Department of Applied Mechanics

Multigrid Acceleration Methods for a Solution of a Turbulent Incompressible and a Laminar Magneto-hydrodynamic Flow

Ari Kruskopf

Aalto University | SCIENCE + TECHNOLOGY | REPORT

Aalto University publication series
SCIENCE + **TECHNOLOGY** 24/2011

Multigrid Acceleration Methods for a Solution of a Turbulent Incompressible and a Laminar Magnetohydrodynamic Flow

Ari Kruskopf

Aalto University
School of Engineering
Department of Applied Mechanics
Fluid Mechanics

Aalto University publication series
SCIENCE + **TECHNOLOGY** 24/2011

ISBN 978-952-60-4361-6 (pdf)
ISBN 978-952-60-4360-9 (printed)
ISSN-L 1799-4896
ISSN 1799-490X (pdf)
ISSN 1799-4896 (printed)

Aalto Print
Helsinki 2011

Finland

Contents

Nomenclature

A	magnetic vector potential in Chapter 3	$\mathrm{Wb/m}$
A	coefficient matrix of linear equations in Chapters 4 and 5	
B	magnetic flux density	T
C^{dif}	dimensionless number	
cfl	courant number	
D	electric flux density	$\mathrm{C/m^2}$
E	electric field strength	$\mathrm{V/m}$
f, F	function	
F	flux of variable ϕ	
F_L	Lorentz force	N
h	nearest cell center distance from the wall	m
H	magnetic field strength	$\mathrm{A/m}$
$\vec{i}, \vec{j}, \vec{k}$	Cartesian vector components	
J	electric current density	$\mathrm{A/m^2}$
k	turbulence kinetic energy	$\mathrm{m^2/s^2}$
m	mass	kg
$\dot{m}$	mass flow rate	$\mathrm{kg/s}$
m_{eq}	highest order of differentiation in equation	
m_P	accuracy of prolongation operator	
m_R	accuracy of restriction operator	
n	surface normal vector	
p	pressure	Pa
$\mathbf{P}$	pressure matrix	
P	turbulence production	
R	residual	
R^*	restricted residual in GMG	
$\overline{R}$	forcing function for residual in GMG	
$\hat{R}$	modified residual in GMG	
Re_m	magnetic Reynolds number	
s	source term	
S	cell-face area	$\mathrm{m^2}$
S_{ij}	strain rate tensor	
t	time	s
u, v, w	velocity components in x-, y- and z-directions	$\mathrm{m/s}$
u_τ	wall-friction velocity	$\mathrm{m/s}$
u_i	fluctuation of velocity component i	$\mathrm{m/s}$
U_i	average of velocity component i	$\mathrm{m/s}$
V	velocity	$\mathrm{m/s}$

$\overline{V}$	contravariant velocity at cell face	$\mathrm{m/s}$
$\mathbf{V}$	volume	m^3
x, y, z	Cartesian coordinates	
y^+	dimensionless cell center height at wall	

Greek letters

α_u, α_p	under-relaxation coefficients for velocity and pressure	
δ_{ij}	Kronecker's delta	
ξ, η	distance between successive cell centers in i- and j-directions	
λ	diffusion coefficient	
Λ	diffusion coefficient and friction factor	
ϵ	turbulence dissipation rate	$\mathrm{m}^2/\mathrm{s}^3$
ϵ_m	permittivity	$\mathrm{F/m}$
ν	kinematic viscosity	m^2/s
ν_m	magnetic viscosity	m^2/s
μ	dynamic viscosity	$\mathrm{Pa\,s}$
μ_T	eddy viscosity	$\mathrm{Pa\,s}$
μ_m	magnetic permeability	$\mathrm{H/m}$
ρ	density	$\mathrm{kg/m}^3$
ρ_m	electric charge density	$\mathrm{C/m}^3$
ϕ	reduced electric scalar potential	V
ψ	source term	
σ	electrical conductivity	$\mathrm{S/m}$
τ_{ij}	$= -\rho\overline{u_i' u_j'}$ Reynolds stress	
ω	modified residual in AMG	

Superscripts

m	grid level
n,l	iteration counts
*	updated velocity value after momentum equation
$'$	fluctuation of variable; also used in pressure-correction algorithm as a correction
L, R	left, right
C, D	convection, diffusion

Subscripts

i,j	grid coordinate directions; also used as tensor notations
k	component of the equation system
k	also relates constants to turbulence kinetic equation
e, w, n, s	compass points in grid; east, west, north and south cell
p	middle cell

Operators

∇	grad operator
$\nabla\cdot$	divergence operator
$\nabla\times$	curl operator
$\max()$	maximum operator
$\min()$	minimum operator

k-ϵ Turbulence Model Constants

$$
\begin{aligned}
C_\mu &= 0.09 \\
C_{\epsilon 1} &= 1.44 \\
C_{\epsilon 2} &= 1.92 \\
\sigma_k &= 1.0 \\
\sigma_\epsilon &= 1.3 \\
C_T &= \sqrt{2} \\
C_k &= 0.5 \\
C_\epsilon &= -2C_k
\end{aligned}
$$

Abbreviations

MG	Multigrid
AMG	Algebraic Multigrid
FAS	Full Approximation Storage
GCA	Galerkin Coarse Grid Approximation
GMG	Geometric Multigrid
LGS	Line-Gauss-Seidel
SIMPLE	Semi-Implicit Method for Pressure-Linked Equations

1 Introduction

Equations and a solution procedure for a two-dimensional turbulent incompressible flow and a magnetohydrodynamic laminar flow in a curvilinear grid are presented in this work. Turbulence is modeled with a k-ϵ model with cross-diffusion terms. Continuity and momentum equations are solved with a pressure-correction algorithm. Electromagnetic phenomena are approximated with a magnetic vector potential equation. The solution procedure consists of an implicit time-stepping algorithm applied in a control-volume method with a multigrid (MG) acceleration. MG schemes are used for both linear and non-linear equation systems. From an engineering point of view MG methods in the literature are analyzed with complex mathematics. Practical MG algorithms for incompressible flow problems are still rare. Therefore, the MG algorithms in this work are presented in detail so that a reader who has basic skills in fluid dynamics, numerical methods and programming can gain an insight into how to transform a single grid solver into an MG solver.

2 Turbulence Model Equations

A simulation of the turbulent Navier-Stokes equations without simplifications is very time consuming. A statistical approach can be used to reduce the computational work. In the following chapters the averaging method of the N-S equations and turbulence energy modeling are described. This chapter was written on the basis of Wilcox's book [1] as well as the article by Rahman and Siikonen [2].

2.1 Reynolds Averaged Incompressible Navier-Stokes Equations

Averaging can be performed in several ways. The most common methods are time averaging, spatial averaging and ensemble averaging. Time averaging is chosen for this work. This is the most suitable choice for stationary turbulence. Time averaging is defined for flow variable $f(x,t)$ as follows:

$$F(x) = \lim_{T\to\infty} \frac{1}{T} \int_t^{t+T} f(x,t)dt \tag{1}$$

In the case of stationary turbulence the flow variables can be expressed as the sum of a mean, $F_i(x)$, and a fluctuating part $f_i^{'}(x,t)$. Thus instantaneous velocity can now be described as

$$u_i(x,t) = U_i(x) + u_i^{'}(x,t) \tag{2}$$

Below, the continuity and momentum equations are presented in a conservation form respectively in the case of incompressibility and constant viscosity.

$$\frac{\partial u_i}{\partial x_i} = 0 \tag{3}$$

$$\frac{\partial \rho u_i}{\partial t} + \frac{\partial \rho u_i u_j}{\partial x_j} = -\frac{\partial p_i}{\partial x_i} + \frac{\partial}{\partial x_j}\left(\mu \frac{\partial u_i}{\partial x_j}\right) \tag{4}$$

Here, the velocity components and pressure include mean and fluctuating parts. The resulting Reynolds-averaged equations have the following form:

$$\frac{\partial U_i}{\partial x_i} = 0 \tag{5}$$

$$\frac{\partial \rho U_i}{\partial t} + \frac{\partial \rho U_i U_j}{\partial x_j} + \frac{\partial \rho \overline{u_i' u_j'}}{\partial x_j} = -\frac{\partial P_i}{\partial x_i} + \frac{\partial}{\partial x_j}\left(\mu \frac{\partial U_i}{\partial x_j}\right) \tag{6}$$

Only the momentum equation differs from the original incompressible laminar equations with a single term. A new term is the Reynolds stress tensor $\tau_{ij} = -\rho\overline{u_i' u_j'}$. The line over the term denotes time averaging the product of the velocity component fluctuation parts. This term is symmetric so it has six unknown components in a three-dimensional case. Hence there is a need for additional equations. Here we use a Boussinesq eddy-viscosity approximation. Eddy viscosity μ_T is a property of the flow and describes how turbulent eddies transport momentum, mass and energy alongside the mean flow convection. The Boussinesq approximation suggests that turbulent eddies are transported by diffusion. For this approximation to be valid it is assumed that turbulence is homogeneous and in a state of equilibrium. In the momentum equation Reynolds stresses are correlated as:

$$\tau_{ij} = \mu_T \frac{\partial U_i}{\partial x_j} \tag{7}$$

Now the momentum equation has a new form as Reynolds stress is considered as a diffusive term:

$$\frac{\partial \rho U_i}{\partial t} + \frac{\partial \rho U_i U_j}{\partial x_j} = -\frac{\partial p_i}{\partial x_i} + \frac{\partial}{\partial x_j}\left[(\mu + \mu_T)\frac{\partial U_i}{\partial x_j}\right] \tag{8}$$

To retain the Boussinesq approximation, turbulence energy has to be modeled using at least one or two equations. Turbulence energy equation models have been developed to incorporate non-local and flow history effects into the eddy-viscosity equation. One-equation models relate the turbulence length scale to some typical flow dimension and are described as incomplete. Two-equation models provide an equation for the turbulence length scale. [1]

2.2 Applied Two-Equation Model for Turbulence Energy

Turbulence kinetic energy k is defined by

$$k = \frac{1}{2}\overline{u_i' u_i'} = \frac{1}{2}\left(\overline{u'^2} + \overline{v'^2} + \overline{w'^2}\right) \tag{9}$$

where squared velocity fluctuations (u', v', w') are time averaged and summed. Turbulence energy is modeled with the transport equations of turbulence kinetic energy k as in Ref.[2]:

$$\frac{\partial \rho k}{\partial t} + \frac{\partial \rho U_j k}{\partial x_j} = \frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_T}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right] + P - \rho\epsilon + E_k \tag{10}$$

and its dissipation rate $\tilde{\epsilon}$:

$$\frac{\partial \rho \tilde{\epsilon}}{\partial t} + \frac{\partial \rho U_j \tilde{\epsilon}}{\partial x_j} = \frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_T}{\sigma_\epsilon}\right)\frac{\partial \tilde{\epsilon}}{\partial x_j}\right] + \frac{C_{\epsilon 1}P - C_{\epsilon 2}\rho\tilde{\epsilon} - \rho D e^{-(R_y/80)^2}}{T_t} + E_\epsilon \tag{11}$$

which determines the rate at which turbulence kinetic energy is converted into thermal energy. In the last two equations $\epsilon = \tilde{\epsilon} + D$ and turbulence timescale $T_t = \max\left(k/\tilde{\epsilon}, C_T\sqrt{\nu/\epsilon}\right)$. The eddy viscosity and some of the other necessary variables are modeled by

$$\mu_T = C_\mu f_\mu \rho k T_t \tag{12}$$

$$D = \frac{2\nu k}{y_n^2} \tag{13}$$

$$R_y = \frac{\sqrt{k} y_n}{\nu} \tag{14}$$

where ν is kinematic viscosity and y_n the normal distance from the wall. The constants associated with the equations are $C_\mu = 0.09$, $C_{\epsilon 1} = 1.44$, $C_{\epsilon 2} = 1.92$, $C_T = \sqrt{2}$, $\sigma_k = 1.0$ and $\sigma_\epsilon = 1.3$. The damping function f_μ located in the eddy viscosity equation is a function of R_λ. These are defined by

$$f_\mu = 1 + e^{\left(-0.01R_\lambda - 0.0068R_\lambda^3\right)} \tag{15}$$

$$R_\lambda = \frac{y_n}{\sqrt{\frac{\nu k}{\tilde{\epsilon}}}} = \frac{y_n}{\sqrt{\nu T_t}} \tag{16}$$

where $\sqrt{\nu T_t}$ is a Taylor microscale. To incorporate nonequilibrium flow effects, which are present in separated and reattaching flows, cross-diffusion terms E_k and E_ϵ are present in the turbulence kinetic energy and dissipation rate equations. The cross-diffusion terms enhance energy dissipation in non-equilibrium flows. They are defined by

$$E_k = C_k C_\mu \frac{\partial}{\partial x_j}\left(\frac{k^3}{\tilde{\epsilon}^2}\frac{\partial \tilde{\epsilon}}{\partial x_j}\right) \tag{17}$$

$$E_\epsilon = C_\epsilon C_\mu \frac{\partial}{\partial x_j}\left(k\frac{\partial k}{\partial x_j}\right) \tag{18}$$

which have a simplified form as

$$E_k = C_k \mu_T \min\left[\frac{\partial\left(k/\epsilon\right)}{\partial x_j}\frac{\partial \tilde{\epsilon}}{\partial x_j}, 0\right] \tag{19}$$

$$E_\epsilon = C_\epsilon \frac{\mu_T}{T_t^2}\left[\frac{\partial\left(k/\epsilon\right)}{\partial x_j}\frac{\partial k}{\partial x_j}\right] \tag{20}$$

where the model constants are $C_k = 0.5$ and $C_\epsilon = -2C_k$. The turbulence production term P in the kinetic energy equation equals

$$P = \tau_{ij}\frac{\partial U_i}{\partial x_j} \tag{21}$$

where the following Boussinesq approximation is used for Reynolds stress:

$$\tau_{ij} = 2\mu_T S_{ij} - \frac{2}{3}\rho k \delta_{ij} \tag{22}$$

Here S_{ij} is the mean-strain rate tensor and δ_{ij} Kronecker's delta. This means that the second term on the right-hand side of the equation contributes only in the tensor diagonal. The strain-rate tensor can be expressed with velocity derivatives:

$$S_{ij} = \frac{1}{2}\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right) \tag{23}$$

After manipulation, the turbulent production term can be presented in 2D-form as a function of derivatives of the velocity components u and v.

$$\tau_{ij}\frac{\partial U_i}{\partial x_j} = 2\mu_T\left[\left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial y}\right)^2\right] + \mu_T\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right)^2 \tag{24}$$

Derivation of the result is shown in Appendix A1.

3 Magnetohydrodynamic Model

Magnetohydrodynamics means that a electromagnetic field is coupled with the flow field. This coupling can be bi-directional. The coupling in the momentum equation is taken into account with a source term that is a result of a force generated by the electromagnetic field. This term is called a Lorentz force. Electromagnetic phenomena can be solved with an induction equation or magnetic vector potential equation. In two-dimensional problems the latter is a better choice since only one equation in addition to the momentum equations need to be solved to take magnetohydrodynamic phenomena into account. But a vector potential describes indirectly a magnetic flux density, therefore, complicated magnetic boundary conditions cannot be included in the magnetic vector potential equation. These two equations can be derived from Maxwell equations

$$\nabla \times \vec{H} = \vec{J} + \frac{\partial \vec{D}}{\partial t} \tag{25}$$

$$\nabla \times \vec{E} = -\frac{\partial \vec{B}}{\partial t} \tag{26}$$

$$\nabla \cdot \vec{B} = 0 \tag{27}$$

$$\nabla \cdot \vec{D} = \rho_m \tag{28}$$

where E, H, D, B, J and ρ_m are an electric field strength, a magnetic field strength, an electric flux density, a magnetic flux density, an electric current density and an electric charge density, respectively. Three

material parameters are also needed. These are permittivity ϵ_m, permeability μ_m and electric conductivity σ, which are presented in the constitutive relations:

$$\vec{D} = \epsilon_m \vec{E} \tag{29}$$

$$\vec{B} = \mu_m \vec{H} \tag{30}$$

$$\vec{J} = \sigma \vec{E} \tag{31}$$

To avoid confusion with the flow parameters, such as the turbulence dissipation rate ϵ, the molecular viscosity μ and density ρ, subindex m is used with the permittivity, the permeability and the electric charge density. It is possible to determine criteria for omitting the time derivative responsible for the displacement current in Eq. (25). By combining Eqs. (25), (29), (30) and (31) we get

$$\nabla \times \vec{B} = \sigma \mu_m \vec{E} - \mu_m \epsilon_m \frac{\partial \vec{E}}{\partial t} \tag{32}$$

The relative orders of magnitude of the right-hand side terms are estimated to be

$$\sigma \mu_m \vec{E} \sim \sigma \mu_m \vec{E}_0 \ , \quad \mu_m \epsilon_m \frac{\partial \vec{E}}{\partial t} \sim \mu_m \epsilon_m \frac{\vec{E}_0}{t_0}$$

where E_0 and t_0 are the characteristic values for the electric field strength and time, respectively. By studying the ratio of these terms an inequality

$$\frac{\epsilon_m}{t_0 \sigma} \ll 1 \tag{33}$$

can be determined. When this is true, Eq. (25) has a new form:

$$\nabla \times \vec{B} = \mu_m \vec{J} \tag{34}$$

When a conducting fluid moves in a magnetic field, an electric field is induced if there exists velocity and magnetic field components that are perpendicular to each other [3]. This additional field can be taken into account with

$$\vec{E'} = \vec{V} \times \vec{B} \tag{35}$$

which is added to Eq. (31):

$$\vec{J} = \sigma \left(\vec{E} + \vec{V} \times \vec{B} \right) \tag{36}$$

In this work, material properties are assumed to be constant. Local charge density and magnetic dipoles are assumed to be in equilibrium. Also, magnetohydrodynamic shock waves are omitted.

3.1 Equation for Magnetic Vector Potential

A magnetic vector potential is defined by:

$$\vec{B} = \nabla \times \vec{A}, \tag{37}$$

but this does not uniquely define the vector potential. Adding the gradient of any scalar function f to the vector potential does not affect the result, since $\nabla \times \nabla f \equiv 0$. Thus $\nabla \cdot \vec{A}$ can be specified for the sake of convenience. In electromagnetic problems considered in this work, the Coulomb gauge

$$\nabla \cdot \vec{A} = 0 \tag{38}$$

is assumed to hold [4]. Substituting magnetic vector potential into Eq. (34), a connection with the vector potential and the current density can be found:

$$\nabla \times \nabla \times \vec{A} = -\nabla^2 \vec{A} = \mu_m \vec{J} \tag{39}$$

where the simplification can be made using

$$\nabla \times \nabla \times \vec{A} = \nabla \left(\nabla \cdot \vec{A} \right) - \nabla^2 \vec{A} = -\nabla^2 \vec{A} \tag{40}$$

and the Coulomb gauge Eq. (38). Eq. (40) is derived in Appendix A2. Next the vector potential is substituted into Eq. (26)

$$\nabla \times \vec{E} = -\frac{\partial}{\partial t} \left(\nabla \times \vec{A} \right) = -\nabla \times \frac{\partial \vec{A}}{\partial t} \tag{41}$$

from where the inverse of the curl is taken:

$$\vec{E} = -\frac{\partial \vec{A}}{\partial t} + \nabla \phi \tag{42}$$

This equation has a gradient of a reduced electric scalar potential $\nabla \phi$ as a new term. This term vanishes when operated by the curl and, therefore, represents the irrotational part of the electric field. In the current density Eq. (36), E is replaced by Eq. (42):

$$\vec{J} = \sigma \left(-\frac{\partial \vec{A}}{\partial t} + \nabla \phi + \vec{V} \times \vec{B} \right) \tag{43}$$

Now an equation for the magnetic vector potential can be derived by combining Eqs. (37), (39) and (43), and reorganizing the terms:

$$\frac{\partial \vec{A}}{\partial t} = \vec{V} \times \nabla \times \vec{A} + \frac{1}{\mu_m \sigma} \nabla^2 \vec{A} + \nabla \phi \tag{44}$$

In order to restrain magnetic and Lorentz force fields into xy-plane, vectors $\vec{A} = A_z\vec{k}$ and $\vec{J} = J_z\vec{k}$. Thus the magnetic flux density in terms of vector potential is

$$\vec{B} = \nabla \times \vec{A} = \begin{vmatrix} \vec{i} & \vec{j} & \vec{k} \\ \dfrac{\partial}{\partial x} & \dfrac{\partial}{\partial y} & \dfrac{\partial}{\partial z} \\ 0 & 0 & A_z \end{vmatrix} = \frac{\partial A_z}{\partial y}\vec{i} - \frac{\partial A_z}{\partial x}\vec{j} \tag{45}$$

and a convection term

$$\vec{V} \times \nabla \times \vec{A} = \begin{vmatrix} \vec{i} & \vec{j} & \vec{k} \\ u & v & 0 \\ \dfrac{\partial A_z}{\partial y} & -\dfrac{\partial A_z}{\partial x} & 0 \end{vmatrix} = \left(-u\frac{\partial A_z}{\partial x} - v\frac{\partial A_z}{\partial y} \right) \vec{k} \tag{46}$$

which is substituted into Eq. (44):

$$\frac{\partial A_z}{\partial t} + u\frac{\partial A_z}{\partial x} + v\frac{\partial A_z}{\partial y} = \frac{1}{\mu_m \sigma}\nabla^2 A_z + \frac{\partial \phi}{\partial z} \tag{47}$$

where the A_x and A_y components are neglected. Since the current density exists only in the z-direction, the reduced scalar potential has the same restriction and acts as a source term for the z-component of the vector potential. Because the flow is incompressible, a continuity equation is $\nabla \cdot \vec{V} = 0$. Thus velocity components u and v can be put inside the partial derivatives:

$$\frac{\partial \rho A_z}{\partial t} + \frac{\partial u \rho A_z}{\partial x} + \frac{\partial v \rho A_z}{\partial y} = \frac{\rho}{\mu_m \sigma}\nabla^2 A_z + \frac{J_s \rho}{\sigma} \tag{48}$$

where a source term is replaced with:

$$\frac{\partial \phi}{\partial z} = \frac{J_s}{\sigma} \tag{49}$$

Now a two-dimensional equation for the magnetic vector potential is essentially a convection-diffusion equation in the conservation form. This equation is multiplied by ρ so it resembles other equations and keeps the notations simpler. The reduced scalar potential depends on the source current density J_S in the z-direction.

