# Implementation and Performance Analyses of a Highly Efficient Algorithm for Pressure-Velocity Coupling

**Implementierung und Untersuchung einer hoch effizienten Methode zur Druck-Geschwindigkeits-Kopplung**

Master-Thesis von Fabian Gabel

Tag der Einreichung:

1. Gutachten: Prof. Dr. rer. nat. Michael Schäfer
2. Gutachten: Dipl.-Ing Ulrich Falk

TECHNISCHE UNIVERSITÄT DARMSTADT

Studienbereich CE
FNB

Implementation and Performance Analyses of a Highly Efficient Algorithm for Pressure-Velocity Coupling
Implementierung und Untersuchung einer hoch effizienten Methode zur Druck-Geschwindigkeits-Kopplung

Vorgelegte Master-Thesis von Fabian Gabel

1. Gutachten: Prof. Dr. rer. nat. Michael Schäfer
2. Gutachten: Dipl.-Ing Ulrich Falk

Tag der Einreichung:

# Erklärung zur Master-Thesis

Hiermit versichere ich, die vorliegende Master-Thesis ohne Hilfe Dritter nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus Quellen entnommen wurden, sind als solche kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Darmstadt, den 13. Februar 2015

(Fabian Gabel)

## Contents

## List of Figures

## List of Tables



## List of Algorithms

## TODO LIST

- midpoint integration rule or midpoint rule of integration
- remove equation numbering when there is no reference
- change the matrix coefficient indexing to $a_P^{u_i,p}$ like in [15].
- extend nomenclature
- align equations using the **alignat** environment
- check pressure correction equation on iteration indices (gradients)
- check simple chapter, since the right hand side lacks of deferred corrector and under-relaxed velocities
- mention the boundary conditions for the pressure correction
- check all headings for correct spelling
- mention that for an unknown velocity field the partial differential equation for the temperature is non-linear as well
- consistent use of either temperature or energy equation
- check the signs in the Boussinesq approximation
- pressure weighted interpolation method for large body forces?
- not only speed but also improvement of robustness
- instationary flows?
- align exponents in the equation for the Newton-Raphson linearization
- define the term consistent
- add citations to clipper, opencascade and maple, ICEM CFD
- a priori exact solution use this formulation
- Falsche Wahl der problem domain, führt zu global nicht erfüllter kontinuitätsgleichung. residuum der druckkorrectur entspricht dem massenfluss
- reference Comparison of finite-volume numerical methods with staggered and colocated
- read klaij again and use some of its arguments grids
- emphasize the flexibility of the implementation

## 4 Implicit Finite Volume Method for Incompressible Flows – Segregated Approach

The purpose of this section is to present the discretization applied to the set of equations (11). Since the system of partial differential equations to be solved always exhibits a coupling, at least between the dependent variables pressure and velocity, a first solution algorithm, namely the *SIMPLE* algorithm, addressed to resolve the pressure velocity coupling, is introduced. Furthermore an under-relaxation factor independent method of calculating mass fluxes by interpolation is introduced and the detailed derivation of all coefficients that result from the discretization process is presented. Finally the boundary conditions, that are relevant for the present thesis will be introduced.

### 4.1 Discretization of the Mass Balance

Integration of equation (11a) over the integration domain of a single control volume $P$, after the application of Gauss' integration theorem and the additivity of the Riemann integral, yields

$$\iint_{S_f} u_i n_i \mathrm{d}S = \sum_{f\in\{w,s,b,t,n,e\}} \iint_{S_f} u_i n_i \mathrm{d}S = 0.$$

In the present work the mass balance is discretized using the midpoint integration rule for the surface integrals and linear interpolation of the velocity to the center of mass of the surface. This leads to the following form of the mass balance:

$$\sum_{f\in\{w,s,b,t,n,e\}} u_{i,f} n_{f_i} S_f = 0, \tag{16}$$

where no interpolation to attain the values of $u_i$ at the face $S_f$ is performed yet, since the straightforward linear interpolation will lead to undesired oscillations in the solution fields. An interpolation method to circumvent this so called *checker boarding* effect is presented in subsection 4.2.

### 4.2 A Pressure-Weighted Interpolation Method for Velocities

The advantages of using a cell-centered variable arrangement are evident: The treatment of non-orthogonality is simplified and the conservation property of finite volume methods is retained [10, 39, 45, 66]. A major drawback with cell centered variable arrangements is that pressure field may delink, which will then lead to unphysical oscillations in both the pressure and the velocity results. If the oscillations are severe enough the solution algorithm might even get unstable and diverge. The described decoupling occurs, when the pressure gradient in the momentum balances and the mass fluxes in the continuity equation are discretized using central differences.

A common practice to eliminate this behaviour is the use of a momentum interpolation technique, also known as *Rhie-Chow Interpolation* [52]. The original interpolation scheme however does not guarantee a unique solution, independent of the amount of under-relaxation. The performance of one of the algorithms that are used in the present thesis heavily relies on the under-relaxation of variables to accomplish stability. Furthermore the original method as proposed by [52] does not account for large body forces which also may lead to unphysical results. These issues will be addressed in this subsection which at the end will present an interpolation method that assures an under-relaxation independent solution: the *pressure-weighted interpolation method* [45].

The starting point of the pressure-weighted interpolation method is formed by the discretized momentum balances at node $P$ and an arbitrary neighbouring node $Q$. The discretization for finite volume methods and details including the incorporation of under-relaxation factors will be handled in subsection 4.5. The semi-discrete implicit momentum balances, if one solves for the velocity at node $P$ or $Q$, read

$$u_{i,P}^{(n)} = -\frac{\alpha_{\mathbf{u}_P}}{a_P^{u_i}}\left(\sum_{F\in NB(P)} a_F^{u_i} u_{i,F}^{(n)} + b_{P,u_i}^{(n-1)} - V_P\left(\frac{\partial p}{\partial x_i}\right)_P^{(n-1)}\right) + (1-\alpha_{\mathbf{u}})\, u_{i,P}^{(n-1)} \tag{17a}$$

$$\text{and} \quad u_{i,Q}^{(n)} = -\frac{\alpha_{\mathbf{u}_Q}}{a_Q^{u_i}}\left(\sum_{F\in NB(Q)} a_F^{u_i} u_{i,F}^{(n)} + b_{Q,u_i}^{(n-1)} - V_Q\left(\frac{\partial p}{\partial x_i}\right)_Q^{(n-1)}\right) + (1-\alpha_{\mathbf{u}})\, u_{i,Q}^{(n-1)} \quad , \tag{17b}$$

where the superscript $(n-1)$ denotes the previous outer iteration number. The reader should note, that the pressure gradient has not been discretized yet. This has the advantage that the selective interpolation technique [54] can be

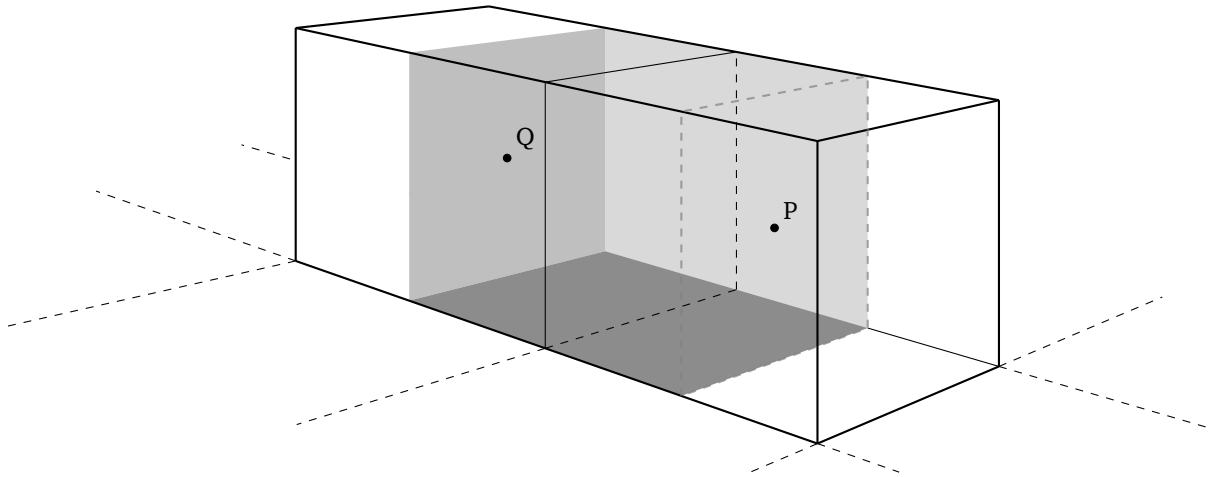


**Figure 4:** Possible interpretation of a virtual control volume (grey) located between nodes $P$ and $Q$

applied, which is crucial for the elimination of the mentioned oscillations. In almost the same way a semi-discrete implicit momentum balance can be formulated for a virtual control volume located between nodes $P$ and $Q$.

$$u_{i,f}^{(n)} = -\frac{\alpha_{\mathbf{u}_f}}{a_f^{u_i}} \left( \sum_{F \in NB(f)} a_F^{u_i} u_{i,F}^{(n)} + b_{f,u_i}^{(n-1)} - V_f \left( \frac{\partial p}{\partial x_i} \right)_f^{(n-1)} \right) + (1 - \alpha_{\mathbf{u}}) u_{i,f}^{(n-1)}. \tag{18}$$

Figure 4.2 gives an interpretation of the virtual control volume. To guarantee convergence of this expression for $u_{i,f}$, under-relaxation is necessary [39]. To eliminate the undefined artifacts surging form the virtualization of a control volume the following assumptions have to be made to derive a closed expression for the velocity on the boundary face $S_f$

$$\frac{\alpha_{\mathbf{u}_f}}{a_f^{u_i}} \left( \sum_{F \in NB(f)} a_F^{u_i} u_{i,F}^{(n)} \right) \approx (1 - \gamma_f) \frac{\alpha_{\mathbf{u}_P}}{a_P^{u_i}} \left( \sum_{F \in NB(P)} a_F^{u_i} u_{i,F}^{(n)} \right) + \gamma_f \frac{\alpha_{\mathbf{u}_Q}}{a_Q^{u_i}} \left( \sum_{F \in NB(Q)} a_F^{u_i} u_{i,F}^{(n)} \right), \tag{19a}$$

$$\frac{\alpha_{\mathbf{u}_f}}{a_f^{u_i}} b_{f,u_i}^{(n-1)} \approx (1 - \gamma_f) \frac{\alpha_{\mathbf{u}_P}}{a_P^{u_i}} b_{P,u_i}^{(n-1)} + \gamma_f \frac{\alpha_{\mathbf{u}}}{a_Q^{u_i}} b_{Q,u_i}^{(n-1)} \tag{19b}$$

$$\text{and} \quad \frac{\alpha_{\mathbf{u}}}{a_f^{u_i}} \approx (1 - \gamma_f) \frac{\alpha_{\mathbf{u}}}{a_P^{u_i}} + \gamma_f \frac{\alpha_{\mathbf{u}}}{a_Q^{u_i}}, \tag{19c}$$

$$\tag{19d}$$

where $\gamma_f$ is a geometric interpolation factor.

