |A_j^i| \end{aligned}$$

and hence

$$\|\mathbf{M}+\mathbf{C}^T\|_\infty \leq 4\varepsilon \max_{i=1,3}\max_{P=1,N}\sum_{j=1}^{n_f(P)}|A_j^i| = c_{h,\varepsilon}.$$

From section 2.2.1 we know that $M^i$ and $C^i$ only differ by the diagonal entries (which is due to boundary conditions), therefore we have $M^i+(C^i)^T=(M^i+(C^i)^T)^T$ and thus $\|\mathbf{M}+\mathbf{C}^T\|_2 \leq \|\mathbf{M}+\mathbf{C}^T\|_\infty$ (see corollary 2.3.2. in [11]). This estimate shows that the term $\langle(\mathbf{M}+\mathbf{C}^T)p,\mathbf{u}\rangle$ can be controlled by the mesh 'uniformity' and the cell size. If the mesh is sufficiently uniform such that $c_{h,\varepsilon} \leq 2\sqrt{\lambda_{\min}(\mathbf{H})\lambda_{\min}(A_p)}$ then we get that $\mathcal{A}_\alpha$ is invertible and all eigenvalues of $\mathcal{A}_\alpha$ have positive real part.

Let us now turn to the solution of the linear system. We will now assume that $\mathcal{A}_\alpha$ is non-singular, which we know due to the previous considerations, is the case if the mesh is sufficiently 'uniform'. For the solution of the linear system (30) we choose the restarted GMRES algorithm [19]. It is known that GMRES converges fast if the eigenvalues are enclosed in a small region in the complex plane, which is usually not the case for the unpreconditioned matrix $\mathcal{A}_\alpha$. Therefore the convergence has to be improved by preconditioning. In this section we present two different preconditioning approaches, a block preconditioner and a preconditioner based on the algebraic multigrid method.

### 3.1 Preconditioners based on Block LU decomposition

The idea is to exploit the block-structure of the linear system for the deduction of an efficient preconditioner. We assume that a relaxation is chosen such that condition (34) is fulfilled, then we know from the previous subsection that $\mathbf{A}_\alpha(\mathbf{u}_n)$ is invertible. Then the block LU-decomposition of the linear system (24) is given by

$$\mathcal{A}_\alpha(\mathbf{u}_n)=\begin{pmatrix}\mathbf{A}_\alpha(\mathbf{u}_n) & \mathbf{M}\\ \mathbf{C} & A_p\end{pmatrix}=\begin{pmatrix}I & 0\\ \mathbf{C}\mathbf{A}_\alpha(\mathbf{u}_n)^{-1} & I\end{pmatrix}\begin{pmatrix}\mathbf{A}_\alpha(\mathbf{u}_n) & 0\\ 0 & S(\mathbf{u}_n)\end{pmatrix}\begin{pmatrix}I & \mathbf{A}_\alpha(\mathbf{u}_n)^{-1}\mathbf{M}\\ 0 & I\end{pmatrix} \tag{37}$$

with the Schur-Complement $S(\mathbf{u}_n)=A_p-\mathbf{C}\mathbf{A}_\alpha(\mathbf{u}_n)^{-1}\mathbf{M}$. Replacing $\mathbf{P}_{A_\alpha}\approx\mathbf{A}_\alpha^{-1}$ and $P_S\approx S^{-1}$ we can find the preconditioner

$$\mathcal{P}=\begin{pmatrix}I & -\mathbf{P}_{A_\alpha}\mathbf{M}\\ 0 & I\end{pmatrix}\begin{pmatrix}\mathbf{P}_{A_\alpha} & 0\\ 0 & P_S\end{pmatrix}\begin{pmatrix}I & 0\\ -\mathbf{C}\mathbf{P}_{A_\alpha} & I\end{pmatrix}. \tag{38}$$

Clearly the Schur-complement matrix cannot be computed exactly, for this reason similar to the SIMPLE method we neglect $\mathbf{C}\mathbf{A}_\alpha(\mathbf{u}_n)^{-1}\mathbf{M}$ and set $P_S=P_{A_p}$. The block preconditioner $\mathcal{P}$ is very easy to implement and allows to use existing implementations for $\mathbf{P}_{A_\alpha}$ and $P_{A_p}$ from the SIMPLE method. In analogy to the SIMPLE method we choose an ILU(0) preconditioner for $\mathbf{P}_{A_\alpha}$ and an AMG preconditioner for $P_{A_p}$. Thus the application of $\mathcal{P}$ involves 3 times the application of the ILU(0) preconditioner and one time the application of the AMG preconditioner. For the AMG preconditioner smoothed aggregation coarsening ([21]) in combination with a standard v-cycle and a Gauss-Seidel smoother is used.