3.2 Equation for a Magnetic Induction and a Magnetic Reynolds Number

An induction equation can be derived by taking the curl from Eq. (44) and changing the vector potential to the magnetic flux with Eq. (37):

$$\frac{\partial \vec{B}}{\partial t} = \nabla \times \left(\vec{V} \times \vec{B} \right) + \frac{1}{\mu_m \sigma} \nabla^2 \vec{B} \tag{50}$$

The magnitudes of the terms on the right-hand side of the equation above are estimated with

$$\nabla \times \left(\vec{V} \times \vec{B} \right) \sim \frac{V_0 B_0}{L_0} \tag{51}$$

$$\frac{1}{\mu_m \sigma} \nabla^2 \vec{B} \sim \frac{B_0}{\mu_m \sigma L_0^2} \tag{52}$$

where B_0, V_0 and L_0 are the characteristic values for the magnetic flux density, velocity and linear dimension, respectively. The magnetic Reynolds number [5] can be obtained by dividing Eq. (51) with Eq. (52), resulting in:

$$Re_m \equiv V_0 L_0 \mu_m \sigma \tag{53}$$

Using a quantity called a magnetic viscosity $\nu_m = 1/(\mu_m \sigma)$, this dimensionless number can be made to resemble the Reynolds number associated with hydrodynamics:

$$Re_m = \frac{V_0 L_0}{\nu_m} \tag{54}$$

3.3 Lorentz Force Source Term for a Momentum Equation

The Lorentz force is an electromagnetic body force which represents the force per unit volume exerted by the magnetic field upon the electric currents [3]. The Lorentz force

$$\vec{F_L} = \vec{J} \times \vec{B} \tag{55}$$

is at right angles to both J and B. In the two-dimensional model explained earlier, current density J has only a z-component and magnetic flux density B x- and y-components. Thus

$$\vec{F_L} = \begin{vmatrix} \vec{i} & \vec{j} & \vec{k} \\ 0 & 0 & J_z \\ B_x & B_y & 0 \end{vmatrix} = -J_z B_y \vec{i} + J_z B_x \vec{j} \tag{56}$$

where the Lorentz force exists only in the xy-plane as magnetic flux density B. Since B can be calculated from the vector potential A the only unknown is current density J_z. This can be calculated from Eq. (43):

$$J_z = \sigma \left(-\frac{\partial A_z}{\partial t} + \frac{\partial \phi}{\partial z} + uB_y - vB_x \right) \tag{57}$$

where $\dfrac{\partial \phi}{\partial z}$ is a source term.

4 Implicit Solution Procedure for a System of Equations

A solution strategy for the system of two-dimensional equations considered in this work is an iterative time stepping algorithm applied with a control volume method. With an initial guess, explicit residuals are calculated for the equations to be solved. The equations are linearized implicitly into delta form and the resulting linear system containing the explicit residual is solved through iterative matrix inversion. Variables are updated using new values from the solved linear equations. Although only steady-state cases are considered here, this algorithm can be easily modified to solve time-accurate cases. All equations are solved as segregated. Continuity and momentum equations are solved with a pressure-correction method. All methods and algorithms presented in this chapter are based on the postgraduate course material written by Siikonen [6].

4.1 Linearization Method

All iteratively solved equations in this work except a pressure-correction equation can be described with a system of equations:

$$\frac{\partial \vec{\Phi}\rho}{\partial t} + \frac{\partial u\vec{\Phi}\rho}{\partial x} + \frac{\partial v\vec{\Phi}\rho}{\partial y} = \nabla \cdot \left(\vec{\Lambda} \cdot \vec{\Phi} \right) - \nabla \cdot \mathbf{P} + \vec{\Psi} \tag{58}$$

Where $\vec{\Phi}$ is a vector that consists of variables to be solved, and $\mathbf{P}$ is a matrix containing pressure in the momentum equation. Λ is a vector containing diffusion coefficients and a nabla operator. Source terms are

inside vector $\vec{\Psi}$. These are defined as:

$$\mathbf{P} = \begin{pmatrix} p & 0 \\ 0 & p \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{pmatrix} \quad \Lambda = \begin{pmatrix} (\mu + \mu_T)\nabla \\ (\mu + \mu_T)\nabla \\ \left(\mu + \frac{\mu_T}{\sigma_k}\right)\nabla \\ \left(\mu + \frac{\mu_T}{\sigma_\epsilon}\right)\nabla \\ (\rho\nu_m)\nabla \end{pmatrix} = \begin{pmatrix} \lambda_1\nabla \\ \lambda_2\nabla \\ \lambda_3\nabla \\ \lambda_4\nabla \\ \lambda_5\nabla \end{pmatrix} \tag{59}$$

$$\Phi = \begin{pmatrix} u \\ v \\ k \\ \tilde{\epsilon} \\ A_z \end{pmatrix} \quad \Psi = \begin{pmatrix} -J_z B_y \\ J_z B_x \\ P - \rho\epsilon + E_k \\ \frac{C_{\epsilon 1}P - C_{\epsilon 2}\rho\tilde{\epsilon} - \rho D e^{-(R_y/80)^2}}{T_t} + E_\epsilon \\ \frac{\rho J_s}{\sigma} \end{pmatrix} \tag{60}$$

where the diffusion term for the vector potential does not contain the eddy viscosity. Eq. (58) is integrated over volume $\mathbf{V}$ and some of the terms are transformed into surface integrals using a divergence theorem of Gauss:

$$\begin{aligned} \int_{\mathbf{V}} \frac{\partial \vec{\Phi}\rho}{\partial t} d\mathbf{V} = &- \int_S \vec{\Phi}\rho\left(\vec{V}\cdot\vec{n}\right) dS + \int_S \left(\vec{\Lambda}\cdot\vec{\Phi}\right)\vec{n} dS \\ &- \int_S \mathbf{P}\vec{n} dS + \int_{\mathbf{V}} \vec{\Psi} d\mathbf{V} \end{aligned} \tag{61}$$

where $\vec{n} = n_x\vec{i} + n_y\vec{j}$ is a surface normal vector. This integral equation is then changed into a discretized form

$$\begin{aligned} \frac{\delta(\vec{\Phi}\rho)}{\delta t}\mathbf{V} = &- \sum_l \left[\vec{\Phi}\rho\left(\vec{V}\cdot\vec{n}S\right)\right]_l + \sum_l \left[\left(\vec{\Lambda}\cdot\vec{\Phi}\right)\vec{n}S\right]_l \\ &- \sum_l \left(\mathbf{P}\vec{n}S\right)_l + \vec{\Psi}\mathbf{V} \end{aligned} \tag{62}$$

where surface and volume integrals are approximated using a midpoint rule. The discretized equation is applied to curvilinear Cartesian two-dimensional cell-centered grid:

$$\rho\frac{\delta\vec{\Phi}_{i,j}}{\delta t}\mathbf{V}_{i,j} = \vec{R}_{i,j} = \vec{F}_{i-1/2} - \vec{F}_{i+1/2} + \vec{F}_{j-1/2} - \vec{F}_{j+1/2} + \left[\vec{\Psi}\mathbf{V}\right]_{i,j} \tag{63}$$

where i and j are indexes in a structured mesh. Eq. (63) is used to calculate explicit residual vector $R_{i,j}$ for control volume (i,j). The residual consists of fluxes $(= \vec{F}_{i-1/2}, \vec{F}_{i+1/2}, \vec{F}_{j-1/2}, \vec{F}_{j+1/2})$ of Φ in and out of a

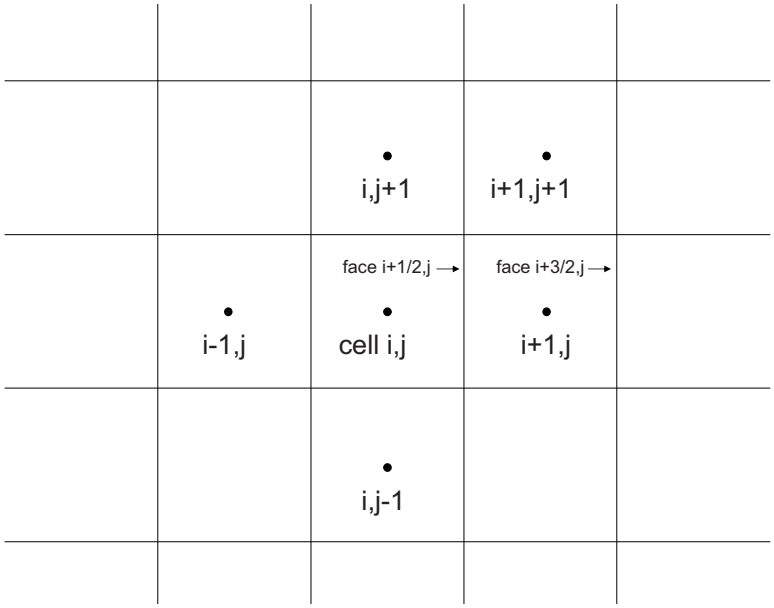

Figure 1: Indexing system

control volume through surfaces $S_{i-1/2,j}, S_{i+1/2,j}, S_{i,j-1/2}, S_{i,j+1/2}$ and the source (Ψ) of Φ inside the volume. In other words the imbalance of Φ inside the control volume produces the residual. For the sake of simplicity surfaces are referred to in the equations with only a single index, since one of them (the whole number) is always the same as the control volume. The index system is presented in Fig. 1. Flux of Φ for the surface $(i-1/2, j)$ is calculated with:

$$\vec{F}_{i-1/2} = \left[\vec{\Phi}\rho\left(\vec{V}\cdot\vec{n}S\right) + \left(\vec{\Lambda}\cdot\vec{\Phi}\right)\vec{n}S - \mathbf{P}\vec{n}S\right]_{i-1/2} \tag{64}$$

where pressure contributes only to the momentum equations. The first component of this vector is

$$F_{1,i-1/2} = \left[\phi_1\rho\left(\vec{V}\cdot\vec{n}S\right) - \lambda_1\left(n_x\frac{\partial\phi_1}{\partial x} + n_y\frac{\partial\phi_1}{\partial y}\right)S - \mathbf{P}\vec{n}S\right]_{i-1/2} \tag{65}$$

where ϕ_1 is the first component of $\vec{\Phi}$ and λ_1 a diffusion coefficient in the first component of Λ. Velocity V and pressure $\mathbf{P}$ at the cell faces

are calculated as averages from cell-center values. To avoid numerical oscillation related to the convection term, an upwind discretization for the convective flux term is formulated as

$$F^C_{1,i-1/2} = \rho S_{i-1/2} \left[\max(0, \vec{V} \cdot \vec{n}) \phi_1^L - \max(0, -\vec{V} \cdot \vec{n}) \phi_1^R \right]_{i-1/2} \tag{66}$$

where values of ϕ_1^R and ϕ_1^L are calculated for the cell faces with a MUSCL-formula from cell-center values. Superscripts L and R denote left-side and right-side upwind interpolation for the face value, respectively. In the MUSCL equation ϕ is limited with a second-order TVD (Total Variation Diminishing) scheme. The partial derivatives located in the diffusion term are approximated by applying the divergence theorem of Gauss. First, the gradient of ϕ is integrated over a volume and transformed into a surface integral:

$$\int_{\mathbf{V}} \nabla \phi d\mathbf{V} = \int_S \phi \vec{n} dS \tag{67}$$

Next, surface integrals are changed to sums over l surfaces

$$\nabla \phi \mathbf{V} = \left(\frac{\partial \phi}{\partial x} + \frac{\partial \phi}{\partial y} \right) \mathbf{V} = \sum_l \left(S \phi \vec{n} \right)_l = \sum_l \left[S \left(\phi n_x + \phi n_y \right) \right]_l \tag{68}$$

which can be separated and divided by volume:

$$\frac{\partial \phi}{\partial x} = \frac{1}{\mathbf{V}} \sum_l \left(S \phi n_x \right)_l \qquad \frac{\partial \phi}{\partial y} = \frac{1}{\mathbf{V}} \sum_l \left(S \phi n_y \right)_l \tag{69}$$

As the derivatives are calculated at the surfaces, a staggered grid is used for geometry information. Staggered grid cell volumes and face surfaces with normal vectors are calculated by averaging geometry information from the original grid. Staggered grid cells in the i- and j-directions are presented with dashed lines in Fig. 2.

The residual equation is linearized using an implicit formulation. The implicit residual R^{n+1} can be calculated using the explicit residual R^n and its linearization. The linearized residual consists of linearized flux and source vectors as follows:

$$\begin{aligned} \rho \frac{\delta \vec{\Phi}_{i,j}}{\delta t} = \vec{R}^{n+1}_{i,j} &= \vec{F}^{n+1}_{i-1/2} - \vec{F}^{n+1}_{i+1/2} + \vec{F}^{n+1}_{j-1/2} - \vec{F}^{n+1}_{j+1/2} + \mathbf{V}_{i,j} \vec{\Psi}^{n+1}_{i,j} \\ &= \vec{F}^{n}_{i-1/2} - \vec{F}^{n}_{i+1/2} + \vec{F}^{n}_{j-1/2} - \vec{F}^{n}_{j+1/2} + \mathbf{V}_{i,j} \vec{\Psi}^{n}_{i,j} \\ &+ \frac{\partial \vec{F}^n_{i-1/2}}{\partial t} \delta t - \frac{\partial \vec{F}^n_{i+1/2}}{\partial t} \delta t + \frac{\partial \vec{F}^n_{j-1/2}}{\partial t} \delta t - \frac{\partial \vec{F}^n_{j+1/2}}{\partial t} \delta t \\ &+ \mathbf{V}_{i,j} \frac{\partial \vec{\Psi}^n_{i,j}}{\partial t} \delta t = \vec{R}^n_{i,j} + \frac{\partial \vec{R}^n_{i,j}}{\partial t} \delta t \end{aligned} \tag{70}$$

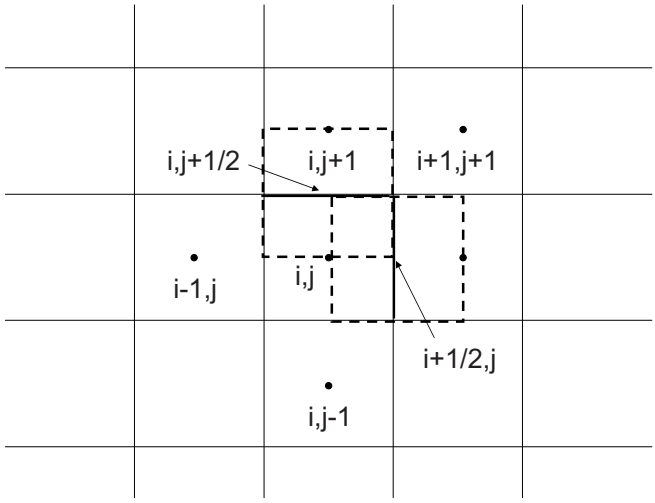

Figure 2: Staggered grid cells at surfaces $(i + 1/2, j)$ and $(i, j + 1/2)$.

In the linearization of the flux vector F in Eq. (64) the third term always produces a zero after derivation, since it does not contain any of the unknowns. Thus only the convection and diffusion terms need to be considered. The first two components in flux vector F have different properties than others, since unknown variables u and v exist outside vector $\vec{\Phi}$. To have a similar form for every linearized flux vector component, the derivation is only performed with respect to Φ. The principle of this approximation is based on Picard iteration. This simplifies the algorithms considerably, since all unknowns can be treated in the same way in the flux term. To show the linearization of the flux terms in greater detail, the first component of these vectors is considered. Since the linearization of the flux vector depends only on the vector $\vec{\Phi}$ the following modification is made:

$$\frac{\partial F(\phi_1)^n}{\partial t}\delta t = \frac{\partial F(\phi_1)^n}{\partial \phi_1}\frac{\delta \phi_1}{\delta t}\delta t = \frac{\partial F(\phi_1)^n}{\partial \phi_1}\delta \phi_1 \tag{71}$$

Before the derivation the convective flux Eq. (66) is changed to

$$F^C_{1,i-1/2} = \rho S_{i-1/2}\left[\mathbf{max}(0, \vec{V}_{i-1/2}\cdot\vec{n}_{i-1/2})\phi_{1,i-1} - \mathbf{max}(0, -\vec{V}_{i-1/2}\cdot\vec{n}_{i-1/2})\phi_{1,i}\right] \tag{72}$$

and the diffusive flux term is approximated with

$$F^D_{1,i-1/2} = -\left[S\lambda_1\left(n_x\frac{\partial \phi_1}{\partial x} + n_y\frac{\partial \phi_1}{\partial y}\right)\right]_{i-1/2} = -\left(S\lambda_1\right)_{i-1/2}\frac{\phi_{1,i}-\phi_{1,i-1}}{\Delta\xi_{i-1/2}} \tag{73}$$

where $\Delta\xi_{i-1/2}$ is the distance between cell centers (i,j) and $(i-1,j)$. Conversely $\Delta\eta_{i-1/2}$ denotes the distance between cell centers (i,j) and $(i,j-1)$. Derivation of the first flux component at surface $(i-1/2,j)$ is performed with respect to $\phi_{1,i}$ and $\phi_{1,i-1}$:

$$\begin{aligned}\frac{\partial F(\phi_1)^n_{i-1/2}}{\partial \phi_1}\delta\phi_1 &= \frac{\partial F(\phi_1)^n_{i-1/2}}{\partial \phi_{1,i-1}}\delta\phi_{1,i-1} + \frac{\partial F(\phi_1)^n_{i-1/2}}{\partial \phi_{1,i}}\delta\phi_{1,i} \\ &= \left[\max(0,\vec{V}_{i-1/2}\cdot\vec{n}_{i-1/2})\rho S_{i-1/2} + \left(\frac{S\lambda_1}{\Delta\xi}\right)_{i-1/2}\right]\delta\phi_{1,i-1} \\ &- \left[\max(0,-\vec{V}_{i-1/2}\cdot\vec{n}_{i-1/2})\rho S_{i-1/2} + \left(\frac{S\lambda_1}{\Delta\xi}\right)_{i-1/2}\right]\delta\phi_{1,i}\end{aligned} \tag{74}$$

which corresponds to a first-order upwind discretization. Linearizations for the source terms and the flux terms for other surfaces $(i+$

$1/2, j)$, $(i, j-1/2)$ and $(i, j+1/2)$ can be found in Appendix A3. Source and flux linearizations from all faces are substituted into Eq. (70) and the terms are grouped to make coefficients for $\delta\phi_1$

$$R^n_{i,j} = A_{w,i,j}\delta\phi_{i-1,j} + A_{e,i,j}\delta\phi_{i+1,j} + A_{s,i,j}\delta\phi_{i,j-1} + A_{n,i,j}\delta\phi_{i,j+1} + A_{p,i,j}\delta\phi_{i,j} \quad (75)$$

where the coefficients are defined as follows:

$$\begin{aligned}
A_{w,i,j} &= -\max(0, \vec{V}_{i-1/2} \cdot \vec{n}_{i-1/2})\rho S_{i-1/2} - \left(\frac{S\lambda_1}{\Delta\xi}\right)_{i-1/2} \\
A_{e,i,j} &= -\max(0, -\vec{V}_{i+1/2} \cdot \vec{n}_{i+1/2})\rho S_{i+1/2} - \left(\frac{S\lambda_1}{\Delta\xi}\right)_{i+1/2} \\
A_{s,i,j} &= -\max(0, \vec{V}_{j-1/2} \cdot \vec{n}_{j-1/2})\rho S_{j-1/2} - \left(\frac{S\lambda_1}{\Delta\eta}\right)_{j-1/2} \\
A_{n,i,j} &= -\max(0, -\vec{V}_{j+1/2} \cdot \vec{n}_{j+1/2})\rho S_{j+1/2} - \left(\frac{S\lambda_1}{\Delta\eta}\right)_{j+1/2} \\
A_{p,i,j} &\approx \mathbf{V}\left(\frac{\rho}{\delta t} - \frac{\partial\psi_1}{\partial t}\right) - A_{w,i,j} - A_{e,i,j} - A_{s,i,j} - A_{n,i,j}
\end{aligned} \quad (76)$$

Subscripts w, e, s, n and p denote west, east, south, north and middle cell, respectively. Coefficient A_p is not exactly a negative sum of the other coefficients added with the volume term. But in this form the value of the diagonal is suitable for an iterative matrix inversion (Scarborough criteria). The linearized equation system can be now written as

$$A\delta\phi = R^n \quad (77)$$

which can be solved by inverting matrix A ($\delta\phi = A^{-1}R^n$). This is done separately for each component of $\vec{\Phi}$. The inversions are performed with an iterative line-Gauss-Seidel (LGS) algorithm. As the solutions are obtained, variables are updated with:

$$\vec{\Phi}^{n+1} = \vec{\Phi}^n + \delta\vec{\Phi} \quad (78)$$

To clarify the numerical principle that is used above a few key issues are now repeated and further explained. In the flux components that are included in the explicit residual R^n the values of $\vec{\Phi}$ at the faces are approximated with MUSCL-interpolations using old values $\vec{\Phi}^n$. In the flux components that are derivated to form the coefficients of matrix A, the values of $\vec{\Phi}$ at the faces are approximated with first order upwind.