Using the assumptions made in equation (19) the expression in equation (18) can be closed in a way that it only depends on the variable values in node $P$ and $Q$

$$
\begin{aligned}
u_{i,f}^{(n)} \approx{}& \left(1-\gamma_f\right)\left(-\frac{\alpha_{\mathbf{u}}}{a_P^{u_i}}\sum_{F\in NB(P)} a_F^{u_i} u_{i,F}^{(n)}\right)+\gamma_f\left(-\frac{\alpha_{\mathbf{u}}}{a_Q^{u_i}}\sum_{F\in NB(Q)} a_F^{u_i} u_{i,F}^{(n)}\right)\\
&+\frac{\alpha_{\mathbf{u}}}{a_f^{u_i}} b_{f,u_i}^{(n-1)}-\frac{\alpha_{\mathbf{u}}}{a_f^{u_i}} V_f\left(\frac{\partial p}{\partial x_i}\right)_f^{(n-1)}+\left(1-\alpha_{\mathbf{u}}\right) u_{i,f}^{(n-1)}\\
={}&\left(1-\gamma_f\right) u_{i,P}^{(n)}-\left(1-\gamma_f\right)\frac{\alpha_{\mathbf{u}}}{a_P^{u_i}}\left(b_{P,u_i}^{(n-1)}-V_P\left(\frac{\partial p}{\partial x_i}\right)_P^{(n-1)}\right)+\gamma_f u_{i,Q}^{(n)}-\gamma_f\frac{\alpha_{\mathbf{u}}}{a_Q^{u_i}}\left(b_{Q,u_i}^{(n-1)}-V_Q\left(\frac{\partial p}{\partial x_i}\right)_Q^{(n-1)}\right)\\
&+\frac{\alpha_{\mathbf{u}}}{a_f^{u_i}} b_{f,u_i}^{(n-1)}-\frac{\alpha_{\mathbf{u}}}{a_f^{u_i}} V_f\left(\frac{\partial p}{\partial x_i}\right)_f^{(n-1)}+\left(1-\alpha_{\mathbf{u}}\right) u_{i,f}^{(n-1)}\\
={}&\left[\left(1-\gamma_f\right) u_{i,P}^{(n)}+\gamma_f u_{i,Q}^{(n)}\right]\\
&-\left[\left(\left(1-\gamma_f\right)\frac{\alpha_{\mathbf{u}} V_P}{a_P^{u_i}}+\gamma_f\frac{\alpha_{\mathbf{u}} V_Q}{a_Q^{u_i}}\right)\left(\frac{\partial p}{\partial x_i}\right)_f^{(n-1)}-\left(1-\gamma_f\right)\frac{\alpha_{\mathbf{u}} V_P}{a_P^{u_i}}\left(\frac{\partial p}{\partial x_i}\right)_P^{(n-1)}-\gamma_f\frac{\alpha_{\mathbf{u}} V_Q}{a_Q^{u_i}}\left(\frac{\partial p}{\partial x_i}\right)_Q^{(n-1)}\right]\\
&+\left(1-\alpha_{\mathbf{u}}\right)\left[u_{i,f}^{(n-1)}-\left(1-\gamma_f\right) u_{i,P}^{(n-1)}-\gamma_f u_{i,Q}^{(n-1)}\right]\\
\approx{}&\left[\left(1-\gamma_f\right) u_{i,P}^{(n)}+\gamma_f u_{i,Q}^{(n)}\right]\\
&-\left(\left(1-\gamma_f\right)\frac{\alpha_{\mathbf{u}} V_P}{a_P^{u_i}}+\gamma_f\frac{\alpha_{\mathbf{u}} V_Q}{a_Q^{u_i}}\right)\underline{\left[\left(\frac{\partial p}{\partial x_i}\right)_f^{(n-1)}-\left(1-\gamma_f\right)\left(\frac{\partial p}{\partial x_i}\right)_P^{(n-1)}-\gamma_f\left(\frac{\partial p}{\partial x_i}\right)_Q^{(n-1)}\right]}\\
&+\left(1-\alpha_{\mathbf{u}}\right)\left[u_{i,f}^{(n-1)}-\left(1-\gamma_f\right) u_{i,P}^{(n-1)}-\gamma_f u_{i,Q}^{(n-1)}\right].
\end{aligned}
\tag{20}
$$

It should be noted that the argumentation that led to the last expression, is that the task of the underlined pressure gradient corrector in equation (20) is to suppress oscillations in the converged solution for the pressure field. If there are no oscillations this part should not become active. As long as the behaviour of this corrector remains consistent, i.e. that there are no oscillations in the pressure field, it can be multiplied with arbitrary constants [22]. This is however true on equidistant grids, where $\gamma_f = 1/2$ and central differences are used to calculate the gradients. On arbitrary orthogonal grids another modification has to be performed which is based on a special case of the mean value theorem of differential calculus and the following

**Proposition.** *Let $x_1, x_2 \in \mathbb{R}$ with $x_1 \neq x_2$ and $p(x) = a_0 + a_1 x + a_2 x^2$ a real polynomial function. Then*

$$\frac{dp}{dx}\left(\frac{x_1+x_2}{2}\right) = \frac{p(x_2)-p(x_1)}{x_2-x_1},$$

*i.e. the slope of the secant equals the value of the first derivative of $p$ exactly half the way between $x_1$ and $x_2$.*

*Proof.* Evaluation of the derivative yields

$$\frac{dp}{dx}\left(\frac{x_1+x_2}{2}\right) = a_1 + 2a_2\frac{x_1+x_2}{2} = a_1 + a_2(x_1+x_2).$$

On the other hand the slope of the secant, using the third binomial rule can be expressed as

$$\begin{aligned}\frac{p(x_2)-p(x_1)}{x_2-x_1} &= \frac{a_0+a_1x_2+a_2x_2^2-\left(a_0+a_1x_1+a_2x_1^2\right)}{x_2-x_1}\\ &= \frac{a_1(x_2-x_1)+a_2\left(x_2^2-x_1^2\right)}{x_2-x_1}\\ &= a_1+a_2(x_2+x_1).\end{aligned}$$

The comparison of both expressions completes the proof. □

It is desirable for the pressure corrector to vanish independent of the grid spacing if the profile of the pressure is quadratic and hence does not exhibit oscillations. According to the preceding proposition this can be accomplished by modifying equation (20) to average the pressure gradients from node $P$ and $Q$ instead of interpolating linearly

$$\begin{aligned}u_{i,f}^{(n)} = &\left[\left(1-\gamma_f\right)u_{i,P}^{(n)}+\gamma_f u_{i,Q}^{(n)}\right]\\ &-\left(\left(1-\gamma_f\right)\frac{\alpha_{\mathbf{u}}V_P}{a_P^{u_i}}+\gamma_f\frac{\alpha_{\mathbf{u}}V_Q}{a_Q^{u_i}}\right)\left[\left(\frac{\partial p}{\partial x_i}\right)_f^{(n-1)}-\frac{1}{2}\left(\frac{\partial p}{\partial x_i}\right)_P^{(n-1)}-\frac{1}{2}\left(\frac{\partial p}{\partial x_i}\right)_Q^{(n-1)}\right]\\ &+\underline{(1-\alpha_{\mathbf{u}})\left[u_{i,f}^{(n-1)}-\left(1-\gamma_f\right)u_{i,P}^{(n-1)}-\gamma_f u_{i,Q}^{(n-1)}\right]}.\end{aligned} \tag{21}$$

Comparing this final expression with the standard interpolation scheme, it is evident that normally, the underlined term is not taken into consideration [22]. However section 7.4 shows, that neglecting this term indeed creates under-relaxation factor dependent results. This section concludes with a final

**Proposition.** *The pressure weighted momentum interpolation scheme (21) guarantees the converged solution for $u_{i,f}$ to be independent of the velocity under-relaxation $\alpha_{\mathbf{u}}$.*

*Proof.* An equivalent formulation of (21) is given by

$$\begin{aligned}\alpha_{\mathbf{u}}u_{i,f}^{(n-1)}+u_{i,f}^{(n-1)}-u_{i,f}^{(n)} = &\alpha_{\mathbf{u}}\left[\left(1-\gamma_f\right)u_{i,P}^{(n-1)}+\gamma_f u_{i,Q}^{(n-1)}\right]\\ &+\left[\left(1-\gamma_f\right)\left(u_{i,P}^{(n)}-u_{i,P}^{(n-1)}\right)+\gamma_f\left(u_{i,Q}^{(n)}-u_{i,Q}^{(n-1)}\right)\right]\\ &-\alpha_{\mathbf{u}}\left(\left(1-\gamma_f\right)\frac{V_P}{a_P^{u_i}}+\gamma_f\frac{V_Q}{a_Q^{u_i}}\right)\left[\left(\frac{\partial p}{\partial x_i}\right)_f^{(n-1)}-\frac{1}{2}\left(\frac{\partial p}{\partial x_i}\right)_P^{(n-1)}-\frac{1}{2}\left(\frac{\partial p}{\partial x_i}\right)_Q^{(n-1)}\right].\end{aligned}$$

Upon convergence $u_{i,P}^{(n)}=u_{i,P}^{(n-1)}$ and $u_{i,f}^{(n)}=u_{i,f}^{(n-1)}$. This leads to

$$\begin{aligned}\alpha_{\mathbf{u}}u_{i,f}^{(n-1)} = &\alpha_{\mathbf{u}}\left[\left(1-\gamma_f\right)u_{i,P}^{(n-1)}+\gamma_f u_{i,Q}^{(n-1)}\right]\\ &-\alpha_{\mathbf{u}}\left(\left(1-\gamma_f\right)\frac{V_P}{a_P^{u_i}}+\gamma_f\frac{V_Q}{a_Q^{u_i}}\right)\left[\left(\frac{\partial p}{\partial x_i}\right)_f^{(n-1)}-\frac{1}{2}\left(\frac{\partial p}{\partial x_i}\right)_P^{(n-1)}-\frac{1}{2}\left(\frac{\partial p}{\partial x_i}\right)_Q^{(n-1)}\right],\end{aligned}$$

which shows, after division by $\alpha_{\mathbf{u}}>0$, that $u_{i,f}$ is independent of the under-relaxation factor. □

## 4.3 Implicit Pressure Correction and the SIMPLE Algorithm

The goal of finite volume methods is to deduce a system of linear algebraic equations from a partial differential equation. In the case of the momentum balances the general structure of this linear equations is

$$u_{i,P}^{(n)} = -\frac{\alpha_{\mathbf{u}_P}}{a_P^{u_i}}\left(\sum_{F\in NB(P)}a_F^{u_i}u_{i,F}^{(n)}+b_{P,u_i}^{(n-1)}-V_P\left(\frac{\partial p}{\partial x_i}\right)_P^{(n-1)}\right)+\left(1-\alpha_{\mathbf{u}}\right)u_{i,P}^{(n-1)}, \tag{22}$$

where the pressure gradient has been discretized only symbolically and $b_{P,u_i}^{(n-1)}$ denotes the source term evaluated at the previous outer iteration.

At this stage the equations are still coupled and non-linear. As described in section 3.4 the Picard iteration process can be used to linearize the equations. Every momentum balance equation then only depends on the one dominant variable $u_i$. Furthermore the coupling of the momentum balances through the convective term $(u_i u_j)$ is resolved in the process of linearization. The decoupled momentum balances can then be solved sequentially for the dominant variable $u_i$. All coefficients $a_{\{P,F\}}^{u_i}$, the source term and the pressure gradient will be evaluated explicitly by using results of the preceding outer iteration $(n-1)$. For the pressure gradient this means to take the pressure of the antecedent outer iteration as a first guess for the following iteration. This guess has to be corrected in an iterative way until all the non-linear equations are fulfilled up to a certain tolerance. Section (6.4.2) presents a suitable convergence criterion and its implementation. This linearization process in conjunction with the pressure guess leads to the linear equation

$$u_{i,P}^{(n*)} = -\frac{\alpha_{\mathbf{u}}}{a_P^{u_i}} \left( \sum_{F \in NB(P)} a_F^{u_i} u_{i,F}^{(n*)} + b_{P,u_i}^{(n-1)} - V_P \left( \frac{\partial p}{\partial x_i} \right)_P^{(n-1)} \right) + (1-\alpha_{\mathbf{u}}) u_{i,P}^{(n-1)}. \tag{23}$$