A better but still crude approximation of the Schur complement would be $S(\mathbf{u}_n) \approx A_p - \mathbf{C}\mathbf{A}_{\alpha,D}(\mathbf{u}_n)^{-1}\mathbf{M}$, where $\mathbf{A}_{\alpha,D} = \mathrm{diag}(\mathbf{A}_\alpha)$. The drawback of this approximation is that the matrix $\mathbf{C}\mathbf{A}_{\alpha,D}(\mathbf{u}_n)^{-1}\mathbf{M}$ is usually less sparse than $A_p$, since every cell is now also connected to the neighbors of the neighbors:

$$[\mathbf{C}\mathbf{A}_{\alpha,D}(\mathbf{u}_n)^{-1}\mathbf{M}p]_P = \sum_{j=1}^{n_f}\left[\omega_j \frac{1}{a_P}\mathbf{g}_P + \frac{1}{a_{P_j}}(1-\omega_j)\mathbf{g}_{P_j}\right]\cdot \mathbf{A}_j.$$

It can be seen that the application of the gradient $\mathbf{g}_{P_j} = [Mp]_{P_j}$ involves the neighbors of a cell $P_j$, i.e. the neighbors of the neighbors of cell $P$. Further for $A_p$ we know that it is symmetric positive definite, whereas for $A_p - \mathbf{C}\mathbf{A}_{\alpha,D}(\mathbf{u}_n)^{-1}\mathbf{M}$ we know that it is not symmetric for non-uniform meshes (due to $[C^i]^T \neq M^i$). For this reason we choose $P_S = P_{A_p}$. This choice can also be motivated by another consideration: For the convection-diffusion operator

$$\mathcal{L}(\mathbf{u}) = \rho(\mathbf{u}\cdot\nabla) - \mu\Delta$$

the commutativity relation

$$\mathrm{div}[\mathcal{L}(\mathbf{u})]^{-1}\nabla \approx \Delta[\mathcal{L}(\mathbf{u})]^{-1}$$

can be deduced (similar to the idea in [20]). Assuming $\mathrm{div}(\mathbf{u}) = 0$ this relation can be translated to our discrete operators (we neglect cross-diffusion terms)

$$V^{-1}\mathbf{C}[\mathbf{V}^{-1}\mathbf{A}(\mathbf{u})]^{-1}\mathbf{V}^{-1}\mathbf{M} \approx -V^{-1}A^{\mathrm{dif}}[V^{-1}A(\mathbf{u})]^{-1},$$

where $V$ is a finite volume 'mass matrix'. More precisely $V$ is a diagonal matrix with $V(P,P) = |\Omega_P|$. $\mathbf{V}$ is the corresponding block diagonal matrix. Hence we get the approximation

$$-\mathbf{C}\mathbf{A}(\mathbf{u})^{-1}\mathbf{M} \approx A^{\mathrm{dif}}[A(\mathbf{u})]^{-1}V.$$

Replacing $[\mathbf{A}(\mathbf{u})]^{-1}$ by $[\mathbf{A}_D(\mathbf{u})]^{-1}$ gives for the matrix-vector product with a vector $p$

$$[A^{\mathrm{dif}}[\mathbf{A}_D(\mathbf{u})]^{-1}Vp]_P = \sum_{j=1}^{n_f}\frac{\|\mathbf{A}_j\|^2}{\mathbf{A}_j\cdot\mathbf{d}_j}\left[\frac{V_P}{a_P}p_P - \frac{V_{P_j}}{a_{P_j}}p_{P_j}\right] \tag{39}$$

If we would subsitute $\dfrac{V_P}{a_P}$ and $\dfrac{V_{P_j}}{a_{P_j}}$ by the average $\dfrac{1}{2}\left(\dfrac{V_P}{a_P} + \dfrac{V_{P_j}}{a_{P_j}}\right)$ we would end up with the matrix $A_p$. This is why (39) can be seen as an approximation of $A_p$. This justifies to use a preconditioner which is constructed based on the matrix $A_p$ for the Schur complement $S$.