A lower order approximation for the implicit equations is used to keep the computational molecule as small as possible. To solve larger computational molecules a more complicated and sophisticated iterative solver would have to be used to invert the matrix A. When a solution has converged, the accuracy of the result is determined by the accuracy of the explicit residual. Thus, the accuracy of the converged solution is unaffected by the lower order approximation of the implicit flux terms. This type of procedure was first suggested by MacCormack [7] for compressible flows. MacCormack proposes that a desirable form for the Navier-Stokes equations is

$$\{numerics\}\delta\phi = \{physics\}$$

Here {physics} is described by the explicit residual and {numerics} by the matrix A. Later this procedure was described as a deferred correction by Ferziger&Perić [8], where the right-hand side of the equation, the explicit residual, is regarded as a correction.

A steady-state solution can be accelerated using a local time-step. An optimal time step size is often empirical and differs in problems. Stability and a speed of convergence of a convective flow is related to the Courant number:

$$cfl = \frac{u \cdot \delta t}{\Delta x} \tag{79}$$

which can be transformed into a curvilinear form:

$$\begin{aligned} cfl_{i,j} = \delta t^C_{i,j} \frac{(un_x)_{i-1/2,j} + (un_x)_{i+1/2,j} + (vn_y)_{i-1/2,j} + (vn_y)_{i+1/2,j}}{\Delta\xi_{i,j}} \\ + \delta t^C_{i,j} \frac{(un_x)_{i,j-1/2} + (un_x)_{i,j+1/2} + (vn_y)_{i,j-1/2} + (vn_y)_{i,j+1/2}}{\Delta\eta_{i,j}} \end{aligned} \tag{80}$$

from where a local time step $\delta t^c_{i,j}$ can be solved. In low Reynolds number flows the effect of the diffusion term should also be taken into account. Thus another numerical parameter is defined:

$$C^{dif}_{i,j} = \frac{\delta t^D_{i,j} \max\left(\frac{1}{\Delta\xi^2_{i,j}}, \frac{1}{\Delta\eta^2_{i,j}}\right)\lambda}{2\rho} \tag{81}$$

where λ is a diffusion coefficient. From these two time-step values the smaller one is chosen for a local cell. Suitable values for C^{dif} and the cfl range in most cases between 0.5 - 5. In magnetohydrodynamic flows the momentum equation and the magnetic vector potential

equation might have completely different ratios between diffusive and convective fluxes. Therefore, diffusive time step sizes for these equations should be calculated separately. Either both equations are calculated with the same local minimum time step or different estimates are used. Another approach for increasing stability or under-relaxation for steady-state calculations locally is to neglect the term containing δt in the coefficient A_p and just divide the diagonal values by α_ϕ, which has a value of $0 < \alpha_\phi \leq 1.0$. This has the same effect as the time-step related under-relaxation, which is to increase the diagonal values for the matrix A.

4.2 Pressure-Correction Method

After the momentum equations have been linearized and the velocity components have been updated, the pressure is solved using the continuity equation. New values of u and v will satisfy the linearized momentum equation if the matrix inversion was fully completed and no under-relaxation was made when updating the values. However, they will not satisfy the continuity equation. New values for the velocities from the solution of the momentum equations are defined now as u^* and v^*. These values have to be corrected with velocities u' and v' to satisfy the continuity equation:

$$u^{n+1} = u^* + u' \quad v^{n+1} = v^* + v' \tag{82}$$

Before a mass balance can be calculated, the incompressible continuity equation is integrated over the volume of a computational cell, transformed into a surface integral and further to a sum over l surfaces:

$$\int_{\mathbf{V}} \nabla \rho \vec{V} d\mathbf{V} = \int_{S} \rho \vec{V} \cdot \vec{n} dS = \sum_{l} \left(\rho \vec{V} \cdot \vec{n} S \right)_l = 0 \tag{83}$$

where $\vec{V} \cdot \vec{n} = \overline{V}$ is later used as a velocity perpendicular to the surface. The corrected velocity values are now substituted into continuity equation to form a mass balance equation for control volume (i, j). Corrections are rearranged into another side of the equation:

$$\begin{aligned} &-(\rho \overline{V}' S)_{i-1/2} + (\rho \overline{V}' S)_{i+1/2} - (\rho \overline{V}' S)_{j-1/2} + (\rho \overline{V}' S)_{j+1/2} = \\ &(\rho \overline{V}^* S)_{i-1/2} - (\rho \overline{V}^* S)_{i+1/2} + (\rho \overline{V}^* S)_{j-1/2} - (\rho \overline{V}^* S)_{j+1/2} = -\Delta \dot{m}_{i,j} \end{aligned} \tag{84}$$

where $-\Delta \dot{m}_{i,j}$ represents the error in mass balance for cell (i, j). During the very first iterations there exist large errors in the form of oscillations in the velocity and pressure values. If only velocity values V^*

were used in the mass balance calculations, the damping of these oscillations would be very slow. In other words, many iterations would be needed to obtain an accurate solution. Fortunately, this damping can be increased with a Rhie&Chow scheme [9]. The scheme introduces a stronger link between the pressure and the velocity. In practice the interpolation for the pressure is combined with the face value of the velocity at $(i-1/2,j)$ as follows:

$$\overline{V}^*_{i-1/2} = \frac{1}{2}\left(\vec{V}^*_{i-1} + \vec{V}^*_i\right)\cdot\vec{n}_{i-1/2} + C\frac{S_{i-1/2}}{4A^V_{p,i-1/2}}\left(p_{i+1} - 3p_i + 3p_{i-1} - p_{i-2}\right) \tag{85}$$

where $C(=0.5)$ is an empirical coefficient for adjusting the magnitude of the pressure term. A^V_p is a coefficient from the linearized momentum equation and can be calculated as an average of cell center values at the face or from equation

$$\frac{1}{A^V_{p,i-1/2}} = \frac{1}{2}\left(\frac{1}{A^V_{p,i}} + \frac{1}{A^V_{p,i-1}}\right) \tag{86}$$

Now the mass balance can be calculated properly. If Eq 76 is used in the calculation of A^V_p, under-relaxation parameters cfl and C^{vis} can have a small effect on the converged result. This happens because the Rhie&Chow damping terms do not vanish completely when a solution has reached the truncation error level. But as the error produced by the Rhie&Chow terms is proportional to $O(\Delta x^2)$, this error gets very small as the grid is successively refined. To remove the effect of the under-relaxation parameters on the solution, one can use a different A^V_p coefficient, where the term containing time step is omitted when calculating the damping terms. Next a connection between the correction values for velocity and pressure have to be established. First the corrected velocity values $\overline{V}^{n+1}$ from the faces are multiplied by the coefficients from a linearized momentum equation for staggered grid cell $(i+1/2,j)$ and terms containing V^* values are combined to form

$$\begin{aligned} A^V_{w,i+1/2,j}\overline{V}^*_{i-1/2,j} + A^V_{e,i+1/2,j}\overline{V}^*_{i+3/2,j} + A^V_{s,i+1/2,j}\overline{V}^*_{i+1/2,j-1} \\ + A^V_{n,i+1/2,j}\overline{V}^*_{i+1/2,j+1} + A^V_{p,i+1/2,j}\overline{V}^*_{i+1/2,j} = 0 \end{aligned} \tag{87}$$

where the equation equals zero since the V^* values are assumed to satisfy the linearized momentum equation. The terms containing velocity

correction values V' equal

$$\begin{aligned} & A^V_{w,i+1/2,j}\overline{V}'_{i-1/2,j} + A^V_{e,i+1/2,j}\overline{V}'_{i+3/2,j} + A^V_{s,i+1/2,j}\overline{V}'_{i+1/2,j-1} \\ & + A^V_{n,i+1/2,j}\overline{V}'_{i+1/2,j+1} + A^V_{p,i+1/2,j}\overline{V}'_{i+1/2,j} \\ & = -(Sp')_{i+1,j} + (Sp')_{i,j} + p'_{i+1/2,j}(S_{i+1,j} - S_{i,j}) \\ & \approx -S_{i+1/2,j}(p'_{i+1,j} - p'_{i,j}) \end{aligned} \tag{88}$$

where the terms containing pressure are from the discretized momentum equation in the local coordinate system. Pressure p' is a correction that is to be calculated from the pressure correction equation. Before that, the equation above is simplified with a SIMPLE approximation:

$$A^V_{p,i+1/2,j}\overline{V}'_{i+1/2,j} \approx -S_{i+1/2,j}(p'_{i+1,j} - p'_{i,j}) \tag{89}$$

where other velocity terms are just neglected. Although the link between the velocity and the pressure gets inaccurate, this will result in a more coherent equation for pressure. No values from the previous time-step are used for corrections. Thus this method is implicit. The corresponding equations for other faces are:

$$\begin{aligned} A^V_{p,i-1/2,j}\overline{V}'_{i-1/2,j} &\approx -S_{i-1/2,j}(p'_{i,j} - p'_{i-1,j}) \\ A^V_{p,i,j-1/2}\overline{V}'_{i,j-1/2} &\approx -S_{i,j-1/2}(p'_{i,j} - p'_{i,j-1}) \\ A^V_{p,i,j+1/2}\overline{V}'_{i,j+1/2} &\approx -S_{i,j+1/2}(p'_{i,j+1} - p'_{i,j}) \end{aligned} \tag{90}$$

where the two lower equations are derived from an equation for a staggered grid cell in the j-direction. In the local coordinate system this equation is

$$\begin{aligned} & A^V_{w,i,j+1/2}\overline{V}'_{i-1,j+1/2} + A^V_{e,i,j+1/2}\overline{V}'_{i+1,j+1/2} + A^V_{s,i,j+1/2}\overline{V}'_{i,j-1/2} \\ & + A^V_{n,i,j+1/2}\overline{V}'_{i,j+3/2} + A^V_{p,i,j+1/2}\overline{V}'_{i,j+1/2} \\ & = -(Sp')_{i,j+1} + (Sp')_{i,j} + p'_{i,j+1/2}(S_{i,j+1} - S_{i,j}) \\ & \approx -S_{i,j+1/2}(p'_{i,j+1} - p'_{i,j}) \end{aligned} \tag{91}$$

Next the velocities are solved from the corrections obtained with the SIMPLE method. These velocities are then substituted into the mass balance equation and rearranged to form:

$$A^p_w p'_{i-1,j} + A^p_e p'_{i+1,j} + A^p_s p'_{i,j-1} + A^p_n p'_{i,j+1} + A^p_p p'_{i,j} = -\Delta\dot{m}_{i,j} \tag{92}$$

where the coefficients A_w^p, A_e^p, A_s^p, A_n^p and A_p^p are defined as:

$$\begin{aligned} A_w^p &= -\left(\frac{\rho S^2}{A_p^V}\right)_{i-1/2} \quad A_e^p = -\left(\frac{\rho S^2}{A_p^V}\right)_{i+1/2} \\ A_s^p &= -\left(\frac{\rho S^2}{A_p^V}\right)_{j-1/2} \quad A_n^p = -\left(\frac{\rho S^2}{A_p^V}\right)_{j+1/2} \\ A_p^p &= -A_w^p - A_e^p - A_s^p - A_n^p \end{aligned} \tag{93}$$

The linear system $Ap' = -\Delta\dot{m}$ is solved iteratively with a line Gauss-Seidel iteration and the pressure is corrected with

$$p^{n+1} = p^n + \alpha_p p' \tag{94}$$

where the update is under-relaxed by a factor of α_p. Finally, the velocity components have to be corrected in order to satisfy the continuity equation. The velocity can be corrected with a combination of the SIMPLE approximation and the momentum equation:

$$A_{p,i,j}^V u'_{i,j} \approx -(Sn_x p')_{i+1/2,j} + (Sn_x p')_{i-1/2,j} - (Sn_x p')_{i,j+1/2} + (Sn_x p')_{i,j-1/2} \tag{95}$$

from where the correction for the u component can be solved. By replacing the normal vector component x with y the equation for v velocity can be formulated. The pressure values at the faces are again calculated as averages from the cell centers. The velocities are corrected with:

$$u^{n+1} = u^* + \alpha_u u' \quad v^{n+1} = v^* + \alpha_u v' \tag{96}$$

where the corrections are under-relaxed with parameter α_u.

4.3 Treatment of Boundaries

To simplify routines at boundaries a ghost-cell method is applied. With the ghost cells, interpolations at the boundaries can be performed in exactly the same way as inside the mesh. Two ghost cells are used at every boundary to make a second order MUSCL interpolation possible. The ghost cells and examples of velocity-vector indexes are presented in Fig. 3. The values that are put into the ghost cells depend on the boundary conditions and the values inside the computational domain.

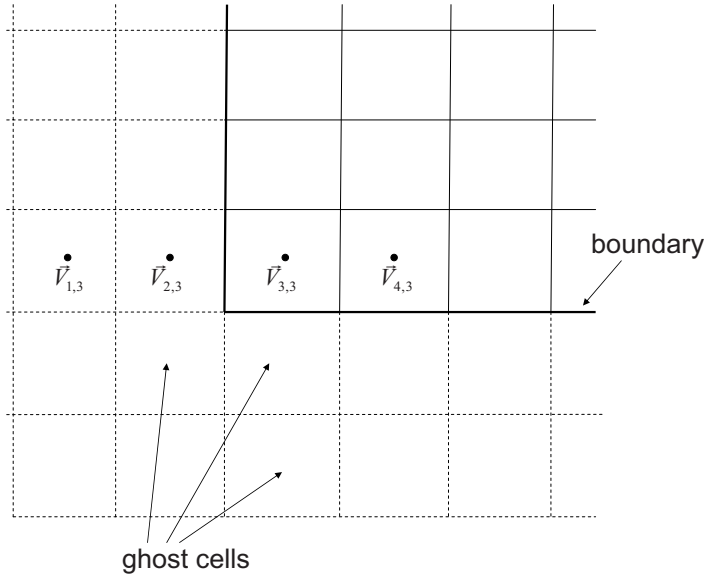

Figure 3: Ghost cells

A MUSCL-equation

$$\phi_{i+1/2} = \phi_i + \frac{1}{4}\left[(1-\kappa)\left(\phi_i - \phi_{i-1}\right) + (1+\kappa)\left(\phi_{i+1} - \phi_i\right)\right], \qquad \overline{V}_{i+1/2} > 0$$
$$\phi_{i+1/2} = \phi_{i+1} + \frac{1}{4}\left[(1-\kappa)\left(\phi_{i+1} - \phi_{i+2}\right) + (1+\kappa)\left(\phi_i - \phi_{i+1}\right)\right], \qquad \overline{V}_{i+1/2} < 0 \tag{97}$$

can be used to interpolate values of ϕ at the faces. Different schemes can be used by changing the value of κ. A central difference is enabled, when $\kappa = 1$. Other schemes are a second-order upwind ($\kappa = 0$), a second-order upwind biased ($\kappa = -1$), QUICK ($\kappa = 1/2$) and a third-order upwind method ($\kappa = 1/3$).

4.3.1 Boundary Conditions for the Velocity Vector

If the left side of the domain in Fig. 3 is considered as a velocity inlet, the ghost-cell velocity values $\vec{V}_1$ and $\vec{V}_2$ can be solved for different

schemes from equation pairs

$$\begin{aligned}
\vec{V}_{in} &= 3/2\vec{V}_2 - 1/2\vec{V}_1 & \vec{V}_{in} &= 1/2(\vec{V}_2 + \vec{V}_3) \\
\vec{V}_{in} &= \vec{V}_2 + 1/4(\vec{V}_3 + \vec{V}_1) & \vec{V}_{in} &= 1/2(\vec{V}_2 + \vec{V}_3) \\
\vec{V}_{in} &= 3/4\vec{V}_2 + 3/8\vec{V}_3 - 1/8\vec{V}_1 & \vec{V}_{in} &= 1/2(\vec{V}_2 + \vec{V}_3) \\
\vec{V}_{in} &= 5/6\vec{V}_2 + 1/3\vec{V}_3 - 1/6\vec{V}_1 & \vec{V}_{in} &= 1/2(\vec{V}_2 + \vec{V}_3)
\end{aligned} \tag{98}$$

where the first equation set is based on the MUSCL-equation and the second one is a central difference (average), which is also used at the faces. Surprisingly, all equation pairs give the same solution for the first ghost cell:

$$\vec{V}_1 = 4\vec{V}_{in} - 3\vec{V}_3 \quad \vec{V}_2 = 2\vec{V}_{in} - \vec{V}_3 \tag{99}$$

A value for the second cell is obviously also the same for each scheme. If the inlet velocity $\vec{V}_{in}$ represents a velocity perpendicular to the inlet, a value $(\vec{n}V)_{in}$ can be used instead. Here V_{in} is a velocity perpendicular to the face. For the outlet surface a zero velocity gradient can be assumed and thus

$$\vec{V}_2 = \vec{V}_3 \quad \vec{V}_1 = \vec{V}_4 \tag{100}$$

In the case of tangentially moving boundary (wall) the ghost-cell velocity values can be solved from averages

$$\begin{aligned}
(\vec{t}V)_{wall} &= 1/2(\vec{V}_2 + \vec{V}_3) \\
(\vec{t}V)_{wall} &= 1/2(\vec{V}_1 + \vec{V}_4)
\end{aligned} \tag{101}$$

where $\vec{t}$ is a tangential unit vector of the cell face at the wall. The solutions

$$\begin{aligned}
\vec{V}_2 &= 2(\vec{t}V)_{wall} - \vec{V}_3 \\
\vec{V}_1 &= 2(\vec{t}V)_{wall} - \vec{V}_4
\end{aligned} \tag{102}$$

will produce nonzero normal velocities with MUSCL-interpolations at the wall, but these velocities will be of an opposite value and, therefore, the total mass flux through wall will be zero when Eq. (66) is used. Also the average normal velocity is always zero at the wall. A tangential velocity will only be correctly calculated with the average. A frictionless wall can be taken into account using a symmetry condition. The ghost-cell values are

$$\begin{aligned}
\begin{pmatrix} u_2 \\ v_2 \end{pmatrix} &= \begin{pmatrix} 1 - 2n_x^2 & -2n_x n_y \\ -2n_x n_y & 1 - 2n_y^2 \end{pmatrix} \begin{pmatrix} u_3 \\ v_3 \end{pmatrix} \\
\begin{pmatrix} u_1 \\ v_1 \end{pmatrix} &= \begin{pmatrix} 1 - 2n_x^2 & -2n_x n_y \\ -2n_x n_y & 1 - 2n_y^2 \end{pmatrix} \begin{pmatrix} u_4 \\ v_4 \end{pmatrix}
\end{aligned} \tag{103}$$

Using this equation the normal velocity of average

$$\vec{V}_{wall} \cdot \vec{n} = 1/2\vec{n} \cdot \left[(u_2 + u_3)\,\vec{i} + (v_2 + v_3)\,\vec{j} \right] \tag{104}$$

produces a zero value, which can be seen after a manipulation:

$$\vec{V}_{wall} \cdot \vec{n} = u_3 n_x \left[1 - \left(n_x^2 + n_y^2 \right) \right] + v_3 n_y \left[1 - \left(n_x^2 + n_y^2 \right) \right] = 0 \tag{105}$$

In the case of a periodic boundary condition for velocity, a mass-flow rate has to be given initially and kept constant during iterations. If the initial velocity profile is given, the mass flow rate can be calculated by summing

$$\dot{m}_{in} = \frac{1}{2} \sum_{j=3}^{J-2} \rho S_{2+1/2,j} \vec{n}_{2+1/2,j} \cdot \left(\vec{V}_{2,j} + \vec{V}_{3,j} \right) \tag{106}$$

where J is the last cell number in the j-direction. In the first iteration normal inlet and outlet methods are used, but afterwards periodicity is enforced. At the end of the iteration cycle the outlet mass-flow rate is calculated with

$$\dot{m}_{out} = \frac{1}{2} \sum_{j=3}^{J-2} \rho S_{I-3/2,j} \vec{n}_{I-3/2,j} \cdot \left(\vec{V}_{I-2,j} + \vec{V}_{I-1,j} \right) \tag{107}$$

where I is the last cell number in the i-direction. Next the ratio of the enforced mass rate and the outlet mass-flow rate is calculated

$$C_r = \frac{\dot{m}_{in}}{\dot{m}_{out}} \tag{108}$$

and new velocity values for the "inlet" are calculated by

$$\vec{V}_{in,j} = 1/2 C_r \left(\vec{V}_{I-2,j} + \vec{V}_{I-1,j} \right) \tag{109}$$

In the next iteration cycle these velocity values at the inlet can be used with Eq. (99) to calculate ghost-cell values. Ghost-cell values for an "outlet" are

$$\vec{V}_{I-1} = \vec{V}_1 \quad \vec{V}_I = \vec{V}_2 \tag{110}$$

where the ghost-cell values from the "inlet" are just put into the ghost cells of an "outlet". Again at the end of the iteration cycle a new outlet mass flow rate and a new ratio C_r are calculated.