Here $(*)$ indicates that the solution of this equation still needs to be corrected to also fulfill the discretized mass balance

$$\sum_{F \in NB(P)} (u_i)_f^{(n)} n_i S_f = 0. \tag{24}$$

Applying the same procedure as in section 4.2 to equation (23) results in the following expression for the face velocities after solving the discretized momentum balances using as pressure guess the pressure from the previous outer iteration

$$\begin{aligned} u_{i,f}^{(n*)} = & \left[ (1-\gamma_f) u_{i,P}^{(n*)} + \gamma_f u_{i,Q}^{(n*)} \right] \\ & - \left( (1-\gamma_f) \frac{\alpha_{\mathbf{u}} V_P}{a_P^{u_i}} + \gamma_f \frac{\alpha_{\mathbf{u}} V_Q}{a_Q^{u_i}} \right) \left[ \left( \frac{\partial p}{\partial x_i} \right)_f^{(n-1)} - (1-\gamma_f) \left( \frac{\partial p}{\partial x_i} \right)_P^{(n-1)} - \gamma_f \left( \frac{\partial p}{\partial x_i} \right)_Q^{(n-1)} \right] \\ & + (1-\alpha_{\mathbf{u}}) \left[ u_{i,f}^{(n-1)} - (1-\gamma_f) u_{i,P}^{(n-1)} - \gamma_f u_{i,Q}^{(n-1)} \right]. \end{aligned} \tag{25}$$

The lack of an equation with the pressure as dominant variable leads to the necessity to alter the mass balance as the only equation left. Methods of this type are called projection methods. A common class of algorithms of this family of methods uses an equation for the additive pressure correction $p'$ instead of the pressure itself and enforces continuity by correcting the velocities with an additive corrector $u_i'$, such that

$$u_{i,P}^{(n)} = u_{i,P}^{(n*)} + u_{i,P}', \quad u_{i,f}^{(n)} = u_{i,f}^{(n*)} + u_{i,f}' \quad \text{and} \quad p_P^{(n)} = p_P^{(n-1)} + p_P'.$$

It is now possible to formulate the discretized momentum balance for the corrected velocities and the corrected pressure as

$$u_{i,P}^{(n)} = -\frac{\alpha_{\mathbf{u}}}{a_P^{u_i}} \left( \sum_{F \in NB(P)} a_F^{u_i} u_{i,F}^{(n)} + b_{P,u_i}^{(n-1)} - V_P \left( \frac{\partial p}{\partial x_i} \right)_P^{(n)} \right) + (1-\alpha_{\mathbf{u}}) u_{i,P}^{(n-1)}. \tag{26}$$

It should be noted that the only difference to the equation which will be solved in the next outer iteration is that the source term $b_{P,u_i}$ has not been updated yet. In the same way the discretized momentum balance for the face velocity $u_{i,f}$ can be formulated as

$$\begin{aligned} u_{i,f}^{(n)} = & \left[ (1-\gamma_f) u_{i,P}^{(n)} + \gamma_f u_{i,Q}^{(n)} \right] \\ & - \left( (1-\gamma_f) \frac{\alpha_{\mathbf{u}} V_P}{a_P^{u_i}} + \gamma_f \frac{\alpha_{\mathbf{u}} V_Q}{a_Q^{u_i}} \right) \left[ \left( \frac{\partial p}{\partial x_i} \right)_f^{(n)} - (1-\gamma_f) \left( \frac{\partial p}{\partial x_i} \right)_P^{(n)} - \gamma_f \left( \frac{\partial p}{\partial x_i} \right)_Q^{(n)} \right] \\ & + (1-\alpha_{\mathbf{u}}) \left[ u_{i,f}^{(n-1)} - (1-\gamma_f) u_{i,P}^{(n-1)} - \gamma_f u_{i,Q}^{(n-1)} \right]. \end{aligned} \tag{27}$$

To couple velocity and pressure correctors one can subtract equation (23) from (26) and equation (25) from (27) to get

$$u'_{i,P} = -\frac{\alpha_{\mathbf{u}}}{a_P^{u_i}} \left( \underline{\sum_{F \in NB(P)} a_F^{u_i} u'_{i,F}} - V_P \left( \frac{\partial p'}{\partial x_i} \right)_P^{(n)} \right) \quad \text{and} \tag{28}$$

$$u'_{i,f} = \left[ (1-\gamma_f) u'_{i,P} + \gamma_f u'_{i,Q} \right] - \left( (1-\gamma_f) \frac{\alpha_{\mathbf{u}} V_P}{a_P^{u_i}} + \gamma_f \frac{\alpha_{\mathbf{u}} V_Q}{a_Q^{u_i}} \right) \left[ \left( \frac{\partial p}{\partial x_i} \right)'_f - (1-\gamma_f) \left( \frac{\partial p}{\partial x_i} \right)'_P - \gamma_f \left( \frac{\partial p}{\partial x_i} \right)'_Q \right]. \tag{29}$$

The majority of the class of pressure correction algorithms has this equations as a common basis. Each algorithm then introduces special distinguishable approximations of the velocity corrections that are, at the moment of solving the pressure equation, still unknown. The method used in the present work is the SIMPLE Algorithm (Semi-Implicit Method for Pressure-Linked Equations [48]). The approximation this algorithm performs is severe since the term containing the unknown velocity corrections is dropped entirely. The respective term has been underlined in equation (28). Since the global purpose of the presented method is to enforce continuity by implicitly calculating a pressure correction, the velocity correction has to be expressed solely in terms of the pressure correction. This can be accomplished by inserting equation (28) into equation (29). This gives an update formula

$$u'_{i,f} = -\left( (1-\gamma_f) \frac{\alpha_{\mathbf{u}} V_P}{a_P^{u_i}} + \gamma_f \frac{\alpha_{\mathbf{u}} V_Q}{a_Q^{u_i}} \right) \left( \frac{\partial p}{\partial x_i} \right)'_f, \tag{30}$$

which is then, together with (25), inserted into the discretized continuity equation (24) to obtain

$$\sum_{F \in NB(P)} \left( (1-\gamma_f) \frac{\alpha_{\mathbf{u}} V_P}{a_P^{u_i}} + \gamma_f \frac{\alpha_{\mathbf{u}} V_F}{a_F^{u_i}} \right) \left( \frac{\partial p}{\partial x_i} \right)'_f n_i S_f = b_{p,p} \quad , \tag{31}$$

where the right hand side $b_{p,p}$ is defined as

$$b_{p,p} := \sum_{F \in NB(P)} u_{i,f}^{(n*)} n_i S_f. \tag{32}$$

The complete discretization with central differences as approximation for the gradient of the pressure correction is straightforward and will be presented in subsection 4.4.

The approximation performed in the SIMPLE algorithm affects convergence in a way that the pressure correction has to be under-relaxed with a parameter $\alpha_p \in [0,1]$

$$p_P^{(n)} = p_P^{(n-1)} + \alpha_{\mathbf{p}} p'_P. \tag{33}$$

It should be noted that there are better approximations for the pressure correction which will not rely on pressure under-relaxation for convergence. On example that uses a more consistent approximation is the *SIMPLEC* algorithm (SIMPLE Consistent) [64]. A similar derivation of the pressure weighted interpolation method for the SIMPLEC algorithm can be found in [45].

As shown in section 4.2 the behaviour of the pressure weighted interpolation method on non-equidistant grids can be improved by replacing the linear interpolation of pressure gradients with a simple average in equation (25). This leads to the following equation for calculating mass fluxes

$$\begin{aligned} u_{i,f}^{(n*)} = & \left[ (1-\gamma_f) u_{i,P}^{(n*)} + \gamma_f u_{i,Q}^{(n*)} \right] \\ & - \left( (1-\gamma_f) \frac{\alpha_{\mathbf{u}} V_P}{a_P^{u_i}} + \gamma_f \frac{\alpha_{\mathbf{u}} V_Q}{a_Q^{u_i}} \right) \left[ \left( \frac{\partial p}{\partial x_i} \right)_f^{(n-1)} - \frac{1}{2} \left( \frac{\partial p}{\partial x_i} \right)_P^{(n-1)} - \frac{1}{2} \left( \frac{\partial p}{\partial x_i} \right)_Q^{(n-1)} \right] \\ & + (1-\alpha_{\mathbf{u}}) \left[ u_{i,f}^{(n-1)} - (1-\gamma_f) u_{i,P}^{(n-1)} - \gamma_f u_{i,Q}^{(n-1)} \right]. \end{aligned} \tag{34}$$

Generally the SIMPLE algorithm can be represented by an iterative procedure as shown in Algorithm 5.1.

**Algorithm 1** SIMPLE Algorithm

*INITIALIZE* variables
**while** (convergence criterion not accomplished) **do**
  *SOLVE* linearized momentum balances, equation **(23)**
  *CALCULATE* mass fluxes using **(27)** or **(34)**
  *SOLVE* pressure correction equation to assure continuity, equation **(31)**
  *UPDATE* pressure using **(33)**
  *UPDATE* velocities and mass fluxes using **(28)**
  **if** (coupled scalar equation) **then**
    *SOLVE* scalar equation as described in **(4.6)**
  **end if**
**end while**

## 4.4 Discretization of the Mass Fluxes and the Pressure Correction Equation

Subsections 4.2 and 4.3 introduced the concept of pressure weighted interpolation to avoid oscillating results and an algorithm to calculate a velocity field that obeys continuity. The derived equations have not been discretized completely, furthermore the approach has not been generalized to non-orthogonal grids.

The discretized mass balance (16) only depends on the normal velocities $u_{i,f}n_{i,f}$. By analogy with equation (34) an interpolated normal face velocity and thus the mass flux can be calculated as

$$\begin{aligned} u_{i,f}^{(n*)}n_{i,f} = &\left[\left(1-\gamma_f\right)u_{i,P}^{(n*)} + \gamma_f u_{i,Q}^{(n*)}\right]n_{i,f} \\ &-\left(\left(1-\gamma_f\right)\frac{\alpha_{\mathbf{u}}V_P}{a_P^{u_i}} + \gamma_f\frac{\alpha_{\mathbf{u}}V_Q}{a_Q^{u_i}}\right)\left[\left(\frac{\partial p}{\partial n}\right)_f^{(n-1)} - \frac{1}{2}\left(\frac{\partial p}{\partial n}\right)_P^{(n-1)} - \frac{1}{2}\left(\frac{\partial p}{\partial n}\right)_Q^{(n-1)}\right] \\ &+(1-\alpha_{\mathbf{u}})\left[u_{i,f}^{(n-1)} - \left(1-\gamma_f\right)u_{i,P}^{(n-1)} - \gamma_f\, u_{i,Q}^{(n-1)}\right]n_{i,f}, \end{aligned} \tag{35}$$

where the scalar product of pressure gradients and the normal vector has been replaced by a directional derivative in the direction of the face normal vector. In the present work pressure gradients in (35) and pressure correction gradients in equation (31) will be discretized by central differences

$$\left(\frac{\partial p}{\partial n}\right)_f \approx \frac{p_P - p_Q}{\left(\mathbf{x}_P - \mathbf{x}_Q\right)\cdot\mathbf{n}_f} \quad \text{and} \quad \left(\frac{\partial p'}{\partial n}\right)_f \approx \frac{p'_P - p'_Q}{\left(\mathbf{x}_P - \mathbf{x}_Q\right)\cdot\mathbf{n}_f}.$$

This discretization can then be inserted into the semi-discretized pressure correction equation (31)

$$\sum_{F\in NB(P)}\left(\left(1-\gamma_f\right)\frac{\alpha_{\mathbf{u}}V_P}{a_P^{u_i}} + \gamma_f\frac{\alpha_{\mathbf{u}}V_F}{a_F^{u_i}}\right)\frac{p'_P - p'_F}{\left(\mathbf{x}_P - \mathbf{x}_F\right)\cdot\mathbf{n}_f}S_f = b_{P,p}. \tag{36}$$

The resulting coefficients for the pressure correction equation

$$a_P^{p'}p'_P + \sum_{F\in NB(P)} a_F^{p'}p'_F = b_{P,p'},$$

can be calculated as

$$a_F^{p'} = -\left(\left(1-\gamma_f\right)\frac{\alpha_{\mathbf{u}}V_P}{a_P^{u_i}} + \gamma_f\frac{\alpha_{\mathbf{u}}V_F}{a_F^{u_i}}\right)\frac{S_f}{\left(\mathbf{x}_P - \mathbf{x}_F\right)\cdot\mathbf{n}_f} \quad \text{and} \quad a_P^{p'} = -\sum_{F\in NB(P)} a_F^{p'}. \tag{37}$$

The right hand side can be calculated as in equation (32), if the presented discretization is applied.