### 3.2 AMG Preconditioners

In a CFD simulation a major part of the computation time is consumed by the solution of linear systems, therefore a fast linear solver is also the key to a fast CFD simulation. Algebraic multigrid methods have been proven to serve as a very efficient and fast preconditioner for the pressure correction equation. Standard methods can be applied since the system matrix

is symmetric and positive definite in the case of incompressible flow. Also for compressible flow AMG methods can be applied (see e.g.[6]). For using algebraic multigrid methods for the solution of block systems there are basically three approaches:
The 'variable based' approach applies AMG to the whole system as a kind of 'black box' as it is and does not consider the origin from a system of PDE's. This approach is problematic for systems like (24) since the interpolation of different physical quantities should be avoided.
If we apply AMG to the 'diagonal blocks' associated with the different physical unknowns, we refer to this ansatz as 'unknown based'. This means the multigrid hierarchy is built up separately for each physical unknown. Interpolation and restriction operators are defined for each block separately. This approach is straightforward to apply, however the coupling between the unknowns is ignored and thus the preconditioner might not be efficient if there is a strong coupling between the two physical unknowns.
We work with the point-based approach as it has turned out to be most efficient for our type of application. In the CFD context this approach has originally been introduced in [17]. The method assigns variables to 'points', in this setting this leads to a reordered system:

$$\tilde{\mathcal{A}}_\alpha(\mathbf{u}) = \begin{pmatrix} A_{1,1} & \dots & A_{1,N} \\ \vdots & \ddots & \vdots \\ A_{N,1} & \dots & A_{N,N} \end{pmatrix}, \quad \text{with} \quad A_{i,j} = \begin{pmatrix} A_{\alpha,ij} & & & M_{ij}^1 \\ & A_{\alpha,ij} & & M_{ij}^2 \\ & & A_{\alpha,ij} & M_{ij}^3 \\ C_{ij}^1 & C_{ij}^2 & C_{ij}^3 & A_{p,ij} \end{pmatrix}. \tag{40}$$

The setup of a multi grid hierarchy requires the definition of a coarsening strategy. A sequence of coarse grid matrices $\tilde{\mathcal{A}}^l \in \mathbb{R}^{n_l \times n_k}$ with $l = 1, \dots, l_{\max}$ has to be constructed. For the coarsening there are basically two strategies described in literature. There is the 'classical' Ruge-Stüben approach and aggregation methods. We work with aggregation methods, where a set of nodes is collected in an aggregate. The prolongation from the coarse grid to the fine grid is then done by assigning the aggregate value to all points, which belong to this aggregate. This means we have a constant interpolation operator. We refer to [8] for the investigation of various aggregation methods for the coupled system and the parallel coarse grid treatment.
For the determination of strong and weak couplings we have to define $\|A_{i,j}\|$. There are many different strategies for the choice of $\|A_{i,j}\|$, our method of choice is to identify the norm with the absolute value of a certain physical unknown. In particular we choose $\|A_{i,j}\| = |A_{p,ij}|$. This strategy is appropriate since velocity and pressure matrices have the same sparsity pattern. Going back to the original block representation of the matrix $\mathcal{A}_\alpha$ we set the prolongation operator

$$\begin{pmatrix} \mathbf{u}_l \\ p_l \end{pmatrix} = \mathcal{P}_l \begin{pmatrix} \mathbf{u}_{l+1} \\ p_{l+1} \end{pmatrix}, \quad \text{with } \mathcal{P}_l = \begin{pmatrix} P_l & & & \\ & P_l & & \\ & & P_l & \\ & & & P_l \end{pmatrix}. \tag{41}$$

The restriction is set as $\mathcal{R}_l = \mathcal{P}_l^T$. As concrete coarsening strategy we choose pairwise aggregation ([13, 15]), which produces aggregates of two fine grid nodes. An advantage of this method is that the setup is cheap and it shows good convergence in combination with standard cycling strategies. A drawback regarding the computing time is that due to the small size of the aggregates a large number of multigrid levels is produced. After having defined the coarse grid matrices and the block-wise prolongation and restriction operators, a standard MG cycling

strategy can be used straightforward. We use the v-cycle in combination with two iterations of an ILU(0) smoother, i.e. $\mathcal{M}_l = L_l^{-1} U_l^{-1}$, where $(L_l, U_l)$ is the ILU(0) decomposition of $\tilde{\mathcal{A}}^l$.