4.3.2 Boundary Conditions for Pressure

In the case of the velocity inlet ghost cell, values for pressure can be extrapolated from averages

$$p_3 = 1/2(p_2 + p_4) \quad p_2 = 1/2(p_1 + p_3) \tag{111}$$

which give

$$p_2 = 2p_3 - p_4 \quad p_1 = 4p_3 - 3p_4 \tag{112}$$

after manipulation. Near the walls a zero gradient condition

$$p_2 = p_3 \quad p_1 = p_4 \tag{113}$$

can be enforced to ensure a zero mass flow at the wall. For the outlet the pressure should be fixed

$$p_2 = p_{out} \quad p_1 = p_{out} \tag{114}$$

If the whole domain is surrounded by walls, no outlet exists. Still the pressure must be fixed at some point to enable convergence. This point can be set at the boundary wall inside the ghost cells using the equation above. If all boundaries are moving, a better result might be obtained by fixing the pressure at some point inside the domain. In the periodic case one of the periodic pairs should be set as the outlet and the other as the inlet, and corresponding conditions used for pressure.

4.3.3 Boundary Conditions for Scalars

In the velocity inlet case scalar variable ϕ can be calculated for ghost cells similarly as for velocity with Eq. (99).

$$\phi_1 = 4\phi_{in} - 3\phi_3 \quad \phi_2 = 2\phi_{in} - \phi_3 \tag{115}$$

In the case of a wall condition, scalar values can be solved from averages. The solutions are

$$\begin{aligned} \phi_2 &= 2\phi_{wall} - \phi_3 \\ \phi_1 &= 2\phi_{wall} - \phi_4 \end{aligned} \tag{116}$$

for ghost cells. Since the scalars are calculated as averages at the faces in Eq. (69), the ghost-cell values can also be based on averages. For the outlet a zero gradient condition is suitable.

4.3.4 Boundary Conditions for Linearized Implicit Equations

Implicit boundary conditions can be determined by linearizing the explicit boundary conditions. On the other hand the boundary conditions of the implicit equations can be treated with some flexibility to ensure iterative matrix inversion (Scarborough criteria). The velocity inlet condition is linearized for the u-component as follows:

$$\begin{aligned} &\frac{\partial}{\partial u_2}(u_2)\,\delta u_2 = 2\frac{\partial}{\partial u_{in}}(u_{in})\,\delta u_{in} - \frac{\partial}{\partial u_3}(u_3)\,\delta u_3 \\ &\delta u_2 = -\,\delta u_3; \quad \delta u_{in} = 0 \\ &\delta u_2 + \delta u_3 = 0 \\ &A_{p,2}\delta u_2 + A_{e,2}\delta u_3 = 0 \end{aligned} \tag{117}$$

where a local derivative for the inlet is zero, since it is a fixed value. The terms are then organized to the left side. Coefficient $A_{p,2} = 1$ and $A_{e,2} = 1$. Coefficient $A_{e,2}$ can also be set to a zero value to ensure a diagonal dominance. The other coefficients $A_{w,2}, A_{s,2}, A_{n,2}$ and residual R_2 are equal to zero. Wall boundary conditions with a frictionless condition can be treated identically. A zero-gradient condition for the u-velocity is linearized as

$$\begin{aligned} &\frac{\partial}{\partial u_2}(u_2)\,\delta u_2 = \frac{\partial}{\partial u_3}(u_3)\,\delta u_3 \\ &\delta u_2 - \delta u_3 = 0 \end{aligned} \tag{118}$$

where coefficient $A_{p,2} = 1$ and $A_{e,2} = -1$.

5 Applied Multigrid Techniques

In practical modeling problems it is important to have efficient algorithms to ensure fast convergence. Hardware and time are wasted if advances in numerical methods are not put to use. A useful way to examine the efficiency of a numerical method is to change the size of the mesh and see how long it takes to get a converged solution. More cells means more equations to be solved. Then, inevitably, more time and storage space are needed. In the optimal case with multigrid methods, work and storage requirements depend linearly on the size of the problem.

5.1 Algebraic Multigrid

During a single time step (=iteration cycle) most of the time is used in the iterative inversion of coefficient matrix A in the linear equation system, $A\delta\phi = R$. This is the case especially for the pressure-correction equation. If only LGS is used in the inversion, the solution time depends quadratically on the mesh size. An algebraic multigrid (AMG) method has been developed to accelerate the solution of a linear equation system. To understand how the multigrid methods work, the error has to be considered more closely. The error present in the solution can be approximated as a spectrum of waves. The size of these waves ranges from $2\Delta x$ to the length of the domain $2L$ in a one-dimensional case. During the iterative inversion of the matrix the error waves are smoothed cycle after cycle. Therefore, the basic matrix-inverting algorithms are called smoothers. The basic property of a smoother like LGS applied in this study is that smaller wavelengths (higher frequencies) are smoothed faster than the longer ones. The bigger the mesh, the longer (relatively) the wavelengths present. The longer the waves, the more inefficient the smoothing and, therefore, the inversion time increases rapidly with mesh size [10]. To improve this smoothing process the problem is divided into several grid levels. In a structured mesh the generation of a new grid level is very straightforward. In a two-dimensional problem four neighboring cells are combined through the domain to make a new level. A third grid level is generated in exactly the same way using the second level. This process can be continued as long as the cell counts in the i- and j-directions of the mesh are divisible by two. The second grid level has four times fewer cells and the third 16 times fewer than the first level. The actual geometrical coordinates are not needed in the coarse grid generation. Only the knowledge of the cell neighbors is relevant. The coefficient matrix A and the residual matrix R have to be transported to the coarser levels. This is called restriction. The coefficients do not change during the inversion and, therefore, can be restricted only at the beginning of the algorithm. In Fig. 4 it can be seen how coefficients can be grouped to the coarser level with a Galerkin coarse grid approximation (GCA) [6]. All the coefficients that are inside the coarse cell are summed together.

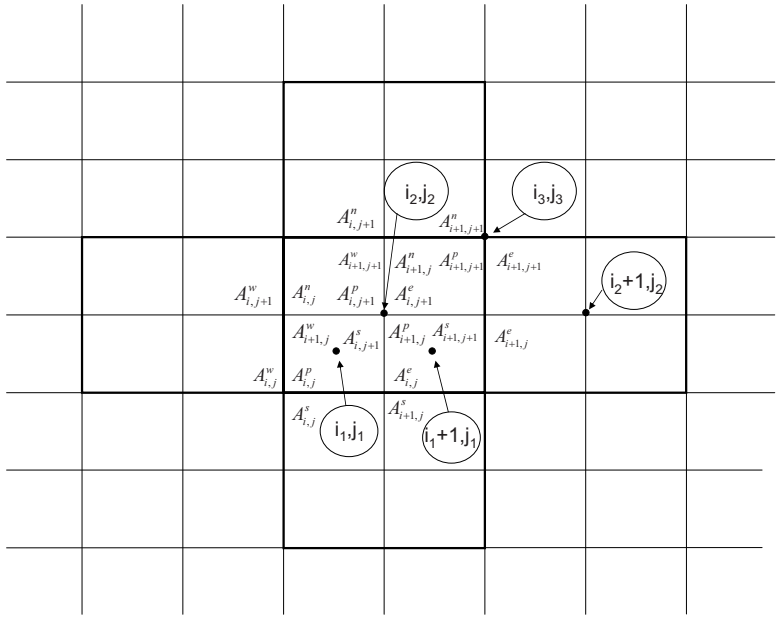

Figure 4: Coarse grid coefficients of AMG

On the second level the coefficients are as follows:

$$\begin{aligned}
A^2_{w,i_2,j_2} &= A_{w,i_1,j_1} + A_{w,i_1,j_1+1} \\
A^2_{e,i_2,j_2} &= A_{e,i_1+1,j_1} + A_{e,i_1+1,j_1+1} \\
A^2_{s,i_2,j_2} &= A_{s,i_1,j_1} + A_{s,i_1+1,j_1} \\
A^2_{n,i_2,j_2} &= A_{n,i_1,j_1+1} + A_{n,i_1+1,j_1+1} \\
A^2_{p,i_2,j_2} &= A_{p,i_1,j_1} + A_{p,i_1+1,j_1} + A_{p,i_1,j_1+1} + A_{p,i_1+1,j_1+1} \\
&\quad + A_{w,i_1+1,j_1} + A_{w,i_1+1,j_1+1} + A_{e,i_1,j_1} + A_{e,i_1,j_1+1} \\
&\quad + A_{s,i_1,j_1+1} + A_{s,i_1+1,j_1+1} + A_{n,i_1,j_1} + A_{n,i_1+1,j_1}
\end{aligned} \tag{119}$$

where indexes i_2 and j_2 refer to the second grid level. The third level coefficients are generated identically from the second level coefficients, and so on. Before the restriction of the residual matrix the lowest wavelengths of the error are smoothed on the finest grid level with a single cycle using LGS. Next the residual to be restricted is defined as:

$$\omega^1 = R^1 - A^1 \delta\phi^1 \tag{120}$$

which is locally

$$\begin{aligned}
\omega^1_{i_1,j_1} = R^1_{i_1,j_1} - (&A^1_{p,i_1,j_1}\delta\phi^1_{i_1,j_1} + A^1_{w,i_1,j_1}\delta\phi^1_{i_1-1,j_1} + A^1_{e,i_1,j_1}\delta\phi^1_{i_1+1,j_1} + \\
&A^1_{s,i_1,j_1}\delta\phi^1_{i_1,j_1-1} + A^1_{n,i_1,j_1}\delta\phi^1_{i_1,j_1+1})
\end{aligned} \tag{121}$$

where ω^1 is a modified residual that describes an error of the linear equation system on the finest level. The residual on the second level is the sum of the modified fine grid residuals ω^1 inside the coarse cell:

$$R^2_{i_2,j_2} = \omega^1_{i_1,j_1} + \omega^1_{i_1+1,j_1} + \omega^1_{i_1,j_1+1} + \omega^1_{i_1+1,j_1+1}. \tag{122}$$

Now, longer wavelengths can be smoothed more efficiently on the second grid level, since it is coarser and cannot contain as high frequency error components as the finest grid level. System $A^2\delta\phi^2 = R^2$ is smoothed by several iterations (=2-5) with LGS . It is more economic to smooth more on the second level as it has four times fewer equations. If there are more grid levels, the second level residuals need to be restricted to the next level in the same way as before. Let us assume there are only two grid levels. The next step is to correct the fine-level values of $\delta\phi^1$. This is called prolongation. The second-level values $\delta\phi^2$ are transported to the fine level by summing them with the old values of $\delta\phi^{1,n}$

as:

$$\begin{array}{ll}
\delta\phi^{1,n+1}_{i_1,j_1} = \delta\phi^{1,n}_{i_1,j_1} + \delta\phi^{2}_{i_2,j_2} & \delta\phi^{1,n+1}_{i_1+2,j_1} = \delta\phi^{1,n}_{i_1+2,j_1} + \delta\phi^{2}_{i_2+1,j_2} \\
\delta\phi^{1,n+1}_{i_1+1,j_1} = \delta\phi^{1,n}_{i_1+1,j_1} + \delta\phi^{2}_{i_2,j_2} & \delta\phi^{1,n+1}_{i_1+3,j_1} = \delta\phi^{1,n}_{i_1+3,j_1} + \delta\phi^{2}_{i_2+1,j_2} \\
\delta\phi^{1,n+1}_{i_1,j_1+1} = \delta\phi^{1,n}_{i_1,j_1+1} + \delta\phi^{2}_{i_2,j_2} & \delta\phi^{1,n+1}_{i_1+2,j_1+1} = \delta\phi^{1,n}_{i_1+2,j_1+1} + \delta\phi^{2}_{i_2+1,j_2} \\
\delta\phi^{1,n+1}_{i_1+1,j_1+1} = \delta\phi^{1,n}_{i_1+1,j_1+1} + \delta\phi^{2}_{i_2,j_2} & \delta\phi^{1,n+1}_{i_1+3,j_1+1} = \delta\phi^{1,n}_{i_1+3,j_1+1} + \delta\phi^{2}_{i_2+1,j_2}
\end{array} \tag{123}$$

where the prolongation is shown for eight fine-grid cells. This type of prolongation is called a piecewise constant interpolation or an injection, where the solution from the coarse grid is just added to those fine-grid values that are inside the coarse cell [11]. After the prolongation a first iteration is performed in AMG. Next, the fine grid problem can be smoothed again and the modified residual restricted to the second level, etc. Convergence can be monitored by taking the L_2-norm from the modified residual $||\omega^1||_2$.

do m = 2 to M
  GCA($A^{m-1} \rightarrow A^m$)
enddo
$\delta\phi^1 = 0$
do n = 1 to N
  LGS($A^1\delta\phi^1 = R^1$)
  do m = 2 to M
    RES($R^{m-1} \rightarrow R^m$)
    $\delta\phi^m = 0$
    LGS($A^m\delta\phi^m = R^m$)
  enddo
  do m = M-1 to 1
    PRO($\delta\phi^{m+1} \rightarrow \delta\phi^m$)
    LGS($A^m\delta\phi^m = R^m$)
  enddo
enddo

Let us take a more detailed look of the algorithm for M grid levels. Above is a loop structure describing the main algorithm. At the beginning the coarse grid coefficients are generated with the GCA routine for every grid level. Next the initial guess is set to zero for $\delta\phi^1$. After that the main iteration loop begins. First the error is smoothed on the finest level with the LGS routine. A new loop begins, where the grid levels are cycled from the second level to the coarsest one. In this loop the residuals are restricted with a RES routine, an initial guess is set to zero, and the error is smoothed. Initialization at every iteration is not mandatory, but it makes the algorithm faster. After the coarsest level the loop ends. To correct the values a new loop starts, where the levels are cycled from the second coarsest to the finest one. In this a solution is prolongated with a PRO routine and the corrected values can be smoothed again. Smoothing is optional in this loop, because it might not improve overall performance. This concludes the first iteration in the main loop. This type of iteration is called a V-cycle. The main loop ends after N iterations. Optionally, convergence can be monitored and the main loop can be ended if an acceptable level of error has been reached. A more accurate interpolation can also be used for

restriction and prolongation, but is not necessary in AMG.

5.2 Geometric Multigrid

For solving linear problems AMG is all that is needed to have an efficient solution. If the problem, as in almost all flow cases, is non-linear, AMG cannot reduce the time steps (or the amount of under-relaxation) needed for a steady-state solution. In non-linear problems a geometric multigrid (GMG) should be applied. With the GMG algorithm the problem is solved on multiple grid levels as with AMG. In the case of GMG the actual geometrical data is needed for grid generation. For every level, nodes, face areas, cell volumes etc. are created using fine-level data. Again, four fine grid cells are combined to form a coarse grid cell, which is the easiest way to proceed with the structured mesh. To get the benefit from GMG, coarser levels have to represent the overall physics of the problem. In many cases there is a minimum resolution that can contain the main features of the problem. If even coarser levels are nevertheless used, convergence is not improved and might even stall or diverge completely. More levels could possibly be applied if the grid were made denser where higher gradients existed. In that case cells should be stretched with moderation so that interpolation error would not ruin the convergence or the accuracy. In the figure below there is a general algorithm for solving a series of non-linear equations with a full approximation storage (FAS-GMG). Differences compared to the AMG-routine are scarce. At the beginning the geometry is generated for every grid level. This is comparable to the coarse grid generation in AMG. Next, variables $\vec{\Phi}^1$ are initialized on the finest grid level. In GMG an initial value can be other than zero. Also in GMG coarse-level variables are initialized by restricting the fine-level values. Subroutines SOLVEVARS, RESTRICTVARS and PROLONGATEVARS have in general the same purpose as LGS, RES and PRO in AMG, respectively. The second argument in SOLVEVARS is related to the solution of the equation system considered in this work and, therefore, might not be necessary in other problems.

do m = 2 to M
  GEOMETRY(m)
enddo
$\vec{\Phi}^1 = \vec{\Phi}^{initial}$
do n = 1 to N
  SOLVEVARS($\vec{\Phi}^1$,true,1)
  do m = 2 to M
    RESTRICTVARS($\vec{R}^{m-1} \rightarrow \vec{R}^m$, $\vec{\Phi}^{m-1} \rightarrow \vec{\Phi}^m$,m)
    SOLVEVARS($\vec{\Phi}^m$,true,m)
  enddo
  do m = M-1 to 1
    PROLONGATEVARS($\vec{\Phi}^{m+1} \rightarrow \vec{\Phi}^m$,m)
    SOLVEVARS($\vec{\Phi}^m$,false,m)
  enddo
enddo