## 4.5 Discretization of the Momentum Balance

The stationary momentum balance integrated over a single control volume $P$ reads as

$$\underbrace{\iint_S (\rho u_i u_j) n_j \mathrm{d}S}_{\text{convective term}} - \underbrace{\iint_S \left(\mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right)\right) n_j \mathrm{d}S}_{\text{diffusive term}} = - \underbrace{\iiint_V \frac{\partial p}{\partial x_i} \mathrm{d}V}_{\text{pressure sourceterm}} - \underbrace{\iiint_V \rho\beta\,(T - T_0)\,\mathrm{d}V}_{\text{temperature sourceterm}}, \tag{38}$$

where the different terms to be addressed individually in the following sections are indicated. The reader should note that the form of this equation has been modified by using Gauss' integration theorem. The terms residing on the left will be treated in an implicit and due to deferred corrections also in an explicit way, whereas the terms on the right will be treated exclusively in an explicit way.

### 4.5.1 Linearization and Discretization of the Convective Term

The convective term $\left(\rho u_i u_j\right)$ of the Navier-Stokes equations is the reason for the non-linearity of the equations. In order to deduce a set of linear algebraic equations from the Navier-Stokes equations this term has to be linearized. As introduced in section (3.4), the non linearity will be dealt with by means of an iterative process, the Picard iteration. The part dependent on the non dominant dependent variable therefore will be approximated by its value from the previous iteration as $\rho u_i^{(n)} u_j^{(n)} \approx \rho u_i^{(n)} u_j^{(n-1)}$. However this linearization will not be directly visible because it will be covered by the mass flux $\dot{m}_f = \iint_{S_f} \rho u_j^{(n-1)} n_j \mathrm{d}S$.

Using the additivity of the Riemann integral the first step is to decompose the surface integral into individual contributions from each boundary face of the control volume $P$

$$\iint_S \rho u_i u_j n_j \mathrm{d}S = \sum_{f \in \{w,s,b,t,n,e\}} \iint_{S_f} \rho u_i u_j n_j \mathrm{d}S = \sum_{f \in \{w,s,b,t,n,e\}} F_{i,f}^c,$$

where $F_{i,f}^c := \iint_{S_f} \rho u_i^{(n)} u_j^{(n-1)} n_j \mathrm{d}S$ is the convective flux of the velocity $u_i$ through the boundary face $S_f$.

To improve diagonal dominance of the resulting linear system while maintaining the smaller discretization error of a higher order discretization, a blended discretization scheme is applied and combined with a deferred correction. Since due to the non-linearity of the equations to be solved, an iterative solution process is needed by all means, the overall convergence does not degrade noticeably when using a deferred correction [22]. Blending and deferred correction result in a decomposition of the convective flux into a lower order approximation, which is treated implicitly, and the explicit difference between the higher and lower order approximation for the same convective flux. Since for coarse grid resolutions the use of higher order approximations may lead to oscillations of the solution, which in turn may degrade or even impede convergence, the schemes can be blended by a control factor $\eta \in [0, 1]$. Furthermore the use of low-order approximations increases the diagonal dominance of the matrix, which then benefits the convergence of iterative solvers for the linear systems [54].

To show the generality of this approach all further derivations are presented for the generic boundary face $S_f$ that separates control volume $P$ from its neighbour $F \in NB(P)$. This decomposition then leads to

$$F_{i,f}^c \approx \underbrace{F_{i,f}^{c,l}}_{\text{implicit}} + \eta \underbrace{\left[F_{i,f}^{c,h} - F_{i,f}^{c,l}\right]^{(n-1)}}_{\text{explicit}}.$$

It should be noted that the convective fluxes carrying an $l$ for *lower* or an $h$ for *higher* as exponent, already have been linearized and discretized. The discretization applied to the convective flux in the present work is using the midpoint integration rule and blends the upwind interpolation scheme with a linear interpolation scheme. Applied to above decomposition one can derive the following approximations

$$\begin{aligned} F_{i,f}^{c,l} &= u_{i,F} \min(\dot{m}_f, 0) + u_{i,P} \max(0, \dot{m}_f) \\ F_{i,f}^{c,h} &= u_{i,F}\, \gamma_f + u_{i,P}\,(1 - \gamma_f), \end{aligned}$$

where the variable values have to be taken from the previous iteration step $(n-1)$ as necessary and the mass flux $\dot{m}_f$ has been used as result of the linearization process. The results can now be summarized by presenting the convective contribution to the matrix coefficients $a_{F,u_i}$ and $a_P^{u_i}$ and the right hand side $b_{P,u_i}$ which are calculated as

$$a_F^{u_i,c} = \min(\dot{m}_f, 0), \qquad a_P^{u_i,c} = \sum_{F \in NB(P)} \max(0, \dot{m}_f) \tag{39a}$$

$$\text{and} \quad b_{P,u_i}^c = \sum_{F \in NB(P)} \eta\left(u_{i,F}^{(n-1)}\left(\min(\dot{m}_f, 0) - \gamma_f\right)\right) + \eta\left(u_{i,P}^{(n-1)}\left(\max(0, \dot{m}_f) - \left(1-\gamma_f\right)\right)\right). \tag{39b}$$

### 4.5.2 Discretization of the Diffusive Term

The diffusive term contains the first partial derivatives of the velocity as a result of the material constitutive equation that characterizes the behaviour of Newtonian fluids. As pointed out in section 3.3, directional derivatives can be discretized using central differences on orthogonal grids or in the more general case of non-orthogonal grids using central differences implicitly and an explicit deferred correction comprising the non-orthogonality of the grid. As seen in equation (8) the diffusive term of the Navier-Stokes equations can be simplified using the mass balance in the case of an incompressible flow with constant viscosity $\mu$. To sustain the generality of the presented approach this simplification will be omitted.

As before, by using the additivity and furthermore linearity of the Riemann integral, the integration of the diffusive term will be divided into integration over individual boundary faces $S_f$

$$\iint_S \left(\mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right)\right) n_j \mathrm{d}S\mathrm{d}S = \sum_{f \in \{w,s,b,t,n,e\}} \left[\iint_{S_f} \mu \underline{\frac{\partial u_i}{\partial x_j} n_j \mathrm{d}S} + \iint_{S_f} \mu \frac{\partial u_j}{\partial x_i} n_j \mathrm{d}S\right] = \sum_{f \in \{w,s,b,t,n,e\}} F_{i,f}^d,$$

where $F_{i,f}^d$ denotes the diffusive flux through an individual boundary face. Section 3.3 only covered the non-orthogonal corrector for directional derivatives. Since the velocity is a vector field and not a scalar field, the results of section 3.3 may only be applied to the underlined term. The other term will be treated explicitly since it is considerably smaller than the underlined term, furthermore does not cause oscillations and thus will not derogate convergence [22]. First all present integrals will be approximated using the midpoint integration rule. The diffusive flux $F_{i,f}^d$ for a generic face $S_f$ located between the control volumes $P$ and $F$ then reads

$$F_{i,f}^d \approx \underline{\mu\left(\frac{\partial u_i}{\partial x_j}\right)_f n_j S_f} + \mu\left(\frac{\partial u_j}{\partial x_i}\right)_f n_j S_f.$$

Using central differences for the implicit discretization of the directional derivative and furthermore using the *orthogonal correction* approach from 3.3.2 the approximation can be derived as

$$F_{i,f}^d \approx \mu\left(\underline{||\boldsymbol{\Delta}_f||_2 \frac{u_{P_i} - u_{F_i}}{||\mathbf{x}_P - \mathbf{x}_F||_2} - (\nabla u_i)_f^{(n-1)} \cdot \left(\boldsymbol{\Delta}_f - \mathbf{S}_f\right)}\right) + \mu\left(\frac{\partial u_j}{\partial x_i}\right)_f^{(n-1)} n_{f_i}$$

$$= \mu\left(\underline{S_f \frac{u_{P_i} - u_{F_i}}{||\mathbf{x}_P - \mathbf{x}_F||_2} - \left(\frac{\partial u_i}{\partial x_j}\right)_f^{(n-1)} \left(\xi_{f_i} - n_{f_i}\right) S_f}\right) + \mu\left(\frac{\partial u_j}{\partial x_i}\right)_f^{(n-1)} n_{f_i},$$

where the unit vector pointing in direction of the straight line connecting control volume $P$ and control volume $F$ is denoted as

$$\xi_f = \frac{\mathbf{x}_P - \mathbf{x}_F}{||\mathbf{x}_p - \mathbf{x}_F||_2}.$$

The interpolation of the cell center gradients to the boundary faces is performed as in (13). Now the contribution of the diffusive part to the matrix coefficients and the right hand side can be calculated as

$$a_F^{u_i,d} = -\frac{\mu S_f}{||\mathbf{x}_P - \mathbf{x}_F||_2}, \qquad a_P^{u_i,d} = \sum_{F \in NB(P)} \frac{\mu S_f}{||\mathbf{x}_P - \mathbf{x}_F||} \tag{40a}$$

$$\begin{aligned} b_{F,u_i}^d &= \sum_{F \in NB(P)} \left(\frac{\partial u_i}{\partial x_j}\right)_f^{(n-1)} \left(\xi_{f_i} - n_{f_i}\right) S_f - \mu \left(\frac{\partial u_j}{\partial x_i}\right)_f^{(n-1)} n_{f_i} S_f \\ &= \sum_{F \in NB(P)} \left(\frac{\partial u_i}{\partial x_j}\right)_f^{(n-1)} \xi_{f_i} S_f - \mu \left(\left(\frac{\partial u_i}{\partial x_j}\right)_f^{(n-1)} - \left(\frac{\partial u_j}{\partial x_i}\right)_f^{(n-1)}\right) n_{f_i} S_f . \end{aligned} \tag{40b}$$

### 4.5.3 Discretization of the Source Terms

Since in the segregated solution approach in every equation all other variables but the dominant one are treated as constants and furthermore the source terms in equation (38) do not depend on the dominant variable the discretization is straightforward. The source terms of the momentum balance are discretized using the midpoint integration rule for volume integrals, which leads to the source term

$$-\iiint_V \frac{\partial p}{\partial x_i} \mathrm{d}V - \iiint_V \rho\beta\left(T - T_0\right) \mathrm{d}V \approx -\left(\frac{\partial p}{\partial x_i}\right)_P^{(n-1)} V_P - \rho\beta\left(T_P^{(n-1)} - T_0\right) V_P = b_{P,u_i}^{sc}. \tag{41}$$

## 4.6 Discretization of the Temperature Equation

The discretization of the temperature equation is performed by the same means as for the momentum balance. The only difference is a simpler diffusion term. The integral form of the temperature equation, after applying the Gauss' theorem of integration, is

$$\underbrace{\iint_S \rho u_j T n_j \mathrm{d}S}_{\text{advective term}} - \underbrace{\iint_S \kappa \frac{\partial T}{\partial x_j} n_j \mathrm{d}V}_{\text{diffusive term}} = \underbrace{\iiint_V q_T \mathrm{d}V}_{\text{source term}} .$$

Proceeding as in the previous subsections one can now discretize the advective, the diffusive term and the source term. Since this process does not provide further insight, just the final results will be presented. The discretization yields the matrix coefficients as

$$a_F^T = \min(\dot{m}_f, 0) + \frac{\kappa S_f}{||\mathbf{x}_P - \mathbf{x}_F||_2} \tag{42a}$$

$$a_P^T = \sum_{F \in NB(P)} \max(0, \dot{m}_f) - \frac{\kappa S_f}{||\mathbf{x}_P - \mathbf{x}_F||_2} \tag{42b}$$

$$\begin{aligned} b_{P,T} = &\sum_{F \in NB(P)} \eta\left(T_F^{(n-1)}\left(\min(\dot{m}_f, 0) - \gamma_f\right)\right) + \eta\left(T_P^{(n-1)}\left(\max(0, \dot{m}_f) - \left(1 - \gamma_f\right)\right)\right) \\ &+ \sum_{F \in NB(P)} \left(\frac{\partial T}{\partial x_j}\right)_f^{(n-1)} \left(\xi_{f_j} - n_{f_j}\right) S_f \\ &+ q_{T_P} V_P . \end{aligned} \tag{42c}$$

Again it is possible though not always necessary, as in the case of the velocities, to under-relax the solution of the resulting linear system with a factor $\alpha_T$. This can be accomplished as shown in subsection 4.9.