# 4 Numerical Examples

In the last section we illustrate the performance of the previous derived methods by numerical examples. We present three different simulation cases, two stationary problems and one time dependent problem. All subsequent simulations were done within the framework of AVL FIRE®. The code runs parallel using a mesh decomposition, the communication is performed through calls to platform-MPI subroutines.
We look at the number of non-linear iterations needed to achieve a certain accuracy, i.e. the velocity and mass residuals are below a certain tolerance $\varepsilon$. The coupled linear system is solved using a preconditioned, restarted GMRES algorithm. We compare results for the block preconditioner introduced in section 3.1 and the AMG preconditioner (PAMG), which was introduced in section 3.2. The comparison is done w.r.t. the average number of linear iterations needed, which is the total number of linear iterations for the whole run divided by the number of non-linear iterations.

## 4.1 Driven Cavity

As a first example we consider the driven cavity problem. The computational domain is a cube, on the boundary the velocity is fixed. On one side of the cube the velocity is set to $(1, 0, 0)$, on all other sides the velocity is set to $(0, 0, 0)$. We set the viscosity such that Re=1000. We consider different discretizations, all meshes are orthogonal and cell sizes shrink with a factor 4 towards the boundary. The convergence tolerance for mass and velocity residuals is set to $\varepsilon = 10^{-7}$. For the discretization of the convective term we use an upwind scheme. For the coupled method no relaxation is used, for SIMPLE we use $\alpha_u = 0.6, \alpha_p = 0.1$. In table 1 we show the number of non-linear iterations needed by SIMPLE and the coupled solver, as well as the average number of linear iterations. The table clearly highlights the $h$-dependency of the block preconditioner, the iteration numbers for the PAMG preconditioner also increase slightly with $h$ becoming smaller. We also show the iteration numbers for an ILU(0) preconditioner applied to the whole block system. Further simulation times (average wall clock time) are given in table 1. It can be observed that for coarser meshes the block preconditioner and PAMG need a similar amount of time while for finer discretizations PAMG clearly shows a better performance. A fair comparison of the simulation time between SIMPLE and the coupled solver is somewhat difficult since the performance of the SIMPLE method depends strongly on the choice of under-relaxation factors. A good guess for relaxation parameters may result in good performance of the SIMPLE iteration, choosing smaller relaxation parameters to be on the safe side regarding convergence of the non-linear iterations may cause a very slow convergence of SIMPLE.

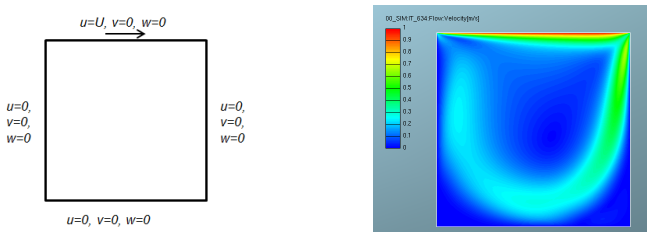



Figure 1: Driven Cavity: problem setting and velocity magnitude for Re=1000

| number of cells | SIMPLE | | | coupled | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | non-linear it. | AMG | | non-linear it. | PAMG | | BlockPrecond | | ILU(0) | |
| | | lin. it. | time | | lin. it. | time | lin. it. | time | lin. it. | time |
| 32768 | 387 | 5.7 | 22 | 108 | 5.0 | 30 | 18.2 | 28 | 23.3 | 23 |
| 262144 | 1036 | 8.3 | 382 | 112 | 5.0 | 214 | 25.5 | 250 | 39.5 | 272 |
| 884736 | 2118 | 10.6 | 2483 | 186 | 5.3 | 1393 | 31.1 | 1625 | 67.5 | 2684 |
| 2097152 | 3531 | 11.6 | 11592 | 291 | 6.7 | 5604 | 35.3 | 6859 | 105.5 | 14058 |