The main differences between the AMG and GMG algorithms can be found in the treatment of the residual, the restriction and the prolongation. The residual is prolongated identically from the first level to the second in GMG using Eq. (122), where the modified residual is replaced by the explicit residual. If several iterations or post-smoothing are performed on the coarse levels, a forcing function is applied to the residual. On the second level, after the explicit residual has been calculated, the difference between the restricted residual $R^{2,*}$ from finest level and the new explicit residual R^2 is calculated as

$$\overline{R^2} = R^{2,*} - R^2 \tag{124}$$

which is the forcing function for the residual. The forcing function is updated only during the first iteration of pre-smoothing. Next the sum of the forcing function and the explicit residual is calculated as

$$\hat{R}^2 = R^2 + \overline{R^2} \tag{125}$$

which equals the restricted residual from the first level. But after consecutive iterations on the same level, this modified residual will differ

from the restricted residual. The modified residual is used in context with the linearized system. After the final iteration on the second level the explicit residual is calculated once more with the updated variables. The latest residual is then summed with the forcing function:

$$\begin{aligned} R^{3,*}_{i_3,j_3} =& R^2_{i_2,j_2} + R^2_{i_2+1,j_2} + R^2_{i_2,j_2+1} + R^2_{i_2+1,j_2+1} \\ & \overline{R^2}_{i_2,j_2} + \overline{R^2}_{i_2+1,j_2} + \overline{R^2}_{i_2,j_2+1} + \overline{R^2}_{i_2+1,j_2+1} \end{aligned} \tag{126}$$

which equals a restricted residual for the third level. The forcing function ensures that only the change in the explicit residual is added to the restricted residual from the previous level. After the prolongation from the third level the same forcing function is used during post-smoothing. This algorithm follows the method of Jameson [12]. In FAS variables are also restricted, unlike in AMG. Values are weighted with cell volume as follows:

$$(\phi^*\mathbf{V})^{m+1}_{i_{m+1},j_{m+1}} = (\phi\mathbf{V})^m_{i_m,j_m} + (\phi\mathbf{V})^m_{i_m+1,j_m} + (\phi\mathbf{V})^m_{i_m,j_m+1} + (\phi\mathbf{V})^m_{i_m+1,j_m+1} \tag{127}$$

where ϕ^* is important during prolongation [6]. To ensure a mesh-size-independent rate of convergence a condition

$$m_P + m_R > m_{eq} \tag{128}$$

should hold [10]. On the right-hand side m_{eq} is the highest order of differentiation in the equation to be solved. On the left the first and the second term are an interpolation order of a prolongation and restriction. The restriction described above has the order of $m_R = 1$. Since all equations to be solved have second-order partial differential terms, prolongation should be of the order of $m_P = 2$. Injection in AMG has the order of one. Therefore, a bilinear interpolation, which has $m_P = 2$, should be implemented in 2-D cases. Using the indexing system presented in Fig. 4 the bilinear interpolation can be calculated by weighting coarse grid values as follows:

$$\begin{aligned} \hat{\phi}^{m-1}_{i_{m-1},j_{m-1}} = \phi^{m-1}_{i_{m-1},j_{m-1}} &+ \frac{9}{16}(\phi - \phi^*)^m_{i_m,j_m} + \frac{3}{16}(\phi - \phi^*)^m_{i_m-1,j_m} \\ &+ \frac{3}{16}(\phi - \phi^*)^m_{i_m,j_m-1} + \frac{1}{16}(\phi - \phi^*)^m_{i_m-1,j_m-1} \end{aligned}$$

Here the difference of the calculated value ϕ^m and the restricted value $\phi^{m,*}$ is added after weighting to the fine-level value. There are four

different weightings inside the grid, depending on how close the neighboring coarse grid cell centers are. Corner fine-grid cell values are interpolated from three coarse cells. All weightings are presented in Appendix A4. The condition in Eq. (128) was confirmed in test calculations for the momentum equation. Injection for pressure was enough during the prolongation to ensure convergence, although faster convergence was achieved with bilinear interpolation. In the case of a compressible flow with a density-based solver, Siikonen [13], [14] has found that injection was sufficient for interpolation during prolongation for all variables. Therefore, at least in some cases in Eq. (128), equality is a sufficient condition for convergence. In Appendix A5 the subroutines SOLVEVARS, RESTRICTVARS and PROLONGATEVARS are presented for solving the equation system considered in this work. In SOLVEVARS the current grid level m is cycled L_m times. In the loop, boundary conditions are set for every variable in BC, and local time-steps are calculated in LOCALDT. If the calculation is at least on the second level and in the first iteration round, mass balances are calculated for every cell in MASSBALANCE. Next the forcing function is applied for the mass residual. Restricted values of velocities are used in the mass balances. It is important that the Rhie&Chow terms are not included in the balance at this stage. The Rhie&Chow damping is applied only on the finest level, and through the mass residual this damping is transported to the coarser levels. During this work it was noted that when the damping terms were applied on the coarser levels, the solution diverged. After the mass residual all dependent variables such as magnetic flux density B, electric current density J, eddy viscosity μ_T, turbulence production P and cross-diffusion terms E_k, E_ϵ are calculated in subroutine CALCULATECV. Since these variables are complementary they can be defined as functions of the independent variables (u, v, k, ϵ,A_z) and, therefore, do not need to be solved. Next, equations for the independent variables are linearized and solved in subroutine SOLVEVAR, except for pressure, which is solved in subroutine PC. Before describing subroutine SOLVEVAR in detail, Eq. (63) is rewritten into form

$$\rho\frac{\delta\phi_k}{\delta t}\mathbf{V} + F_k(\phi_k) = s_k \tag{129}$$

and further to

$$R_k = s_k - F_k(\phi_k) \tag{130}$$

where subindex k denotes a component of Eq. (63), s a source term, R a residual and F flux terms, which are functions of ϕ. In SOLVEVAR

the explicit residual is calculated for grid level m and equation k. The forcing function is calculated in the first iteration round during presmoothing and the explicit residual is modified with it. Flux terms are linearized in subroutine LINEARIZE, and the resulting linear system is solved with subroutine AMG. Finally, variable ϕ_k is updated with an under-relaxation of α_k. In subroutine PC mass balances are calculated and Rhie&Chow damping is added in them on the finest grid level. The mass residual is modified with the forcing function, and coefficients for the pressure-correction equation are calculated in PCECOEFFS. The pressure-correction equation is solved using AMG and, finally, the velocities and the pressure field are corrected. Intergrid operations are performed in RESTRICTVARS and PROLONGATEVARS. In the former the boundary conditions are updated first. Equations are cycled and the explicit residual is calculated for them. The residuals and the variables are then restricted to a coarser level in subroutines RESTRICT1 and RESTRICT2. Mass residuals are calculated in MASSBALANCE and damping terms are added for the finest-level residuals. Finally, mass residuals and pressure are restricted to the coarser level. Boundary conditions are again updated in PROLONGATEVARS and then variables are prolongated to the finer level.

6 Test Calculations

Test cases include a laminar cavity flow, a laminar magneto-hydrodynamic flow, a turbulent channel flow and geometric multigrid testing. All interpolations of the independent variables were done with the second-order upwind method. The code was written in C++. All calculations were done with Windows XP OS using a single core of the Intel Core 2 CPU 6700 2.66GHz processor.

6.1 Laminar Cavity Flow

The laminar cavity flow was calculated with Re=100 and Re=1000. A uniform 128×128 grid ($1\,\mathrm{m} \times 1\,\mathrm{m}$) was used in calculations. The top surface velocity was set to $1.0\,\mathrm{m/s}$ and viscosity to 0.01 (Re=100) and 0.001 Pa s (Re=1000). Contours for the u- and v-velocity components from Re = 100 case are presented in Fig. 5.

A comparison between cases Re=100 and Re=1000 is presented in Fig. 6. When the Reynolds number increases, boundary layers near the walls get thinner. Also the center of the main clockwise vortex moves

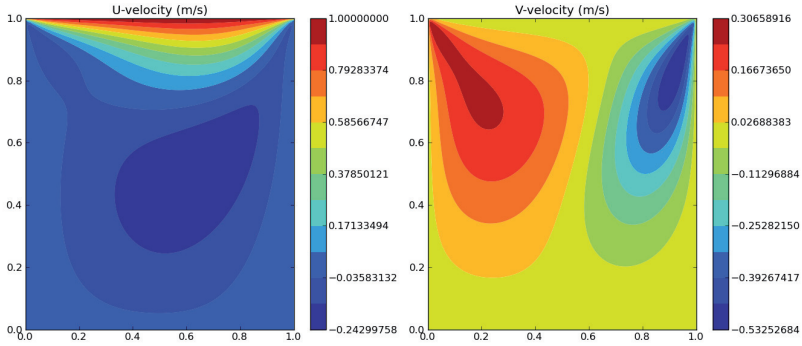

Figure 5: Cavity flow at Re=100. u-velocity on the left, v-velocity on the right.

closer to the center of the grid as Re increases. The velocity profiles are in good agreement with the calculations of Goyon [15].

6.2 Laminar Magnetohydrodynamic flow

The magnetohydrodynamic interaction is taken into account by coupling the momentum equations with the magnetic vector-potential equation. To reduce the magnetic field and the Lorentz force into the xy-plane the magnetic vector potential and the current density vector have only a z-component. Thus $\vec{A} = (0, 0, A_z\vec{k})$ and $\vec{J} = (0, 0, J_z\vec{k})$. A $1\,\mathrm{m} \times 1\,\mathrm{m}$ square cavity is chosen for the geometry. Current density $J_s\vec{k}$ is set as a source term for the vector potential equation. To produce a counter-clockwise rotation for the fluid the source term is set as a function of the x-coordinate as follows: $J_s = 10(0.5 - x)\,\mathrm{A/m^2}$. This sets the current density at the left boundary to $5\,\mathrm{A/m^2}$ and at the right boundary to $-5\,\mathrm{A/m^2}$. In the middle the source term changes linearly between the boundary values. Viscosity is set to $0.01\,\mathrm{Pa\,s}$ and density to $1.0\,\mathrm{kg/m^3}$. Magnetic permeability is $0.1\,\mathrm{H/m}$ and electrical conductivity $300\,\mathrm{S/m}$.

First, only the magnetic vector-potential equation is solved. The effect of the source term and flow on the vector potential and the other variables are compared later. The boundary conditions for the vector potential include a zero normal gradient for the left and the right boundaries as well as constant values 0.1 and 0.0 $\mathrm{Wb/m}$ for the bottom and the top boundaries. Constant values suggest that the magnetic

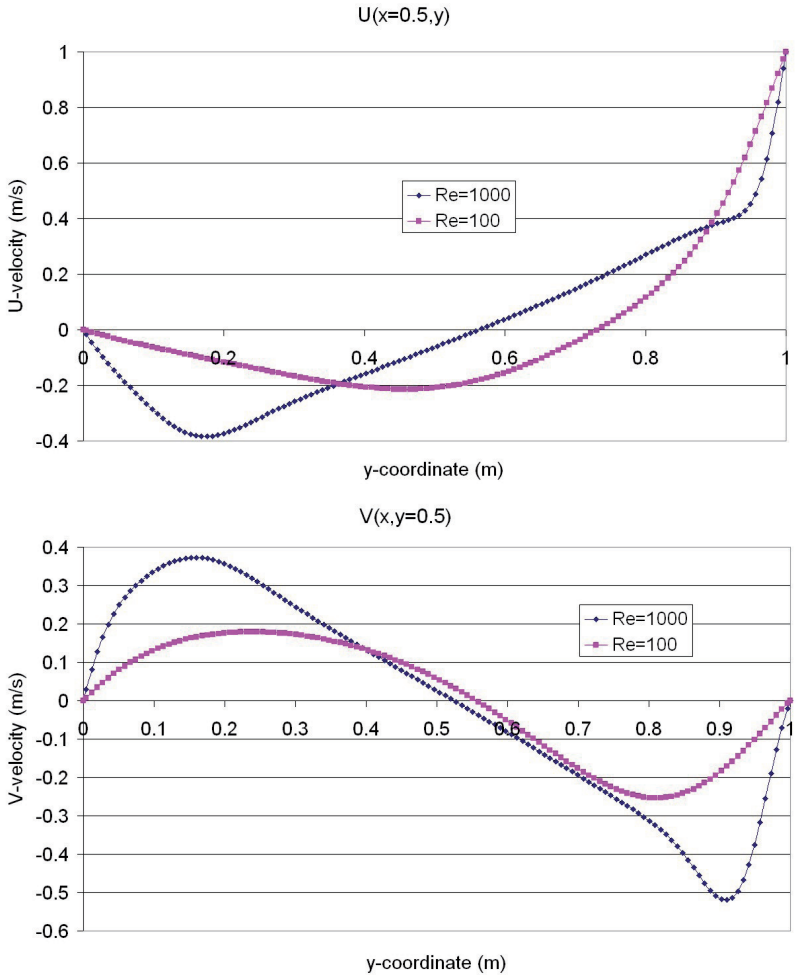

Figure 6: Comparison of u- and v-velocity profiles.

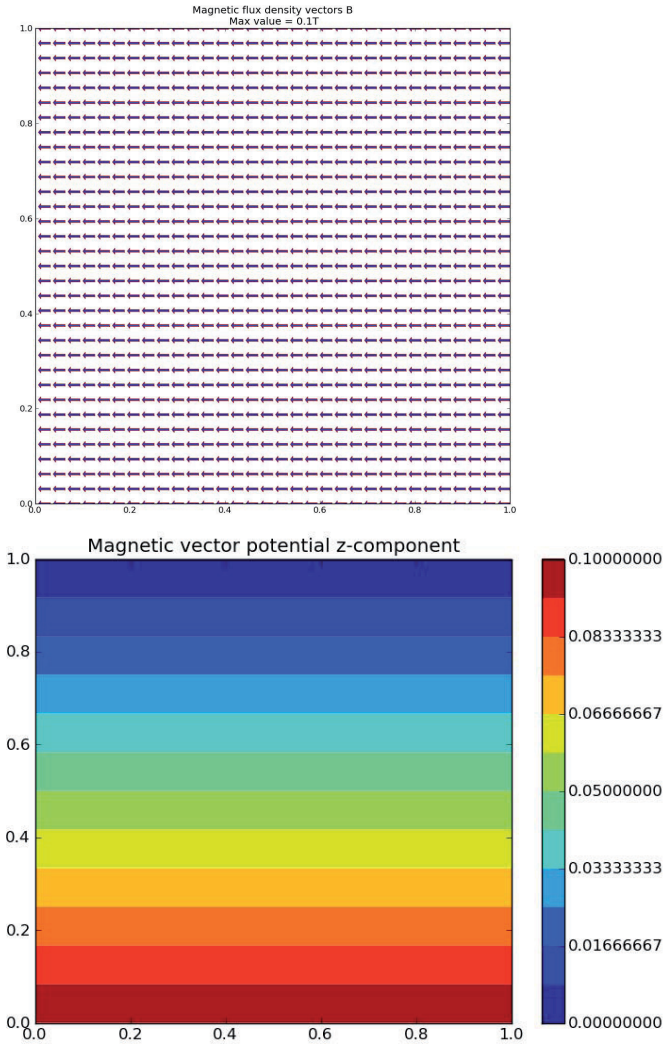

Figure 7: Magnetic flux density vectors and magnetic vector potential contours.

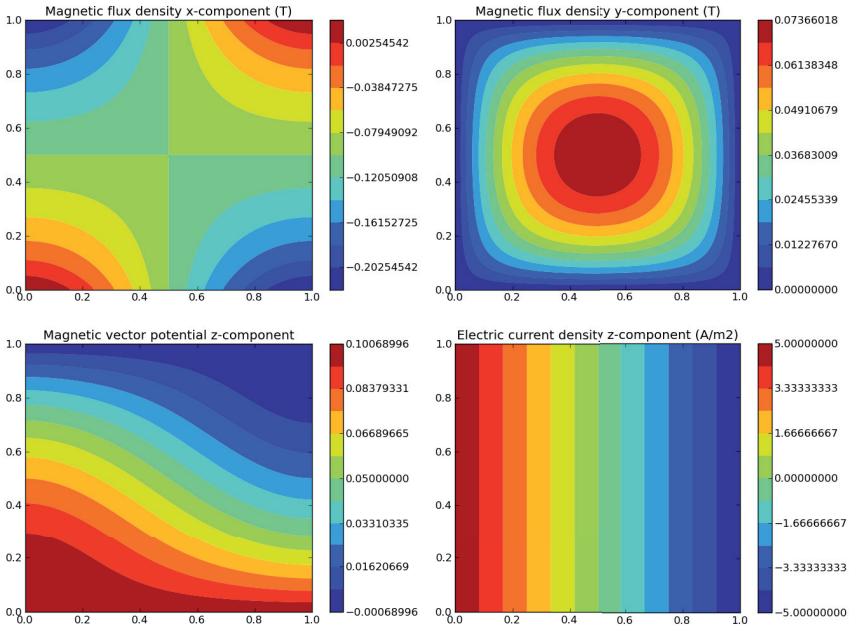

Figure 8: Contours for a magnetic vector potential, a flux density and a source current density.

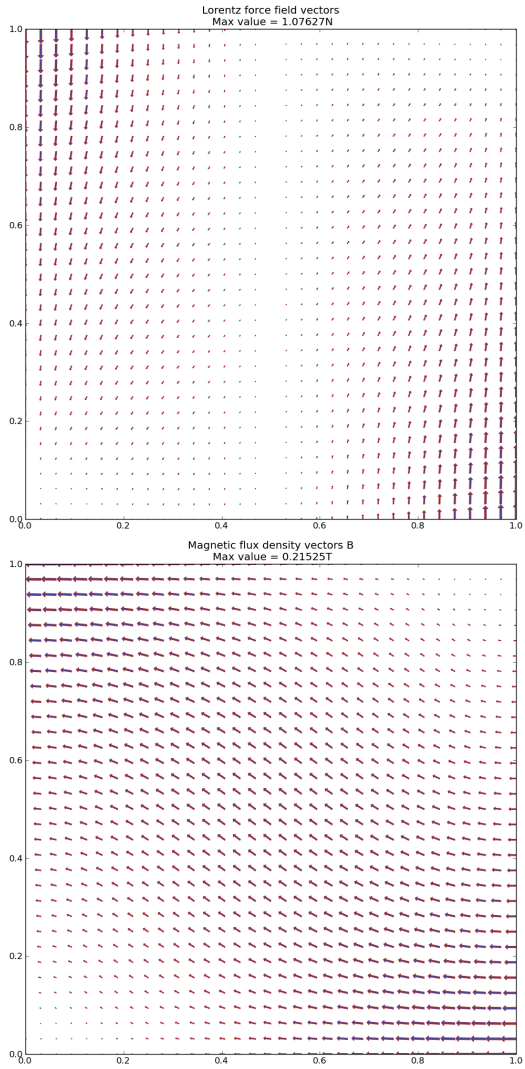

Figure 9: Magnetic flux density and Lorentz force vectors.

field is parallel to the boundary. The zero normal gradient holds the field perpendicular to the boundary. Contours for the magnetic vector potential equation and magnetic flux density vectors are presented in Fig. 7. This linear system generates a 0.1 T magnetic field pointing to the left. Next, the source term is added to the system. The magnetic vector potential is presented again in Fig. 8. The current density component and the x and y components for magnetic flux density can also be found there. High and low value areas of the vector potential are expanded due to the source current term. Current density component J_z is linear from the left to the right side, since it contains only the source current. x- and y-components of the magnetic flux density are of equal size in the middle, but in the upper left and lower right corners the x-component dominates, which can also be seen in Fig. 9. The Lorentz forces are strongest near the upper left and the lower right corners. In the middle no forces are present, because there the current density is also zero.

When the momentum equations are coupled with the vector potential, the Lorentz forces start to rotate the fluid and the flow vectors distort the vector potential and the current density. The impact of the flow on the electromagnetic properties can be seen in Fig. 10. The flow reduces the electric currents in the middle. It can be interpreted that some of the ohmic resistive power $(P = UJ)_z$ is transformed into viscous heating power of the flow field. Here, potential difference $U_z = J_s/\sigma$. The vector potential contour lines are pushed down on the left side and up on the right side by the flow. The highest values of the magnetic flux density are reduced and the field vectors are mainly pointing to the left (Fig. 11). The Lorentz forces are also weakened. The velocity vectors are presented in Fig. 12. The maximum velocity is approximately $0.1\,\mathrm{m/s}$ and the fluid is rotating in a counterclockwise direction.