## 4.7 Boundary Conditions

As the antecedent subsections showed, it is possible to deduce a linear algebraic equation for each control volume by the finite volume method. The approach presented in the preceding subsections however did not cover the treatment of control volumes at domain boundaries yet. This subsection introduces the boundary conditions which are relevant for the present work and furthermore deals with transitional conditions at block boundaries.

### 4.7.1 Dirichlet Boundary Conditions

The first boundary condition is the Dirichlet boundary condition. This type of boundary condition is used to model inlet conditions for flow problems. For the temperature equation it may also be used at walls as will be shown in subsection 4.7.2. It is characterized by specifying the value of the variable for which the equation is solved explicitly. As a result boundary fluxes can be calculated directly. Especially the mass flux $\dot{m}_f$ is known and hence does not have to be calculated using the pressure weighted interpolation method. Since no special modifications have to be made, as the resulting coefficient for a neighbouring control volume laying past the boundary is considered on the right hand side of the linear system, the implementation approach will be presented only for the temperature equation. Since there is no boundary condition that fixes the gradient at Dirichlet boundaries it is assumed that the partial derivatives of the respective variable are constant and can hence be extrapolated

$$\left(\frac{\partial T}{\partial x_j}\right)_f \approx \left(\frac{\partial T}{\partial x_j}\right)_P .$$

The modification to the central coefficient of the linear equation can be recursively formulated as

$$a_P^T = a_P^T + \left(\max(0,\dot{m}_f) - \frac{\kappa S_f}{||\mathbf{x}_P - \mathbf{x}_f||_2}\right),$$

whereas the contribution to the right hand side reads

$$b_{P,T} = b_{P,T} - \left(\min(\dot{m}_f,0) - \frac{\kappa S_f}{||\mathbf{x}_P - \mathbf{x}_f||_2}\right) T_f + \left(\frac{\partial T}{\partial x_j}\right)_P^{(n-1)} \left(\xi_{f_j} - n_{f_j}\right) S_f$$

The reader should note, that even though the gradient discretization at domain boundaries is realized by a one sided forward differencing scheme instead of a central differencing scheme. This does not drastically affect accuracy because the distance used in the differential quotient is half the distance used on a central difference inside the domain [54].

### 4.7.2 Treatment of Wall Boundaries

A common boundary to the solution domain is given by solid walls. For all kind of flows this boundary condition first of all has a kinematic character, since the concept of impermeable walls dictates a zero normal velocity at the wall. In viscous flows wall boundaries can be interpreted furthermore as a no-slip condition, i.e. a Dirichlet boundary condition for the velocities. Convective fluxes through solid walls are thus zero by definition however the diffusive fluxes require special treatment not only for the velocities but also for the temperature. To approximate the fluid behavior on a wall boundary correctly, special modifications have to be taken into account to model the normal a shear tension. Furthermore diffusive fluxes for the temperature can be given by Neumann or Dirichlet boundary conditions.

The derivation of the discretized diffusive flux through wall boundaries starts from the integral momentum balance (2) for the vector $\mathbf{u}$. Here only the term for surface forces is needed

$$\mathbf{F}_w = \iint_{S_w} \mathbf{t}\mathrm{d}S = \iint_{S_w} \mathbf{T}(\mathbf{n}_w)\mathrm{d}S \tag{43}$$

For the purpose of treating wall boundary conditions it is appropriate to use a local coordinate system $n,t,s$ where $n$ denotes the wall normal coordinate, $t$ denotes the coordinate tangential to the wall shear force and, $s$ is the binormal coordinate. (FIGURE). With respect to this coordinate system the wall normal vector is represented by $\mathbf{n}_w = (1,0,0)^T$ and the image of the wall normal vector $\mathbf{T}(\mathbf{n_w})$ is represented by

$$\mathbf{T}(\mathbf{n}_w) = \begin{bmatrix} \tau_{nn} & \tau_{nt} & \tau_{ns} \\ \tau_{nt} & \tau_{tt} & \tau_{ts} \\ \tau_{ns} & \tau_{ts} & \tau_{ss} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} \tau_{nn} \\ \tau_{nt} \\ \tau_{ns} \end{bmatrix}$$

The velocity of the wall is assumed to be constant, so the directional derivative of the tangential velocity vanishes on the wall

$$\tau_{tt} = \frac{\partial u_t}{\partial x_t} = 0,$$

which in conjunction with the continuity equation in differential form leads to

$$\frac{\partial u_n}{\partial x_n} + \frac{\partial u_t}{\partial x_t} = \frac{\partial u_n}{\partial x_n} = 0,$$

what is equivalent to $\tau_{nn} = 0$ at the wall. A physical interpretation would be that the transfer of momentum at the wall occurs by shear forces exclusively. Furthermore the coordinate direction $t$ is chosen to be parallel to the shear force which is no restriction because of the possibility to rotate the coordinate system within the plane. This leads to $\tau_{ns} = 0$. The absolute value of the surface force hence only depends on the normal derivative of the velocity tangential to the wall. After transforming the coordinates back to the system $(x_1, x_2, x_3)$ the surface force can be calculated and the integral can be discretized using the midpoint integration rule by

$$\mathbf{F}_w = \iint_{S_w} \mathbf{t_w}\, \tau_{nt} \mathrm{d}S = \iint_{S_w} \mathbf{t_w}\, \mu \frac{\partial u_t}{\partial n} \mathrm{d}S = \mathbf{t_w}\, \mu \left(\frac{\partial u_t}{\partial n}\right)_w S_w, \tag{44}$$

where $\mathbf{t_w}$ denotes the transformed tangential vector $(0,1,0)^T$ with respect to the coordinate system $(x_1, x_2, x_3)$. In the discretization process this tangential vector will be calculated from the velocity vector as

$$\mathbf{t_w} = \frac{\mathbf{u}_t}{||\mathbf{u}_t||_2} \quad \text{, where} \quad \mathbf{u}_t = \mathbf{u} - (\mathbf{u} \cdot \mathbf{n}_w)\, \mathbf{n}_w.$$

According to [22] this force should not be handled explicitly for convergence reasons. On the other side, if the surface force is expressed by the velocities $u_i$ the discretization process would lead to different central matrix coefficients, which would affect memory efficiency.

The discretization approach used in the present thesis uses a simpler implicit discretization coupled with a deferred correction that combines the explicit discretization of the surface force (44) with a central difference. At first the contained directional derivative is discretized implicitly by

$$\left(\frac{\partial u_t}{\partial n}\right) t_{wi} \approx \left(\frac{\partial u_i}{\partial \xi}\right) \approx \frac{u_{i,P} - u_{i,w}}{||\mathbf{x}_P - \mathbf{x}_w||_2}$$

and explicitly by

$$\left(\frac{\partial u_t}{\partial n}\right) t_{wi} \approx \frac{\left(u_{i,P} - u_{i,w}\right) - \left(u_{j,P} - u_{j,w}\right) n_j n_i}{(\mathbf{x}_P - \mathbf{x}_w) \cdot \mathbf{n}_w}.$$

Therefore the contributions to the central coefficient and the right hand side of the linear equation that results from the presented discretization process are

$$a_P^{u_i} = a_P^{u_i} + \mu \frac{S_f}{||\mathbf{x}_P - \mathbf{x}_w||_2} \quad \text{and}$$

$$b_{P,u_i} = b_P^{u_i} + \mu \frac{S_f}{||\mathbf{x}_P - \mathbf{x}_w||_2} u_{i,P}^{(n-1)} + \frac{\left(u_{i,P}^{(n-1)} - u_{i,w}\right) - \left(u_{j,P}^{(n-1)} - u_{j,w}\right) n_j n_i}{(\mathbf{x}_P - \mathbf{x}_w) \cdot \mathbf{n}_w}.$$

The reader should note that since the deferred correction uses a Dirichlet boundary condition, no correction of the value of the wall velocity $u_{i,w}$ has to be accounted for.

If the solution of the flow field is coupled to the solution of a temperature equation, different options for the boundary condition at walls may be chosen. If the wall temperature is known, a Dirichlet boundary condition for the temperature is the choice. A wall of this type is called to be *isothermal*. If on the other side only the heat flux is known a Neumann boundary condition is used. In the special case of zero heat flux the wall is called to be *adiabatic*. For adiabatic walls, which are besides isothermal walls used for the present thesis the implementation is straight forward since no coefficients have to be calculated.

### 4.7.3 Treatment of Block Boundaries

If block structured grids are used to decompose the problem domain, the characterizing property of structured grids, which is the constant amount of grid cells in each direction, gives each inner cell exactly six neighbours. If more then one grid block is used this property is violated if the number of grid cells of each block is chosen to be arbitrary. The arbitrariness of the grid resolution is a main benefactor for the adaptivity of block structured grids.

To maintain the conservation property of finite volume methods block boundaries should not be interpreted as boundaries in the classical sense. Instead conditions have to be formulated to guarantee that flows through block boundaries are conserved. The method to treat block boundaries used in the present thesis was presented by [38]. Another method was presented in [36]. However the first method was chosen since it allows a fully implicit consideration of the block boundary fluxes. This method calculates fluxes through separate face segments $S_l$ that come up on *non-matching* grid blocks. Each of this face segments gets assigned a control volume $L$ and a control volume $R$ which exclusively share this face segment. Figure **??** shows a block boundary when non matching blocks are used. An algorithm to provide this needed information will be presented in REFERENCE.

Once the geometric data including the information regarding the interpolation has been provided, the fluxes through this boundaries can be calculated as presented in the antecedent subsections. The connectivity of neighbouring control volumes of different grid blocks is represented by matrix coefficients $a_L^\phi$ and $a_R^\phi$, where $\phi \in \{u_1, u_2, u_3, p, T\}$.

## 4.8 Treatment of the Singularity of the Pressure Correction Equation with Neumann Boundaries

It has to be noted that the derived pressure correction equation is a Poisson equation. As can be proven **??**, the linear $N \times N$ systems surging from the presented discretization on a grid with $N$ control volumes have a nullspace of dimension one, i.e.

$$\mathrm{null}(A_{p'}) = \mathrm{span}(\mathbb{1}),$$

where $\mathbb{1} = (1)_{i=1,\dots,N} \in \mathbb{R}^N$ is the vector spanning the nullspace. This singularity accounts for the property of incompressible flows, that pressure can only be determined up to a constant. To fix this constraint various possibilities exist [22]. A common method is to set the pressure correction to zero in one reference control volume and hence fix the pressure at one reference point in the problem domain. This can be done before solving the system by applying this Dirichlet-type condition, or it can be done afterwards when pressure is calculated from the pressure correction. This approach is not suitable for grid convergence studies since without proper interpolation it is not guaranteed that the reference pressure correction is taken at the correct location.

Since some comparisons performed in the present work rely on grid convergence studies another approach for reducing the lose constraint of the pressure correction system has been used: The reference pressure correction is taken to be the mean value of the pressure correction over the domain

$$p'_{\mathrm{ref}} = \frac{\iiint_V p' \mathrm{d}V}{\iiint_V \mathrm{d}V} \approx \frac{\sum_{P=1}^N p'_P V_P}{\sum_{P=1}^N V_P}$$

such that the net pressure correction amounts to zero. This modifies equation (33) to read

$$p_P^{(n)} = p_P^{(n-1)} + \alpha_{\mathbf{p}} \left( p'_P - p'_{\mathrm{ref}} \right). \tag{45}$$

## 4.9 Structure of the Assembled Linear Systems

The objective of a finite volume method is to create a set of linear algebraic equations by discretizing partial differential equations. In the case of the discretized momentum balance, taking all contributions together leads to the following linear algebraic equation for each control volume $P$

$$a_P^{u_i} u_{P_i} + \sum_{F \in NB(P)} a_F u_{F_i} = b_{P,u_i},$$

where the coefficients are composed as

$$a_P^{u_i} = a_P^{u_i,c} - a_P^{u_i,d} \tag{46a}$$

$$a_F^{u_i} = a_F^{u_i,c} - a_F^{u_i,d} \tag{46b}$$

$$b_{P,u_i} = b_{P,u_i,c} - b_{P,u_i,d} + b_{P,u_i}^{sc}. \tag{46c}$$

Similar expressions for the pressure correction equation and the temperature equation exist. In the case of control volumes located at boundaries some of the coefficients will be calculated in a different way. This aspect is addressed in section 4.7.