Table 1: Driven Cavity: Average linear iterations and average wall clock time in s on 8CPU's

## 4.2 Mixer

We consider a geometry, which models a mixer. The mixer stirs water in a cylindrical vessel (see figure 2). The sliding blades of the mixer are rotating with a speed of 1.15m/s, this is modeled with a moving mesh. The problem is time-dependent, we set the time step $\Delta t = 0.0025$ and the end time $t_{\text{end}} = 0.5$. For turbulence modeling we use the $k - \varepsilon$ model. For the SIMPLE iterations we use the under-relaxation factors $\alpha_u = 0.4, \alpha_p = 0.1$. On the surface of the mixer we set wall boundary conditions.

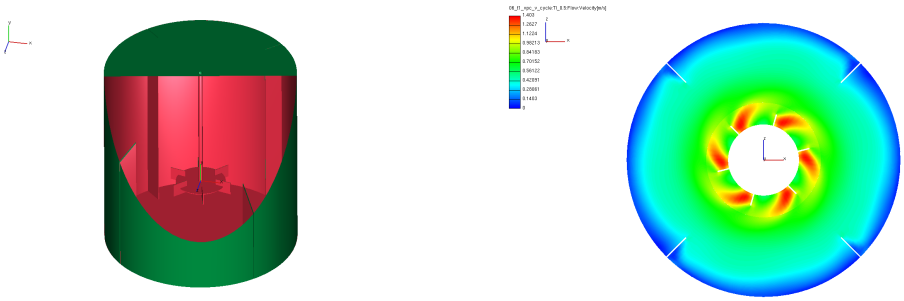


Figure 2: Mixer geometry and velocity magnitude for $t = 0.5$

| number of cells | SIMPLE | | | coupled | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | non-linear it. | AMG | | non-linear it. | PAMG | | BlockPrecond | | ILU(0) | |
| | | lin. it. | time | | lin. it. | time | lin. it. | time | lin. it. | time |
| 339504 | 9463 | 7.8 | 5986 | 1392 | 2.3 | 5284 | 5.3 | 2502 | 12.7 | 4926 |

Table 2: Mixer: Average linear iterations and average wall clock time in s on 10 CPU's

In table 2 the average iteration numbers for the point based AMG and the block preconditioner are given. The table shows that the number of iterations is a little bit higher than the number of iterations needed by AMG. However the simulation using the block preconditioner takes only half the time as the simulation with the PAMG preconditioner.

### 4.3 Intake Port

This problem stems from engine simulation, the geometry describes a cylinder with a valve. The valve lift is 12mm (see figure 3). At the inlet we prescribe the total pressure, at the outlet 'static pressure' boundary conditions are set. The fluid is air. We have two discretizations with 1.9mio and 5mio finite volumes. The convergence tolerance for mass and velocity residuals is set to $10^{-4}$. For turbulence modeling we use the $k-\varepsilon$ model (see [12]). For the coupled problem no relaxation is used, for SIMPLE we use $\alpha_u = 0.6, \alpha_p = 0.1$.