Comparative calculations were made with COMSOL software. The solution strategy for the equation system in COMSOL is based on the finite-element method. All calculations were made with Windows XP version 3.5a. With COMSOL, different physical phenomena can be coupled by combining modules that are focused on specific isolated areas. Magnetohydrodynamic calculations were done by using "Incompressible Navier-Stokes" (IN-S) and "Magnetostatics" (MS) modules with identical material properties, boundary conditions and source current term as in the previous calculation. The former module solves velocity components u and v as well as pressure. The latter solves the z-component of the magnetic vector potential. Other necessary variables

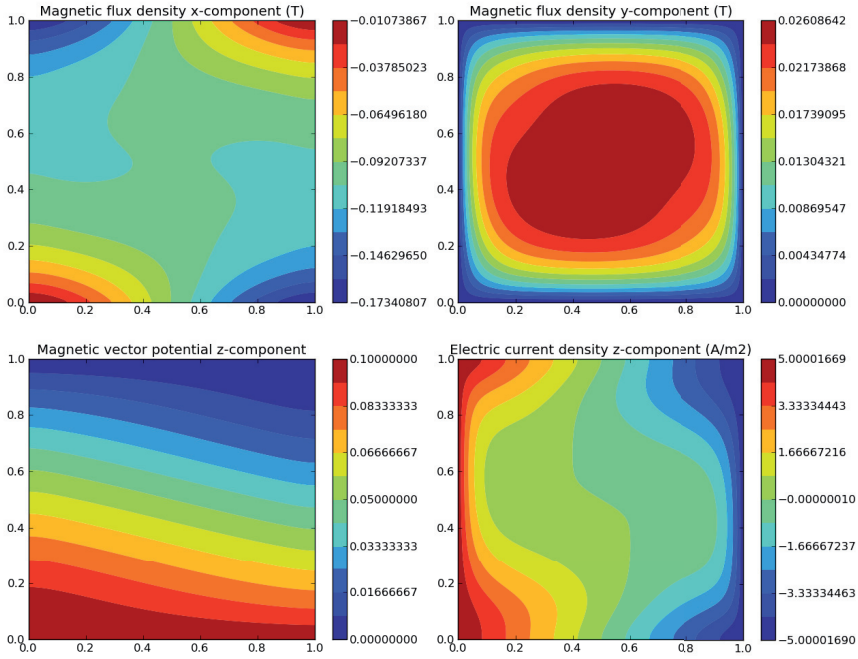

Figure 10: Contours for the magnetic vector potential, the flux density and the source current density, when the momentum equations have been coupled with the vector-potential equation.

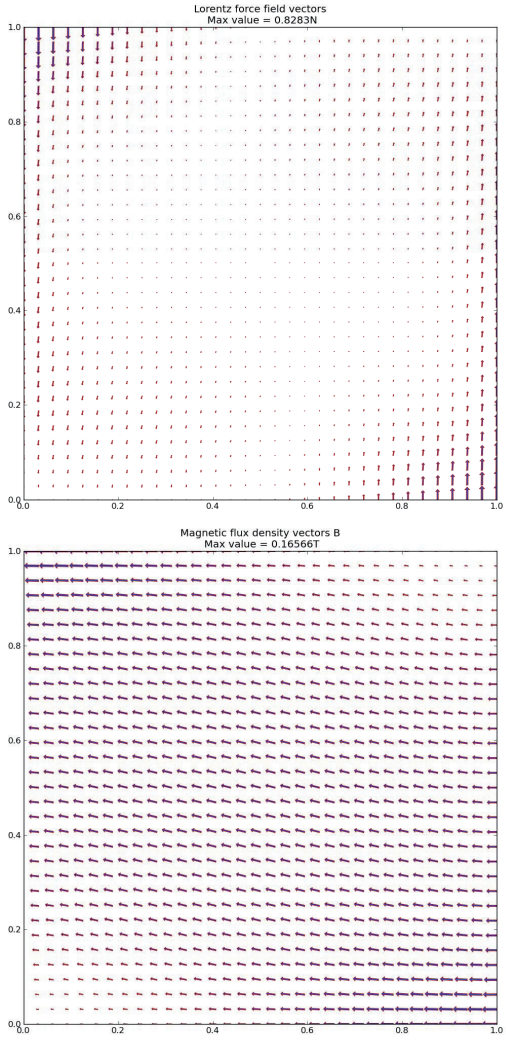

Figure 11: Magnetic flux density and Lorentz force vectors.

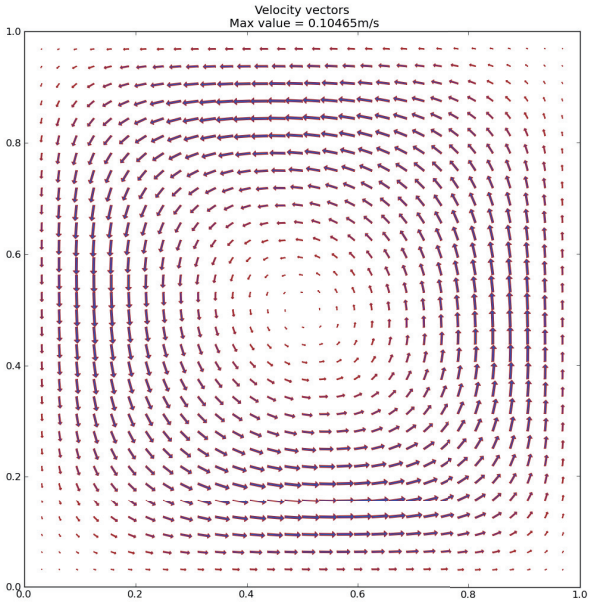

Figure 12: Velocity vectors.

can be calculated from these four unknowns and from the source current. Convection was coupled with the MS-module by connecting velocity components u and v into the MS-module's subdomain settings. The vector potential was coupled with the IN-S-module by setting the x- and y-components of the Lorentz force as volume forces in the subdomain. It is possible to add customized equations to COMSOL to be solved, but this was not necessary. Triangular elements were used in the mesh. The vector potential equation was interpolated with a quadratic Lagrangian polynomial. In the IN-S module the element type was set to Lagrange-P_2P_1. Calculations were made with 9,440 elements that consisted of 4,841 nodes. A higher resolution mesh (37,760 elements, 19,121 nodes) was also used to check the results. In the IN-S module the type of all boundaries was set as wall with a no-slip condition. Pressure was set to $0.0\,\mathrm{Pa}$ at coordinate point (0,0) by using a point constraint setting. In the MS-module the left and right boundary conditions were set to electric insulation. A magnetic potential was used for the top and bottom with 0.0 and 0.1 values, respectively. The default setting for the error norm is L_2-norm, which was used for the monitoring of the residuals. The error tolerance for the modules were set to $1 \cdot 10^{-6}$. In Fig. 13 the contours of the magnetic properties are presented in the same order as in Fig. 10. Also the contours of the velocity components are compared in Fig. 14, where the COMSOL results are above. It can be noted that the results are almost identical with each other.

6.3 Turbulent Channel Flow

The turbulence model was tested with a channel entrance and with a fully developed flow in $1.0\,\mathrm{m} \times 0.1\,\mathrm{m}$ (length$\times$width) geometry. The mesh was stretched bi-directionally in the y-direction with a ratio of 1.1 starting from the walls in the former case. Also in the entrance case the x-direction was stretched with a ratio of 1.05 starting from the inlet. The total cell count was 80×80 and the Reynolds-number was 60000. Inlet values were set as follows: $u_{in} = 30.0\,\mathrm{m/\,s}$, $k_{in} = 0.9\,\mathrm{m^2/\,s^2}$ $(= 0.001 * u_{in}^2)$, $\epsilon_{in} = 10.0\,\mathrm{m^2/\,s^3}$ and $\mu_T = 0.01\mu$. The Reynolds-number is defined for channel flow as:

$$Re = \frac{L/2V_{in}\rho}{\mu} \tag{131}$$

where L is the channel width. Contours of the k, ϵ, turbulence production and relative eddy viscosity (μ_T/μ) from the entrance flow are

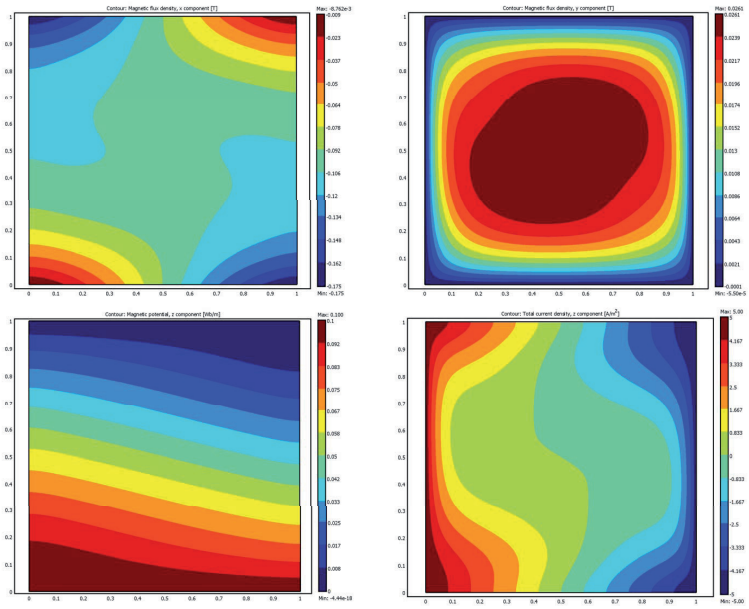

Figure 13: Contours for the electromagnetic properties from COMSOL-solution.

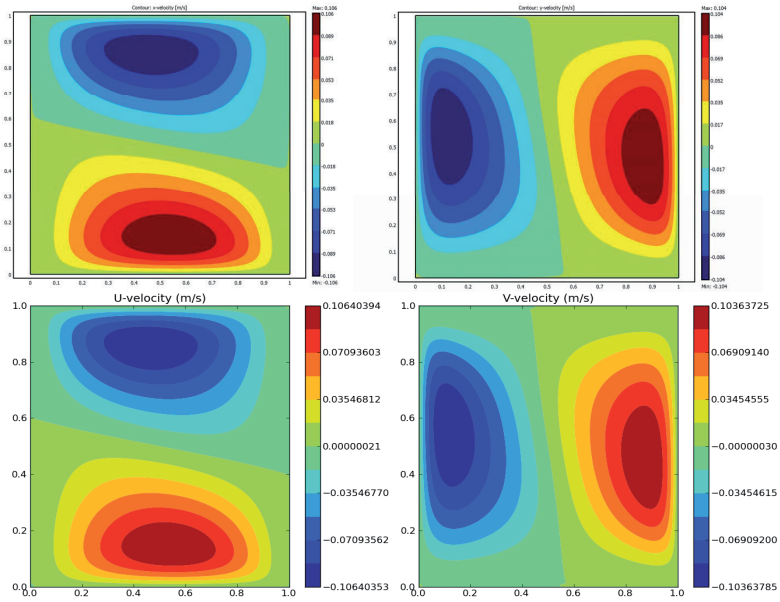

Figure 14: Comparison of the velocity contours. The u- and v-velocity components from the COMSOL-solution are in the upper left and upper right corners, respectively.

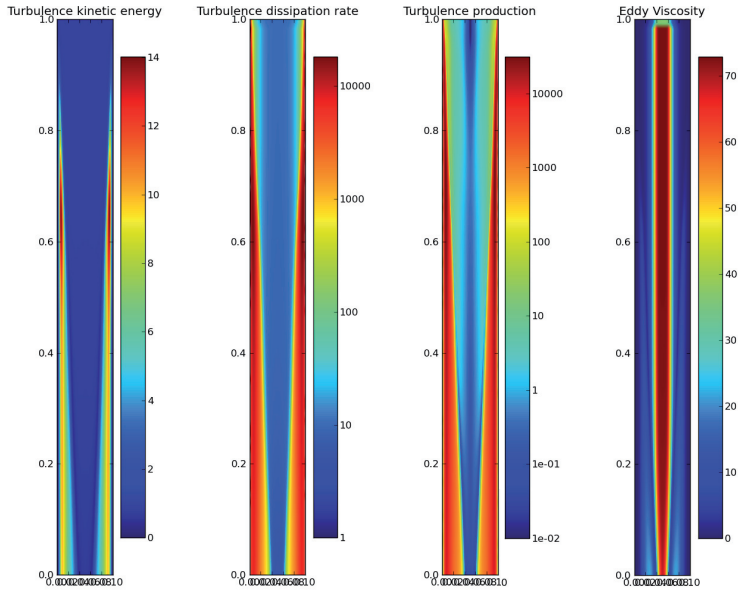

Figure 15: Turbulence contours of a channel entrance at Re=60000.

presented in Fig. 15.

A fully developed flow was accomplished by using periodic conditions for the inlet and outlet boundaries. At the beginning the inlet velocity value was set to $u_{in} = 30.0$ and the initial condition for velocity $u_i = 30.0$. Next, the total mass flow was integrated over the inlet area. This "inlet" mass flow value was set as the target mass flow value that was enforced during iterations. Calculations were done with and without the cross-diffusion terms for $Re = 10^4, 3 \cdot 10^4, 6 \cdot 10^4, 10^5$ by varying the viscosity. Results for the case of Re=60000 with the cross-diffusion terms are presented in Fig. 16 and Fig. 17. Some disturbance from the outlet can be seen in Fig. 17. Although small, the v-velocity component has a maximum ($\approx |0.3|$ mm$/s$) next to the outlet. In Fig. 18 developing velocity profiles from the entrance case were combined with the fully developed case at Re=60000. For all fully developed cases a friction

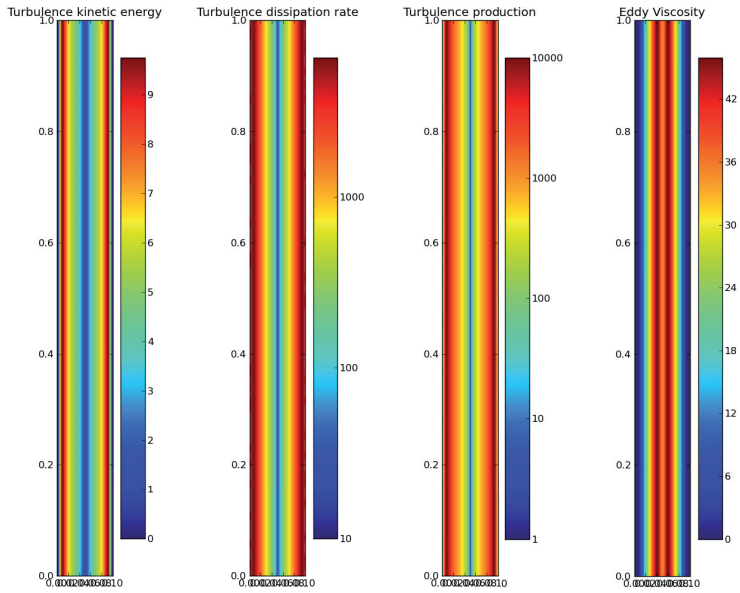

Figure 16: Turbulence contours of the fully developed channel flow at Re=60000.

factor Λ was calculated with:

$$\Lambda = \frac{8\tau_w}{\rho V_{in}^2} \tag{132}$$

where the shear force at the wall, τ_w, is calculated as:

$$\tau_w = \mu \left(\frac{\partial u}{\partial y} \right)_w \tag{133}$$

The friction factor is presented as a function of Re-number in Fig. 19 and results are compared with values obtained from literature [16]. It can be seen that at least in the range of $10^4 < Re < 10^5$, calculations without the cross-diffusion terms predict slightly higher friction factor at the wall than they should. This happens because the velocity derivative at the wall is too high. The effect of the cross-diffusion terms clearly lower the velocity derivative at the wall, but this correction is too high and the error is larger than in the cases without the terms. Different grids were used in these cases. To check that the grid is refined enough at the wall, a dimensionless height y^+ can be calculated with

$$y^+ = \frac{\rho u_\tau h}{\mu} \tag{134}$$

where h is the distance from the nearest cell center to the wall and u_τ is the wall friction velocity. The friction velocity can be calculated from:

$$u_\tau = \sqrt{\frac{\tau_w}{\rho}} \tag{135}$$

For these fully developed cases y^+ values ranged from 0.45 to 0.95. The grid is considered refined enough at the wall, when $y^+ \approx 1$. [16]

6.4 Multigrid Acceleration

Multigrid calculations were done with all of the previous cases. Boundary conditions and material parameters were identical with equivalent cases unless mentioned otherwise. The geometric multigrid algorithm was FAS, as described in Chapter 5.2. The algebraic multigrid was used for the pressure-correction, turbulence kinetic energy, turbulent dissipation rate and magnetic vector potential equation even for a single grid case. The maximum level count used in the AMG was five. For the pressure-correction equation AMG iterations were stopped after the relative L_2-norm dropped below 0.45 in the cavity flow case and

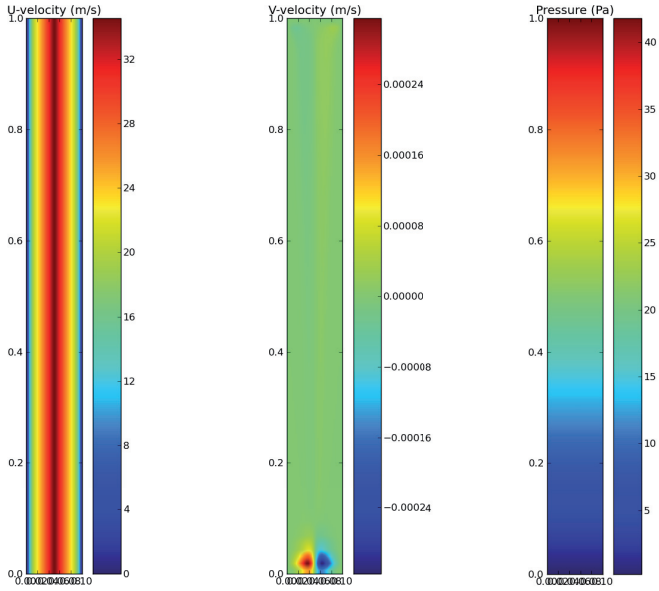

Figure 17: Velocity and pressure contours of the fully developed channel flow at Re=60000.

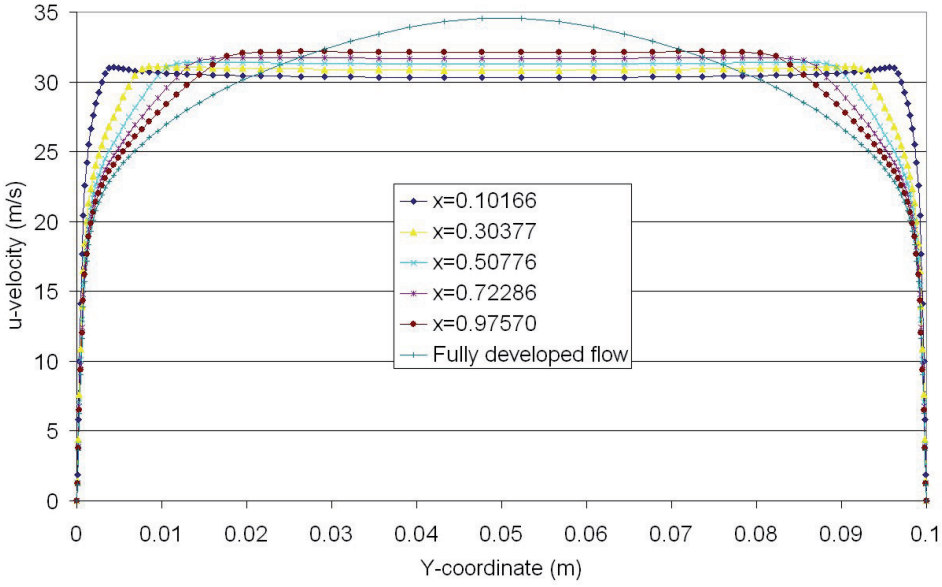

Figure 18: Developing u-velocity profiles for a channel flow at Re=60000.

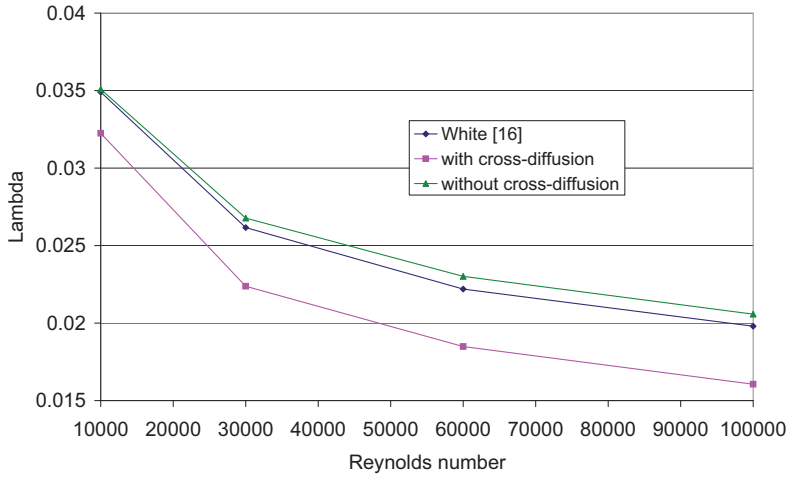

Figure 19: Friction factor as a function of Reynolds-number.