For the decoupled iterative solution process of the Navier-Stokes equations it is necessary to reduce the change of each dependent variable in each iteration. Normally this is done by an *under-relaxation* technique, a convex combination of the solution of the linear system for the present iteration $(n)$ and from the previous iteration $(n-1)$ with the under-relaxation parameter $\alpha_{u_i} \in (0,1]$, where $\alpha_{u_i} = 1$ refers to no under-relaxation. Generally speaking this parameter can be chosen individually for each equation. Since there are no rules for choosing this parameters in a general setting the under-relaxation parameter for the velocities is chosen to be equal for all three velocities, $\alpha_{u_i} = \alpha_{\mathbf{u}}$ [54]. This has the further advantage that, in case the boundary conditions are implemented with the same intention, the linear system for each of the velocities remains unchanged except for the right hand side. This helps to increase memory efficiency.

Let the solution for the linear system without under-relaxation be denoted as

$$\tilde{u}_{P_i}^{(n)} := \frac{b_{P,u_i} - \sum_{F \in NB(P)} a_F u_{F_i}}{a_P^{u_i}},$$

Which is only a formal expression for the case the underlying linear system is solved exactly. A convex combination as described yields

$$\begin{aligned} u_{P_i}^{(n)} &:= \alpha_{\mathbf{u}} \tilde{u}_{P_i}^{(n)} + (1-\alpha_{\mathbf{u}}) u_{P_i}^{(n-1)} \\ &= \alpha_{\mathbf{u}} \frac{b_{P,u_i} - \sum_{F \in NB(P)} a_F u_{F_i}}{a_P^{u_i}} + (1-\alpha_{\mathbf{u}}) u_{P_i}^{(n-1)}, \end{aligned}$$

an expression that can be modified to derive a linear system whose solution is the under-relaxed velocity

$$\frac{a_P^{u_i}}{\alpha_{\mathbf{u}}} u_{i,P} + \sum_{F \in NB(P)} a_F^{u_i} u_{i,F} = b_{P,u_i} + \frac{(1-\alpha_{\mathbf{u}}) a_P^{u_i}}{\alpha_{\mathbf{u}}} u_{i,P}^{(n-1)}.$$

After all matrix coefficients have been successfully calculated the linear system system can be represented by a system matrix $A$ and a right hand side vector $b$. Figure 5.4 shows the non-zero structure of a linear system for a grid consisting in a $2 \times 2 \times 2$ cell and a $3 \times 3 \times 3$ cell block.

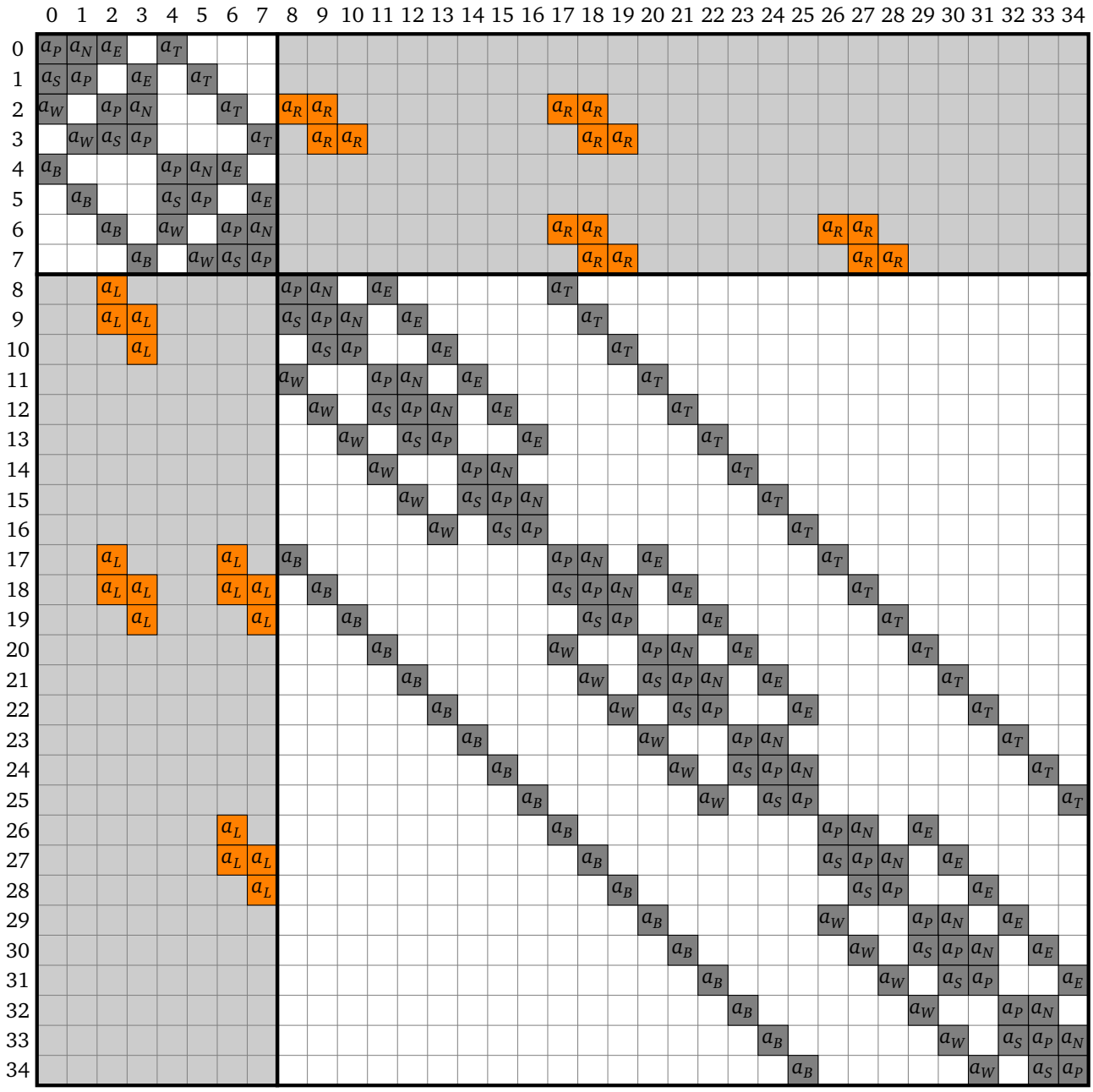

**Figure 5:** Non-zero structure of the linear systems used in the SIMPLE algorithm for a block structured grid consisting of one $2 \times 2 \times 2$ cell and one $3 \times 3 \times 3$ cell block

## 5 Implicit Finite Volume Method for Incompressible Flows – Fully Coupled Approach

Since the antecedent section 4 already discussed the discretization details on the involved equations, this section aims at a comparison at the algorithmic level of the SIMPLE algorithm, presented in section 4.3, as a method to resolve the pressure-velocity coupling, and an implementation of a fully coupled solution algorithm. It should be noted that the discretization of the equations to be solved is not changed in any way to maintain comparability, so the presented differences are exclusively due to difference in the solution algorithm. Successful implementations of a fully coupled solution algorithm for incompressible Navier-Stokes equations have been presented in [8, 15, 21, 63]. In addition the presented work will extend the solution approach presented in [21] to three-dimensional domains. Furthermore this section will present various approaches to incorporate different degrees of velocity-to-temperature and temperature-to-velocity/pressure coupling. Finally the structure of the resulting linear system to be solved is discussed.

### 5.1 The Fully Coupled Algorithm – Pressure-Velocity Coupling Revised

This subsection motivates the use of a fully coupled algorithm to resolve pressure-velocity coupling and mentions the differences to the approach presented in subsection 4.3. The mentioned subsection presented a common solution approach to solve incompressible Navier-Stokes equations: After the linearization of the equations, momentum balances were solved using the pressure from the previous iteration as a guess. Since in general the velocity field obtained by solving a momentum balance with a guessed pressure does not obey continuity, the velocity field and the pressure field had to be corrected. This in turn would lead to an inferior solution regarding the residual of the momentum balances. To avoid this iterative guess-and-correct solution process another class of approaches to the pressure-velocity coupling problematic, represented by algorithms that are *fully coupled*, will be introduced now. .

The central aspect of fully coupled solution methods for Navier-Stokes equations is, that instead of solving for the velocities and pressure corrections sequentially, the velocity field and the pressure are solved for simultaneously [54], so every velocity field that is calculated obeys conservation of momentum and mass, without the need to be corrected. As a result the under-relaxation of pressure and velocities is no longer required, which accelerates convergence significantly in terms of needed outer iterations. The only reason for still needing an iterative solution process is the non-linearity of the Navier-Stokes equations which is accounted for by the use of the Picard iteration process, which has been presented in subsection 3.4. The iterations fortunately make room for deferred corrections as introduced in subsection 4.5. It should be noted that it is also possible to use the coupled solution algorithm to calculate pressure corrections. Reference [34] uses this approach to solve the Navier-Stokes equations.

On the downside implementations of fully coupled solution methods require significantly more system memory than segregated methods. This is due to the higher amount of information that has to be available at the same time and the resulting linear systems which also require more memory, due to the increased amount of unknowns to solve for. Furthermore the bad condition of the linear algebraic system [54] tends to slow down the convergence of the equation solver algorithm.

As in the case of segregated methods it is not advisable to solve the mass balance equation directly [54]. Instead of deriving an equation for the pressure correction $p'$, as shown in 4.3, an equation for the pressure is derived by using the pressure-weighted interpolation method, which has been introduced in 4.2 and is used to approximate the velocities in the discretized mass balance (16). Concretely equation (21) is adapted for the use in a fully coupled algorithm

$$u_{i,f}^{(n)} = \left[\left(1-\gamma_f\right)u_{i,P}^{(n)} + \gamma_f u_{i,Q}^{(n)}\right] - \left(\left(1-\gamma_f\right)\frac{V_P}{a_P^{u_i}} + \gamma_f \frac{V_Q}{a_Q^{u_i}}\right)\left[\underline{\left(\frac{\partial p}{\partial x_i}\right)_f^{(n)}} - \frac{1}{2}\left(\left(\frac{\partial p}{\partial x_i}\right)_P^{(n-1)} + \left(\frac{\partial p}{\partial x_i}\right)_Q^{(n-1)}\right)\right].$$

The changes comprise the removal of last term of (21) accounting for the under-relaxation, since no under-relaxation is needed in fully coupled algorithms which is equivalent to an under-relaxation factor $\alpha_{\mathbf{u}} = 1$. Furthermore the underlined partial derivatives are now going to be treated implicitly. This leads to the semi-implicit pressure equation

$$\begin{aligned}\sum_{F\in NB(P)} \rho\left[\left(1-\gamma_f\right)u_{i,P}^{(n)} + \gamma_f u_{i,F}^{(n)}\right] S_f - \rho\left(\left(1-\gamma_f\right)\frac{V_P}{a_P^{u_i}} + \gamma_f\frac{V_F}{a_F^{u_i}}\right)\left[\left(\frac{\partial p}{\partial x_i}\right)_f^{(n)}\right] S_f \\ = -\sum_{F\in NB(P)} \rho\left(\left(1-\gamma_f\right)\frac{V_P}{a_P^{u_i}} + \gamma_f\frac{V_F}{a_F^{u_i}}\right)\left[\frac{1}{2}\left(\left(\frac{\partial p}{\partial x_i}\right)_P^{(n-1)} + \left(\frac{\partial p}{\partial x_i}\right)_F^{(n-1)}\right)\right] S_f. \end{aligned} \tag{47}$$