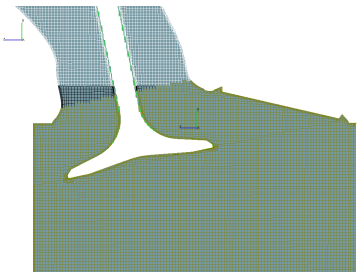
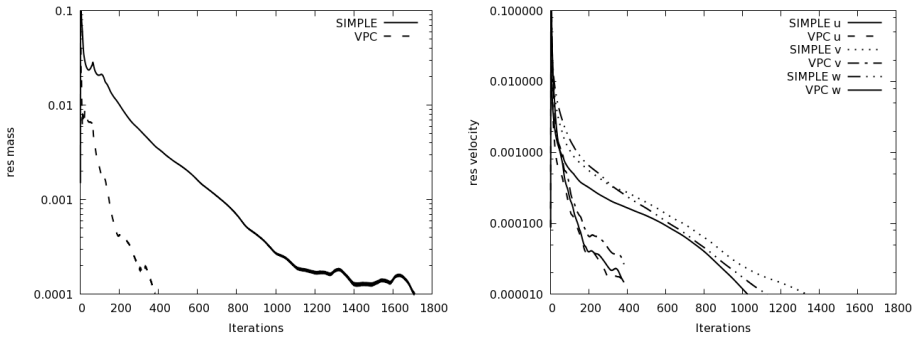
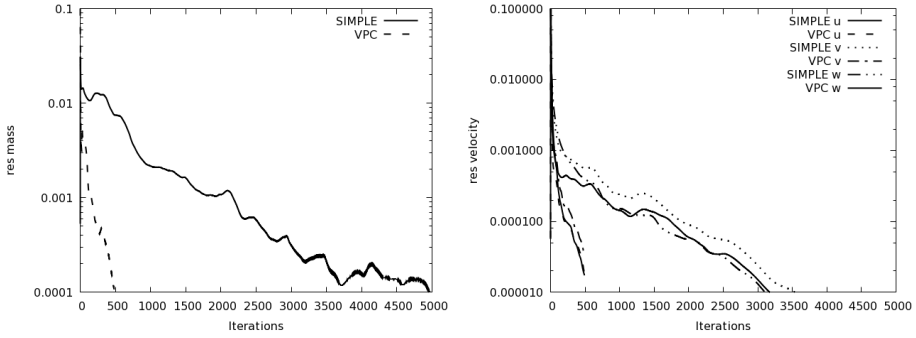


Figure 3: Intake Port mesh




(a) Residuals 1.9mio mesh




(b) Residuals 5mio mesh

Figure 4: Intake Port mass and pressure residuals

As shown in table 3 the number of non-linear iterations is significantly smaller for the coupled approach compared to the SIMPLE method. The mass and pressure residuals for the coupled and the segregated approach can be found in figure 4. The average linear iterations for the PAMG and the block preconditioner can also be found in table 3. For both PAMG and the block preconditioner the (linear) iteration numbers increase as the cell size becomes smaller. The block preconditioner needs significantly more iterations than PAMG however it can be observed that for the coarser discretization the simulation time for PAMG and block preconditioner is the same. For the fine discretization the block preconditioner needs about 20% more time than PAMG. With the ILU(0) preconditioner the linear solver did not converge, which caused divergence of the non-linear iteration.

| number of cells | SIMPLE | | | coupled | | | | |
|---|---|---|---|---|---|---|---|---|
| | non-linear it. | AMG | | non-linear it. | PAMG | | BlockPrecond | |
| | | lin. it. | time | | lin. it. | time | lin. it. | time |
| 1905323 | 1711 | 11.4 | 3494 | 381 | 7.6 | 5486 | 38.9 | 5253 |
| 4997098 | 4967 | 11.2 | 26353 | 508 | 10.0 | 22661 | 52.8 | 26968 |

Table 3: Intake port: Average linear iterations and average wall clock time in s on 16CPU's

# 5 Conclusion and Outlook

This work provides a formulation for the numerical solution of the incompressible Navier-Stokes equations. We have derived a non-linear set of equations, which results from the finite volume discretization using a coupled approach. The non-linear system is solved via a fixed-point method, each non-linear iteration requires the approximate solution of a block system. We have analyzed the linear system and introduced two different preconditioner approaches, a block preconditioner and an AMG variant, which can be applied to the block system.
Numerical examples have been given, which illustrate the performance of the preconditioners. The block preconditioner needs usually more iterations than the monolithic AMG approach. However for smaller grids the computation time is similar to the AMG approach or even smaller. An advantage of the block preconditioner is that it is very easy to implement if one has preconditioners for the single blocks available. For larger grids AMG clearly outperforms the block preconditioner. The monolithic AMG approach seems to be promising, though further decreasing the computing time would still be desirable. The ILU(0) smoother takes a major part of the computation time. So a potential for reducing the simulation time would be to find a faster but still efficient smoother.
In applications there is also the need to simulate the compressible case. The compressible case is characterized by a variable density, which affects the continuity equation. As a consequence the matrix $A_p$ is not symmetric anymore. Basically both preconditioner approaches presented in this paper can also be used for the compressible case, yet a careful analysis and testing would still be needed.

# Acknowledgement

I would like to thank the colleagues from AVL List for interesting and fruitful discussions and for proofreading the manuscript.