0.35 in the two other cases. For the other equations the ending criteria with AMG was 0.01. It should be noted that the fastest convergence might not happen with the least number of iterations. The total iteration count can be reduced by increasing the matrix inversions, but this usually decreases the overall performance. The linearized momentum equation was iterated with the LGS and without the AMG. Under-relaxation for pressure and velocities were kept as constants at 0.3 and 0.7, respectively. During the iterations all numerical parameters were kept as constants. One iteration (V-cycle) in GMG consisted of a single iteration on the densest level and two iterations per post- and pre-smoothing. Two iterations were made on the coarsest level. Timekeeping was realized with the help of a CPU-timer. This timer is affected by other processes but they were kept to a minimum during the iterations. Some of the calculations were repeated and only less than one percent fluctuation in the total computing time was experienced. Every step of the program code was included in the total time.

First GMG was tested with laminar cavity flow at Re = 1000. The iteration count in LGS was varied for every case to get the fastest convergence. Also the Courant number and C^{vis} were varied to achieve the best (fastest) performance. The Courant number was changed with 0.5 increments and C^{vis} with 0.05. The ending criterion for the outer iterations was the absolute L_2-norm of the pressure-correction p'. Calculations were automatically stopped after the residual dropped below $1 \cdot 10^{-16}$, which is practically the machine accuracy. The coarsest grid size was 8×8. Even coarser sizes were not improving the performance. The results of the testing are presented in Table 1. It is clear that the ratio between the single-grid time and the best GMG time rises, when mesh size increases. With a grid size of 32×32 the ratio is 1.6, but with 256×256 the ratio is 39.8. Interestingly the optimal cfl number decreases when the mesh size increases. Also GMG results need a smaller cfl than a single grid result, at least on the smaller grids. A strange anomaly can be seen in C^{vis} with sizes 128×128 and 256×256 using two GMG levels. This parameter is considerably smaller than with any other settings. This might make it hard to predict the optimal value in different flow problems. The pressure residuals for grid size 256×256 are presented in Fig. 20.

Next, GMG was tested with a turbulent flow in a channel entrance at Re=60000. This time the grid size was $1.0\,\mathrm{m} \times 0.05\,\mathrm{m}$ with 120×60 cells. A symmetry condition was used at the boundary parallel to the wall. The grid was stretched with a ratio of 1.05 in the y-direction and between 0.0 to $0.2\,\mathrm{m}$ in the x-direction. Evenly spaced grid points were

Table 1. Comparison of the geometric multigrid results.

Cavity flow

size	cells	GMG levels	time (s)	iterations	cfl	C^{vis}	LGS u,v
32X32	1024	1	5.1	996	5.5	1.25	3
		2	3.4	352	5	1.25	3
		3	3.1	234	5	1	3
64X64	4096	1	43.1	2115	5	1.1	3
		2	23.6	576	3.5	1	3
		3	12.3	232	3.5	1	3
		4	11.7	206	3.5	1.1	3
128X128	16384	1	526	5663	4	1.05	3
		2	256	1355	3	0.4	3
		3	77.5	354	4	1	3
		4	51.1	225	3.5	1	3
		5	47.9	212	3	1	3
256X256	65536	1	12330	27850	2.5	1	4
		2	4220	4654	2.5	0.7	4
		3	1440	991	2.5	1.1	4
		4	561	405	2.5	1.1	3
		5	362	277	3	1	3
		6	310	251	2.5	1	3

MHD

size	cells	GMG levels	relative time	iterations	cfl	C^{vis}
128X128	16384	1	21.3	8360	3.8	2.5
		2	5.6	1329	2.5	1.75
		3	2.9	495	0.905	0.85
		4	1.3	198	0.905	0.85
		5	1	156	0.905	0.85

Channel entrance

size	cells	GMG levels	relative time	iterations	cfl	C^{vis}
60x120	7200	1	6.5	14692	3	0.5
		2	1.0	1220	2	0.25
		3	1.5	1191	1.1	0.075

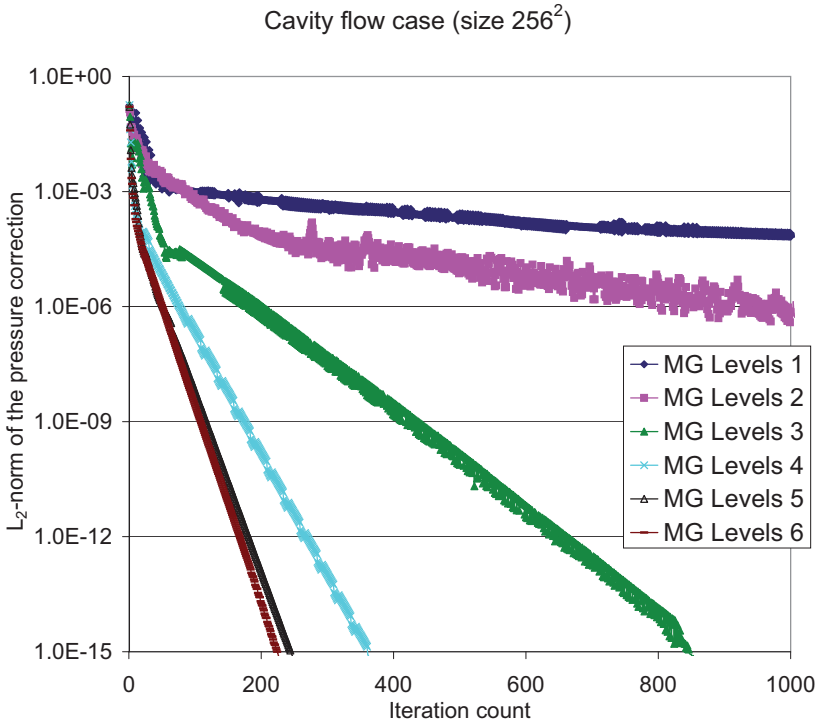

Figure 20: L_2-norm of pressure correction in the case of cavity flow of grid size 256×256.

used between 0.2 to $1.0\,\mathrm{m}$. The cross-diffusion terms were included in the calculations. It was noted during testing that convergence was ensured only if the cross-diffusion terms and eddy viscosity were restricted to the coarser grids. The solution time was registered after the relative residual of the turbulence kinetic energy dropped below $1 \cdot 10^{-13}$. The results are presented for one, two and three grid levels in Table 1. Solution times have been divided by the shortest time. For these three cases L_2-norms are presented for the turbulence kinetic energy residual and pressure correction in Fig. 21. Although the three-level case had converged with the fewest number of iterations (V-cycles), it was about 50% slower than the two-level case. The single-grid case was 6.5 times slower than the fastest MG case.

Finally, GMG was applied to the laminar magnetohydrodynamic case. With grid size 128×128 the solution was calculated using one, two, three, four and five grid levels. The solution time was registered after the relative residual of the magnetic vector potential dropped below $1 \cdot 10^{-13}$. A comparison of the solution times can be found again from Table 1. L_2-norms are presented for magnetic vector potential residual and pressure correction in Fig. 22 and Fig. 23, respectively. Again, a significant reduction in the solution time can be had with GMG. The best case is 21.3 times faster than the single-grid solution. With a cavity flow case of similar size the best case ratio is 11.0. This could suggest that, when an additional equation is coupled with the flow, the use of GMG can be even more beneficial. In the channel entrance and magnetohydrodynamic cases the values of the under-relaxation parameters cfl and C^{vis} had to be lowered considerably to ensure convergence when the number of grid levels was increased. Only in the turbulent case with three levels did this seem to have an adverse effect on performance.

7 Conclusions

In this study a computer program has been written that solves an incompressible turbulent and a laminar magnetohydrodynamic flow in a curvilinear two-dimensional structured grid. The program includes a Python code for pre- and post-processing as well as a C++ code for a solver. At the beginning the Python code creates an input file that contains information about mesh, boundary conditions, initial conditions and material properties. The solver program is started automatically via Python after the input file has been made. The solver reads the

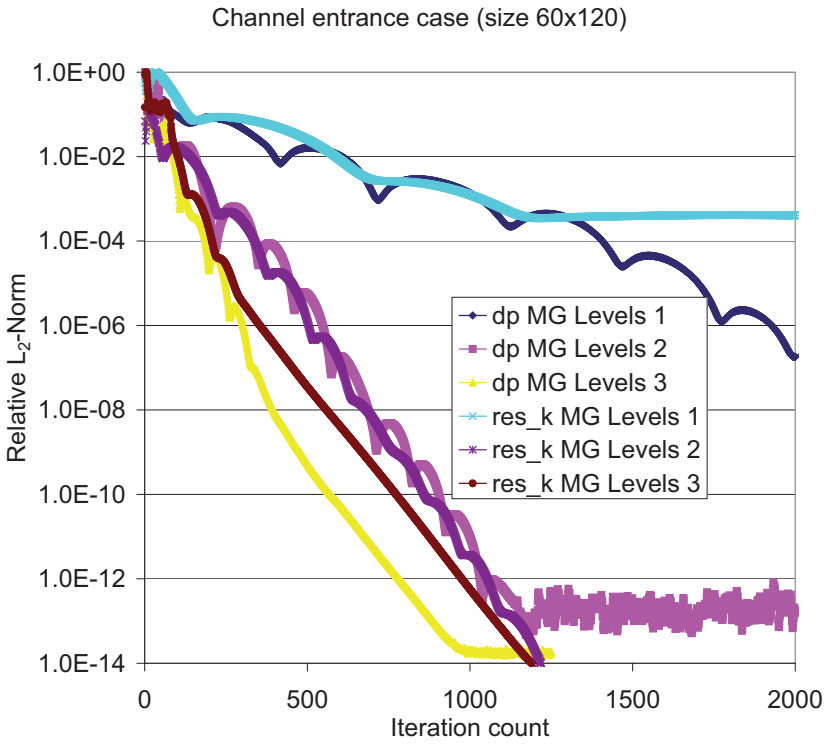

Figure 21: Relative L_2-norms of pressure correction and turbulence kinetic energy residual in the case of channel entrance flow of grid size 60×120.

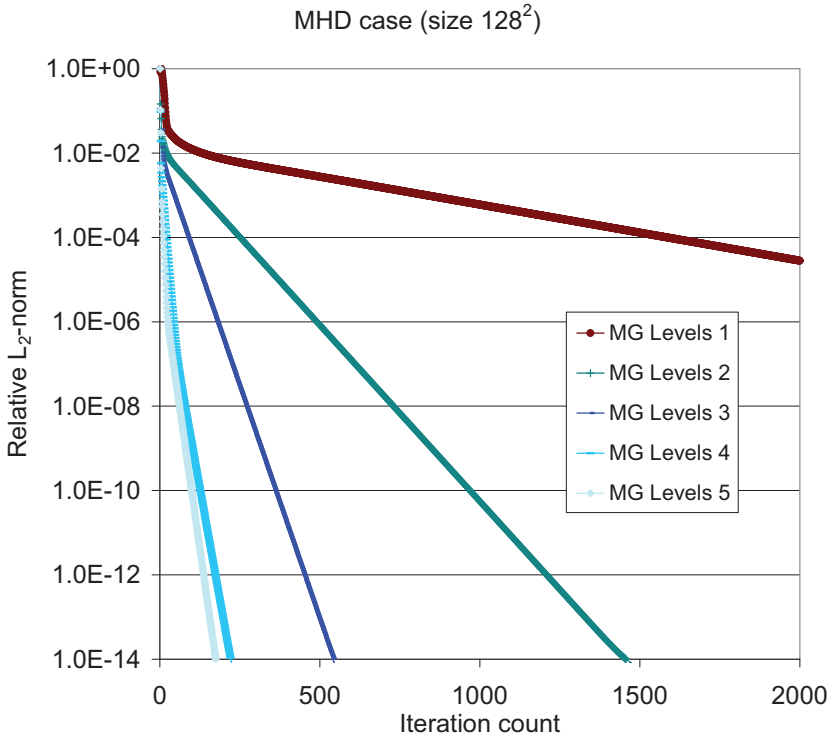

Figure 22: Relative L_2-norm of magnetic vector potential residual in the case of laminar magnetohydrodynamic flow of grid size 128×128.

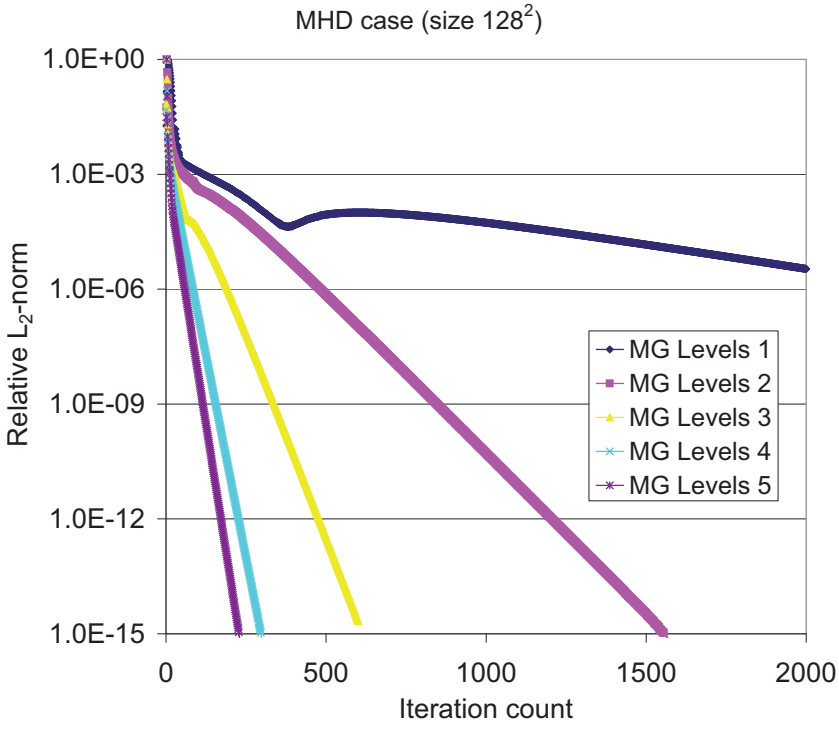

Figure 23: Relative L_2-norm of pressure correction in the case of laminar magnetohydrodynamic flow of grid size 128×128.

input file as well as a mesh file and initial values if these are necessary. Equations are iterated by the solver until conditions for finishing are met and solutions are then stored into files. Finally, the Python code reads automatically the solution data and generates contour and vector plots for the desired variables.

Continuity and momentum equations are solved with a pressure-correction algorithm and turbulence energy is modeled with the $k - \epsilon$ model using cross-diffusion terms. Magnetohydrodynamic flow is implemented by coupling a convection-diffusion equation for the magnetic vector potential with the momentum equation. All equations are solved using an implicit time-stepping algorithm that is accelerated with multigrid procedures. Linearized equations are solved with an algebraic multigrid and non-linear equations with a full approximation storage scheme.

The program was tested by solving several classical test cases. Laminar flow was tested for the cavity flow case and the results for the velocity profiles were identical with the literature [15]. A turbulent flow was modeled for a channel entrance at Re=60000. Friction factors were calculated and compared with values from literature [16] for a fully developed channel flow at $Re = 10^4 - 10^5$ with and without cross-diffusion terms. The cases without the terms had a smaller error in friction factor values. A magnetohydrodynamic flow was modeled in a cavity and the results were confirmed with commercial COMSOL software. The FAS-algorithm was tested with all of the cases mentioned above. Numerical parameters were varied to achieve the fastest solution time for every case. In all cases, GMG significantly decreased the time needed for a solution. In the case of cavity flow with a grid size of 256^2 the multigrid solution proved to be approximately forty times faster than a single-grid solution.

References

[1] Wilcox D.C., Turbulence Modeling for CFD, DCW Industries Inc., California, 1994, ISBN 0-9636051-0-0.

[2] Rahman M.M. and Siikonen T., Improved Low-Reynolds-Number k-E Model, American Institute of Aeronautics and Astronautics Inc, Vol. 38, Number 7, pp. 1298-1299, 2000.

[3] Kendall P.C. and Plumpton C., Magnetohydrodynamics with Hydrodynamics Volume 1, Pergamon Press Ltd, Oxford, 1964.

[4] Luomi J., Finite Element Methods for Electrical Machines, Lecture Notes for a Postgraduate Course in Electrical Machines, Chalmers University of Technology, Göteborg, 1993.

[5] Kulikovskiy A.G. and Lyubimov G.A., Magnetohydrodynamics, Addison-Wesley Publishing Company Inc, Massachusetts, 1965.

[6] Siikonen T., Laskennallisen virtausmekaniikan ja lämmönsiirron jatko-opintojakso, Helsinki University of Technology, Laboratory of Applied Thermodynamics, Finland, 2006 (in Finnish).

[7] MacCormack, R.W., A Numerical Method for Solving the Equations of Compressible Viscous Flows, AIAA Journal, Vol. 20, No. 9, pp. 1275-1281, 1981.

[8] Ferziger J.H. and Perić M., Computational Methods for Fluid Dynamics, Springer-Verlag Berlin Heidelberg New York, 3.rev.ed., 2002, ISBN 3-540-42074-6.

[9] Rhie, C.M. and Chow, W.L., A Numerical Study of the Turbulent Flow Past an Isolated Airfoil with Trailing Edge Separation., AIAA J., Vol. 21, pp. 1525-1532, 1983.

[10] Brandt A., Guide to Multigrid Development, Department of Applied Mathematics, The Weizmann Institute of Science, Rehovot, Israel.

[11] Wesseling P., An Introduction to Multigrid Methods, John Wiley & Sons Ltd., 1992, Corrected Reprint. Philadelphia: R.T. Edwards, Inc., 2004.

[12] Jameson, A., Yoon, S., Multigrid Solution of the Euler Equations Using Implicit Schemes, AIAA Journal, Vol. 24, No. 11, 1986.

[13] Siikonen T., A Three-Dimensional Multigrid Algorithm for the Euler and the Thin-Layer Navier-Stokes Equations, Helsinki University of Technology, Laboratory of Applied Thermodynamics, Finland, 1991.

[14] Siikonen T., An Application of Roe's Flux-Difference Splitting for k-ϵ Turbulence Model, Helsinki University of Technology, Laboratory of Applied Thermodynamics, Finland, 1994.

[15] Goyon O., High-Reynolds number solutions of Navier-Stokes equations using incremental unknowns, Computational methods in Applied Mechanics and Engineering, vol. 130, pp. 319-335, 1996.

[16] White F.M., Viscous Fluid Flow, McGraw-Hill International Edition, Third Edition, p. 428, 2006, ISBN 007-124493-X.