Even though this equation is solved for the pressure $p$ while equation (31) is solved for the pressure correction $p'$ the matrix coefficients from the pressure correction discretization of equation (37) and the calculation of the right hand

side is given in equation (32). For the implicit coupling to the velocities $u_i$ additional matrix coefficients have to be considered. These can be calculated as

$$a_F^{p,u_i} = \rho\,\gamma_f S_f \quad \text{and} \quad a_P^{p,u_i} = \sum_{F\in NB(P)} \rho\,(1-\gamma_f) S_f .$$

On the other side the discretized momentum balance gives matrix coefficients to account for the implicit velocity-pressure coupling. They are calculated as

$$a_F^{u_i,p} = \gamma_f S_f \quad \text{and} \quad a_P^{u_i,p} = \sum_{F\in NB(P)} (1-\gamma_f) S_f .$$

If the Boussinesq approximation is used implicitly, an additional coefficient $a_P^{u_i,T}$ to account for the velocity temperature coupling has to be considered. This will be handled in subsection 5.2

**Algorithm 2** Fully Coupled Solution Algorithm

```
INITIALIZE variables
while (convergence criterion not accomplished) do
    if (temperature coupling) then
        SOLVE the linear system for velocities, pressure and temperature
    else
        SOLVE the linear system for velocities and pressure
    end if
    CALCULATE mass fluxes using REFERENCE
    if (decoupled scalar equation) then
        SOLVE scalar equation as described in (4.6)
    end if
end while
```

## 5.2 Coupling to the Temperature Equation

Since the present work also aims at analyzing the efficiency of different methods to couple the temperature equation to the velocities and vice-versa this subsection discusses different approaches to handle the coupling of the temperature equation to the Navier-Stokes equations if the Boussinesq-Approximation, as introduced in subsection 2.5.2, is used. The effectiveness of realizing a strong coupling of the temperature equation to the Navier-Stokes equations depends on the flow problem. The physical coupling tends to increase in flow scenarios of natural convection, where the temperature difference and hence the buoyancy term in the Navier-Stokes equations dominates the fluid movement. Generally speaking: The higher the physical coupling of temperature and flow, the greater are the benefits of using an implementation that uses a strong coupling to the temperature equation. The following subsections present different approaches that can be used for different intensities of coupling.

### 5.2.1 Decoupled Approach – Explicit Velocity-to-Temperature Coupling

The decoupled approach is similar to the treatment of the velocity-temperature coupling described in subsection 4.5. If this approach is chosen no special measures have to be taken. The momentum balances receive a contribution $b_P^{u_i,T}$ with

$$b_P^{u_i,T} := -\rho\beta\left(T_P^{(n-1)} - T_0\right) V_P,$$

that handles the coupling explicitly by using the temperature result of the previous outer iteration. In some cases it might be necessary to under-relax the temperature each iteration

### 5.2.2 Implicit Velocity-to-Temperature Coupling

In the same way it is possible to realize the coupling described in the previous subsection by implicitly considering the temperature in the momentum balances, which leads to an additional matrix coefficient $a_P^{u_i,T}$ and an additional contribution to the right hand side accounting for the coupling to the temperature equation

$$a_P^{u_i,T} = \rho\beta V_P \quad \text{and} \quad b_P^{u_i,T} = \rho\beta T_0 V_P .$$

### 5.2.3 Temperature-to-Velocity/Pressure Coupling – Newton-Raphson Linearization

The previous section discussed the velocity temperature coupling and so far only the momentum balances were affected by the realization of the coupling methods. Independently it is possible to couple the temperature equation not only to the momentum balances but also to the pressure equation. In section 4.6 the non-linear partial differential equation was linearized using a Picard iteration method. Specifically the convective term $\rho u_j T$ was linearized by taking the mass flux from the antecedent solve of the momentum and pressure correction equations. If the decoupled approach is used, this treatment remains valid. But, if an implicit temperature coupling is sought, a coupling of the temperature equation to the momentum balances which considers the mass fluxes is desirable. Methods that exhibit the described property can be found in [23, 47, 58, 63]. A common denomination of the therein used linearization method is called *Newton-Raphson linearization* which can be interpreted as a bilinear approximation of the convective term by the linear terms of the respective Taylor polynomial.

The Newton-Raphson linearization technique is applied to the convective term of the temperature equation as follows. A first order Taylor approximation of the the mass specific convective flux through the boundary face $S_f$ around the $n-1$nt iteration value yields for the approximation at the $n$th iteration

$$\begin{aligned}
\left(u_{j,f}\,T_f\right)^{(n)} &\approx \left(u_{j,f}\,T_f\right)^{(n-1)} + \frac{\partial}{\partial u_{j,f}}\left(u_{j,f}\,T_f\right)^{(n-1)}\left(u_{j,f}^{(n)} - u_{j,f}^{(n-1)}\right) + \frac{\partial}{\partial T_f}\left(u_{j,f}\,T_f\right)^{(n-1)}\left(T_f^{(n)} - T_f^{(n-1)}\right) \\
&= (u_{j,f}\,T_f)^{(n-1)} + {T_f}^{(n-1)}({u_{j,f}}^{(n)} - {u_{j,f}}^{(n-1)}) + {u_{j,f}}^{(n-1)}({T_f}^{(n)} - {T_f}^{(n-1)}) \\
&= \underline{{u_{j,f}}^{(n-1)}{T_f}^{(n)}} + {u_{j,f}}^{(n)}{T_f}^{(n-1)} - {u_{j,f}}^{(n-1)}{T_f}^{(n-1)}.
\end{aligned}$$

A comparison with section 4.5.1 shows, that the underlined term coincides with the term from the usual linearization. The first of the two new terms will be treated implicitly and explicitly after using the pressure-weighted interpolation method from section **??** to interpolate the value of $u_{j,f}$. The second term will be treated explicitly. The use of the pressure-weighted interpolation method does not only create a temperature-to-velocity but also a temperature-to-pressure coupling. Applying the pressure-weighted interpolation, the equation for the mass specific convective flux reads

$$\begin{aligned}
\left(u_{j,f}\,T_f\right)^{(n)} \approx\ & \underline{{u_{j,f}}^{(n-1)}{T_f}^{(n)}} + {T_f}^{(n-1)}\left[\left(1-\gamma_f\right)u_{j,P}^{(n)} + \gamma_f u_{j,Q}^{(n)}\right] \\
& - T_f^{(n-1)}\left(\left(1-\gamma_f\right)\frac{\alpha_{\mathbf{u}}V_P}{a_P^{u_j}} + \gamma_f\frac{\alpha_{\mathbf{u}}V_Q}{a_Q^{u_j}}\right)\left[\left(\frac{\partial p}{\partial x_j}\right)_f^{(n)} - \frac{1}{2}\left(\frac{\partial p}{\partial x_j}\right)_P^{(n-1)} - \frac{1}{2}\left(\frac{\partial p}{\partial x_j}\right)_Q^{(n-1)}\right] \\
& - {u_{j,f}}^{(n-1)}{T_f}^{(n-1)}.
\end{aligned}$$

If this semi-implicit, semi-discrete equation is discretized completely new matrix and right hand side contributions can be calculated.

$$a_F^{T,u_i} = \rho\, T_f^{(n-1)}\gamma_f\, n_{f,i} S_f$$

$$a_P^{T,u_i} = \sum_{F\in NB(P)} \rho\, T_f^{(n-1)}\left(1-\gamma_f\right) n_{f,i} S_f$$

$$a_F^{T,p} = -\rho\, T_f^{(n-1)}\left(\left(1-\gamma_f\right)\frac{\alpha_{\mathbf{u}}V_P}{a_P^{u_i}} + \gamma_f\frac{\alpha_{\mathbf{u}}V_F}{a_F^{u_i}}\right)\frac{S_f}{(\mathbf{x}_P-\mathbf{x}_F)\cdot\mathbf{n}_f} = \rho\, T_f^{(n-1)} a_F^p$$

$$\text{and}\quad a_P^{T,p} = -\sum_{F\in NB(P)} T_f^{(n-1)} a_F^{T,p}.$$

## 5.3 Boundary Conditions on Domain and Block Boundaries

### 5.3.1 Dirichlet Boundary Condition for Velocity

Mention that the Newton-Raphson linearization is not considered on Dirichlet boundaries.

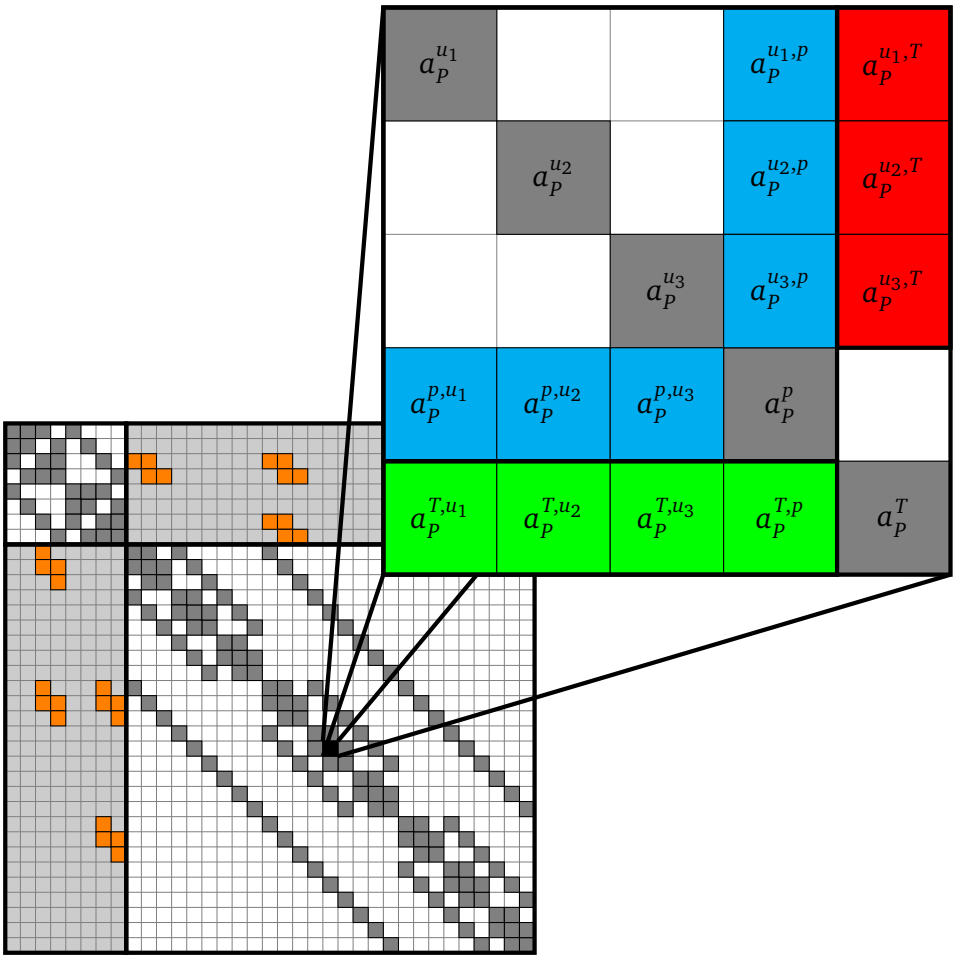


**Figure 6:** Non-zero structure of block submatrices of the linear systems used in the coupled solution algorithm for a block structured grid consisting of one $2 \times 2 \times 2$ cell and one $3 \times 3 \times 3$ cell block. The blue coefficients represent the pressure-velocity coupling, the red coefficients correspond to the velocity-to-temperature coupling and the green coefficients result from the Newton-Raphson linearization technique.

### 5.3.2 Wall Boundary Condition

### 5.3.3 Block Boundary Condition

## 5.4 Assembly of Linear Systems – Final Form of Equations

## References


[1]

[2] Anderson, D. A., Tannehill, J. C. and Pletcher, R. H. *Computational Fluid Mechanics and Heat Transfer*. Hemisphere Publishing Corporation, Wahington, 1984.