# References

[1] M. Benzi. Preconditioning techniques for large linear systems: A survey. *Journal of Computational Physics*, 182(2):418 – 477, 2002.

[2] M. Benzi, G. H. Golub, and J. Liesen. Numerical solution of saddle point problems. *Acta Numerica*, 14:1–137, 5 2005.

[3] Z. Chen and A. Przekwas. A coupled pressure-based computational method for incompressible/compressible flows. *Journal of Computational Physics*, 229(24):9150 – 9165, 2010.

[4] T. Clees. AMG strategies for PDE systems with applications in industrial semiconductor simulation. *Dissertation, Universität zu Köln*, 2004.

[5] M. Darwish, I. Sraj, and F. Moukalled. A coupled finite volume solver for the solution of incompressible flows on unstructured grids. *Journal of Computational Physics*, 228(1):180 – 201, 2009.

[6] M. Emans. Benchmarking aggregation AMG for linear systems in CFD simulations of compressible internal flows. *Electronic Transactions on Numerical Analysis*, 10:351 – 366, 2010.

[7] M. Emans. Efficient parallel AMG methods for approximate solutions of linear systems in CFD applications. *SIAM Journal on Scientific Computing*, 32(4):2235–2254, 2010.

[8] M. Emans. Coarse-grid treatment in parallel AMG for coupled systems in CFD applications. *Journal of Computational Science*, 2(4):365 – 376, 2011.

[9] M. Emans and M. Liebmann. Velocity - pressure coupling on GPUs. *Computing*, 95(1):123–143, 2013.

[10] J. Ferziger and M.Perić. *Computational Methods for Fluid Dynamics*. Springer, 1996.

[11] G. H. Golub and C. F. Van Loan. *Matrix Computations (4th Ed.)*. Johns Hopkins University Press, Baltimore, MD, USA, 2013.

[12] W. Jones and B. Launder. The prediction of laminarization with a two-equation model of turbulence. *International Journal of Heat and Mass Transfer*, 15(2):301 – 314, 1972.

[13] H. Kim, J. Xu, and L. Zikatanov. A multigrid method based on graph matching for convection-diffusion equations. *Numerical Linear Algebra with Applications*, 10(1-2):181–195, 2003.

[14] B. Metsch. Algebraic multigrid (AMG) for saddle point systems. *Dissertation, Rheinische Friedrich-Wilhelms-Universität Bonn*, 2013.

[15] Y. Notay. An aggregation-based algebraic multigrid method. *Electronic Transactions on Numerical Analysis*, 37:123–146, 2010.

[16] S. Patankar and D. Spalding. A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows. *International Journal of Heat and Mass Transfer*, 15(10):1787 – 1806, 1972.

[17] M. Raw. A coupled algebraic multigrid method for the 3D navier-stokes equations. In W. Hackbusch and G. Wittum, editors, *Fast Solvers for Flow Problems*, Notes on Numerical Fluid Mechanics (NNFM), pages 204–215. Vieweg+Teubner Verlag, 1995.

[18] C. M. Rhie and W. L. Chow. Numerical study of the turbulent flow past an airfoil with trailing edge separation. *AIAA Journal*, 21(11):1525–1532, 1983.

[19] Y. Saad and M. H. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, 7(3):856–869, 1986.

[20] D. Silvester, H. Elman, D. Kay, and A. Wathen. Efficient preconditioning of the linearized Navier-Stokes equations for incompressible flow. *Journal of Computational and Applied Mathematics*, 128(1-2):261–279, 2001. Numerical Analysis 2000. Vol. VII: Partial Differential Equations.

[21] P. Vaněk, J. Mandel, and M. Brezina. Algebraic multigrid by smoothed aggregation for second and fourth order elliptic problems. *Computing*, 56(3):179–196, 1996.

[22] M. Wabro. Algebraic multigrid methods for the numerical solution of the incompressible navier-stokes equations. *Dissertation, Institut für Numerische Mathematik, Johannes Kepler Universität, Linz*, 2003.

[23] R. Webster. Stability of algebraic multigrid for stokes problems. *International Journal for Numerical Methods in Fluids*, 71(4):488–505, 2013.

[24] R. Webster. Stabilisation of AMG solvers for saddle-point stokes problems. *International Journal for Numerical Methods in Fluids*, 2015. FLD-14-0460.R1.