Appendix A1

A strain rate tensor S_{ij} in a matrix form is as follows:

$$S_{ij} = \frac{1}{2}\begin{pmatrix} 2\frac{\partial u}{\partial x} & \frac{\partial u}{\partial y}+\frac{\partial v}{\partial x} & \frac{\partial u}{\partial z}+\frac{\partial w}{\partial x} \\ \frac{\partial v}{\partial x}+\frac{\partial u}{\partial y} & 2\frac{\partial v}{\partial y} & \frac{\partial v}{\partial z}+\frac{\partial w}{\partial y} \\ \frac{\partial w}{\partial x}+\frac{\partial u}{\partial z} & \frac{\partial w}{\partial y}+\frac{\partial v}{\partial z} & 2\frac{\partial w}{\partial z} \end{pmatrix} \tag{136}$$

Reynolds stress tensor is presented in a matrix form using the Boussinesq approximation:

$$\tau_{ij} = \begin{pmatrix} 2\mu_T\frac{\partial u}{\partial x}-\frac{2}{3}\rho k & \mu_T\left(\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\right) & \mu_T\left(\frac{\partial u}{\partial z}+\frac{\partial w}{\partial x}\right) \\ \mu_T\left(\frac{\partial v}{\partial x}+\frac{\partial u}{\partial y}\right) & 2\mu_T\frac{\partial v}{\partial y}-\frac{2}{3}\rho k & \mu_T\left(\frac{\partial v}{\partial z}+\frac{\partial w}{\partial y}\right) \\ \mu_T\left(\frac{\partial w}{\partial x}+\frac{\partial u}{\partial z}\right) & \mu_T\left(\frac{\partial w}{\partial y}+\frac{\partial v}{\partial z}\right) & 2\mu_T\frac{\partial w}{\partial z}-\frac{2}{3}\rho k \end{pmatrix} \tag{137}$$

Turbulent production term $P = \tau_{ij}\cdot\partial u_i/\partial x_j$ can be calculated as a product of the tensors, resulting as a scalar:

$$\begin{aligned} \tau_{ij}\frac{\partial U_i}{\partial x_j} = {} & \frac{\partial u}{\partial x}\left(2\mu_T\frac{\partial u}{\partial x}-\frac{2}{3}\rho k\right)+\mu_T\frac{\partial u}{\partial y}\left(\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\right)+\mu_T\frac{\partial u}{\partial z}\left(\frac{\partial u}{\partial z}+\frac{\partial w}{\partial x}\right) \\ & +\mu_T\frac{\partial v}{\partial x}\left(\frac{\partial v}{\partial x}+\frac{\partial u}{\partial y}\right)+\frac{\partial v}{\partial y}\left(2\mu_T\frac{\partial v}{\partial y}-\frac{2}{3}\rho k\right)+\mu_T\frac{\partial v}{\partial z}\left(\frac{\partial v}{\partial z}+\frac{\partial w}{\partial y}\right) \\ & +\mu_T\frac{\partial w}{\partial x}\left(\frac{\partial w}{\partial x}+\frac{\partial u}{\partial z}\right)+\mu_T\frac{\partial w}{\partial y}\left(\frac{\partial w}{\partial y}+\frac{\partial v}{\partial z}\right)+\frac{\partial w}{\partial z}\left(2\mu_T\frac{\partial w}{\partial z}-\frac{2}{3}\rho k\right) \end{aligned} \tag{138}$$

which can be rearranged to

$$\begin{aligned} \tau_{ij}\frac{\partial U_i}{\partial x_j} = {} & 2\mu_T\left[\left(\frac{\partial u}{\partial x}\right)^2+\left(\frac{\partial v}{\partial y}\right)^2+\left(\frac{\partial w}{\partial z}\right)^2\right] \\ & +\mu_T\left[\left(\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\right)^2+\left(\frac{\partial u}{\partial z}+\frac{\partial w}{\partial x}\right)^2+\left(\frac{\partial v}{\partial z}+\frac{\partial w}{\partial y}\right)^2\right] \\ & -\frac{2}{3}\rho k\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}+\frac{\partial w}{\partial z}\right) \end{aligned} \tag{139}$$

where the last term on the right side can be omitted in the incompressible case, since it includes the continuity equation in parentheses and thus equals zero. In the 2D incompressible case the turbulence production term simplifies to:

$$\tau_{ij}\frac{\partial U_i}{\partial x_j} = 2\mu_T\left[\left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial y}\right)^2\right] + \mu_T\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right)^2$$

Appendix A2

Curl of $\vec{A}$:

$$\begin{aligned}\nabla \times \vec{A} &= \begin{vmatrix} \vec{i} & \vec{j} & \vec{k} \\ \dfrac{\partial}{\partial x} & \dfrac{\partial}{\partial y} & \dfrac{\partial}{\partial z} \\ A_x & A_y & A_z \end{vmatrix} \\ &= \left(\frac{\partial A_z}{\partial y} - \frac{\partial A_y}{\partial z}\right)\vec{i} - \left(\frac{\partial A_z}{\partial x} - \frac{\partial A_x}{\partial z}\right)\vec{j} + \left(\frac{\partial A_y}{\partial x} - \frac{\partial A_x}{\partial y}\right)\vec{k}\end{aligned} \tag{140}$$

Curl of $\nabla \times \vec{A}$:

$$\begin{aligned}\nabla \times \nabla \times \vec{A} &= \begin{vmatrix} \vec{i} & \vec{j} & \vec{k} \\ \dfrac{\partial}{\partial x} & \dfrac{\partial}{\partial y} & \dfrac{\partial}{\partial z} \\ \dfrac{\partial A_z}{\partial y} - \dfrac{\partial A_y}{\partial z} & \dfrac{\partial A_x}{\partial z} - \dfrac{\partial A_z}{\partial x} & \dfrac{\partial A_y}{\partial x} - \dfrac{\partial A_x}{\partial y} \end{vmatrix} \\ &= \left[\frac{\partial}{\partial y}\left(\frac{\partial A_y}{\partial x} - \frac{\partial A_x}{\partial y}\right) - \frac{\partial}{\partial z}\left(\frac{\partial A_x}{\partial z} - \frac{\partial A_z}{\partial x}\right)\right]\vec{i} \\ &- \left[\frac{\partial}{\partial x}\left(\frac{\partial A_y}{\partial x} - \frac{\partial A_x}{\partial y}\right) - \frac{\partial}{\partial z}\left(\frac{\partial A_z}{\partial y} - \frac{\partial A_y}{\partial z}\right)\right]\vec{j} \\ &+ \left[\frac{\partial}{\partial x}\left(\frac{\partial A_x}{\partial z} - \frac{\partial A_z}{\partial x}\right) - \frac{\partial}{\partial y}\left(\frac{\partial A_z}{\partial y} - \frac{\partial A_y}{\partial z}\right)\right]\vec{k}\end{aligned} \tag{141}$$

Next terms $\frac{\partial^2 A_x}{\partial x^2}$, $-\frac{\partial^2 A_x}{\partial x^2}$ are added to the $\vec{i}$-component, terms $\frac{\partial^2 A_y}{\partial y^2}$, $-\frac{\partial^2 A_y}{\partial y^2}$ to the $\vec{j}$-component and terms $\frac{\partial^2 A_z}{\partial z^2}$, $-\frac{\partial^2 A_z}{\partial z^2}$ to the $\vec{k}$-component:

$$\begin{aligned}
&\nabla \times \nabla \times \vec{A} = \\
&\left[\frac{\partial}{\partial x}\left(\frac{\partial A_x}{\partial x}\right) + \frac{\partial}{\partial x}\left(\frac{\partial A_y}{\partial y}\right) + \frac{\partial}{\partial x}\left(\frac{\partial A_z}{\partial z}\right) - \frac{\partial}{\partial x}\left(\frac{\partial A_x}{\partial x}\right) - \frac{\partial}{\partial y}\left(\frac{\partial A_x}{\partial y}\right) - \frac{\partial}{\partial z}\left(\frac{\partial A_x}{\partial z}\right)\right]\vec{i} \\
&+ \left[\frac{\partial}{\partial y}\left(\frac{\partial A_x}{\partial x}\right) + \frac{\partial}{\partial y}\left(\frac{\partial A_y}{\partial y}\right) + \frac{\partial}{\partial y}\left(\frac{\partial A_z}{\partial z}\right) - \frac{\partial}{\partial x}\left(\frac{\partial A_y}{\partial x}\right) - \frac{\partial}{\partial y}\left(\frac{\partial A_y}{\partial y}\right) - \frac{\partial}{\partial z}\left(\frac{\partial A_y}{\partial z}\right)\right]\vec{j} \\
&+ \left[\frac{\partial}{\partial z}\left(\frac{\partial A_x}{\partial x}\right) + \frac{\partial}{\partial z}\left(\frac{\partial A_y}{\partial y}\right) + \frac{\partial}{\partial z}\left(\frac{\partial A_z}{\partial z}\right) - \frac{\partial}{\partial x}\left(\frac{\partial A_z}{\partial x}\right) - \frac{\partial}{\partial y}\left(\frac{\partial A_z}{\partial y}\right) - \frac{\partial}{\partial z}\left(\frac{\partial A_z}{\partial z}\right)\right]\vec{k}
\end{aligned} \tag{142}$$

Terms can be rearranged to:

$$\begin{aligned}
\nabla \times \nabla \times \vec{A} = &\left[\frac{\partial}{\partial x}\left(\nabla \cdot \vec{A}\right) - \nabla \cdot (\nabla A_x)\right]\vec{i} \\
&+ \left[\frac{\partial}{\partial y}\left(\nabla \cdot \vec{A}\right) - \nabla \cdot (\nabla A_y)\right]\vec{j} \\
&+ \left[\frac{\partial}{\partial z}\left(\nabla \cdot \vec{A}\right) - \nabla \cdot (\nabla A_z)\right]\vec{k}
\end{aligned} \tag{143}$$

and further to:

$$\nabla \times \nabla \times \vec{A} = \nabla\left(\nabla \cdot \vec{A}\right) - \nabla^2 \vec{A} \tag{144}$$

Appendix A3

Flux term linearizations on the west, east, south and north faces respectively:

$$
\begin{aligned}
\frac{\partial F(\phi_1)^n_{i-1/2}}{\partial \phi_1}\delta\phi_1 &= \left[\max(0,\vec{V}\cdot\vec{n})\rho S + \left(\frac{S\lambda_1}{\Delta\xi}\right)\right]_{i-1/2}\delta\phi_{1,i-1} \\
&\quad - \left[\max(0,-\vec{V}\cdot\vec{n})\rho S + \left(\frac{S\lambda_1}{\Delta\xi}\right)\right]_{i-1/2}\delta\phi_{1,i} \\
\frac{\partial F(\phi_1)^n_{i+1/2}}{\partial \phi_1}\delta\phi_1 &= \left[\max(0,\vec{V}\cdot\vec{n})\rho S + \left(\frac{S\lambda_1}{\Delta\xi}\right)\right]_{i+1/2}\delta\phi_{1,i} \\
&\quad - \left[\max(0,-\vec{V}\cdot\vec{n})\rho S + \left(\frac{S\lambda_1}{\Delta\xi}\right)\right]_{i+1/2}\delta\phi_{1,i+1} \\
\frac{\partial F(\phi_1)^n_{j-1/2}}{\partial \phi_1}\delta\phi_1 &= \left[\max(0,\vec{V}\cdot\vec{n})\rho S + \left(\frac{S\lambda_1}{\Delta\eta}\right)\right]_{j-1/2}\delta\phi_{1,j-1} \\
&\quad - \left[\max(0,-\vec{V}\cdot\vec{n})\rho S + \left(\frac{S\lambda_1}{\Delta\eta}\right)\right]_{j-1/2}\delta\phi_{1,j} \\
\frac{\partial F(\phi_1)^n_{j+1/2}}{\partial \phi_1}\delta\phi_1 &= \left[\max(0,\vec{V}\cdot\vec{n})\rho S + \left(\frac{S\lambda_1}{\Delta\eta}\right)\right]_{j+1/2}\delta\phi_{1,j} \\
&\quad - \left[\max(0,-\vec{V}\cdot\vec{n})\rho S + \left(\frac{S\lambda_1}{\Delta\eta}\right)\right]_{j+1/2}\delta\phi_{1,j+1}
\end{aligned}
\tag{145}
$$

Since some of the source terms consist of variables that are not part of the local ϕ-component, their derivation with respect to ϕ would produce zero. Because there exists a strong coupling between the equations, the effect of the other variables have to be damped. Damping is especially important during the first few iterations, when a solution contains large oscillations, which induce large gradients. Damping is implemented so that the value of coefficient A_p increases. If the terms have negative values, absolute values are forced. The Lorentz force terms are linearized just by taking an absolute value from them. A source term for the k equation is linearized as follows: From the eddy viscosity equation ϵ is solved without taking the wall correction and damping function f_μ into account. This equation is substituted into the source term and derived with respect to k:

$$
\frac{\partial \psi_k}{\partial k} = \frac{\partial}{\partial k}\left(P - \frac{C_\mu \rho^2 k^2}{\mu_T} + E_k\right) = \frac{P}{k_c} + \frac{2C_\mu \rho^2 k}{\mu_T} \tag{146}
$$

where the derivation of E_k is set to zero and an absolute value is taken from the second term on the right. It is assumed that the production and the dissipation terms dominate over the cross-diffusion term. The purpose of k_c is to decrease the damping effect of the production term. Value $k_c = 1.0$ was used in the test calculations. Finally, the eddy viscosity is substituted into the linearized equation.

$$\frac{\partial \psi_k}{\partial k} = \frac{P}{k_c} + \frac{2\rho\tilde{\epsilon}}{k} \tag{147}$$

In the linearization of the source term in the equation of the $\tilde{\epsilon}$ the cross-diffusion term and the damping term are assumed to be small and are neglected. The derivation of $\psi_{\tilde{\epsilon}}$ is approximated as

$$\frac{\partial \psi_{\tilde{\epsilon}}}{\partial \tilde{\epsilon}} = \frac{C_{\epsilon 2}\rho}{T_t} + \frac{C_{\epsilon 1}P}{T_t \epsilon_c} \tag{148}$$

where ϵ_c (=300 in calculations) decreases the effect of turbulent production P. Finally, in the vector potential equation the source term linearization was not necessary, since source ψ_{A_z} was not affected by any other equation. The linearization of vector Ψ is presented here:

$$\frac{\partial \vec{\Psi}}{\partial t} = -\begin{pmatrix} |J_z B_y| \\ |J_z B_x| \\ \dfrac{2\rho\epsilon}{k} + \dfrac{P}{k_c} \\ \dfrac{C_{\epsilon 2}\rho}{T_t} + \dfrac{C_{\epsilon 1}P}{T_t \epsilon_c} \\ 0 \end{pmatrix} \tag{149}$$

Appendix A4

Weighting functions are presented for bilinear prolongation in locations (i,j), $(i+1,j)$, $(i,j+1)$ and $(i+1,j+1)$:

$$\begin{aligned}
\hat{\phi}^{m-1}_{i_{m-1},j_{m-1}} &= \phi^{m-1}_{i_{m-1},j_{m-1}} && + \frac{9}{16}(\phi-\phi^*)^m_{i_m,j_m} + \frac{3}{16}(\phi-\phi^*)^m_{i_m-1,j_m} \\
& && + \frac{3}{16}(\phi-\phi^*)^m_{i_m,j_m-1} + \frac{1}{16}(\phi-\phi^*)^m_{i_m-1,j_m-1} \\
\hat{\phi}^{m-1}_{i_{m-1}+1,j_{m-1}} &= \phi^{m-1}_{i_{m-1}+1,j_{m-1}} && + \frac{9}{16}(\phi-\phi^*)^m_{i_m,j_m} + \frac{3}{16}(\phi-\phi^*)^m_{i_m+1,j_m} \\
& && + \frac{3}{16}(\phi-\phi^*)^m_{i_m,j_m-1} + \frac{1}{16}(\phi-\phi^*)^m_{i_m+1,j_m-1} \\
\hat{\phi}^{m-1}_{i_{m-1},j_{m-1}+1} &= \phi^{m-1}_{i_{m-1},j_{m-1}+1} && + \frac{9}{16}(\phi-\phi^*)^m_{i_m,j_m} + \frac{3}{16}(\phi-\phi^*)^m_{i_m-1,j_m} \\
& && + \frac{3}{16}(\phi-\phi^*)^m_{i_m,j_m+1} + \frac{1}{16}(\phi-\phi^*)^m_{i_m-1,j_m+1} \\
\hat{\phi}^{m-1}_{i_{m-1}+1,j_{m-1}+1} &= \phi^{m-1}_{i_{m-1}+1,j_{m-1}+1} && + \frac{9}{16}(\phi-\phi^*)^m_{i_m,j_m} + \frac{3}{16}(\phi-\phi^*)^m_{i_m+1,j_m} \\
& && + \frac{3}{16}(\phi-\phi^*)^m_{i_m,j_m+1} + \frac{1}{16}(\phi-\phi^*)^m_{i_m+1,j_m+1}
\end{aligned}$$

For the corner nodes, the values are interpolated using three cells from the coarse grid. Two ghost cells are present at each boundary. Therefore the index of the first cell inside the domain is 3. Total cell counts including the ghost cells are for the i- and j-directions I and J respectively. The index numbers of the last cells inside the domain are $I-2$ and $J-2$. Weighting equations for the corner nodes are as follows:

$$
\begin{aligned}
\hat{\phi}^{m-1}_{3_{m-1},3_{m-1}} = \phi^{m-1}_{3_{m-1},3_{m-1}} \quad & + \frac{8}{16}(\phi-\phi^*)^m_{3_m,3_m} + \frac{4}{16}(\phi-\phi^*)^m_{2_m,3_m} \\
& + \frac{4}{16}(\phi-\phi^*)^m_{3_m,2_m} \\
\hat{\phi}^{m-1}_{I_{m-1}-2,3_{m-1}} = \phi^{m-1}_{I_{m-1}-2,3_{m-1}} \quad & + \frac{8}{16}(\phi-\phi^*)^m_{I_m-2,3_m} + \frac{4}{16}(\phi-\phi^*)^m_{I_m-1,3_m} \\
& + \frac{4}{16}(\phi-\phi^*)^m_{I_m-2,2_m} \\
\hat{\phi}^{m-1}_{3_{m-1},J_{m-1}-2} = \phi^{m-1}_{3_{m-1},J_{m-1}-2} \quad & + \frac{8}{16}(\phi-\phi^*)^m_{3_m,J_m-2} + \frac{4}{16}(\phi-\phi^*)^m_{3_m,J_m-1} \\
& + \frac{4}{16}(\phi-\phi^*)^m_{2_m,J_m-2} \\
\hat{\phi}^{m-1}_{I_{m-1}-2,J_{m-1}-2} = \phi^{m-1}_{I_{m-1}-2,J_{m-1}-2} \quad & + \frac{8}{16}(\phi-\phi^*)^m_{I_m-2,J_m-2} + \frac{4}{16}(\phi-\phi^*)^m_{I_m-2,J_m-1} \\
& + \frac{4}{16}(\phi-\phi^*)^m_{I_m-1,J_m-2}
\end{aligned}
$$

The interpolation error for these methods are of the order $O(\Delta x^2_{m-1}, \Delta y^2_{m-1})$, which can be shown with the Taylor expansion.

Appendix A5

Subroutines, which are present in the FAS-GMG algorithm:

SOLVEVARS($\vec{\Phi}^m$, presmooth, m) {
  do l = 1 to L_m
    BC(m)
    LOCALDT(m)
    if(l=1 **and** m>1 **and** presmooth) {
      MASSBALANCE(Δm^m)
      $\overline{\Delta m^m} = \Delta m^{m,*} - \Delta m^m$
    }
    CALCULATECV(m)
    do k = 1 to K
      SOLVEVAR(ϕ_k^m,k,l,m,presmooth)
    enddo
    PC(p^m,l,m)
  enddo
}

RESTRICTVARS($\vec{R}^{m-1} \rightarrow \vec{R}^{m}$, $\vec{\Phi}^{m-1} \rightarrow \vec{\Phi}^{m}$,m) {
  BC(m)
  do k = 1 to K
    $R_k^{m-1} = s_k^{m-1} - F_k^{m-1}(\phi_k^{m-1})$
    RESTRICT1($R_k^{m-1} \rightarrow R_k^m$)
    RESTRICT2($\phi_k^{m-1} \rightarrow \phi_k^m$)
  enddo
  MASSBALANCE(Δm^{m-1})
  if(m=2) {
    RHIE&CHOW(Δm^{m-1})
  }
  RESTRICT1($\Delta m^{m-1} \rightarrow \Delta m^m$)
  RESTRICT2($p^{m-1} \rightarrow p^m$)
}

PROLONGATEVARS($\vec{\Phi}^{m+1} \rightarrow \vec{\Phi}^{m}$,m) {
  BC(m)
  do k = 1 to K
    PRO($\phi_k^{m+1} \rightarrow \phi_k^m$)
  enddo
  PRO($p^{m+1} \rightarrow p^m$)
}

SOLVEVAR(ϕ_k^m,k,l,m,presmooth) {

$R_k^m = s_k^m - F_k^m(\phi_k^m)$

if(l=1 **and** m>1 **and** presmooth) {

$\overline{R_k^m} = R_k^{m,*} - R_k^m$

}

$\hat{R}_k^m = R_k^m + \overline{R_k^m}$

LINEARIZE(F_k^m)

AMG($F_k^{'m} d\phi_k^m = \hat{R}_k^m$)

$\phi_k^{m,l+1} = \phi_k^{m,l} + \alpha_k d\phi_k^m$

}

PC(p^m,l,m) {

MASSBALANCE(Δm^m)

if(m=1) {

RHIE&CHOW(Δm^m)

}

$\hat{\Delta m}^m = \Delta m^m + \overline{\Delta m^m}$

PCECOEFFS(m)

AMG($A_p^m p^{'m} = -\hat{\Delta m}_k^m$)

$u^{m,l+1} = u^{m,*} + \alpha_u u^{'m}$

$v^{m,l+1} = v^{m,*} + \alpha_u v^{'m}$

$p^{m,l+1} = p^{m,l} + \alpha_p p^{'m}$

}

Aalto-ST 24/2011

BUSINESS + ECONOMY

ART + DESIGN + ARCHITECTURE

SCIENCE + TECHNOLOGY

CROSSOVER

DOCTORAL DISSERTATIONS

ISBN 978-952-60-4361-6 (pdf)
ISBN 978-952-60-4360-9
ISSN-L 1799-4896
ISSN 1799-490X (pdf)
ISSN 1799-4896

Aalto University
School of Engineering
Department of Applied Mechanics
www.aalto.fi