[3] Aris, R. *Vectors, Tensors and the Basic Equations of Fluid Mechanics*. Prentice-Hall, Inc., 1962.

[4] Balay, S., Abhyankar, S., Adams, M. F., Brown, J., Brune, P., Buschelman, K., Eijkhout, V., Gropp, W. D., Kaushik, D., Knepley, M. G., McInnes, L. C., Rupp, K., Smith, B. F. and Zhang, H. PETSc users manual. Tech. Rep. ANL-95/11 - Revision 3.5, Argonne National Laboratory, 2014.

[5] Balay, S., Abhyankar, S., Adams, M. F., Brown, J., Brune, P., Buschelman, K., Eijkhout, V., Gropp, W. D., Kaushik, D., Knepley, M. G., McInnes, L. C., Rupp, K., Smith, B. F. and Zhang, H. PETSc Web page. `http://www.mcs.anl.gov/petsc`, 2014.

[6] Balay, S., Gropp, W. D., McInnes, L. C. and Smith, B. F. Efficient management of parallelism in object oriented numerical software libraries. In *Modern Software Tools in Scientific Computing* (1997), E. Arge, A. M. Bruaset and H. P. Langtangen, Eds., Birkhäuser Press, pp. 163–202.

[7] Bond, R. B., Ober, C. C., Knupp, P. M. and Bova, S. W. Manufactured solution for computational fluid dynamics boundary condition verification. *AIAA Journal 45*, 9 (Sep 2007), 2224–2236.

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

[9] Chisholm, T. and Zingg, D. W. A fully coupled newton-krylov solver for turbulent aerodynamics flows, 2002.

[10] Choi, S. K. Note on the use of momentum interpolation method for unsteady flows. *Numerical Heat Transfer, Part A: Applications 36*, 5 (1999), 545–550.

[11] Choi, S.-K., Kim, S.-O., Lee, C.-H. and Choi, H.-K. Use of the momentum interpolation method for flows with a large body force. *Numerical Heat Transfer, Part B: Fundamentals 43*, 3 (2003), 267–287.

[12] Choudhary, A., Roy, C., Luke, E. and Veluri, S. Fluid Dynamics and Co-located Conferences. American Institute of Aeronautics and Astronautics, Jun 2011, ch. Issues in Verifying Boundary Conditions for 3D Unstructured CFD Codes.

[13] Christon, M. A., Gresho, P. M. and Sutton, S. B. Computational predictability of time-dependent natural convection flows in enclosures (including a benchmark solution). *International Journal for Numerical Methods in Fluids 40*, 8 (2002), 953–980.

[14] Dahmen, W. and Reusken, A. *Numerik für Ingenieure und Naturwissenschaftler*, 2. ed. Springer Verlag, Berlin, 2008.

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

[16] De Vahl Davis, G. Natural convection of air in a square cavity: A bench mark numerical solution. *International Journal for Numerical Methods in Fluids 3*, 3 (1983), 249–264.

[17] Deng, G., Piquet, J., Vasseur, X. and Visonneau, M. A new fully coupled method for computing turbulent flows. *Computers & Fluids 30*, 4 (2001), 445 – 472.

[18] Deng, G. B., Piquet, J., Queutey, P. and Visonneau, M. A new fully coupled solution of the navier-stokes equations. *International Journal for Numerical Methods in Fluids 19*, 7 (1994), 605–639.

[19] Elman, H., Howle, V. E., Shadid, J., Shuttleworth, R. and Tuminaro, R. A taxonomy and comparison of parallel block multi-level preconditioners for the incompressible navier-stokes equations. *J. Comput. Phys. 227*, 3 (Jan. 2008), 1790–1808.

[20] Elman, H., Howle, V. E., Shadid, J. and Tuminaro, R. A parallel block multi-level preconditioner for the 3d incompressible navier-stokes equations. *Journal of Computational Physics 187* (2003), 504–523.

[21] Falk, U. and Schäfer, M. A fully coupled finite volume solver for the solution of incompressible flows on locally refined non-matching block-structured grids. In *Adaptive Modeling and Simulation 2013* (Barcelona, Spain, June 2013), J. P. M. de Almeida, P. Diez, C. Tiago and N. Parez, Eds., pp. 235–246.

[22] Ferziger, J. H. and Perić, M. *Numerische Strömungsmechanik*. Springer Verlag, Berlin, 2002.

[23] Galpin, P. F. and Raithby, G. D. Numerical solution of problems in incompressible fluid flow: Treatment of the temperature-velocity coupling. *Numerical Heat Transfer 10*, 2 (1986), 105–129.

[24] Gray, D. D. and Giorgini, A. The validity of the boussinesq approximation for liquids and gases. *International Journal of Heat and Mass Transfer 19*, 5 (1976), 545 – 551.

[25] Gresho, P. M. and Sani, R. L. On pressure boundary conditions for the incompressible navier-stokes equations. *International Journal for Numerical Methods in Fluids 7*, 10 (1987), 1111–1145.

[26] Gropp, W., Lusk, E. and Skjellum, A. *Using MPI: portable parallel programming with the message-passing interface*, 2. ed. The MIT Press, Cambridge, Massachussets, 1999.

[27] Gropp, W. D., Kaushik, D. K., Keyes, D. E. and Smith, B. F. High performance parallel implicit cfd. *Parallel Computing 27* (2000), 337–362.

[28] Hadji, S. and Dhatt, G. Asymptotic–newton method for solving incompressible flows. *International Journal for Numerical Methods in Fluids 25*, 8 (1997), 861–878.

[29] Hager, G. and Wellein, G. *Introduction to High Performance Computing for Scientists and Engineers*. CRC, Boca Raton, 2011.

[30] Henderson, A. Paraview guide, a parallel visualization application, 2007.

[31] Hortmann, M., Perić, M. and Scheuerer, G. Finite volume multigrid prediction of laminar natural convection: Bench-mark solutions. *International Journal for Numerical Methods in Fluids 11*, 2 (1990), 189–207.

[32] Hui, W. Exact solutions of the unsteady two-dimensional navier-stokes equations. *Journal of Applied Mathematics and Physics ZAMP 38*, 5 (1987), 689–702.

[33] Jasak, H. *Error Analysis and Estimation for the Finite Volume Method with Applications to Fluid Flows*. PhD thesis, Imperial College of Science, Technology and Medicine, Jun 1996.

[34] Klaij, C. M. and Vuik, C. Simple-type preconditioners for cell-centered, colocated finite volume discretization of incompressible reynolds-averaged navier–stokes equations. *International Journal for Numerical Methods in Fluids 71*, 7 (2013), 830–849.

[35] Kundu, P. K., Cohen, I. M. and Downling, D. R. *Fluid Mechanics*, 5 ed. Elsevier, 2012.

[36] Lange, C. F., Schäfer, M. and Durst, F. Local block refinement with a multigrid flow solver. *International Journal for Numerical Methods in Fluids 38*, 1 (2002), 21–41.

[37] Li, W., Yu, B., Wang, X.-r. and Sun, S.-y. Calculation of cell face velocity of non-staggered grid system. *Applied Mathematics and Mechanics 33*, 8 (2012), 991–1000.

[38] Lilek, Ž., Muzaferija, S., Perić, M. and Seidl, V. An implicit finite-volume method using nonmatching blocks of structured grid. *Numerical Heat Transfer, Part B: Fundamentals 32*, 4 (1997), 385–401.

[39] Majumdar, S. Role of underrelaxation in momentum interpolation for calculation of flow with nonstaggered grids. *Numerical Heat Transfer 13*, 1 (1988), 125–132.

[40] Maple. *version 18*. Waterloo Maple Inc. (Maplesoft), Waterloo, Ontario, 2014.

[41] MATLAB. *version 7.10.0 (R2010a)*. The MathWorks Inc., Natick, Massachusetts, 2010.

[42] McCalpin, J. D. Stream: Sustainable memory bandwidth in high performance computers. Tech. rep., University of Virginia, Charlottesville, Virginia, 1991-2007. A continually updated technical report. `http://www.cs.virginia.edu/stream/`.

[43] McCalpin, J. D. Memory bandwidth and machine balance in current high performance computers. *IEEE Computer Society Technical Committee on Computer Architecture (TCCA) Newsletter* (Dec. 1995), 19–25.

[44] Meister, A. *Numerik linearer Gleichungssysteme*, 4. ed. Vieweg+Teubner, Wiesbaden, 2011.

[45] Miller, T. F. and Schmidt, F. W. Use of a pressure-weighted interpolation method for the solution of the incompressible navier-stokes equations on a nonstaggered grid system. *Numerical Heat Transfer 14*, 2 (1988), 213–233.

[46] Oberkampf, W. L. and Trucano, T. G. Verification and validation in computational fluid dynamics. *Progress in Aerospace Sciences 38*, 3 (2002), 209 – 272.

[47] Oliveira, P. J. and Issa, R. I. An improved piso algorithm for the computation of buoyancy-driven flows. *Numerical Heat Transfer, Part B: Fundamentals 40*, 6 (2001), 473–493.

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

[49] Perić, M. Analysis of pressure-velocity coupling on nonorthogonal grids. *Numerical Heat Transfer 17* (Jan. 1990), 63–82.

[50] Pope, S. B. *Turbulent Flows*. Cambridge University Press, New York, 2000.

[51] Ramamurti, R. and Löhner, R. A parallel implicit incompressible flow solver using unstructured meshes. *Computers & Fluids 25*, 2 (1996), 119 – 132.

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

[53] Salari, K. and Knupp, P. Code verification by the method of manufactured solutions. Tech. Rep. SAND2000-1444, Sandia National Labs., Albuquerque, NM (US); Sandia National Labs., Livermore, CA (US), Jun 2000.

[54] Schäfer, M. *Numerik im Maschinenbau*. Springer Verlag, Berlin, 1999.

[55] Schäfer, M. and Turek, S. Recent benchmark computations of laminar flow around a cylinder, 1996.

[56] Sheng, Y., Sheng, G., Shoukri, M. and Wood, P. New version of simplet and its application to turbulent buoyancy-driven flows. *Numerical Heat Transfer, Part A: Applications 34*, 8 (1998), 821–846.

[57] Sheng, Y., Shoukri, M., Sheng, G. and Wood, P. A modification to the simple method for buoyancy-driven flows. *Numerical Heat Transfer, Part B: Fundamentals 33*, 1 (1998), 65–78.

[58] Sheu, T. W. H. and Lin, R. K. Newton linearization of the incompressible navier–stokes equations. *International Journal for Numerical Methods in Fluids 44*, 3 (2004), 297–312.

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

[60] Sotiropoulos, F. and Abdallah, S. Coupled fully implicit solution procedure for the steady incompressible navier-stokes equations. *Journal of Computational Physics 87*, 2 (1990), 328 – 348.

[61] Spurk, J. H. and Aksel, N. *Strömungslehre: Einführung in die Theorie der Strömungen*, 8. ed. Sringer Verlag, Berlin, 2010.

[62] Taylor, G. I. and Green, A. E. Mechanism of the production of small eddies from large ones. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences 158*, 895 (1937), pp. 499–521.

[63] Vakilipour, S. and Ormiston, S. J. A coupled pressure-based co-located finite-volume solution method for natural-convection flows. *Numerical Heat Transfer, Part B: Fundamentals 61*, 2 (2012), 91–115.

[64] Van Doormaal, J. P. and Raithby, G. D. Enhancements of the simple method for predicting incompressible fluid flows. *Numerical Heat Transfer 7*, 2 (1984), 147–163.

[65] Vanka, S. Block-implicit multigrid solution of navier-stokes equations in primitive variables. *Journal of Computational Physics 65*, 1 (1986), 138 – 158.

[66] ZHANG, S., ZHAO, X. AND BAYYUK, S. Generalized formulations for the rhie–chow interpolation. *Journal of Computational Physics 258*, 0 (2014), 880 – 914.